AST2600

Integrated Remote Management Processor

A3 Datasheet

ASPEED Technology Inc.

Version 1.2

June 2, 2022

ASPEED Technology Inc. retains the right to make changes to its products or specifications. While the information furnished herein is held to be accurate and reliable, no responsibility will be assumed by ASPEED Technology for its use. Furthermore, the information contained herein does not convey to the purchaser of microelectronic devices any license under the patent right of any manufactures.

ASPEED products are not intended for use in life support products where failure of an ASPEED product could reasonably be expected to result in death or personal injury. Anyone using an ASPEED product in such an application without express written consent of an officer of ASPEED does so at their own risk, and agrees to fully indemnify ASPEED for any damages that may result from such use or sale.

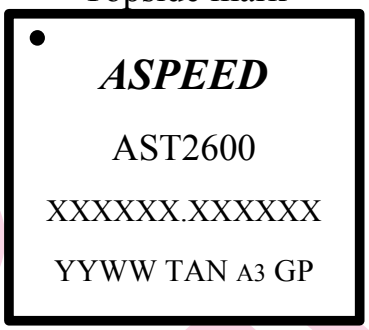
All other trademarks or register trademarks mentioned herein are the property of their respective holders.

Headquarters
4F., No. 1, Sec. 3, Gongdao 5th Rd., East Dist.,
Hsinchu City 30069, Taiwan, R. O. C.
TEL: 886-3-5751185
FAX: 886-3-5751183
http://www.ASPEEDtech.com

Ordering Information

Part number :AST2600A3 - GP
Solder ball type :Lead-free
Substrate type :RoHS Green package
Package size :21mm x 21mm
Ball pitch : 0.8mm

Topside mark



Top Marking Definition:

Line 1: [ASPEED] = Company Logo
Line 2: [ASTxxxx] = Product Code
Line 3: [XXXXXX.XXXXXX] = IC Foundry Lot Number
Line 4: [YYWW] = Date code, [TAN], [xx] = Chip Revision Number, [GP] = RoHS

Revision History

Date	Revision	Description
Jan. 2, 2019	0.1	Initial draft.
May.31, 2019	0.2	Second draft.
Sep.12, 2019	0.3	1. Merge R3VDD and R4VDD into one ”RVDD” power domain. 2. Change PV33D ”Pin K11” and ”Pin K12” to indepent ”PV33D_RGM” domain. 3. Add Electrical Specifications. 4. Fix some typo.
Nov. 21, 2019	0.4	1. Correct WDTRST and RSTIND signal name with low active symbol. 2. Remove BMCINT function on Pin A15. 3. Add new HBLED# function on Pin Y23. 4. First release for AST2600 A1.
Jan. 3, 2020	0.5	1. Add section ”Reset Source Table”. 2. Add descriptions for section ”SRAM Memory Buffer”. 3. Add reset sources for all register descriptions. 4. Fix IO pull ups and driving strangths in pin descrition. 5. Add AST2620 comparison table. 6. Fix some typo.
May.18, 2020	0.6	1. Add and modify SCU0C8, SCU0D8, SCU300[14:11], SCU310[3:0], SCU314[12:6], SCU338, SCU33C, SCU500, SCU510, SCUC20[18:16] detail descriptions. 2. Add OTP Memory descriptions in section 61. 3. Add new MAC58[28] and MAC58[27] register descriptions. 4. Modify I2CD04[19:16] and I2CD04[15:12] register descriptions. 5. Change desciprtions for Uart Debug Interface section 11. 6. Remove interrupt #197 from interrupt source table in section 8. 7. Remove register SWVIC10, SWVIC18[15], SWVIC1C[15], SWVIC20, SWVIC28[15], SWVIC2C[15]. 8. Add new function LPC Mem/FWH to AHB with register HICR6[17]. 9. Extend Mailbox from 16 to 32 registers. 10. Change SPI tCSS and I3C setup time SPEC. 11. Remove description ”Only 2 out of the interfaces can be enabled simultaneously”. 12. OTPSTRAP Description table was deleted because it is same as SCU500, SCU510. 13. Add SCU to Strap Source Mappings to represent the strap sources clearly. 14. Update VR040 Multi-JPEG Data Buffer descriptions. 15. Change feature of Master Serial GPIO to 2 sets 16. Change feature of Slave Serial GPIO to 2 sets 17. Add new section 9 ”Hardware Strap Registers” 18. Add new section 10 ”ARM TrustZone” 19. Add OTP programming temperature SPEC. 20. Add SPI electical SPEC when clock speed higher than 100MHz. 21. Add RSTIND# electrical SPEC on power sequence section. 22. Add ”List of Tables” 23. Update A1 power consumption data on electrical SPEC. 24. Fix page number mismatch in ”Contents” 25. Fix some typo.

to next page

from previous page

Aug.18, 2020	0.7	1. Add watchdog related protection registers for security: SCU830, SCU834, SCU850, SCU854, SCU858, SCU85C, SCU870, SCU874, SCU878, SCU87C. 2. Add seperated UART DMA control registers: UDMA04C, UDMA05C, UDMA06C, UDMA07C, UDMA08C, UDMA09C, UDMA0AC, UDMA0BC, UDMA0CC, UDMA0DC, UDMA0EC, UDM0FC, UDMA10C, UDMA11C, UDMA12C, UDMA13C, UDMA14C, UDMA15C, UDMA16C, UDMA17C, UDMA18C, UDM19C, UDMA1AC, UDMA1BC, UDMA1CC, UDMA1DC, UDMA1EC, UDMA1FC. And increased buffer size for VUART TX/RX DMA. 3. Add 4 independent I2C controllers for security usage (I2CS) 53. And Add new address map table for I2CS. And add interrupt 75-78 for I2CS. 4. Change A-PLL register descriptions: SCU210, SCU214. The frequency calculation formula is changed. 5. Add secure boot registers: SEC50, SEC54, SEC58, SEC5C, SEC60, SEC64, SEC68, SEC6C, SEC80, SEC84, SEC88, SEC90, SEC94, SEC98, SECB0 and SECB4. 6. Add new hardware pin straps: GPIOZ3, GPIOZ4, GPIOZ5, GPIOZ6, GPIOZ7 in 2.3 and update register SCU51C[10:6]. 7. Add core power good and I/O power good detection registers and CHASI# raw status to register CHAI10 8. Add detail I3C Register revision change in Section 54.3 9. Remove LPC Host function. 10. Update ”I/O DC Electrical Specification” for A2 version. 11. Add and modify PTCR0F0[20], ACPIE3E0, ACPIE3, MBXSTS0, MBXBCR, MBXFCR1, MBXFCR2, SMBXFCR1, SMBXFCR2, HICR6, HICR9, HICRA, SV1UART24, SIRQCR0, SIRQCR1, SIRQCR2, SIRQCR3, SNPWADR, BTR0, BTFVSR0, BTFVSR1, PCCR0, PCCR1, PCCR2, PCCR3, PCCR5, SIOR730, SIOR731, SIOR732, SIOR739, iBTCR1, iBTCR3.
Dec.08, 2020	0.8	1. First release for A3. 2. Add Features Comparison for AST2600A0/AST2600A1/AST2600A2&A3. 3. Add Boot Flow and CPU feature. 4. Correct timing control block diagram for MAC1/MAC2/MAC3/MAC4. 5. Add NCSI 50MHz clock output frequency stability from PinF24/PinH24. 6. Add minimum pulse timing requirement for CHASI#. 7. Fix some typo.

to next page

from previous page

May.6, 2021	0.9	1. Add description of behavior of OOB_FREE in ESPI000[4]. 2. Add A2 to A3 change notice. 3. Remove system reset control for AHB bridges, SCU040[1], and change to reserve. 4. Provide OTP strap option, OTPSTRAP[23] and SCU500[24], to enable PCIe root complex reset on SSPRST# pin. 5. Provide programming option, SCU040[18], to reset PCIe root complex when OTPSTRAP[23]=0. (similar to dedicated reset pin). 6. Add revision ID for A3 in SCU004. 7. Change default values of SCU0C8 and SCU0D8. See details in ”A2 to A3 Change Descriptions” item 1. 8. Boot from UART can be UART1 and UART5 automatically. Change descriptions of strap function pin FWSPICK in section ”Hardware Pin/OTP Strap Definition” 2.3. Change descriptions of OTPSTRAP[40] in table ”OTPSTRAP Mappings” 13. See details in ”A2 to A3 Change Descriptions” item 3. 9. Fix typo in table ”OTPSTRAP Mappings” 13. The OTPSTRAP[5] should be MAC 1 RGMII mode. The OTPSTRAP[6] should be MAC 2 RGMII mode. The OTPSTRAP[32] should be MAC 3 RGMII mode. The OTPSTRAP[34] should be MAC 4 RGMII mode. 10. Fix typo of the password in SEC00 11. Change SCUC08, PEHR04, SCUC24[9], HICRB[6], HICRB[29] , PECI00[24], PECI08[30]. 12. Add I3C010[29], I3C010[28], I3C020[29], I3C020[28], I3C030[29], I3C030[28], I3C040[29], I3C040[28], I3C050[29], I3C050[28], I3C060[29], I3C060[28]. 13. Add ADDR_MASK, please refer to I3CD280, I3CD284, I3CD288, I3CD28c, I3CD290, I3CD294, I3CD298, I3CD29c. 14. Modify I3CD01C, IBI Data Threshold Value field, the maximum supported size is 31. 15. Modify I3CD084, memory access from R to RW. 16. Remove typo for WDT20[24:25]. Add new items, WDT20[27] and WDT28[27], for I2CS. 17. Fix typo in Uart DMA, there are 14 sets of DMAs. The base address is Uart DMA instead of Timer. 18. Fix baud rate calculation in UART Controller, UART_DLL/UART_DLH. 19. Fix typo of SDIO0F4[25:21] to Slot 1 Input Clock Phase. 20. Add PHYA00 and PHYB00[11] bit to enable lower transmit FIFO threshold setting. 21. Add PHYA00 and PHYB00[10] bit to improve asynchronous list performance. 22. Add more detail hardware strap description on Hardware Pin Strap Section. 23. Remove power sequence requirement for ESPICLK and ESPIRST#. 24. Update AC electrical SPEC for Strap Input Interface. 25. Update peak temperature of SMT Soldering Reflow Chart. 26. Add I3C Low voltage IO VOL/VOH SPEC. 27. Add RGMII AC timing SPEC with PHY RXCLK delay mode. 28. Remove description of SEC78[3]. It is a typo.

to next page

from previous page

Dec.1, 2021	1.0	1. Fix typo for reset sources of CHAI10[2], CHAI10[10] and CHAI10[18]. 2. Fix UART DMA sources to UART1-4, UART6-13. UART DMA dose not support for UART5. All of the related register descriptions are modified. 3. Fix and reword descriptions in SBSTRAPStraps for Secure Boot Enable. 4. Reword descriptions of SCU51C[1]. 5. Reword descriptions of SCU510[31]. 6. Reword descriptions of OTPCFG0[1]. 7. Reword descriptions of OTPCFG0[6]. 8. Change OTPCFG4[23:16] to reserved. 9. Add four Chapters: XSuperIO Controller, XVirtual UART, XLPC Controller and XMailBox Controller. 10. Add DC/AC electrical SPEC for UART. 11. Add Diagram of Secure Boot enablement control in Section 9.7. 12. Remove sequence requirement for LPCCLK and LPCRST# in Section 3.7. 13. Remove SDR104 description due to Errata item 69. 14. Add UART routing in Section 1.7.3 15. Add Flash Interface Architecture in Section 1.7.4 16. Add I3C I/O Diagram in Section 1.7.5 17. Add Recommended Operating Conditions for I3CVDDL. 18. Add PCIe to LPC section. 19. Add PCIe to MailBox section. 20. Add PCIe to SuperIO section. 21. Add PCIe to UART section.
Dec.24, 2021	1.1	1. Update AST2600 Chip Architecture Block Diagram on Section 1. 2. Fix some typo.
May.31, 2022	1.2	1. Add description of adding delay after deasserting SCU040 or SCU050 2. Correct the electrical SPEC (Vih/Vil/Vhys) for I3C Low-voltage I/O. 3. Fix a typo of SCU300[22:20] from HPLL to EPLL. 4. Add Note of Negotiated Speed in PECI1C. 5. Add Snoop 4-byte mode in HICR6[19]. 6. Add PEHR2C[8] 3.5 dB. 7. Add XHICRB[30]. 8. Modify SCU300[10:8] and SCU308. 9. add VESA in PCIS44[1:0]. 10. Move IV PCIE2LPC/PCIE2UART to III PCIE2LPC/PCIE2VUART. 11. Change P-Bus from PCI Bus Controller to PCI Bus VGA Controller 12. Remove P2A Section. 13. PEHR14 wording revised. 14. Define NCSI Tco electrical AC timing. 15. Correct DAC Driver full swing calculation on SCU0D4[8]. 16. Add additional Note in power up sequence section for AP-Note #17. 17. Add PCIe Architecture Diagram. 18. Fix some typo.

Text color definition

1. Initial draft
2. A1 revision
3. A2/A3 revision

A1 to A2 Change Highlight:

1. Reference to revision history for V0.7.

2. In A2 revision, more hardware pin straps are supported on pins GPIOZ3, GPIOZ4, GPIOZ5, GPIOZ6 and GPIOZ7. The external circuit must keep correct level of those pins during strap period. Please reference to AC timing specification of Strap Input Interface. In AST2600/AST2620 A1 platform.S, LPC/eSPI selection is set by checking GPIOZ4. It must be removed after A1.
3. There are some modifications for UART DMA control registers. The driver must change for them.
4. There are some modifications for Secure boot. Please reference Secure Boot Specification V5 for details.
5. The PLLAHVDD (PinH14) voltage requirement is changed from 3.3V to 1.2V.
6. For GPIOZ4 strap pin on A2/A3 silicon version, this pin include internal pull-down and it is actual hardware strap pin. You can stuff external Pull-up 4.7K ohm resistor to PV33D if host is LPC mode. For A2/A3 version, you do NOT need to enable path code on the SDK firmware for this pin as A0/A1 version.
7. For GPIOZ6 (PinAD11), it is used as GPIO Pass Through selection:
   0b: Disable GPIO Pass Through function on GPIOP[5:0]
   1b: Enable GPIO Pass Through function on GPIOP[5:0]
   Please note GPIOP[5:0] pins are defined as Pass Through application after SRST# de-asserted when your system includes a pull-up on GPIOZ6 pins. The GPIOZ6 strap setting will affect several pins functionality in your system, please check your schematic and ensure it is meeting your usage case.
8. For GPIOZ7 (PinAF10), it is used as ACPI function selection:
   0b: Disable ACPI function
   1b: Enable ACPI function
   Please note some pin are defined as ACPI functional pin such as (SIOS3#, SIOS5#, SIOPWREQ#, SIOONCTRL#, SIOPBI#, SIOPBO#, SIOPWRGD, SIOSCI#) after SRST# de-asserted when your system includes a pull-up on GPIOZ7 pins. The GPIOZ7 strap setting will affect several pin functionality in your system, please check your schematic and ensure it is meeting your usage case.
9. Add eMMC reset source from GPIOY3GPIO18A2GPIO18B6, selected by OTPSTRAP[28:27], if eMMC boot is needed.
10. For MAC3/MAC4 in AST2600A3, the RGMII delay time in each step is NOT the same as AST2600 A0/A1. So the clock delay time setting on SCU350 will be changed in your firmware setting after you find the optimum window in mactest result.

A2 to A3 Change Highlight:

1. Reference to revision history for V0.8 and V0.9.
2. All of the debug interfaces controlled by SCU0C8 and SCU0D8 are disabled in default. Set OTPCFG7 registers can change their default values. The hardware strap pin, FWSPIMISO, can enable Low Secure Boot function. It also can change the default values of SCU0C8 and SCU0D8 and keep UART1/UART5 debug port enabled.
3. Add big-endian key mode. The RSA keys and signature in secure standards are typically in big-endian format. Only little-endian key modes are supported in A0, A1 and A2 revision. That means those data need be transferred to little-endian format before programming into OTP memory. In A3 revision, big-endian key types are supported.
4. Add UART1/UART5 for Auto Boot from UART/VUART . In A0 to A2 revision, the auto boot can only boot from either UART1 or from UART5. It is selected by OTPCFG3[16]. In A3 revision, secure boot engine will auto detect UART1 and UART5 both. When OTPCFG3[16]=0, the auto boot can boot from UART1, UART5 or VUART by automatically detection. In this condition, GPIOM7 must be keep high during secure boot detection. Otherwise, if the secure boot engine detected GPIOM7=0, it will skip boot from UART1. When OTPCFG3[16]=1, the auto boot can boot from UART1, or VUART by auto detection.

5. Boot from UART can detect UART1 and UART5 automatically. In A0 to A2 revision, the boot from UART can only boot from either UART1 or from UART5. It is selected by OTPCFG3[16]. In A3 revision, secure boot engine will auto detect UART1 and UART5 both. When OTPCFG3[16]=0, the auto boot can boot from UART1 or UART5 by automatically detection. In this condition, GPIOM7 must be keep high during secure boot detection. Otherwise, if the secure boot engine detected GPIOM7=0, it will skip boot from UART1. When OTPCFG3[16]=1, the secure boot can boot from UART1 only.
6. Changing default value of GPIO2D0 from 32'h10CCC1 to 32'h109A61.
7. OOB_FREE in eSPI status is set automatically without BMC FW involvement. However BMC FW still needs to program ESPI000[4] to inform eSPI controller that FW is ready to process OOB packets.
8. Changing bit 31 of ESPI008 from de-assertion to both de-assertion and assertion. Also adding bit 15 of ESPI004 to indicate current ESPI_RSTN pin state.
9. Fix Errata 51: 1-byte dummy occurs before IPMI BT/iBT response message Length byte in eSPI mode
10. Fix Errata 52: I3C IBI read status register 1-bit shift
11. Fix Errata 53: FSI IO signals are inverse and not functional
12. Fix Errata 55: Vulnerability concern for default debug ports enable control
13. Fix Errata 59: New I3C active Open Drain class pull-up option, 750 ohm.
14. Fix Errata 64: MCTP00 bit 15 is changing to enable patch.
15. There are some modifications for Secure boot. Please reference Secure Boot Specification V5 for details.
16. BMC FW has to set SCU040[18] high to reset internal PCI Express Root complex controller as AC power-on. For initial reference fw, please refer u-boot platform.S or add following code in platform.S.
    The SCU040[18] setting is required to for the A3 silicon to boot.
    ldr r0, =0x1e6e2040
    movw r1, #0x0000
    movt r1, #0x0004
    str r1, [r0]
17. Add PHYA00 and PHYB00[11:10] bit to improve USB2.0 Host Controller performance.
18. The register HICR9[15:12] is set to 0x1. The register HICRA[2:0] is set to 0x4. They are set by secure boot engine after ARM reset or power on reset. If IO1 of UART1 or UART10 will be used, software must set these registers with correct settings.

Contents

List of Figures

List of Tables

Part I
Functional Specification

1 General Information

1.1 Introduction

This manual provides the related technical information about the newly developed Integrated Remote Management Processor (IRMP) – AST2600.

This document is intended for product planners, system designers, and software developers who are going to adopt or have adopted this device to support graphics acceleration & display, baseboard management controller (BMC), KVM-over-IP, and virtual storage functions for highly manageable server platforms or iKVM switches.

1.2 Chip Architecture

AST2600 is the 7th generation of Integrated Remote Management Processor introduced by ASPEED Technology Inc. Its a vastly integrated SOC device playing as a service processor to support various functions required for highly manageable server platforms. In this generation, the CPU performance is improved significantly by integrating 1.2GHz dual-core ARM Cortex A7 (r0p5) 32-bit CPU with FPU. Debug access is through ARM CoreSight SOC-400 into CPU. Additionally, most of the controllers are improved with more features or performance. AST2600 also supports more interfaces including PCIe Gen2 1x bus interface and root complex which can make BMC to have expended control capacity. New adopted DisplayPort 1.1a also fits next generation display interface. Finally real secure boot function with secure OTP memory can improve the BMC security. Figure-1 clearly illustrates the chip architecture of the BMC. The detailed features of the individual internal blocks will be descried in the following chapters.

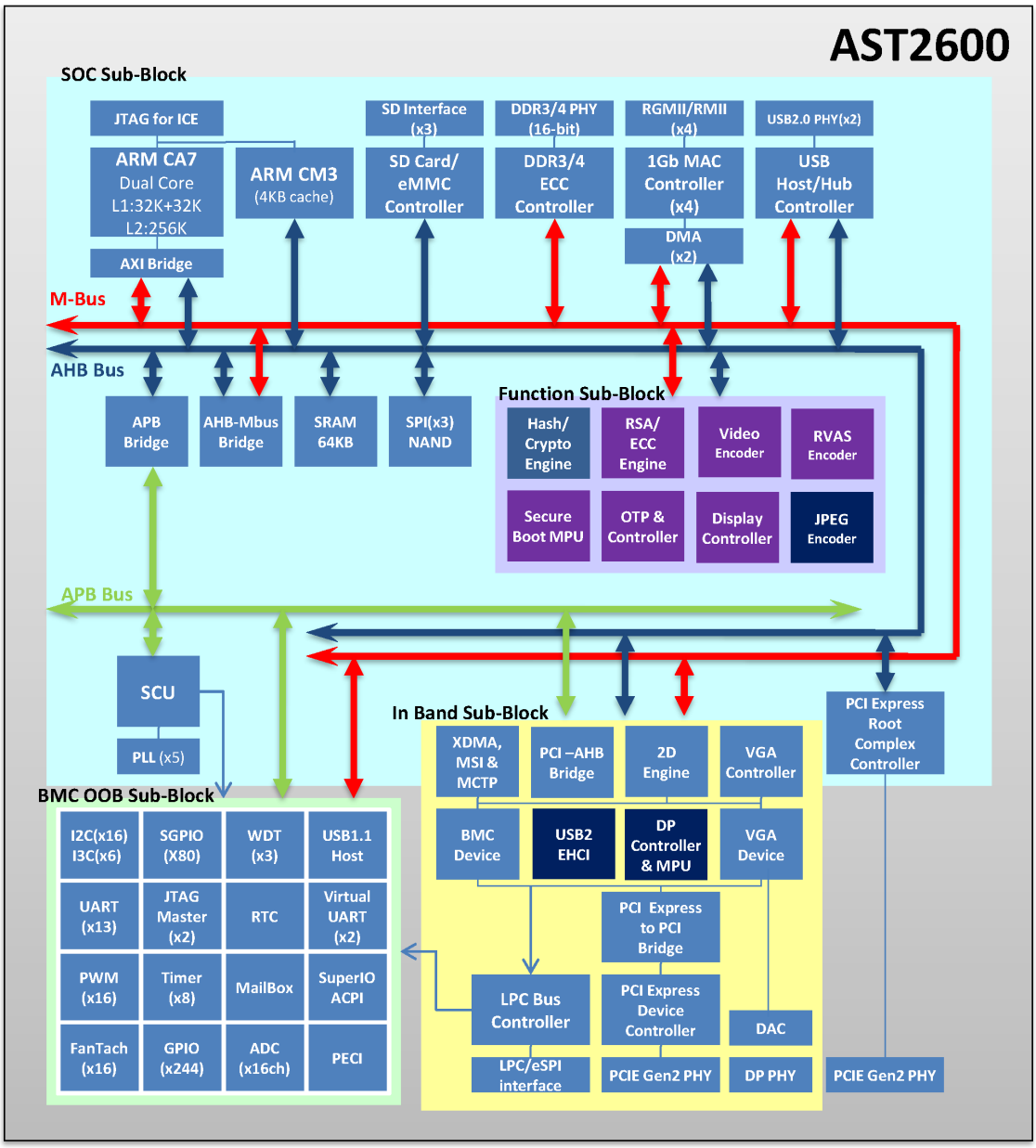


Figure 1: AST2600 Chip Architecture

1.3 Summary of Feature Set

1.3.1 Package

- 21mmx21mm 624-pin TFBGA package
- 0.8 mm ball pitch
- RoHS compliant 4-layer substrate design
- Alpha Particle: ULA (ultra low alpha) $<=$ 0.002 cph/cm2
- Typical condition chip power consumption (including DRAM power)

  Peak power consumption : $<$ 1.84 W
  Average power consumption : $<$ 1.74 W

1.3.2 Design for Testability

- Adopt full-scan-chain design methodology for testing internal logic by Automatic Test Pattern Generation (ATPG)
- Support Built-In-Self-Test (BIST) for testing internal SRAM macros and PCIe PHY
- Support JTAG-compliant boundary scan (BSDL) for board manufacturing tests

1.3.3 PCI Express 2.0 Bus Device Controller

- Built-in PCI Express 2.0 Bridge Controller & PCI Express Gen 2 PHY
- Support 1-lane 5-Gbps PCI Express Bus
- Compliant with PCI Express Base Spec. Revision 2.0
- Compliant with PCI Bus Power Management Interface Spec. Revision 1.1
- Integrate VGA device
- Integrate BMC device
- Support optional BMC KCS device
- Support all memory, I/O, configuration, and message transactions with decoding windows implementation
- Support 32-bit memory space and 16-bit I/O space
- Support standard VGA memory and I/O address decoding
- Support PCIe-to-LPC Bridge if LPC interface disabled
- Support a memory region accessible for Host CPU, ARM CPU, and the coprocessor
- Support INTX, power management, error signaling
- Support native active state power management L0s and L1 states
- Support beacon (wake-up) function
- Support OBFF Msg decoding and interrupt BMC
- VGA device frame buffer memory allocation can be determined by external strapping resistors (16/64MB)
- VGA device supports one way to access all BMC internal IPs with write protection control
- BMC device supports one mapping window to access all BMC internal IPs and memory with write protection control

- BMC device supports leagcy I/O access all LPC registers with write protection control
- BMC device supports one mapping window to access all LPC registers with write protection control
- BMC device supports MCTP function
- BMC device supports X-DMA function

1.3.4 PCI Express 2.0 Root Complex Controller

- Built-in PCI Express 2.0 Root Complex Controller & PCI Express Gen 2 PHY
- Support 1x (1-lane) 5-Gbps PCI Express Bus
- Compliant with PCI Express Base Spec. Revision 2.0
- Compliant with PCI Bus Power Management Interface Spec. Revision 1.1
- Support all memory, I/O, configuration, and message transactions with decoding windows implementation
- Support 32-bit memory space and 16-bit I/O space
- Support INTX, power management, error signaling
- Support native active state power management L0s and L1 states

1.3.5 VGA Display Controller

- Fully IBM VGA compliant
- Maximum Display resolution: 1920x1200 32bpp@60Hz
- Support widescreen resolutions:
  - WXGA : 1280x800 32/16bpp @60Hz
  - WXGA+ : 1440x900 32/16bpp @60Hz
  - WSXGA+ : 1680x1050 32/16bpp @60Hz
  - FullHD : 1920x1080p 32/16bpp @60Hz
- Integrate one dedicated PLL for video clock generation, which can be directly turned off by ARM CPU for power saving
- Support VESA-compliant DDC interfaces for the display monitor
- Support one dedicated SPI master interface for video option ROM
- Support 64x64 hardware overlay cursor with mono and color formats
- RGB analog output
  - Integrate 200MHz triple DACs compliant with VESA monitor timing specification
  - Integrate 1.2V reference voltage generator
  - Support DAC power down function, directly controlled by ARM CPU or Host CPU
- DisplayPort 1.1a output options
  - Includes 2-lane main link with data rate of 2.7 or 1.62 Gbps per lane.
  - Includes AUX Channel transceiver.
  - Supports post-cursor 1 FFE for Main Link transmitter.
  - Supports 0.8 V TX output swing for Main Link transmitter.
  - Supports 0/3.5/6.0 dB Pre-emphasis for Main Link transmitter.
  - Supports built-in internal loopback mode for testing.
  - NOT support DP++ application.
- Triple DAC display output supports dynamic switching between VGA and Graphics Controller

1.3.6 64-bit 2D Graphics Accelerator

- Directly access data through M-Bus
- High performance pipelined one-cycle 64-bit 2D engine
- Optimized for RGB565 and XRGB8888 pixel formats
- 2D engine commands
  - BitBlt Rectangle Fill
  - BitBlt Pattern Fill
  - BitBlt Rectangle Copy from Source to Destination
  - Support 256 Raster Operations
  - Integrate 8x8 Pattern Registers
  - Integrate 8x8 Mask Registers
  - Support Rectangle Clip
  - Support Color Expansion
  - Support Enhanced Color Expansion
  - Support Line Drawing with Style Pattern
  - Support Color Blending
- Integrate 32 stages of hardware command queue for 2D command pre-fetch
- Integrate 64x32 source buffer and 64x32 destination buffer to improve 2D engine performance

1.3.7 DDR4 SDRAM Controller

- Support external 16-bit DDR4 SDRAM data bus width
- Maximum memory clock frequency
  - DDR4 ：800MHz (DDR4-1600)
- Integrate DDR IO PHY with automatic timing and driving calibration capability
- Support Internal 128-bit DRAM data bus width
- Only support the DRAM size that has Column address (CA) = 10 bits (A0∼A9).
- Supported DDR4 DRAM Types: 128Mb x 16, 256Mb x 16, 512Mb x 16, 1Gb x 16, 1Gb x 16 (TwinDie 1 rank)
- Support Error-Correction Check (ECC) option
  - Unified code/data memory, graphics memory, video memory and ECC memory in one SDRAM device
  - No extra external memory cost when enabling ECC function
  - No ECC cycle for graphics and video memory access by automatic memory cycle detection logic
  - Support SECDED (Single bit Error Correction, Double bit Error Detect) ECC algorithm
  - Allocate 8 bits of ECC memory for each 64 bits of code/data memory
  - Support ECC auto scrub function by hardware (for single-bit error)
  - Support interrupt option whenever a recoverable or unrecoverable ECC error occurs
  - Support reset option whenever a unrecoverable ECC error occurs
  - 8-bit ECC error counter with reset control, overflow protection and interrupt option

1.3.8 Dual-core ARM Cortex A7 CPU

- Embedded dual-core ARM Cortex A7 32-bit RISC CPU (r0p5)
- Maximum running frequency: 1.2GHz
- L1 Instruction cache: 32KB
- L1 Data cache: 32KB
- L2 cache: 256KB
- Support Memory Management Unit(MMU)
- Support floating processing unit (FPU)
- Support trust-zone
- Support LPAE
- Support integrated GIC controller
- Support integrated general timer
- Interface: Multiple AXI buses for instruction, data and peripheral access
- Integrated PLL for CPU clock generation
- AXI bus clock speed divider options: 1/2(default), 1/1 to CPU clock
- AHB bus clock speed divider options: 1/2(default), 1/3, 1/4, 1/5, 1/6, 1/7, 1/8
- Integrated AMBA APB bus with embedded AHB to APB bridge
- JTAG interface with CPU reset control for code debugging
- Support L1/L2 cache parity error check

1.3.9 Embedded ARM Cortex M3 CPU

- Embedded one more 32-bit ARM Cortex M3 CPU (r2p1)
- Maximum running frequency: 200MHz (AHB Bus clock)
- Instruction cache size: 4KB
- Data cache size: 4KB
- Interface: AHB buses for both instruction and data access
- The coprocessor can be used for handling some low speed tasks.
- ARM and coprocessor can communicate each other through a shared memory (SRAM or DRAM), or interrupt.

1.3.10 Video Compression Engine

- Fully hardware video compression engine that can maximumly reduce CPU loading.
- Directly connected to AHB bus interface for register programming
- Directly access video data through M-Bus
- Video source can be from internal VGA output or from external DVO input
- Internal Video source capture mode:
  - Video capture mode: capture internal VGA RGB video stream (for legacy VGA display modes, like text modes or 16/256 color graphics modes)
  - Quick fetch mode: directly fetch RGB video data from graphics frame buffer (for 16bpp and 32bpp graphics modes)
- Support three quality levels of video compression scheme:
  - Lossy video compression scheme ($<$40dB)
  - Visually lossless compression scheme (40 $\sim$ 50dB)
  - Lossless compression scheme
- Target average compression ratio: $>$10 (visually lossless compression scheme)
- Engine clock can be turned off when engine is idle, to save power
- Support two video compression modes
  - YUV420: for lower video quality but higher frame rate
  - YUV444: for higher video quality but lower frame rate
- Support two video compression format
  - ASPEED proprietary compression mode: for multi-frame and differentical compression
  - JPEG standard mode: for single frame and management compression
- Support video compression for high graphics display resolution modes up to 1920x1200x32bpp
- Target frame rate: 30 frame/sec for 1280x1024 32bpp@60Hz
- Support independent management view capturing for special purpose like last frame recording.
- Support Quick Cursor (**ASPEED Patent**)
  - Achieve 60 frames/sec cursor refresh rate
  - Directly transmit cursor patterns and X/Y locations to remote site
  - Directly overlay cursor pattern at remote site
- Support Quick Fetch (**Patent pending by ASPEED**)
  - Significantly reduce memory bandwidth requirements for video compression
  - 16 bits of DRAM bus width is enough for 1600x1200x32bpp video compression
  - Only enabled for high resolution modes (high color and true color modes)
  - Quick Cursor must be enabled (cursor overlay will be done in client site)
  - Regular VGA display refresh can be turned off to save power and reduce DRAM utilization rate
- Support arbitrary video down scaling with horizontal & vertical video filtering options
- Integrate RC4, AES128 encryption engine for video stream encryption
  - 1 set of loadable 256x8 SRAM for expanded key buffers

  - Key expansion is done by firmware
  - Provide enable/disable option
- Support two-pass video compression scheme (**ASPEED Patent**)
  - Applied to YUV444 video format only
  - Provide visually lossless video compression quality for reducing network average loading under intranet KVM applications
- Provide industry-leading video compression performance and quality

1.3.11 Internal SRAM

- Support 3 portion of internal SRAM buffer
  1. 64KB.
     - Available at boot.
     - With Parity Check. Parity Check only support Double-Word access.
  2. 24KB.
     - Available at boot.
     - Without Parity Check.
  3. 1KB.
     - Available at boot.
     - Without Parity Check.
     - Each byte is write once. Unlock until primary CPU reset.
     - One global control bit to lock all bytes at once. Unlock until chip reset.
- Directly accessed by CPU through AHB

1.3.12 Battery Backed SRAM

- Support 64 bytes internal SRAM buffer
- Directly accessed by CPU through APB
- SRAM content is retained even after power-off or hardware reset conditions when powered by an external battery

1.3.13 System Control Unit (SCU)

- Directly connected to internal APB bus
- Centralize clock and reset control registers of all modules
- Support programmable CPU clock divider (1/1 ∼ 1/16) for power saving
- Support programmable auto-CPU clock slow down function to reduce power when system is idle, which is controlled by hardware automatically, no firmware effort.
- Integrate all PLL control registers
- Integrate all power saving control registers
- Integrate all multi-function pins selection registers
- Integrate 2 sets of ring oscillators for process window monitoring
- Integrate hardware strap registers

- Integrate PCI ID setting registers
- Integrate 32 bytes of scratch registers for Host CPU to BMC CPU message passing
- Integrate 8 bytes of scratch registers for BMC CPU to Host CPU message passing
- Integrate a real random number generator
- Supports 64 bits of chip unique ID

1.3.14 AHB Controller (AHBC)

- AHB master arbitration
- AHB memory address remapping control with register-write protection
- AHB bus read/write command logging function. This is a new function that can record a specific address of AHB bus read/write operation to SRAM buffer or DRAM. Such as the firmware console output message, which can be recorded into the DRAM as a ring buffer. The logging function is very helpful for debug purpose that can record all bus accessing operation on this address without any loss.
- Support bus lock prevention watchdog capability.

1.3.15 Firmware SPI Memory Controller (FMC)

- Directly connected to internal AHB bus interface
- Support maximum 256 MBytes of direct SPI addressing, and unlimited size of indirect SPI addressing.
- Support a flexible command mode for any types and any size of flash access
- Support 1, 2, and 4 bits input/output command modes
- Support booting from 24 or 32 bits address mode, switch automatically.
- Support 3 chip select pins
- The first chip select FWSPICS0# is the BMC default code boot source
- Integrate DMA controller for large amount of data copying between flash and DRAM, it is very helpful for code shadowing operation.
- Support dual boot function for firmware fail-over recovery. System will auto reboot from the second flash (FWSPICS1#) if the main flash (FWSPICS0#) does not boot up successfully within 22 seconds.
- Support write command and address filter function
- Support register lock until reset function
- For more detailed features, refer section 14

1.3.16 SPI Master Controller (SPI)

- Support more 2 sets of SPI master interfaces
- All registers definition are the same as the firmware SPI memory interface.
- Support a flexible command mode for any types and any size of flash access
- Support 1, 2, and 4 bits input/output command modes
- Support 24 or 32 bits address mode.
- Support 2 chip selects for SPI1, and 3 chip selects for SPI2
- Support write command and address filter function
- Support register lock until reset function
- For more detailed features, refer section 15

1.3.17 SD/SDIO/eMMC Host Controller

- Directly connected to AHB bus
- Support SD Memory Card v2.00/v3.00 (include SDHC)
- SDIO Host Specification v2.00 Compliant
- eMMC Specification v5.1 Compliant
- Support maximum 32GB flash card size
- Support 2 slots of 4-bits SD/SDIO/eMMC and 1 slot of 8-bits eMMC interfaces
- Integrate DMA controller
- Independent interrupt
- Power and clock of each slot can be controlled independently
- Support boot from eMMC
- Not support eMMC HS-400 mode.

1.3.18 Hardware Secure Boot

- Support hardware boot image measurement and decryption
- Programmable option to select Secure Boot image in SPI or eMMC. The boot image size of eMMC is limited to 64KB
- Support second boot image measurement
- Support up to 8 Secure keys with retirement and auto switching functions
- Programmable option to select SHA224, SHA256, SHA384 or SHA512 for Secure Boot image measurement
- Programmable option to select RSA1024, RSA2048, RSA3072 or RSA4096 for Secure Boot image digest encryption
- Supports AES 256 for Secure Boot firmware image encryption
- Built-in 64Kbit One Time Programmable (OTP) memory for configuration, Key Storage and store user programmable data. The OTP memory has the following features:
  - Support ECC for OTP data access
  - Built-in redundancy rows
  - Built-in charge pump
  - Secure key area can be accessed only by Secure Boot engine connected in hardware. There is no other way to read the stored keys either by firmware, scan or JTAG
  - Write protection bits for multiple regions of OTP memory
  - 64-bit OTP memory reserved for programming user defined random/Unique number/Key
- Built-in 64-bit unique chip ID programmed at the time of factory testing
- Built-in true random number generator
- Built-in AES256 secret vault key encryption and decryption
- For detail hardware architecture and software implementation guide, please refer to ”AST2600 Secure Boot User Guide” document.

1.3.19 USB2.0 Virtual Hub Controller

- Compliant with USB Specification Revision 2.0
- Integrate 1 set of USB2.0 Virtual Hub Controller
- Integrate 1 set of USB2.0 PHY, shared with USB2.0 Host function
- Directly connected to AHB bus interface for register programming
- Directly accessible DMA data through M-Bus
- Support USB2.0 standard and backward compatible with USB1.1 standard
- Support one hub port and 7 downstream ports with configurable endpoint type
- Support total 21 configurable endpoints
- Support each downstream port with:
  - Control endpoint : 1 set
  - Interrupt/Bulk/Isochronous endpoint : 1-15 sets (total 15 sets in endpoint pool)
- Integrated DMA engine for direct memory bus accesses (bypass AHB bus)
- Support automatic retry of failure packets and PING flow control
- Support USB remote wake-up (Suspend/Resume)
- Flexible architecture that can emulate at most 7 of any types of USB devices concurrently
- Support USB over LAN architecture

1.3.20 USB2.0 Device Controller

- Compliant with USB Specification Revision 2.0
- Integrate 1 set of USB2.0 PHY, shared with USB2.0 Host function
- Directly connected to AHB bus interface for register programming
- Directly accessible DMA data through M-Bus
- Support USB2.0 standard and backward compatible with USB1.1 standard
- Support endpoints:
  - Control endpoint : 1 set
  - Interrupt/Bulk/Isochronous endpoint : 4 sets
- Integrated DMA engine for direct memory bus accesses (bypass AHB bus)
- Support automatic retry of failure packets and PING flow control
- Support USB remote wake-up (Suspend/Resume)

1.3.21 USB1.1 HID Device Controller

- Compliant with USB Specification Revision 1.1/2.0
- Integrate 1 set of USB2.0 PHY, shared with USB2.0 Host/Device function
- Support Low speed mode transfers
- Support suspend, wake-up resume and remote wake-up resume
- Support 1 control and 2 interrupt endpoints
- Support 8 bytes of buffer for each endpoints
- Target to emulate keyboard and mouse devices over LAN

1.3.22 USB2.0 Host Controller

- Compliant with USB Specification Revision 2.0
- Compliant with EHCI Specification Revision 1.0
- Directly connected to AHB bus
- Support 2 EHCI controllers and 2 Host ports
- Integrate 2 set of USB2.0 PHY, shared with USB2.0 device
- Support Low/Full/High speed mode transfers
- Compliant to EHCI Host controller driver

1.3.23 USB2.0 Host Controller on PCIe bus

- Compliant with USB Specification Revision 2.0
- Compliant with EHCI Specification Revision 1.0
- Directly connected to PCIe bus
- Support internal direct head to head connection for EHCI to Virtual Hub controller, So it can eliminate the external USB connection for BMC remote USB redirection function.
- Compliant to EHCI Host controller driver

1.3.24 USB1.1 Host Controller

- Compliant with USB Specification Revision 1.1/2.0
- Compliant with UHCI Specification Revision 1.1
- Directly connected to AHB bus
- Support Low/Full speed mode transfers
- Support 1 UHCI controller and 2 Host ports, shared with the 2 USB2.0 Host ports
- Compliant to UHCI Host controller driver

1.3.25 10/100/1000 Mbps Fast Ethernet MAC

- Integrate 4 MACs compliant with IEEE802.3 and IEEE802.3z specification
- Support 10/100/1000M bps transfer rate
- Support Reduced Media Independent Interface (RMII/NCSI x2) or Reduced Gigabit Media Independent Interface (RGMII x4)
- Support 2 NCSI interfaces up to 4 loads on 50 ohm impedance traces up to 20 inches in length
- Support total 4 interfaces: 2 RMII/RGMII interfaces and 2 RGMII/RGMII/NCSI interfaces
  - The MAC#1 and MAC#2 RMII/RGMII interfaces can support 1.8V
  - The MAC#3 and MAC#4 RMII/RGMII/NCSI interfaces can support 1.8V/3.3V and can't have separate voltage for both interfaces
- DMA engine for transmitting and receiving packets
- Integrated link list DMA engine for M-Bus access
- Support IEEE 802.1Q VLAN tag insertion and removal

- Support High Priority Transmit Queue
- Independent TX/RX FIFO
- Support half and full duplex (1000 Mbps mode only supports full duplex)
- Support flow control for full duplex and backpressure for half duplex
- Support zero-copy data transfer off-load engine
- Support IP, TCP, UDP TX/RX checksum offloads
- Support Jumbo packets (9.6K bytes)

1.3.26 I2C/SMBus Serial Interface Controller

- Directly connected to APB bus
- Integrate 16 sets of multi-function I2C/SMBus bus controllers
  - Each controller can be programmed as a master or slave controller
  - Support DMA mode for transfer data to/from DRAM
  - Programmable Size ($<=$ 32 Byte) of Pool Buffer Mode for improving performance
  - Programmable Size ($<=$ 4096 Byte) of DMA buffer for large amount of data transfer
  - Support 1 dedicated hardware alert pin for each set of SMBus controller
  - Support 3 slave address when working at slave mode
- Extra 4 Secure I2C controllers (I2CS) are built in A2 revision. They can be assigned to TrustZone or SSP for security usage.
- Support High-Speed mode (3.4M bps) capability
- Schmitt type of input data buffer and input clock buffer to add noise immunity
- Embedded anti-glitch input data filter
- Bus lock condition detection:
  - Programmable interrupt option for SDA data line lock with programmable SDA-low time out period
  - Programmable interrupt option for SCL data line lock with programmable SCL-low time out period
- Support auto or manual recovery capability for SDA data line locked case
- Need external pull-up resistors

1.3.27 I3C Serial Interface Controller

- Integrate 6 sets of MIPI I3C bus controllers. The MIPI I3C controller supports the following general features:
  - Two Wire I3C serial interface - consists of a serial data line (SDA) and a serial clock (SCL)
  - Configurable Device Roles V I3C Main Master and I3C APB Slave
  - Separate Command and Data buffers for ease of transfers
  - Configurable Buffer Depths
  - Single port RAM support for Command and Data Buffers
  - Dedicated buffer for capturing device characteristic information when configured as a Master
  - Supports up to 128 Write or Read bytes with single command
  - Hardware assisted Dynamic Address Assignment (DAA) support

  - Hot-Join support with user controllable filter
  - In-Band Interrupt support with user controllable filter
  - Supports Data transfer to Legacy I2C Slaves
  - Supports various Data rates (FM/FM+/SDR/HDR-DDR,TSP/TSL)
  - CRC/Parity Generation and Validation
  - Support for broadcast and directed CCC transfers
  - Use of Duty Cycle to achieve Lower Effective Speed in a Mixed Fast Bus when configured as a Master
  - Debug Interface
- MIPI I3C Master Features
  - Clock Stalling support in Master Mode
  - Supports Device Address Table for addressing Multiple Slaves
  - Detects arbitration loss due to incoming IBI and subsequently re-transmits the command
  - Programmable Retry Count for transfers that are NACK'ed (Address) by Slaves
- MIPI I3C APB Slave Features
  - Static or Dynamic Slave Device Support.
  - Built-in Hardware Dynamic Address Allocation support (ENTDAA/SETDASA).
  - Built-in CCC transfer handler (does not involve the application)
  - APB3 Slave Interface for register access
  - External DMA support through hardware handshake interface
  - Configurable Queue Depths (up to 16 locations)
  - Configurable Data Buffer Depths (up to 4 KB)
  - Auto Hot-Join request generation support
  - Slave Interrupt Request generation support
  - I2C mode support
  - Adaptive mode of operation between I2C and I3C depending on I3C bus traffic
- MIPI I3C controller and port combination
  - MIPI I3C controller 1 and 2 are using dedicated ports I3C1 and I3C2, which support 1.0V/1.2V voltage.
  - MIPI I3C controller 3 and 4 have 2 configurations
    - Dedicated ports I3C3 and I3C4 which support 1.0V/1.2V voltage. Or
    - Sharing with IO ports of I2C which support 1.8V/3.3V voltage.
  - MIPI I3C controller 5 and 6 are sharing with IO ports of I2C which support 1.8V/3.3V voltage.

1.3.28 GPIO Controller

- Directly connected to APB bus
- Support up to 244 GPIO pins, which are 31 sets
- Each GPIO sets can be programmed to accept command from ARM, LPC(SIO), or Coprocessor.
  - Total 228 GPIO pins are full featured and can be programmed to support below capabilities:
    - Input or output option (input mode or output mode)
    - Interrupt generation option (enabled or disabled interrupt generation)

    - Interrupt sensitivity option (level-high, level-low, rising-edge, falling-edge or both-edge trigger mode)
    - Interrupt direction option (BMC or LPC), by group assignment
    - WDT reset tolerance (for non-interrupted related registers only)
    - Three de-bouncing timer options (the time range is from microsecond to 100 millisecond)
    - Input signal masking option (the registers will maintain the signal state that existed prior to the masking taking effect)
  - Total 16 GPIO pins are input mode only and can be programmed to support the following capabilities:
    - Interrupt generation option (enabled or disabled interrupt generation)
    - Interrupt sensitivity option (level-high, level-low, rising-edge, falling-edge or both-edge trigger mode)
    - Interrupt direction option (BMC or LPC), by group assignment
    - WDT reset tolerance (for non-interrupted related registers only)
    - Three de-bouncing timer options (the time range is from microsecond to 100 millisecond)
    - Input signal masking option (the registers will maintain the signal state that existed prior to the masking taking effect)
- Programmable output driving mode: Push-Pull or Open-Drain
- Some GPIOs support Schmitt type input buffer for noise immunity
- 4 out of the 244 GPIO pins have 16mA driving strength, others have 8mA driving strength
- 36 out of the 244 GPIO pins that support 1.8V mode.
- 40 out of the 244 GPIO pins that support 1.8V/3.3V modes.
- Support 3 pairs of GPIO pass-through (1 GPIO IN $\rightarrow$ 1 GPIO OUT) pin with internal switch control, which is useful for some button function control

1.3.29 Master Serial GPIO Controller

- Directly connected to APB bus
- Co-work with external serial-chained TTL components (74LV165/74LV595)
- Suuport 2 masters. First one is up to 128 input and output. The other is up to 80 input and output. Each master only needs 4 control pins.
- Shift clock is from APB bus clock divided by a programmable value.
- All of the 4 control pins (In/Out/Clock/Load) are multiplexed with GPIO pins
- Programmable shift-load clock length (8/16/24/32/40/48/56/64/72/80/88/96/104/112/120/128 clocks)
- Support interrupt option for each input port
- Support interrupt sensitivity option: Level-High, Level-Low, Edge-High, Edge-Low
- Support reset tolerance option for each output port
- Support input signal masking option (the registers will maintain the signal state that existed prior to the masking taking effect)

1.3.30 Slave Serial GPIO Monitor

- Slave Serial GPIO monitors SGPIO bus between Initiator and Target that follows SFF-8485/8489.
- Support 2 sets, and 2 channels of monitor input on each set
- Support maximum 32 drives recording capability for each channel

1.3.31 UART (16550)

- Directly connected to APB bus
- Support 13 UART controllers and 13 UART IO interfaces, UART controller 1-4 can be directed to UART IO 1-4 interface and UART5 with Tx/Rx only for firmware console
- UART IO 5 and UART IO 7-13 are one to one binding to internal controller
- Support 4 sets UART IO interface with full flow control pins
- Support 4 UART controllers that can be redirected to be controlled by LPC bus as physical UARTs of host system
- Support baud-rate change detection for UART1 ~ UART4
- Separate transmit & receive FIFO buffer (16x8) to reduce CPU interrupts
- Support up to 3686.4K baud-rate except UART5 921.6K baud-rate
- Programmable baud rate generator
- Standard asynchronous communication bits – Start/Stop/Parity
- Independent masking of transmit FIFO, receive FIFO, receiving timeout and error condition Interrupts
- False start-bit detection
- Line break generation and detection
- Fully programmable serial interface characteristics
- 5/6/7/8 data length
- Even, odd, stick and none parity generation and detection
- 1/2 stop-bit generation
- Support DMA mode for each UART
- Support hardware UART debug command interface, which can be used for debugging and updating firmware. Without firmware supporting required. The debug interface can be chosen by hardware strap to work at UART1 or UART5 port. And it requires an authentication to enable the debug mode.

1.3.32 Timer

- Directly connected to APB Bus
- Built-in 8 sets of 32-bit timer modules
- Free-running or periodic mode
- Maskable interrupts

1.3.33 Watchdog Timer (WDT)

- Directly connected to APB bus
- Watchdog function
- Built-in 8 sets of 32-bit WDT modules
- Generate either interrupt or reset after counting down to zero (programmable)
  - **System reset signal**: to reset BMC or full chip
  - **Interrupt signal**: to interrupt CPU

  - **External signal**: to externally reset controller, maximum pulse width 1.048 second, programmable polarity

- Generate 2 types of reset pulse (programmable) for resetting BMC part or full chip
- Configurable reset selection control for each BMC function module

1.3.34 Real Time Clock (RTC)

- Directly connected to APB bus
- Reference clock is divided from CLKIN (25MHz) input
- Support Full Calendar function with correct leap years
- Clock mode for calculating:
  - seconds (0-59)
  - minutes (0-59)
  - hours (0-23)
  - days of month (1-28,29,30,31)
  - month (1-12)
  - year (0-99)
  - century (0-31)
- Programmable alarm with interrupt generation
  - Periodic alarm for second, minute, hour or day setting separately
  - Periodic alarm for a specified day/hour/minute/second time within a month
- Maskable interrupt
- No battery backup supported

1.3.35 LPC Bus Interface

- Directly connected to APB bus interface
- Maximum running frequency: 33MHz
- Operation mode
  - Salve mode: designed for BMC functions (I/O read write cycles) and SBIOS boot (memory or firmware read write cycles)
- Support Serial IRQ (reduce polling time)
- Support port 80H/81H (programmable address) snooping registers with interrupt options and size 256 bytes (FIFO mode) or 64MB (DMA mode)
- Support 2 virtual UART for Serial-over-Lan (SOL) application
- Support up to 4 sets of KCS mode registers and 1 BT mode registers (IPMI 2.0 Complaint), or 3 sets of KCS mode registers and 1+1 BT mode registers
- Support IPMI [SOL] serial port sharing

1.3.36 eSPI Interface

- Support 66, 50, 33, 25, and 20MHz of eSPI clock frequency
- Support both alert mode, shared IO and dedicate
- Support Quad, Dual, and Single IO mode
- Support 4 channels
- Shared same pins with LPC device interface
- Peripheral channel:
  - Maximum payload size is 64 bytes
- Virtual Wire channel:
  - Maximum virtual wire count is 8
  - Support 32 interrupts
  - Support 32 GPIOs
- Out-of-Band channel:
  - Maximum payload size is 64 bytes
- Run-time Flash Sharing channel:
  - Maximum payload size is 64 bytes
  - Support both master- and slave-attached flash sharing

1.3.37 System SPI Flash Controller

- Support 3 types of application mode, which can be defined at hardware strap or exchange on-the-fly
- SPI Master (the same as SPI1 master of BMC):
  - Connected to the AHB bus, can be accessed by ARM, Coprocessor or from LPC/eSPI interface by host system
  - Support system boot BIOS flash memory
  - Support 1, 2 and 4 bits input/output command modes
  - Support 24 or 32 bits address mode.
  - Support 2 chip select pins
  - Support write command and address filter function
  - Support register lock until reset function

1.3.38 Super I/O Controller

- Directly Connected to LPC bus controller & AHB Bus
- Support 4 sets of 16550-compliant UART controllers with full flow control pins
- Support port 80h/81h (programmable address) snooping registers with interrupt options and size 256 bytes (FIFO mode) or 64MB (DMA mode)
- Support Port 80h/81h redirect to two GPIO Groups for LED indicator
- Support Port 80h/81h redirect to Serial GPIO Port1 & Port2 for LED indicator
- Support 7+8 general purpose registers for communication between the host and the BMC with interrupt capability

- Support mailbox with 32 registers
- Support ACPI/PM logic
  - ACPI Complaint
  - SMI Support
  - SCI Support
  - SerIRQ Support
  - S3# and S5# Support
  - Programmable Wake-up Events
  - Plug and Play Register Set
  - Power Supply Control
  - Power Button Control
  - GPIO Support

1.3.39 Hash & Crypto Engine

- Directly connected to APB bus
- Direct data access through internal memory bus
- Programmable AES/DES/3DES/RC4 encryption or decryption mode with programmable context buffer location for fast context switching
- Support RSA algorithm hardware acceleration. The supported bit numbers for modulator and exponent are from 256 bits to 4096 bits.
- Support ECC algorithm hardware acceleration. The curves are NIST Prime Fields P-192, P-224, P-256, P-384.
- Support multiple message digest standards: MD5/SHA1/SHA2-224/SHA2-256/SHA2-384/SHA2-512, HMAC-MD5/HMAC-SHA1/HMAC-SHA224/HMAC-SHA256
- Support 4 types of engine trigger modes:
  - Encryption/decryption only
  - Message digest only
  - Encryption/decryption first, message digest second
  - Message digest first, encryption/decryption second
- Support AES crypto standard with the following modes:
  - Electronic Code Book (ECB), Cipher Block Chaining (CBC), Cipher Feedback (CFB), Output Feedback (OFB), Counter (CTR)
  - Support Three Different Key Sizes : 128, 192 or 256 bits
  - Key expansion task is done by software
  - Encryption/Decryption Throughput: $>$ 100M bps

1.3.40 ADC Controller

- Directly connected to APB bus
- Integrate 10-bit analog-to-digital converter (ADC)
- Support 16 low-leakage ($<$ 1.0uA) inputs to measure up to 16 analog voltages
- Integrate band-gap reference voltage generator with 2% precision
- Support intelligent hardware monitor function for all of the 16 analog input with interrupt option
- All of the 16 ADC channel pins can be programmed as GPI pins of the GPIO controller

1.3.41 PWM Controller

- Support 16 PWM outputs
- Each PWM output has its own PWM frequency
- Duty cycle from 0 to 100% with 1/256 resolution
- Shared with GPIO pins

1.3.42 Fan Tachometer Controller

- Directly connected to AHB bus
- Support up to 16 tachometer inputs
- Measurement schemes: rising edge, falling edge or both edges
- Support Interrupt trigger when over fan speed limitation setting
- Shared with DVO input pins

1.3.43 PECI Controller

- Directly connected to APB bus
- Intel PECI 4.1 compliant
- Maximum packet length is 256 bytes (Baseline transmission unit)
- Support up to 8 CPUs and 2 domains per CPU
- Integrate PECI compliant I/O buffers, can connect to PECI bus directly
- Transmit buffer 64 bytes and receive buffer 64 bytes

1.3.44 JTAG Master Controller

- Directly connected to APB bus
- TCK is divided from AHB clock with a programmable value
- Support flexible instruction value
- Support interrupt when transmission pause or complete
- Support software mode which controls TCK, TMS, and TDI directly through APB
- Support Intel At-Scale Debug

1.3.45 MCTP Controller

- Directly connected to APB bus
- Support maximum packet length 64/128/256/512 bytes.
- Independent send and receive message engine
- Flexible PCI Express header
- Interrupt for receiving or sending completion
- Integrated 8KB internal buffer. It can be programmed as 2K/6K, 4K/4K, or 6K/2K for TX and RX, respectively.
- Support DMA option

1.3.46 MSI Controller

- 2 exclusive interrupt types: INTx and MSI(Message Signaled Interrupt)
- Support 4 interrupt source
- 1 of 4 interrupt source is controlled by APB

1.3.47 X-DMA Controller

- Transfer data between memory of BMC and Host.
- Independent descriptor space storged in memory for BMC and Host.
- Both BMC and Host can transfer data upward(BMC to Host) or downward(Host to BMC).
- Programmable pitch for line-based transfer.
- Each descriptor can interrupt BMC or Host.
- Interrupt for completion or receive non-successful completion.

1.3.48 Software Specifications

- Manufacturer Test Program under UEFI for VGA testing only
- VGA BIOS and flash utility under UEFI
- Windows Driver
  - Support Windows Server 2012 R2 x86/x64 (with WHQL)
  - Support Windows Server 2016 x86/x64 (with WHQL)
  - Support Windows Server 2019 x86/x64 (with WHQL)
- Linux X Window Driver
  - Support XORG 7.x
- FreeBSD X Window Driver
  - Support XORG 7.x
- Solaris 11.4 x86
  - Support XORG 7.x
- Supported Linux Distribution:
  - RHEL LTS
  - Ubuntu LTS

1.4 Features Comparison (AST2600/AST2500/AST2400)

The following table shows the major feature comparisons between AST2600, AST2500 and AST2400 product specification.

Table 1: Comparisons Between AST2600, AST2500 and AST2400

Feature	AST2600	AST2500	AST2400
VGA/2D Controller	Yes	Yes	Yes
VGA Bus Interafce	PCI Express Gen2	PCI Express Gen2	PCI Express Gen1
PCI Express Root Complex	PCI Express Gen2	PCI Express Gen2	No
SOC Display Controller	Yes	Yes	Yes
Storage Redirection	Yes	Yes	Yes
KVM Redirection	Yes	Yes	Yes
Package	21mmx21mm TFBGA	19mmx19mm TFBGA	19mmx19mm LFBGA
Pin Count Ball Pitch	624 Pins 0.8 mm	456 Pins 0.8 mm	408 Pins 0.8 mm
Peak Power Consumption (include DRAM)	< 2W	< 2.1W	< 2.7W
Embedded CPU	ARM Coetex A7 Dual Core	ARM1176JZS	ARM926EJ-S
CPU Frequency	1.2GHz (max)	800MHz (max)	400MHz (max)
32Bit Coprocessor	200MHz (Cortex M3)	200MHz (max)	No
SDRAM Memory Bus Width	16 Bits	16 Bits	16 Bits
SDRAM Memory Types	DDR4	DDR3L/DDR4	DDR2/DDR3
Maximum Memory Capacity	2048MB	1024MB	512MB
Maximum Memory Clock Frequency	800MHz/1600Mbps	800MHz/1600Mbps	400MHz/800Mbps
ECC Support	Yes (1/8 size)	Yes (1/8 size)	Yes (1/8 size)
Maximum Graphics Display Resolutions	1920x1200 32bpp@60Hz	1920x1200 32bpp@60Hz	1920x1200 32bpp@60Hz
Maximum Video Clock Frequency	165MHz	165MHz	165MHz
USB 2.0 Virtual Hub Controller	Yes (x1)	Yes (x1)	Yes (x1)
USB 2.0 Device Controller	Yes (x1)	~~Yes (x1)~~	No
USB 1.1 HID Controller	Yes (x1)	Yes (x1)	Yes (x1)
USB 2.0 Host Controller	Yes (x2)	Yes (x2)	Yes (x1)
USB 1.1 Host Ports	Yes (x2)	Yes (x2)	Yes (x2)
USB2.0 Host Controller on PCIe	Yes	No	No
Hash Engine	Yes (MD5/SHA)	Yes (MD5/SHA)	Yes (MD5/SHA)
Crypto Engine	Yes (AES/RC4/DES/3DES)	Yes (AES/RC4/DES/3DES)	Yes (AES/RC4/DES/3DES)
Asynchronous Crypto Engine	Yes (RSA/ECC)	Yes (RSA)	Yes (RSA)
I2C/SMBus Controller	Yes (x16)	Yes (x14)	Yes (x14)
PWM Outputs	Yes (x16)	Yes (x8)	Yes (x8)
Fan Tech	Yes (x16)	Yes (x16)	Yes (x16)
PECI 4.0	Yes	Yes (W/R Length ¡ 32)	No
PECI IO Buffer	Yes	Yes	Yes
GPIO	244 (max)	226 (max)	216 (max)
SGPIO Master	128 bit + 80 bit (2 sets)	80 (max)	80 (max)
SGPIO Monitor	10 devices x2ch x2	10 devices x2ch x2	10 devices x2ch
UART	Yes (x13) (Flow control x4)	Yes (x5) (Flow control x4)	Yes (x5) (Flow control x4)
UART Baudrate	3686.4K	921.6K	115.2K

to next page

from previous page

Virtual UART	Yes (2)	Yes (1)	Yes (1)
System UART Interface	Yes (4)	Yes (4)	Yes (4)
UART DMA	Yes	Yes	No
Hardware UART debug	Yes	Yes	No
LPC Bus Controller	Yes (Slave & Master)	Yes (Slave & Master)	Yes (Slave & Master)
eSPI Bus Controller	Yes (Slave)	Yes (Slave)	No
Watchdog Timer	Yes (x8)	Yes (x3)	Yes (x2)
Timer	Yes (x8)	Yes (x8)	Yes (x8)
Real Time Clock (RTC)	Yes	Yes	Yes
Digital Video Output	No	Yes (24-Bit Single Edge or 12-Bit Dual Edge)	Yes (24-Bit Single Edge or 12-Bit Dual Edge)
DisplayPort 1.1a Output	Yes	No	No
Digital Video Input	No	Yes (24-Bit Single Edge)	Yes (24-Bit Single Edge)
Ethernet MAC Module	Yes (x4)	Yes (x2)	Yes (x2)
Ethernet MAC Throughput	10/100/1000M bps	10/100/1000M bps	10/100/1000M bps
Ethernet MAC Interface	RGMII/RMII (x4) or NCSI (x2)	RGMII (x2) or RMII/NCSI (x2)	RGMII (x2) or RMII/NCSI (x2)
Flash Memory Controller	SPI Flash	SPI Flash	SPI Flash NOR Flash (x8/x16) NAND Flash
SD/SDIO/eMMC 4-bits Interface eMMC 8-bits Interface	Yes (x2) and Yes (x1)	Yes (x2) or Yes (x1)	Yes (x2) or No
Embedded SRAM	Yes (64KB)	Yes (36KB)	Yes (32KB)
Battery Backed SRAM	Yes (128 bytes)	Yes (64 bytes)	No
Chip unique ID	Yes (64 bits)	Yes (64 bits)	No
System SPI Interface	Yes	Yes	Yes
MCTP function	Yes (with DMA)	Yes (with DMA)	Yes (with DMA)
MSI function	Yes	Yes	Yes
DMA between System and BMC memory	Yes	Yes	Yes
Super I/O Function	Yes	Yes	Yes
ADC Function	Yes (16 voltage)	Yes (16 voltage)	Yes (16 voltage)
ADC reference voltage	2 reference voltages	No	No
JTAG-boundary scan	Yes	Yes	Yes
JTAG Master Function	Yes (x2)	Yes (x1)	Yes

1.5 Features Comparison (AST2605/AST2600/AST2620)

The following table shows the major feature comparisons between AST2600 / AST2605 / AST2620 product specification.

Table 2: Comparisons Between AST2600, AST2605 and AST2620

Feature	AST2605	AST2600	AST2620
VGA/2D Controller	Yes	Yes	No
VGA Bus Interafce	PCI Express Gen2	PCI Express Gen2	No
PCI Express Root Complex	PCI Express Gen2	PCI Express Gen2	PCI Express Gen2
SOC Display Controller	Yes	Yes	No
Storage Redirection	Yes	Yes	Yes
KVM Redirection	Yes	Yes	No
Package	21mmx21mm TFBGA	21mmx21mm TFBGA	21mmx21mm TFBGA
Pin Count Ball Pitch	624 Pins 0.8 mm	624 Pins 0.8 mm	624 Pins 0.8 mm
Peak Power Consumption (include DRAM)	< 2W	< 2W	TBD
Embedded CPU	ARM Coetex A7 Dual Core	ARM Coetex A7 Dual Core	ARM Coetex A7 Dual Core
CPU Frequency	1.2GHz (max)	1.2GHz (max)	1.2GHz (max)
32Bit Coprocessor	No	200MHz (Cortex M3)	200MHz (Cortex M3)
32Bit Secure Processor	200MHz (Cortex M3)	No	No
SDRAM Memory Bus Width	16 Bits	16 Bits	16 Bits
SDRAM Memory Types	DDR4	DDR4	DDR4
Maximum Memory Capacity	2048MB	2048MB	2048MB
Maximum Memory Clock Frequency	800MHz/1600Mbps	800MHz/1600Mbps	800MHz/1600Mbps
ECC Support	Yes (1/8 size)	Yes (1/8 size)	Yes (1/8 size)
Maximum Graphics Display Resolutions	1920x1200 32bpp@60Hz	1920x1200 32bpp@60Hz	No
Maximum Video Clock Frequency	165MHz	165MHz	No
USB 2.0 Virtual Hub Controller	Yes (x1)	Yes (x1)	Yes (x1)
USB 2.0 Device Controller	Yes (x1)	Yes (x1)	Yes (x1)
USB 1.1 HID Controller	Yes (x1)	Yes (x1)	Yes (x1)
USB 2.0 Host Controller	Yes (x2)	Yes (x2)	Yes (x2)
USB 1.1 Host Ports	Yes (x2)	Yes (x2)	Yes (x2)
USB2.0 Host Controller on PCIe	Yes	Yes	No
Hash Engine	Yes (MD5/SHA)	Yes (MD5/SHA)	Yes (MD5/SHA)
Crypto Engine	Yes (AES/RC4/DES/3DES)	Yes (AES/RC4/DES/3DES)	Yes (AES/RC4/DES/3DES)
Asynchronous Crypto Engine	Yes (RSA/ECC)	Yes (RSA/ECC)	Yes (RSA/ECC)
I2C/SMBus Controller	Yes (x16)	Yes (x16)	Yes (x16)
PWM Outputs	Yes (x16)	Yes (x16)	Yes (x16)
Fan Tech	Yes (x16)	Yes (x16)	Yes (x16)
PECI 4.0	Yes	Yes	Yes
PECI IO Buffer	Yes	Yes	Yes
GPIO	244 (max)	244 (max)	244 (max)
SGPIO Master	128 bit + 80 bit (2 sets	128 bit + 80 bit (2 sets)	80 (max)
SGPIO Monitor	10 devices x2ch x2	10 devices x2ch x2	10 devices x2ch x2
UART	Yes (x13) (Flow control x4)	Yes (x13) (Flow control x4)	Yes (x13) (Flow control x4)

to next page

from previous page

UART Baudrate	3686.4K	3686.4K	3686.4K
Virtual UART	Yes (2)	Yes (2)	Yes (2)
System UART Interface	Yes (4)	Yes (4)	Yes (4)
UART DMA	Yes	Yes	Yes
Hardware UART debug	Yes	Yes	Yes
LPC Bus Controller	Yes (Slave & Master)	Yes (Slave & Master)	Yes (Slave & Master)
eSPI Bus Controller	Yes (Slave)	Yes (Slave)	Yes (Slave)
Watchdog Timer	Yes (x8)	Yes (x8)	Yes (x8)
Timer	Yes (x8)	Yes (x8)	Yes (x8)
Real Time Clock (RTC)	Yes	Yes	Yes
Digital Video Output	No	No	No
DisplayPort 1.1a Output	Yes	Yes	No
Digital Video Input	No	No	No
Ethernet MAC Module	Yes (x4)	Yes (x4)	Yes (x4)
Ethernet MAC Throughput	10/100/1000M bps	10/100/1000M bps	10/100/1000M bps
Ethernet MAC Interface	RGMII/RMII (x4) or NCSI (x2)	RGMII/RMII (x4) or NCSI (x2)	RGMII/RMII (x4) or NCSI (x2)
Flash Memory Controller	SPI Flash	SPI Flash	SPI Flash
SD/SDIO/eMMC 4-bits Interface eMMC 8-bits Interface	Yes (x2) and Yes (x1)	Yes (x2) and Yes (x1)	Yes (x2) and Yes (x1)
Embedded SRAM	Yes (64KB)	Yes (64KB)	Yes (64KB)
Battery Backed SRAM	Yes (128 bytes)	Yes (128 bytes)	Yes (128 bytes)
Chip unique ID	Yes (64 bits)	Yes (64 bits)	Yes (64 bits)
System SPI Interface	Yes	Yes	Yes
MCTP function	Yes (with DMA)	Yes (with DMA)	Yes (with DMA)
MSI function	Yes	Yes	Yes
DMA between System and BMC memory	Yes	Yes	No
Super I/O Function	Yes	Yes	Yes
ADC Function	Yes (16 voltage)	Yes (16 voltage)	Yes (16 voltage)
ADC reference voltage	2 reference voltages	2 reference voltages	2 reference voltages
JTAG-boundary scan	Yes	Yes	Yes
JTAG Master Function	Yes (x2)	Yes (x2)	Yes (x2)

1.6 Features Comparison (AST2600A0/AST2600A1/AST2600A2/A3)

The following table shows the major feature comparisons between AST2600A0, AST2600A1 and AST2600A2/A3 product specification.

Table 3: Comparisons Between AST2600A0, AST2600A1 and AST2600A2/A3

Feature	AST2600A0	AST2600A1	AST2600A2/A3
Power on default ARM CA7 frequency	800MHz	1.2GHz	1.2GHz
Bus and Memory TrustZone capability	No	Yes	Yes
Embedded SRAM size	64KB	64KB + 24KB + 1KB	64KB + 24KB + 1KB
IO driving strength for RGMII1/RGMII2	O8	O16	O16
IO driving strength for SPI bus	O8	O16	O16
SOC status Heart Beat output	No	Yes	Yes
GPIO pass-through	4 pairs	3 pairs	3 pairs
GPIO pass-through enable by pin strap	No	Yes	Yes
SGPIO master	80 bit	128 bit + 80 bit (2 sets)	128 bit + 80 bit (2 sets)
SGPIO slave	32 bit	32 bit + 32 bit (2 sets)	32 bit + 32 bit (2 sets)
LPC Memory/FWH to AHB bridge	No	Yes	Yes
PostCode to GPIO	80h to GPIOx8	80h+81h to GPIOx16	80h+81h to GPIOx16
MailBox buffer	16 bytes	32 bytes	32 bytes
Watchdog timer	4 sets	8 sets	8 sets
Secure Boot AES GCM mode	No	Yes	Yes
Boot from UART	No	Yes	Yes
Boot from eMMC	No	Yes	Yes
TruztZone support for WDT ***1	No	No	Yes
TruztZone support for eSPI ***2	No	No	Yes
TruztZone support for I2C (I2CS) ***3	No	No	Yes
TruztZone support for GPIO ***8	No	No	Yes
New hardstrap pins ***4	No	No	Yes
Seperated Uart DMA registers ***5	No	No	Yes
SBE: Support boot from VUART	No	No	Yes
SBE: Support DICE	No	No	Yes
CHASI# raw read back ***6	No	No	Yes
Core and I/O power detection ***7	No	No	Yes
I/O and Command for LPC/eSPI SAFS ***9	No	No	Yes

***1: Reference to revision history v0.7 - 1.
***2: Reference to revision history v0.7 - 1.
***3: Reference to revision history v0.7 - 3.
***4: Reference to revision history v0.7 - 6.
***5: Reference to revision history v0.7 - 2.
***6: Reference to revision history v0.7 - 7.
***7: Reference to revision history v0.7 - 7.
***8: Reference to revision history v0.7 - 12.
***9: Reference to revision history v0.7 - 13.

1.7 Applications

1.7.1 Display Output Interface

1. Path 1: VGA output, the output target can choose either or both to the DAC or DisplayPort 1.1a interface.
2. Path 2: Graphics CRT output, the output target can choose either or both to the DAC or DisplayPort 1.1a interface.

1.7.2 USB Port Configuration

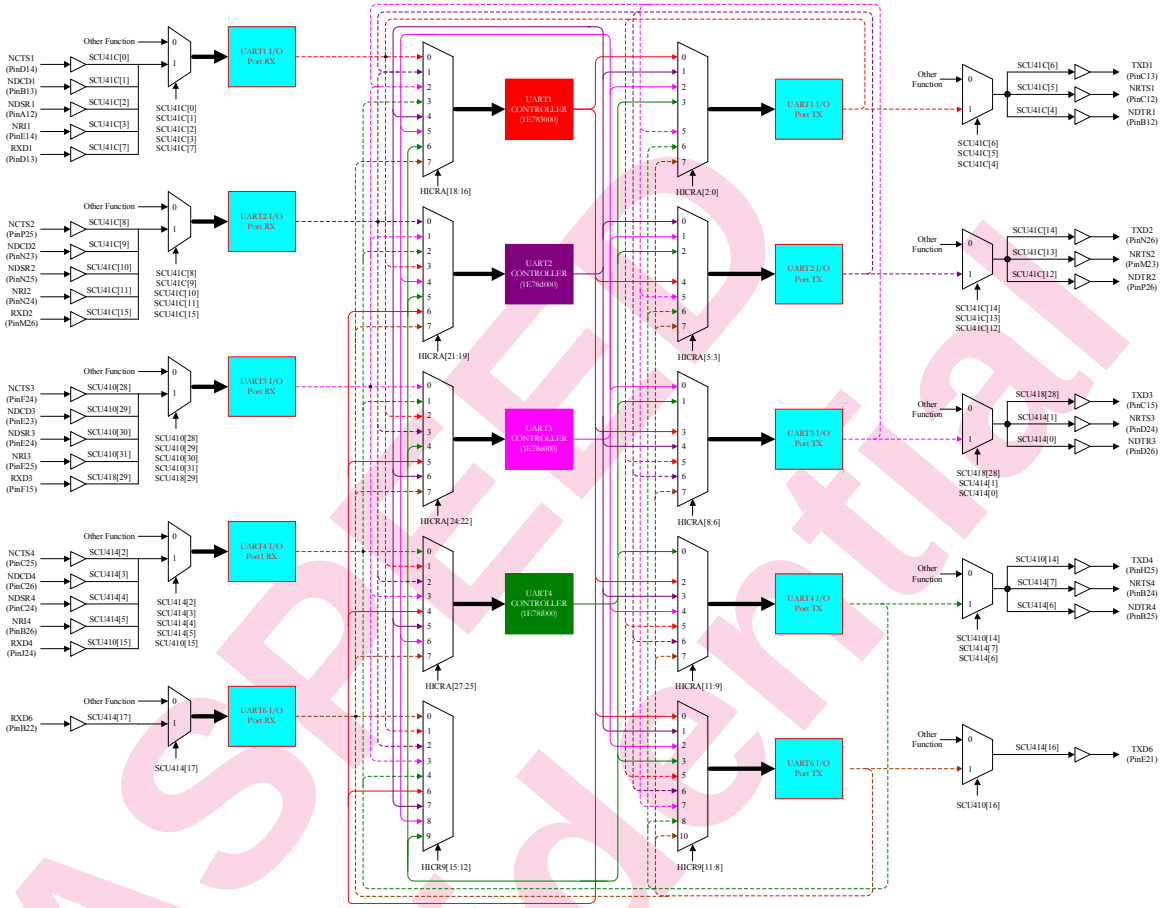
AST2600 supports 2 USB ports, which can be configured as below application types.

- USB port A:
  1. USB2.0 Full/High Speed Virtual Hub device, function as remote USB storage/keyboard/mouse redirection.
  2. USB2.0 Full/High Speed Host port.
- USB port B:
  1. USB1.1 Low Speed HID device, function as remote USB keyboard/mouse redirection. This is a backup port now, firmware mainly use the USB2.0 Virtual Hub controller to emulate the keyboard/mouse redirection function.
  2. USB2.0 Full/High Speed device, an extended general purpose USB2.0 device port, which can be used for different BMC chips link, or other purpose.
  3. USB2.0 Full/High Speed Host port.

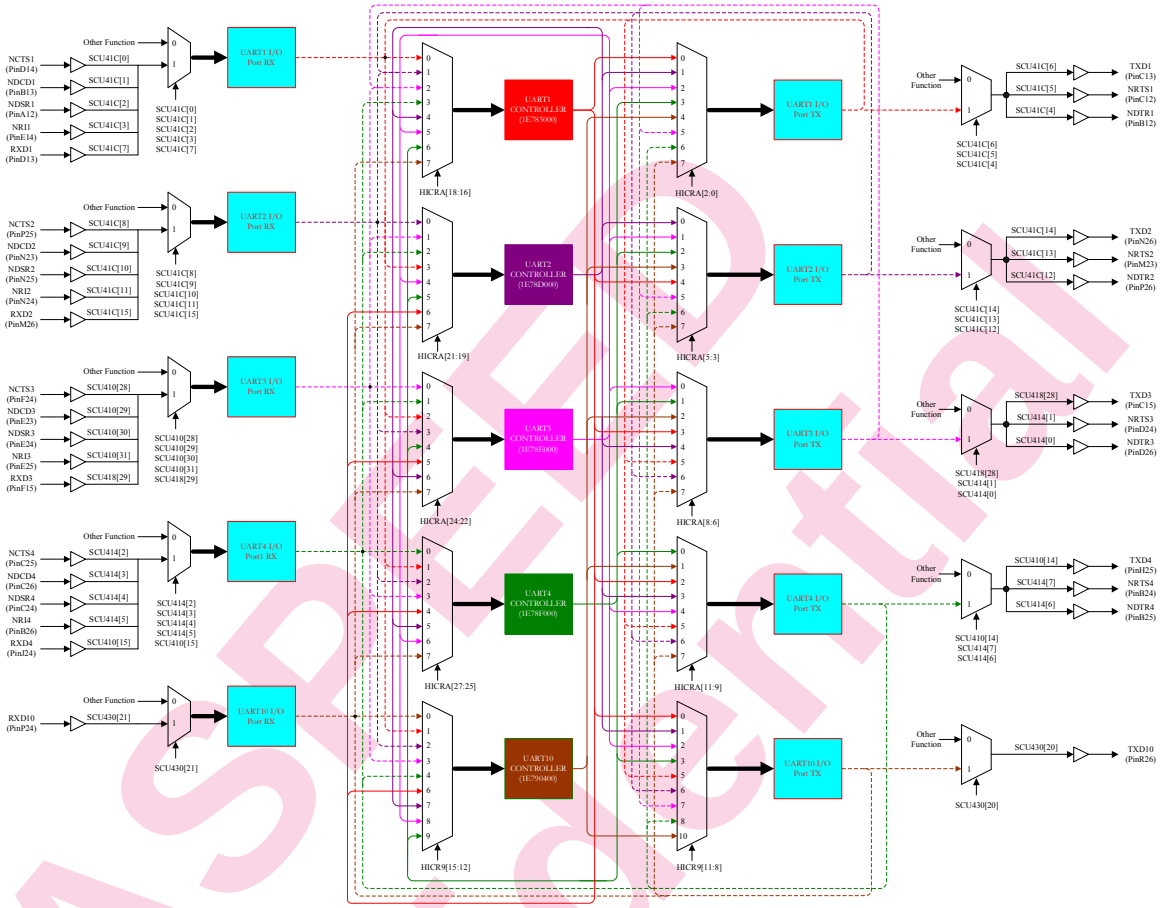
1.7.3 UART routing topology

AST2600 UARTs designed with a highly flexibile internal routing to meet any types of system design requirements. Below figure shows the topology of chip internal UART routing. The UART routing is controlled by LPC controller register HICR9 and HICRA.
Please notice UART routing topology is not the same between A0/A1 and A2. Please check below figure for more detail.

UART routing topology for A0/A1:



UART routing topology for A2/A3:



1.7.4 Flash Interface Architecture

AST2600 include three independent SPI controller for SPI flash access. Please refer to the Flash Interface Architecture as below figuew.

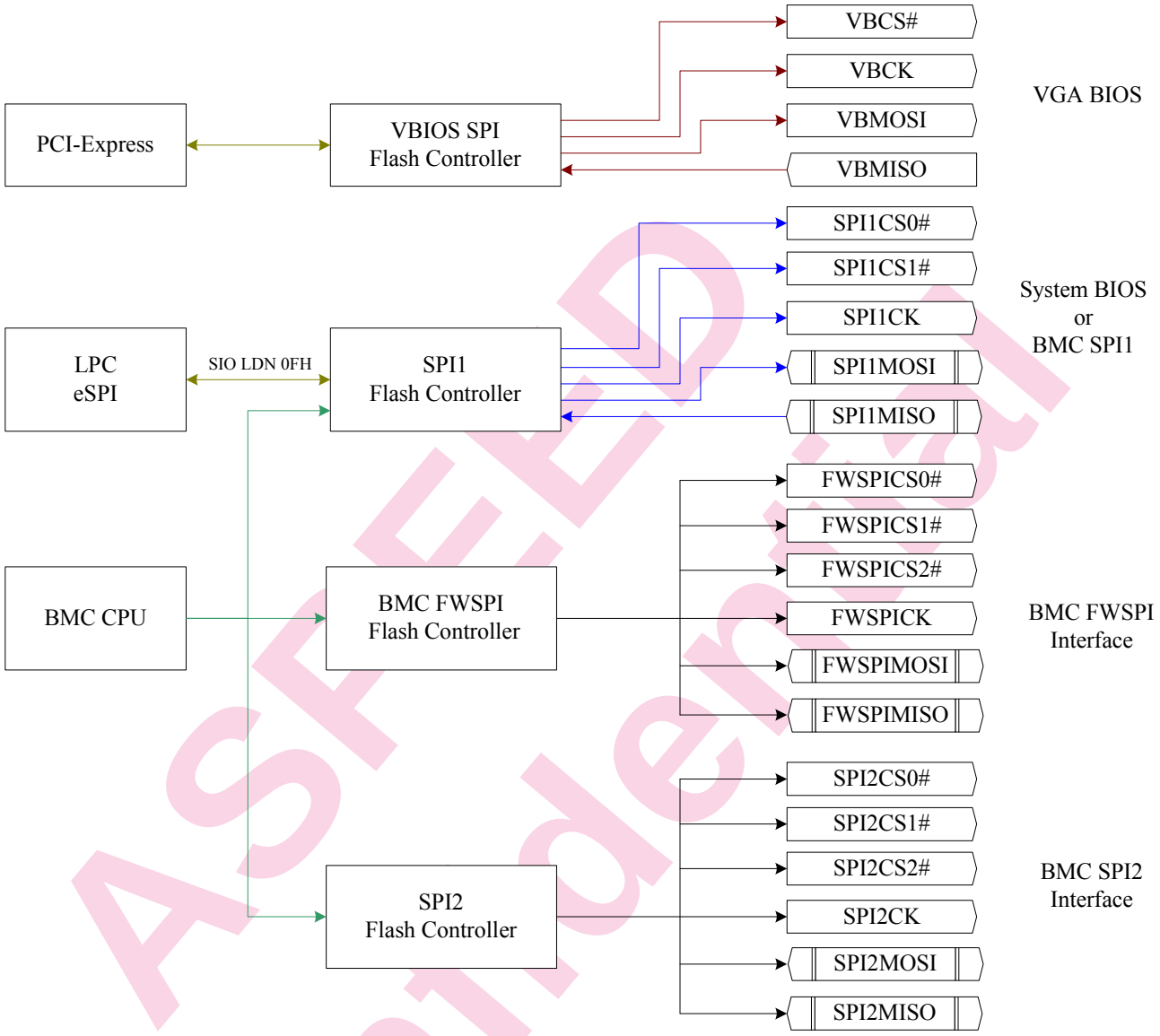


Figure 2: AST2600 Flash Interface Architecture

1.7.5 I3C I/O Diagram

AST2600 include six I3C controller internally and voltage level can support High voltage level (1.8V) or Low voltage level (1.0V). Please refer to the I/O mapping between internal controller and external I/O as below figure.

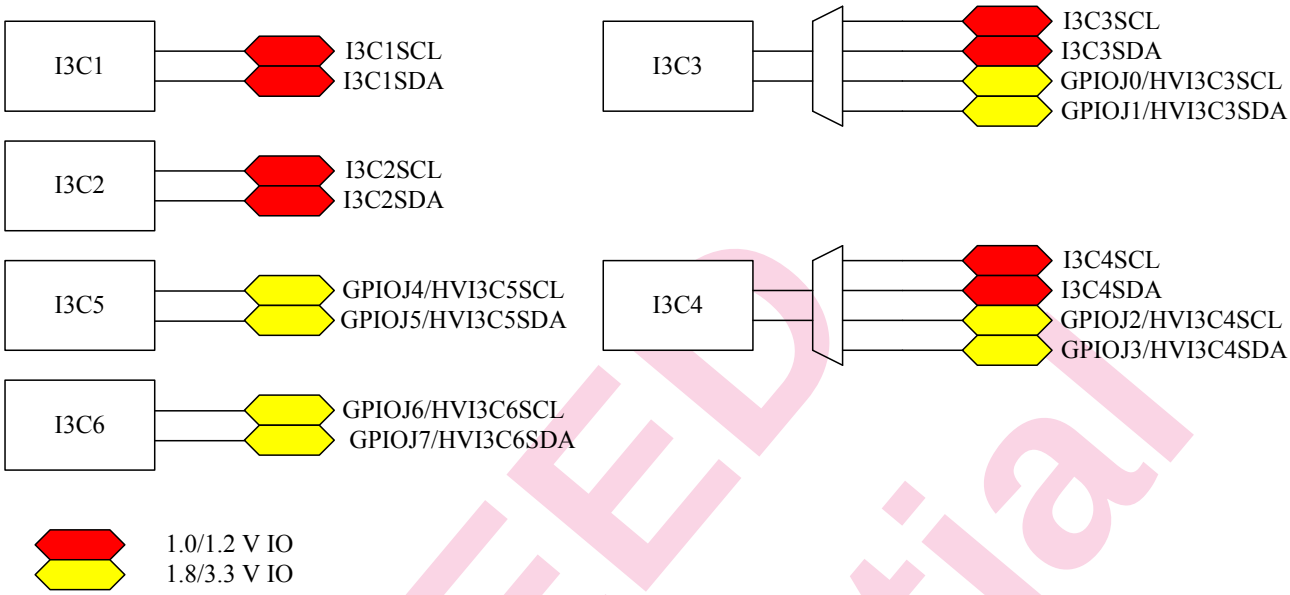


Figure 3: AST2600 I3C I/O Diagram

2 Pin & I/O Related Specification

2.1 Pin Description

Notice: This chip do not support 5V tolerant I/O.

Abbreviation Definition:

Symbol	:	Description
#	:	Denotes active low signal
I	:	Input buffer
IU	:	Input buffer with internal pull high resistor (58 ~ 133 KΩ)
ID	:	Input buffer with internal pull low resistor (52 ~ 128 KΩ)
IUS	:	Schmitt-trigger type input buffer with internal pull high resistor (58 ~ 133 KΩ)
IDS	:	Schmitt-trigger type input buffer with internal pull low resistor (52 ~ 128 KΩ)
IR	:	Input buffer with programmable internal pull low resistor (52 ~ 128 KΩ), default OFF
IS	:	Schmitt-trigger type input buffer
On	:	Output buffer with different driving capability (n = 8, 12, 16)
Op	:	Output buffer with programmable (O8/O16) driving capability
P	:	Power/Ground pin
A	:	Analog pin
CMOS	:	3.3V CMOS protocol I/O buffer with only 3.3V tolerant input buffer.
CMOS12	:	1.2V/1.0V CMOS protocol I/O buffer with 1.2V or 1.0V tolerant input buffer.
CMOS18	:	1.8V CMOS protocol I/O buffer with only 1.8V tolerant input buffer.
DDR	:	DDR4 compliant SDRAM buffer type
VREF	:	Reference voltage

Note1 :
IU with internal pull high is only used for input buffer, it can not be used to drive external loads. The system design must use Standby power domain on all paths connected on these pins to prevent current leakage from the internal pull-up resistor.
Note2 :
Please refer section ?? for the firmware programming method of GPIO push-pull and open-drain driving mode.

Pin Group List

DDR4 DRAM Interface – 54 pins
PCI Express End Point Interface – 7 pins
PCI Express Root Complex Interface – 6 pins
DisplayPort Interface – 7 pins
LPC/ESPI Interface – 8 pins
1.8V RGMII/RMII Dual Interface – 24 pins
3.3V RGMII/RMII/NCSI Dual Interface – 33 pins
UART Port – 50 pins
Firmware SPI Memory Interface – 10 pins
Extend BMC SPI Memory Interface – 17 pins
I2C/SMBUS Interface – 32 pins
I3C Interface – 16 pins
SD/SDIO/eMMC Interface – 28 pins

Serial GPIO Interface – 8 pins
VGA Interface – 4 pins
PWM/Fan Tachometer – 32 pins
ACPI Interface – 11 pins
GPIO Interface – 10 pins
PECI Port – 2 pins
JTAG Slave / Master Port 1 – 5 pins
JTAG Master Port 2 – 5 pins
Miscellaneous – 7 pins
USB 2.0 Host/Slave Port – 6 pins
DAC – 6 pins
ADC – 26 pins
PLL Power – 7 pins
Power and Ground – 242 pins

DDR4 DRAM Interface – 54 pins

Ball	Signal	I/O	Type	Description
T3 R4 U1 R3 R1 P1 P2 P3 W2 V3 W1 W3 W4 U4 V4 U3	MDQ0 MDQ1 MDQ2 MDQ3 MDQ4 MDQ5 MDQ6 MDQ7 MDQ8 MDQ9 MDQ10 MDQ11 MDQ12 MDQ13 MDQ14 MDQ15	I/O	DDR	**DRAM data bus**
N3 R5	MDM0 MDM1	O	DDR	**DRAM byte mask bus**
T2 V2	MDQS0 MDQS1	I/O	DDR	**DRAM data bidirectional strobe pins**
T1 V1	MDQS0# MDQS1#	I/O	DDR	**DRAM data bidirectional strobe pins complement phase**
F3 K5 F1 H3 J3 L1 M5 M2	MA0 MA1 MA2 MA3 MA4 MA5 MA6 MA7	O	DDR	**DRAM address bus**

to next page

from previous page

M1 N1 F2 G1 L4 K3 H1 M3	MA8 MA9 MA10 MA11 MA12 MA13 MA14/MACT# MA15/MBG1			**MACT# for DDR4 DRAM** **MBG1 for DDR4 Twin-die DRAM**
G4 H5 G3	MBA0 MBA1 MBA2/MBG0	O	DDR	**DRAM bank address** MBG0 for DDR4 DRAM
K2	MCK	O	DDR	**DRAM clock pin (Reference MVDD_CK)**
K1	MCK#	O	DDR	**DRAM clock pin complement phase (Reference MVDD_CK)**
H2	MCS#	O	DDR	**DRAM chip select pin**
J5	MRAS#	O	DDR	**DRAM row address select pin** **It is also used as MA16 funtion for DDR4**
J4	MCAS#	O	DDR	**DRAM column address select pin** **It is also used as MA15 funtion for DDR4**
G5	MWE#	O	DDR	**DRAM write enable pin** **It is also used as MA14 funtion for DDR4**
L5	MCKE	O	DDR	**DRAM clock enable control pin (Reference MVDD_CK)**
J1	MODT	O	DDR	**On-die termination enable control**
L3	MRESET#	O	DDR	**DRAM reset pin (Reference MVDD_CK)**
N4	MALERT#	I/O	DDR	**DDR4 Alert function pin**
P5	MIOZ	-	A	**IO calibration reference pin** An external 240 Ω±1% resistor should be connected between MIOZ and ground.
T5 U5	MVREF MVREF	-	VREF	**IO reference voltage** DDR IO reference voltage input pin. The voltage is 0.5*MVDD.

PCI Express End Point Interface – 7 pins

Ball	Signal	I/O	Type	Description
AD5	PEREFCLKP	I	A	**PCI Express Reference clock input** 100MHz positive input of the differential clock pair
AD6	PEREFCLKN	I	A	**PCI Express Reference clock input** 100MHz negative input of the differential clock pair
AF5	PERXP	I	A	**PCI Express Serial Data Receiver** It receives positive input of the differential signal pair.
AF6	PERXN	I	A	**PCI Express Serial Data Receiver** It receives negative input of the differential signal pair.
AE4	PETXP	O	A	**PCI Express Serial Data Transmitter** It transmits positive output of the differential signal pair.
AE5	PETXN	O	A	**PCI Express Serial Data Transmitter** It transmits negative output of the differential signal pair.
PERST# Signal Power Domain : PV33D_RGM				

to next page

from previous page

A7	PERST#	IU/O8	CMOS	**PCI Express reset pin** This reset signal reset PCI Express bus controller and VGA/2D device.

PCI Express Root Complex Interface – 6 pins

Ball	Signal	I/O	Type	Description
AE1	RCREFCLKP	I	A	**PCI Express Root Complex Port differential Clock input.** 100MHz positive input of the differential clock pair
AE2	RCREFCLKN	I	A	**PCI Express Root Complex Port differential Clock input.** 100MHz negative input of the differential clock pair
AD2	RCRXP	I	A	**PCI Express Root Complex Port differential Receive Signal.** It receives positive input of the differential signal pair.
AD3	RCRXN	I	A	**PCI Express Root Complex Port differential Receive Signal.** It receives negative input of the differential signal pair.
AF2	RCTXP	O	A	**PCI Express Root Complex Port differential Transmit Signal.** It transmits positive output of the differential signal pair.
AF3	RCTXN	O	A	**PCI Express Root Complex Port differential Transmit Signal.** It transmits negative output of the differential signal pair.

DisplayPort Interface – 7 pins

Ball	Signal	I/O	Type	Description
AC3	DPAUXP	I/O	A	**DisplayPort Aux channel differential output.** It transmits/receives positive output of the differential signal pair.
AB3	DPAUXN	I/O	A	**DisplayPort Aux channel differential output.** It transmits/receives negative output of the differential signal pair.
AC1	DPTXP0	O	A	**DisplayPort Data channel differential bus 0 output.** It transmits positive output of the differential signal pair.
AB1	DPTXN0	O	A	**DisplayPort Data channel differential bus 0 output.** It transmits negative output of the differential signal pair.
AB2	DPTXP1	O	A	**DisplayPort Data channel differential bus 1 output.** It transmits positive output of the differential signal pair.
AA2	DPTXN1	O	A	**DisplayPort Data channel differential bus 1 output.** It transmits negative output of the differential signal pair.
DPHPD Signal Power Domain : PV33D_RGM				
C7	DPHPD	IS	CMOS	**DisplayPort Hot plug detection.**

LPC/ESPI Interface – 8 pins

Ball	Signal	I/O	Type	Description
IO Power Domain : LPVDD				
AB7	GPIOW0	I/O8	CMOS	**GPIO group W bit 0 (default)**
	LAD0	I/O8	CMOS	**LPC address and data bus bit 0**
	ESPID0	I/O8	CMOS18	**eSPI data bus bit 0**
AB8	GPIOW1	I/O8	CMOS	**GPIO group W bit 1 (default)**
	LAD1	I/O8	CMOS	**LPC address and data bus bit 1**
	ESPID1	I/O8	CMOS18	**eSPI data bus bit 1**
AC8	GPIOW2	I/O8	CMOS	**GPIO group W bit 2 (default)**
	LAD2	I/O8	CMOS	**LPC address and data bus bit 2**

to next page

from previous page

	ESPID2	I/O8	CMOS18	**eSPI data bus bit 2**
AC7	GPIOW3	I/O8	CMOS	**GPIO group W bit 3 (default)**
	LAD3	I/O8	CMOS	**LPC address and data bus bit 3**
	ESPID3	I/O8	CMOS18	**eSPI data bus bit 3**
AE7	GPIOW4	I/O8	CMOS	**GPIO group W bit 4 (default)**
	LCLK	I	CMOS	**LPC bus clock input**
	ESPICK	I	CMOS18	**eSPI clock input**
AF7	GPIOW5	I/O8	CMOS	**GPIO group W bit 5 (default)**
	LFRAME#	I	CMOS	**LPC FRAME#**
	ESPICS#	I	CMOS18	**eSPI chip select input**
AD7	GPIOW6	I/O8	CMOS	**GPIO group W bit 6 (default)**
	LSIRQ#	I/O8	CMOS	**LPC serial IRQ**
	ESPIALT#	O8	CMOS18	**eSPI Alert**
AD8	GPIOW7	I/O8	CMOS	**GPIO group W bit 7 (default)**
	LPCRST#	I	CMOS	**LPC reset input**
	ESPIRST#	I/O8	CMOS18	**eSPI reset input**

1.8V RGMII/RMII Dual Interface – 24 pins

Ball	Signal	I/O	Type	Description
MAC1 IO Power Domain : PV18D				
C6	RGMII1TXCK	O16	CMOS18	**RGMII 1 transmit clock (1.8V only)**
	RMII1RCLKO	O16	CMOS18	**RMII 1 50MHz reference clock output (1.8V only)**
	GPIO18A0	ID/O16	CMOS18	**1.8V GPIO group A bit 0 (1.8V only) (default)** When used as RMII 50MHz reference clock output, this pin should be connected to RMII1RCLKI and PHY RCLK input.
D6	RGMII1TXCTL	O16	CMOS18	**RGMII 1 transmit control (1.8V only)**
	RMII1TXEN	O16	CMOS18	**RMII 1 transmit enable (1.8V only)**
	GPIO18A1	ID/O16	CMOS18	**1.8V GPIO group A bit 1 (1.8V only) (default)**
D5	RGMII1TXD0	O16	CMOS18	**RGMII 1 transmit data bus to PHY bit 0 (1.8V only)**
	RMII1TXD0	O16	CMOS18	**RMII 1 transmit data bus to PHY bit 0 (1.8V only)**
	GPIO18A2	ID/O16	CMOS18	**1.8V GPIO group A bit 2 (1.8V only) (default)**
A3	RGMII1TXD1	O16	CMOS18	**RGMII 1 transmit data bus to PHY bit 1 (1.8V only)**
	RMII1TXD1	O16	CMOS18	**RMII 1 transmit data bus to PHY bit 1 (1.8V only)**
	GPIO18A3	ID/O16	CMOS18	**1.8V GPIO group A bit 3 (1.8V only) (default)**
C5	RGMII1TXD2	O16	CMOS18	**RGMII 1 transmit data bus to PHY bit 2 (1.8V only)**
	GPIO18A4	ID/O16	CMOS18	**1.8V GPIO group A bit 4 (1.8V only) (default)**
E6	RGMII1TXD3	O16	CMOS18	**RGMII 1 transmit data bus to PHY bit 3 (1.8V only)**
	GPIO18A5	ID/O16	CMOS18	**1.8V GPIO group A bit 5 (1.8V only) (default)**
B3	RGMII1RXCK	I	CMOS18	**RGMII 1 receive clock (1.8V only)**
	RMII1RCLKI	I	CMOS18	**RMII 1 50MHz reference clock input (1.8V only)**
	GPIO18A6	ID/O16	CMOS18	**1.8V GPIO group A bit 6 (1.8V only) (default)**
A2	RGMII1RXCTL	I	CMOS18	**RGMII 1 receive control (1.8V only)**
	GPIO18A7	ID/O16	CMOS18	**1.8V GPIO group A bit 7 (1.8V only) (default)**

to next page

from previous page

B2	RGMII1RXD0	I	CMOS18	**RGMII 1 receive data bus from PHY bit 0 (1.8V only)**
	RMII1RXD0	I	CMOS18	**RMII 1 receive data bus from PHY bit 0 (1.8V only)**
	GPIO18B0	ID/O16	CMOS18	**1.8V GPIO group B bit 0 (1.8V only)** (default)
B1	RGMII1RXD1	I	CMOS18	**RGMII 1 receive data bus from PHY bit 1 (1.8V only)**
	RMII1RXD1	I	CMOS18	**RMII 1 receive data bus from PHY bit 1 (1.8V only)**
	GPIO18B1	ID/O16	CMOS18	**1.8V GPIO group B bit 1 (1.8V only)** (default)
C4	RGMII1RXD2	I	CMOS18	**RGMII 1 receive data bus from PHY bit 2 (1.8V only)**
	RMII1CRSDV	I	CMOS18	**RMII 1 receive carrier sense and data valid (1.8V only)**
	GPIO18B2	ID/O16	CMOS18	**1.8V GPIO group B bit 2 (1.8V only)** (default)
E5	RGMII1RXD3	I	CMOS18	**RGMII 1 receive data bus from PHY bit 3 (1.8V only)**
	RMII1RXER	I	CMOS18	**RMII 1 receive data error (1.8V only)**
	GPIO18B3	ID/O16	CMOS18	**1.8V GPIO group B bit 3 (1.8V only)** (default)
Note : MAC1 is 1.8V I/O only, please note it did not support NCSI 3.3V I/O application.				
MAC2 IO Power Domain : PV18D				
D4	RGMII2TXCK	O16	CMOS18	**RGMII 2 transmit clock (1.8V only)**
	RMII2RCLKO	O16	CMOS18	**RMII 2 50MHz reference clock output (1.8V only)**
	GPIO18B4	ID/O16	CMOS18	**1.8V GPIO group B bit 4 (1.8V only)** (default) When used as RMII 50MHz reference clock output, this pin should be connected to RMII1RCLKI and PHY RCLK input.
C2	RGMII2TXCTL	O16	CMOS18	**RGMII 2 transmit control (1.8V only)**
	RMII2TXEN	O16	CMOS18	**RMII 2 transmit enable (1.8V only)**
	GPIO18B5	ID/O16	CMOS18	**1.8V GPIO group B bit 5 (1.8V only)** (default)
C1	RGMII2TXD0	O16	CMOS18	**RGMII 2 transmit data bus to PHY bit 0 (1.8V only)**
	RMII2TXD0	O16	CMOS18	**RMII 2 transmit data bus to PHY bit 0 (1.8V only)**
	GPIO18B6	ID/O16	CMOS18	**1.8V GPIO group B bit 6 (1.8V only)** (default)
D3	RGMII2TXD1	O16	CMOS18	**RGMII 2 transmit data bus to PHY bit 1 (1.8V only)**
	RMII2TXD1	O16	CMOS18	**RMII 2 transmit data bus to PHY bit 1 (1.8V only)**
	GPIO18B7	ID/O16	CMOS18	**1.8V GPIO group B bit 7 (1.8V only)** (default)
E4	RGMII2TXD2	O16	CMOS18	**RGMII 2 transmit data bus to PHY bit 2 (1.8V only)**
	GPIO18C0	ID/O16	CMOS18	**1.8V GPIO group C bit 0 (1.8V only)** (default)
F5	RGMII2TXD3	O16	CMOS18	**RGMII 2 transmit data bus to PHY bit 3 (1.8V only)**
	GPIO18C1	ID/O16	CMOS18	**1.8V GPIO group C bit 1 (1.8V only)** (default)
D2	RGMII2RXCK	I	CMOS18	**RGMII 2 receive clock (1.8V only)**
	RMII2RCLKI	I	CMOS18	**RMII 2 50MHz reference clock input (1.8V only)**
	GPIO18C2	ID/O16	CMOS18	**1.8V GPIO group C bit 2 (1.8V only)** (default)
E3	RGMII2RXCTL	I	CMOS18	**RGMII 2 receive control (1.8V only)**
	GPIO18C3	ID/O16	CMOS18	**1.8V GPIO group C bit 3 (1.8V only)** (default)
D1	RGMII2RXD0	I	CMOS18	**RGMII 2 receive data bus from PHY bit 0 (1.8V only)**
	RMII2RXD0	I	CMOS18	**RMII 2 receive data bus from PHY bit 0 (1.8V only)**
	GPIO18C4	ID/O16	CMOS18	**1.8V GPIO group C bit 4 (1.8V only)** (default)
F4	RGMII2RXD1	I	CMOS18	**RGMII 2 receive data bus from PHY bit 1 (1.8V only)**
	RMII2RXD1	I	CMOS18	**RMII 2 receive data bus from PHY bit 1 (1.8V only)**
	GPIO18C5	ID/O16	CMOS18	**1.8V GPIO group C bit 5 (1.8V only)** (default)

to next page

from previous page

E2	RGMII2RXD2	I	CMOS18	**RGMII 2 receive data bus from PHY bit 2 (1.8V only)**
	RMII2CRSDV	I	CMOS18	**RMII 2 receive carrier sense and data valid (1.8V only)**
	GPIO18C6	ID/O16	CMOS18	**1.8V GPIO group C bit 6 (1.8V only)** (default)
E1	RGMII2RXD3	I	CMOS18	**RGMII 2 receive data bus from PHY bit 3 (1.8V only)**
	RMII2RXER	I	CMOS18	**RMII 2 receive data error (1.8V only)**
	GPIO18C7	ID/O16	CMOS18	**1.8V GPIO group C bit 7 (1.8V only)** (default)
Note : MAC2 is 1.8V I/O only, please note it did not support NCSI 3.3V I/O application.				

3.3V RGMII/RMII/NCSI Dual Interface – 33 pins

Ball	Signal	I/O	Type	Description
MAC3 IO Power Domain : RVDD				
H24	RGMII3TXCK	Op	CMOS	**RGMII 3 transmit clock**
	RMII3RCLKO	Op	CMOS	**RMII/NCSI 3 50MHz reference clock output**
	GPIOC0	I/O8	CMOS	**GPIO group C bit 0** (default) When used as RMII/NCSI 50MHz reference clock output, this pin should be connected to RMII1RCLKI and PHY RCLK input.
J22	RGMII3TXCTL	Op	CMOS	**RGMII 3 transmit control**
	RMII3TXEN	Op	CMOS	**RMII/NCSI 3 transmit enable**
	GPIOC1	I/O8	CMOS	**GPIO group C bit 1** (default)
H22	RGMII3TXD0	Op	CMOS	**RGMII 3 transmit data bus to PHY bit 0**
	RMII3TXD0	Op	CMOS	**RMII/NCSI 3 transmit data bus to PHY bit 0**
	GPIOC2	I/O8	CMOS	**GPIO group C bit 2** (default)
H23	RGMII3TXD1	Op	CMOS	**RGMII 3 transmit data bus to PHY bit 1**
	RMII3TXD1	Op	CMOS	**RMII/NCSI 3 transmit data bus to PHY bit 1**
	GPIOC3	I/O8	CMOS	**GPIO group C bit 3** (default)
G22	RGMII3TXD2	Op	CMOS	**RGMII 3 transmit data bus to PHY bit 2**
	GPIOC4	I/O8	CMOS	**GPIO group C bit 4** (default)
F22	RGMII3TXD3	Op	CMOS	**RGMII 3 transmit data bus to PHY bit 3**
	GPIOC5	I/O8	CMOS	**GPIO group C bit 5** (default)
G23	RGMII3RXCK	I	CMOS	**RGMII 3 receive clock**
	RMII3RCLKI	I	CMOS	**RMII/NCSI 3 50MHz reference clock input**
	GPIOC6	I/O8	CMOS	**GPIO group C bit 6** (default)
G24	RGMII3RXCTL	I	CMOS	**RGMII 3 receive control**
	GPIOC7	I/O8	CMOS	**GPIO group C bit 7** (default)
F23	RGMII3RXD0	I	CMOS	**RGMII 3 receive data bus from PHY bit 0**
	RMII3RXD0	I	CMOS	**RMII/NCSI 3 receive data bus from PHY bit 0**
	GPIOD0	I/O8	CMOS	**GPIO group D bit 0** (default)
F26	RGMII3RXD1	I	CMOS	**RGMII 3 receive data bus from PHY bit 1**
	RMII3RXD1	I	CMOS	**RMII/NCSI 3 receive data bus from PHY bit 1**
	GPIOD1	I/O8	CMOS	**GPIO group D bit 1** (default)
F25	RGMII3RXD2	I	CMOS	**RGMII 3 receive data bus from PHY bit 2**
	RMII3CRSDV	I	CMOS	**RMII/NCSI 3 receive carrier sense and data valid**
	GPIOD2	I/O8	CMOS	**GPIO group D bit 2** (default)

to next page

from previous page

E26	RGMII3RXD3	I	CMOS	**RGMII 3 receive data bus from PHY bit 3**
	RMII3RXER	I	CMOS	**RMII/NCSI 3 receive data error**
	GPIOD3	I/O8	CMOS	**GPIO group D bit 3 (default)**
MAC4 IO Power Domain : RVDD				
F24	RGMII4TXCK	Op	CMOS	**RGMII 4 transmit clock**
	RMII4RCLKO	Op	CMOS	**RMII/NCSI 4 50MHz reference clock output**
	NCTS3	I	CMOS	**UART 3 clear to send modem status**
	GPIOD4	I/O8	CMOS	**GPIO group D bit 4 (default)** When used as RMII/NCSI 50MHz reference clock output, this pin should be connected to RMII2RCLKI and PHY RCLK input.
E23	RGMII4TXCTL	Op	CMOS	**RGMII 4 transmit control**
	RMII4TXEN	Op	CMOS	**RMII/NCSI 4 transmit enable**
	NDCD3	I	CMOS	**UART 3 data carrier detect modem status**
	GPIOD5	I/O8	CMOS	**GPIO group D bit 5 (default)**
E24	RGMII4TXD0	Op	CMOS	**RGMII 4 transmit data bus to PHY bit 0**
	RMII4TXD0	Op	CMOS	**RMII/NCSI 4 transmit data bus to PHY bit 0**
	NDSR3	I	CMOS	**UART 3 data set ready modem status**
	GPIOD6	I/O8	CMOS	**GPIO group D bit 6 (default)**
E25	RGMII4TXD1	Op	CMOS	**RGMII 4 transmit data bus to PHY bit 1**
	RMII4TXD1	Op	CMOS	**RMII/NCSI 4 transmit data bus to PHY bit 1**
	NRI3	I	CMOS	**UART 3 ring indicator modem status**
	GPIOD7	I/O8	CMOS	**GPIO group D bit 7 (default)**
D26	RGMII4TXD2	Op	CMOS	**RGMII 4 transmit data bus to PHY bit 2**
	NDTR3	O8	CMOS	**UART 3 data terminate ready modem status**
	GPIOE0	I/O8	CMOS	**GPIO group E bit 0 (default)**
D24	RGMII4TXD3	Op	CMOS	**RGMII 4 transmit data bus to PHY bit 3**
	NRTS3	O8	CMOS	**UART 3 request to send modem status**
	GPIOE1	I/O8	CMOS	**GPIO group E bit 1 (default)**
C25	RGMII4RXCK	I	CMOS	**RGMII 4 receive clock**
	RMII4RCLKI	I	CMOS	**RMII/NCSI 4 50MHz reference clock input**
	NCTS4	I	CMOS	**UART 4 clear to send modem status**
	GPIOE2	I/O8	CMOS	**GPIO group E bit 2 (default)**
C26	RGMII4RXCTL	I	CMOS	**RGMII 4 receive control**
	NDCD4	I	CMOS	**UART 4 data carrier detect modem status**
	GPIOE3	I/O8	CMOS	**GPIO group E bit 3 (default)**
C24	RGMII4RXD0	I	CMOS	**RGMII 4 receive data bus from PHY bit 0**
	RMII4RXD0	I	CMOS	**RMII/NCSI 4 receive data bus from PHY bit 0**
	NDSR4	I	CMOS	**UART 4 data set ready modem status**
	GPIOE4	I/O8	CMOS	**GPIO group E bit 4 (default)**
B26	RGMII4RXD1	I	CMOS	**RGMII 4 receive data bus from PHY bit 1**
	RMII4RXD1	I	CMOS	**RMII/NCSI 4 receive data bus from PHY bit 1**
	NRI4	I	CMOS	**UART 4 ring indicator modem status**
	GPIOE5	I/O8	CMOS	**GPIO group E bit 5 (default)**

to next page

from previous page

B25	RGMII4RXD2	I	CMOS	**RGMII 4 receive data bus from PHY bit 2**
	RMII4CRSDV	I	CMOS	**RMII/NCSI 4 receive carrier sense and data valid**
	NDTR4	O8	CMOS	**UART 4 data terminate ready modem status**
	GPIOE6	I/O8	CMOS	**GPIO group E bit 6 (default)**
B24	RGMII4RXD3	I	CMOS	**RGMII 4 receive data bus from PHY bit 3**
	RMII4RXER	I	CMOS	**RMII/NCSI 4 receive data error**
	NRTS4	O8	CMOS	**UART 4 request to send modem status**
	GPIOE7	I/O8	CMOS	**GPIO group E bit 7 (default)**
RGMIICK Signal Power Domain : PV33D_RGM				
A10	RGMIICK	I	CMOS	**RGMII external 125MHz reference clock input** RGMII 125MHz can be generated internally. When use internal source, add external 100K pull down for this pin.
Note :				
MAC1 MII interface – IO Power Domain : PV33D				
R23	GPIOS0	ID/O8	CMOS	**GPIO group S bit 0 (default)**
	MDC1	O8	CMOS	**MAC1 management interface clock output**
T25	GPIOS1	ID/O8	CMOSD	**GPIO group S bit 1 (default)**
	MDIO1	ID/O8	CMOS	**MAC1 management interface data input/output** When works as MDIO, requires external 4.7K pull high to PV33D.
MAC2 MII interface — IO Power Domain : PV33D				
J23	GPIOB4	ID/O8	CMOS	**GPIO group B bit 4 (default)**
	MDC2	O8	CMOS	**MAC2 management interface clock output**
G26	GPIOB5	ID/O8	CMOS	**GPIO group B bit 5 (default)**
	MDIO2	ID/O8	CMOS	**MAC2 management interface data input/output**
MAC3 MII interface — IO Power Domain : PV33D				
Note : List MDC3 / MDIO3 pins on I2C/SMBUS signal group				
MAC4 MII interface — IO Power Domain : PV33D				
Note : List MDC4 / MDIO4 pins on I2C/SMBUS signal group				

UART Port – 50 pins

Ball	Signal	I/O	Type	Description
UART port 1 — IO Power domain : PV33D				
D14	GPIOM0	ID/O8	CMOS	**GPIO group M bit 0 (default)**
	NCTS1	ID	CMOS	**UART 1 clear to send modem status**
B13	GPIOM1	ID/O8	CMOS	**GPIO group M bit 1 (default)**
	NDCD1	ID	CMOS	**UART 1 data carrier detect modem status**
A12	GPIOM2	ID/O8	CMOS	**GPIO group M bit 2 (default)**
	NDSR1	ID	CMOS	**UART 1 data set ready modem status**
E14	GPIOM3	ID/O8	CMOS	**GPIO group M bit 3 (default)**
	NRI1	ID	CMOS	**UART 1 ring indicator modem status**
B12	GPIOM4	ID/O8	CMOS	**GPIO group M bit 4 (default)**
	NDTR1	O8	CMOS	**UART 1 data terminate ready modem status**

to next page

from previous page

C12	GPIOM5	ID/O8	CMOS	**GPIO group M bit 5 (default)**
	NRTS1	O8	CMOS	**UART 1 request to send modem status**
C13	GPIOM6	ID/O8	CMOS	**GPIO group M bit 6 (default)**
	TXD1	O8	CMOS	**UART 1 transmit serial data output**
D13	GPIOM7	ID/O8	CMOS	**GPIO group M bit 7 (default)**
	RXD1	ID	CMOS	**UART 1 receive serial data input**
UART port 2 — IO Power domain : PV33D				
P25	GPION0	ID/O8	CMOS	**GPIO group N bit 0 (default)**
	NCTS2	ID	CMOS	**UART 2 clear to send modem status**
N23	GPION1	ID/O8	CMOS	**GPIO group N bit 1 (default)**
	NDCD2	ID	CMOS	**UART 2 data carrier detect modem status**
N25	GPION2	ID/O8	CMOS	**GPIO group N bit 2 (default)**
	NDSR2	ID	CMOS	**UART 2 data set ready modem status**
N24	GPION3	ID/O8	CMOS	**GPIO group N bit 3 (default)**
	NRI2	ID	CMOS	**UART 2 ring indicator modem status**
P26	GPION4	ID/O8	CMOS	**GPIO group N bit 4 (default)**
	NDTR2	O8	CMOS	**UART 2 data terminate ready modem status**
M23	GPION5	ID/O8	CMOS	**GPIO group N bit 5 (default)**
	NRTS2	O8	CMOS	**UART 2 request to send modem status**
N26	GPION6	ID/O8	CMOS	**GPIO group N bit 6 (default)**
	TXD2	O8	CMOS	**UART 2 transmit serial data output**
M26	GPION7	ID/O8	CMOS	**GPIO group N bit 7 (default)**
	RXD2	ID	CMOS	**UART 2 receive serial data input**
UART port 3 — IO Power domain:PV33D				
C15	GPIOL4	ID/O8	CMOS	**GPIO group L bit 4 (default)**
	TXD3	O8	CMOS	**UART 3 transmit serial data output**
F15	GPIOL5	ID/O8	CMOS	**GPIO group L bit 5 (default)**
	RXD3	ID	CMOS	**UART 3 receive serial data input**
Note : UART3 support full interface UART, other flow control pins are listed on MAC4 signal group.				
UART port 4 — IO Power domain:PV33D				
H25	GPIOB6	ID/O8	CMOS	**GPIO group B bit 6 (default)**
	TXD4	O8	CMOS	**UART 4 transmit serial data output**
J24	GPIOB7	ID/O8	CMOS	**GPIO group B bit 7 (default)**
	RXD4	ID	CMOS	**UART 4 receive serial data input**
Note : UART4 support full interface UART, other flow control pins are listed on MAC4 signal group.				
UART port 5 — IO Power domain:PV33D_RGM				
C8	TXD5	O8	CMOS	**UART 5 transmit serial data output** **Note: TXD5 must add external 10k ohm pull-up for A0 chip.**
D8	RXD5	I	CMOS	**UART 5 receive serial data input**
UART port 6 — IO Power domain:PV33D				
Note : UART6 don't support full flow control interface. List the TXD and RXD pins on SD2 signal group.				

to next page

from previous page

UART port 7 — IO Power domain:PV33D				
Note : UART7 don't support full flow control interface. List the TXD and RXD pins on SD2 signal group.				
UART port 8 — IO Power domain:PV33D				
Note : UART8 don't support full flow control interface. List the TXD and RXD pins on SD2 signal group.				
UART port 9 — IO Power domain:PV33D				
Note : UART9 don't support full flow control interface. List the TXD and RXD pins on SD2 signal group.				
UART port 10 — IO Power domain:PV33D				
R26	GPIOS4 TXD10	ID/O8 O8	CMOS CMOS	**GPIO group S bit 4 (default)** **UART 10 transmit serial data output**
P24	GPIOS5 RXD10	ID/O8 ID	CMOS CMOS	**GPIO group S bit 5 (default)** **UART 10 receive serial data input**
Note : UART10 don't support full flow control interface.				
UART port 11 — IO Power domain:PV33D				
P23	GPIOS6 TXD11	ID/O8 O8	CMOS CMOS	**GPIO group S bit 6 (default)** **UART 11 transmit serial data output**
T24	GPIOS7 RXD11	ID/O8 ID	CMOS CMOS	**GPIO group S bit 7 (default)** **UART 11 receive serial data input**
Note : UART11 don't support full flow control interface.				
UART port 12 — IO Power domain:PV33D				
Note : UART12 don't support full flow control interface. List the TXD and RXD pins on SPI2 signal group.				
UART port 13 — IO Power domain:PV33D				
Note : UART13 don't support full flow control interface. List the TXD and RXD pins on SPI1 signal group.				

Firmware SPI Memory Interface – 10 pins

Ball	Signal	I/O	Type	Description
Firmware SPI interface – IO Power Domain : PV33D				
AB14	FWSPICS0#	O16	CMOS	**Firmware booting SPI memory chip select 0** Default booting chip select.
AF13	FWSPICK	O16	CMOS	**Firmware booting SPI memory clock output**
AC14	FWSPIMOSI	ID/O16	CMOS	**Firmware booting SPI memory MOSI** Firmware SPI flash data MOSI output (single bit IO) Firmware SPI flash data D0 input/output (Dual/Quad bit IO)
AB13	FWSPIMISO	ID/O16	CMSO	**Firmware booting SPI memory MISO** Firmware SPI flash data MISO input (single bit IO) Firmware SPI flash data D1 input/output (Dual/Quad bit IO)
AE12	GPIOY4 FWSPIDQ2	ID/O16 ID/O16	CMOS CMOS	**GPIO group Y bit 4 (default)** **Firmware SPI flash data D2 input/output (Quad bit IO)**
AF12	GPIOY5 FWSPIDQ3	ID/O16 ID/O16	CMOS CMOS	**GPIO group Y bit 5 (default)** **Firmware SPI flash data D3 input/output (Quad bit IO)**

to next page

from previous page

AC12	GPIOY6 FWSPIABR	ID/O8 ID	CMOS CMOS	**GPIO group Y bit 6 (default)** **Firmware SPI Alternate Boot Selection** This signal is used for selecting alternate Boot area of the BOOT SPI Flash.
AB12	GPIOY7 FWSPIWP#	IS/O8 IS	CMOS CMOS	**GPIO group Y bit 7 (default)** **Firmware SPI Write Protection** This signal is used for preventing writes to the BOOT SPI Flash. When 0, areas of SPI are protected from ERASE/WRITE. When 1 the entire chip is allowed to update.
AA13	FWSPICS1#	O16	CMOS	**Firmware booting SPI memory chip select 1**
AC13	FWSPICS2#	O16	CMOS	**Firmware booting SPI memory chip select 2**

Extend BMC SPI Memory Interface – 17 pins

Ball	Signal	I/O	Type	Description
SPI1 Interface – IO Power Domain : PV33D				
AD13	SPI1CS0#	O16	CMOS	**SPI1 master interface CS output channel 0**
AC10	GPIOZ0 SPI1CS1#	ID/O16 O16	CMOS CMOS	**GPIO group Z bit 0 (default)** **SPI1 master interface CS output channel 1**
AD10	GPIOZ1 SPI1ABR	ID/O8 ID	CMOS CMOS	**GPIO group Z bit 1 (default)** **SPI1 Alternate Boot Selection** This signal is used for selecting alternate Boot area of the SPI Flash.
AE10	GPIOZ2 SPI1WP#	IS/O8 IS/O8	CMOS CMOS	**GPIO group Z bit 2 (default)** **SPI1 Write Protection** This signal is used for preventing writes to the SPI1 Flash. When 0, areas of SPI are protected from ERASE/WRITE. When 1 the entire chip is allowed to update.
AB11	GPIOZ3 SPI1CK	ID/O8 O8	CMOS CMOS	**GPIO group Z bit 3 (default)** **SPI1 master interface CK output**
AC11	GPIOZ4 SPI1MOSI	ID/O16 ID/O16	CMOS CMOS	**GPIO group Z bit 4 (default)** **SPI1 master interface MOSI output/input** SPI1 flash data MOSI output (single bit IO) SPI1 flash data D0 input/output (Dual/Quad bit IO)
AA11	GPIOZ5 SPI1MISO	ID/O16 ID/O16	CMOS CMOS	**GPIO group Z bit 5 (default)** **SPI1 master interface MISO input/output** SPI1 flash data MISO input (single bit IO) SPI1 flash data D1 input/output (Dual/Quad bit IO)
AD11	GPIOZ6 SPI1DQ2 TXD13	ID/O16 ID/O16 O16	CMOS CMOS CMOS	**GPIO group Z bit 6 (default)** **SPI1 flash data D2 input/output (Quad bit IO)** **UART 13 transmit serial data output (Dual Ball-out)**
AF10	GPIOZ7 SPI1DQ3 RXD13	ID/O16 ID/O16 ID	CMOS CMOS CMOS	**GPIO group Z bit 7 (default)** **SPI1 flash data D3 input/output (Quad bit IO)** **UART 13 receive serial data input (Dual Ball-out)**
SPI2 Interface – IO Power Domain : PV33D				
AE8	GPIOX0 SPI2CS0#	ID/O16 O16	CMOS CMOS	**GPIO group X bit 0 (default)** **SPI2 master interface CS output channel 0**

to next page

from previous page

AA9	GPIOX1	ID/O16	CMOS	**GPIO group X bit 1 (default)**
	SPI2CS1#	O16	CMOS	**SPI2 master interface CS output channel 1**
AC9	GPIOX2	ID/O16	CMOS	**GPIO group X bit 2 (default)**
	SPI2CS2#	O16	CMOS	**SPI2 master interface CS output channel 2**
AF8	GPIOX3	ID/O16	CMOS	**GPIO group X bit 3 (default)**
	SPI2CK	O16	CMOS	**SPI2 master interface CK output**
AB9	GPIOX4	ID/O16	CMOS	**GPIO group X bit 4 (default)**
	SPI2MOSI	ID/O16	CMOS	**SPI2 master interface MOSI output/input** SPI2 flash data MOSI output (single bit IO) SPI2 flash data D0 input/output (Dual/Quad bit IO)
AD9	GPIOX5	ID/O16	CMOS	**GPIO group X bit 5 (default)**
	SPI2MISO	ID/O16	CMOS	**SPI2 master interface MISO input/output** SPI2 flash data MISO input (single bit IO) SPI2 flash data D1 input/output (Dual/Quad bit IO)
AF9	GPIOX6	ID/O16	CMOS	**GPIO group X bit 6 (default)**
	SPI2DQ2	ID/O16	CMOS	**SPI2 flash data D2 input/output (Quad bit IO)**
	TXD12	O16	CMOS	**UART 12 transmit serial data output (Dual Ball-out)**
AB10	GPIOX7	ID/O16	CMOS	**GPIO group X bit 7 (default)**
	SPI2DQ3	ID/O16	CMOS	**SPI2 flash data D3 input/output (Quad bit IO)**
	RXD12	ID	CMOS	**UART 12 receive serial data input (Dual Ball-out)**

I2C/SMBUS Interface – 32 pins

Ball	Signal	I/O	Type	Description
		I2C Port 1 — Port 4		
Note : List the I2C CLK and DATA Pins on I3C signal group.				
		I2C Port 5 — Port 14 (IO Power domain : PV33D)		
A11	GPIOK0	IS/O8	CMOS	**GPIO group K bit 0 (default)**
	SCL5	IS/O8	CMOS	**I2C/SMBUS 5 clock pin**
C11	GPIOK1	IS/O8	CMOS	**GPIO group K bit 1 (default)**
	SDA5	IS/O8	CMOS	**I2C/SMBUS 5 data pin**
D12	GPIOK2	IS/O8	CMOS	**GPIO group K bit 2 (default)**
	SCL6	IS/O8	CMOS	**I2C/SMBUS 6 clock pin**
E13	GPIOK3	IS/O8	CMOS	**GPIO group K bit 3 (default)**
	SDA6	IS/O8	CMOS	**I2C/SMBUS 6 data pin**
D11	GPIOK4	IS/O8	CMOS	**GPIO group K bit 4 (default)**
	SCL7	IS/O8	CMOS	**I2C/SMBUS 7 clock pin**
E11	GPIOK5	IS/O8	CMOS	**GPIO group K bit 5 (default)**
	SDA7	IS/O8	CMOS	**I2C/SMBUS 7 data pin**
F13	GPIOK6	IS/O8	CMOS	**GPIO group K bit 6 (default)**
	SCL8	IS/O8	CMOS	**I2C/SMBUS 8 clock pin**
E12	GPIOK7	IS/O8	CMOS	**GPIO group K bit 7 (default)**
	SDA8	IS/O8	CMOS	**I2C/SMBUS 8 data pin**
D15	GPIOL0	IS/O8	CMOS	**GPIO group L bit 0 (default)**
	SCL9	IS/O8	CMOS	**I2C/SMBUS 9 clock pin**

to next page

from previous page

A14	GPIOL1	IS/O8	CMOS	**GPIO group L bit 1 (default)**
	SDA9	IS/O8	CMOS	**I2C/SMBUS 9 data pin**
E15	GPIOL2	IS/O8	CMOS	**GPIO group L bit 2 (default)**
	SCL10	IS/O8	CMOS	**I2C/SMBUS 10 clock pin**
A13	GPIOL3	IS/O8	CMOS	**GPIO group L bit 3 (default)**
	SDA10	IS/O8	CMOS	**I2C/SMBUS 10 data pin**
M24	GPIOA0	IS/O16	CMOS	**GPIO group A bit 0 (default)**
	SCL11	IS/O16	CMOS	**I2C/SMBUS 11 clock pin**
	MDC3	O16	CMOS	**MAC3 management interface clock output**
M25	GPIOA1	IS/O16	CMOS	**GPIO group A bit 1 (default)**
	SDA11	IS/O16	CMOS	**I2C/SMBUS 11 data pin**
	MDIO3	IS/O16	CMOS	**MAC3 management interface data input/output** When works as MDIO, requires external 4.7K pull high to PV33D.
L26	GPIOA2	IS/O16	CMOS	**GPIO group A bit 2 (default)**
	SCL12	IS/O16	CMOS	**I2C/SMBUS 12 clock pin**
	MDC4	O16	CMOS	**MAC4 management interface clock output**
K24	GPIOA3	IS/O16	CMOS	**GPIO group A bit 3 (default)**
	SDA12	IS/O16	CMOS	**I2C/SMBUS 12 data pin**
	MDIO4	IS/O16	CMOS	**MAC4 management interface data input/output** When works as MDIO, requires external 4.7K pull high to PV33D.
K26	GPIOA4	IS/O8	CMOS	**GPIO group A bit 4 (default)**
	MAC1LINK	IS	CMOS	**MAC1 PHY link status input**
	SCL13	IS/O8	CMOS	**I2C/SMBUS 13 clock pin**
	SGPM2CK	IS/O8	CMOS	**Master serial GPIO bus 2 clock output**
	SGPS2CK	IS	CMOS	**Slave serial GPIO bus 2 clock input** MAC1LINK is programmable high/low polarity.
L24	GPIOA5	IS/O8	CMOS	**GPIO group A bit 5 (default)**
	MAC2LINK	IS	CMOS	**MAC2 PHY link status input**
	SDA13	IS/O8	CMOS	**I2C/SMBUS 13 data pin**
	SGPM2LD	IS/O8	CMOS	**Master serial GPIO bus 2 parallel data load output**
	SGPS2LD	IS	CMOS	**Slave serial GPIO bus 2 parallel data load input** MAC2LINK is programmable high/low polarity.
L23	GPIOA6	IS/O8	CMOS	**GPIO group A bit 6 (default)**
	MAC3LINK	IS	CMOS	**MAC3 PHY link status input**
	SCL14	IS/O8	CMOS	**I2C/SMBUS 14 clock pin**
	SGPM2O	IS/O8	CMOS	**Master serial GPIO bus 2 serial data output**
	SGPS2I0	IS	CMOS	**Slave serial GPIO bus 2 serial data input channel 0** MAC3LINK is programmable high/low polarity.
K25	GPIOA7	IS/O8	CMOS	**GPIO group A bit 7 (default)**
	MAC4LINK	IS	CMOS	**MAC4 PHY link status input**
	SDA14	IS/O8	CMOS	**I2C/SMBUS 14 data pin**
	SGPM2I	IS	CMOS	**Master serial GPIO bus 2 serial data input**
	SGPS2I1	IS	CMOS	**Slave serial GPIO bus 2 serial data input channel 1** MAC4LINK is programmable high/low polarity.

to next page

from previous page

I2C Port 15 — IO Power domain : PV33D
Note : List the I2C CLK and DATA Pins on Serial GPIO signal group.

I2C Port 16 — IO Power domain : PV33D
Note : List the I3C CLK and DATA Pins on Serial GPIO signal group.

I3C Interface – 16 pins

Ball	Signal	I/O	Type	Description
I3C Low-voltage mode Bus 1 - 2, IO Power Domain : I3CVDDL1				
AF23	I3C1SCL	IU/O8	CMOS12	**Low-voltage (1.2V/1.0V) I3C Bus 1 clock pin**
AE24	I3C1SDA	IU/O8	CMOS12	**Low-voltage (1.2V/1.0V) I3C Bus 1 data pin**
AF22	I3C2SCL	IU/O8	CMOS12	**Low-voltage (1.2V/1.0V) I3C Bus 2 clock pin**
AE22	I3C2SDA	IU/O8	CMOS12	**Low-voltage (1.2V/1.0V) I3C Bus 2 data pin**
I3C Low-voltage mode Bus 3 - 4, IO Power Domain : I3CVDDL2				
AF25	I3C3SCL	IU/O8	CMOS12	**Low-voltage (1.2V/1.0V) I3C Bus 3 clock pin** **Share the same inetnal I3C controller with HVI3C3SCL pin**
	FSI1CLK	ID/O8	CMOS12	**1.2V FSI Bus 1 clock pin**
AE26	I3C3SDA	IU/O8	CMOS12	**Low-voltage (1.2V/1.0V) I3C Bus 3 data pin** **Share the same inetnal I3C controller with HVI3C3SDA pin**
	FSI1DATA	ID/O8	CMOS12	**1.2V FSI Bus 1 data pin**
AE25	I3C4SCL	IU/O8	CMOS12	**Low-voltage (1.2V/1.0V) I3C Bus 4 clock pin** **Share the same inetnal I3C controller with HVI3C4SCL pin**
	FSI2CLK	ID/O8	CMOS12	**1.2V FSI Bus 2 clock pin**
AF24	I3C4SDA	IU/O8	CMOS12	**Low-voltage (1.2V/1.0V) I3C Bus 4 data pin** **Share the same inetnal I3C controller with HVI3C4SDA pin**
	FSI2DATA	ID/O8	CMOS12	**1.2V FSI Bus 2 data pin**
I3C High-voltage mode Bus 3 — 6, IO Power Domain : I3CVDDH				
B20	GPIOJ0	IU/O12		**GPIO group J bit 0 (default)**
	HVI3C3SCL	IU/O12		**High-voltage (1.8V/3.3V) I3C Bus 3 clock pin** **Share the same inetnal I3C controller with I3C3SCL pin**
	SCL1	IU/O12		**I2C/SMBUS 1 clock pin**
A20	GPIOJ1	IU/O12		**GPIO group J bit 1 (default)**
	HVI3C3SDA	IU/O12		**High-voltage (1.8V/3.3V) I3C Bus 3 data pin** **Share the same inetnal I3C controller with I3C3SDA pin**
	SDA1	IU/O12		**I2C/SMBUS 1 data pin**
E19	GPIOJ2	IU/O12		**GPIO group J bit 2 (default)**
	HVI3C4SCL	IU/O12		**High-voltage (1.8V/3.3V) I3C Bus 4 clock pin** **Share the same inetnal I3C controller with I3C4SCL pin**
	SCL2	IU/O12		**I2C/SMBUS 2 clock pin**
D20	GPIOJ3	IU/O12		**GPIO group J bit 3 (default)**
	HVI3C4SDA	IU/O12		**High-voltage (1.8V/3.3V) I3C Bus 4 data pin** **Share the same inetnal I3C controller with I3C4SDA pin**
	SDA2	IU/O12		**I2C/SMBUS 2 data pin**
C19	GPIOJ4	IU/O12		**GPIO group J bit 4 (default)**
	HVI3C5SCL	IU/O12		**High-voltage (1.8V/3.3V) I3C Bus 5 clock pin**

to next page

from previous page

	SCL3	IU/O12		**I2C/SMBUS 3 clock pin**
A19	GPIOJ5	IU/O12		**GPIO group J bit 5 (default)**
	HVI3C5SDA	IU/O12		**High-voltage (1.8V/3.3V) I3C Bus 5 data pin**
	SDA3	IU/O12		**I2C/SMBUS 3 data pin**
C20	GPIOJ6	IU/O12		**GPIO group J bit 6 (default)**
	HVI3C6SCL	IU/O12		**High-voltage (1.8V/3.3V) I3C Bus 6 clock pin**
	SCL4	IU/O12		**I2C/SMBUS 4 clock pin**
D19	GPIOJ7	IU/O12		**GPIO group J bit 7 (default)**
	HVI3C6SDA	IU/O12		**High-voltage (1.8V/3.3V) I3C Bus 6 data pin**
	SDA4	IU/O12		**I2C/SMBUS 4 data pin**

SD/SDIO/eMMC Interface – 28 pins

Ball	Signal	I/O	Type	Description
SD1 interface — IO Power Domain : SD1VDD				
D22	GPIOF0	I/Op	CMOS	**GPIO group F bit 0 (default)**
	SD1CLK	Op		**SD 1 clock output**
	PWM8	Op	CMOS	**PWM output bus 8 (Dual Ball-out)**
E22	GPIOF1	I/Op	CMOS	**GPIO group F bit 1 (default)**
	SD1CMD	I/Op		**SD 1 command input/output**
	PWM9	Op	CMOS	**PWM output bus 9 (Dual Ball-out)**
D23	GPIOF2	I/Op	CMOS	**GPIO group F bit 2 (default)**
	SD1DAT0	I/Op		**SD 1 data bus bit 0**
	PWM10	Op	CMOS	**PWM output bus 10 (Dual Ball-out)**
C23	GPIOF3	I/Op	CMOS	**GPIO group F bit 3 (default)**
	SD1DAT1	I/Op		**SD 1 data bus bit 1**
	PWM11	Op	CMOS	**PWM output bus 11 (Dual Ball-out)**
C22	GPIOF4	I/Op	CMOS	**GPIO group F bit 4 (default)**
	SD1DAT2	I/Op		**SD 1 data bus bit 2**
	PWM12	Op	CMOS	**PWM output bus 12 (Dual Ball-out)**
A25	GPIOF5	I/Op	CMOS	**GPIO group F bit 5 (default)**
	SD1DAT3	I/Op		**SD 1 data bus bit 3**
	PWM13	Op	CMOS	**PWM output bus 13 (Dual Ball-out)**
A24	GPIOF6	I/Op	CMOS	**GPIO group F bit 6 (default)**
	SD1CD#	I		**SD 1 data card detection input**
	PWM14	Op	CMOS	**PWM output bus 14 (Dual Ball-out)**
A23	GPIOF7	I/Op	CMOS	**GPIO group F bit 7 (default)**
	SD1WP#	I		**SD 1 write protect input**
	PWM15	Op	CMOS	**PWM output bus 15 (Dual Ball-out)**
SD2 interface — IO Power Domain : SD2VDD				
E21	GPIOG0	I/Op	CMOS	**GPIO group G bit 0 (default)**
	TXD6	Op	CMOS	**UART 6 transmit serial data output**
	SD2CLK	Op		**SD 2 clock output**
	SALT9	I	CMOS	**SMBus 9 Alert pin (Dual Ball-out)**

to next page

from previous page

B22	GPIOG1	I/Op	CMOS	**GPIO group G bit 1 (default)**
	RXD6	I	CMOS	**UART 6 receive serial data input**
	SD2CMD	I/Op		**SD 2 command input/output**
	SALT10	I	CMOS	**SMBus 10 Alert pin (Dual Ball-out)**
C21	GPIOG2	I/Op	CMOS	**GPIO group G bit 2 (default)**
	TXD7	Op	CMOS	**UART 7 transmit serial data output**
	SD2DAT0	I/Op		**SD 2 data bus bit 0**
	SALT11	I	CMOS	**SMBus 11 Alert pin (Dual Ball-out)**
A22	GPIOG3	I/Op	CMOS	**GPIO group G bit 3 (default)**
	RXD7	I	CMOS	**UART 7 receive serial data input**
	SD2DAT1	I/Op		**SD 2 data bus bit 1**
	SALT12	I	CMOS	**SMBus 12 Alert pin (Dual Ball-out)**
A21	GPIOG4	I/Op	CMOS	**GPIO group G bit 4 (default)**
	TXD8	Op	CMOS	**UART 8 transmit serial data output**
	SD2DAT2	I/Op		**SD 2 data bus bit 2**
	SALT13	I	CMOS	**SMBus 13 Alert pin (Dual Ball-out)**
E20	GPIOG5	I/Op	CMOS	**GPIO group G bit 5 (default)**
	RXD8	I	CMOS	**UART 8 receive serial data input**
	SD2DAT3	I/Op		**SD 2 data bus bit 3**
	SALT14	I	CMOS	**SMBus 14 Alert pin (Dual Ball-out)**
D21	GPIOG6	I/Op	CMOS	**GPIO group G bit 6 (default)**
	TXD9	Op	CMOS	**UART 9 transmit serial data output**
	SD2CD#	I		**SD 2 data card detection input**
	SALT15	I	CMOS	**SMBus 15 Alert pin (Dual Ball-out)**
B21	GPIOG7	I/Op	CMOS	**GPIO group G bit 7 (default)**
	RXD9	I	CMOS	**UART 9 receive serial data input**
	SD2WP#	I		**SD 2 write protect input**
	SALT16	I	CMOS	**SMBus 16 Alert pin (Dual Ball-out)**
eMMC interface — IO Power Domain : PV18D				
AB4	GPIO18D0	ID/O16	CMOS18	**1.8V GPIO group D bit 0 (1.8V only)(default)**
	EMMCCLK	O16	CMOS18	**EMMC clock output (1.8V only)**
AA4	GPIO18D1	ID/O16	CMOS18	**1.8V GPIO group D bit 1 (1.8V only)(default)**
	EMMCCMD	ID/O16	CMOS18	**EMMC command input/output (1.8V only)**
AC4	GPIO18D2	ID/O16	CMOS18	**1.8V GPIO group D bit 2 (1.8V only)(default)**
	EMMCDAT0	ID/O16	CMOS18	**EMMC data bus bit 0 (1.8V only)**
AA5	GPIO18D3	ID/O16	CMOS18	**1.8V GPIO group D bit 3 (1.8V only)(default)**
	EMMCDAT1	ID/O16	CMOS18	**EMMC data bus bit 1 (1.8V only)**
Y5	GPIO18D4	ID/O16	CMOS18	**1.8V GPIO group D bit 4 (1.8V only)(default)**
	EMMCDAT2	ID/O16	CMOS18	**EMMC data bus bit 2 (1.8V only)**
AB5	GPIO18D5	ID/O16	CMOS18	**1.8V GPIO group D bit 5 (1.8V only)(default)**
	EMMCDAT3	ID/O16	CMOS18	**EMMC data bus bit 3 (1.8V only)**
AC5	GPIO18D6	ID/O16	CMOS18	**1.8V GPIO group D bit 7 (1.8V only)(default)**
	EMMCCD#	ID	CMOS18	**EMMC data card detection input (1.8V only)**

to next page

from previous page

AB6	GPIO18D7	ID/O16	CMOS18	**1.8V GPIO group D bit 6 (1.8V only)(default)**
	EMMCWP#	ID	CMOS18	**EMMC write protect input (1.8V only)**
Y1	GPIO18E0	ID/O16	CMOS18	**1.8V GPIO group E bit 0 (1.8V only)(default)**
	EMMCDAT4	ID/O16	CMOS18	**EMMC data bus bit 4 (1.8V only)**
	FWSPI18CS#	O16	CMOS18	**Firmware boot 1.8V SPI flash CS output**
	VBCS#	O16	CMOS18	**VGA BIOS SPI flash CS output**
Y2	GPIO18E1	ID/O16	CMOS18	**1.8V GPIO group E bit 1 (1.8V only)(default)**
	EMMCDAT5	ID/O16	CMOS18	**EMMC data bus bit 5 (1.8V only)**
	FWSPI18CK	O16	CMOS18	**Firmware boot 1.8V SPI flash CK output**
	VBCK	O16	CMOS18	**VGA BIOS SPI flash CK output**
Y3	GPIO18E2	ID/O16	CMOS18	**1.8V GPIO group E bit 2 (1.8V only)(default)**
	EMMCDAT6	ID/O16	CMOS18	**EMMC data bus bit 6 (1.8V only)**
	FWSPI18MOSI	O16	CMOS18	**Firmware boot 1.8V SPI flash MOSI output**
	VBMOSI	O16	CMOS18	**VGA BIOS SPI flash MOSI output**
Y4	GPIO18E3	ID/O16	CMOS18	**1.8V GPIO group E bit 3 (1.8V only)(default)**
	EMMCDAT7	ID/O16	CMOS18	**EMMC data bus bit 7 (1.8V only)**
	FWSPI18MISO	ID	CMOS18	**Firmware boot 1.8V SPI flash MISO input**
	VBMISO	ID	CMOS18	**VGA BIOS SPI flash MISO input**

Serial GPIO Interface – 8 pins

Ball	Signal	I/O	Type	Description
Master SGPIO and Slave SGPIO Bus 1 — IO Power Domain : PV33D				
A18	GPIOH0	ID/O8	CMOS	**GPIO group H bit 0 (default)**
	SGPM1CK	O8	CMOS	**Master Serial GPIO bus 1 clock output**
B18	GPIOH1	ID/O8	CMOS	**GPIO group H bit 1 (default)**
	SGPM1LD	O8	CMOS	**Master Serial GPIO bus 1 parallel data load output**
C18	GPIOH2	ID/O8	CMOS	**GPIO group H bit 2 (default)**
	SGPM1O	O8	CMOS	**Master Serial GPIO bus 1 serial data output**
A17	GPIOH3	ID/O8	CMOS	**GPIO group H bit 3 (default)**
	SGPM1I	ID	CMOS	**Master Serial GPIO bus 1 serial data input**
D18	GPIOH4	ID/O8	CMOS	**GPIO group H bit 4 (default)**
	SGPS1CK	I	CMOS	**Slave serial GPIO bus 1 clock input**
	SCL15	ID/O8	CMOS	**I2C/SMBUS 15 clock pin**
B17	GPIOH5	ID/O8	CMOS	**GPIO group H bit 5 (default)**
	SGPS1LD	I	CMOS	**Slave serial GPIO bus 1 parallel data load input**
	SDA15	ID/O8	CMOS	**I2C/SMBUS 15 data pin**
C17	GPIOH6	ID/O8	CMOS	**GPIO group H bit 6 (default)**
	SGPS1I0	I	CMOS	**Slave serial GPIO bus 1 serial data input channel 0**
	SCL16	ID/O8	CMOS	**I2C/SMBUS 16 clock pin**
E18	GPIOH7	ID/O8	CMOS	**GPIO group H bit 7 (default)**
	SGPS1I1	I	CMOS	**Slave serial GPIO bus 1 serial data input channel 1**
	SDA16	ID/O8	CMOS	**I2C/SMBUS 16 data pin**
Slave SGPIO Bus 2 — IO Power Domain : PV33D				

to next page

from previous page

Note :
List the Slave GPIO Bus 2 on I2C/SMBus signal group.

VGA Interface – 4 pins

Ball	Signal	I/O	Type	Description
HSYNC and VSYNC — IO Power Domain : PV33D				
B14	GPIOL6	ID/O8	CMOS	**GPIO group L bit 6**
	VGAHS	O8	CMOS	**VGA horizontal sync output (default)**
C14	GPIOL7	ID/O8	CMOS	**GPIO group L bit 7**
	VGAVS	O8	CMOS	**VGA vertical sync output (default)**
DDCCLK and DDCDAT — IO Power Domain : PV33D_RGM				
B8	DDCCLK	O8	CMOS	**VGA DDC clock pin**
A8	DDCDAT	O8	CMOS	**VGA DDC data pin**

PWM/Fan Tachometer – 32 pins

Ball	Signal	I/O	Type	Description
PWM and TACH — IO Power Domain : PV33D				
AD26	GPIOO0	ID/O8	CMOS	**GPIO group O bit 0 (default)**
	PWM0	O8	CMOS	**PWM output bus 0**
AD22	GPIOO1	ID/O8	CMOS	**GPIO group O bit 1 (default)**
	PWM1	O8	CMOS	**PWM output bus 1**
AD23	GPIOO2	ID/O8	CMOS	**GPIO group O bit 2 (default)**
	PWM2	O8	CMOS	**PWM output bus 2**
AD24	GPIOO3	ID/O8	CMOS	**GPIO group O bit 3 (default)**
	PWM3	O8	CMOS	**PWM output bus 3**
AD25	GPIOO4	ID/O8	CMOS	**GPIO group O bit 4 (default)**
	PWM4	O8	CMOS	**PWM output bus 4**
AC22	GPIOO5	ID/O8	CMOS	**GPIO group O bit 5 (default)**
	PWM5	O8	CMOS	**PWM output bus 5**
AC24	GPIOO6	ID/O8	CMOS	**GPIO group O bit 6 (default)**
	PWM6	O8	CMOS	**PWM output bus 6**
AC23	GPIOO7	ID/O8	CMOS	**GPIO group O bit 7 (default)**
	PWM7	O8	CMOS	**PWM output bus 7**
AB22	GPIOP0	ID/O8	CMOS	**GPIP group P bit 0 (default)**
	PWM8	O8	CMOS	**PWM output bus 8 (Dual Ball-out)**
	THRUIN0	ID	CMOS	**PASS Through Input bus 0**
W24	GPIOP1	ID/O8	CMOS	**GPIP group P bit 1 (default)**
	PWM9	O8	CMOS	**PWM output bus 9 (Dual Ball-out)**
	THRUOUT0	O8	CMOS	**PASS Through Output bus 0**
AA23	GPIOP2	ID/O8	CMOS	**GPIP group P bit 2 (default)**
	PWM10	O8	CMOS	**PWM output bus 10 (Dual Ball-out)**
	THRUIN1	ID	CMOS	**PASS Through Input bus 1**
AA24	GPIOP3	ID/O8	CMOS	**GPIP group P bit 3 (default)**
	PWM11	O8	CMOS	**PWM output bus 11 (Dual Ball-out)**

to next page

from previous page

	THRUOUT1	O8	CMOS	**PASS Through Output bus 1**
W23	GPIOP4	ID/O8	CMOS	**GPIP group P bit 4 (default)**
	PWM12	O8	CMOS	**PWM output bus 12 (Dual Ball-out)**
	THRUIN2	ID	CMOS	**PASS Through Input bus 2**
AB23	GPIOP5	ID/O8	CMOS	**GPIP group P bit 5 (default)**
	PWM13	O8	CMOS	**PWM output bus 13 (Dual Ball-out)**
	THRUOUT2	O8	CMOS	**PASS Through Output bus 2**
AB24	GPIOP6	ID/O8	CMOS	**GPIP group P bit 6 (default)**
	PWM14	O8	CMOS	**PWM output bus 14 (Dual Ball-out)**
Y23	GPIOP7	ID/O8	CMOS	**GPIP group P bit 7**
	PWM15	O8	CMOS	**PWM output bus 15 (Dual Ball-out)**
	HBLED#	O8	CMOS	**Heart Beat LED indicator output.(default)** It is open-drain active low output. The HW blinking modes are shown below: 1 0Hz : LED dark indicates ARM_CA7 is reset or secure boot is running (LED off) 2 80Hz : Secure boot failed (LED 50% on) 3 20Hz : Boot passed with ABR (show about 3 seconds and then exit) 4 4Hz : ARM CA7 is reading address range 0x0000_0000 to 0x0FFF_FFFF (reading flash or SRAM) 5 1Hz : ARM CA7 is reading address range 0x1000_0000 to 0x7FFF_FFFF (reading peripherals) 6 0.5Hz : ARM CA7 is reading address range 0x8000_0000 to 0xFFFF_FFFF (reading DRAM) 7 0.2Hz : No ARM CA7 read access cycles 8 0.1Hz : No ARM CA7 read access cycles over 6-8 seconds Stay in mode 8 if there is no ARM_CA7 reading access Exit to mode 4-6 if there is any ARM_CA7 reading access HW blinking modes can be disabled by HW pin strap. SW driven mode of Heart Beat needs to disable Multi-funtion pin control 1-3 and use GPIOP7.
AA25	GPIOQ0	IS/O8	CMOS	**GPIO group Q bit 0 (default)**
	TACH0	IS	CMOS	**Fan Tachometer input**
AB25	GPIOQ1	IS/O8	CMOS	**GPIO group Q bit 1 (default)**
	TACH1	IS	CMOS	**Fan Tachometer input**
Y24	GPIOQ2	IS/O8	CMOS	**GPIO group Q bit 2 (default)**
	TACH2	IS	CMOS	**Fan Tachometer input**
AB26	GPIOQ3	IS/O8	CMOS	**GPIO group Q bit 3 (default)**
	TACH3	IS	CMOS	**Fan Tachometer input**
Y26	GPIOQ4	IS/O8	CMOS	**GPIO group Q bit 4 (default)**
	TACH4	IS	CMOS	**Fan Tachometer input**
AC26	GPIOQ5	IS/O8	CMOS	**GPIO group Q bit 5 (default)**
	TACH5	IS	CMOS	**Fan Tachometer input**

to next page

from previous page

Y25	GPIOQ6	IS/O8	CMOS	**GPIO group Q bit 6 (default)**
	TACH6	IS	CMOS	**Fan Tachometer input**
AA26	GPIOQ7	IS/O8	CMOS	**GPIO group Q bit 7 (default)**
	TACH7	IS	CMOS	**Fan Tachometer input**
V25	GPIOR0	ID/O8	CMOS	**GPIO group R bit 0 (default)**
	TACH8	ID	CMOS	**Fan Tachometer input**
U24	GPIOR1	ID/O8	CMOS	**GPIO group R bit 1 (default)**
	TACH9	ID	CMOS	**Fan Tachometer input**
V24	GPIOR2	ID/O8	CMOS	**GPIO group R bit 2 (default)**
	TACH10	ID	CMOS	**Fan Tachometer input**
V26	GPIOR3	ID/O8	CMOS	**GPIO group R bit 3 (default)**
	TACH11	ID	CMOS	**Fan Tachometer input**
U25	GPIOR4	ID/O8	CMOS	**GPIO group R bit 4 (default)**
	TACH12	ID	CMOS	**Fan Tachometer input**
T23	GPIOR5	ID/O8	CMOS	**GPIO group R bit 5 (default)**
	TACH13	ID	CMOS	**Fan Tachometer input**
W26	GPIOR6	ID/O8	CMOS	**GPIO group R bit 6 (default)**
	TACH14	ID	CMOS	**Fan Tachometer input**
U26	GPIOR7	ID/O8	CMOS	**GPIO group R bit 7 (default)**
	TACH15	ID	CMOS	**Fan Tachometer input**

ACPI Interface – 11 pins

Ball	Signal	I/O	Type	Description
		ACPI — IO Power Domain : PV33D		
E16	GPIOI5	ID/O8	CMOS	**GPIO group I bit 5 (default)**
	SIOPBO#	ID/O8	CMOS	**Super IO PWRBUTTON_O# control (Optional for SIO function)**
B16	GPIOI6	ID/O8	CMOS	**GPIO group I bit 6 (default)**
	SIOPBI#	ID	CMOS	**Super IO PWRBUTTON_I# control (Optional for SIO function)**
A15	GPIOI7	ID/O8	CMOS	**GPIO group I bit 7 (default)**
	SIOSCI#	O8	CMOS	**System SCI output (Optional for SIO function)**
AB15	GPIOV0	ID/O8	CMOS	**GPIO group V bit 0 (default)**
	SIOS3#	ID	CMOS	**Super IO S3# control (Optional for SIO function)**
AF14	GPIOV1	ID/O8	CMOS	**GPIO group V bit 1 (default)**
	SIOS5#	ID	CMOS	**Super IO S5# control (Optional for SIO function)**
AD14	GPIOV2	ID/O8	CMOS	**GPIO group V bit 2 (default)**
	SIOPWREQ#	O8	CMOS	**Super IO PWREQ# control (Optional for SIO function)**
AC15	GPIOV3	ID/O8	CMOS	**GPIO group V bit 3 (default)**
	SIOONCTRL#	ID/O8	CMOS	**Super IO ONCONTROL# control (Optional for SIO function)**
AE15	GPIOV4	ID/O8	CMOS	**GPIO group V bit 4 (default)**
	SIOPWRGD	ID	CMOS	**System power good input (Optional for SIO function)**

to next page

from previous page

AE14	GPIOV5	ID/O8	CMOS	**GPIO group V bit 5 (default)**
	LPCPD#	ID/O8	CMOS	**Dedicated LPC power down control (Optional for LPC function)**
AD15	GPIOV6	ID/O8	CMOS	**GPIO group V bit 6 (default)**
	LPCPME#	ID/O8	CMOS	**Dedicated LPC PME# control (Optional for LPC function)**
AF15	GPIOV7	ID/O8	CMOS	**GPIO group V bit 7 (default)**
	LPCSMI#	ID/O8	CMOS	**Dedicated LPC SMI control (Optional for LPC function)**

GPIO Interface – 10 pins

Ball	Signal	I/O	Type	Description
		IO Power Domain : PV33D		
J26	GPIOB0	ID/O12	CMOS	**GPIO group B bit 0 (default)**
	SALT1	ID	CMOS	**SMBus 1 Alert pin**
K23	GPIOB1	ID/O12	CMOS	**GPIO group B bit 1 (default)**
	SALT2	ID	CMOS	**SMBus 2 Alert pin**
H26	GPIOB2	ID/O12	CMOS	**GPIO group B bit 2 (default)**
	SALT3	ID	CMOS	**SMBus 3 Alert pin**
J25	GPIOB3	ID/O12	CMOS	**GPIO group B bit 3 (default)**
	SALT4	ID	CMOS	**SMBus 4 Alert pin**
T26	GPIOS2	ID/O12	CMOS	**GPIO group S bit 2 (default)**
	PEWAKE#	ID/O12		**PCI Express Wake up function pin** PEWAKE# signal is an open drain type signal.
R24	GPIOS3	ID/O12	CMOS	**GPIO group S bit 3 (default)**
	OSCCLK	O12		**Internal oscillator ring clock output**
AF11	GPIOY0	ID/O12	CMOS	**GPIO group Y bit 0 (default)**
	SALT5	ID	CMOS	**SMBus 5 Alert pin**
	WDTRST1#	O12	CMOS	**Watchdog timer 1 timeout pulse output**
AD12	GPIOY1	ID/O12	CMOS	**GPIO group Y bit 1 (default)**
	SALT6	ID	CMOS	**SMBus 6 Alert pin**
	WDTRST2#	O12	CMOS	**Watchdog timer 2 timeout pulse output**
AE11	GPIOY2	ID/O12	CMOS	**GPIO group Y bit 2 (default)**
	SALT7	ID	CMOS	**SMBus 7 Alert pin**
	WDTRST3#	O12	CMOS	**Watchdog timer 3 timeout pulse output**
AA12	GPIOY3	ID/O12	CMOS	**GPIO group Y bit 3 (default)**
	SALT8	ID	CMOS	**SMBus 8 Alert pin**
	WDTRST4#	O12	CMOS	**Watchdog timer 4 timeout pulse output**

PECI Port – 2 pins

Ball	Signal	I/O	Type	Description
AF16	PECI	I/O	A	**PECI signal input/output**
AA18	PECIVDD	-	P	**PECI power** Input power range 0.85V-1.21V, connect to CPU V_{TT} power. If not used, connect to IV10D.

JTAG Slave / Master Port 1 – 5 pins

Ball	Signal	I/O	Type	Description
		IO Power Domain : PV33D_RGM		
F11	NTRST	IU	CMOS	**ARM JTAG test reset input** **Note: Add 10K ohm external pull down.**
	MNTRST1	O8	CMOS	**JTAG master 1 reset output**
B9	TCK	IU	CMOS	**ARM JTAG test clock input**
	MTCK1	O8	CMOS	**JTAG master 1 clock output**
A9	TMS	IU	CMOS	**ARM JTAG test mode select input**
	MTMS1	O8	CMOS	**JTAG master 1 mode select output**
D9	TDI	IU	CMOS	**ARM JTAG test data input**
	MTDI1	O8	CMOS	**JTAG master 1 data output** **Note: It is output pin on AST2600**
C9	TDO	O8	CMOS	**ARM JTAG test data output**
	MTDO1	IU	CMOS	**JTAG master 1 data input** **Note: It is input pin on AST2600**

JTAG Master Port 2 – 5 pins

Ball	Signal	I/O	Type	Description
		IO Power Domain : PV33D		
D17	GPIOI0	ID/O8	CMOS	**GPIO group I bit 0 (default)**
	MNTRST2	O8	CMOS	**JTAG master 2 reset output**
	TXD12	O8	CMOS	**UART 12 transmit serial data output (Dual Ball-out)**
A16	GPIOI1	ID/O8	CMOS	**GPIO group I bit 1 (default)**
	MTDI2	O8	CMOS	**JTAG master 2 data output** **Note: It is output pin on AST2600**
	RXD12	ID	CMOS	**UART 12 receive serial data input (Dual Ball-out)**
E17	GPIOI2	ID/O8	CMOS	**GPIO group I bit 2 (default)**
	MTCK2	O8	CMOS	**JTAG master 2 clock output**
	TXD13	O8	CMOS	**UART 13 transmit serial data output (Dual Ball-out)**
D16	GPIOI3	ID/O8	CMOS	**GPIO group I bit 3 (default)**
	MTMS2	O8	CMOS	**JTAG master 2 mode select output**
	RXD13	ID	CMOS	**UART 13 receive serial data input (Dual Ball-out)**
C16	GPIOI4	ID/O8	CMOS	**GPIO group I bit 4 (default)**
	MTDO2	ID	CMOS	**JTAG master 2 data input** **Note: It is input pin on AST2600**

Miscellaneous – 7 pins

Ball	Signal	I/O	Type	Description
		IO Power Domain : PV33D_RGM		
C10	ENTEST	IUS	CMOS	**Enable test mode input pin** Pull low the ENTEST to ground (1KΩ) in normal operation mode.
E10	SRST#	IUS	CMOS	**Chip level reset input pin** Keep SRST# Low after power-on and power stable for a period of time (minimum 1ms). Do not trigger this pin when host VGA function is ON. This reset input will reset whole chip functions, including PCIe and VGA. Please use EXTRST# for run-time reset request.

to next page

from previous page

D7	SSPRST#	IUS	CMOS	**Reset input pin for SSP CPU (ARM Cortex-M3)**
B10	EXTRST#	IUS	CMOS	**SOC level reset input pin** SOC level reset with programmable reset control at register SCU060. Use this pin to reset firmware at normal working mode. This pin has the same capability as internal watchdog reset.
D10	CLKIN	IU	CMOS	**External 25MHz reference clock input pin**
B7	RSTIND#	O8	CMOS	**CPU reset indication output**
IO Power Domain : BATVDD				
AB21	CHASI#	IS	CMOS	**Chassis intrusion detection**

USB 2.0 Host/Slave Port – 6 pins

Ball	Signal	I/O	Type	Description
A4	USB2A_DP	I/O	A	$D+$ **signal of USB 2.0 port A**
B4	USB2A_DN	I/O	A	$D-$ **signal of USB 2.0 port A** USB2.0 port A can be configured to below modes: 1. USB2.0 Virtual Hub 2. USB2.0 Host port 1 3. PCIe-EHCI host port
A6	USB2B_DP	I/O	A	$D+$ **signal of USB 2.0 port B**
B6	USB2B_DN	I/O	A	$D-$ **signal of USB 2.0 port B** USB2.0 port B can be configured to below modes: 1. USB2.0 device 2. USB2.0 Host port 2 3. USB1.1 HID device
K10	USB2AV18	-	P	**USB PHY 1.8V analog power** 1uF + 0.1uF decouple capacitor required.
J10	USB2AV33	-	P	**USB PHY 3.3V analog power** 1uF + 0.1uF decouple capacitor required.

DAC – 6 pins

Ball	Signal	I/O	Type	Description
AF21	DACR	O	A	**DAC R channel output**
AE21	DACG	O	A	**DAC G channel output**
AD21	DACB	O	A	**DAC B channel output**
AC21	DACRSET	-	A	**DAC reference resistor** Connect an external resistor of 2.74KΩ to GND for adjusting the magnitude of DAC full-scale output current.
V21 W21	DACAV33 DACAV33	- -	P P	**3.3V DAC analog power**

ADC – 26 pins

Ball	Signal	I/O	Type	Description
AD20	ADC0	I	A	**ADC channel 0 analog input**
	GPIT0	I	CMOS	**GPIO group T bit 0**
AC18	ADC1	I	A	**ADC channel 1 analog input**
	GPIT1	I	CMOS	**GPIO group T bit 1**
AE19	ADC2	I	A	**ADC channel 2 analog input**
	GPIT2	I	CMOS	**GPIO group T bit 2**
AD19	ADC3	I	A	**ADC channel 3 analog input**
	GPIT3	I	CMOS	**GPIO group T bit 3**
AC19	ADC4	I	A	**ADC channel 4 analog input**
	GPIT4	I	CMOS	**GPIO group T bit 4**
AB19	ADC5	I	A	**ADC channel 5 analog input**
	GPIT5	I	CMOS	**GPIO group T bit 5**
AB18	ADC6	I	A	**ADC channel 6 analog input**
	GPIT6	I	CMOS	**GPIO group T bit 6**
AE18	ADC7	I	A	**ADC channel 7 analog input**
	GPIT7	I	CMOS	**GPIO group T bit 7**
AB16	ADC8	I	A	**ADC channel 8 analog input**
	GPIU0	I	CMOS	**GPIO group U bit 0**
	SALT9	I	CMOS	**SMBus 9 Alert pin (Dual Ball-out)**
AA17	ADC9	I	A	**ADC channel 9 analog input**
	GPIU1	I	CMOS	**GPIO group U bit 1**
	SALT10	I	CMOS	**SMBus 10 Alert pin (Dual Ball-out)**
AB17	ADC10	I	A	**ADC channel 10 analog input**
	GPIU2	I	CMOS	**GPIO group U bit 2**
	SALT11	I	CMOS	**SMBus 11 Alert pin (Dual Ball-out)**
AE16	ADC11	I	A	**ADC channel 11 analog input**
	GPIU3	I	CMOS	**GPIO group U bit 3**
	SALT12	I	CMOS	**SMBus 12 Alert pin (Dual Ball-out)**
AC16	ADC12	I	A	**ADC channel 12 analog input**
	GPIU4	I	CMOS	**GPIO group U bit 4**
	SALT13	I	CMOS	**SMBus 13 Alert pin (Dual Ball-out)**
AA16	ADC13	I	A	**ADC channel 13 analog input**
	GPIU5	I	CMOS	**GPIO group U bit 5**
	SALT14	I	CMOS	**SMBus 14 Alert pin (Dual Ball-out)**
AD16	ADC14	I	A	**ADC channel 14 analog input**
	GPIU6	I	CMOS	**GPIO group U bit 6**
	SALT15	I	CMOS	**SMBus 15 Alert pin (Dual Ball-out)**
AC17	ADC15	I	A	**ADC channel 15 analog input**
	GPIU7	I	CMOS	**GPIO group U bit 7**
	SALT16	I	CMOS	**SMBus 16 Alert pin (Dual Ball-out)**
AC20	ADCVREFP0	O	A	**ADC positive reference voltage for ADC[7:0] pins**
AD17	ADCVREFP1			**ADC positive reference voltage for ADC[15:8] pins**

to next page

from previous page

AB20 AD18	ADCVREFN0 ADCVREFN1	O	A	**ADC negative reference voltage for ADC[7:0] pins** **ADC negative reference voltage for ADC[15:8] pins**
AF20 AF18	ADCREXT0 ADCREXT1	O	A	**ADC external resistor connection pin for ADC[7:0] pins.** **ADC external resistor connection pin for ADC[15:8] pins.** Connect resistor 50KΩ to ground for sensor source current adjustment.
AF19 AF17	ADCVREFEXT0 ADCVREFEXT1	-	P	**ADC external reference voltage for ADC[7:0] pins** **ADC external reference voltage for ADC[15:8] pins** Voltage Range: 0.9V to 2.7V
Y21 AA21	ADCAV33 ADCAV33	- -	P P	**3.3V ADC analog power**

PLL Power – 7 pins

Ball	Signal	I/O	Type	Description
J9	DPLLAVDD	-	P	**1.0V DPLL analog power**
E7 F7	PLLAVDD PLLAVDD	- -	P P	**1.0V EPLL/MPLL/HPLL analog power**
E9 F9	PLLDVDD PLLDVDD	- -	P P	**1.0V DPLL/EPLL/MPLL/HPLL digital power**
H14	PLLAHVDD	-	P	**1.2V PLL analog power for A2/A3 version** **3.3V PLL analog power for A0/A1 version**
N5	AVDDPLL	-	P	**1.8V DDR4 PHY PLL power**

Power and Ground – 242 pins

Ball	Signal	I/O	Type	Description
L9	IV10D	-	P	**1.0V Core logic power**
L10	IV10D	-	P	
M8	IV10D	-	P	
N8	IV10D	-	P	
P8	IV10D	-	P	
M11	IV10D	-	P	
N11	IV10D	-	P	
P11	IV10D	-	P	
R9	IV10D	-	P	
R10	IV10D	-	P	
K16	IV12D	-	P	**1.2V Core logic power**
K17	IV12D	-	P	
K18	IV12D	-	P	
K19	IV12D	-	P	
L14	IV12D	-	P	
M14	IV12D	-	P	
N14	IV12D	-	P	
P14	IV12D	-	P	
R14	IV12D	-	P	
T14	IV12D	-	P	

to next page

from previous page

L21	IV12D	-	P	
M21	IV12D	-	P	
N21	IV12D	-	P	
P21	IV12D	-	P	
R21	IV12D	-	P	
T21	IV12D	-	P	
U15	IV12D	-	P	
U16	IV12D	-	P	
U17	IV12D	-	P	
U18	IV12D	-	P	
G6	MVDD	-	P	**1.2V DDR4 SDRAM PHY and Memory I/O power**
H6	MVDD	-	P	
J6	MVDD	-	P	
K6	MVDD	-	P	
L6	MVDD	-	P	
P6	MVDD	-	P	
R6	MVDD	-	P	
T6	MVDD	-	P	
M6	MVDD_CK	-	P	**1.2V DDR4 SDRAM PHY CK I/O power**
T8	DP_VCC10A	-	P	**DisplayPort PHY 1.0V analog power**
V8	DP_VCC18A	-	P	**DisplayPort PHY 1.8V analog power**
T10	PE_VCC10A	-	P	**PCIe End-point PHY 1.0V analog power**
V10	PE_VCC18A	-	P	**PCIe End-point PHY 1.8V analog power**
T9	RC_VCC10A	-	P	**PCIe Root-complex PHY 1.0V analog power**
V9	RC_VCC18A	-	P	**PCIe Root-complex PHY 1.8V analog power**
W15	LPVDD	-	P	**1.8V/3.3V Adjustable I/O power for LPC/eSPI**
L22	RVDD	-	P	**1.8V/2.5V/3.3V Adjustable I/O power for RGMII3/RGMII4**
M22	RVDD	-	P	**Note: RGMII3 and RGMII4 must feed the same voltage level**
J21	RVDD	-	P	
K21	RVDD	-	P	
G21	SD1VDD	-	P	**1.8V/3.3V Adjustable I/O power for SD1**
H21	SD1VDD	-	P	
H18	SD2VDD	-	P	**1.8V/3.3V Adjustable I/O power for SD2**
H19	SD2VDD	-	P	
V22	I3CVDDL1	-	P	**1.2V/1.0V Adjustable I/O power for I3C Low-voltage Bus 1/2**
U21	I3CVDDL2	-	P	**1.2V/1.0V Adjustable I/O power for I3C Low-voltage Bus 3/4**
F19	I3CVDDH	-	P	**1.8V/3.3V Adjustable I/O power for I3C High-voltage Bus 3/4/5/6**
F20	I3CVDDH	-	P	
V23	BATVDD	-	P	**Battery backed SRAM power, not including RTC**
L13	PV18D_SLI	-	P	**1.8V SLI power domain**
M13	PV18D_SLI	-	P	
N13	PV18D_SLI	-	P	

to next page

from previous page

P13	PV18D_SLI	-	P	
R13	PV18D_SLI	-	P	
T13	PV18D_SLI	-	P	
U13	PV18D_SLI	-	P	
V13	PV18D_SLI	-	P	
W13	PV18D_SLI	-	P	
W14	PV18D_SLI	-	P	
H12	PV18D_RGM	-	P	**1.8V RGM power domain**
J12	PV18D_RGM	-	P	
U6	PV18D	-	P	**1.8V General purposeI/O power**
V6	PV18D	-	P	
H8	PV18D	-	P	
J8	PV18D	-	P	
N12	PV18D	-	P	
P12	PV18D	-	P	
R12	PV18D	-	P	
K11	PV33D_RGM	-	P	**3.3V RGM power domain**
K12	PV33D_RGM	-	P	
H15	PV33D	-	P	**3.3V General purpose I/O power**
H16	PV33D	-	P	
H17	PV33D	-	P	
W16	PV33D	-	P	
W17	PV33D	-	P	
W18	PV33D	-	P	
W19	PV33D	-	P	
N22	PV33D	-	P	
P22	PV33D	-	P	
R22	PV33D	-	P	
T22	PV33D	-	P	
A1	GND	-	P	**Ground**
A5	GND	-	P	
A26	GND	-	P	
B5	GND	-	P	
B11	GND	-	P	
B15	GND	-	P	
B19	GND	-	P	
B23	GND	-	P	
C3	GND	-	P	
D25	GND	-	P	
E8	GND	-	P	
F6	GND	-	P	
F8	GND	-	P	
F10	GND	-	P	

to next page

from previous page

F12	GND	-	P	
F14	GND	-	P	
F16	GND	-	P	
F17	GND	-	P	
F18	GND	-	P	
F21	GND	-	P	
G2	GND	-	P	
G25	GND	-	P	
H4	GND	-	P	
H9	GND	-	P	
H10	GND	-	P	
H11	GND	-	P	
H13	GND	-	P	
J2	GND	-	P	
J11	GND	-	P	
J13	GND	-	P	
J14	GND	-	P	
J15	GND	-	P	
J16	GND	-	P	
J17	GND	-	P	
J18	GND	-	P	
J19	GND	-	P	
K4	GND	-	P	
K8	GND	-	P	
K9	GND	-	P	
K13	GND	-	P	
K14	GND	-	P	
K15	GND	-	P	
K22	GND	-	P	
L2	GND	-	P	
L8	GND	-	P	
L11	GND	-	P	
L12	GND	-	P	
L15	GND	-	P	
L16	GND	-	P	
L17	GND	-	P	
L18	GND	-	P	
L19	GND	-	P	
L25	GND	-	P	
M4	GND	-	P	
M9	GND	-	P	
M10	GND	-	P	
M12	GND	-	P	

to next page

from previous page

M15	GND	-	P	
M16	GND	-	P	
M17	GND	-	P	
M18	GND	-	P	
M19	GND	-	P	
N2	GND	-	P	
N6	GND	-	P	
N9	GND	-	P	
N10	GND	-	P	
N15	GND	-	P	
N16	GND	-	P	
N17	GND	-	P	
N18	GND	-	P	
N19	GND	-	P	
P4	GND	-	P	
P9	GND	-	P	
P10	GND	-	P	
P15	GND	-	P	
P16	GND	-	P	
P17	GND	-	P	
P18	GND	-	P	
P19	GND	-	P	
R2	GND	-	P	
R8	GND	-	P	
R11	GND	-	P	
R15	GND	-	P	
R16	GND	-	P	
R17	GND	-	P	
R18	GND	-	P	
R19	GND	-	P	
R25	GND	-	P	
T4	GND	-	P	
T11	GND	-	P	
T12	GND	-	P	
T15	GND	-	P	
T16	GND	-	P	
T17	GND	-	P	
T18	GND	-	P	
T19	GND	-	P	
U2	GND	-	P	
U8	GND	-	P	
U9	GND	-	P	
U10	GND	-	P	

to next page

from previous page

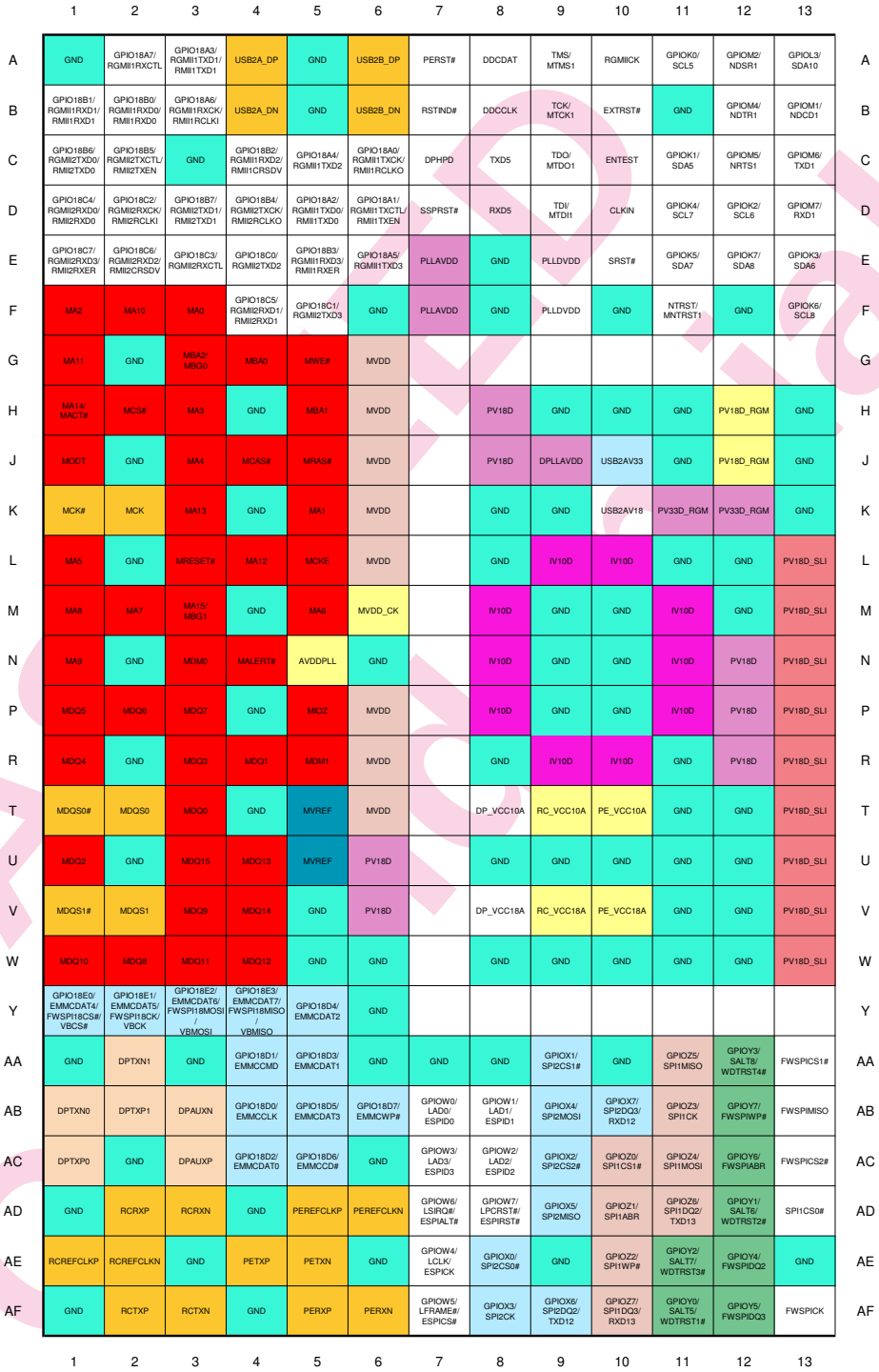
U11	GND	-	P	
U12	GND	-	P	
U14	GND	-	P	
U19	GND	-	P	
U22	GND	-	P	
V5	GND	-	P	
V11	GND	-	P	
V12	GND	-	P	
V14	GND	-	P	
V15	GND	-	P	
V16	GND	-	P	
V17	GND	-	P	
V18	GND	-	P	
V19	GND	-	P	
W5	GND	-	P	
W6	GND	-	P	
W8	GND	-	P	
W9	GND	-	P	
W10	GND	-	P	
W11	GND	-	P	
W12	GND	-	P	
W22	GND	-	P	
W25	GND	-	P	
Y6	GND	-	P	
Y22	GND	-	P	
AA1	GND	-	P	
AA3	GND	-	P	
AA6	GND	-	P	
AA7	GND	-	P	
AA8	GND	-	P	
AA10	GND	-	P	
AA14	GND	-	P	
AA15	GND	-	P	
AA19	GND	-	P	
AA20	GND	-	P	
AA22	GND	-	P	
AC2	GND	-	P	
AC6	GND	-	P	
AC25	GND	-	P	
AD1	GND	-	P	
AD4	GND	-	P	
AE3	GND	-	P	
AE6	GND	-	P	

to next page

from previous page

AE9	GND	-	P	
AE13	GND	-	P	
AE17	GND	-	P	
AE20	GND	-	P	
AE23	GND	-	P	
AF1	GND	-	P	
AF4	GND	-	P	
AF26	GND	-	P	
U23	GND	-	P	

2.2 Ball Map



	1	2	3	4	5	6	7	8	9	10	11	12	13	
A	GND	GPIO18A7/ RGMII1RXCTL	GPIO18A3/ RGMII1TXD1/ RMII1TXD1	USB2A_DP	GND	USB2B_DP	PERST#	DDCDAT	TMS/ MTMS1	RGMIICK	GPIOK0/ SCL5	GPIOM2/ NDSR1	GPIOL3/ SDA10	A
B	GPIO18B1/ RGMII1RXD1/ RMII1RXD1	GPIO18B0/ RGMII1RXD0/ RMII1RXD0	GPIO18A6/ RGMII1RXCK/ RMII1RCLKI	USB2A_DN	GND	USB2B_DN	RSTIND#	DDCCLK	TCK/ MTCK1	EXTRST#	GND	GPIOM4/ NDTR1	GPIOM1/ NDCD1	B
C	GPIO18B6/ RGMII2TXD0/ RMII2TXD0	GPIO18B5/ RGMII2TXCTL/ RMII2TXEN	GND	GPIO18B2/ RGMII1RXD2/ RMII1CRSDV	GPIO18A4/ RGMII1TXD2	GPIO18A0/ RGMII1TXCK/ RMII1RCLKO	DPHPD	TXD5	TDO/ MTDO1	ENTEST	GPIOK1/ SDA5	GPIOM5/ NRTS1	GPIOM6/ TXD1	C
D	GPIO18C4/ RGMII2RXD0/ RMII2RXD0	GPIO18C2/ RGMII2RXCK/ RMII2RCLKI	GPIO18B7/ RGMII2TXD1/ RMII2TXD1	GPIO18B4/ RGMII2TXCK/ RMII2RCLKO	GPIO18A2/ RGMII1TXD0/ RMII1TXD0	GPIO18A1/ RGMII1TXCTL/ RMII1TXEN	SSPRST#	RXD5	TDI/ MTDI1	CLKIN	GPIOK4/ SCL7	GPIOK2/ SCL6	GPIOM7/ RXD1	D
E	GPIO18C7/ RGMII2RXD3/ RMII2RXER	GPIO18C6/ RGMII2RXD2/ RMII2CRSDV	GPIO18C3/ RGMII2RXCTL	GPIO18C0/ RGMII2TXD2	GPIO18B3/ RGMII1RXD3/ RMII1RXER	GPIO18A5/ RGMII1TXD3	PLLAVDD	GND	PLLDVDD	SRST#	GPIOK5/ SDA7	GPIOK7/ SDA8	GPIOK3/ SDA6	E
F	MA2	MA10	MA0	GPIO18C5/ RGMII2RXD1/ RMII2RXD1	GPIO18C1/ RGMII2TXD3	GND	PLLAVDD	GND	PLLDVDD	GND	NTRST/ MNTRST1	GND	GPIOK6/ SCL8	F
G	MA11	GND	MBA2/ MBG0	MBA0	MWE#	MVDD								G
H	MA14/ MACT#	MCS#	MA3	GND	MBA1	MVDD		PV18D	GND	GND	GND	PV18D_RGM	GND	H
J	MODT	GND	MA4	MCAS#	MRAS#	MVDD		PV18D	DPLLAVDD	USB2AV33	GND	PV18D_RGM	GND	J
K	MCK#	MCK	MA13	GND	MA1	MVDD		GND	GND	USB2AV18	PV33D_RGM	PV33D_RGM	GND	K
L	MA5	GND	MRESET#	MA12	MCKE	MVDD		GND	IV10D	IV10D	GND	GND	PV18D_SLI	L
M	MA8	MA7	MA15/ MBG1	GND	MA6	MVDD_CK		IV10D	GND	GND	IV10D	GND	PV18D_SLI	M
N	MA9	GND	MDM0	MALERT#	AVDDPLL	GND		IV10D	GND	GND	IV10D	PV18D	PV18D_SLI	N
P	MDQ5	MDQ6	MDQ7	GND	MIOZ	MVDD		IV10D	GND	GND	IV10D	PV18D	PV18D_SLI	P
R	MDQ4	GND	MDQ3	MDQ1	MDM1	MVDD		GND	IV10D	IV10D	GND	PV18D	PV18D_SLI	R
T	MDQS0#	MDQS0	MDQ0	GND	MVREF	MVDD		DP_VCC10A	RC_VCC10A	PE_VCC10A	GND	GND	PV18D_SLI	T
U	MDQ2	GND	MDQ15	MDQ13	MVREF	PV18D		GND	GND	GND	GND	GND	PV18D_SLI	U
V	MDQS1#	MDQS1	MDQ9	MDQ14	GND	PV18D		DP_VCC18A	RC_VCC18A	PE_VCC18A	GND	GND	PV18D_SLI	V
W	MDQ10	MDQ8	MDQ11	MDQ12	GND	GND		GND	GND	GND	GND	GND	PV18D_SLI	W
Y	GPIO18E0/ EMMCDAT4/ FWSPI18CS#/ VBCS#	GPIO18E1/ EMMCDAT5/ FWSPI18CK/ VBCK	GPIO18E2/ EMMCDAT6/ FWSPI18MOSI / VBMOSI	GPIO18E3/ EMMCDAT7/ FWSPI18MISO / VBMISO	GPIO18D4/ EMMCDAT2	GND								Y
AA	GND	DPTXN1	GND	GPIO18D1/ EMMCCMD	GPIO18D3/ EMMCDAT1	GND	GND	GND	GPIOX1/ SPI2CS1#	GND	GPIOZ5/ SPI1MISO	GPIOY3/ SALT8/ WDTRST4#	FWSPICS1#	AA
AB	DPTXN0	DPTXP1	DPAUXN	GPIO18D0/ EMMCCLK	GPIO18D5/ EMMCDAT3	GPIO18D7/ EMMCWP#	GPIOW0/ LAD0/ ESPID0	GPIOW1/ LAD1/ ESPID1	GPIOX4/ SPI2MOSI	GPIOX7/ SPI2DQ3/ RXD12	GPIOZ3/ SPI1CK	GPIOY7/ FWSPIWP#	FWSPIMISO	AB
AC	DPTXP0	GND	DPAUXP	GPIO18D2/ EMMCDAT0	GPIO18D6/ EMMCCD#	GND	GPIOW3/ LAD3/ ESPID3	GPIOW2/ LAD2/ ESPID2	GPIOX2/ SPI2CS2#	GPIOZ0/ SPI1CS1#	GPIOZ4/ SPI1MOSI	GPIOY6/ FWSPIABR	FWSPICS2#	AC
AD	GND	RCRXP	RCRXN	GND	PEREFCLKP	PEREFCLKN	GPIOW6/ LSIRQ#/ ESPIALT#	GPIOW7/ LPCRST#/ ESPIRST#	GPIOX5/ SPI2MISO	GPIOZ1/ SPI1ABR	GPIOZ6/ SPI1DQ2/ TXD13	GPIOY1/ SALT6/ WDTRST2#	SPI1CS0#	AD
AE	RCREFCLKP	RCREFCLKN	GND	PETXP	PETXN	GND	GPIOW4/ LCLK/ ESPICK	GPIOX0/ SPI2CS0#	GND	GPIOZ2/ SPI1WP#	GPIOY2/ SALT7/ WDTRST3#	GPIOY4/ FWSPIDQ2	GND	AE
AF	GND	RCTXP	RCTXN	GND	PERXP	PERXN	GPIOW5/ LFRAME#/ ESPICS#	GPIOX3/ SPI2CK	GPIOX6/ SPI2DQ2/ TXD12	GPIOZ7/ SPI1DQ3/ RXD13	GPIOY0/ SALT5/ WDTRST1#	GPIOY5/ FWSPIDQ3	FWSPICK	AF
	1	2	3	4	5	6	7	8	9	10	11	12	13	

Figure 4: Ball Map – Left Side

	14	15	16	17	18	19	20	21	22	23	24	25	26	
A	GPIOL1/ SDA9	GPIOI7/ SIOSCI#	GPIOI1/ MTDI2/ RXD12	GPIOH3/ SGPM1I	GPIOH0/ SGPM1CK	GPIOJ5/ HVI3C5SDA/ SDA3	GPIOJ1/ HVI3C3SDA/ SDA1	GPIOG4/ TXD8/ SD2DAT2/ SALT13	GPIOG3/ RXD7/ SD2DAT1/ SALT12	GPIOF7/ SD1WPI#/ PWM15	GPIOF6/ SD1CDI#/ PWM14	GPIOF5/ SD1DAT3/ PWM13	GND	A
B	GPIOL6/ VGAHS	GND	GPIOI6/ SIOPBI#	GPIOH5/ SGPS1LD/ SDA15	GPIOH1/ SGPM1LD	GND	GPIOJ0/ HVI3C3SCL/ SCL1	GPIOG7/ RXD9/ SD2WPI#/ SALT16	GPIOG1/ RXD6/ SD2CMD/ SALT10	GND	GPIOE7/ NRTS4/ RGMII4RXD3/ RMII4RXER	GPIOE6/ NDTR4/ RGMII4RXD2/ RMII4CRSDV	GPIOE5/ NRI4/ RGMII4RXD1/ RMII4RXD1	B
C	GPIOL7/ VGAVS	GPIOL4/ TXD3	GPIOI4/ MTDO2	GPIOH6/ SGPS1I0/ SCL16	GPIOH2/ SGPM1O	GPIOJ4/ HVI3C5SCL/ SCL3	GPIOJ6/ HVI3C6SCL/ SCL4	GPIOG2/ TXD7/ SD2DAT0/ SALT11	GPIOF4/ SD1DAT2/ PWM12	GPIOF3/ SD1DAT1/ PWM11	GPIOE4/ NDSR4/ RGMII4RXD0/ RMII4RXD0	GPIOE2/ NCTS4/ RGMII4RXCK/ RMII4RCLKI	GPIOE3/ NDCD4/ RGMII4RXCTL	C
D	GPIOM0/ NCTS1	GPIOL0/ SCL9	GPIOI3/ MTMS2/ RXD13	GPIOI0/ MNTRST2/ TXD12	GPIOH4/ SGPS1CK/ SCL15	GPIOJ7/ HVI3C6SDA/ SDA4	GPIOJ3/ HVI3C4SDA/ SDA2	GPIOG6/ TXD9/ SD2CD#/ SALT15	GPIOF0/ SD1CLK/ PWM8	GPIOF2/ SD1DAT0/ PWM10	GPIOE1/ NRTS3/ RGMII4TXD3	GND	GPIOE0/ NDTR3/ RGMII4TXD2	D
E	GPIOM3/ NRI1	GPIOL2/ SCL10	GPIOI5/ SIOPBO#	GPIOI2/ MTCK2/ TXD13	GPIOH7/ SGPS1I1/ SDA16	GPIOJ2/ HVI3C4SCL/ SCL2	GPIOG5/ RXD8/ SD2DAT3/ SALT14	GPIOG0/ TXD6/ SD2CLK/ SALT9	GPIOF1/ SD1CMD/ PWM9	GPIOD5/ NDCD3/ RGMII4TXCTL/ RMII4TXEN	GPIOD6/ NDSR3/ RGMII4TXD0/ RMII4TXD0	GPIOD7/ NRI3/ RGMII4TXD1/ RMII4TXD1	GPIOD3/ RGMII3RXD3/ RMII3RXER	E
F	GND	GPIOL5/ RXD3	GND	GND	GND	I3CVDDH	I3CVDDH	GND	GPIOC5/ RGMII3TXD3	GPIOD0/ RGMII3RXD0/ RMII3RXD0	GPIOD4/ NCTS3/ RGMII4TXCK/ RMII4RCLKO	GPIOD2/ RGMII3RXD2/ RMII3CRSDV	GPIOD1/ RGMII3RXD1/ RMII3RXD1	F
G								SD1VDD	GPIOC4/ RGMII3TXD2	GPIOC6/ RGMII3RXCK/ RMII3RCLKI	GPIOC7/ RGMII3RXCTL	GND	GPIOB5/ MDIO2	G
H	PLLAHVDD	PV33D	PV33D	PV33D	SD2VDD	SD2VDD		SD1VDD	GPIOC2/ RGMII3TXD0/ RMII3TXD0	GPIOC3/ RGMII3TXD1/ RMII3TXD1	GPIOC0/ RGMII3TXCK/ RMII3RCLKO	GPIOB6/ TXD4	GPIOB2/ SALT3	H
J	GND	GND	GND	GND	GND	GND		RVDD	GPIOC1/ RGMII3TXCTL/ RMII3TXEN	GPIOB4/ MDC2	GPIOB7/ RXD4	GPIOB3/ SALT4	GPIOB0/ SALT1	J
K	GND	GND	IV12D	IV12D	IV12D	IV12D		RVDD	GND	GPIOB1/ SALT2	GPIOA3/ MDIO4/ SDA12	GPIOA7/ MAC4LINK/ SDA14/ SGPS2I1/ SGPM2I	GPIOA4/ MAC1LINK/ SCL13/ SGPS2CK/ SGPM2CK	K
L	IV12D	GND	GND	GND	GND	GND		IV12D	RVDD	GPIOA6/ MAC3LINK/ SCL14/ SGPS2I0/ SGPM2O	GPIOA5/ MAC2LINK/ SDA13/ SGPS2LD/ SGPM2LD	GND	GPIOA2/ MDC4/ SCL12	L
M	IV12D	GND	GND	GND	GND	GND		IV12D	RVDD	GPION5/ NRTS2	GPIOA0/ MDC3/ SCL11	GPIOA1/ MDIO3/ SDA11	GPION7/ RXD2	M
N	IV12D	GND	GND	GND	GND	GND		IV12D	PV33D	GPION1/ NDCD2	GPION3/ NRI2	GPION2/ NDSR2	GPION6/ TXD2	N
P	IV12D	GND	GND	GND	GND	GND		IV12D	PV33D	GPIOS6/ TXD11	GPIOS5/ RXD10	GPION0/ NCTS2	GPION4/ NDTR2	P
R	IV12D	GND	GND	GND	GND	GND		IV12D	PV33D	GPIOS0/ MDC1	GPIOS3/ OSCCLK	GND	GPIOS4/ TXD10	R
T	IV12D	GND	GND	GND	GND	GND		IV12D	PV33D	GPIOR5/ TACH13	GPIOS7/ RXD11	GPIOS1/ MDIO1	GPIOS2/ PEWAKE#	T
U	GND	IV12D	IV12D	IV12D	IV12D	GND		I3CVDDL2	GND	GND	GPIOR1/ TACH9	GPIOR4/ TACH12	GPIOR7/ TACH15	U
V	GND	GND	GND	GND	GND	GND		DACAV33	I3CVDDL1	BATVDD	GPIOR2/ TACH10	GPIOR0/ TACH8	GPIOR3/ TACH11	V
W	PV18D_SLI	LPVDD	PV33D	PV33D	PV33D	PV33D		DACAV33	GND	GPIOP4/ PWM12/ THRUIN2	GPIOP1/ PWM9/ THRUOUT0	GND	GPIOR6/ TACH14	W
Y								ADCAV33	GND	GPIOP7/ PWM15/ HBLED#	GPIOQ2/ TACH2	GPIOQ6/ TACH6	GPIOQ4/ TACH4	Y
AA	GND	GND	ADC13/ GPIU5/ SALT14	ADC9/ GPIU1/ SALT10	PECIVDD	GND	GND	ADCAV33	GND	GPIOP2/ PWM10/ THRUIN1	GPIOP3/ PWM11/ THRUOUT1	GPIOQ0/ TACH0	GPIOQ7/ TACH7	AA
AB	FWSPICS0#	GPIOV0/ SIOS3#	ADC8/ GPIU0/ SALT9	ADC10/ GPIU2/ SALT11	ADC6/ GPIT6	ADC5/ GPIT5	ADCVREFN0	CHASI#	GPIOP0/ PWM8/ THRUIN0	GPIOP5/ PWM13/ THRUOUT2	GPIOP6/ PWM14	GPIOQ1/ TACH1	GPIOQ3/ TACH3	AB
AC	FWSPIMOSI	GPIOV3/ SIOONCTRL#	ADC12/ GPIU4/ SALT13	ADC15/ GPIU7/ SALT16	ADC1/ GPIT1	ADC4/ GPIT4	ADCVREFP0	DACRSET	GPIOO5/ PWM5	GPIOO7/ PWM7	GPIOO6/ PWM6	GND	GPIOQ5/ TACH5	AC
AD	GPIOV2/ SIOPWREQ#	GPIOV6/ LPCPME#	ADC14/ GPIU6/ SALT15	ADCVREFP1	ADCVREFN1	ADC3/ GPIT3	ADC0/ GPIT0	DACB	GPIOO1/ PWM1	GPIOO2/ PWM2	GPIOO3/ PWM3	GPIOO4/ PWM4	GPIOO0/ PWM0	AD
AE	GPIOV5/ LPCPD#	GPIOV4/ SIOPWRGD	ADC11/ GPIU3/ SALT12	GND	ADC7/ GPIT7	ADC2/ GPIT2	GND	DACG	I3C2SDA	GND	I3C1SDA	I3C4SCL/ FSI2CLK	I3C3SDA/ FSI1DATA	AE
AF	GPIOV1/ SIOS5#	GPIOV7/ LPCSMI#	PECI	ADCVREFEXT1	ADCREXT1	ADCVREFEXT0	ADCREXT0	DACR	I3C2SCL	I3C1SCL	I3C4SDA/ FSI2DATA	I3C3SCL/ FSI1CLK	GND	AF
	14	15	16	17	18	19	20	21	22	23	24	25	26	

Figure 5: Ball Map – Right Side

2.3 Hardware Pin Strap

The strap registers, SCU500 and SCU510, define the functions that are set after power on reset. Please reference SCU500 and SCU510 for detail function descriptions. Each bit in the strap register has its strap sources. The source can be an external pin, OTP memory bit or combination of pin and OTP. Please reference to 9.5Hardware Strap Source Mappings for more detail.
AST2600 defines 2 methods for hardware functional strap setting input.

1. Method 1: Pin Strap: Load from dedicated strap pins.
2. Method 2: OTP Strap: Load from the OTP memory when SRST# active.

The hardware strap registers strap the external pin voltage state or value in OTPSTRAP registers when SRST# reset input is low and keep the value unchanged after SRST# goes high. The strap values are stored to SCU500 and SCU510 in the System Control Unit (SCU) module. After SRST# reset load, the SCU500 and SCU510 registers are still writable. Set selected bits in register SCU508, SCU508, OTPCFG28 and OTPCFG29 can enable the write protection for the corresponding bits in SCU500 and SCU510.

When GPIO is used as hardware strap, it will limit the active level selection of the GPIO application. Ex. when strap setting is pulled high, then it can be used as a GPIO of active low that have a power up state of high.

2.3.1 Method 1: Load from dedicated strap pins

Strap function pins when SCU510[30]=0

There are 9 dedicated strap function pins when SCU510[30]=0. Listed as below:

Table 4: Dedicated Strap Function Pins (SCU510[30]=0)

TXD5*	SCU500[0]	Disable ARM CA7 CPU boot	
GPIOZ3	SCU51C[6]	Disable VGA device(PCIe device 0)	(no similar OTPSTRAP)
GPIOZ4	SCU51C[7]	Select LPC/eSPI	(or with OTPSTRAP[38])
GPIOZ5	SCU51C[8]	Disable HeartBeat LED	(no similar OTPSTRAP)
GPIOZ6	SCU51C[9]	Enable GPIO Pass Through	(or with OTPSTRAP[60])
GPIOZ7	SCU51C[10]	Enable ACPI function	(or with OTPSTRAP[37])
FWSPICK	SCU510[8]	Enable Boot from Uart1 or Uart5	
FWSPIMOSI*	SCU510[31]	Enable Secure Boot by Pin Strap	
FWSPIMISO	SCU51C[0]	Enable Low-Security Boot function	

Notes

- If it is necessary to use these hardware strap as GPIO application, please note it must be latched after SRST# de-asserted every time.
- Don't drive these hadware strap pin from other output source directly.
- *. The TXD5 hardware strap is reversed. It means a value 0 on strap pin will trap value 1 into strap register SCU500[0], a value 0 on strap pin will trap value 1 into strap register SCU500[0]. All of the other hardware strap pins are not reversed. Please reference to section 2.3.3 for more detail.
- Please ensure that the HIGH/LOW on these pins is meeting the TIH hold time on AC timing specification of Strap Input Interface..
- For the hardware strap function setting on GPIOZ[7:3] pins, it can also be set the HIGH/LOW status by programming internal OTP strap memory. After SRST# de-asserted, the internal strap value on SCU500/ SCU510/ SCU51C register is latched by external hardware strap level OR OTP strap memory value. This gives you the flexibility of setting OTP strap value =1b or stuff the external pull-up resistor on these strap pin if you need to set this hardware strap function =HIGH.

- All of the debug interfaces controlled by SCU0C8 and SCU0D8 are are disabled in default. Set OTPCFG7 registers can change their default values. The hardware strap pin, FWSPIMISO, can enable Low Secure Boot function. It also can change the default values of SCU0C8 and SCU0D8 and keep UART1/UART5 debug port enabled.
- For GPIOZ4 strap pin on A2/A3 silicon version, this pin include internal pull-down and it is actual hardware strap pin. You can stuff external Pull-up 4.7K ohm resistor to PV33D if host is LPC mode. For A2/A3 version, you do NOT need to enable path code on the SDK ?rmware for this pin as A0/A1 version.
- For GPIOZ6 (PinAD11), it is used as GPIO Pass Through selection:
  0b: Disable GPIO Pass Through function on GPIOP[5:0]
  1b: Enable GPIO Pass Through function on GPIOP[5:0]
  Please note GPIOP[5:0] pins are defined as Pass Through application after SRST# de-asserted when your system includes a pull-up on GPIOZ6 pins. The GPIOZ6 strap setting will affect several pins functionality in your system, please check your schematic and ensure it is meeting your usage case.
- For GPIOZ7 (PinAF10), it is used as ACPI function selection:
  0b: Disable ACPI function
  1b: Enable ACPI function
  Please note some pin are defined as ACPI functional pin such as (SIOS3#, SIOS5#, SIOPWREQ#, SIOONCTRL#, SIOPBI#, SIOPBO#, SIOPWRGD, SIOSCI#) after SRST# de-asserted when your system includes a pull-up on GPIOZ7 pins. The GPIOZ7 strap setting will affect several pin functionality in your system, please check your schematic and ensure it is meeting your usage case.

Strap function pins when SCU510[30]=1

There are 11 dedicated strap function pins when SCU510[30]=1. Listed as below:

Table 5: Dedicated Strap Function Pins (SCU510[30]=1)

TXD5*	SCU500[0]	Disable ARM CA7 CPU boot	
GPIOZ6	SCU510[13]	Boot SPI flash size Bit [0]	(replace OTPSTRAP[45])
GPIOZ7	SCU510[14]	Boot SPI flash size Bit [1]	(replace OTPSTRAP[46])
FWSPICK	SCU510[15]	Boot SPI flash size Bit [2]	(replace OTPSTRAP[47])
GPIOZ3	SCU510[19]	Host SPI flash size Bit [0]	(replace OTPSTRAP[51])
GPIOZ4	SCU510[20]	Host SPI flash size Bit [1]	(replace OTPSTRAP[52])
GPIOZ5	SCU510[21]	Host SPI flash size Bit [2]	(replace OTPSTRAP[53])
GPIOY4	SCU510[25]	Host SPI CRTM size Bit [0]	(replace OTPSTRAP[57])
GPIOY5	SCU510[26]	Host SPI CRTM size Bit [1]	(replace OTPSTRAP[58])
FWSPIMOSI*	SCU510[31]	Enable Secure Boot by Pin Strap	
FWSPIMISO	SCU51C[0]	Enable Low Secure Boot function	

2.3.2 Method 2: OTP Strap – Load from the OTP memory configuration space

The OTP memory is a one time programmable memory. The OTPSTRAP are 64 bits internal signals from OTP memory and can be source of hardware strap registers. Please reference to section 9 ”Hardware Strap Registers” for more details.

2.3.3 Hardware pin strap configuration method

- TXD5: embedded internal pull high resistor of value 26K $\sim$ 71K.
  Please add external 10k ohm pull-up for A0 chip.

  - Strap value = 0, just leave the pin floating or pull up.
  - Strap value = 1, pull-down the pin to ground by a resistor of value 1K $\sim$ 4.7K Ω.
- FWSPICK, FWSPIMOSI, FWSPIMISO: embedded internal pull low of value 33K $\sim$ 89K
  - Strap value = 0, just leave the pin floating.
  - Strap value = 1, pull-high the pin to 3.3V standby power by a resistor of value 1K $\sim$ 4.7K Ω.
- GPIOY[5:4], GPIOZ[7:3]: embedded internal pull low of value 33K $\sim$ 89K
  - Strap value = 0, just leave the pin floating.
  - Strap value = 1, pull-high the pin to 3.3V standby power by a resistor of value 1K $\sim$ 4.7K Ω.

2.3.4 How to disable the Graphics device

This section describe how to disable the Graphics device without disabling the PCIE bridge, MCTP and Host2BMC device.
Set SCUC20[0]=0 (power-on default 1) to disable VGA device (device 0). Below features are valid after this setting.

- PCI scan cant enumerate and allocate device 0.
- It makes device 0 disappear completely.
- It makes AST2600 not respond to any config cycles on device0.

2.4 GPIO Summary

2.4.1 GPIO Table

GPIO	Ball	Pin Name	Driving	Input Buffer	Internal Resistor
GPIOA0	M24	GPIOA0/MDC3/SCL11	16mA	Schmitt	
GPIOA1	M25	GPIOA1/MDIO3/SDA11	16mA	Schmitt	
GPIOA2	L26	GPIOA2/MDC4/SCL12	16mA	Schmitt	
GPIOA3	K24	GPIOA3/MDIO4/SDA12	16mA	Schmitt	
GPIOA4	K26	GPIOA4/MACLINK1/SCL13	8mA	Schmitt	
GPIOA5	L24	GPIOA5/MACLINK2/SDA13	8mA	Schmitt	
GPIOA6	L23	GPIOA6/MACLINK3/SCL14	8mA	Schmitt	
GPIOA7	K25	GPIOA7/MACLINK4/SDA14	8mA	Schmitt	
GPIOB0	J26	GPIOB0/SALT1	16mA	CMOS	PD
GPIOB1	K23	GPIOB1/SALT2	16mA	CMOS	PD
GPIOB2	H26	GPIOB2/SALT3	16mA	CMOS	PD
GPIOB3	J25	GPIOB3/SALT4	16mA	CMOS	PD
GPIOB4	J23	GPIOB4/MDC2	16mA	CMOS	PD
GPIOB5	G26	GPIOB5/MDIO2	16mA	CMOS	PD
GPIOB6	H25	GPIOB6/TXD4	16mA	CMOS	PD
GPIOB7	J24	GPIOB7/RXD4	16mA	CMOS	PD
GPIOC0	H24	GPIOC0/RGMII3TXCK/RMII3RCLKO	8mA	CMOS	
GPIOC1	J22	GPIOC1/RGMII3TXCTL/RMII3TXEN	8mA	CMOS	
GPIOC2	H22	GPIOC2/RGMII3TXD0/RMII3TXD0	8mA	CMOS	
GPIOC3	H23	GPIOC3/RGMII3TXD1/RMII3TXD1	8mA	CMOS	
GPIOC4	G22	GPIOC4/RGMII3TXD2	8mA	CMOS	
GPIOC5	F22	GPIOC5/RGMII3TXD3	8mA	CMOS	
GPIOC6	G23	GPIOC6/RGMII3RXCK/RMII3RCLKI	8mA	CMOS	
GPIOC7	G24	GPIOC7/RGMII3RXCTL	8mA	CMOS	
GPIOD0	F23	GPIOD0/RGMII3RXD0/RMII3RXD0	8mA	CMOS	
GPIOD1	F26	GPIOD1/RGMII3RXD1/RMII3RXD1	8mA	CMOS	
GPIOD2	F25	GPIOD2/RGMII3RXD2/RMII3CRSDV	8mA	CMOS	
GPIOD3	E26	GPIOD3/RGMII3RXD3/RMII3RXER	8mA	CMOS	
GPIOD4	F24	GPIOD4/NCTS3/RGMII4TXCK/RMII4RCLKO	8mA	CMOS	
GPIOD5	E23	GPIOD5/NDCD3/RGMII4TXCTL/RMII4TXEN	8mA	CMOS	
GPIOD6	E24	GPIOD6/NDSR3/RGMII4TXD0/RMII4TXD0	8mA	CMOS	
GPIOD7	E25	GPIOD7/NRI3/RGMII4TXD1/RMII4TXD1	8mA	CMOS	
GPIOE0	D26	GPIOE0/NDTR3/RGMII4TXD2	8mA	CMOS	
GPIOE1	D24	GPIOE1/NRTS3/RGMII4TXD3	8mA	CMOS	
GPIOE2	C25	GPIOE2/NCTS4/RGMII4RXCK/RMII4RCLKI	8mA	CMOS	
GPIOE3	C26	GPIOE3/NDCD4/RGMII4RXCTL	8mA	CMOS	
GPIOE4	C24	GPIOE4/NDSR4/RGMII4RXD0/RMII4RXD0	8mA	CMOS	
GPIOE5	B26	GPIOE5/NRI4/RGMII4RXD1/RMII4RXD1	8mA	CMOS	

to next page

from previous page

GPIOE6	B25	GPIOE6/NDTR4/RGMII4RXD2/RMII4CRSDV	8mA	CMOS	
GPIOE7	B24	GPIOE7/NRTS4/RGMII4RXD3/RMII4RXER	8mA	CMOS	
GPIOF0	D22	GPIOF0/SD1CLK/PWM8	4/8/12/16mA	CMOS	
GPIOF1	E22	GPIOF1/SD1CMD/PWM9	4/8/12/16mA	CMOS	
GPIOF2	D23	GPIOF2/SD1DAT0/PWM10	4/8/12/16mA	CMOS	
GPIOF3	C23	GPIOF3/SD1DAT1/PWM11	4/8/12/16mA	CMOS	
GPIOF4	C22	GPIOF4/SD1DAT2/PWM12	4/8/12/16mA	CMOS	
GPIOF5	A25	GPIOF5/SD1DAT3/PWM13	4/8/12/16mA	CMOS	
GPIOF6	A24	GPIOF6/SD1CD#/PWM14	4/8/12/16mA	CMOS	
GPIOF7	A23	GPIOF7/SD1WP#/PWM15	4/8/12/16mA	CMOS	
GPIOG0	E21	GPIOG0/TXD6/SD2CLK/SALT9	4/8/12/16mA	CMOS	
GPIOG1	B22	GPIOG1/RXD6/SD2CMD/SALT10	4/8/12/16mA	CMOS	
GPIOG2	C21	GPIOG2/TXD7/SD2DAT0/SALT11	4/8/12/16mA	CMOS	
GPIOG3	A22	GPIOG3/RXD7/SD2DAT1/SALT12	4/8/12/16mA	CMOS	
GPIOG4	A21	GPIOG4/TXD8/SD2DAT2/SALT13	4/8/12/16mA	CMOS	
GPIOG5	E20	GPIOG5/RXD8/SD2DAT3/SALT14	4/8/12/16mA	CMOS	
GPIOG6	D21	GPIOG6/TXD9/SD2CD#/SALT15	4/8/12/16mA	CMOS	
GPIOG7	B21	GPIOG7/RXD9/SD2WP#/SALT16	4/8/12/16mA	CMOS	
GPIOH0	A18	GPIOH0/SGPM1CK	8mA	CMOS	PD
GPIOH1	B18	GPIOH1/SGPM1LD	8mA	CMOS	PD
GPIOH2	C18	GPIOH2/SGPM1O	8mA	CMOS	PD
GPIOH3	A17	GPIOH3/SGPM1I	8mA	CMOS	PD
GPIOH4	D18	GPIOH4/SGPS1CK/SCL15	8mA	Schmitt	
GPIOH5	B17	GPIOH5/SGPS1LD/SDA15	8mA	Schmitt	
GPIOH6	C17	GPIOH6/SGPS1I0/SCL16	8mA	Schmitt	
GPIOH7	E18	GPIOH7/SGPS1I1/SDA16	8mA	Schmitt	
GPIOI0	D17	GPIOI0/MTRSTN2/TXD12	8mA	CMOS	PD
GPIOI1	A16	GPIOI1/MTDI2/RXD12	8mA	CMOS	PD
GPIOI2	E17	GPIOI2/MTCK2/TXD13	8mA	CMOS	PD
GPIOI3	D16	GPIOI3/MTMS2/RXD13	8mA	CMOS	PD
GPIOI4	C16	GPIOI4/MTDO2	8mA	CMOS	PD
GPIOI5	E16	GPIOI5/SIOPBO#	8mA	CMOS	PD
GPIOI6	B16	GPIOI6/SIOPBI#	8mA	CMOS	PD
GPIOI7	A15	GPIOI7/SIOSCI#	8mA	CMOS	PD
GPIOJ0	B20	GPIOJ0/HVI3C3SCL/SCL1	12mA	Schmitt	PU
GPIOJ1	A20	GPIOJ1/HVI3C3SDA/SDA1	12mA	Schmitt	PU
GPIOJ2	E19	GPIOJ2/HVI3C4SCL/SCL2	12mA	Schmitt	PU
GPIOJ3	D20	GPIOJ3/HVI3C4SDA/SDA2	12mA	Schmitt	PU
GPIOJ4	C19	GPIOJ4/HVI3C5SCL/SCL3	12mA	Schmitt	PU
GPIOJ5	A19	GPIOJ5/HVI3C5SDA/SDA3	12mA	Schmitt	PU
GPIOJ6	C20	GPIOJ6/HVI3C6SCL/SCL4	12mA	Schmitt	PU
GPIOJ7	D19	GPIOJ7/HVI3C6SDA/SDA4	12mA	Schmitt	PU

to next page

from previous page

GPIOK0	A11	GPIOK0/SCL5	8mA	Schmitt	
GPIOK1	C11	GPIOK1/SDA5	8mA	Schmitt	
GPIOK2	D12	GPIOK2/SCL6	8mA	Schmitt	
GPIOK3	E13	GPIOK3/SDA6	8mA	Schmitt	
GPIOK4	D11	GPIOK4/SCL7	8mA	Schmitt	
GPIOK5	E11	GPIOK5/SDA7	8mA	Schmitt	
GPIOK6	F13	GPIOK6/SCL8	8mA	Schmitt	
GPIOK7	E12	GPIOK7/SDA8	8mA	Schmitt	
GPIOL0	D15	GPIOL0/SCL9	8mA	Schmitt	
GPIOL1	A14	GPIOL1/SDA9	8mA	Schmitt	
GPIOL2	E15	GPIOL2/SCL10	8mA	Schmitt	
GPIOL3	A13	GPIOL3/SDA10	8mA	Schmitt	
GPIOL4	C15	GPIOL4/TXD3	8mA	CMOS	PD
GPIOL5	F15	GPIOL5/RXD3	8mA	CMOS	PD
GPIOL6	B14	GPIOL6/VGAHS	8mA	CMOS	PD
GPIOL7	C14	GPIOL7/VGAVS	8mA	CMOS	PD
GPIOM0	D14	GPIOM0/NCTS1	8mA	CMOS	PD
GPIOM1	B13	GPIOM1/NDCD1	8mA	CMOS	PD
GPIOM2	A12	GPIOM2/NDSR1	8mA	CMOS	PD
GPIOM3	E14	GPIOM3/NRI1	8mA	CMOS	PD
GPIOM4	B12	GPIOM4/NDTR1	8mA	CMOS	PD
GPIOM5	C12	GPIOM5/NRTS1	8mA	CMOS	PD
GPIOM6	C13	GPIOM6/TXD1	8mA	CMOS	PD
GPIOM7	D13	GPIOM7/RXD1	8mA	CMOS	PD
GPION0	P25	GPION0/NCTS2	8mA	CMOS	PD
GPION1	N23	GPION1/NDCD2	8mA	CMOS	PD
GPION2	N25	GPION2/NDSR2	8mA	CMOS	PD
GPION3	N24	GPION3/NRI2	8mA	CMOS	PD
GPION4	P26	GPION4/NDTR2	8mA	CMOS	PD
GPION5	M23	GPION5/NRTS2	8mA	CMOS	PD
GPION6	N26	GPION6/TXD2	8mA	CMOS	PD
GPION7	M26	GPION7/RXD2	8mA	CMOS	PD
GPIOO0	AD26	GPIOO0/PWM0	8mA	CMOS	PD
GPIOO1	AD22	GPIOO1/PWM1	8mA	CMOS	PD
GPIOO2	AD23	GPIOO2/PWM2	8mA	CMOS	PD
GPIOO3	AD24	GPIOO3/PWM3	8mA	CMOS	PD
GPIOO4	AD25	GPIOO4/PWM4	8mA	CMOS	PD
GPIOO5	AC22	GPIOO5/PWM5	8mA	CMOS	PD
GPIOO6	AC24	GPIOO6/PWM6	8mA	CMOS	PD
GPIOO7	AC23	GPIOO7/PWM7	8mA	CMOS	PD
GPIOP0	AB22	GPIOP0/PWM8/THRUIN0	8mA	CMOS	PD
GPIOP1	W24	GPIOP1/PWM9/THRUOUT0	8mA	CMOS	PD

to next page

from previous page

GPIOP2	AA23	GPIOP2/PWM10/THRUIN1	8mA	CMOS	PD
GPIOP3	AA24	GPIOP3/PWM11/THRUOUT1	8mA	CMOS	PD
GPIOP4	W23	GPIOP4/PWM12/THRUIN2	8mA	CMOS	PD
GPIOP5	AB23	GPIOP5/PWM13/THRUOUT2	8mA	CMOS	PD
GPIOP6	AB24	GPIOP6/PWM14	8mA	CMOS	PD
GPIOP7	Y23	GPIOP7/HBLED#/PWM15	8mA	CMOS	PD
GPIOQ0	AA25	GPIOQ0/TACH0	8mA	Schmitt	
GPIOQ1	AB25	GPIOQ1/TACH1	8mA	Schmitt	
GPIOQ2	Y24	GPIOQ2/TACH2	8mA	Schmitt	
GPIOQ3	AB26	GPIOQ3/TACH3	8mA	Schmitt	
GPIOQ4	Y26	GPIOQ4/TACH4	8mA	Schmitt	
GPIOQ5	AC26	GPIOQ5/TACH5	8mA	Schmitt	
GPIOQ6	Y25	GPIOQ6/TACH6	8mA	Schmitt	
GPIOQ7	AA26	GPIOQ7/TACH7	8mA	Schmitt	
GPIOR0	V25	GPIOR0/TACH8	8mA	CMOS	PD
GPIOR1	U24	GPIOR1/TACH9	8mA	CMOS	PD
GPIOR2	V24	GPIOR2/TACH10	8mA	CMOS	PD
GPIOR3	V26	GPIOR3/TACH11	8mA	CMOS	PD
GPIOR4	U25	GPIOR4/TACH12	8mA	CMOS	PD
GPIOR5	T23	GPIOR5/TACH13	8mA	CMOS	PD
GPIOR6	W26	GPIOR6/TACH14	8mA	CMOS	PD
GPIOR7	U26	GPIOR7/TACH15	8mA	CMOS	PD
GPIOS0	R23	GPIOS0/MDC1	8mA	CMOS	PD
GPIOS1	T25	GPIOS1/MDIO1	8mA	CMOS	PD
GPIOS2	T26	GPIOS2/PEWAKE#	8mA	CMOS	PD
GPIOS3	R24	GPIOS3/OSCCLK	8mA	CMOS	PD
GPIOS4	R26	GPIOS4/TXD10	8mA	CMOS	PD
GPIOS5	P24	GPIOS5/RXD10	8mA	CMOS	PD
GPIOS6	P23	GPIOS6/TXD11	8mA	CMOS	PD
GPIOS7	T24	GPIOS7/RXD11	8mA	CMOS	PD
GPIT0	AD20	ADC0/GPIT0		CMOS	PD(P)
GPIT1	AC18	ADC1/GPIT1		CMOS	PD(P)
GPIT2	AE19	ADC2/GPIT2		CMOS	PD(P)
GPIT3	AD19	ADC3/GPIT3		CMOS	PD(P)
GPIT4	AC19	ADC4/GPIT4		CMOS	PD(P)
GPIT5	AB19	ADC5/GPIT5		CMOS	PD(P)
GPIT6	AB18	ADC6/GPIT6		CMOS	PD(P)
GPIT7	AE18	ADC7/GPIT7		CMOS	PD(P)
GPIU0	AB16	ADC8/GPIU0/SALT9		CMOS	PD(P)
GPIU1	AA17	ADC9/GPIU1/SALT10		CMOS	PD(P)
GPIU2	AB17	ADC10/GPIU2/SALT11		CMOS	PD(P)
GPIU3	AE16	ADC11/GPIU3/SALT12		CMOS	PD(P)

to next page

from previous page

GPIU4	AC16	ADC12/GPIU4/SALT13		CMOS	PD(P)
GPIU5	AA16	ADC13/GPIU5/SALT14		CMOS	PD(P)
GPIU6	AD16	ADC14/GPIU6/SALT15		CMOS	PD(P)
GPIU7	AC17	ADC15/GPIU7/SALT16		CMOS	PD(P)
GPIOV0	AB15	GPIOV0/SIOS3#	8mA	CMOS	PD
GPIOV1	AF14	GPIOV1/SIOS5#	8mA	CMOS	PD
GPIOV2	AD14	GPIOV2/SIOPWREQ#	8mA	CMOS	PD
GPIOV3	AC15	GPIOV3/SIOONCTRL#	8mA	CMOS	PD
GPIOV4	AE15	GPIOV4/SIOPWRGD	8mA	CMOS	PD
GPIOV5	AE14	GPIOV5/LPCPD#	8mA	CMOS	PD
GPIOV6	AD15	GPIOV6/LPCPME#	8mA	CMOS	PD
GPIOV7	AF15	GPIOV7/LPCSMI#	8mA	CMOS	PD
GPIOW0	AB7	GPIOW0/LAD0/ESPID0	4/8/12/16mA	CMOS	
GPIOW1	AB8	GPIOW1/LAD1/ESPID1	4/8/12/16mA	CMOS	
GPIOW2	AC8	GPIOW2/LAD2/ESPID2	4/8/12/16mA	CMOS	
GPIOW3	AC7	GPIOW3/LAD3/ESPID3	4/8/12/16mA	CMOS	
GPIOW4	AE7	GPIOW4/LCLK/ESPICK	8mA	CMOS	
GPIOW5	AF7	GPIOW5/LFRAME#/ESPICS#	8mA	CMOS	
GPIOW6	AD7	GPIOW6/LSIRQ#/ESPIALT#	8mA	CMOS	
GPIOW7	AD8	GPIOW7/LPCRST#/ESPIRST#	8mA	CMOS	
GPIOX0	AE8	GPIOX0/SPI2CS0#	8mA	CMOS	PD
GPIOX1	AA9	GPIOX1/SPI2CS1#	8mA	CMOS	PD
GPIOX2	AC9	GPIOX2/SPI2CS2#	8mA	CMOS	PD
GPIOX3	AF8	GPIOX3/SPI2CK	8mA	CMOS	PD
GPIOX4	AB9	GPIOX4/SPI2MOSI	8mA	CMOS	PD
GPIOX5	AD9	GPIOX5/SPI2MISO	8mA	CMOS	PD
GPIOX6	AF9	GPIOX6/SPI2DQ2/TXD12	8mA	CMOS	PD
GPIOX7	AB10	GPIOX7/SPI2DQ3/RXD12	8mA	CMOS	PD
GPIOY0	AF11	GPIOY0/SALT5/WDTRST1#	8mA	CMOS	PD
GPIOY1	AD12	GPIOY1/SALT6/WDTRST2#	8mA	CMOS	PD
GPIOY2	AE11	GPIOY2/SALT7/WDTRST3#	8mA	CMOS	PD
GPIOY3	AA12	GPIOY3/SALT8/WDTRST4#	8mA	CMOS	PD
GPIOY4	AE12	GPIOY4/FWSPIDQ2	8mA	CMOS	PD
GPIOY5	AF12	GPIOY5/FWSPIDQ3	8mA	CMOS	PD
GPIOY6	AC12	GPIOY6/FWSPIABR	8mA	CMOS	PD
GPIOY7	AB12	GPIOY7/FWSPIWP#	8mA	Schmitt	
GPIOZ0	AC10	GPIOZ0/SPI1CS1#	8mA	CMOS	PD
GPIOZ1	AD10	GPIOZ1/SPI1ABR	8mA	CMOS	PD
GPIOZ2	AE10	GPIOZ2/SPI1WP#	8mA	Schmitt	
GPIOZ3	AB11	GPIOZ3/SPI1CK	8mA	CMOS	PD
GPIOZ4	AC11	GPIOZ4/SPI1MOSI	8mA	CMOS	PD
GPIOZ5	AA11	GPIOZ5/SPI1MISO	8mA	CMOS	PD

to next page

from previous page

GPIOZ6	AD11	GPIOZ6/SPI1DQ2/TXD13	8mA	CMOS	PD
GPIOZ7	AF10	GPIOZ7/SPI1DQ3/RXD13	8mA	CMOS	PD
GPIO18A0	C6	GPIO18A0/RGMII1TXCK/RMII1RCLKO	16mA	CMOS18	PD
GPIO18A1	D6	GPIO18A1/RGMII1TXCTL/RMII1TXEN	16mA	CMOS18	PD
GPIO18A2	D5	GPIO18A2/RGMII1TXD0/RMII1TXD0	16mA	CMOS18	PD
GPIO18A3	A3	GPIO18A3/RGMII1TXD1/RMII1TXD1	16mA	CMOS18	PD
GPIO18A4	C5	GPIO18A4/RGMII1TXD2	16mA	CMOS18	PD
GPIO18A5	E6	GPIO18A5/RGMII1TXD3	16mA	CMOS18	PD
GPIO18A6	B3	GPIO18A6/RGMII1RXCK/RMII1RCLKI	16mA	CMOS18	PD
GPIO18A7	A2	GPIO18A7/RGMII1RXCTL	16mA	CMOS18	PD
GPIO18B0	B2	GPIO18B0/RGMII1RXD0/RMII1RXD0	16mA	CMOS18	PD
GPIO18B1	B1	GPIO18B1/RGMII1RXD1/RMII1RXD1	16mA	CMOS18	PD
GPIO18B2	C4	GPIO18B2/RGMII1RXD2/RMII1CRSDV	16mA	CMOS18	PD
GPIO18B3	E5	GPIO18B3/RGMII1RXD3/RMII1RXER	16mA	CMOS18	PD
GPIO18B4	D4	GPIO18B4/RGMII2TXCK/RMII2RCLKO	16mA	CMOS18	PD
GPIO18B5	C2	GPIO18B5/RGMII2TXCTL/RMII2TXEN	16mA	CMOS18	PD
GPIO18B6	C1	GPIO18B6/RGMII2TXD0/RMII2TXD0	16mA	CMOS18	PD
GPIO18B7	D3	GPIO18B7/RGMII2TXD1/RMII2TXD1	16mA	CMOS18	PD
GPIO18C0	E4	GPIO18C0/RGMII2TXD2	16mA	CMOS18	PD
GPIO18C1	F5	GPIO18C1/RGMII2TXD3	16mA	CMOS18	PD
GPIO18C2	D2	GPIO18C2/RGMII2RXCK/RMII2RCLKI	16mA	CMOS18	PD
GPIO18C3	E3	GPIO18C3/RGMII2RXCTL	16mA	CMOS18	PD
GPIO18C4	D1	GPIO18C4/RGMII2RXD0/RMII2RXD0	16mA	CMOS18	PD
GPIO18C5	F4	GPIO18C5/RGMII2RXD1/RMII2RXD1	16mA	CMOS18	PD
GPIO18C6	E2	GPIO18C6/RGMII2RXD2/RMII2CRSDV	16mA	CMOS18	PD
GPIO18C7	E1	GPIO18C7/RGMII2RXD3/RMII2RXER	16mA	CMOS18	PD
GPIO18D0	AB4	GPIO18D0/EMMCCLK	16mA	CMOS18	PD
GPIO18D1	AA4	GPIO18D1/EMMCCMD	16mA	CMOS18	PD
GPIO18D2	AC4	GPIO18D2/EMMCDAT0	16mA	CMOS18	PD
GPIO18D3	AA5	GPIO18D3/EMMCDAT1	16mA	CMOS18	PD
GPIO18D4	Y5	GPIO18D4/EMMCDAT2	16mA	CMOS18	PD
GPIO18D5	AB5	GPIO18D5/EMMCDAT3	16mA	CMOS18	PD
GPIO18D6	AC5	GPIO18D6/EMMCCD#	16mA	CMOS18	PD
GPIO18D7	AB6	GPIO18D7/EMMCWP#	16mA	CMOS18	PD
GPIO18E0	Y1	GPIO18E0/EMMCDAT4/FWSPI18CS#/VBCS#	16mA	CMOS18	PD
GPIO18E1	Y2	GPIO18E1/EMMCDAT5/FWSPI18CK/VBCK	16mA	CMOS18	PD
GPIO18E2	Y3	GPIO18E2/EMMCDAT6/FWSPI18MOSI/VBMOSI	16mA	CMOS18	PD
GPIO18E3	Y4	GPIO18E3/EMMCDAT7/FWSPI18MISO/VBMISO	16mA	CMOS18	PD

2.4.2 GPIO Pass Through

GPIOP groups support maximum 3 sets of pass through pin pair. When enabling the pass through function, the signal from input will directly pass to the output. And the BMC can only read the pin status from GPIO

register. It can not control the output value.

This is useful for system power and reset button control. The pass through pin pairs listed below:

GPIOP0 → GPIOP1
GPIOP2 → GPIOP3
GPIOP4 → GPIOP5

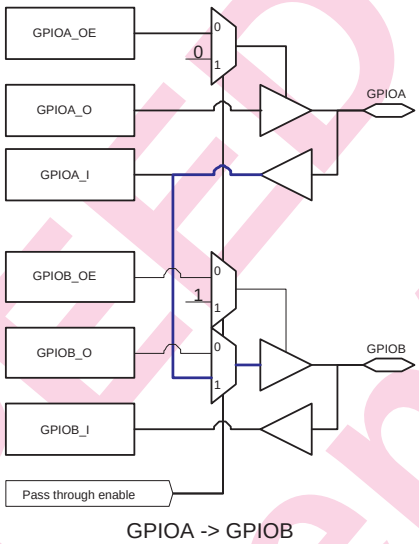


Figure 6: GPIO Pass Through Control

2.5 Serial GPIO Master

- 2 Master supports.
  - First one is 128 Full featured SGPIO: SGPIOA, SGPIOB, SGPIOC, SGPIOD, SGPIOE, SGPIOF, SGPIOG, SGPIOH, SGPIOI, SGPIOJ, GPIOK, GPIOL, GPIOM, GPION, GPIOO, GPIOP
  - Second ont is 80 Full featured SGPIO: SGPIOA, SGPIOB, SGPIOC, SGPIOD, SGPIOE, SGPIOF, SGPIOG, SGPIOH, SGPIOI, SGPIOJ
- Programmable byte length of SGPIO selection
  - 8 bits : SGPIOA
  - 16 bits : SGPIOA, SGPIOB
  - 24 bits : SGPIOA, SGPIOB, SGPIOC
  - 32 bits : SGPIOA, SGPIOB, SGPIOC, SGPIOD
  - 40 bits : SGPIOA, SGPIOB, SGPIOC, SGPIOD, SGPIOE
  - 48 bits : SGPIOA, SGPIOB, SGPIOC, SGPIOD, SGPIOE, SGPIOF
  - 56 bits : SGPIOA, SGPIOB, SGPIOC, SGPIOD, SGPIOE, SGPIOF, SGPIOG
  - 64 bits : SGPIOA, SGPIOB, SGPIOC, SGPIOD, SGPIOE, SGPIOF, SGPIOG, SGPIOH
  - 72 bits : SGPIOA, SGPIOB, SGPIOC, SGPIOD, SGPIOE, SGPIOF, SGPIOG, SGPIOH, SGPIOI
  - 80 bits : SGPIOA, SGPIOB, SGPIOC, SGPIOD, SGPIOE, SGPIOF, SGPIOG, SGPIOH, SGPIOI, SGPIOJ
  - 88 bits : SGPIOA, SGPIOB, SGPIOC, SGPIOD, SGPIOE, SGPIOF, SGPIOG, SGPIOH, SGPIOI, SGPIOJ, GPIOK

- 96 bits : SGPIOA, SGPIOB, SGPIOC, SGPIOD, SGPIOE, SGPIOF, SGPIOG, SGPIOH, SGPIOI, SGPIOJ, GPIOK, GPIOL
- 104 bits : SGPIOA, SGPIOB, SGPIOC, SGPIOD, SGPIOE, SGPIOF, SGPIOG, SGPIOH, SGPIOI, SGPIOJ, GPIOK, GPIOL, GPIOM
- 112 bits : SGPIOA, SGPIOB, SGPIOC, SGPIOD, SGPIOE, SGPIOF, SGPIOG, SGPIOH, SGPIOI, SGPIOJ, GPIOK, GPIOL, GPIOM, GPION
- 120 bits : SGPIOA, SGPIOB, SGPIOC, SGPIOD, SGPIOE, SGPIOF, SGPIOG, SGPIOH, SGPIOI, SGPIOJ, GPIOK, GPIOL, GPIOM, GPION, GPIOO
- 128 bits : SGPIOA, SGPIOB, SGPIOC, SGPIOD, SGPIOE, SGPIOF, SGPIOG, SGPIOH, SGPIOI, SGPIOJ, GPIOK, GPIOL, GPIOM, GPION, GPIOO, GPIOP

- Input interrupt with sensitive high/low level trigger, rising/falling edge trigger mode
- Watchdog reset tolerance

2.5.1 Timing Waveform

Below shows some operating waveform based on the 74165/74595 TTL shift register.

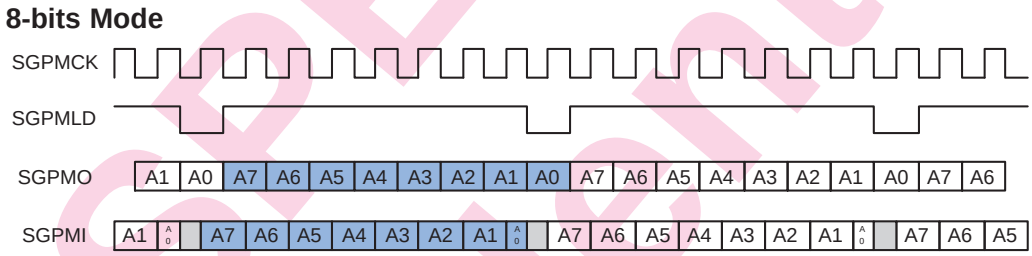


Figure 7: 8 Bits SGPIO Waveform

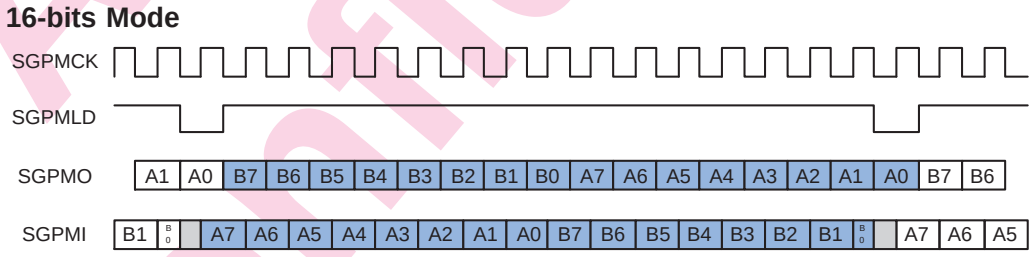


Figure 8: 16 Bits SGPIO Waveform

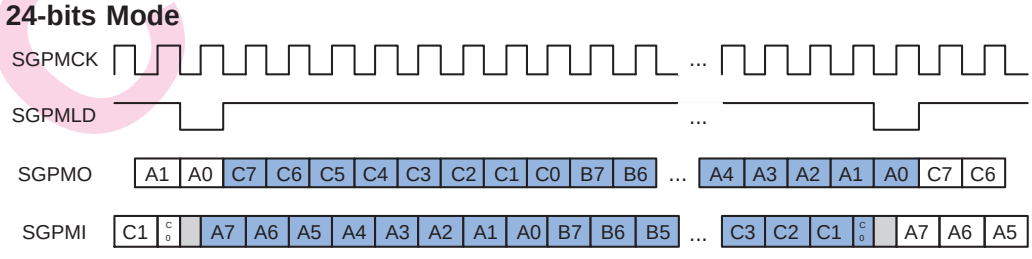


Figure 9: 24 Bits SGPIO Waveform

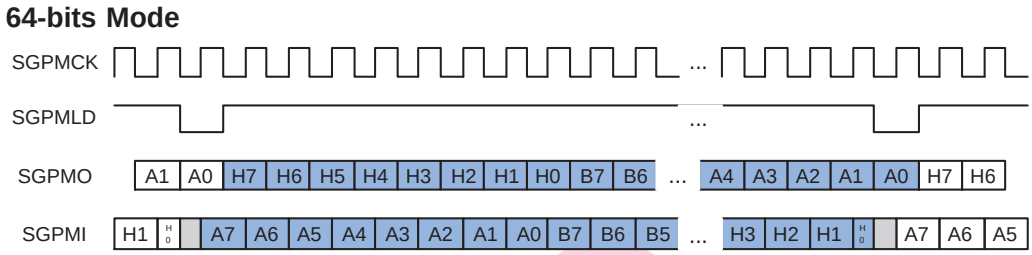


Figure 10: 64 Bits SGPIO Waveform

2.5.2 Example Verilog Code for SGPIO Shifter

```
module SGPIO_S2P(SGPMCK, SGPMLD, SGPMO, SGPMI, Parallel_In, Parallel_Out);

parameter  NUM = 8;

input                 SGPMCK;
input                 SGPMLD;
input                 SGPMO;
output                SGPMI;

input      [NUM-1:0]  Parallel_In;
output     [NUM-1:0]  Parallel_Out;
reg        [NUM-1:0]  Parallel_Out;

reg        [NUM-1:0]  p2s;
reg        [NUM-1:0]  s2p;
wire                  SGPMI;

always@(posedge SGPMCK)
begin
    if(!SGPMLD)
      p2s <= Parallel_In;
    else begin
      p2s <= {p2s[NUM-2:0], 1'b0};
    end
end

assign SGPMI = p2s[NUM-1];

always@(posedge SGPMCK)
begin
    s2p <= {s2p[NUM-2:0], SGPMO};
end
always@(posedge SGPMLD)
begin
    Parallel_Out <= s2p;
end
endmodule
```

2.6 Serial GPIO Slave Monitor

- Slave Serial GPIO monitors SGPIO bus between Initiator and Target that follows SFF-8485/8489.
- Support 2 sets of Slave SGPIO interfaces.
- Each set of SGPIO Slave support 2 channels monitor input
- Each set of SGPIO Slave support maximum 32 drives recording capability for each channel

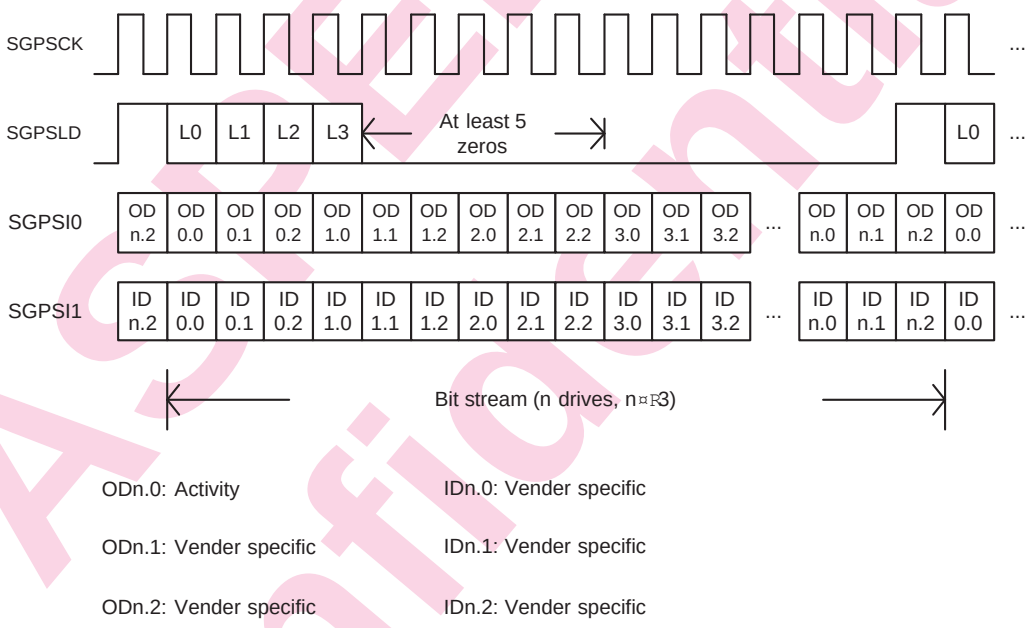


Figure 11: Slave SGPIO Waveform

3 Electrical Specifications

3.1 Absolute Maximum Ratings

PARAMETER	SYMBOL	MIN	TYP	MAX	UNIT
1.0V Core, PLL, PCIe and DP power	IV10D DPLLAVDD PLLAVDD PLLDVDD PE_VCC10A RC_VCC10A DP_VCC10A	GND-0.3		1.1	V
1.2V Core power	IV12D	GND-0.3		1.26	V
DDR4 power	MVDD MVDD_CK	GND-0.3		1.26	V
I3C/FSI power	I3CVDDL1 I3CVDDL2	GND-0.3		1.26	V
MAC3 & MAC4 RGMII/RMII/NCSI power	RVDD	GND-0.3		3.6	V
3.3V I/O, ADC/DAC, PLL and USB power	ADCAV33 DACAV33 BATVDD PV33D PV33D_RGM USB2AV33	GND-0.3		3.6	V
1.2V PLL power	PLLAHVDD	GND-0.3		1.26	V
1.8V I/O, PLL, USB, PCIe and DP power	PV18D PV18D_RGM PV18D_SLI AVDDPLL USB2AV18 PE_VCC18A RC_VCC18A DP_VCC18A	GND-0.3		1.98	V
1.8V/3.3V LPC/eSPI, SD and I3C power	LPVDD SD1VDD SD2VDD I3CVDDH	GND-0.3		3.6	V
3.3V CMOS IO input range		GND-0.3		3.6	V
1.8V CMOS IO input range		GND-0.3		1.98	V
MAC3 & MAC4 IO input range		GND-0.3		3.6	V
DDR IO input range		GND-0.3		MVDD+0.3	V
PECI IO input range		GND-0.3		PECIVDD+0.15	V
USB IO input range		GND-0.3		3.6	V
DAC IO input range		GND-0.3		3.6	V
ADC IO input range		GND-0.3		3.6	V
Storage temperature	TSTG	-40		125	^oC

3.2 Recommended Operating Conditions

PARAMETER	SYMBOL	MIN	TYP	MAX	UNIT
1.0V Core, PLL, PCIe and DP power	IV10D DPLLAVDD PLLAVDD PLLDVDD PE_VCC10A RC_VCC10A DP_VCC10A	0.95	1.0	1.05	V
PECI supply power	PECIVDD	0.7		1.26	V
DDR4 power	MVDD MVDD_CK	1.14	1.2	1.26	V
I3C/FSI power	I3CVDDL1 I3CVDDL2	0.95	1.0	1.05	V
MAC3 & MAC4 RGMII/RMII power	RVDD	1.71		3.465	V
3.3V I/O, ADC/DAC, PLL and USB power	ADCAV33 DACAV33 PV33D PV33D_RGM USB2AV33	3.135	3.3	3.465	V
1.2V PLL power	PLLAHVDD	1.14	1.2	1.26	V
1.8V I/O, PLL, USB, PCIe and DP power	PV18D PV18D_RGM PV18D_SLI AVDDPLL USB2AV18 PE_VCC18A RC_VCC18A DP_VCC18A	1.71	1.8	1.89	V
Adjustable I/O power - 1.8V	LPVDD RVDD SD1VDD SD2VDD I3CVDDH	1.71	1.8	1.89	V
Adjustable I/O power - 3.3V	LPVDD RVDD SD1VDD SD2VDD I3CVDDH	3.135	3.3	3.465	V
Battery Power	BATVDD	1.0		3.465	V
AC Power Noise (PLLAHVDD) (others)	1.0V 1.2V 1.2V 1.8V 2.5V 3.3V			80 100 100 100 150 200	mV_{pp}
Ambient operation temperature	T_A	-40		85	^oC
OTP programming temperature	T_OTP	0		85	^oC
Chip top surface temperature	T_C	-40		90	^oC

to next page

from previous page

Note :
The validation of operation temperature -40C is on-going on A1

3.3 Input Overshoot/Undershoot Tolerance

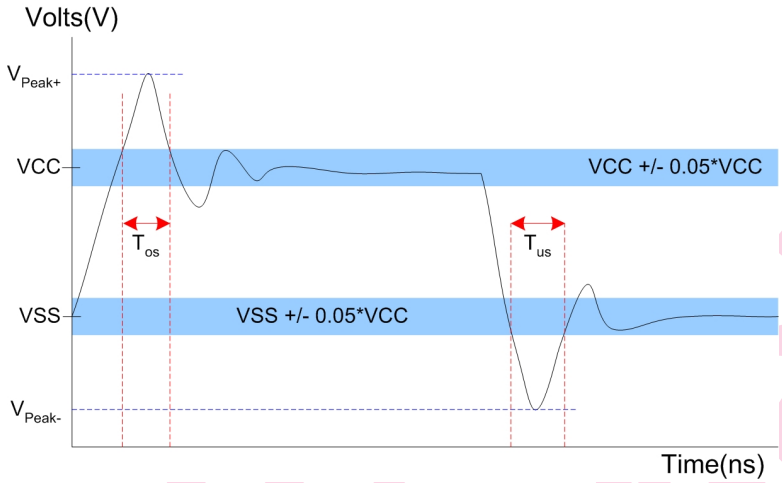


Figure 12: Overshoot/Undershoot

PARAMETER	SYMBOL	MIN	TYP	MAX	UNIT
Overshoot peak voltage	V_{Peak+}			min(1.3*VCC, 4)	V
Overshoot period	T_{OS}			5	ns
Undershoot peak voltage	V_{Peak-}			VSS - 0.3*VCC	V
Undershoot period	T_{US}			5	ns
Note : The peak overshoot voltage can not exceed 4.0V.					

3.4 Input Ringback Tolerance

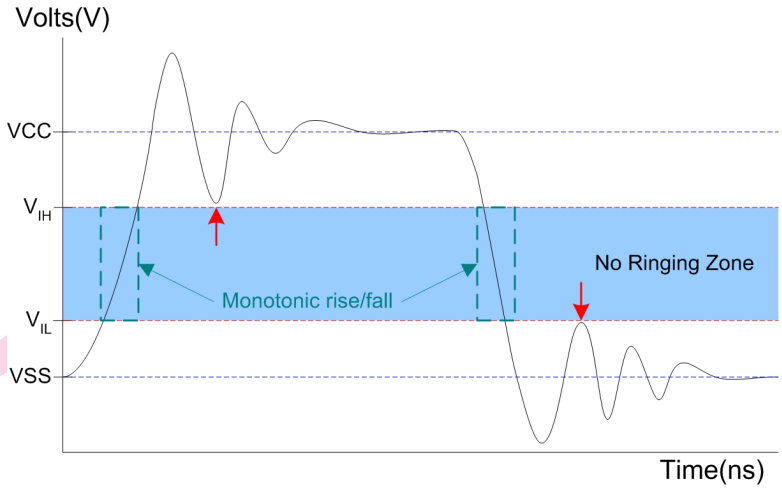


Figure 13: Ringback

- The input signal rise/fall between V_{IH} and V_{IL} should be monotonic style of rise/fall.

- The ringback from overshoot should not lower than V_{IH}.
- The ringback from undershoot should not higher than V_{IL}.

3.5 ESD Capability

- ESD Model: CDM
  - Based on JEDEC22-C101-E standard
  - Pass the JEDEC Class II ($\pm 250V$) classification
- ESD Model: HBM
  - Based on JEDEC-JS-001-2010 standard
  - Pass the JEDEC Class 1C ($\pm 1KV$) classification

3.6 Power Consumption

The power was measured based on the following hardware configurations:

- (DDR4) 1GB DDR4 x1 SDRAM: Micron MT40A1G16RC
- 32MB SPI Flash x1: Macronix MX25L25635F or Winbond W25Q256JVFIQ
- ARM frequency: 1200 MHz
- AHB frequency: 200 MHz
- APB frequency: 100 MHz
- DDR frequency: DDR4-1600 (800 MHz Clock Rate)
- Ambient Temperature: Room temperature 25 degree C

The power was measured based on the following conditions:

- **COND1**: Maximum Condition Power (ARM CPU 1.2GHz)
  1. Windows 1920x1080x32bpp@60Hz + 2D Bltblt stress
  2. Firmware run R-KVM + Iperf LAN stress on LAN2 and LAN3
  3. No other digital IO swing, except LAN2 and LAN3 (RGMII 125MHz)
  4. ARM CPU stress tool for utilization 100
- **COND2**: Normal Condition Power (ARM CPU 1.2GHz)
  1. Windows 1920x1080x32bpp@60Hz + 2D Bltblt stress
  2. Firmware run Iperf LAN stress on LAN3
  3. No other digital IO swing, except LAN3 (RGMII 125MHz)
- **COND3**: Minimum Condition Power (ARM CPU 1.2GHz)
  1. VGA OFF
  2. Firmware Idle
  3. No any digital IO swing
- **COND4**: Initial Condition Power (ARM CPU 1.2GHz)
  1. VGA OFF
  2. Firmware booting
  3. No other digital IO swing, except firmware SPI

- **COND5**: Standby Condition Power (ARM CPU 1.2GHz)
  1. VGA OFF
  2. Firmware run Iperf LAN stress on LAN3
  3. No other digital IO swing, except LAN3 (RGMII 125MHz)

Power rail	COND1		COND2		COND3		COND4		COND5	
(units: mA)	mean	peak	mean	peak	mean	peak	mean	peak	mean	peak
Core 1.0V power	580	710	568	677	332	556	395	652	493	672
DDR4 1.20V power *(Include DRAM)*	592	722	587	656	408	580	443	659	482	639
1.8V power *All 1.8V domain*	70	96	57	76	38	54	38	65	38	62
3.3V power *All 3.3V domain*	91	116	87	111	15	29	18	37	32	60
Chip total power *(DDR4, Watt)*	1.72	2.13	1.66	1.97	0.94	1.44	1.05	1.69	1.25	1.75

3.7 Power Up Sequence

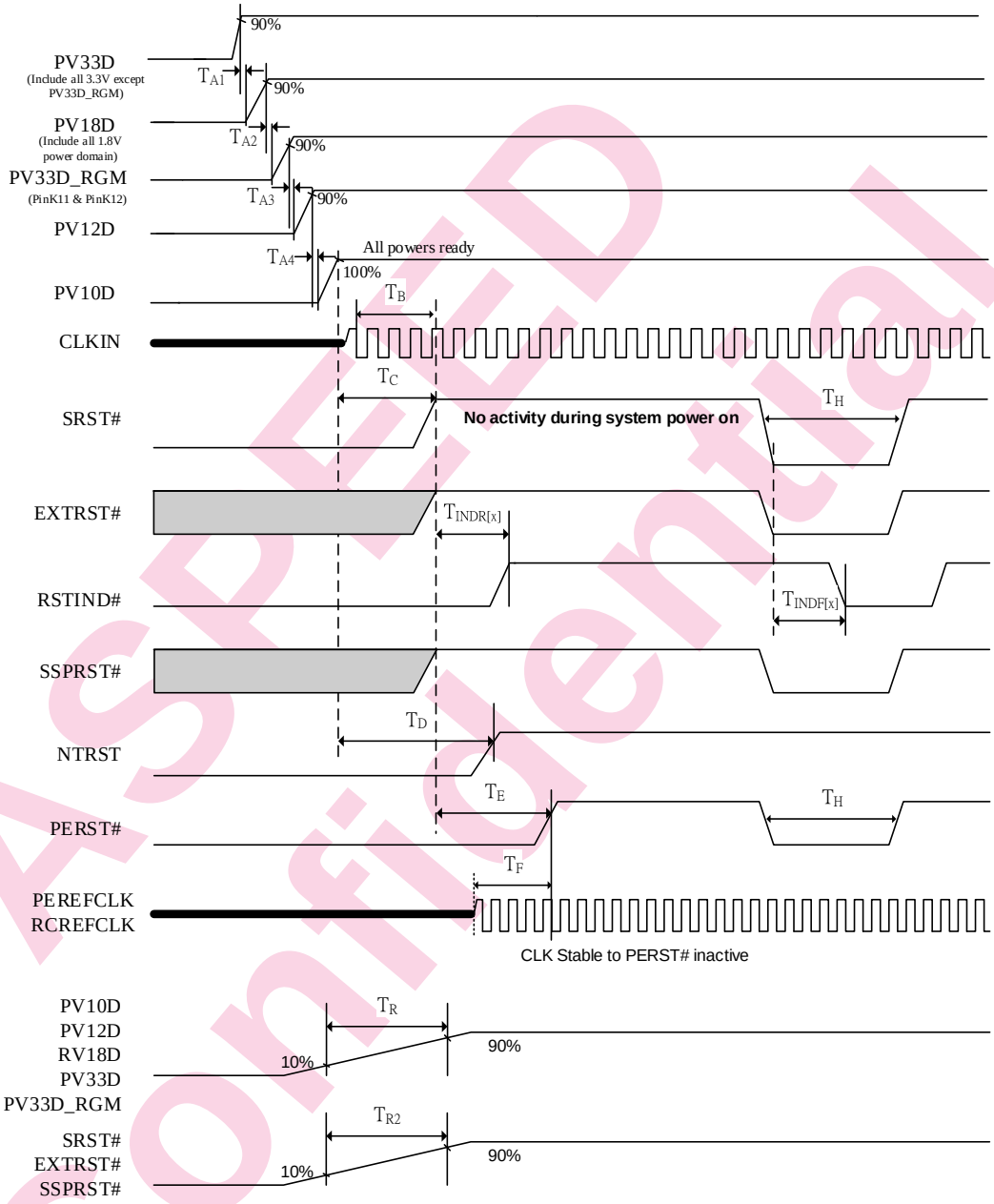


Figure 14: Power-up sequence

Power-up Sequence Parameters

PARAMETER	SYMBOL	MIN	TYP	MAX	UNIT
PV33D to PV18D	T_{A1}	0			ms
PV18D to PV33D_RGM	T_{A2}	20			us
PV33D_RGM to PV12D	T_{A3}	0			ms
PV12D to PV10D	T_{A4}	0			ms
CLKIN stable to SRST# inactive	T_B	500			us
Power stable to SRST# inactive	T_C	1			ms
Power stable to JTAG NTRST inactive	T_D	1			ms
SRST# to PCI-E reset	T_E	50			ms
PCI-E clock stable to reset inactive	T_F	1			ms
Minimum valid reset pulse width	T_H	10			us
Power ramp up time	T_R	10			us
All reset inputs rise time	T_{R2}			300	ns
SRST# inactive to RSTIND# inactive	T_{INDR1}		2		ms
EXTRST# inactive to RSTIND# inactive	T_{INDR2}		2		us
Watchdog reset to RSTIND# inactive	T_{INDR3}		12		us
SRST# active to RSTIND# active	T_{INDF1}		0		us
EXTRST# active to RSTIND# active	T_{INDF2}		130		ns
Watchdog reset to RSTIND# active	T_{INDF3}		0		us

Note :
1. The maximum rise time 300ns for SRST#/EXTRST# is required for the AST2600 to boot properly.
2. There is additional power sequence requrirement for PE_VCC18A and RC_VCC18A on AP Note #17. It is nice to have but NOT must be, please refer to AP Note #17 for more detail.
3. For SSPRST#, the sequnece requirement can be ignored becasue this pin will be used as output function for AST26s0 PCIe Root Complex.
4. For NTRST, the sequnece requirement is defined when external JTAG ICE Debugger is used.

Notes:

1. There is no power-up sequence requirement for PECIVDD, it is connected to the CPU V_{TT} power.
2. PV33D include USB2AV33,ADCAV33, DACAV33, RVDD (3.3V mode), SD1VDD, SD2VDD, I3CVDDH (3.3V mode), LPVDD (3.3V mode) and PV33D.
3. PV18D include USB2AV18, RC_VCC18A, PE_VCC18A, DP_VCC18A, AVDDPLL, RVDD (1.8V mode), I3CVDDH (1.8V mode), LPVDD (1.8V mode), PV18D_SLI, PV18D and PV18D_RGM.
4. PV33D_RGM only include PinK11 and PinK12.
5. PV12D include MVDD, MVDD_CK, I3CVDDL1 (1.2V mode), I3CVDDL2 (1.2V mode), IV12D, PLLAHVDD.
6. PV10D include DPLLAVDD, PLLAVDD, PLLDVDD, RC_VCC10A, PE_VCC10A, DP_VCC10A, I3CVDDL1 (1.0V mode), I3CVDDL2 (1.0V mode) and IV10D.

3.8 Power Down Sequence

Below timing is a reference for safe power down sequence.

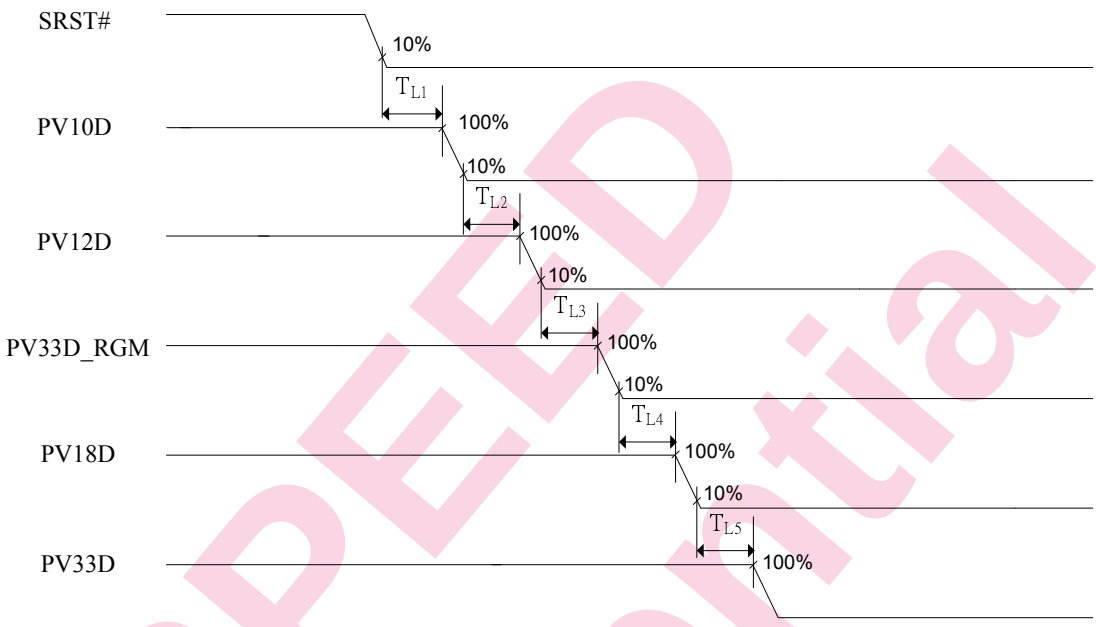


Figure 15: Power-down sequence

Power-down Sequence Parameters

PARAMETER	SYMBOL	MIN	TYP	MAX	UNIT
SRST# or EXTRST# (either one or both) active to PV10D power drop delay.	T_{L1}	0			ms
PV10D to PV12D	T_{L2}	0			ms
PV12D to PV33D_RGM	T_{L3}	0			ms
PV33D_RGM to PV18D	T_{L4}	20			us
PV18D to PV33D	T_{L5}	0			ms

3.9 Power Down Sequence for Firmware

§ The power down sequence is only defined for firmware to be stopped safely.
§ For chip hardware power down sequence, please refer to previous section.

Since firmware may have flash erase or program operation at run time, for a safe power down, all of the flash erase and program operation should be stopped and finished before power start to drop.

It ever happened that flash content (at random address) was destructed if update process is on-going and suddenly power dropped. So firmware and system design should do something to avoid this case to happen. Below are 2 ways that is helpful to avoid this issue.

- Power supply should have a certain amount of capacitor capacity to supply BMC to work for a while after AC loss, and BMC should can detect the AC loss event at the first time and then stop or finish the on-going flash update process immediately.
- Use another flash part for firmware to store information at run time. So no frequent update operation on firmware flash part is executed at run time.

Below timing is a reference for safe power down sequence.

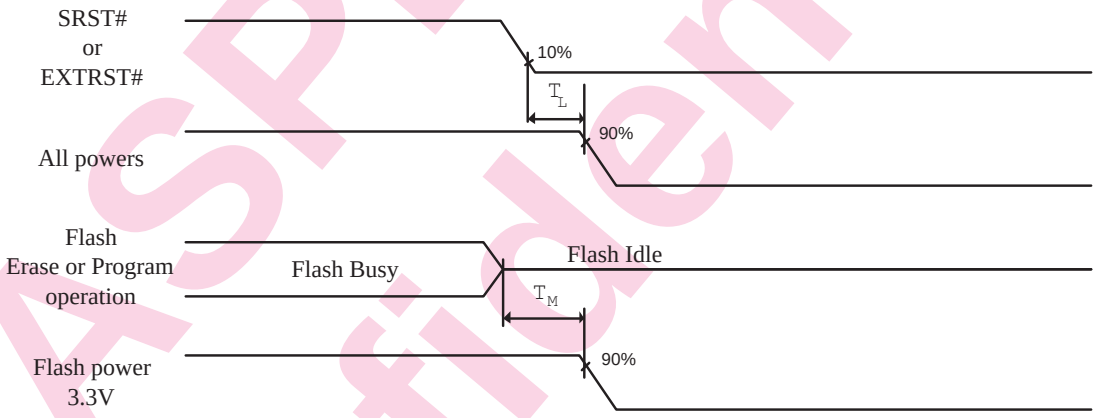


Figure 16: Power-down sequence for Firmware

Power-down Sequence Parameters

PARAMETER	SYMBOL	MIN	TYP	MAX	UNIT
SRST# or EXTRST# (either one or both) active to power drop delay. Activate reset is to make sure the firmware can be terminated before power start to drop, where unstable power may cause firmware run abnormally.	T_L	0			ms
Flash enters idle state before power start to drop.	T_M	0			ms

3.10 I/O DC Electrical Specification

3.10.1 3.3V CMOS I/O Parameters

PARAMETER	SYMBOL	MIN	TYP	MAX	UNIT
Input Low Voltage	V_{IL}	-0.3		0.8	V
Input High Voltage	V_{IH}	2.0		3.6	V
Output Low Voltage @ I_{OL}(min)	V_{OL}			0.4	V
Output High Voltage @ I_{OH}(min)	V_{OH}	2.4			V
Threshold Point	V_T	1.44	1.57	1.73	V
Threshold Point with PU Resistor Enabled	V_Tpu	1.42	1.56	1.72	V
Threshold Point with PD Resistor Enabled	V_Tpd	1.45	1.59	1.74	V
Schmitt Trig. Low to High Threshold	V_T+	1.6	1.74	1.89	V
Schmitt Trig. High to Low Threshold	V_T-	1.27	1.4	1.56	V
Schmitt Trig. Low to High Threshold with PD Resistor Enabled	$V_T + pd$	1.61	1.75	1.9	V
Schmitt Trig. High to Low Threshold with PD Resistor Enabled	$V_T - pd$	1.27	1.41	1.57	V
Input Leakage Current @ V_I=3.3V or 0V	I_L			$\pm$10	μA
Tri-state Output Leakage Current @ V_O=3.3V or 0V	I_{OZ}			$\pm$10	μA
Pull-up Resistor[1]	R_{PU}	34	51	81	KΩ
Pull-down Resistor	R_{PD}	34	51	84	KΩ
Low Level Output Current - O8 @ V_{OL}=0.4V	I_{OL}	11.8	17.6	23.2	mA
Low Level Output Current - O12 @ V_{OL}=0.4V	I_{OL}	15.8	23.5	31.1	mA
Low Level Output Current - O16 @ V_{OL}=0.4V	I_{OL}	19.8	29.5	38.9	mA
High Level Output Current - O8 @ V_{OH}=2.4V	I_{OH}	17.2	34.1	58.9	mA
High Level Output Current - O12 @ V_{OH}=2.4V	I_{OH}	23.9	47.2	81.5	mA
High Level Output Current - O16 @ V_{OH}=2.4V	I_{OH}	30.5	60.3	104.1	mA

3.10.2 1.8V CMOS I/O Parameters

PARAMETER	SYMBOL	MIN	TYP	MAX	UNIT
Input Low Voltage	V_{IL}	-0.3		0.63	V
Input High Voltage	V_{IH}	1.17		1.98	V
Output Low Voltage @ I_{OL}(min)	V_{OL}			0.45	V
Output High Voltage @ I_{OH}(min)	V_{OH}	1.17			V

to next page

[1]This pull-up only acts on the input path, it will not function normally on the output path, so if need pull-up function externally, add pull-up resistor externally.

from previous page

Threshold Point	V_T	0.82	0.89	0.97	V
Threshold Point with PU Resistor Enabled	V_Tpu	0.81	0.88	0.97	V
Threshold Point with PD Resistor Enabled	V_Tpd	0.82	0.89	0.98	V
Schmitt Trig. Low to High Threshold	V_T+	0.96	1.03	1.1	V
Schmitt Trig. High to Low Threshold	V_T-	0.64	0.75	0.86	V
Schmitt Trig. Low to High Threshold with PU Resistor Enabled	$V_T + pu$	0.95	1.02	1.09	V
Schmitt Trig. High to Low Threshold with PU Resistor Enabled	$V_T - pu$	0.63	0.75	0.85	V
Schmitt Trig. Low to High Threshold with PD Resistor Enabled	$V_T + pd$	0.96	1.05	1.11	V
Schmitt Trig. High to Low Threshold with PD Resistor Enabled	$V_T - pd$	0.65	0.76	0.86	V
Input Leakage Current @ V_I=1.8V or 0V	I_L			±10	μA
Tri-state Output Leakage Current @ V_O=1.8V or 0V	I_{OZ}			±10	μA
Pull-up Resistor	R_{PU}	60	89	137	KΩ
Pull-down Resistor	R_{PD}	61	104	196	KΩ
Low Level Output Current - O4 @ V_{OL}=0.45V	I_{OL}	5.5	9.2	13	mA
Low Level Output Current - O8 @ V_{OL}=0.45V	I_{OL}	11.1	18.2	25.6	mA
Low Level Output Current - O12 @ V_{OL}=0.45V	I_{OL}	16.5	27	37.7	mA
Low Level Output Current - O16 @ V_{OL}=0.45V	I_{OL}	21.9	35.6	49.4	mA
High Level Output Current - O4 @ V_{OH}=1.17V	I_{OH}	6.6	9.6	13.2	mA
High Level Output Current - O8 @ V_{OH}=1.17V	I_{OH}	13.1	19.1	26.2	mA
High Level Output Current - O12 @ V_{OH}=1.17V	I_{OH}	19.5	28.5	39	mA
High Level Output Current - O16 @ V_{OH}=1.17V	I_{OH}	25.9	37.9	51.8	mA

3.10.3 3.3V/1.8V I2C/I3C I/O Parameters (GPIOJ Group)

PARAMETER	SYMBOL	MIN	TYP	MAX	UNIT
Input Low Voltage @ 3.3V mode	V_{IL}	-0.3		0.8	V
Input Low Voltage @ 1.8V mode	V_{IL}	-0.3		0.5	V
Input High Voltage @ 3.3V mode	V_{IH}	2.0		3.6	V
Input High Voltage @ 1.8V mode	V_{IH}	1.3		1.98	V
Output Low Voltage @ I_{OL}(min) @ 3.3V mode	V_{OL}			0.4	V

to next page

from previous page

Output Low Voltage @ I_{OL}(min) @ 1.8V mode	V_{OL}			0.45	V
Output High Voltage @ I_{OH}(min) @ 3.3V mode	V_{OH}	2.4			V
Output High Voltage @ I_{OH}(min) @ 1.8V mode	V_{OH}	1.35			V
Schmitt Trig. Low to High Threshold @ 3.3V mode	V_T+	1.59	1.74	1.9	V
Schmitt Trig. Low to High Threshold @ 1.8V mode	V_T+	0.87	1.05	1.25	V
Schmitt Trig. High to Low Threshold @ 3.3V mode	V_T-	1.25	1.4	1.57	V
Schmitt Trig. High to Low Threshold @ 1.8V mode	V_T-	0.55	0.73	0.9	V
Tri-state Output Leakage Current @ V_O=3.3V or 0V @ 3.3V mode	I_{OZ}			±10	μA
Tri-state Output Leakage Current @ V_O=1.8V or 0V @ 1.8V mode	I_{OZ}			±10	μA
Pull-up Resistor[1] @ 3.3V mode	R_{PU}	34	51	81	KΩ
Pull-up Resistor[1] @ 3.3V mode + I3C application	R_{PU}	1.4	2	2.6	KΩ
Pull-up Resistor @ 1.8V mode	R_{PU}	60	117	220	KΩ
Pull-up Resistor @ 1.8V mode + I3C applicatio	R_{PU}	1.4	2	2.6	KΩ
Low Level Output Current - O12 @ V_{OL}=0.4V @ 3.3V mode	I_{OL}	15.9	23.6	31.2	mA
Low Level Output Current - O16 @ V_{OL}=0.4V @ 3.3V mode	I_{OL}	19.8	29.5	38.9	mA
Low Level Output Current - O12 @ V_{OL}=0.45V @ 1.8V mode	I_{OL}	6	13.6	25.5	mA
Low Level Output Current - O16 @ V_{OL}=0.45V @ 1.8V mode	I_{OL}	7.7	17.2	32	mA
High Level Output Current - O12 @ V_{OH}=2.4V @ 3.3V mode	I_{OH}	30.5	60.3	104.1	mA
High Level Output Current - O16 @ V_{OH}=2.4V @ 3.3V mode	I_{OH}	28.2	54.8	88.8	mA
High Level Output Current - O12 @ V_{OH}=1.17V @ 1.8V mode	I_{OH}	7	13.9	25	mA
High Level Output Current - O16 @ V_{OH}=1.17V @ 1.8V mode	I_{OH}	9	17.7	31.5	mA

3.10.4 MAC3/MAC4/SD1/SD2/LPC/eSPI I/O Parameters

PARAMETER	SYMBOL	MIN	TYP	MAX	UNIT
Input Low Voltage @ 3.3V mode	V_{IL}	-0.3		0.99	V
Input Low Voltage @ 1.8V mode	V_{IL}	-0.3		0.54	V

to next page

from previous page

Input High Voltage @ 3.3V mode	V_{IH}	2.31		3.6	V
Input High Voltage @ 1.8V mode	V_{IH}	1.26		2.1	V
Output Low Voltage @ I_{OL}(min) 3.3V mode	V_{OL}			0.99	V
Output Low Voltage @ I_{OL}(min) 1.8V mode	V_{OL}			0.54	V
Output High Voltage @ I_{OH}(min) 3.3V mode	V_{OH}	2.31			V
Output High Voltage @ I_{OH}(min) 1.8V mode	V_{OH}	1.26			V
Low Level Output Current - O4 @ V_{OL}=0.75V 3.3V mode	I_{OL}	7	13.5	18	mA
Low Level Output Current - O8 @ V_{OL}=0.75V 3.3V mode	I_{OL}	10.5	20	27	mA
Low Level Output Current - O12 @ V_{OL}=0.75V 3.3V mode	I_{OL}	14	27	36	mA
Low Level Output Current - O16 @ V_{OL}=0.75V 3.3V mode	I_{OL}	21	40	54	mA
Low Level Output Current - O4 @ V_{OL}=0.75V 1.8V mode	I_{OL}	4.5	6.5	12	mA
Low Level Output Current - O8 @ V_{OL}=0.75V 1.8V mode	I_{OL}	6.75	9.75	18	mA
Low Level Output Current - O12 @ V_{OL}=0.75V 1.8V mode	I_{OL}	9	13	24	mA
Low Level Output Current - O16 @ V_{OL}=0.75V 1.8V mode	I_{OL}	13.5	19.5	36	mA
High Level Output Current - O4 @ V_{OH}=2.55V 3.3V mode	I_{OH}	7	13.5	20	mA
High Level Output Current - O8 @ V_{OH}=2.55V 3.3V mode	I_{OH}	10.5	20	30	mA
High Level Output Current - O12 @ V_{OH}=2.55V 3.3V mode	I_{OH}	14	27	40	mA
High Level Output Current - O16 @ V_{OH}=2.55V 3.3V mode	I_{OH}	21	40	60	mA
High Level Output Current - O4 @ V_{OH}=1.05V 1.8V mode	I_{OH}	4.5	6.5	12	mA
High Level Output Current - O8 @ V_{OH}=1.05V 1.8V mode	I_{OH}	6.75	9.75	18	mA
High Level Output Current - O12 @ V_{OH}=1.05V 1.8V mode	I_{OH}	9	13	24	mA
High Level Output Current - O16 @ V_{OH}=1.05V 1.8V mode	I_{OH}	13.5	19.5	36	mA

3.10.5 3.3V I/O Parameters for dedicated pins

Including following signals: ENTEST, SRST#, PERST#, SSPRST#, EXTRST#, CLKIN, RSTIND#, DPHPD, RGMIICK, TXD5, RXD5, DDCCLK, DDCDAT, TRST, TCK, TDI, TDO, TMS

PARAMETER	SYMBOL	MIN	TYP	MAX	UNIT
Input Low Voltage	V_{IL}	-0.3		0.8	V
Input High Voltage	V_{IH}	2.0		3.465	V
Output Low Voltage @ I_{OL}(min)	V_{OL}			0.4	V
Output High Voltage @ I_{OH}(min)	V_{OH}	2.4			V
Threshold Point	V_T	1.02	1.18	1.37	V
Threshold Point with PU Resistor Enabled	V_Tpu	1	1.16	1.34	V
Threshold Point with PD Resistor Enabled	V_Tpd	1.02	1.19	1.39	V
Schmitt Trig. Low to High Threshold	V_T+	1.23	1.34	1.5	V
Schmitt Trig. High to Low Threshold	V_T-	0.97	1.13	1.33	V
Schmitt Trig. Low to High Threshold with PD Resistor Enabled	$V_T + pd$	1.23	1.36	1.52	V
Schmitt Trig. High to Low Threshold with PD Resistor Enabled	$V_T - pd$	0.97	1.14	1.34	V
Input Leakage Current @ V_I=3.3V or 0V	I_L			±10	μA
Tri-state Output Leakage Current @ V_O=3.3V or 0V	I_{OZ}			±10	μA
Pull-up Resistor	R_{PU}	26	46	71	KΩ
Pull-down Resistor	R_{PD}	27	48	102	KΩ
Low Level Output Current @ V_{OL}=0.4V	I_{OL}	17.8	27.7	38.5	mA
High Level Output Current @ V_{OH}=2.4V	I_{OH}	27.1	41.8	63.8	mA

3.10.6 DDR4 I/O Parameters

PARAMETER	SYMBOL	MIN	TYP	MAX	UNIT
Supply power voltage	$MVDD$	1.14	1.20	1.26	V
Note : Follow JEDEC JESD79-4 standard specification.					

3.10.7 DAC I/O Parameters

PARAMETER	SYMBOL	MIN	TYP	MAX	UNIT
DACAV33 supply DC current	$\|I\|$		66	132	mA
Power down current	$\|I\|$		30	50	uA
DACVREF output voltage		0.94	1	1.06	V

3.10.8 ADC I/O Parameters

PARAMETER	SYMBOL	MIN	TYP	MAX	UNIT
ADCAV33 supply DC current	$\|I\|$		2.6	3.2	mA
Power down current	$\|I\|$		10	60	uA
Analog input impedance		15			KΩ
Analog input Capacitance				12	pF
Analog VBAT input Voltage on ADC7 and ADC15		0		3.3	V
ADCVREFEXT voltage range		0.9		2.7	V
Resolution			10		Bit
VREFP voltage (ENVR=1 and VRSEL[1:0]=00b)			2.5		V
VREFP voltage (ENVR=0 and VRSEL[1:0]=01b)			1.2		V
VREFN voltage			0		V
Voltage accuracy			±1		%VREFP
Analog input range ADC[15:0]		0		2.5	V
Differential Non-Linearity (DNL)	DNL	-1		1	LSB
Integral Non-Linearity (INL)	INL	-1.5		1.5	LSB
Sampling rate	F_S	10	500		KHz

3.10.9 ADC7 and ADC15 for VBAT sense Parameters

PARAMETER	SYMBOL	MIN	TYP	MAX	UNIT
Operation current @ BMC Power Off	$I_{leak_{pd}}$			1.16	uA
Operation current @ BMC Power On	$I_{leak_{po}}$			0.28	uA

3.10.10 CHASI# I/O Parameters

PARAMETER	SYMBOL	MIN	TYP	MAX	UNIT
Operation current @ BMC Power Off	$I_{leak_{pd}}$			0.29	uA
Operation current @ BMC Power On	$I_{leak_{pd}}$			0.55	uA
Capacitance				2.68	pF
Minimum pulse requirement			7		ns

3.10.11 Battery Backed SRAM Parameters

PARAMETER	SYMBOL	MIN	TYP	MAX	UNIT
BATVDD supply power voltage	$BATVDD$	1.0		3.6	V
Operation current @ BMC Power Off	I_{leak_pd}			0.31	uA
Operation current @ BMC Power On	I_{leak_po}			1.16	uA

3.10.12 USB I/O Parameters

PARAMETER	SYMBOL	MIN	TYP	MAX	UNIT
USB2AV33 supply power voltage		3.135	3.3	3.465	V
Maximum current for USB2AV18 (2 ports)	I_{max}			72	mA
Maximum current for USB2AV33 (2 ports)	I_{max}			50	mA
Idle state current for USB2AV18 (1 port)	I_{idle}			8.2	mA
Idle state current for USB2AV33 (1 port)	I_{idle}			9.4	mA
Suspend state current for USB2AV18 (1 port)	I_{spd}			75	uA
Suspend state current for USB2AV33 (1 port)	I_{spd}			40	uA

3.10.13 PECI I/O Parameters

PARAMETER	SYMBOL	MIN	TYP	MAX	UNIT
PECIVDD supply power voltage	V_{TT}	0.7		1.26	V
Input Low Voltage	V_{IL}	0.275V_{TT}		0.5V_{TT}	V
Input High Voltage	V_{IH}	0.55V_{TT}		0.725V_{TT}	V
Output Low Voltage	V_{OL}		0.25V_{TT}		V
Output High Voltage	V_{OH}		0.75V_{TT}		V
Output Low Current @ 0.25V_{TT}	I_{OL}	0.5		1.51	mA
Output High Current @ 0.75V_{TT}	I_{OH}	-6			mA
Bit time		0.495		500	uS
Capacitance				10	pF

3.10.14 FSI I/O Parameters

PARAMETER	SYMBOL	MIN	TYP	MAX	UNIT
IO Supply Voltage	V_{DD}	1.0	1.2	1.3	V
Input Low Voltage	V_{IL}	-0.3		0.4	V
Input High Voltage	V_{IH}	0.8		1.5	V
Output Low Voltage @ I_{OL}=-2mA	V_{OL}	0		V_{DD}+0.1	V
Output High Voltage @ I_{OH}=2mA	V_{OH}	V_{DD}-0.1		V_{DD}	V
Pull-up Resistor select 0	R_{PU}	0.6	0.75	1.5	KΩ
Pull-down Resistor	R_{PD}	7	10	15	KΩ

3.10.15 Low-voltage I3C I/O Parameters

PARAMETER	SYMBOL	MIN	TYP	MAX	UNIT
IO Supply Voltage	V_{DD}	0.95	1.0	1.25	V
Input Low Voltage	V_{IL}	-0.1*V_{DD}		0.3*V_{DD}	V
Input High Voltage	V_{IH}	0.7*V_{DD}		1.1*V_{DD}	V
Schmitt Input Hysteresis	V_{hys}	30	100	150	mV
Pull-up Resistor select 0	R_{PU}	0.6	0.75	1.5	KΩ
Pull-up Resistor select 1	R_{PU}	1.2	2	2.8	KΩ
Note : For internal 750 ohm pull-up resistor selection, please refer to Errata Item 59 for more detail.					

3.10.16 UART I/O Parameters

PARAMETER	SYMBOL	MIN	TYP	MAX	UNIT
Input Low Voltage	V_{IL}	-0.3		0.8	V
Input High Voltage	V_{IH}	2.0		3.6	V
Output Low Voltage @ I_{OL}(min)	V_{OL}			0.4	V
Output High Voltage @ I_{OH}(min)	V_{OH}	2.4			V

3.11 AC Timing Specification

Below figure defines the timing of Rise/Fall time and Duty cycle in this datasheet.

- Rise/Fall time: ramp time between V_{IH} and V_{IL}.
- Duty cycle: ratio of T_{Duty+}/T_{Duty-}

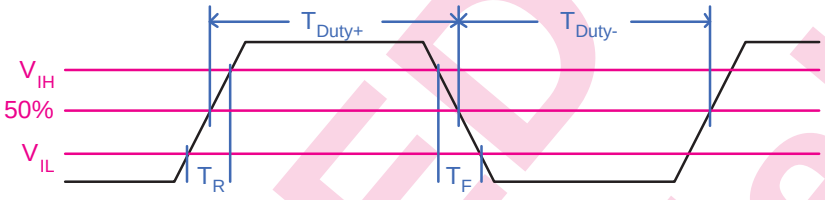


Figure 17: Timing Definition

3.11.1 Reference Clock Input

CLKIN Input Requirements

PARAMETER	SYMBOL	MIN	TYP	MAX	UNIT
Input Low Voltage	V_{IL}	-0.3		0.8	V
Input High Voltage	V_{IH}	2.0		3.6	V
Nominal Frequency			25		MHz
Frequency Stability/Tolerance		-30		+30	ppm
Duty Cycle Ratio		45		55	%
Input Rise time	T_R			5	ns
Input Fall time	T_F			5	ns

Clock name	CLKIN=25MHz	Units
CPUCLK	1200	MHz
HCLK	200	MHz
PCLK	100	MHz
DDRCK	800	MHz

3.11.2 LPC Interface

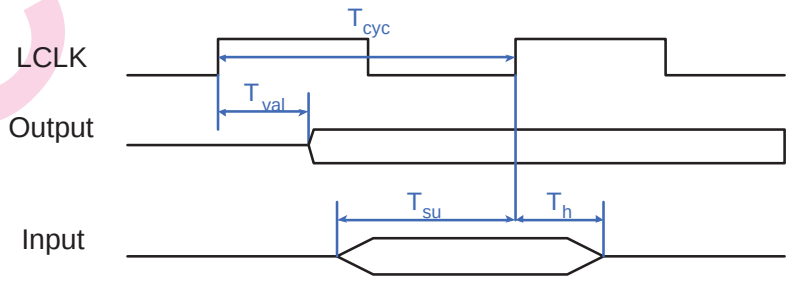


Figure 18: LPC Timing Waveform

LPC 33MHz

PARAMETER	SYMBOL	MIN	TYP	MAX	UNIT
Clock cycle time	T_{cyc}	29		42	ns
Output valid time	T_{val}	2		11	ns
Input setup time	T_{su}	7			ns
Input hold time	T_h	0.5			ns
Input Rise time	T_R			5	ns
Input Fall time	T_F			5	ns

3.11.3 RGMII/RMII/NCSI Interface

AST2600 embedded programmable delay chain of 32 stages for MAC interfaces transmit and receive timing fine-tune. It is suggested to do timing margin analysis during development by using ASPEED utility. (mactest ot ncsitest) And adjust the timing setting in firmware if the best timing point is not match the default setting.

The default timing setting for RMII/NCSI interface was set to the best timing margin point.

The default timing setting with/without PHY RXCLK delay for RGMII interface was set as below:

- Use internal generated 125MHz reference clock.
- When RXCLK delay is disabled on ethernet PHY setting.
- TX path : center-aligned output
- RX path : edge-aligned input

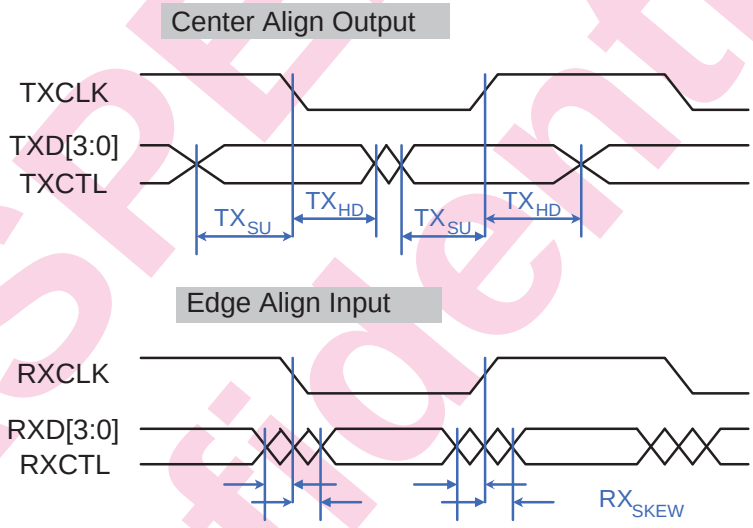


Figure 19: RGMII Timing Waveform

RGMII Timing without PHY RXCLK delay mode

PARAMETER	SYMBOL	MIN	TYP	MAX	UNIT
Transmit/Receive Clock frequency (1G)	TCK_{CYC}		125		MHz
Transmit/Receive Clock frequency (100M)	TCK_{CYC}		25		MHz
Transmit/Receive Clock frequency (10M)	TCK_{CYC}		2.5		MHz
Transmit/Receive Clock duty cycle	TCK_{Duty}	45		55	%
Data output setup time	TX_{SU}	1			ns
Data output hold time	TX_{HD}	1			ns
Data input skew time	RX_{SKEW}	-1		1	ns
Input Rise time (RVDD=3.3V)	T_R			2.0	ns
Input Fall time (RVDD=3.3V)	T_F			2.0	ns
Input Rise time (RVDD=1.8V)	T_R			1.5	ns
Input Fall time (RVDD=1.8V)	T_F			1.5	ns

- Use internal generated 125MHz reference clock.
- When RXCLK delay (2ns) is enabled on ethernet PHY setting.
- TX path : center-aligned output
- RX path : center-aligned input

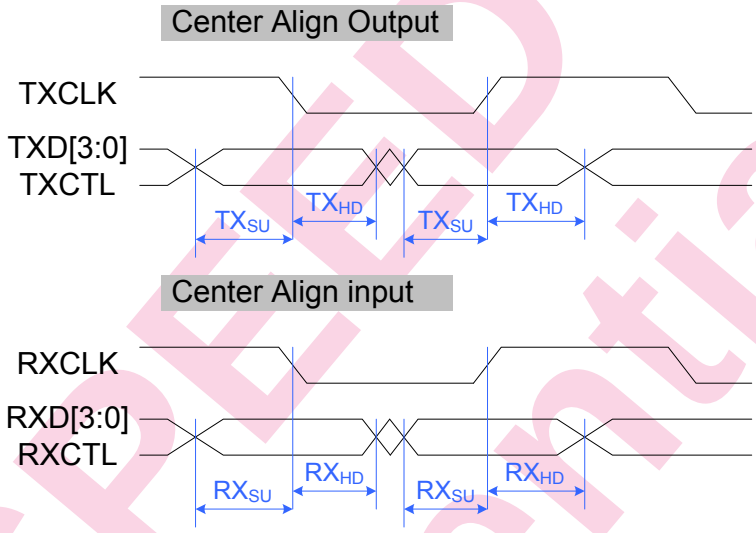


Figure 20: RGMII Timing Waveform with RXCLK Delay

RGMII Timing with PHY RXCLK delay mode

PARAMETER	SYMBOL	MIN	TYP	MAX	UNIT
Transmit/Receive Clock frequency (1G)	TCK_{CYC}		125		MHz
Transmit/Receive Clock frequency (100M)	TCK_{CYC}		25		MHz
Transmit/Receive Clock frequency (10M)	TCK_{CYC}		2.5		MHz
Transmit/Receive Clock duty cycle	TCK_{Duty}	45		55	%
Data output setup time	TX_{SU}	1			ns
Data output hold time	TX_{HD}	1			ns
Data input setup time	RX_{SU}	1			ns
Data input hold time	RX_{HD}	1			ns
Input Rise time (RVDD=3.3V)	T_R			2.0	ns
Input Fall time (RVDD=3.3V)	T_F			2.0	ns
Input Rise time (RVDD=1.8V)	T_R			1.5	ns
Input Fall time (RVDD=1.8V)	T_F			1.5	ns

Note :
Please make sure target solution (such as Ethernet PHY, SwitchKetc) support RX delay capability and the delay capability is larger than 1.2ns when MAC1/MAC2 is used. Please refer to AST2600 AP Note-05 for more detail.

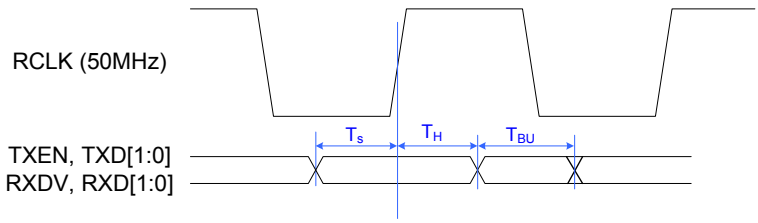


Figure 21: RMII/NCSI Timing Waveform

RMII/NCSI Timing

PARAMETER	SYMBOL	MIN	TYP	MAX	UNIT
Reference Clock cycle time	TCK_{CYC}		20		ns
Reference Clock duty cycle	TCK_{Duty}	35		65	%
Input / Output setup time	T_S	4			ns
Input / Output hold time	T_H	2			ns
Input / Output bus unstable time	T_{BU}			8	ns
Input Rise time	T_R			3.5	ns
Input Fall time	T_F			3.5	ns
NCSI 50MHz clock output frequency stability from PinF24/PinH24		-100		100	ppm
NCSI Clock Rising to Data Output	T_{CO}	7		12.5	ns

Note :
For NCSI Tco timing, it can be adjusted by internal delay cell setting on SCU350. The minimum 7ns is defined by zero internal delay condition.

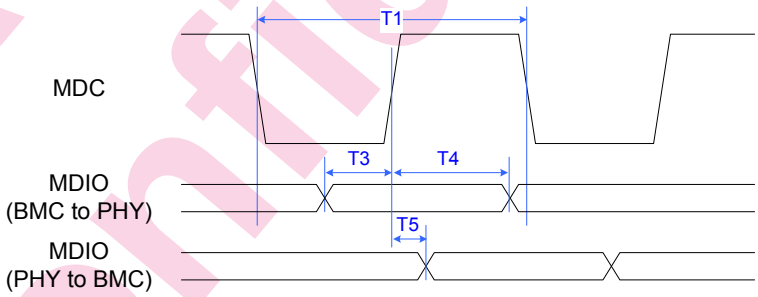


Figure 22: MDC/MDIO Timing Waveform

MDC/MDIO Timing

PARAMETER	SYMBOL	MIN	TYP	MAX	UNIT
MDC cycle time	$T1$	400			ns
MDC rise/fall time	$T2$			10	ns
MDIO input setup time (PHY Side)	$T3$	10			ns
MDIO input hold time (PHY Side)	$T4$	10			ns
MDIO output delay (PHY Side)	$T5$			300	ns

3.11.4 DDR4 Interface

Embedded a physical layer PHY for DDR4 interface timing optimization. This PHY will do real time driving, timing and termination calibration at power up.

PARAMETER	SYMBOL	MIN	TYP	MAX	UNIT
Clock frequency	f_{CK}		800		MHz
Clock cycle time	t_{CK}	1.245	1.25	1.255	ns

Note :
1. Follow JEDEC JESD79-4 specification.
2. For Clock cycle time, please follow ASPEED's criteria directly (The "Mean" value with 1K samples at least). You can ignore the 1.25ns to 1.5ns requirement on JEDEC SPEC when electrical AC timing SPEC is PASS.

3.11.5 PCIe Interface

The AC and DC specifications for these signals are identical to those defined in Section 4.3 of the PCI Express Base Specification, Rev 2.0.

3.11.6 DisplayPort Interface

The AC and DC specifications for these signals are identical to those defined in Section 3.4 &3.5 of the VESA DisplayPort Standard Specification, Ver 1.1a.

3.11.7 SPI Master Interface

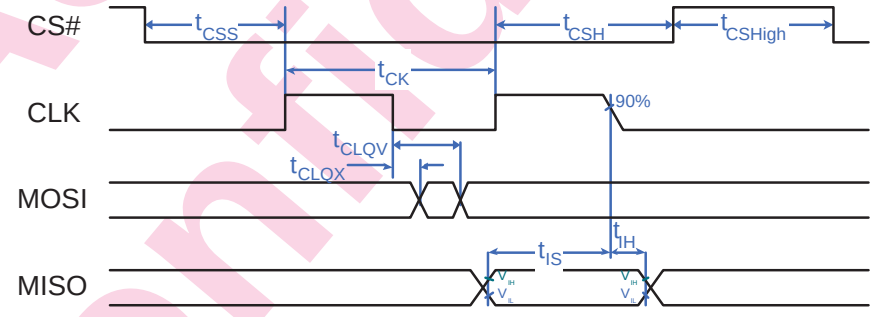


Figure 23: SPI Master Timing Waveform

SPI Master: t_{AHB} = AHB bus clock period (1/HCLK)

PARAMETER	SYMBOL	MIN	TYP	MAX	UNIT
Power on default SPI clock frequency	f_{SPI}		HCLK/8		MHz
Programmable SPI clock frequency	f_{SPI}	HCLK/256		100	MHz
Programmable SPI Clock period	t_{CK}	10		$256t_{AHB}$	ns
CS# Setup time	t_{CSS}	$2t_{AHB}$+0.5			t_{CK}
CS# Hold time	t_{CSH}	0.5			t_{CK}

to next page

from previous page

CS# Inactive time (programmable)	t_{CSHigh}	16		40	t_{AHB}
Data output Hold time	t_{CLQX}	-1			ns
Data output Valid time	t_{CLQV}			2	ns
Input Setup time (no path delay compensation)	t_{IS}	2			ns
Input Hold time (no path delay compensation)	t_{IH}	0			ns
Input Rise time ($f_{SPI} <= 50MHz$)	T_R			5	ns
Input Fall time ($f_{SPI} <= 50MHz$)	T_F			5	ns
Input Rise time ($f_{SPI} > 50MHz$)	T_R			3	ns
Input Fall time ($f_{SPI} > 50MHz$)	T_F			3	ns

Note :
The data input path has path delay compensation capability. It can shift the data input latch point by units of 1 $\sim$ 5 HCLK clocks cycle (t_{AHB}). And the data input setup/hold time should add the compensation delay based on the setting of FMC94/SPIR94 accordingly.

3.11.8 eSPI Slave Interface

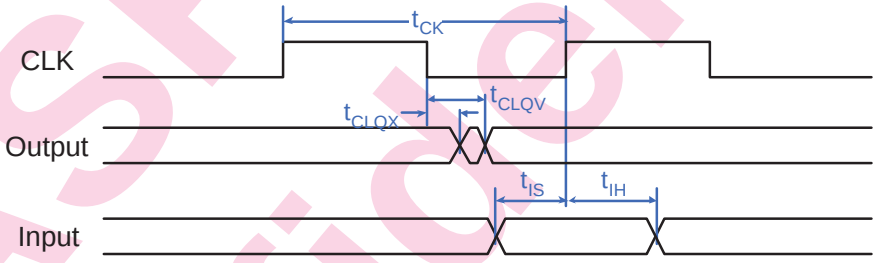


Figure 24: eSPI Slave Timing Waveform

PARAMETER	SYMBOL	MIN	TYP	MAX	UNIT
eSPI clock frequency (load $\leq$ 10pF)	f_{eSPI}			66	MHz
eSPI clock frequency (load $\leq$ 30pF)	f_{eSPI}			48	MHz
eSPI clock frequency (load $>$ 30pF)	f_{eSPI}			33	MHz
Data output Hold time	t_{CLQX}	3			ns
Data output Valid time (load $\leq$ 10pF)	t_{CLQV}			6	ns
Data output Valid time (load $\leq$ 30pF)	t_{CLQV}			8	ns
Data input from master Setup time	t_{IS}	3			ns
Data input from master Hold time	t_{IH}	3			ns
Input Rise time	T_R			3.5	ns
Input Fall time	T_F			3.5	ns

3.11.9 GPIO Pass-through Propagation Delay

PARAMETER	SYMBOL	MIN	TYP	MAX	UNIT
GPIOPx to GPIOPy	tpd_{GPIO}		4.5	5	ns

3.11.10 JTAG Master Interface

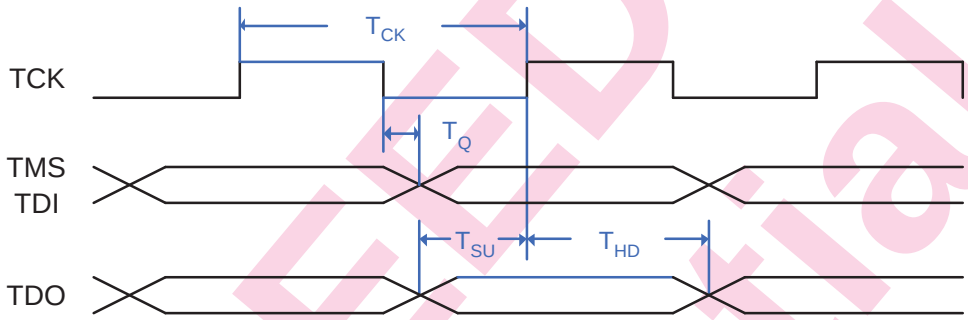


Figure 25: JTAG Master Timing Waveform

PARAMETER	SYMBOL	MIN	TYP	MAX	UNIT
Clock period (programmable)	T_{CK}	100			ns
Data input setup time	T_{SU}	5			ns
Data input hold time	T_{HD}	5			ns
Data output valid time after falling edge	T_Q			40	ns

3.11.11 I2C/SMBus Interface

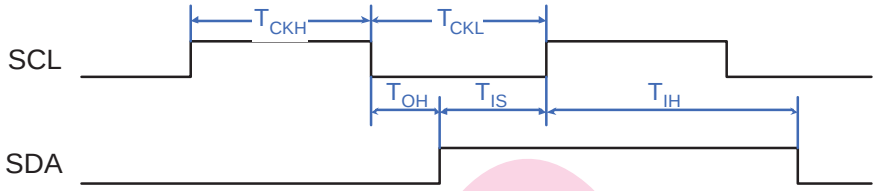


Figure 26: I2C Timing Waveform

PARAMETER	SYMBOL	MIN	TYP	MAX	UNIT
Clock rate (programmable)	f_{CK}	0.1		3400	KHz
Signal rise time - 100KHz	T_R			1000	ns
Signal rise time - 400KHz	T_R			300	ns
Signal rise time - 3.4MHz	T_R			80	ns
Signal fall time	T_F			300	ns
Data input setup time	T_{IS}	0			ns
Data input hold time	T_{IH}	T_{CKH}			ns
Data output valid time	T_{OH}	5			ns
Note : For rsie time, it must fine tune suitable external resistor for different speed.					

3.11.12 I3C Interface

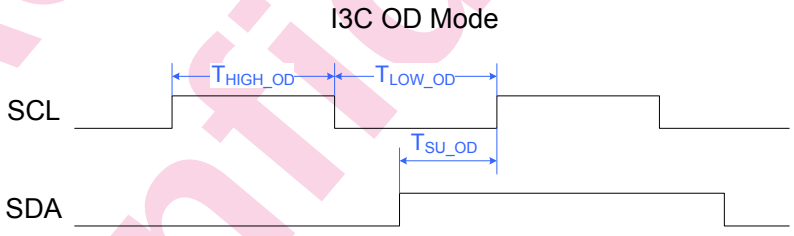


Figure 27: I3C Open-drain mode Timing Waveform

I3C Interface on open-drain mode

PARAMETER	SYMBOL	MIN	TYP	MAX	UNIT
High Period of SCL Clock	T_{HIGH_OD}		50		ns
Low Period of SCL Clock	T_{LOW_OD}		80		ns
SDA Inpput Setup Time	T_{SU_OD}	5			ns

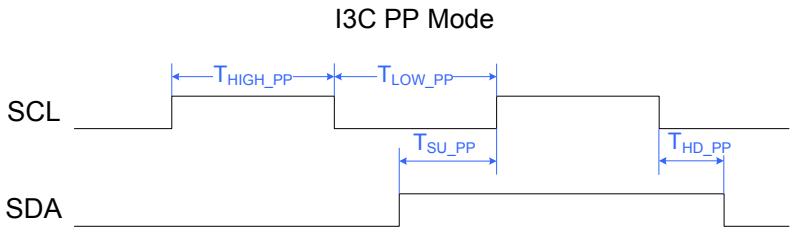


Figure 28: I3C Push-pull mode Timing Waveform

I3C Interface on push-pull mode

PARAMETER	SYMBOL	MIN	TYP	MAX	UNIT
SCL Clock High Period	T_{HIGH_PP}		40		ns
SCL Clock Low Period	T_{LOW_PP}		40		ns
SDA Input Setup Time	T_{SU_PP}	5			ns
SDA Input Hold Time	T_{HD_PP}	5			ns
SCL Pulse Width of Slope Reversal	T_{SLPR}			2.6	ns

3.11.13 SD/eMMC Interface

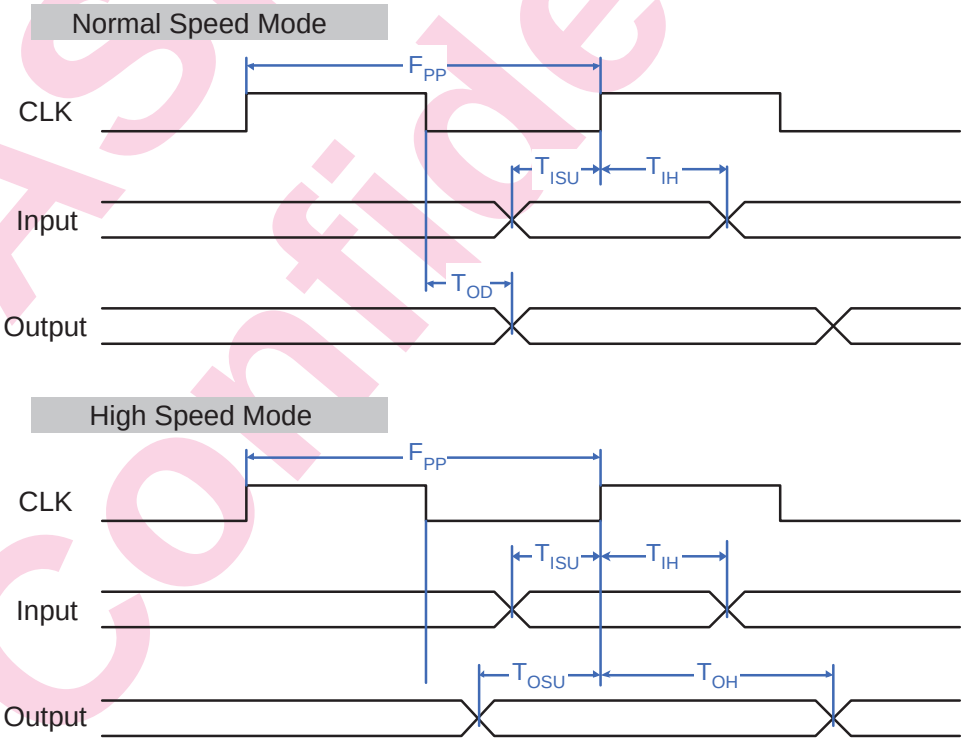


Figure 29: SD/eMMC Timing Waveform

Normal Speed Mode

PARAMETER	SYMBOL	MIN	TYP	MAX	UNIT
Clock frequency	F_{PP}			25	MHz
Clock duty cycle	TCK_{Duty}	45		55	%
Output delay	T_{OD}			10	ns
Input Setup time	T_{ISU}	2			ns
Input Hold time	T_{IH}	5			ns
Input Rise time	T_R			6	ns
Input Fall time	T_F			6	ns

High Speed Mode

PARAMETER	SYMBOL	MIN	TYP	MAX	UNIT
Clock frequency	F_{PP}			50	MHz
Clock duty cycle	TCK_{Duty}	45		55	%
Output Setup time	T_{OSU}	10			ns
Output Hold time	T_{OH}	5			ns
Input Setup time	T_{ISU}	2			ns
Input Hold time	T_{IH}	2			ns
Input Rise time	T_R			3	ns
Input Fall time	T_F			3	ns

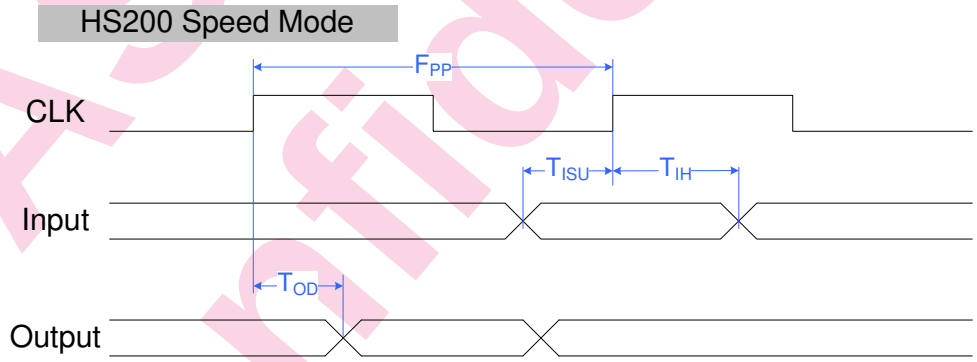


Figure 30: eMMC HS200 Timing Waveform

HS200 Speed Mode

PARAMETER	SYMBOL	MIN	TYP	MAX	UNIT
Clock frequency	F_{PP}			200	MHz
Clock duty cycle	TCK_{Duty}	45		55	%
Output delay	T_{OD}		1		ns
Input Setup time	T_{ISU}	1.4			ns
Input Hold time	T_{IH}	0.8			ns
Input Rise time (Cload $<$ 6pF)	T_R			1	ns
Input Fall time (Cload $<$ 6pF)	T_F			1	ns

Note :
When eMMC controller in HS200 mode, it supports self-tune and manual adjustment of the sampling phase Setup/Hold time. ASPEED recommends to center the clock edge w.r.t data period. This recommendation helps finding the best sampling point easier.

3.11.14 SGPIO Master Interface

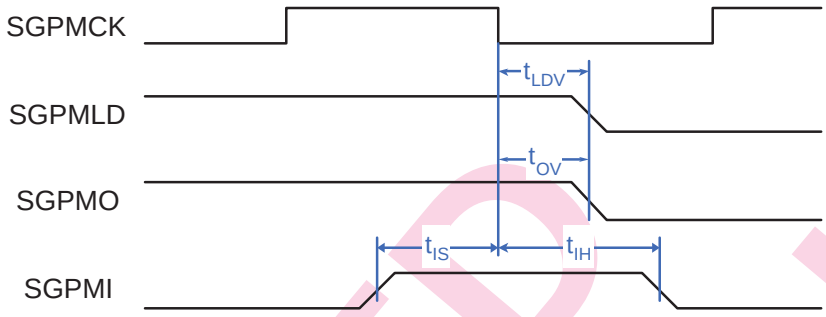


Figure 31: SGPIO Master Timing Waveform

PARAMETER	SYMBOL	MIN	TYP	MAX	UNIT
Clock frequency (programmable)	f_{CK}		0.194	25	MHz
Output valid of Load	t_{LDV}			5	ns
Output valid of Data	t_{OV}			5	ns
Input Setup time	t_{IS}	10			ns
Input Hold time	t_{IH}	1			ns
Input Rise time	T_R			6	ns
Input Fall time	T_F			6	ns

3.11.15 SGPIO Slave Interface

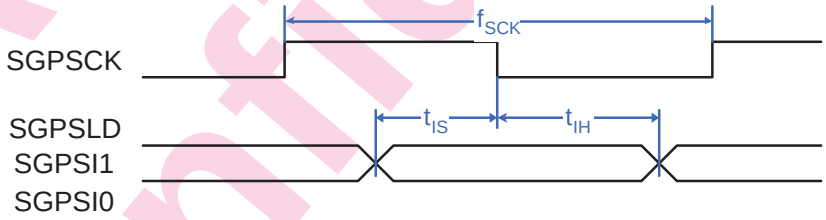


Figure 32: SGPIO Slave Timing Waveform

PARAMETER	SYMBOL	MIN	TYP	MAX	UNIT
Clock frequency (programmable)	f_{SCK}			1	MHz
Input Setup time	t_{IS}	50			ns
Input Hold time	t_{IH}	300			ns
Input Rise time	T_R			6	ns
Input Fall time	T_F			6	ns

3.11.16 Strap Input Interface

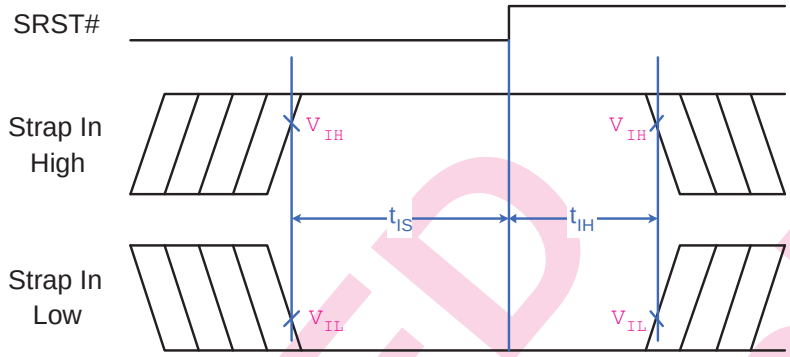


Figure 33: Strap Input Timing Waveform

PARAMETER	SYMBOL	MIN	TYP	MAX	UNIT
Input Setup time	t_{IS}	100			ns
Input Hold time	t_{IH}	50		1000	us

Note :
The transaction time from SRST# de-assertion is 2.6ms at the AS2600 input pin to the first external FWSPI transaction (FWSPICS0# going low). That's why TIH define maximum 1ms SPEC. You can ingore this TIH maximum 1ms SPEC if system just use external pull-up resitor control

3.11.17 FSI Interface

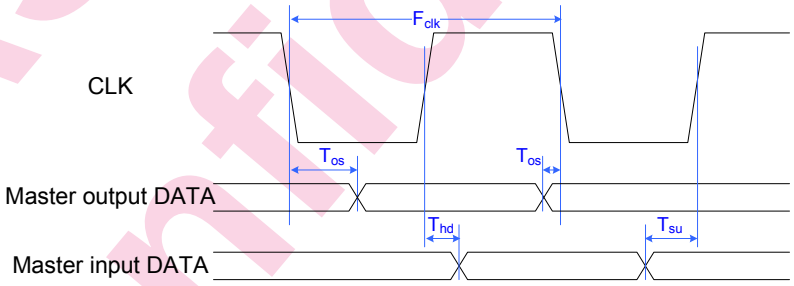


Figure 34: FSI Timing Waveform

FSI Interface

PARAMETER	SYMBOL	MIN	TYP	MAX	UNIT
Clock cycle time	F_{clk}	1		166	MHz
CLK/DATA output skew time	T_{os}	-300		300	ps
Input setup time	T_{su}	2			ns
Input hold time	T_{hd}	2			ns
Input/Output Rise time	T_R	0.3		1.5	ns
Input/Output Fall time	T_F	0.3		1.5	ns

3.11.18 UART Interface

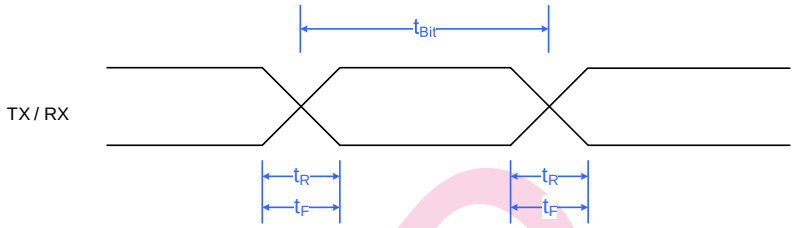


Figure 35: UART Timing Waveform

UART Interface

PARAMETER	SYMBOL	MIN	TYP	MAX	UNIT
Baud Rate				3.6864	MBits/s
Baud Rate tolerance		-3		3	%
Bit Time	T_{Bit}	271			ns
Input/Output Rise time	T_R			0.05	T_{Bit}
Input/Output Fall time	T_F			0.05	T_{Bit}
Note : The Rise time and Fall time is calculated IO supply voltage from 10% to 90%.					

3.12 Thermal Specification (Simulation Result)

3.12.1 Terminology

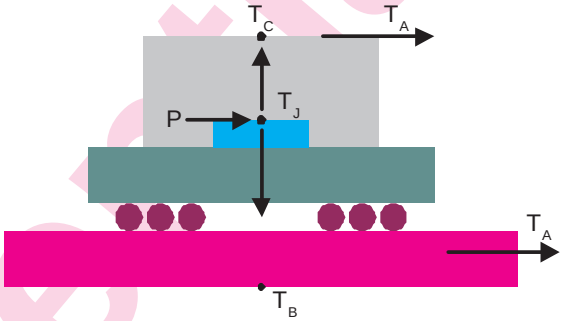
The major thermal dissipation paths can be illustrated as following:

- T_J : the maximum junction temperature
- T_T : the maximum top-center temperature
- T_A : the ambient or environment temperature
- T_C : the maximum compound surface temperature
- T_B : the maximum surface temperature of PCB bottom
- P : total input power

The thermal parameters can be defined as following figure:

1. Junction to ambient thermal resistance, Theta $_{JA}$

$$\theta_{JA} = \frac{T_J - T_A}{P}$$



Thermal Dissipation of PBGA Package

2. Junction to case thermal resistance, Theta $_{JC}$

$$\theta_{JC} = \frac{T_J - T_C}{P}$$

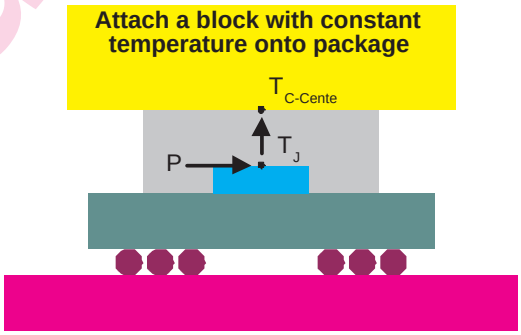


Figure 36: Thermal Terminology

3.12.2 Testing Conditions

Package Conditions	
Package Type	TFBGA Package
Ball Count	624
Package Dimension (L x W)	21 x 21 mm
Ball Pitch	0.8 mm
Number of Cu Layer-Substrate	4 layers
Substrate thickness	0.36 mm
PCB Conditions	
PCB layers	6 layers
PCB Dimensions (L x W x H)	130 x 120 x 1.66 mm^3
Environment Conditions	
Maximum Junction temperature (C)	125
Maximum Ambient temperature (C)	70
Input Power (watt)	2.5
Control Condition	Air Flow = 0, 1, 2 m/s

3.12.3 Thermal Data

TDP	$\theta_{JA}(^oC/W)$			$\theta_{JB}(^oC/W)$	$\theta_{JC}(^oC/W)$
DDR4	0 m/s	1 m/s	2 m/s		
	27.6	25.2	24.3	20	11.1

TDP power includes the DRAM and termination power.

3.12.4 Substrate Material Properties

Item	Material	Thermal conductivity K(W/mK)
Die	Si	147
Substrate Metal	Cu	400
Film	FH900	0.2
Mold Compound	KE-G1250	0.9
Solder ball	SAC	55
Wire	Cu	400

4 Package Information

Package Information List

- Alpha Particle Level : ULA (Ultra Low Alpha) ¡= 0.002 cph/cm2
- JEDEC Moisture Sensitivity Level : MSL-3
- Package Type: TFBGA (MCM)
- Package Ball Count: 624 (26 row x 26 row)
- Package Size: 21 mm x 21 mm
- Package Height: Max 1.53 mm
- Ball Pitch / Ball Size: 0.80 mm / 0.5mm
- Package Ball: Lead Free

4.1 Package Outline Drawing

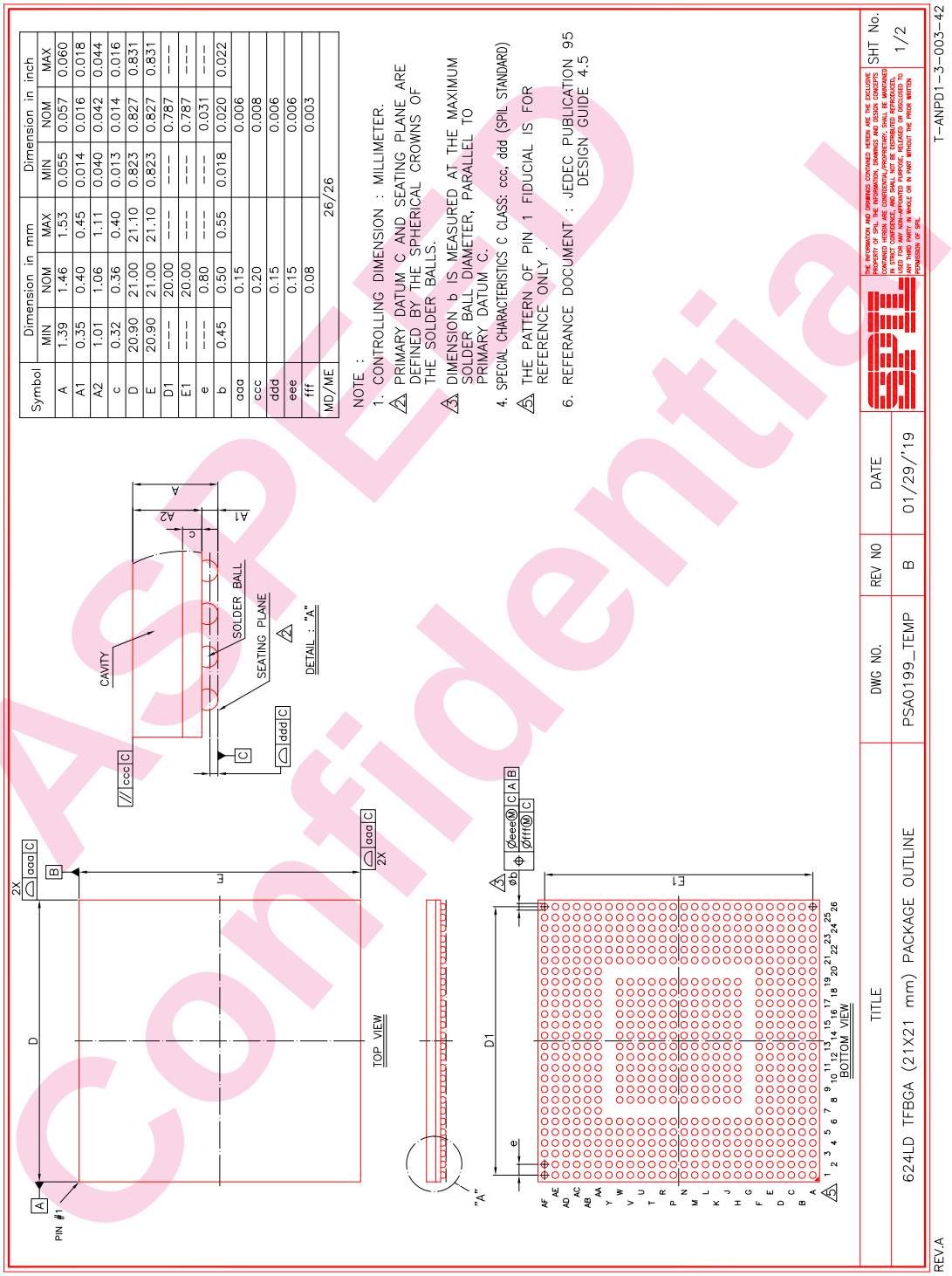
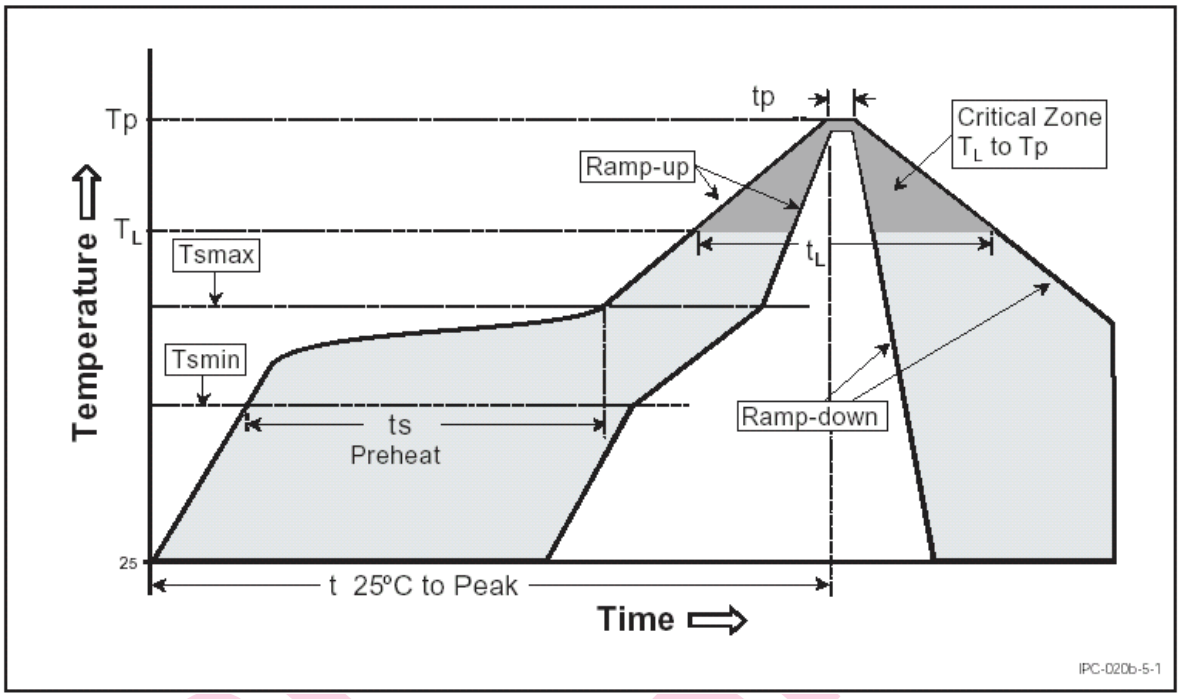


Figure 37: IC Package

4.2 SMT Soldering Reflow Chart



Profile Feature	Pb-Free Assembly	
	Large Body	Small Body
Average ramp-up rate (T_L to T_p)	1.5℃/second max	
Preheat -Temperature Min (Ts_{min}) -Temperature Max (Ts_{max}) -Time (min to max) (ts)	150℃ 200℃ 60-120 seconds	
Time maintained above: -Temperature (T_L) -Time (t_L)	217℃ 60-150 seconds	
Peak Temperature (Tp)	260+5/-5℃	
Time within 5℃ of actual Peak Temperature (tp)	30 seconds max	
Ramp-down Rate	3℃/second max.	
Time 25℃ to Peak Temperature	8 minutes max.	

Note:
1.All temperatures refer to topside of the package, measured on the package body surface.
2.Actual board assembly depends on other parts on board density and follower solder paste manufa

Figure 38: SMT Soldering Reflow Chart

Part II
Firmware Programming Guide

5 Multi-function Pins Mapping and Control

The following table defines the working function of all multi-function pins. The control priority is from ”Function 1” (Highest) to ”Function 3” (Lowest).

5.1 Function 1-3

Ball	Default	Function 1	Control 1	Function 2	Control 2	Function 3	Control 3	Others
AF23	Hi-Z, Input	I3C1SCL	SCU438[16]=1					Hi-Z, Input
AE24	Hi-Z, Input	I3C1SDA	SCU438[17]=1					Hi-Z, Input
AF22	Hi-Z, Input	I3C2SCL	SCU438[18]=1					Hi-Z, Input
AE22	Hi-Z, Input	I3C2SDA	SCU438[19]=1					Hi-Z, Input
AF25	Hi-Z, Input	I3C3SCL	SCU438[20]=1	FSI1CLK	SCU4D8[20]=1			Hi-Z, Input
AE26	Hi-Z, Input	I3C3SDA	SCU438[21]=1	FSI1DATA	SCU4D8[21]=1			Hi-Z, Input
AE25	Hi-Z, Input	I3C4SCL	SCU438[22]=1	FSI2CLK	SCU4D8[22]=1			Hi-Z, Input
AF24	Hi-Z, Input	I3C4SDA	SCU438[23]=1	FSI2DATA	SCU4D8[23]=1			Hi-Z, Input
M24	GPIOA0	MDC3	SCU410[0]=1	SCL11	SCU4B0[0]=1			GPIOA0
M25	GPIOA1	MDIO3	SCU410[1]=1	SDA11	SCU4B0[1]=1			GPIOA1
L26	GPIOA2	MDC4	SCU410[2]=1	SCL12	SCU4B0[2]=1			GPIOA2
K24	GPIOA3	MDIO4	SCU410[3]=1	SDA12	SCU4B0[3]=1			GPIOA3
K26	GPIOA4	MACLINK1	SCU410[4]=1	SCL13	SCU4B0[4]=1	SGPS2CK	SCU690[4]=1	GPIOA4
L24	GPIOA5	MACLINK2	SCU410[5]=1	SDA13	SCU4B0[5]=1	SGPS2LD	SCU690[5]=1	GPIOA5
L23	GPIOA6	MACLINK3	SCU410[6]=1	SCL14	SCU4B0[6]=1	SGPS2I0	SCU690[6]=1	GPIOA6
K25	GPIOA7	MACLINK4	SCU410[7]=1	SDA14	SCU4B0[7]=1	SGPS2I1	SCU690[7]=1	GPIOA7
J26	GPIOB0	SALT1	SCU410[8]=1			SALTS1	SCU690[8]=1	GPIOB0
K23	GPIOB1	SALT2	SCU410[9]=1			SALTS2	SCU690[9]=1	GPIOB1
H26	GPIOB2	SALT3	SCU410[10]=1			SALTS3	SCU690[10]=1	GPIOB2
J25	GPIOB3	SALT4	SCU410[11]=1			SALTS4	SCU690[11]=1	GPIOB3
J23	GPIOB4	MDC2	SCU410[12]=1					GPIOB4
G26	GPIOB5	MDIO2	SCU410[13]=1					GPIOB5
H25	GPIOB6	TXD4	SCU410[14]=1					GPIOB6
J24	GPIOB7	RXD4	SCU410[15]=1					GPIOB7
H24	GPIOC0	RGMII3TXCK	SCU410[16]=1 & SCU510[0]=1	RMII3RCLKO	SCU410[16]=1 & SCU510[0]=0			GPIOC0
J22	GPIOC1	RGMII3TXCTL	SCU410[17]=1 & SCU510[0]=1	RMII3TXEN	SCU410[17]=1 & SCU510[0]=0			GPIOC1
H22	GPIOC2	RGMII3TXD0	SCU410[18]=1 & SCU510[0]=1	RMII3TXD0	SCU410[18]=1 & SCU510[0]=0			GPIOC2

to next page

from previous page

Ball	Default	Function 1	Control 1	Function 2	Control 2	Function 3	Control 3	Others
H23	GPIOC3	RGMII3TXD1	SCU410[19]=1 & SCU510[0]=1	RMII3TXD1	SCU410[19]=1 & SCU510[0]=0			GPIOC3
G22	GPIOC4	RGMII3TXD2	SCU410[20]=1 & SCU510[0]=1					GPIOC4
F22	GPIOC5	RGMII3TXD3	SCU410[21]=1 & SCU510[0]=1					GPIOC5
G23	GPIOC6	RGMII3RXCK	SCU410[22]=1 & SCU510[0]=1	RMII3RCLKI	SCU410[22]=1 & SCU510[0]=0			GPIOC6
G24	GPIOC7	RGMII3RXCTL	SCU410[23]=1 & SCU510[0]=1					GPIOC7
F23	GPIOD0	RGMII3RXD0	SCU410[24]=1 & SCU510[0]=1	RMII3RXD0	SCU410[24]=1 & SCU510[0]=0			GPIOD0
F26	GPIOD1	RGMII3RXD1	SCU410[25]=1 & SCU510[0]=1	RMII3RXD1	SCU410[25]=1 & SCU510[0]=0			GPIOD1
F25	GPIOD2	RGMII3RXD2	SCU410[26]=1 & SCU510[0]=1	RMII3CRSDV	SCU410[26]=1 & SCU510[0]=0			GPIOD2
E26	GPIOD3	RGMII3RXD3	SCU410[27]=1 & SCU510[0]=1	RMII3RXER	SCU410[27]=1 & SCU510[0]=0			GPIOD3
F24	GPIOD4	NCTS3	SCU410[28]=1	RGMII4TXCK	SCU4B0[28]=1 & SCU510[1]=1	RMII4RCLKO	SCU4B0[28]=1 & SCU510[1]=0	GPIOD4
E23	GPIOD5	NDCD3	SCU410[29]=1	RGMII4TXCTL	SCU4B0[29]=1 & SCU510[1]=1	RMII4TXEN	SCU4B0[29]=1 & SCU510[1]=0	GPIOD5
E24	GPIOD6	NDSR3	SCU410[30]=1	RGMII4TXD0	SCU4B0[30]=1 & SCU510[1]=1	RMII4TXD0	SCU4B0[30]=1 & SCU510[1]=0	GPIOD6
E25	GPIOD7	NRI3	SCU410[31]=1	RGMII4TXD1	SCU4B0[31]=1 & SCU510[1]=1	RMII4TXD1	SCU4B0[31]=1 & SCU510[1]=0	GPIOD7
D26	GPIOE0	NDTR3	SCU414[0]=1 & SCU470[16]=0	RGMII4TXD2	SCU4B4[0]=1 & SCU510[1]=1 & SCU470[16]=0			GPIOE0
D24	GPIOE1	NRTS3	SCU414[1]=1 & SCU470[17]=0	RGMII4TXD3	SCU4B4[1]=1 & SCU510[1]=1 & SCU470[17]=0			GPIOE1

to next page

from previous page

Ball	Default	Function 1	Control 1	Function 2	Control 2	Function 3	Control 3	Others
C25	GPIOE2	NCTS4	SCU414[2]=1 & SCU470[18]=0	RGMII4RXCK	SCU4B4[2]=1 & SCU510[1]=1 & SCU470[18]=0	RMII4RCLKI	SCU4B4[2]=1 & SCU510[1]=0 & SCU470[18]=0	GPIOE2
C26	GPIOE3	NDCD4	SCU414[3]=1 & SCU470[19]=0	RGMII4RXCTL	SCU4B4[3]=1 & SCU510[1]=1 & SCU470[19]=0			GPIOE3
C24	GPIOE4	NDSR4	SCU414[4]=1 & SCU470[20]=0	RGMII4RXD0	SCU4B4[4]=1 & SCU510[1]=1 & SCU470[20]=0	RMII4RXD0	SCU4B4[4]=1 & SCU510[1]=0 & SCU470[20]=0	GPIOE4
B26	GPIOE5	NRI4	SCU414[5]=1 & SCU470[21]=0	RGMII4RXD1	SCU4B4[5]=1 & SCU510[1]=1 & SCU470[21]=0	RMII4RXD1	SCU4B4[5]=1 & SCU510[1]=0 & SCU470[21]=0	GPIOE5
B25	GPIOE6	NDTR4	SCU414[6]=1 & SCU470[22]=0	RGMII4RXD2	SCU4B4[6]=1 & SCU510[1]=1 & SCU470[22]=0	RMII4CRSDV	SCU4B4[6]=1 & SCU510[1]=0 & SCU470[22]=0	GPIOE6
B24	GPIOE7	NRTS4	SCU414[7]=1 & SCU470[23]=0	RGMII4RXD3	SCU4B4[7]=1 & SCU510[1]=1 & SCU470[23]=0	RMII4RXER	SCU4B4[7]=1 & SCU510[1]=0 & SCU470[23]=0	GPIOE7
D22	GPIOF0	SD1CLK	SCU414[8]=1	PWM8	SCU4B4[8]=1			GPIOF0
E22	GPIOF1	SD1CMD	SCU414[9]=1	PWM9	SCU4B4[9]=1			GPIOF1
D23	GPIOF2	SD1DAT0	SCU414[10]=1	PWM10	SCU4B4[10]=1			GPIOF2
C23	GPIOF3	SD1DAT1	SCU414[11]=1	PWM11	SCU4B4[11]=1			GPIOF3
C22	GPIOF4	SD1DAT2	SCU414[12]=1	PWM12	SCU4B4[12]=1			GPIOF4
A25	GPIOF5	SD1DAT3	SCU414[13]=1	PWM13	SCU4B4[13]=1			GPIOF5
A24	GPIOF6	SD1CD#	SCU414[14]=1	PWM14	SCU4B4[14]=1			GPIOF6
A23	GPIOF7	SD1WP#	SCU414[15]=1	PWM15	SCU4B4[15]=1			GPIOF7
E21	GPIOG0	TXD6	SCU414[16]=1	SD2CLK	SCU4B4[16]=1 & SCU450[1]=1	SALT9	SCU694[16]=1	GPIOG0
B22	GPIOG1	RXD6	SCU414[17]=1	SD2CMD	SCU4B4[17]=1 & SCU450[1]=1	SALT10	SCU694[17]=1	GPIOG1

to next page

from previous page

Ball	Default	Function 1	Control 1	Function 2	Control 2	Function 3	Control 3	Others
C21	GPIOG2	TXD7	SCU414[18]=1	SD2DAT0	SCU4B4[18]=1 & SCU450[1]=1	SALT11	SCU694[18]=1	GPIOG2
A22	GPIOG3	RXD7	SCU414[19]=1	SD2DAT1	SCU4B4[19]=1 & SCU450[1]=1	SALT12	SCU694[19]=1	GPIOG3
A21	GPIOG4	TXD8	SCU414[20]=1	SD2DAT2	SCU4B4[20]=1 & SCU450[1]=1	SALT13	SCU694[20]=1	GPIOG4
E20	GPIOG5	RXD8	SCU414[21]=1	SD2DAT3	SCU4B4[21]=1 & SCU450[1]=1	SALT14	SCU694[21]=1	GPIOG5
D21	GPIOG6	TXD9	SCU414[22]=1	SD2CD#	SCU4B4[22]=1 & SCU450[1]=1	SALT15	SCU694[22]=1	GPIOG6
B21	GPIOG7	RXD9	SCU414[23]=1	SD2WP#	SCU4B4[23]=1 & SCU450[1]=1	SALT16	SCU694[23]=1	GPIOG7
A18	GPIOH0	SGPM1CK	SCU414[24]=1					GPIOH0
B18	GPIOH1	SGPM1LD	SCU414[25]=1					GPIOH1
C18	GPIOH2	SGPM1O	SCU414[26]=1					GPIOH2
A17	GPIOH3	SGPM1I	SCU414[27]=1					GPIOH3
D18	GPIOH4	SGPS1CK	SCU414[28]=1	SCL15	SCU4B4[28]=1	SCLS3	SCU694[28]=1	GPIOH4
B17	GPIOH5	SGPS1LD	SCU414[29]=1	SDA15	SCU4B4[29]=1	SDAS3	SCU694[29]=1	GPIOH5
C17	GPIOH6	SGPS1I0	SCU414[30]=1	SCL16	SCU4B4[30]=1	SCLS4	SCU694[30]=1	GPIOH6
E18	GPIOH7	SGPS1I1	SCU414[31]=1	SDA16	SCU4B4[31]=1	SDAS4	SCU694[31]=1	GPIOH7
D17	GPIOI0	MTRSTN2	SCU418[0]=1	TXD12	SCU4B8[0]=1			GPIOI0
A16	GPIOI1	MTDI2	SCU418[1]=1	RXD12	SCU4B8[1]=1			GPIOI1
E17	GPIOI2	MTCK2	SCU418[2]=1	TXD13	SCU4B8[2]=1			GPIOI2
D16	GPIOI3	MTMS2	SCU418[3]=1	RXD13	SCU4B8[3]=1			GPIOI3
C16	GPIOI4	MTDO2	SCU418[4]=1					GPIOI4
E16	GPIOI5	SIOPBO#	SCU418[5]=1 ‖ SCU510[5]=1					GPIOI5
B16	GPIOI6	SIOPBI#	SCU418[6]=1 ‖ SCU510[5]=1					GPIOI6

to next page

from previous page

Ball	Default	Function 1	Control 1	Function 2	Control 2	Function 3	Control 3	Others
A15	GPIOI7	SIOSCI#	SCU418[7]=1 ‖ SCU510[5]=1					GPIOI7
B20	GPIOJ0	HVI3C3SCL	SCU418[8]=1	SCL1	SCU4B8[8]=1	SCLS1	SCU698[8]=1	GPIOJ0
A20	GPIOJ1	HVI3C3SDA	SCU418[9]=1	SDA1	SCU4B8[9]=1	SDAS1	SCU698[9]=1	GPIOJ1
E19	GPIOJ2	HVI3C4SCL	SCU418[10]=1	SCL2	SCU4B8[10]=1	SCLS2	SCU698[10]=1	GPIOJ2
D20	GPIOJ3	HVI3C4SDA	SCU418[11]=1	SDA2	SCU4B8[11]=1	SDAS2	SCU698[11]=1	GPIOJ3
C19	GPIOJ4	HVI3C5SCL	SCU418[12]=1	SCL3	SCU4B8[12]=1	SCLS3	SCU698[12]=1	GPIOJ4
A19	GPIOJ5	HVI3C5SDA	SCU418[13]=1	SDA3	SCU4B8[13]=1	SDAS3	SCU698[13]=1	GPIOJ5
C20	GPIOJ6	HVI3C6SCL	SCU418[14]=1	SCL4	SCU4B8[14]=1	SCLS4	SCU698[14]=1	GPIOJ6
D19	GPIOJ7	HVI3C6SDA	SCU418[15]=1	SDA4	SCU4B8[15]=1	SDAS4	SCU698[15]=1	GPIOJ7
A11	GPIOK0	SCL5	SCU418[16]=1	SCLS1	SCU4B8[16]=1			GPIOK0
C11	GPIOK1	SDA5	SCU418[17]=1	SDAS1	SCU4B8[17]=1			GPIOK1
D12	GPIOK2	SCL6	SCU418[18]=1	SCLS2	SCU4B8[18]=1			GPIOK2
E13	GPIOK3	SDA6	SCU418[19]=1	SDAS2	SCU4B8[19]=1			GPIOK3
D11	GPIOK4	SCL7	SCU418[20]=1	SCLS3	SCU4B8[20]=1			GPIOK4
E11	GPIOK5	SDA7	SCU418[21]=1	SDAS3	SCU4B8[21]=1			GPIOK5
F13	GPIOK6	SCL8	SCU418[22]=1	SCLS4	SCU4B8[22]=1			GPIOK6
E12	GPIOK7	SDA8	SCU418[23]=1	SDAS4	SCU4B8[23]=1			GPIOK7
D15	GPIOL0	SCL9	SCU418[24]=1	SCLS1	SCU4B8[24]=1			GPIOL0
A14	GPIOL1	SDA9	SCU418[25]=1	SDAS1	SCU4B8[25]=1			GPIOL1
E15	GPIOL2	SCL10	SCU418[26]=1	SCLS2	SCU4B8[26]=1			GPIOL2
A13	GPIOL3	SDA10	SCU418[27]=1	SDAS2	SCU4B8[27]=1			GPIOL3
C15	GPIOL4	TXD3	SCU418[28]=1					GPIOL4
F15	GPIOL5	RXD3	SCU418[29]=1					GPIOL5
B14	GPIOL6	VGAHS	SCU418[30]=1					GPIOL6
C14	GPIOL7	VGAVS	SCU418[31]=1					GPIOL7
D14	GPIOM0	NCTS1	SCU41C[0]=1					GPIOM0
B13	GPIOM1	NDCD1	SCU41C[1]=1					GPIOM1

to next page

from previous page

Ball	Default	Function 1	Control 1	Function 2	Control 2	Function 3	Control 3	Others
A12	GPIOM2	NDSR1	SCU41C[2]=1					GPIOM2
E14	GPIOM3	NRI1	SCU41C[3]=1					GPIOM3
B12	GPIOM4	NDTR1	SCU41C[4]=1					GPIOM4
C12	GPIOM5	NRTS1	SCU41C[5]=1					GPIOM5
C13	GPIOM6	TXD1	SCU41C[6]=1					GPIOM6
D13	GPIOM7	RXD1	SCU41C[7]=1					GPIOM7
P25	GPION0	NCTS2	SCU41C[8]=1					GPION0
N23	GPION1	NDCD2	SCU41C[9]=1					GPION1
N25	GPION2	NDSR2	SCU41C[10]=1					GPION2
N24	GPION3	NRI2	SCU41C[11]=1					GPION3
P26	GPION4	NDTR2	SCU41C[12]=1					GPION4
M23	GPION5	NRTS2	SCU41C[13]=1					GPION5
N26	GPION6	TXD2	SCU41C[14]=1					GPION6
M26	GPION7	RXD2	SCU41C[15]=1					GPION7
AD26	GPIOO0	PWM0	SCU41C[16]=1					GPIOO0
AD22	GPIOO1	PWM1	SCU41C[17]=1					GPIOO1
AD23	GPIOO2	PWM2	SCU41C[18]=1					GPIOO2
AD24	GPIOO3	PWM3	SCU41C[19]=1					GPIOO3
AD25	GPIOO4	PWM4	SCU41C[20]=1					GPIOO4
AC22	GPIOO5	PWM5	SCU41C[21]=1					GPIOO5
AC24	GPIOO6	PWM6	SCU41C[22]=1					GPIOO6
AC23	GPIOO7	PWM7	SCU41C[23]=1					GPIOO7
AB22	GPIOP0	PWM8	SCU41C[24]=1	THRUIN0	SCU4BC[24]=1 || SCU510[28]=1 || SCU51C[9]=1			GPIOP0
W24	GPIOP1	PWM9	SCU41C[25]=1	THRUOUT0	SCU4BC[25]=1 || SCU510[28]=1 || SCU51C[9]=1			GPIOP1

to next page

from previous page

Ball	Default	Function 1	Control 1	Function 2	Control 2	Function 3	Control 3	Others
AA23	GPIOP2	PWM10	SCU41C[26]=1	THRUIN1	SCU4BC[26]=1 || SCU510[28]=1 || SCU51C[9]=1			GPIOP2
AA24	GPIOP3	PWM11	SCU41C[27]=1	THRUOUT1	SCU4BC[27]=1 || SCU510[28]=1 || SCU51C[9]=1			GPIOP3
W23	GPIOP4	PWM12	SCU41C[28]=1	THRUIN2	SCU4BC[28]=1 || SCU510[28]=1 || SCU51C[9]=1			GPIOP4
AB23	GPIOP5	PWM13	SCU41C[29]=1	THRUOUT2	SCU4BC[29]=1 || SCU510[28]=1 || SCU51C[9]=1			GPIOP5
AB24	GPIOP6	PWM14	SCU41C[30]=1					GPIOP6
Y23	HEARTBEAT	PWM15	SCU41C[31]=1			SCU69C[31]=1	HEARTBEAT	GPIOP7
AA25	GPIOQ0	TACH0	SCU430[0]=1					GPIOQ0
AB25	GPIOQ1	TACH1	SCU430[1]=1					GPIOQ1
Y24	GPIOQ2	TACH2	SCU430[2]=1					GPIOQ2
AB26	GPIOQ3	TACH3	SCU430[3]=1					GPIOQ3
Y26	GPIOQ4	TACH4	SCU430[4]=1					GPIOQ4
AC26	GPIOQ5	TACH5	SCU430[5]=1					GPIOQ5
Y25	GPIOQ6	TACH6	SCU430[6]=1					GPIOQ6
AA26	GPIOQ7	TACH7	SCU430[7]=1					GPIOQ7
V25	GPIOR0	TACH8	SCU430[8]=1					GPIOR0
U24	GPIOR1	TACH9	SCU430[9]=1					GPIOR1
V24	GPIOR2	TACH10	SCU430[10]=1					GPIOR2
V26	GPIOR3	TACH11	SCU430[11]=1					GPIOR3
U25	GPIOR4	TACH12	SCU430[12]=1					GPIOR4
T23	GPIOR5	TACH13	SCU430[13]=1					GPIOR5
W26	GPIOR6	TACH14	SCU430[14]=1					GPIOR6
U26	GPIOR7	TACH15	SCU430[15]=1					GPIOR7

to next page

from previous page

Ball	Default	Function 1	Control 1	Function 2	Control 2	Function 3	Control 3	Others
R23	GPIOS0	MDC1	SCU430[16]=1					GPIOS0
T25	GPIOS1	MDIO1	SCU430[17]=1					GPIOS1
T26	GPIOS2	PEWAKE#	SCU430[18]=1					GPIOS2
R24	GPIOS3	OSCCLK	SCU430[19]=1					GPIOS3
R26	GPIOS4	TXD10	SCU430[20]=1					GPIOS4
P24	GPIOS5	RXD10	SCU430[21]=1					GPIOS5
P23	GPIOS6	TXD11	SCU430[22]=1					GPIOS6
T24	GPIOS7	RXD11	SCU430[23]=1					GPIOS7
AD20	ADC0	GPIT0	SCU430[24]=1					ADC0
AC18	ADC1	GPIT1	SCU430[25]=1					ADC1
AE19	ADC2	GPIT2	SCU430[26]=1					ADC2
AD19	ADC3	GPIT3	SCU430[27]=1					ADC3
AC19	ADC4	GPIT4	SCU430[28]=1					ADC4
AB19	ADC5	GPIT5	SCU430[29]=1					ADC5
AB18	ADC6	GPIT6	SCU430[30]=1					ADC6
AE18	ADC7	GPIT7	SCU430[31]=1					ADC7
AB16	ADC8	SALT9	SCU434[0]=1 & SCU694[16]=0 & SCU4D4[0]=1	GPIU0	SCU434[0]=1 & SCU4D4[0]=0			ADC8
AA17	ADC9	SALT10	SCU434[1]=1 & SCU694[17]=0 & SCU4D4[1]=1	GPIU1	SCU434[1]=1 & SCU4D4[1]=0			ADC9
AB17	ADC10	SALT11	SCU434[2]=1 & SCU694[18]=0 & SCU4D4[2]=1	GPIU2	SCU434[2]=1 & SCU4D4[2]=0			ADC10
AE16	ADC11	SALT12	SCU434[3]=1 & SCU694[19]=0 & SCU4D4[3]=1	GPIU3	SCU434[3]=1 & SCU4D4[3]=0			ADC11

to next page

from previous page

Ball	Default	Function 1	Control 1	Function 2	Control 2	Function 3	Control 3	Others
AC16	ADC12	SALT13	SCU434[4]=1 & SCU694[20]=0 & SCU4D4[4]=1	GPIU4	SCU434[4]=1 & SCU4D4[4]=0			ADC12
AA16	ADC13	SALT14	SCU434[5]=1 & SCU694[21]=0 & SCU4D4[5]=1	GPIU5	SCU434[5]=1 & SCU4D4[5]=0			ADC13
AD16	ADC14	SALT15	SCU434[6]=1 & SCU694[22]=0 & SCU4D4[6]=1	GPIU6	SCU434[6]=1 & SCU4D4[6]=0			ADC14
AC17	ADC15	SALT16	SCU434[7]=1 & SCU694[23]=0 & SCU4D4[7]=1	GPIU7	SCU434[7]=1 & SCU4D4[7]=0			ADC15
AB15	GPIOV0	SIOS3#	SCU434[8]=1 || SCU510[5]=1					GPIOV0
AF14	GPIOV1	SIOS5#	SCU434[9]=1 || SCU510[5]=1					GPIOV1
AD14	GPIOV2	SIOPWREQ#	SCU434[10]=1 || SCU510[5]=1					GPIOV2
AC15	GPIOV3	SIOONCTRL#	SCU434[11]=1 || SCU510[5]=1					GPIOV3
AE15	GPIOV4	SIOPWRGD	SCU434[12]=1 || SCU510[5]=1					GPIOV4
AE14	GPIOV5	LPCPD#	SCU434[13]=1					GPIOV5
AD15	GPIOV6	LPCPME	SCU434[14]=1					GPIOV6
AF15	GPIOV7	LPCSMI#	SCU434[15]=1					GPIOV7
AB7	GPIOW0	ESPID0	SCU434[16]=1 & SCU510[6]=0	LAD0	SCU434[16]=1 & SCU510[6]=1			GPIOW0
AB8	GPIOW1	ESPID1	SCU434[17]=1 & SCU510[6]=0	LAD1	SCU434[17]=1 & SCU510[6]=1			GPIOW1
AC8	GPIOW2	ESPID2	SCU434[18]=1 & SCU510[6]=0	LAD2	SCU434[18]=1 & SCU510[6]=1			GPIOW2

to next page

from previous page

Ball	Default	Function 1	Control 1	Function 2	Control 2	Function 3	Control 3	Others
AC7	GPIOW3	ESPID3	SCU434[19]=1 & SCU510[6]=0	LAD3	SCU434[19]=1 & SCU510[6]=1			GPIOW3
AE7	GPIOW4	ESPICK	SCU434[20]=1 & SCU510[6]=0	LCLK	SCU434[20]=1 & SCU510[6]=1			GPIOW4
AF7	GPIOW5	ESPICS#	SCU434[21]=1 & SCU510[6]=0	LFRAME#	SCU434[21]=1 & SCU510[6]=1			GPIOW5
AD7	GPIOW6	ESPIALT#	SCU434[22]=1 & SCU510[6]=0	LSIRQ#	SCU434[22]=1 & SCU510[6]=1			GPIOW6
AD8	GPIOW7	ESPIRST#	SCU434[23]=1 & SCU510[6]=0	LPCRST#	SCU434[23]=1 & SCU510[6]=1			GPIOW7
AE8	GPIOX0	SPI2CS0#	SCU434[24]=1					GPIOX0
AA9	GPIOX1	SPI2CS1#	SCU434[25]=1					GPIOX1
AC9	GPIOX2	SPI2CS2#	SCU434[26]=1					GPIOX2
AF8	GPIOX3	SPI2CK	SCU434[27]=1					GPIOX3
AB9	GPIOX4	SPI2MOSI	SCU434[28]=1					GPIOX4
AD9	GPIOX5	SPI2MISO	SCU434[29]=1					GPIOX5
AF9	GPIOX6	SPI2DQ2	SCU434[30]=1	TXD12	SCU4D4[30]=1			GPIOX6
AB10	GPIOX7	SPI2DQ3	SCU434[31]=1	RXD12	SCU4D4[31]=1			GPIOX7
AF11	GPIOY0	SALT5	SCU438[0]=1	WDTRST1#	SCU4D8[0]=1	SALTS1	SCU6B8[0]=1	GPIOY0
AD12	GPIOY1	SALT6	SCU438[1]=1	WDTRST2#	SCU4D8[1]=1	SALTS2	SCU6B8[1]=1	GPIOY1
AE11	GPIOY2	SALT7	SCU438[2]=1	WDTRST3#	SCU4D8[2]=1	SALTS3	SCU6B8[2]=1	GPIOY2
AA12	GPIOY3	SALT8	SCU438[3]=1	WDTRST4#	SCU4D8[3]=1	SALTS4	SCU6B8[3]=1	GPIOY3
AE12	GPIOY4	FWSPIDQ2	SCU438[4]=1 || SCU510[22]=1					GPIOY4
AF12	GPIOY5	FWSPIDQ3	SCU438[5]=1 || SCU510[22]=1					GPIOY5
AC12	GPIOY6	FWSPIABR	SCU438[6]=1 || SCU510[22]=1					GPIOY6
AB12	GPIOY7	FWSPIWP#	SCU438[7]=1 || SCU510[22]=1					GPIOY7

to next page

from previous page

Ball	Default	Function 1	Control 1	Function 2	Control 2	Function 3	Control 3	Others
AC10	GPIOZ0	SPI1CS1#	SCU438[8]=1 || (SCU510[16]=1 & SCU510[18]=0)					GPIOZ0
AD10	GPIOZ1	SPI1ABR	SCU438[9]=1 || SCU510[17]=1 || SCU510[27]=1					GPIOZ1
AE10	GPIOZ2	SPI1WP#	SCU438[10]=1 || SCU510[27]=1					GPIOZ2
AB11	GPIOZ3	SPI1CK	SCU438[11]=1					SPI1CK
AC11	GPIOZ4	SPI1MOSI	SCU438[12]=1					SPI1MOSI
AA11	GPIOZ5	SPI1MISO	SCU438[13]=1					SPI1MISO
AD11	GPIOZ6	SPI1DQ2	SCU438[14]=1 || SCU510[27]=1	TXD13	SCU4D8[14]=1 & SCU4B8[2]=0			GPIOZ6
AF10	GPIOZ7	SPI1DQ3	SCU438[15]=1 || SCU510[27]=1	RXD13	SCU4D8[15]=1 & SCU4B8[3]=0			GPIOZ7
C6	GPIO18A0	RGMII1TXCK	SCU400[0]=1 & SCU500[6]=1	RMII1RCLKO	SCU400[0]=1 & SCU500[6]=0			GPIO18A0
D6	GPIO18A1	RGMII1TXCTL	SCU400[1]=1 & SCU500[6]=1	RMII1TXEN	SCU400[1]=1 & SCU500[6]=0			GPIO18A1
D5	GPIO18A2	RGMII1TXD0	SCU400[2]=1 & SCU500[6]=1	RMII1TXD0	SCU400[2]=1 & SCU500[6]=0			GPIO18A2
A3	GPIO18A3	RGMII1TXD1	SCU400[3]=1 & SCU500[6]=1	RMII1TXD1	SCU400[3]=1 & SCU500[6]=0			GPIO18A3
C5	GPIO18A4	RGMII1TXD2	SCU400[4]=1 & SCU500[6]=1					GPIO18A4
E6	GPIO18A5	RGMII1TXD3	SCU400[5]=1 & SCU500[6]=1					GPIO18A5
B3	GPIO18A6	RGMII1RXCK	SCU400[6]=1 & SCU500[6]=1	RMII1RCLKI	SCU400[6]=1 & SCU500[6]=0			GPIO18A6
A2	GPIO18A7	RGMII1RXCTL	SCU400[7]=1 & SCU500[6]=1					GPIO18A7

to next page

from previous page

Ball	Default	Function 1	Control 1	Function 2	Control 2	Function 3	Control 3	Others
B2	GPIO18B0	RGMII1RXD0	SCU400[8]=1 & SCU500[6]=1	RMII1RXD0	SCU400[8]=1 & SCU500[6]=0			GPIO18B0
B1	GPIO18B1	RGMII1RXD1	SCU400[9]=1 & SCU500[6]=1	RMII1RXD1	SCU400[9]=1 & SCU500[6]=0			GPIO18B1
C4	GPIO18B2	RGMII1RXD2	SCU400[10]=1 & SCU500[6]=1	RMII1CRSDV	SCU400[10]=1 & SCU500[6]=0			GPIO18B2
E5	GPIO18B3	RGMII1RXD3	SCU400[11]=1 & SCU500[6]=1	RMII1RXER	SCU400[11]=1 & SCU500[6]=0			GPIO18B3
D4	GPIO18B4	RGMII2TXCK	SCU400[12]=1 & SCU500[7]=1	RMII2RCLKO	SCU400[12]=1 & SCU500[7]=0			GPIO18B4
C2	GPIO18B5	RGMII2TXCTL	SCU400[13]=1 & SCU500[7]=1	RMII2TXEN	SCU400[13]=1 & SCU500[7]=0			GPIO18B5
C1	GPIO18B6	RGMII2TXD0	SCU400[14]=1 & SCU500[7]=1	RMII2TXD0	SCU400[14]=1 & SCU500[7]=0			GPIO18B6
D3	GPIO18B7	RGMII2TXD1	SCU400[15]=1 & SCU500[7]=1	RMII2TXD1	SCU400[15]=1 & SCU500[7]=0			GPIO18B7
E4	GPIO18C0	RGMII2TXD2	SCU400[16]=1 & SCU500[7]=1					GPIO18C0
F5	GPIO18C1	RGMII2TXD3	SCU400[17]=1 & SCU500[7]=1					GPIO18C1
D2	GPIO18C2	RGMII2RXCK	SCU400[18]=1 & SCU500[7]=1	RMII2RCLKI	SCU400[18]=1 & SCU500[7]=0			GPIO18C2
E3	GPIO18C3	RGMII2RXCTL	SCU400[19]=1 & SCU500[7]=1					GPIO18C3
D1	GPIO18C4	RGMII2RXD0	SCU400[20]=1 & SCU500[7]=1	RMII2RXD0	SCU400[20]=1 & SCU500[7]=0			GPIO18C4
F4	GPIO18C5	RGMII2RXD1	SCU400[21]=1 & SCU500[7]=1	RMII2RXD1	SCU400[21]=1 & SCU500[7]=0			GPIO18C5
E2	GPIO18C6	RGMII2RXD2	SCU400[22]=1 & SCU500[7]=1	RMII2CRSDV	SCU400[22]=1 & SCU500[7]=0			GPIO18C6
E1	GPIO18C7	RGMII2RXD3	SCU400[23]=1 & SCU500[7]=1	RMII2RXER	SCU400[23]=1 & SCU500[7]=0			GPIO18C7

to next page

from previous page

Ball	Default	Function 1	Control 1	Function 2	Control 2	Function 3	Control 3	Others
AB4	GPIO18D0	EMMCCLK	SCU400[24]=1					GPIO18D0
AA4	GPIO18D1	EMMCCMD	SCU400[25]=1					GPIO18D1
AC4	GPIO18D2	EMMCDAT0	SCU400[26]=1					GPIO18D2
AA5	GPIO18D3	EMMCDAT1	SCU400[27]=1					GPIO18D3
Y5	GPIO18D4	EMMCDAT2	SCU400[28]=1					GPIO18D4
AB5	GPIO18D5	EMMCDAT3	SCU400[29]=1					GPIO18D5
AB6	GPIO18D6	EMMCCD#	SCU400[30]=1					GPIO18D6
AC5	GPIO18D7	EMMCWP#	SCU400[31]=1					GPIO18D7
Y1	GPIO18E0	FWSPI18CS#	SCU500[3]=1	VBCS#	SCU500[5]=1	EMMCDAT4	SCU404[0]=1	GPIO18E0
Y2	GPIO18E1	FWSPI18CK	SCU500[3]=1	VBCK	SCU500[5]=1	EMMCDAT5	SCU404[1]=1	GPIO18E1
Y3	GPIO18E2	FWSPI18MOSI	SCU500[3]=1	VBMOSI	SCU500[5]=1	EMMCDAT6	SCU404[2]=1	GPIO18E2
Y4	GPIO18E3	FWSPI18MISO	SCU500[3]=1	VBMISO	SCU500[5]=1	EMMCDAT7	SCU404[3]=1	GPIO18E3

5.2 Function 4

Ball	Default	Function 4	Control 4	Function 5	Control 5	Reserved 2	Reserved-CTRL 2	Others
M24	GPIOA0			SCLS3	SCU710[0]=1			GPIOA4
M25	GPIOA1			SDAS3	SCU710[1]=1			GPIOA5
L26	GPIOA2			SCLS4	SCU710[2]=1			GPIOA6
K24	GPIOA3			SDAS4	SCU710[3]=1			GPIOA7
K26	GPIOA4	SGPM2CK	SCU6D0[4]=1	SCLS1	SCU710[4]=1			GPIOA4
L24	GPIOA5	SGPM2LD	SCU6D0[5]=1	SDAS1	SCU710[5]=1			GPIOA5
L23	GPIOA6	SGPM2O	SCU6D0[6]=1	SCLS2	SCU710[6]=1			GPIOA6
K25	GPIOA7	SGPM2I	SCU6D0[7]=1	SDAS2	SCU710[7]=1			GPIOA7
E21	GPIOG0	SATLS1	SCU6D4[16]=1					GPIOG0
B22	GPIOG1	SATLS2	SCU6D4[17]=1					GPIOG1
C21	GPIOG2	SATLS3	SCU6D4[18]=1					GPIOG2
A22	GPIOG3	SATLS4	SCU6D4[19]=1					GPIOG3
A21	GPIOG4	SATLS1	SCU6D4[20]=1					GPIOG4
E20	GPIOG5	SATLS2	SCU6D4[21]=1					GPIOG5
D21	GPIOG6	SATLS3	SCU6D4[22]=1					GPIOG6
B21	GPIOG7	SATLS4	SCU6D4[23]=1					GPIOG7

5.3 Pin Function Configuration

Ball	Name	I/O	Type	Power-Up	Multi-function Control
LPC Slave Interface					
AB7	LAD0	I/O8	Bidir	Hi-Z,Input	SCU434[16]=1 & SCU510[6]=1
AB8	LAD1	I/O8	Bidir	Hi-Z,Input	SCU434[17]=1 & SCU510[6]=1
AC8	LAD2	I/O8	Bidir	Hi-Z,Input	SCU434[18]=1 & SCU510[6]=1
AC7	LAD3	I/O8	Bidir	Hi-Z,Input	SCU434[19]=1 & SCU510[6]=1
AE7	LCLK	I	In	Input	SCU434[20]=1 & SCU510[6]=1
AF7	LFRAME#	I	In	Input	SCU434[21]=1 & SCU510[6]=1
AD7	LSIRQ	I/O8	Bidir	Hi-Z,Input	SCU434[22]=1 & SCU510[6]=1
AD8	LPCRST#	I	In	Input	SCU434[23]=1 & SCU510[6]=1
AE14	LPCPD#	ID/O8	In	Hi-Z,Input	SCU434[13]=1
AF15	LPCPME#	ID/O8	Bidir	Hi-Z,Input	SCU434[14]=1
AD15	LPCSMI#	ID/O8	Out	Hi-Z,Input	SCU434[15]=1
SuperIO ACPI Interface					
AB15	SIOS3#	ID	In	Hi-Z,Input	SCU434[8]=1
AF14	SIOS5#	ID	In	Hi-Z,Input	SCU434[9]=1
AD14	SIOPWREQ#	O8	Out	Hi-Z	SCU434[10]=1
AC15	SIOONCTRL#	ID/O8	Bidir	Hi-Z,Input	SCU434[11]=1
B16	SIOPBI#	ID	In	Hi-Z,Input	SCU418[6]=1
AE15	SIOPWRGD	ID	In	Hi-Z,Input	SCU434[12]=1

Ball	Name	I/O	Type	Power-Up	Multi-function Control
E16	SIOPBO#	ID/O8	Bidir	Hi-Z,Input	SCU418[5]=1
A15	SIOSCI#	O8	Out	Hi-Z	SCU4B8[7]=1
eSPI Interface					
AB7	ESPID0	I/O8	Bidir	Hi-Z,Input	SCU434[16]=1 & SCU510[6]=0
AB8	ESPID1	I/O8	Bidir	Hi-Z,Input	SCU434[17]=1 & SCU510[6]=0
AC8	ESPID2	I/O8	Bidir	Hi-Z,Input	SCU434[18]=1 & SCU510[6]=0
AC7	ESPID3	I/O8	Bidir	Hi-Z,Input	SCU434[19]=1 & SCU510[6]=0
AE7	ESPICK	I	In	Input	SCU434[20]=1 & SCU510[6]=0
AF7	ESPICS#	I	In	Input	SCU434[21]=1 & SCU510[6]=0
AD7	ESPIALT#	I/O8	Bidir	Hi-Z,Input	SCU434[22]=1 & SCU510[6]=0
AD8	ESPIRST#	I	In	Input	SCU434[23]=1 & SCU510[6]=0
Firmware SPI Interface					
AE12	FWSPIDQ2	ID/O8	Bidir	Hi-Z	SCU438[4]=1
AF12	FWSPIDQ3	ID/O8	Bidir	Hi-Z	SCU438[5]=1
AC12	FWSPIABR	ID	In	Hi-Z	SCU438[6]=1
AB12	FWSPIWP#	IS	In	Hi-Z	SCU438[7]=1
Y1	FWSPI18CS#	O8	Out	Hi-Z	SCU500[3]=1
Y2	FWSPI18CK	O8	Out	Hi-Z	SCU500[3]=1
Y3	FWSPI18MOSI	ID/O8	Bidir	Hi-Z	SCU500[3]=1
Y4	FWSPI18MISO	ID/O8	Bidir	Hi-Z	SCU500[3]=1
SPI1 Interface					
AB11	SPI1CK	O8	Out	Hi-Z	SCU438[11]=1
AC10	SPI1CS1#	O8	Out	Hi-Z	SCU438[8]=1
AC11	SPI1MOSI	ID/O8	Out	Hi-Z	SCU438[12]=1
AA11	SPI1MISO	ID/O8	In	Hi-Z	SCU438[13]=1
AD11	SPI1DQ2	ID/O8	Bidir	Hi-Z	SCU438[14]=1
AF10	SPI1DQ3	ID/O8	Bidir	Hi-Z	SCU438[15]=1
AD10	SPI1ABR	ID	Bidir	Hi-Z	SCU438[9]=1
AE10	SPI1WP#	IS	In	Hi-Z	SCU438[10]=1
SPI2 Interface					
AE8	SPI2CS0#	O8	Out	Hi-Z	SCU434[24]=1
AA9	SPI2CS1#	O8	Out	Hi-Z	SCU434[25]=1
AC9	SPI2CS2#	O8	Out	Hi-Z	SCU434[26]=1
AF8	SPI2CK	O8	Out	Hi-Z	SCU434[27]=1
AB9	SPI2MOSI	ID/O8	Bidir	Hi-Z	SCU434[28]=1
AD9	SPI2MISO	ID/O8	Bidir	Hi-Z	SCU434[29]=1
AF9	SPI2DQ2	ID/O8	Bidir	Hi-Z	SCU434[30]=1
AB10	SPI2DQ3	ID/O8	Bidir	Hi-Z	SCU434[31]=1
SD/SDIO/eMMC Interface					
D22	SD1CLK	Op	Out	Hi-Z	SCU414[8]=1
E22	SD1CMD	I/Op	Bidir	Hi-Z	SCU414[9]=1
D22	SD1DAT0	I/Op	Bidir	Hi-Z	SCU414[10]=1
C23	SD1DAT1	I/Op	Bidir	Hi-Z	SCU414[11]=1

Ball	Name	I/O	Type	Power-Up	Multi-function Control
C22	SD1DAT2	I/Op	Bidir	Hi-Z	SCU414[12]=1
A25	SD1DAT3	I/Op	Bidir	Hi-Z	SCU414[13]=1
A24	SD1CD#	I	In	Input	SCU414[14]=1
A23	SD1WP#	I	In	Input	SCU414[15]=1
E21	SD2CLK	Op	Out	Hi-Z	SCU414[16]=0 & SCU4B4[16]=1 & SCU450[1]=1
B22	SD2CMD	I/Op	Bidir	Hi-Z	SCU414[17]=0 & SCU4B4[17]=1 & SCU450[1]=1
C21	SD2DAT0	I/Op	Bidir	Hi-Z	SCU414[18]=0 & SCU4B4[18]=1 & SCU450[1]=1
A22	SD2DAT1	I/Op	Bidir	Hi-Z	SCU414[19]=0 & SCU4B4[19]=1 & SCU450[1]=1
A21	SD2DAT2	I/Op	Bidir	Hi-Z	SCU414[20]=0 & SCU4B4[20]=1 & SCU450[1]=1
E20	SD2DAT3	I/Op	Bidir	Hi-Z	SCU414[21]=0 & SCU4B4[21]=1 & SCU450[1]=1
D21	SD2CD#	I	In	Input	SCU414[22]=0 & SCU4B4[22]=1 & SCU450[1]=1
B21	SD2WP#	I	In	Input	SCU414[23]=0 & SCU4B4[23]=1 & SCU450[1]=1
AB4	EMMCCLK	Op	Out	Hi-Z	SCU400[24]=1
AA4	EMMCCMD	I/Op	Bidir	Hi-Z	SCU400[25]=1
AC4	EMMCDAT0	I/Op	Bidir	Hi-Z	SCU400[26]=1
AA5	EMMCDAT1	I/Op	Bidir	Hi-Z	SCU400[27]=1
Y5	EMMCDAT2	I/Op	Bidir	Hi-Z	SCU400[28]=1
AB5	EMMCDAT3	I/Op	Bidir	Hi-Z	SCU400[29]=1
Y1	EMMCDAT4	I/Op	Bidir	Hi-Z	SCU500[3]=0 & SCU500[5]=0 & SCU404[0]=1
Y2	EMMCDAT5	I/Op	Bidir	Hi-Z	SCU500[3]=0 & SCU500[5]=0 & SCU404[1]=1
Y3	EMMCDAT6	I/Op	Bidir	Hi-Z	SCU500[3]=0 & SCU500[5]=0 & SCU404[2]=1
Y4	EMMCDAT7	I/Op	Bidir	Hi-Z	SCU500[3]=0 & SCU500[5]=0 & SCU404[3]=1
AC5	EMMCCD#	I	In	Input	SCU400[30]=1
AB6	EMMCWP#	I	In	Input	SCU400[31]=1
JTAG Master Interface					
D17	MNTRST2	O8	Out	Hi-Z	SCU418[0]=1
A16	MTDI2	O8	Out	Hi-Z	SCU418[1]=1
E17	MTCK2	O8	Out	Hi-Z	SCU418[2]=1
D16	MTMS2	O8	Out	Hi-Z	SCU418[3]=1
C16	MTDO2	ID	In	Hi-Z	SCU418[4]=1
MII Interface					
R23	MDC1	O8	Out	Hi-Z	SCU430[16]=1
T25	MDIO1	ID/O8	Bidir	Hi-Z	SCU430[17]=1
J23	MDC2	O8	Out	Hi-Z	SCU410[12]=1
G26	MDIO2	ID/O8	Bidir	Hi-Z	SCU410[13]=1
M24	MDC3	O16	Out	Hi-Z	SCU410[0]=1
M25	MDIO3	IS/O16	Bidir	Hi-Z	SCU410[1]=1
L26	MDC4	O16	Out	Hi-Z	SCU410[2]=1
K24	MDIO4	IS/O16	Bidir	Hi-Z	SCU410[3]=1
RGMII1 Interface					
B3	RGMII1RXCK	I	In	Hi-Z,Input	SCU400[6]=1 & SCU500[6]=1
A2	RGMII1RXCTL	I	In	Hi-Z,Input	SCU400[7]=1 & SCU500[6]=1
B2	RGMII1RXD0	I	In	Hi-Z,Input	SCU400[8]=1 & SCU500[6]=1

Ball	Name	I/O	Type	Power-Up	Multi-function Control
B1	RGMII1RXD1	I	In	Hi-Z,Input	SCU400[9]=1 & SCU500[6]=1
C4	RGMII1RXD2	I	In	Hi-Z,Input	SCU400[10]=1 & SCU500[6]=1
E5	RGMII1RXD3	I	In	Hi-Z,Input	SCU400[11]=1 & SCU500[6]=1
C6	RGMII1TXCK	O8	Out	Hi-Z	SCU400[0]=1 & SCU500[6]=1
D6	RGMII1TXCTL	O8	Out	Hi-Z	SCU400[1]=1 & SCU500[6]=1
D5	RGMII1TXD0	O8	Out	Hi-Z	SCU400[2]=1 & SCU500[6]=1
A3	RGMII1TXD1	O8	Out	Hi-Z	SCU400[3]=1 & SCU500[6]=1
C5	RGMII1TXD2	O8	Out	Hi-Z	SCU400[4]=1 & SCU500[6]=1
E6	RGMII1TXD3	O8	Out	Hi-Z	SCU400[5]=1 & SCU500[6]=1
RGMII2 Interface					
D2	RGMII2RXCK	I	In	Hi-Z,Input	SCU400[18]=1 & SCU500[7]=1
E3	RGMII2RXCTL	I	In	Hi-Z,Input	SCU400[19]=1 & SCU500[7]=1
D1	RGMII2RXD0	I	In	Hi-Z,Input	SCU400[20]=1 & SCU500[7]=1
F4	RGMII2RXD1	I	In	Hi-Z,Input	SCU400[21]=1 & SCU500[7]=1
E2	RGMII2RXD2	I	In	Hi-Z,Input	SCU400[22]=1 & SCU500[7]=1
E1	RGMII2RXD3	I	In	Hi-Z,Input	SCU400[23]=1 & SCU500[7]=1
D4	RGMII2TXCK	O8	Out	Hi-Z	SCU400[12]=1 & SCU500[7]=1
C2	RGMII2TXCTL	O8	Out	Hi-Z	SCU400[13]=1 & SCU500[7]=1
C1	RGMII2TXD0	O8	Out	Hi-Z	SCU400[14]=1 & SCU500[7]=1
D3	RGMII2TXD1	O8	Out	Hi-Z	SCU400[15]=1 & SCU500[7]=1
E4	RGMII2TXD2	O8	Out	Hi-Z	SCU400[16]=1 & SCU500[7]=1
F5	RGMII2TXD3	O8	Out	Hi-Z	SCU400[17]=1 & SCU500[7]=1
RGMII3 Interface					
G23	RGMII3RXCK	I	In	Hi-Z,Input	SCU410[22]=1 & SCU510[0]=1
G24	RGMII3RXCTL	I	In	Hi-Z,Input	SCU410[23]=1 & SCU510[0]=1
F23	RGMII3RXD0	I	In	Hi-Z,Input	SCU410[24]=1 & SCU510[0]=1
F26	RGMII3RXD1	I	In	Hi-Z,Input	SCU410[25]=1 & SCU510[0]=1
F25	RGMII3RXD2	I	In	Hi-Z,Input	SCU410[26]=1 & SCU510[0]=1
E26	RGMII3RXD3	I	In	Hi-Z,Input	SCU410[27]=1 & SCU510[0]=1
H24	RGMII3TXCK	Op	Out	Hi-Z	SCU410[16]=1 & SCU510[0]=1
J22	RGMII3TXCTL	Op	Out	Hi-Z	SCU410[17]=1 & SCU510[0]=1
H22	RGMII3TXD0	Op	Out	Hi-Z	SCU410[18]=1 & SCU510[0]=1
H23	RGMII3TXD1	Op	Out	Hi-Z	SCU410[19]=1 & SCU510[0]=1
G22	RGMII3TXD2	Op	Out	Hi-Z	SCU410[20]=1 & SCU510[0]=1
F22	RGMII3TXD3	Op	Out	Hi-Z	SCU410[21]=1 & SCU510[0]=1
RGMII4 Interface					
C25	RGMII4RXCK	I	In	Hi-Z,Input	SCU414[2]=0 & SCU4B4[2]=1 & SCU510[1]=1
C26	RGMII4RXCTL	I	In	Hi-Z,Input	SCU414[3]=0 & SCU4B4[3]=1 & SCU510[1]=1
C24	RGMII4RXD0	I	In	Hi-Z,Input	SCU414[4]=0 & SCU4B4[4]=1 & SCU510[1]=1
B26	RGMII4RXD1	I	In	Hi-Z,Input	SCU414[5]=0 & SCU4B4[5]=1 & SCU510[1]=1
B25	RGMII4RXD2	I	In	Hi-Z,Input	SCU414[6]=0 & SCU4B4[6]=1 & SCU510[1]=1
B24	RGMII4RXD3	I	In	Hi-Z,Input	SCU414[7]=0 & SCU4B4[7]=1 & SCU510[1]=1
F24	RGMII4TXCK	Op	Out	Hi-Z	SCU410[28]=0 & SCU4B0[28]=1 & SCU510[1]=1

Ball	Name	I/O	Type	Power-Up	Multi-function Control
E23	RGMII4TXCTL	Op	Out	Hi-Z	SCU410[29]=0 & SCU4B0[29]=1 & SCU510[1]=1
E24	RGMII4TXD0	Op	Out	Hi-Z	SCU410[30]=0 & SCU4B0[30]=1 & SCU510[1]=1
E25	RGMII4TXD1	Op	Out	Hi-Z	SCU410[31]=0 & SCU4B0[31]=1 & SCU510[1]=1
D26	RGMII4TXD2	Op	Out	Hi-Z	SCU414[0]=0 & SCU4B4[0]=1 & SCU510[1]=1
D24	RGMII4TXD3	Op	Out	Hi-Z	SCU414[1]=0 & SCU4B4[1]=1 & SCU510[1]=1
RMII1 Interface					
B3	RMII1RCLKI	I	In	Hi-Z,Input	SCU400[6]=1 & SCU500[6]=0
B2	RMII1RXD0	I	In	Hi-Z,Input	SCU400[8]=1 & SCU500[6]=0
B1	RMII1RXD1	I	In	Hi-Z,Input	SCU400[9]=1 & SCU500[6]=0
C4	RMII1CRSDV	I	In	Hi-Z,Input	SCU400[10]=1 & SCU500[6]=0
E5	RMII1RXER	I	In	Hi-Z,Input	SCU400[11]=1 & SCU500[6]=0
C6	RMII1RCLKO	O8	Out	Hi-Z	SCU400[0]=1 & SCU500[6]=0
D6	RMII1TXEN	O8	Out	Hi-Z	SCU400[1]=1 & SCU500[6]=0
D5	RMII1TXD0	O8	Out	Hi-Z	SCU400[2]=1 & SCU500[6]=0
A3	RMII1TXD1	O8	Out	Hi-Z	SCU400[3]=1 & SCU500[6]=0
RMII2 Interface					
D2	RMII2RCLKI	I	In	Hi-Z,Input	SCU400[18]=1 & SCU500[7]=1
D1	RMII2RXD0	I	In	Hi-Z,Input	SCU400[20]=1 & SCU500[7]=1
F4	RMII2RXD1	I	In	Hi-Z,Input	SCU400[21]=1 & SCU500[7]=1
E2	RMII2CRSDV	I	In	Hi-Z,Input	SCU400[22]=1 & SCU500[7]=1
E1	RMII2RXER	I	In	Hi-Z,Input	SCU400[23]=1 & SCU500[7]=1
D4	RMII2RCLKO	O8	Out	Hi-Z,	SCU400[12]=1 & SCU500[7]=0
C2	RMII2TXEN	O8	Out	Hi-Z,	SCU400[13]=1 & SCU500[7]=0
C1	RMII2TXD0	O8	Out	Hi-Z,	SCU400[14]=1 & SCU500[7]=0
D3	RMII2TXD1	O8	Out	Hi-Z,	SCU400[15]=1 & SCU500[7]=0
RMII3 Interface					
G23	RMII3RCLKI	I	In	Hi-Z,Input	SCU410[22]=1 & SCU510[0]=0
F23	RMII3RXD0	I	In	Hi-Z,Input	SCU410[24]=1 & SCU510[0]=0
F26	RMII3RXD1	I	In	Hi-Z,Input	SCU410[25]=1 & SCU510[0]=0
F25	RMII3CRSDV	I	In	Hi-Z,Input	SCU410[26]=1 & SCU510[0]=0
E26	RMII3RXER	I	In	Hi-Z,Input	SCU410[27]=1 & SCU510[0]=0
H24	RMII3RCLKO	Op	Out	Hi-Z,	SCU410[16]=1 & SCU510[0]=0
J22	RMII3TXEN	Op	Out	Hi-Z,	SCU410[17]=1 & SCU510[0]=0
H22	RMII3TXD0	Op	Out	Hi-Z,	SCU410[18]=1 & SCU510[0]=0
H23	RMII3TXD1	Op	Out	Hi-Z,	SCU410[19]=1 & SCU510[0]=0
RMII4 Interface					
C25	RMII4RCLKI	I	In	Hi-Z,Input	SCU414[2]=0 & SCU4B4[2]=1 & SCU510[1]=0
C24	RMII4RXD0	I	In	Hi-Z,Input	SCU414[4]=0 & SCU4B4[4]=1 & SCU510[1]=0
B26	RMII4RXD1	I	In	Hi-Z,Input	SCU414[5]=0 & SCU4B4[5]=1 & SCU510[1]=0
B25	RMII4CRSDV	I	In	Hi-Z,Input	SCU414[6]=0 & SCU4B4[6]=1 & SCU510[1]=0
B24	RMII4RXER	I	In	Hi-Z,Input	SCU414[7]=0 & SCU4B4[7]=1 & SCU510[1]=0
F24	RMII4RCLKO	Op	Out	Hi-Z,	SCU410[28]=0 & SCU4B0[28]=1 & SCU510[1]=0
E23	RMII4TXEN	Op	Out	Hi-Z,	SCU410[29]=0 & SCU4B0[29]=1 & SCU510[1]=0

Ball	Name	I/O	Type	Power-Up	Multi-function Control
E24	RMII4TXD0	Op	Out	Hi-Z,	SCU410[30]=0 & SCU4B0[30]=1 & SCU510[1]=0
E25	RMII4TXD1	Op	Out	Hi-Z,	SCU410[31]=0 & SCU4B0[31]=1 & SCU510[1]=0
UART Interface					
D14	NCTS1	ID	In	Hi-Z	SCU41C[0]=1
B13	NDCD1	ID	In	Hi-Z	SCU41C[1]=1
A12	NDSR1	ID	In	Hi-Z	SCU41C[2]=1
E14	NRI1	ID	In	Hi-Z	SCU41C[3]=1
B12	NDTR1	O8	Out	Hi-Z	SCU41C[4]=1
C12	NRTS1	O8	Out	Hi-Z	SCU41C[5]=1
C13	TXD1	O8	Out	Hi-Z	SCU41C[6]=1
D13	RXD1	ID	In	Hi-Z	SCU41C[7]=1
P25	NCTS2	ID	In	Hi-Z	SCU41C[8]=1
N23	NDCD2	ID	In	Hi-Z	SCU41C[9]=1
N25	NDSR2	ID	In	Hi-Z	SCU41C[10]=1
N24	NRI2	ID	In	Hi-Z	SCU41C[11]=1
P26	NDTR2	O8	Out	Hi-Z	SCU41C[12]=1
M23	NRTS2	O8	Out	Hi-Z	SCU41C[13]=1
N26	TXD2	O8	Out	Hi-Z	SCU41C[14]=1
M26	RXD2	ID	In	Hi-Z	SCU41C[15]=1
F24	NCTS3	ID	In	Hi-Z	SCU410[28]=1
E23	NDCD3	ID	In	Hi-Z	SCU410[29]=1
E24	NDSR3	ID	In	Hi-Z	SCU410[30]=1
E25	NRI3	ID	In	Hi-Z	SCU410[31]=1
D26	NDTR3	O8	Out	Hi-Z	SCU414[0]=1
D24	NRTS3	O8	Out	Hi-Z	SCU414[1]=1
C15	TXD3	O8	Out	Hi-Z	SCU418[28]=1
F15	RXD3	ID	In	Hi-Z	SCU418[29]=1
C25	NCTS4	ID	In	Hi-Z	SCU414[2]=1
C26	NDCD4	ID	In	Hi-Z	SCU414[3]=1
C24	NDSR4	ID	In	Hi-Z	SCU414[4]=1
B26	NRI4	ID	In	Hi-Z	SCU414[5]=1
B25	NDTR4	O8	Out	Hi-Z	SCU414[6]=1
B24	NRTS4	O8	Out	Hi-Z	SCU414[7]=1
H25	TXD4	O8	Out	Hi-Z	SCU410[14]=1
J24	RXD4	ID	In	Hi-Z	SCU410[15]=1
E21	TXD6	O8	Out	Hi-Z	SCU414[16]=1
B22	RXD6	ID	In	Hi-Z	SCU414[17]=1
C21	TXD7	O8	Out	Hi-Z	SCU414[18]=1
A22	RXD7	ID	In	Hi-Z	SCU414[19]=1
A21	TXD8	O8	Out	Hi-Z	SCU414[20]=1
E20	RXD8	ID	In	Hi-Z	SCU414[21]=1
D21	TXD9	O8	Out	Hi-Z	SCU414[22]=1
B21	RXD9	ID	In	Hi-Z	SCU414[23]=1

Ball	Name	I/O	Type	Power-Up	Multi-function Control
R26	TXD10	O8	Out	Hi-Z	SCU430[20]=1
P24	RXD10	ID	In	Hi-Z	SCU430[21]=1
P23	TXD11	O8	Out	Hi-Z	SCU430[22]=1
T24	RXD11	ID	In	Hi-Z	SCU430[23]=1
D17	TXD12	O8	Out	Hi-Z	SCU4B8[0]=1 & SCU418[0]=0
AF9	TXD12	O8	Out	Hi-Z	SCU4B8[0]=0 & SCU434[30]=0 & SCU4D4[30]=1
A16	RXD12	ID	In	Hi-Z	SCU4B8[1]=1 & SCU418[1]=0
AB10	RXD12	ID	In	Hi-Z	SCU4B8[1]=0 & SCU434[31]=0 & SCU4D4[31]=1
E17	TXD13	O8	Out	Hi-Z	SCU4B8[2]=1 & SCU418[2]=0
AD11	TXD13	O8	Out	Hi-Z	SCU4B8[2]=0 & SCU438[14]=0 & SCU4D8[14]=1
D16	RXD13	ID	In	Hi-Z	SCU4B8[3]=1 & SCU418[3]=0
AF10	RXD13	ID	In	Hi-Z	SCU4B8[3]=0 & SCU438[15]=0 & SCU4D8[15]=1
VGA Interface					
B14	VGAHS	O8	Out	Hi-Z	SCU418[30]=1
C14	VGAVS	O8	Out	Hi-Z	SCU418[31]=1
I2C Interface					
B20	SCL1	IU/O12	Bidir	Hi-Z	SCU418[8]=0 & SCU4B8[8]=1
A20	SDA1	IU/O12	Bidir	Hi-Z	SCU418[9]=0 & SCU4B8[9]=1
E19	SCL2	IU/O12	Bidir	Hi-Z	SCU418[10]=0 & SCU4B8[10]=1
D20	SDA2	IU/O12	Bidir	Hi-Z	SCU418[11]=0 & SCU4B8[11]=1
C19	SCL3	IU/O12	Bidir	Hi-Z	SCU418[12]=0 & SCU4B8[12]=1
A19	SDA3	IU/O12	Bidir	Hi-Z	SCU418[13]=0 & SCU4B8[13]=1
C20	SCL4	IU/O12	Bidir	Hi-Z	SCU418[14]=0 & SCU4B8[14]=1
D19	SDA4	IU/O12	Bidir	Hi-Z	SCU418[15]=0 & SCU4B8[15]=1
A11	SCL5	IS/O8	Bidir	Hi-Z	SCU418[16]=1
C11	SDA5	IS/O8	Bidir	Hi-Z	SCU418[17]=1
D12	SCL6	IS/O8	Bidir	Hi-Z	SCU418[18]=1
E13	SDA6	IS/O8	Bidir	Hi-Z	SCU418[19]=1
D11	SCL7	IS/O8	Bidir	Hi-Z	SCU418[20]=1
E11	SDA7	IS/O8	Bidir	Hi-Z	SCU418[21]=1
F13	SCL8	IS/O8	Bidir	Hi-Z	SCU418[22]=1
E12	SDA8	IS/O8	Bidir	Hi-Z	SCU418[23]=1
D15	SCL9	IS/O8	Bidir	Hi-Z	SCU418[24]=1
A14	SDA9	IS/O8	Bidir	Hi-Z	SCU418[25]=1
E15	SCL10	IS/O8	Bidir	Hi-Z	SCU418[26]=1
A13	SDA10	IS/O8	Bidir	Hi-Z	SCU418[27]=1
M24	SCL11	IS/O8	Bidir	Hi-Z	SCU410[0]=0 & SCU4B0[0]=1
M25	SDA11	IS/O8	Bidir	Hi-Z	SCU410[1]=0 & SCU4B0[1]=1
L26	SCL12	IS/O8	Bidir	Hi-Z	SCU410[2]=0 & SCU4B0[2]=1
K24	SDA12	IS/O8	Bidir	Hi-Z	SCU410[3]=0 & SCU4B0[3]=1
K26	SCL13	IS/O8	Bidir	Hi-Z	SCU410[4]=0 & SCU4B0[4]=1
L24	SDA13	IS/O8	Bidir	Hi-Z	SCU410[5]=0 & SCU4B0[5]=1
L23	SCL14	IS/O8	Bidir	Hi-Z	SCU410[6]=0 & SCU4B0[6]=1

Ball	Name	I/O	Type	Power-Up	Multi-function Control
K25	SDA14	IS/O8	Bidir	Hi-Z	SCU410[7]=0 & SCU4B0[7]=1
D18	SCL15	ID/O8	Bidir	Hi-Z	SCU414[28]=0 & SCU4B4[28]=1
B17	SDA15	ID/O8	Bidir	Hi-Z	SCU414[29]=0 & SCU4B4[29]=1
C17	SCL16	ID/O8	Bidir	Hi-Z	SCU414[30]=0 & SCU4B4[30]=1
E18	SDA16	ID/O8	Bidir	Hi-Z	SCU414[31]=0 & SCU4B4[31]=1
J26	SALT1	ID/O8	Bidir	Hi-Z	SCU410[8]=1
K23	SALT2	ID/O8	Bidir	Hi-Z	SCU410[9]=1
H26	SALT3	ID/O8	Bidir	Hi-Z	SCU410[10]=1
J25	SALT4	ID/O8	Bidir	Hi-Z	SCU410[11]=1
AF11	SALT5	ID/O8	Bidir	Hi-Z	SCU438[0]=1
AD12	SALT6	ID/O8	Bidir	Hi-Z	SCU438[1]=1
AE11	SALT7	ID/O8	Bidir	Hi-Z	SCU438[2]=1
AA12	SALT8	ID/O8	Bidir	Hi-Z	SCU438[3]=1
E21	SALT9	I/Op	Bidir	Hi-Z	SCU414[16]=0 & SCU4B4[16]=0 & SCU694[16]=1
AB16	SALT9	I	In	Hi-Z	SCU434[0]=1 & SCU694[16]=0 & SCU4D4[0]=1
B22	SALT10	I/Op	Bidir	Hi-Z	SCU414[17]=0 & SCU4B4[17]=0 & SCU694[17]=1
AA17	SALT10	I	In	Hi-Z	SCU434[1]=1 & SCU694[17]=0 & SCU4D4[1]=1
C21	SALT11	I/Op	Bidir	Hi-Z	SCU414[18]=0 & SCU4B4[18]=0 & SCU694[18]=1
AB17	SALT11	I	In	Hi-Z	SCU434[2]=1 & SCU694[18]=0 & SCU4D4[2]=1
A22	SALT12	I/Op	Bidir	Hi-Z	SCU414[19]=0 & SCU4B4[19]=0 & SCU694[19]=1
AE16	SALT12	I	In	Hi-Z	SCU434[3]=1 & SCU694[19]=0 & SCU4D4[3]=1
A21	SALT13	I/Op	Bidir	Hi-Z	SCU414[20]=0 & SCU4B4[20]=0 & SCU694[20]=1
AC16	SALT13	I	In	Hi-Z	SCU434[4]=1 & SCU694[20]=0 & SCU4D4[4]=1
E20	SALT14	I/Op	Bidir	Hi-Z	SCU414[21]=0 & SCU4B4[21]=0 & SCU694[21]=1
AA16	SALT14	I	In	Hi-Z	SCU434[5]=1 & SCU694[21]=0 & SCU4D4[5]=1
D21	SALT15	I/Op	Bidir	Hi-Z	SCU414[22]=0 & SCU4B4[22]=0 & SCU694[22]=1
AD16	SALT15	I	In	Hi-Z	SCU434[6]=1 & SCU694[22]=0 & SCU4D4[6]=1
B21	SALT16	I/Op	Bidir	Hi-Z	SCU414[23]=0 & SCU4B4[23]=0 & SCU694[23]=1
AC17	SALT16	I	In	Hi-Z	SCU434[7]=1 & SCU694[23]=0 & SCU4D4[7]=1
I3C LV Interface					
AF23	I3C1SCL	IU/O8	Bidir	Hi-Z	SCU438[16]=1
AE24	I3C1SDA	IU/O8	Bidir	Hi-Z	SCU438[17]=1
AF22	I3C2SCL	IU/O8	Bidir	Hi-Z	SCU438[18]=1
AE22	I3C2SDA	IU/O8	Bidir	Hi-Z	SCU438[19]=1
AF25	I3C3SCL	IU/O8	Bidir	Hi-Z	SCU438[20]=1
AE26	I3C3SDA	IU/O8	Bidir	Hi-Z	SCU438[21]=1
AE25	I3C4SCL	IU/O8	Bidir	Hi-Z	SCU438[22]=1
AF24	I3C4SDA	IU/O8	Bidir	Hi-Z	SCU438[23]=1
I3C HV Interface					
B20	HVI3C3SCL	IU/O12	Bidir	Hi-Z	SCU418[8]=1
A20	HVI3C3SDA	IU/O12	Bidir	Hi-Z	SCU418[9]=1
E19	HVI3C4SCL	IU/O12	Bidir	Hi-Z	SCU418[10]=1
D20	HVI3C4SDA	IU/O12	Bidir	Hi-Z	SCU418[11]=1

Ball	Name	I/O	Type	Power-Up	Multi-function Control
C19	HVI3C5SCL	IU/O12	Bidir	Hi-Z	SCU418[12]=1
A19	HVI3C5SDA	IU/O12	Bidir	Hi-Z	SCU418[13]=1
C20	HVI3C6SCL	IU/O12	Bidir	Hi-Z	SCU418[14]=1
D19	HVI3C6SDA	IU/O12	Bidir	Hi-Z	SCU418[15]=1
FSI Interface					
AF25	FSI1CLK	ID/O8	Bidir	Hi-Z	SCU438[20]=0 & SCU4D8[20]=1
AE26	FSI1DATA	ID/O8	Bidir	Hi-Z	SCU438[21]=0 & SCU4D8[21]=1
AE25	FSI2CLK	ID/O8	Bidir	Hi-Z	SCU438[22]=0 & SCU4D8[22]=1
AF24	FSI2DATA	ID/O8	Bidir	Hi-Z	SCU438[23]=0 & SCU4D8[23]=1
PWM Interface					
AD26	PWM0	O8	Out	Hi-Z	SCU41C[16]=1
AD22	PWM1	O8	Out	Hi-Z	SCU41C[17]=1
AD23	PWM2	O8	Out	Hi-Z	SCU41C[18]=1
AD24	PWM3	O8	Out	Hi-Z	SCU41C[19]=1
AD25	PWM4	O8	Out	Hi-Z	SCU41C[20]=1
AC22	PWM5	O8	Out	Hi-Z	SCU41C[21]=1
AC24	PWM6	O8	Out	Hi-Z	SCU41C[22]=1
AC23	PWM7	O8	Out	Hi-Z	SCU41C[23]=1
AB22	PWM8	O8	Out	Hi-Z	SCU41C[24]=1
D22	PWM8	Op	Out	Hi-Z	SCU414[8]=0 & SCU4B4[8]=1
W24	PWM9	O8	Out	Hi-Z	SCU41C[25]=1
E22	PWM9	Op	Out	Hi-Z	SCU414[9]=0 & SCU4B4[9]=1
AA23	PWM10	O8	Out	Hi-Z	SCU41C[26]=1
D23	PWM10	Op	Out	Hi-Z	SCU414[10]=0 & SCU4B4[10]=1
AA24	PWM11	O8	Out	Hi-Z	SCU41C[27]=1
C23	PWM11	Op	Out	Hi-Z	SCU414[11]=0 & SCU4B4[11]=1
W23	PWM12	O8	Out	Hi-Z	SCU41C[28]=1
C22	PWM12	Op	Out	Hi-Z	SCU414[12]=0 & SCU4B4[12]=1
AB23	PWM13	O8	Out	Hi-Z	SCU41C[29]=1
A25	PWM13	Op	Out	Hi-Z	SCU414[13]=0 & SCU4B4[13]=1
AB24	PWM14	O8	Out	Hi-Z	SCU41C[30]=1
A24	PWM14	Op	Out	Hi-Z	SCU414[14]=0 & SCU4B4[14]=1
Y23	PWM15	O8	Out	Hi-Z	SCU41C[31]=1
A23	PWM15	Op	Out	Hi-Z	SCU414[15]=0 & SCU4B4[15]=1
Tachometer Interface					
AA25	TACH0	IS	In	Hi-Z,Input	SCU430[0]=1
AB25	TACH1	IS	In	Hi-Z,Input	SCU430[1]=1
Y24	TACH2	IS	In	Hi-Z,Input	SCU430[2]=1
AB26	TACH3	IS	In	Hi-Z,Input	SCU430[3]=1
Y26	TACH4	IS	In	Hi-Z,Input	SCU430[4]=1
AC26	TACH5	IS	In	Hi-Z,Input	SCU430[5]=1
Y25	TACH6	IS	In	Hi-Z,Input	SCU430[6]=1
AA26	TACH7	IS	In	Hi-Z,Input	SCU430[7]=1

Ball	Name	I/O	Type	Power-Up	Multi-function Control
V25	TACH8	ID	In	Hi-Z,Input	SCU430[8]=1
U24	TACH9	ID	In	Hi-Z,Input	SCU430[9]=1
V24	TACH10	ID	In	Hi-Z,Input	SCU430[10]=1
V26	TACH11	ID	In	Hi-Z,Input	SCU430[11]=1
U25	TACH12	ID	In	Hi-Z,Input	SCU430[12]=1
T23	TACH13	ID	In	Hi-Z,Input	SCU430[13]=1
W26	TACH14	ID	In	Hi-Z,Input	SCU430[14]=1
U26	TACH15	ID	In	Hi-Z,Input	SCU430[15]=1
ADC Interface					
AD20	ADC0	I	In	Input	SCU430[24]=0
AC18	ADC1	I	In	Input	SCU430[25]=0
AE19	ADC2	I	In	Input	SCU430[26]=0
AD19	ADC3	I	In	Input	SCU430[27]=0
AC19	ADC4	I	In	Input	SCU430[28]=0
AB19	ADC5	I	In	Input	SCU430[29]=0
AB18	ADC6	I	In	Input	SCU430[30]=0
AE18	ADC7	I	In	Input	SCU430[31]=0
AB16	ADC8	I	In	Input	SCU434[0]=0
AA17	ADC9	I	In	Input	SCU434[1]=0
AB17	ADC10	I	In	Input	SCU434[2]=0
AE16	ADC11	I	In	Input	SCU434[3]=0
AC16	ADC12	I	In	Input	SCU434[4]=0
AA16	ADC13	I	In	Input	SCU434[5]=0
AD16	ADC14	I	In	Input	SCU434[6]=0
AC17	ADC15	I	In	Input	SCU434[7]=0
SGPIO Master Interface					
A18	SGPM1CK	O8	Out	Hi-Z	SCU414[24]=1
B18	SGPM1LD	O8	Out	Hi-Z	SCU414[25]=1
C18	SGPM1O	O8	Out	Hi-Z	SCU414[26]=1
A17	SGPM1I	ID	In	Hi-Z	SCU414[27]=1
K26	SGPM2CK	O8	Out	Hi-Z	SCU410[4]=0 & SCU4B0[4]=0 & SCU690[4]=0 & SCU6D0[4]=1
L24	SGPM2LD	O8	Out	Hi-Z	SCU410[5]=0 & SCU4B0[5]=0 & SCU690[5]=0 & SCU6D0[5]=1
L23	SGPM2O	O8	Out	Hi-Z	SCU410[6]=0 & SCU4B0[6]=0 & SCU690[6]=0 & SCU6D0[6]=1
K25	SGPM2I	ID	In	Hi-Z	SCU410[7]=0 & SCU4B0[7]=0 & SCU690[7]=0 & SCU6D0[7]=1
SGPIO Slave Interface					
D18	SGPS1CK	ID	In	Hi-Z	SCU414[28]=1
B17	SGPS1LD	ID	In	Hi-Z	SCU414[29]=1
C17	SGPS1I0	ID	In	Hi-Z	SCU414[30]=1
E18	SGPS1I1	ID	In	Hi-Z	SCU414[31]=1
K26	SGPS2CK	TBD	In	Hi-Z	SCU410[4]=0 & SCU4B0[4]=0 & SCU690[4]=1

Ball	Name	I/O	Type	Power-Up	Multi-function Control
L24	SGPS2LD	TBD	In	Hi-Z	SCU410[5]=0 & SCU4B0[5]=0 & SCU690[5]=1
L23	SGPS2I0	TBD	In	Hi-Z	SCU410[6]=0 & SCU4B0[6]=0 & SCU690[6]=1
K25	SGPS2I1	TBD	In	Hi-Z	SCU410[7]=0 & SCU4B0[7]=0 & SCU690[7]=1
3.3V GPIO Interface					
M24	GPIOA0	IS/O16	Bidir	Hi-Z,Input	SCU410[0]=0 & SCU4B0[0]=0
M25	GPIOA1	IS/O16	Bidir	Hi-Z,Input	SCU410[1]=0 & SCU4B0[1]=0
L26	GPIOA2	IS/O16	Bidir	Hi-Z,Input	SCU410[2]=0 & SCU4B0[2]=0
K24	GPIOA3	IS/O16	Bidir	Hi-Z,Input	SCU410[3]=0 & SCU4B0[3]=0
K26	GPIOA4	IS/O8	Bidir	Hi-Z,Input	SCU410[4]=0 & SCU4B0[4]=0
L24	GPIOA5	IS/O8	Bidir	Hi-Z,Input	SCU410[5]=0 & SCU4B0[5]=0
L23	GPIOA6	IS/O8	Bidir	Hi-Z,Input	SCU410[6]=0 & SCU4B0[6]=0
K25	GPIOA7	IS/O8	Bidir	Hi-Z,Input	SCU410[7]=0 & SCU4B0[7]=0
J26	GPIOB0	ID/O8	Bidir	Hi-Z,Input	SCU410[8]=0 & SCU4B0[8]=0
K23	GPIOB1	ID/O8	Bidir	Hi-Z,Input	SCU410[9]=0 & SCU4B0[9]=0
H26	GPIOB2	ID/O8	Bidir	Hi-Z,Input	SCU410[10]=0 & SCU4B0[10]=0
J25	GPIOB3	ID/O8	Bidir	Hi-Z,Input	SCU410[11]=0 & SCU4B0[11]=0
J23	GPIOB4	ID/O8	Bidir	Hi-Z,Input	SCU410[12]=0 & SCU4B0[12]=0
G26	GPIOB5	ID/O8	Bidir	Hi-Z,Input	SCU410[13]=0 & SCU4B0[13]=0
H25	GPIOB6	ID/O8	Bidir	Hi-Z,Input	SCU410[14]=0 & SCU4B0[14]=0
J24	GPIOB7	ID/O8	Bidir	Hi-Z,Input	SCU410[15]=0 & SCU4B0[15]=0
H24	GPIOC0	I/O8	Bidir	Hi-Z,Input	SCU410[16]=0
J22	GPIOC1	I/O8	Bidir	Hi-Z,Input	SCU410[17]=0
H22	GPIOC2	I/O8	Bidir	Hi-Z,Input	SCU410[18]=0
H23	GPIOC3	I/O8	Bidir	Hi-Z,Input	SCU410[19]=0
G22	GPIOC4	I/O8	Bidir	Hi-Z,Input	SCU410[20]=0
F22	GPIOC5	I/O8	Bidir	Hi-Z,Input	SCU410[21]=0
G23	GPIOC6	I/O8	Bidir	Hi-Z,Input	SCU410[22]=0
G24	GPIOC7	I/O8	Bidir	Hi-Z,Input	SCU410[23]=0
F23	GPIOD0	I/O8	Bidir	Hi-Z,Input	SCU410[24]=0
F26	GPIOD1	I/O8	Bidir	Hi-Z,Input	SCU410[25]=0
F25	GPIOD2	I/O8	Bidir	Hi-Z,Input	SCU410[26]=0
E26	GPIOD3	I/O8	Bidir	Hi-Z,Input	SCU410[27]=0
F24	GPIOD4	I/O8	Bidir	Hi-Z,Input	SCU410[28]=0 & SCU4B0[28]=0
E23	GPIOD5	I/O8	Bidir	Hi-Z,Input	SCU410[29]=0 & SCU4B0[29]=0
E24	GPIOD6	I/O8	Bidir	Hi-Z,Input	SCU410[30]=0 & SCU4B0[30]=0
E25	GPIOD7	I/O8	Bidir	Hi-Z,Input	SCU410[31]=0 & SCU4B0[31]=0
D26	GPIOE0	I/O8	Bidir	Hi-Z,Input	SCU414[0]=0 & SCU4B4[0]=0
D24	GPIOE1	I/O8	Bidir	Hi-Z,Input	SCU414[1]=0 & SCU4B4[1]=0
C25	GPIOE2	I/O8	Bidir	Hi-Z,Input	SCU414[2]=0 & SCU4B4[2]=0
C26	GPIOE3	I/O8	Bidir	Hi-Z,Input	SCU414[3]=0 & SCU4B4[3]=0
C24	GPIOE4	I/O8	Bidir	Hi-Z,Input	SCU414[4]=0 & SCU4B4[4]=0
B26	GPIOE5	I/O8	Bidir	Hi-Z,Input	SCU414[5]=0 & SCU4B4[5]=0
B25	GPIOE6	I/O8	Bidir	Hi-Z,Input	SCU414[6]=0 & SCU4B4[6]=0

Ball	Name	I/O	Type	Power-Up	Multi-function Control
B24	GPIOE7	I/O8	Bidir	Hi-Z,Input	SCU414[7]=0 & SCU4B4[7]=0
D22	GPIOF0	I/Op	Bidir	Hi-Z,Input	SCU414[8]=0 & SCU4B4[8]=0
E22	GPIOF1	I/Op	Bidir	Hi-Z,Input	SCU414[9]=0 & SCU4B4[9]=0
D23	GPIOF2	I/Op	Bidir	Hi-Z,Input	SCU414[10]=0 & SCU4B4[10]=0
C23	GPIOF3	I/Op	Bidir	Hi-Z,Input	SCU414[11]=0 & SCU4B4[11]=0
C22	GPIOF4	I/Op	Bidir	Hi-Z,Input	SCU414[12]=0 & SCU4B4[12]=0
A25	GPIOF5	I/Op	Bidir	Hi-Z,Input	SCU414[13]=0 & SCU4B4[13]=0
A24	GPIOF6	I/Op	Bidir	Hi-Z,Input	SCU414[14]=0 & SCU4B4[14]=0
A23	GPIOF7	I/Op	Bidir	Hi-Z,Input	SCU414[15]=0 & SCU4B4[15]=0
E21	GPIOG0	I/Op	Bidir	Hi-Z,Input	SCU414[16]=0 & SCU4B4[16]=0 & SCU694[16]=0
B22	GPIOG1	I/Op	Bidir	Hi-Z,Input	SCU414[17]=0 & SCU4B4[17]=0 & SCU694[17]=0
C21	GPIOG2	I/Op	Bidir	Hi-Z,Input	SCU414[18]=0 & SCU4B4[18]=0 & SCU694[18]=0
A22	GPIOG3	I/Op	Bidir	Hi-Z,Input	SCU414[19]=0 & SCU4B4[19]=0 & SCU694[19]=0
A21	GPIOG4	I/Op	Bidir	Hi-Z,Input	SCU414[20]=0 & SCU4B4[20]=0 & SCU694[20]=0
E20	GPIOG5	I/Op	Bidir	Hi-Z,Input	SCU414[21]=0 & SCU4B4[21]=0 & SCU694[21]=0
D21	GPIOG6	I/Op	Bidir	Hi-Z,Input	SCU414[22]=0 & SCU4B4[22]=0 & SCU694[22]=0
B21	GPIOG7	I/Op	Bidir	Hi-Z,Input	SCU414[23]=0 & SCU4B4[23]=0 & SCU694[23]=0
A18	GPIOH0	ID/O8	Bidir	Hi-Z,Input	SCU414[24]=0
B18	GPIOH1	ID/O8	Bidir	Hi-Z,Input	SCU414[25]=0
C18	GPIOH2	ID/O8	Bidir	Hi-Z,Input	SCU414[26]=0
A17	GPIOH3	ID/O8	Bidir	Hi-Z,Input	SCU414[27]=0
D18	GPIOH4	ID/O8	Bidir	Hi-Z,Input	SCU414[28]=0 & SCU4B4[28]=0
B17	GPIOH5	ID/O8	Bidir	Hi-Z,Input	SCU414[29]=0 & SCU4B4[29]=0
C17	GPIOH6	ID/O8	Bidir	Hi-Z,Input	SCU414[30]=0 & SCU4B4[30]=0
E18	GPIOH7	ID/O8	Bidir	Hi-Z,Input	SCU414[31]=0 & SCU4B4[31]=0
D17	GPIOI0	ID/O8	Bidir	Hi-Z,Input	SCU418[0]=0 & SCU4B8[0]=0
A16	GPIOI1	ID/O8	Bidir	Hi-Z,Input	SCU418[1]=0 & SCU4B8[1]=0
E17	GPIOI2	ID/O8	Bidir	Hi-Z,Input	SCU418[2]=0 & SCU4B8[2]=0
D16	GPIOI3	ID/O8	Bidir	Hi-Z,Input	SCU418[3]=0 & SCU4B8[3]=0
C16	GPIOI4	ID/O8	Bidir	Hi-Z,Input	SCU418[4]=0
E16	GPIOI5	ID/O8	Bidir	Hi-Z,Input	SCU418[5]=0
B16	GPIOI6	ID/O8	Bidir	Hi-Z,Input	SCU418[6]=0
A15	GPIOI7	ID/O8	Bidir	Hi-Z,Input	SCU418[7]=0 & SCU4B8[7]=0
B20	GPIOJ0	IU/O12	Bidir	Hi-Z,Input	SCU418[8]=0 & SCU4B8[8]=0
A20	GPIOJ1	IU/O12	Bidir	Hi-Z,Input	SCU418[9]=0 & SCU4B8[9]=0
E19	GPIOJ2	IU/O12	Bidir	Hi-Z,Input	SCU418[10]=0 & SCU4B8[10]=0
D20	GPIOJ3	IU/O12	Bidir	Hi-Z,Input	SCU418[11]=0 & SCU4B8[11]=0
C19	GPIOJ4	IU/O12	Bidir	Hi-Z	SCU418[12]=0 & SCU4B8[12]=0
A19	GPIOJ5	IU/O12	Bidir	Hi-Z	SCU418[13]=0 & SCU4B8[13]=0
C20	GPIOJ6	IU/O12	Bidir	Hi-Z	SCU418[14]=0 & SCU4B8[14]=0
D19	GPIOJ7	IU/O12	Bidir	Hi-Z	SCU418[15]=0 & SCU4B8[15]=0
A11	GPIOK0	IS/O8	Bidir	Hi-Z,Input	SCU418[16]=0
C11	GPIOK1	IS/O8	Bidir	Hi-Z,Input	SCU418[17]=0

Ball	Name	I/O	Type	Power-Up	Multi-function Control
D12	GPIOK2	IS/O8	Bidir	Hi-Z,Input	SCU418[18]=0
E13	GPIOK3	IS/O8	Bidir	Hi-Z,Input	SCU418[19]=0
D11	GPIOK4	IS/O8	Bidir	Hi-Z,Input	SCU418[20]=0
E11	GPIOK5	IS/O8	Bidir	Hi-Z,Input	SCU418[21]=0
F13	GPIOK6	IS/O8	Bidir	Hi-Z,Input	SCU418[22]=0
E12	GPIOK7	IS/O8	Bidir	Hi-Z,Input	SCU418[23]=0
D15	GPIOL0	IS/O8	Bidir	Hi-Z,Input	SCU418[24]=0
A14	GPIOL1	IS/O8	Bidir	Hi-Z,Input	SCU418[25]=0
E15	GPIOL2	IS/O8	Bidir	Hi-Z,Input	SCU418[26]=0
A13	GPIOL3	IS/O8	Bidir	Hi-Z,Input	SCU418[27]=0
C15	GPIOL4	ID/O8	Bidir	Hi-Z,Input	SCU418[28]=0
F15	GPIOL5	ID/O8	Bidir	Hi-Z,Input	SCU418[29]=0
B14	GPIOL6	ID/O8	Bidir	Hi-Z,Input	SCU418[30]=0
C14	GPIOL7	ID/O8	Bidir	Hi-Z,Input	SCU418[31]=0
D14	GPIOM0	ID/O8	Bidir	Hi-Z,Input	SCU41C[0]=0
B13	GPIOM1	ID/O8	Bidir	Hi-Z,Input	SCU41C[1]=0
A12	GPIOM2	ID/O8	Bidir	Hi-Z,Input	SCU41C[2]=0
E14	GPIOM3	ID/O8	Bidir	Hi-Z,Input	SCU41C[3]=0
B12	GPIOM4	ID/O8	Bidir	Hi-Z,Input	SCU41C[4]=0
C12	GPIOM5	ID/O8	Bidir	Hi-Z,Input	SCU41C[5]=0
C13	GPIOM6	ID/O8	Bidir	Hi-Z,Input	SCU41C[6]=0
D13	GPIOM7	ID/O8	Bidir	Hi-Z,Input	SCU41C[7]=0
P25	GPION0	ID/O8	Bidir	Hi-Z,Input	SCU41C[8]=0
N23	GPION1	ID/O8	Bidir	Hi-Z,Input	SCU41C[9]=0
N25	GPION2	ID/O8	Bidir	Hi-Z,Input	SCU41C[10]=0
N24	GPION3	ID/O8	Bidir	Hi-Z,Input	SCU41C[11]=0
P26	GPION4	ID/O8	Bidir	Hi-Z,Input	SCU41C[12]=0
M23	GPION5	ID/O8	Bidir	Hi-Z,Input	SCU41C[13]=0
N26	GPION6	ID/O8	Bidir	Hi-Z,Input	SCU41C[14]=0
M26	GPION7	ID/O8	Bidir	Hi-Z,Input	SCU41C[15]=0
AD26	GPIOO0	ID/O8	Bidir	Hi-Z,Input	SCU41C[16]=0
AD22	GPIOO1	ID/O8	Bidir	Hi-Z,Input	SCU41C[17]=0
AD23	GPIOO2	ID/O8	Bidir	Hi-Z,Input	SCU41C[18]=0
AD24	GPIOO3	ID/O8	Bidir	Hi-Z,Input	SCU41C[19]=0
AD25	GPIOO4	ID/O8	Bidir	Hi-Z,Input	SCU41C[20]=0
AC22	GPIOO5	ID/O8	Bidir	Hi-Z,Input	SCU41C[21]=0
AC24	GPIOO6	ID/O8	Bidir	Hi-Z,Input	SCU41C[22]=0
AC23	GPIOO7	ID/O8	Bidir	Hi-Z,Input	SCU41C[23]=0
AB22	GPIOP0	ID/O8	Bidir	Hi-Z,Input	SCU41C[24]=0 & SCU4BC[24]=0
W24	GPIOP1	ID/O8	Bidir	Hi-Z,Input	SCU41C[25]=0 & SCU4BC[25]=0
AA23	GPIOP2	ID/O8	Bidir	Hi-Z,Input	SCU41C[26]=0 & SCU4BC[26]=0
AA24	GPIOP3	ID/O8	Bidir	Hi-Z,Input	SCU41C[27]=0 & SCU4BC[27]=0
W23	GPIOP4	ID/O8	Bidir	Hi-Z,Input	SCU41C[28]=0 & SCU4BC[28]=0

Ball	Name	I/O	Type	Power-Up	Multi-function Control
AB23	GPIOP5	ID/O8	Bidir	Hi-Z,Input	SCU41C[29]=0 & SCU4BC[29]=0
AB24	GPIOP6	ID/O8	Bidir	Hi-Z,Input	SCU41C[30]=0 & SCU4BC[30]=0
Y23	GPIOP7	ID/O8	Bidir	Hi-Z,Input	SCU41C[31]=0 & SCU4BC[31]=0
AA25	GPIOQ0	IS/O8	Bidir	Hi-Z,Input	SCU430[0]=0
AB25	GPIOQ1	IS/O8	Bidir	Hi-Z,Input	SCU430[1]=0
Y24	GPIOQ2	IS/O8	Bidir	Hi-Z,Input	SCU430[2]=0
AB26	GPIOQ3	IS/O8	Bidir	Hi-Z,Input	SCU430[3]=0
Y26	GPIOQ4	IS/O8	Bidir	Hi-Z,Input	SCU430[4]=0
AC26	GPIOQ5	IS/O8	Bidir	Hi-Z,Input	SCU430[5]=0
Y25	GPIOQ6	IS/O8	Bidir	Hi-Z,Input	SCU430[6]=0
AA26	GPIOQ7	IS/O8	Bidir	Hi-Z,Input	SCU430[7]=0
V25	GPIOR0	ID/O8	Bidir	Hi-Z	SCU430[8]=0
U24	GPIOR1	ID/O8	Bidir	Hi-Z	SCU430[9]=0
V24	GPIOR2	ID/O8	Bidir	Hi-Z,Input	SCU430[10]=0
V26	GPIOR3	ID/O8	Bidir	Hi-Z,Input	SCU430[11]=0
U25	GPIOR4	ID/O8	Bidir	Hi-Z,Input	SCU430[12]=0
T23	GPIOR5	ID/O8	Bidir	Hi-Z,Input	SCU430[13]=0
W26	GPIOR6	ID/O8	Bidir	Hi-Z,Input	SCU430[14]=0
U26	GPIOR7	ID/O8	Bidir	Hi-Z,Input	SCU430[15]=0
R23	GPIOS0	ID/O8	Bidir	Hi-Z,Input	SCU430[16]=0
T25	GPIOS1	ID/O8	Bidir	Hi-Z,Input	SCU430[17]=0
T26	GPIOS2	ID/O8	Bidir	Hi-Z,Input	SCU430[18]=0
R24	GPIOS3	ID/O8	Bidir	Hi-Z,Input	SCU430[19]=0
R26	GPIOS4	ID/O8	Bidir	Hi-Z,Input	SCU430[20]=0
P24	GPIOS5	ID/O8	Bidir	Hi-Z,Input	SCU430[21]=0
P23	GPIOS6	ID/O8	Bidir	Hi-Z,Input	SCU430[22]=0
T24	GPIOS7	ID/O8	Bidir	Hi-Z,Input	SCU430[23]=0
AD20	GPIT0	I	Input	Hi-Z	SCU430[24]=1
AC18	GPIT1	I	Input	Hi-Z	SCU430[25]=1
AE19	GPIT2	I	Input	Hi-Z	SCU430[26]=1
AD19	GPIT3	I	Input	Hi-Z	SCU430[27]=1
AC19	GPIT4	I	Input	Hi-Z	SCU430[28]=1
AB19	GPIT5	I	Input	Hi-Z	SCU430[29]=1
AB18	GPIT6	I	Input	Hi-Z	SCU430[30]=1
AE18	GPIT7	I	Input	Hi-Z	SCU430[31]=1
AB16	GPIU0	I	Input	Hi-Z	SCU434[0]=1 & SCU4D4[0]=0
AA17	GPIU1	I	Input	Hi-Z	SCU434[1]=1 & SCU4D4[1]=0
AB17	GPIU2	I	Input	Hi-Z	SCU434[2]=1 & SCU4D4[2]=0
AE16	GPIU3	I	Input	Hi-Z	SCU434[3]=1 & SCU4D4[3]=0
AC16	GPIU4	I	Input	Hi-Z	SCU434[4]=1 & SCU4D4[4]=0
AA16	GPIU5	I	Input	Hi-Z	SCU434[5]=1 & SCU4D4[5]=0
AD16	GPIU6	I	Input	Hi-Z	SCU434[6]=1 & SCU4D4[6]=0
AC17	GPIU7	I	Input	Hi-Z	SCU434[7]=1 & SCU4D4[7]=0

Ball	Name	I/O	Type	Power-Up	Multi-function Control
AB15	GPIOV0	ID/O8	Bidir	Hi-Z	SCU434[8]=0
AF14	GPIOV1	ID/O8	Bidir	Hi-Z	SCU434[9]=0
AD14	GPIOV2	ID/O8	Bidir	Hi-Z	SCU434[10]=0
AC15	GPIOV3	ID/O8	Bidir	Hi-Z	SCU434[11]=0
AE15	GPIOV4	ID/O8	Bidir	Hi-Z	SCU434[12]=0
AE14	GPIOV5	ID/O8	Bidir	Hi-Z	SCU434[13]=0 & SCU4D4[13]=0
AD15	GPIOV6	ID/O8	Bidir	Hi-Z	SCU434[14]=0
AF15	GPIOV7	ID/O8	Bidir	Hi-Z	SCU434[15]=0
AB7	GPIOW0	I/Op	Bidir	Hi-Z	SCU434[16]=0
AB8	GPIOW1	I/Op	Bidir	Hi-Z	SCU434[17]=0
AC8	GPIOW2	I/Op	Bidir	Hi-Z	SCU434[18]=0
AC7	GPIOW3	I/Op	Bidir	Hi-Z	SCU434[19]=0
AE7	GPIOW4	I/O8	Bidir	Hi-Z	SCU434[20]=0
AF7	GPIOW5	I/O8	Bidir	Hi-Z	SCU434[21]=0
AD7	GPIOW6	I/O8	Bidir	Hi-Z	SCU434[22]=0
AD8	GPIOW7	I/O8	Bidir	Hi-Z	SCU434[23]=0
AE8	GPIOX0	ID/O8	Bidir	Hi-Z	SCU434[24]=0
AA9	GPIOX1	ID/O8	Bidir	Hi-Z	SCU434[25]=0
AC9	GPIOX2	ID/O8	Bidir	Hi-Z	SCU434[26]=0
AF8	GPIOX3	ID/O8	Bidir	Hi-Z	SCU434[27]=0
AB9	GPIOX4	ID/O8	Bidir	Hi-Z	SCU434[28]=0
AD9	GPIOX5	ID/O8	Bidir	Hi-Z	SCU434[29]=0
AF9	GPIOX6	ID/O8	Bidir	Hi-Z	SCU434[30]=0 & SCU4D4[30]=0
AB10	GPIOX7	ID/O8	Bidir	Hi-Z	SCU434[31]=0 & SCU4D4[31]=0
AF11	GPIOY0	ID/O8	Bidir	Hi-Z,Input	SCU438[0]=0 & SCU4D8[0]=0
AD12	GPIOY1	ID/O8	Bidir	Hi-Z,Input	SCU438[1]=0 & SCU4D8[1]=0
AE11	GPIOY2	ID/O8	Bidir	Hi-Z,Input	SCU438[2]=0 & SCU4D8[2]=0
AA12	GPIOY3	ID/O8	Bidir	Hi-Z,Input	SCU438[3]=0 & SCU4D8[3]=0
AE12	GPIOY4	ID/O8	Bidir	Hi-Z	SCU438[4]=0
AF12	GPIOY5	ID/O8	Bidir	Hi-Z	SCU438[5]=0
AC12	GPIOY6	ID/O8	Bidir	Hi-Z	SCU438[6]=0
AB12	GPIOY7	IS/O8	Bidir	Hi-Z	SCU438[7]=0
AC10	GPIOZ0	ID/O8	Bidir	Hi-Z,Input	SCU438[8]=0
AD10	GPIOZ1	ID/O8	Bidir	Hi-Z,Input	SCU438[9]=0
AE10	GPIOZ2	IS/O8	Bidir	Hi-Z,Input	SCU438[10]=0
AB11	GPIOZ3	ID/O8	Bidir	Hi-Z,Input	SCU438[11]=0
AC11	GPIOZ4	ID/O8	Bidir	Hi-Z,Input	SCU438[12]=0
AA11	GPIOZ5	ID/O8	Bidir	Hi-Z,Input	SCU438[13]=0
AD11	GPIOZ6	ID/O8	Bidir	Hi-Z,Input	SCU438[14]=0
AF10	GPIOZ7	ID/O8	Bidir	Hi-Z,Input	SCU438[15]=0
1.8V GPIO Interface					
C6	GPIO18A0	ID/O8	Bidir	Hi-Z,Input	SCU400[0]=0
D6	GPIO18A1	ID/O8	Bidir	Hi-Z,Input	SCU400[1]=0

Ball	Name	I/O	Type	Power-Up	Multi-function Control
D5	GPIO18A2	ID/O8	Bidir	Hi-Z,Input	SCU400[2]=0
A3	GPIO18A3	ID/O8	Bidir	Hi-Z,Input	SCU400[3]=0
C5	GPIO18A4	ID/O8	Bidir	Hi-Z,Input	SCU400[4]=0
E6	GPIO18A5	ID/O8	Bidir	Hi-Z,Input	SCU400[5]=0
B3	GPIO18A6	ID/O8	Bidir	Hi-Z,Input	SCU400[6]=0
A2	GPIO18A7	ID/O8	Bidir	Hi-Z,Input	SCU400[7]=0
B2	GPIO18B0	ID/O8	Bidir	Hi-Z,Input	SCU400[8]=0
B1	GPIO18B1	ID/O8	Bidir	Hi-Z,Input	SCU400[9]=0
C4	GPIO18B2	ID/O8	Bidir	Hi-Z,Input	SCU400[10]=0
E5	GPIO18B3	ID/O8	Bidir	Hi-Z,Input	SCU400[11]=0
D4	GPIO18B4	ID/O8	Bidir	Hi-Z,Input	SCU400[12]=0
C2	GPIO18B5	ID/O8	Bidir	Hi-Z,Input	SCU400[13]=0
C1	GPIO18B6	ID/O8	Bidir	Hi-Z,Input	SCU400[14]=0
D3	GPIO18B7	ID/O8	Bidir	Hi-Z,Input	SCU400[15]=0
E4	GPIO18C0	ID/O8	Bidir	Hi-Z,Input	SCU400[16]=0
F5	GPIO18C1	ID/O8	Bidir	Hi-Z,Input	SCU400[17]=0
D2	GPIO18C2	ID/O8	Bidir	Hi-Z,Input	SCU400[18]=0
E3	GPIO18C3	ID/O8	Bidir	Hi-Z,Input	SCU400[19]=0
D1	GPIO18C4	ID/O8	Bidir	Hi-Z,Input	SCU400[20]=0
F4	GPIO18C5	ID/O8	Bidir	Hi-Z,Input	SCU400[21]=0
E2	GPIO18C6	ID/O8	Bidir	Hi-Z,Input	SCU400[22]=0
E1	GPIO18C7	ID/O8	Bidir	Hi-Z,Input	SCU400[23]=0
AB4	GPIO18D0	ID/O8	Bidir	Hi-Z,Input	SCU400[24]=0
AA4	GPIO18D1	ID/O8	Bidir	Hi-Z,Input	SCU400[25]=0
AC4	GPIO18D2	ID/O8	Bidir	Hi-Z,Input	SCU400[26]=0
AA5	GPIO18D3	ID/O8	Bidir	Hi-Z,Input	SCU400[27]=0
Y5	GPIO18D4	ID/O8	Bidir	Hi-Z,Input	SCU400[28]=0
AB5	GPIO18D5	ID/O8	Bidir	Hi-Z,Input	SCU400[29]=0
AB6	GPIO18D6	ID/O8	Bidir	Hi-Z,Input	SCU400[30]=0
AC5	GPIO18D7	ID/O8	Bidir	Hi-Z,Input	SCU400[31]=0
Y1	GPIO18E0	ID/O8	Bidir	Hi-Z,Input	SCU404[0]=0
Y2	GPIO18E1	ID/O8	Bidir	Hi-Z,Input	SCU404[1]=0
Y3	GPIO18E2	ID/O8	Bidir	Hi-Z,Input	SCU404[2]=0
Y4	GPIO18E3	ID/O8	Bidir	Hi-Z,Input	SCU404[3]=0
Misc. Interface					
AF11	WDTRST1#	O8	Out	Hi-Z	SCU438[0]=0 & SCU4D8[0]=1
AD12	WDTRST2#	O8	Out	Hi-Z	SCU438[1]=0 & SCU4D8[1]=1
AE11	WDTRST3#	O8	Out	Hi-Z	SCU438[2]=0 & SCU4D8[2]=1
AA12	WDTRST4#	O8	Out	Hi-Z	SCU438[3]=0 & SCU4D8[3]=1
K26	MACLINK1	IS	In	Hi-Z	SCU410[4]=1
L24	MACLINK2	IS	In	Hi-Z	SCU410[5]=1
L23	MACLINK3	IS	In	Hi-Z	SCU410[6]=1
K25	MACLINK4	IS	In	Hi-Z	SCU410[7]=1

Ball	Name	I/O	Type	Power-Up	Multi-function Control
R24	OSCCLK	O8	Out	Hi-Z	SCU430[19]=1
Y23	HEARTBEAT	O8	Out	Output	SCU69C[31]=1
T26	PEWAKE#	ID/O8	Out	Hi-Z	SCU430[18]=1

//local reset signals

6 Reset Source Table

Syntax:

Symbol	:	Description
#	:	Denotes active low signal
!	:	Denotes invert value, that is register value equals to 0
R[n]	:	Denotes bit index **n** of register **R**
||	:	Logical OR
&	:	Logical AND
S ? A : B	:	Logical function: if(S = 1) then A else B
SRST#	:	Reset input pin
EXTRST#	:	Reset input pin
PERST#	:	Reset input pin
SSPRST#	:	SSP Reset input pin
LPCRST#	:	Reset input pin
LHRST#	:	Reset input pin
ESPIRST#	:	Reset input pin
ESPICS#	:	eSPI chip select input pin
PLTRST#	:	eSPI virtual wire of platform reset command input (ESPI098[5])
WDT_Full	:	Watchdog full reset event
WDT_SOC1	:	Watchdog SOC reset event 1
WDT_SOC2	:	Watchdog SOC reset event 2
WDT_ARM	:	Watchdog ARM reset event

Logical operation in following tagle is Active-based, not Level-based.

Table 6: Reset Functions

Reset Function List		
Name	**Reset Source Function**	**Event Log**
ResetSOC1[n]	(SCU060[n] & EXTRST#) || ([n] & WDT_SOC1)	
ResetSOC2[n]	(SCU070[n] & EXTRST#) || ([n] & WDT_SOC2)	
RstPwr	SRST#	SCU064[0], SCU074[0]
RstFull	SRST# || ((SCU0C8[9] || SCU500[19] || SCU500[25]) & WDT_Full)	SCU064[11], SCU074[11]
RstCM3	!SCU500[24] & SSPRST#	SCU064[6]
RstPE	PERST#	SCU064[4]
RstLPC	LPCRST#	SCU074[4]
RstRC	SCU500[24] ? SSPRST# : PERST#	SCU064[5]

RstARM	RstFull || (ResetSOC1[0] & ResetSOC2[0]) || WDT_ARM	SCU064[12], SCU074[12]
Rst00	RstFull || ResetSOC1[1] || (SCU040[0] & RstFull)	SCU068[2]
Rst01	RstFull || (ResetSOC1[2] & ResetSOC2[2]) || SCU040[1] || SCU050[1]	SCU064[14], SCU074[14]
Rst02	RstFull || (ResetSOC1[3] & ResetSOC2[3]) || Rst01	SCU064[15], SCU074[15]
Rst03	RstFull || (ResetSOC1[4] & ResetSOC2[4])	SCU064[13], SCU074[13]
Rst04	RstPwr || RstCM3 || ResetSOC1[5] || !SCUA00[0] || SCUA00[1] || WDT_ARM	SCU068[3]
Rst05	RstFull || ResetSOC1[6] || SCU040[14]	SCU068[18]
Rst06	RstFull || ResetSOC1[6] || SCU040[14] || SCU440[25]	
Rst07	RstFull || ResetSOC1[6] || SCU040[14] || (SCU440[25:24]!=2)	
Rst08	RstFull || RstPE || (SCU440[25:24]=1) || (SCU440[25:24]=2)	SCU068[19]
Rst09	RstFull || ResetSOC1[7] || SCU040[3] || (SCU440[29:28]!=0)	SCU068[17]
Rst10	RstFull || ResetSOC1[7] || SCU040[3] || (SCU440[29:28]=0)	SCU068[20]
Rst11	RstFull || ResetSOC1[7] || SCU040[3] || (SCU440[29:28]!=1)	
Rst12	RstFull || ResetSOC1[7] || SCU040[3] || !SCU440[29]	
Rst13	RstFull || ResetSOC1[8] || SCU040[15]	SCU068[16]
Rst14	RstFull || ResetSOC1[9] || SCU040[13]	SCU068[5]
Rst15	RstFull || ResetSOC1[9] || SCU040[13] || SCU040[26]	
Rst16	SCU0C0[7] ? (RstFull || ResetSOC1[10] || SCU040[7]) : RstPE	SCU068[6]
Rst17	RstFull || ResetSOC1[11] || SCU040[6]	SCU068[21]
Rst18	RstFull || ResetSOC1[12] || SCU040[4]	SCU068[4]
Rst19	RstFull || ResetSOC1[13] || SCU040[28] || (SCU040[5] & RstPE)	SCU068[25]
Rst20	RstFull || ResetSOC1[14] || SCU040[29]	SCU068[26]
Rst21	RstFull || ResetSOC1[15] || SCU040[30]	SCU068[27]
Rst22	RstFull || ResetSOC1[16] || SCU040[11]	SCU068[9]
Rst23	RstFull || ResetSOC1[17] || SCU040[12]	SCU068[10]
Rst24	Rst22 || Rst23	
Rst25	RstFull || ResetSOC1[18] || SCU040[16]	SCU068[15]
Rst26	RstFull || ResetSOC1[19] || SCU040[22]	SCU068[1]
Rst27	RstFull || ResetSOC1[20] || SCU040[24] || RstPE	SCU068[11]
Rst28	RstFull || ResetSOC1[21] || SCU040[23] || RstRC	SCU068[12]
Rst29	SCU0C0[8] ? (RstFull || ResetSOC1[22] || SCU040[25]) : RstPE	SCU068[23]
Rst30	RstFull || ResetSOC1[23] || SCU040[27] || RstRC	SCU068[24]
Rst31	RstFull || ResetSOC1[24]	SCU068[0]
Rst32	RstFull || ResetSOC1[25] || SCU040[9]	SCU068[22]
Rst33	RstPE || SCUC20[7]	SCU068[14]
Rst34	RstFull || ResetSOC1[26] || SCU040[31] || Rst01	SCU068[28]
Rst35	RstFull || Rst01	SCU068[13]
Rst36	RstFull	SCU068[7]
Rst37	RstFull	SCU068[8]

Rst38	RstFull || ResetSOC2[1]	SCU078[26]
Rst39	RstFull || ResetSOC2[5] || SCU050[20]	SCU078[12]
Rst40	RstFull || ResetSOC2[6] || SCU050[21]	SCU078[13]
Rst41	Rst39 || Rst40	
Rst42	RstFull || ResetSOC2[7] || SCU050[24]	SCU078[25]
Rst43	RstFull || ResetSOC2[8] || SCU050[22]	SCU078[1]
Rst44	RstFull || ResetSOC2[9]	SCU078[0]
Rst45	RstFull || ResetSOC2[10] || SCU050[3]	SCU078[6]
Rst47	RstFull || ResetSOC2[11] || SCU050[0]	SCU078[5]
Rst48	!LHCR0[0] || LHRST#	SCU074[5]
Rst49	RstFull || ResetSOC2[12] || SCU050[4]	SCU078[14]
Rst50	RstFull || ResetSOC2[13] || SCU050[5]	SCU078[24]
Rst51	RstFull || ResetSOC2[14] || SCU050[23]	SCU078[2]
Rst52	RstFull || ResetSOC2[15] || SCU050[27]	SCU078[3]
Rst53	RstFull || ResetSOC2[16] || SCU050[2]	SCU078[4]
Rst54	RstFull || ResetSOC2[17] || SCU050[7]	SCU078[15]
Rst55	RstFull || ResetSOC2[18] || SCU050[8]	SCU078[16]
Rst56	RstFull || ResetSOC2[19] || SCU050[9]	SCU078[17]
Rst57	RstFull || ResetSOC2[20] || SCU050[10]	SCU078[18]
Rst58	RstFull || ResetSOC2[21] || SCU050[11]	SCU078[19]
Rst59	RstFull || ResetSOC2[22] || SCU050[12]	SCU078[20]
Rst60	RstFull || ResetSOC2[23] || SCU050[13]	SCU078[21]
Rst61	RstFull || ResetSOC2[26] || SCU050[25]	SCU078[27]
Rst46	RstFull || LPCRST# || (SCU510[6] & PLTRST#)	SCU078[7]
Rst62	ESPICS#	
Rst63	HICR9[4] ? LPCRST# : RstFull	SCU078[8]
Rst64	HICR9[5] ? LPCRST# : RstFull	SCU078[9]
Rst65	HICR9[6] ? LPCRST# : RstFull	SCU078[10]
Rst66	HICR9[7] ? LPCRST# : RstFull	SCU078[11]
Rst67	RstFull || (SCUF80[0] & SCUF90[0] & RstARM)	
Rst68	RstFull || (SCUF80[8] & SCUF90[8] & RstARM)	
Rst69	RstFull || (SCUF80[12] & RstARM)	
Rst70	RstFull || (SCUF90[12] & RstARM)	
Rst71	RstFull || (SCUF80[16] & RstARM)	
Rst72	RstFull || (SCUF90[16] & RstARM)	
Rst73	RstFull || (SCUF80[24] & RstARM)	
Rst74	RstFull || (SCUF80[25] & RstARM)	
Rst75	RstPwr || (SCUF80[25] & RstARM)	
Rst73	RstFull || (SCUF80[24] & RstARM)	
Rst74	RstFull || (SCUF80[25] & RstARM)	
Rst75	RstPwr || (SCUF80[25] & RstARM)	
Rst76	RstFull || (SCUF90[31] & RstARM)	
Rst76	RstFull || (SCUF84[0] & RstARM)	

Rst77	RstFull || (SCUF84[1] & RstARM)	
Rst78	RstFull || (SCUF84[1] & RstARM)	
Rst79	RstFull || (SCUF84[1] & RstARM)	
Rst80	RstFull || (SCUF84[16] & RstARM)	
Rst81	RstFull || (SCUF84[17] & RstARM)	
Rst82	RstFull || (SCUF84[18] & RstARM)	
Rst83	RstFull || (SCUF84[19] & RstARM)	
Rst84	RstFull || (SCUF84[20] & RstARM)	
Rst85	RstFull || (SCUF84[21] & RstARM)	
Rst86	RstFull || (SCUF84[22] & RstARM)	
Rst87	RstFull || (SCUF84[23] & RstARM)	
Rst88	RstFull || (SCUF88[0] & RstARM)	
Rst89	RstFull || (SCUF88[1] & RstARM)	
Rst90	RstFull || (SCUF98[0] & RstARM)	
Rst91	RstFull || (SCUF98[1] & RstARM)	
Rst92	RstFull || (SCUF88[4] & RstARM)	
Rst93	RstFull || (SCUF88[5] & RstARM)	
Rst94	RstFull || (SCUF88[8] & RstARM)	
Rst95	RstFull || (SCUF88[9] & RstARM)	
Rst96	RstFull || (SCUF88[12] & RstARM)	
Rst97	RstFull || (SCUF88[13] & RstARM)	
Rst98	RstFull || (SCUF8C[0] & RstARM)	
Rst99	RstFull || (SCUF8C[1] & RstARM)	
Rst100	RstFull || (SCUF8C[2] & RstARM)	
Rst101	RstFull || (SCUF9C[0] & RstARM)	
Rst102	RstFull || (SCUF9C[1] & RstARM)	
Rst103	RstFull || (SCUF8C[4] & RstARM)	
Rst104	RstFull || (SCUF8C[5] & RstARM)	
Rst105	RstFull || (SCUF8C[7] & RstARM)	
Rst106	RstFull || (SCUF9C[4] & RstARM)	
Rst107	RstFull || (SCUF9C[5] & RstARM)	
Rst108	RstFull || (SCUF9C[6] & RstARM)	
Rst109	RstFull || (SCUF9C[7] & RstARM)	
Rst110	RstFull || (SCUF8C[8] & RstARM)	
Rst111	RstFull || (SCUF8C[10] & RstARM)	
Rst112	RstFull || (SCUF8C[11] & RstARM)	
Rst113	RstFull || (SCUF9C[8] & RstARM)	
Rst114	RstFull || (SCUF9C[10] & RstARM)	
Rst115	RstFull || (SCUF9C[11] & RstARM)	
Rst113	RstFull || (SCUF8C[12] & RstARM)	
Rst114	RstFull || (SCUF8C[13] & RstARM)	
Rst115	RstFull || (SCUF8C[14] & RstARM)	
Rst116	RstFull || (SCUF9C[12] & RstARM)	

Rst117	RstFull || (SCUF9C[13] & RstARM)	
Rst124	RstFull || (SCUFA0[0] & RstARM)	
Rst125	RstFull || (SCUFA0[1] & RstARM)	
Rst126	RstFull || (SCUFA0[3] & RstARM)	
Rst127	RstFull || (SCUFB0[0] & RstARM)	
Rst128	RstFull || (SCUFB0[1] & RstARM)	
Rst129	RstFull || (SCUFB0[2] & RstARM)	
Rst130	RstFull || (SCUFB0[3] & RstARM)	
Rst131	RstFull || (SCUFB0[4] & RstARM)	
Rst132	RstFull || (SCUFB0[5] & RstARM)	
Rst133	RstFull || (SCUFB0[6] & RstARM)	
Rst134	RstFull || (SCUFB0[7] & RstARM)	
Rst135	RstFull || (SCUFA0[8] & RstARM)	
Rst136	RstFull || (SCUFB0[8] & RstARM)	
Rst137	RstFull || (SCUFB0[9] & RstARM)	
Rst138	RstFull || (SCUFB0[10] & RstARM)	
Rst139	RstFull || (SCUFB0[12] & RstARM)	
Rst140	RstFull || (SCUFB0[20] & RstARM)	
Rst141	RstFull || (SCUFB0[21] & RstARM)	
Rst142	RstFull || (SCUFB0[22] & RstARM)	
Rst143	RstFull || (SCUFB0[23] & RstARM)	
Rst144	RstFull || (SCUFB0[25] & RstARM)	
Rst145	RstFull || (SCUFB0[26] & RstARM)	
Rst146	RstFull || (SCUFB0[29] & RstARM)	
Rst147	RstFull || (SCUFB8[16] & RstARM)	
Rst148	RstFull || (SCUFB8[17] & RstARM)	
Rst149	RstFull || (SCUFB8[18] & RstARM)	
Rst150	RstFull || (SCUFB8[19] & RstARM)	
Rst151	RstFull || (SCUFB8[24] & RstARM)	
Rst152	RstFull || (SCUFA4[2] & RstARM)	
Rst153	RstFull || (SCUFB4[2] & RstARM)	
Rst154	RstFull || (SCUFA4[4] & RstARM)	
Rst155	RstFull || (SCUFA4[9] & RstARM)	
Rst156	RstFull || (SCUFB4[8] & RstARM)	
Rst157	RstFull || (SCUFA4[12] & RstARM)	
Rst158	RstFull || (SCUFB4[12] & RstARM)	
Rst159	RstFull || (SCUFA4[17] & RstARM)	
Rst160	RstFull || (SCUFB4[16] & RstARM)	
Rst161	RstFull || (SCUFB8[0] & RstARM)	
Rst162	RstFull || (SCUFB8[1] & RstARM)	
Rst163	RstFull || (SCUFB8[2] & RstARM)	
Rst164	RstFull || (SCUFB8[3] & RstARM)	
Rst165	RstFull || (SCUFB8[4] & RstARM)	

Rst166	RstFull || (SCUFB8[5] & RstARM)	
Rst167	RstFull || (SCUFB8[6] & RstARM)	
Rst168	RstFull || (SCUFB8[8] & RstARM)	
Rst169	RstFull || (SCUFC0[0] & RstARM)	
Rst170	RstFull || (SCUFC0[2] & RstARM)	
Rst171	RstFull || (SCUFC0[3] & RstARM)	
Rst172	RstFull || (SCUFC0[4] & RstARM)	
Rst173	RstFull || (SCUFC0[5] & RstARM)	
Rst174	RstFull || (SCUFC8[0] & RstARM)	
Rst175	RstFull || (SCUFC8[1] & RstARM)	
Rst176	RstFull || (SCUFC8[2] & RstARM)	
Rst177	RstFull || (SCUFC8[3] & RstARM)	
Rst178	RstFull || (SCUFC8[8] & RstARM)	
Rst179	RstFull || (SCUFC8[10] & RstARM)	
Rst180	RstFull || (SCUFE0[0] & RstARM)	
Rst181	RstFull || (SCUFE0[1] & RstARM)	
Rst182	RstFull || (SCUFE0[2] & RstARM)	
Rst183	RstFull || (SCUFE0[4] & RstARM)	
Rst184	RstFull || (SCUFE0[5] & RstARM)	
Rst185	RstFull || (SCUFE0[6] & RstARM)	
Rst186	RstFull || (SCUFE0[7] & RstARM)	
Rst187	RstFull || (SCUFE0[8] & RstARM)	
Rst188	RstFull || (SCUFE0[9] & RstARM)	
Rst189	RstFull || (SCUFE0[10] & RstARM)	
Rst190	RstFull || (SCUFE0[11] & RstARM)	
Rst191	RstFull || (SCUFE0[12] & RstARM)	
Rst192	RstFull || (SCUFE0[14] & RstARM)	
Rst193	RstFull || (SCUFE0[16] & RstARM)	
Rst194	RstFull || (SCUFEC[0] & RstARM)	
Rst195	RstFull || (SCUFEC[1] & RstARM)	
Rst196	RstFull || (SCUFEC[2] & RstARM)	
Rst197	RstFull || (SCUFEC[3] & RstARM)	
Rst198	RstFull || (SCUFFC[0] & RstARM)	
Rst199	RstFull || (SCUFFC[2] & RstARM)	
Rst200	RstFull || (SCUFFC[3] & RstARM)	
Rst201	RstFull || (SCUFEC[4] & RstARM)	
Rst202	RstFull || (SCUFEC[5] & RstARM)	
Rst203	RstFull || (SCUFFC[4] & RstARM)	
Rst204	RstFull || (SCUFFC[5] & RstARM)	
Rst205	RstFull || (SCUFFC[6] & RstARM)	
Rst206	RstFull || (SCUFEC[8] & RstARM)	
Rst207	RstFull || (SCUFEC[10] & RstARM)	
Rst208	RstFull || (SCUFEC[12] & RstARM)	

Rst209	RstFull || (SCUFEC[15] & RstARM)	
Rst210	RstFull || (SCUFFC[15] & RstARM)	
Rst211	Rst01 || (AHBC208[0] & RstARM)	
Rst212	Rst01 || (AHBC208[1] & RstARM)	
Rst213	Rst25 || EMMC12C[24]	
Rst214	Rst25 || EMMC12C[25] || EMMC12C[24]	
Rst215	Rst25 || EMMC12C[26] || EMMC12C[24]	
Rst216	Rst42 || SDIO12C[24]	
Rst217	Rst42 || SDIO12C[25] || SDIO12C[24]	
Rst218	Rst42 || SDIO12C[26] || SDIO12C[24]	
Rst219	Rst42 || SDIO22C[24]	
Rst220	Rst42 || SDIO22C[25] || SDIO22C[24]	
Rst221	Rst42 || SDIO22C[26] || SDIO22C[24]	
Rst222	RstFull || LPCRST#	
Rst223	RstFull || !ESPI000[24]	
Rst224	RstFull || !ESPI000[25]	
Rst225	RstFull || !ESPI000[26]	
Rst226	RstFull || !ESPI000[27]	
Rst227	RstFull || !ESPI000[28]	
Rst228	RstFull || !ESPI000[29]	
Rst229	RstFull || !ESPI000[30]	
Rst230	RstFull || !ESPI000[31]	
Rst231	RstFull || (SCUFB8[22] & RstARM)	
Rst232	RstFull || (SCUFB8[24] & RstARM)	
Rst233	RstFull || (SCUFB8[25] & RstARM)	
Rst234	RstFull || (SCUFD0[4] & RstARM)	
Rst235	RstFull || (SCUFD0[5] & RstARM)	
Rst236	RstFull || (SCUFD0[8] & RstARM)	
Rst237	RstFull || (SCUFD0[9] & RstARM)	
Rst238	RstFull || (SCUFD0[10] & RstARM)	
Rst239	RstFull || (SCUFD0[11] & RstARM)	
Rst240	RstFull || (SCUFD0[12] & RstARM)	
Rst241	RstFull || (SCUFD0[13] & RstARM)	
Rst242	RstFull || (SCUFD0[14] & RstARM)	
Rst243	RstFull || (SCUFD0[15] & RstARM)	

6.1 Function Level Reset Source Table

Table 7: Function Level Reset Source

Function Module	Reset Source
AHB Bus Controller	RstFull, RstARM, Rst01

Dual-core ARM Cortex A7 CPU	RstARM
Embedded ARM Coetex M3 CPU	RstCM3
AHB to SDRAM Bridge	Rst01
AHB to APB Bridge	Rst01
Firmware SPI Memory Controller	RstFull, RstARM
SPI1/SPI2 Flash Controller	RstFull, Rst38
10/100/1G Ethernet MAC Controller (MAC1)	RstFull, Rst24, Rst22
10/100/1G Ethernet MAC Controller (MAC2)	RstFull, Rst24, Rst23
10/100/1G Ethernet MAC Controller (MAC3)	RstFull, Rst41, Rst39
10/100/1G Ethernet MAC Controller (MAC4)	RstFull, Rst41, Rst40
Ethernet MDC/MDIO Bus Controller	Rst45
USB2.0 Virtual Hub Controller	RstFull, Rst05, Rst06
USB2.0 Device Controller	RstFull, Rst10, Rst11
USB1.1 HID Controller	Rst09
USB1.1 UHCI Host Controller	Rst13
USB2.0 EHCI Host Controller (EHCI1)	RstFull, Rst05, Rst07
USB2.0 EHCI Host Controller (EHCI2)	RstFull, Rst10, Rst12
Software Interrupt Controller	RstARM, RstCM3, Rst01
SDRAM Memory Controller (MMC)	Rst00
System Control Unit (SCU)	SRST#, RstFull, RstARM, Rst01
Hash & Crypto Engine (HACE)	Rst18
ECC/RSA Engine	Rst18
JTAG Master Controller (JTAGM1)	Rst26
JTAG Master Controller (JTAGM2)	Rst43
SOC Display Controller	Rst14, Rst15
X-DMA Controller (XDMA)	RstPE, Rst29
PCIe RC X-DMA Controller (XDMA_RC)	RstRC, Rst30
MCTP Controller (MCTP)	RstPE, Rst27
PCIe RC MCTP Controller (MCTP_RC)	RstRC, Rst28
ADC Controller	Rst51
AHB to PCIe Bus Bridge	RstFull, RstPE, RstRC, Rst34
eSPI Controller	RstFull, Rst61, Rst62
MMBI Controller	RstFull, RstARM
Battery Backed SRAM	SRST#, Rst03
Video Engine	Rst17, Rst15
SRAM Memory Buffer	RstARM, Rst03
SD/SDIO Host Controller	Rst42
eMMC Controller	Rst25
eMMC Boot Controller	RstARM
GPIO Controller (Parallel GPIO)	RstFull, Rst44
GPIO Controller (1.8V Parallel GPIO)	RstFull, Rst31
Real Time Clock	RstFull, Rst03
Timer Controller	RstFull, Rst03

UART Controller (UART1)	Rst63
UART Controller (UART2)	Rst64
UART Controller (UART3)	Rst65
UART Controller (UART4)	Rst66
UART Controller (UART5)	Rst36
UART Controller (UART6 - UART13)	Rst03
UART DMA	Rst03, RstARM
Watchdog Timer	SRST#, Rst03
PWM & Fan Tacho Controller	RstFull, Rst50
Virtual UART (VUART)	RstFull, Rst47, Rst46
LPC Device Controller	RstFull, Rst47, Rst46
SuperIO Controller	RstFull, Rst47, Rst46
System Wake-Up Control	RstFull
MailBox Controller	RstFull
I2C/SMBus Controller	Rst53
I3C Controller (I3C1)	Rst54, Rst55
I3C Controller (I3C2)	Rst54, Rst56
I3C Controller (I3C3)	Rst54, Rst57
I3C Controller (I3C4)	Rst54, Rst58
I3C Controller (I3C5)	Rst54, Rst59
I3C Controller (I3C6)	Rst54, Rst60
I3C HDMA Controller	Rst54
PECI Controller	Rst49
PCIe Host Controller (RC/Bridge)	SRST#, RstPE
PCIe Host Controller (RC)	SRST#, RstRC
Internal Bridge Controller	Rst02
DisplayPort Interface	Rst19
DisplayPort Micro Controller Unit	Rst20
Secure Boot Controller	SRST#, RstARM
FSI Controller	Rst52
PCI Express Controller	SRST#, RstPE
PCI Express Authentication	SRST#, RstPE, RstARM
PCI Express Authentication Micro Controller Unit	Rst21
PCI Bus Controller (P-Bus)	RstFull, RstPE, Rst01
VGA Display Controller	RstFull, RstPE
2D Graphics Engine	Rst16
Message Signaled Interrupts	Rst33, Rst27
PCIe BMC Device	RstFull, RstPE
PCIe EHCI Controller	RstFull, RstPE, Rst08
PCIe Host to BMC Controller	Rst35 || RstPE

7 Memory Space Allocation Table

7.1 ARM Address Space Mapping

Table 8: ARM Address Mapping Table

Address Range	Size (Byte)	Write Mode (Byte)	Read Mode (Byte)	IP Module
0000:0000-0FFF:FFFF	256M	1/2/4	1/2/4	Firmware SPI Memory (boot-up default)
1000:0000-1001:63FF	89K	1/2/4	1/2/4	SRAM Memory Buffer
1400:0000-14FF:FFFF	16M	4	1/2/4	ARM Coresight Interface
1600:0000-17FF:FFFF	32M	1/2/4	1/2/4	AHB Bus to LPC Bus Bridge
1800:0000-1803:FFFF	256K	4	1/2/4	DisplayPort Micro Controller Unit Memory
1880:0000-1883:FFFF	256K	4	1/2/4	PCIe Authentication Micro Controller Unit Memory
1900:0000-1903:FFFF	256K	4	1/2/4	Secure Boot Controller Memory
1A00:0000-1A03:FFFF	256K	4	1/2/4	Cortex M3 Memory
1E60:0000-1E60:FFFF	64K	4	1/2/4	AHB Bus Controller
1E61:0000-1E61:FFFF	64K	4	1/2/4	PWM & Fan Tacho Controller
1E62:0000-1E62:FFFF	64K	4	1/2/4	Firmware SPI Memory Controller
1E63:0000-1E63:0FFF	4K	4	1/2/4	SPI1 Memory Controller
1E63:1000-1E63:1FFF	4K	4	1/2/4	SPI2 Memory Controller
1E65:0000-1E65:0FFF	4K	1/2/4	1/2/4	Ethernet MDC/MDIO Bus Controller
1E65:1000-1E65:1FFF	4K	1/2/4	1/2/4	I3C HDMA Controller
1E66:0000-1E66:FFFF	64K	1/2/4	1/2/4	10/100/1G Ethernet MAC Controller (MAC1)
1E67:0000-1E67:FFFF	64K	1/2/4	1/2/4	10/100/1G Ethernet MAC Controller (MAC3)
1E68:0000-1E68:FFFF	64K	1/2/4	1/2/4	10/100/1G Ethernet MAC Controller (MAC2)
1E69:0000-1E69:FFFF	64K	1/2/4	1/2/4	10/100/1G Ethernet MAC Controller (MAC4)
1E6A:0000-1E6A:0FFF	4K	4	1/2/4	USB2.0 Virtual Hub Controller
1E6A:1000-1E6A:17FF	2K	4	1/2/4	USB2.0 EHCI Host Controller (EHCI1)
1E6A:1800-1E6A:1FFF	2K	RO	1/2/4	PCIe USB2.0 EHCI Host Controller
1E6A:2000-1E6A:2FFF	4K	4	1/2/4	USB2.0 Device Controller
1E6A:3000-1E6A:3FFF	4K	4	1/2/4	USB2.0 EHCI Host Controller (EHCI2)
1E6B:0000-1E6B:FFFF	64K	4	1/2/4	USB1.1 UHCI Host Controller
1E6C:0000-1E6C:FFFF	64K	4	1/2/4	Software Interrupt Controller
1E6D:0000-1E6D:FFFF	64K	4	1/2/4	Hash & Crypto Engine (HACE)
1E6E:0000-1E6E:0FFF	4K	4	1/2/4	SDRAM Memory Controller (MMC)
1E6E:1000-1E6E:1FFF	4K	4	1/2/4	USB1.1 HID Controller
1E6E:2000-1E6E:2FFF	4K	4	1/2/4	System Control Unit (SCU)
1E6E:4000-1E6E:4FFF	4K	4	1/2/4	JTAG Master Controller
1E6E:6000-1E6E:6FFF	4K	4	1/2/4	SOC Display Controller (GFX)
1E6E:7000-1E6E:7FFF	4K	4	1/2/4	X-DMA Controller (XDMA)
1E6E:8000-1E6E:8FFF	4K	4	1/2/4	MCTP Controller (MCTP)
1E6E:9000-1E6E:9FFF	4K	4	1/2/4	ADC Controller
1E6E:B000-1E6E:BFFF	4K	4	1/2/4	DisplayPort Interface

to next page

from previous page

1E6E:C000-1E6E:CFFF	4K	4	1/2/4	RVAS VGA Snoop Controller
1E6E:D000-1E6E:D1FF	512B	4	1/2/4	PCIe Host Controller (RC/Bridge)
1E6E:D200-1E6E:D3FF	512B	4	1/2/4	PCIe Host Controller (RC)
1E6E:E000-1E6E:EFFF	4K	4	1/2/4	eSPI Controller
1E6E:F000-1E6E:FFFF	4K	4	1/2/4	Battery Backed SRAM
1E6F:0000-1E6F:0FFF	4K	4	1/2/4	AHB to PCIe Bus Bridge (H2X)
1E6F:2000-1E6F:2FFF	4K	4	1/2/4	Secure Boot Controller
1E6F:3000-1E6F:3FFF	4K	4	1/2/4	Internal Bridge Controller
1E6F:5000-1E6F:5FFF	4K	4	1/2/4	eMMC Boot Controller
1E6F:7000-1E6F:7FFF	4K	4	1/2/4	APB to PCIe RC Bridge
1E6F:8000-1E6F:8FFF	4K	4	1/2/4	PCIe RC X-DMA Controller (XDMA_RC)
1E6F:9000-1E6F:9FFF	4K	4	1/2/4	PCIe RC MCTP Controller (MCTP_RC)
1E6F:A000-1E6F:AFFF	4K	4	1/2/4	ECC/RSA Engine (ACRY)
1E6F:B000-1E6F:BFFF	4K	4	1/2/4	ESPI Master
1E70:0000-1E71:FFFF	64K	1/2/4	1/2/4	Video Engine
1E71:0000-1E71:FFFF	64K	4	1/2/4	ECC/RSA Engine Memory (ACRY)
1E72:0000-1E73:FFFF	128K	1/2/4	1/2/4	SRAM Memory Controller
1E74:0000-1E74:FFFF	64K	4	1/2/4	SD/SDIO Host Controller
1E75:0000-1E75:FFFF	64K	4	1/2/4	eMMC Controller
1E76:0000-1E76:01FF	512B	4	1/2/4	2D Graphics Engine
1E77:0000-1E77:0FFF	64K	4	1/2/4	AHB to PCIe RC bridge controller
1E78:0000-1E78:07FF	4K	4	1/2/4	GPIO Controller (Parallel GPIO)
1E78:0800-1E78:0FFF	4K	4	1/2/4	GPIO Controller (1.8V Parallel GPIO)
1E78:1000-1E78:1FFF	4K	4	1/2/4	Real Time Clock (RTC)
1E78:2000-1E78:2FFF	4K	4	1/2/4	Timer Controller #1 ~ #8
1E78:3000-1E78:3FFF	4K	4	1/2/4	UART Controller (UART1)
1E78:4000-1E78:4FFF	4K	4	1/2/4	UART Controller (UART5)
1E78:5000-1E78:5FFF	4K	4	1/2/4	Watchdog Timer (WDT)
1E78:7000-1E78:7FFF	4K	4	1/2/4	Virtual UART1 (VUART1)
1E78:8000-1E78:8FFF	4K	4	1/2/4	Virtual UART2 (VUART2)
1E78:9000-1E78:9FFF	4K	4	1/2/4	LPC Controller
1E78:A000-1E78:AFFF	4K	4	1/2/4	I2C/SMBus Controller
1E78:B000-1E78:BFFF	4K	4	1/2/4	PECI Controller
1E78:C000-1E78:CFFF	4K	4	1/2/4	APB to PCIe Bridge
1E78:D000-1E78:DFFF	4K	4	1/2/4	UART Controller (UART2)
1E78:E000-1E78:EFFF	4K	4	1/2/4	UART Controller (UART3)
1E78:F000-1E78:FFFF	4K	4	1/2/4	UART Controller (UART4)
1E79:0000-1E79:00FF	256B	4	1/2/4	UART Controller (UART6)
1E79:0100-1E79:01FF	256B	4	1/2/4	UART Controller (UART7)
1E79:0200-1E79:02FF	256B	4	1/2/4	UART Controller (UART8)
1E79:0300-1E79:03FF	256B	4	1/2/4	UART Controller (UART9)
1E79:0400-1E79:04FF	256B	4	1/2/4	UART Controller (UART10)

to next page

from previous page

1E79:0500-1E79:05FF	256B	4	1/2/4	UART Controller (UART11)
1E79:0600-1E79:06FF	256B	4	1/2/4	UART Controller (UART12)
1E79:0700-1E79:07FF	256B	4	1/2/4	UART Controller (UART13)
1E79:B000-1E79:BFFF	4K	4	1/2/4	FSI Controller
1E79:E000-1E79:EFFF	4K	4	1/2/4	UART DMA
1E79:F000-1E79:F7FF	4K	4	1/2/4	UART5 - debug
1E79:F800-1E79:FFFF	4K	4	1/2/4	UART1 - debug
1E7A:0000-1E7A:7FFF	32K	4	1/2/4	I3C Controller
1E7A:8000-1E7A:8FFF	4K	4	1/2/4	I2CS Controller
1E7C:0000-1E7C:1FFF	8K	4	1/2/4	RVAS LMEM
1E7C:8000-1E7C:8FFF	4K	4	1/2/4	RVAS
1E7D:0000-1E7D:8FFF	4K	4	1/2/4	FSI Controller AHB
1E7E:0000-1E7E:FFFF	64K	4	1/2/4	PCIe Host to BMC Controller
1E7F:0000-1E7F:FFFF	64K	4	1/2/4	PCI Express Authentication
2000:0000-2FFF:FFFF	256M	1/2/4	1/2/4	BMC SPI Flash Memory
3000:0000-3FFF:FFFF	256M	1/2/4	1/2/4	SPI1 Flash Memory
5000:0000-5FFF:FFFF	256M	1/2/4	1/2/4	SPI2 Flash Memory
6000:0000-7FFF:FFFF	512M	1/2/4	1/2/4	PCIe RC memory aperture
8000:0000-FFFF:FFFF	2048M	1/2/4	1/2/4	SDRAM

Note: Program access the IP using un-supported access mode will get an un-predictable result.

7.2 ARM CoreSight Address Map and Target ID

The following contains the list of addresses of the ARM CoreSight debug components present in the system.

Table 9: ARM CoreSight Address Mapping Table

Address in the debugger view	Component
9400:0000-9400:0FFF	ROM Table
9402:0000-9402:0FFF	Cortex-A7 CoreSight Integration Level ROM Table
9403:0000-9403:0FFF	Cortex-A7 Core0 DBG
9403:1000-9403:1FFF	Cortex-A7 Core0 PMU
9403:2000-9403:2FFF	Cortex-A7 Core1 DBG
9403:3000-9403:3FFF	Cortex-A7 Core1 PMU
9404:2000-9404:2FFF	CXCTI
9404:3000-9404:3FFF	CXTSGEN
E00F:F000-E00F:FFFF	Cortex-M3 CoreSight Integration Level ROM Table
E000:E000-E000:EFFF	Cortex-M3 v7M System Control Space (SCS)
E000:1000-E000:1FFF	Cortex-M3 v7M Data Watchpoint and Trace (DWT)
E000:2000-E000:2FFF	Cortex-M3 v7M FlashPatch and Breakpoint (FPB)
E004:2000-E004:2FFF	Cortex-M3 CoreSight Cross Trigger Interface (CTI)

ARM CoreSight Target ID has been fixed as 3f0f0f0fh or 0011_111100001111_0000_1111_0000111_1b. Please refer to the following table to see the corresponding fields.

Table 10: ARM CoreSight ID Table

CoreSight JTAG Target ID	Description	Value (binary)
31:28	REVISION	0011b
27:16	PART_NUM	111100001111b
15:12	tied to 0	0000b
11:8	JEDEC JEP-106 Continuation code	1111b
7:1	JEDEC JEP-106 Identity code	0000111b
0	tied to 1	1b

7.3 VGA Memory Space map to ARM Memory Space

VGA Size	Strap[3:2]	DRAM Size			
		256MB	512MB	1024MB	2048MB
8MB	0	8F80:0000 ~ 8FFF:FFFF	9F80:0000 ~ 9FFF:FFFF	BF80:0000 ~ BFFF:FFFF	FF80:0000 ~ FFFF:FFFF
16MB	1	8F00:0000 ~ 8FFF:FFFF	9F00:0000 ~ 9FFF:FFFF	BF00:0000 ~ BFFF:FFFF	FF00:0000 ~ FFFF:FFFF
32MB	2	8E00:0000 ~ 8FFF:FFFF	9E00:0000 ~ 9FFF:FFFF	BE00:0000 ~ BFFF:FFFF	FE00:0000 ~ FFFF:FFFF
64MB	3	8C00:0000 ~ 8FFF:FFFF	9C00:0000 ~ 9FFF:FFFF	BC00:0000 ~ BFFF:FFFF	FC00:0000 ~ FFFF:FFFF

8 Interrupt Source Table

Table 11: ARM Interrupt Source Table for Shared Peripheral Interrupt (SPI)

INT#	Description	Attribute
0	SDRAM interrupt	Sensitive high level trigger
1	Reserved(level 2)	Sensitive high level trigger
2	MAC1 interrupt	Sensitive high level trigger
3	MAC2 interrupt	Sensitive high level trigger
4	Hash/Crypto interrupt	Sensitive high level trigger
5	USB 2.0 Virtual Hub/Host interrupt	Sensitive high level trigger
6	X-DMA interrupt	Sensitive high level trigger
7	Video Engine interrupt	Sensitive high level trigger
8	UART5 (ARM console) interrupt	Sensitive high level trigger
9	USB 1.1 HID/ USB2.0 Host2 / USB2.0 Device interrupt	Sensitive high level trigger
10	USB 1.1 Host interrupt	Sensitive high level trigger
11	1.8V GPIO interrupt	Sensitive high level trigger
12	SCU interrupt	Sensitive high level trigger
13	RTC alarm interrupt	Sensitive high level trigger
14	Graphics CRT interrupt	Sensitive high level trigger
15	eMMC interrupt	Sensitive high level trigger
16	Timer 1 interrupt	Sensitive high level trigger
17	Timer 2 interrupt	Sensitive high level trigger
18	Timer 3 interrupt	Sensitive high level trigger
19	Timer 4 interrupt	Sensitive high level trigger
20	Timer 5 interrupt	Sensitive high level trigger
21	Timer 6 interrupt	Sensitive high level trigger
22	Timer 7 interrupt	Sensitive high level trigger
23	Timer 8 interrupt	Sensitive high level trigger
24	WDT alarm interrupt	Sensitive high level trigger
25	Graphics 2D interrupt	Sensitive high level trigger
26	MCTP interrupt	Sensitive high level trigger
27	JTAG1 Master interrupt	Sensitive high level trigger
28	AHBC interrupt	Sensitive high level trigger
29	AHB to PCIe Bus Bridge T interrupt	Sensitive high level trigger
30	DisplayPort interrput	Sensitive high level trigger
31	Reserved	Reserved
32	MAC3 interrupt	Sensitive high level trigger
33	MAC4 interrupt	Sensitive high level trigger
34	Reserved	Reserved
35	LPC interrupt	Sensitive high level trigger
36	Reserved	Reserved
37	Reserved	Reserved

to next page

from previous page

38	PECI interrupt	Sensitive high level trigger
39	FMC interrupt	Sensitive high level trigger
40	GPIO interrupt	Sensitive high level trigger
41	SCU interrupt 2	Sensitive high level trigger
42	eSPI interrupt	Sensitive high level trigger
43	SD/SDIO interrupt	Sensitive high level trigger
44	PWM/Tachometer interrupt	Sensitive high level trigger
45	System Wakeup Control	Sensitive high level trigger
46	ADC interrupt	Sensitive high level trigger
47	UART1 interrupt	Sensitive high level trigger
48	UART2 interrupt	Sensitive high level trigger
49	UART3 interrupt	Sensitive high level trigger
50	UART4 interrupt	Sensitive high level trigger
51	SGPIO Master 1 interrupt	Sensitive high level trigger
52	SGPIO Slave 1 interrupt	Sensitive high level trigger
53	JTAG Master interrupt 2	Sensitive high level trigger
54	MailBox interrupt	Sensitive high level trigger
55	AHBC interrupt 2	Sensitive high level trigger
56	UART DMA interrupt	Sensitive high level trigger
57	UART6 interrupt	Sensitive high level trigger
58	UART7 interrupt	Sensitive high level trigger
59	UART8 interrupt	Sensitive high level trigger
60	UART9 interrupt	Sensitive high level trigger
61	UART10 interrupt	Sensitive high level trigger
62	UART11 interrupt	Sensitive high level trigger
63	UART12 interrupt	Sensitive high level trigger
64	UART13 interrupt	Sensitive high level trigger
65	SPI DMA interrupt	Sensitive high level trigger
66	Reserved	Reserved
67	eSPI VW interrupt	Sensitive high level trigger
68	eSPI OOB interrupt	Sensitive high level trigger
69	eSPI Flash interrupt	Sensitive high level trigger
70	SGPIO Master 2 interrupt	Sensitive high level trigger
71	SGPIO Slave 2 interrupt	Sensitive high level trigger
72	GPIO interrupt for CM3	Sensitive high level trigger
73	GPIO interrupt for LPC	Sensitive high level trigger
74	ADC interrupt for CM3	Sensitive high level trigger
75	I2CS 0 interrupt	Sensitive high level trigger
76	I2CS 1 interrupt	Sensitive high level trigger
77	I2CS 2 interrupt	Sensitive high level trigger
78	I2CS 3 interrupt	Sensitive high level trigger

to next page

from previous page

79-96	Reserved	Reserved
97	FGB FIRQ interrupt	Sensitive high level trigger
98	FGB INTR interrupt	Sensitive high level trigger
99	HOST2BMC interrupt	Sensitive high level trigger
100	FSI 1 interrupt	Sensitive high level trigger
101	FSI 2 interrupt	Sensitive high level trigger
102	I3C 0 interrupt	Sensitive high level trigger
103	I3C 1 interrupt	Sensitive high level trigger
104	I3C 2 interrupt	Sensitive high level trigger
105	I3C 3 interrupt	Sensitive high level trigger
106	I3C 4 interrupt	Sensitive high level trigger
107	I3C 5 interrupt	Sensitive high level trigger
108	eSPI MMBI interrupt	Sensitive high level trigger
109	Reserved	Sensitive high level trigger
110	I2C 0 interrupt	Sensitive high level trigger
111	I2C 1 interrupt	Sensitive high level trigger
112	I2C 2 interrupt	Sensitive high level trigger
113	I2C 3 interrupt	Sensitive high level trigger
114	I2C 4 interrupt	Sensitive high level trigger
115	I2C 5 interrupt	Sensitive high level trigger
116	I2C 6 interrupt	Sensitive high level trigger
117	I2C 7 interrupt	Sensitive high level trigger
118	I2C 8 interrupt	Sensitive high level trigger
119	I2C 9 interrupt	Sensitive high level trigger
120	I2C 10 interrupt	Sensitive high level trigger
121	I2C 11 interrupt	Sensitive high level trigger
122	I2C 12 interrupt	Sensitive high level trigger
123	I2C 13 interrupt	Sensitive high level trigger
124	I2C 14 interrupt	Sensitive high level trigger
125	I2C 15 interrupt	Sensitive high level trigger
126	MAC Phy link 0 interrupt	Sensitive high level trigger
127	MAC Phy link 1 interrupt	Sensitive high level trigger
128	MAC Phy link 2 interrupt	Sensitive high level trigger
129	MAC Phy link 3 interrupt	Sensitive high level trigger
130	GPIO direct 0 interrupt	Sensitive high level trigger
131	GPIO direct 1 interrupt	Sensitive high level trigger
132	GPIO direct 2 interrupt	Sensitive high level trigger
133	GPIO direct 3 interrupt	Sensitive high level trigger
134	GPIO direct 4 interrupt	Sensitive high level trigger
135	GPIO direct 5 interrupt	Sensitive high level trigger
136	GPIO direct 6 interrupt	Sensitive high level trigger

to next page

from previous page

137	GPIO direct 7 interrupt	Sensitive high level trigger
138	KCS1 IBF interrupt	Sensitive high level trigger
139	KCS2 IBF interrupt	Sensitive high level trigger
140	KCS3 IBF interrupt	Sensitive high level trigger
141	KCS4 IBF interrupt	Sensitive high level trigger
142	KCS5 IBF interrupt	Sensitive high level trigger
143	iBT interrupt	Sensitive high level trigger
144	SNOOP 1/2 interrupt	Sensitive high level trigger
145	Post Code FIFO/DMA interrupt	Sensitive high level trigger
146	SIO SW interrupt	Sensitive high level trigger
147	VUART1 interrupt	Sensitive high level trigger
148	VUART2 interrupt	Sensitive high level trigger
149	Intrusion Interrupt (CHASI#)	Sensitive high level trigger
150-158	Reserved	Reserved
159	Inversed FWSPIWP# pin interrupt	Sensitive high level trigger
160	ECC/RSA engine interrupt	Sensitive high level trigger
161	Boot from eMMC engine interrupt	Sensitive high level trigger
162	Internal bridge interrupt	Sensitive high level trigger
163	RVAS VGA interrupt	Sensitive high level trigger
164	PCIe LPC interrupt	Sensitive high level trigger
165	PCIe LPC mailbox interrupt	Sensitive high level trigger
166	PCIe LPC SWC interrupt	Sensitive high level trigger
167	AHB to PCIe Bus Bridge L interrupt	Sensitive high level trigger
168	AHB to PCIe Bus Bridge H interrupt	Sensitive high level trigger
169-170	Reserved	Reserved
171	PCIe LPC SNOOP 1/2 interrupt	Sensitive high level trigger
172	PCIe LPC iBT interrupt	Sensitive high level trigger
173	PCIe LPC Post Code FIFO/DMA interrupt	Sensitive high level trigger
174	PCIe LPC KCS1 IBF interrupt	Sensitive high level trigger
175	PCIe LPC KCS2 IBF interrupt	Sensitive high level trigger
176	PCIe LPC KCS3 IBF interrupt	Sensitive high level trigger
177	PCIe LPC KCS4 IBF interrupt	Sensitive high level trigger
178	PCIe LPC KCS5 IBF interrupt	Sensitive high level trigger
179	PCIe LPC SIO SW interrupt	Sensitive high level trigger
180	PCIe LPC VUART1 interrupt	Sensitive high level trigger
181	PCIe LPC VUART2 interrupt	Sensitive high level trigger
182	Software interrupt #0 (CM3SIRQ0)	Sensitive high level trigger
183	Software interrupt #1 (CM3SIRQ1)	Sensitive high level trigger
184	Software interrupt #2 (CM3SIRQ2)	Sensitive high level trigger
185	Software interrupt #3 (CM3SIRQ3)	Sensitive high level trigger

to next page

from previous page

186	Software interrupt #4 (CM3SIRQ4)	Sensitive high level trigger
187	Software interrupt #5 (CM3SIRQ5)	Sensitive high level trigger
188	Software interrupt #6 (CM3SIRQ6)	Sensitive high level trigger
189	Software interrupt #7 (CM3SIRQ7)	Sensitive high level trigger
190	Software interrupt #8 (CM3SIRQ8)	Sensitive high level trigger
191	Software interrupt #9 (CM3SIRQ9)	Sensitive high level trigger
192	Software interrupt #10 (CM3SIRQ10)	Sensitive high level trigger
193	Software interrupt #11 (CM3SIRQ11)	Sensitive high level trigger
194	Software interrupt #12 (CM3SIRQ12)	Sensitive high level trigger
195	Software interrupt #13 (CM3SIRQ13)	Sensitive high level trigger
196	Software interrupt #14 (CM3SIRQ14)	Sensitive high level trigger
197	Reserved	Reserved

Table 12: SSP CM3 Interrupt Source Table for Shared Peripheral Interrupt (SPI)

INT#	Description	Attribute
0	SDRAM interrupt	Sensitive high level trigger
1	Reserved(level 2)	Sensitive high level trigger
2	MAC1 interrupt	Sensitive high level trigger
3	MAC2 interrupt	Sensitive high level trigger
4	Hash/Crypto interrupt	Sensitive high level trigger
5	USB 2.0 Virtual Hub/Host interrupt	Sensitive high level trigger
6	X-DMA interrupt	Sensitive high level trigger
7	Video Engine interrupt	Sensitive high level trigger
8	UART5 (ARM console) interrupt	Sensitive high level trigger
9	USB 1.1 HID/ USB2.0 Host2 / USB2.0 Device interrupt	Sensitive high level trigger
10	USB 1.1 Host interrupt	Sensitive high level trigger
11	1.8V GPIO interrupt	Sensitive high level trigger
12	SCU interrupt	Sensitive high level trigger
13	RTC alarm interrupt	Sensitive high level trigger
14	Graphics CRT interrupt	Sensitive high level trigger
15	eMMC interrupt	Sensitive high level trigger
16	Timer 1 interrupt	Sensitive high level trigger
17	Timer 2 interrupt	Sensitive high level trigger
18	Timer 3 interrupt	Sensitive high level trigger
19	Timer 4 interrupt	Sensitive high level trigger
20	Timer 5 interrupt	Sensitive high level trigger
21	Timer 6 interrupt	Sensitive high level trigger
22	Timer 7 interrupt	Sensitive high level trigger
23	Timer 8 interrupt	Sensitive high level trigger
24	WDT alarm interrupt	Sensitive high level trigger
25	Graphics 2D interrupt	Sensitive high level trigger
26	MCTP interrupt	Sensitive high level trigger
27	JTAG1 Master interrupt	Sensitive high level trigger
28	AHBC interrupt	Sensitive high level trigger
29	AHB to PCIe Bus Bridge interrupt	Sensitive high level trigger
30	DisplayPort interrput	Sensitive high level trigger
31	Reserved	Reserved
32	MAC3 interrupt	Sensitive high level trigger
33	MAC4 interrupt	Sensitive high level trigger
34	Reserved	Reserved
35	LPC interrupt	Sensitive high level trigger
36	Reserved	Reserved
37	Reserved	Reserved
38	PECI interrupt	Sensitive high level trigger

to next page

from previous page

39	FMC interrupt	Sensitive high level trigger
40	GPIO interrupt	Sensitive high level trigger
41	SCU interrupt 2	Sensitive high level trigger
42	eSPI interrupt	Sensitive high level trigger
43	SD/SDIO interrupt	Sensitive high level trigger
44	PWM/Tachometer interrupt	Sensitive high level trigger
45	System Wakeup Control	Sensitive high level trigger
46	ADC interrupt	Sensitive high level trigger
47	UART1 interrupt	Sensitive high level trigger
48	UART2 interrupt	Sensitive high level trigger
49	UART3 interrupt	Sensitive high level trigger
50	UART4 interrupt	Sensitive high level trigger
51	SGPIO Master interrupt	Sensitive high level trigger
52	SGPIO Slave interrupt	Sensitive high level trigger
53	JTAG Master interrupt 2	Sensitive high level trigger
54	MailBox interrupt	Sensitive high level trigger
55	AHBC interrupt 2	Sensitive high level trigger
56	UART DMA interrupt	Sensitive high level trigger
57	UART6 interrupt	Sensitive high level trigger
58	UART7 interrupt	Sensitive high level trigger
59	UART8 interrupt	Sensitive high level trigger
60	UART9 interrupt	Sensitive high level trigger
61	UART10 interrupt	Sensitive high level trigger
62	UART11 interrupt	Sensitive high level trigger
63	UART12 interrupt	Sensitive high level trigger
64	UART13 interrupt	Sensitive high level trigger
65	SPI DMA interrupt	Sensitive high level trigger
66	Reserved	Reserved
67	eSPI VW interrupt	Sensitive high level trigger
68	eSPI OOB interrupt	Sensitive high level trigger
69	eSPI Flash interrupt	Sensitive high level trigger
70	SGPIO Master 2 interrupt	Sensitive high level trigger
71	SGPIO Slave 2 interrupt	Sensitive high level trigger
72	GPIO interrupt for CM3	Sensitive high level trigger
73	GPIO interrupt for LPC	Sensitive high level trigger
74	ADC interrupt for CM3	Sensitive high level trigger
75	I2CS 0 interrupt	Sensitive high level trigger
76	I2CS 1 interrupt	Sensitive high level trigger
77	I2CS 2 interrupt	Sensitive high level trigger
78	I2CS 3 interrupt	Sensitive high level trigger
79-96	Reserved	Reserved

to next page

from previous page

97	FGB FIRQ interrupt	Sensitive high level trigger
98	FGB INTR interrupt	Sensitive high level trigger
99	HOST2BMC interrupt	Sensitive high level trigger
100	FSI 1 interrupt	Sensitive high level trigger
101	FSI 2 interrupt	Sensitive high level trigger
102	I3C 0 interrupt	Sensitive high level trigger
103	I3C 1 interrupt	Sensitive high level trigger
104	I3C 2 interrupt	Sensitive high level trigger
105	I3C 3 interrupt	Sensitive high level trigger
106	I3C 4 interrupt	Sensitive high level trigger
107	I3C 5 interrupt	Sensitive high level trigger
108	eSPI MMBI interrupt	Sensitive high level trigger
109	Reserved	Sensitive high level trigger
110	I2C 0 interrupt	Sensitive high level trigger
111	I2C 1 interrupt	Sensitive high level trigger
112	I2C 2 interrupt	Sensitive high level trigger
113	I2C 3 interrupt	Sensitive high level trigger
114	I2C 4 interrupt	Sensitive high level trigger
115	I2C 5 interrupt	Sensitive high level trigger
116	I2C 6 interrupt	Sensitive high level trigger
117	I2C 7 interrupt	Sensitive high level trigger
118	I2C 8 interrupt	Sensitive high level trigger
119	I2C 9 interrupt	Sensitive high level trigger
120	I2C 10 interrupt	Sensitive high level trigger
121	I2C 11 interrupt	Sensitive high level trigger
122	I2C 12 interrupt	Sensitive high level trigger
123	I2C 13 interrupt	Sensitive high level trigger
124	I2C 14 interrupt	Sensitive high level trigger
125	I2C 15 interrupt	Sensitive high level trigger
126	MAC Phy link 0 interrupt	Sensitive high level trigger
127	MAC Phy link 1 interrupt	Sensitive high level trigger
128	MAC Phy link 2 interrupt	Sensitive high level trigger
129	MAC Phy link 3 interrupt	Sensitive high level trigger
130	GPIO direct 0 interrupt	Sensitive high level trigger
131	GPIO direct 1 interrupt	Sensitive high level trigger
132	GPIO direct 2 interrupt	Sensitive high level trigger
133	GPIO direct 3 interrupt	Sensitive high level trigger
134	GPIO direct 4 interrupt	Sensitive high level trigger
135	GPIO direct 5 interrupt	Sensitive high level trigger
136	GPIO direct 6 interrupt	Sensitive high level trigger
137	GPIO direct 7 interrupt	Sensitive high level trigger
138	KCS1 IBF interrupt	Sensitive high level trigger

to next page

from previous page

139	KCS2 IBF interrupt	Sensitive high level trigger
140	KCS3 IBF interrupt	Sensitive high level trigger
141	KCS4 IBF interrupt	Sensitive high level trigger
142	KCS5 IBF interrupt	Sensitive high level trigger
143	iBT interrupt	Sensitive high level trigger
144	SNOOP 1/2 interrupt	Sensitive high level trigger
145	Post Code FIFO/DMA interrupt	Sensitive high level trigger
146	SIO SW interrupt	Sensitive high level trigger
147	VUART1 interrupt	Sensitive high level trigger
148	VUART2 interrupt	Sensitive high level trigger
149	Intrusion Interrupt (CHASI#)	Sensitive high level trigger
150-158	Reserved	Reserved
159	Inversed FWSPIWP# pin interrupt	Sensitive high level trigger
160	ECC/RSA engine interrupt	Sensitive high level trigger
161	Boot from eMMC engine interrupt	Sensitive high level trigger
162	Internal bridge interrupt	Sensitive high level trigger
163	RVAS VGA interrupt	Sensitive high level trigger
164	PCIe LPC interrupt	Sensitive high level trigger
165	LPC mailbox interrupt	Sensitive high level trigger
166	LPC SWC interrupt	Sensitive high level trigger
167-169	Reserved	Reserved
170	PCIe LPC? interrupt	Sensitive high level trigger
171	PCIe KCS7 IBF interrupt	Sensitive high level trigger
172	PCIe BT interrupt	Sensitive high level trigger
173	PCIe SNOOP PW interrupt	Sensitive high level trigger
174	PCIe KCS1 IBF interrupt	Sensitive high level trigger
175	PCIe KCS2 IBF interrupt	Sensitive high level trigger
176	PCIe KCS3 IBF interrupt	Sensitive high level trigger
177	PCIe KCS4 IBF interrupt	Sensitive high level trigger
178	PCIe KCS5 IBF interrupt	Sensitive high level trigger
179	PCIe Post Code FIFO interrupt	Sensitive high level trigger
180	PCIe VUART1 interrupt	Sensitive high level trigger
181	PCIe VUART2 interrupt	Sensitive high level trigger
182	Software interrupt #0 (CA7SIRQ0)	Sensitive high level trigger
183	Software interrupt #1 (CA7SIRQ1)	Sensitive high level trigger
184	Software interrupt #2 (CA7SIRQ2)	Sensitive high level trigger
185	Software interrupt #3 (CA7SIRQ3)	Sensitive high level trigger
186	Software interrupt #4 (CA7SIRQ4)	Sensitive high level trigger
187	Software interrupt #5 (CA7SIRQ5)	Sensitive high level trigger
188	Software interrupt #6 (CA7SIRQ6)	Sensitive high level trigger

to next page

from previous page

189	Software interrupt #7 (CA7SIRQ7)	Sensitive high level trigger
190	Software interrupt #8 (CA7SIRQ8)	Sensitive high level trigger
191	Software interrupt #9 (CA7SIRQ9)	Sensitive high level trigger
192	Software interrupt #10 (CA7SIRQ10)	Sensitive high level trigger
193	Software interrupt #11 (CA7SIRQ11)	Sensitive high level trigger
194	Software interrupt #12 (CA7SIRQ12)	Sensitive high level trigger
195	Software interrupt #13 (CA7SIRQ13)	Sensitive high level trigger
196	Software interrupt #14 (CA7SIRQ14)	Sensitive high level trigger
197	Reserved	Reserved

9 Hardware Strap Registers

SCU500 and SCU510 are hardware strap registers. They define the functions that are set after power on reset. The sources of the register values are:

- Physical strap pins
- OTPCFG in OTP memory
- Regsiter programming

9.1 Hardware Strap Registers

The strap registers, SCU500 and SCU510, define the functions that are set after power on reset. Please reference SCU500 and SCU510 for detail function descriptions. Each bit in the strap register has its strap sources. The source can be an external pin, OTP memory bit or combination of pin and OTP. Please reference to 9.5Hardware Strap Source Mappings for more details.

9.2 Hardware Strap Registers Protection

The strap registers are still programmable after power on reset unless the write protection are enabled. The write protection can be set per bit. OTPCFG28 and OTPCFG29 are OTP memory to enable the write protection for strap registers. SCU508 and SCU518 are software programmable registers to enable the write protection for strap registers.

9.3 Physical Strap Pins

The hardware strap registers strap the external pin voltage state or value in OTPSTRAP registers when SRST# reset input is low and keep the value unchanged after SRST# goes high. Please reference to section 2.3 for more details.

9.4 OTPSTRAP in OTP memory

The OTP memory is a one time programmable memory. The default values in OTP are blank. User can program te OTP memory after manufacture. The OTPCFG are inside OTP memory. They define some chip configuration options. Please reference section 61 for more details. OTPSTRAP is part of OTPCFG. It is one source of hardware strap registers.
OTPSTRAP are 64 bits internal signals. They cannot be read or written and are combinations of OTPCFG16-OTPCFG31.

OTPSTRAP[31: 0] = OTPCFG16 ^ OTPCFG18 ^ OTPCFG20 ^ OTPCFG22 ^ OTPCFG24 ^ OTPCFG26
OTPSTRAP[63:32] = OTPCFG17 ^ OTPCFG19 ^ OTPCFG21 ^ OTPCFG23 ^ OTPCFG25 ^ OTPCFG27

^ is bitwise exclusive-OR (XOR) operation. For example:
OTPSTRAP[1] = OTPCFG16[1] **XOR** OTPCFG18[1] **XOR** OTPCFG20[1] **XOR** OTPCFG22[1] **XOR** OTPCFG24[1] **XOR** OTPCFG26[1].
OTPCFG0-OTPCFG31 are inside OTP memory. Please reference to document ”SecureBoot_spec.pdf” for more details.

9.4.1 OTPSTRAP to SCU Strap Register Mappings

Here list the mappings for OTPSTRAP to SCU strap registers. The physical functions of OTPSTRAP are defined in SCU500 and SCU510. Please reference them for mor details.

Table 13: OTPSTRAP Mappings

OTPSTRAP[0]	SCU51C[1]	Enable Secure Boot by OTP Strap *
OTPSTRAP[1]	SCU500[2]	Boot from eMMC
OTPSTRAP[2]	SCU500[3]	Boot from debug SPI
OTPSTRAP[3]	SCU500[4]	Disable ARM CM3
OTPSTRAP[4]	SCU500[5]	VGA expansion ROM
OTPSTRAP[5]	SCU500[6]	MAC 1 RGMII mode
OTPSTRAP[6]	SCU500[7]	MAC 2 RGMII mode
OTPSTRAP[9:7]	SCU500[10:8]	CPU frequency
OTPSTRAP[11:10]	SCU500[12:11]	HCLK ratio[1:0]
OTPSTRAP[13:12]	SCU500[14:13]	VGA memory size[1:0]
OTPSTRAP[14]	SCU500[15]	Reserved
OTPSTRAP[15]	SCU500[16]	CPU/AXI clock frequency ratio selection
OTPSTRAP[16]	SCU500[17]	Disable ARM JTAG debug
OTPSTRAP[17]	SCU500[18]	VGA class code
OTPSTRAP[18]	SCU500[19]	Disable debug Interfaces (All of SCU0C8 & SCU0D8)
OTPSTRAP[19]	SCU500[20]	Boot from eMMC speed mode
OTPSTRAP[20]	SCU500[21]	Enable PCIe EHCI (default disable)
OTPSTRAP[21]	SCU500[22]	Disable ARM JTAG trust world debug
OTPSTRAP[22]	SCU500[23]	Disable Dedicated BMC function
OTPSTRAP[23]	SCU500[24]	Dedicate PCIe root complex reset
OTPSTRAP[24]	SCU500[25]	Disable watchdog to reset full chip
OTPSTRAP[26:25]	SCU500[27:26]	Internal Bridge Speed Selection
OTPSTRAP[28:27]	SCU500[29:28]	Select Reset Source of eMMC part
OTPSTRAP[29]	SCU500[30]	Disable RVAS function
OTPSTRAP[31:30]	NA	Reserved

OTPSTRAP[32]	SCU510[0]	MAC 3 RGMII mode
OTPSTRAP[33]	SCU510[1]	MAC 4 RGMII mode
OTPSTRAP[34]	SCU510[2]	SuperIO configuration address selection
OTPSTRAP[35]	SCU510[3]	Disable LPC to decode SuperIO 0x2E/0x4E address
OTPSTRAP[36]	SCU510[4]	Disable Debug Interfaces (All of SCUD8)
OTPSTRAP[37]	SCU510[5]	Enable ACPI function
OTPSTRAP[38]	SCU510[6]	Enable LPC mode
OTPSTRAP[39]	SCU510[7]	Enable SAFS mode
OTPSTRAP[40]	SCU510[8]	Enable Boot from Uart1 or Uart5 by Pin Strap
OTPSTRAP[41]	SCU510[9]	Enable boot SPI 3B address mode auto-clear
OTPSTRAP[42]	SCU510[10]	Enable SPI 3B/4B address mode auto detection
OTPSTRAP[43]	SCU510[11]	Enable boot SPI or eMMC ABR (second boot)
OTPSTRAP[44]	SCU510[12]	Boot SPI ABR Mode
OTPSTRAP[47:45]	SCU510[15:13]	Boot SPI flash size
OTPSTRAP[48]	SCU510[16]	Enable host SPI ABR (second boot)
OTPSTRAP[49]	SCU510[17]	Enable host SPI ABR mode select pin
OTPSTRAP[50]	SCU510[18]	Host SPI ABR Mode
OTPSTRAP[53:51]	SCU510[21:19]	Host SPI flash size
OTPSTRAP[54]	SCU510[22]	Enable boot SPI auxiliary control pins
OTPSTRAP[56:55]	SCU510[24:23]	Boot SPI CRTM size
OTPSTRAP[58:57]	SCU510[26:25]	Host SPI CRTM size
OTPSTRAP[59]	SCU510[27]	Enable host SPI auxiliary control pins
OTPSTRAP[60]	SCU510[28]	Enable GPIO Pass Through
OTPSTRAP[61]	SCU510[29]	Reserved
OTPSTRAP[62]	SCU510[30]	Enable Dedicate GPIO Strap Pins
OTPSTRAP[63]	NA	Reserved

9.5 Strap Source to SCU Strap Register Mappings

Here list the mappings for SCU strap registers. Each bit in the strap register has its strap sources. The source can be an external pin, OTP memory bit or combination of pin and OTP.

Table 14: Hardware Strap Source Mappings

SCU500[0]	= pin strap(!TXD5)
SCU500[1]	= SEC14[6] ‖ SCU510[8] SCU510[31] = pin strap(FWSPIMOSI) SCU51C[1] = OTPSTRAP[0] SEC14[6] = OTPCFG0[6] ? OTPCFG0[1] : (OTPCFG0[1] ‖ SCU51C[1] ‖ SCU510[31])
SCU500[2]	= OTPSTRAP[1]
SCU500[3]	= OTPSTRAP[2]
SCU500[4]	= OTPSTRAP[3]
SCU500[5]	= OTPSTRAP[4]
SCU500[6]	= OTPSTRAP[5]
SCU500[7]	= OTPSTRAP[6]
SCU500[10:8]	= OTPSTRAP[9:7]
SCU500[12:11]	= OTPSTRAP[11:10]
SCU500[14:13]	= OTPSTRAP[13:12]
SCU500[15]	= OTPSTRAP[14]
SCU500[16]	= OTPSTRAP[15]
SCU500[17]	= OTPSTRAP[16]
SCU500[18]	= OTPSTRAP[17]
SCU500[19]	= OTPSTRAP[18]
SCU500[20]	= OTPSTRAP[19]
SCU500[21]	= OTPSTRAP[20]
SCU500[22]	= OTPSTRAP[21]
SCU500[23]	= OTPSTRAP[22]
SCU500[24]	= OTPSTRAP[23]
SCU500[25]	= OTPSTRAP[24]
SCU500[27:26]	= OTPSTRAP[26:25]
SCU500[30:28]	= OTPSTRAP[29:27]
SCU500[31]	= OTPSTRAP[54]

SCU510[0]	= OTPSTRAP[32]
SCU510[1]	= OTPSTRAP[33]
SCU510[2]	= OTPSTRAP[34]
SCU510[3]	= OTPSTRAP[35]
SCU510[4]	= OTPSTRAP[36]
SCU510[5]	= OTPSTRAP[37]
SCU510[6]	= OTPSTRAP[38]
SCU510[7]	= OTPSTRAP[39]
SCU510[8]	= SCU510[30] ? OTPSTRAP[40] : FWSPICK
SCU510[9]	= OTPSTRAP[41]
SCU510[10]	= OTPSTRAP[42]
SCU510[11]	= OTPSTRAP[43]
SCU510[12]	= OTPSTRAP[44]
SCU510[13]	= SCU510[30] ? GPIOZ6 : OTPSTRAP[45]
SCU510[14]	= SCU510[30] ? GPIOZ7 : OTPSTRAP[46]
SCU510[15]	= SCU510[30] ? FWSPICK : OTPSTRAP[47]
SCU510[16]	= OTPSTRAP[48]
SCU510[17]	= OTPSTRAP[49]
SCU510[18]	= OTPSTRAP[50]
SCU510[19]	= SCU510[30] ? GPIOZ3 : OTPSTRAP[51]
SCU510[20]	= SCU510[30] ? GPIOZ4 : OTPSTRAP[52]
SCU510[21]	= SCU510[30] ? GPIOZ5 : OTPSTRAP[53]
SCU510[22]	= OTPSTRAP[54]
SCU510[24:23]	= OTPSTRAP[56:55]
SCU510[25]	= SCU510[30] ? GPIOY4 : OTPSTRAP[57]
SCU510[26]	= SCU510[30] ? GPIOY5 : OTPSTRAP[58]
SCU510[27]	= OTPSTRAP[59]
SCU510[28]	= OTPSTRAP[60]
SCU510[29]	= OTPSTRAP[61]
SCU510[30]	= OTPSTRAP[62]
SCU510[31]	= FWSPIMOSI

9.6 OTPSTRAP Description

The OTPSTRAP signals are strapped by SCU500 and SCU510 strap registers. The physical functions are controlled by strap registers. Please refrence to SCU500 and SCU510 for more details of strap controlled functions. (The OTPSTRAP Description in previous datasheet revision was removed because it is duplicated with SCU500 and SCU510 in section 24 System Control Unit (SCU).

9.7 Straps for Secure Boot Enable

The Key registers and straps that play into controlling the Secure Boot enablement are:

- OTPSTRAP[0] : Secure boot Enable using OTPSTRAP, readable at SCU51C[1]
- FWSPIMOSI Hardware pin strap V Secure boot Enable using FWSPIMOSI hardware pin, readable at SCU510[31]

- OTPCFG0[1] : One bit master enable for Secure boot. Setting this bit to 1 will have the secure boot enabled irrespective of other enable sources.
- SEC14[6] : Master bit to indicate status of the secure boot enable, computed as below.
- SEC14[6] = OTPCFG0[6] ? OTPCFG0[1] : (OTPCFG0[1] || SCU51C[1] || SCU510[31])
- AST2600 Secure Boot Enable supports the capability to enable this feature in multiple ways for supporting different usage cases.
- Always enable : In usa cases where the customer wants to configure the chip one time in the OTP and enforce secure boot always can do so by programming the OTPCFG0[1]
- Enable by external pin strap : In usa cases where the customer wants to control the secure boot enable by external HW strap pin(FWSPIMOSI), i.e. only enable secure boot mode when the external strap is selected to enable it. This is achieved by setting OTPCFG0[1]=0 and OTPCFG0[6]=0 and OTPSTRAP[0]=0
- Enable by OTPSTRAP : In this usa case customer may initially want to have the secure boot disabled or controlled by external pin strap during development phase and later on wants to programmatically enable secure boot. This is achieved by OTPCFG0[1]=0 and OTPCFG0[6]=0 and OTPSTRAP[0]=1

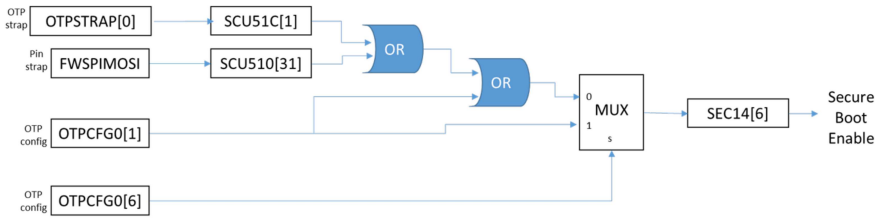


Figure 39: Diagram of Secure Boot enablement control

10 Boot Flow and CPU feature

10.1 Boot Flow

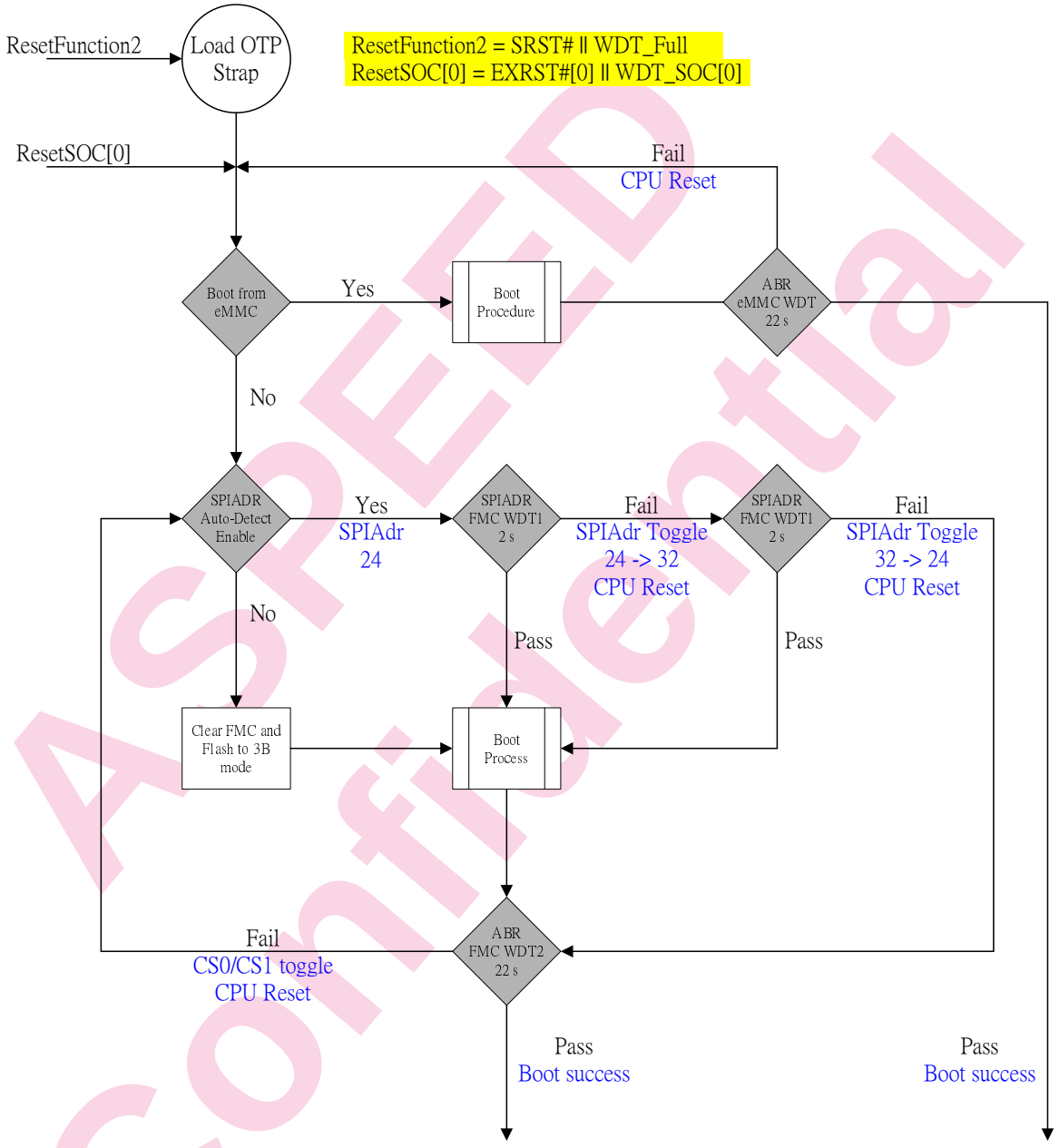
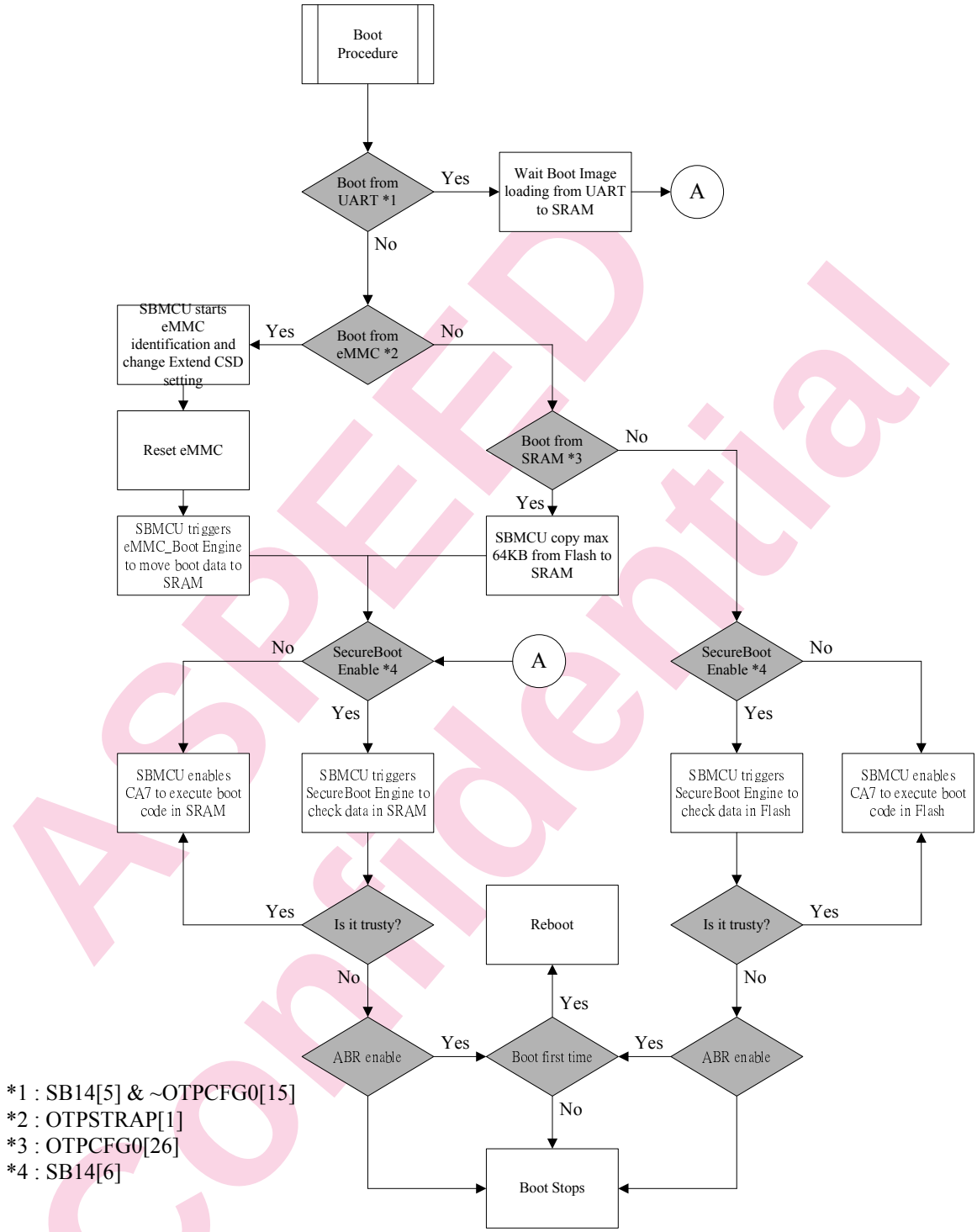


Figure 40: Top view of Boot-Up sequence



*1 : SB14[5] & ~OTPCFG0[15]
*2 : OTPSTRAP[1]
*3 : OTPCFG0[26]
*4 : SB14[6]

SB14[5] = (pin strap(FWSPICK) | SCU510[8])
SB14[6] = OFTCFG0[6] ? OTPCFG0[1] : (pin strap(FWSPIMOSI) | OTPSTRAP[0] | OTPCFG0[1])
SCU510[8] = OTPSTRAP[30] ? OTPSTRAP[40] : pin strap(FWSPICK)

Figure 41: Detail of Boot-Up sequence

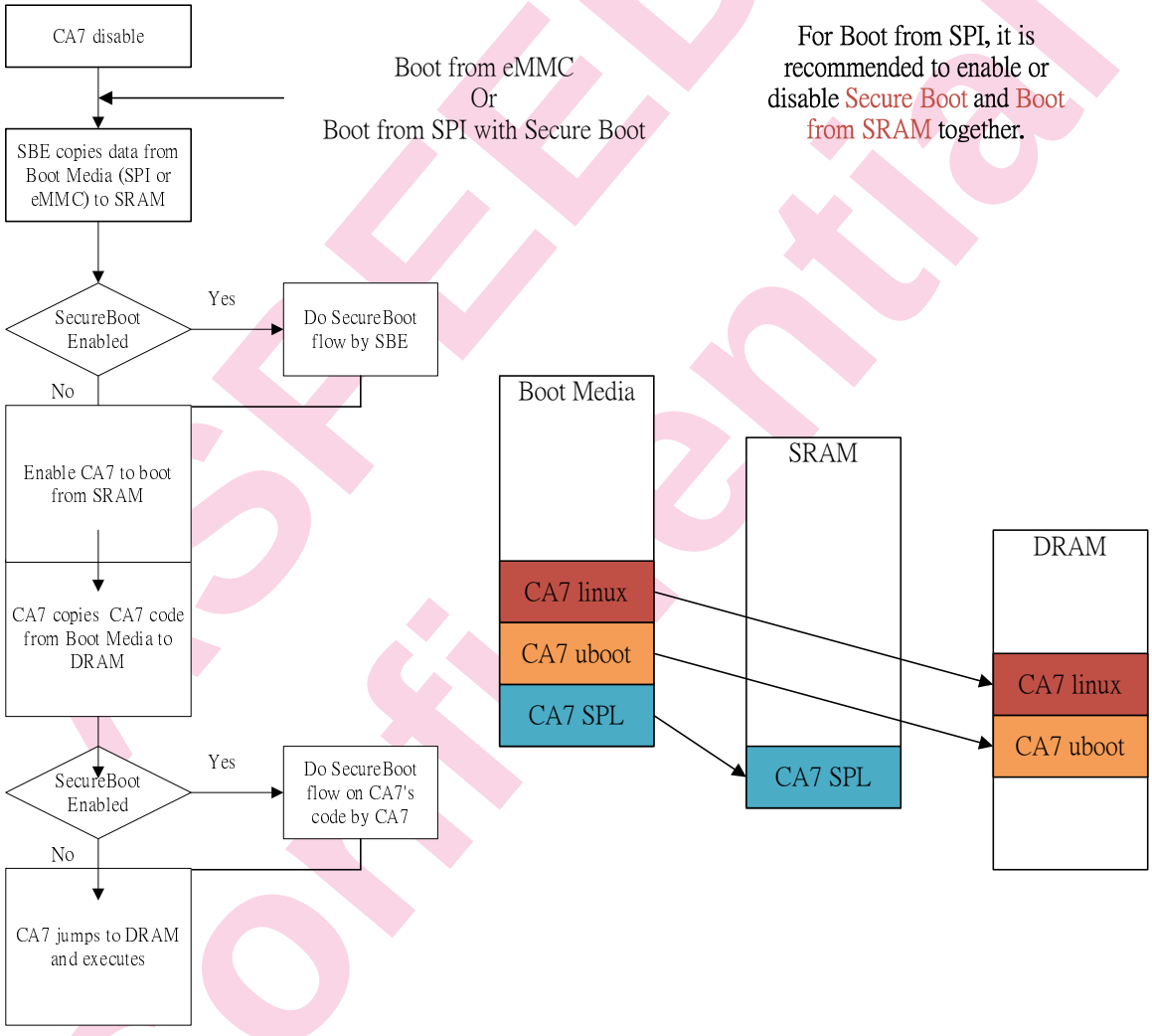


Figure 42: Description of Boot-Up sequence

10.2 AST2600 TrustZone

AST2600 includes an ARM dual core Cortex A7 processer with TrustZone capabilities. Refer to the link at Arme Developer site for the background on the ARM TrustZone description and usage cases to establist security https://developer.arm.com/ip-products/security-ip/trustzone/trustzone-for-cortex-a

The TrustZone involves establishing security by partiotioning as secure world and non secure world that are separated in hardware. The secure world is established by carving out memory and peripherals and running a separate operating system referred as the Trusted Execution Environment (TEE). The non secure world would be the regular feature rich application environment running the regular OS refereed to as the Rich OS Environment(REE).

10.2.1 AST2600 TrustZone Feature

AST2600 is equipped with a memory protection unit and a peripheral protection unit to fulfill the security isolation requirement of TrustZone. This allows establishing a hardware restriction for TEE resources, including memory and peripheral devices, not be accessible by REE. The related registers in AST2600 are AHBC200 - AHBC3B8 can be found in the datasheet. In the following sections, couple of example scenarios is illustrated.

TEE and REE IP Identification

AST2600 SOC is composed of multiple IPs, where some are master and others are slave. The hardware allows programming each IP as secure or non-secure one. This allows the interaction between TEE and REE to be monitored and even restricted.

By default, all of the masters IPs on AST2600 are treated as secure. Firmware developers have to configure which IP belongs to REE through the register AHBC200. For example, if the UART DMA engine should not be able to access TEE resource, then AHBC200[24] should be set to disable the secure privilege of UART DMA engine. The configuration should be done at early booting stage. Once the configuration is done, the value of AHBC200 can be protected by AHBC204[28] to defend against REE instrusion. Note that once the write protection enforced by AHBC204 neither TEE nor REE are able to modify it. Only power-on reset can restore the configuration.

TEE Memory Access Protection

The TrustZone memory management requires partitioning the physical memory into secure and non-secure regions. This allows monitoring a non-secure access to secure memory regions can be captured and aborted. AST2600 provides a set of memory protection registers AHBC300 - AHBC3B8 to achieve the memory paritition as well as the protection.
There are total 12 configurable memory regions supported by AST2600 . When a memory region is configured, firmware developers have to further decide whose memory request is permitted to access according to the DRAM REQ ID. Note that the memory protection region granularity is on the basis of 0x1000 bytes.

Table 15: Memory Request Priority Table

Priority	Request ID	Group	Request Source
1	REQ0	1	VGA hardware cursor read
2	REQ1	1	VGA CRT read
3	REG2	1	SOC Display Controller read
4	REQ3	1	PCI-E bus1 read/write
5	REQ4	1	Video high priority write
6	REQ5	1	CPU read/write
7	REQ6	1	SLI read/write
8	REQ7	1	PCI-E bus2 read/write
9	REQ8	2-1	USB2.0 Hub/EHCI1 DMA read/write
10	REQ9	2-1	USB2.0 Device/EHCI2 DMA read/write
11	REQ10	2-1	USB1.1 UHCI host read/write
12	REQ11	2-1	AHB bus read/write
13	REQ12	2-1	CM3 data read/write
14	REQ13	2-1	CM3 instruction read
15	REQ14	2-1	MAC0 DMA read/write
16	REQ15	2-1	MAC1 DMA read/write
17	REQ16	2-1	SDIO DMA read/write
18	REQ17	2-1	Pilot Engine read/write
19	REQ18	2-1	XDMA1 read/write
20	REQ19	2-1	MCTP1 read/write
21	REQ20	2-1	Video flag read/write
22	REQ21	2-1	Video low priority write
23	REQ22	2-1	2D engine data read/write
24	REQ23	2-1	Encryption engine read/write
25	REQ24	2-1	MCTP2 read/write
26	REQ25	2-1	XDMA2 read/write
27	REQ26	2-1	ECC/RSA read/write

Example: protect TEE OS secure memory

Consider TEE OS is loaded into the physical memory region 0x90000000 - 0x9FFFFFFF. In addition, assume only CPU running with SCR.NS=0 is allowed to access the memory region. The following steps are to secure such region from REE access.

1. AHBC300[31:12] = 90000h
2. AHBC304[31:12] = 9FFFFh
3. AHBC308[5] = 1
4. AHBC300[0] = 1
5. AHBC204[16] = 1

Steps 1 and 2 together define the protected memory region ranges from 0x90000000 - 0x9FFFFFFF. Step 3 allows only DRAM request issued from CPU to access the region. Step 4 enables the protection. The last step lockdown this security configuration.

TEE Peripheral Access Protection

Similar to memory protection, peripheral resource for TEE use should also be protected from REE access. This can be achieved by AHBC280 - AHBC28C. If only REE write access is prohibited, AHBC290 - AHBC29C apply.

Example: protect UART9 secure peripheral
Consider UART9 is reserved for TEE use only. The following steps is conducted to secure such region from REE access.

1. AHBC28C[0] = 1
2. AHBC204[11] = 1

The first step ensures that only master with secure privilege can access UART10 control registers and the second step lockdown the configuration.

11 UART Debug Controller

AST2600 integrates two UART debug controllers which can input commands to AST2600 by using simple terminal program without the assistance of CPU. The UART debug controllers are indenpedent to other controllers, like UART controllers and UART boot mode.

There are two debug ports physically attached on UART1 and UART5.

The debug UART ports are default in normal UART function mode. They keep snooping the UART RX pin with 1200 baud rate. If they received UART debug port password, they will switch the port to UART debug controller and the UART port will enter UART port will enter the debug mode.

For security issue, these features can be disabled by setting register SCU0C8[1] = 1 to disable port UART5 debug and SCU0D8[3] = 1 to disable port UART1 debug.

To enter the UART debug mode, follow the steps as below on terminal program then the UART port will enter the debug mode:

1. Set terminal baud rate to 1200
2. Paste the password at the bottom of this page
3. When the terminal shows '$ ', it indicates that the UART port has switched to the debug mode.
4. Set terminal baud rate to 115200 and start debug

Under UART debug mode, it supports following command sets:

Command Format	Description
i address	Read 1byte from the [address]
o address b_data	Write 1byte [b_data] to the [address]
r address	Read 4byte from the [address]
w address d_data	Write 4byte [d_data] to the [address]
d address length	Read 4byte data starting from [address] to [address+length]
t address length [Key Enter] list_data	Write [list_data] to the destination starting from [address] to [address+length] with 1byte mode command
u address length [Key Enter] list_data	Write [list_data] to the destination starting from [address] to [address+length] with 4byte mode command
q	Quit debug mode
[Key BackSpace]	Press [BackSpace] key, move cursor to left
[Key Esc]	Press [Esc] key, ignore latest command and change to next line

The above commands and arguements must follow below format:

'address'	is 32-bit hexadecimal mode (no prefix 0x). Maximum 8 digits. No leading zeros.
'b_data'	is 8-bit hexadecimal mode (no prefix 0x). Maximum 2 digits. No leading zeros.
'd_data'	is 32-bit hexadecimal mode (no prefix 0x). Maximum 8 digits. No leading zeros.
'length'	is 32-bit hexadecimal mode (no prefix 0x). Maximum 8 digits. No leading zeros.
'list_data'	is a list of binary data. The list length is described by 'length' in the unit of byte. All bytes in 'list_data' are consecutive without any delimiter character.
' '	is required to seperate command and arguments.
[Key Enter]	is added at the end of a command to fire the command. It contains 2 binary codes 0x0d and 0x0a.

5z&0VK{@`HW}H~V310=l=JB+M]IV-f;Sz98XfCA&Rp)i|Jo=2?IBN$QaQ2"Kb|Ov

Part III
Function Registers

12 Register Type Abbreviations

The following abbreviations are used to indicate the Register Type throughout this document.
RW = Capable of Read & Write.
R = Capable of Read
W = Capable of Write
RO = Read Only. Writing to the register will not cause any effects.
WO = Write Only. Reading from the register returns unpredictable values.
WT = Write to Trigger. Writing a specified value to a register to trigger some hardware operations.
W1C = Write 1 to Clear. Writing 1 to a register to clear its value to 0. Writing 0 has no effect.
W1S = Write 1 to Set. Writing 1 to a register to set its value to 1. Writing 0 has no effect.
W1T = Write 1 to Trigger. Writing 1 to a register to trigger some hardware operations. Writing 0 has no effect.
RWT = R + WT
RW1C = R + W1C
RW1S = R + W1S
OTP = One time programmable. It can be programmed once during whole chip life cycle.

13 AHB Bus Controller (AHBC)

13.1 Overview

Advanced High-performance Bus Controller (AHBC) supports a mechanism, including a priority arbiter, an address decoder and a data multiplexer, to control the overall operations of Advanced High-performance Bus (AHB), which is the main system bus for ARM CPU to communicate with the related peripherals. The priority arbiter, by round-robin arbitration scheme, assigns which bus master gets the right to access AHB for the moment. Each bus master has its own REQUEST/GRANT interface to the priority arbiter. The address decoder performs a centralized address decoding function.

AHBC also provide remapping mechanism to speed up the access time of program code.

There is a new feature added to log a specific address read/write data into a log buffer. The log buffer can be placed in DRAM or SRAM. The log buffer size supported is from 4K bytes $\sim$ 1M bytes. The log buffer is helpful for firmware debugging. Such as logging the firmware console output message. Or it can be used to monitor a specific control register for debugging the firmware behavior.

Base address of AHBC = 0x1E60_0000
Physical address = (Base address of AHBC) + Offset

AHBC00: Protection Key Register
AHBC40: AHB Bus Command Recording Control/Status Register
AHBC44: Log Buffer Base/Write Pointer Register
AHBC48: Polling Address Register
AHBC4C: Hardware FIFO Status Register
AHBC5C: Hardware FIFO Merge Register
AHBC60: Hardware FIFO Stage #0 Register
AHBC64: Hardware FIFO Stage #1 Register

AHBC68: Hardware FIFO Stage #2 Register
AHBC6C: Hardware FIFO Stage #3 Register
AHBC70: Hardware FIFO Stage #4 Register
AHBC74: Hardware FIFO Stage #5 Register
AHBC78: Hardware FIFO Stage #6 Register
AHBC7C: Hardware FIFO Stage #7 Register
AHBC80: Priority Control Register
AHBC84: Interrupt Control/Status Register
AHBC88: AHB Bus Target Disable Control Register
AHBC8C: Address Remapping Register
AHBC90: Watchdog Counter Status Register
AHBC94: Watchdog Counter Reload Value Register
AHBC9C: SCU Password Snoop Register
AHBCC0: Internal AHB Bus Flush Enable Register
AHBCC4: VIC CPU Lock Control Register
AHBC200: TrustZone AHBC Master Secure Disable Register
AHBC204: TrustZone Regiters Write Protection
AHBC208: AHBC and TrustZone Regiters Reset Selection Register
AHBC240: Region Access Protection Register 1
AHBC244: Region Access Protection Register 2
AHBC248: Region Access Protection Register 3
AHBC24C: Region Access Protection Register 4
AHBC250: Region Write Protection Register 1
AHBC254: Region Write Protection Register 2
AHBC258: Region Write Protection Register 3
AHBC25C: Region Write Protection Register 4
AHBC280: TrustZone Access Protection Register 1
AHBC284: TrustZone Access Protection Register 2
AHBC288: TrustZone Access Protection Register 3
AHBC28C: TrustZone Access Protection Register 4
AHBC290: TrustZone Write Protection Register 1
AHBC294: TrustZone Write Protection Register 2
AHBC298: TrustZone Write Protection Register 3
AHBC29C: TrustZone Write Protection Register 4
AHBC300: TrustZone Memory Region 1 Protection Start Register
AHBC304: TrustZone Memory Region 1 Protection End Register
AHBC308: TrustZone Memory Region 1 Protection Master Register
AHBC310: TrustZone Memory Region 2 Protection Start Register
AHBC314: TrustZone Memory Region 2 Protection End Register
AHBC318: TrustZone Memory Region 2 Protection Master Register
AHBC320: TrustZone Memory Region 3 Protection Start Register
AHBC324: TrustZone Memory Region 3 Protection End Register
AHBC328: TrustZone Memory Region 3 Protection Master Register
AHBC330: TrustZone Memory Region 4 Protection Start Register
AHBC334: TrustZone Memory Region 4 Protection End Register
AHBC338: TrustZone Memory Region 4 Protection Master Register
AHBC340: TrustZone Memory Region 5 Protection Start Register
AHBC344: TrustZone Memory Region 5 Protection End Register
AHBC348: TrustZone Memory Region 5 Protection Master Register
AHBC350: TrustZone Memory Region 6 Protection Start Register
AHBC354: TrustZone Memory Region 6 Protection End Register
AHBC358: TrustZone Memory Region 6 Protection Master Register
AHBC360: TrustZone Memory Region 7 Protection Start Register
AHBC364: TrustZone Memory Region 7 Protection End Register
AHBC368: TrustZone Memory Region 7 Protection Master Register
AHBC370: TrustZone Memory Region 8 Protection Start Register
AHBC374: TrustZone Memory Region 8 Protection End Register

AHBC378: TrustZone Memory Region 8 Protection Master Register
AHBC380: TrustZone Memory Region 9 Protection Start Register
AHBC384: TrustZone Memory Region 9 Protection End Register
AHBC388: TrustZone Memory Region 9 Protection Master Register
AHBC390: TrustZone Memory Region 10 Protection Start Register
AHBC394: TrustZone Memory Region 10 Protection End Register
AHBC398: TrustZone Memory Region 10 Protection Master Register
AHBC3A0: TrustZone Memory Region 11 Protection Start Register
AHBC3A4: TrustZone Memory Region 11 Protection End Register
AHBC3A8: TrustZone Memory Region 11 Protection Master Register
AHBC3B0: TrustZone Memory Region 12 Protection Start Register
AHBC3B4: TrustZone Memory Region 12 Protection End Register
AHBC3B8: TrustZone Memory Region 12 Protection Master Register

13.2 Features

- Directly connected to internal AHB bus
- AHB master and slave controller
- AHB bus multiplexer
- AHB slave address decoder
- AHB master controller with two-level arbitration (round-robin arbitration for each arbitration level)
- AHB memory address remapping control with register-write protection
- AHB bus lock prevention watchdog function

13.3 Registers : Base Address = 0x1E60:0000

Offset: 00h	AHBC00: Protection Key Register	Init = 0

Bit	R/W	Reset	Description
31:0	RW	Rst211	**Protect Key** Write 0xAEED_1A03: set to '1', key unlocked. Write Others: set to '0', key locked. The key protected registers as below: AHBC80, AHBC84[15:0], AHBC88, AHBC8C, AHBC94

Offset: 40h	AHBC40: AHB Bus Command Recording Control/Status Register	Init = 0

Bit	R/W	Reset	Description
31:11	RO	-	**Reserved (0)**
10:8	RW	RstFull	**Buffer Size** 000: 4KB 001: 8KB 010: 16KB 011: 32KB 100: 128KB 101: 256KB 110: 512KB 111: 1024KB
7	RO	-	**Reserved (0)**

to next page

from previous page

6:4	RW	RstFull	**Polling Data Size** 000: Byte, bit[7:0] 001: Byte, bit[15:8] 010: Byte, bit[23:16] 011: Byte, bit[31:24] 100: Word, bit[15:0] 101: Word, bit[31:16] 11x: DWord, bit[31:0]
3	RO	-	**Reserved (0)**
2	RW	RstFull	**Flush Temp Polling Buffer** Flush the data from hardware FIFO (AHBC60 ∼ AHBC7C) to log buffer. If the last data (AHBC5C) is not DWord, it will not be flushed to the log buffer.
1	RW	RstFull	**Polling Mode** 0: polling read command 1: polling write command
0	RW	RstFull	**Polling Enable** 0: Disable the polling function will reset the FIFO 1: Enable polling mode

Offset: 44h **AHBC44: Log Buffer Base/Write Pointer Register** **Init = X**

Bit	R/W	Reset	Description
31:2	RW	RstFull	**Buffer Base Address/Write Pointer** Buffer Size = 000 => Base[31:12], Wptr[11:2] Buffer Size = 001 => Base[31:13], Wptr[12:2] Buffer Size = 010 => Base[31:14], Wptr[13:2] Buffer Size = 011 => Base[31:15], Wptr[14:2] Buffer Size = 100 => Base[31:16], Wptr[15:2] Buffer Size = 101 => Base[31:17], Wptr[16:2] Buffer Size = 110 => Base[31:18], Wptr[17:2] Buffer Size = 111 => Base[31:19], Wptr[18:2] The write pointer points to the next write address. So the actual write data < write pointer.
1	RO	-	**Reserved (0)**
0	RW1C	RstFull	**Log Buffer Ring Back Indicator** 0: Log buffer not over the maximum size. 1: Log buffer reach the maximum size and ring back. Write operation to clear this flag.

Note :
The log buffer operates as a ring buffer, starts from the init value, and ring back after reach the maximum buffer size.
To guarantee normal operation, this register can be written only when Polling is disabled.

Offset: 48h **AHBC48: Polling Address Register** **Init = X**

Bit	R/W	Reset	Description
31:2	RW	RstFull	**Polling Address**
1:0	RO	-	**Reserved (0)**

Note :
To guarantee normal operation, this register can be written only when Polling is disabled.

Offset: 4Ch	AHBC4C: Hardware FIFO Status Register	Init = 0

Bit	R/W	Reset	Description
31:16	RO	-	**Reserved (0)**
15:12	RO	RstFull	**FIFO Length (FIFO Stage)**
11	RO	-	**Reserved (0)**
10:8	RO	RstFull	**FIFO Write Pointer (FIFO Stage)**
7	RO	-	**Reserved (0)**
6:4	RO	RstFull	**FIFO Read Pointer (FIFO Stage)**
3:2	RO	-	**Reserved (0)**
1:0	RO	RstFull	**Merge Buffer Write Index (Byte)**

Offset: 5Ch	AHBC5C: Hardware FIFO Merge Register	Init = X

Bit	R/W	Reset	Description
31:0	RO	-	**FIFO Merge Data**

Offset: 60h	AHBC60: Hardware FIFO Stage #0 Register	Init = X
Offset: 64h	AHBC64: Hardware FIFO Stage #1 Register	Init = X
Offset: 68h	AHBC68: Hardware FIFO Stage #2 Register	Init = X
Offset: 6Ch	AHBC6C: Hardware FIFO Stage #3 Register	Init = X
Offset: 70h	AHBC70: Hardware FIFO Stage #4 Register	Init = X
Offset: 74h	AHBC74: Hardware FIFO Stage #5 Register	Init = X
Offset: 78h	AHBC78: Hardware FIFO Stage #6 Register	Init = X
Offset: 7Ch	AHBC7C: Hardware FIFO Stage #7 Register	Init = X

Bit	Attr.	Reset	Description
31:0	RO	-	**FIFO Data**

Offset: 80h	AHBC80: Priority Control Register	Init = 0x1

Bit	R/W	Reset	Description
31:10	RO	-	**Reserved (0)**

to next page

from previous page

9:0	RW	Rst211	**Priority Level Selection** Bit n represents the level of master n on AHB 0: lower priority level 1: higher priority level AHBC total support up to 9 bus masters. The arbiter supports a two-level mechanism to arbitrate master requests. Each master can be programmed to a higher level or lower level. The following table shows the bit assignment of priority control. Bit[8] : UART-DMA Bit[7] : eSPI Bit[6] : Coprocessor Bit[5] : SPI Slave to AHB Bridge Bit[4] : Firmware Flash DMA Bit[3] : LPC to AHB Bridge Bit[2] : PCI-E to AHB Bridge Bit[1] : ARM CPU Bit[0] : AHBC

Offset: 84h **AHBC84: Interrupt Control/Status Register** **Init = 0x0**

Bit	R/W	Reset	Description
31:17	RO	-	**Reserved (0)**
16	RW1C	Rst211	**Bus lock interrupt status**
15:1	RO	-	**Reserved (0)**
0	RW	Rst211	**Enable bus lock interrupt**

Note :
Write '1' to the specific status bits can clear it to '0'.

Offset: 88h **AHBC88: AHB Bus Target Disable Control Register** **Init = 0x0**

Bit	R/W	Reset	Description
31:24	RO	-	**Reserved (0)**
23	RW	Rst211	**Disable target PCIe Security Controller**
22	RW	Rst211	**Disable target Host 2 BMC bridge**
21	RW	Rst211	**Disable target DP MCU**
20	RW	Rst211	**Disable target ECC/RSA Engine**
19	RW	Rst211	**Disable target PCIe RC memory aperture**
18	RW	Rst211	**Disable target AHB to PCIe RC bridge controller**
17	RW	Rst211	**Disable target HACE**
16	RW	Rst211	**Disable target CM3**
15	RW	Rst211	**Disable target USB2.0 EHCI Host Controller on PCIe**
14	RW	Rst211	**Disable targets of APB bus part 1**
13	RW	Rst211	**Disable target DP MCU**
12	RW	Rst211	**Disable target AHB-to-DRAM**
11	RW	Rst211	**Disable target Firmware Flash**
10	RW	Rst211	**Disable target eMMC**

to next page

from previous page

9	RW	Rst211	**Disable target 2D Engine**
8	RW	Rst211	**Disable target Video Engine**
7	RW	Rst211	**Disable target APB Bridge**
6	RW	Rst211	**Disable target SWVIC**
5	RW	Rst211	**Disable target USB2.0 port1**
4	RW	Rst211	**Disable target USB Host**
3	RW	Rst211	**Disable target MAC1/MAC2**
2	RW	Rst211	**Disable target RVAS**
1	RW	Rst211	**Disable target SRAM**
0	RW	Rst211	**Disable target RVAS LMEM**
Note : This specific register bit will be set to '1' when watchdog timeout if the AHBC is locked by the target.			

Offset: 8Ch **AHBC8C: Address Remapping Register** **Init = 0**

Bit	R/W	Reset	Description
31:8	R	-	**Reserved (0)**
7	RW	Rst211	**Reserved**
6	RW	Rst211	**Enable extend PCIe RC memory aperture** 0: Disable mapping 1: Enable mapping This bit enables the mapping of large physical address extension address space range behind 0x1_0000_0000 to PCIe RC memory aperture.
5	RW	Rst211	**Enable PCIe RC memory aperture** 0: Disable mapping 1: Enable mapping This bit enables the mapping of physical address space range 0x6000_0000 ~ 0x7FFF_FFFF to PCIe RC memory aperture.
4:2	RW	Rst211	**Reserved (0)**
1	RW	Rst211	**Disable BSPI Remap** 0: Enable mapping 1: Disable mapping This bit disables the mapping of physical address space range 0x0000_0000 ~ 0x0FFF_FFFF to BSPI address space range 0x2000_0000 ~ 0x1FFF_FFFF.
0	RW	Rst211	**Boot Area Remap** 0: Disable mapping 1: Enable mapping This bit enables the mapping of physical address space range 0x0000_0000 ~ 0x0FFF_FFFF to Embedded SRAM address space range 0x1000_0000 ~ 0x1FFF_FFFF. This is used for Secure Boot.

Offset: 90h **AHBC90: Watchdog Counter Status Register** **Init = 0xC000**

Bit	R/W	Reset	Description
31:16	RO	-	**Reserved (0)**

to next page

from previous page

15:0	RO	-	**Counter status** This register stores the current status of counter. This register reloads from AHBC94 automatically when bus is idle. The counter unit is the number of HCLK. The counter starts to decrement when bus is busy. And when counts to 0, the bus target that holds the bus busy will be disabled.

Offset: 94h	**AHBC94: Watchdog Counter Reload Value Register**		**Init = 0xC000**
Bit	**R/W**	**Reset**	**Description**
31:16	RO	-	**Reserved (0)**
15:0	RW	Rst01	**Counter reload value register** Reload register contains value which will be loaded into AHBC90 register. When the reload value set to 0, it means to disable the watchdog timer.

Offset: 9Ch	**AHBC9C: SCU Password Snoop Register**		**Init = 0x0**
Bit	**R/W**	**Reset**	**Description**
31:1	RO	-	**Reserved (0)**
0	RW	Rst211	**SCU Password Snoop Register** 0: Enable snoop for SCU000 and SCU010 1: Disable snoop SCU000 and SCU010

Offset: C0h	**AHBCC0: Internal AHB Bus Flush Enable Register**		**Init = 0x0**
Bit	**R/W**	**Reset**	**Description**
31:0	RW	Rst212	**Internal AHB bus flush enable register** The registers control the internal bus read flush write data in pipelines. This register is reserved and please keep 0.

Offset: C4h	**AHBCC4: SWVIC CPU Lock Control Register**		**Init = 0x0**
Bit	**R/W**	**Reset**	**Description**
31:1	RO	-	**Reserved (0)**
0	RW	Rst212	**SWVIC CPU Lock Control Register** The SWVIC registers can be locked to prevent ARM CA7 to access CM3 controlled SWVIC registers (SWVIC18 and SWVIC1C) and also prevent CM3 to access CA7 controlled SWVIC registers (SWVIC28 and SWVIC2C).

Offset: 200h	**AHBC200: TrustZone AHBC Master Secure Disable Register**		**Init = 0x0**
Bit	**R/W**	**Reset**	**Description**
31:28	RW	Rst212	**Reserved (0)**
27	RW	Rst212	**Disable I3C DMA secure privilege**
26	RW	Rst212	**Disable secure boot MPU secure privilege**
25	RW	Rst212	**Reserved (0)**
24	RW	Rst212	**Disable Uart DMA secure privilege**
23	RW	Rst212	**Disable eSPI bus to AHB bus bridge secure privilege**
22	RW	Rst212	**Disable 80 port DMA (from LPC/eSPI) secure privilege**

to next page

from previous page

21	RW	Rst212	**Disable SPI DMA secure privilege**
20	RW	Rst212	**Disable boot SPI DMA secure privilege**
19	RW	Rst212	**Disable LPC bus to AHB bus bridge secure privilege**
18	RW	Rst212	**Disable SD controller secure privilege**
17:10	RW	Rst212	**Reserved (0)**
9	RW	Rst212	**Disable DP MPU secure privilege**
8	RW	Rst212	**Disable 80 port DMA (from PCIe) secure privilege**
7	RW	Rst212	**Reserved (0)**
6	RW	Rst212	**Disable CM3 secure privilege**
5	RW	Rst212	**Disable GP MPU secure privilege**
4	RW	Rst212	**Disable eMMC controller secure privilege**
3	RW	Rst212	**Disable Uart debug secure privilege**
2	RW	Rst212	**Disable PCIe to AHB bridge TrustZone secure privilege**
1:0	RW	Rst212	**Reserved (0)**

Offset: 204h **AHBC204: TrustZone Regiters Write Protection** **Init = 0x0**

Bit	R/W	Reset	Description
31	RW1S	Rst212	**Reserved (0)**
30	RW1S	Rst212	**Enable write protect for register AHBC3C4**
29	RW1S	Rst212	**Enable write protect for register AHBC208**
28	RW1S	Rst212	**Enable write protect for register AHBC200**
27	RW1S	Rst212	**Enable write protect for register AHBC3B0, AHBC3B4 and AHBC3B8**
26	RW1S	Rst212	**Enable write protect for register AHBC3A0, AHBC3A4 and AHBC3A8**
25	RW1S	Rst212	**Enable write protect for register AHBC390, AHBC394 and AHBC398**
24	RW1S	Rst212	**Enable write protect for register AHBC380, AHBC384 and AHBC388**
23	RW1S	Rst212	**Enable write protect for register AHBC370, AHBC374 and AHBC378**
22	RW1S	Rst212	**Enable write protect for register AHBC360, AHBC364 and AHBC368**
21	RW1S	Rst212	**Enable write protect for register AHBC350, AHBC354 and AHBC358**
20	RW1S	Rst212	**Enable write protect for register AHBC340, AHBC344 and AHBC348**
19	RW1S	Rst212	**Enable write protect for register AHBC330, AHBC334 and AHBC338**
18	RW1S	Rst212	**Enable write protect for register AHBC320, AHBC324 and AHBC328**
17	RW1S	Rst212	**Enable write protect for register AHBC310, AHBC314 and AHBC318**
16	RW1S	Rst212	**Enable write protect for register AHBC300, AHBC304 and AHBC308**
15	RW1S	Rst212	**Enable write protect for register AHBC29C**
14	RW1S	Rst212	**Enable write protect for register AHBC298**
13	RW1S	Rst212	**Enable write protect for register AHBC294**
12	RW1S	Rst212	**Enable write protect for register AHBC290**
11	RW1S	Rst212	**Enable write protect for register AHBC28C**
10	RW1S	Rst212	**Enable write protect for register AHBC288**
9	RW1S	Rst212	**Enable write protect for register AHBC284**
8	RW1S	Rst212	**Enable write protect for register AHBC280**
7	RW1S	Rst212	**Enable write protect for register AHBC25C**

to next page

from previous page

6	RW1S	Rst212	**Enable write protect for register AHBC258**
5	RW1S	Rst212	**Enable write protect for register AHBC254**
4	RW1S	Rst212	**Enable write protect for register AHBC250**
3	RW1S	Rst212	**Enable write protect for register AHBC24C**
2	RW1S	Rst212	**Enable write protect for register AHBC248**
1	RW1S	Rst212	**Enable write protect for register AHBC244**
0	RW1S	Rst212	**Enable write protect for register AHBC240**

Offset: 208h AHBC208: AHBC and TrustZone Regiters Reset Selection Register Init = 0x0

Bit	R/W	Reset	Description
31:2	RO	-	**Reserved (0)**
1	RW1S	Rst212	**TrustZone register reset selection** 0: AHBC200-AHBC3B8 registers reset by AHB bus reset 1: AHBC200-AHBC3B8 registers reset by ARM CA7 reset
0	RW1S	Rst212	**AHBC register reset selection** 0: AHBC00-AHBCC4 registers reset by AHB bus reset 1: AHBC00-AHBCC4 registers reset by ARM CA7 reset

Offset: 240h AHBC240: Region Access Protection Register 1 Init = 0x0

Bit	R/W	Reset	Description
31:27	RW	Rst212	**Reserved (0)**
26	RW	Rst212	**Enable access protection for range 0x1e7f0000 - 0x1e7f3fff (PCIe Security))**
25	RW	Rst212	**Enable access protection for range 0x1e7e0000 - 0x1e7effff (HOST2BMC))**
24	RW	Rst212	**Enable access protection for range 0x18800000 - 0x1883ffff (GP MPU Memory)**
23	RW	Rst212	**Enable access protection for range 0x1e710000 - 0x1e71ffff (ACRYM))**
22	RW	Rst212	**Enable access protection for range 0x60000000 - 0x7fffffff (PCIe RC Aperture**
21	RW	Rst212	**Enable access protection for range 0x1e770000 - 0x1e77ffff (AHB to PCIe RC Bridge))**
20	RW	Rst212	**Enable access protection for range 0x1e6d0000 - 0x1e6d00ff (HACE)**
19	RW	Rst212	**Enable access protection for range 0x1a000000 - 0x1a03ffff (CM3 Memory)**
18	RW	Rst212	**Enable access protection for range 0x1e6a2000 - 0x1e6a21ff (EHCI Controller)**
17	RW	Rst212	**Enable access protection for range 0x18000000 - 0x1803ffff (DP MPU Memory)**
16	RW	Rst212	**Enable access protection for range 0x80000000 - 0xffffffff (SDRAM Memory)**
15	RW	Rst212	**Enable access protection for range 0x1e620000 - 0x1e62ffff (Firmware SPI) and 0x20000000 - 0x2fffffff (Firmware SPI Memory)**
14	RW	Rst212	**Enable access protection for range 0x1e750000 - 0x1e75ffff (eMMC)**
13	RW	Rst212	**Enable access protection for range 0x1e760000 - 0x1e7601ff (2D Engine)**
12	RW	Rst212	**Enable access protection for range 0x1e700000 - 0x1e70ffff (Video Engine)**
11	RW	Rst212	**Enable access protection for range 0x1e6c0000 - 0x1e6c7000 (VIC part 1)**
10	RW	Rst212	**Enable access protection for range 0x1e6a0000 - 0x1e6a01ff (USB2.0 Virtual Hub)**

to next page

from previous page

9	RW	Rst212	**Enable access protection for range 0x1e6b0000 - 0x1e6bffff (USB1.1 UHCI Host Controller)**
8	RW	Rst212	**Enable access protection for range 0x1e660000 - 0x1e6603ff (MAC1) and 0x1e680000 - 0x1e6803ff (MAC2)**
7	RW	Rst212	**Enable access protection for range 0x1e7c8000 - 0x1e7cffff (RVAS)**
6	RW	Rst212	**Enable access protection for range 0x1e720000 - 0x1e73ffff (SRAM register)**
5	RW	Rst212	**Enable access protection for range 0x10016000 - 0x1001ffff (SRAM part 3)**
4	RW	Rst212	**Enable access protection for range 0x10010000 - 0x10015fff (SRAM part 2)**
3	RW	Rst212	**Enable access protection for range 0x10000000 - 0x1000ffff (SRAM part 1)**
2	RW	Rst212	**Enable access protection for range 0x1e7c0000 - 0x1e7c1fff (RVAS LMEM)**
1	RW	Rst212	**Enable access protection for range 0x1e600000 - 0x1e607fff (AHBC part 2)**
0	RW	Rst212	**Enable access protection for range 0x1e600000 - 0x1e6000ff (AHBC part 1)**

Offset: 244h **AHBC244: Region Access Protection Register 2** **Init = 0x0**

Bit	R/W	Reset	Description
31:29	RW	Rst212	**Reserved (0)**
28	RW	Rst212	**Enable access protection for range 0x1e6e8800 - 0x1e6e8fff (PUART over PCIe)**
27	RW	Rst212	**Enable access protection for range 0x1e6e7800 - 0x1e6e7fff (VUART over PCIe)**
26	RW	Rst212	**Enable access protection for range 0x1e6ec000 - 0x1e6ecfff (APB to PCIe Bridge/Device Bridge)**
25	RW	Rst212	**Enable access protection for range 0x1e785800 - 0x1e7859ff (WDT Mirror)**
24	RW	Rst212	**Enable access protection for range 0x1e79f000 - 0x1e79f7ff (UART5 debug)**
23	RW	Rst212	**Enable access protection for range 0x1e79e800 - 0x1e79efff (UART DMA part 2)**
22	RW	Rst212	**Enable access protection for range 0x1e784000 - 0x1e784fff (UART5)**
21	RW	Rst212	**Enable access protection for range 0x1e782000 - 0x1e782fff (Timer)**
20	RW	Rst212	**Enable access protection for range 0x1e781000 - 0x1e781fff (RTC)**
19	RW	Rst212	**Enable access protection for range 0x1e780800 - 0x1e780fff (1.8V GPIO)**
18	RW	Rst212	**Enable access protection for range 0x1e6f2800 - 0x1e6f2fff (Secure Boot MPU part 2)**
17	RW	Rst212	**Enable access protection for range 0x1e6f9000 - 0x1e6f9fff (PCIe RC MCTP)**
16	RW	Rst212	**Enable access protection for range 0x1e6f8000 - 0x1e6f8fff (PCIe RC XDMA)**
15	RW	Rst212	**Enable access protection for range 0x1e6e7000 - 0x1e6e7fff (APB to PCIe RC Bridge)**
14	RW	Rst212	**Enable access protection for range 0x1e6ed200 - 0x1e6ed3ff (PCIe RC Controller)**
13	RW	Rst212	**Enable access protection for range 0x1e6f5000 - 0x1e6f50ff (eMMC Boot Controller)**
12	RW	Rst212	**Enable access protection for range 0x1e6f3000 - 0x1e6f307f (Internal Bridge Controller part 1)**
11	RW	Rst212	**Enable access protection for range 0x1e6ed000 - 0x1e6ed1ff (PCIe Bridge/Device Controller)**
10	RW	Rst212	**Enable access protection for range 0x1e6e8000 - 0x1e6e8fff (MCTP)**

to next page

from previous page

9	RW	Rst212	**Enable access protection for range 0x1e6e7000 - 0x1e6e7fff (XDMA)**
8	RW	Rst212	**Enable access protection for range 0x1e6e6000 - 0x1e6e6fff (GFX)**
7	RW	Rst212	**Enable access protection for range 0x1e6ec000 - 0x1e6ecfff (VGA)**
6	RW	Rst212	**Enable access protection for range 0x1e6eb000 - 0x1e6eb1ff (DP)**
5	RW	Rst212	**Enable access protection for range 0x1e6e4000 - 0x1e6e40ff (JTAG Master 1)**
4	RW	Rst212	**Enable access protection for range 0x1e6fa000 - 0x1e6fa07f (ACRY)**
3	RW	Rst212	**Enable access protection for range 0x1e6e2000 - 0x1e6e2fff (SCU part 1)**
2	RW	Rst212	**Enable access protection for range 0x1e6e1000 - 0x1e6e107f (USB 1.1 HID)**
1	RW	Rst212	**Enable access protection for range 0x1e6e0000 - 0x1e6e07ff (MMC)**
0	RW	Rst212	**Enable access protection for range 0x14000000 - 0x14ffffff (ARM Coresight Interface)**

Offset: 248h **AHBC248: Region Access Protection Register 3** **Init = 0x0**

Bit	R/W	Reset	Description
31	RW	Rst212	**Enable access protection for range 0x1e790200 - 0x1e7902ff (UART8)**
30	RW	Rst212	**Enable access protection for range 0x1e790100 - 0x1e7901ff (UART7)**
29	RW	Rst212	**Enable access protection for range 0x1e790000 - 0x1e7900ff (UART6)**
28	RW	Rst212	**Enable access protection for range 0x1e78f000 - 0x1e78ffff (UART4)**
27	RW	Rst212	**Enable access protection for range 0x1e78e000 - 0x1e78efff (UART3)**
26	RW	Rst212	**Enable access protection for range 0x1e78d000 - 0x1e78dfff (UART2)**
25	RW	Rst212	**Enable access protection for range 0x1e783000 - 0x1e783fff (UART1)**
24	RW	Rst212	**Enable access protection for range 0x1e780000 - 0x1e7807ff (GPIO)**
23	RW	Rst212	**Enable access protection for range 0x1e6e4100 - 0x1e6e41ff (JTAG Master 2)**
22	RW	Rst212	**Enable access protection for range 0x1e6f3800 - 0x1e6f387f (Internal Bridge Controller part 2)**
21	RW	Rst212	**Enable access protection for range 0x1e6fb000 - 0x1e6fbfff (ESPI Master)**
20	RW	Rst212	**Enable access protection for range 0x1e6f2000 - 0x1e6f27ff (Secure Boot MPU part 1)**
19	RW	Rst212	**Enable access protection for range 0x1e6ef000 - 0x1e6effff (BSRAM)**
18	RW	Rst212	**Enable access protection for range 0x1e6ee000 - 0x1e6eefff (ESPI)**
17	RW	Rst212	**Reserved (0)**
16	RW	Rst212	**Enable access protection for range 0x1e6e9000 - 0x1e6e9fff (ADC)**
15	RW	Rst212	**Enable access protection for range 0x1e6e2000 - 0x1e6e2fff (SCU part 2)**
14	RW	Rst212	**Enable access protection for range 0x1e6c8000 - 0x1e6cffff (VIC part 2)**
13	RW	Rst212	**Enable access protection for range 0x19000000 - 0x1903ffff (AST MCU Memory)**
12	RW	Rst212	**Enable access protection for range 0x1e7d0000 - 0x1e7dffff (FSI AHB)**
11	RW	Rst212	**Enable access protection for range 0x1e610000 - 0x1e6101ff (PWM/TACHO)**
10	RW	Rst212	**Enable access protection for range 0x1e651000 - 0x1e651fff (I3C HDMA)**
9	RW	Rst212	**Enable access protection for range 0x1e740000 - 0x1e74ffff (SD/SDIO)**
8	RW	Rst212	**Enable access protection for range 0x50000000 - 0x5fffffff (SPI2 Memory)**
7	RW	Rst212	**Enable access protection for range 0x30000000 - 0x3fffffff (SPI1 Memory)**

to next page

from previous page

6	RW	Rst212	**Reserved (0)**
5	RW	Rst212	**Enable access protection for range 0x1e630000 - 0x1e63ffff (SPI controllers)**
4	RW	Rst212	**Enable access protection for range 0x1e690000 - 0x1e6903ff (MAC4)**
3	RW	Rst212	**Enable access protection for range 0x1e670000 - 0x1e6503ff (MAC3)**
2	RW	Rst212	**Enable access protection for range 0x1e650000 - 0x1e65007f (MII)**
1	RW	Rst212	**Enable access protection for range 0x16000000 - 0x17ffffff (AHB Bus to LPC Bus Bridge)**
0	RW	Rst212	**Reserved (0)**

Offset: 24Ch **AHBC24C: Region Access Protection Register 4** **Init = 0x0**

Bit	R/W	Reset	Description
31:23	RW	Rst212	**Reserved (0)**
22	RW	Rst212	**Enable access protection for range 0x1e7a8000 - 0x1e7a8fff (I2CS)**
21	RW	Rst212	**Enable access protection for range 0x1e7a7000 - 0x1e7a7fff (I3C 6 register)**
20	RW	Rst212	**Enable access protection for range 0x1e7a6000 - 0x1e7a6fff (I3C 5 register)**
19	RW	Rst212	**Enable access protection for range 0x1e7a5000 - 0x1e7a5fff (I3C 4 register)**
18	RW	Rst212	**Enable access protection for range 0x1e7a4000 - 0x1e7a4fff (I3C 3 register)**
17	RW	Rst212	**Enable access protection for range 0x1e7a3000 - 0x1e7a3fff (I3C 2 register)**
16	RW	Rst212	**Enable access protection for range 0x1e7a2000 - 0x1e7a2fff (I3C 1 register)**
15	RW	Rst212	**Reserved (0)**
14	RW	Rst212	**Enable access protection for range 0x1e7a0000 - 0x1e7a0fff (I3C global register)**
13	RW	Rst212	**Enable access protection for range 0x1e79b000 - 0x1e79bfff (FSI)**
12	RW	Rst212	**Enable access protection for range 0x1e78b000 - 0x1e78bfff (PECI)**
11	RW	Rst212	**Enable access protection for range 0x1e78a000 - 0x1e78afff (I2C)**
10	RW	Rst212	**Enable access protection for range 0x1e789000 - 0x1e7897ff (LPC)**
9	RW	Rst212	**Enable access protection for range 0x1e788000 - 0x1e7887ff (PUART)**
8	RW	Rst212	**Enable access protection for range 0x1e787000 - 0x1e7877ff (VUART)**
7	RW	Rst212	**Enable access protection for range 0x1e785000 - 0x1e7851ff (WDT)**
6	RW	Rst212	**Enable access protection for range 0x1e79f800 - 0x1e79ffff (UART1 debug)**
5	RW	Rst212	**Enable access protection for range 0x1e79e000 - 0x1e79e7ff (UART DMA part 1)**
4	RW	Rst212	**Enable access protection for range 0x1e790700 - 0x1e7907ff (UART13)**
3	RW	Rst212	**Enable access protection for range 0x1e790600 - 0x1e7906ff (UART12)**
2	RW	Rst212	**Enable access protection for range 0x1e790500 - 0x1e7905ff (UART11)**
1	RW	Rst212	**Enable access protection for range 0x1e790400 - 0x1e7904ff (UART10)**
0	RW	Rst212	**Enable access protection for range 0x1e790300 - 0x1e7903ff (UART9)**

Offset: 250h **AHBC250: Region Write Protection Register 1** **Init = 0x0**

Bit	R/W	Reset	Description
31:27	RW	Rst212	**Reserved (0)**
26	RW	Rst212	**Enable write protection for range 0x1e7f0000 - 0x1e7f3fff (PCIe Security))**
25	RW	Rst212	**Enable write protection for range 0x1e7e0000 - 0x1e7effff (HOST2BMC))**

to next page

from previous page

24	RW	Rst212	**Enable write protection for range 0x18800000 - 0x1883ffff (GP MPU Memory)**
23	RW	Rst212	**Enable write protection for range 0x1e710000 - 0x1e71ffff (ACRYM))**
22	RW	Rst212	**Enable write protection for range 0x60000000 - 0x7fffffff (PCIe RC Aperture**
21	RW	Rst212	**Enable write protection for range 0x1e770000 - 0x1e77ffff (AHB to PCIe RC Bridge))**
20	RW	Rst212	**Enable write protection for range 0x1e6d0000 - 0x1e6d00ff (HACE)**
19	RW	Rst212	**Enable write protection for range 0x1a000000 - 0x1a03ffff (CM3 Memory)**
18	RW	Rst212	**Enable write protection for range 0x1e6a2000 - 0x1e6a21ff (EHCI Controller)**
17	RW	Rst212	**Enable write protection for range 0x18000000 - 0x1803ffff (DP MPU Memory)**
16	RW	Rst212	**Enable write protection for range 0x80000000 - 0xffffffff (SDRAM Memory)**
15	RW	Rst212	**Enable write protection for range 0x1e620000 - 0x1e62ffff (Firmware SPI) and 0x20000000 - 0x2fffffff (Firmware SPI Memory)**
14	RW	Rst212	**Enable write protection for range 0x1e750000 - 0x1e75ffff (eMMC)**
13	RW	Rst212	**Enable write protection for range 0x1e760000 - 0x1e7601ff (2D Engine)**
12	RW	Rst212	**Enable write protection for range 0x1e700000 - 0x1e70ffff (Video Engine)**
11	RW	Rst212	**Enable write protection for range 0x1e6c0000 - 0x1e6c7000 (VIC part 1)**
10	RW	Rst212	**Enable write protection for range 0x1e6a0000 - 0x1e6a01ff (USB2.0 Virtual Hub)**
9	RW	Rst212	**Enable write protection for range 0x1e6b0000 - 0x1e6bffff (USB1.1 UHCI Host Controller)**
8	RW	Rst212	**Enable write protection for range 0x1e660000 - 0x1e6603ff (MAC1) and 0x1e680000 - 0x1e6803ff (MAC2)**
7	RW	Rst212	**Enable write protection for range 0x1e7c8000 - 0x1e7cffff (RVAS)**
6	RW	Rst212	**Enable write protection for range 0x1e720000 - 0x1e73ffff (SRAM register)**
5	RW	Rst212	**Enable write protection for range 0x10016000 - 0x1001ffff (SRAM part 3)**
4	RW	Rst212	**Enable write protection for range 0x10010000 - 0x10015fff (SRAM part 2)**
3	RW	Rst212	**Enable write protection for range 0x10000000 - 0x1000ffff (SRAM part 1)**
2	RW	Rst212	**Enable write protection for range 0x1e7c0000 - 0x1e7c1fff (RVAS LMEM)**
1	RW	Rst212	**Enable write protection for range 0x1e600000 - 0x1e607fff (AHBC part 2)**
0	RW	Rst212	**Enable write protection for range 0x1e600000 - 0x1e6000ff (AHBC part 1)**

Offset: 254h **AHBC254: Region Write Protection Register 2** **Init = 0x0**

Bit	R/W	Reset	Description
31:29	RW	Rst212	**Reserved (0)**
28	RW	Rst212	**Enable write protection for range 0x1e6e8800 - 0x1e6e8fff (PUART over PCIe)**
27	RW	Rst212	**Enable write protection for range 0x1e6e7800 - 0x1e6e7fff (VUART over PCIe)**
26	RW	Rst212	**Enable write protection for range 0x1e6ec000 - 0x1e6ecfff (APB to PCIe Bridge/Device Bridge)**
25	RW	Rst212	**Enable write protection for range 0x1e785800 - 0x1e7859ff (WDT Mirror)**
24	RW	Rst212	**Enable write protection for range 0x1e79f000 - 0x1e79f7ff (UART5 debug)**
23	RW	Rst212	**Enable write protection for range 0x1e79e800 - 0x1e79efff (UART DMA part 2)**
22	RW	Rst212	**Enable write protection for range 0x1e784000 - 0x1e784fff (UART5)**
21	RW	Rst212	**Enable write protection for range 0x1e782000 - 0x1e782fff (Timer)**

to next page

from previous page

20	RW	Rst212	**Enable write protection for range 0x1e781000 - 0x1e781fff (RTC)**
19	RW	Rst212	**Enable write protection for range 0x1e780800 - 0x1e780fff (1.8V GPIO)**
18	RW	Rst212	**Enable write protection for range 0x1e6f2800 - 0x1e6f2fff (Secure Boot MPU part 2)**
17	RW	Rst212	**Enable write protection for range 0x1e6f9000 - 0x1e6f9fff (PCIe RC MCTP)**
16	RW	Rst212	**Enable write protection for range 0x1e6f8000 - 0x1e6f8fff (PCIe RC XDMA)**
15	RW	Rst212	**Enable write protection for range 0x1e6e7000 - 0x1e6e7fff (APB to PCIe RC Bridge)**
14	RW	Rst212	**Enable write protection for range 0x1e6ed200 - 0x1e6ed3ff (PCIe RC Controller)**
13	RW	Rst212	**Enable write protection for range 0x1e6f5000 - 0x1e6f50ff (eMMC Boot Controller)**
12	RW	Rst212	**Enable write protection for range 0x1e6f3000 - 0x1e6f307f (Internal Bridge Controller part 1)**
11	RW	Rst212	**Enable write protection for range 0x1e6ed000 - 0x1e6ed1ff (PCIe Bridge/Device Controller)**
10	RW	Rst212	**Enable write protection for range 0x1e6e8000 - 0x1e6e8fff (MCTP)**
9	RW	Rst212	**Enable write protection for range 0x1e6e7000 - 0x1e6e7fff (XDMA)**
8	RW	Rst212	**Enable write protection for range 0x1e6e6000 - 0x1e6e6fff (GFX)**
7	RW	Rst212	**Enable write protection for range 0x1e6ec000 - 0x1e6ecfff (VGA)**
6	RW	Rst212	**Enable write protection for range 0x1e6eb000 - 0x1e6eb1ff (DP)**
5	RW	Rst212	**Enable write protection for range 0x1e6e4000 - 0x1e6e40ff (JTAG Master 1)**
4	RW	Rst212	**Enable write protection for range 0x1e6fa000 - 0x1e6fa07f (ACRY)**
3	RW	Rst212	**Enable write protection for range 0x1e6e2000 - 0x1e6e2fff (SCU part 1)**
2	RW	Rst212	**Enable write protection for range 0x1e6e1000 - 0x1e6e107f (USB 1.1 HID)**
1	RW	Rst212	**Enable write protection for range 0x1e6e0000 - 0x1e6e07ff (MMC)**
0	RW	Rst212	**Enable write protection for range 0x14000000 - 0x14ffffff (ARM Coresight Interface)**

Offset: 258h **AHBC258: Region Write Protection Register 3** **Init = 0x0**

Bit	R/W	Reset	Description
31	RW	Rst212	**Enable write protection for range 0x1e790200 - 0x1e7902ff (UART8)**
30	RW	Rst212	**Enable write protection for range 0x1e790100 - 0x1e7901ff (UART7)**
29	RW	Rst212	**Enable write protection for range 0x1e790000 - 0x1e7900ff (UART6)**
28	RW	Rst212	**Enable write protection for range 0x1e78f000 - 0x1e78ffff (UART4)**
27	RW	Rst212	**Enable write protection for range 0x1e78e000 - 0x1e78efff (UART3)**
26	RW	Rst212	**Enable write protection for range 0x1e78d000 - 0x1e78dfff (UART2)**
25	RW	Rst212	**Enable write protection for range 0x1e783000 - 0x1e783fff (UART1)**
24	RW	Rst212	**Enable write protection for range 0x1e780000 - 0x1e7807ff (GPIO)**
23	RW	Rst212	**Enable write protection for range 0x1e6e4100 - 0x1e6e41ff (JTAG Master 2)**
22	RW	Rst212	**Enable write protection for range 0x1e6f3800 - 0x1e6f387f (Internal Bridge Controller part 2)**
21	RW	Rst212	**Enable write protection for range 0x1e6fb000 - 0x1e6fbfff (ESPI Master)**
20	RW	Rst212	**Enable write protection for range 0x1e6f2000 - 0x1e6f27ff (Secure Boot MPU part 1)**

to next page

from previous page

19	RW	Rst212	**Enable write protection for range 0x1e6ef000 - 0x1e6effff (BSRAM)**
18	RW	Rst212	**Enable write protection for range 0x1e6ee000 - 0x1e6eefff (ESPI)**
17	RW	Rst212	**Reserved (0)**
16	RW	Rst212	**Enable write protection for range 0x1e6e9000 - 0x1e6e9fff (ADC)**
15	RW	Rst212	**Enable write protection for range 0x1e6e2000 - 0x1e6e2fff (SCU part 2)**
14	RW	Rst212	**Enable write protection for range 0x1e6c8000 - 0x1e6cffff (VIC part 2)**
13	RW	Rst212	**Enable write protection for range 0x19000000 - 0x1903ffff (AST MCU Memory)**
12	RW	Rst212	**Enable write protection for range 0x1e7d0000 - 0x1e7dffff (FSI AHB)**
11	RW	Rst212	**Enable write protection for range 0x1e610000 - 0x1e6101ff (PWM/TACHO)**
10	RW	Rst212	**Enable write protection for range 0x1e651000 - 0x1e651fff (I3C HDMA)**
9	RW	Rst212	**Enable write protection for range 0x1e740000 - 0x1e74ffff (SD/SDIO)**
8	RW	Rst212	**Enable write protection for range 0x50000000 - 0x5fffffff (SPI2 Memory)**
7	RW	Rst212	**Enable write protection for range 0x30000000 - 0x3fffffff (SPI1 Memory)**
6	RW	Rst212	**Reserved (0)**
5	RW	Rst212	**Enable write protection for range 0x1e630000 - 0x1e63ffff (SPI controllers)**
4	RW	Rst212	**Enable write protection for range 0x1e690000 - 0x1e6903ff (MAC4)**
3	RW	Rst212	**Enable write protection for range 0x1e670000 - 0x1e6503ff (MAC3)**
2	RW	Rst212	**Enable write protection for range 0x1e650000 - 0x1e65007f (MII)**
1	RW	Rst212	**Enable write protection for range 0x16000000 - 0x17ffffff (AHB Bus to LPC Bus Bridge)**
0	RW	Rst212	**Reserved (0)**

Offset: 25Ch **AHBC25C: Region Write Protection Register 4** **Init = 0x0**

Bit	R/W	Reset	Description
31:23	RW	Rst212	**Reserved (0)**
22	RW	Rst212	**Enable write protection for range 0x1e7a8000 - 0x1e7a8fff (I2CS)**
21	RW	Rst212	**Enable write protection for range 0x1e7a7000 - 0x1e7a7fff (I3C 6 register)**
20	RW	Rst212	**Enable write protection for range 0x1e7a6000 - 0x1e7a6fff (I3C 5 register)**
19	RW	Rst212	**Enable write protection for range 0x1e7a5000 - 0x1e7a5fff (I3C 4 register)**
18	RW	Rst212	**Enable write protection for range 0x1e7a4000 - 0x1e7a4fff (I3C 3 register)**
17	RW	Rst212	**Enable write protection for range 0x1e7a3000 - 0x1e7a3fff (I3C 2 register)**
16	RW	Rst212	**Enable write protection for range 0x1e7a2000 - 0x1e7a2fff (I3C 1 register)**
15	RW	Rst212	**Reserved (0)**
14	RW	Rst212	**Enable write protection for range 0x1e7a0000 - 0x1e7a0fff (I3C global register)**
13	RW	Rst212	**Enable write protection for range 0x1e79b000 - 0x1e79bfff (FSI)**
12	RW	Rst212	**Enable write protection for range 0x1e78b000 - 0x1e78bfff (PECI)**
11	RW	Rst212	**Enable write protection for range 0x1e78a000 - 0x1e78afff (I2C)**
10	RW	Rst212	**Enable write protection for range 0x1e789000 - 0x1e7897ff (LPC)**
9	RW	Rst212	**Enable write protection for range 0x1e788000 - 0x1e7887ff (PUART)**
8	RW	Rst212	**Enable write protection for range 0x1e787000 - 0x1e7877ff (VUART)**
7	RW	Rst212	**Enable write protection for range 0x1e785000 - 0x1e7851ff (WDT)**

to next page

from previous page

6	RW	Rst212	**Enable write protection for range 0x1e79f800 - 0x1e79ffff (UART1 debug)**
5	RW	Rst212	**Enable write protection for range 0x1e79e000 - 0x1e79e7ff (UART DMA part 1)**
4	RW	Rst212	**Enable write protection for range 0x1e790700 - 0x1e7907ff (UART13)**
3	RW	Rst212	**Enable write protection for range 0x1e790600 - 0x1e7906ff (UART12)**
2	RW	Rst212	**Enable write protection for range 0x1e790500 - 0x1e7905ff (UART11)**
1	RW	Rst212	**Enable write protection for range 0x1e790400 - 0x1e7904ff (UART10)**
0	RW	Rst212	**Enable write protection for range 0x1e790300 - 0x1e7903ff (UART9)**

Offset: 280h **AHBC280: TrustZone Access Protection Register 1** **Init = 0x0**

Bit	R/W	Reset	Description
31:27	RW	Rst212	**Reserved (0)**
26	RW	Rst212	**Enable access protection for range 0x1e7f0000 - 0x1e7f3fff (PCIe Security))**
25	RW	Rst212	**Enable access protection for range 0x1e7e0000 - 0x1e7effff (HOST2BMC))**
24	RW	Rst212	**Enable access protection for range 0x18800000 - 0x1883ffff (GP MPU Memory)**
23	RW	Rst212	**Enable access protection for range 0x1e710000 - 0x1e71ffff (ACRYM))**
22	RW	Rst212	**Enable access protection for range 0x60000000 - 0x7fffffff (PCIe RC Aperture**
21	RW	Rst212	**Enable access protection for range 0x1e770000 - 0x1e77ffff (AHB to PCIe RC Bridge))**
20	RW	Rst212	**Enable access protection for range 0x1e6d0000 - 0x1e6d00ff (HACE)**
19	RW	Rst212	**Enable access protection for range 0x1a000000 - 0x1a03ffff (CM3 Memory)**
18	RW	Rst212	**Enable access protection for range 0x1e6a2000 - 0x1e6a21ff (EHCI Controller)**
17	RW	Rst212	**Enable access protection for range 0x18000000 - 0x1803ffff (DP MPU Memory)**
16	RW	Rst212	**Enable access protection for range 0x80000000 - 0xffffffff (SDRAM Memory)**
15	RW	Rst212	**Enable access protection for range 0x1e620000 - 0x1e62ffff (Firmware SPI) and 0x20000000 - 0x2fffffff (Firmware SPI Memory)**
14	RW	Rst212	**Enable access protection for range 0x1e750000 - 0x1e75ffff (eMMC)**
13	RW	Rst212	**Enable access protection for range 0x1e760000 - 0x1e7601ff (2D Engine)**
12	RW	Rst212	**Enable access protection for range 0x1e700000 - 0x1e70ffff (Video Engine)**
11	RW	Rst212	**Enable access protection for range 0x1e6c0000 - 0x1e6c7000 (VIC part 1)**
10	RW	Rst212	**Enable access protection for range 0x1e6a0000 - 0x1e6a01ff (USB2.0 Virtual Hub)**
9	RW	Rst212	**Enable access protection for range 0x1e6b0000 - 0x1e6bffff (USB1.1 UHCI Host Controller)**
8	RW	Rst212	**Enable access protection for range 0x1e660000 - 0x1e6603ff (MAC1) and 0x1e680000 - 0x1e6803ff (MAC2)**
7	RW	Rst212	**Enable access protection for range 0x1e7c8000 - 0x1e7cffff (RVAS)**
6	RW	Rst212	**Enable access protection for range 0x1e720000 - 0x1e73ffff (SRAM register)**
5	RW	Rst212	**Enable access protection for range 0x10016000 - 0x1001ffff (SRAM part 3)**
4	RW	Rst212	**Enable access protection for range 0x10010000 - 0x10015fff (SRAM part 2)**
3	RW	Rst212	**Enable access protection for range 0x10000000 - 0x1000ffff (SRAM part 1)**

to next page

from previous page

2	RW	Rst212	**Enable access protection for range 0x1e7c0000 - 0x1e7c1fff (RVAS LMEM)**
1	RW	Rst212	**Enable access protection for range 0x1e600000 - 0x1e607fff (AHBC part 2)**
0	RW	Rst212	**Enable access protection for range 0x1e600000 - 0x1e6000ff (AHBC part 1)**

Offset: 284h **AHBC284: TrustZone Access Protection Register 2** **Init = 0x0**

Bit	R/W	Reset	Description
31:29	RW	Rst212	**Reserved (0)**
28	RW	Rst212	**Enable access protection for range 0x1e6e8800 - 0x1e6e8fff (PUART over PCIe)**
27	RW	Rst212	**Enable access protection for range 0x1e6e7800 - 0x1e6e7fff (VUART over PCIe)**
26	RW	Rst212	**Enable access protection for range 0x1e6ec000 - 0x1e6ecfff (APB to PCIe Bridge/Device Bridge)**
25	RW	Rst212	**Enable access protection for range 0x1e785800 - 0x1e7859ff (WDT Mirror)**
24	RW	Rst212	**Enable access protection for range 0x1e79f000 - 0x1e79f7ff (UART5 debug)**
23	RW	Rst212	**Enable access protection for range 0x1e79e800 - 0x1e79efff (UART DMA part 2)**
22	RW	Rst212	**Enable access protection for range 0x1e784000 - 0x1e784fff (UART5)**
21	RW	Rst212	**Enable access protection for range 0x1e782000 - 0x1e782fff (Timer)**
20	RW	Rst212	**Enable access protection for range 0x1e781000 - 0x1e781fff (RTC)**
19	RW	Rst212	**Enable access protection for range 0x1e780800 - 0x1e780fff (1.8V GPIO)**
18	RW	Rst212	**Enable access protection for range 0x1e6f2800 - 0x1e6f2fff (Secure Boot MPU part 2)**
17	RW	Rst212	**Enable access protection for range 0x1e6f9000 - 0x1e6f9fff (PCIe RC MCTP)**
16	RW	Rst212	**Enable access protection for range 0x1e6f8000 - 0x1e6f8fff (PCIe RC XDMA)**
15	RW	Rst212	**Enable access protection for range 0x1e6e7000 - 0x1e6e7fff (APB to PCIe RC Bridge)**
14	RW	Rst212	**Enable access protection for range 0x1e6ed200 - 0x1e6ed3ff (PCIe RC Controller)**
13	RW	Rst212	**Enable access protection for range 0x1e6f5000 - 0x1e6f50ff (eMMC Boot Controller)**
12	RW	Rst212	**Enable access protection for range 0x1e6f3000 - 0x1e6f307f (Internal Bridge Controller part 1)**
11	RW	Rst212	**Enable access protection for range 0x1e6ed000 - 0x1e6ed1ff (PCIe Bridge/Device Controller)**
10	RW	Rst212	**Enable access protection for range 0x1e6e8000 - 0x1e6e8fff (MCTP)**
9	RW	Rst212	**Enable access protection for range 0x1e6e7000 - 0x1e6e7fff (XDMA)**
8	RW	Rst212	**Enable access protection for range 0x1e6e6000 - 0x1e6e6fff (GFX)**
7	RW	Rst212	**Enable access protection for range 0x1e6ec000 - 0x1e6ecfff (VGA)**
6	RW	Rst212	**Enable access protection for range 0x1e6eb000 - 0x1e6eb1ff (DP)**
5	RW	Rst212	**Enable access protection for range 0x1e6e4000 - 0x1e6e40ff (JTAG Master 1)**
4	RW	Rst212	**Enable access protection for range 0x1e6fa000 - 0x1e6fa07f (ACRY)**
3	RW	Rst212	**Enable access protection for range 0x1e6e2000 - 0x1e6e2fff (SCU part 1)**
2	RW	Rst212	**Enable access protection for range 0x1e6e1000 - 0x1e6e107f (USB 1.1 HID)**

to next page

from previous page

1	RW	Rst212	**Enable access protection for range 0x1e6e0000 - 0x1e6e07ff (MMC)**
0	RW	Rst212	**Enable access protection for range 0x14000000 - 0x14ffffff (ARM Coresight Interface)**

Offset: 288h **AHBC288: TrustZone Access Protection Register 3** **Init = 0x0**

Bit	R/W	Reset	Description
31	RW	Rst212	**Enable access protection for range 0x1e790200 - 0x1e7902ff (UART8)**
30	RW	Rst212	**Enable access protection for range 0x1e790100 - 0x1e7901ff (UART7)**
29	RW	Rst212	**Enable access protection for range 0x1e790000 - 0x1e7900ff (UART6)**
28	RW	Rst212	**Enable access protection for range 0x1e78f000 - 0x1e78ffff (UART4)**
27	RW	Rst212	**Enable access protection for range 0x1e78e000 - 0x1e78efff (UART3)**
26	RW	Rst212	**Enable access protection for range 0x1e78d000 - 0x1e78dfff (UART2)**
25	RW	Rst212	**Enable access protection for range 0x1e783000 - 0x1e783fff (UART1)**
24	RW	Rst212	**Enable access protection for range 0x1e780000 - 0x1e7807ff (GPIO)**
23	RW	Rst212	**Enable access protection for range 0x1e6e4100 - 0x1e6e41ff (JTAG Master 2)**
22	RW	Rst212	**Enable access protection for range 0x1e6f3800 - 0x1e6f387f (Internal Bridge Controller part 2)**
21	RW	Rst212	**Enable access protection for range 0x1e6fb000 - 0x1e6fbfff (ESPI Master)**
20	RW	Rst212	**Enable access protection for range 0x1e6f2000 - 0x1e6f27ff (Secure Boot MPU part 1)**
19	RW	Rst212	**Enable access protection for range 0x1e6ef000 - 0x1e6effff (BSRAM)**
18	RW	Rst212	**Enable access protection for range 0x1e6ee000 - 0x1e6eefff (ESPI)**
17	RW	Rst212	**Reserved (0)**
16	RW	Rst212	**Enable access protection for range 0x1e6e9000 - 0x1e6e9fff (ADC)**
15	RW	Rst212	**Enable access protection for range 0x1e6e2000 - 0x1e6e2fff (SCU part 2)**
14	RW	Rst212	**Enable access protection for range 0x1e6c8000 - 0x1e6cffff (VIC part 2)**
13	RW	Rst212	**Enable access protection for range 0x19000000 - 0x1903ffff (AST MCU Memory)**
12	RW	Rst212	**Enable access protection for range 0x1e7d0000 - 0x1e7dffff (FSI AHB)**
11	RW	Rst212	**Enable access protection for range 0x1e610000 - 0x1e6101ff (PWM/TACHO)**
10	RW	Rst212	**Enable access protection for range 0x1e651000 - 0x1e651fff (I3C HDMA)**
9	RW	Rst212	**Enable access protection for range 0x1e740000 - 0x1e74ffff (SD/SDIO)**
8	RW	Rst212	**Enable access protection for range 0x50000000 - 0x5fffffff (SPI2 Memory)**
7	RW	Rst212	**Enable access protection for range 0x30000000 - 0x3fffffff (SPI1 Memory)**
6	RW	Rst212	**Reserved (0)**
5	RW	Rst212	**Enable access protection for range 0x1e630000 - 0x1e63ffff (SPI controllers)**
4	RW	Rst212	**Enable access protection for range 0x1e690000 - 0x1e6903ff (MAC4)**
3	RW	Rst212	**Enable access protection for range 0x1e670000 - 0x1e6503ff (MAC3)**
2	RW	Rst212	**Enable access protection for range 0x1e650000 - 0x1e65007f (MII)**
1	RW	Rst212	**Enable access protection for range 0x16000000 - 0x17ffffff (AHB Bus to LPC Bus Bridge)**
0	RW	Rst212	**Reserved (0)**

Offset: 28Ch	AHBC28C: TrustZone Access Protection Register 4		Init = 0x0
Bit	**R/W**	**Reset**	**Description**
31:23	RW	Rst212	**Reserved (0)**
22	RW	Rst212	**Enable access protection for range 0x1e7a8000 - 0x1e7a8fff (I2CS)**
21	RW	Rst212	**Enable access protection for range 0x1e7a7000 - 0x1e7a7fff (I3C 6 register)**
20	RW	Rst212	**Enable access protection for range 0x1e7a6000 - 0x1e7a6fff (I3C 5 register)**
19	RW	Rst212	**Enable access protection for range 0x1e7a5000 - 0x1e7a5fff (I3C 4 register)**
18	RW	Rst212	**Enable access protection for range 0x1e7a4000 - 0x1e7a4fff (I3C 3 register)**
17	RW	Rst212	**Enable access protection for range 0x1e7a3000 - 0x1e7a3fff (I3C 2 register)**
16	RW	Rst212	**Enable access protection for range 0x1e7a2000 - 0x1e7a2fff (I3C 1 register)**
15	RW	Rst212	**Reserved (0)**
14	RW	Rst212	**Enable access protection for range 0x1e7a0000 - 0x1e7a0fff (I3C global register)**
13	RW	Rst212	**Enable access protection for range 0x1e79b000 - 0x1e79bfff (FSI)**
12	RW	Rst212	**Enable access protection for range 0x1e78b000 - 0x1e78bfff (PECI)**
11	RW	Rst212	**Enable access protection for range 0x1e78a000 - 0x1e78afff (I2C)**
10	RW	Rst212	**Enable access protection for range 0x1e789000 - 0x1e7897ff (LPC)**
9	RW	Rst212	**Enable access protection for range 0x1e788000 - 0x1e7887ff (PUART)**
8	RW	Rst212	**Enable access protection for range 0x1e787000 - 0x1e7877ff (VUART)**
7	RW	Rst212	**Enable access protection for range 0x1e785000 - 0x1e7851ff (WDT)**
6	RW	Rst212	**Enable access protection for range 0x1e79f800 - 0x1e79ffff (UART1 debug)**
5	RW	Rst212	**Enable access protection for range 0x1e79e000 - 0x1e79e7ff (UART DMA part 1)**
4	RW	Rst212	**Enable access protection for range 0x1e790700 - 0x1e7907ff (UART13)**
3	RW	Rst212	**Enable access protection for range 0x1e790600 - 0x1e7906ff (UART12)**
2	RW	Rst212	**Enable access protection for range 0x1e790500 - 0x1e7905ff (UART11)**
1	RW	Rst212	**Enable access protection for range 0x1e790400 - 0x1e7904ff (UART10)**
0	RW	Rst212	**Enable access protection for range 0x1e790300 - 0x1e7903ff (UART9)**

Offset: 290h	AHBC290: TrustZone Write Protection Register 1		Init = 0x0
Bit	**R/W**	**Reset**	**Description**
31:27	RW	Rst212	**Reserved (0)**
26	RW	Rst212	**Enable write protection for range 0x1e7f0000 - 0x1e7f3fff (PCIe Security))**
25	RW	Rst212	**Enable write protection for range 0x1e7e0000 - 0x1e7effff (HOST2BMC))**
24	RW	Rst212	**Enable write protection for range 0x18800000 - 0x1883ffff (GP MPU Memory)**
23	RW	Rst212	**Enable write protection for range 0x1e710000 - 0x1e71ffff (ACRYM))**
22	RW	Rst212	**Enable write protection for range 0x60000000 - 0x7fffffff (PCIe RC Aperture**
21	RW	Rst212	**Enable write protection for range 0x1e770000 - 0x1e77ffff (AHB to PCIe RC Bridge))**
20	RW	Rst212	**Enable write protection for range 0x1e6d0000 - 0x1e6d00ff (HACE)**
19	RW	Rst212	**Enable write protection for range 0x1a000000 - 0x1a03ffff (CM3 Memory)**
18	RW	Rst212	**Enable write protection for range 0x1e6a2000 - 0x1e6a21ff (EHCI Controller)**
17	RW	Rst212	**Enable write protection for range 0x18000000 - 0x1803ffff (DP MPU Memory)**
16	RW	Rst212	**Enable write protection for range 0x80000000 - 0xffffffff (SDRAM Memory)**

to next page

from previous page

15	RW	Rst212	**Enable write protection for range 0x1e620000 - 0x1e62ffff (Firmware SPI) and 0x20000000 - 0x2fffffff (Firmware SPI Memory)**
14	RW	Rst212	**Enable write protection for range 0x1e750000 - 0x1e75ffff (eMMC)**
13	RW	Rst212	**Enable write protection for range 0x1e760000 - 0x1e7601ff (2D Engine)**
12	RW	Rst212	**Enable write protection for range 0x1e700000 - 0x1e70ffff (Video Engine)**
11	RW	Rst212	**Enable write protection for range 0x1e6c0000 - 0x1e6c7000 (VIC part 1)**
10	RW	Rst212	**Enable write protection for range 0x1e6a0000 - 0x1e6a01ff (USB2.0 Virtual Hub)**
9	RW	Rst212	**Enable write protection for range 0x1e6b0000 - 0x1e6bffff (USB1.1 UHCI Host Controller)**
8	RW	Rst212	**Enable write protection for range 0x1e660000 - 0x1e6603ff (MAC1) and 0x1e680000 - 0x1e6803ff (MAC2)**
7	RW	Rst212	**Enable write protection for range 0x1e7c8000 - 0x1e7cffff (RVAS)**
6	RW	Rst212	**Enable write protection for range 0x1e720000 - 0x1e73ffff (SRAM register)**
5	RW	Rst212	**Enable write protection for range 0x10016000 - 0x1001ffff (SRAM part 3)**
4	RW	Rst212	**Enable write protection for range 0x10010000 - 0x10015fff (SRAM part 2)**
3	RW	Rst212	**Enable write protection for range 0x10000000 - 0x1000ffff (SRAM part 1)**
2	RW	Rst212	**Enable write protection for range 0x1e7c0000 - 0x1e7c1fff (RVAS LMEM)**
1	RW	Rst212	**Enable write protection for range 0x1e600000 - 0x1e607fff (AHBC part 2)**
0	RW	Rst212	**Enable write protection for range 0x1e600000 - 0x1e6000ff (AHBC part 1)**

Offset: 294h **AHBC294: TrustZone Write Protection Register 2** **Init = 0x0**

Bit	R/W	Reset	Description
31:29	RW	Rst212	**Reserved (0)**
28	RW	Rst212	**Enable write protection for range 0x1e6e8800 - 0x1e6e8fff (PUART over PCIe)**
27	RW	Rst212	**Enable write protection for range 0x1e6e7800 - 0x1e6e7fff (VUART over PCIe)**
26	RW	Rst212	**Enable write protection for range 0x1e6ec000 - 0x1e6ecfff (APB to PCIe Bridge/Device Bridge)**
25	RW	Rst212	**Enable write protection for range 0x1e785800 - 0x1e7859ff (WDT Mirror)**
24	RW	Rst212	**Enable write protection for range 0x1e79f000 - 0x1e79f7ff (UART5 debug)**
23	RW	Rst212	**Enable write protection for range 0x1e79e800 - 0x1e79efff (UART DMA part 2)**
22	RW	Rst212	**Enable write protection for range 0x1e784000 - 0x1e784fff (UART5)**
21	RW	Rst212	**Enable write protection for range 0x1e782000 - 0x1e782fff (Timer)**
20	RW	Rst212	**Enable write protection for range 0x1e781000 - 0x1e781fff (RTC)**
19	RW	Rst212	**Enable write protection for range 0x1e780800 - 0x1e780fff (1.8V GPIO)**
18	RW	Rst212	**Enable write protection for range 0x1e6f2800 - 0x1e6f2fff (Secure Boot MPU part 2)**
17	RW	Rst212	**Enable write protection for range 0x1e6f9000 - 0x1e6f9fff (PCIe RC MCTP)**
16	RW	Rst212	**Enable write protection for range 0x1e6f8000 - 0x1e6f8fff (PCIe RC XDMA)**
15	RW	Rst212	**Enable write protection for range 0x1e6e7000 - 0x1e6e7fff (APB to PCIe RC Bridge)**
14	RW	Rst212	**Enable write protection for range 0x1e6ed200 - 0x1e6ed3ff (PCIe RC Controller)**

to next page

from previous page

13	RW	Rst212	**Enable write protection for range 0x1e6f5000 - 0x1e6f50ff (eMMC Boot Controller)**
12	RW	Rst212	**Enable write protection for range 0x1e6f3000 - 0x1e6f307f (Internal Bridge Controller part 1)**
11	RW	Rst212	**Enable write protection for range 0x1e6ed000 - 0x1e6ed1ff (PCIe Bridge/Device Controller)**
10	RW	Rst212	**Enable write protection for range 0x1e6e8000 - 0x1e6e8fff (MCTP)**
9	RW	Rst212	**Enable write protection for range 0x1e6e7000 - 0x1e6e7fff (XDMA)**
8	RW	Rst212	**Enable write protection for range 0x1e6e6000 - 0x1e6e6fff (GFX)**
7	RW	Rst212	**Enable write protection for range 0x1e6ec000 - 0x1e6ecfff (VGA)**
6	RW	Rst212	**Enable write protection for range 0x1e6eb000 - 0x1e6eb1ff (DP)**
5	RW	Rst212	**Enable write protection for range 0x1e6e4000 - 0x1e6e40ff (JTAG Master 1)**
4	RW	Rst212	**Enable write protection for range 0x1e6fa000 - 0x1e6fa07f (ACRY)**
3	RW	Rst212	**Enable write protection for range 0x1e6e2000 - 0x1e6e2fff (SCU part 1)**
2	RW	Rst212	**Enable write protection for range 0x1e6e1000 - 0x1e6e107f (USB 1.1 HID)**
1	RW	Rst212	**Enable write protection for range 0x1e6e0000 - 0x1e6e07ff (MMC)**
0	RW	Rst212	**Enable write protection for range 0x14000000 - 0x14ffffff (ARM Coresight Interface)**

Offset: 298h **AHBC298: TrustZone Write Protection Register 3** **Init = 0x0**

Bit	R/W	Reset	Description
31	RW	Rst212	**Enable write protection for range 0x1e790200 - 0x1e7902ff (UART8)**
30	RW	Rst212	**Enable write protection for range 0x1e790100 - 0x1e7901ff (UART7)**
29	RW	Rst212	**Enable write protection for range 0x1e790000 - 0x1e7900ff (UART6)**
28	RW	Rst212	**Enable write protection for range 0x1e78f000 - 0x1e78ffff (UART4)**
27	RW	Rst212	**Enable write protection for range 0x1e78e000 - 0x1e78efff (UART3)**
26	RW	Rst212	**Enable write protection for range 0x1e78d000 - 0x1e78dfff (UART2)**
25	RW	Rst212	**Enable write protection for range 0x1e783000 - 0x1e783fff (UART1)**
24	RW	Rst212	**Enable write protection for range 0x1e780000 - 0x1e7807ff (GPIO)**
23	RW	Rst212	**Enable write protection for range 0x1e6e4100 - 0x1e6e41ff (JTAG Master 2)**
22	RW	Rst212	**Enable write protection for range 0x1e6f3800 - 0x1e6f387f (Internal Bridge Controller part 2)**
21	RW	Rst212	**Enable write protection for range 0x1e6fb000 - 0x1e6fbfff (ESPI Master)**
20	RW	Rst212	**Enable write protection for range 0x1e6f2000 - 0x1e6f27ff (Secure Boot MPU part 1)**
19	RW	Rst212	**Enable write protection for range 0x1e6ef000 - 0x1e6effff (BSRAM)**
18	RW	Rst212	**Enable write protection for range 0x1e6ee000 - 0x1e6eefff (ESPI)**
17	RW	Rst212	**Reserved (0)**
16	RW	Rst212	**Enable write protection for range 0x1e6e9000 - 0x1e6e9fff (ADC)**
15	RW	Rst212	**Enable write protection for range 0x1e6e2000 - 0x1e6e2fff (SCU part 2)**
14	RW	Rst212	**Enable write protection for range 0x1e6c8000 - 0x1e6cffff (VIC part 2)**
13	RW	Rst212	**Enable write protection for range 0x19000000 - 0x1903ffff (AST MCU Memory)**
12	RW	Rst212	**Enable write protection for range 0x1e7d0000 - 0x1e7dffff (FSI AHB)**

to next page

from previous page

11	RW	Rst212	**Enable write protection for range 0x1e610000 - 0x1e6101ff (PWM/TACHO)**
10	RW	Rst212	**Enable write protection for range 0x1e651000 - 0x1e651fff (I3C HDMA)**
9	RW	Rst212	**Enable write protection for range 0x1e740000 - 0x1e74ffff (SD/SDIO)**
8	RW	Rst212	**Enable write protection for range 0x50000000 - 0x5fffffff (SPI2 Memory)**
7	RW	Rst212	**Enable write protection for range 0x30000000 - 0x3fffffff (SPI1 Memory)**
6	RW	Rst212	**Reserved (0)**
5	RW	Rst212	**Enable write protection for range 0x1e630000 - 0x1e63ffff (SPI controllers)**
4	RW	Rst212	**Enable write protection for range 0x1e690000 - 0x1e6903ff (MAC4)**
3	RW	Rst212	**Enable write protection for range 0x1e670000 - 0x1e6503ff (MAC3)**
2	RW	Rst212	**Enable write protection for range 0x1e650000 - 0x1e65007f (MII)**
1	RW	Rst212	**Enable write protection for range 0x16000000 - 0x17ffffff (AHB Bus to LPC Bus Bridge)**
0	RW	Rst212	**Reserved (0)**

Offset: 29Ch **AHBC29C: TrustZone Write Protection Register 4** **Init = 0x0**

Bit	R/W	Reset	Description
31:23	RW	Rst212	**Reserved (0)**
22	RW	Rst212	**Enable write protection for range 0x1e7a8000 - 0x1e7a8fff (I2CS)**
21	RW	Rst212	**Enable write protection for range 0x1e7a7000 - 0x1e7a7fff (I3C 6 register)**
20	RW	Rst212	**Enable write protection for range 0x1e7a6000 - 0x1e7a6fff (I3C 5 register)**
19	RW	Rst212	**Enable write protection for range 0x1e7a5000 - 0x1e7a5fff (I3C 4 register)**
18	RW	Rst212	**Enable write protection for range 0x1e7a4000 - 0x1e7a4fff (I3C 3 register)**
17	RW	Rst212	**Enable write protection for range 0x1e7a3000 - 0x1e7a3fff (I3C 2 register)**
16	RW	Rst212	**Enable write protection for range 0x1e7a2000 - 0x1e7a2fff (I3C 1 register)**
15	RW	Rst212	**Reserved (0)**
14	RW	Rst212	**Enable write protection for range 0x1e7a0000 - 0x1e7a0fff (I3C global register)**
13	RW	Rst212	**Enable write protection for range 0x1e79b000 - 0x1e79bfff (FSI)**
12	RW	Rst212	**Enable write protection for range 0x1e78b000 - 0x1e78bfff (PECI)**
11	RW	Rst212	**Enable write protection for range 0x1e78a000 - 0x1e78afff (I2C)**
10	RW	Rst212	**Enable write protection for range 0x1e789000 - 0x1e7897ff (LPC)**
9	RW	Rst212	**Enable write protection for range 0x1e788000 - 0x1e7887ff (PUART)**
8	RW	Rst212	**Enable write protection for range 0x1e787000 - 0x1e7877ff (VUART)**
7	RW	Rst212	**Enable write protection for range 0x1e785000 - 0x1e7851ff (WDT)**
6	RW	Rst212	**Enable write protection for range 0x1e79f800 - 0x1e79ffff (UART1 debug)**
5	RW	Rst212	**Enable write protection for range 0x1e79e000 - 0x1e79e7ff (UART DMA part 1)**
4	RW	Rst212	**Enable write protection for range 0x1e790700 - 0x1e7907ff (UART13)**
3	RW	Rst212	**Enable write protection for range 0x1e790600 - 0x1e7906ff (UART12)**
2	RW	Rst212	**Enable write protection for range 0x1e790500 - 0x1e7905ff (UART11)**
1	RW	Rst212	**Enable write protection for range 0x1e790400 - 0x1e7904ff (UART10)**
0	RW	Rst212	**Enable write protection for range 0x1e790300 - 0x1e7903ff (UART9)**

Offset: 300h	AHBC300: TrustZone Memory Region 1 Protection Start Register		Init = 0x0
Bit	**R/W**	**Reset**	**Description**
31:12	RW	Rst212	**Protection Region 1 Start Address**
11:1	RO	-	**Reserved (0)**
0	RW	Rst212	**Protection Region 1 Enable**

Offset: 304h	AHBC304: TrustZone Memory Region 1 Protection End Register		Init = 0x0
Bit	**R/W**	**Reset**	**Description**
31:12	RW	Rst212	**Protection Region 1 End Address**
11:0	RO	-	**Reserved (0)**

Offset: 308h	AHBC308: TrustZone Memory Region 1 Protection Master Register		Init = 0x0
Bit	**R/W**	**Reset**	**Description**
31:28	RO	-	**Reserved (0)**
27	RW	Rst212	**Allow Access from DRAM REQ27**
26	RW	Rst212	**Allow Access from DRAM REQ26**
25	RW	Rst212	**Allow Access from DRAM REQ25**
24	RW	Rst212	**Allow Access from DRAM REQ24**
23	RW	Rst212	**Allow Access from DRAM REQ23**
22	RW	Rst212	**Allow Access from DRAM REQ22**
21	RW	Rst212	**Allow Access from DRAM REQ21**
20	RW	Rst212	**Allow Access from DRAM REQ20**
19	RW	Rst212	**Allow Access from DRAM REQ19**
18	RW	Rst212	**Allow Access from DRAM REQ18**
17	RW	Rst212	**Allow Access from DRAM REQ17**
16	RW	Rst212	**Allow Access from DRAM REQ16**
15	RW	Rst212	**Allow Access from DRAM REQ15**
14	RW	Rst212	**Allow Access from DRAM REQ14**
13	RW	Rst212	**Allow Access from DRAM REQ13**
12	RW	Rst212	**Allow Access from DRAM REQ12**
11	RW	Rst212	**Allow Access from DRAM REQ11**
10	RW	Rst212	**Allow Access from DRAM REQ10**
9	RW	Rst212	**Allow Access from DRAM REQ9**
8	RW	Rst212	**Allow Access from DRAM REQ8**
7	RW	Rst212	**Allow Access from DRAM REQ7**
6	RW	Rst212	**Allow Access from DRAM REQ6**
5	RW	Rst212	**Allow Access from DRAM REQ5**
4	RW	Rst212	**Allow Access from DRAM REQ4**
3	RW	Rst212	**Allow Access from DRAM REQ3**
2	RW	Rst212	**Allow Access from DRAM REQ2**
1	RW	Rst212	**Allow Access from DRAM REQ1**
0	RW	Rst212	**Allow Access from DRAM REQ0**

Offset: 310h	AHBC310: TrustZone Memory Region 2 Protection Start Register		Init = 0x0
Bit	**R/W**	**Reset**	**Description**
31:12	RW	Rst212	**Protection Region 2 Start Address**
11:1	RO	-	**Reserved (0)**
0	RW	Rst212	**Protection Region 2 Enable**

Offset: 314h	AHBC314: TrustZone Memory Region 2 Protection End Register		Init = 0x0
Bit	**R/W**	**Reset**	**Description**
31:12	RW	Rst212	**Protection Region 2 End Address**
11:0	RO	-	**Reserved (0)**

Offset: 318h	AHBC318: TrustZone Memory Region 2 Protection Master Register		Init = 0x0
Bit	**R/W**	**Reset**	**Description**
31:28	RO	-	**Reserved (0)**
27	RW	Rst212	**Allow Access from DRAM REQ27**
26	RW	Rst212	**Allow Access from DRAM REQ26**
25	RW	Rst212	**Allow Access from DRAM REQ25**
24	RW	Rst212	**Allow Access from DRAM REQ24**
23	RW	Rst212	**Allow Access from DRAM REQ23**
22	RW	Rst212	**Allow Access from DRAM REQ22**
21	RW	Rst212	**Allow Access from DRAM REQ21**
20	RW	Rst212	**Allow Access from DRAM REQ20**
19	RW	Rst212	**Allow Access from DRAM REQ19**
18	RW	Rst212	**Allow Access from DRAM REQ18**
17	RW	Rst212	**Allow Access from DRAM REQ17**
16	RW	Rst212	**Allow Access from DRAM REQ16**
15	RW	Rst212	**Allow Access from DRAM REQ15**
14	RW	Rst212	**Allow Access from DRAM REQ14**
13	RW	Rst212	**Allow Access from DRAM REQ13**
12	RW	Rst212	**Allow Access from DRAM REQ12**
11	RW	Rst212	**Allow Access from DRAM REQ11**
10	RW	Rst212	**Allow Access from DRAM REQ10**
9	RW	Rst212	**Allow Access from DRAM REQ9**
8	RW	Rst212	**Allow Access from DRAM REQ8**
7	RW	Rst212	**Allow Access from DRAM REQ7**
6	RW	Rst212	**Allow Access from DRAM REQ6**
5	RW	Rst212	**Allow Access from DRAM REQ5**
4	RW	Rst212	**Allow Access from DRAM REQ4**
3	RW	Rst212	**Allow Access from DRAM REQ3**
2	RW	Rst212	**Allow Access from DRAM REQ2**
1	RW	Rst212	**Allow Access from DRAM REQ1**
0	RW	Rst212	**Allow Access from DRAM REQ0**

Offset: 320h	AHBC320: TrustZone Memory Region 3 Protection Start Register		Init = 0x0
Bit	**R/W**	**Reset**	**Description**
31:12	RW	Rst212	**Protection Region 3 Start Address**
11:1	RO	-	**Reserved (0)**
0	RW	Rst212	**Protection Region 3 Enable**

Offset: 324h	AHBC324: TrustZone Memory Region 3 Protection End Register		Init = 0x0
Bit	**R/W**	**Reset**	**Description**
31:12	RW	Rst212	**Protection Region 3 End Address**
11:0	RO	-	**Reserved (0)**

Offset: 328h	AHBC328: TrustZone Memory Region 3 Protection Master Register		Init = 0x0
Bit	**R/W**	**Reset**	**Description**
31:28	RO	-	**Reserved (0)**
27	RW	Rst212	**Allow Access from DRAM REQ27**
26	RW	Rst212	**Allow Access from DRAM REQ26**
25	RW	Rst212	**Allow Access from DRAM REQ25**
24	RW	Rst212	**Allow Access from DRAM REQ24**
23	RW	Rst212	**Allow Access from DRAM REQ23**
22	RW	Rst212	**Allow Access from DRAM REQ22**
21	RW	Rst212	**Allow Access from DRAM REQ21**
20	RW	Rst212	**Allow Access from DRAM REQ20**
19	RW	Rst212	**Allow Access from DRAM REQ19**
18	RW	Rst212	**Allow Access from DRAM REQ18**
17	RW	Rst212	**Allow Access from DRAM REQ17**
16	RW	Rst212	**Allow Access from DRAM REQ16**
15	RW	Rst212	**Allow Access from DRAM REQ15**
14	RW	Rst212	**Allow Access from DRAM REQ14**
13	RW	Rst212	**Allow Access from DRAM REQ13**
12	RW	Rst212	**Allow Access from DRAM REQ12**
11	RW	Rst212	**Allow Access from DRAM REQ11**
10	RW	Rst212	**Allow Access from DRAM REQ10**
9	RW	Rst212	**Allow Access from DRAM REQ9**
8	RW	Rst212	**Allow Access from DRAM REQ8**
7	RW	Rst212	**Allow Access from DRAM REQ7**
6	RW	Rst212	**Allow Access from DRAM REQ6**
5	RW	Rst212	**Allow Access from DRAM REQ5**
4	RW	Rst212	**Allow Access from DRAM REQ4**
3	RW	Rst212	**Allow Access from DRAM REQ3**
2	RW	Rst212	**Allow Access from DRAM REQ2**
1	RW	Rst212	**Allow Access from DRAM REQ1**
0	RW	Rst212	**Allow Access from DRAM REQ0**

Offset: 330h	AHBC330: TrustZone Memory Region 4 Protection Start Register		Init = 0x0
Bit	**R/W**	**Reset**	**Description**
31:12	RW	Rst212	**Protection Region 4 Start Address**
11:1	RO	-	**Reserved (0)**
0	RW	Rst212	**Protection Region 4 Enable**

Offset: 334h	AHBC334: TrustZone Memory Region 4 Protection End Register		Init = 0x0
Bit	**R/W**	**Reset**	**Description**
31:12	RW	Rst212	**Protection Region 4 End Address**
11:0	RO	-	**Reserved (0)**

Offset: 338h	AHBC338: TrustZone Memory Region 4 Protection Master Register		Init = 0x0
Bit	**R/W**	**Reset**	**Description**
31:28	RO	-	**Reserved (0)**
27	RW	Rst212	**Allow Access from DRAM REQ27**
26	RW	Rst212	**Allow Access from DRAM REQ26**
25	RW	Rst212	**Allow Access from DRAM REQ25**
24	RW	Rst212	**Allow Access from DRAM REQ24**
23	RW	Rst212	**Allow Access from DRAM REQ23**
22	RW	Rst212	**Allow Access from DRAM REQ22**
21	RW	Rst212	**Allow Access from DRAM REQ21**
20	RW	Rst212	**Allow Access from DRAM REQ20**
19	RW	Rst212	**Allow Access from DRAM REQ19**
18	RW	Rst212	**Allow Access from DRAM REQ18**
17	RW	Rst212	**Allow Access from DRAM REQ17**
16	RW	Rst212	**Allow Access from DRAM REQ16**
15	RW	Rst212	**Allow Access from DRAM REQ15**
14	RW	Rst212	**Allow Access from DRAM REQ14**
13	RW	Rst212	**Allow Access from DRAM REQ13**
12	RW	Rst212	**Allow Access from DRAM REQ12**
11	RW	Rst212	**Allow Access from DRAM REQ11**
10	RW	Rst212	**Allow Access from DRAM REQ10**
9	RW	Rst212	**Allow Access from DRAM REQ9**
8	RW	Rst212	**Allow Access from DRAM REQ8**
7	RW	Rst212	**Allow Access from DRAM REQ7**
6	RW	Rst212	**Allow Access from DRAM REQ6**
5	RW	Rst212	**Allow Access from DRAM REQ5**
4	RW	Rst212	**Allow Access from DRAM REQ4**
3	RW	Rst212	**Allow Access from DRAM REQ3**
2	RW	Rst212	**Allow Access from DRAM REQ2**
1	RW	Rst212	**Allow Access from DRAM REQ1**
0	RW	Rst212	**Allow Access from DRAM REQ0**

Offset: 340h	AHBC340: TrustZone Memory Region 5 Protection Start Register		Init = 0x0
Bit	**R/W**	**Reset**	**Description**
31:12	RW	Rst212	**Protection Region 5 Start Address**
11:1	RO	-	**Reserved (0)**
0	RW	Rst212	**Protection Region 5 Enable**

Offset: 344h	AHBC344: TrustZone Memory Region 5 Protection End Register		Init = 0x0
Bit	**R/W**	**Reset**	**Description**
31:12	RW	Rst212	**Protection Region 5 End Address**
11:0	RO	-	**Reserved (0)**

Offset: 348h	AHBC348: TrustZone Memory Region 5 Protection Master Register		Init = 0x0
Bit	**R/W**	**Reset**	**Description**
31:28	RO	-	**Reserved (0)**
27	RW	Rst212	**Allow Access from DRAM REQ27**
26	RW	Rst212	**Allow Access from DRAM REQ26**
25	RW	Rst212	**Allow Access from DRAM REQ25**
24	RW	Rst212	**Allow Access from DRAM REQ24**
23	RW	Rst212	**Allow Access from DRAM REQ23**
22	RW	Rst212	**Allow Access from DRAM REQ22**
21	RW	Rst212	**Allow Access from DRAM REQ21**
20	RW	Rst212	**Allow Access from DRAM REQ20**
19	RW	Rst212	**Allow Access from DRAM REQ19**
18	RW	Rst212	**Allow Access from DRAM REQ18**
17	RW	Rst212	**Allow Access from DRAM REQ17**
16	RW	Rst212	**Allow Access from DRAM REQ16**
15	RW	Rst212	**Allow Access from DRAM REQ15**
14	RW	Rst212	**Allow Access from DRAM REQ14**
13	RW	Rst212	**Allow Access from DRAM REQ13**
12	RW	Rst212	**Allow Access from DRAM REQ12**
11	RW	Rst212	**Allow Access from DRAM REQ11**
10	RW	Rst212	**Allow Access from DRAM REQ10**
9	RW	Rst212	**Allow Access from DRAM REQ9**
8	RW	Rst212	**Allow Access from DRAM REQ8**
7	RW	Rst212	**Allow Access from DRAM REQ7**
6	RW	Rst212	**Allow Access from DRAM REQ6**
5	RW	Rst212	**Allow Access from DRAM REQ5**
4	RW	Rst212	**Allow Access from DRAM REQ4**
3	RW	Rst212	**Allow Access from DRAM REQ3**
2	RW	Rst212	**Allow Access from DRAM REQ2**
1	RW	Rst212	**Allow Access from DRAM REQ1**
0	RW	Rst212	**Allow Access from DRAM REQ0**

Offset: 350h	AHBC350: TrustZone Memory Region 6 Protection Start Register		Init = 0x0
Bit	**R/W**	**Reset**	**Description**
31:12	RW	Rst212	**Protection Region 6 Start Address**
11:1	RO	-	**Reserved (0)**
0	RW	Rst212	**Protection Region 6 Enable**

Offset: 354h	AHBC354: TrustZone Memory Region 6 Protection End Register		Init = 0x0
Bit	**R/W**	**Reset**	**Description**
31:12	RW	Rst212	**Protection Region 6 End Address**
11:0	RO	-	**Reserved (0)**

Offset: 358h	AHBC358: TrustZone Memory Region 6 Protection Master Register		Init = 0x0
Bit	**R/W**	**Reset**	**Description**
31:28	RO	-	**Reserved (0)**
27	RW	Rst212	**Allow Access from DRAM REQ27**
26	RW	Rst212	**Allow Access from DRAM REQ26**
25	RW	Rst212	**Allow Access from DRAM REQ25**
24	RW	Rst212	**Allow Access from DRAM REQ24**
23	RW	Rst212	**Allow Access from DRAM REQ23**
22	RW	Rst212	**Allow Access from DRAM REQ22**
21	RW	Rst212	**Allow Access from DRAM REQ21**
20	RW	Rst212	**Allow Access from DRAM REQ20**
19	RW	Rst212	**Allow Access from DRAM REQ19**
18	RW	Rst212	**Allow Access from DRAM REQ18**
17	RW	Rst212	**Allow Access from DRAM REQ17**
16	RW	Rst212	**Allow Access from DRAM REQ16**
15	RW	Rst212	**Allow Access from DRAM REQ15**
14	RW	Rst212	**Allow Access from DRAM REQ14**
13	RW	Rst212	**Allow Access from DRAM REQ13**
12	RW	Rst212	**Allow Access from DRAM REQ12**
11	RW	Rst212	**Allow Access from DRAM REQ11**
10	RW	Rst212	**Allow Access from DRAM REQ10**
9	RW	Rst212	**Allow Access from DRAM REQ9**
8	RW	Rst212	**Allow Access from DRAM REQ8**
7	RW	Rst212	**Allow Access from DRAM REQ7**
6	RW	Rst212	**Allow Access from DRAM REQ6**
5	RW	Rst212	**Allow Access from DRAM REQ5**
4	RW	Rst212	**Allow Access from DRAM REQ4**
3	RW	Rst212	**Allow Access from DRAM REQ3**
2	RW	Rst212	**Allow Access from DRAM REQ2**
1	RW	Rst212	**Allow Access from DRAM REQ1**
0	RW	Rst212	**Allow Access from DRAM REQ0**

Offset: 360h	AHBC360: TrustZone Memory Region 7 Protection Start Register		Init = 0x0
Bit	**R/W**	**Reset**	**Description**
31:12	RW	Rst212	**Protection Region 7 Start Address**
11:1	RO	-	**Reserved (0)**
0	RW	Rst212	**Protection Region 7 Enable**

Offset: 364h	AHBC364: TrustZone Memory Region 7 Protection End Register		Init = 0x0
Bit	**R/W**	**Reset**	**Description**
31:12	RW	Rst212	**Protection Region 7 End Address**
11:0	RO	-	**Reserved (0)**

Offset: 368h	AHBC368: TrustZone Memory Region 7 Protection Master Register		Init = 0x0
Bit	**R/W**	**Reset**	**Description**
31:28	RO	-	**Reserved (0)**
27	RW	Rst212	**Allow Access from DRAM REQ27**
26	RW	Rst212	**Allow Access from DRAM REQ26**
25	RW	Rst212	**Allow Access from DRAM REQ25**
24	RW	Rst212	**Allow Access from DRAM REQ24**
23	RW	Rst212	**Allow Access from DRAM REQ23**
22	RW	Rst212	**Allow Access from DRAM REQ22**
21	RW	Rst212	**Allow Access from DRAM REQ21**
20	RW	Rst212	**Allow Access from DRAM REQ20**
19	RW	Rst212	**Allow Access from DRAM REQ19**
18	RW	Rst212	**Allow Access from DRAM REQ18**
17	RW	Rst212	**Allow Access from DRAM REQ17**
16	RW	Rst212	**Allow Access from DRAM REQ16**
15	RW	Rst212	**Allow Access from DRAM REQ15**
14	RW	Rst212	**Allow Access from DRAM REQ14**
13	RW	Rst212	**Allow Access from DRAM REQ13**
12	RW	Rst212	**Allow Access from DRAM REQ12**
11	RW	Rst212	**Allow Access from DRAM REQ11**
10	RW	Rst212	**Allow Access from DRAM REQ10**
9	RW	Rst212	**Allow Access from DRAM REQ9**
8	RW	Rst212	**Allow Access from DRAM REQ8**
7	RW	Rst212	**Allow Access from DRAM REQ7**
6	RW	Rst212	**Allow Access from DRAM REQ6**
5	RW	Rst212	**Allow Access from DRAM REQ5**
4	RW	Rst212	**Allow Access from DRAM REQ4**
3	RW	Rst212	**Allow Access from DRAM REQ3**
2	RW	Rst212	**Allow Access from DRAM REQ2**
1	RW	Rst212	**Allow Access from DRAM REQ1**
0	RW	Rst212	**Allow Access from DRAM REQ0**

Offset: 370h	AHBC370: TrustZone Memory Region 8 Protection Start Register		Init = 0x0
Bit	**R/W**	**Reset**	**Description**
31:12	RW	Rst212	**Protection Region 8 Start Address**
11:1	RO	-	**Reserved (0)**
0	RW	Rst212	**Protection Region 8 Enable**

Offset: 374h	AHBC374: TrustZone Memory Region 8 Protection End Register		Init = 0x0
Bit	**R/W**	**Reset**	**Description**
31:12	RW	Rst212	**Protection Region 8 End Address**
11:0	RO	-	**Reserved (0)**

Offset: 378h	AHBC378: TrustZone Memory Region 8 Protection Master Register		Init = 0x0
Bit	**R/W**	**Reset**	**Description**
31:28	RO	-	**Reserved (0)**
27	RW	Rst212	**Allow Access from DRAM REQ27**
26	RW	Rst212	**Allow Access from DRAM REQ26**
25	RW	Rst212	**Allow Access from DRAM REQ25**
24	RW	Rst212	**Allow Access from DRAM REQ24**
23	RW	Rst212	**Allow Access from DRAM REQ23**
22	RW	Rst212	**Allow Access from DRAM REQ22**
21	RW	Rst212	**Allow Access from DRAM REQ21**
20	RW	Rst212	**Allow Access from DRAM REQ20**
19	RW	Rst212	**Allow Access from DRAM REQ19**
18	RW	Rst212	**Allow Access from DRAM REQ18**
17	RW	Rst212	**Allow Access from DRAM REQ17**
16	RW	Rst212	**Allow Access from DRAM REQ16**
15	RW	Rst212	**Allow Access from DRAM REQ15**
14	RW	Rst212	**Allow Access from DRAM REQ14**
13	RW	Rst212	**Allow Access from DRAM REQ13**
12	RW	Rst212	**Allow Access from DRAM REQ12**
11	RW	Rst212	**Allow Access from DRAM REQ11**
10	RW	Rst212	**Allow Access from DRAM REQ10**
9	RW	Rst212	**Allow Access from DRAM REQ9**
8	RW	Rst212	**Allow Access from DRAM REQ8**
7	RW	Rst212	**Allow Access from DRAM REQ7**
6	RW	Rst212	**Allow Access from DRAM REQ6**
5	RW	Rst212	**Allow Access from DRAM REQ5**
4	RW	Rst212	**Allow Access from DRAM REQ4**
3	RW	Rst212	**Allow Access from DRAM REQ3**
2	RW	Rst212	**Allow Access from DRAM REQ2**
1	RW	Rst212	**Allow Access from DRAM REQ1**
0	RW	Rst212	**Allow Access from DRAM REQ0**

Offset: 380h	AHBC380: TrustZone Memory Region 9 Protection Start Register		Init = 0x0
Bit	**R/W**	**Reset**	**Description**
31:12	RW	Rst212	**Protection Region 9 Start Address**
11:1	RO	-	**Reserved (0)**
0	RW	Rst212	**Protection Region 9 Enable**

Offset: 384h	AHBC384: TrustZone Memory Region 9 Protection End Register		Init = 0x0
Bit	**R/W**	**Reset**	**Description**
31:12	RW	Rst212	**Protection Region 9 End Address**
11:0	RO	-	**Reserved (0)**

Offset: 388h	AHBC388: TrustZone Memory Region 9 Protection Master Register		Init = 0x0
Bit	**R/W**	**Reset**	**Description**
31:28	RO	-	**Reserved (0)**
27	RW	Rst212	**Allow Access from DRAM REQ27**
26	RW	Rst212	**Allow Access from DRAM REQ26**
25	RW	Rst212	**Allow Access from DRAM REQ25**
24	RW	Rst212	**Allow Access from DRAM REQ24**
23	RW	Rst212	**Allow Access from DRAM REQ23**
22	RW	Rst212	**Allow Access from DRAM REQ22**
21	RW	Rst212	**Allow Access from DRAM REQ21**
20	RW	Rst212	**Allow Access from DRAM REQ20**
19	RW	Rst212	**Allow Access from DRAM REQ19**
18	RW	Rst212	**Allow Access from DRAM REQ18**
17	RW	Rst212	**Allow Access from DRAM REQ17**
16	RW	Rst212	**Allow Access from DRAM REQ16**
15	RW	Rst212	**Allow Access from DRAM REQ15**
14	RW	Rst212	**Allow Access from DRAM REQ14**
13	RW	Rst212	**Allow Access from DRAM REQ13**
12	RW	Rst212	**Allow Access from DRAM REQ12**
11	RW	Rst212	**Allow Access from DRAM REQ11**
10	RW	Rst212	**Allow Access from DRAM REQ10**
9	RW	Rst212	**Allow Access from DRAM REQ9**
8	RW	Rst212	**Allow Access from DRAM REQ8**
7	RW	Rst212	**Allow Access from DRAM REQ7**
6	RW	Rst212	**Allow Access from DRAM REQ6**
5	RW	Rst212	**Allow Access from DRAM REQ5**
4	RW	Rst212	**Allow Access from DRAM REQ4**
3	RW	Rst212	**Allow Access from DRAM REQ3**
2	RW	Rst212	**Allow Access from DRAM REQ2**
1	RW	Rst212	**Allow Access from DRAM REQ1**
0	RW	Rst212	**Allow Access from DRAM REQ0**

Offset: 390h	AHBC390: TrustZone Memory Region 10 Protection Start Register		Init = 0x0
Bit	**R/W**	**Reset**	**Description**
31:12	RW	Rst212	**Protection Region 10 Start Address**
11:1	RO	-	**Reserved (0)**
0	RW	Rst212	**Protection Region 10 Enable**

Offset: 394h	AHBC394: TrustZone Memory Region 10 Protection End Register		Init = 0x0
Bit	**R/W**	**Reset**	**Description**
31:12	RW	Rst212	**Protection Region 10 End Address**
11:0	RO	-	**Reserved (0)**

Offset: 398h	AHBC398: TrustZone Memory Region 10 Protection Master Register		Init = 0x0
Bit	**R/W**	**Reset**	**Description**
31:28	RO	-	**Reserved (0)**
27	RW	Rst212	**Allow Access from DRAM REQ27**
26	RW	Rst212	**Allow Access from DRAM REQ26**
25	RW	Rst212	**Allow Access from DRAM REQ25**
24	RW	Rst212	**Allow Access from DRAM REQ24**
23	RW	Rst212	**Allow Access from DRAM REQ23**
22	RW	Rst212	**Allow Access from DRAM REQ22**
21	RW	Rst212	**Allow Access from DRAM REQ21**
20	RW	Rst212	**Allow Access from DRAM REQ20**
19	RW	Rst212	**Allow Access from DRAM REQ19**
18	RW	Rst212	**Allow Access from DRAM REQ18**
17	RW	Rst212	**Allow Access from DRAM REQ17**
16	RW	Rst212	**Allow Access from DRAM REQ16**
15	RW	Rst212	**Allow Access from DRAM REQ15**
14	RW	Rst212	**Allow Access from DRAM REQ14**
13	RW	Rst212	**Allow Access from DRAM REQ13**
12	RW	Rst212	**Allow Access from DRAM REQ12**
11	RW	Rst212	**Allow Access from DRAM REQ11**
10	RW	Rst212	**Allow Access from DRAM REQ10**
9	RW	Rst212	**Allow Access from DRAM REQ9**
8	RW	Rst212	**Allow Access from DRAM REQ8**
7	RW	Rst212	**Allow Access from DRAM REQ7**
6	RW	Rst212	**Allow Access from DRAM REQ6**
5	RW	Rst212	**Allow Access from DRAM REQ5**
4	RW	Rst212	**Allow Access from DRAM REQ4**
3	RW	Rst212	**Allow Access from DRAM REQ3**
2	RW	Rst212	**Allow Access from DRAM REQ2**
1	RW	Rst212	**Allow Access from DRAM REQ1**
0	RW	Rst212	**Allow Access from DRAM REQ0**

Offset: 3A0h	AHBC3A0: TrustZone Memory Region 11 Protection Start Register		Init = 0x0
Bit	**R/W**	**Reset**	**Description**
31:12	RW	Rst212	**Protection Region 11 Start Address**
11:1	RO	-	**Reserved (0)**
0	RW	Rst212	**Protection Region 11 Enable**

Offset: 3A4h	AHBC3A4: TrustZone Memory Region 11 Protection End Register		Init = 0x0
Bit	**R/W**	**Reset**	**Description**
31:12	RW	Rst212	**Protection Region 11 End Address**
11:0	RO	-	**Reserved (0)**

Offset: 3A8h	AHBC3A8: TrustZone Memory Region 11 Protection Master Register		Init = 0x0
Bit	**R/W**	**Reset**	**Description**
31:28	RO	-	**Reserved (0)**
27	RW	Rst212	**Allow Access from DRAM REQ27**
26	RW	Rst212	**Allow Access from DRAM REQ26**
25	RW	Rst212	**Allow Access from DRAM REQ25**
24	RW	Rst212	**Allow Access from DRAM REQ24**
23	RW	Rst212	**Allow Access from DRAM REQ23**
22	RW	Rst212	**Allow Access from DRAM REQ22**
21	RW	Rst212	**Allow Access from DRAM REQ21**
20	RW	Rst212	**Allow Access from DRAM REQ20**
19	RW	Rst212	**Allow Access from DRAM REQ19**
18	RW	Rst212	**Allow Access from DRAM REQ18**
17	RW	Rst212	**Allow Access from DRAM REQ17**
16	RW	Rst212	**Allow Access from DRAM REQ16**
15	RW	Rst212	**Allow Access from DRAM REQ15**
14	RW	Rst212	**Allow Access from DRAM REQ14**
13	RW	Rst212	**Allow Access from DRAM REQ13**
12	RW	Rst212	**Allow Access from DRAM REQ12**
11	RW	Rst212	**Allow Access from DRAM REQ11**
10	RW	Rst212	**Allow Access from DRAM REQ10**
9	RW	Rst212	**Allow Access from DRAM REQ9**
8	RW	Rst212	**Allow Access from DRAM REQ8**
7	RW	Rst212	**Allow Access from DRAM REQ7**
6	RW	Rst212	**Allow Access from DRAM REQ6**
5	RW	Rst212	**Allow Access from DRAM REQ5**
4	RW	Rst212	**Allow Access from DRAM REQ4**
3	RW	Rst212	**Allow Access from DRAM REQ3**
2	RW	Rst212	**Allow Access from DRAM REQ2**
1	RW	Rst212	**Allow Access from DRAM REQ1**
0	RW	Rst212	**Allow Access from DRAM REQ0**

Offset: 3B0h	AHBC3B0: TrustZone Memory Region 12 Protection Start Register		Init = 0x0
Bit	**R/W**	**Reset**	**Description**
31:12	RW	Rst212	**Protection Region 12 Start Address**
11:1	RO	-	**Reserved (0)**
0	RW	Rst212	**Protection Region 12 Enable**

Offset: 3B4h	AHBC3B4: TrustZone Memory Region 12 Protection End Register		Init = 0x0
Bit	**R/W**	**Reset**	**Description**
31:12	RW	Rst212	**Protection Region 12 End Address**
11:0	RO	-	**Reserved (0)**

Offset: 3B8h	AHBC3B8: TrustZone Memory Region 12 Protection Master Register		Init = 0x0
Bit	**R/W**	**Reset**	**Description**
31:28	RO	-	**Reserved (0)**
27	RW	Rst212	**Allow Access from DRAM REQ27**
26	RW	Rst212	**Allow Access from DRAM REQ26**
25	RW	Rst212	**Allow Access from DRAM REQ25**
24	RW	Rst212	**Allow Access from DRAM REQ24**
23	RW	Rst212	**Allow Access from DRAM REQ23**
22	RW	Rst212	**Allow Access from DRAM REQ22**
21	RW	Rst212	**Allow Access from DRAM REQ21**
20	RW	Rst212	**Allow Access from DRAM REQ20**
19	RW	Rst212	**Allow Access from DRAM REQ19**
18	RW	Rst212	**Allow Access from DRAM REQ18**
17	RW	Rst212	**Allow Access from DRAM REQ17**
16	RW	Rst212	**Allow Access from DRAM REQ16**
15	RW	Rst212	**Allow Access from DRAM REQ15**
14	RW	Rst212	**Allow Access from DRAM REQ14**
13	RW	Rst212	**Allow Access from DRAM REQ13**
12	RW	Rst212	**Allow Access from DRAM REQ12**
11	RW	Rst212	**Allow Access from DRAM REQ11**
10	RW	Rst212	**Allow Access from DRAM REQ10**
9	RW	Rst212	**Allow Access from DRAM REQ9**
8	RW	Rst212	**Allow Access from DRAM REQ8**
7	RW	Rst212	**Allow Access from DRAM REQ7**
6	RW	Rst212	**Allow Access from DRAM REQ6**
5	RW	Rst212	**Allow Access from DRAM REQ5**
4	RW	Rst212	**Allow Access from DRAM REQ4**
3	RW	Rst212	**Allow Access from DRAM REQ3**
2	RW	Rst212	**Allow Access from DRAM REQ2**
1	RW	Rst212	**Allow Access from DRAM REQ1**
0	RW	Rst212	**Allow Access from DRAM REQ0**

13.4 Programming Guide

Giving an example of polling UART5 TXD output, and record to DRAM base 0x81000000 of size 256KB.

13.4.1 Enable Polling

1. write 0x1E600040 = 0x00000502
2. write 0x1E600048 = 0x1E784000
3. write 0x1E600044 = 0x81000000
4. write 0x1E600040 = 0x00000503

13.4.2 Memory Access Protection

1. AHBC300 = 0x83000000
2. AHBC304 = 0x8300F000
3. AHBC308 = 0x3020
4. AHBC300 = 0x83000001

This example shows 0x83000000 ~ 0x8300FFFF are only accessible by CA7 and CM3. Others are not allowed to access this region.

14 Firmware SPI Memory Controller (BSPI)

14.1 Overview

14.1.1 Register List

Base address of FMC controller = 0x1E62_0000
Base address of Boot SPI memory = 0x0000_0000
Base address of FMC SPI memory = 0x2000_0000
Physical address = (Base address of FMC) + Offset

FMC00: CE Type Setting Register
FMC04: CE Control Register
FMC08: Interrupt Control and Status Register
FMC0C: Command Control Register
FMC10: CE0 Control Register
FMC14: CE1 Control Register
FMC18: CE2 Control Register
FMC30: CE0 Address Decoding Range Register
FMC34: CE1 Address Decoding Range Register
FMC38: CE2 Address Decoding Range Register
FMC50: Auto Soft-Reset Command Control
FMC54: SPI Dummy Cycle Data Register
FMC60: FMC_WDT1 Control/Status Register for Address Mode Detection
FMC64: FMC_WDT2 Control/Status Register for Alternate Boot
FMC68: FMC_WDT2 Timer Reload Value Register
FMC6C: FMC_WDT2 Timer Restart Register
FMC7C: DMA Buffer Mode Length Register
FMC80: DMA Control/Status Register
FMC84: DMA Flash Side Address
FMC88: DMA DRAM Side Address
FMC8C: DMA Length Register
FMC90: CheckSum Calculation Result
FMC94: CE0 Read Timing Compensation
FMC98: CE1 Read Timing Compensation
FMC9C: CE2 Read Timing Compensation
FMCA0: Command Filter Control Register
FMCA4: Write Address Filter Control Register
FMCA8: Register Lock Control Register (SRST#)
FMCAC: Register Lock Control Register (Watchdog)
FMCB0: Write Address Filter Register #1
FMCB4: Write Address Filter Register #2
FMCB8: Write Address Filter Register #3
FMCBC: Write Address Filter Register #4
FMCC0: Write Address Filter Register #5
FMCC4: Write Address Filter Register #6
FMCC8: Write Address Filter Register #7
FMCCC: Write Address Filter Register #8
FMC100:FQCD00: Fully Qualified Command
FMC104:FQCD01: Fully Qualified Command
FMC108:FQCD02: Fully Qualified Command
FMC10C:FQCD03: Fully Qualified Command
FMC110:FQCD04: Fully Qualified Command
FMC114:FQCD05: Fully Qualified Command
FMC118:FQCD06: Fully Qualified Command
FMC11C:FQCD07: Fully Qualified Command

FMC120:FQCD08: Fully Qualified Command
FMC124:FQCD09: Fully Qualified Command
FMC128:FQCD10: Fully Qualified Command
FMC12C:FQCD11: Fully Qualified Command
FMC130:FQCD12: Fully Qualified Command
FMC134:FQCD13: Fully Qualified Command
FMC138:FQCD14: Fully Qualified Command
FMC13C:FQCD15: Fully Qualified Command
FMC140:FQCD16: Fully Qualified Command
FMC144:FQCD17: Fully Qualified Command
FMC148:FQCD18: Fully Qualified Command
FMC14C:FQCD19: Fully Qualified Command
FMC150:AQCD00: Address Qualified Command
FMC154:AQCD01: Address Qualified Command
FMC158:AQCD02: Address Qualified Command
FMC15C:AQCD03: Address Qualified Command
FMC160:AQCD04: Address Qualified Command
FMC164:AQCD05: Address Qualified Command
FMC168:AQCD06: Address Qualified Command
FMC16C:AQCD07: Address Qualified Command
FMC170:AQCD08: Address Qualified Command
FMC174:AQCD09: Address Qualified Command
FMC178:AQCD10: Address Qualified Command
FMC17C:AQCD11: Address Qualified Command

FMC200:DMA FIFO Mode Data Port Register
FMCBUF:Offset 0x200 - 0x2FF: DMA Buffer R/W Port

14.1.2 Changed items compared to AST2500

- Remove parallel NOR interface supporting.
- Remove software strap function.
- Merge watchdog timer into flash controller for 3/4 bytes address detection (FMC_WDT1) and alternate (2nd) boot function (FMC_WDT2).
- Change the SPI clock divider range from (HCLK/1 - HCLK/16) to (HCLK/2 - HCLK/256), since HCLK clock rate is higher than 200MHz.
- Re-define the CE minimum inactive timing selection.
- Re-define the CE address decoding range register, increase the resolution to 1MB.
- Re-define the command filters, supports 40 non-address check commands and 24 address check commands.
- Add 0x13 read command option for 4B address Auto-Read command mode.
- Add programmable fine delay tuning on data input path for high speed read timing compensation.
- Add the address filters to 8 sets.
- Add another alternate boot option to support single flash ABR.
- Add support Quad-IO and QPI mode.
- Add support auto-generate soft-reset command to CE0 after watchdog reset.
- Add support auxiliary control pins for BMC SPI ABR/WriteProtect functions, which is enabled by strap bit OTP trap_en_bspi_auxpin .

14.1.3 OTP configuration for BMC SPI function

- OTP trap_en_bspi_auxpin : enable auxiliary control pins for ABR and Write Protect control.
- OTP trap_en_bspiadrswap : enable address mode detection FMC_WDT1 counter.
- OTP trap_en_bspiabr : enable ABR boot FMC_WDT2 counter.
- OTP trap_bspi_abrmode : select BMC SPI ABR function mode, 1 chip or 2 chips.
  - 0 : 2 chips ABR mode
  - 1 : 1 chips ABR mode
  - FMC64[6] is determined by the ABR mode selection, not changable.
- OTP trap_bspi_size [2:0] : define the BMC SPI CE0/(and CE1 if OTP trap_bspi_abrmode =0) flash size.
  - Value definition:
    - 000 : no define size
    - 001 : 2MB
    - 010 : 4MB
    - 011 : 8MB
    - 100 : 16MB
    - 101 : 32MB
    - 110 : 64MB
    - 111 : 128MB
  - When OTP trap_bspi_size [2:0] $\neq$ 0, FMC04[0] and FMC04[4] is determined by SPI size after reset if OTP trap_en_bspiadrswap = 0. Bit 0 and 4 is 0 if OTP trap_bspi_size [2:0] $\leq$ 16MB and 1 otherwise. Also these 2 bits are RO if OTP trap_bspi_size [2:0] $\neq$ 0.
  - When OTP trap_bspi_size [2:0] $\neq$ 0, FMC30[31:16] is determined by SPI size after reset. They are RO if OTP trap_bspi_size [2:0] $\neq$ 0.
  - When OTP trap_bspi_size [2:0] $\neq$ 0, FMC34[31:00] is determined by SPI size after reset if OTP trap_bspi_abrmode = 0. They are RO if OTP trap_bspi_size [2:0] $\neq$ 0.
  - To support SPI size larger than 128MB, set OTP trap_bspi_size [2:0] = 0. Then firmware can program FMC30 to configure the maximum direct fetch address space for CE0.
- OTP trap_bspi_crtmsize [1:0] : define CRTM write protection size, the CRTM protect region is located at the bottom of code area.
  - CRTM only works when OTP trap_en_bspi_auxpin = 1, OTP trap_en_bspiabr = 1, and FWSPIWP# is active
  - Value definition:
    - 00 : disable CRTM
    - 01 : 256KB
    - 10 : 512KB
    - 11 : 1MB
  - When CRTM is enabled, FMCA4[3:0] = 0xF. And FMCB0/FMCB4 are determined by ABR mode, SPI size, and CRTM size, not changable.

14.1.4 Auxiliary control pins behavior (OTP trap_en_bspi_auxpin = 1)

- FWSPIABR : alternate boot source selection pin
  - FWSPIABR = low state, boot source is determined by internal ABR watchdog FMC_WDT2, initial boot from primary code
  - FWSPIABR = high state, force to boot from alternate code
- FWSPIWP# : write protect control input pin, active low
  - When active, FWSPIWP# = low state (FMCA0[0]=1)

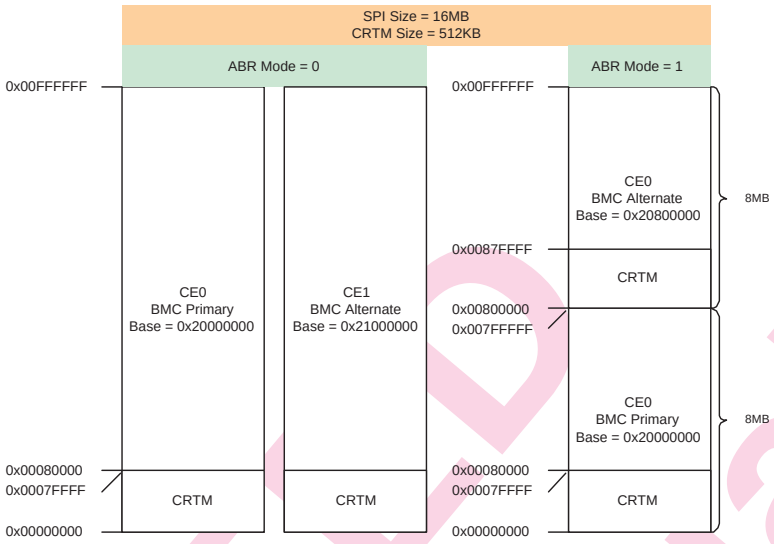


Figure 43: BMC SPI CRTM Allocation

- CRTM area take effect.
- SPI transaction block for BMC access is enabled.
- BMC access to SPI commands are filtered by FQCD and AQCD.
- Registers from FMCA0 to FMC17C are write protected.

* When external pin control of transaction block is not enabled, then software can control the transaction block through related registers.

– Enable additional 2 data pins to support 4-bits IO mode.

– Registers between FMCA0 to FMC17C are only writable by ARM CPU when FWSPIWP# is inactive.

14.1.5 SPI flash selection requirement

– Minimum read speed $\geq$ 35MHz

– When OTP trap_bspi_size is defined and larger than 16MB, it should choose the SPI flash part that support **0x13** read command for CE0.

14.2 Features

This SPI controller provides 3 chip select pins (CE0 $\sim$ CE2) to control at most 3 flash memory devices. And CE0 $\sim$ CE2 can be assigned to different non-overlapping and continuous address regions with programmable starting/ending address. The allowed flash memory address space is:

– 0x2000_0000 $\sim$ 0x2FFF_FFFF.

Due to hardware pin share limitation, only 1 CE can be activated at a time.

CE0 is the default firmware booting source. The base address of CE0 is default set at:

1. 0x0000_0000
2. 0x2000_0000

For different size support, firmware must change the accessed CE or modify the address decoding range defined at FMC30.

14.2.1 SPI master controller features

- Support maximum 256 MBytes direct access memory space for each chip select.
- Programmable clock speed, HCLK/2 ∼ HCLK/256.
- Support 1, 2, 4 bits and QPI SPI IO modes.
- Support 24bits and 32bits address mode.
- Support high clock rate (> 50MHz) read timing calibration function.
- Support DMA for bulk data transfer between SPI flash and DRAM/SRAM.

14.2.2 Alternate (2nd) Boot Recovery Function (ABR)

- Alternate boot recovery function is enabled by strap OTP trap_en_bspiabr .
- When the firmware booting successfully, it should disable FMC_WDT2 to stop booting switch.
- The default expire time setting for boot switching is 22 seconds. If firmware booting time is longer than 22 seconds, it should change the expire time and restart counting within 22 seconds.
- Supports 2 modes of alternate boot function:
  1. Use 2 SPI flash parts, CE0 and CE1. CE0 is primary boot source, and alternate boot from CE1. Flash size information is optional for this boot mode.
  2. Use 1 SPI flash part, CE0. Uniformly split the flash space into 2 parts of same size. Lower partition is the default boot area, and alternate boot from upper partition. Flash size information is required for this boot mode.
- After the FMC_WDT2 timeout, then it will reset CPU and switch the boot source between 2 code areas. The 2nd boot function will keep running until firmware disable it.
- When use 2 flash mode and boot from CE1 flash, at this time the address mapping for CE0 and CE1 are swapped. To restore the address mapping, firmware should clear the boot select flag at FMC64.bit[4].
- When use 1 flash mode and boot from upper partition, at this time the address mapping for lower and upper partition are swapped. To restore the address mapping, firmware should clear the boot select flag at FMC64.bit[4].
- The firmware code in CE1 flash can be the same as the main flash.

14.2.3 Address mode Detect Function

- Address mode 3B/4B toggling function is enabled by strap OTP trap_en_bspiadrswap .
- Flash controller has the capability that cooperate with firmware to detect the SPI flash address mode.
- It use an watchdog timer FMC_WDT1 with expire time of 2 seconds and enabled at CPU booting start. This watchdog timer will be disabled by firmware at very early stage if firmware can be executed normally. If timer timeout, then CPU will be reset and SPI address mode will be switched from 3B to 4B or from 4B to 3B.
- This operation will be repeated at most 2 times and then stop. Which means the firmware may have problem and can not boot normally.
- This operation is only applied on the booting flash, CE0.

14.2.4 DMA Function

- Support maximum 32 MByte data transfer length.
- Support 2 operation modes.
  1. Memory Mode: transfer data between DRAM/SRAM and SPI memory.

2. Buffer Mode: transfer data between DMA buffer and SPI memory.

- Support flash data checksum calculation (32 bits data unit).
- Support 256 bytes DMA buffer. The DMA buffer can be accessed at offset range 0x200 - 0x2FF when DMA function is disabled.
- Support 4-byte boundary for SPI memory side address, and 4-byte boundary for DRAM/SRAM side address.
- Support 1-byte boundary for transfer length.
- When work at DMA buffer mode:
  * Firmware can preload TX data into DMA buffer and set preload length in FMC7C before DMA is enabled.
  * DMA buffer works as FIFO mode after DMA is enabled.
  * When DMA is on-going, firmware can push/pop data at dedicated data port for more data of TX/RX.
  * The total transfer length is defined in FMC8C.
  * DMA will stop temporarily and keep CE# active when DMA buffer is full or empty.
  * Support FIFO full/empty interrupt.

14.3 Registers : Base Address = 0x1E62:0000

Offset: 00h	FMC00: CE Type Setting Register	Init = 0x2A

Bit	R/W	Description
31:19	RO	**Reserved (0)**
18	RW	**Enable CE2 default write type**
17	RW	**Enable CE1 default write type**
16	RW	**Enable CE0 default write type** 0: CEx is default at write disable mode 1: CEx is default at write enable mode The default write type is a global range control for a CE. This default write type in companion with the Write Address Filter register at FMCA4 control the write protection behavior of a CE. Protection mode as below: 1. If FMC00 is write disabled, the write can be enabled by FMCA4. 2. If FMC00 is write enabled, then write can be disabled by FMCA4.
15:6	RO	**Reserved (0)**
5:4	RO	**CE2 flash type selection**
3:2	RO	**CE1 flash type selection**
1:0	RO	**CE0 flash type selection** 10: Select SPI flash type

Offset: 04h	FMC04: CE Control Register	Init = 0x00002A00

Bit	R/W	Description
31:14	RO	**Reserved (0)**
13:12	RW	**Select CE2 minimum inactive timing (tCSHigh)**
11:10	RW	**Select CE1 minimum inactive timing (tCSHigh)**
9:8	RW	**Select CE0 minimum inactive timing (tCSHigh)** 00: 16 x HCLK 01: 24 x HCLK 10: 32 x HCLK 11: 40 x HCLK
7	RO	**Reserved (0)**
6	RW	**CE2 4B address Auto-Read command selection**
5	RW	**CE1 4B address Auto-Read command selection**
4	RW	**CE0 4B address Auto-Read command selection** 0: use 0x03 command 1: use 0x13 command
3	RO	**Reserved (0)**
2	RW	**CE2 SPI address mode selection**
1	RW	**CE1 SPI address mode selection**
0	RW	**CE0 SPI address mode selection** 0: (3B)3 bytes (smaller than or equal to 16MB) 1: (4B)4 bytes (larger than 16MB) CE0 selection would be toggled for each 2 seconds by address detection FMC_WDT1.

to next page

from previous page

Note :
If OTP trap_en_bspiadrswap = 0 and OTP trap_bspi_size [2:0] != 0, then bit0 and bit4 are determined by OTP trap_bspi_size [2:0] and not changable.
When defined size $\leq$ 16MB, then bit0/bit4 = 0, otherwise bit0/bit4 = 1.

Offset: 08h		**FMC08: Interrupt Control and Status Register**	**Init = 0x0**
Bit	**R/W**	**Description**	
31:14	RO	**Reserved (0)**	
13	RO	**DMA Buffer Mode FIFO Full Status** 0: FIFO is non-full 1: FIFO is full	
12	RO	**DMA Buffer Mode FIFO Empty Status** 0: FIFO is non-empty 1: FIFO is empty	
11	RO	**DMA Status** 0: Busy when DMA was enabled, or Idle when DMA was disabled. 1: DMA Finish Disable DMA will also clear this bit.	
10	RW	**SPI Command Abort Status** Command abort conditions as below: – User-Mode read/write command when command or address filter are enabled. – Normal-Read with un-supported command when command filter is enabled. – Normal-Write with un-supported command when command filter is enabled. – Normal-Write to protected address when write disabled or address filter is enabled. This bit is write ’1’ cleared.	
9	RW	**SPI Write Address Protected Status** This status indicates a write command is written to the write protected address range. The write protected address is defined by the ”Default Write Type” and ”Write Address Filter” registers. This bit is write ’1’ cleared.	
8:5	RO	**Reserved (0)**	
4	RW	**DMA Buffer Mode FIFO Full/Empty Interrupt Enable**	
3	RW	**DMA Interrupt Enable**	
2	RW	**SPI Command Abort Interrupt Enable**	
1	RW	**SPI Write Address Protected Interrupt Enable** 0: Disable 1: Enable	
0	RO	**Reserved (0)**	

Offset: 0Ch		**FMC0C: Command Control Register**	**Init = 0x0**
Bit	**R/W**	**Description**	
31:8	RO	**Reserved (0)**	

to next page

from previous page

7:4	RW	**SPI Address Byte lane disable** xxx1: disable address byte 0 xx1x: disable address byte 1 x1xx: disable address byte 2 1xxx: disable address byte 3 (only valid for 4-byte address mode) This register is used to constraint the address byte issued. Useful for special command that do not require full address bytes. Only applied to non-User-Mode command. When all address and data byte lane are disabled, only command field will be issued.
3:0	RW	**Data Byte lane disable** xxx1: disable data byte 0 xx1x: disable data byte 1 x1xx: disable data byte 2 1xxx: disable data byte 3 This register is used to constrain the command data size, the same as byte enable. Useful for byte specific access with a 4-byte access command. Applied to all flash types.

Offset: 10h	**FMC10: SPI CE0 Control Register**	**Init = 0x0000_0400**
Offset: 14h	**FMC14: SPI CE1 Control Register**	**Init = 0x0000_0400**
Offset: 18h	**FMC18: SPI CE2 Control Register**	**Init = 0x0000_0400**

Bit	Attr.	Description
31:28	RW	**IO Mode** 0000: single bit. 0010: dual bit read/write, data cycle only. 0011: dual bit read/write, including address and dummy byte cycle. 0100: quad bit read/write, data cycle only. 0101: quad bit read/write, including address and dummy byte cycle. 1xxx: QPI mode, quad bit on command/address/data cycles. others: reserved The IO mode also apply to User-Mode, and User mode only has data cycles, no command and address/dummy cycles.
27:24	RW	**SPI base clock selection** 0000: baseclk = 0 * HCLK 0001: baseclk = 16 * HCLK 0010: baseclk = 32 * HCLK 0011: baseclk = 48 * HCLK .... 1111: baseclk = 240 * HCLK
23:16	RW	**SPI Command** The content of this register is used for Normal-Read or Normal-Write CMD phase.
15	RW	**Dummy cycle command output** 0: dummy cycle no command output 1: first dummy cycle has command output
14	RW	**Dummy cycles before data for Normal-Read command (high bits)**
13	RW	**Reserved**

to next page

from previous page

12	RW	**Disable SPI flash read/write command merge** 0: Enable 1: Disable (with performance penalty) Set this bit will disable the SPI controller to merge continuous address read and write. By default, continuous address read and write will be merged to reduce the command overhead while read or write commands continuously occur within 256 HCLK clocks.
11:8	RW	**SPI clock frequency selection (t-CK)** 0000: baseclk + (16 * HCLK) 0001: baseclk + (14 * HCLK) 0010: baseclk + (12 * HCLK) 0011: baseclk + (10 * HCLK) 0100: baseclk + (8 * HCLK)(default) 0101: baseclk + (6 * HCLK) 0110: baseclk + (4 * HCLK) 0111: baseclk + (2 * HCLK) 1000: baseclk + (15 * HCLK) 1001: baseclk + (13 * HCLK) 1010: baseclk + (11 * HCLK) 1011: baseclk + (9 * HCLK) 1100: baseclk + (7 * HCLK) 1101: baseclk + (5 * HCLK) 1110: baseclk + (3 * HCLK) 1111: baseclk + (1 * HCLK) (only valid for baseclk selection not equal to 0)
7:6	RW	**Dummy cycles before data for Normal-Read command (low bits)** bit[14,7:6] = 000: 0 Byte (default) 001: 1 Byte 010: 2 Byte —- 111: 7 Byte The dummy cycle is affected by the setting of IO mode.
5	RW	**MSB/LSB first control** 0: MSB First (default for boot code) 1: LSB First
4	RW	**Reserved**
3	RW	**Reserved** ~~Enable dual data input mode~~ Insteaded by using bit[30:28] = ”010” mode.
2	RW	**CE# Stop Active Control** Set this bit to 1 will terminate the CE# active state immediately. When in User-Mode, SPI command cycle will be activated (CE# low) until set this bit to 1. Thats to say setting this bit to 1 will stop activating SPI interface after Read/Write operation finished or immediately if no Read/Write operation is in progress. But when different CE command is entered, SPI interface will be deactivated immediately.

to next page

from previous page

1:0	RW	**Command Mode** 00: Auto-Read (0x03/0x13 + Address + Read data [1/2/3/4 bytes]) 01: Normal-Read (CMD + Address + Read data [1/2/3/4 bytes]) 10: Normal-Write (CMD + Address + Write data [1/2/3/4 bytes]) 11: User-Mode (Read/write data [1/2/3/4 bytes]) At User-Mode, address only used for decoding CE, all address decoded in the same CE address range are valid, and data will be read/write from/to the SPI flash in the order of LSB byte first. This mode provides a flexible programming method for specific command type other than Normal-Read/Write command supported. CMD = 1 byte of data from bit[23:16] of this register Address = 3 or 4 bytes from AHB address bus and output in the order from MSB byte to LSB byte Read/write data = 1~4 bytes data from AHB data bus and output in the order from LSB byte to MSB byte.

Offset: 30h **FMC30: CE0 Address Decoding Range Register** **Init = 0x07F0_0000**
Offset: 34h **FMC34: CE1 Address Decoding Range Register** **Init = 0x0000_0000**
Offset: 38h **FMC38: CE2 Address Decoding Range Register** **Init = 0x0000_0000**

Bit	Attr.	Description
31:16	RW	**End address A[31:16]** Here defines the CEx upper bound address limit with the unit of 1MB. Only A[27:20] are used for decoding.

to next page

from previous page

15:0	RW	**Start address A[31:16]** Here defines the CEx lower bound address limit with the unit of 1MB. Only A[27:20] are used for decoding. The Start address of CE0 is fixed at 0x0 and not changeable. The valid address range for each CE is: StartAdr $\leq$ CEx $\leq$ EndAdr. When configure Start address = End address, then the decoding is disabled. The valid flash decoding address is as blelow, and the total decoding space is 256MB: 0x0000_0000 - 0x0FFF_FFFF (decoded to CE0 only) 0x2000_0000 - 0x2FFF_FFFF The default address segments for each CE listed as below: CE0 = 0x0000_0000 - 0x07FF_FFFF (128MB, when OTP trap_bspi_size [2:0] = 0) CE0 = 0x2000_0000 - 0x27FF_FFFF (128MB, when OTP trap_bspi_size [2:0] = 0) CE1 = disabled CE2 = disabled When OTP trap_bspi_size [2:0] $\neq$ 0, the default address segment for CE0 is determined by OTP trap_bspi_size [2:0]. When OTP trap_bspi_size [2:0] $\neq$ 0 and OTP trap_bspi_abrmode = 0, the default address segment for CE1 is determined by OTP trap_bspi_size [2:0]. The address segments setting of 3 chips select must not be overlapped. It will cause error when overlapped. There is a rule must be followed in defining the Start address. For example: 0x20000000, valid address mask = 0x0FFFFFF, can address 0 $\sim$ 256MB flash chip 0x24000000, valid address mask = 0x03FFFFF, can address 0 $\sim$ 64MB flash chip 0x22000000, valid address mask = 0x01FFFFF, can address 0 $\sim$ 32MB flash chip 0x21000000, valid address mask = 0x00FFFFF, can address 0 $\sim$ 16MB flash chip 0x20800000, valid address mask = 0x007FFFF, can address 0 $\sim$ 8MB flash chip

Offset: 50h **FMC50: Auto Soft-Reset Command Control** **Init = 0**

Bit	R/W	Description
31:16	RO	**Reserved (0)**
15:8	RO	**Latest read SPI status register**
7:3	RO	**Reserved (0)**
2	RO	**QPI mode for reset command** This bit is copied from FMC10[31], and cleared after reset command.
1	RW	**Enable wait SPI WIP idle** Set this bit to '1' and after watchdog reset, hardware will check the SPI status register until WIP bit idle, then CPU can start to access the code.
0	RW	**Enable generate soft-reset command** Set this bit to '1' will enable hardware to issue soft-reset command (0x66-0x99) to boot CE0 or CE1 automatically after watchdog reset. This operation can reset the volatile bits of SPI flash to power on default value. The procedure for soft-reset operation as below: watchdog reset $\rightarrow$ 0x05 wait WIP idle $\rightarrow$ 0x66 $\rightarrow$ 0x99 $\rightarrow$ 0x05 wait WIP idle $\rightarrow$ done

to next page

from previous page

Note :
1. Soft-reset can be used to restore the SPI flash state into default, such as QPI mode, address mode, and multi-die selection. Also it can wait until flash idle if firmware issued an erase or program command before watchdog.
2. Wait WIP idle can avoid CPU access to a busy flash that don't responsd for read command. But it can not work if flash work at QPI or multi-die mode before watchdog reset.

Offset: 54h **FMC54: SPI Dummy Cycle Data Register** **Init = 0**

Bit	R/W	Description
31:8	RO	**Reserved (0)**
7:0	RW	**SPI dummy cycle output data** This register will be outputed as data at the head of dummy cycle. Shared by all chip selects.

Offset: 60h **FMC60: FMC_WDT1 Control/Status Register for Address Mode Detection** **Init = 0**

Bit	R/W	Description
31:12	RO	**Reserved (0)**
11:8	RO	**Watchdog active event counter** The counter value can be cleared by writing bit[31:24] = 0xEA.
7:1	RO	**Reserved (0)**
0	RW	**Enable watchdog**

Note :
The expire time for FMC_WDT1 is hardware fixed on 2 seconds and can not be changed.

Offset: 64h **FMC64: FMC_WDT2 Control/Status Register for Alternate Boot** **Init = 0**

Bit	R/W	Description
31:16	RO	**Reserved (0)**
15:8	RO	**Watchdog active event counter** The counter value can be cleared by writing bit[31:24] = 0xEA.
7	RO	**Reserved (0)**
6	RO	**Alternate Boot Mode that defined by OTP** 0: 2 chips mode 1: 1 chip mode
5	RO	**Single chip boot mode source select indicator**
4	RW	**Boot flash source select indicator** 0: boot from primary source 1: boot from alternate source This bit can be set to '1' only by FMC_WDT2 or external ABR pin FWSPIABR (OTP trap_en_bspi_auxpin = 1), and cleared to '0' by writing bit[23:16] = 0xEA.
3:1	RO	**Reserved (0)**
0	RW	**Enable watchdog**

Offset: 68h **FMC68: FMC_WDT2 Timer Reload Value Register** **Init = 0xE0**

Bit	R/W	Description
31:16	RO	**Counter value status**
15:13	RO	**Reserved (0)**

to next page

from previous page

12:0	RW	**Reload value of expire time** The time unit is 0.1 second. Default set at 22 seconds.

Offset: 6Ch **FMC6C: FMC_WDT2 Timer Restart Register** **Init = 0**

Bit	R/W	Description
31:16	RO	**Reserved (0)**
15:0	WT	**Restart register** Write 0x4755 value to load the reload value into watchdog counter.

Offset: 7Ch **FMC7C: DMA Buffer Mode Length Register** **Init = 0**

Bit	R/W	Description
6:0	RW	**FIFO Length Status** Actual data size = FIFO length * 4 – This register reflect the current FIFO length status when DMA buffer mode is on-going. So firmware can read this status to get current FIFO condition for buffer mode data push/pop indication. Value = 0x40 is full condition Value = 0x00 is empty condition – The value can be updated by firmware when DMA is in disabled state, so firmware can preload a value for TX transfer. – After DMA is enabled, this register is read only. And firmware can write(TX)/read(RX) FIFO data port FMC200 to update the length counter.

Offset: 80h **FMC80: DMA Control/Status Register** **Init = 0**

Bit	R/W	Description
31:20	RO	**Reserved (0)**
19:16	RW	**SPI clock frequency setting** Calibration mode use only. The value definition is the same as CEx Control Register bit[11:8] and with bit[27:24] = 0.
15:8	RW	**SPI read data input delay cycle setting** Calibration mode use only. The value definition is the same as FMC94.
7:5	RO	**Reserved (0)**
4	RW	**DMA Buffer Mode** 0: Memory mode, transfer to/from DRAM/SRAM. 1: Buffer mode, transfer to/from DMA buffer. The DMA buffer works as FIFO mode for data transfer, and DMA will stop temporarily when FIFO is empty or full.
3	RW	**Calibration Mode** 0: Normal mode 1: Calibration mode. At this mode, SPI clock rate and read delay cycle parameters setting will be replaced by bit[15:4]. Calibration mode should be enabled in companion with bit[2]=1.
2	RW	**CheckSum Calculation Only** 0: Normal DMA operation 1: Checksum accumulation only This bit is valid only when the DMA direction = 0. And transfer length should be 4 bytes boundary.

to next page

from previous page

1	RW	**DMA Direction** 0: Read flash, move from flash to external memory 1: Write flash, move from external memory to flash
0	RW	**DMA Enable** 0: Disable 1: Enable DMA operation
Note : After each DMA cycle completion, it should clear FMC80[0] = 0 to reset DMA status and to start new DMA cycle. Refer 14.4 for the DMA programming guidance.		

Offset: 84h **FMC84: DMA Flash Side Address** **Init = X**

Bit	R/W	Description
31:28	RO	**Reserved**
27:2	RW	**Flash side start address** For DMA Read flash, this is the source address. For DMA Write flash, this is the destination address. DMA can only execute on 4 bytes boundary. The valid address range is 0x2000_0000 - 0x2FFF_FFFF. When read, it shows the current working address.
1:0	RO	**Reserved**

Offset: 88h **FMC88: DMA DRAM/SRAM Side Address** **Init = X**

Bit	R/W	Description
31:2	RW	**DRAM side start address (Memory mode only)** For DMA Read flash, this is the destination address. For DMA Write flash, this is the source address. DMA can only execute on 4 bytes boundary. When read, it shows the current working address.
1:0	RO	**Reserved**

Offset: 8Ch **FMC8C: DMA Length Register** **Init = X**

Bit	R/W	Description
31:25	RO	**Reserved (0)**
24:0	RW	**DMA Length** From 1 bytes to 32MB 0: 1 byte 0x1FFFFFF: 32M bytes When DMA is on-going, the length register will count down, and reach 0 when DMA finished.

Offset: 90h	FMC90: CheckSum Calculation Result	Init = 0

Bit	R/W	Description
31:0	RO	**CheckSum Calculation Result** Accumulate the flash read data checksum result, 32bits base. This register will be reset when DMA disabled.

Offset: 94h	FMC94: CE0 SPI Flash Read Timing Compensation	Init = 0
Offset: 98h	FMC98: CE1 SPI Flash Read Timing Compensation	Init = 0
Offset: 9Ch	FMC9C: CE2 SPI Flash Read Timing Compensation	Init = 0

Bit	Attr.	Description
31:24	RW	**SPICLK = HCLK/5, delay cycle for data input latch point**
23:16	RW	**SPICLK = HCLK/4, delay cycle for data input latch point**
15:8	RW	**SPICLK = HCLK/3, delay cycle for data input latch point**
7:0	RW	**SPICLK = HCLK/2, delay cycle for data input latch point** Bit[7:4]: DI input delay selection, each step is about 0.5ns. Bit[3]: 0: disable DI input delay 1: enable DI input delay Bit[2:0]: 0: no delay 1: delay 1 HCLK 2: delay 2 HCLK 3: delay 3 HCLK 4: delay 4 HCLK others: delay 5 HCLK
Note : This read timing applied to all CE's SPI flash. So it may need to change the timing for different CE SPI flash.		

Offset: A0h	FMCA0: Command Filter Control Register	Init = 0

Bit	R/W	Description
31:3	RO	**Reserved**
2	RW	**Enable read/write command filter for CE2**
1	RW	**Enable read/write command filter for CE1**
0	RW	**Enable read/write command filter for CE0** 0: disable filter, all commands are allowed. 1: enable filter, only commands defined in FMC100 ~ FMC17C are allowed. When OTP trap_en_bspi_auxpin = 1 and external FWSPIWP# pin is active, CE0/(and CE1 if OTP trap_bspi_abrmode =0) filter will be always enabled. If FWSPIWP# is not active, then CE0/CE1 filter can be controlled by software. When command filter is enabled, below limitation would be applied: 1. User-Mode command (Command Mode = "11") would be disabled. 2. Normal-Write operation only can be applied within the address range 0x20000000 ~ 0x2FFFFFFF (256MB).

Offset: A4h	FMCA4: Write Address Filter Control Register	Init = 0

Bit	R/W	Description
31:16	RO	**Reserved**
15:14	RW	**Mode of write address filter #8**
13:12	RW	**Mode of write address filter #7**
11:10	RW	**Mode of write address filter #6**
9:8	RW	**Mode of write address filter #5**
7:6	RW	**Mode of write address filter #4**
5:4	RW	**Mode of write address filter #3**
3:2	RW	**Mode of write address filter #2**
1:0	RW	**Mode of write address filter #1** 0x: NOP 10: write enabled 11: write disabled When OTP trap_en_bspi_auxpin = 1, FWSPIWP# = 0 and CRTM is enabled, filter #1 & #2 will be forced to "11" mode. And address range at FMCB0 & FMCB4 are determined by OTP trap_bspi_size and OTP trap_bspi_crtmsize . When FWSPIWP# = 1 or CRTM is disabled, filter #1 & #2 can be set by firmware.

Note :
Address filters take effect under 2 conditions:

1. When command filter is enabled, then write is allowed when command matches AQCD and address pass address filters.
2. When command filter is disabled, then write is allowed when address pass address filters or configured in User-Mode.

The write filter mode is defined by 2 registers: FMC00 bit[18:16] and FMCA4.
Protection mode as below:

1. If FMC00 is write disabled, the write can be enabled by FMCA4.
2. If FMC00 is write enabled, then write can be disabled by FMCA4.

The address segments defined on FMCB0 ~ FMCCC controls the address range for write allowed or disallowed.

Offset: A8h	FMCA8: Register Lock Control Register (SRST#)	Init = 0

Bit	R/W	Description
31:11	RO	**Reserved**
10	RW	**Lock FMCA4 and Address Filter from write until reset by SRST#**
9	RW	**Lock FMCA0 and FQCD/AQCD from write until reset by SRST#**
8	RW	**Lock FMC94/FMC98/FMC9C from write until reset by SRST#**
7	RW	**Lock FMC38 from write until reset by SRST#**
6	RW	**Lock FMC34 from write until reset by SRST#**
5	RW	**Lock FMC30 from write until reset by SRST#**
4	RW	**Lock FMC18 from write until reset by SRST#**
3	RW	**Lock FMC14 from write until reset by SRST#**
2	RW	**Lock FMC10 from write until reset by SRST#**
1	RW	**Lock FMC04 from write until reset by SRST#**
0	RW	**Lock FMC00 from write until reset by SRST#**

Note :
This register is write '1' only, and reset to '0' only when SRST# active.

Offset: ACh	FMCAC: Register Lock Control Register (Watchdog)	Init = 0

Bit	R/W	Description
31:11	RO	**Reserved**
10	RW	**Lock FMCA4 and Address Filter from write until reset by watchdog**
9	RW	**Lock FMCA0 and FQCD/AQCD from write until reset by watchdog**
8	RW	**Lock FMC94/FMC98/FMC9C from write until reset by watchdog**
7	RW	**Lock FMC38 from write until reset by watchdog**
6	RW	**Lock FMC34 from write until reset by watchdog**
5	RW	**Lock FMC30 from write until reset by watchdog**
4	RW	**Lock FMC18 from write until reset by watchdog**
3	RW	**Lock FMC14 from write until reset by watchdog**
2	RW	**Lock FMC10 from write until reset by watchdog**
1	RW	**Lock FMC04 from write until reset by watchdog**
0	RW	**Lock FMC00 from write until reset by watchdog**

Note :
This register is write '1' only, and reset to '0' only when watchdog reset.

Offset: B0h	FMCB0: Write Address Filter Register #1	Init = 0
Offset: B4h	FMCB4: Write Address Filter Register #2	Init = 0
Offset: B8h	FMCB8: Write Address Filter Register #3	Init = 0
Offset: BCh	FMCBC: Write Address Filter Register #4	Init = 0
Offset: C0h	FMCC0: Write Address Filter Register #5	Init = 0
Offset: C4h	FMCC4: Write Address Filter Register #6	Init = 0
Offset: C8h	FMCC8: Write Address Filter Register #7	Init = 0
Offset: CCh	FMCCC: Write Address Filter Register #8	Init = 0

Bit	Attr.	Description
31:16	RW	**Segments Upper bound address bit[27:12]**
15:0	RW	**Segments Lower bound address bit[27:12]** The address range is defined between 0x20000000 ~ 0x2FFFFFFF. And active to CE0, CE1, or CE2 based on the address range defined for the CE.

Note :
The segments address range is defined as below:
lower_bound $\leq$ address $\leq$ upper_bound

Offset: 100h	**FQCD00: Fully Qualified Command**	**Init = 0x80000003**
Offset: 104h	**FQCD01: Fully Qualified Command**	**Init = 0x8000000B**
Offset: 108h	**FQCD02: Fully Qualified Command**	**Init = 0x8000009F**
Offset: 10Ch	**FQCD03: Fully Qualified Command**	**Init = 0x80000005**
Offset: 110h	**FQCD04: Fully Qualified Command**	**Init = 0x80000001**
Offset: 114h	**FQCD05: Fully Qualified Command**	**Init = 0x80000006**
Offset: 118h	**FQCD06: Fully Qualified Command**	**Init = 0x80000004**
Offset: 11Ch	**FQCD07: Fully Qualified Command**	**Init = 0x00000000**
Offset: 120h-14Ch	**FQCD08-FQCD19: Fully Qualified Command**	**Init = 0x00000000**

Bit	Attr.	Description
31	RW	**Enable Entry**
30:16	RO	**Reserved**
15:8	RW	**Fully qualified Command**
7:0	RW	**Fully qualified Command**

Note :
If OTP trap_en_bspi_auxpin = 1, then the write of these FQCD registers is controlled by external pin FWSPIWP# .
When FWSPIWP# = 0, then all these registers are Read Only.
When FWSPIWP# = 1, then all these registers are Read/Write.

When FQCD register is enabled by bit31 but not defined, please set the command as 0x0.
Below commands would be used by controller for auto command generation:
0x03, 0x13, 0x05, 0x66, 0x99

Offset: 150h	**AQCD00: Address Qualified Command**	**Init = 0x80000002**
Offset: 154h	**AQCD01: Address Qualified Command**	**Init = 0x800000D8**
Offset: 158h	**AQCD02: Address Qualified Command**	**Init = 0x00000000**
Offset: 15Ch	**AQCD03: Address Qualified Command**	**Init = 0x00000000**
Offset: 160h	**AQCD04: Address Qualified Command**	**Init = 0x00000000**
Offset: 164h	**AQCD05: Address Qualified Command**	**Init = 0x00000000**
Offset: 168h	**AQCD06: Address Qualified Command**	**Init = 0x00000000**
Offset: 16Ch	**AQCD07: Address Qualified Command**	**Init = 0x00000000**
Offset: 170h	**AQCD08: Address Qualified Command**	**Init = 0x00000000**
Offset: 174h	**AQCD09: Address Qualified Command**	**Init = 0x00000000**
Offset: 178h	**AQCD10: Address Qualified Command**	**Init = 0x00000000**
Offset: 17Ch	**AQCD11: Address Qualified Command**	**Init = 0x00000000**

Bit	Attr.	Description
31	RW	**Enable Entry**
30:16	RO	**Reserved**
15:8	RW	**4-Byte Addressing Command** Check address bits A[27:12] to avoid address aliasing, and address byte lane configuration should be 4 bytes.
7:0	RW	**3-Byte Addressing Command** Check address bits A[27:12] to avoid address aliasing, and address byte lane configuration should be 3 bytes.

to next page

from previous page

Note :
If OTP trap_en_bspi_auxpin = 1, then the write of these FQCD registers is controlled by external pin FWSPIWP# .
When FWSPIWP# = 0, then all these registers are Read Only.
When FWSPIWP# = 1, then all these registers are Read/Write.

When AQCD register is enabled by bit31 but not defined, please set the command as 0x0.
For commands listed in these registers can be executed correctly and avoid address aliasing, the address size should be 3 or 4 bytes that matching the command definition, and the address should pass the address filter.

Offset: 200h	FMC200: DMA FIFO Mode Data Port Register	Init = X

Bit	R/W	Description
31:0	RW	**FIFO data read/write port** 1. This data port work when DMA buffer mode and DMA are both enabled. 2. When in TX mode transfer and FIFO is not full, firmware can write this port to push data into FIFO for TX transfer. And the length counter will increment by 1 for each write. Write would be ignored if FIFO is full. 3. When in RX mode transfer and FIFO is not empty, firmware can read this port to pop data from FIFO to get RX data. And the length counter will decrement by 1 for each read. Read return dummy data if FIFO is empty. 4. Read/Write of this register is 32 bits format.

Offset: 200h-2FFh	FMCBUF: DMA Buffer Data Register	Init = X

Bit	R/W	Description
31:0	RW	**DMA Buffer read/write port** 1. This DMA buffer can be accessed by CPU when DMA is disabled. 2. DMA is 256 bytes, ranging from address offset 0x200 to 0x2FF. 3. For DMA buffer TX mode transfer, firmware can write this buffer to preload data into FIFO for TX transfer. 4. Read/Write of this buffer is 32 bits format.

14.4 Programming Guide

14.4.1 DMA Operation - Memory Mode

- DMA operation can not across the CE address segment boundary.
- DMA DRAM/SRAM side address can not over the maximum supported DRAM size. It will wrap back if overflow.
- Start DMA procedure:
  1. Set FMC84
  2. Set FMC88
  3. Set FMC8C
  4. Set FMC80[1:0]
  5. Wait DMA done
  6. Clear FMC80[0] = 0 to reset DMA and start new DMA cycle.
  7. If more DMA cycles to run, goto step 1, otherwise quit.
- Reset or terminate DMA operation:
  1. Set FMC80[0] = 0
  2. DMA should be reset once at each DMA operation end.

14.4.2 DMA Operation - Buffer Mode

- Before enable DMA, it can preload the TX data into DMA buffer and set the preload length in FMC7C (4-bytes unit).

- After DMA is enabled and DMA is on-going, the DMA buffer works as FIFO mode for firmware to push/pop TX/RX data at the same data port address.
- Start DMA procedure:
  1. For TX, it can preload data into buffer
  2. Set FMC7C
  3. Set FMC84
  4. Set FMC8C
  5. Set FMC80[1:0] and bit[4]=1
  6. It can check FMC7C status to push/pop more data for TX/RX when DMA is on-going.
  7. Wait DMA done
  8. Clear FMC80[0] = 0 to reset DMA and start new DMA cycle.
  9. If more DMA cycles to run, goto step 1, otherwise quit.
- Reset or terminate DMA operation:
  1. Set FMC80[0] = 0
  2. DMA should be reset once at each DMA operation end.

14.4.3 DMA CheckSum Calculation Mode

- CheckSum calculation is only applied to flash read mode.
- Command procedure:
  1. Set FMC84
  2. Set FMC8C
  3. Set FMC80[1:0] = ”01”
  4. FMC80[2] is set when only checksum calculation is necessary, no data movement
  5. FMC80[3] is set when wants to try different clock rate and input compensation delay for DMA cycle
  6. Wait DMA done

14.4.4 Address Continuity Rule

For non-”User-Mode” command to access the SPI flash, hardware will check the address continuity to keep contiguous read or write commands to be within the same SPI CE# active cycle.

The continuity rule designed by hardware was based on the full double word basis, that is all 4 byte lanes on a 32bits command. Any below condition will terminate the address continuity and inactive the SPI CS# before new command.

- Read or write to flash registers.
- Non-4-byte lane access to flash will stop the continuity of **NEXT** flash access command.
- LSB byte lane disabled will stop the continuity of **CURRENT** flash access.
- Crossover the 4K-bytes address boundary.
- Read-to-write or write-to-read operation switching.
- DMA cycle beginning and ending.

15 SPI1/SPI2 Flash Controller (SPI1/SPI2)

15.1 Overview

15.1.1 Register List

Base address of SPI Controller #1 = 0x1E63_0000
Base address of SPI Controller #2 = 0x1E63_1000
Base address of SPI Memory #1 = 0x3000_0000
Base address of SPI Memory #2 = 0x5000_0000
Physical address = (Base address of SPI) + Offset

SPIR00: SPI Flash Configuration Register
SPIR04: CE Control Register
SPIR08: Interrupt Control and Status Register
SPIR0C: Command Control Register
SPIR10: CE0 Control Register
SPIR14: CE1 Control Register
SPIR18: CE2 Control Register (SPI2 only)
SPIR30: CE0 Address Decoding Range Register
SPIR34: CE1 Address Decoding Range Register
SPIR38: CE2 Address Decoding Range Register (SPI2 only)
SPIR54: SPI Dummy Cycle Data Register
SPIR64: Alternate Boot Control/Status (SPI1 only)
SPIR6C: Host Direct Access Commands #4 (SPI1 only)
SPIR70: Host Direct Access Commands #1 (SPI1 only)
SPIR74: Host Direct Access Commands #2 (SPI1 only)
SPIR78: Host Direct Access Commands #3 (SPI1 only)
SPIR7C: DMA Buffer Mode Length Register
SPIR80: DMA Control/Status Register
SPIR84: DMA Flash Side Address
SPIR88: DMA DRAM Side Address
SPIR8C: DMA Length Register
SPIR90: CheckSum Calculation Result
SPIR94: CE0 SPI Flash Read Timing Compensation
SPIR98: CE1 SPI Flash Read Timing Compensation
SPIR9C: CE2 SPI Flash Read Timing Compensation (SPI2 only)
SPIRA0: Command Filter Control Register
SPIRA4: Write Address Filter Control Register
SPIRA8: Register Lock Control Register (SRST#)
SPIRAC: Register Lock Control Register (Watchdog)
SPIRB0: Write Address Filter Register #1
SPIRB4: Write Address Filter Register #2
SPIRB8: Write Address Filter Register #3
SPIRBC: Write Address Filter Register #4
SPIRC0: Write Address Filter Register #5
SPIRC4: Write Address Filter Register #6
SPIR100:FQCD00: Fully Qualified Command
SPIR104:FQCD01: Fully Qualified Command
SPIR108:FQCD02: Fully Qualified Command
SPIR10C:FQCD03: Fully Qualified Command
SPIR110:FQCD04: Fully Qualified Command
SPIR114:FQCD05: Fully Qualified Command
SPIR118:FQCD06: Fully Qualified Command
SPIR11C:FQCD07: Fully Qualified Command
SPIR120:FQCD08: Fully Qualified Command

SPIR124:FQCD09: Fully Qualified Command
SPIR128:FQCD10: Fully Qualified Command
SPIR12C:FQCD11: Fully Qualified Command
SPIR130:FQCD12: Fully Qualified Command
SPIR134:FQCD13: Fully Qualified Command
SPIR138:FQCD14: Fully Qualified Command
SPIR13C:FQCD15: Fully Qualified Command
SPIR140:FQCD16: Fully Qualified Command
SPIR144:FQCD17: Fully Qualified Command
SPIR148:FQCD18: Fully Qualified Command
SPIR14C:FQCD19: Fully Qualified Command
SPIR150:AQCD00: Address Qualified Command
SPIR154:AQCD01: Address Qualified Command
SPIR158:AQCD02: Address Qualified Command
SPIR15C:AQCD03: Address Qualified Command
SPIR160:AQCD04: Address Qualified Command
SPIR164:AQCD05: Address Qualified Command
SPIR168:AQCD06: Address Qualified Command
SPIR16C:AQCD07: Address Qualified Command
SPIR170:AQCD08: Address Qualified Command
SPIR174:AQCD09: Address Qualified Command
SPIR178:AQCD10: Address Qualified Command
SPIR17C:AQCD11: Address Qualified Command

SPIR200:DMA FIFO Mode Data Port Register
SPIRBUF:Offset 0x200 - 0x2FF: DMA Buffer R/W Port

15.1.2 Changed items compared to AST2500

- Change the SPI clock divider range from (HCLK/1 - HCLK/16) to (HCLK/2 - HCLK/256), since HCLK clock rate is higher than 200MHz.
- Re-define the CE minimum inactive timing selection.
- Re-define the CE address decoding range register, increase the resolution to 1MB.
- Re-define the command filters, supports 40 non-address check commands and 24 address check commands.
- Add 0x13 read command option for 4B address Auto-Read command mode.
- Add programmable fine delay tuning on data input path for high speed read timing compensation.
- Add the address filters to 6 sets.
- Add another alternate boot option to support single flash ABR.
- Add support Quad-IO and QPI mode.
- Add Host Direct Access Port on SPI1.
- Add DMA controller for bulk data transfer between SPI1/SPI2 flash and DRAM/SRAM.
- Add support auxiliary control pins for SPI1 ABR/WriteProtect functions, which is enabled by strap bit OTP trap_en_hspi_auxpin .

15.1.3 OTP configuration for Host SPI function

- OTP trap_en_hspi_auxpin : enable auxiliary control pins for ABR and Write Protect control.
- OTP trap_en_hspiabr : enable host SPI ABR function.
- OTP trap_en_hspiabr_selpin : enable external pin SPI1ABR for ABR boot code selection.

- OTP trap_hspi_abrmode : select host SPI ABR function mode, 1 chip or 2 chips.
  - 0 : 2 chips ABR mode
  - 1 : 1 chips ABR mode
  - SPIR64[6] is determined by the ABR mode selection, not changable.
- OTP trap_hspi_size [2:0] : define the host SPI CE0 flash size.
  - Value definition:
    - 000 : no define size
    - 001 : 2MB
    - 010 : 4MB
    - 011 : 8MB
    - 100 : 16MB
    - 101 : 32MB
    - 110 : 64MB
    - 111 : 128MB
  - When OTP trap_hspi_size [2:0] $\neq$ 0, SPIR04 bit0 and bit4 is determined by SPI size after reset, not changable.
  - When OTP trap_hspi_size [2:0] $\neq$ 0, SPIR30[31:16] is determined by SPI size after reset, not changable.
  - When OTP trap_hspi_size [2:0] $\neq$ 0, SPIR34[31:00] is determined by SPI size after reset if OTP trap_hspi_abrmode = 0, not changable.
  - Host access to SPI CE0 address is clamped based on SPI size, not changable.
  - To support SPI size larger than 128MB, set OTP trap_hspi_size [2:0] = 0. Then firmware can program SPIR30 to configure the maximum direct fetch address space for CE0.
- trap_hspi_crtmsize [1:0] : define CRTM write protection size, the CRTM protect region is located at the top of code area.
  - CRTM only works when OTP trap_en_hspi_auxpin = 1, OTP trap_en_hspiabr = 1, and SPI1WP# is active
  - Value definition:
    - 00 : disable CRTM
    - 01 : 1MB
    - 10 : 2MB
    - 11 : 4MB
  - When CRTM is enabled, SPIRA4[3:0] = 0xF. And SPIRB0/SPIRB4 are determined by ABR mode, SPI size, and CRTM size, not changable.

15.1.4 Auxiliary control pins behavior for SPI1 (OTP trap_en_hspi_auxpin = 1)

- SPI1ABR : alternate boot source selection pin.
  - This pin function is enabled when OTP trap_en_hspi_auxpin = 1 or OTP trap_en_hspiabr_selpin = 1.
  - SPI1ABR = low state, force to boot from primary code.
  - SPI1ABR = high state, force to boot from alternate code.
  - SPIR64[4] is determined by external SPI1ABR pin state, not changable.
- HOST_CFGLOCK_N : host write protect control signal from SIO register SIORFF9[0].
- SPI1WP# : write protect control input pin.
- FWSPIWP# : write protect control input pin.

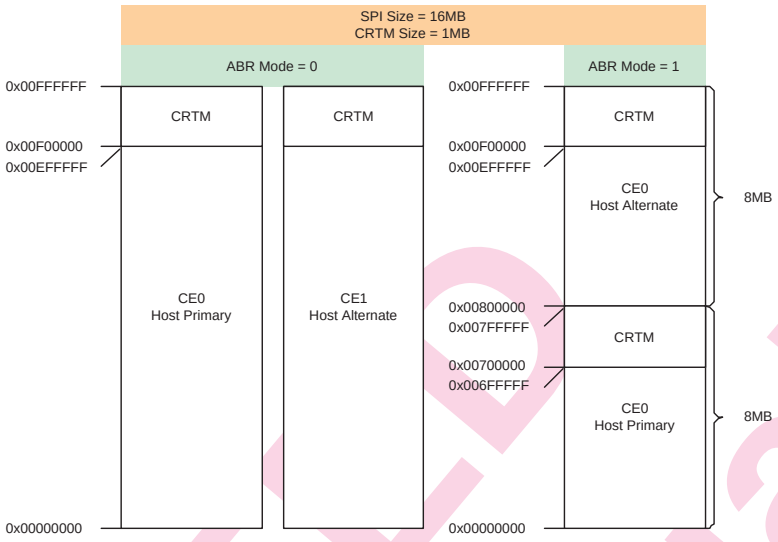


Figure 44: Host SPI CRTM Allocation

– Registers between SPIRA0 to SPIR17C are only writable by Host and ARM CPU, the access right matrix as below:

HOST_CFGLOCK_N	FWSPIWP#	SPI1WP#	Host	BMC
0	0	0	R	R
0	0	1	R/W	R
0	1	0	R	R
0	1	1	R/W	R
1	0	0	R	R
1	0	1	R/W	R
1	1	0	R	R/W
1	1	1	R/W	R/W

– Enable additional 2 data pins to support 4-bits IO mode.
– SPI transaction block for Host access is enabled when SPI1WP# = low state (SPIRA0[2]=1). FQCD and AQCD are write protected from host write. And host access to SPI commands are filtered by FQCD and AQCD.
– SPI transaction block for BMC access is enabled when either FWSPIWP# = low state or HOST_CFGLOCK_N = 0 state (SPIRA0[0]=1). FQCD and AQCD are write protected from BMC write. And BMC access to SPI commands are filtered by FQCD and AQCD.
– When external pin control of transaction block is not enabled, then software can control the transaction block through related registers.

15.1.5 SPI flash Selection requirement

– Minimum read speed $\geq$ 35MHz
– When OTP trap_hspi_size is defined and larger than 16MB, it should choose the SPI flash part that support **0x13** read command for CE0.

15.2 Features

15.2.1 SPI master controller features

– SPI1 (Host SPI) supports 2 chip select pins (CE0, CE1)

- SPI2 supports 3 chip select pins (CE0, CE1, CE2)
- Support maximum 256 MBytes direct access memory space for each chip select.
- Support maximum 256 MBytes decoding space for SPI1 and SPI2.
- Programmable clock speed, HCLK/2 ∼ HCLK/256.
- Support 1, 2, 4 bits and QPI SPI IO modes.
- Support 24bits and 32bits address mode.
- Support high clock rate (> 50MHz) read timing compensation function.
- Support DMA for bulk data transfer between SPI flash and DRAM/SRAM.

15.2.2 Alternate Boot (ABR) Code Selection for SPI1 (Host SPI)

- Alternate boot function is enabled by strap OTP trap_en_hspiabr .
- Alternate boot code source selection policy:
  - OTP trap_en_hspi_auxpin = 1 or OTP trap_en_hspiabr_selpin = 1, boot source is selected by external ABR pin SPI1ABR .
  - OTP trap_en_hspi_auxpin = 0 and OTP trap_en_hspiabr_selpin = 0, boot source is selected by register SPIR64[4].
- Supports 2 modes of alternate boot function:
  1. Use 2 SPI flash parts, CE0 and CE1. CE0 is primary boot source, and alternate boot from CE1.
  2. Use 1 SPI flash part, CE0. Uniformly split the flash space into 2 parts of same size. Upper partition is the default boot area, and alternate boot from lower partition.
     Flash size information is required for this boot mode.
- When use 2 flash mode and boot from CE1 flash, at this time the address mapping for CE0 and CE1 are swapped. To restore the address mapping, firmware should clear the boot select flag at SPIR64.bit[4].
- When use 1 flash mode and boot from lower partition, at this time the address mapping for lower and upper partition are swapped. To restore the address mapping, firmware should clear the boot select flag at SPIR64.bit[4].

15.2.3 DMA Function

- Support maximum 32 MByte data transfer length.
- Support 2 operation modes.
  1. Memory Mode: transfer data between DRAM/SRAM and SPI memory.
  2. Buffer Mode: transfer data between DMA buffer and SPI memory.
- Support flash data checksum calculation (32 bits data unit).
- Support 256 bytes DMA buffer. The DMA buffer can be accessed at offset range 0x200 - 0x2FF when DMA function is disabled.
- Support 4-byte boundary for SPI memory side address, and 4-byte boundary for DRAM/SRAM side address.
- Support 1-byte boundary for transfer length.
- The DMA controller is shared by SPI1 and SPI2 controllers.
  1. Before using the DMA, driver of SPI1 or SPI2 should get the right of using DMA controller, by writing SPIR80[31]=1 and waiting for SPIR80[30]=1.
  2. After finished using the DMA controller, driver should release the DMA owner for another SPI1/SPI2 driver to use the DMA.
- When work at DMA buffer mode:

* Firmware can preload TX data into DMA buffer and set preload length in SPIR7C before DMA is enabled.
* DMA buffer works as FIFO mode after DMA is enabled.
* When DMA is on-going, firmware can push/pop data at dedicated data port for more data of TX/RX.
* The total transfer length is defined in SPIR8C.
* DMA will stop temporarily and keep CE# active when DMA buffer is full or empty.
* Support FIFO full/empty interrupt.

15.3 Registers : Base Address = 0x1E63:x000

Offset: 00h	SPIR00: SPI Flash Configuration Register	Init = 0x0000000A

Bit	R/W	Description
31:19	RO	**Reserved (0)**
18	RW	**Enable CE2 default write type**
17	RW	**Enable CE1 default write type**
16	RW	**Enable CE0 default write type** 0: CEx is default at write disable mode 1: CEx is default at write enable mode The default write type is a global range control for a CE. This default write type in companion with the Write Address Filter register at SPIRA4 control the write protection behavior of a CE. Protection mode as below: 1. If SPIR00 is write disabled, the write can be enabled by SPIRA4. 2. If SPIR00 is write enabled, then write can be disabled by SPIRA4.
15:0	RO	**Reserved**

Offset: 04h	SPIR04: CE Control Register	Init = 0x00002A00

Bit	R/W	Description
31:14	RO	**Reserved (0)**
13:12	RW	**Select CE2 minimum inactive timing (tCSHigh)**
11:10	RW	**Select CE1 minimum inactive timing (tCSHigh)**
9:8	RW	**Select CE0 minimum inactive timing (tCSHigh)** 00: 16 x HCLK 01: 24 x HCLK 10: 32 x HCLK 11: 40 x HCLK
7	RO	**Reserved (0)**
6	RW	**CE2 4B address Auto-Read command selection**
5	RW	**CE1 4B address Auto-Read command selection**
4	RW	**CE0 4B address Auto-Read command selection** 0: use 0x03 command 1: use 0x13 command
3	RO	**Reserved (0)**
2	RW	**CE2 SPI address mode selection**
1	RW	**CE1 SPI address mode selection**
0	RW	**CE0 SPI address mode selection** 0: (3B)3 bytes (smaller than or equal to 16MB) 1: (4B)4 bytes (larger than 16MB)

Offset: 08h	SPIR08: Interrupt Control and Status Register	Init = 0x0

Bit	R/W	Description
31:14	RO	**Reserved (0)**
13	RO	**DMA Buffer Mode FIFO Full Status** 0: FIFO is non-full 1: FIFO is full

to next page

from previous page

12	RO	**DMA Buffer Mode FIFO Empty Status** 0: FIFO is non-empty 1: FIFO is empty
11	RO	**DMA Status** 0: Busy when DMA was enabled, or Idle when DMA was disabled. 1: DMA Finish Disable DMA will also clear this bit.
10	RW	**SPI Command Abort Status** Command abort conditions as below: – User-Mode read/write command when command or address filter are enabled. – Normal-Read with un-supported command when command filter is enabled. – Normal-Write with un-supported command when command filter is enabled. – Normal-Write to protected address when write disabled or address filter is enabled. This bit is write '1' cleared.
9	RW	**SPI Write Address Protected Status** This status indicates a write command is written to the write protected address range. The write protected address is defined by the "Default Write Type" and "Write Address Filter" registers. This bit is write '1' cleared.
8:5	RO	**Reserved (0)**
4	RW	**DMA Buffer Mode FIFO Full/Empty Interrupt Enable**
3	RW	**DMA Interrupt Enable**
2	RW	**SPI Command Abort Interrupt Enable**
1	RW	**SPI Write Address Protected Interrupt Enable** 0: Disable 1: Enable
0	RO	**Reserved (0)**

Offset: 0Ch **SPIR0C: Command Control Register** **Init = 0x0**

Bit	R/W	Description
31:8	RO	**Reserved (0)**
7:4	RW	**SPI Address Byte lane disable** xxx1: disable address byte 0 xx1x: disable address byte 1 x1xx: disable address byte 2 1xxx: disable address byte 3 (only valid for 4-byte address mode) This register is used to constraint the address byte issued. Useful for special command that do not require full address bytes. Only applied to non-User-Mode command. When all address and data byte lane are disabled, only command field will be issued.
3:0	RW	**Data Byte lane disable** xxx1: disable data byte 0 xx1x: disable data byte 1 x1xx: disable data byte 2 1xxx: disable data byte 3 This register is used to constrain the command data size, the same as byte enable. Useful for byte specific access with a 4-byte access command. Applied to all flash types.

Offset: 10h	SPIR10: CE0 Control Register	Init = 0x0000_0400
Offset: 14h	SPIR14: CE1 Control Register	Init = 0x0000_0400
Offset: 18h	SPIR14: CE2 Control Register (SPI2 only)	Init = 0x0000_0400

Bit	Attr.	Description
31:28	RW	**IO Mode** 0000: single bit. 0010: dual bit read/write, data cycle only. 0011: dual bit read/write, including address and dummy byte cycle. 0100: quad bit read/write, data cycle only. 0101: quad bit read/write, including address and dummy byte cycle. 1xxx: QPI mode, quad bit on command/address/data cycles. others: reserved The IO mode also apply to User-Mode, and User-mode only has data cycles, no command and address/dummy cycles.
27:24	RW	**SPI base clock selection** 0000: baseclk = 0 * t-HCLK 0001: baseclk = 16 * t-HCLK 0010: baseclk = 32 * t-HCLK 0011: baseclk = 48 * t-HCLK .... 1111: baseclk = 240 * t-HCLK
23:16	RW	**SPI Command** The content of this register is used as the data for Normal-Read or Normal-Write CMD phase.
15	RW	**Dummy cycle command output** 0: dummy cycle no command output 1: first dummy cycle has command output
14	RW	**Dummy cycles before data for Normal-Read command (high bits)**
13	RW	**Reserved**
12	RW	**Disable SPI flash read/write command merge** 0: Enable 1: Disable (with performance penalty) Set this bit will disable the SPI controller to merge continuous address read and write. By default, continuous address read and write will be merged to reduce the command overhead while read or write commands continuously occur within 256 HCLK clocks.
11:8	RW	**SPI clock frequency selection (t-CK)** 0000: baseclk + (16 * t-HCLK) 0001: baseclk + (14 * t-HCLK) 0010: baseclk + (12 * t-HCLK) 0011: baseclk + (10 * t-HCLK) 0100: baseclk + (8 * t-HCLK)(default) 0101: baseclk + (6 * t-HCLK) 0110: baseclk + (4 * t-HCLK) 0111: baseclk + (2 * t-HCLK) 1000: baseclk + (15 * t-HCLK) 1001: baseclk + (13 * t-HCLK) 1010: baseclk + (11 * t-HCLK) 1011: baseclk + (9 * t-HCLK) 1100: baseclk + (7 * t-HCLK) 1101: baseclk + (5 * t-HCLK) 1110: baseclk + (3 * t-HCLK) 1111: baseclk + (1 * t-HCLK) (only valid for baseclk selection not equal to 0)

to next page

from previous page

7:6	RW	**Dummy cycles before data for Normal-Read command (low bits)** bit[14,7:6] = 000: 0 Byte (default) 001: 1 Byte 010: 2 Byte —- 111: 7 Byte The dummy cycle is affected by the setting of IO mode.
5	RW	**MSB/LSB first control** 0: MSB First (default for boot code) 1: LSB First
4	RW	**Reserved**
3	RW	**Reserved** ~~Enable dual data input mode~~ Insteaded by using bit[30:28] = ”010” mode.
2	RW	**CE# Stop Active Control** Set this bit to 1 will terminate the CE# active state immediately. When in User-Mode, SPI command cycle will be activated (CE# low) until set this bit to 1. Thats to say setting this bit to 1 will stop activating SPI interface after Read/Write operation finished or immediately if no Read/Write operation is in progress. But when different CE command is entered, SPI interface will be deactivated immediately.
1:0	RW	**Command Mode** 00: Auto-Read (0x03/0x13 + Address + Read data [1/2/3/4 bytes]) 01: Normal-Read (CMD + Address + Read data [1/2/3/4 bytes]) 10: Normal-Write (CMD + Address + Write data [1/2/3/4 bytes]) 11: User-Mode (Read/write data [1/2/3/4 bytes]) At User-Mode, address only used for decoding CE, all address decoded in the same CE address range are valid, and data will be read/write from/to the SPI flash in the order of LSB byte first. This mode provides a flexible programming method for specific command type other than Normal-Read/Write command supported. CMD = 1 byte of data from bit[23:16] of this register Address = 3 or 4 bytes from AHB address bus and output in the order from MSB byte to LSB byte Read/write data = 1∼4 bytes data from AHB data bus and output in the order from LSB byte to MSB byte.

Offset: 30h	**SPIR30: CE0 Address Decoding Range Register**	**Init = 0x07F0_0000**
Offset: 34h	**SPIR34: CE1 Address Decoding Range Register**	**Init = 0x0000_0000**
Offset: 38h	**SPIR38: CE2 Address Decoding Range Register**	**Init = 0x0000_0000**

Bit	Attr.	Description
31:16	RW	**End address A[31:16]** Here defines the CEx upper bound address limit with the unit of 1MB. Only A[27:20] are used for decoding.

to next page

from previous page

15:0	RW	**Start address A[31:16]** Here defines the CEx lower bound address limit with the unit of 1MB. Only A[27:20] are used for decoding. The Start address of CE0 is fixed at 0x0 and not changeable. The decoding range is only applied on the AHB bus command, host access SPI1 that go through direct path is not constrained by this range. The valid address range for each CE is: StartAdr $\leq$ CEx $\leq$ EndAdr. When configure Start address = End address, then the decoding for this range is disabled. The valid flash decoding address is as blelow, and the total decoding space is 256MB: SPI1: 0x3000_0000 - 0x3FFF_FFFF SPI2: 0x5000_0000 - 0x5FFF_FFFF The default address segment for each CE of SPI1 listed below: CE0 = 0x3000_0000 - 0x37FF_FFFF (128MB, when OTP trap_hspi_size [2:0] = 0) CE1 = disabled When OTP trap_hspi_size [2:0] $\neq$ 0, the default address segment for CE0 is determined by OTP trap_hspi_size [2:0]. When OTP trap_hspi_size [2:0] $\neq$ 0 and OTP trap_hspi_abrmode = 0, the default address segment for CE1 is determined by OTP trap_hspi_size [2:0]. The default address segment for each CE of SPI2 listed below: CE0 = 0x5000_0000 - 0x57FF_FFFF (128MB) CE1 = disabled CE2 = disabled The address segments setting of 3 chips select must not be overlapped. It will cause error when overlapped. There is a rule must be followed in defining the Start address. For example: 0x30000000, valid address mask = 0x0FFFFFF, can address 0 $\sim$ 256MB flash chip 0x34000000, valid address mask = 0x03FFFFF, can address 0 $\sim$ 64MB flash chip 0x32000000, valid address mask = 0x01FFFFF, can address 0 $\sim$ 32MB flash chip 0x31000000, valid address mask = 0x00FFFFF, can address 0 $\sim$ 16MB flash chip 0x30800000, valid address mask = 0x007FFFF, can address 0 $\sim$ 8MB flash chip

Offset: 54h **SPIR54: SPI Dummy Cycle Data Register** **Init = 0**

Bit	R/W	Description
31:8	RO	**Reserved (0)**
7:0	RW	**SPI dummy cycle output data** This register will be outputed as data at the head of dummy cycle. Shared by all chip selects.

Offset: 64h **SPIR64: Alternate Boot Control/Status (SPI1 only)** **Init = 0**

Bit	R/W	Description
31:7	RO	**Reserved (0)**

to next page

from previous page

6	RW	**Alternate Boot Mode selection** 0: 2 chips mode 1: 1 chip mode When OTP trap_en_hspiabr = 1, then this bit is determined by OTP trap_hspi_abrmode and not changable. When OTP trap_en_hspiabr = 0, then this bit can be configured by firmware. Firmware set 1 by writing '1', and clear to 0 by writing bit[31:24] = 0xEA.
5	RO	**Reserved (0)**
4	RW	**Boot flash source select indicator** 0: boot from primary source 1: boot from alternate source When OTP trap_en_hspiabr = 1 and (OTP trap_en_hspi_auxpin = 1 or OTP trap_en_hspiabr_selpin = 1), then this bit is determined by external pin SPI1ABR . When OTP trap_en_hspiabr = 0, then this bit can be configured by firmware. Firmware set 1 by writing '1', and clear to 0 by writing bit[23:16] = 0xEA.
3:1	RW	**SPI size of boot flash** 000: no define 001: 2 MB 010: 4 MB 011: 8 MB 100: 16 MB 101: 32 MB 110: 64 MB 111: 128 MB When OTP trap_hspi_size != 0, then the size is determined by OTP setting. When OTP trap_hspi_size == 0, then the size can be configured by firmware. When the SPI size value is not zero, then below setting will be forced to specific value depends on the SPI size: 1. High/Low address partition of 1 chip ABR mode on CE0. 2. Host direct access address clamping. 3. SPIR04 bit0 and bit4. 4. SPIR30. 5. SPIR34 if ABR mode select 2 chips mode. 6. SPIRB0 and SPIRB4 if CRTM size != 0.
0	RO	**Reserved (0)**

Offset: 6Ch SPIR6C: Host Direct Access Commands #4 (SPI1 only) Init = 0x00001303

Bit	R/W	Description
31:28	RW	**IO Mode for Read** 0000: single bit. 0010: dual bit read/write, data cycle only. 0011: dual bit read/write, including address and dummy byte cycle. 0100: quad bit read/write, data cycle only. 0101: quad bit read/write, including address and dummy byte cycle. 1xxx: QPI mode, quad bit on command/address/data cycles. others: reserved

to next page

from previous page

27:24	RW	**IO Mode for Page Program** 0000: single bit. 0010: dual bit read/write, data cycle only. 0011: dual bit read/write, including address and dummy byte cycle. 0100: quad bit read/write, data cycle only. 0101: quad bit read/write, including address and dummy byte cycle. 1xxx: QPI mode, quad bit on command/address/data cycles. others: reserved
23:16	RO	**Reserved (0)**
15:8	RW	**Command for Read 4B mode**
7:0	RW	**Command for Read 3B mode**
Note : The host direct access command also protected by command and address filter. The command selection of 3B address or 4B address mode is depended on SPIR04[0].		

Offset: 70h SPIR70: Host Direct Access Commands #1 (SPI1 only) Init = 0x80000605

Bit	R/W	Description
31	RW	**Enable command encoding for direct access** 0: Disable, host should manage the controller to issue the required commands. 1: Enable, support encoded page program and erase commands. Refer Section 15.5.
30:26	RO	**Direct access command execution step** **(for debugging purpose only)** 00000: Idle others: busy
25	RO	**Reserved (0)**
24	RW	**Host access SPI1 CE selection** 0: CE0 1: CE1
23:16	RO	**SPI flash status register** **(for debugging purpose only)** bit16: WIP, write in progress bit17: WEL, write enable latch bit18 ~ bit22: flash specific definition bit23: SRWD, status register write protect
15:8	RW	**Command for Write Enable**
7:0	RW	**Command for Read Status** Check status bit0 = 0 for program in progress status check.
Note : The host direct access command also protected by command and address filter.		

Offset: 74h SPIR74: Host Direct Access Commands #2 (SPI1 only) Init = 0x21201202

Bit	R/W	Description
31:24	RW	**Command for Sector (4K) Erase 4B mode**
23:16	RW	**Command for Sector (4K) Erase 3B mode**
15:8	RW	**Command for Page Program 4B mode**
7:0	RW	**Command for Page Program 3B mode**
Note : The host direct access command also protected by command and address filter. The command selection of 3B address or 4B address mode is depended on SPIR04[0].		

Offset: 78h	SPIR78: Host Direct Access Commands #3 (SPI1 only)	Init = 0xDCD85C52
Bit	**R/W**	**Description**
31:24	RW	**Command for Block (64K) Erase 4B mode**
23:16	RW	**Command for Block (64K) Erase 3B mode**
15:8	RW	**Command for Block (32K) Erase 4B mode**
7:0	RW	**Command for Block (32K) Erase 3B mode**
Note : The host direct access command also protected by command and address filter. The command selection of 3B address or 4B address mode is depended on SPIR04[0].		

Offset: 7Ch	SPIR7C: DMA Buffer Mode Length Register	Init = 0
Bit	**R/W**	**Description**
6:0	RW	**FIFO Length Status** Actual data size = FIFO length * 4 – This register reflect the current FIFO length status when DMA buffer mode is on-going. So firmware can read this status to get current FIFO condition for buffer mode data push/pop indication. Value = 0x40 is full condition Value = 0x00 is empty condition – The value can be updated by firmware when DMA is in disabled state, so firmware can preload a value for TX transfer. – After DMA is enabled, this register is read only. And firmware can write(TX)/read(RX) FIFO data port SPIR200 to update the length counter.

Offset: 80h	SPIR80: DMA Control/Status Register	Init = 0
Bit	**R/W**	**Description**
31	RW	**DMA Request** Write SPIR80 = 0xAEED0000 to set this bit to '1'. And hardware will clear the request to '0' after DMA done, or FW can clear to '0' by writing SPIR80 = 0xDEEA0000.
30	RO	**DMA Grant** 0: DMA is not allowed to be used. All DMA related control registers are not allowed to be written. 1: DMA is granted to be used.
29:20	RO	**Reserved (0)**
19:16	RW	**SPI clock frequency setting** Calibration mode use only. The value definition is the same as CEx Control Register bit[11:8] and with bit[27:24] = 0.
15:8	RW	**SPI read data input delay cycle setting** Calibration mode use only. The value definition is the same as SPIR94.
7:5	RO	**Reserved (0)**
4	RW	**DMA Buffer Mode** 0: Memory mode, transfer to/from DRAM/SRAM. 1: Buffer mode, transfer to/from DMA buffer. The DMA buffer works as FIFO mode for data transfer, and DMA will stop temporarily when FIFO is empty or full. Host direct access port do not support DMA buffer mode.

to next page

from previous page

3	RW	**Calibration Mode** 0: Normal mode 1: Calibration mode. At this mode, SPI clock rate and read delay cycle parameters setting will be replaced by bit[15:4]. Calibration mode should be enabled in companion with bit[2]=1.
2	RW	**CheckSum Calculation Only** 0: Normal DMA operation 1: Checksum accumulation only This bit is valid only when the DMA direction = 0. And transfer length should be 4 bytes boundary.
1	RW	**DMA Direction** 0: Read flash, move from flash to external memory 1: Write flash, move from external memory to flash
0	RW	**DMA Enable** 0: Disable 1: Enable DMA operation
Note : SPIR80[19:0] to SPIR8C are writing prohibited if SPIR80[30] = 0. After each DMA cycle completion, it should clear SPIR80[0] = 0 to reset DMA status and to start new DMA cycle. Refer 15.4 for the DMA programming guidance.		

Offset: 84h **SPIR84: DMA Flash Side Address** **Init = X**

Bit	R/W	Description
31:28	RO	**Reserved**
27:2	RW	**Flash side start address** For DMA Read flash, this is the source address. For DMA Write flash, this is the destination address. DMA can only execute on 4 bytes boundary. And the valid address range is as below: SPI1 : 0x3000_0000 - 0x3FFF_FFFF. SPI2 : 0x5000_0000 - 0x5FFF_FFFF. When read, it shows the current working address.
1:0	RO	**Reserved**

Offset: 88h **SPIR88: DMA DRAM/SRAM Side Address** **Init = X**

Bit	R/W	Description
31:2	RW	**DRAM side start address (Memory mode only)** For DMA Read flash, this is the destination address. For DMA Write flash, this is the source address. DMA can only execut on 4 bytes boundary. When read, it shows the current working address.
1:0	RO	**Reserved**

Offset: 8Ch **SPIR8C: DMA Length Register** **Init = X**

Bit	R/W	Description
31:25	RO	**Reserved (0)**

to next page

from previous page

24:0	RW	**DMA Length** From 1 byte to 32MB 0: 1 byte 0x7FFFFF: 32M bytes When DMA is on-going, the length register will count down, and reach 0 when DMA finished.

Offset: 90h **SPIR90: CheckSum Calculation Result** **Init = 0**

Bit	R/W	Description
31:0	RO	**CheckSum Calculation Result** Accumulate the flash read data checksum result, 32bits base. This register will be reset when DMA disabled.

Offset: 94h **SPIR94: CE0 SPI Flash Read Timing Compensation** **Init = 0**
Offset: 98h **SPIR98: CE1 SPI Flash Read Timing Compensation** **Init = 0**
Offset: 9Ch **SPIR9C: CE2 SPI Flash Read Timing Compensation** **Init = 0**

Bit	Attr.	Description
31:24	RW	**SPICLK = HCLK/5, delay cycle for data input latch point**
23:16	RW	**SPICLK = HCLK/4, delay cycle for data input latch point**
15:8	RW	**SPICLK = HCLK/3, delay cycle for data input latch point**
7:0	RW	**SPICLK = HCLK/2, delay cycle for data input latch point** Bit[7:4]: DI input delay selection, each step is about 0.5ns. Bit[3]: 0: disable DI input delay 1: enable DI input delay Bit[2:0]: 0: no delay 1: delay 1 HCLK 2: delay 2 HCLK 3: delay 3 HCLK 4: delay 4 HCLK others: delay 5 HCLK

Note :
This read timing applied to all CE's SPI flash. So it may need to change the timing for different CE SPI flash.

Offset: A0h **SPIRA0: Command Filter Control Register** **Init = 0**

Bit	R/W	Description
31:3	RO	**Reserved**
2	RW	**Enable read/write command filter for CE2**
1	RW	**Enable read/write command filter for CE1**

to next page

from previous page

0	RW	**Enable read/write command filter for CE0** 0: disable filter, all commands are allowed. 1: enable filter, only commands defined in SPIR100 ~ SPIR17C are allowed. When OTP trap_en_hspi_auxpin = 1 and external SPI1WP# pin is active, CE0 filter will be always enabled. If SPI1WP# is not active, then CE0 filter can be controlled by software. When command filter is enabled, below limitation would be applied: 1. User-Mode command (Command Mode = "11") would be disabled.

Offset: A4h **SPIRA4: Write Address Filter Control Register** **Init = 0**

Bit	R/W	Description
31:12	RO	**Reserved**
11:10	RW	**Mode of write address filter #6**
9:8	RW	**Mode of write address filter #5**
7:6	RW	**Mode of write address filter #4**
5:4	RW	**Mode of write address filter #3**
3:2	RW	**Mode of write address filter #2**
1:0	RW	**Mode of write address filter #1** 0x: NOP 10: write enabled 11: write disabled When OTP trap_en_hspi_auxpin = 1 and CRTM is enabled, filter #1 & #2 will be forced to "11" mode. And address range at SPIRB0 & SPIRB4 are determined by HW based on SPI size and CRTM size. When CRTM is disabled, filter #1 & #2 can be set by firmware.

Note :
Address filters take effect under 2 conditions:

1. When command filter is enabled, then write is allowed when command matches AQCD and address pass address filters.
2. When command filter is disabled, then write is allowed when address pass address filters or configured in User-Mode.

The write filter mode is defined by 2 registers: SPIR00 bit[18:16] and SPIRA4.
Protection mode as below:

1. If SPIR00 is write disabled, the write can be enabled by SPIRA4.
2. If SPIR00 is write enabled, then write can be disabled by SPIRA4.

The address segments defined on SPIRB0 ~ SPIRC4 controls the address range for write allowed or disallowed.

Offset: A8h **SPIRA8: Register Lock Control Register (SRST#)** **Init = 0**

Bit	R/W	Description
31:13	RO	**Reserved**
12	RW	**Lock SPIR70/SPIR74/SPIR78 from write until reset by SRST# (SPI1)**
11	RW	**Lock SPIR64 from write until reset by SRST# (SPI1)**
10	RW	**Lock SPIRA4 and Address Filter from write until reset by SRST#**
9	RW	**Lock SPIRA0 and FQCD/AQCD from write until reset by SRST#**

to next page

from previous page

8	RW	**Lock SPIR94/SPIR98/SPIR9C from write until reset by SRST#**
7	RW	**Lock SPIR38 from write until reset by SRST#**
6	RW	**Lock SPIR34 from write until reset by SRST#**
5	RW	**Lock SPIR30 from write until reset by SRST#**
4	RW	**Lock SPIR18 from write until reset by SRST#**
3	RW	**Lock SPIR14 from write until reset by SRST#**
2	RW	**Lock SPIR10 from write until reset by SRST#**
1	RW	**Lock SPIR04 from write until reset by SRST#**
0	RW	**Lock SPIR00 from write until reset by SRST#**
Note : This register is write '1' only, and reset to '0' only when SRST# active.		

Offset: ACh **SPIRAC: Register Lock Control Register (Watchdog)** **Init = 0**

Bit	R/W	Description
31:13	RO	**Reserved**
12	RW	**Lock SPIR70/SPIR74/SPIR78 from write until reset by watchdog (SPI1)**
11	RW	**Lock SPIR64 from write until reset by watchdog (SPI1)**
10	RW	**Lock SPIRA4 and Address Filter from write until reset by watchdog**
9	RW	**Lock SPIRA0 and FQCD/AQCD from write until reset by watchdog**
8	RW	**Lock SPIR94/SPIR98/SPIR9C from write until reset by watchdog**
7	RW	**Lock SPIR38 from write until reset by watchdog**
6	RW	**Lock SPIR34 from write until reset by watchdog**
5	RW	**Lock SPIR30 from write until reset by watchdog**
4	RW	**Lock SPIR18 from write until reset by watchdog**
3	RW	**Lock SPIR14 from write until reset by watchdog**
2	RW	**Lock SPIR10 from write until reset by watchdog**
1	RW	**Lock SPIR04 from write until reset by watchdog**
0	RW	**Lock SPIR00 from write until reset by watchdog**
Note : This register is write '1' only, and reset to '0' only when watchdog reset.		

Offset: B0h **SPIRB0: Write Address Filter Register #1** **Init = 0**
Offset: B4h **SPIRB4: Write Address Filter Register #2** **Init = 0**
Offset: B8h **SPIRB8: Write Address Filter Register #3** **Init = 0**
Offset: BCh **SPIRBC: Write Address Filter Register #4** **Init = 0**
Offset: C0h **SPIRC0: Write Address Filter Register #5** **Init = 0**
Offset: C4h **SPIRC4: Write Address Filter Register #6** **Init = 0**

Bit	Attr.	Description
31:16	RW	**Segments Upper bound address bit[27:12]**
15:0	RW	**Segments Lower bound address bit[27:12]** The address range is defined between 0x30000000 ~ 0x3FFFFFFF. And active to CE0, CE1, or CE2 based on the address range defined for the CE.

to next page

from previous page

Note :
The segments address range is defined as below:
lower_bound $\leq$ address $\leq$ upper_bound

Offset: 100h	**FQCD00: Fully Qualified Command**	**Init = 0x80001303**
Offset: 104h	**FQCD01: Fully Qualified Command**	**Init = 0x80000C0B**
Offset: 108h	**FQCD02: Fully Qualified Command**	**Init = 0x80009F9F**
Offset: 10Ch	**FQCD03: Fully Qualified Command**	**Init = 0x80000505**
Offset: 110h	**FQCD04: Fully Qualified Command**	**Init = 0x80000101**
Offset: 114h	**FQCD05: Fully Qualified Command**	**Init = 0x80000606**
Offset: 118h	**FQCD06: Fully Qualified Command**	**Init = 0x80000404**
Offset: 11Ch	**FQCD07: Fully Qualified Command**	**Init = 0x00000000**
Offset: 120h-14Ch	**FQCD08-FQCD19: Fully Qualified Command**	**Init = 0x00000000**

31	RW	**Enable Entry**
30:16	RO	**Reserved**
15:8	RW	**Fully qualified Command**
7:0	RW	**Fully qualified Command**

Note :
If OTP trap_en_hspi_auxpin = 1, then the write of these FQCD registers is controlled by external pin SPI1WP# .
When SPI1WP# = 0, then all these registers are Read Only.
When SPI1WP# = 1, then all these registers are Read/Write.
When FQCD register is enabled by bit31 but not defined, please set the command as 0x0.
Below commands would be used by controller for auto command generation:
0x03, 0x13, 0x05

Offset: 150h	**AQCD00: Address Qualified Command**	**Init = 0x80001202**
Offset: 154h	**AQCD01: Address Qualified Command**	**Init = 0x8000DCD8**
Offset: 158h	**AQCD02: Address Qualified Command**	**Init = 0x00000000**
Offset: 15Ch	**AQCD03: Address Qualified Command**	**Init = 0x00000000**
Offset: 160h	**AQCD04: Address Qualified Command**	**Init = 0x00000000**
Offset: 164h	**AQCD05: Address Qualified Command**	**Init = 0x00000000**
Offset: 168h	**AQCD06: Address Qualified Command**	**Init = 0x00000000**
Offset: 16Ch	**AQCD07: Address Qualified Command**	**Init = 0x00000000**
Offset: 170h	**AQCD08: Address Qualified Command**	**Init = 0x00000000**
Offset: 174h	**AQCD09: Address Qualified Command**	**Init = 0x00000000**
Offset: 178h	**AQCD10: Address Qualified Command**	**Init = 0x00000000**
Offset: 17Ch	**AQCD11: Address Qualified Command**	**Init = 0x00000000**

Bit	Attr.	Description
31	RW	**Enable Entry**
30:16	RO	**Reserved**
15:8	RW	**4-Byte Addressing Command** Check address bits A[27:12] to avoid address aliasing, and address byte lane configuration should be 4 bytes.
7:0	RW	**3-Byte Addressing Command** Check address bits A[27:12] to avoid address aliasing, and address byte lane configuration should be 3 bytes.

to next page

from previous page

Note :
If OTP trap_en_hspi_auxpin = 1, then the write of these FQCD registers is controlled by external pin SPI1WP# .
When SPI1WP# = 0, then all these registers are Read Only.
When SPI1WP# = 1, then all these registers are Read/Write.
When AQCD register is enabled by bit31 but not defined, please set the command as 0x0.
For commands listed in these registers can be executed correctly and avoid address aliasing, the address size should be 3 or 4 bytes that matching the command definition, and the address should pass the address filter.

Offset: 200h	SPIR200: DMA FIFO Mode Data Port Register	Init = X

Bit	R/W	Description
31:0	RW	**FIFO data read/write port** 1. This data port work when DMA buffer mode and DMA are both enabled. 2. When in TX mode transfer and FIFO is not full, firmware can write this port to push data into FIFO for TX transfer. And the length counter will increment by 1 for each write. Write would be ignored if FIFO is full. 3. When in RX mode transfer and FIFO is not empty, firmware can read this port to pop data from FIFO to get RX data. And the length counter will decrement by 1 for each read. Read return dummy data if FIFO is empty. 4. Read/Write of this register is 32 bits format.

Offset: 200h-2FFh	SPIRBUF: DMA Buffer R/W Port	Init = X

Bit	R/W	Description
31:0	RW	**DMA Buffer read/write port** 1. This DMA buffer can be accessed by CPU when DMA is disabled. 2. DMA is 256 bytes, ranging from address offset 0x200 to 0x2FF. 3. For DMA buffer TX mode transfer, firmware can write this buffer to preload data into FIFO for TX transfer. 4. Read/Write of this buffer is 32 bits format.

15.4 Programming Guide

15.4.1 DMA Operation - Memory Mode

- DMA operation can not across the CE address segment boundary.
- DMA DRAM/SRAM side address can not over the maximum supported DRAM size. It will wrap back if overflow.
- Start DMA procedure:
  1. Write SPIR80 = 0xAEED0000 to get the DMA owner right
  2. Wait until SPIR80[30] = 1
  3. Set SPIR84
  4. Set SPIR88
  5. Set SPIR8C
  6. Set SPIR80[1:0]
  7. Wait DMA done

8. Clear SPIR80[0] = 0 to reset DMA and start new DMA cycle.
9. If more DMA cycles to run, goto step 3, otherwise next.
10. After finished using DMA, release the DMA owner by writing SPIR80 = 0xDEEA0000

– Reset or terminate DMA operation:

1. Set SPIR80[0] = 0
2. DMA should be reset once at each DMA operation end.

15.4.2 DMA Operation - Buffer Mode

– Before enable DMA, it can preload the TX data into DMA buffer and set the preload length in SPIR7C (4-bytes unit).
– After DMA is enabled and DMA is on-going, the DMA buffer works as FIFO mode for firmware to push/pop TX/RX data at the same data port address.
– Start DMA procedure:

1. Write SPIR80 = 0xAEED0000 to get the DMA owner right
2. Wait until SPIR80[30] = 1
3. For TX, it can preload data into buffer
4. Set SPIR7C
5. Set SPIR84
6. Set SPIR8C
7. Set SPIR80[1:0] and bit[4]=1
8. It can check SPIR7C status to push/pop more data for TX/RX when DMA is on-going.
9. Wait DMA done
10. Clear SPIR80[0] = 0 to reset DMA and start new DMA cycle.
11. If more DMA cycles to run, goto step 3, otherwise next.
12. After finished using DMA, release the DMA owner by writing SPIR80 = 0xDEEA0000

– Reset or terminate DMA operation:

1. Set SPIR80[0] = 0
2. DMA should be reset once at each DMA operation end.

15.4.3 DMA CheckSum Calculation Mode

– CheckSum calculation is only applied to flash read mode.
– Command procedure:

1. Write SPIR80 = 0xAEED0000 to get the DMA owner right
2. Wait until SPIR80[30] = 1
3. Set SPIR84
4. Set SPIR8C
5. Set SPIR80[1:0] = ”01”
6. SPIR80[2] is set when only checksum calculation is necessary, no data movement
7. SPIR80[3] is set when wants to try different clock rate and input compensation delay for DMA cycle
8. Wait DMA done
9. After finished using DMA, release the DMA owner by writing SPIR80 = 0xDEEA0000

15.5 Host Direct Access Command Decoding

SPI1 supports a direct path for eSPI or LPC to access SPI1 controller. The supported commands as below:

1. Memory Write
2. Memory Read
3. Erase
4. Register Write
5. Register Read

Memory Write

Memory write supports below SPI Page Program sequence:

1. Issue write enable command (0x06)
2. Issue page program command (0x02/0x12)
3. Send write address/data come from Host
4. Check Status and wait ”write in progress” flag idle

Memory Read

Memory read supports below SPI Read sequence:

1. Use Auto-Read command (0x03/0x13) for SPI read

Memory Erase

Memory erase supports below SPI Erase sequence:

1. Issue write enable command (0x06)
2. There are 3 erase types can be chosen by Host:
   - Sector (4KB) erase : Issue SE erase command (0x20/0x21)
   - Block (32KB) erase : Issue BE32K erase command (0x52/0x5C)
   - Block (64KB) erase : Issue BE64K erase command (0xD8/0xDC)
3. Check Status and wait ”write in progress” flag idle

Register Write

Implement register write.

Register Read

Implement register read.

16 10/100/1G Ethernet MAC Controller (MAC1/MAC2/MAC3/MAC4)

16.1 Overview

AST2600 integrates four sets of high performance Ethernet MAC modules which can operate at 10 Mbps, 100 Mbps, or 1000 Mbps. The two MAC modules are totally identical and can be enabled or disabled independently. The only difference is that each MAC has its own base addresses for register programming. The digital interface of each MAC can be RMII, RGMII.

Base address of Ethernet MAC #1 = 0x1E66_0000
Base address of Ethernet MAC #2 = 0x1E68_0000
Base address of Ethernet MAC #3 = 0x1E67_0000
Base address of Ethernet MAC #4 = 0x1E69_0000
Physical address = (Base address of Ethernet MAC) + Offset

MAC00: Interrupt Status Register (ISR)
MAC04: Interrupt Enable Register (IME)
MAC08: MAC Most Significant Address Register #0 (MAC_MADR0)
MAC0C: MAC Least Significant Address Register #0 (MAC_LADR0)
MAC10: Multicast Address Hash Table 0 Register (MAHT0)
MAC14: Multicast Address Hash Table 1 Register (MAHT1)
MAC18: Normal Priority Transmit Poll Demand Register (NPTXPD)
MAC1C: Receive Poll Demand Register (RXPD)
MAC20: Normal Priority Transmit Ring Base Address Register (NPTXR_BADR)
MAC24: Receive Ring Base Address Register (RXR_BADR)
MAC28: High Priority Transmit Poll Demand Register (HPTXPD)
MAC2C: High Priority Transmit Ring Base Address Register (HPTXR_BADR)
MAC30: Interrupt Timer Control Register (ITC)
MAC34: Automatic Polling Timer Control Register (APTC)
MAC38: DMA Burst Length and Arbitration Control Register (DBLAC)
MAC3C: Debug Status Register (DBG_STS)
MAC40: Feature Register (FEAR)
MAC48: Transmit Priority Arbitration and FIFO Control Register (TPAFCR)
MAC4C: Receive Buffer Size Register (RBSR)
MAC50: MAC Control Register (MACCR)
MAC54: MAC Status Register (MACSR)
MAC58: Control Mode Register (TM)
MAC68: Flow Control Register (FCR)
MAC6C: Back Pressure Register (BPR)
MAC70: Wake-On-LAN Control Register (WOL_CR)
MAC74: Wake-On-LAN Status Register (WOL_SR)
MAC78: MAC Most Significant Address Register #1 (MAC_MADR1)
MAC7C: MAC Least Significant Address Register #1 (MAC_LADR1)
MAC80: MAC Most Significant Address Register #2 (MAC_MADR2)
MAC84: MAC Least Significant Address Register #2 (MAC_LADR2)
MAC88: MAC Most Significant Address Register #3 (MAC_MADR3)
MAC8C: MAC Least Significant Address Register #3 (MAC_LADR3)
MAC90: Debug Normal Priority Tx-Ring Pointer Register DBG_NTXR_PTR)
MAC94: Debug Rx-Ring Pointer Register (DBG_RXR_PTR)
MAC9C: Debug High Priority Tx-Ring Pointer Register (DBG_HTXR_PTR)
MACA0: Debug Tx-Counter Register #0 (DBG_TXCNT0)
MACA4: Debug Tx-Counter Register #1 (DBG_TXCNT1)
MACA8: Debug Tx-Counter Register #2 (DBG_TXCNT2)
MACAC: Debug Rx-Counter Register #0 (DBG_RXCNT0)
MACB0: Debug Rx-Counter Register #1 (DBG_RXCNT1)
MACB4: Debug Rx-Counter Register #2 (DBG_RXCNT2)

MACB8: Debug Rx-Counter Register #3 (DBG_RXCNT3)
MACBC: Debug Rx-Counter Register #4 (DBG_RXCNT4)
MACC0: Debug Rx-Counter Register #5 (DBG_RXCNT5)
MACC4: Debug Rx-Counter Register #6 (DBG_RXCNT6)
MACC8: Debug Rx-Counter Register #7 (DBG_RXCNT7)
MACD0: MAC Most Significant Address Register #4 (MAC_MADR4)
MACD4: MAC Least Significant Address Register #4 (MAC_LADR4)
MACD8: MAC Most Significant Address Register #5 (MAC_MADR5)
MACDC: MAC Least Significant Address Register #5 (MAC_LADR5)
MACE0: MAC Most Significant Address Register #6 (MAC_MADR6)
MACE4: MAC Least Significant Address Register #6 (MAC_LADR6)
MACE8: MAC Most Significant Address Register #7 (MAC_MADR7)
MACEC: MAC Least Significant Address Register #7 (MAC_LADR7)
MACF0: Filter Control Register #0 (FL_CTRL0)
MACF4: Filter Control Register #1 (FL_CTRL1)
MACF8: Protocol Filter Byte Register #0 (FL_PROL0)
MACFC: Protocol Filter Byte Register #1 (FL_PROL1)
MAC100: Source & Destination Port Filter Word Register #0 (FL_SDPORTW0)
MAC104: Source & Destination Port Filter Word Register #1 (FL_SDPORTW1)
MAC108: Source & Destination Port Filter Word Register #2 (FL_SDPORTW2)
MAC10C: Source & Destination Port Filter Word Register #3 (FL_SDPORTW3)
MAC110: Source & Destination Port Filter Word Register #4 (FL_SDPORTW4)
MAC114: Source & Destination Port Filter Word Register #5 (FL_SDPORTW5)
MAC118: Source & Destination Port Filter Word Register #6 (FL_SDPORTW6)
MAC11C: Source & Destination Port Filter Word Register #7 (FL_SDPORTW7)
MAC120: Source Address Filter Most Significant Address Register #0 (FL_SAM0)
MAC124: Source Address Filter Least Significant Address Register #0 (FL_SAL0)
MAC128: Source Address Filter Most Significant Address Register #1 (FL_SAM1)
MAC12C: Source Address Filter Least Significant Address Register #1 (FL_SAL1)
MAC130: Source Address Filter Most Significant Address Register #2 (FL_SAM2)
MAC134: Source Address Filter Least Significant Address Register #2 (FL_SAL2)
MAC138: Source Address Filter Most Significant Address Register #3 (FL_SAM3)
MAC13C: Source Address Filter Least Significant Address Register #3 (FL_SAL3)
MAC140: Type Filter Word Register #0 (FL_TYPE0)
MAC144: Type Filter Word Register #1 (FL_TYPE1)
MAC148: Type Filter Word Register #2 (FL_TYPE2)
MAC14C: Type Filter Word Register #3 (FL_TYPE3)

16.2 Features

- Integrate dual MAC modules compliant with IEEE802.3 and IEEE802.3z specification
- Support 10/100/1000M bps transfer rates
- Support Reduced Media Independent Interface (RMIIx4), Reduced Gigabit Media Independent Interface (RGMIIx4)
- Support IEEE 802.1Q VLAN tag insertion and deletion
- Support High Priority Transmit Queue for QoS and CoS applications
- Independent TX/RX FIFO
- Support half and full duplex (1000 Mbps mode only supports full duplex)
- Support flow control for full duplex and backpressure for half duplex
- Integrated link list DMA engine with direct M-Bus accesses for transmitting and receiving packets
- Support Wake-on-LAN function with three wake-up events: Link status change, Magic packet and Wake-up frame

16.3 Registers :

Base address of Ethernet MAC #1 = 0x1E66:0000
Base address of Ethernet MAC #2 = 0x1E68:0000
Base address of Ethernet MAC #3 = 0x1E67:0000
Base address of Ethernet MAC #4 = 0x1E69:0000

Offset: 00h	MAC00: Interrupt Status Register (ISR)	Init = 0
Bit	**R/W**	**Description**
31:13	RO	**Reserved (0)**
12	RW	**WOL_STUS: Wake-On-Lan Status** Clearing MAC74 Bit3~0 to clear this status flag.
11	RO	**Reserved (0)**
10	RW	**HPTXBUF_UNAVA: High priority transmit buffer unavailable (Writing "1" to clear)**
9 :8	RO	**Reserved (0)**
7	RW	**TPKT_LOST: Tx-packets lost due to late/excessive collision (Writing "1" to clear)**
6	RW	**NPTXBUF_UNAVA: Normal priority transmit buffer unavailable (Writing "1" to clear)**
5	RW	**TPKT2F: Tx-packets be moved into the the TX FIFO (Writing "1" to clear)**
4	RW	**TPKT2E: Tx-packets be transmitted to Ethernet (Writing "1" to clear)**
3	RW	**RPKT_LOST: Rx-packet lost due to RX FIFO full (Writing "1" to clear)**
2	RW	**RXBUF_UNAVA: Receiving buffer unavailable (Writing "1" to clear)**
1	RW	**RPKT2F: Writing the rx-packet to the RX FIFO (Writing "1" to clear)**
0	RW	**RPKT2B: Writing the rx-packet to the RX buffer (Writing "1" to clear)**

Offset: 04h	MAC04: Interrupt Enable Register (IME)	Init = 0
Bit	**R/W**	**Description**
31:13	RO	**Reserved (0)**
12	RW	**Interrupt enable of MAC00 Bit12**
11	RO	**Reserved (0)**
10	RW	**Interrupt enable of MAC00 Bit10**
9: 8	RO	**Reserved (0)**
7: 0	RW	**Interrupt enable of MAC00 Bit7~0**

Offset: 08h	MAC08: MAC Most Significant Address Register #0 (MAC_MADR0)	Init = 0
Bit	**R/W**	**Description**
31	RW	**Disable MAC address #0 control bit** 0: Enable MAC address #0 1: Disable MAC address #0
30:16	RO	**Reserved (0)**
15:0	RW	**MAC_MADR0** The most significant 2 bytes of MAC address #0

Offset: 0Ch	MAC0C: MAC Least Significant Address Register #0 (MAC_LADR0)	Init = 0
Bit	**R/W**	**Description**
31:0	RW	**MAC_LADR0** The least significant 4 bytes of MAC address #0

Offset: 10h	MAC10: Multicast Address Hash Table 0 Register (MAHT0)	Init = 0
Bit	**R/W**	**Description**
31:0	RW	**MAHT0** Multicast address hash table bytes 3~0 (Hash table 31:0)

Offset: 14h	MAC14: Multicast Address Hash Table 1 Register (MAHT1)	Init = 0
Bit	**R/W**	**Description**
31:0	RW	**MAHT1** Multicast address hash table bytes 7~4 (Hash table 63:32)

Offset: 18h	MAC18: Normal Priority Transmit Poll Demand Register (NPTXPD)	Init = 0
Bit	**R/W**	**Description**
31:0	WT	**NPTXPD** When writing any value to the register, MAC engine reads the normal priority transmit descriptor, process and checks the TXDMA_OWN (TXDES#0 [31]) bit. If TXDMA_OWN (TXDES#0 [31]) = 1, it will move the transmit buffer data into the TX FIFO.

Offset: 1Ch	MAC1C: Receive Poll Demand Register (RXPD)	Init = 0
Bit	**R/W**	**Description**
31:0	WT	**RXPD** When writing any value to the register, MAC engine reads the receive descriptor, process and checks the RXPKT_RDY (RXDES#0 [31]) bit. If RXPKT_RDY (RXDES#0 [31]) = 0, it will move the receive packet data from the RX FIFO into the receiving buffer in the local memory.

MAC20: Normal Priority Transmit Ring Base Address Register (NPTXR_BADR)		
Offset: 20h		**Init = 0**
Bit	**Attr.**	**Description**
31	RO	**Reserved (0)**
30:4	RW	**NPTXR_BADR: Base address of the normal priority transmit ring [30:4]** **The base address must be 16 bytes aligned**
3 :0	RO	**Reserved (0)**

Offset: 24h	MAC24: Receive Ring Base Address Register (RXR_BADR)	Init = 0
Bit	**R/W**	**Description**
31	RO	**Reserved (0)**
30:4	RW	**RXR_BADR: Base address of the receive ring [30:4]** **The base address must be 16 bytes aligned**

to next page

from previous page

3 :0	RO	**Reserved (0)**

Offset: 28h	**MAC28: High Priority Transmit Poll Demand Register (HPTXPD)**	**Init = 0**
Bit	**R/W**	**Description**
31:0	WT	**HPTXPD** When writing any value to the register, MAC engine reads the high priority transmit descriptor, process and check the TXDMA_OWN (TXDES#0 [31]) bit. if TXDMA_OWN (TXDES#0 [31]) = 1, it will move the transmit buffer data into the TX FIFO.

MAC2C: High Priority Transmit Ring Base Address Register (HPTXR_BADR)		
Offset: 2Ch		**Init = 0**
Bit	**Attr.**	**Description**
31	RO	**Reserved (0)**
30:4	RW	**HPTXR_BADR : Base address of the high priority transmit ring [30:4]** **The base address must be 16 bytes aligned**
3 :0	RO	**Reserved (0)**

Offset: 30h	**MAC30: Interrupt Timer Control Register (ITC)**	**Init = 0**
Bit	**R/W**	**Description**
31:16	RO	**Reserved (0)**
15	RW	**TXINT_TIME_SEL** This field defines the period of **TXINT time**. When set, **TXINT time** is **4096 * (MHCLK clock period)**. When cleared, **TXINT time** is **256 * (MHCLK clock period)**.
14:12	RW	**TXINT_THR** This field defines the maximum number of transmit interrupts that can be pending before an interrupt is generated. When TXINT_THR != 0, MAC engine issues a transmit interrupt if the transmit packet number transmitted by MAC engine reaches TXINT_THR. When TXINT_THR = 0 and TXINT_CNT = 0, issuing a transmit interrupt or not depends on TXIC in TXDES#1.
11:8	RW	**TXINT_CNT** This field defines the maximum wait time to issue transmit interrupt after a packet has been transmitted by MAC engine. The time unit is **1 TXINT time**. When TXINT_CNT = 0, the function would be disabled. When TXINT_THR = 0 and TXINT_CNT = 0, issuing a transmit interrupt or not depends on TXIC in TXDES#1.

to next page

from previous page

7	RW	**RXINT_TIME_SEL** This field defines the period of **RXINT time**. When set, **RXINT time** is **4096 * (MHCLK clock period)**. When cleared, **RXINT time** is **256 * (MHCLK clock period)**.
6 :4	RW	**RXINT_THR** This field defines the maximum number of receive interrupts that can be pending before an interrupt is generated. When RXINT_THR != 0, MAC engine issues a receive interrupt if the receive packet number received by MAC engine reaches RXINT_THR. If RXINT_THR = 0 and RXINT_CNT = 0, a receive interrupt will be issued when MAC engine finishes receiving a receive packet.
3 :0	RW	**RXINT_CNT** This field defines the maximum wait time to issue receive interrupt after a packet has been received by MAC engine. The time unit is **1 RXINT time**. When RXINT_CNT = 0, the function is disabled. If RXINT_THR = 0 and RXINT_CNT = 0, a receive interrupt is issued when a packet is received by MAC engine.

Note :
Recommended value = 0000_1010h

The Interrupt Timer Control Register allows the software driver to reduce the number of transmit interrupt (MAC00 [4]) and receive interrupt (MAC00 [0]) by setting the register. This can lower CPU utilization for handling a large number of interrupts.

The register defines two threshold values for the receive packet number and transmit packet number, and two associated timers.
The threshold value defines the maximum number of receive or transmit interrupts that can be pending before an interrupt is generated.
The timer defines the maximum wait time to issue transmit/receive interrupt after a packet has been transmit-ted/received by MAC engine.

The threshold value and timer combination allows for batching of several packets into a single interrupt with a limit for how long it can be pending.
The combination prevents throughput from being impeded in heavy traffic, and the time limit prevents resources from being held for too long in low traffic.

The mitigation mechanism is similar for both receive and transmit interrupts.
There is a counter (TXPKT_CNT) in MAC engine to count the packets transmitted by MAC engine.
When the counter reaches TXINT_THR and TXINT_THR != 0, MAC engine issues transmit interrupt.

There is also a counter (RXPKT_CNT) in MAC engine to count the packets received by MAC engine.
When the counter reaches RXINT_THR and RXINT_THR != 0, MAC engine issues receive interrupt.

TXPKT_CNT is cleared when transmit interrupt is issued.
RXPKT_CNT is cleared when receive interrupt is issued.

† The following is the condition for MAC engine to issue a transmit interrupt.

TXINT_THR	TXINT_CNT	MAC engine Action
0	0	1. Issues transmit interrupt after a packet is transmitted and TXIC of the packet is set. 2. Clears TXPKT_CNT.
0	1	1. Issues transmit interrupt after a packet is transmitted and timer reaches the value of TXINT_CNT. 2. Clears TXPKT_CNT.
1	0	1. Issues transmit interrupt if TXPKT_CNT = TXINT_THR. 2. Clears TXPKT_CNT.
1	1	1. Issues transmit interrupt if the following condition holds: * TXPKT_CNT = TXINT_THR * TXPKT_CNT = 1 and timer reaches the value of TXINT_CNT 2. Clears TXPKT_CNT.

† The following is the condition for MAC engine to issue a receive interrupt.

RXINT_THR	RXINT_CNT	MAC engine Action
0	0	1. Issues receive interrupt after a packet is received by MAC engine. 2. Clears RXPKT_CNT.
0	1	1. Issues receive interrupt after a packet is received by MAC engine and timer reaches the value of RXINT_CNT. 2. Clears RXPKT_CNT.
1	0	1. Issues receive interrupt if RXPKT_CNT = RXINT_THR. 2. Clears TXPKT_CNT.
1	1	1. Issues receive interrupt if the following condition holds: * RXPKT_CNT = RXINT_THR * RXPKT_CNT = 1 and timer reaches the value of RXINT_CNT 2. Clears RXPKT_CNT.

Offset: 34h **MAC34: Automatic Polling Timer Control Register (APTC)** **Init = 0**

Bit	R/W	Description
31:13	RO	**Reserved (0)**
12	RW	**TXPOLL_TIME_SEL** This field defines the period of **TXPOLL time**. When set, **TXPOLL time** is **4096 * (MCLK clock period)**. When cleared, **TXPOLL time** is **256 * (MCLK clock period)**.
11:8	RW	**TXPOLL_CNT** This field defines the period of transmit automatic poll time. The time unit is **1 TXPOLL time**. When TXPOLL_CNT != 0, MAC engine polls the transmit descriptor automatically. If TXPOLL_CNT = 0, MAC engine does not poll the transmit descriptor automatically.

to next page

from previous page

7 :5	RO	**Reserved (0)**
4	RW	**RXPOLL_TIME_SEL** This field defines the period of **RXPOLL time**. When set, **RXPOLL time** is **4096 * (MCLK clock period)**. When cleared, **RXPOLL time** is **256 * (MCLK clock period)**.
3 :0	RW	**RXPOLL_CNT** This field defines the period of receive automatic poll time. The time unit is **1 RXPOLL time**. When RXPOLL_CNT != 0, MAC engine polls the receive descriptor automatically. If RXPOLL_CNT = 0, MAC engine does not poll the receive descriptor automatically.

Note :
Recommended value = 0000_0000h

The Automatic Polling Timer Control Register allows MAC engine to automatically poll the descriptors. This could lower CPU utilization.

If the transmit automatic poll function is enabled, MAC engine automatically polls the transmit descriptor when the transmit automatic poll timer expires.
If the function is disabled, software needs to write Transmit Poll Demand Register (MAC18) to trigger MAC engine to read transmit descriptors after software has prepared the transmit packets in transmit buffers.

If the receive automatic poll function is enabled, MAC engine automatically polls the receive descriptor when the receive automatic poll timer expires.
If the function is disabled, software needs to write Receive Poll Demand Register (MAC1C) to trigger MAC engine to read receive descriptors after software has released the receive descriptors to MAC engine.

MAC38: DMA Burst Length and Arbitration Control Register (DBLAC)

Offset: 38h **Init = 0002_2500h**

Bit	Attr.	Description
31:24	RO	**Reserved (0)**
23	RW	**IFG_INC: IFG(InterFrame Gap) increase** The field defines the increase or decrease of IFG in Ethernet. When IFG_INC=1'b1, the IFG would increase. When IFG_INC=1'b0, the IFG would decrease.

to next page

from previous page

22:20	RW	**IFG_CNT: IFG(InterFrame Gap) count** The field defines the increase or decrease number of IFG in Ethernet. When IFG_INC=1'b1, the IFG would increase. When IFG_INC=1'b0, the IFG would decrease. The time unit is **1 transmit clock in Ethernet**. (8 ns in 1000 Mbps mode, 40ns in 100 Mbps mode, 400 ns in 10 Mbps mode). When in 1000 Mbps mode, if IFG_INC= 1'b0, the value in the field should not be set more than 2. For example: When IFG_CNT=3'h1 and IFG_INC=1'b1 in 1000 Mbps mode, then the IFG = 96+1x8 = 104 ns. When IFG_CNT=3'h1 and IFG_INC=1'b0 in 1000 Mbps mode, then the IFG = 96-1x8 = 88 ns.
19:16	RW	**TXDES_SIZE: Transmit descriptor size** This field defines the transmit descriptor size. Writing 0 to this field is illegal. **The unit is 8 bytes**. **The descriptor size MUST be 16 bytes aligned.**
15:12	RW	**RXDES_SIZE: Receive descriptor size** This field defines the receive descriptor size. Writing 0 to this field is illegal. **The unit is 8 bytes**. **The descriptor size MUST be 16 bytes aligned.**
11:10	RW	**TXBST_SIZE: TXDMA maximum burst size per TXDMA burst** This field sets the maximum size of TXDMA burst. The burst sizes are as follows: 00: 64 bytes 01: 128 bytes 10: 256 bytes 11: 512 bytes
9 :8	RW	**RXBST_SIZE: RXDMA maximum burst size per RXDMA burst** This field sets the maximum size of RXDMA burst. The burst sizes are as follows: 00: 64 bytes 01: 128 bytes 10: 256 bytes 11: 512 bytes
7 :0	RO	**Reserved (0)**
Note : Recommended value = 0002_2500h		

Offset: 3Ch **MAC3C: Debug Status Register (DBG_STS)** **Init = 0**

Bit	R/W	Description
31	RO	**TX_DMABUF_EMPTY** **(for debugging purpose only)**
30	RO	**RX_DMABUF_EMPTY** **(for debugging purpose only)**
29:27	RO	**Reserved (0)**
26:24	RO	**RX_RqFSM[2:0]** **(for debugging purpose only)**

to next page

from previous page

23:20	RO	**RX_OLFSM[3:0]**	**(for debugging purpose only)**
19	RO	**Reserved (0)**	
18:16	RO	**RX_MACFSM[2:0]**	**(for debugging purpose only)**
15:11	RO	**Reserved (0)**	
10:8	RO	**TX_MACFSM[2:0]**	**(for debugging purpose only)**
7 :4	RO	**TX_OLFSM[3:0]**	**(for debugging purpose only)**
3	RO	**Reserved (0)**	
2 :0	RO	**TX_RqFSM[2:0]**	**(for debugging purpose only)**

Offset: 40h **MAC40: Feature Register (FEAR)** **Init = 0000_0000h**

Bit	R/W	Description
31	RO	**Reserved (0)**
30	RO	**Internal loop back** 0: Disable internal loop back 1: Enable internal loop back
29:0	RO	**Reserved (0)**

MAC48: Transmit Priority Arbitration and FIFO Control Register (TPAFCR)

Offset: 48h **Init = 0000_00F1**

Bit	Attr.	Description
31:16	RO	**Reserved (0)**
15:8	RW	**EARLY_RXTHR: Early Receive Threshold** This field specifies the threshold level in the RX FIFO to move packet data to the local memory. When the byte count of the data in the RX FIFO reaches the threshold or there is at least one packet in RX FIFO, hardware would begin to move the packet from RX FIFO to the local memory. Writing 0 to this field indicates that hardware should begin to move the packet after one whole packet has been stored in RX FIFO. The value software programs in this field should be less than RX FIFO size. **The unit is 64 bytes**.
7 :4	RW	**HPKT_THR: High Priority Transmit Packet Threshold**
3 :0	RW	**NPKT_THR: Normal Priority Transmit Packet Threshold**
Note : Recommended value = 0000_02F1h		

Offset: 4Ch **MAC4C: Receive Buffer Size Register (RBSR)** **Init = 0000_0640h**

Bit	R/W	Description
31:14	RO	**Reserved (0)**
13:0	RW	**RXBUF_SIZE: Receive buffer size [13:0]** **The unit is 1 byte**

Offset: 50h	MAC50: MAC Control Register (MACCR)	Init = 0

Bit	R/W	Description
31	RW	**SW_RST: Software reset** Writing 1 to this bit enables software reset. Software reset would last **256** MHCLK clocks, and then be auto-cleared. After setting the SW_RST (MAC50 [31]) = 1 to do the software reset, it need to delay at lease 10 us and then setting the SW_RST (MAC50 [31]) = 1 to do the software reset again. This means the software reset need to be done at least twice.
30:20	RO	**Reserved (0)**
19	RW	**SPEED_100: Speed mode** 1: 100 Mbps 0: 10 Mbps The field and GMAC_MODE (Bit 9) are used to determine MAC engine speed mode. GMAC_MODE / SPEED_100 / Function 0 / 1 / 100 Mbps mode 0 / 0 / 10 Mbps mode 1 / 0 / 1000 Mbps mode 1 / 1 / 1000 Mbps mode This field cannot be software reset. Any change of this bit must also set bit31 to 1 to do the software reset.
18	RW	**DISCARD_CRCERR** Discard the CRC error packet if there is CRC error status in the transmit packet.
17	RW	**RX_BROADPKT_EN** Receive broadcast packets.
16	RW	**RX_MULTIPKT_EN** Receive all multicast packets.
15	RW	**RX_HT_EN** Enable storing incoming packet if the packet passes hash table address filtering and is a multicast packet.
14	RW	**RX_ALLADR** Destination address of incoming packet not checked
13	RW	**JUMBO_LF: Jumbo Long Frame** When set and for MAC #1~#2, received packets with length more than **9596** (**9600** for the VLAN tag packet) bytes are treated as long frames. When set and for MAC #3~#4, received packets with length more than **2044** (**2048** for the VLAN tag packet) bytes are treated as long frames. When cleared, received packets with length more than **1518** (**1522** for the VLAN tag packet) bytes are treated as long frames.
12	RW	**RX_RUNT** Receive the incoming packet even if its length is less than **64** bytes (**68** bytes if VLAN tag is inserted). The incoming packet length must be longer than or equal to **10** bytes.
11	RO	**Reserved (0)**
10	RW	**CRC_APD: Append CRC to transmitted packets**

to next page

from previous page

9	RW	**GMAC_MODE: GMAC mode** If GMAC_MODE = 1, MAC engine is in 1000 Mbps mode; otherwise, MAC engine is in 10/100 Mbps mode. This field cannot be software reset. Any change of this bit must also set bit31 to 1 to do the software reset.
8	RW	**FULLDUP: Full duplex** If FULLDUP = 1, MAC engine is in full duplex mode; otherwise, MAC engine is in half duplex mode.
7	RW	**ENRX_IN_HALFTX** Enable packet reception when transmitting packets in half duplex mode.
6	RO	**Reserved (0)**
5	RW	**HPTXR_EN: High priority transmit ring enable** If HPTXR_EN = 1, software can use the high priority transmit ring; otherwise, software cannot use the high priority transmit ring.
4	RW	**REMOVE_VLAN** Remove VLAN tag from packets received with VLAN tag.
3	RW	**RXMAC_EN: RXMAC enable** When set, enable RXMAC to receive packets.
2	RW	**TXMAC_EN: TXMAC enable** When set, enable TXMAC to transmit packets.
1	RW	**RXDMA_EN: Enable receive DMA channel** If this bit is zero, reception is stopped immediately.
0	RW	**TXDMA_EN: Enable transmit DMA channel** If this bit is zero, transmission is stopped immediately.

Note :
Any chang to the speed of the MAC (MAC50 [9], MAC50 [19]) need to do the software reset (MAC50 [31]).

After setting the SW_RST (MAC50 [31]) = 1 to do the software reset, it need to delay at lease 10 us and then setting the SW_RST (MAC50 [31]) = 1 to do the software reset again.

This means the software reset need to be done at least twice.

Offset: 54h **MAC54: MAC Status Register (MACSR)** **Init = 0**

Bit	R/W	Description
31:12	RO	**Reserved (0)**
11	RW	**COL_EXCEED: Tx-packets lost due to excessive collision (Writing ”1” to clear)**
10	RW	**LATE_COL: Tx-packets lost due to late collision (Writing ”1” to clear)**
9	RW	**TPKT_LOST: Tx-packets lost due to late/excessive collision (Writing ”1” to clear)**
8	RW	**TPKT_OK: Tx-packets be transmitted to Ethernet (Writing ”1” to clear)**
7	RW	**RUNT: Receiver detects a runt packet (Writing ”1” to clear)**
6	RW	**FTL: Receiver detects a frame that is too long (Writing ”1” to clear)**
5	RW	**CRC_ERR: Incoming packet with CRC error (Writing ”1” to clear)**
4	RW	**RPKT_LOST: Rx-packet lost due to RX FIFO full (Writing ”1” to clear)**
3	RW	**RPKT_SAVE: RXMAC finish to receive rx-packet (Writing ”1” to clear)**
2	RW	**COL: Incoming packet had a collision event (Writing ”1” to clear)**
1	RW	**BROADCAST: Incoming packet for broadcast address (Writing ”1” to clear)**
0	RW	**MULTICAST: Incoming packet for multicast address (Writing ”1” to clear)**

Offset: 58h	MAC58: Control Mode Registers (TM)	Init = 0
Bit	**R/W**	**Description**
31:29	RO	**Reserved (0)**
28	RW	**Rq_TxVldDis: Disable MBUS arbiter Tx valid option** Set 1 to be compatible with A0
27	RW	**Rq_RoundRobin: Tx/Rx MBUS arbiter option** 0: Idle in Rx mode 1: Idle in Round Robin Set 1 to be compatible with A0
26:0	RW	**Reserved (0)** For internal test only. Please keep default value 0.
Note : Recommended value = 1800_0000h		

Offset: 68h	MAC68: Flow Control Register (FCR)	Init = 0000_0400h
Bit	**R/W**	**Description**
31:16	RW	**PAUSE_TIME: Pause time in pause frame** The time unit is **1 slot time**.
15:9	RW	**FC_HIGH/FC_LOW** **RX FIFO free space high threshold:** A pause frame is sent with pause time = 0 when RX FIFO free space is larger than the high threshold. **The unit is 256 bytes**, and the default value is 7'h5. **RX FIFO free space low threshold:** A pause frame is sent with pause time set in bits 31~16 when RX FIFO free space is lower than the low threshold. **The unit is 256 bytes**, and the default value is 7'h2. When FC_HTHR_SEL = 1, RX FIFO free space high threshold is selected. When FC_HTHR_SEL = 0, RX FIFO free space low threshold is selected. The value software programs in this field should be less than RX FIFO size.
8	RW	**FC_HTHR_SEL: RX FIFO free space high threshold select** 0: RX FIFO free space low threshold is selected, and the MAC68 [15:9] will show this value. 1: RX FIFO free space high threshold is selected, and the MAC68 [15:9] will show this value.
7	RW	**FC_THR_WREN: RX FIFO free space threshold write enable** When set and FC_HTHR_SEL = 0, write vlaue of the MAC68 [15:9] will be saved to RX FIFO free space low threshold. When set and FC_HTHR_SEL = 1, write vlaue of the MAC68 [15:9] will be saved to RX FIFO free space high threshold.
6 :5	RO	**Reserved (0)**
4	RW	**RX_PAUSE: Receive pause frame (Writing "1" to clear)**
3	RO	**TXPAUSED** Packet transmission paused due to receive pause frame
2	RW	**FCTHR_EN: Enable flow control threshold mode** This bit enables transmit pause frame for high/low threshold.

to next page

from previous page

1	RW	**TX_PAUSE: Transmit pause frame** Software can set this bit to send pause frames. This bit is auto-cleared after the pause frame has been transmitted.
0	RW	**FC_EN: Flow control mode enable**

Offset: 6Ch **MAC6C: Back Pressure Register (BPR)** **Init = 0000_0200h**

Bit	R/W	Description
31:15	RO	**Reserved (0)**
14:8	RW	**BK_LOW: RX FIFO free space low threshold** MAC generates a jam pattern if RX FIFO free space is lower than the low threshold when packets are incoming. **The unit is 256 bytes**, and the default value is 7'h2.
7 :4	RW	**BKJAM_LEN: Back pressure jam length** 0000: 4 bytes 0001: 8 bytes 0010: 16 bytes 0011: 32 bytes 0100: 64 bytes 0101: 128 bytes 0110: 256 bytes 0111: 512 bytes 1000: 1024 bytes 1001: 1518 bytes 1010: 2048 bytes 1011 ~ 1111: 4 bytes
3 :2	RO	**Reserved (0)**
1	RW	**BKADR_MODE: Back pressure address mode** 0: Generate jam pattern when any packet is incoming. 1: Generate jam pattern when packet address matches.
0	RW	**BK_EN: Back pressure mode enable**

Offset: 70h **MAC70: Wake-On-LAN Control Register (WOL_CR)** **Init = 0**

Bit	R/W	Description
31:16	RO	**Reserved (0)**
15	RW	**PWRSAV: Power saving mode** This field is used to determine the current power state. When set, MAC engine enters power saving mode. When cleared, MAC engine is in normal mode.
14:3	RO	**Reserved (0)**
2	RW	**MAGICPKT_EN: Magic packet event enable**
1 :0	RO	**Reserved (0)**

Offset: 74h	MAC74: Wake-On-LAN Status Register (WOL_SR)	Init = 0
Bit	**R/W**	**Description**
31:3	RO	**Reserved (0)**
2	RW	**MAGICPKT_STS: Magic packet event status (Writing "1" to clear)**
1 :0	RO	**Reserved (0)**

Offset: 78h	MAC78: MAC Most Significant Address Register #1 (MAC_MADR1)	Init = 0
Offset: 80h	MAC80: MAC Most Significant Address Register #2 (MAC_MADR2)	Init = 0
Offset: 88h	MAC88: MAC Most Significant Address Register #3 (MAC_MADR3)	Init = 0
Bit	**Attr.**	**Description**
31	RW	**Enable MAC address #1 ~ #3 control bit** 0: Disable MAC address #1 ~ #3 1: Enable MAC address #1 ~ #3
30:16	RO	**Reserved (0)**
15:0	RW	**MAC_MADR1 ~ MAC_MADR3** The most significant 2 bytes of MAC address #1 ~ #3

Offset: 7Ch	MAC7C: MAC Least Significant Address Register #1 (MAC_LADR1)	Init = 0
Offset: 84h	MAC84: MAC Least Significant Address Register #2 (MAC_LADR2)	Init = 0
Offset: 8Ch	MAC8C: MAC Least Significant Address Register #3 (MAC_LADR3)	Init = 0
Bit	**Attr.**	**Description**
31:0	RW	**MAC_LADR1 ~ MAC_LADR3** The least significant 4 bytes of MAC address #1 ~ #3

Offset: 90h	MAC90: Debug Normal Priority Tx-Ring Pointer Register DBG_NTXR_PTR)	Init = 0
Bit	**R/W**	**Description**
31:0	RO	**NPTXR_PTR: Normal Priority Transmit Ring Pointer Register (for debugging purpose only)**

Offset: 94h	MAC94: Debug Rx-Ring Pointer Register (DBG_RXR_PTR)	Init = 0
Bit	**R/W**	**Description**
31:0	RO	**RXR_PTR: Receive Ring Pointer Register (for debugging purpose only)**

Offset: 9Ch	MAC9C: Debug High Priority Tx-Ring Pointer Register (DBG_HTXR_PTR)	Init = 0
Bit	**R/W**	**Description**
31:0	RO	**HPTXR_PTR: High Priority Transmit Ring Pointer Register (for debugging purpose only)**

Offset: A0h	MACA0: Debug Tx-Counter Register #0 (DBG_TXCNT0)	Init = 0
Bit	**R/W**	**Description**
31:0	RO	**TXCNT_PKT_OUT: Packets transmitted to Ethernet successfully (for debugging purpose only)**

Offset: A4h	MACA4: Debug Tx-Counter Register #1 (DBG_TXCNT1)	Init = 0
Bit	**R/W**	**Description**
31:26	RO	**Reserved (0)**
25:16	RO	**TXCNT_MCOL: Packets transmitted OK with 2∼15 collisions (for debugging purpose only)**
15:10	RO	**Reserved (0)**
9 :0	RO	**TXCNT_SCOL: Packets transmitted OK with single collision (for debugging purpose only)**

Offset: A8h	MACA8: Debug Tx-Counter Register #2 (DBG_TXCNT2)	Init = 0
Bit	**R/W**	**Description**
31:26	RO	**Reserved (0)**
25:16	RO	**TXCNT_LCOL: Packets failed in transmission (due to late collision) (for debugging purpose only)**
15:10	RO	**Reserved (0)**
9 :0	RO	**TXCNT_ECOL: Packets failed in transmission (due to collision count >= 16) (for debugging purpose only)**

Offset: ACh	MACAC: Debug Rx-Counter Register #0 (DBG_RXCNT0)	Init = 0
Bit	**R/W**	**Description**
31:16	RO	**Reserved (0)**
15:0	RO	**RXCNT_PKT_FLUSH: Flushed packets when the descriptor is empty (for debugging purpose only)**

Offset: B0h	MACB0: Debug Rx-Counter Register #1 (DBG_RXCNT1)	Init = 0
Bit	**R/W**	**Description**
31:0	RO	**RXCNT_PKT_IN: Packets received to RX FIFO successfully (for debugging purpose only)**

Offset: B4h	MACB4: Debug Rx-Counter Register #2 (DBG_RXCNT2)	Init = 0
Bit	**R/W**	**Description**
31:0	RO	**RXCNT_PKT_WR: Packets received to RX buffer successfully (for debugging purpose only)**

Offset: B8h	MACB8: Debug Rx-Counter Register #3 (DBG_RXCNT3)	Init = 0
Bit	**R/W**	**Description**
31:26	RO	**Reserved (0)**
25:16	RO	**RXCNT_COL: Receive collision counter (for debugging purpose only)**
15:10	RO	**Reserved (0)**
9 :0	RO	**RXCNT_BUFOF: Loss of received packets (due to RX FIFO full) (for debugging purpose only)**

Offset: BCh	MACBC: Debug Rx-Counter Register #4 (DBG_RXCNT4)	Init = 0
Bit	**R/W**	**Description**
31:26	RO	**Reserved (0)**
25:16	RO	**RXCNT_CRCERR_ODD: Discarded CRC error packets with odd nibble.** (for debugging purpose only)
15:10	RO	**Reserved (0)**
9 :0	RO	**RXCNT_CRCERR_EVEN: Discarded CRC error packets with even nibble.** (for debugging purpose only)

Offset: C0h	MACC0: Debug Rx-Counter Register #5 (DBG_RXCNT5)	Init = 0
Bit	**R/W**	**Description**
31:26	RO	**Reserved (0)**
25:16	RO	**RXCNT_MUL: Received multicast packets** (for debugging purpose only)
15:10	RO	**Reserved (0)**
9 :0	RO	**RXCNT_BRO: Received broadcast packets** (for debugging purpose only)

Offset: C4h	MACC4: Debug Rx-Counter Register #6 (DBG_RXCNT6)	Init = 0
Bit	**R/W**	**Description**
31:26	RO	**Reserved (0)**
25:16	RO	**RXCNT_FTL: Received FTL packets** (for debugging purpose only)
15:10	RO	**Reserved (0)**
9 :0	RO	**RXCNT_RUNT: Received runt packets** (for debugging purpose only)

Offset: C8h	MACC8: Debug Rx-Counter Register #7 (DBG_RXCNT7)	Init = 0
Bit	**R/W**	**Description**
31:10	RO	**Reserved (0)**
9 :0	RO	**RXCNT_PAUSE: Receive pause frame counter** (for debugging purpose only)

Offset: D0h	MACD0: MAC Most Significant Address Register #4 (MAC_MADR4)	Init = 0
Offset: D8h	MACD8: MAC Most Significant Address Register #5 (MAC_MADR5)	Init = 0
Offset: E0h	MACE0: MAC Most Significant Address Register #6 (MAC_MADR6)	Init = 0
Offset: E8h	MACE8: MAC Most Significant Address Register #7 (MAC_MADR7)	Init = 0
Bit	**Attr.**	**Description**
31	RW	**Enable MAC address #4 ~ #7 control bit** 0: Disable MAC address #4 ~ #7 1: Enable MAC address #4 ~ #7
30:16	RO	**Reserved (0)**
15:0	RW	**MAC_MADR4 ~ MAC_MADR7** The most significant 2 bytes of MAC address #4 ~ #7

Offset: D4h	MACD4: MAC Least Significant Address Register #4 (MAC_LADR4)	Init = 0
Offset: DCh	MACDC: MAC Least Significant Address Register #5 (MAC_LADR5)	Init = 0
Offset: E4h	MACE4: MAC Least Significant Address Register #6 (MAC_LADR6)	Init = 0
Offset: ECh	MACEC: MAC Least Significant Address Register #7 (MAC_LADR7)	Init = 0

Bit	Attr.	Description
31:0	RW	**MAC_LADR4 ~ MAC_LADR7** The least significant 4 bytes of MAC address #4 ~ #7

Offset: F0h	MACF0: Filter Control Register #0 (FL_CTRL0)	Init = 0

Bit	R/W	Description
31:28	RO	**Reserved (0)**
27:24	RW	**Enable source address filter set control bit** 0xxx: Disable source address filter set #3 1xxx: Enable source address filter set #3 x0xx: Disable source address filter set #2 x1xx: Enable source address filter set #2 xx0x: Disable source address filter set #1 xx1x: Enable source address filter set #1 xxx0: Disable source address filter set #0 xxx1: Enable source address filter set #0
23:21	RW	**Enable source port filter set #5 ~ #7 control bit**
20	RW	**Enable source port filter set #4 control bit**
19:16	RW	**Enable source port filter set #0 ~ #3 control bit**
15:8	RW	**Enable destination port filter set #0 ~ #7 control bit** Bit16~23 / Bit8~15 / Function 0 / 0 / Disable source & destination port filter set #0 ~ #7 0 / 1 / Only enable destination port filter set #0 ~ #7 1 / 0 / Only enable source port filter set #0 ~ #7 1 / 1 / Enable source & destination port filter set #0 ~ #7
7 :0	RW	**Enable protocol filter set control bit** 0xxxxxxx: Disable protocol filter set #7 1xxxxxxx: Enable protocol filter set #7 x0xxxxxx: Disable protocol filter set #6 x1xxxxxx: Enable protocol filter set #6 xx0xxxxx: Disable protocol filter set #5 xx1xxxxx: Enable protocol filter set #5 xxx0xxxx: Disable protocol filter set #4 xxx1xxxx: Enable protocol filter set #4 xxxx0xxx: Disable protocol filter set #3 xxxx1xxx: Enable protocol filter set #3 xxxxx0xx: Disable protocol filter set #2 xxxxx1xx: Enable protocol filter set #2 xxxxxx0x: Disable protocol filter set #1 xxxxxx1x: Enable protocol filter set #1 xxxxxxx0: Disable protocol filter set #0 xxxxxxx1: Enable protocol filter set #0

Offset: F4h	MACF4: Filter Control Register #1 (FL_CTRL1)	Init = 0
Bit	**R/W**	**Description**
31:8	RO	**Reserved (0)**
7 :0	RW	**Enable type filter set control bit** 0xxxxxxx: Disable type filter set #7 1xxxxxxx: Enable type filter set #7 x0xxxxxx: Disable type filter set #6 x1xxxxxx: Enable type filter set #6 xx0xxxxx: Disable type filter set #5 xx1xxxxx: Enable type filter set #5 xxx0xxxx: Disable type filter set #4 xxx1xxxx: Enable type filter set #4 xxxx0xxx: Disable type filter set #3 xxxx1xxx: Enable type filter set #3 xxxxx0xx: Disable type filter set #2 xxxxx1xx: Enable type filter set #2 xxxxxx0x: Disable type filter set #1 xxxxxx1x: Enable type filter set #1 xxxxxxx0: Disable type filter set #0 xxxxxxx1: Enable type filter set #0

Offset: F8h	MACF8: Protocol Filter Byte Register #0 (FL_PROL0)	Init = X
Bit	**R/W**	**Description**
31:24	RW	**The protocol filter set #3 byte data**
23:16	RW	**The protocol filter set #2 byte data**
15:8	RW	**The protocol filter set #1 byte data**
7 :0	RW	**The protocol filter set #0 byte data**
Note : When the protocol field of the received frame match this register, the frame will be dropped.		

Offset: FCh	MACFC: Protocol Filter Byte Register #1 (FL_PROL1)	Init = X
Bit	**R/W**	**Description**
31:24	RW	**The protocol filter set #7 byte data**
23:16	RW	**The protocol filter set #6 byte data**
15:8	RW	**The protocol filter set #5 byte data**
7 :0	RW	**The protocol filter set #4 byte data**
Note : When the protocol field of the received frame match this register, the frame will be dropped.		

Offset: 100h MAC100: Source & Destination Port Filter Word Register #0 (FL_SDPORTW0) Init = X
Offset: 104h MAC104: Source & Destination Port Filter Word Register #1 (FL_SDPORTW1) Init = X
Offset: 108h MAC108: Source & Destination Port Filter Word Register #2 (FL_SDPORTW2) Init = X
Offset: 10Ch MAC10C: Source & Destination Port Filter Word Register #3 (FL_SDPORTW3) Init = X
Offset: 110h MAC110: Source & Destination Port Filter Word Register #4 (FL_SDPORTW4) Init = X
Offset: 114h MAC114: Source & Destination Port Filter Word Register #5 (FL_SDPORTW5) Init = X
Offset: 118h MAC118: Source & Destination Port Filter Word Register #6 (FL_SDPORTW6) Init = X
Offset: 11Ch MAC11C: Source & Destination Port Filter Word Register #7 (FL_SDPORTW7) Init = X

Bit	Attr.	Description
31:16	RW	**The source port filter set #0 ~ #7 word data**
15:0	RW	**The destination port filter set #0 ~ #7 word data**

Note :
When the source or the destination port field of the received frame match this register, the frame will be dropped.

Offset: 120h MAC120: Source Address Filter Most Significant Address Register #0 (FL_SAM0) Init = X
Offset: 128h MAC128: Source Address Filter Most Significant Address Register #1 (FL_SAM1) Init = X
Offset: 130h MAC130: Source Address Filter Most Significant Address Register #2 (FL_SAM2) Init = X
Offset: 138h MAC138: Source Address Filter Most Significant Address Register #3 (FL_SAM3) Init = X

Bit	Attr.	Description
31:16	RO	**Reserved (0)**
15:0	RW	**FL_SAM0 ~ FL_SAM3** The most significant 2 bytes of source address filter set #0 ~ #3

Note :
When the MACF0 [27:24] != 0, the MAC will only receive those frames which the source address field match this register.

Offset: 124h MAC124: Source Address Filter Least Significant Address Register #0 (FL_SAL0) Init = X
Offset: 12Ch MAC12C: Source Address Filter Least Significant Address Register #1 (FL_SAL1) Init = X
Offset: 134h MAC134: Source Address Filter Least Significant Address Register #2 (FL_SAL2) Init = X
Offset: 13Ch MAC13C: Source Address Filter Least Significant Address Register #3 (FL_SAL3) Init = X

Bit	Attr.	Description
31:0	RW	**FL_SAL0 ~ FL_SAL3** The least significant 4 bytes of source address filter set #0 ~ #3

Note :
When the MACF0 [27:24] != 0, the MAC will only receive those frames which the source address field match this register.

Offset: 140h MAC140: Type Filter Word Register #0 (FL_TYPE0) Init = X

Bit	R/W	Description
31:16	RW	**The type filter set #1 word data**
15:0	RW	**The type filter set #0 word data**

Note :
When the type field of the received frame match this register, the frame will be dropped.

Offset: 144h	MAC144: Type Filter Word Register #1 (FL_TYPE1)	Init = X

Bit	R/W	Description
31:16	RW	**The type filter set #3 word data**
15:0	RW	**The type filter set #2 word data**

Note :
When the type field of the received frame match this register, the frame will be dropped.

Offset: 148h	MAC148: Type Filter Word Register #2 (FL_TYPE2)	Init = X

Bit	R/W	Description
31:16	RW	**The type filter set #5 word data**
15:0	RW	**The type filter set #4 word data**

Note :
When the type field of the received frame match this register, the frame will be dropped.

Offset: 14Ch	MAC14C: Type Filter Word Register #3 (FL_TYPE3)	Init = X

Bit	R/W	Description
31:16	RW	**The type filter set #7 word data**
15:0	RW	**The type filter set #6 word data**

Note :
When the type field of the received frame match this register, the frame will be dropped.

16.4 Function Description

16.4.1 Transmit Descriptor

MAC engine uses a descriptor ring to manage transmit buffers. The transmit descriptors and data buffers are all located in the local memory. MAC engine moves the transmit packet data from the transmit buffers in the local memory to the TX FIFO inside MAC engine and then transmits the packet to Ethernet. The transmit descriptors reside in the local memory act as pointers to the transmit buffers.

Each transmit descriptor contains a transmit buffer. A transmit buffer consists of either an entire frame or part of a frame, but not multiple frames. The transmit descriptor contains transmit buffer status and the transmit buffer can only contain the transmit data. MAC engine supports two descriptor rings for transmission. These descriptor rings are normal priority transmit ring and high priority transmit ring. The normal priority transmit ring is for normal packet transmission; the high priority transmit ring is for high priority packet transmission. Higher priority packets can be put into the high priority transmit ring for quicker transmission.

- **The size of each transmit descriptor must be 16 bytes aligned**.
- **The start address of each transmit descriptor must be 16 bytes aligned**.
- The maximum transmit packet size including CRC is **9604** bytes for MAC #1∼#2.
- The maximum transmit packet size including CRC is **2052** bytes for MAC #3∼#4.
- LLC packet is IEEE 802.3/802.2/SNAP format packet.

Offset: 00h		**TXDES#0: Control Bits and Ownership Information.**	**Init = X**
Bit	**R/W**	**Description**	
31	RW	**TXDMA_OWN: TXDMA ownership bit** 0: The software owns the descriptor. 1: The descriptor is owned by the MAC engine. MAC engine clears this bit when it completes the frame transmission.	
30	RW	**EDOTR: End Descriptor of Transmit Ring** When set, it indicates that the descriptor is the last descriptor of the transmit ring.	
29	RW	**FTS: First Transmit Segment descriptor** When set, it indicates that this is the first descriptor of a TX packet.	
28	RW	**LTS: Last Transmit Segment descriptor.** When set, it indicates that this is the last descriptor of a TX packet.	
27:20	RW	**Reserved (0)**	
19	RW	**CRC_ERR: CRC error** When CRC_ERR=1 and DISCARD_CRCERR (MAC50 [18] = 1), TXDMA would discard the transmit packet, not send it to Ethernet.	
18:14	RW	**Reserved (0)**	

to next page

from previous page

13:0	RW	**TXBUF_SIZE: Transmit buffer size in byte** The transmit buffer size can not be zero. For MAC #1~#2: Maximum byte count / TXDES#0[13:0] / Transmit packet TXDES#1[16] (INS_VLAN) = 0 / 9600 / 9604 TXDES#1[16] (INS_VLAN) = 1 / 9596 / 9604 For MAC #3~#4: Maximum byte count / TXDES#0[13:0] / Transmit packet TXDES#1[16] (INS_VLAN) = 0 / 2048 / 2052 TXDES#1[16] (INS_VLAN) = 1 / 2044 / 2052
Note : **Bits 27 ~ 14 are valid only when the FTS = 1.**		

Offset: 04h TXDES#1: VLAN Control Bits and VLAN Tag Control Information. Init = X

Bit	R/W	Description
31	RW	**TXIC: Transmit Interrupt on Completion** When set, the MAC engine would assert transmit interrupt after the present frame has been transmitted. It is valid when MAC30 [14:8] = 0 (TXINT_THR, TXINT_CNT).
30	RW	**TX2FIC: Transmit to FIFO Interrupt on Completion** When set, the MAC engine would assert transmit interrupt after the present frame has been moved into the TX FIFO.
29:20	RW	**Reserved (0)**
19	RW	**IPCS_EN: IP checksum offload enable** When set, MAC engine would offload IP checksum.
18	RW	**UDPCS_EN: UDP checksum offload enable** When set, MAC engine would offload UDP checksum.
17	RW	**TCPCS_EN: TCP checksum offload enable** When set, MAC engine would offload TCP checksum.
16	RW	**INS_VLAN: Insert VLAN Tag** When set, 0x8100 (IEEE 802.1Q VLAN Tag Type) is inserted after source address, and 2 bytes VLAN_TAGC are inserted after IEEE 802.1Q VLAN Tag Type. When clear, the packet content would not be changed when transmitting to network.
15:0	RW	**VLAN_TAGC: VLAN Tag Control Information** The 2-byte VLAN Tag Control Information contains information, from the upper layer, of user priority, canonical format indicator, and VLAN ID. Please refer to IEEE 802.1Q for more VLAN tag information. Bits / Function 15 - 13 / User priority 12 / CFI (Canonical Format Indicator) 11 - 0 / VID (VLAN Identifier)
Note : **Bits 31 ~ 0 are valid only when the FTS = 1.**		

Offset: 08h	TXDES#2: Reserved	Init = X
Bit	**R/W**	**Description**
31:0	RW	**Reserved (0)**

Offset: 0Ch	TXDES#3: Transmit Buffer Base Address	Init = X
Bit	**R/W**	**Description**
31	RW	**Reserved (0)**
30:0	RW	**TXBUF_BADR: Transmit buffer base address [30:0]**

16.4.2 Receive Descriptor

MAC engine uses a descriptor ring to manage the receive buffers. The receive descriptors and data buffers are all located in the local memory. MAC engine first stores the packet received from the network in the RX FIFO and then moves the received packet data to the receive buffers in the local memory. The receive descriptors reside in the local memory act as pointers to the receive buffers.

There is a descriptor ring for reception. The base address of the receive ring is in the Receive Ring Base Address Register (MAC24). Each receive descriptor contains a receive buffer. A receive buffer consists of either an entire frame or part of a frame, but not multiple frames. The receive descriptor contains receive buffer status and the receive buffer can only contain the receive packet data.

MAC engine supports the receive buffer base address as 1 byte aligned for the zero-copy feature.

- **The size of each receive descriptor must be 16 bytes aligned**.
- **The start address of each receive descriptor must be 16 bytes aligned**.
- The maximum receive packet size including CRC is **9596** (**9600** for the VLAN tag packet) bytes for MAC #1∼#2.
- The maximum receive packet size including CRC is **2044** (**2048** for the VLAN tag packet) bytes for MAC #3∼#4.
- LLC packet is IEEE 802.3/802.2/SNAP format packet.
- MAC engine doesn't support the following tow packets to do checksum offload; they are IEEE 802.3 with IEEE 802.2 packet and IEEE 802.3 with 802.1Q and 802.2 packet.

Offset: 00h	RXDES#0: Frame Status and Descriptor Ownership Information.	Init = X
Bit	**R/W**	**Description**
31	RW	**RXPKT_RDY: RX packet ready** 0: The descriptor is owned by the MAC engine. 1: The software owns the descriptor. MAC engine set this bit when it completes the frame reception or when the receive buffer of the receive descriptor is full.
30	RW	**EDORR: End Descriptor of Receive Ring** When set, it indicates that the descriptor is the last descriptor of the receive ring.
29	RO	**FRS: First Receive Segment descriptor** When set, it indicates that this is the first descriptor of a received packet.
28	RO	**LRS: Last Receive Segment descriptor.** When set, it indicates that this is the last descriptor of a received packet.

to next page

from previous page

<table>
<tr><td>27:26</td><td>RO</td><td>Reserved (0)</td></tr>
<tr><td>25</td><td>RO</td><td>PAUSE_FRMAE: PAUSE_FRAME Pause frame
When set, it indicates that the receive packet is a pause frame.</td></tr>
<tr><td>24</td><td>RO</td><td>PAUSE_OPCODE: Pause frame OP code
When set, it indicates that there is pause frame OP code in the receive packet.</td></tr>
<tr><td>23</td><td>RO</td><td>FIFO_FULL: FIFO full
When set, it indicates that RX FIFO is full when the packet is received.</td></tr>
<tr><td>22</td><td>RO</td><td>RX_ODD_NB: Receive Odd Nibbles
When set, it indicates receiving a packet with odd nibbles.</td></tr>
<tr><td>21</td><td>RO</td><td>RUNT: Runt packet
When set, it indicates that the received packet length is less than 64 bytes (68 bytes if VLAN tag is inserted).</td></tr>
<tr><td>20</td><td>RO</td><td>FTL: Frame Too Long
When set, it indicates that the received packet length exceeds the long frame length.

If JUMBO_LF=1 and for MAC #1∼#2, the long frame length is 9596 (9600 for the VLAN tag packet) bytes.
If JUMBO_LF=1 and for MAC #3∼#4, the long frame length is 2044 (2048 for the VLAN tag packet) bytes.

If JUMBO_LF=0, the long frame length is 1518 (1522 for the VLAN tag packet) bytes.</td></tr>
<tr><td>19</td><td>RO</td><td>CRC_ERR: CRC error
When set, it indicates that the CRC error occurs on the received packet.</td></tr>
<tr><td>18</td><td>RO</td><td>RX_ERR: Receive error
When set, it indicates that receive error happens when receiving a packet.</td></tr>
<tr><td>17</td><td>RO</td><td>BROADCAST: Broadcast frame.
When set, it indicates that the received packet is a broadcast frame.</td></tr>
<tr><td>16</td><td>RO</td><td>MULTICAST: Multicast frame.
When set, it indicates that the received packet is a multicast frame.</td></tr>
<tr><td>15</td><td>RO</td><td>UDP_OPT_CHKSUM: UDP Optional Checksum
It indicates that the UDP header with optional checksum.</td></tr>
<tr><td>14</td><td>RO</td><td>Reserved (0)</td></tr>
<tr><td>13:0</td><td>RO</td><td>VDBC: valid data byte count.
The field indicates the valid data in the receive buffer.
The unit is 1 byte.

For MAC #1∼#2:
<table>
<tr><th>Maximum byte count</th><th>Receive packet</th><th>RXDES#0[13:0]</th></tr>
<tr><td>Without VLAN tag</td><td>1518 / 9596</td><td>1518 / 9596</td></tr>
<tr><td>With VLAN tag and (MAC50[4](REMOVE_VLAN) = 0)</td><td>1522 / 9600</td><td>1522 / 9600</td></tr>
<tr><td>With VLAN tag and (MAC50[4](REMOVE_VLAN) = 1)</td><td>1522 / 9600</td><td>1518 / 9596</td></tr>
</table>
For MAC #3∼#4:
<table>
<tr><th>Maximum byte count</th><th>Receive packet</th><th>RXDES#0[13:0]</th></tr>
<tr><td>Without VLAN tag</td><td>1518 / 2044</td><td>1518 / 2044</td></tr>
<tr><td>With VLAN tag and (MAC50[4](REMOVE_VLAN) = 0)</td><td>1522 / 2048</td><td>1522 / 2048</td></tr>
<tr><td>With VLAN tag and (MAC50[4](REMOVE_VLAN) = 1)</td><td>1522 / 2048</td><td>1518 / 2044</td></tr>
</table></td></tr>
</table>

to next page

from previous page

Note :
Bits 27 ~ 14 are valid only when the FRS = 1.

Offset: 04h RXDES#1: VLAN Status Bits and VLAN Tag Control Information. Init = X

Bit	R/W	Description
31:28	RO	**Reserved (0)**
27	RO	**IPCS_FAIL: IP checksum failure** When set, MAC engine detects IP checksum failure.
26	RO	**UDPCS_FAIL: UDP checksum failure** When set, MAC engine detects UDP checksum failure.
25	RO	**TCPCS_FAIL: TCP checksum failure** When set, MAC engine detects TCP checksum failure.
24	RO	**VLAN_AVA: VLAN Tag Available** When set, the receive packet is a packet with IEEE 802.1Q VLAN Tag Type.
23	RO	**DF: Datagram Fragment** When set, the IP packet is not fragment. When clear, the IP packet may fragment. Checksum status is valid only when DF=1.
22	RO	**LLC_PKT: LLC packet** When set, it indicates that the receive packet is LLC packet.
21:20	RO	**PROTL_TYPE: Protocol Type** These 2 bits indicate which protocol in the receive packet. 00: Not IP protocol. 01: IP protocol. 10: TCP/IP protocol. 11: UDP/IP protocol.
19	RO	**IPv6: IPv6 packet** When set, it indicates that the frame is IPv6 packet
18	RO	**SKIP_FRAME: Skip Frame** When set, it indicates that the frame is skipped
17	RO	**CHKSUM_LEN_FAIL: Checksum Length Fail** When set, it indicates that the checksum length fail
16	RO	**Reserved (0)**
15:0	RO	**VLAN_TAGC: VLAN Tag Control Information** If the receive packet contains VLAN tag. MAC engine would extracts 4 bytes from the receive packet. The 4 bytes data contains 0x8100 and 2 bytes VLAN Tag Control Information. MAC engine would move the 2 bytes VLAN Tag Control Information to this field. The 2-byte VLAN Tag Control Information contains information, from the upper layer, of user priority, canonical format indicator, and VLAN ID. Please refer to IEEE 802.1Q for more VLAN tag information. Bits | Function 15 - 13 | User priority 12 | CFI (Canonical Format Indicator) 11 - 0 | VID (VLAN Identifier)

to next page

from previous page

Note :
Bits 31 ∼ 0 are valid only when the FRS = 1.

Offset: 08h		**RXDES#2: Length/Type Filed**	**Init = X**
Bit	**R/W**	**Description**	
31:16	RO	**Reserved (0)**	
15:0	RO	**Length/Type: The Length/Type field of a MAC frame**	
Note : **Bits 31 ∼ 0 are valid only when the FRS = 1.**			

Offset: 0Ch		**RXDES#3: Receive Buffer Base Address**	**Init = X**
Bit	**R/W**	**Description**	
31	RO	**Reserved (0)**	
30:0	RW	**RXBUF_BADR: Receive buffer base address [30:0]**	

16.4.3 Transmitting Packet

When software wants to transmit a packet to the Ethernet, it moves the packet data into the transmit buffer first. Then software writes the packet length and position into the transmit descriptor and triggers MAC engine to send the packet. After the entire packet has been moved into the TX FIFO, MAC engine begins to transmit it to the Ethernet. When the packet has been transmitted, MAC engine asserts interrupt to notify software that the packet has been transmitted successfully. Higher priority packets can be put into the high priority descriptor for quicker transmission.

16.4.4 Receiving Packet

When there is an incoming packet, MAC engine first saves the received packet in the RX FIFO if the address check result is correct. After the incoming packet is successfully saved in RX FIFO, MAC engine initiates Direct Memory Access (DMA) function to move the received packet data from the RX FIFO to the local memory. Then MAC engine asserts interrupt to notify software that the packet has been received successfully.

16.4.5 Ethernet Address Filtering

- ALL: bit 14 of MAC Control Register (MAC50 [14])
- MULTI: bit 16 of MAC Control Register (MAC50 [16])
- BROAD: bit 17 of MAC Control Register (MAC50 [17])
- HT: bit 15 of MAC Control Register (MAC50 [15])
- MAC_ADR: MAC Addres Register (MAC08 & MAC0C)
- MAHT: Multicast Address Hash Table Register (MAC10 & MAC14)
- multicast: Multicast Address
- broadcast: Broadcast Address

Group	ALL	MULTI	BROAD	HT	MAC_ADR	MAHT	multicast	broadcast
A	0	0	0	0	**O**	**X**	**X**	**X**
B	0	0	0	1	**O**	**O**	**X**	**O (MAC10 [0] == 1)**
C	0	0	1	0	**O**	**X**	**X**	**O**
D	0	0	1	1	**O**	**O**	**X**	**O**
E	0	1	x	x	**O**	**O**	**O**	**O**
F	1	x	x	x	**O**	**O**	**O**	**O**

”O” : MAC Controller receives a frame whose destination address exactly matches the register/address listed in the column.
”X” : MAC Controller does not compare destination address with the register/address listed in the column.

16.4.6 MII Management Interface

MAC contains an MII Management Interface for an MII compliant PHY device. This allows control and status parameters to be passed between MAC and PHY by MDIO and MDC, thereby reducing the number of control pins required for PHY mode control.

The protocol consists of the bit stream that is sampled at rising edge of the MDC, the bit stream format is described below.

Clause 22	PRE	ST	OP	PHYAD	REGAD	TA	DATA	IDLE
Write	1...1	01	01	AAAAA	RRRRR	10	D..D(16)	Z
Read	1...1	01	10	AAAAA	RRRRR	Z0	D..D(16)	Z

Clause 45	PRE	ST	OP	PRTAD	DEVAD	TA	DATA	IDLE
Address	1...1	00	00	PPPPP	EEEEE	10	A..A(16)	Z
Write	1...1	00	01	PPPPP	EEEEE	10	D..D(16)	Z
Read	1...1	00	11	PPPPP	EEEEE	Z0	D..D(16)	Z
Read Inc.	1...1	00	10	PPPPP	EEEEE	Z0	D..D(16)	Z

16.5 Initialization

16.5.1 Frame Transmitting Procedure

The frame transmitting procedure is as follows:

Initialization:

1. Set GMAC_MODE (MAC50 [9]) and SPEED_100 (MAC50 [19]) to proper setting.
2. Set SW_RST (MAC50 [31]) = 1 to do the software reset.
3. Delay at least 10 us and then set SW_RST (MAC50 [31]) = 1 to do the software reset again. It takes about **256** MHCLK clocks for hardware to finish the software reset. When the hardware finished the software reset, the SW_RST (MAC50 [31]) will be cleared to 0.
4. Allocate the local memory for the transmit descriptor ring and transmit buffer.
5. Initialize the transmit descriptor ring.
6. Set the Normal Priority Transmit Ring Base Address Register (MAC20) to the base address of the normal priority transmit descriptor ring in the local memory.
7. Set the High Priority Transmit Ring Base Address Register (MAC2C) to the base address of the high priority transmit descriptor ring in the local memory if necessary.
8. Set Interrupt Enable Register (MAC04).
9. Set MAC Address Register (MAC08).
10. Set Multicast Address Hash Table Register (MAC10).
11. Set Interrupt Timer Control Register (MAC30) to select the manner of the transmit interrupt.
12. Set Automatic Polling Timer Control Register (MAC34) to select the manner of transmit poll.
13. Set Transmit Priority Arbitration and FIFO Control Register (MAC48) to set transmit priority arbitration.
14. Set DMA Burst Length and Arbitration Control Register (MAC38) to set proper TX/RX descriptor size and DMA burst length.
15. Set MAC Control Register (MAC50) to set valid configuration for MAC engine and to enable transmit channel.

Transmit procedure:

1. Software checks if the remainder of the normal priority transmit descriptors is enough for the next packet transmission. If not, software needs to wait until the transmit descriptors are enough.
2. Prepare the transmit packet data to the transmit buffer.
3. Set the normal priority transmit descriptor.
4. Write the Normal Priority Transmit Poll Demand Register (MAC18) to trigger MAC engine to poll the transmit descriptor if necessary when the packet is put in the normal priority transmit ring.
5. Wait for interrupt.
6. When interrupt occurs, software checks if it is a transmit interrupt. If MAC00 [4] = 1, it means the packet has been transmitted to network successfully. If MAC00 [7] = 1, it means the packet has been aborted during transmission due to late collision or excessive collision .
7. Steps 1 through 6 are for normal packets in the normal priority transmit ring. If software wants to transmit high priority packets, repeat these steps for the high priority transmit ring.

Note:

1. When setting the transmit descriptor, **TXDES#0 must be set last**.
2. When preparing a transmit packet which contains more than one transmit descriptors, **the first transmit descriptor must be the last set descriptor of the transmit packet**.

16.5.2 Frame Receiving Procedure

The frame receiving procedure is as follows:

Initialization:

1. Set GMAC_MODE (MAC50 [9]) and SPEED_100 (MAC50 [19]) to proper setting.
2. Set SW_RST (MAC50 [31]) = 1 to do the software reset.
3. Delay at least 10 us and then set SW_RST (MAC50 [31]) = 1 to do the software reset again. It takes about **256** MHCLK clocks for hardware to finish the software reset. When the hardware finished the software reset, the SW_RST (MAC50 [31]) will be cleared to 0.
4. Allocate the local memory for the receive descriptor ring and receive buffer.
5. Initialize the receive descriptor ring.
6. Set Receive Ring Base Address Register (MAC24) to the base address of the receive descriptor ring in the local memory.
7. Set Interrupt Enable Register (MAC04).
8. Set MAC Address Register (MAC08).
9. Set Multicast Address Hash Table Register (MAC10).
10. Set Interrupt Timer Control Register (MAC30) to select the manner of the receive interrupt.
11. Set Automatic Polling Timer Control Register (MAC34) to select the manner of receive poll.
12. Set Transmit Priority Arbitration and FIFO Control Register (MAC48) to set transmit priority arbitration.
13. Set DMA Burst Length and Arbitration Control Register (MAC38) to set proper TX/RX descriptor size and DMA burst length.
14. Set MAC Control Register (MAC50) to set valid configuration for MAC engine and to enable receive channel.
15. Write Receive Poll Demand Register (MAC1C) to trigger MAC engine to poll the receive descriptor.

Receive procedures:

1. Wait for interrupt.
2. When interrupt occurs, software checks if it is a receive interrupt. If MAC00 [0] = 1, it means the packet has been moved to the receive buffer successfully. Then software needs to fetch the receive descriptor to get the receive packet until the owner bit of the next receive descriptor does not belong to software.
3. Software releases the receive descriptors to MAC engine after accessing the received packet.
4. If the receive automatic poll function is disabled, software needs to write Receive Poll Demand Register (MAC1C) to trigger MAC engine to poll the receive descriptor.

17 Ethernet MDC/MDIO Bus Controller (PMI)

17.1 Overview

Base address of PMI = 0x1E65_0000
Physical address = (Base address of PMI) + Offset

PMI00: MDC/MDIO #0 Control Register
PMI04: MDC/MDIO #0 Read Data Register
PMI08: MDC/MDIO #1 Control Register
PMI0C: MDC/MDIO #1 Read Data Register
PMI10: MDC/MDIO #2 Control Register
PMI14: MDC/MDIO #2 Read Data Register
PMI18: MDC/MDIO #3 Control Register
PMI1C: MDC/MDIO #3 Read Data Register
PMI7C: PHY Interrup Activity Level

17.2 Registers : Base address of PMI = 0x1E65:0000

Offset: 00h	**PMI00: MDC/MDIO #0 Control Register**	**Init = X**
Offset: 08h	**PMI08: MDC/MDIO #1 Control Register**	**Init = X**
Offset: 10h	**PMI10: MDC/MDIO #2 Control Register**	**Init = X**
Offset: 18h	**PMI18: MDC/MDIO #3 Control Register**	**Init = X**

Bit	Attr.	Description
31	RW	**FIRE/BUSY** Setting this bit to 1 initializes a operation sequence to PHY. This bit would be auto cleared after the operation is finished.
30:29	RO	**Reserved (0)**
28	RW	**ST Code** 0: clause 45 1: clause 22
27:26	RW	**OP Code** Clause 22 / Clause 45 00 reserved / Address 01 Write / Write 10 Read / Post Read Increment Address 11 reserved / Read
25:21	RW	**PHYAD/PRTAD: PHY address(clause 22) or Port address(clause 45)**
20:16	RW	**REGAD/DEVAD: PHY register address(clause 22) or Device address(clause 45)**
15:0	RW	**MIIWDATA: Write data to PHY**

Offset: 04h	**PMI04: MDC/MDIO #0 Read Data Register**	**Init = c001_0000h**
Offset: 0Ch	**PMI0C: MDC/MDIO #1 Read Data Register**	**Init = c001_0000h**
Offset: 14h	**PMI14: MDC/MDIO #2 Read Data Register**	**Init = c001_0000h**
Offset: 1Ch	**PMI1C: MDC/MDIO #3 Read Data Register**	**Init = c001_0000h**

Bit	Attr.	Description
31:24	RW	**MDC cycle threshold [7:0]** This field defines the period of **MDC #0∼#3 time**. The period of **MDC #0 time** is **(PMI04 [31:24] + 1) * 2 * (HCLK clock period)**. The period of **MDC #1 time** is **(PMI0c [31:24] + 1) * 2 * (HCLK clock period)**. The period of **MDC #2 time** is **(PMI14 [31:24] + 1) * 2 * (HCLK clock period)**. The period of **MDC #3 time** is **(PMI1c [31:24] + 1) * 2 * (HCLK clock period)**.
23	RW	**MDIO latch edge control** **(for debugging purpose only)** 0: Latch MDIO signal at the rising edge of the MDC 1: Latch MDIO signal at the falling edge of the MDC
22:20	RW	**MDIO latch timing control** **(for debugging purpose only)** 0: Normal timing, 1: Delay 1 MDC cycle, 2: Delay 2 MDC cycle, 3: Delay 3 MDC cycle 4: 4 MDC cycle early, 5: 3 MDC cycle early, 6: 2 MDC cycle early, 7: 1 MDC cycle early
19:17	RO	**Reserved (0)**
16	RO	**IDLE** 0: The operation sequence to PHY is busy 1: The controller is IDLE, and the read data from PHY is ready
15:0	RO	**MIIRDATA: Read data from PHY**

Offset: 7Ch		PMI7C: PHY Interrup Activity Level	Init = 0h
31:4	RO	**Reserved (0)**	
3	RW	**MAC4LINK activity level**	
2	RW	**MAC3LINK activity level**	
1	RW	**MAC2LINK activity level**	
0	RW	**MAC1LINK activity level** 0: Input signal is activity low 1: Input signal is activity high	

18 USB2.0 Virtual Hub Controller (USB20)

18.1 Overview

USB2.0 Controller implements 1 set of USB Hub register and 7 sets of USB Device registers. The physical address of these registers can be derived as the following:

Base address of USB Hub= 0x1E6A_0000
Physical address = (Base address) + Offset

PHYA00: PHY Control/Status #1
PHYA04: PHY Control/Status #2
PHYA08: PHY Control/Status #3
PHYA0C: PHY Control/Status #4
HUB00: Root Function Control & Status Register
HUB04: Root Configuration Setting Register
HUB08: Interrupt Control Register
HUB0C: Interrupt Status Register
HUB10: Programmable Endpoint Pool ACK Interrupt Enable Register
HUB14: Programmable Endpoint Pool NAK Interrupt Enable Register
HUB18: Programmable Endpoint Pool ACK Interrupt Status Register
HUB1C: Programmable Endpoint Pool NAK Interrupt Status Register
HUB20: Device Controller Soft Reset Enable Register
HUB24: USB Status Register
HUB28: Programmable Endpoint Pool Data Toggle Value Set
HUB2C: Isochronous Transaction Fail Accumulator
HUB30: Endpoint 0 Control/Status Register
HUB34: Base Address of Endpoint 0 IN/OUT Data Buffer Register
HUB38: Endpoint 1 Control/Status Register
HUB3C: Endpoint 1 Status Change Bitmap Data
HUB40: SOF Counter
HUB44: DMA to Memory Synchronization Status
DEV00: Downstream Device Function Enable Control Register
DEV04: Interrupt Status
DEV08: Endpoint 0 Control/Status Register
DEV0C: Base Address of Endpoint 0 IN/OUT Data Buffer Register
EPP00: Endpoint Configuration Register
EPP04: DMA Descriptor List Control/Status Register
EPP08: DMA Descriptor/Buffer Base Address
EPP0C: DMA Descriptor List Read(DMA)/Write(CPU) Pointer and Status
DES_0: Data Buffer Base Address
DES_1: Descriptor Control/Status

18.2 Features

- Complies with the Universal Serial Bus specification Rev. 2.0, Supports USB Full Speed (12Mb/sec) and High Speed (480 Mb/sec), backward compatible with USB1.1.
- USB Hub architecture, supports 1 hub device port and 7 downstream device ports.
- Supports 21 programmable endpoints that can be assigned to any devices, and can be configured to Bulk IN/OUT, Interrupt IN/OUT and Isochronous IN/OUT type endpoint.
- For Hub device, supports :
  1. 1 default Control endpoint.
  2. 1 dedicated hub status Interrupt IN endpoint.

3. Any number (1-15) of programmable endpoints.

– For each Downstream device controller, supports :
1. 1 default Control endpoint.
2. Any number (1-15) of programmable endpoints.

– Automatic retry of failed packets, and PING Flow control.
– Separate data buffers for the SETUP data of a CONTROL transfer
– Integrated DMA engine for direct memory bus accesses (bypass AHB bus).
– Supports independent DMA channel for each endpoint.
– Supports 256 stages descriptor mode for all 21 programmable endpoints.
– Supports USB remote wake-up function (Suspend/Resume operation).

18.3 Comparison to AST2500

1. Add support downstream device number to 7.
2. Add support programmable endpoint number to 21.
3. Support byte alignment for all DMA buffers.

18.4 Procedure to enable USB2.0 Hub port

18.4.1 PCIe EHCI to Hub path

1. Set SCU440 bit[25:24] = ”00”
2. Set SCU040 bit[14] = 1, enable controller reset
3. Set SCU080 bit[14] = 1, enable PHY clock
4. wait 10 ms for PLL locking
5. Set SCU044 bit[14] = 1, disable controller reset
6. Set HUB00 bit[11] = 1, disable PHY reset
7. Load driver

18.4.2 Hub to USB2.0 port 1

1. Set SCU440 bit[25:24] = ”01”, select USB2.0 port 1 mode as device
2. Set PHYA00 bit[8] = 0, set UTMI 16 bits mode
3. Set SCU040 bit[14] = 1, enable controller reset
4. Set SCU080 bit[14] = 1, enable PHY clock
5. wait 10 ms for PLL locking
6. Set SCU044 bit[14] = 1, disable controller reset
7. Set HUB00 bit[11] = 1, disable PHY reset
8. Load driver

18.5 Procedure to enable USB2.0 eye diagram measurement

- Measure Host TX waveform, enable device port termination
  1. Measurement point on the AST2600 side that is mostly close to the USB pin position.
  2. Force host (PCH) to send out Test Packet pattern, and follow below sequence to set AST2600 into high speed termination mode.
  3. Enter firmware u-boot
  4. mw 1e6e2000 1688a8a8
  5. mw 1e6e2440 1000000
  6. mw 1e6e2040 4000
  7. mw 1e6e2080 4000
  8. mw 1e6e2044 4000
  9. mw 1e6a0000 b41
- Measure Device TX waveform
  1. Measurement point on the host (PCH) side that is mostly close to the USB pin position.
  2. Force host (PCH) USB port to enable high speed termination for device eye diagram measurement.
  3. Enter BMC firmware u-boot
  4. mw 1e6e2000 1688a8a8
  5. mw 1e6e2440 1000000
  6. mw 1e6e2040 4000
  7. mw 1e6e2080 4000
  8. mw 1e6e2044 4000
  9. mw 1e6a0000 c41

18.6 Registers : Base Address = 0x1E6A:0000

18.6.1 Address Definition

Offset	Size(Byte)	Description
0x07F-0x000	128	Root/Global Device Register
0x087-0x080	8	Root Device SETUP Data Buffer
0x08F-0x088	8	Device 1 SETUP Data Buffer
0x097-0x090	8	Device 2 SETUP Data Buffer
0x09F-0x098	8	Device 3 SETUP Data Buffer
0x0A7-0x0A0	8	Device 4 SETUP Data Buffer
0x0AF-0x0A8	8	Device 5 SETUP Data Buffer
0x0B7-0x0B0	8	Device 6 SETUP Data Buffer
0x0BF-0x0B8	8	Device 7 SETUP Data Buffer
0x0FF-0x0C0	64	Reserved
0x10F-0x100	16	Device 1 Register
0x11F-0x110	16	Device 2 Register
0x12F-0x120	16	Device 3 Register
0x13F-0x130	16	Device 4 Register
0x14F-0x140	16	Device 5 Register
0x15F-0x150	16	Device 6 Register
0x16F-0x160	16	Device 7 Register
0x1FF-0x170	144	Reserved
0x20F-0x200	16	Programmable Endpoint 0 Register
0x21F-0x210	16	Programmable Endpoint 1 Register
0x22F-0x220	16	Programmable Endpoint 2 Register
0x23F-0x230	16	Programmable Endpoint 3 Register
0x24F-0x240	16	Programmable Endpoint 4 Register
0x25F-0x250	16	Programmable Endpoint 5 Register
0x26F-0x260	16	Programmable Endpoint 6 Register
0x27F-0x270	16	Programmable Endpoint 7 Register
0x28F-0x280	16	Programmable Endpoint 8 Register
0x29F-0x290	16	Programmable Endpoint 9 Register
0x2AF-0x2A0	16	Programmable Endpoint 10 Register
0x2BF-0x2B0	16	Programmable Endpoint 11 Register
0x2CF-0x2C0	16	Programmable Endpoint 12 Register
0x2DF-0x2D0	16	Programmable Endpoint 13 Register
0x2EF-0x2E0	16	Programmable Endpoint 14 Register
0x2FF-0x2F0	16	Programmable Endpoint 15 Register
0x30F-0x300	16	Programmable Endpoint 16 Register
0x31F-0x310	16	Programmable Endpoint 17 Register
0x32F-0x320	16	Programmable Endpoint 18 Register
0x33F-0x330	16	Programmable Endpoint 19 Register
0x34F-0x340	16	Programmable Endpoint 20 Register
0x81F-0x800	32	PHY Control Register

18.6.2 PHY Control Register (Host/Device)

Offset: 00	PHYA00: USB PHY Control/Status #1	Init = 0x0300
Bit	**R/W**	**Description**
31:16	RO	**PHY debug output**
15:12	RW	**PHY debug mode selection**
11	RW	**Enable USB2.0 Host Controller #1 128/256 Bytes Transmit FIFO Threshold** **(for debugging purpose only)** 0: disable 1: enable When enabled, EHCI84 and BEHCI84[7:6] can be set to 00/01
10	RW	**Improve USB2.0 Host Controller #1 asynchronous list performance** **(for debugging purpose only)** 0: disable 1: enable
9	RW	**Enable PHY to output 120MHz and 480MHz clock** 0: disable 1: enable
8	RW	**Enable 8 bit UTMI interface** 0: 16 bits UTMI 1: 8 bits UTMI When configured as EHCI to Hub direct connection mode, it will be forced to 8 bits UTMI mode.
7	RW	**DN 15K pull down enable (Device mode)**
6	RW	**DP 15K pull down enable (Device mode)**
5	RW	**PHY output Low Swing mode**
4	RW	**PHY BIST mode load control**
3:0	RW	**PHY BIST mode selection**

Offset: 04	PHYA04: USB PHY Control/Status #2	Init = 0x00C00000
Bit	**R/W**	**Description**
31:0	RW	**PHY XCFGI[31:0]**

Offset: 08	PHYA08: USB PHY Control/Status #3	Init = 0x00000001
Bit	**R/W**	**Description**
31:0	RW	**PHY XCFGI[63:32]**

Offset: 0C	PHYA0C: USB PHY Control/Status #4	Init = 0x00024903
Bit	**R/W**	**Description**
31:24	RO	**PHY XCFGO**
23:18	RO	**Reserved**
17:15	RW	**PHY XCfg_LOCK_RANGE_MIN**
14:12	RW	**PHY XCfg_LOCK_RANGE_MAX**
11:9	RW	**PHY XCfg_FINE_TUNE_NUM**
8:6	RW	**PHY XCfg_COARSE_TUNE_NUM**
5:0	RW	**PHY XCFGI[69:64]**

18.6.3 Root/Global Register Definition

Offset: 00	HUB00: Root Function Control & Status Register	Init = 0x80000000

Bit	R/W	Description
31	RO	**Reserved (1)**
30:20	RO	**Reserved (0)**
19	RW	**Enable FIFO dynamic power down** 0: disable 1: enable
18	RW	**Programmable endpoint long descriptor list mode** 0: 32 stages descriptor 1: 256 stages descriptor This bit will affect the full endpoint pool.
17	RW	**Isochronous IN null data response control** 0: No response, wait host timeout 1: Return 0 byte DATA0 packet This bit controls the response action for Isochronous IN type endpoints when the IN data for transmitting not ready. The action can be no response or return an 0 byte DATA0 packet. When no response action is selected, then CSPLIT IN retry is possible.
16	RW	**Complete a "SPLIT IN Transaction" after SOF has been received** 0: Disable 1: Enable Because "Complete a SPLIT IN Transaction" has no ACK, its very difficult to define the end of the transaction. Set this bit to '1' will add SOF packet into the transaction finish check list. This bit must be set to 1 at Set_Address transfer cycle if keep the old address update methodology.
15:14	RO	**Reserved (0)**
13	RO	**USB PHY BIST result** 0: Fail 1: Pass This flag will be cleared by disabling USB PHY BIST function (HUB00 [12] = 0). Note: BIST stands for Built-In-Self-Test
12	RW	**USB PHY BIST control** 0: Turn off USB PHY BIST 1: Turn on USB PHY BIST
11	RW	**Disable USB PHY reset** 0: Enable USB PHY reset 1: Disable USB PHY reset
10:8	RW	**USB Test Mode selection** 000: Disable 001: Enable Test J 010: Enable Test K 011: Enable Test SE0_NAK 100: Enable Test Packet 101: Reserved 110: Reserved 111: Reserved ~~Enable Test Loop Back~~ (for debugging purpose only)
7	RW	**Force USB bus state timer to work at test mode** **(for debugging purpose only)** 0: Normal operation mode 1: Force USB bus state to High Speed mode (X32 faster)

to next page

from previous page

6	RW	**USB Force to High Speed State Mode** **(for debugging purpose only)** 0: Normal operation 1: Force the bus state to High Speed
5	RW	**USB Remote Wakeup signaling pulse width selection** 0: 8ms 1: 12ms
4	RW	**Enable manual Remote Wakeup** 0: No operation 1: Enable manual Remote Wakeup, can be set only in Suspend state This register can be effectively set to ”1” only when USB Controller enters Suspend state, and this register will be automatically cleared by H/W after Remote Wakeup finished.
3	RW	**Enable automatic Remote Wakeup** 0: Disable 1: Enable automatic Remote Wakeup When this register is enabled, Remote Wakeup singling will be automatically issued whenever firmware write commands has been received in Suspend state.
2	RW	**Enable clock stopping in suspend state** 0: USB Controller won’t stop clock even in Suspend state 1: USB Controller will stop clock in Suspend state
1	RW	**Upstream port connection speed selection** 0: Select High Speed and Full Speed modes 1: Select Full Speed mode only
0	RW	**Enable upstream port connection** 0: Disable upstream port connection 1: Enable upstream port connection

Offset: 04 **HUB04: Root Configuration Setting Register** **Init = 0**

Bit	R/W	Description
31:24	RO	**Reserved**
23:16	RO	**Status of DMA page buffer** **(for debugging purpose only)** 0: Page Free 1: Page Allocated Bit16: Status of Page #0 .... Bit23: Status of Page #7 USB Controller totally integrates 2K byes of SRAM to be allocated for data transmit of IN transaction. The 2K bytes of SRAM is uniformly divided into 8 pages, each of them is 256 bytes long. Each EP can only allocate one page buffer from them, but there are 2 pages (Page 0 and Page 1) are arranged as a ring buffer and dedicated for the usage of the active EP. Therefore, the status bits of the two pages are reserved.
15:7	RO	**Reserved**
6:0	RW	**Root function device address** Change the address whenever Set_Address command was received. The following is the Set_Address command sequence: 1. SETUP data packet, contains the new address information. 2. Change to the new address. 3. IN data packet, status phase with zero byte data returned.

Offset: 08	HUB08: Interrupt Control Register	Init = 0

Bit	R/W	Description
31:18	RO	**Reserved (0)**
17	RW	**Enable Programmable Endpoint Pool NAK Interrupt**
16	RW	**Enable Programmable Endpoint Pool ACK/STALL Interrupt**
15	RW	**Enable Device #7 Controller Interrupt**
14	RW	**Enable Device #6 Controller Interrupt**
13	RW	**Enable Device #5 Controller Interrupt**
12	RW	**Enable Device #4 Controller Interrupt**
11	RW	**Enable Device #3 Controller Interrupt**
10	RW	**Enable Device #2 Controller Interrupt**
9	RW	**Enable Device #1 Controller Interrupt**
8	RW	**Enable USB Suspend Resume Interrupt** This interrupt is used to notify software that the USB host is leaving suspend mode. Software can do something because USB device will wake up. If software doesn't need to know the resume event, please don't enable this bit, so that hardware will skip to wait software to do something.
7	RW	**Enable USB Suspend Entry Interrupt** This interrupt is used to notify software that the USB host is entering suspend mode. Software can do something because USB device will entering sleep state. If software doesn't need to know the suspend event, please don't enable this bit, so that hardware will skip to wait software to do something.
6	RW	**Enable USB Bus Reset Interrupt** When Bus Reset occurred, hardware will automatically reset the connection status and return to the not configured state. Software must also clear the related states and prepare for the new connection.
5	RW	**Enable Hub EP1 IN Data packet ACK Interrupt**
4	RW	**Enable Hub EP0 IN Data packet NAK Interrupt**
3	RW	**Enable Hub EP0 IN Data packet ACK/STALL Interrupt**
2	RW	**Enable Hub EP0 OUT Data packet NAK Interrupt**
1	RW	**Enable Hub EP0 OUT Data packet ACK/STALL Interrupt**
0	RW	**Enable Hub EP0 SETUP Data packet ACK Interrupt**

Note :
The definition of this register is :
0 : Disable
1 : Enable

The enable control for Endpoint Pool is the first level interrupt enable bit for all the EPs allocated from Endpoint Pool.

The enable control for Device is the first level interrupt enable bit for all the downstream device controller.

Offset: 0C	HUB0C: Interrupt Status Register	Init = 0

Bit	R/W	Description
31	RO	**DMA to Memory flush idle status** 0: DMA is in busy to flush data to memory. 1: DMA is idle.
30:19	RO	**Reserved (0)**

to next page

from previous page

18	RO	~~**USB command bus is dead locked**~~
17	RO	**Programmable Endpoint Pool NAK Interrupt Occurs**
16	RO	**Programmable Endpoint Pool ACK/STALL Interrupt Occurs**
15	RO	**Device #7 Controller Interrupt Occurs**
14	RO	**Device #6 Controller Interrupt Occurs**
13	RO	**Device #5 Controller Interrupt Occurs**
12	RO	**Device #4 Controller Interrupt Occurs**
11	RO	**Device #3 Controller Interrupt Occurs**
10	RO	**Device #2 Controller Interrupt Occurs**
9	RO	**Device #1 Controller Interrupt Occurs**
8	RW	**USB Suspend Resume event has occurred** When this register is set to ”1”, it indicates that USB Upstream Port has resumed from Suspend state.
7	RW	**USB Suspend Entry event has occurred** When this register is set ”1”, it indicates that the USB Upstream Port has entered Suspend state.
6	RW	**USB Bus Reset has Occurred** When set, indicates a USB Bus Reset state occurs.
5	RW	**EP1 IN Data Packet ACK/STALL Returned** When set, indicates an IN transaction finished with ACK transmitted.
4	RW	**EP0 IN Data Packet NAK Returned** When set, indicates an IN packet received and responded with NAK. Control Endpoint IN/OUT NAK response interrupt can used to indicate the current Host transaction type, and S/W can based on this to determine the current Control Transfer stage.
3	RW	**EP0 IN Data Packet ACK/STALL Returned** When set, indicates an IN transaction finished with ACK or STALL transmitted.
2	RW	**EP0 OUT Data Packet NAK Returned** When set, indicates an OUT/PING packet received and responded with NAK. Control Endpoint IN/OUT NAK response interrupt can used to indicate the current Host transaction type, and S/W can based on this to determine the current Control Transfer stage.
1	RW	**EP0 OUT Data Packet ACK/STALL Returned** When set, indicates an OUT transaction finished with ACK/STALL returned, or a PING packet received and responded with STALL.
0	RW	**EP0 Setup Data Arrives** When set, indicates an SETUP transaction has been received successfully.

Note :
Each status bit of this register is set by H/W automatically when the corresponding event occurred or the transaction finished. The corresponding interrupt enable bit wont impact the setting of the bit. The enable bit only determines the interrupt generation.
Whenever S/W has finished the handling of the status bit, it must clear the status bit by writing ’1’, else it will not be able to recognize new events.

Offset: 10	**HUB10: Programmable Endpoint Pool ACK Interrupt Enable Register**	**Init = 0**
Bit	**R/W**	**Description**
31:21	RO	**Reserved (0)**

to next page

from previous page

20:0	RW	**Programmable Endpoint ACK Interrupt Enable** bit 0: Programmable Endpoint number #0 bit 1: Programmable Endpoint number #1 .... bit20: Programmable Endpoint number #20 This register is to control the ACK interrupt enable bits of the 21 programmable Endpoints in the Endpoint Pool. The definition of each bit in this register is as the following: 0: Disable interrupt 1: Enable interrupt

Offset: 14 HUB14: Programmable Endpoint Pool NAK Interrupt Enable Register Init = 0

Bit	R/W	Description
31:21	RO	**Reserved (0)**
20:0	RW	**Programmable Endpoint NAK Interrupt Enable** bit 0: Programmable Endpoint number #0 bit 1: Programmable Endpoint number #1 .... bit20: Programmable Endpoint number #20 This register is to control the ACK interrupt enable bits of the 21 programmable Endpoints in the Endpoint Pool. The definition of each bit in this register is as the following: 0: Disable interrupt 1: Enable interrupt

Offset: 18 HUB18: Programmable Endpoint Pool ACK Interrupt Status Register Init = 0

Bit	R/W	Description
31:21	RO	**Reserved (0)**
20:0	RW	**Programmable Endpoint ACK Interrupt Occurs** bit 0: Programmable Endpoint number #0 bit 1: Programmable Endpoint number #1 .... bit20: Programmable Endpoint number #20 This status flag will be set to '1' under any one of the following conditions: 1. STALL response. 2. When short packet received from OUT transaction. 3. When Interrupt Generation Enable been detected from the DMA descriptor, or for a single descriptor mode. 4. When the DMA descriptor list becomes empty, this indicates that the last descriptor been used.

Note :
Each status is automatically set to "1" whenever the ACK event has been occurred, no matter the corresponding interrupt enable bit has been enabled or not.
S/W must clear each status by writing '1' to its bit after finished the process; otherwise the next ACK event will not be recognized.

Offset: 1C	HUB1C: Programmable Endpoint Pool NAK Interrupt Status Register	Init = 0

Bit	R/W	Description
31:20	RO	**Reserved (0)**
20:0	RW	**Programmable Endpoint NAK Interrupt Occurs** bit 0: Programmable Endpoint number #0 bit 1: Programmable Endpoint number #1 .... bit20: Programmable Endpoint number #20 This status flag will be set to '1' when the endpoint response with NAK.

Note :
Each status is automatically set to "1" whenever the NAK event has been occurred, no matter the corresponding interrupt enable bit has been enabled or not.
S/W must clear each status by writing '1' to its bit after finished the process; otherwise the next NAK event will not be recognized.

Offset: 20	HUB20: Device Controller Soft Reset Enable Register	Init = 0x3FF

Bit	R/W	Description
31:10	RO	**Reserved (0)**
9	RW	**Enable Programmable Endpoint Pool software Reset**
8	RW	**Enable DMA Controller software Reset**
7	RW	**Enable Device #7 Controller software Reset**
6	RW	**Enable Device #6 Controller software Reset**
5	RW	**Enable Device #5 Controller software Reset**
4	RW	**Enable Device #4 Controller software Reset**
3	RW	**Enable Device #3 Controller software Reset**
2	RW	**Enable Device #2 Controller software Reset**
1	RW	**Enable Device #1 Controller software Reset**
0	RW	**Enable Root HUB Controller software Reset**

Note :
0 : Normal operation
1 : Reset the device controller
These software reset bits only reset the controllers status registers, not including all the registers supported by the controller. To reset all the registers of the controllers, please reference the registers of SCU040.
Software sets the specific bit to '1' to start the reset process, and sets the specific bit to '0' to stop the reset process. There is no need to put time delay between the two processes.

Offset: 24	HUB24: USB Status Register (for debugging purpose only)	Init = X

Bit	R/W	Description
31	RO	**USB Suspend State**
30	RO	**USB Bus Reset State**
29	RO	**USB Bus Line State DN**
28	RO	**USB Bus Line State DP**
27	RO	**USB Bus Speed** 0: Full Speed 1: High Speed SW reads this bit to determine the current host connection speed. But SW must read this bit after the first packet is received that behind the bus reset cycle.

to next page

from previous page

26:16	RO	**USB Last Frame Number record**
15	RO	**UTMI State XcvrSelect** 0: High Speed 1: Full Speed
14	RO	**UTMI State TermSelect** 0: High Speed 1: Full Speed
13:12	RO	**UTMI State OPMode** 0: Normal mode 1: Non Driving 2: Disable Bit-Stuff and NRZI encoding 3: Reserved
11:8	RO	**Endpoint Number of the Last USB Transaction**
7	RO	**Reserved (0)**
6:0	RO	**Device Address of the Last USB Transaction**

Offset: 28 **HUB28: Programmable Endpoint Pool Data Toggle Value Set** **Init = X**

Bit	R/W	Description
31:9	RO	**Reserved (0)**
8	WO	**Endpoint data toggle bit initial value set** 0: Initial to sequence DATA0 1: Initial to sequence DATA1 The indexed data toggle bit is determined by Bit [4:0] of this register. Only one data toggle bit can be initialized for each register write. Reading this register always returns ”0”.
7:5	RO	**Reserved (0)**
4:0	WO	**Programmable Endpoint Index** 0-20: Endpoint number index 21-31: Invalid This index value determines which Endpoint data toggle bit will be initialized. Reading this register always returns ”0”.

Note :
Data toggle sequence initialization can only be applied to Control/Bulk/Interrupt type Endpoints.
Isochronous type Endpoints should not be initialized; it will be reset automatically when SOF receives.

HUB2C: Isochronous Transaction Fail Accumulator (for debugging purpose only)

Offset: 2C **Init = 0**

Bit	Attr.	Description
31:26	RO	**Reserved (0)**
25:16	RW	**Isochronous OUT Failure Counter**
15:10	RO	**Reserved**
9:0	RW	**Isochronous IN Failure Counter**

Note :
Writing any data to this register will clear the two counters to 0.
The two counter values show the number of isochronous transaction failures which are due to either buffer unavailable or data not yet ready.

Offset: 30	HUB30: Endpoint 0 Control/Status Register	Init = 0
Bit	**R/W**	**Description**
31:23	RO	**Reserved (0)**
22:16	RO	**Endpoint 0 OUT received data byte count**
15	RO	**Reserved (0)**
14:8	RW	**Endpoint 0 IN data byte count for transfer**
7:3	RO	**Reserved (0)**
2	RW	**Endpoint 0 OUT buffer ready for receiving data** 0: Buffer is not ready to receive data 1: Buffer is ready to receive data S/W can set this bit to '1' when it is ready to receive data from Host by OUT transactions. When H/W receives the data successfully, this bit will be automatically cleared to "0". S/W can monitor this bit to check the data has been received or not.
1	RW	**Endpoint 0 IN buffer ready for transferring data** 0: Data is not ready to transfer data 1: Data is ready to transfer data S/W can set this bit to '1' when there is a need to transfer data to Host by IN transactions. When H/W has finished the data transfer, this bit will be automatically cleared. S/W can monitor this bit to check the data has been transferred or not. Only one of Bit [1] and Bit [2] can be set to "1" for each time. The two settings are exclusive.
0	RW	**Endpoint 0 STALL control** When this register is set to "1", Endpoint 0 returns STALL response for Data and Status stages. This bit is automatically cleared to "0" after a new SETUP packet is received.

Offset: 34	HUB34: Base Address of Endpoint 0 IN/OUT Data Buffer Register	Init = X
Bit	**R/W**	**Description**
31	RO	**Reserved (0)**
30:0	RW	**Base address of Endpoint 0 IN/OUT data buffer** This register defines the base address of Default Control Endpoint transaction data buffer, ~~which is 64 bytes long and at 64-bit boundary~~. The direction of this buffer is determined by S/W and depends on the Control Transfer Command mode.

Offset: 38	HUB38: Endpoint 1 Control/Status Register	Init = 0
Bit	**R/W**	**Description**
31:3	RO	**Reserved (0)**
2	W1T	**Reset Endpoint 1 data toggle bit to DATA0** 0: No operation 1: Reset data toggle bit to DATA0
1	RW	**Endpoint 1 STALL control** When this register is set to 1, Endpoint 1 will return STALL response for this endpoint polling.
0	RW	**Enable Endpoint 1** 0 : Disabled 1 : Enabled

Offset: 3C	HUB3C: Endpoint 1 Status Change Bitmap Data	Init = 0
Bit	**R/W**	**Description**
31:8	RO	**Reserved**
7	RW	**Port #7 Status Change Bit (Device #7)**
6	RW	**Port #6 Status Change Bit (Device #6)**
5	RW	**Port #5 Status Change Bit (Device #5)**
4	RW	**Port #4 Status Change Bit (Device #4)**
3	RW	**Port #3 Status Change Bit (Device #3)**
2	RW	**Port #2 Status Change Bit (Device #2)**
1	RW	**Port #1 Status Change Bit (Device #1)**
0	RW	**Hub Port Status Change Bit**
Note : When any bits of this register is not equal to zero, then USB Controller will automatically response with this byte of data; otherwise it will response with NAK when being polled for this Endpoint.		

Offset: 40	HUB40: SOF Counter	Init = 0
Bit	**R/W**	**Description**
31:0	RW	**SOF Counter** SOF Counter for SOF period synchronization between host and client side. Write command clear this counter to 0.

Offset: 44	HUB44: DMA to Memory Synchronization Status	Init = 0
Bit	**R/W**	**Description**
31:16	RO	**Reserved (0)**
15:8	RO	**DMA to Memory ACK counter**
7:0	RO	**DMA to Memory REQ counter**

18.6.4 Device #1 — #7 Register Definition

Offset: 00	DEV00: Downstream Device Function Enable Control Register	Init = 0
Bit	**R/W**	**Description**
31:15	RO	**Reserved (0)**
14:8	RW	**Downstream Device Address** Change the address whenever Set_Address command was received. The following is the Set_Address command sequence: 1. SETUP data packet, contains the new address information. 2. Change to the new address. 3. IN data packet, status phase with zero byte data returned.
7	RO	**Reserved (0)**
6	RW	**Enable Endpoint 0 IN data packet NAK interrupt** 0: Disable 1: Enable
5	RW	**Enable Endpoint 0 IN data packet ACK/STALL interrupt** 0: Disable 1: Enable

to next page

from previous page

4	RW	**Enable Endpoint 0 OUT data packet NAK interrupt** 0: Disable 1: Enable
3	RW	**Enable Endpoint 0 OUT data packet ACK/STALL interrupt** 0: Disable 1: Enable
2	RW	**Enable Endpoint 0 SETUP data packet ACK interrupt** 0: Disable 1: Enable
1	RW	**Device port speed selection** 0 : Full Speed mode or Low Speed mode 1 : High Speed mode
0	RW	**Enable device port** 0 : Disable device port 1 : Enable device port Whenever Upstream Port Bus Reset occurs, this bit will be cleared.

Offset: 04 **DEV04: Interrupt Status** **Init = 0**

Bit	R/W	Description
31:5	RO	**Reserved (0)**
4	RW	**Endpoint 0 IN data packet NAK returned** When this register is set to ”1”, it indicates that an IN transaction has been finished with NAK response. Endpoint 0 IN/OUT NAK response interrupt can be used to indicate the current Host transaction type, and S/W can based on this to determine the current Control Transfer stage.
3	RW	**Endpoint 0 IN data packet ACK received or STALL returned** When this register is set to ”1”, it indicates that an IN transaction has been finished with ACK/STALL response.
2	RW	**Endpoint 0 OUT data packet NAK returned** When this register is set to ”1”, it indicates that an OUT transaction has been finished with NAK response. Endpoint 0 IN/OUT NAK response interrupt can be used to indicate the current Host transaction type, and S/W can based on this to determine the current Control Transfer stage.
1	RW	**Endpoint 0 OUT data packet ACK/STALL returned** When this register is set to ”1”, it indicates that an OUT transaction has been finished with ACK/STALL response. Or a PING packet received and responded with STALL.
0	RW	**Endpoint 0 SETUP data packet received** When this register is set to ”1”, it indicates that a SETUP transaction been finished.

Note :
Each status bit of this register is set by H/W automatically when the corresponding event occurred or the transaction finished. The corresponding interrupt enable bit wont impact the setting of the bit. The enable bit only determines the interrupt generation.
Whenever S/W has finished the handling of the status bit, it must clear the status bit by writing ’1’, else it will not be able to recognize new events.

Offset: 08 **DEV08: Endpoint 0 Control/Status Register** **Init = 0**

Bit	R/W	Description
31:29	RO	**Reserved (0)**

to next page

from previous page

28	RO	**CSPLIT IN Wait** **(for debugging purpose only)**
27	RO	**Normal IN Wait** **(for debugging purpose only)**
26	RO	**Start SPLIT Cycle** **(for debugging purpose only)**
25:24	RO	**Status of Transmit DMA State Machine** **(for debugging purpose only)** 00 : Idle 01 : DMA Request 10 : DMA Done and Data Ready 11 : Reserved
23	RO	**Reserved (0)**
22:16	RO	**Endpoint 0 OUT received data byte count**
15	RO	**Reserved (0)**
14:8	RW	**Endpoint 0 IN data byte count for transfer**
7:3	RO	**Reserved (0)**
2	RW	**Endpoint 0 OUT buffer ready for receiving data** 0: Buffer is not ready to receive data 1: Buffer is ready to receive data S/W can set this bit to '1' when it is ready to receive data from Host by OUT transactions. When H/W receives the data successfully, this bit will be automatically cleared to "0". S/W can monitor this bit to check the data has been received or not.
1	RW	**Endpoint 0 IN buffer ready for transferring data** 0: Data is not ready to transfer data 1: Data is ready to transfer data S/W can set this bit to '1' when there is a need to transfer data to Host by IN transactions. When H/W has finished the data transfer, this bit will be automatically cleared. S/W can monitor this bit to check the data has been transferred or not. Only one of Bit [1] and Bit [2] can be set to "1" for each time. The two settings are exclusive.
0	RW	**Endpoint 0 STALL control** When this register is set to "1", Endpoint 0 returns STALL response for Data and Status stages. This bit is automatically cleared to "0" after a new SETUP packet is received.

Offset: 0C DEV0C: Base Address of Endpoint 0 IN/OUT Data Buffer Register Init = X

Bit	R/W	Description
31	RO	**Reserved (0)**
30:0	RW	**Base address of data buffer** This register defines the base address of Default Control Endpoint transaction data buffer, ~~which is 64 bytes long and at 64 bit boundary~~. The direction of this buffer is determined by S/W and depends on the Control Transfer Command mode.

18.6.5 Programmable Endpoint #0 — #20 Register Definition

Offset: 00 **EPP00: Endpoint Configuration Register** **Init = 0**

Bit	R/W	Description
31:28	RW	**List mode pre-empty interrupt control** 0: disable pre-empty interrupt control. Define the stages number before list empty to issue interrupt. This is used by firmware to cover the time penalty to enter interrupt service routine. Interrupt will be issued whenever a transaction finished and the reset descriptor stages equal to this register setting. Only applicable at 256 stages descriptor mode.
27	RW	**Burst transactions long idle interrupt control** 0: Disable 1: Enable long idle (over 1 SOF period idle) interrupt control for a burst transaction. Interrupt will be issued when a burst transaction stop received for over 1 SOF cycle period. This can help firmware to reduce interrupt event number for a burst transfer.
26	RO	**Reserved (0)**
25:16	RW	**Endpoint Maximum Packet Size Definition** 0: 1024 bytes 1: 1 byte .... 1023: 1023 bytes – For OUT direction endpoint, this definition limits the maximum packet size can be received. If extra bytes received, it will not be transfered to DRAM and return with NAK response to retry this transaction. – For IN direction endpoint, it must be used in companion with EPP04[3]. It defines the maximum bytes to be returned within 1 packet. This enables a possibility to pack several packets into a descriptor stage. The total size that can be packed in a descriptor stage equals to 4096 bytes or 8 sub-packets, depends on which criteria meet first.
15:14	RW	**Endpoint Data Fetch Control** **1. Isochronous type endpoint.** Defines the auto data toggle stage setting under High speed. Value / Stages / Isochronous IN / Isochronous OUT 00 / 1 / DATA0 / DATA0 01 / 2 / DATA1→DATA0 / MDATA→DATA1 1x / 3 / DATA2→DATA1→DATA0 / MDATA→MDATA→DATA2 When SOF received, the data PID sequencing will be reset to the start data PID. For long packet (transmit byte count > maximum packet size) or high bandwidth Isochronous IN, please enable the auto data toggle mode, EPP00[13] = 0. **2. Interrupt type endpoint.** Defines the prefetch mode when long packet or high bandwidth Interrupt IN. 00: No prefetch. Next data will be fetched after transaction successfully. 01: Prefetch the next packet data once after its was transmitted.

to next page

from previous page

13	RW	**Endpoint Auto Data Toggle Disable** 0 : Enable auto data toggle 1 : Disable auto data toggle When auto data toggle is disabled for OUT direction endpoint, then all packets will be received without error, ignoring the data sequence errors. When auto data toggle is disabled for IN direction endpoint, then the data sequence PID will be replaced by the PID fetched from the descriptor list.
12	RW	**Endpoint Stall Control** When this bit is set to 1, the endpoint will always return STALL response until it is cleared to 0. The Stall control can only be set for Bulk/Interrupt type endpoints.
11:8	RW	**Endpoint Number** This value defines the endpoint number of this endpoint.
7	RO	**Reserved (0)**
6:4	RW	**Endpoint type selection** 00x : Disable 010 : Bulk In 011 : Bulk Out 100 : Interrupt In 101 : Interrupt Out 110 : Isochronous In 111 : Isochronous Out
3:1	RW	**Allocated Device Port Number** 000 : Root device 001 : Downstream device 1 010 : Downstream device 2 011 : Downstream device 3 100 : Downstream device 4 101 : Downstream device 5 110 : Downstream device 6 111 : Downstream device 7
0	RW	**Enable Endpoint** 0 : Disabled (this endpoint will be reset) 1 : Enabled

Note :
Endpoint Reset can be initiated by the following method:

1. System global reset controlled in SCU, this will reset full controller including registers.
2. Release the endpoint, disable endpoint.
3. Set the reset bit at HUB20 Bit[9] or HUB20 Bit[n], where n is device number set in the Port Number field.

Only the first item will reset the register value, others don't.

Offset: 04 EPP04: DMA Descriptor List Control/Status Register Init = 0

Bit	R/W	Description
31:20	RO	**Reserved (0)**
19	RO	**Occupied Transmit IN Buffer Status** **(for debugging purpose only)** 0 : No buffer 1 : 1 buffer occupied
18:16	RO	**Occupied Transmit IN Buffer Index** **(for debugging purpose only)**

to next page

from previous page

15	RW	**Long idle interrupt status** 0: No interrupt. 1: Interrupt status of long idle. Write '1' to clear this status bit.
14	RW	**Descrptor pre-empty interrupt status** 0: No interrupt. 1: Interrupt status of descriptor list pre-empty. Write '1' to clear this status bit.
13	RW	**OUT transaction short packet interrupt status** 0: No interrupt. 1: Interrupt status of OUT short packet. Write '1' to clear this status bit.
12	RO	**The Current Interrupt Generation Flag** **(for debugging purpose only)** This bit shows the interrupt generation status, fetched from the current stage descriptor.
11	RO	**CSPLIT IN Wait** **(for debugging purpose only)**
10:9	RO	**Auto Data Toggle Count** **(for debugging purpose only)**
8	RO	**Start SPLIT Cycle** **(for debugging purpose only)**
7:4	RO	**Current Descriptor Processing Status** **(for debugging purpose only)** 00 : RX Idle 01 : RX Read Descriptor Request 02 : RX Read Descriptor Grant 03 : RX Read Descriptor Data Back and Buffer Ready 04 : RX OUT Data Receive Cycle 05 : RX OUT Transaction ACK and Descriptor Write Back Request 06 : RX Descriptor Write Back Grant 07 : RX DMA Done 08 : TX Idle 09 : TX Read Descriptor Request 10 : TX Read Descriptor Grant 11 : TX Read Descriptor Data Back and Buffer Ready 12 : TX IN Data DMA Fetch Request 13 : TX IN Data Ready 14 : TX IN Data Transfer Cycle 15 : TX IN Transaction DONE

to next page

from previous page

3	RM	**DMA buffer mode selection** **For OUT direction endpoint** Defines the DMA buffer cascade mode. This mode can shorten the inter-packet latency, and increase the performance. 0: DMA buffers can be allocated anywhere. 1: DMA buffers are allocated continuously, each has size of Maximum packet size defined at EPP00[25:16]. **For IN direction endpoint** Enables the long data packet mode defined in one descriptor stage. In general, one descriptor only contains one packet data. Enable this mode can merge at most 8 packets or 4096 bytes of data into one descriptor stage. The data size of each packet is defined at EPP00[25:16]. This can decrease the interrupt number and increase the performance, especially for high speed high bandwidth transfer. 0: Normal mode, one descriptor only can contain one packet data. 1: Enable long packet support. **Any time this bit is turned ON or OFF, descriptor Reset is necessary.**
2	RW	**Descriptor List Operation Reset** Sets this bit to '1' will reset the descriptor operation and flush buffers. Sets this bit to '0' to disable reset.
1	RW	**Single Stage Descriptor Mode** 0 : 32/256 stages descriptor 1 : 1 stage descriptor, when the OUT/IN transaction done, the CPU Write pointer will be cleared to 0 automatically. The single mode operation is started by set the Write pointer at EPP0C to 1, this means the data/buffer ready for TX/RX transfer. And when transfer done, the Write pointer will be cleared to 0 by H/W.
0	RW	**Descriptor List Operation Enable** 0 : Disable, for single stage mode 1 : Enable normal operation The descriptor list operates at 32/256 stages Ring mode. Descriptor List Operation Enable and Single Stage Descriptor modes are mutually exclusive. They cannot be set at the same time. Single mode has the higher priority.

Offset: 08 EPP08: DMA Descriptor/Buffer Base Address Init = X

Bit	R/W	Description
31	RO	**Reserved (0)**
30:0	RW	**Base address** Descriptor Enabled : Descriptor list base address (64 bits boundary). Descriptor Disabled : DMA data buffer base address (byte boundary).

EPP0C: DMA Descriptor List Read(DMA)/Write(CPU) Pointer and Status		
Offset: 0C		**Init = 0**
Bit	**Attr.**	**Description**
31	RO	**Descriptor List Empty Flag (Default=0)** 0 : not empty 1 : empty
30	RW	**Long idle interrupt status** 0: No interrupt. 1: Interrupt status of long idle. Alias with EPP04[15]. Write '1' to clear this status bit.
29:28	RW	**Endpoint Current Data Toggle Sequence Value (Default=0)** 00 : Indicates the Data PID of next transaction would be DATA0 01 : Indicates the Data PID of next transaction would be DATA2 10 : Indicates the Data PID of next transaction would be DATA1 11 : Indicates the Data PID of next transaction would be MDATA **1.** When read, this value indicates the current internal register state used for next transaction Data sequence PID. **2.** When write, used to setting the next transaction Data sequence PID. Only valid at Single descriptor mode, endpoint type "IN" and auto data toggle disabled.
27	RO	**Reserved (0)**

to next page

from previous page

<table>
<tr><td>26:16</td><td>RW</td><td>
Packet Size (Default=X)
The unit is byte.
This field has 4 definitions:

<table>
<tr><td rowspan="2">Descriptor Type</td><td colspan="2">Endpoint Type</td></tr>
<tr><td>IN</td><td>OUT</td></tr>
<tr><td>Single</td><td>TxDataByteCnt(1)</td><td>RxDataByteCnt(3)</td></tr>
<tr><td>List</td><td>TxDataByteCnt(2)</td><td>MaxPacketSize(4)</td></tr>
</table>
(1) When endpoint type = IN and single mode enabled :
Transmit data packet length
This is the transmit data packet length for IN transfer.
Setting by SW.
(2) When endpoint type = IN and descriptor list mode :
Transmit data packet length
This is the transmit data packet length for IN transfer.
Fetched from descriptor list entry.
(3) When endpoint type = OUT and single mode enabled :
Received data packet length
This is the received data packet length of OUT transfer.
Setting by RXDMA controller.
(4) When endpoint type = OUT and descriptor list mode :
Endpoint Maximum Packet Size
This is used for OUT transaction short packet interrupt trigger.
When the OUT data length received less than the Maximum Packet Size,
then the OUT ACK interrupt flag will raise.
Setting by SW.
SW can write this value when descriptor is under disabled state, including single descriptor mode.
</td></tr>
<tr><td>15:8</td><td>RW</td><td>
Descriptor List DMA Read Pointer (Default=0)
This shows the current descriptor read position that will be read by DMA controller if it is not empty(equal to CPU write pointer).
This pointer can be initialized by S/W when descriptor list mode is disabled and not set at single mode. The correct initialize procedure is as follows:
1. Disable the descriptor list first, set EPP04 = 0
2. Wait descriptor operation status Idle EPP04[7:4] == 0 or 0x8
3. Update the Read Pointer
4. Enable the descriptor operation, set EPP04[0] = 1
</td></tr>
</table>

to next page

from previous page

7:0	RW	**Descriptor List CPU Write Pointer (Default=0)** For transmit (IN) direction, this pointer indicates the transmit data buffer allocated to DMA. For receive (OUT) direction, this pointer indicates the receiver free buffer allocated to DMA. When descriptor operation is enabled, this value indicates the next descriptor write position that will be writing by CPU. And the DMA operation will increment until read pointer equals to write pointer that indicates the Empty condition. The descriptor stage that Write Pointer addressed is not valid, and DMA will not process it. The descriptor list usage cannot be fully, there needs 1 free space for differentiation full and empty cases. That is when WPTR = RPTR, it means empty status. And full status equals WPTR = RPTR-1. The SW can maximum fills the descriptor entry until the full condition; otherwise it will conflict with the empty condition.
Note : The descriptor operation is that the Write Pointer is the leading pointer, and Read Pointer will track at the tail of Write Pointer. Read and Write pointer will be reset to 0 when device reset (HUB20) or USB bus reset occurs.		

18.6.6 Programmable Endpoint DMA Descriptor Definition

Offset: 31:0	DES_0: Data Buffer Base Address	Init = X

Bit	R/W	Description
31:0	RW	**DMA Data Buffer Base Address** Both OUT and IN direction endpoint support byte boundary.

Offset: 63:32	DES_1: Descriptor Control/Status (OUT)	Init = X

Bit	R/W	Description
63	RW	**Enable Interrupt Generation** 0 : No interrupt 1 : Interrupt generated when transaction done. This bit is used by SW to control the generation of interrupt, when SW wants to reduce the interrupts number. By this interrupt control, interrupt will be generated on this endpoint only when : a. Interrupt generation enable = 1 and transaction done for this descriptor. b. Descriptor list empty and the last transaction done. c. Stall response occurs. d. NAK response occurs and descriptor list empty and NAK interrupt enabled.
62:60	RO	**Device Port Number** 000 : Root Hub 001 : Port1 010 : Port2 011 : Port3 100 : Port4 101 : Port5 110 : Port6 111 : Port7 This port number status only used for indicating the received transaction status. No any meaning for transmit path.
59:56	RO	**Endpoint Number** This endpoint number status only used for indicating the received transaction status. No any meaning for transmit path.
55	RO	**Reserved**
54:48	RO	**Device Address** This device address status only used for indicating the received transaction status. No any meaning for transmit path.
47:46	RO	**Data Packet PID** 00 : DATA0 01 : DATA2 10 : DATA1 11 : MDATA This is the Data PID that current packet received.
45	RO	**End of Packet(E)** This is used for Isochronous OUT to a Full Speed device.

to next page

from previous page

44	RO	**Start of a Packet(S)** S E High-Speed to Full-Speed Isochronous OUT Data Relation 0 0 High-speed data is the middle of the full-speed data payload 0 1 High-speed data is the end of the full-speed data payload 1 0 High-speed data is the beginning of the full-speed data payload 1 1 High-speed data is all of the full-speed data payload This is used for Isochronous OUT to a Full Speed device.
43	RO	**OUT Packet Valid Flag** This bit will be written 1 by hardware when OUT Transaction DMA done.
42:32	RO	**Packet Length in Bytes** This field will be written by DMA to indicate the received packet length, not include the CRC length.

Offset: 63:32 **DES_1: Descriptor Control/Status (IN)** **Init = X**

Bit	R/W	Description
63	RW	**Enable Interrupt Generation** 0 : No interrupt 1 : Interrupt generated when transaction done. This bit is used by SW to control the generation of interrupt, when SW wants to reduce the interrupts number. By this interrupt control, interrupt will be generated on this endpoint only when : a. Interrupt generation enable = 1 and transaction done for this descriptor. b. Descriptor list empty and the last transaction done. c. Stall response occurs. d. NAK response occurs and descriptor list empty and NAK interrupt enabled.
62:48	RO	**Reserved**
47:46	RW	**Data Packet Start PID** 00 : DATA0 01 : DATA2 10 : DATA1 11 : MDATA The Data PID will be used for transmitting if the HW auto data toggle (EPP00.bit[13]) is disabled.
45	RO	**Reserved**
44:32	RW	**Packet Length in Bytes** CPU must initialize this field to indicate the packet length for transmit. Allowable length is 0 ~ 4096 bytes.

18.6.7 Register Reset Table

Reg	Bit	SCU bit[14]	Bus Reset	HUB20 bit[0]	HUB20 bit[7:1]	HUB20 bit[9]	EPP00 bit[0]	Others
HUB00	30:16	Y						
HUB00	15:14	Y						HUB00[10:8] != 111
HUB00	13:0	Y						
HUB04	6:0	Y	Y					
HUB08	17:0	Y						
HUB0C	17:16	Y	Y			Y		
HUB0C	15:9	Y	Y		Y(Device specific)			
HUB0C	8:0	Y						
HUB10	20:0	Y						
HUB14	20:0	Y						
HUB18	20:0	Y	Y	Y(EPP00[3:1] port specific)		Y	Y(EP Specific)	
HUB1C	20:0	Y	Y	Y(EPP00[3:1] port specific)		Y	Y(EP Specific)	
HUB20	9:0	Y(All 1)						
HUB2C	25:16	Y	Y	Y				
HUB2C	9:0	Y	Y	Y				
HUB30	2:0	Y	Y	Y				
HUB38	1:0	Y						
HUB3C	7:0	Y						
DEV00	14:8	Y	Y					
DEV00	6:1	Y						
DEV00	0	Y	Y					
DEV04	4:0	Y	Y		Y(Device specific)			
DEV08	28:24	Y	Y		Y(Device specific)			
DEV08	2:0	Y	Y		Y(Device specific)			
EPP00	25:0	Y						
EPP04	12:8	Y	Y	Y(EPP00[3:1] port specific)		Y	Y	
EPP04	7:4	Y	Y	Y(EPP00[3:1] port specific)		Y	Y	EPP04[2] = 1
EPP04	3:0	Y						
EPP0C	15:8	Y	Y	Y(EPP00[3:1] port specific)		Y	Y	EPP04[1] = 1
EPP0C	7:0	Y	Y	Y(EPP00[3:1] port specific)		Y	Y	

1. For all other registers not included, no reset control implemented.

18.7 Software Programming Guide

18.7.1 Reset Control

The RESET control of USB2.0 Virtual Hub Controller includes the following types :

Global Reset

This is an asynchronous reset control and will reset full controller to its initial state, including all registers.
This reset can be initiated from SCU040[14] reset control register.

Bus State Reset

When upstream USB host controller issues a Bus Reset state, then the root hub controller will be reset to an un-configured state, which the device address equals ZERO. And all downstream device controllers and endpoints will be removed from plugging state and all state machines and counters will be reset to its initial state.
SW must send a bus reset command to remote USB host controller and restart the device initialization process.

Device Reset

HUB20 contains reset control for each device. The reset control only reset the device state into its initial state; register value will not be cleared. The states that will be reset including all state machines, FIFO pointers and descriptor pointers. For all unused downstream device ports, the device reset must be enabled. The following is a sequence for controlling device reset:

- Attach a device.
  1. Disable the specific device port reset.
  2. Enable device port and do the initialization sequence.
- Remove a device.
  1. Disable device port function.
  2. Enable the specific device port reset.

Endpoint Pool Reset

The Pool of 21 programmable endpoints can be reset by the following conditions:

1. Global reset at SCU040[14] will reset all registers to its initial value defined.
2. HUB20 [9] will reset all endpoints in the endpoint pool.
3. Upstream USB Bus Reset will reset all endpoints.
4. The specific device's reset that it was allocated.
5. Endpoints disabled will also been reset.

18.7.2 Initialization Sequence

Hub Connection

1. Enable USB2.0 clock running(SCU080[14] = 1), wait 10 ms for clock stable
2. Disable USB2.0 global reset by setting SCU044[14] = 1

3. Enable VIC interrupt setting.
4. Write HUB0C = 0xffffffff
5. Write HUB18 = 0xffffffff
6. Write HUB1C = 0xffffffff
7. Write HUB20 = 0x2FE
8. Write HUB08 = interrupt required
9. Write HUB38 = 0x5
10. Write HUB00 = 0x800
11. Write HUB00 = 0x805
12. Start handshake with USB host controller for Hub configuration

Hub Disconnection

1. Enable USB2.0 global reset by setting SCU040[14] = 1
2. Disable USB2.0 clock running by setting SCU084[14] = 1

Device Enable

1. Disable device reset at HUB20
2. Clear interrupt status at HUB0C
3. Write DEV04 = 0xff
4. Write DEV00 to the desired setting
5. Enable interrupt at HUB08

Downstream Device Attachment

1. Disable device and endpoint pool reset by setting the specific bit in HUB20 to '0'
2. Enable device and endpoint pool interrupt by setting the specific bit in HUB08 to '1'
3. Assign endpoint to device
4. Write DEV04 = 0xffffffff
5. Write DEV00 = 0x01 + (interrupt necessarily)
6. Set the specific port status change bit in HUB3C
7. Start handshake with USB host controller for device configuration

Endpoint Enable

1. Write EPP00 to the desired setting
2. Select single buffer mode
   (a) Write EPP04 = 0x6
   (b) Write EPP04 = 0x2
   (c) Enable buffer

i. Write EPP08 = buffer base
ii. Write EPP0C = 0x1
iii. Wait interrupt

3. Select descriptor list mode
   (a) Write EPP04 = 0x4
   (b) Write EPP0C = 0x0
   (c) Write EPP08 = descriptor base address
   (d) Write EPP04 = 0x1 or 0x9
   (e) Enable buffer
      i. Set descriptor list
      ii. Write EPP0C = valid descriptor stage
      iii. Wait interrupt

Assign Endpoints to Device

1. Enable endpoint interrupt by setting the specific bit in HUB10 and HUB14 to '1'
2. Enable endpoint, assign the device port, set the endpoint type and endpoint number
3. Create endpoint's DMA, single buffer mode or descriptor list mode

18.7.3 Set Device Address

Hub Controller

– When received Set_Address command, **Modify the controller device address field with the new address**.
– Prepare a zero length data packet for the status phase IN transaction.
– Wait and clear the IN transaction ACK interrupt.

Downstream Device Controller

– When received Set_Address command, **Modify the controller device address field with the new address**.
– Prepare a zero length data packet for the status phase IN transaction.
– Wait and clear the IN transaction ACK interrupt.

18.7.4 Response STALL

When software wants to response a STALL handshake to Host, then Host will send a Clear_Feature command to endpoint. When software received this command, it must clear the STALL bit and do a toggle reset operation to reset the data toggle sequence of this endpoint to DATA0.

The toggle reset command is defined at HUB28 and HUB38.

18.7.5 Programmable Endpoint OUT Transfer Finish Check

For programmable endpoints OUT transfer that using descriptor list mode. Software must check both the descriptor read pointer and the ”valid” bit in the descriptor bit[43] to determine a finished transfer. Especially for the ”valid” bit in the descriptor list. Sometimes there is a timing racing problem between software find a change of the read pointer and the valid bit updated. This is caused by DRAM controller arbitration priority mechanism. So somtimes CPU read an empty descriptor before the correct descriptor write into DRAM. This case happens rarely, but it is possible to happen. So software must polling the valid bit several times after it received an interrupt and detected the change of descriptor read pointer.

18.7.6 Prevent a Transient Read Pointer Value

There is a possibility to read a temp value for the descriptor read pointer. This is caused by the USB registers read and the pointer update happened on 2 clock domains. So it is possible to read a wrong transient value when the pointer is updating. This fail case often will be encountered when multiple stages descriptor enabled concurrently. To prevent the fail case, SW must continuously read the descriptor read pointer until 2 consecutive of same value got.

18.7.7 Procedure to enable Interrupt

The USB2.0 interrupt function must follow the sequence to enable.

1. Enable USB2.0 clock at SCU080[14]=1 and wait 10 ms for clock stable.
2. Disable USB2.0 global reset at SCU040[14]=0.
3. Set interrupt type as sensitive-high level trigger at VIC24[5]=1 and VIC2C[5]=1.
4. Enable USB2.0 interrupt at VIC10[5]=1.

18.7.8 OUT Direction Endpoint Maximum Packet Size Setting

- Control endpoint, the default maximum packet size setting is 64 bytes. SW must allocate 64 bytes buffer for DMA buffer, no matter what speed it belongs to.
- Programmable endpoint, the maximum packet size setting was defined at EPP00[25:16]. SW must allocate with the maximum size memory for each DMA buffer.

19 USB2.0 Device Controller (USB20D)

19.1 Overview

USB2.0 Device Controller implements 1 control endpoint and 4 programmable endpoints. The physical address of related registers can be derived as the following:

Base address of USB2.0 Device Controller = 0x1E6A_2000
Physical address = (Base address) + Offset

PHYB00: PHY Control/Status #1
PHYB04: PHY Control/Status #2
PHYB08: PHY Control/Status #3
PHYB0C: PHY Control/Status #4
UBD00: Root Function Control & Status Register
UBD04: Root Configuration Setting Register
UBD08: Interrupt Control Register
UBD0C: Interrupt Status Register
UBD10: Programmable Endpoint Pool ACK Interrupt Enable Register
UBD14: Programmable Endpoint Pool NAK Interrupt Enable Register
UBD18: Programmable Endpoint Pool ACK Interrupt Status Register
UBD1C: Programmable Endpoint Pool NAK Interrupt Status Register
UBD20: Device Controller Soft Reset Enable Register
UBD24: USB Status Register
UBD28: Programmable Endpoint Pool Data Toggle Value Set
UBD2C: Isochronous Transaction Fail Accumulator
UBD30: Endpoint 0 Control/Status Register
UBD34: Base Address of Endpoint 0 IN/OUT Data Buffer Register
EPP00: Endpoint Configuration Register
EPP04: DMA Descriptor List Control/Status Register
EPP08: DMA Descriptor/Buffer Base Address
EPP0C: DMA Descriptor List Read(DMA)/Write(CPU) Pointer and Status
DES_0: Data Buffer Base Address
DES_1: Descriptor Control/Status

19.2 Features

- Complies with the Universal Serial Bus specification Rev. 2.0, Supports USB Full Speed (12Mb/sec) and High Speed (480 Mb/sec), backward compatible with USB1.1.
- Supports 1 Control endpoint.
- Supports 4 programmable endpoints that can be assigned to any devices, and can be configured to Bulk IN/OUT, Interrupt IN/OUT and Isochronous IN/OUT type endpoint.
- Automatic retry of failed packets, and PING Flow control.
- Separate data buffers for the SETUP data of a CONTROL transfer
- Integrated DMA engine for direct memory bus accesses (bypass AHB bus).
- Supports independent DMA channel for each endpoint.
- Supports 256 stages descriptor mode for all 4 programmable endpoints.
- Supports USB remote wake-up function (Suspend/Resume operation).
- All registers are backward compatible to USB2.0 UBD controller.

19.3 Procedure to enable USB2.0 Device port

1. Set SCU440 bit[29:28] = ”01”, select USB2.0 port 2 mode as device
2. Set SCU040 bit[3] = 1, enable controller reset
3. Set SCU084 bit[7] = 1, enable PHY clock
4. wait 10 ms for PLL locking
5. Set SCU044 bit[3] = 1, disable controller reset
6. Set UBD00 bit[11] = 1, disable PHY reset
7. Load driver

19.4 Procedure to enable USB2.0 eye diagram measurement

- Measure Host TX waveform, enable device port termination
  1. Measurement point on the AST2600 side that is mostly close to the USB pin position.
  2. Force host (PCH) to send out Test Packet pattern, and follow below sequence to set AST2600 into high speed termination mode.
  3. Enter firmware u-boot
  4. mw 1e6e2000 1688a8a8
  5. mw 1e6e2440 10000000
  6. mw 1e6e2084 80
  7. mw 1e6e2044 8
  8. mw 1e6a2000 b41
- Measure Device TX waveform
  1. Measurement point on the host (PCH) side that is mostly close to the USB pin position.
  2. Force host (PCH) USB port to enable high speed termination for device eye diagram measurement.
  3. Enter BMC firmware u-boot
  4. mw 1e6e2000 1688a8a8
  5. mw 1e6e2440 10000000
  6. mw 1e6e2084 80
  7. mw 1e6e2044 8
  8. mw 1e6a2000 c41

19.5 Registers : Base Address = 0x1E6A:2000

19.5.1 Address Definition

Offset	Size(Byte)	Description
0x03F-0x000	64	Root/Global Device Register
0x087-0x080	8	Root Device SETUP Data Buffer
0x20F-0x200	16	Programmable Endpoint 0 Register
0x21F-0x210	16	Programmable Endpoint 1 Register
0x22F-0x220	16	Programmable Endpoint 2 Register
0x23F-0x230	16	Programmable Endpoint 3 Register
0x81F-0x800	32	PHY Control Register

19.5.2 PHY Control Register (Host/Device)

Offset: 00	PHYB00: USB PHY Control/Status #1	Init = 0
Bit	**R/W**	**Description**
31:16	RO	**PHY debug output**
15:12	RW	**PHY debug mode selection**
11	RW	**Enable USB2.0 Host Controller #2 128/256 Bytes Transmit FIFO Threshold (for debugging purpose only)** 0: disable 1: enable When enabled, EHCI84[7:6] can be set to 00/01
10	RW	**Improve USB2.0 Host Controller #2 asynchronous list performance (for debugging purpose only)** 0: disable 1: enable
9	RW	**Enable PHY to output 120MHz and 480MHz clock** 0: disable 1: enable
8	RW	**Enable 8 bit UTMI interface** 0: 16 bits UTMI 1: 8 bits UTMI When configured as host to device direct connected mode, it should be set as 8 bits UTMI mode.
7	RW	**DN 15K pull down enable (Device mode)**
6	RW	**DP 15K pull down enable (Device mode)**
5	RW	**PHY output Low Swing mode**
4	RW	**PHY BIST mode load control**
3:0	RW	**PHY BIST mode selection**

Offset: 04	PHYB04: USB PHY Control/Status #2	Init = 0x00C00000
Bit	**R/W**	**Description**
31:0	RW	**PHY XCFGI[31:0]**

Offset: 08	PHYB08: USB PHY Control/Status #3	Init = 0x00000001
Bit	**R/W**	**Description**
31:0	RW	**PHY XCFGI[63:32]**

Offset: 0C	PHYB0C: USB PHY Control/Status #4	Init = 0x00024903
Bit	**R/W**	**Description**
31:24	RO	**PHY XCFGO**
23:18	RO	**Reserved**
17:15	RW	**PHY XCfg_LOCK_RANGE_MIN**
14:12	RW	**PHY XCfg_LOCK_RANGE_MAX**
11:9	RW	**PHY XCfg_FINE_TUNE_NUM**
8:6	RW	**PHY XCfg_COARSE_TUNE_NUM**
5:0	RW	**PHY XCFGI[69:64]**

19.5.3 Root/Global Register Definition

Offset: 00	UBD00: Root Function Control & Status Register	Init = 0
Bit	**R/W**	**Description**
31	RO	**USB PHY clock enable status** 0: USB PHY clock is disabled by SCU080 bit[7]. 1: USB PHY clock is enabled. The procedure to enable USB2.0 controller was below : 1. Enable USB2.0 clock running(SCU080[7] = 0), wait 10 ms for clock stable 2. Disable USB2.0 global reset by setting SCU044[3] = 1 3. Disable USB2.0 PHY reset by setting UBD00[11] = 1 4. Start using USB2.0 controller
30:20	RO	**Reserved (0)**
19	RW	**Enable FIFO dynamic power down** 0: disable 1: enable
18	RW	**Programmable endpoint long descriptor list mode** 0: 32 stages descriptor 1: 256 stages descriptor This bit will affect the full endpoint pool.
17	RW	**Isochronous IN null data response control** 0: No response, wait host timeout 1: Return 0 byte DATA0 packet This bit controls the response action for Isochronous IN type endpoints when the IN data for transmitting not ready. The action can be no response or return an 0 byte DATA0 packet. When no response action is selected, then CSPLIT IN retry is possible.
16:14	RO	**Reserved (0)**
13	RO	**USB PHY BIST result** 0: Fail 1: Pass This flag will be cleared by disabling USB PHY BIST function (UBD00 [12] = 0). Note: BIST stands for Built-In-Self-Test

to next page

from previous page

12	RW	**USB PHY BIST control** 0: Turn off USB PHY BIST 1: Turn on USB PHY BIST
11	RW	**Disable USB PHY reset** 0: Enable USB PHY reset 1: Disable USB PHY reset
10:8	RW	**USB Test Mode selection** 000: Disable 001: Enable Test J 010: Enable Test K 011: Enable Test SE0_NAK 100: Enable Test Packet 101: Reserved 110: Reserved 111: Reserved ~~Enable Test Loop Back~~ (for debugging purpose only)
7	RW	**Force USB bus state timer to work at test mode** **(for debugging purpose only)** 0: Normal operation mode 1: Force USB bus state to High Speed mode (X32 faster)
6	RW	**USB Force to High Speed State Mode** **(for debugging purpose only)** 0: Normal operation 1: Force the bus state to High Speed
5	RW	**USB Remote Wakeup signaling pulse width selection** 0: 8ms 1: 12ms
4	RW	**Enable manual Remote Wakeup** 0: No operation 1: Enable manual Remote Wakeup, can be set only in Suspend state This register can be effectively set to ”1” only when USB Controller enters Suspend state, and this register will be automatically cleared by H/W after Remote Wakeup finished.
3	RW	**Enable automatic Remote Wakeup** 0: Disable 1: Enable automatic Remote Wakeup When this register is enabled, Remote Wakeup singling will be automatically issued whenever firmware write commands has been received in Suspend state.
2	RW	**Enable clock stopping in suspend state** 0: USB Controller won’t stop clock even in Suspend state 1: USB Controller will stop clock in Suspend state
1	RW	**Upstream port connection speed selection** 0: Select High Speed and Full Speed modes 1: Select Full Speed mode only
0	RW	**Enable upstream port connection** 0: Disable upstream port connection 1: Enable upstream port connection

Offset: 04 **UBD04: Root Configuration Setting Register** **Init = 0**

Bit	R/W	Description
31:20	RO	**Reserved**

to next page

from previous page

19:16	RO	**Status of DMA page buffer** **(for debugging purpose only)** 0: Page Free 1: Page Allocated Bit16: Status of Page #0 .... Bit19: Status of Page #3 USB Controller totally integrates 1K byes of SRAM to be allocated for data transmit of IN transaction. The 1K bytes of SRAM is uniformly divided into 4 pages, each of them is 256 bytes long. Each EP can only allocate one page buffer from them, but there are 2 pages (Page 0 and Page 1) are arranged as a ring buffer and dedicated for the usage of the active EP. Therefore, the status bits of the two pages are reserved.
15:7	RO	**Reserved**
6:0	RW	**Root function device address** Change the address whenever Set_Address command was received. The following is the Set_Address command sequence: 1. SETUP data packet, contains the new address information. 2. Change to the new address. 3. IN data packet, status phase with zero byte data returned.

Offset: 08 **UBD08: Interrupt Control Register** **Init = 0**

Bit	R/W	Description
31:18	RO	**Reserved (0)**
17	RW	**Enable Programmable Endpoint Pool NAK Interrupt**
16	RW	**Enable Programmable Endpoint Pool ACK/STALL Interrupt**
15:9	RO	**Reserved (0)**
8	RW	**Enable USB Suspend Resume Interrupt** This interrupt is used to notify software that the USB host is leaving suspend mode. Software can do something because USB device will wake up. If software doesn't need to know the resume event, please don't enable this bit, so that hardware will skip to wait software to do something.
7	RW	**Enable USB Suspend Entry Interrupt** This interrupt is used to notify software that the USB host is entering suspend mode. Software can do something because USB device will entering sleep state. If software doesn't need to know the suspend event, please don't enable this bit, so that hardware will skip to wait software to do something.
6	RW	**Enable USB Bus Reset Interrupt** When Bus Reset occurred, hardware will automatically reset the connection status and return to the not configured state. Software must also clear the related states and prepare for the new connection.
4	RW	**Enable EP0 IN Data packet NAK Interrupt**
3	RW	**Enable EP0 IN Data packet ACK/STALL Interrupt**
2	RW	**Enable EP0 OUT Data packet NAK Interrupt**
1	RW	**Enable EP0 OUT Data packet ACK/STALL Interrupt**
0	RW	**Enable EP0 SETUP Data packet ACK Interrupt**

to next page

from previous page

Note :
The definition of this register is :
0 : Disable
1 : Enable

The enable control for Endpoint Pool is the first level interrupt enable bit for all the EPs allocated from Endpoint Pool.

The enable control for Device is the first level interrupt enable bit for all the downstream device controller.

Offset: 0C	UBD0C: Interrupt Status Register	Init = 0

Bit	R/W	Description
31	RO	**Received memory FIFO empty flag** (for debugging purpose only) 0: FIFO is not empty 1: FIFO is empty Monitor the FIFO is empty or not. It can receive next packet if FIFO is empty.
30:18	RO	**Reserved (0)**
17	RO	**Programmable Endpoint Pool NAK Interrupt Occurs**
16	RO	**Programmable Endpoint Pool ACK/STALL Interrupt Occurs**
15:9	RO	**Reserved (0)**
8	RW	**USB Suspend Resume event has occurred** When this register is set to ”1”, it indicates that USB Upstream Port has resumed from Suspend state.
7	RW	**USB Suspend Entry event has occurred** When this register is set ”1”, it indicates that the USB Upstream Port has entered Suspend state.
6	RW	**USB Bus Reset has Occurred** When set, indicates a USB Bus Reset state occurs.
4	RW	**EP0 IN Data Packet NAK Returned** When set, indicates an IN packet received and responded with NAK. Control Endpoint IN/OUT NAK response interrupt can used to indicate the current Host transaction type, and S/W can based on this to determine the current Control Transfer stage.
3	RW	**EP0 IN Data Packet ACK/STALL Returned** When set, indicates an IN transaction finished with ACK or STALL transmitted.
2	RW	**EP0 OUT Data Packet NAK Returned** When set, indicates an OUT/PING packet received and responded with NAK. Control Endpoint IN/OUT NAK response interrupt can used to indicate the current Host transaction type, and S/W can based on this to determine the current Control Transfer stage.
1	RW	**EP0 OUT Data Packet ACK/STALL Returned** When set, indicates an OUT transaction finished with ACK/STALL returned, or a PING packet received and responded with STALL.
0	RW	**EP0 Setup Data Arrives** When set, indicates an SETUP transaction has been received successfully.

Note :
Each status bit of this register is set by H/W automatically when the corresponding event occurred or the transaction finished. The corresponding interrupt enable bit wont impact the setting of the bit. The enable bit only determines the interrupt generation.
Whenever S/W has finished the handling of the status bit, it must clear the status bit by writing ’1’, else it will not be able to recognize new events.

Offset: 10	UBD10: Programmable Endpoint Pool ACK Interrupt Enable Register	Init = 0
Bit	**R/W**	**Description**
31:4	RO	**Reserved (0)**
3:0	RW	**Programmable Endpoint ACK Interrupt Enable** bit 0: Programmable Endpoint number #0 bit 1: Programmable Endpoint number #1 bit 2: Programmable Endpoint number #2 bit 3: Programmable Endpoint number #3 This register is to control the ACK interrupt enable bits of the 4 programmable Endpoints in the Endpoint Pool. The definition of each bit in this register is as the following: 0: Disable interrupt 1: Enable interrupt

Offset: 14	UBD14: Programmable Endpoint Pool NAK Interrupt Enable Register	Init = 0
Bit	**R/W**	**Description**
31:4	RO	**Reserved (0)**
3:0	RW	**Programmable Endpoint NAK Interrupt Enable** bit 0: Programmable Endpoint number #0 bit 1: Programmable Endpoint number #1 bit 2: Programmable Endpoint number #2 bit 3: Programmable Endpoint number #3 This register is to control the ACK interrupt enable bits of the 4 programmable Endpoints in the Endpoint Pool. The definition of each bit in this register is as the following: 0: Disable interrupt 1: Enable interrupt

Offset: 18	UBD18: Programmable Endpoint Pool ACK Interrupt Status Register	Init = 0
Bit	**R/W**	**Description**
31:4	RO	**Reserved (0)**
3:0	RW	**Programmable Endpoint ACK Interrupt Occurs** bit 0: Programmable Endpoint number #0 bit 1: Programmable Endpoint number #1 bit 2: Programmable Endpoint number #2 bit 3: Programmable Endpoint number #3 This status flag will be set to ’1’ under any one of the following conditions: 1. STALL response. 2. When short packet received from OUT transaction. 3. When Interrupt Generation Enable been detected from the DMA descriptor, or for a single descriptor mode. 4. When the DMA descriptor list becomes empty, this indicates that the last descriptor been used.
Note : Each status is automatically set to ”1” whenever the ACK event has been occurred, no matter the corresponding interrupt enable bit has been enabled or not. S/W must clear each status by writing ’1’ to its bit after finished the process; otherwise the next ACK event will not be recognized.		

Offset: 1C	UBD1C: Programmable Endpoint Pool NAK Interrupt Status Register	Init = 0
Bit	**R/W**	**Description**
31:4	RO	**Reserved (0)**
3:0	RW	**Programmable Endpoint NAK Interrupt Occurs** bit 0: Programmable Endpoint number #0 bit 1: Programmable Endpoint number #1 bit 2: Programmable Endpoint number #2 bit 3: Programmable Endpoint number #3 This status flag will be set to '1' when the endpoint response with NAK.
Note : Each status is automatically set to "1" whenever the NAK event has been occurred, no matter the corresponding interrupt enable bit has been enabled or not. S/W must clear each status by writing '1' to its bit after finished the process; otherwise the next NAK event will not be recognized.		

Offset: 20	UBD20: Device Controller Soft Reset Enable Register	Init = 0x301
Bit	**R/W**	**Description**
31:10	RO	**Reserved (0)**
9	RW	**Enable Programmable Endpoint Pool software Reset**
8	RW	**Enable DMA Controller software Reset**
7:1	RO	**Reserved (0)**
0	RW	**Enable Root UBD Controller software Reset**
Note : 0 : Normal operation 1 : Reset the device controller These software reset bits only reset the controllers status registers, not including all the registers supported by the controller. To reset all the registers of the controllers, please reference the registers of SCU04. Software sets the specific bit to '1' to start the reset process, and sets the specific bit to '0' to stop the reset process. There is no need to put time delay between the two processes.		

Offset: 24	UBD24: USB Status Register (for debugging purpose only)	Init = X
Bit	**R/W**	**Description**
31	RO	**USB Suspend State**
30	RO	**USB Bus Reset State**
29	RO	**USB Bus Line State DN**
28	RO	**USB Bus Line State DP**
27	RO	**USB Bus Speed** 0: Full Speed 1: High Speed SW reads this bit to determine the current host connection speed. But SW must read this bit after the first packet is received that behind the bus reset cycle.
26:16	RO	**USB Last Frame Number record**
15	RO	**UTMI State XcvrSelect** 0: High Speed 1: Full Speed
14	RO	**UTMI State TermSelect** 0: High Speed 1: Full Speed

to next page

from previous page

13:12	RO	**UTMI State OPMode** 0: Normal mode 1: Non Driving 2: Disable Bit-Stuff and NRZI encoding 3: Reserved
11:8	RO	**Endpoint Number of the Last USB Transaction**
7	RO	**Reserved (0)**
6:0	RO	**Device Address of the Last USB Transaction**

Offset: 28 **UBD28: Programmable Endpoint Pool Data Toggle Value Set** **Init = X**

Bit	R/W	Description
31:9	RO	**Reserved (0)**
8	WO	**Endpoint data toggle bit initial value set** 0: Initial to sequence DATA0 1: Initial to sequence DATA1 The indexed data toggle bit is determined by Bit [1:0] of this register. Only one data toggle bit can be initialized for each register write. Reading this register always returns ”0”.
7:2	RO	**Reserved (0)**
1:0	WO	**Programmable Endpoint Index** 0-3: Endpoint number index This index value determines which Endpoint data toggle bit will be initialized. Reading this register always returns ”0”.

Note :
Data toggle sequence initialization can only be applied to Control/Bulk/Interrupt type Endpoints.
Isochronous type Endpoints should not be initialized; it will be reset automatically when SOF receives.

UBD2C: Isochronous Transaction Fail Accumulator (for debugging purpose only)

Offset: 2C **Init = 0**

Bit	Attr.	Description
31:26	RO	**Reserved (0)**
25:16	RW	**Isochronous OUT Failure Counter**
15:10	RO	**Reserved (0)**
9:0	RW	**Isochronous IN Failure Counter**

Note :
Writing any data to this register will clear the two counters to 0.
The two counter values show the number of isochronous transaction failures which are due to either buffer unavailable or data not yet ready.

Offset: 30 **UBD30: Endpoint 0 Control/Status Register** **Init = 0**

Bit	R/W	Description
31:23	RO	**Reserved (0)**
22:16	RO	**Endpoint 0 OUT received data byte count**
15	RO	**Reserved (0)**
14:8	RW	**Endpoint 0 IN data byte count for transfer**
7:3	RO	**Reserved (0)**

to next page

from previous page

2	RW	**Endpoint 0 OUT buffer ready for receiving data** 0: Buffer is not ready to receive data 1: Buffer is ready to receive data S/W can set this bit to '1' when it is ready to receive data from Host by OUT transactions. When H/W receives the data successfully, this bit will be automatically cleared to "0". S/W can monitor this bit to check the data has been received or not.
1	RW	**Endpoint 0 IN buffer ready for transferring data** 0: Data is not ready to transfer data 1: Data is ready to transfer data S/W can set this bit to '1' when there is a need to transfer data to Host by IN transactions. When H/W has finished the data transfer, this bit will be automatically cleared. S/W can monitor this bit to check the data has been transferred or not. Only one of Bit [1] and Bit [2] can be set to "1" for each time. The two settings are exclusive.
0	RW	**Endpoint 0 STALL control** When this register is set to "1", Endpoint 0 returns STALL response for Data and Status stages. This bit is automatically cleared to "0" after a new SETUP packet is received.

Offset: 34 UBD34: Base Address of Endpoint 0 IN/OUT Data Buffer Register Init = X

Bit	R/W	Description
31	RO	**Reserved (0)**
30:0	RW	**Base address of Endpoint 0 IN/OUT data buffer** This register defines the base address of Default Control Endpoint transaction data buffer, which is 64 bytes long and at 64-bit boundary. The direction of this buffer is determined by S/W and depends on the Control Transfer Command mode.

19.5.4 Programmable Endpoint #0 — #3 Register Definition

Offset: 00 EPP00: Endpoint Configuration Register Init = 0

Bit	R/W	Description
31:28	RW	**List mode pre-empty interrupt control** 0: disable pre-empty interrupt control. Define the stages number before list empty to issue interrupt. This is used by firmware to cover the time penalty to enter interrupt service routine. Interrupt will be issued whenever a transaction finished and the reset descriptor stages equal to this register setting. Only applicable at 256 stages descriptor mode.
27	RW	**Burst transactions long idle interrupt control** 0: Disable 1: Enable long idle (over 1 SOF period idle) interrupt control for a burst transaction. Interrupt will be issued when a burst transaction stop received for over 1 SOF cycle period. This can help firmware to reduce interrupt event number for a burst transfer.
26	RO	**Reserved (0)**

to next page

from previous page

<table>
<tr><td>25:16</td><td>RW</td><td>Endpoint Maximum Packet Size Definition
0: 1024 bytes
1: 1 byte
....
1023: 1023 bytes

– For OUT direction endpoint, this definition limits the maximum packet size can be received. If extra bytes received, it will not be transfered to DRAM and return with NAK response to retry this transaction.
– For IN direction endpoint, it must be used in companion with EPP04[3]. It defines the maximum bytes to be returned within 1 packet. This enables a possibility to pack several packets into a descriptor stage. The total size that can be packed in a descriptor stage equals to 4095 bytes or 8 sub-packets, depends on which criteria meet first.</td></tr>
<tr><td>15:14</td><td>RW</td><td>Endpoint Data Fetch Control
1. Isochronous type endpoint.
Defines the auto data toggle stage setting under High speed.
<table><tr><th>Value</th><th>Stages</th><th>Isochronous IN</th><th>Isochronous OUT</th></tr><tr><td>00</td><td>1</td><td>DATA0</td><td>DATA0</td></tr><tr><td>01</td><td>2</td><td>DATA1→DATA0</td><td>MDATA→DATA1</td></tr><tr><td>1x</td><td>3</td><td>DATA2→DATA1→DATA0</td><td>MDATA→MDATA→DATA2</td></tr></table>When SOF received, the data PID sequencing will be reset to the start data PID.
For long packet (transmit byte count > maximum packet size) or high bandwidth Isochronous IN, please enable the auto data toggle mode, EPP00[13] = 0.

2. Interrupt type endpoint.
Defines the prefetch mode when long packet or high bandwidth Interrupt IN.
00: No prefetch. Next data will be fetched after transaction successfully.
01: Prefetch the next packet data once after its was transmitted.</td></tr>
<tr><td>13</td><td>RW</td><td>Endpoint Auto Data Toggle Disable
0 : Enable auto data toggle
1 : Disable auto data toggle
When auto data toggle is disabled for OUT direction endpoint, then all packets will be received without error, ignoring the data sequence errors.
When auto data toggle is disabled for IN direction endpoint, then the data sequence PID will be replaced by the PID fetched from the descriptor list.</td></tr>
<tr><td>12</td><td>RW</td><td>Endpoint Stall Control
When this bit is set to 1, the endpoint will always return STALL response until it is cleared to 0.
The Stall control can only be set for Bulk/Interrupt type endpoints.</td></tr>
<tr><td>11:8</td><td>RW</td><td>Endpoint Number
This value defines the endpoint number of this endpoint.</td></tr>
<tr><td>7</td><td>RO</td><td>Reserved (0)</td></tr>
</table>

to next page

from previous page

6:4	RW	**Endpoint type selection** 00x : Disable 010 : Bulk In 011 : Bulk Out 100 : Interrupt In 101 : Interrupt Out 110 : Isochronous In 111 : Isochronous Out
3:1	RO	**Reserved (0)**
0	RW	**Enable Endpoint** 0 : Disabled (this endpoint will be reset) 1 : Enabled

Note :
Endpoint Reset can be initiated by the following method:

1. System global reset controlled in SCU, this will reset full controller including registers.
2. Release the endpoint, disable endpoint.
3. Set the reset bit at UBD20 Bit[9] or UBD20 Bit[n], where n is device number set in the Port Number field.

Only the first item will reset the register value, others don't.

Offset: 04 **EPP04: DMA Descriptor List Control/Status Register** **Init = 0**

Bit	R/W	Description
31:19	RO	**Reserved (0)**
18	RO	**Occupied Transmit IN Buffer Status** **(for debugging purpose only)** 0 : No buffer 1 : 1 buffer occupied
17:16	RO	**Occupied Transmit IN Buffer Index** **(for debugging purpose only)**
15	RW	**Long idle interrupt status** 0: No interrupt. 1: Interrupt status of long idle. Write '1' to clear this status bit.
14	RW	**Descrptor pre-empty interrupt status** 0: No interrupt. 1: Interrupt status of descriptor list pre-empty. Write '1' to clear this status bit.
13	RW	**OUT transaction short packet interrupt status** 0: No interrupt. 1: Interrupt status of OUT short packet. Write '1' to clear this status bit.
12	RO	**The Current Interrupt Generation Flag** **(for debugging purpose only)** This bit shows the interrupt generation status, fetched from the current stage descriptor.
11	RO	**CSPLIT IN Wait** **(for debugging purpose only)**
10:9	RO	**Auto Data Toggle Count** **(for debugging purpose only)**
8	RO	**Start SPLIT Cycle** **(for debugging purpose only)**

to next page

from previous page

7:4	RO	**Current Descriptor Processing Status** **(for debugging purpose only)** 00 : RX Idle 01 : RX Read Descriptor Request 02 : RX Read Descriptor Grant 03 : RX Read Descriptor Data Back and Buffer Ready 04 : RX OUT Data Receive Cycle 05 : RX OUT Transaction ACK and Descriptor Write Back Request 06 : RX Descriptor Write Back Grant 07 : RX DMA Done 08 : TX Idle 09 : TX Read Descriptor Request 10 : TX Read Descriptor Grant 11 : TX Read Descriptor Data Back and Buffer Ready 12 : TX IN Data DMA Fetch Request 13 : TX IN Data Ready 14 : TX IN Data Transfer Cycle 15 : TX IN Transaction DONE
3	RM	**DMA buffer mode selection** **For OUT direction endpoint** Defines the DMA buffer cascade mode. This mode can shorten the inter-packet latency, and increase the performance. 0: DMA buffers can be allocated anywhere. 1: DMA buffers are allocated continuously, each has size of Maximum packet size defined at EPP00[25:16]. **For IN direction endpoint** Enables the long data packet mode defined in one descriptor stage. In general, one descriptor only contains one packet data. Enable this mode can merge at most 8 packets or 4095 bytes of data into one descriptor stage. The data size of each packet is defined at EPP00[25:16]. This can decrease the interrupt number and increase the performance, especially for high speed high bandwidth transfer. 0: Normal mode, one descriptor only can contain one packet data. 1: Enable long packet support. **Any time this bit is turned ON or OFF, descriptor Reset is necessary.**
2	RW	**Descriptor List Operation Reset** Sets this bit to '1' will reset the descriptor operation and flush buffers. Sets this bit to '0' to disable reset.
1	RW	**Single Stage Descriptor Mode** 0 : 32/256 stages descriptor 1 : 1 stage descriptor, when the OUT/IN transaction done, the CPU Write pointer will be cleared to 0 automatically. The single mode operation is started by set the Write pointer at EPP0C to 1, this means the data/buffer ready for TX/RX transfer. And when transfer done, the Write pointer will be cleared to 0 by H/W.
0	RW	**Descriptor List Operation Enable** 0 : Disable, for single stage mode 1 : Enable normal operation The descriptor list operates at 32/256 stages Ring mode. Descriptor List Operation Enable and Single Stage Descriptor modes are mutually exclusive. They cannot be set at the same time. Single mode has the higher priority.

Offset: 08	EPP08: DMA Descriptor/Buffer Base Address	Init = X

Bit	R/W	Description
31	RO	**Reserved (0)**
30:0	RW	**Base address** Descriptor Enabled : Descriptor list base address (64 bits boundary). Descriptor Disabled : DMA data buffer base address (byte boundary).

EPP0C: DMA Descriptor List Read(DMA)/Write(CPU) Pointer and Status

Offset: 0C	Init = 0

Bit	Attr.	Description
31	RO	**Descriptor List Empty Flag (Default=0)** 0 : not empty 1 : empty
30	RW	**Long idle interrupt status** 0: No interrupt. 1: Interrupt status of long idle. Alias with EPP04[15]. Write '1' to clear this status bit.
29:28	RW	**Endpoint Current Data Toggle Sequence Value (Default=0)** 00 : Indicates the Data PID of next transaction would be DATA0 01 : Indicates the Data PID of next transaction would be DATA2 10 : Indicates the Data PID of next transaction would be DATA1 11 : Indicates the Data PID of next transaction would be MDATA **1.** When read, this value indicates the current internal register state used for next transaction Data sequence PID. **2.** When write, used to setting the next transaction Data sequence PID. Only valid at Single descriptor mode, endpoint type "IN" and auto data toggle disabled.
27	RO	**Reserved (0)**

to next page

from previous page

<table>
<tr><td>26:16</td><td>RW</td><td>
Packet Size (Default=X)
The unit is byte.
This field has 4 definitions:

<table>
<tr><td rowspan="2">Descriptor Type</td><td colspan="2">Endpoint Type</td></tr>
<tr><td>IN</td><td>OUT</td></tr>
<tr><td>Single</td><td>TxDataByteCnt(1)</td><td>RxDataByteCnt(3)</td></tr>
<tr><td>List</td><td>TxDataByteCnt(2)</td><td>MaxPacketSize(4)</td></tr>
</table>
(1) When endpoint type = IN and single mode enabled :
Transmit data packet length
This is the transmit data packet length for IN transfer.
Setting by SW.
(2) When endpoint type = IN and descriptor list mode :
Transmit data packet length
This is the transmit data packet length for IN transfer.
Fetched from descriptor list entry.
(3) When endpoint type = OUT and single mode enabled :
Received data packet length
This is the received data packet length of OUT transfer.
Setting by RXDMA controller.
(4) When endpoint type = OUT and descriptor list mode :
Endpoint Maximum Packet Size
This is used for OUT transaction short packet interrupt trigger.
When the OUT data length received less than the Maximum Packet Size,
then the OUT ACK interrupt flag will raise.
Setting by SW.
SW can write this value when descriptor is under disabled state, including single descriptor mode.
</td></tr>
<tr><td>15:8</td><td>RW</td><td>
Descriptor List DMA Read Pointer (Default=0)
This shows the current descriptor read position that will be read by DMA controller if it is not empty(equal to CPU write pointer).
This pointer can be initialized by S/W when descriptor list mode is disabled and not set at single mode. The correct initialize procedure is as follows:
1. Disable the descriptor list first, set EPP04 = 0
2. Wait descriptor operation status Idle EPP04[7:4] == 0 or 0x8
3. Update the Read Pointer
4. Enable the descriptor operation, set EPP04[0] = 1
</td></tr>
</table>

to next page

from previous page

7:0	RW	**Descriptor List CPU Write Pointer (Default=0)** For transmit (IN) direction, this pointer indicates the transmit data buffer allocated to DMA. For receive (OUT) direction, this pointer indicates the receiver free buffer allocated to DMA. When descriptor operation is enabled, this value indicates the next descriptor write position that will be writing by CPU. And the DMA operation will increment until read pointer equals to write pointer that indicates the Empty condition. The descriptor stage that Write Pointer addressed is not valid, and DMA will not process it. The descriptor list usage cannot be fully, there needs 1 free space for differentiation full and empty cases. That is when WPTR = RPTR, it means empty status. And full status equals WPTR = RPTR-1. The SW can maximum fills the descriptor entry until the full condition; otherwise it will conflict with the empty condition.
Note : The descriptor operation is that the Write Pointer is the leading pointer, and Read Pointer will track at the tail of Write Pointer. Read and Write pointer will be reset to 0 when device reset (UBD20) or USB bus reset occurs.		

19.5.5 Programmable Endpoint DMA Descriptor Definition

Offset: 31:0	DES_0: Data Buffer Base Address	Init = X

Bit	R/W	Description
31:0	RW	**DMA Data Buffer Base Address** Both OUT and IN direction endpoint support byte boundary.

Offset: 63:32	DES_1: Descriptor Control/Status (OUT)	Init = X

Bit	R/W	Description
63	RW	**Enable Interrupt Generation** 0 : No interrupt 1 : Interrupt generated when transaction done. This bit is used by SW to control the generation of interrupt, when SW wants to reduce the interrupts number. By this interrupt control, interrupt will be generated on this endpoint only when : a. Interrupt generation enable = 1 and transaction done for this descriptor. b. Descriptor list empty and the last transaction done. c. Stall response occurs. d. NAK response occurs and descriptor list empty and NAK interrupt enabled.
59:56	RO	**Endpoint Number** This endpoint number status only used for indicating the received transaction status. No any meaning for transmit path.
55	RO	**Reserved**
54:48	RO	**Device Address** This device address status only used for indicating the received transaction status. No any meaning for transmit path.
47:46	RO	**Data Packet PID** 00 : DATA0 01 : DATA2 10 : DATA1 11 : MDATA This is the Data PID that current packet received.
43	RO	**OUT Packet Valid Flag** This bit will be written 1 by hardware when OUT Transaction DMA done.
42:32	RO	**Packet Length in Bytes** This field will be written by DMA to indicate the received packet length, not include the CRC length.

Offset: 63:32	DES_1: Descriptor Control/Status (IN)	Init = X

Bit	R/W	Description
63	RW	**Enable Interrupt Generation** 0 : No interrupt 1 : Interrupt generated when transaction done. This bit is used by SW to control the generation of interrupt, when SW wants to reduce the interrupts number. By this interrupt control, interrupt will be generated on this endpoint only when : a. Interrupt generation enable = 1 and transaction done for this descriptor. b. Descriptor list empty and the last transaction done. c. Stall response occurs. d. NAK response occurs and descriptor list empty and NAK interrupt enabled.
62:48	RO	**Reserved**
47:46	RW	**Data Packet Start PID** 00 : DATA0 01 : DATA2 10 : DATA1 11 : MDATA The Data PID will be used for transmitting if the HW auto data toggle (EPP00.bit[13]) is disabled.
45	RO	**Reserved**
44:32	RW	**Packet Length in Bytes** CPU must initialize this field to indicate the packet length for transmit. Allowable length is 0 ~ 4096 bytes.

19.5.6 Register Reset Table

Reg	Bit	SCU bit[14]	Bus Reset	UBD20 bit[0]	UBD20 bit[9]	EPP00 bit[0]	Others
UBD00	30:16	Y					
UBD00	15:14	Y					UBD00[10:8] != 111
UBD00	13:0	Y					
UBD04	6:0	Y	Y				
UBD08	17:0	Y					
UBD0C	17:16	Y	Y		Y		
UBD0C	15:9	Y	Y				
UBD0C	8:0	Y					
UBD10	20:0	Y					
UBD14	20:0	Y					
UBD18	20:0	Y	Y	Y(EPP00[3:1] port specific)	Y	Y(EP Specific)	
UBD1C	20:0	Y	Y	Y(EPP00[3:1] port specific)	Y	Y(EP Specific)	
UBD20	9:0	Y(All 1)					
UBD2C	25:16	Y	Y	Y			
UBD2C	9:0	Y	Y	Y			
UBD30	2:0	Y	Y*	Y*			
EPP00	25:0	Y					
EPP04	12:8	Y	Y	Y(EPP00[3:1] port specific)	Y	Y	
EPP04	7:4	Y	Y	Y(EPP00[3:1] port specific)	Y	Y	EPP04[2] = 1
EPP04	3:0	Y					
EPP0C	15:8	Y	Y	Y(EPP00[3:1] port specific)	Y	Y	EPP04[1] = 1
EPP0C	7:0	Y	Y	Y(EPP00[3:1] port specific)	Y	Y	

1. For all other registers not included, no reset control implemented.
2. '*' indicates new added in this chip.

19.6 Software Programming Guide

19.6.1 Reset Control

The RESET control of USB2.0 Virtual Hub Controller includes the following types :

Global Reset

This is an asynchronous reset control and will reset full controller to its initial state, including all registers.
This reset can be initiated from SCU040[3] reset control register.

Bus State Reset

When upstream USB host controller issues a Bus Reset state, then the root hub controller will be reset to an un-configured state, which the device address equals ZERO. And all downstream device controllers and endpoints will be removed from plugging state and all state machines and counters will be reset to its initial state.
SW must send a bus reset command to remote USB host controller and restart the device initialization process.

Endpoint Pool Reset

The Pool of 4 programmable endpoints can be reset by the following conditions:

1. Global reset at SCU040[3] will reset all registers to its initial value defined.
2. UBD20[9] will reset all endpoints in the endpoint pool.
3. Upstream USB Bus Reset will reset all endpoints.
4. The specific device's reset that it was allocated.
5. Endpoints disabled will also been reset.

19.6.2 Initialization Sequence

Root Connection

1. Enable USB2.0 clock running(SCU080[7] = 0), wait 10 ms for clock stable
2. Disable USB2.0 global reset by setting SCU040[3] = 0
3. Enable VIC interrupt setting.
4. Write UBD0C = 0xffffffff
5. Write UBD18 = 0xffffffff
6. Write UBD1C = 0xffffffff
7. Write UBD20 = 0x2FE
8. Write UBD08 = interrupt required
9. Write UBD00 = 0x800
10. Write UBD00 = 0x805
11. Start handshake with USB host controller for Hub configuration

Root Disconnection

1. Enable USB2.0 global reset by setting SCU040[3] = 1
2. Disable USB2.0 clock running by setting SCU080[7] = 1

Endpoint Enable

1. Write EPP00 to the desired setting
2. Select single buffer mode
   (a) Write EPP04 = 0x6
   (b) Write EPP04 = 0x2
   (c) Enable buffer
      i. Write EPP08 = buffer base
      ii. Write EPP0C = 0x1
      iii. Wait interrupt
3. Select descriptor list mode
   (a) Write EPP04 = 0x4
   (b) Write EPP0C = 0x0
   (c) Write EPP08 = descriptor base address
   (d) Write EPP04 = 0x1 or 0x9
   (e) Enable buffer
      i. Set descriptor list
      ii. Write EPP0C = valid descriptor stage
      iii. Wait interrupt

Assign Endpoints to Device

1. Enable endpoint interrupt by setting the specific bit in UBD10 and UBD14 to '1'
2. Enable endpoint, assign the device port, set the endpoint type and endpoint number
3. Create endpoint's DMA, single buffer mode or descriptor list mode

19.6.3 Set Device Address

- When received Set_Address command, **Modify the controller device address field with the new address**.
- Prepare a zero length data packet for the status phase IN transaction.
- Wait and clear the IN transaction ACK interrupt.

19.6.4 Response STALL

When software wants to response a STALL handshake to Host, then Host will send a Clear_Feature command to endpoint. When software received this command, it must clear the STALL bit and do a toggle reset operation to reset the data toggle sequence of this endpoint to DATA0.

The toggle reset command is defined at UBD28.

19.6.5 Programmable Endpoint OUT Transfer Finish Check

For programmable endpoints OUT transfer that using descriptor list mode. Software must check both the descriptor read pointer and the ”valid” bit in the descriptor bit[43] to determine a finished transfer. Especially for the ”valid” bit in the descriptor list. Sometimes there is a timing racing problem between software find a change of the read pointer and the valid bit updated. This is caused by DRAM controller arbitration priority mechanism. So somtimes CPU read an empty descriptor before the correct descriptor write into DRAM. This case happens rarely, but it is possible to happen. So software must polling the valid bit several times after it received an interrupt and detected the change of descriptor read pointer.

19.6.6 Prevent a Transient Read Pointer Value

There is a possibility to read a temp value for the descriptor read pointer. This is caused by the USB registers read and the pointer update happened on 2 clock domains. So it is possible to read a wrong transient value when the pointer is updating. This fail case often will be encountered when multiple stages descriptor enabled concurrently. To prevent the fail case, SW must continuously read the descriptor read pointer until 2 consecutive of same value got.

19.6.7 Procedure to enable Interrupt

The USB2.0 interrupt function must follow the sequence to enable.

1. Enable USB2.0 clock at SCU080[7]=0 and wait 10 ms for clock stable.
2. Disable USB2.0 global reset at SCU040[3]=0.
3. Set interrupt type as sensitive-high level trigger at VIC24[5]=1 and VIC2C[5]=1.
4. Enable USB2.0 interrupt at VIC10[9]=1.

19.6.8 OUT Direction Endpoint Maximum Packet Size Setting

- Control endpoint, the default maximum packet size setting is 64 bytes. SW must allocate 64 bytes buffer for DMA buffer, no matter what speed it belongs to.
- Programmable endpoint, the maximum packet size setting was defined at EPP00[25:16]. SW must allocate with the maximum size memory for each DMA buffer.

20 USB1.1 HID Controller (USB11)

20.1 Overview

USB1.1 Controller (USB1.1), compliant with Universal Bus Specification Revision 1.1/2.0, supports only Low Speed USB transactions.

This controller integrates one set of Control Endpoint and two sets of Interrupt IN Endpoints, equipped with 8 bytes of data buffer for each Endpoint. It also supports additional features like Suspend, Wake-Up Resume and Remote Wake-Up Resume.

USB1.1 totally implements 17 sets of 32-bit registers, which are listed below, to program the various supported functions. Each register has its own specific offset value, ranging from 0x00 to 0x40h, to derive its physical address location.

Base address of USB1.1 = 0x1E6E_1000
Physical address = (Base address of USB1.1) + Offset

USBL00: Function Control and Status Register
USBL04: Function Configuration Setting Register
USBL08: Endpoint Toggle Bit Reset Register
USBL0C: USB Status Register
USBL10: Interrupt Control Register
USBL14: Interrupt Status Register
USBL18: Endpoint 0 Control and Status Register
USBL1C: Endpoint 1 Control and Status Register
USBL20: Endpoint 2 Control and Status Register
USBL24: Endpoint 0 SETUP/OUT Data Buffer Register (Low)
USBL28: Endpoint 0 SETUP/OUT Data Buffer Register (High)
USBL2C: Endpoint 0 IN Data Buffer Register (Low)
USBL30: Endpoint 0 IN Data Buffer Register (High)
USBL34: Endpoint 1 IN Data Buffer Register (Low)
USBL38: Endpoint 1 IN Data Buffer Register (High)
USBL3C: Endpoint 2 IN Data Buffer Register (Low)
USBL40: Endpoint 2 IN Data Buffer Register (High)

20.2 Features

- Compliant with Universal Serial Bus Specification Revision 1.1/2.0
- Integrate 1 set of USB1.1 PHY
- Clock source is from a divided clock from USB PHY PLL
- Support only Low Speed Transfer
- Support Suspend, Wake-up resume and Remote Wake-up resume
- Support 1 Control and 2 Interrupt IN Endpoint
- Support 8 Bytes buffer for each Endpoints

20.3 Procedure to enable USB1.1 Device port

1. Set SCU440[29:28] = ”00”, select USB2.0 port 2 mode as HID device
2. Set SCU040[3] = 1, enable controller reset
3. Set SCU080[7] = 0, enable PHY clock

4. wait 10 ms for PLL locking
5. Set SCU040[3] = 0, disable controller reset
6. Load driver

20.4 Registers : Base Address = 0x1E6E:1000

Offset: 00h	USBL00: Function Control and Status Register	Init = 0
Bit	**R/W**	**Description**
31:9	RO	**Reserved (0)**
8:6	RW	**Test Mode selection** 000 : Disable 001 : Test J 010 : Test K 011 : Test SE0_NAK 100 : Test Packet 101 : Test Force SE0 110 : Test Force SE1 111 : PHY BIST enable ~~Enable Test Loop Back~~ (for debugging purpose only)
5:4	RW	**USB remote wakeup signaling width selection** 00 : 4ms 01 : 8ms 10 : 12ms 11 : 15ms
3	RW	**Enable USB remote wakeup** 0: Disable USB remote wakeup 1: Enable USB remote wakeup USB remote wakeup can be enabled only when USB host enters suspend state. This bit will be automatically cleared by H/W whenever wakeup signal issued.
2	RW	**Stop clock when USB enters suspend sate** 0: Never stop clock even when USB1.1 enters suspend state 1: Stop clock when USB1.1 enters suspend state Stopping clock feature is designed to reduce the power consumption from USB1.1 Controller.
1	RW	**USB Low Speed Mode Enable** 0 : Full Speed 1 : Low Speed AST2600 not support Full speed mode, so this bit can only be set to '1'.
0	RW	**Enable USB connection** 0: Disable USB1.1 connection, driving to SE0 (default) 1: Enable USB1.1 connection

Offset: 04h	USBL04: Function Configuration Setting Register	Init = 0
Bit	**R/W**	**Description**
31:8	RO	**Reserved (0)**

to next page

from previous page

7:1	RW	**USB Function Device Address** Change the address will affect the packet receiving immediately. The address should be set after the status phase of the Set_Address control transfer command. The following is the Set_Address command sequence: 1. SETUP data packet, contains the new address information. 2. IN data packet, status phase with zero byte data returned. 3. Change to the new address.
0	RW	**Function configuration status** 0: Function configuration has not yet been done 1: Function configuration has been done Software needs to set this bit to ”1” when USB has been configured by receiving Set_Config_Feature.

Offset: 08h **USBL08: Endpoint Toggle Bit Reset Register** **Init = X**

Bit	R/W	Description
31:2	RO	**Reserved (0)**
1	W1T	**Reset Endpoint 2 data toggle bit** Writing ”1” to this bit will reset Endpoint 2 data toggle bit. Reading this bit will always return ”0”.
0	W1T	**Reset Endpoint 1 data toggle bit** Writing ”1” to this bit will reset Endpoint 1 data toggle bit. Reading this bit will always return ”0”.

Offset: 0Ch **USBL0C: USB Status Register (for debugging purpose only)** **Init = X**

Bit	R/W	Description
31	RO	**USB Suspend State**
30	RO	**USB Bus Reset State**
29	RO	**USB Bus Line State FullSpeedMode = DN, LowSpeedMode = DP**
28	RO	**USB Bus Line State FullSpeedMode = DP, LowSpeedMode = DN**
27	RO	**Reserved (0)**
26:16	RO	**USB Last Frame Number**
15:11	RO	**Reserved (0)**
10:4	RO	**USB Last Transaction Device Address**
3:0	RO	**USB Last Transaction Endpoint Number**

Offset: 10h **USBL10: Interrupt Control Register** **Init = 0**

Bit	R/W	Description
31:9	RO	**Reserved (0)**
8	RW	**Enable EP0 In/Out NAK Response Interrupt** S/W opens this interrupt enable after receiving Setup Data, and when receiving this interrupt, in ISR, S/W first prepare the required IN/OUT request and then clear this interrupt status. This bit can be disabled after status phase finished, or S/W can always sets this bit on. For whether In or Out NAK response, S/W can check the USBL14[9].
7	RW	**Enable EP2 In Data packet ACK Interrupt**
6	RW	**Enable EP1 In Data packet ACK Interrupt**

to next page

from previous page

5	RW	**Enable EP0 In Data packet ACK Interrupt**
4	RW	**Enable EP0 Out Data packet ACK Interrupt**
3	RW	**Enable EP0 Setup Data Arrives Interrupt**
2	RW	**Enable USB Suspend Resume Interrupt**
1	RW	**Enable USB Suspend Entry Interrupt**
0	RW	**Enable USB Bus Reset Interrupt**

Offset: 14h **USBL14: Interrupt Status Register** **Init = 0**

Bit	R/W	Description
31:10	RO	**Reserved (0)**
9	RO	**EP0 In/Out Transaction Cycle Status** 0 : EP0 is under IN cycle 1 : EP0 is under OUT cycle
8	RW1C	**EP0 In/Out NAK Response Occurs (WC)**
7	RW1C	**EP2 In Data packet ACK returned (WC)**
6	RW1C	**EP1 In Data packet ACK returned (WC)**
5	RW1C	**EP0 In Data packet ACK/STALL returned (WC)**
4	RW1C	**EP0 Out Data packet ACK returned (WC)**
3	RW1C	**EP0 Setup Data Arrives (WC)**
2	RW1C	**USB Suspend Resume Occurs (WC)**
1	RW1C	**USB Suspend Entry Occurs (WC)**
0	RW1C	**USB Bus Reset Occurs (WC)**

Note :
WC : means this status is write '1' clear.
Each status bit of this register is set by H/W automatically when the corresponding event occurred or the transaction finished. The corresponding interrupt enable bit wont impact the setting of the bit. The enable bit only determines the interrupt generation.
Whenever S/W has finished the handling of the status bit, it must clear the status bit by writing '1', else it will not be able to recognize new events.

Offset: 18h **USBL18: Endpoint 0 Control and Status Register** **Init = 0x00000XX0**

Bit	R/W	Description
31:12	RO	**Reserved (0)**
11:8	RO	**Endpoint 0 OUT received data byte count** This register determines the number of valid bytes in Endpoint 0 SETUP/OUT data buffer.
7:4	RW	**Endpoint 0 IN data byte count for transfer** This register determines the number of valid bytes in Endpoint 0 IN data buffer.
3	RO	**Reserved (0)**
2	RW	**Endpoint 0 OUT buffer ready for receiving data** 0: Buffer is not ready to receive data 1: Buffer is ready to receive data S/W can set this bit to '1' when it is ready to receive data from Host by OUT transactions. When H/W receives the data successfully, this bit will be automatically cleared to "0". S/W can monitor this bit to check the data has been received or not.

to next page

from previous page

1	RW	**Endpoint 0 IN buffer ready for transferring data** 0: Data is not ready to transfer data 1: Data is ready to transfer data S/W can set this bit to '1' when there is a need to transfer data to Host by IN transactions. When H/W has finished the data transfer, this bit will be automatically cleared. S/W can monitor this bit to check the data has been transferred or not. Only one of Bit [1] and Bit [2] can be set to "1" for each time. The two settings are exclusive. For H/W timing concerns, this bit cannot be set at the same time as Bit [7:4] of this register. Two steps of write commands are required: Write Bit [7:4] the first, update Bit [1] the second.
0	RW	**Stall Control** When this register is set to "1", Endpoint 0 returns STALL response for Data and Status stages. This bit is automatically cleared to "0" after a new SETUP packet is received.

Offset: 1Ch **USBL1C: Endpoint 1 Control and Status Register** **Init = 0x000000X0**

Bit	R/W	Description
31:8	RO	**Reserved (0)**
7:4	RW	**Endpoint 1 IN data byte count for transfer** This register determines the number of valid bytes in Endpoint 1 IN data buffer.
3:2	RO	**Reserved (0)**
1	RW	**Endpoint 1 IN buffer ready for transferring data** 0: Data is not ready to transfer data 1: Data is ready to transfer data S/W can set this bit to '1' when there is a need to transfer data to Host by IN transactions. When H/W has finished the data transfer, this bit will be automatically cleared. S/W can monitor this bit to check the data has been transferred or not. For H/W timing concerns, this bit cannot be set at the same time as Bit [7:4] of this register. Two steps of write commands are required: Write Bit [7:4] the first, update Bit [1] the second.
0	RW	**Stall Control** When this register is set to "1", Endpoint 1 returns STALL response for all IN transactions until S/W clears this bit to '0'.

Offset: 20h **USBL20: Endpoint 2 Control and Status Register** **Init = 0x000000X0**

Bit	R/W	Description
31:8	RO	**Reserved (0)**
7:4	RW	**Endpoint 2 IN data byte count for transfer** This register determines the number of valid bytes in Endpoint 2 IN data buffer.
3:2	RO	**Reserved (0)**
1	RW	**Endpoint 2 IN buffer ready for transferring data** 0: Data is not ready to transfer data 1: Data is ready to transfer data S/W can set this bit to '1' when there is a need to transfer data to Host by IN transactions. When H/W has finished the data transfer, this bit will be automatically cleared. S/W can monitor this bit to check the data has been transferred or not. For H/W timing concerns, this bit cannot be set at the same time as Bit [7:4] of this register. Two steps of write commands are required: Write Bit [7:4] the first, update Bit [1] the second.

to next page

from previous page

0	RW	**Stall Control** When this register is set to ”1”, Endpoint 2 returns STALL response for all IN transactions until S/W clears this bit to ’0’.

Offset: 24h **USBL24: Endpoint 0 SETUP/OUT Data Buffer Register (Low)** **Init = X**
Offset: 28h **USBL28: Endpoint 0 SETUP/OUT Data Buffer Register (High)** **Init = X**

Bit	Attr.	Description
63:0	RO	**Endpoint 0 Setup/OUT Data buffer** When received SETUP/OUT interrupt, S/W must read back this register as input commands or data. Since SETUP data and OUT data share the same buffer and SETUP data cannot be retried, whenever SETUP transaction phase happens, this data buffer will be automatically overwritten by SETUP data.

Offset: 2Ch **USBL2C: Endpoint 0 IN Data Buffer Register (Low)** **Init = X**
Offset: 30h **USBL30: Endpoint 0 IN Data Buffer Register (High)** **Init = X**

Bit	Attr.	Description
63:0	RW	**Endpoint 0 IN Data buffer** S/W must initialize this data buffer firstly, and then enable this buffer for IN data response.

Offset: 34h **USBL34: Endpoint 1 IN Data Buffer Register (Low)** **Init = X**
Offset: 38h **USBL38: Endpoint 1 IN Data Buffer Register (High)** **Init = X**

Bit	Attr.	Description
63:0	RW	**Endpoint 1 IN Data buffer** S/W must initialize this data buffer firstly, and then enable this buffer for IN data response.

Offset: 3Ch **USBL3C: Endpoint 2 IN Data Buffer Register (Low)** **Init = X**
Offset: 40h **USBL40: Endpoint 2 IN Data Buffer Register (High)** **Init = X**

Bit	Attr.	Description
63:0	RW	**Endpoint 2 IN Data buffer** S/W must initialize this data buffer firstly, and then enable this buffer for IN data response.

20.5 Software Programming Guide

1. The Base address please referenced the system memory allocation definition.
2. All registers programming are Double Word(32 bits) base, no byte enable control.
3. USB Controller Initialization
   (a) W USBL00 = 0
   (b) W USBL04 = 0
   (c) W USBL10 = value, this is depends on the SW programming requirement.
   (d) W USBL14 = ffffffff
   (e) W USBL18 = 0
   (f) W USBL1C = 0
   (g) W USBL20 = 0
   (h) W USBL00 = 0x03 for Low Speed Connection
   bit[2] is depend on the power saving consideration.
   (i) And SW can starts to wait Host's configuration handshake
4. Endpoint 0 Handshake Control
   (a) When SETUP transaction received, SW can be ACKed by interrupt or check the status bit USBL14[3]
   (b) After SW processed the SETUP data, SW needs to decide the next transaction cycle is IN or OUT, and sets the specific bits in USBL18 to enable the next command cycle, the host IN/OUT command for this Endpoint will be NAKed until it is enabled, but SETUP command can be ACKed.
   (c) Or SW can enable the NAK interrupt enable at USBL10.bit8, and wait for NAK interrupt occurs to decide what to do next.
   (d) Continue step2/3 until Status Phase is finished.
5. Endpoint 1/2 Handshake Control
   (a) When SW have data to be returned from this Endpoint, do the following steps else HW always returns nothing(NAK) for this Endpoints command.
   (b) Fill the data buffer
   (c) Write Byte Count
   (d) Enable Data Buffer Ready for transfer
   (e) Polling Buffer Empty Status at the same bit as Buffer Ready to check the data is returned successfully by HW or not, or by interrupt to be ACKed when HW transfer done.
6. Remote Wakeup
   (a) When Host is in Suspend state
   (b) Sets the Remote Wakeup Signaling Width
   (c) Enable Remote Wakeup
7. Other Register : USBL04,USBL08
   (a) These registers setting are depend on the Host command, and SW decide when to set these registers.

21 USB2.0/1.1 Host Controller (USBHC)

21.1 Overview

USB1.1 Host Controller (UHCI) is adapted from UHCI Rev1.1; and only USB Host Controller I/O registers are implemented. For the PCI Configuration Registers, they are enumerated by software system. A RootHub is embedded in the core, by default, 2 USB1.1 downstream ports are implemented.

USB2.0 Host Controller (EHCI) is adapted from EHCI Rev1.0. A RootHub is embedded in the core, by default, only one USB2.0 downstream port is implemented.

Base address of UHCI Host Controller = 0x1E6B_0000
Physical address = (Base address of UHCI) + Offset

UHCI Register Set

UHCI00: USB Command Register (USBCMD)
UHCI04: USB Status Register (USBSTS)
UHCI08: USB Interrupt Enable Register (USBINT)
UHCI0C: Frame List Based Address Register (FRBASEADD)
UHCI40: Test Control Register
UHCI80: Frame Number Register (FRNUM)
UHCI84: Start of Frame Modify Register (SOFMOD)
UHCI88: Port1 Status/Control Register (PORTSC1)
UHCI8C: Port2 Status/Control Register (PORTSC2)

Base address of EHCI Host Controller #1 = 0x1E6A_1000 (Physical PORTA)
Base address of EHCI Host Controller #2 = 0x1E6A_3000 (Physical PORTB)

EHCI Register Set

EHCI00: Capability Registers Length (CAPLENGTH)
EHCI04: Structural Parameters (HCSPARAMS)
EHCI08: Capability Parameters (HCCPARAMS)
EHCI0C: Companion Port Route Description (HCSP-PORTROUTE)
EHCI20: USB Command Register (USBCMD)
EHCI24: USB Status Register (USBSTS)
EHCI28: USB Interrupt Enable Register (USBINTR)
EHCI2C: Frame Index Register (FRINDEX)
EHCI34: Periodic Frame List Base Address Register (PERIODICLISTBASE)
EHCI38: Current Asynchronous List Address Register (ASYNCLISTADDR)
EHCI60: Configure Flag Register (CONFIGFLAG)
EHCI64: Port1 Status/Control Register (PORTSC1)
EHCI80: Frame Length Adjustment Register (FLADJ)
EHCI84: Controler Fine-tune Register
EHCI88: Frame Timing Adjustment
EHCI8C: Hardware Revision Number Register

21.2 UHCI Features

- Complies with USB Specification Rev 2.0

- Adapted from Universal Host Controller Interface (UHCI) Specification Rev1.1
- Supports 2 ports data transfer at full-speed (12 Mbit/s) and low-speed (1.5 Mbit/s) (UHCI)
- Supports all four types of USB transfers: control, bulk, interrupt and isochronous
- Includes a RootHub with multi-port architecture
- Directly addressable memory architecture; memory can be updated on-the-fly
- Supports USB Host Controller I/O registers for software communication channel
- Re-uses Linux UHCI Host Controller driver with minor change on register offset
- Register set are only capable of double-word read/write

21.3 EHCI Features

- Complies with USB Specification Rev 2.0
- Adapted from Enhanced Host Controller Interface (EHCI) Specification Rev1.0
- Supports 1 port data transfer at high-speed (480 Mbit/s)(EHCI)
- Supports all four types of USB transfers: control, bulk, interrupt and isochronous
- Includes a RootHub with multi-port architecture
- Directly addressable memory architecture; memory can be updated on-the-fly
- Supports USB Host Controller I/O registers for software communication channel
- Re-uses Linux EHCI Host Controller driver with minor change on register offset
- Register set are only capable of double-word read/write
- Supported capabilities:
  - 1 down-stream port
  - no port power control (PPC = 0)
  - 1 companion controller (N_CC = 1)
  - no port indicator
  - no debug port
  - no 64-bit addressing
  - programmable frame list size: 1024, 512, 256
  - programmable asynchronous schedule park (EHCI84[1])
  - no EHCI extended capabilities

21.4 Procedure to enable USB Host port

21.4.1 USB2.0 Host port 1

1. Set SCU440 bit[25:24] = ”10”, select USB2.0 port 1 mode as host
2. Set SCU040 bit[14] = 1, enable controller reset
3. Set SCU080 bit[14] = 1, enable PHY clock
4. wait 10 ms for PLL locking
5. Set SCU044 bit[14] = 1, disable controller reset

21.4.2 USB2.0 Host port 2

1. Set SCU440 bit[29:28] = ”10”, select USB2.0 port 2 mode as host
2. Set SCU040 bit[3] = 1, enable controller reset
3. Set SCU084 bit[7] = 1, enable PHY clock
4. wait 10 ms for PLL locking
5. Set SCU044 bit[3] = 1, disable controller reset
6. Load EHCI driver

21.4.3 UHCI controller

1. USB2.0 Port 2 must be enabled first
2. Set SCU040 bit[15] = 1, enable controller reset
3. Set SCU084 bit[9] = 1, enable controller clock
4. wait 10 us for clock stable
5. Set SCU044 bit[15] = 1, disable controller reset
6. Load UHCI driver

21.5 UHCI Registers : Base Address = 0x1E6B:0000

Offset: 00h	UHCI00: USB Command Register (USBCMD)	Init = 0

Bit	R/W	Description
31:8	RO	**Reserved (0)**
7	RW	**Max Packet (MAXP)** 0: 32 bytes. 1: 64 bytes. This bit selects the maximum packet size that can be used for full speed bandwidth reclamation at the end of a frame. This value is used by the Host Controller to determine whether it should initiate another transaction based on the time remaining in the SOF counter. Use of reclamation packets larger than the programmed size will cause a Babble error if executed during the critical window at frame end. The Babble error results in the offending endpoint being stalled. Software is responsible for ensuring that any packet which could be executed under bandwidth reclamation be within this size limit.
6	RW	**Configure Flag (CF)** HCD software sets this bit as the last action in its process of configuring the Host Controller. This bit has no effect on the hardware. It is provided only as a semaphore service for software.
5	RW	**Software Debug (SWDBG)** 0: Normal Mode. 1: Debug mode. In SW Debug mode, the Host Controller clears the Run/Stop bit after the completion of each USB transaction. The next transaction is executed when software sets the Run/Stop bit back to 1. The SWDBG bit must only be manipulated when the controller is in the stopped state. This can be determined by checking the HCHalted bit in the USBSTS register.
4	RW	**Force Global Resume (FGR)** When set to 1, Host Controller sends the Global Resume signal on the USB. Software sets this bit to 0 after 20 ms has elapsed to stop sending the Global Resume signal. At that time all USB devices should be ready for bus activity. The Host Controller sets this bit to 1 when a resume event (connect, disconnect, or K-state) is detected while in global suspend mode. Software resets this bit to 0 to end Global Resume signaling. The 1 to 0 transition causes the port to send a low speed EOP signal. This bit will remain a 1 until the EOP has completed.
3	RW	**Enter Global Suspend Mode (EGSM)** When set to 1, Host Controller enters the Global Suspend mode. No USB transactions occurs during this time. The Host Controller is able to receive resume signals from USB and interrupt the system. Software resets this bit to 0 to come out of Global Suspend mode. Software writes this bit to 0 at the same time that Force Global Resume (bit 4) is written to 0 or after writing bit 4 to 0. Software must also ensure that the Run/Stop bit (bit 0) is cleared prior to setting this bit.

to next page

from previous page

2	RW	**Global Reset (GRESET)** When this bit is set to 1, the Host Controller sends the global reset signal on the USB and then resets all its logic, including the internal hub registers. The hub registers are reset to their power on state. This bit is reset by the software after a minimum of 10 ms has elapsed as specified in Chapter 7 of the USB Specification. Note: Chip Hardware Reset has the same effect as Global Reset (bit 2), except that the Host Controller does not send the Global Reset on USB.
1	RW	**Host Controller Reset (HCRESET)** When this bit is set to 1, the Host Controller module resets its internal timers, counters, state machines, etc. to their initial value. Any transaction currently in progress on USB is immediately terminated. This bit is reset by the Host Controller when the reset process is complete. The HCReset effects on Hub registers are slightly different from Chip Hardware Reset and Global USB Reset. The HCReset affects bits [8,3:0] of the Port Status and Control Register (PORTSC) of each port. HCReset resets the state machines of the Host Controller including the Connect/Disconnect state machine (one for each port). When the Connect/Disconnect state machine is reset, the output that signals connect/disconnect are negated to 0, effectively signaling a disconnect even if a device is attached to the port. This virtual disconnect causes the port to be disabled. This disconnect and disabling of the port causes bit 1 (connect status change) and bit 3 (port enable/disable change) of the PORTSC to get set. The disconnect also causes bit 8 of PORTSC to reset. About 64 bit times after HCReset goes to 0, the connect and low-speed detect will take place and bits 0 and 8 of the PORTSC will change accordingly.
0	RW	**Run/Stop (RS)** 0: Stop. 1: Run. When set to a 1, the Host Controller proceeds with execution of the schedule. The Host Controller continues execution as long as this bit is set. When this bit is set to 0, the Host Controller completes the current transaction on the USB and then halts. The HC Halted bit in the status register indicates when the Host Controller has finished the transaction and has entered the stopped state. The Host Controller clears this bit when the following fatal errors occur: consistency check failure, AHB Bus errors.

Offset: 04h **UHCI04: USB Status Register (USBSTS)** **Init = 0x0000_0020**

Bit	R/W	Description
31:6	RO	**Reserved (0)**
5	RW1C	**HCHalted bit (WC)** The Host Controller sets this bit to 1 after it has stopped executing as a result of the Run/Stop bit being set to 0, either by software or by the Host Controller hardware (debug mode or an internal error).
4	RW1C	**Host Controller Process Error (WC)** The Host Controller sets this bit to 1 when it detects a fatal error and indicates that the Host Controller suffered a consistency check failure while processing a Transfer Descriptor. An example of a consistency check failure would be finding an illegal PID field while processing the packet header portion of the TD. When this error occurs, the Host Controller clears the Run/Stop bit in the Command register to prevent further schedule execution. A hardware interrupt is generated to the system.

to next page

from previous page

3	RW1C	**Host System Error (WC)** The Host Controller sets this bit to 1 when a serious error occurs during a host system access involving the Host Controller module. In an AHB system, conditions that set this bit to 1 include ERROR response and 4G boundary detection. When this error occurs, the Host Controller clears the Run/Stop bit in the Command register to prevent further execution of the scheduled TDs. A hardware interrupt is generated to the system.
2	RW1C	**Resume Detect (WC)** The Host Controller sets this bit to 1 when it receives a RESUME signal from a USB device. This is only valid if the Host Controller is in a global suspend state (bit 3 of Command register = 1).
1	RW1C	**USB Error Interrupt (WC)** The Host Controller sets this bit to 1 when completion of a USB transaction results in an error condition (e.g., error counter underflow). If the TD on which the error interrupt occurred also had its IOC bit set, both this bit and Bit 0 are set.
0	RW1C	**USB Interrupt (USBINT) (WC)** The Host Controller sets this bit to 1 when the cause of an interrupt is a completion of a USB transaction whose Transfer Descriptor had its IOC bit set. The Host Controller also sets this bit to 1 when a short packet is detected (actual length field in TD is less than maximum length field in TD), and short packet detection is enabled in that TD.

Note :
WC : means this status is write '1' clear, write '0' has no any effect.
Each status bit of this register is set by H/W automatically when the corresponding event occurred or the transaction finished. The corresponding interrupt enable bit wont impact the setting of the bit. The enable bit only determines the interrupt generation.
Whenever S/W has finished the handling of the status bit, it must clear the status bit by writing '1', else it will not be able to recognize new events.

Offset: 08h **UHCI08: USB Interrupt Enable Register (USBINT)** **Init = 0**

Bit	R/W	Description
31:4	RO	**Reserved (0)**
3	RW	**Short Packet Interrupt Enable** 0: Disabled. 1: Enabled.
2	RW	**Interrupt On Complete (IOC) Enable** 0: Disabled. 1: Enabled.
1	RW	**Resume Interrupt Enable** 0: Disabled. 1: Enabled.
0	RW	**Timeout/CRC Interrupt Enable** 0: Disabled. 1: Enabled.

Offset: 0Ch **UHCI0C: Frame List Based Address Register (FRBASEADD)** **Init = X**

Bit	R/W	Description
31:12	RW	**Base Address** These bits correspond to memory address signals [31:12], respectively.
11:0	RO	**Reserved (0)**

Offset: 40h	UHCI40: Test Control Register	Init = 0
Bit	**R/W**	**Description**
31:15	RO	**Reserved (0)**
14	RW	**AHB Bus control selection**
13	RW	**Enable FIFO Auto Power Down** 0: Disable 1: Enable
12	RW	**Delay List operation 8 clock cycles** 0: No delay 1: Delay 8 cycles This delay is mainly for TXFIFO power down wakeup time.
11:10	RO	**Reserved (0)**
9	RO	**Loop Back test result** 0: Fail 1: Pass This status will be cleared when disabling USB Test Mode
8	RO	**Loop Back test status** 0: Not finish 1: Finished This status will be cleared when disabling USB Test Mode
7	RW	**Transmit FIFO threshold** 0: 32 bytes 1: 64 bytes
6	RW	**Port LineState sampling mode** 0: Normal, sample at EOF2 1: Real time
5:4	RW	**SOF Test mode** 00: 1 ms, Normal mode 01: 64 us 10: 128 us 11: 256 us
3	RW	**Test Mode speed selection** 0: Full speed 1: Low speed
2:0	RW	**USB Test Mode selection** 000: Disable 001: Enable Test J 010: Enable Test K 011: Reserved 100: Enable Test Packet 101: Enable Test SE0 110: Enable Test SE1 111: Enable Test Loop Back (for debugging purpose only)

Offset: 80h	UHCI80: Frame Number Register (FRNUM)	Init = 0
Bit	**R/W**	**Description**
31:11	RO	**Reserved (0)**
10:0	RW	**Frame List Current Index/Frame Number** Bits [10:0] provide the frame number in the SOF Frame. The value in this register increments at the end of each time frame (approximately every 1 ms). In addition, bits [9:0] are used for the Frame List current index and correspond to memory address signals [11:2].

Offset: 84h	UHCI84: Start of Frame Modify Register (SOFMOD)	Init = 0x0000_0040

Bit	R/W	Description
31:7	RO	**Reserved (0)**
6:0	RW	**SOF Timing Value** Guidelines for the modification of frame time are contained in Chapter 7 of the USB Specification. The SOF cycle time (number of SOF counter clock periods to generate a SOF frame length) is equal to 11936 + value in this field. The default value is decimal 64 which gives a SOF cycle time of 12000. For a 12 MHz SOF counter clock input, this produces a 1 ms Frame period. The following table indicates what SOF Timing Value to program into this field for a certain frame period. Frame Length (# 12Mhz Clocks) (decimal) / SOF Reg. Value (decimal) 11936 — 0 11937 — 1 ... ... 11999 — 63 12000 — 64 12001 — 65 ... ... 12062 — 126 12063 — 127

Offset: 88h	UHCI88: Port1 Status/Control Register (PORTSC1)	Init = 0x0000_0080
Offset: 8Ch	UHCI8C: Port2 Status/Control Register (PORTSC2)	Init = 0x0000_0080

Bit	Attr.	Description
31:13	RO	**Reserved (0)**
12	RW	**Suspend** 0: Port not in suspend state. 1: Port in suspend state. This bit should not be written to a 1 if global suspend is active (bit[3]=1 in the USBCMD register). Bit[2] and bit[12] of this register define the hub states as follows: Bits [12,2] — Hub State x0 — Disable 01 — Enable 11 — Suspend When in suspend state, downstream propagation of data is blocked on this port, except for single ended 0 resets (global reset and port reset). The blocking occurs at the end of the current transaction, if a transaction was in progress when this bit was written to 1. In the suspend state, the port is sensitive to resume detection. Note that the bit status does not change until the port is suspended and that there may be a delay in suspending a port if there is a transaction currently in progress on the USB.
11:10	RO	**Reserved (0)**

to next page

from previous page

9	RW	**Port Reset** 0: Port is not in Reset. 1: Port is in Reset. When in the Reset State, the port is disabled and sends the USB Reset signaling. Note that host software must guarantee that the RESET signaling is active for the proper amount of time as specified in the USB Specification.
8	RO	**Low Speed Device Attached** 0: Full speed device. 1: Low speed device is attached to this port. Writes have no effect.
7	RO	**Reserved (1)**
6	RW	**Resume Detect** 0: No resume (K-state) detected/driven on port. 1: Resume detected/driven on port. Software sets this bit to a 1 to drive resume signaling. The Host Controller sets this bit to a 1 if a J-to-K transition is detected while the port is in the Suspend state. Note that when this bit is 1, a K-state is driven on the port as long as this bit remains 1 and the port is still in suspend state. Writing a 0 (from 1) causes the port to send a low speed EOP. This bit will remain a 1 until the EOP has completed.
5:4	RO	**Line Status** These bits reflect the D+ (bit[4]) and D- (bit[5]) signals line's logical levels. These bits are used for fault detect and recovery as well as for USB diagnostics. This field is updated at EOF2 time (See Chapter 11 of the USB Specification).
3	RW1C	**Port Enable/Disable Change (WC)** 0: No change. 1: Port enabled/disabled status has changed. For the root hub, this bit gets set only when a port is disabled due to disconnect on the that port or due to the appropriate conditions existing at the EOF2 point (See Chapter 11 of the USB Specification). Software clears this bit by writing a 1 to it.
2	RW	**Port Enabled/Disabled** 0: Disable. 1: Enable. Ports can be enabled by host software only. Ports can be disabled by either a fault condition (disconnect event or other fault condition) or by host software. Note that the bit status does not change until the port state actually changes and that there may be a delay in disabling or enabling a port if there is a transaction currently in progress on the USB.
1	RW1C	**Connect Status Change (WC)** 0: No change. 1: Change in Current Connect Status. Indicates a change has occurred in the port's Current Connect Status (see bit 0). The hub device sets this bit for any changes to the port device connect status, even if system software has not cleared a connect status change. If, for example, the insertion status changes twice before system software has cleared the changed condition, hub hardware will be setting an already-set bit (i.e., the bit will remain set). However, the hub transfers the change bit only once when the Host Controller requests a data transfer to the Status Change endpoint. System software is responsible for determining state change history in such a case. Software sets this bit to 0 by writing a 1 to it.

to next page

from previous page

0	RO	**Current Connect Status** 0: No device is present. 1: Device is present on port. This value reflects the current state of the port, and may not correspond directly to the event that caused the Connect Status Change bit (Bit[1]) to be set.

21.6 EHCI Registers

Offset: 00h **EHCI00: Capability Registers Length (CAPLENGTH)** **Init = 0x01000020**

Bit	R/W	Description
31:16	RO	**Host Controller Interface Version Number (HCIVERSION)**
15:8	RO	**Reserved (0)**
7:0	RO	**Offset to Operational Registers** This register is used as an offset to add to register base to find the beginning of the Operational Register Space.

Offset: 04h **EHCI04: Structural Parameters (HCSPARAMS)** **Init = 0x00001101**

Bit	R/W	Description
31:16	RO	**Reserved (0)**
23:20	RO	**Debug Port Number**
19:17	RO	**Reserved (0)**
16	RO	**Port Indicators (P_INDICATOR)** This bit indicates whether the ports support port indicator control. When this bit is a one, the port status and control registers include a read/writeable field for controlling the state of the port indicator.
15:12	RO	**Number of Companion Controller (N_CC)** This field indicates the number of companion controllers associated with this USB 2.0 host controller. A zero in this field indicates there are no companion host controllers. Port-ownership hand-off is not supported. Only high-speed devices are supported on the host controller root ports. A value larger than zero in this field indicates there are companion USB 1.1 host controller(s). Port-ownership hand-offs are supported. High, Full- and Low-speed devices are supported on the host controller root ports.
11:8	RO	**Number of Ports per Companion Controller (N_PCC)** This field indicates the number of ports supported per companion host controller. It is used to indicate the port routing configuration to system software.
7	RO	**Port Routing Rules** This field indicates the method used by this implementation for how all ports are mapped to companion controllers. The value of this field has the following interpretation: **0:** The first N_PCC ports are routed to the lowest numbered function companion host controller, the next N_PCC port are routed to the next lowest function companion controller, and so on. **1:** The port routing is explicitly enumerated by the first N_PORTS elements of the HCSP-PORTROUTE array.
6:5	RO	**Reserved (0)**

to next page

from previous page

4	RO	**Port Power Control (PPC)** This field indicates whether the host controller implementation includes port power control. A one in this bit indicates the ports have port power switches. A zero in this bit indicates the port do not have port power switches. The value of this field affects the functionality of the Port Power field in each port status and control register.
3:0	RO	**N_PORTS** This field specifies the number of physical downstream ports implemented on this host controller. The value of this field determines how many port registers are addressable in the Operational Register Space. Valid values are in the range of 1H to FH.

Offset: 08h EHCI08: Capability Parameters (HCCPARAMS) Init = 0x00000016

Bit	R/W	Description
31:16	RO	**Reserved (0)**
15:8	RO	**EHCI Extended Capabilities Pointer (EECP)** This optional field indicates the existence of a capabilities list. A value of 00h indicates no extended capabilities are implemented. A non-zero value in this register indicates the offset in PCI configuration space of the first EHCI extended capability. The pointer value must be 40h or greater if implemented to maintain the consistency of the PCI header defined for this class of device.
7:4	RO	**Isochronous Scheduling Threshold** This field indicates, relative to the current position of the executing host controller, where software can reliably update the isochronous schedule. When bit [7] is zero, the value of the least significant 3 bits indicates the number of micro-frames a host controller can hold a set of isochronous data structures (one or more) before flushing the state. When bit [7] is a one, then host software assumes the host controller may cache an isochronous data structure for an entire frame.
3	RO	**Reserved (0)**
2	RO	**Asynchronous Schedule Park Capability** If this bit is set to a one, then the host controller supports the park feature for high-speed queue heads in the Asynchronous Schedule. The feature can be disabled or enabled and set to a specific level by using the Asynchronous Schedule Park Mode Enable and Asynchronous Schedule Park Mode Count fields in the USBCMD register.
1	RO	**Programmable Frame List Flag** If this bit is set to a zero, then system software must use a frame list length of 1024 elements with this host controller. The USBCMD register Frame List Size field is a read-only register and should be set to zero. If set to a one, then system software can specify and use a smaller frame list and configure the host controller via the USBCMD register Frame List Size field. The frame list must always be aligned on a 4K page boundary. This requirement ensures that the frame list is always physically contiguous.
0	RO	**64-bit Addressing Capability** 0: data structures using 32-bit address memory pointers 1: data structures using 64-bit address memory pointers

Offset: 0Ch-1Ch EHCI0C: Companion Port Route Description (HCSP-PORTROUTE) Init = 0x0

Bit	R/W	Description
159:0	RO	**Reserved (0)**

Offset: 20h	EHCI20: USB Command Register (USBCMD)	Init = 0x00080B00
Bit	**R/W**	**Description**
31:24	RO	**Reserved (0)**
23:16	RW	**Interrupt Threshold Control** This field is used by system software to select the maximum rate at which the host controller will issue interrupts. The only valid values are defined below. If software writes an invalid value to this register, the results are undefined. Value : Maximum Interrupt Interval 00h : Reserved 01h : 1 micro-frame 02h : 2 micro-frames 04h : 4 micro-frames 08h : 8 micro-frames (default, equates to 1 ms) 10h : 16 micro-frames (2 ms) 20h : 32 micro-frames (4 ms) 40h : 64 micro-frames (8 ms) Any other value in this register yields undefined results. Software modifications to this bit while HCHalted bit is equal to zero results in undefined behavior.
15:12	RO	**Reserved (0)**
11	RW	**Asynchronous Schedule Park Mode Enable** If the Asynchronous Park Capability bit in the HCCPARAMS register is a one, then this bit defaults to a 1h and is R/W. Otherwise the bit must be a zero and is RO. Software uses this bit to enable or disable Park mode. When this bit is one, Park mode is enabled. When this bit is a zero, Park mode is disabled.
10	RO	**Reserved (0)**
9:8	RW	**Asynchronous Schedule Park Mode Count** If the Asynchronous Park Capability bit in the HCCPARAMS register is a one, then this field defaults to 3h and is R/W. Otherwise it defaults to zero and is RO. It contains a count of the number of successive transactions the host controller is allowed to execute from a high-speed queue head on the Asynchronous schedule before continuing traversal of the Asynchronous schedule. Valid values are 1h to 3h. Software must not write a zero to this bit when Park Mode Enable is a one as this will result in undefined behavior.
7	RO	**Reserved (0)**
6	RW	**Interrupt on Async Advance Doorbell** This bit is used as a doorbell by software to tell the host controller to issue an interrupt the next time it advances asynchronous schedule. Software must write a 1 to this bit to ring the doorbell. When the host controller has evicted all appropriate cached schedule state, it sets the Interrupt on Async Advance status bit in the USBSTS register. If the Interrupt on Async Advance Enable bit in the USBINTR register is a one then the host controller will assert an interrupt at the next interrupt threshold. The host controller sets this bit to a zero after it has set the Interrupt on Async Advance status bit in the USBSTS register to a one. Software should not write a one to this bit when the asynchronous schedule is disabled. Doing so will yield undefined results.

to next page

from previous page

5	RW	**Asynchronous Schedule Enable** This bit controls whether the host controller skips processing the Asynchronous Schedule. Values mean: 0: Do not process the Asynchronous Schedule 1: Use the ASYNCLISTADDR register to access the Asynchronous Schedule.
4	RW	**Periodic Schedule Enable** This bit controls whether the host controller skips processing the Periodic Schedule. Values mean: 0: Do not process the Periodic Schedule 1: Use the PERIODICLISTBASE register to access the Periodic Schedule.
3:2	RW	**List Size** This field specifies the size of the frame list. The size the frame list controls which bits in the Frame Index Register should be used for the Frame List Current index. Values mean: 00: 1024 elements (4096 bytes) Default value 01: 512 elements (2048 bytes) 10: 256 elements (1024 bytes) – for resource-constrained environments 11: Reserved
1	RW	**Host Controller Reset (HCRESET)** This control bit is used by software to reset the host controller. The effects of this on Root Hub registers are similar to a Chip Hardware Reset. When software writes a one to this bit, the Host Controller resets its internal pipelines, timers, counters, state machines, etc. to their initial value. Any transaction currently in progress on USB is immediately terminated. A USB reset is not driven on downstream ports. PCI Configuration registers are not affected by this reset. All operational registers, including port registers and port state machines are set to their initial values. Port ownership reverts to the companion host controller(s). Software must reinitialize the host controller in order to return the host controller to an operational state. This bit is set to zero by the Host Controller when the reset process is complete. Software cannot terminate the reset process early by writing a zero to this register. Software should not set this bit to a one when the HCHalted bit in the USBSTS register is a zero. Attempting to reset an actively running host controller will result in undefined behavior.
0	RW	**Run/Stop (RS)** 0=Stop. 1=Run. When set to a 1, the Host Controller proceeds with execution of the schedule. The Host Controller continues execution as long as this bit is set to a 1. When this bit is set to 0, the Host Controller completes the current and any actively pipelined transactions on the USB and then halts. The Host Controller must halt within 16 micro-frames after software clears the Run bit. The HC Halted bit in the status register indicates when the Host Controller has finished its pending pipelined transactions and has entered the stopped state. Software must not write a one to this field unless the host controller is in the Halted state (i.e. HCHalted in the USBSTS register is a one). Doing so will yield undefined results.

Offset: 24h	**EHCI24: USB Status Register (USBSTS)**	**Init = 0x00001000**
Bit	**R/W**	**Description**
31:16	RO	**Reserved (0)**

to next page

from previous page

15	RO	**Asynchronous Schedule Status** The bit reports the current real status of the Asynchronous Schedule. If this bit is a zero then the status of the Asynchronous Schedule is disabled. If this bit is a one then the status of the Asynchronous Schedule is enabled. The Host Controller is not required to immediately disable or enable the Asynchronous Schedule when software transitions the Asynchronous Schedule Enable bit in the USBCMD register. When this bit and the Asynchronous Schedule Enable bit are the same value, the Asynchronous Schedule is either enabled (1) or disabled (0).
14	RO	**Periodic Schedule Status** The bit reports the current real status of the Periodic Schedule. If this bit is a zero then the status of the Periodic Schedule is disabled. If this bit is a one then the status of the Periodic Schedule is enabled. The Host Controller is not required to immediately disable or enable the Periodic Schedule when software transitions the Periodic Schedule Enable bit in the USBCMD register. When this bit and the Periodic Schedule Enable bit are the same value, the Periodic Schedule is either enabled (1) or disabled (0).
13	RO	**Reclamation** This is a read-only status bit, which is used to detect an empty asynchronous schedule.
12	RO	**HCHalted bit** This bit is a zero whenever the Run/Stop bit is a one. The Host Controller sets this bit to one after it has stopped executing as a result of the Run/Stop bit being set to 0, either by software or by the Host Controller hardware (e.g. internal error).
11:6	RO	**Reserved (0)**
5	RW1C	**Interrupt on Async Advance (WC)** System software can force the host controller to issue an interrupt the next time the host controller advances the asynchronous schedule by writing a one to the Interrupt on Async Advance Doorbell bit in the USBCMD register. This status bit indicates the assertion of that interrupt source.
4	RW1C	**Host System Error (WC)** The Host Controller sets this bit to 1 when a serious error occurs during a host system access involving the Host Controller module. In a PCI system, conditions that set this bit to 1 include PCI Parity error, PCI Master Abort, and PCI Target Abort. When this error occurs, the Host Controller clears the Run/Stop bit in the Command register to prevent further execution of the scheduled TDs.
3	RW1C	**Frame List Rollover (WC)** The Host Controller sets this bit to a one when the Frame List Index rolls over from its maximum value to zero. The exact value at which the rollover occurs depends on the frame list size. For example, if the frame list size (as programmed in the Frame List Size field of the USBCMD register) is 1024, the Frame Index Register rolls over every time FRINDEX[13] toggles. Similarly, if the size is 512, the Host Controller sets this bit to a one every time FRINDEX[12] toggles.
2	RW1C	**Port Change Detect (WC)** The Host Controller sets this bit to a one when any port for which the Port Owner bit is set to zero has a change bit transition from a zero to a one or a Force Port Resume bit transition from a zero to a one as a result of a J-K transition detected on a suspended port. This bit will also be set as a result of the Connect Status Change being set to a one after system software has relinquished ownership of a connected port by writing a one to a port's Port Owner bit. This bit is allowed to be maintained in the Auxiliary power well. Alternatively, it is also acceptable that on a D3 to D0 transition of the EHCI HC device, this bit is loaded with the OR of all of the PORTSC change bits (including: Force port resume, over-current change, enable/disable change and connect status change).

to next page

from previous page

1	RW1C	**USB Error Interrupt (WC)** The Host Controller sets this bit to 1 when completion of a USB transaction results in an error condition (e.g., error counter underflow). If the TD on which the error interrupt occurred also had its IOC bit set, both this bit and USBINT bit are set.
0	RW1C	**USB Interrupt (USBINT) (WC)** The Host Controller sets this bit to 1 on the completion of a USB transaction, which results in the retirement of a Transfer Descriptor that had its IOC bit set. The Host Controller also sets this bit to 1 when a short packet is detected (actual number of bytes received was less than the expected number of bytes).

Note :
WC : means this status is write '1' clear, write '0' has no any effect.
Each status bit of this register is set by H/W automatically when the corresponding event occurred or the transaction finished. The corresponding interrupt enable bit wont impact the setting of the bit. The enable bit only determines the interrupt generation.
Whenever S/W has finished the handling of the status bit, it must clear the status bit by writing '1', else it will not be able to recognize new events.

Offset: 28h **EHCI28: USB Interrupt Enable Register (USBINTR)** **Init = 0**

Bit	R/W	Description
31:6	RO	**Reserved (0)**
5	RW	**Interrupt on Async Advance Enable** When this bit is a one, and the Interrupt on Async Advance bit in the USBSTS register is a one, the host controller will issue an interrupt at the next interrupt threshold. The interrupt is acknowledged by software clearing the Interrupt on Async Advance bit.
4	RW	**Host System Error Enable** When this bit is a one, and the Host System Error Status bit in the USBSTS register is a one, the host controller will issue an interrupt. The interrupt is acknowledged by software clearing the Host System Error bit.
3	RW	**Frame List Rollover Enable** When this bit is a one, and the Frame List Rollover bit in the USBSTS register is a one, the host controller will issue an interrupt. The interrupt is acknowledged by software clearing the Frame List Rollover bit.
2	RW	**Port Change Interrupt Enable** When this bit is a one, and the Port Change Detect bit in the USBSTS register is a one, the host controller will issue an interrupt. The interrupt is acknowledged by software clearing the Port Change Detect bit.
1	RW	**USB Error Interrupt Enable** When this bit is a one, and the USBERRINT bit in the USBSTS register is a one, the host controller will issue an interrupt at the next interrupt threshold. The interrupt is acknowledged by software clearing the USBERRINT bit.
0	RW	**USB Interrupt Enable** When this bit is a one, and the USBINT bit in the USBSTS register is a one, the host controller will issue an interrupt at the next interrupt threshold. The interrupt is acknowledged by software clearing the USBINT bit.

Offset: 2Ch **EHCI2C: Frame Index Register (FRINDEX)** **Init = 0**

Bit	R/W	Description
31:14	RO	**Reserved (0)**

to next page

from previous page

13:0	RW	**Frame Index** The value in this register increments at the end of each time frame (e.g. micro-frame). Bits [N:3] are used for the Frame List current index. This means that each location of the frame list is accessed 8 times (frames or micro-frames) before moving to the next index. The following illustrates values of N based on the value of the Frame List Size field in the USBCMD register. USBCMD[Frame List Size] Number Elements N 00 (1024) 12 01 (512) 11 10 (256) 10 11 Reserved

Offset: 34h EHCI34: Periodic Frame List Base Address Register (PERIODICLISTBASE) Init = X

Bit	R/W	Description
31:12	RW	**Base Address** These bits correspond to memory address signals [31:12], respectively.
11:0	RO	**Reserved (0)**

Offset: 38h EHCI38: Current Asynchronous List Address Register (ASYNCLISTADDR) Init = X

Bit	R/W	Description
31:5	RW	**Link Pointer Low (LPL)** These bits correspond to memory address signals [31:5], respectively. This field may only reference a Queue Head (QH).
4:0	RO	**Reserved (0)**

Offset: 60h EHCI60: Configure Flag Register (CONFIGFLAG) Init = 0

Bit	R/W	Description
31:1	RO	**Reserved (0)**
0	RW	**Configure Flag (CF)** Host software sets this bit as the last action in its process of configuring the Host Controller (see Section 4.1). This bit controls the default port-routing control logic. Bit values and side-effects are listed below. **0:** Port routing control logic default-routes each port to an implementation dependent classic host controller. **1:** Port routing control logic default-routes all ports to this host controller.

Offset: 64h EHCI64: Port1 Status/Control Register (PORTSC1) Init = 0x0000_3000

31:22	RO	**Reserved (0)**
21	RW	**Wake on Disconnect Enable (WKDSCNNT_E)** Writing this bit to a one enables the port to be sensitive to device disconnects as wake-up events.
20	RW	**Wake on Connect Enable (WKCNNT_E)** Writing this bit to a one enables the port to be sensitive to device connects as wake-up events. See Section 4.3 for effects of this bit on resume event behavior. Refer to Section 4.3.1 for operational model.

19:14	RO	**Reserved (0)**
13	RW	**Port Owner** This bit unconditionally goes to a 0b when the Configured bit in the CONFIGFLAG register makes a 0b to 1b transition. This bit unconditionally goes to 1b whenever the Configured bit is zero. System software uses this field to release ownership of the port to a selected host controller (in the event that the attached device is not a high-speed device). Software writes a one to this bit when the attached device is not a high-speed device. A one in this bit means that a companion host controller owns and controls the port.
12	RO	**Reserved (1)**
11:10	RO	**Line Status** These bits reflect the current logical levels of the D+ (bit 11) and D- (bit 10) signal lines. These bits are used for detection of low-speed USB devices prior to the port reset and enable sequence. This field is valid only when the port enable bit is zero and the current connect status bit is set to a one. The encoding of the bits are: Bits[11:10] USB State Interpretation 00b SE0 Not Low-speed device, perform EHCI reset 10b J-state Not Low-speed device, perform EHCI reset 01b K-state Low-speed device, release ownership of port 11b Undefined Not Low-speed device, perform EHCI reset.
9	RO	**Reserved (0)**
8	RW	**Port Reset** 0: Port is not in Reset. 1: Port is in Reset. When software writes a one to this bit (from a zero), the bus reset sequence as defined in the USB Specification Revision 2.0 is started. Software writes a zero to this bit to terminate the bus reset sequence. Software must keep this bit at a one long enough to ensure the reset sequence, as specified in the USB Specification Revision 2.0, completes. Note: when software writes this bit to a one, it must also write a zero to the Port Enable bit. Note that when software writes a zero to this bit there may be a delay before the bit status changes to a zero. The bit status will not read as a zero until after the reset has completed. If the port is in high-speed mode after reset is complete, the host controller will automatically enable this port (e.g. set the Port Enable bit to a one). A host controller must terminate the reset and stabilize the state of the port within 2 milliseconds of software transitioning this bit from a one to a zero. For example: if the port detects that the attached device is high-speed during reset, then the host controller must have the port in the enabled state within 2ms of software writing this bit to a zero. The HCHalted bit in the USBSTS register should be a zero before software attempts to use this bit. The host controller may hold Port Reset asserted to a one when the HCHalted bit is a one.

7	RW	**Suspend** 0: Port not in suspend state. 1: Port in suspend state. Port Enabled Bit and Suspend bit of this register define the port states as follows: Bits [Port Enabled, Suspend] Port State 0X Disable 10 Enable 11 Suspend When in suspend state, downstream propagation of data is blocked on this port, except for port reset. The blocking occurs at the end of the current transaction, if a transaction was in progress when this bit was written to 1. In the suspend state, the port is sensitive to resume detection. Note that the bit status does not change until the port is suspended and that there may be a delay in suspending a port if there is a transaction currently in progress on the USB. A write of zero to this bit is ignored by the host controller. The host controller will unconditionally set this bit to a zero when: – Software sets the Force Port Resume bit to a zero (from a one). – Software sets the Port Reset bit to a one (from a zero). If host software sets this bit to a one when the port is not enabled (i.e. Port enabled bit is a zero) the results are undefined.
6	RW	**Force Port Resume** 0: No resume (K-state) detected/driven on port. 1: Resume detected/driven on port. This functionality defined for manipulating this bit depends on the value of the Suspend bit. For example, if the port is not suspended (Suspend and Enabled bits are a one) and software transitions this bit to a one, then the effects on the bus are undefined. Software sets this bit to a 1 to drive resume signaling. The Host Controller sets this bit to a 1 if a J-to-K transition is detected while the port is in the Suspend state. When this bit transitions to a one because a J-to-K transition is detected, the Port Change Detect bit in the USBSTS register is also set to a one. If software sets this bit to a one, the host controller must not set the Port Change Detect bit. Note that when the EHCI controller owns the port, the resume sequence follows the defined sequence documented in the USB Specification Revision 2.0. The resume signaling (Full-speed 'K') is driven on the port as long as this bit remains a one. Software must appropriately time the Resume and set this bit to a zero when the appropriate amount of time has elapsed. Writing a zero (from one) causes the port to return to high- speed mode (forcing the bus below the port into a high-speed idle). This bit will remain a one until the port has switched to the high-speed idle. The host controller must complete this transition within 2 milliseconds of software setting this bit to a zero.
5:4	RO	**Reserved (0)**
3	RW1C	**Port Enable/Disable Change (WC)** 0: No change. 1: Port enabled/disabled status has changed. For the root hub, this bit gets set to a one only when a port is disabled due to the appropriate conditions existing at the EOF2 point (See Chapter 11 of the USB Specification for the definition of a Port Error). Software clears this bit by writing a 1 to it.

2	RW	**Port Enabled/Disabled** 0: Disable. 1: Enable. Ports can only be enabled by the host controller as a part of the reset and enable. Software cannot enable a port by writing a one to this field. The host controller will only set this bit to a one when the reset sequence determines that the attached device is a high-speed device. Ports can be disabled by either a fault condition (disconnect event or other fault condition) or by host software. Note that the bit status does not change until the port state actually changes. There may be a delay in disabling or enabling a port due to other host controller and bus events. When the port is disabled (0b) downstream propagation of data is blocked on this port, except for reset.
1	RW1C	**Connect Status Change (WC)** 0: No change. 1: Change in Current Connect Status. Indicates a change has occurred in the ports Current Connect Status. The host controller sets this bit for all changes to the port device connect status, even if system software has not cleared an existing connect status change. For example, the insertion status changes twice before system software has cleared the changed condition, hub hardware will be ”setting” an already-set bit (i.e., the bit will remain set). Software sets this bit to 0 by writing a 1 to it.
0	RO	**Current Connect Status** 0: No device is present. 1: Device is present on port. This value reflects the current state of the port, and may not correspond directly to the event that caused the Connect Status Change bit (Bit 1) to be set.

Offset: 80h **EHCI80: Frame Length Adjustment Register (FLADJ)** **Init = 0x0000_0020**

Bit	R/W	Description
31:6	RO	**Reserved (0)**
5:0	RW	**Frame Length Timing Value** Each decimal value change to this register corresponds to 16 high-speed bit times. The SOF cycle time (number of SOF counter clock periods to generate a SOF micro-frame length) is equal to 59488 + value in this field. The default value is decimal 32 (20h), which gives a SOF cycle time of 60000. Frame Length (# High Speed bit times) (decimal) / FLADJ Value (decimal) 59488 — 0 (00h) 59504 — 1 (01h) 59520 — 2 (02h) ... 59984 — 31 (1Fh) 60000 — 32 (20h) ... 60480 — 62 (3Eh) 60496 — 63 (3Fh)

Offset: 84h	EHCI84: Controller Fine-tune Register	Init = 0x0000_0A47

Bit	R/W	Description
31:12	RO	**Reserved (0)**
11	RW	**Reserved (1)**
10	RW	**Enable FIFO auto power down** 0: Disable (default) 1: Enable
9	RW	**Companion Controller Selection** 0 : No companion controller reported. 1 : One companion controller reported.
8	RW	**High speed Isochronous IN MaxPacketSize selection** 0 : MaxPacketSize is determined by Transaction_X_Length or Maximum Packet Size (whichever is less). 1 : MaxPacketSize is determined by Maximum Packet Size field.
7:6	RW	**Transmit FIFO Threshold** 00 : 128 bytes 01 : 256 bytes 10 : 512 bytes 11 : 768 bytes
5:2	RW	**Isochronous Scheduling Threshold** This field value reflect to HCCPARAMS bit[7:4].
1	RW	**Asynchronous Schedule Park Capability** This bit value reflect to HCCPARAMS bit[2].
0	RW	**Programmable Frame List Flag** This bit value reflect to HCCPARAMS bit[1].

Offset: 88h	EHCI88: Frame Timing Adjustment	Init = 0x40100000

Bit	R/W	Description
31:22	RW	**preEOF1 timing** EOF1 timing point before SOF is the last executable transaction time, the actual EOF1 time calculation equals MaxPacketSize + preEOF1. The unit is byte.
21:12	RW	**preEOF2 timing** EOF2 timing point before SOF is the time for host controller to stop processing list structure. The unit is byte.
11:0	RW	**SOF transmit delay timing** Setting this field to delay the SOF packet transmission after actual internal SOF timing point. This is used to extend EOF2 timing. The unit is byte.

Offset: 8Ch	EHCI8C: Hardware Revision Number Register	Init = 0x0000_0002

Bit	R/W	Description
31:8	RO	**Reserved (0)**
7:0	RO	**Hardware revision number**

22 Software Interrupt Controller (SWVIC)

22.1 Overview

Software Interrupt Controller (SWVIC) is an AMBA salve device directly connected to AHB bus. It provides hardware interrupt interfaces to interrupt ARM Cortex A7 and Cortex M3.

Base address of SWVIC = 0x1E6C_0000
Physical address = (Base address of SWVIC) + Offset

Offset: 18h	SWVIC18: CA7 Software Interrupt Register		Init = 0
Bit	**R/W**	**Reset**	**Description**
31:15	RO	-	**reserved (0)**
14:0	RW1S	Rst04	**Software Interrupt Generation (CA7SIRQ0-CA7SIRQ14) to ARM Cortex A7** Setting a bit generates a software interrupt for the specific source interrupt before interrupt masking. Write '1' sets the corresponding bit to value "1", and write '0' has no effect.
Note : SWVIC10, SWVIC18 and SWVIC1C are typically used for Cortex M3 to assert software interrupt to Cortex A7.			

Offset: 1Ch	SWVIC1C: CA7 Software Interrupt Clear Register		Init = 0
Bit	**R/W**	**Reset**	**Description**
31:15	RO	-	**reserved (0)**
14:0	W1C	-	**Clear bits in the SWVIC18 register (CA7SIRQ0-CA7SIRQ14)** Write '1' clears the corresponding bit in the SWVIC18 register to value "0", and write '0' has no effect.

Offset: 28h	SWVIC28: CM3 Software Interrupt Register		Init = 0
Bit	**R/W**	**Reset**	**Description**
31:15	RO	-	**reserved (0)**
14:0	RW1S	RstARM	**Software Interrupt Generation (CM3SIRQ0-CM3SIRQ14) to ARM Cortex M3** Setting a bit generates a software interrupt for the specific source interrupt before interrupt masking. Write '1' sets the corresponding bit to value "1", and write '0' has no effect.
Note : SWVIC20, SWVIC28 and SWVIC2C are typically used for Cortex A7 to assert software interrupt to Cortex M3.			

Offset: 2Ch	SWVIC2C: CM3 Software Interrupt Clear Register		Init = 0
Bit	**R/W**	**Reset**	**Description**
31:15	RO	-	**reserved (0)**
14:0	W1C	-	**Clear bits in the SWVIC28 register (CM3SIRQ0-CM3SIRQ14)** Write '1' clears the corresponding bit in the SWVIC28 register to value "0", and write '0' has no effect.

23 SDRAM Memory Controller (MMC)

23.1 Overview

The DRAM controller of AST2600 is not backward compatible to AST2500.

23.1.1 Features

- Only support the DRAM size that has Column address (CA) = 10 bits (A0∼A9).
- Only support the DRAM BurstLen = 8 mode.
- Supported DRAM configurations.
  - DDR4: 128Mb x 16 (2Gb), 256Mb x 16 (4Gb), 512Mb x 16 (8Gb), 1Gb x 16 (16Gb), 1Gb x 16 (16Gb TwinDie 1 rank)
- Support I/O timing calibration function.
- Support programmable size of Vertical ECC protection function, the overhead of memory size equals to 1/8 ECC protected memory size.
  - Actual DRAM Size required when ECC enabled = ECC_protected_size * 9 / 8
- Support 2 write protect regions for DDR memory write protect.

23.1.2 ECC Features

- SECDED ECC algorithm that using Hamming Code Sequence
- ECC Structure: 1 byte ECC code for each 8 bytes of data
- Memory Organization: for each 128 bytes data cascaded by 16 bytes of ECC code. This is gotten by considering the best DRAM access efficiency, same page and burst 8 continuity. The address remapping is done by hardware, it is completely firmware transparent.
- Support ECC verify function at MCRB0.

23.1.3 ECC Verify

The ECC verify procedure as below:

1. Set error injection control at MCRB0. Then at next write to DRAM in memory controller, the bits indicated by MCRB0 will be toggled and write to DRAM.
2. Wait MCRB0[7] = 0 for error injection done. And the memory address of write command with error injected will be recorded at MCRB4.
3. Read MCRB4 for the error injected address.
4. Do a read on the error injected address to trigger ECC error event.
5. Check MCR50 for the ECC error event. And check if MCR58 or MCR5C match to MCRB4.

23.1.4 Register Summary

Base address of MCR = 0x1E6E_0000
Physical address = (Base address of SDRAM Controller) + Offset

MCR00: Protection Key Register
MCR04: Configuration Register *(Key protected)*
MCR08: Graphics Memory Protection Register *(Key protected)*
MCR0C: Refresh Timing Register *(Key protected)*

MCR10: AC Timing Register #1 *(Key protected)*
MCR14: AC Timing Register #2 *(Key protected)*
MCR18: AC Timing Register #3 *(Key protected)*
MCR1C: AC Timing Register #4 *(Key protected)*
MCR20: MR0/MR1 Mode Setting Register *(Key protected)*
MCR24: MR2/MR3 Mode Setting Register *(Key protected)*
MCR28: MR4/MR5 Mode Setting Register *(Key protected)*
MCR2C: MR6 Mode Setting Register *(Key protected)*
MCR30: Mode Setting Control Register *(Key protected)*
MCR34: Power Control Register *(Key protected)*
MCR38: Arbitration Control Register *(Key protected)*
MCR3C: Request Queued Limitation Mask Register *(Key protected)*
MCR40: Maximum Grant Length Register #1 *(Key protected)*
MCR44: Maximum Grant Length Register #2 *(Key protected)*
MCR48: Maximum Grant Length Register #3 *(Key protected)*
MCR4C: Maximum Grant Length Register #4 *(Key protected)*
MCR50: Interrupt Control/Status Register
MCR54: ECC Address Range Control Register *(Key protected)*
MCR58: Address of First Un-Recoverable ECC Error Address *(Key protected)*
MCR5C: Address of Last Recoverable ECC Error Address *(Key protected)*
MCR60: DDR PHY Control/Status Register #1 *(Key protected)*
MCR64: DDR PHY Control/Status Register #2 *(Key protected)*
MCR68: DDR PHY I2C Debug Interface Register *(Key protected)*
MCR6C: Reserved Register
MCR70: ECC and Testing Control/Status Register
MCR74: Testing Start Address and Length Register
MCR78: Testing Fail DQ Bit Register
MCR7C: Test Initial Value Register
MCR80: Enable Request Input Control Register *(Key protected)*
MCR84: High Priority Request Setting Register *(Key protected)*
MCR88: DRAM Controller Software Release Date
MCR8C: DRAM Controller Initial Time
MCR90: Write Protect Region0 Lower Bound Register *(Key protected)*
MCR94: Write Protect Region0 Upper Bound Register *(Key protected)*
MCR98: Write Protect Region1 Lower Bound Register *(Key protected)*
MCR9C: Write Protect Region1 Upper Bound Register *(Key protected)*
MCRA0: Write Protect Registers Lock Control Register *(Key protected)*
MCRB0: ECC Test Control Register *(Key protected)*
MCRB4: ECC Error Bits Injected Address
MCRD0: Write Priviledge Control of Pretect Region0 Register *(Key protected)*
MCRD4: Read Priviledge Control of Pretect Region0 Register *(Key protected)*
MCRD8: Write Priviledge Control of Pretect Region1 Register *(Key protected)*
MCRDC: Read Priviledge Control of Pretect Region1 Register *(Key protected)*

Chang the setting of Memory Controller registers often results in significant impact on SOC operations. Therefore, all these registers have to be well protected.

23.2 Fixed Priority DRAM Request

Priority	Request ID	Group	Request Source
1	REQ0	1	VGA hardware cursor read
2	REQ1	1	VGA CRT read
3	REG2	1	SOC Display Controller read
4	REQ3	1	PCI-E bus1 read/write
5	REQ4	1	Video high priority write
6	REQ5	1	CPU read/write
7	REQ6	1	SLI read/write
8	REQ7	1	PCI-E bus2 read/write
9	REQ8	2-1	USB2.0 Hub/EHCI1 DMA read/write
10	REQ9	2-1	USB2.0 Device/EHCI2 DMA read/write
11	REQ10	2-1	USB1.1 UHCI host read/write
12	REQ11	2-1	AHB bus read/write
13	REQ12	2-1	CM3 data read/write
14	REQ13	2-1	CM3 instruction read
15	REQ14	2-1	MAC0 DMA read/write
16	REQ15	2-1	MAC1 DMA read/write
17	REQ16	2-1	SDIO DMA read/write
18	REQ17	2-1	Pilot Engine read/write
19	REQ18	2-1	XDMA1 read/write
20	REQ19	2-1	MCTP1 read/write
21	REQ20	2-1	Video flag read/write
22	REQ21	2-1	Video low priority write
23	REQ22	2-1	2D engine data read/write
24	REQ23	2-1	Encryption engine read/write
25	REQ24	2-1	MCTP2 read/write
26	REQ25	2-1	XDMA2 read/write
27	REQ26	2-1	ECC/RSA read/write

23.3 Registers: Base Address = 0x1E6E:0000

Offset: 00h	MCR00: Protection Key Register	Init = 0

Bit	R/W	Description
31:0	RW	**Protection key** This register is designed to protect all registers of SDRAM controller from unpredictable updates. **Unlock SDRAM registers**: Write 0xFC600309 to this register **Soft-Lock until next unlock**: Write others value to this register **Hard-Lock until reset**: Write 0xDEADDEAD to this register Whenever finished the initialization of SDRAM registers, please always set SDRAM registers into **locking** mode. The initial state of this register is soft-lock mode. When this register is soft-locked, the read back value of this register is 0x00000000. When this register is hard-locked, the read back value of this register is 0x00000010. When this register is unlocked, the read back value of this register is 0x00000001.

Offset: 04h	MCR04: Configuration Register	Init = 0x30000044

Bit	R/W	Description
31:28	RO	**DRAM Controller Hardware Version** 0-2: earlier chip revision 3: AST2600 others: Reserved
27:20	RW	**DRAM Controller Software Version** 0: earlier chip revision others: Software defined
19:18	RW	**Group 2 (Level 2) requests bank interleaving selection** bit18: apply to normal priority requests bit19: apply to high priority requests 0: disable bank interleaving 1: enable bank interleaving
17:16	RW	**Group 1 (Level 2) requests bank interleaving selection** bit16: apply to normal priority requests bit17: apply to high priority requests 0: disable bank interleaving 1: enable bank interleaving
15:14	RW	**Group 2 (Level 1) requests arbitratrion policy** bit14: apply to normal priority requests bit15: apply to high priority requests 0: round robin 1: fixed
13:12	RW	**Group 1 (Level 1) requests arbitratrion policy** bit12: apply to normal priority requests bit13: apply to high priority requests 0: round robin 1: fixed
11	RW	**RQ Output Bank (Level 3) Arbitration mode** 0: round robin 1: fixed, read higher than write, lower bank number first
10	RW	**Enable swap BA0 and BA2 address bit** 0: normal order 1: swapped order This may can improve the Read/Write efficiency for DDR4.

to next page

from previous page

9	RW	**Enable ECC auto scrubbing for recoverable errors** 0: No scrub memory 1: Auto scrub memory after recoverable ECC data error detected.
8	RW	**Enable Data Scramble function** 0: Disable scramble function 1: Enable data scramble function to reduce maximum SSO current.
7	RW	**Enable ECC function** 0: Disable ECC function 1: Enable ECC function, need extra 1/8 memory size cost for ECC data.
6	RO	**Reserved for VGA driver backward compatible** 1: 16 bits
5	RW	**Select DDR4 16Gb TwinDie Single Rank Component**
4	RW	**Select DRAM type** 0: DDR3L 1: DDR4
3:2	RW	**Select VGA memory size** 00: Select 8 M bytes of graphics memory aperture size 01: Select 16M bytes of graphics memory aperture size 10: Select 32M bytes of graphics memory aperture size 11: Select 64M bytes of graphics memory aperture size Set from the SCU500[14:13]
1:0	RW	**Select the total memory capacity for oversized write protection** 00: Select 256M bytes or smaller as the total data memory capacity 01: Select 512M bytes as the total data memory capacity 10: Select 1024M bytes as the total data memory capacity 11: Select 2048M bytes as the total data memory capacity (DDR4 only) The total memory size setting will enable the protection for memory write that exceed the defined size. It will not wrap around. For memory read, oversize will wrap around.

Offset: 08h **MCR08: Graphics Memory Protection Register** **Init = 0**

Bit	R/W	Description
31:0	RW	**Enable graphics memory request protection** Bit[n]: Protect REQn .... Bit[1]: Protect REQ1 Bit[0]: Protect REQ0 1: re-allocated to the highest memory space 0: not changed This register is designed to protect SDRAM memory from improper graphics memory updates by host CPU. SDRAM memory controller can serve a bunch of memory access requests from REQ0 to REQn. All the memory requests from VGA Display Controller and PCI Bus Controller are among them. When the register bit corresponding to REQn is programmed to be 1, all the accesses of REQn will be re-allocated by address re-mapping to the highest memory space defined by MCR04 [1:0]. When being programmed to be 0, the request is not changed.

Offset: 0Ch	MCR0C: Refresh Timing Register	Init = 0
Bit	**R/W**	**Description**
31:16	RW	**Period of ZQCS cycle** This value define the period of executing ZQCS command. The unit is the number of refresh cycles. For example, 128ms = 7.5 us * 17066(0x42AA).
15:8	RW	**Period of high-priority SDRAM refresh cycle** AST2600 uses 12.5MHz clock source as the reference clock to generate high-priority SDRAM refresh cycle request according to the following equation. Value = round_to_minimum(12.5 * refresh_ period_in_us_unit)
7	RW	**Enable ZQCS command** 0: Disable 1: Enable, ZQCS command will be issued periodically defined by ZQCS preiod.
6	RW	**Enable ZQCL when reset DDR DLL** 0: Disable 1: Enable, ZQCL command will be issued automatically after reset DLL MRS command.
5	RW	**Enable low-priority SDRAM refresh cycle** 0: Disable low-priority SDRAM refresh cycle 1: Enable low-priority SDRAM refresh cycle When this bit is enabled, SDRAM Controller will issue SDRAM refresh cycle requests with the lowest priority to best utilize the available memory bandwidth. Whenever memory requests bus are idled and then SDRAM Controller will automatically issue low-priority SDRAM refresh cycle requests. The maximum accumulated number of refresh cycle is 8. Low-Priority refresh cycles will stop whenever any request is pending for execution.
4	RW	**Force all banks to be pre-charged before refresh cycles** 0: Disabled 1: Enabled In general, SDRAM will only pre-charge the banks needed to be pre-charged before refresh cycle. When this bit is enabled, SDRAM will pre-charge all the banks, no matter the bank status. This register is designed for insurance policy only.
3:1	RW	**Reserved**
0	RW	**Refresh enable**
Note : When working at higher ambient temperature (85^oC), the refresh period must be 2X faster.		

Offset: 10h	MCR10: AC Timing Register #1	Init = 0
Bit	**R/W**	**Description**
31:24	RW	**t-RTP timing setting (read to pre-charge)** Timing = 2n + 1 (CK), n is the register value. This timing setting determines the number of delay cycles from read command to pre-charge command of the same bank.
23:16	RW	**t-WTP Timing Setting (write to pre-charge)** Timing = 2n + 1 (CK), n is the register value. This timing setting determines the number of delay cycles from write command to pre-charge command of the same bank.
15:8	RW	**t-RTW Timing Setting (Read to Write command)** Timing = 2n + 3 (CK), n is the register value. This timing setting determines the number of delay cycles from read command to write command.

to next page

from previous page

7:0	RW	**t-WTR Timing Setting (Write to Read command)** Timing = 2n + 3 (CK), n is the register value. This timing setting determines the number of delay cycles from write command to read command.

Offset: 14h **MCR14: AC Timing Register #2** **Init = 0**

Bit	R/W	Description
31:24	RW	**t-RAS timing setting (active to precharge timing)** Timing = 2n + 3 (CK), n is the register value.
23:20	RW	**DRAM Write Latency (WL) timing setting** Timing = WL - 5, supports WL from 5 to 20
19:16	RW	**DRAM Read Latency (CL) timing setting** Timing = CL - 4, supports CL from 4 to 19
15:12	RW	**t-XP CKE exit delay from power down mode to any command** Timing = 2n + 4 (CK), n is the register value.
11:8	RW	**t-CKE, CKE minimum high/low pulse width control** MIN pulse width = 2n + 2 (CK), n is the register value.
7:0	RW	**t-RFC all bank refresh interval timing setting** Timing = 2n + 4 (CK), n is the register value.

Offset: 18h **MCR18: AC Timing Register #3** **Init = 0**

Bit	R/W	Description
31:24	RW	**t-FAW four bank active window** Timing = 2n + 1 (CK), n is the register value.
23:16	RW	**t-RP timing setting** Timing = 2n + 4 (CK), n is the register value.
15:8	RW	**t-RRD timing setting (active-to-active)** Timing = 2n + 1 (CK), n is the register value.
7:0	RW	**t-RCD timing setting (active-to-read/write)** Timing = 2n + 7 (CK), n is the register value.

Offset: 1Ch **MCR1C: AC Timing Register #4** **Init = 0**

Bit	R/W	Description
15	RW	**Write data delay phase selection** In normal operation, this bit indicates the WL is even or odd. 1: even 0: odd
14	RW	**t-CCD_L timing mode** 0: apply to DDR4 same bank group only (BG = BA2) 1: apply always
13:12	RW	**t-CCD_L timing setting** Timing = n + 4 (CK) This timing is required for DDR4 SDRAM only.
11:10	RW	**t-RRD_L timing setting** Timing = 2n (CK) This value equals the difference of t-RRD_L - t-RRD_S. This timing is required for DDR4 SDRAM only.

to next page

from previous page

9:8	RW	**t-WTR_L timing setting** Timing = 2n (CK) This value equals the difference of t-WTR_L - t-WTR_S. This timing is required for DDR4 SDRAM only.
7:0	RW	**t-ZQCS timing setting** Timing = 2n + 2 (CK), n is the register value.

Offset: 20h **MCR20: MR0/MR1 Mode Setting Register** **Init = X**

Bit	R/W	Description
31:29	RO	**Reserved (0)**
28:16	RW	**Mode Register 1 Setting (MR1)** The definition of this register depends on the populated SDRAM type. **For DDR3 SDRAM type**: Bit [12] (Qoff) 0 : Output buffer enabled (DQs, DQSs, DQS#s) 1 : Output buffer disabled Bit [11] (TDQS Enable) 0 : Disabled 1 : Enable Bit [10,8]: Reserved (0) Bit [7] (Write leveling enable) 0: Disabled 1: Enable (Not supported) Bit [4:3] (Additive Latency) 00: 0 01: CL-1 (Not supported) 10: CL-2 (Not supported) 11: Reserved Bit [9,6,2] (Rtt_Nom) 000: Rtt_Nom disabled 001: RZQ/4 010: RZQ/2 011: RZQ/6 100: RZQ/12 101: RZQ/8 11x: Reserved others: Invalid Bit [5,1] (Output driver impedance control) 00: RZQ/6 01: RZQ/7 1x: Reserved Bit [0] (DLL Disable) 0: Enable DLL 1: Disable DLL **For DDR4 SDRAM type**: Bit [12] (Qoff) 0 : Output buffer enabled (DQs, DQSs, DQS#s) 1 : Output buffer disabled

to next page

from previous page

		Bit [11] (TDQS Enable) 0 : Disabled 1 : Enable Bit [10:8] (Rtt_Nom) 000: Rtt_Nom disabled 001: RZQ/4 010: RZQ/2 011: RZQ/6 100: RZQ/1 101: RZQ/5 110: RZQ/3 111: RZQ/7 others: Invalid Bit [7] (Write leveling enable) 0: Disabled 1: Enable (Not supported) Bit [6,5]: Reserved (0) Bit [4:3] (Additive Latency) 00: 0 01: CL-1 (Not supported) 10: CL-2 (Not supported) 11: Reserved Bit [2:1] (Output driver impedance control) 00: RZQ/7 01: RZQ/5 1x: Reserved Bit [0] (DLL Disable) 1: Enable DLL 0: Disable DLL
15:13	RO	**Reserved (0)**
12: 0	RW	**Mode Register 0 Setting (MR0)** The definition of this register depends on the populated SDRAM type. **For DDR3 SDRAM type**: Bit [12] (DLL Control for Precharge PD) 0: Slow exit (DLL off), use t-XPDLL 1: Fast exit (DLL on), use t-XP Bit [11:9] (Write Recovery for auto precharge) 000: 16T 001: 5T 010: 6T 011: 7T 100: 8T 101: 10T 110: 12T 111: 14T Bit [8] (DLL Reset) Bit [7] (Test Mode)

to next page

from previous page

Bit [6:4,2] (CAS Latency)
0000: Reserved
0010: CAS latency = 5T
0100: CAS latency = 6T
0110: CAS latency = 7T
1000: CAS latency = 8T
1010: CAS latency = 9T
1100: CAS latency = 10T
1110: CAS latency = 11T
0001: CAS latency = 12T
0011: CAS latency = 13T
0101: CAS latency = 14T

Bit [3] (Read Burst Type)
0: Nibble Sequential
1: Interleave (Not Supported)

Bit [1:0] (Burst Length)
00: Burst 8
01: BC4 or BL8 (on the fly)
10: BC4 (Fixed)
11: Reserved

For DDR4 SDRAM type:

Bit [12] (Set to 0)

Bit [11:9] (Write Recovery and Read to Precharge for auto precharge)
000: WR = 10T, RTP = 5
001: WR = 12T, RTP = 6
010: WR = 14T, RTP = 7
011: WR = 16T, RTP = 8
100: WR = 18T, RTP = 9
101: WR = 20T, RTP = 10
110: WR = 24T, RTP = 12
111: Reserved

Bit [8] (DLL Reset)

Bit [7] (Test Mode)

Bit [6:4,2] (CAS Latency)
0000: CAS latency = 9T
0001: CAS latency = 10T
0010: CAS latency = 11T
0011: CAS latency = 12T
0100: CAS latency = 13T
0101: CAS latency = 14T
0110: CAS latency = 15T
0111: CAS latency = 16T
1000: CAS latency = 18T
1001: CAS latency = 20T
1010: CAS latency = 22T
1011: CAS latency = 24T
11xx: Reserved

Bit [3] (Read Burst Type)
0: Nibble Sequential
1: Interleave (Not Supported)

to next page

from previous page

		Bit [1:0] (Burst Length) 00: Burst 8 01: BC4 or BL8 (on the fly) 10: BC4 (Fixed) 11: Reserved

Offset: 24h **MCR24: MR2/MR3 Mode Setting Register** **Init = X**

Bit	R/W	Description
31:29	RO	**Reserved (0)**
28:16	RW	**Mode Register 3 Setting (MR3)** The definition of this register depends on the populated SDRAM type. **For DDR3 SDRAM type**: Bit [12:0] (all '0') **For DDR4 SDRAM type**: Bit [12:11] (MPR Read Format) 00: Serial 01: Parallel 10: Staggered 11: ReservedTemperature Bit [10:9]: Write CMD Latency when CRC and DM are enabled 00: 4nCK (1600) 01: 5nCK (1866,2133,2400) 10: 6nCK (TBD) 11: RFU Bit [8:6] (Fine Granularity Refresh Mode) 000: Normal (Fixed 1x) 001: Fixed 2x 010: Fixed 4x 011: Reserved 100: Reserved 101: Enable on the fly 2x (non support) 110: Enable on the fly 2x (non support) 111: Reserved Bit [5] (Temperature sensor readout) 0: Disable 1: Enable Bit [4] (Per DRAM Addressability) 0: Disable 1: Enable (non support) Bit [3] (Geardown Mode) 0: 1/2 Rate 1: 1/4 Rate Bit [2] (MPR Operation) 0: Normal 1: Dataflow from/to MPR Bit [1:0] (MPR page Selection) 00: Page 0 01: Page 1 10: Page 2 11: Page 3

to next page

from previous page

15:13	RO	**Reserved (0)**
12:0	RW	**Mode Register 2 Setting (MR2)** The definition of this register depends on the populated SDRAM type. **For DDR3 SDRAM type**: Bit [12:11] (all '0') Bit [10:9] (Rtt_WR) 00: Dynamic ODT off (Write does not affect Rtt value) 01: RZQ/4 10: RZQ/2 11: Reserved Bit [8] (all '0') Bit [7] (Self-Refresh Temperature (SRT) Range) 0: Normal operating temperature range 1: Extended (optional) operating temperature range Bit [6] (Auto Self-Refresh (ASR)) 0: Manual SR Reference (SRT) 1: ASR enable (Optional) Bit [5:3] (CAS write Latency (CWL)) 000: 5 (tCK(avg) $\geq$ 2.5ns) 001: 6 (2.5ns $>$ tCK(avg) $\geq$ 1.875ns) 010: 7 (1.875ns $>$ tCK(avg) $\geq$ 1.5ns) 011: 8 (1.5ns $>$ tCK(avg) $\geq$ 1.25ns) 100: 9 (1.25ns $>$ tCK(avg) $\geq$ 1.071ns) 101: 10(1.071ns $>$ tCK(avg) $\geq$ 0.938ns) 11x: Reserved Bit [2:0] (Partial Array Self-Refresh) 000: Full Array 001: HalfArray (BA[2:0]=000,001,010,011) 010: Quarter Array (BA[2:0]=000,001) 011: 1/8th Array (BA[2:0] = 000) 100: 3/4 Array (BA[2:0] = 010,011,100,101,110,111) 101: HalfArray (BA[2:0] = 100,101,110,111) 110: Quarter Array (BA[2:0]=110,111) 111: 1/8th Array (BA[2:0]=111) **For DDR4 SDRAM type**: Bit [12] (Write CRC) 0: Disable 1: Enable (mom support) Bit [11] (all '0') Bit [10:9] (Rtt_WR) 00: Dynamic ODT off (Write does not affect Rtt value) 01: RZQ/2 10: RZQ/1 11: Hi-Z Bit [8] (all '0') Bit [7] (Self-Refresh Temperature (SRT) Range) 0: Normal operating temperature range 1: Extended (optional) operating temperature range

to next page

from previous page

		Bit [7:6] (Low Power Array Self Refresh (LP ASR)) 00: Manual Mode (Normal Operaing Temperature Range) 01: Manual Mode (Reduced Operating Temperature Range) 10: Manual Mode (Extended Operating Temperature Range) 11: ASR Mode (Auto Self Refresh) Bit [5:3] (CAS write Latency (CWL)) 000: 9 (1600) 001: 10(1866) 010: 11(2133,1600) 011: 12(2400,1866) 100: 14(2133) 101: 16(2400) 110: 18 111: Reserved Bit [2:0] (all '0')

Offset: 28h **MCR28: MR4/MR5 Mode Setting Register** **Init = X**

Bit	R/W	Description
31:29	RO	**Reserved (0)**
28:16	RW	**Mode Register 5 Setting (MR5)** The definition of this register depends on the populated SDRAM type. **For DDR3 SDRAM type**: Bit [12:0] (all '0') **For DDR4 SDRAM type**: Bit [12] (Read DBI) 0: Disable 1: Enable Bit [11] (Write DBI) 0: Disable 1: Enable (non support) Bit [10] (Data Mask) 0: Disable 1: Enable Bit [9] (CA parity Persistent Error) 0: Disable 1: Enable (non support) Bit [8:6] (RTT_PARK) 000: Disable 001: RZQ/4 010: RZQ/2 011: RZQ/6 100: RZQ/1 101: RZQ/5 110: RZQ/3 111: RZQ/7 Bit [5] (ODT Input Buffer during Power Down mode) 0: ODT input buffer is activated 1: ODT input buffer is deactivated

to next page

from previous page

		Bit [4] (C/A Parity Error Status) 0: Clear 1: Error (non support) Bit [3] (CRC Error Clear) 0: Clear 1: Error (no support) Bit [2:0] (C/A Parity Latency Mode) 000: Disable others: Non support
15:13	RO	**Reserved (0)**
12: 0	RW	**Mode Register 4 Setting (MR4)** The definition of this register depends on the populated SDRAM type. **For DDR3 SDRAM type**: Bit [12:0] (all '0') **For DDR4 SDRAM type**: Bit [12] (Write Preamble) 0: 1nCK 1: 2nCK Bit [11] (Read Preamble) 0: 1nCK 1: 2nCK Bit [10] (Read Preamble Training Mode) 0: Disable 1: Enable Bit [9] (Self Refresh Abort) 0: Disable 1: Enable Bit [8:6] (CS to CMD/ADDR Latency Mode (cycles)) 000: Disable others: None support Bit [5] (all '0') Bit [4] (Internal Vref Monitor) 0: Disable 1: Enable Bit [3] (Temperature Controlled Refresh Mode) 0: Disable 1: Enable Bit [2] (Temperature Controlled Refresh Range) 0: Normal 1: Extended Bit [1] (Maximum Power Down Mode) 0: Disable 1: Enable Bit [0] (all '0')

Offset: 2Ch	MCR2C: MR6 Mode Setting Register	Init = X

Bit	R/W	Description
31:16	RO	**Reserved (0)**
15:13	RO	**Reserved (0)**
12:0	RW	**Mode Register 6 Setting (MR6)** The definition of this register depends on the populated SDRAM type. **For DDR3 SDRAM type**: Bit [12:0] (all '0') **For DDR4 SDRAM type**: Bit [12:10] (tCCD_L) 000: tCCD_L.min = 4 nCK, tDLLKmin = 597 nCK (≤ 1333Mbps) 001: tCCD_L.min = 5 nCK, tDLLKmin = 597 nCK (≤ 1866Mbps) 010: tCCD_L.min = 6 nCK, tDLLKmin = 768 nCK (≤ 2400Mbps) 011: tCCD_L.min = 7 nCK, tDLLKmin = 1024 nCK (TBD) 100: tCCD_L.min = 8 nCK, tDLLKmin = 1024 nCK (TBD) others: Reserved Bit [9:8] (all '0') Bit [7] (VrefDQ Training Enable) 0: Disable(Normal operation Mode) 1: Enable(Training Mode) Bit [6] (VrefDQ Training Range) 0: Range 1 1: Range 2 Bit [5:0] (VrefDQ Training Value)

Offset: 30h	MCR30: Mode Setting Control Register	Init = 0

Bit	R/W	Description
31:5	RO	**Reserved (0)**
4	RW	**Enable Reset DLL delay timer** 0: No delay 1: Enable delay The timer is defined in the memory controller to block command execution.
3:1	RW	**Mode register selections** 000 : MR0 : Mode register 0 001 : MR1 : Mode register 1 010 : MR2 : Mode register 2 011 : MR3 : Mode register 3 100 : MR4 : Mode register 4 101 : MR5 : Mode register 5 110 : MR6 : Mode register 6 111 : Reserved
0	RW	**Fire mode register setting and status flag** 0: No fire/command setting done flag 1: Fire command Set this bit to '1' will fire mode register setting. When finished, HW will automatically clear this bit to 0. Then SW can do the next mode setting command; HW will automatically control the timing requirement. Before this command has been done, AHB bus will be locked to prevent other command entering SDRAM controller, so SW can set the command continuously without delay.

Offset: 34h	MCR34: Power Control Register	Init = 0

Bit	R/W	Description
31	RO	**Current CKE pin output value** (for debugging purpose only)
30:28	RO	**Self Refresh Control State Machine Status** (for debugging purpose only) 000 : Idle 001 : wait memory controller all parts entering idle state 010 : entering self refresh state 011 : wait tXSNR for starting refresh command 100 : wait MRS DLL reset command 101 : wait 512 clock cycle for DLL recovery
27:16	RO	**Reserved (0)**
15:12	RW	**SDRAM ODT delay from write command** 0000: delay 2 DDR clocks 0001: delay 3 DDR clocks .... 1111: delay 17 DDR clocks Generally set as 0 for DDR3 and DDR4. For DDR3/DDR4, ODTLon = WL - 2, ODTLoff = WL - 2
11:10	RW	**Enable extend SDRAM ODT cycle** 00: No extend 01: extend 1 DDR clock 10: extend 2 DDR clock 11: extend 3 DDR clock The default ODT pulse width for each write command is 6 DDR clocks
9	RW	**Enable SDRAM ODT Auto-ON/OFF mode** 0: Disable auto mode 1: Auto turn ON/OFF SDRAM ODT for write cycles. (ODT must be enabled first.)
8	RW	**Enable DDR3/DDR4 SDRAM ODT** 0: Disable 1: Enable
7	RW	**SDRAM ResetN Output** 0: Enable SDRAM reset, output low. 1: Disable SDRAM reset, output high.
6	RW	**Disable memory request input** 0: Enable, normal operation 1: Disable, when DDR init cycle
5	RW	**Disable memory request bypass mode** 0: Request bypass, all internal FIFOs will be reset 1: Normal operation mode
4	RW	**Enable SDRAM Read Gap Control** 0: no gap control 1: enable gap control for PHY read output
3	RW	**Disable CK/CK# output when entering self refresh mode** 0 : Enable (Output buffers are always enabled) 1 : Disable (Output buffers are ON only when not in self refresh mode)
2	RW	**Force SDRAM to enter self refresh mode** 0 : Normal mode or exit from self refresh mode 1 : Force SDRAM to enter self refresh mode

to next page

from previous page

1	RW	**Enable SDRAM auto power down function** 0: Disable auto power down function 1: Enable auto power down function when no SDRAM request When enabling auto power down function, SDRAM Controller will dynamically drive CKE signal to control the power consumption of SDRAM chips.
0	RW	**SDRAM CKE Enable** 0 : Disable CKE function (force CKE signal at 0 state after power-on reset) 1 : Enable CKE function

Offset: 38h **MCR38: Arbitration Control Register** **Init = 0**

Bit	R/W	Description
31:21	RO	**Reserved (0)**
20:16	RW	**R/W max-grant count for RQ output arbitration** Define the max-grant count to switch between Read and Write requests at RQ output.
15	RO	**Reserved (0)**
14:8	RW	**Group 2 requests queued number control threshold value** This register is designed to limit the request count in Request Queue. Used in combination with MCR3C.
7	RO	**Reserved (0)**
6:0	RW	**Group 1 requests queued number control threshold value** This register is designed to limit the request count in Request Queue. Used in combination with MCR3C.

Offset: 3Ch **MCR3C: Request Queued Limitation Mask Register** **Init = 0**

Bit	R/W	Description
31:0	RW	**Mask control bits** Bit [0]: Mask bit for REQ0 Bit [1]: Mask bit for REQ1 .... 0: This request will not be masked forever 1: This request will be masked (when the queued requests number over the threshold value defined at MCR38). This register is designed to guarantee high priority requests with low latency, especially the CRT display request, which if suffering long latency time from waiting many low priority requests pending in the request queue, may cause serious screen noise. Therefore, high priority requests are usually not masked so that they will not be ignored even when the queued request number over the threshold value.

Offset: 40h **MCR40: Maximum Grant Length Register #1** **Init = 0**
Offset: 44h **MCR44: Maximum Grant Length Register #2** **Init = 0**
Offset: 48h **MCR48: Maximum Grant Length Register #3** **Init = 0**
Offset: 4Ch **MCR4C: Maximum Grant Length Register #4** **Init = 0**

Bit	Attr.	Description
123:120	RW	**MCR4C [27:24]: Maximum grant length for REQ30**
119:116	RW	**MCR4C [23:20]: Maximum grant length for REQ29**

to next page

from previous page

115:112	RW	**MCR4C [19:16]: Maximum grant length for REQ28**
111:108	RW	**MCR4C [15:12]: Maximum grant length for REQ27**
107:104	RW	**MCR4C [11:08]: Maximum grant length for REQ26**
103:100	RW	**MCR4C [07:04]: Maximum grant length for REQ25**
99 :96	RW	**MCR4C [03:00]: Maximum grant length for REQ24**
95:92	RW	**MCR48 [31:28]: Maximum grant length for REQ23**
91:88	RW	**MCR48 [27:24]: Maximum grant length for REQ22**
87:84	RW	**MCR48 [23:20]: Maximum grant length for REQ21**
83:80	RW	**MCR48 [19:16]: Maximum grant length for REQ20**
79:76	RW	**MCR48 [15:12]: Maximum grant length for REQ19**
75:72	RW	**MCR48 [11:08]: Maximum grant length for REQ18**
71:68	RW	**MCR48 [07:04]: Maximum grant length for REQ17**
67:64	RW	**MCR48 [03:00]: Maximum grant length for REQ16**
63:60	RW	**MCR44 [31:28]: Maximum grant length for REQ15**
59:56	RW	**MCR44 [27:24]: Maximum grant length for REQ14**
55:52	RW	**MCR44 [23:20]: Maximum grant length for REQ13**
51:48	RW	**MCR44 [19:16]: Maximum grant length for REQ12**
47:44	RW	**MCR44 [15:12]: Maximum grant length for REQ11**
43:40	RW	**MCR44 [11:08]: Maximum grant length for REQ10**
39:36	RW	**MCR44 [07:04]: Maximum grant length for REQ9**
35:32	RW	**MCR44 [03:00]: Maximum grant length for REQ8**
31:28	RW	**MCR40 [31:28]: Maximum grant length for REQ7**
27:24	RW	**MCR40 [27:24]: Maximum grant length for REQ6**
23:20	RW	**MCR40 [23:20]: Maximum grant length for REQ5**
19:16	RW	**MCR40 [19:16]: Maximum grant length for REQ4**
15:12	RW	**MCR40 [15:12]: Maximum grant length for REQ3**
11:8	RW	**MCR40 [11:08]: Maximum grant length for REQ2**
7:4	RW	**MCR40 [07:04]: Maximum grant length for REQ1**
3:0	RW	**MCR40 [03:00]: Maximum grant length for REQ0**

to next page

from previous page

Note :
SDRAM Controller totally supports up to 31 SDRAM requests (REQ0 ~ REQ30).
The maximum grant length of each request can be programmed by 4 bits assigned by registers MCR4C ~ MCR40.
The maximum grant length is defined by the following table:

Bit[3:0]	Maximum Grant Length
0, 1	2 x 128 bits
2, 3	4 x 128 bits
4, 5	6 x 128 bits
6, 7	8 x 128 bits
8, 9	10 x 128 bits
10, 11	12 x 128 bits
12, 13	14 x 128 bits
14, 15	16 x 128 bits

Register MCR4C ~ MCR40 defines the maximum grant length allowed to be put into SDRAM request FIFO for each SDRAM request. Properly setting of these registers can effectively control the bandwidth allocations for different SDRAM requests.

Offset: 50h **MCR50: Interrupt Control/Status Register** **Init = 0**

Bit	R/W	Description
31	RW	**Clear interrupt flags and error counters** 0: No operation 1: Reset all interrupts flag and ECC error counters. SW must set this bit back to '0' after cleared the status.
30	RW	**Enable reset CPU when un-recoverable error occur** 0: no reset 1: enable reset
29:24	RO	**Reserved (0)**
23:16	RO	**ECC recoverable error counter** This counter with saturation logic can record up to 255 ECC recoverable error events.
15:12	RO	**ECC un-recoverable error counter** This counter with saturation logic can record up to 15 ECC un-recoverable error events.
11:9	RO	**Reserved (0)**
8	RO	**Interrupt flag of blocked read/write event on Group 2-2**
7	RO	**Interrupt flag of blocked read/write event on Group 2-1**
6	RO	**Interrupt flag of blocked read/write event on Group 1** 0: No interrupt pending 1: Interrupt pending (whenever a blocked read/write event happens) This bit will be cleared by MCR50 [31]. The blocked command has below conditions: 1. Over-sized write 2. Write to protected region 1/2 3. Over-sized read if read wrap mode was disabled 4. Read to protected region 1/2 if read was disabled
5	RO	**Interrupt flag of ECC recoverable error events** 0: No interrupt pending 1: Interrupt pending (whenever a recoverable error event happens) This bit will be cleared by MCR50 [31].

to next page

from previous page

4	RO	**Interrupt flag of ECC un-recoverable error events** 0: No interrupt pending 1: Interrupt pending (whenever an un-recoverable error event happens) This bit will be cleared by MCR50 [31].
3	RO	**Reserved (0)**
2	RW	**Enable blocked read/write event interrupt** 0: Disable 1: Enable over sized error interrupt
1	RW	**Enable ECC recoverable error interrupt** 0: Disable 1: Enable ECC recoverable error interrupt
0	RW	**Enable ECC un-recoverable error interrupt** 0: Disable 1: Enable ECC un-recoverable error interrupt

Offset: 54h **MCR54: ECC Address Range Control Register** **Init = 0x1FF00000**

Bit	R/W	Description
31	RO	**Reserved (0)**
30:20	RW	**Maximum ECC work address range** ECC work area = address $\leq$ max_range. The ECC work area must be defined in the unit of 1M Bytes. The value for this field = max_range - 1, the default value is 512MB. When ECC is enabled, firmware must reserve extra 1/8 address space for ECC spare data storage.
19:18	RO	**Reserved (0)**
17	RO	**Last blocked read/write mode on Group 2-2**
16	RO	**Request ID of the last blocked read/write on Group 2-2**
13	RO	**Last blocked read/write mode on Group 2-1**
12:8	RO	**Request ID of the last blocked read/write on Group 2-1**
3	RO	**Last blocked read/write mode on Group 1** 0: read 1: write
2:0	RO	**Request ID of the last blocked read/write on Group 1**

Note :
The blocked command has below conditions:
1. Over-sized write
2. Write to protected region 1/2
3. Over-sized read if read wrap mode was disabled
4. Read to protected region 1/2 if read was disabled

Offset: 58h **MCR58: Address of First Un-Recoverable ECC Error Address** **Init = 0**

Bit	R/W	Description
31	RO	**Reserved (0)**
30:4	RO	**Address of first un-recoverable ECC error** The address will be recorded again when the error accumulation counter be reset.
3:0	RO	**Reserved (0)**

Offset: 5Ch	MCR5C: Address of Last Recoverable ECC Error Address	Init = 0

Bit	R/W	Description
31	RO	**Reserved (0)**
30:4	RO	**Address of last recoverable ECC error** The address will be recorded again when the error accumulation counter be reset.
3:0	RO	**Reserved (0)**

Offset: 60h	MCR60: DDR PHY Control/Status Register #1	Init = 0

Bit	R/W	Description
31:15	RW	**Reserved**
14:8	RW	**DDR PHY I2C Slave Address**
4	RO	**DDR PHY PLL Lock Status**
2	RW	**DDR PHY Reset Control** 0: Enable reset 1: Disable Reset
1	RW	**Reserved**
0	RW	**DDR PHY Init Control/Status** 0: Idle 1: Enable PHY Init cycle or busy state

Offset: 64h	MCR64: DDR PHY Control/Staus Register #2	Init = 0

Bit	R/W	Description
31:3	RW	**Reserved**
2	RW	**DDR PHY Reset CKE Latch** 0: Enable reset 1: Disable reset
1	RW	**DDR PHY IDDQ Control** 0: Disable IDDQ 1: Enable IDDQ
1	RW	**DDR PHY Power Off Control** 0: Disable power off 1: Enable power off

Offset: 68h	MCR68: DDR PHY I2C Debug Interface Register	Init = 0

Bit	R/W	Description
3	RO	**SDA line state**
2	RO	**SCL line state**
1	RW	**SDA OE** 0: SDA pull high 1: SDA drive low
0	RW	**SCL OE** 0: SCL pull high 1: SCL drive low

Offset: 6Ch	MCR6C: Reserved Register	Init = 0
Bit	**R/W**	**Description**
31:0	RW	**Reserved**

Offset: 70h	MCR70: ECC and Testing Control/Status Register	Init = 0
Bit	**R/W**	**Description**
31:16	RO	**Testing Fail Count** This value shows the accumulated fail count. It is record once for each 64 bits data unit. This value will not overflow if the error numbers over the maximum count.
15:14		**Reserved (0)**
13	RO	**Testing result flag** 0: Pass 1: Fail This flag will be cleared whenever disabling SDRAM tests. AST2600 provides a sequential logic that can effectively test SDRAM memory in a very short period of time. The memory range to be tested is programmable (MCR74 and MCR78). This module can be a good instrument for SDRAM stress tests or SDRAM self tests during boot-up process.
12	RO	**Testing finish flag** 0: Busy 1: Finish This flag will be cleared whenever disabling SDRAM tests.
11:10	RO	**Reserved (0)**
9	RW	**ECC memory segment initialization mode** 0: Normal test mode 1: ECC memory initialization mode ECC memory initialization sequence: 1. Enable ECC mode by setting MCR04[7] = 1 2. Set ECC memory range at MCR54 3. Set initial value, MCR7C = 0 4. Start initialization by setting MCR70 = 0x221 5. Wait MCR70[12] = 1 6. Clear MCR70 = 0 7. Set MCR50 = 0x80000000 8. Clear MCR50 = 0
8:6	RW	**Testing auxiliary control** xx1: Enable Test mode 1 to be executed after Test mode 0. x1x: Enable terminate testing after read error.
5:3	RW	**Testing data generation mode** 000: Byte Toggled data, 0 → 1 → 0 → 1 001: Byte Rotate left, LSB → MSB 010: Byte Rotate right, MSB → LSB 011: Byte Increment at each 8 bits unit 100: Byte sequence data 101: 4 bit sequence data 110: 2 bit sequence data 111: 1 bit sequence data The initial value of testing data will be from MCR7C. The sequential testing data will be generated based on the above selected mode.

to next page

from previous page

2:1	RW	**Testing mode** 00: Write memory only (testing result flag is always 0 after testing) 01: Read back and compare for each location 10: Write one memory location first then read back the location and compare with the expected value 11: Reserved
0	RW	**Enable testing** 0: Disable, reset test function and related status 1: Enable Since SDRAM testing will impact normal graphics display functions, Its not recommended to enable this function after Watchdog Timer reset. Enabling SDRAM testing after power-on reset should be a right time frame.

Offset: 74h **MCR74: Testing Start Address and Length Register** **Init = X**

Bit	R/W	Description
31	RW	**Queue Merge Balance Enable #1**
30:26	RW	**Testing start address base** This value defines the testing base address segment. It is defined at 64MB boundary.
25:4	RW	**Testing total length** This value defines the testing final address (16 bytes boundary). Testing starts from offset 0 (relative to the base address segment) and ends at this final address. The maximum testing space can up to 64MB.
3:1	RO	**Reserved (0)**
0	RW	**Queue Merge Balance Enable #2** It is recommended to set both 31 and 0 to same value. For example, both bits are 1.

Offset: 78h **MCR78: Testing Fail DQ Bit Register** **Init = X**

Bit	R/W	Description
31:0	RO	**Fail DQ bit position** Bit 0 : DQ0 rising edge Bit 1 : DQ1 rising edge .... Bit15 : DQ15 rising edge Bit16 : DQ0 falling edge Bit17 : DQ1 falling edge .... Bit31 : DQ15 falling edge

Offset: 7Ch	MCR7C: Test Initial Value Register	Init = X

Bit	R/W	Description
31:0	RW	**Initial value** **When Data Generation Mode = 000, 001, 010, 011** 07:0 : Initial Value for positive edge DQ 15:8 : Initial Value for negative edge DQ The values are the same for all DQS group. **When Data Generation Mode = 100** 07:0, 23:16 = Value for positive edge DQ and turn by sequence. 15:8, 31:24 = Value for negative edge DQ and turn by sequence. The values are the same for all DQS group. **When Data Generation Mode = 101** 3:0, 11:08, 19:16, 27:24 = Value for positive edge DQ and turn by sequence. 7:4, 15:12, 23:20, 31:28 = Value for negative edge DQ and turn by sequence. The values are the same for each 4 bits DQ. **When Data Generation Mode = 110** 1:0, 5:4, 09:08, 13:12, 17:16, 21:20, 25:24, 29:28 = Value for positive edge DQ and turn by sequence. 3:2, 7:6, 11:10, 15:14, 19:18, 23:22, 27:26, 31:30 = Value for negative edge DQ and turn by sequence. The values are the same for each 2 bits DQ. **When Data Generation Mode = 111** 0, 2, 4, 6, 8, 10, 12, 14 = Value for positive edge DQ and turn by sequence. 1, 3, 5, 7, 9, 11, 13, 15 = Value for negative edge DQ and turn by sequence. The values are the same for each 1-bit DQ.

Offset: 80h	MCR80: Enable Request Input Control Register	Init = 0

Bit	R/W	Description
31:0	RW	**Enable requests control bit** Bit [0]: Enable REQ0 input Bit [1]: Enable REQ1 input ....

Offset: 84h	MCR84: High Priority Request Setting Register	Init = 0

Bit	R/W	Description
n	RW	**Enable REQ(n) as high priority**
...	RW	
1	RW	**Enable REQ(1) as high priority**
0	RW	**Enable REQ(0) as high priority**

Offset: 88h	MCR88: DRAM Controller Software Release Date	Init = 0

Bit	R/W	Description
31:0	RW	**Release date [YYYYMMDD]**

Offset: 8Ch	MCR8C: DRAM Controller Initial Time	Init = 0

Bit	R/W	Description
31:0	RW	**Initial time in us**

Offset: 90h	MCR90: Write Protect Region0 Lower Bound Register	Init = 0
Offset: 94h	MCR94: Write Protect Region0 Upper Bound Register	Init = 0

Bit	Attr.	Description
		MCR90: Lower Bound Register
31:12	RW	**Write Protect Region0 Lower Bound Address**
11:1	RO	**Reserved**
1	RW	**Disable Read to Write Protect Region0**
0	RW	**Enable Write Protect Region0**
		MCR94: Upper Bound Register
31:12	RW	**Write Protect Region0 Upper Bound Address**
11:0	RO	**Reserved**

Note :
The protected region is Lower_Bound $\leq$ write address $\leq$ Upper_Bound.
This register is writable only when MCRA0[0] = 0.

Offset: 98h	MCR98: Write Protect Region1 Lower Bound Register	Init = 0
Offset: 9Ch	MCR9C: Write Protect Region1 Upper Bound Register	Init = 0

Bit	Attr.	Description
		MCR98: Lower Bound Register
31:12	RW	**Write Protect Region1 Lower Bound Address**
11:2	RO	**Reserved**
1	RW	**Disable Read to Write Protect Region1**
0	RW	**Enable Write Protect Region1**
		MCR9C: Upper Bound Register
31:12	RW	**Write Protect Region1 Upper Bound Address**
11:0	RO	**Reserved**

Note :
The protected region is Lower_Bound $\leq$ write address $\leq$ Upper_Bound.
This register is writable only when MCRA0[1] = 0.

Offset: A0h	MCRA0: Write Protect Registers Lock Control Register	Init = 0

Bit	R/W	Description
31:3	RO	**Reserved**
2	RW	**Disable DRAM read to oversized address** 0: read to oversized region would return aliased lower address data 1: read to oversized region would return 0
1	RW	**Lock MCR98/MCR9C/MCRD0/MCRD4**
0	RW	**Lock MCR90/MCR94/MCRD8/MCRDC**

to next page

from previous page

Note :
MCR90 ~ MCRA0 and MCRD0 ~ MCRDC can only be written by ARM CPU.
When enable SPI auxiliary pins configurable mode (SCU500[31]=1), then the write is controlled by external pin FWSPIWP#.
When external pin FWSPIWP# = 0, then MCRA0 is write '1' only.
When external pin FWSPIWP# = 1, then MCRA0 can be write 1 and 0.
When disable SPI auxiliary pin configurable mode (SCU500[31]=0), then this register is write '1' possible only.

Offset: B0h	MCRB0: ECC Test Control Register	Init = 0
Bit	**R/W**	**Description**
31:16	RO	**Reserved**
15	RW	**Enable ECC 2nd bit error injection**
14:8	RW	**ECC 2nd bit error inject bit position**
7	RW	**Enable ECC 1st bit error injection**
6:0	RW	**ECC 1st bit error inject bit position** Memory write is 128bits unit. This register defines bit position of the ECC error injection.

Offset: B4h	MCRB4: ECC Error Bits Injected Address	Init = X
Bit	**R/W**	**Description**
30:4	RO	**ECC Error injected address record** The ECC error is injected at the first write address after fired.

Offset: D0h	MCRD0: Write Privilege Control of Pretect Region0 Register	Init = 0x0
Bit	**R/W**	**Description**
31:28	RO	**Reserved (0)**
27	RW	**Allow write from DRAM REQ27**
26	RW	**Allow write from DRAM REQ26**
25	RW	**Allow write from DRAM REQ25**
24	RW	**Allow write from DRAM REQ24**
23	RW	**Allow write from DRAM REQ23**
22	RW	**Allow write from DRAM REQ22**
21	RW	**Allow write from DRAM REQ21**
20	RW	**Allow write from DRAM REQ20**
19	RW	**Allow write from DRAM REQ19**
18	RW	**Allow write from DRAM REQ18**
17	RW	**Allow write from DRAM REQ17**
16	RW	**Allow write from DRAM REQ16**
15	RW	**Allow write from DRAM REQ15**
14	RW	**Allow write from DRAM REQ14**
13	RW	**Allow write from DRAM REQ13**
12	RW	**Allow write from DRAM REQ12**
11	RW	**Allow write from DRAM REQ11**
10	RW	**Allow write from DRAM REQ10**
9	RW	**Allow write from DRAM REQ9**

to next page

from previous page

8	RW	**Allow write from DRAM REQ8**
7	RW	**Allow write from DRAM REQ7**
6	RW	**Allow write from DRAM REQ6**
5	RW	**Allow write from DRAM REQ5**
4	RW	**Allow write from DRAM REQ4**
3	RW	**Allow write from DRAM REQ3**
2	RW	**Allow write from DRAM REQ2**
1	RW	**Allow write from DRAM REQ1**
0	RW	**Allow write from DRAM REQ0**

Offset: D4h MCRD4: Read Privilege Control of Pretect Region0 Register Init = 0x0

Bit	R/W	Description
31:28	RO	**Reserved (0)**
27	RW	**Allow read from DRAM REQ27**
26	RW	**Allow read from DRAM REQ26**
25	RW	**Allow read from DRAM REQ25**
24	RW	**Allow read from DRAM REQ24**
23	RW	**Allow read from DRAM REQ23**
22	RW	**Allow read from DRAM REQ22**
21	RW	**Allow read from DRAM REQ21**
20	RW	**Allow read from DRAM REQ20**
19	RW	**Allow read from DRAM REQ19**
18	RW	**Allow read from DRAM REQ18**
17	RW	**Allow read from DRAM REQ17**
16	RW	**Allow read from DRAM REQ16**
15	RW	**Allow read from DRAM REQ15**
14	RW	**Allow read from DRAM REQ14**
13	RW	**Allow read from DRAM REQ13**
12	RW	**Allow read from DRAM REQ12**
11	RW	**Allow read from DRAM REQ11**
10	RW	**Allow read from DRAM REQ10**
9	RW	**Allow read from DRAM REQ9**
8	RW	**Allow read from DRAM REQ8**
7	RW	**Allow read from DRAM REQ7**
6	RW	**Allow read from DRAM REQ6**
5	RW	**Allow read from DRAM REQ5**
4	RW	**Allow read from DRAM REQ4**
3	RW	**Allow read from DRAM REQ3**
2	RW	**Allow read from DRAM REQ2**
1	RW	**Allow read from DRAM REQ1**
0	RW	**Allow read from DRAM REQ0**

Offset: D8h	MCRD8: Write Privilege Control of Pretect Region1 Register	Init = 0x0
Bit	**R/W**	**Description**
31:28	RO	**Reserved (0)**
27	RW	**Allow write from DRAM REQ27**
26	RW	**Allow write from DRAM REQ26**
25	RW	**Allow write from DRAM REQ25**
24	RW	**Allow write from DRAM REQ24**
23	RW	**Allow write from DRAM REQ23**
22	RW	**Allow write from DRAM REQ22**
21	RW	**Allow write from DRAM REQ21**
20	RW	**Allow write from DRAM REQ20**
19	RW	**Allow write from DRAM REQ19**
18	RW	**Allow write from DRAM REQ18**
17	RW	**Allow write from DRAM REQ17**
16	RW	**Allow write from DRAM REQ16**
15	RW	**Allow write from DRAM REQ15**
14	RW	**Allow write from DRAM REQ14**
13	RW	**Allow write from DRAM REQ13**
12	RW	**Allow write from DRAM REQ12**
11	RW	**Allow write from DRAM REQ11**
10	RW	**Allow write from DRAM REQ10**
9	RW	**Allow write from DRAM REQ9**
8	RW	**Allow write from DRAM REQ8**
7	RW	**Allow write from DRAM REQ7**
6	RW	**Allow write from DRAM REQ6**
5	RW	**Allow write from DRAM REQ5**
4	RW	**Allow write from DRAM REQ4**
3	RW	**Allow write from DRAM REQ3**
2	RW	**Allow write from DRAM REQ2**
1	RW	**Allow write from DRAM REQ1**
0	RW	**Allow write from DRAM REQ0**

Offset: DCh	MCRDC: Read Privilege Control of Pretect Region1 Register	Init = 0x0
Bit	**R/W**	**Description**
31:28	RO	**Reserved (0)**
27	RW	**Allow read from DRAM REQ27**
26	RW	**Allow read from DRAM REQ26**
25	RW	**Allow read from DRAM REQ25**
24	RW	**Allow read from DRAM REQ24**
23	RW	**Allow read from DRAM REQ23**
22	RW	**Allow read from DRAM REQ22**
21	RW	**Allow read from DRAM REQ21**
20	RW	**Allow read from DRAM REQ20**

to next page

from previous page

19	RW	**Allow read from DRAM REQ19**
18	RW	**Allow read from DRAM REQ18**
17	RW	**Allow read from DRAM REQ17**
16	RW	**Allow read from DRAM REQ16**
15	RW	**Allow read from DRAM REQ15**
14	RW	**Allow read from DRAM REQ14**
13	RW	**Allow read from DRAM REQ13**
12	RW	**Allow read from DRAM REQ12**
11	RW	**Allow read from DRAM REQ11**
10	RW	**Allow read from DRAM REQ10**
9	RW	**Allow read from DRAM REQ9**
8	RW	**Allow read from DRAM REQ8**
7	RW	**Allow read from DRAM REQ7**
6	RW	**Allow read from DRAM REQ6**
5	RW	**Allow read from DRAM REQ5**
4	RW	**Allow read from DRAM REQ4**
3	RW	**Allow read from DRAM REQ3**
2	RW	**Allow read from DRAM REQ2**
1	RW	**Allow read from DRAM REQ1**
0	RW	**Allow read from DRAM REQ0**

23.4 Self Refresh Command Sequence

23.4.1 Enter Self Refresh

1. Disable all IP working and swap ARM code area to static flash memory
2. Set MCR34 bit[2:1] = ”10” and all other bits keep original value, bit[3] is an option for more power saving

23.4.2 Exit Self Refresh

1. Set MCR34 bit[2] = 0 and all other bits keep original value
2. Reset DRAM DLL. Set MCR2C bit[8] = 1 and all other bits keep old value then set MCR28 = 1

23.5 Region Protect

- There are 14 set register of region control.
  - 2 regions are read/write control.
  - 12 regions are access control.
- For read/write control regions:
  1. Set MCR90 and MCR94 to a space under protect.
  2. Set MCRD0 and MCRD4 for read and/or write control.
  3. Set MCR90[1] if this region is not readable.
  4. Set MCR90[0] to enable this region protection.

  RO Example:

1. Set MCR90 to 0x80000000 and MCR94 to 0x8001F000 for 0x80000000 to 0x8001FFFF.
2. Set MCRD0 to 0x0 and MCRD4 to 0x3020 to allow CA7 and CM3 to read-only.
3. Set MCR90[0] to enable protection.

WO Example:

1. Set MCR90 to 0x80000000 and MCR94 to 0x8001F000 for 0x80000000 to 0x8001FFFF.
2. Set MCRD0 to 0x3020 and MCRD4 to 0x0 to allow CA7 and CM3 to write-only.
3. Set MCR90[1] to eanble read control.
4. Set MCR90[0] to enable protection.

– For access control regions:

1. Set AHBC300 and AHBC304 to a space under protect.
2. Set AHBC308 for access control.
3. Set AHBC300[0] to enable this region protection.

Example:

1. Set AHBC300 to 0x80000000 and AHBC304 to 0x8001F000 for 0x80000000 to 0x8001FFFF.
2. Set AHBC308 0x3020 to allow CA7 and CM3 to access.
3. Set AHBC300[0] to enable protection.

24 System Control Unit (SCU)

24.1 Overview

System Control Unit (SCU) implements chip level control registers, which is listed below, to control the various functions supported by AST2600 . Each register has its own specific offset value to derive its physical address location.

Base address of SMC = 0x1E6E_2000
Physical address = (Base address of SCU Controller) + Offset

SCU000: Protection Key Register
SCU010: Protection Key Register 2
SCU004: Silicon Revision ID Register
SCU014: Silicon Revision ID Register 2

SCU040: Module Reset Control Register Set 1
SCU044: Module Reset Control Clear Register 1
SCU050: Module Reset Control Register Set 2
SCU054: Module Reset Control Clear Register 2
SCU060: EXTRST# Reset Selection Set 1
SCU064: System Reset Status Register Set 1-1 (non-key protected)
SCU068: System Reset Status Register Set 1-2 (non-key protected)
SCU06C: System Reset Status Register Set 1-3 (non-key protected)
SCU070: EXTRST# Reset Selection Set 2
SCU074: System Reset Status Register Set 2-1 (non-key protected)
SCU078: System Reset Status Register Set 2-2 (non-key protected)
SCU07C: System Reset Status Register Set 2-3 (non-key protected)
SCU080: Clock Stop Control Register Set 1
SCU090: Clock Stop Control Register Set 2
SCU0C0: Misc. 1 Control Register
SCU0C4: Misc. 2 Control Register
SCU0C8: Debug Control Register
SCU0D0: Misc. 3 Control Register
SCU0D4: Misc. 4 Control Register
SCU0D8: Debug Control Register 2
SCU0FC: DAC CRC Control Register

SCU100: SOC Scratch Register
SCU104: SOC Scratch Register
SCU108: SOC Scratch Register
SCU10C: SOC Scratch Register
SCU180: CPU Scratch Register. Reserved for SMP booting
SCU184: CPU Scratch Register. Reserved for SMP booting
SCU188: CPU Scratch Register. Reserved for SMP booting
SCU18C: CPU Scratch Register. Reserved for SMP booting
SCU190: CPU Scratch Register. Reserved for SMP booting
SCU194: CPU Scratch Register. Reserved for SMP booting
SCU198: CPU Scratch Register. Reserved for SMP booting
SCU19C: CPU Scratch Register. Reserved for SMP booting
SCU1A0: CPU Scratch Register. Reserved for SMP booting
SCU1A4: CPU Scratch Register. Reserved for SMP booting
SCU1A8: CPU Scratch Register. Reserved for SMP booting
SCU1AC: CPU Scratch Register. Reserved for SMP booting
SCU1B0: CPU Scratch Register. Available for SW use

SCU1B4: CPU Scratch Register. Available for SW use
SCU1B8: CPU Scratch Register. Available for SW use
SCU1BC: CPU Scratch Register. Available for SW use
SCU1C0: CPU Scratch Register. Available for SW use
SCU1C4: CPU Scratch Register. Available for SW use
SCU1C8: CPU Scratch Register. Available for SW use
SCU1CC: CPU Scratch Register. Available for SW use
SCU1D0: CPU Scratch Register. Available for SW use
SCU1D4: CPU Scratch Register. Available for SW use
SCU1D8: CPU Scratch Register. Available for SW use
SCU1DC: CPU Scratch Register. Available for SW use
SCU1E0: CPU Scratch Register. Available for SW use
SCU1E4: CPU Scratch Register. Available for SW use
SCU1E8: CPU Scratch Register. Available for SW use
SCU1EC: CPU Scratch Register. Available for SW use
SCU1F0: CPU Scratch Register. Available for SW use
SCU1F4: CPU Scratch Register. Available for SW use
SCU1F8: CPU Scratch Register. Available for SW use
SCU1FC: CPU Scratch Register. Available for SW use

SCU200: H-PLL Parameter register
SCU204: H-PLL Extend Parameter register (non-key protected)
SCU210: A-PLL Parameter register
SCU214: A-PLL Extend Parameter register (non-key protected)
SCU220: M-PLL Parameter register
SCU224: M-PLL Extend registers (non-key protected)
SCU240: E-PLL Parameter register
SCU244: E-PLL Extend registers (non-key protected)
SCU260: D-PLL Parameter register
SCU264: D-PLL Extend registers (non-key protected)
SCU300: Clock Source Selection Register
SCU304: Clock Source Selection Register Set 2
SCU308: Clock Source Selection Register Set 3
SCU310: Clock Source Selection Register Set 4
SCU314: Clock Source Selection Register Set 5
SCU320: Frequency Counter Control Register
SCU324: Frequency Counter Comparison Range
SCU32C: Reserve
SCU330: Frequency Counter Control Register 2 (non-key protected)
SCU334: Frequency Counter Comparison Range 2 (non-key protected)
SCU338: Generate UARTCLK from UXCLK (non-key protected)
SCU33C: Generate HUARTCLK from HUXCLK (non-key protected)
SCU340: MAC 12 Interface Clock Delay Setting
SCU348: MAC 12 Interface Clock Delay 100M Setting
SCU34C: MAC 12 Interface Clock Delay 10M Setting
SCU350: MAC 34 Interface Clock Delay Setting
SCU358: MAC 34 Interface Clock Delay 100M Setting
SCU35C: MAC 34 Interface Clock Delay 10M Setting
SCU360: Clock Duty Measurement Control
SCU364: Clock Duty Selection (non-key protected)
SCU368: Clock Duty Selection (non-key protected)
SCU36C: Clock Duty Measurement Result
SCU370: Clock Duty Measurement Control 2
SCU374: Clock Duty Selection 2 (non-key protected)
SCU37C: Clock Duty Measurement Result 2
SCU400: Multi-function Pin Control #1

SCU404: Multi-function Pin Control #2
SCU40C: Multi-function Pin Control #3
SCU410: Multi-function Pin Control #4
SCU414: Multi-function Pin Control #5
SCU418: Multi-function Pin Control #6
SCU41C: Multi-function Pin Control #7
SCU430: Multi-function Pin Control #8
SCU434: Multi-function Pin Control #9
SCU438: Multi-function Pin Control #10
SCU440: USB Multi-function Control
SCU450: Multi-function Pin Control #14
SCU454: Multi-function Pin Control #15
SCU458: Multi-function Pin Control #16
SCU470: Multi-function Pin Control
SCU4B0: Multi-function Pin Control #17
SCU4B4: Multi-function Pin Control #18
SCU4B8: Multi-function Pin Control #19
SCU4BC: Multi-function Pin Control #20
SCU4D4: Multi-function Pin Control #22
SCU4D8: Multi-function Pin Control #23
SCU690: Multi-function Pin Control #24
SCU694: Multi-function Pin Control #25
SCU698: Multi-function Pin Control #26
SCU69C: Multi-function Pin Control #27
SCU6B8: Multi-function Pin Control #28
SCU6D0: Multi-function Pin Control #29

SCU4F4: UART Debug interface Baud Rate Control
SCU500: Hardware Strap1 Register
SCU504: Hardware Strap1 Clear Register
SCU508: Hardware Strap1 Protection Register
SCU510: Hardware Strap2 Register
SCU514: Hardware Strap2 Clear Register
SCU518: Hardware Strap2 Protection Register
SCU51C: Hardware Strap3 Register
SCU520: Random Number Generator Control
SCU524: Random Number Generator Data Output
SCU530: Random Number Generator 2 Control
SCU534: Random Number Generator 2 Data Output (non-key protected)
SCU544: Power Saving Wakeup Control Register (Reserved)
SCU550: Power Saving Wakeup Enable Register 2 (Reserved) (non-key protected)
SCU560: Interrupt Control and Status Register (non-key protected)
SCU564: Interrupt Control and Status Register 1
SCU570: Interrupt Control and Status Register 2 (non-key protected)
SCU590: OTP Control (non-key protected)
SCU594: Hardware Configuration Register
SCU5B0: Chip Unique ID 0
SCU5B4: Chip Unique ID 1
SCU5B8: Reserved Read Only ID 0
SCU5BC: Reserved Read Only ID 1
SCU5D0: Reserved Read Only ID 2
SCU5D4: Reserved Read Only ID 3
SCU610: Disable GPIO Internal Pull-Down #0
SCU614: Disable GPIO Internal Pull-Down #1
SCU618: Disable GPIO Internal Pull-Down #2
SCU61C: Disable GPIO Internal Pull-Down #3

SCU630: Disable GPIO Internal Pull-Down #4
SCU634: Disable GPIO Internal Pull-Down #5
SCU638: Disable GPIO Internal Pull-Down #6
SCU650: SLI Driving Strength

SCU800: CA7 processor Control
SCU804: CA7 processor Control
SCU808: CA7 processor Control
SCU80C: CA7 processor Control

SCU820: CA7 processor parity check control (non-key protected)
SCU824: CA7 processor parity clear (non-key protected)
SCU830: Watchdog Controller Protection Control 1
SCU834: Security Protection Control
SCU850: Watchdog Controller Protection Control 2
SCU854: Watchdog Controller Protection Control 3
SCU858: Watchdog Controller Protection Control 4
SCU85C: Watchdog Controller Protection Control 5
SCU870: Watchdog Controller Protection Control 6
SCU874: Watchdog Controller Protection Control 7
SCU878: Watchdog Controller Protection Control 8
SCU87C: Watchdog Controller Protection Control 9
SCUA00: CM3 Coprocessor Control Register (non-key protected)
SCUA04: CM3 Memory Base Address Register (non-key protected)
SCUA08: CM3 Instruction Memory Address Limit Register (non-key protected)
SCUA0C: CM3 Data Memory Address Limit Register (non-key protected)
SCUA40: CM3 Cacheable Area Declaration Register (non-key protected)
SCUA44: CM3 Cache Invalidation Control Register (non-key protected)
SCUA48: CM3 Cache Function Control Register (non-key protected)

SCUC00: PCI Configuration Setting Register #1
SCUC04: PCI Configuration Setting Register #2
SCUC08: PCI Configuration Setting Register #3
SCUC20: PCI Express Configuration Setting Contol Register (non-key protected)
SCUC24: BMC MMIO Decode Setting Register (non-key protected)
SCUC28: First relocated controller decode area location (non-key protected)
SCUC2C: Second relocated controller decode area location (non-key protected)
SCUC40: Mailbox decode area location (non-key protected)
SCUC44: Shared memory area decode location 1 (non-key protected)
SCUC48: Shared memory area decode location 2 (non-key protected)
SCUC60: BMC Device ID
SCUC80: EHCI Device ID
SCUE00: VGA Scratch Register #1
SCUE04: VGA Scratch Register #2
SCUE08: VGA Scratch Register #3
SCUE0C: VGA Scratch Register #4
SCUE20: VGA Scratch Register #5
SCUE24: VGA Scratch Register #6
SCUE28: VGA Scratch Register #7
SCUE2C: VGA Scratch Register #8

SCUF00: Write Protect Register #1
SCUF04: Write Protect Register #2
SCUF08: Write Protect Register #3
SCUF0C: Write Protect Register #4
SCUF10: Write Protect Register #5

SCUF18: Write Protect Register #6
SCUF1C: Write Protect Register #7
SCUF20: Write Protect Register #8
SCUF24: Write Protect Register #9
SCUF30: Write Protect Register #10
SCUF34: Write Protect Register #11
SCUF38: Write Protect Register #12
SCUF3C: Write Protect Register #13
SCUF40: Write Protect Register #14
SCUF48: Write Protect Register #15
SCUF50: Write Protect Register #16
SCUF60: Write Protect Register #17
SCUF7C: Write Protect Register #18
SCUF80: Reset Source Control Register #1
SCUF84: Reset Source Control Register #2
SCUF88: Reset Source Control Register #3
SCUF8C: Reset Source Control Register #4
SCUF90: Reset Source Control Register #5
SCUF98: Reset Source Control Register #6
SCUF9C: Reset Source Control Register #7
SCUFA0: Reset Source Control Register #8
SCUFA4: Reset Source Control Register #9
SCUFB0: Reset Source Control Register #10
SCUFB4: Reset Source Control Register #11
SCUFB8: Reset Source Control Register #12
SCUFC0: Reset Source Control Register #13
SCUFC8: Reset Source Control Register #14
SCUFD0: Reset Source Control Register #18
SCUFE0: Reset Source Control Register #15
SCUFEC: Reset Source Control Register #16
SCUFFC: Reset Source Control Register #17

Changing SCU registers usually results in significant impact on SOC operations. Therefore, all these registers have to be well protected.

24.2 Registers : Base Address = 0x1E6E:2000

Offset: 000h	SCU000: Protection Key Register	Init = 0
Offset: 010h	**SCU010: Protection Key Register 2**	**Init = 0**

Bit	Attr.	Reset	Description
31:0	RW	Rst67	**Protection Key** This register is designed to protect SCU registers from unpredictable updates, especially when ARM CPU is out of control. The password of the protection key is **0x1688A8A8**. **Unlock SCU registers**: Write 0x1688A8A8 to this register **Lock SCU registers**: Write others value to this register **Only firmware can lock the SCU registers, other softwares (ex. system BIOS/driver) can not do this to prevent disturbing the operation of firmware.** When this register is unlocked, the read back value of this register is 0x00000001. When this register is locked, the read back value of this register is 0x00000000.

Offset: 004h	SCU004: Silicon Revision ID Register	Init = -
Offset: 014h	**SCU014: Silicon Revision ID Register**	**Init = -**

Bit	Attr.	Reset	Description
31:24	RO	-	**Reserved (0x05)**
23:16	RO	-	**Hardware Revision ID** Combined of SCU004[23:16] and SCU014[23:16] represent silicon revision
15:8	RO	-	**Chip bounding option** The read back value of this register will reflect the status of the chip bonding option which is designed for product differentiation.
7:0	RO	-	**Reserved for backward compatible**

The following table shows a list of silicon revision ID. The register offset is different from previous generation ASPEED BMC. The corresponding register for previous generation ASPEED BMC is SCU7C.

	SCU004	SCU014
AST1100-A0	0x00000200	NA
AST1100-A1	0x00000201	NA
AST1100-A2	0x00000202	NA
AST1100-A3	0x00000202	NA
AST2050-A0	0x00000200	NA
AST2050-A1	0x00000201	NA
AST2050-A2	0x00000202	NA
AST2050-A3	0x00000202	NA
AST2100-A0	0x00000300	NA
AST2100-A1	0x00000301	NA
AST2100-A2	0x00000302	NA
AST2100-A3	0x00000302	NA
AST2150-A0	0x00000202	NA
AST2150-A1	0x00000202	NA
AST2200-A0	0x00000102	NA
AST2200-A1	0x00000102	NA
AST2300-A0	0x01000003	NA
AST2300-A1	0x01010303	NA
AST1300-A1	0x01010003	NA
AST1050-A1	0x01010203	NA
AST2400-A0	0x02000303	NA
AST2400-A1	0x02010303	NA
AST1400-A1	0x02010103	NA
AST1250-A1	0x02010303	NA
AST2500-A0	0x04000303	NA
AST2510-A0	0x04000103	NA
AST2520-A0	0x04000203	NA
AST2530-A0	0x04000403	NA
AST2500-A1	0x04010303	NA
AST2510-A1	0x04010103	NA
AST2520-A1	0x04010203	NA
AST2530-A1	0x04010403	NA
AST2500-A2	0x04030303	NA
AST2510-A2	0x04030103	NA
AST2520-A2	0x04030203	NA
AST2530-A2	0x04030403	NA
AST2600-A0	0x05000303	0x05000303
AST2600-A1	0x05010303	0x05010303
AST2600-A2	0x05010303	0x05020303
AST2600-A3	0x05030303	0x05030303
AST2620-A1	0x05010203	0x05010203
AST2620-A2	0x05010203	0x05020203
AST2620-A3	0x05030203	0x05030203

Offset: 040h	SCU040: System Reset Control Register	Init = 0xF7C3FED8

Bit	R/W	Reset	Description
31	RW1S	Rst68	**Reset AHB to PCIe bus bridge (H2X) controller** 0: No operation 1: Reset AHB to PCIe bus bridge (H2X) controller (asynchronous reset) (default)
30	RW1S	Rst68	**Reset general purpose MCU (GPMCU) controller** 0: No operation 1: Reset general purpose MCU (GPMCU) controller (asynchronous reset) (default)
29	RW1S	Rst68	**Reset DisplayPort MCU (DPMCU) controller** 0: No operation 1: Reset DisplayPort MCU (DPMCU) controller (asynchronous reset) (default)
28	RW1S	Rst68	**Reset DisplayPort controller** 0: No operation 1: Reset DisplayPort controller (asynchronous reset) (default)
27	RW1S	Rst68	**Reset PCI-E Root Complex X-DMA controller** 0: No operation 1: Reset PCI-E Root Complex X-DMA controller (asynchronous reset) (default)
26	RW1S	Rst68	**Graphics controller** 0: No operation 1: Reset Graphics controller (asynchronous reset) (default)
25	RW1S	Rst68	**Reset X-DMA controller** 0: No operation 1: Reset X-DMA controller (asynchronous reset) (default)
24	RW1S	Rst68	**Reset MCTP controller** 0: No operation 1: Reset MCTP controller (asynchronous reset) (default)
23	RW1S	Rst68	**Reset PCI-E Root Complex MCTP controller** 0: No operation 1: Reset PCI-E Root Complex MCTP controller (asynchronous reset) (default)
22	RW1S	Rst68	**Reset JTAG Master controller** 0: No operation 1: Reset JTAG Master controller (asynchronous reset) (default) The reset control bit also control the JTAG interface mode. 0: Enable JTAG master mode, JTAG slave was reset 1: Enable JTAG slave mode (ARM ICE debugger) **Firmware must enable JTAG master mode when Mass Production, for solving ARM JTAG reset incomplete issue.**
21	RW1S	Rst68	**PCI Express device reset output enable** 0: Device reset (PERST#) output disable (default) 1: Device reset (PERST#) output enable
20	RW1S	Rst68	**PCI Express device reset output control** 0: Device reset (PERST#) output high (default) 1: Device reset (PERST#) output low
19	RW1S	Rst68	**PCI Express Root complex reset (SSPRST#) output enable** 0: Root complex reset output disable (default) 1: Root complex reset output enable This bit is reset inactive by default and is used only when SCU500[24]=1. BMC FW need to set this bit high when it wants to assert reset to pin SSPRST#.

to next page

from previous page

18	RW1S	Rst68	**PCI Express Root complex reset (SSPRST#) output control** 0: Root complex reset output high (default) 1: Root complex reset output low This bit is reset inactive by default. BMC FW has to set this bit high to reset internal PCI Express Root complex controller as AC power-on. **This SCU040[18] setting is required to for the A3 silicon to boot.**
17	RW1S	Rst68	**Reserved, must keep at value "1"**
16	RW1S	Rst68	**Reset eMMC controller** 0: No operation 1: Reset eMMC controller (asynchronous reset) (default)
15	RW1S	Rst68	**Reset USB1.1 Host controller** 0: No operation 1: Reset USB1.1 Host controller (asynchronous reset) (default)
14	RW1S	Rst68	**Reset USB2.0 Hub/Host controller** 0: No operation 1: Reset USB2.0 controller (asynchronous reset) (default) USB2.0 Host and Hub controller shared the same port, so only one can work at a time, which is determined by SCU440[25:24]. This reset bit affects both Host and Hub, and the controller not selected by SCU440[25:24] is always stayed at reset state.
13	RW1S	Rst68	**Reset CRT controller** 0: No operation 1: Reset CRT controller (asynchronous reset) (default)
12	RW1S	Rst68	**Reset MAC#2 controller** 0: No operation 1: Reset MAC#2 controller (asynchronous reset) (default)
11	RW1S	Rst68	**Reset MAC#1 controller** 0: No operation 1: Reset MAC#1 controller (asynchronous reset) (default)
10	RW1S	Rst68	**Reserved, must keep at value "1"**
9	RW1S	Rst68	**Reset RVAS engine** 0: No operation 1: Reset RVAS engine (default)
8	RW1S	Rst68	**Disable PCI bus controller and VGA controller** 0: No operation (default) 1: Disable PCI bus controller and VGA controller
7	RW1S	Rst68	**Reset 2D engine** 0: No operation 1: Reset 2D engine (default) This register is valid only at CRT Mode (SCU2C[7]).
6	RW1S	Rst68	**Reset Video engine** 0: No operation 1: Reset Video engine (asynchronous reset) (default)
5	RW1S	Rst68	**Enable PCI reset DisplayPort controller** 0: No operation (default) 1: Enable DisplayPort controller reset source from PERST#
4	RW1S	Rst68	**Reset HAC engine** 0: No operation 1: Reset HAC engine (asynchronous reset) (default)

to next page

from previous page

3	RW1S	Rst68	**Reset USB1.1 HID/USB2.0 Host2~~/USB2.0 Device~~ controller** 0: No operation 1: Reset USB1.1 HID/USB2.0 Host2~~/USB2.0 Device~~ controller (asynchronous reset) (default) USB1.1 HID, USB2.0 Host2 ~~and USB2.0 Device~~ controller shared the same port, so only one can work at a time, which is determined by SCU440[29:28]. This reset bit affects all 3 controllers, and the controllers not selected by SCU440[29:28] are always stayed at reset state.
2	RW1S	Rst68	**Reserved (0)**
1	RW1S	Rst68	**Reserved (0)**
0	RW1S	Rst68	**Enable reset SDRAM controller when full chip Watchdog reset occur** 0: No operation (default) 1: Enable reset SDRAM controller (asynchronous reset) This bit enables the reset of SDRAM controller when Watchdog SOC reset occur and SRAM controller Watchdog reset mask disabled (WDT1C[1]=1). Because reset DRAM controller will cause all DRAM operation failed, including VGA function. So it must be carefully to control the reset of SDRAM controller.
Note : SCU040[i] can be set by write SCU040[i]=1 and be cleared by write SCU044[i]=1			

Offset: 044h	**SCU044: System Reset Control Clear Register 2**		**Init = 0**
Bit	**R/W**	**Reset**	**Description**
31:0	W1C	-	**SCU040 System Reset Control Clear Register**

Offset: 050h	**SCU050: System Reset Control Register Set 2**		**Init = 0x0DFFFFFC**
Bit	**R/W**	**Reset**	**Description**
31:28	RW1S	Rst68	**Reserved, must keep at value "0000"**
27	RW1S	Rst68	**Reset FSI controller** 0: No operation 1: Reset FSI controller (asynchronous reset) (default)
26	RW1S	Rst68	**Reserved, must keep at value "1"**
25	RW1S	Rst68	**Reset eSPI controller** 0: No operation (default) 1: Reset eSPI controller (asynchronous reset)
24	RW1S	Rst68	**Reset SD controller** 0: No operation 1: Reset eMMC controller (asynchronous reset) (default)
23	RW1S	Rst68	**Reset ADC controller** 0: No operation 1: Reset ADC controller (asynchronous reset) (default)
22	RW1S	Rst68	**Reset JTAG Master 2 controller** 0: No operation 1: Reset JTAG Master 2 controller (asynchronous reset) (default) The reset control bit also control the JTAG interface mode. 0: Enable JTAG master mode, JTAG slave was reset 1: Enable JTAG slave mode (ARM ICE debugger) **Firmware must enable JTAG master mode when Mass Production, for solving ARM JTAG reset incomplete issue.**

to next page

from previous page

21	RW1S	Rst68	**Reset MAC#4 controller** 0: No operation 1: Reset MAC#4 controller (asynchronous reset) (default)
20	RW1S	Rst68	**Reset MAC#3 controller** 0: No operation 1: Reset MAC#3 controller (asynchronous reset) (default)
19:14	RW1S	Rst68	**Reserved, must keep at value ”111111”**
13	RW1S	Rst68	**Reset I3C5 controller** 0: No operation 1: Reset I3C5 controller (default)
12	RW1S	Rst68	**Reset I3C4 controller** 0: No operation 1: Reset I3C4 controller (default)
11	RW1S	Rst68	**Reset I3C3 controller** 0: No operation 1: Reset I3C3 controller (default)
10	RW1S	Rst68	**Reset I3C2 controller** 0: No operation 1: Reset I3C2 controller (default)
9	RW1S	Rst68	**Reset I3C1 controller** 0: No operation 1: Reset I3C1 controller (default)
8	RW1S	Rst68	**Reset I3C0 controller** 0: No operation 1: Reset I3C0 controller (default)
7	RW1S	Rst68	**Reset I3C REG/DMA controller** 0: No operation 1: Reset I3C DMA controller (default)
6	RW1S	Rst68	**Reserved, must keep at value ”1”**
5	RW1S	Rst68	**Reset PWM controller** 0: No operation 1: Reset PWM controller (default)
4	RW1S	Rst68	**Reset PECI controller** 0: No operation 1: Reset PECI controller (default)
3	RW1S	Rst68	**Reset MII controller** 0: No operation 1: Reset MII controller (asynchronous reset) (default)
2	RW1S	Rst68	**Reset I2C/SMBus controller** 0: No operation 1: Reset SMBus/I2C controller (asynchronous reset) (default) Writing ”1” to this register will cause all the 16 sets of SMBus/I2C controllers to be **asynchronously** reset. Actually each of the 16 controllers has its own control register to **synchronously** reset itself.
1	RW1S	Rst68	**Reserved, must keep at value ”0”**

to next page

from previous page

0	RW1S	Rst68	**Reset LPC controller** 0: No operation (default) 1: Reset LPC controller (asynchronous reset) The reset command will only be applied to the BMC controller embedded in LPC controller.
Note : SCU050[i] can be set by write SCU050[i]=1 and be cleared by write SCU054[i]=1			

Offset: 054h	**SCU054: System Reset Control Clear Register 2**		**Init = 0**
Bit	**R/W**	**Reset**	**Description**
31:0	W1C	-	**SCU050 System Reset Control Clear Register 2**

Offset: 060h	**SCU060: EXTRST# Reset Selection**		**Init = 0x070F1FF1**
Bit	**R/W**	**Reset**	**Description**
31:27	RO	-	**Reserved(0)**
26	RW	Rst69	**Enable reset AHB to PCIe bus (H2X) controller**
25	RW	Rst69	**Enable reset RVAS controller**
24	RW	Rst69	**Enable reset GPIO #1 controller**
23	RW	Rst69	**Enable reset XDMA #2 controller**
22	RW	Rst69	**Enable reset XDMA #1 controller**
21	RW	Rst69	**Enable reset MCTP #2 controller**
20	RW	Rst69	**Enable reset MCTP #1 controller**
19	RW	Rst69	**Enable reset JTAG #1 master controller**
18	RW	Rst69	**Enable reset eMMC controller**
17	RW	Rst69	**Enable reset MAC#2 controller**
16	RW	Rst69	**Enable reset MAC#1 controller**
15	RW	Rst69	**Enable reset general purpose (GP) MCU controller**
14	RW	Rst69	**Enable reset DisplayPort (DP) MCU controller**
13	RW	Rst69	**Enable reset DP controller**
12	RW	Rst69	**Enable reset HAC engine**
11	RW	Rst69	**Enable reset Video engine**
10	RW	Rst69	**Enable reset CRT mode 2D engine**
9	RW	Rst69	**Enable reset Graphics CRT controller**
8	RW	Rst69	**Enable reset USB1.1 UHCI Host controller**
7	RW	Rst69	**Enable reset USB portB Host/Dev controller**
6	RW	Rst69	**Enable reset USB portA Host/Hub controller**
5	RW	Rst69	**Enable reset Coprocessor**
4	RW	Rst69	**Enable reset SOC controller** SOC controller includes: WDT, RTC, Timer, UART, SRAM. This bit must be same as SCU70[4].
3	RW	Rst69	**Enable reset internal bridge. This bit must be same as SCU70[3].**
2	RW	Rst69	**Enable reset AHB bridges. This bit must be same as SCU70[2].**
1	RW	Rst69	**Enable reset SDRAM controller**

to next page

from previous page

0	RO	-	**Enable reset ARM. This bit is always 1.**

Note :
This register controls the IPs to be reset by the EXTRST# pin input.

Offset: 064h **SCU064: System Reset Event Log Register Set 1-1** **Init = 0x0000FF31**

Bit	R/W	Reset	Description
31	W1C	RstPwr	**WDT4 Software Reset event log**
30	W1C	RstPwr	**WDT4 ARM Reset event log**
29	W1C	RstPwr	**WDT4 Full Reset event log**
28	W1C	RstPwr	**WDT4 SOC Reset event log**
27	W1C	RstPwr	**WDT3 Software Reset event log**
26	W1C	RstPwr	**WDT3 ARM Reset event log**
25	W1C	RstPwr	**WDT3 Full Reset event log**
24	W1C	RstPwr	**WDT3 SOC Reset event**
23	W1C	RstPwr	**WDT2 Software Reset event log**
22	W1C	RstPwr	**WDT2 ARM Reset event log**
21	W1C	RstPwr	**WDT2 Full Reset event log**
20	W1C	RstPwr	**WDT2 SOC Reset event log**
19	W1C	RstPwr	**WDT1 Software Reset event log**
18	W1C	RstPwr	**WDT1 ARM Reset event log**
17	W1C	RstPwr	**WDT1 Full Reset event log**
16	W1C	RstPwr	**WDT1 SOC Reset event log**
15	W1C	Rst02	**Reset event log – SLI_HRST_N Internal Bridge Controller reset**
14	W1C	Rst01	**Reset event log – AHB_HRST_N AHB Bus Controller reset**
13	W1C	Rst03	**Reset event log – SOC_HRST_N AHB SOC reset**
12	W1C	RstARM	**Reset event log – ARMRST_N ARM CA7 CPU reset**
11	W1C	RstFull	**Reset event log – PWRST_N Power-on reset**
10	W1C	RstPwr	**Reset event log – PWRSTNinPLL PLL power-on reset**
9	W1C	RstPwr	**Reset event log – PWRSTNinTrap Power-on strap enable signal**
8	W1C	RstFull	**Reset event log – PWRSTNinR Power-on strap enable signal**
7	W1C	RstPwr	**reverved**
6	W1C	RstPwr	**Coprocessor external reset SSPRST# flag**
5	W1C	RstRC	**PCIe bus #2 reset PERST# flag**
4	W1C	RstPE	**PCIe bus #1 reset RCRST# flag**
3	W1C	RstPwr	**MMC ECC fail reset flag**
2	W1C	RstPwr	**Boot flash reset flag – ABR or AdrSwap**
1	W1C	RstPwr	**External reset EXTRST# flag**
0	W1C	RstPwr	**Power on reset SRST# flag**

Note :
All register bits are write ’1’ clear.

Offset: 068h	SCU068: System Reset Event Log Register Set 1-2		Init = 0x1FFFFFFF
Bit	**R/W**	**Reset**	**Description**
31:29	RO	-	**reverved(0)**
28	W1C	Rst34	**Reset event log – H2X_HRST_N AHB to PCIe RC bridge controller reset**
27	W1C	Rst21	**Reset event log – GPMCU_HRST_N GP MCU reset**
26	W1C	Rst20	**Reset event log – DPMCU_HRST_N DP MCU reset**
25	W1C	Rst19	**Reset event log – DP_PRST_N DisplayPort controller reset**
24	W1C	Rst30	**Reset event log – XDMA8PRST_N PCIe RC XDMA reset**
23	W1C	Rst29	**Reset event log – XDMA_PRST_N PCIe device XDMA reset**
22	W1C	Rst32	**Reset event log – RVAS_HRST_N RVAS engine reset**
21	W1C	Rst17	**Reset event log – VCE_HRST_N Video engine reset**
20	W1C	Rst10	**Reset event log – UD2_HRST_N USB2.0 device controller reset**
19	W1C	Rst08	**Reset event log – UB2B_HRST_N reset**
18	W1C	Rst05	**Reset event log – UB2_HRST_N USB2.0 Hub reset**
17	W1C	Rst09	**Reset event log – UB1_PRST_N USB1.1 HID controller reset**
16	W1C	Rst12	**Reset event log – UB11H_HRST_N USB1.1 UHCI Host reset**
15	W1C	Rst25	**Reset event log – SDC_HRST_N eMMC Controller reset**
14	W1C	Rst33	**Reset event log – MSI_BRST_N reset**
13	W1C	Rst35	**Reset event log – P2A_HRST_N reset**
12	W1C	Rst28	**Reset event log – MCTP8_PRST_N PCIe RC MCTP reset**
11	W1C	Rst27	**Reset event log – MCTP_PRST_N PCIe device MCTP reset**
10	W1C	Rst23	**Reset event log – MAC1_HRST_N Ethernet MAC #1 reset**
9	W1C	Rst22	**Reset event log – MAC0_HRST_N Ethernet MAC #0 reset**
8	W1C	Rst37	**Reset event log – UARTDB_RST_N UART DMA reset**
7	W1C	Rst36	**Reset event log – UART1_RST_N UART #1 reset**
6	W1C	Rst16	**Reset event log – G2D_HRST_N 2D engine reset**
5	W1C	Rst14	**Reset event log – GFX_PRST_N SOC display controller reset**
4	W1C	Rst18	**Reset event log – AES_PRST_N Hash & crypto engine reset**
3	W1C	Rst04	**Reset event log – CM3_HRST_N ARM CM3 reset**
2	W1C	Rst00	**Reset event log – MMC_PRST_N DRAM controller reset**
1	W1C	Rst26	**Reset event log – JTAG_PRST_N JTAG master reset**
0	W1C	Rst31	**Reset event log – GPIO_PRST_N 1.8V GPIO reset**

Note :
All register bits are write '1' clear.

Offset: 06Ch	SCU06C: System Reset Event Log Register Set 1-3		Init = 0
Bit	**R/W**	**Reset**	**Description**
31:16	RO	-	**reverved(0)**
15	W1C	RstPwr	**WDT8 Software Reset event log**
14	W1C	RstPwr	**WDT8 ARM Reset event log**
13	W1C	RstPwr	**WDT8 Full Reset event log**
12	W1C	RstPwr	**WDT8 SOC Reset event log**
11	W1C	RstPwr	**WDT7 Software Reset event log**

to next page

from previous page

10	W1C	RstPwr	**WDT7 ARM Reset event log**
9	W1C	RstPwr	**WDT7 Full Reset event log**
8	W1C	RstPwr	**WDT7 SOC Reset event**
7	W1C	RstPwr	**WDT6 Software Reset event log**
6	W1C	RstPwr	**WDT6 ARM Reset event log**
5	W1C	RstPwr	**WDT6 Full Reset event log**
4	W1C	RstPwr	**WDT6 SOC Reset event log**
3	W1C	RstPwr	**WDT5 Software Reset event log**
2	W1C	RstPwr	**WDT5 ARM Reset event log**
1	W1C	RstPwr	**WDT5 Full Reset event log**
0	W1C	RstPwr	**WDT5 SOC Reset event log**
Note : All register bits are write '1' clear.			

Offset: 070h	**SCU070: EXTRST# Reset Selection**		**Init = 0x3FFFFF1**
Bit	**R/W**	**Reset**	**Description**
31:27	RO	-	**Reserved(0)**
26	RW	Rst70	**Enable reset eSPI controller**
25:24	RW	Rst70	**Reserved(0)**
23	RW	Rst70	**Enable reset I3C bus6 controller**
22	RW	Rst70	**Enable reset I3C bus5 controller**
21	RW	Rst70	**Enable reset I3C bus4 controller**
20	RW	Rst70	**Enable reset I3C bus3 controller**
19	RW	Rst70	**Enable reset I3C bus2 controller**
18	RW	Rst70	**Enable reset I3C bus1 controller**
17	RW	Rst70	**Enable reset I3C Global controller**
16	RW	Rst70	**Enable reset I2C controller**
15	RW	Rst70	**Enable reset FSI controller**
14	RW	Rst70	**Enable reset ADC controller**
13	RW	Rst70	**Enable reset PWM controller**
12	RW	Rst70	**Enable reset PECI controller**
11	RW	Rst70	**Enable reset LPC controller**
10	RW	Rst70	**Enable reset MDC/MDIO controller**
9	RW	Rst70	**Enable reset GPIO #2 controller**
8	RW	Rst70	**Enable reset JTAG #2 master controller**
7	RW	Rst70	**Enable reset SD/SDIO controller**
6	RW	Rst70	**Enable reset MAC#4 controller**
5	RW	Rst70	**Enable reset MAC#3 controller**
4	RW	Rst70	**Enable reset SOC controller** SOC controller includes: WDT, UART, BSRAM. This bit must be same as SCU60[4].
3	RW	Rst70	**Enable reset internal bridge. This bit must be same as SCU60[3].**
2	RW	Rst70	**Enable reset AHB2 bridges. This bit must be same as SCU60[2].**

to next page

from previous page

1	RW	Rst70	**Enable reset SPI1/SPI2 controller**
0	RO	-	**Enable reset ARM SPI/Secure Controller. This bit is always 1.**

Note :
This register controls the IPs to be reset by the EXTRST# pin input.

Offset: 074h **SCU074: System Reset Event Log Register Set 2-1** **Init = 0x0000FF31**

Bit	R/W	Reset	Description
31	W1C	RstPwr	**WDT4 Software Reset event log**
30	W1C	RstPwr	**WDT4 ARM Reset event log**
29	W1C	RstPwr	**WDT4 Full Reset event log**
28	W1C	RstPwr	**WDT4 SOC Reset event log**
27	W1C	RstPwr	**WDT3 Software Reset event log**
26	W1C	RstPwr	**WDT3 ARM Reset event log**
25	W1C	RstPwr	**WDT3 Full Reset event log**
24	W1C	RstPwr	**WDT3 SOC Reset event log**
23	W1C	RstPwr	**WDT2 Software Reset event log**
22	W1C	RstPwr	**WDT2 ARM Reset event log**
21	W1C	RstPwr	**WDT2 Full Reset event log**
20	W1C	RstPwr	**WDT2 SOC Reset event log**
19	W1C	RstPwr	**WDT1 Software Reset event log**
18	W1C	RstPwr	**WDT1 ARM Reset event log**
17	W1C	RstPwr	**WDT1 Full Reset event log**
16	W1C	RstPwr	**WDT1 SOC Reset event log**
15	W1C	Rst02	**Reset event log – SLI_HRST_N Internal Bridge Controller reset**
14	W1C	Rst01	**Reset event log – AHB_HRST_N AHB Bus Controller reset**
13	W1C	Rst03	**Reset event log – SOC_HRST_N AHB SOC reset**
12	W1C	RstARM	**Reset event log – ARMRST_N ARM CA7 CPU reset**
11	W1C	RstFull	**Reset event log – PWRST_N Power-on reset**
10	W1C	RstPwr	**Reset event log – PWRSTNinPLL PLL power-on reset**
9	W1C	RstPwr	**Reset event log – PWRSTNinTrap Power-on strap enable signal**
8	W1C	RstFull	**Reset event log – PWRSTNinR Power-on strap enable signal**
7:6	RO	-	**Reserved(0)**
5	W1C	Rst48	**LPC Host bus reset LPCRST# event log**
4	W1C	Rst46	**LPC/eSPI bus reset ESPIRST# event log**
3	W1C	RstPwr	**MMC ECC fail reset event log**
2	W1C	RstPwr	**Boot flash reset event log – ABR or AdrSwap**
1	W1C	RstPwr	**External reset EXTRST# event log**
0	W1C	RstPwr	**Power on reset SRST# event log**

Note :
All register bits are write ’1’ clear.

Offset: 078h	SCU078: System Reset Event Log Register Set 2-2		Init = 0x07FFFFFF
Bit	**R/W**	**Reset**	**Description**
31:28	RO	-	**Reserved(0)**
27	W1C	Rst61	**Reset event log – ESPI_HRST_N eSPI Controller reset**
26	W1C	Rst38	**Reset event log – SPI_HRST_N SPI Controller reset**
25	W1C	Rst42	**Reset event log – SDC_HRST_N SD/SDIO Controller reset**
24	W1C	Rst50	**Reset event log – PWM_HRST_N PWM/TACHO reset**
23	W1C	RstFull	**Reserved**
22	W1C	RstFull	**Reserved**
21	W1C	Rst60	**Reset event log – I3C5_HRST_N I3C #6 reset**
20	W1C	Rst59	**Reset event log – I3C4_HRST_N I3C #5 reset**
19	W1C	Rst58	**Reset event log – I3C3_HRST_N I3C #4 reset**
18	W1C	Rst57	**Reset event log – I3C2_HRST_N I3C #3 reset**
17	W1C	Rst56	**Reset event log – I3C1_HRST_N I3C #2 reset**
16	W1C	Rst55	**Reset event log – I3C0_HRST_N I3C #1 reset**
15	W1C	Rst54	**Reset event log – I3C_PRST_N I3C DMA reset**
14	W1C	Rst49	**Reset event log – PECI_PRST_N PECI reset**
13	W1C	Rst40	**Reset event log – MAC4_HRST_N Ethernet MAC #4 reset**
12	W1C	Rst39	**Reset event log – MAC3_HRST_N Ethernet MAC #3 reset**
11	W1C	Rst66	**Reset event log – UART4_RST_N UART #4 reset**
10	W1C	Rst65	**Reset event log – UART3_RST_N UART #3 reset**
9	W1C	Rst64	**Reset event log – UART2_RST_N UART #2 reset**
8	W1C	Rst63	**Reset event log – UART1_RST_N UART #1 reset**
7	W1C	PLTRST#	**Reset event log – PLTRST#Platform Reset** When AST2600 is in eSPI mode, it is reset source of SIO, KCS, BT, etc.
6	W1C	Rst45	**Reset event log – MDC_PRST_N MII Controller reset**
5	W1C	Rst47	**Reset event log – LPC_PRST_N LPC controller reset**
4	W1C	Rst53	**Reset event log – I2C_PRST_N I2C reset**
3	W1C	Rst52	**Reset event log – FSI_PRST_N FSI reset**
2	W1C	Rst51	**Reset event log – ADC_PRST_N ADC reset**
1	W1C	Rst43	**Reset event log – JTAG_PRST_N JTAG reset**
0	W1C	Rst44	**Reset event log – GPIO_PRST_N GPIO reset**

Note :
All register bits are write '1' clear.

Offset: 07Ch	SCU07C: System Reset Event Log Register Set 2-3		Init = 0
Bit	**R/W**	**Reset**	**Description**
31:16	RO	-	**reverved(0)**
15	W1C	RstPwr	**WDT8 Software Reset event log**
14	W1C	RstPwr	**WDT8 ARM Reset event log**
13	W1C	RstPwr	**WDT8 Full Reset event log**
12	W1C	RstPwr	**WDT8 SOC Reset event log**
11	W1C	RstPwr	**WDT7 Software Reset event log**
10	W1C	RstPwr	**WDT7 ARM Reset event log**

to next page

from previous page

9	W1C	RstPwr	**WDT7 Full Reset event log**
8	W1C	RstPwr	**WDT7 SOC Reset event**
7	W1C	RstPwr	**WDT6 Software Reset event log**
6	W1C	RstPwr	**WDT6 ARM Reset event log**
5	W1C	RstPwr	**WDT6 Full Reset event log**
4	W1C	RstPwr	**WDT6 SOC Reset event log**
3	W1C	RstPwr	**WDT5 Software Reset event log**
2	W1C	RstPwr	**WDT5 ARM Reset event log**
1	W1C	RstPwr	**WDT5 Full Reset event log**
0	W1C	RstPwr	**WDT5 SOC Reset event log**
Note : All register bits are write '1' clear.			

Offset: 080h **SCU080: Clock Stop Control Register** **Init = 0xFFFF7F8A**

Bit	R/W	Reset	Description
Note : The initial sequence for the clock and reset must follow below steps: 1. Enable engine reset at SCU040/SCU050 2. Delay 100 us 3. Enable clock running SCU080/SCU090 4. Delay 10 ms 5. Disable engine reset at SCU040/SCU050 6. Delay 1 us The sequence must do once whenever the engine is started from the clock stopped state.			
31:28	RW1S	Rst71	**Reserved, must keep at value "111"**
27	RW1S	Rst71	**Stop eMMC Clock (For eMMC Controller)** 0: Enable clock running 1: Stop clock running (default)
26	RW1S	Rst71	**Reserved, must keep at value "1"**
25	RW1S	Rst71	**Stop RVASCLK (For RVAS Engine)** 0: Enable clock running 1: Stop clock running (default)
24	RW1S	Rst71	**Stop RSAECCCLK (For RSA/ECC Controller) Clock** 0: Enable clock running 1: Stop clock running (default)
23:22	RW1S	Rst71	**Reserved, must keep at value "11"**
21	RW1S	Rst71	**Stop MAC#2 (For MAC#2 Controller) Clock** 0: Enable clock running 1: Stop clock running (default)
20	RW1S	Rst71	**Stop MAC#1 (For MAC#1 Controller) Clock** 0: Enable clock running 1: Stop clock running (default)
19:16	RW1S	Rst71	**Reserved, must keep at value "1111"**
15	RW1S	Rst71	**Stop UART5CLK (For UART5 controller)** 0: Enable clock running (default) 1: Stop clock running

to next page

from previous page

14	RW1S	Rst71	**Enable USB2.0 Host/Hub clock** 0: Stop USB2.0 clock running, power-down USB2.0 PHY. (default) 1: Enable USB2.0 clock running The procedure to enable USB2.0 controller: 1. Select the operation mode at SCU440[25:24] 2. Enable USB2.0 global reset by setting SCU040[14] = 1 3. Enable USB2.0 clock running, wait 10 ms for clock stable 4. Disable USB2.0 global reset by setting SCU044[14] = 1 5. Disable USB2.0 PHY reset by setting HUB00[11] = 1 (Hub mode only) 6. Start using USB2.0 controller USB2.0 Host and Hub controller shared the same port, so only one can work at a time. It is determined by SCU440[25:24]. This clock bit controls both Host and Hub, decided by SCU440[25:24].
13	RW1S	Rst71	**Stop YCLK (For HACE)** 0: Enable clock running 1: Stop clock running (default)
12:11	RW1S	Rst71	**Reserved, must keep at value "11"**
10	RW1S	Rst71	**Stop D1CLK (For SOC Display)** 0: Enable clock running 1: Stop clock running (default)
9	RW1S	Rst71	**Stop USB1.1 Host Controller Clock** 0: Enable clock running 1: Stop clock running (default)
8	RW1S	Rst71	**Reserved, must keep at value "1"**
7	RW1S	Rst71	**Stop clock of USB1.1/USB2.0 Host2**~~/USB2.0 Device~~ 0: Enable clock running 1: Stop clock running, power-down USB2.0 PHY. (default) The procedure to enable USB2.0 port#2 controller: 1. Select the operation mode at SCU440[29:28] 2. Enable USB2.0 global reset by setting SCU040[3] = 1 3. Enable USB2.0 clock running, wait 10 ms for clock stable 4. Disable USB2.0 global reset by setting SCU044[3] = 1 5. Disable USB2.0 PHY reset by setting UBD00[11] = 1 (Device mode only) 6. Start using USB2.0 controller USB1.1 HID, USB2.0 Host2 and USB2.0 Device controller shared the same port, so only one can work at a time, which is determined by SCU440[29:28]. This bit enables the clock of controller selected by SCU440[29:28].
6	RW1S	Rst71	**REFCLK1 Stop Enable** 0: Enable clock running (default) 1: Stop clock running Recommand to keep 0
5	RW1S	Rst71	**Stop DCLK (For DAC)** 0: Enable clock running (default) 1: Stop clock running
4	RW1S	Rst71	**Stop BCLK (For PCIe/PCI Bus)** 0: Enable clock running (default) 1: Stop clock running

to next page

from previous page

3	RW1S	Rst71	**Stop VCLK (For Video Capture)** 0: Enable clock running 1: Stop clock running (default)
2	RW1S	Rst71	**Stop GCLK (For 2D Engine)** 0: Enable clock running 1: Stop clock running (default) This register is valid only at CRT Mode (SCU2C[7]).
1	RW1S	Rst71	**Stop ECLK (For Video Engine)** 0: Enable clock running 1: Stop clock running (default)
0	RW1S	Rst71	**Stop MCLK (For SDRAM Controller)** 0: Enable clock running (default) 1: Stop clock running

Offset: 084h	SCU084: Clock Stop Control Clear Register	Init = 0

Bit	R/W	Reset	Description
31:0	W1C	-	**SCU080 Clock Stop Control Clear Register**

Offset: 090h	SCU090: Clock Stop Control Register Set 2	Init = 0xFFF0FFF0

Bit	R/W	Reset	Description
31	RW1S	Rst72	**Reserved, must keep at value "1"**
30	RW1S	Rst72	**Stop FSICLK** 0: Enable clock running 1: Stop clock running (default)
29	RW1S	Rst72	**Stop UART13CLK (For UART13 controller)** 0: Enable clock running 1: Stop clock running (default)
28	RW1S	Rst72	**Stop UART12CLK (For UART12 controller)** 0: Enable clock running 1: Stop clock running (default)
27	RW1S	Rst72	**Stop UART11CLK (For UART11 controller)** 0: Enable clock running 1: Stop clock running (default)
26	RW1S	Rst72	**Stop UART10CLK (For UART10 controller)** 0: Enable clock running 1: Stop clock running (default)
25	RW1S	Rst72	**Stop UART9CLK (For UART9 controller)** 0: Enable clock running 1: Stop clock running (default)
24	RW1S	Rst72	**Stop UART8CLK (For UART8 controller)** 0: Enable clock running 1: Stop clock running (default)
23	RW1S	Rst72	**Stop UART7CLK (For UART7 controller)** 0: Enable clock running 1: Stop clock running (default)
22	RW1S	Rst72	**Stop UART6CLK (For UART6 controller)** 0: Enable clock running 1: Stop clock running (default)
21	RW1S	Rst72	**Stop MAC#4 (For MAC#4 Controller) Clock** 0: Enable clock running 1: Stop clock running (default)
20	RW1S	Rst72	**Stop MAC#3 (For MAC#3 Controller) Clock** 0: Enable clock running 1: Stop clock running (default)
19	RW1S	Rst72	**Stop UART4CLK (For UART4 controller)** 0: Enable clock running (default) 1: Stop clock running
18	RW1S	Rst72	**Stop UART3CLK (For UART3 controller)** 0: Enable clock running (default) 1: Stop clock running
17	RW1S	Rst72	**Stop UART2CLK (For UART2 controller)** 0: Enable clock running (default) 1: Stop clock running

to next page

from previous page

16	RW1S	Rst72	**Stop UART1CLK (For UART1 controller)** 0: Enable clock running (default) 1: Stop clock running
15:14	RW1S	Rst72	**Reserved**
13	RW1S	Rst72	**Stop I3C5CLK (for I3C5 Controller)** 0: Enable clock running 1: Stop clock running (default)
12	RW1S	Rst72	**Stop I3C4CLK (for I3C4 Controller)** 0: Enable clock running 1: Stop clock running (default)
11	RW1S	Rst72	**Stop I3C3CLK (for I3C3 Controller)** 0: Enable clock running 1: Stop clock running (default)
10	RW1S	Rst72	**Stop I3C2CLK (for I3C2 Controller)** 0: Enable clock running 1: Stop clock running (default)
9	RW1S	Rst72	**Stop I3C1CLK (for I3C1 Controller)** 0: Enable clock running 1: Stop clock running (default)
8	RW1S	Rst72	**Stop I3C0CLK (for I3C0 Controller)** 0: Enable clock running 1: Stop clock running (default)
7	RW1S	Rst72	**Reserved**
6	RW1S	Rst72	**Reserved**
5	RW1S	Rst72	**Stop LHCLK (For LPC Master and LPC Plus Controller)** 0: Enable clock running 1: Stop clock running (default)
4	RW1S	Rst72	**Stop SDMCLK (For SD/SDIO Controller)** 0: Enable clock running 1: Stop clock running (default)
3	RW1S	Rst72	**Reserved, must keep at value "1"**
2	RW1S	Rst72	**REFCLK2 Stop Enable** 0: Enable clock running (default) 1: Stop clock running Recommand to keep 0
1	RW1S	Rst72	**Stop eSPI Clock** 0: Enable clock running (default) 1: Stop clock running
0	RW1S	Rst72	**Stop LCLK (for LPC Controller)** 0: Enable clock running (default) 1: Stop clock running

Offset: 094h	**SCU094: Clock Stop Control Clear Register Set 2**		**Init = 0**
Bit	**R/W**	**Reset**	**Description**
31:0	W1C	-	**SCU090 Clock Stop Control Clear Register Set 2**

Offset: 0C0h			SCU0C0: Misc. Control Register	Init = 0x00001000
Bit	**R/W**	**Reset**	**Description**	
31:26	RW	Rst73	**Reserved (0)**	
25	RW	Rst73	**Disable DRAM address space write from PCI bus to AHB bus bridge (P2A)** When disabled, P2A will mask all write command to address space: 1. 0x80000000 - 0xFFFFFFFF	
24	RW	Rst73	**Disable LPC Host/Plus address space write from PCI bus to AHB bus bridge (P2A)** When disabled, P2A will mask all write command to address space: 1. 0x60000000 - 0x7FFFFFFF	
23	RW	Rst73	**Disable SOC address space write from PCI bus to AHB bus bridge (P2A)** When disabled, P2A will mask all write command to address space: 1. 0x10000000 - 0x1FFFFFFF 2. 0x40000000 - 0x5FFFFFFF	
22	RW	Rst73	**Disable flash address space write from PCI bus to AHB bus bridge (P2A)** When disabled, P2A will mask all write command to address space: 1. 0x00000000 - 0x0FFFFFFF 2. 0x20000000 - 0x3FFFFFFF	
21	RW	Rst73	**Reserved (0)**	
20	RW	Rst73	**Select D-PLL parameter source** 0: from VGA. (default) 1: from SCU260 and SCU264	
19	RW	Rst73	**Enable VGA Config Space Prefetch bit setting** 0: PCI VGA Config prefetch bit return 0 (default) 1: PCI VGA Config prefetch bit return 1 This bit setting controls the PCI Config Space Prefetch bit return value.	
18	RW	Rst73	**Select the DisplayPort source for display output** 0: VGA mode (default) 1: SOC Display mode	
17	RW	Rst73	**Disable DAC clock** 0: Normal mode (default) 1: Disable DAC clock	
16	RW	Rst73	**Select the DAC source for display output** 0: VGA mode (default) 1: SOC Display mode	
15:14	RW	Rst73	**JTAG routing selection** 00: Normal, JTAG IO to ARM when SCU040[22]=0 or JTAG IO to JTAG Master when SCU040[22]=1 01: Enable direct JTAG IO to PCIe PHY 10: Enable direct JTAG master to PCIe PHY when SCU040[22]=1 11: Enable direct JTAG master to ARM when SCU040[22]=1	
13	RW	Rst73	**Reserved**	
12	RW	Rst73	**Enable the reference clock divider (div13) for UART5** Combined with SCU0C0[12], SCU304[14] 00: baud rate = 24MHz / (16*divisor) 01: baud rate = 192MHz / (16*divisor) 10: baud rate = (24MHz/13) / (16*divisor) (default) 11: baud rate = (192MHz/13) / (16*divisor)	
11	RW	Rst73	**Enable 2X YCLK** 0: YCLK from MCLK 1: YCLK from MCLK2X	

to next page

from previous page

10	RW	Rst73	**Reserved**
9	RW	Rst73	~~**Disable UART debug port function**~~ The control register is replaced by SCU0C8[1].
8	RW	Rst73	~~**Disable PCI bus to AHB bus bridge**~~ The control register is replaced by SCU0C8[0].
7	RW	Rst73	**Enable 2D ARM SOC Mode function** 0: VGA mode (default) 1: ARM SOC mode
6	RW	Rst73	**Disable VGA CRT display** 0: Enable VGA CRT display (default) 1: Disable VGA CRT display when using Video Direct Fetch mode to reduce VGA memory access bandwidth.
5	RW	Rst73	**Enable VGA legacy registers access when not strap in VGA mode** 0: VGA legacy registers access is controlled by VGA mode strap (default) 1: Force enables VGA legacy registers access
4	RW	Rst73	**Disable E-PLL** 0: Enable E-PLL 1: Disable E-PLL (default) Highest Priority.
3	RW	Rst73	**Disable DAC display** 0: Enable DAC display (default) 1: Disable DAC display
2	RW	Rst73	**Disable D-PLL** 0 : Enable D-PLL (default) 1 : Disable D-PLL Highest Priority.
1:0	RW	Rst73	**Reserved**

Offset: 0C4h **SCU0C4: Misc. 2 Control Register** **Init = 0x2100**

Bit	R/W	Reset	Description
31:16	RW	Rst74	**Reserved (0)**
15: 8	RW	Rst74	**Partical GIC interrupt mask register** 00-15: reserved 16: Mask INT[159:96] 17: Mask INT[159:100] 18: Mask INT[159:104] ... n: Mask INT[159:n*4+32] ... 32 and others: No mask
7	RW	Rst74	**Reserved (0)**
6	RW	Rst74	**LPC IRQ source** 0: LPC related IRQ from LPC/eSPI controller 1: LPC related IRQ from PCI Express LPC controller
5	RW	Rst74	**GIC interrupt timing setting (0)**
4:3	RW	Rst74	**RGMII DMA mode** Set '00' for RGMII1 and RGMII2 as Ethernet mode
2	RW	Rst74	**Enable SRAM Write Protection for Upper 1KB**

to next page

from previous page

1	RW	Rst74	**MALI RC mode** 0: PCI Express PE interface is bridge/device mode (default) 1: PCI Express PE interface is root complex mode
0	RW	Rst74	**Reserved (0)**

Offset: 0C8h **SCU0C8: Debug Control Register** **Init = 0xFFF**

Bit	R/W	Reset	Description
31:12	RO	-	**Reserved (0)**
11	RW	Rst75	**Disable DPMCU and GPMCU debug capability** There are debug features for DPMCU and GPMCU. They can be disabled by set this bit.
10	RW	Rst75	**Disable RSA private key read back** The keys and data of RSA engine are stored in RSA SRAM. The memory private key can be write only to prevent read back by other non-secure programs.
9	RW	Rst75	**Force disable system reset signal of WDT** This bit will force to disable system reset signal of WDT. It is same as SCU500[25] and must be set at the same time with SCUD8[4].
8	RW	Rst75	**Force disable PCIe DMA (XDMA) access for VGA** This bit will force to disable XDMA for VGA 2D engine.
7	RW	Rst75	**Reserved (1)**
6	RW	Rst75	**Force disable the capability for PCIe device port as a Root Complex** The PCIe device port has the capability to be programmed to a Root Complex. This bit can avoid the PCIe device port worked as a Root Complex.
5	RW	Rst75	**Force disable PCIe bus trace buffer** The PCIe controller built-in a trace buffer which can log bus traffics. Set this bit will disable this fucntion.
4	RW	Rst75	**Force disable APB to PCIe bridge config access** Recommend to disable this debug function.
3	RW	Rst75	**Force disable APB to PCIe device bridge** Recommend to disable this debug function.
2	RW	Rst75	**Force disable PCIe DMA (XDMA) access for host and BMC** This bt will force to disable XDMA of host and BMC. The XDMA is capable for host to access BMC memory or BMC to access host memory through PCIe. Recommend to disable this bridge if there are security concerns.
1	RW	Rst75	**Force disable UART5 debug port function** This brige is for debug purpose and is capable to access whole BMC memory space through UART5 port. Recommend to disable this bridge if there are security concerns.
0	RW	Rst75	**Force disable PCI bus to AHB bus bridge** This brige is for debug purpose and is capable to access whole BMC memory space through PCIe. Recommend to disable this bridge if there are security concerns.

Note :
The debug control register values can be automatically load from OTP memory configure space OTPCFG7. In default, the register value will be set to 0xFFF after power on or ARM reset. The default value will change to 0xFFD when SCU51C[0]=1 and OTPCFG0[5]=0 to enable UART5 debug port in default. The auto load is enabled when OTPCFG7[31]=1 and OTPCFG7[11:0] will be load to SCU0C8[11:0] when power on or ARM reset.

Offset: 0D0h	SCU0D0: Misc. 3 Control Register	Init = 0x00000010

Bit	R/W	Reset	Description
31	RW	Rst73	**Reserved (0)**
30	RW	Rst73	**Enable PCIe PEWAKE# output low** 0: inactive, tri-stated output 1: active, output low
29	RW	Rst73	**Enable PCIe PEWAKE# function pin**
28:16	RW	Rst73	**Reserved (0)**
15:14	RW	Rst73	**JTAG routing selection** 00: Normal, IO to ARM or IO to Master 01: Enable direct JTAG IO to PCIe PHY 10: Enable direct JTAG master to PCIe PHY 11: Enable direct JTAG master to ARM
13	RW	Rst73	**Enable the reference clock divider (div13) for RTC** 0: baud rate = 24MHz / (16*divisor) (default) 1: baud rate = (24MHz/13) / (16*divisor)
12: 8	RW	Rst73	**Reserved**
7	RW	Rst73	**Polarity of GPIOB0** 0: Active Low 1: Active High
6	RW	Rst73	**Enable GPIOB0 as eSPI VW reset source** Bit 6 and 7 are for Intel next generation platform.
5: 4	RW	Rst73	**Reserved**
3	RW	Rst73	**Power down video DAC** 0: Enable video DAC (default) 1: Power down video DAC Highest Priority.
2	RW	Rst73	**Reserved**
1	RW	Rst73	**OSC clock output pin selection (For test mode only)** 0: No OSC clock output (default) 1: OSC clock will output from GPIOxx (Y25) pin
0	RW	Rst73	**Disable LPC to SPI flash interfac** 0: Enable (default) 1: Disable

Offset: 0D4h	SCU0D4: Misc. 4 Control Register	Init = 0x03002103

Bit	R/W	Reset	Description
31:24	RW	Rst74	**Reserved (3)**
23:22	RW	Rst74	**Reserved (0)**
21	RW	Rst74	**GPIO Interrupt Enable**
20:16	RW	Rst74	**GPIO Interrupt Selection** 0x0: GPIOA Interrupt Event 0x1: GPIOB Interrupt Event ... 0x1A : GPIOZ Interrupt Event

to next page

from previous page

15: 8	RW	Rst74	**Partical GIC interrupt mask register** 00-15: reserved 16: Mask INT[159:96] 17: Mask INT[159:100] 18: Mask INT[159:104] ... n: Mask INT[159:n*4+32] ... 32 and others: No mask
7:6	RW	Rst74	**Reserved (0)**
5:3	RW	Rst74	**DAC Mode Selection** 000: Normal mode others: Test mode
2:0	RW	Rst74	**DAC Driver full swing (FS) setting (Unit:mA)** $I_{out} = FS * 12.8/DACRSET$

Offset: 0D8h **SCU0D8: Debug Control Register 2** **Init = 0xFF**

Bit	R/W	Reset	Description
31: 6	RW	Rst75	**Reserved (0)**
5	RW	Rst75	**Disable SBMCU debug capability** It is capable to debug for Secure Boot MCU and can be disabled by set this bit. Recommend to disable this debug function.
4	RW	Rst75	**Force disable system reset signal of WDT** This bit will force to disable system reset signal of WDT. It is same as SCU500[25] and must be set at the same time with SCUC8[9].
3	RW	Rst75	**Force disable UART1 debug port function** This brige is for debug purpose and is capable to access whole BMC memory space through UART1 port. Recommend to disable this bridge if there are security concerns.
2	RW	Rst75	**Force disable Firmware Hub to AHB bus bridge** This bridge is for capable to access whole BMC memory space through LPC or ESPI memory and firmware cycle. It can be turned on by setting BMC HICR6[17] and its default is off.
1	RW	Rst75	**Force disable LPC bus to AHB bus bridge** This bridge is for debug purpose and is capable to access whole BMC memory space through LPC or ESPI IO space. The bridge is can be enabled by host and BMC can disable it (HICRB[29]) after power on. Recommend to disable this bridge if there are security concerns.
0	RW	Rst75	**Force disable ESPI bus to AHB bus bridge** This bit will disable ESPI Peripheral Channel Memory access. It is same as to set ESPI080[4]=1 and ESPI080[6]=1.

Note :
The debug control register values can be automatically load from OTP memory configure space OTPCFG7. In default, the register value will be set to 0xFF after power on or ARM reset. The default value will change to 0xF7 when SCU51C[0]=1 and OTPCFG0[5]=0 to enable UART1 debug port in default. The auto load is enabled when OTPCFG7[31]=1 and OTPCFG7[19:16] will be load to SCU0D8[3:0] when power on or ARM reset.

Offset: 0FCh	SCU0FC: DAC CRC Control Register	Init = 0

Bit	R/W	Reset	Description
31: 8	RO	-	**DAC CRC**
7	RO	-	**DAC CRC Done**
6: 2	RO	-	**Reserved (0)**
1	RW	Rst76	**DAC CRC Trig**
0	RW	Rst76	**DAC CRC polarity**

Offset: 100h	SCU100: ASPEED Defined for VGA Function Handshake	Init = 0

Bit	R/W	Reset	Description
31:24	RW	Rst76	**Reserved for ASPEED SDK firmware and SLT test program handshaking** 0x5A: Embedded Linux boot to Linux Properly others: Not defined
23:16	RW	Rst76	**Reserved for Customers definition**
15:8	RW	Rst76	**Reserved for ASPEED definition**
7	RW	Rst76	**DRAM Initial Selection (see note 1)** 0: VBIOS Initial the DRAM 1: SOC Firmware Initial the DRAM
6	RW	Rst76	**SOC Firmware Initial DRAM Status (see note 1)** 0: DRAM Initial is not ready 1: DRAM Initial is Ready
5	RW	Rst76	**Reserved (AST2000 use only)**
4	RW	Rst76	**KVM Virtual EDID Function Selection (see note 2)** 0: disable 1: enable
3	RW	Rst76	**Reserved 0**
2	RW	Rst76	**Reserved 0**
1	RW	Rst76	**BMC Firmware Protection** 1: Forbid SOCFlash to update flash(support from SOCFlash v.1.02.01)
0	RW	Rst76	**iKVM support Wide Screen resolution** 0: iKVM can NOT support wide screen resolution 1: iKVM support wide screen resolution

Note :

1.

```
if (0x1e6e2040 D[7] == 0)
    VBIOS initial the DRAM
Else
    SOC Firmware initial the DRAM
    SOC Firmware set 0x1e6e2040 D[6] to 1 if DRAM initial is ready
    VBIOS POST will wait until 0x1e6e2040 D[6] set by SOC Firmware
End if
```

2.

```
if (0x1e6e2040 D[4] == 0)
    VBIOS get EDID from DDC
Else
    If the Monitor Attached
        Get EDID from DDC
    Else
        Use Virtual EDID as EDID
    End if
End if
```

Offset: 104h	SCU104: ASPEED Defined for VGA Function Handshake		Init = 0
Bit	**R/W**	**Reset**	**Description**
31:0	RW	Rst77	**The last serviced IRQ number**

Offset: 108h	SCU108: VGA Scratch Register		Init = 0
Bit	**R/W**	**Reset**	**Description**
31:0	RW	Rst78	**VGA Scratch Register**

Offset: 10Ch	SCU10C: VGA Scratch Register		Init = 0
Bit	**R/W**	**Reset**	**Description**
31:0	RW	Rst79	**VGA Scratch Register**

Offset: 180h–1ACh SCU180 ~ SCU1AC: CPU Scratch Register. Reserved for SMP booting Init = 0

Offset	Bit	Attr.		Reset Description
180h	31:0	RW	Rst80	**CPU scratch register SCU180 bit[31:0]**
184h	31:0	RW	Rst80	**CPU scratch register SCU184 bit[31:0]**
188h	31:0	RW	Rst80	**CPU scratch register SCU188 bit[31:0]**
18Ch	31:0	RW	Rst80	**CPU scratch register SCU18C bit[31:0]**
190h	31:0	RW	Rst81	**CPU scratch register SCU190 bit[31:0]**
194h	31:0	RW	Rst81	**CPU scratch register SCU194 bit[31:0]**
198h	31:0	RW	Rst81	**CPU scratch register SCU198 bit[31:0]**
19Ch	31:0	RW	Rst81	**CPU scratch register SCU19C bit[31:0]**
1A0h	31:0	RW	Rst82	**CPU scratch register SCU1A0 bit[31:0]**
1A4h	31:0	RW	Rst82	**CPU scratch register SCU1A4 bit[31:0]**
1A8h	31:0	RW	Rst82	**CPU scratch register SCU1A8 bit[31:0]**
1ACh	31:0	RW	Rst82	**CPU scratch register SCU1AC bit[31:0]**

Offset: 1B0h–1FCh SCU1B0 ~ SCU1FC: CPU Scratch Register. Available for SW use Init = 0

Offset	Bit	Attr.		Reset Description
1B0h	31:0	RW	Rst83	**CPU scratch register SCU1B0 bit[31:0]**
1B4h	31:0	RW	Rst83	**CPU scratch register SCU1B4 bit[31:0]**
1B8h	31:0	RW	Rst83	**CPU scratch register SCU1B8 bit[31:0]**
1BCh	31:0	RW	Rst83	**CPU scratch register SCU1BC bit[31:0]**
1C0h	31:0	RW	Rst84	**CPU scratch register SCU1C0 bit[31:0]**
1C4h	31:0	RW	Rst84	**CPU scratch register SCU1C4 bit[31:0]**
1C8h	31:0	RW	Rst84	**CPU scratch register SCU1C8 bit[31:0]**
1CCh	31:0	RW	Rst84	**CPU scratch register SCU1CC bit[31:0]**
1D0h	31:0	RW	Rst85	**CPU scratch register SCU1D0 bit[31:0]**
1D4h	31:0	RW	Rst85	**CPU scratch register SCU1D4 bit[31:0]**
1D8h	31:0	RW	Rst85	**CPU scratch register SCU1D8 bit[31:0]**
1DCh	31:0	RW	Rst85	**CPU scratch register SCU1DC bit[31:0]**

to next page

from previous page

1E0h	31:0	RW	Rst86	**CPU scratch register SCU1E0 bit[31:0]**
1E4h	31:0	RW	Rst86	**CPU scratch register SCU1E4 bit[31:0]**
1E8h	31:0	RW	Rst86	**CPU scratch register SCU1E8 bit[31:0]**
1ECh	31:0	RW	Rst86	**CPU scratch register SCU1EC bit[31:0]**
1F0h	31:0	RW	Rst87	**CPU scratch register SCU1F0 bit[31:0]**
1F4h	31:0	RW	Rst87	**CPU scratch register SCU1F4 bit[31:0]**
1F8h	31:0	RW	Rst87	**CPU scratch register SCU1F8 bit[31:0]**
1FCh	31:0	RW	Rst87	**CPU scratch register SCU1FC bit[31:0]**

Offset: 200h **SCU200: H-PLL Parameter Register** **Init = 0x1000408F**

Bit	R/W	Reset	Description
31:29	RW	Rst88	**Reserved(0)**
28:26	RW	Rst88	**H-PLL parameters** Reserved, must keep value at default value
25	RW	Rst88	**Enable H-PLL reset** 0: Normal operation 1: Reset PLL
24	RW	Rst88	**Enable H-PLL bypass mode** 0: No operation 1: Enable H-PLL bypass mode When enabling H-PLL bypass mode, the output clock of H-PLL is bypassed from the CLKIN input pin.
23	RW	Rst88	**Turn off H-PLL** 0: No operation 1: Turn Off H-PLL When H-PLL is turned off, it will enter power down mode. The output signal may be ”0” or ”1”.
22:19	RW	Rst88	**H-PLL Post Divider (P)**
18:13	RW	Rst88	**H-PLL Denumerator (N)**
12:0	RW	Rst88	**H-PLL Numerator (M)**

Note :
H-PLL is preliminarily designed to generate the running frequency of ARM CPU. The output frequency of H-PLL is based on the following equation:

(Output frequency) = CLKIN(25MHz) * [(M+1) / (N+1)] / (P+1)
(Divide 1 Output frequency range) = 720MHz - 1.8GHz
(Divided reference frequency range) = 439KHz - 1.8GHz
(Output divider values) = 1, 2-16 (even only)
(Bandwidth Adjustment) = NB = BWADJ[11:0] + 1
(typically: NB = (M+1) / 2
The default frequency of H-PLL is 1200MHz when CLKIN = 25MHz.
Note: if SCUF88[0] is 1, when RstARM comes, SCU200 will be 0x1000405f, which is 800MHz.
This is a kind of fail-safe that if SCU200 is incorrect and hang, it will return to a frequency slow enough to start over.

Offset: 204h **SCU204: Extended H-PLL Parameter Register** **Init = 0x31**

Bit	R/W	Reset	Description
31	RO	-	**H-PLL Lock ready reak back**

to next page

from previous page

30:12	RO	-	**Reserved(0)**
11: 0	RW	Rst89	**H-PLL Bandwidth Adjustment bit[15:10]**

Note :
(Bandwidth Adjustment) = NB = BWADJ[11:0] + 1
(typically: NB = (M+1) / 2

Offset: 210h **SCU210: A-PLL Parameter Register** **Init = 0x1000405F**

Bit	R/W	Reset	Description
31:29	RW	Rst90	**Reserved(0)**
28:26	RW	Rst90	**A-PLL parameters** Reserved, must keep value at default value
25	RW	Rst90	**Enable A-PLL reset** 0: Normal operation 1: Reset PLL
24	RW	Rst90	**Enable A-PLL bypass mode** 0: No operation 1: Enable A-PLL bypass mode When enabling A-PLL bypass mode, the output clock of A-PLL is bypassed from the CLKIN input pin.
23	RW	Rst90	**Turn off A-PLL** 0: No operation 1: Turn Off A-PLL When A-PLL is turned off, it will enter power down mode. The output signal may be ”0” or ”1”.
22:19	RW	Rst90	**A-PLL Post Divider (P)**
18:13	RW	Rst90	**A-PLL Denumerator (N)**
12:0	RW	Rst90	**A-PLL Numerator (M)**

Note :
The output frequency of A-PLL is based on the following equation:

(Output frequency) = CLKIN(25MHz) * [(M+1) / (N+1)] / (P+1)
(Divide 1 Output frequency range) = 720MHz - 1.8GHz
(Divided reference frequency range) = 439KHz - 1.8GHz
(Output divider values) = 1, 2-16 (even only)
(Bandwidth Adjustment) = NB = BWADJ[11:0] + 1
(typically: NB = (M+1) / 2
The default frequency of A-PLL is 800MHz when CLKIN = 25MHz.

Offset: 214h **SCU214: Extended A-PLL Parameter Register** **Init = 0x2F**

Bit	R/W	Reset	Description
31	RO	-	**a-PLL Lock ready reak back**
30:12	RO	-	**Reserved(0)**
11: 0	RW	Rst91	**a-PLL Bandwidth Adjustment bit[15:10]**

Note :
(Bandwidth Adjustment) = NB = BWADJ[11:0] + 1
(typically: NB = (M+1) / 2

Offset: 220h	SCU220: M-PLL Parameter Register	Init = 0x1008405F

Bit	R/W	Reset	Description
31:29	RW	Rst92	**Reserved(0)**
28:26	RW	Rst92	**M-PLL parameters** Reserved, must keep value at default value
25	RW	Rst92	**Enable M-PLL reset** 0: Normal operation 1: Reset PLL
24	RW	Rst92	**Enable M-PLL bypass mode** 0: No operation 1: Enable M-PLL bypass mode When enabling M-PLL bypass mode, the output clock of M-PLL is bypassed from the CLKIN input pin.
23	RW	Rst92	**Turn off M-PLL** 0: No operation 1: Turn Off M-PLL When M-PLL is turned off, it will enter power down mode. The output signal may be ”0” or ”1”.
22:19	RW	Rst92	**M-PLL Post Divider (P)**
18:13	RW	Rst92	**M-PLL Denumerator (N)**
12:0	RW	Rst92	**M-PLL Numerator (M)**

Note :
M-PLL is preliminarily designed to generate the running frequency of DDR controller. The output frequency of M-PLL is based on the following equation:

(Output frequency) = CLKIN(25MHz) * [(M+1) / (N+1)] / (P+1)
(Divide 1 Output frequency range) = 720MHz - 1.8GHz
(Divided reference frequency range) = 439KHz - 1.8GHz
(Output divider values) = 1, 2-16 (even only)
The default frequency of M-PLL is 400MHz when CLKIN = 25MHz and create DDR-1600 speed.

Offset: 224h	SCU224: Extended M-PLL Parameter Register	Init = 0x2F

Bit	R/W	Reset	Description
31	RO	-	**M-PLL Lock ready reak back**
30:12	RO	-	**Reserved(0)**
11: 0	RW	Rst93	**M-PLL Bandwidth Adjustment bit[15:10]**

Note :
(Bandwidth Adjustment) = NB = BWADJ[11:0] + 1
(typically: NB = (M+1) / 2

Offset: 240h	SCU240: E-PLL Parameter Register	Init = 0x10004077

Bit	R/W	Reset	Description
31:29	RW	Rst94	**Reserved(0)**
28:26	RW	Rst94	**E-PLL parameters** Reserved, must keep value at default value
25	RW	Rst94	**Enable E-PLL reset** 0: Normal operation 1: Reset PLL

to next page

from previous page

24	RW	Rst94	**Enable E-PLL bypass mode** 0: No operation 1: Enable E-PLL bypass mode When enabling E-PLL bypass mode, the output clock of E-PLL is bypassed from the CLKIN input pin.
23	RW	Rst94	**Turn off E-PLL** 0: No operation 1: Turn Off E-PLL When E-PLL is turned off, it will enter power down mode. The output signal may be ”0” or ”1”.
22:19	RW	Rst94	**E-PLL Post Divider (P)**
18:13	RW	Rst94	**E-PLL Denumerator (N)**
12:0	RW	Rst94	**E-PLL Numerator (M)**

Note :
E-PLL is preliminarily designed to generate the running frequency of some controllers. The output frequency of E-PLL is based on the following equation:

(Output frequency) = CLKIN(25MHz) * [(M+1) / (N+1)] / (P+1)
(Divide 1 Output frequency range) = 720MHz - 1.8GHz
(Divided reference frequency range) = 439KHz - 1.8GHz
(Output divider values) = 1, 2-16 (even only)
The default frequency of E-PLL is 1000MHz when CLKIN = 25MHz.

Offset: 244h **SCU244: Extended E-PLL Parameter Register** **Init = 0x3D**

Bit	R/W	Reset	Description
31	RO	-	**E-PLL Lock ready reak back**
30:12	RO	-	**Reserved(0)**
11: 0	RW	Rst95	**E-PLL Bandwidth Adjustment bit[15:10]**

Note :
(Bandwidth Adjustment) = NB = BWADJ[11:0] + 1
(typically: NB = (M+1) / 2

Offset: 260h **SCU260: D-PLL Parameter Register** **Init = 0x1078405F**

Bit	R/W	Reset	Description
31:29	RW	Rst96	**Reserved(0)**
28:26	RW	Rst96	**D-PLL parameters** Reserved, must keep value at default value
25	RW	Rst96	**Enable D-PLL reset** 0: Normal operation 1: Reset PLL
24	RW	Rst96	**Enable D-PLL bypass mode** 0: No operation 1: Enable D-PLL bypass mode When enabling D-PLL bypass mode, the output clock of D-PLL is bypassed from the CLKIN input pin.

to next page

from previous page

23	RW	Rst96	**Turn off D-PLL** 0: No operation 1: Turn Off D-PLL When D-PLL is turned off, it will enter power down mode. The output signal may be ”0” or ”1”.
22:19	RW	Rst96	**D-PLL Post Divider (P)**
18:13	RW	Rst96	**D-PLL Denumerator (N)**
12:0	RW	Rst96	**D-PLL Numerator (M)**

Note :
D-PLL is preliminarily designed to generate the running frequency of some controllers. The output frequency of D-PLL is based on the following equation:

(Output frequency) = CLKIN(25MHz) * [(M+1) / (N+1)] / (P+1)
(Divide 1 Output frequency range) = 720MHz - 1.8GHz
(Divided reference frequency range) = 439KHz - 1.8GHz
(Output divider values) = 1, 2-16 (even only)
The default frequency of D-PLL is 50MHz when CLKIN = 25MHz

Offset: 264h **SCU264: Extended D-PLL Parameter Register** **Init = 0x31**

Bit	R/W	Reset	Description
31	RO	-	**D-PLL Lock ready reak back**
30:12	RO	-	**Reserved(0)**
11: 0	RW	Rst97	**D-PLL Bandwidth Adjustment bit[15:10]**

Note :
(Bandwidth Adjustment) = NB = BWADJ[11:0] + 1
(typically: NB = (M+1) / 2

Offset: 300h **SCU300: Clock Selection Register** **Init = 0xF3940000**

Bit	R/W	Reset	Description
31	RW	Rst98	**Enable Video Engine clock dynamic slow down** 0: disable 1: enable When slow down, the clock will be divided to ECLK_source/8.
30:28	RW	Rst98	**Video Engine clock source divider** 000: ECLK_source/2 001: ECLK_source/2 010: ECLK_source/3 011: ECLK_source/4 100: ECLK_source/5 101: ECLK_source/6 110: ECLK_source/7 111: ECLK_source/8 The clock source is selected by bit[3:2].
27:26	RW	Rst98	**ECC/RSA engine clock divider selection** 00: ECCCLK/RSACLK = (M-PLL or H-PLL)/2 01: ECCCLK/RSACLK = (M-PLL or H-PLL)/2 10: ECCCLK/RSACLK = (M-PLL or H-PLL)/3 11: ECCCLK/RSACLK = (M-PLL or H-PLL)/4

to next page

from previous page

25:23	RW	Rst98	**APB Bus PCLK divider selection** 000: PCLK = H-PLL/4 001: PCLK = H-PLL/8 010: PCLK = H-PLL/12 011: PCLK = H-PLL/16 100: PCLK = H-PLL/20 101: PCLK = H-PLL/24 110: PCLK = H-PLL/28 111: PCLK = H-PLL/32 There is a limitation on the PCLK frequency allowed. PCLK > 0.5*LCLK (33MHz).
22:20	RW	Rst98	**P-Bus BCLK divider selection** 000: BCLK = E-PLL/4 001: BCLK = E-PLL/8 010: BCLK = E-PLL/12 011: BCLK = E-PLL/16 100: BCLK = E-PLL/20 101: BCLK = E-PLL/24 110: BCLK = E-PLL/28 111: BCLK = E-PLL/32
19	RW	Rst98	**ECC/RSA engine clock selection** 0: M-PLL 1: H-PLL
18:16	RW	Rst98	**MAC AHB bus clock divider selection** 000: H-PLL/4 001: H-PLL/4 010: H-PLL/6 011: H-PLL/8 100: H-PLL/10 101: H-PLL/12 110: H-PLL/14 111: H-PLL/16 This register is designed to select the MAC controller bridge clock. If all MAC ports are running at 10/100Mbps, then the bridge clock should be at least 25MHz. If any one port is running at 1000Mbps, then it should be set at least 100MHz. The frequency of bridge may affect the MAC performance, but higher frequency would lead to higher power consumption. So an appropriate setting is required.
15	RW	Rst98	**eMMC clock running enable** 0: Stop clock 1: Enable clock
14:12	RW	Rst98	**eMMC divider selection** 000: eMMCCLK = eMMCCLK_Src/2 001: eMMCCLK = eMMCCLK_Src/4 010: eMMCCLK = eMMCCLK_Src/6 011: eMMCCLK = eMMCCLK_Src/8 100: eMMCCLK = eMMCCLK_Src/10 101: eMMCCLK = eMMCCLK_Src/12 110: eMMCCLK = eMMCCLK_Src/14 111: eMMCCLK = eMMCCLK_Src/16
11	RW	Rst98	**Select source of eMMCCLK_Src** 0: H-PLL/2 1: M-PLL

to next page

from previous page

10:8	RW	Rst98	**SOC Display clock selection** 000: D-PLL 001: Reserved 010: 40MHz from USB2.0 port1 PHY 011: GPIOC6 100: Reserved 101: E-PLL divided by SCU308[17:12] 110: M-PLL divided by SCU308[17:12] 111: H-PLL divided by SCU308[17:12]
7	RW	Rst98	**ARM CPU/AHB clock static slow down enable** 0: Disable 1: Enable slow down Enable this bit will disable dynamic mode.
6:4	RW	Rst98	**ARM CPU clock static/dynamic slow down setting** 000: H-PLL/2 001: H-PLL/4 010: H-PLL/6 011: H-PLL/8 100: H-PLL/10 101: H-PLL/12 110: H-PLL/14 111: H-PLL/16 This register is designed to slow down ARM CPU clock for reducing power consumption in standby mode. The clock divider is embedded with anti-glitch logic to protect CPU operations.
3:2	RW	Rst98	**Video Engine ECLK clock source selection** 00: The clock source of ECLK is from M-PLL clock output 01: The clock source of ECLK is from H-PLL clock output 1x: The clock source of ECLK is from D-PLL clock output Before issuing command to change this register, it had better to stop ECLK, and Video Engine must be reset in advance in order to the potential risk in changing this clock. The clock divider ratio is defined at bit[30:28].
1	RW	Rst98	**CPU/AHB clock dynamic slow down idle timer** 0: 128 HCLK idle (default) 1: 256 HCLK idle
0	RW	Rst98	**CPU/AHB clock dynamic slow down enable** 0: Disable dynamic slow down 1: Enable dynamic slow down The clock slow down ratio is defined at bit[6:4].

Offset: 304h	**SCU304: Clock Selection Register Set 2**		**Init = 0x00700000**
Bit	**R/W**	**Reset**	**Description**
31:29	RO	-	**Reserved(0)**
28	RW	Rst99	**PCIe LPC virtual UART clock selection** 0: 24MHz/13 1: 192MHz/13

to next page

from previous page

27:24	RW	Rst99	**Internal PCIe 100MHz reference clock divide selection** 0000: E-PLL/2 0001: E-PLL/4 0010: E-PLL/6 0011: E-PLL/8 0100: E-PLL/10 0101: E-PLL/12 0110: E-PLL/14 0111: E-PLL/16 1000: H-PLL/2 1001: H-PLL/4 1010: H-PLL/6 1011: H-PLL/8 1100: H-PLL/10 1101: H-PLL/12 1110: H-PLL/14 1111: H-PLL/16
23	RW	Rst99	**RGMII 12 125MHz clock divider source selection** 0: E-PLL 1: H-PLL
22:20	RW	Rst99	**RGMII 12 125MHz clock divider ratio** 000: div2 001: div2 010: div3 .... 111: div8
19	RW	Rst99	**RMII 12 50MHz clock divider source selection** 0: E-PLL 1: H-PLL
18:16	RW	Rst99	**RMII 12 50MHz clock divider ratio** 000: div4 001: div8 010: div12 011: div16 100: div20 101: div24 110: div28 111: div32
15	RW	Rst99	**UART debug port clock selection** 0: 24MHz/13 1: 192MHz/13
14	RW	Rst99	**UART5 clock selection** Combined with SCU0C0[12], SCU304[14]: 00: 24MHz 01: 192MHz 10: 24MHz/13 11: 192MHz/13
13:12	RW	Rst99	**VCLK Source Selection** 00: DCLK 01: D1CLK 10: HCLK 11: MCLK

to next page

from previous page

11:6	RW	Rst99	**Video input port clock delay selection** SCU304[11]: inverted output SCU304[10:6]: delay line stage
5:0	RW	Rst99	**Video port output clock delay control** SCU304[5]: inverted output SCU304[4:0]: delay line stage

Offset: 308h **SCU308: Clock Selection Register Set 3** **Init = 0**

Bit	R/W	Reset	Description
31:18	RO	-	**Reserved**
17:12	RW	Rst100	**SOC Display clock divider selection** 000000: div 1 011011: div 16 others: Reserved
11:0	RW	Rst100	**Reserved**

Offset: 310h **SCU310: Clock Selection Register Set 4** **Init = 0xF3F40000**

Bit	R/W	Reset	Description
31	RW	Rst101	**SD/SDIO clock (SDMCLK) running enable** 0: Stop clock 1: Enable clock
30:28	RW	Rst101	**SD/SDIO clock (SDMCLK) divider selection** Clock Source depends on bit 8 000: SDMCLK = Clock Source/2 001: SDMCLK = Clock Source/4 010: SDMCLK = Clock Source/6 011: SDMCLK = Clock Source/8 100: SDMCLK = Clock Source/10 101: SDMCLK = Clock Source/12 110: SDMCLK = Clock Source/14 111: SDMCLK = Clock Source/1
27	RW	Rst101	**Reserved**
26:24	RW	Rst101	**MAC AHB bus clock divider selection** 000: HCLK/2 001: HCLK/2 010: HCLK/3 011: HCLK/4 100: HCLK/5 101: HCLK/6 110: HCLK/7 111: HCLK/8 This register is designed to select the MAC controller bridge clock. If all MAC ports are running at 10/100Mbps, then the bridge clock should be at least 25MHz. If any one port is running at 1000Mbps, then it should be set at least 100MHz. The frequency of bridge may affect the MAC performance, but higher frequency would lead to higher power consumption. So an appropriate setting is required.
23	RW	Rst101	**Reserved**

to next page

from previous page

22:20	RW	Rst101	**RGMII 34 125MHz clock divider ratio** 000: HCLK / 2 001: HCLK / 2 010: HCLK / 3 .... 111: HCLK / 8
19	RW	Rst101	**Reserved**
18:16	RW	Rst101	**RMII 34 50MHz clock divider ratio** 000: HCLK / 2 001: HCLK / 2 010: HCLK / 3 .... 111: HCLK / 8
15	RW	Rst101	**LPC Host LHCLK clock generation/output enable control** 0: Disable, LHCLK come from external source 1: Enable, LHCLK is generated and output internally
14:12	RW	Rst101	**LPC Host LHCLK divider selection** 000: LHCLK = HCLK/2 001: LHCLK = HCLK/4 010: LHCLK = HCLK/6 011: LHCLK = HCLK/8 100: LHCLK = HCLK/10 101: LHCLK = HCLK/12 110: LHCLK = HCLK/14 111: LHCLK = HCLK/16
11:9	RW	Rst101	**APB Bus PCLK divider selection** 000: PCLK = HCLK/2 001: PCLK = HCLK/4 010: PCLK = HCLK/6 011: PCLK = HCLK/8 100: PCLK = HCLK/10 101: PCLK = HCLK/12 110: PCLK = HCLK/14 111: PCLK = HCLK/16 There is a limitation on the PCLK frequency allowed. PCLK > 0.5*LCLK (33MHz).
8	RW	Rst101	**Select clock source of SD/SDIO clock (SDMCLK)** 0: 200MHz HCLK clock 1: APLL clock
7	RW	Rst101	**Select PECI clock source** 0: 25MHz REFCLK clock 1: 200MHz HCLK clock
6	RW	Rst101	**Select UART debug port clock source** 0: UARTCLK 24MHz / 13 (SCU338) 1: HUARTCLK 24MHz * 32 / 13 (SCU33C)
5	RW	Rst101	**Select UART6 clock source** 0: UARTCLK 24MHz / 13 (SCU338) 1: HUARTCLK 24MHz * 32 / 13 (SCU33C)
4	RW	Rst101	**Reserved** Reserved

to next page

from previous page

3	RW	Rst101	**Select UART4 clock source** 0: UARTCLK 24MHz / 13 (SCU338) 1: HUARTCLK 24MHz * 32 / 13 (SCU33C)
2	RW	Rst101	**Select UART3 clock source** 0: UARTCLK 24MHz / 13 (SCU338) 1: HUARTCLK 24MHz * 32 / 13 (SCU33C)
1	RW	Rst101	**Select UART2 clock source** 0: UARTCLK 24MHz / 13 (SCU338) 1: HUARTCLK 24MHz * 32 / 13 (SCU33C)
0	RW	Rst101	**Select UART1 clock source** 0: UARTCLK 24MHz / 13 (SCU338) 1: HUARTCLK 24MHz * 32 / 13 (SCU33C)

Offset: 314h **SCU314: Clock Selection Register Set 5** **Init = 0x30000000**

Bit	R/W	Reset	Description
31	RW	Rst102	**I3C clock selection** 0: HCLK 1: APLLdivN
30:28	RW	Rst102	**APLL divider selection** 000: APLLdivN = APLL/2 001: APLLdivN = APLL/2 010: APLLdivN = APLL/3 011: APLLdivN = APLL/4 100: APLLdivN = APLL/5 101: APLLdivN = APLL/6 110: APLLdivN = APLL/7 111: APLLdivN = APLL/8
27:22	RW	Rst102	**VPICLK delay selection**
21:16	RW	Rst102	**VPCLK delay selection**
15	RW	Rst102	**Reserved**
14:13	RW	Rst102	**I3CHCLK selection** 00b: APLL / 4 01b: APLL / 2 10b: APLL 11b: HCLK
12	RW	Rst102	**Select UART13 clock source** 0: UARTCLK 24MHz / 13 (SCU338) 1: HUARTCLK 24MHz * 32 / 13 (SCU33C)
11	RW	Rst102	**Select UART12 clock source** 0: UARTCLK 24MHz / 13 (SCU338) 1: HUARTCLK 24MHz * 32 / 13 (SCU33C)
10	RW	Rst102	**Select UART11 clock source** 0: UARTCLK 24MHz / 13 (SCU338) 1: HUARTCLK 24MHz * 32 / 13 (SCU33C)
9	RW	Rst102	**Select UART10 clock source** 0: UARTCLK 24MHz / 13 (SCU338) 1: HUARTCLK 24MHz * 32 / 13 (SCU33C)
8	RW	Rst102	**Select UART9 clock source** 0: UARTCLK 24MHz / 13 (SCU338) 1: HUARTCLK 24MHz * 32 / 13 (SCU33C)

to next page

from previous page

7	RW	Rst102	**Select UART8 clock source** 0: UARTCLK 24MHz / 13 (SCU338) 1: HUARTCLK 24MHz * 32 / 13 (SCU33C)
6	RW	Rst102	**Select UART7 clock source** 0: UARTCLK 24MHz / 13 (SCU338) 1: HUARTCLK 24MHz * 32 / 13 (SCU33C)
5:3	RW	Rst102	**HUXCLK selection** 00b: APLL / 4 01b: APLL / 2 10b: APLL 11b: HCLK
2:0	RW	Rst102	**UXCLK selection** 00b: APLL / 4 01b: APLL / 2 10b: APLL 11b: HCLK

Offset: 320h **SCU320: Frequency Counter Control Register** **Init = 0**

Bit	R/W	Reset	Description
31:30	RO	-	**Reserved (0)**
29:16	RO	Rst103	**Value of frequency measurement counter** Reset to 0 automatically when starts counting Algorithm of frequency measurement: Frequency = (25MHz / 512) * (Value + 1)
15:9	RW	Rst103	**Delay ring stage control for DLY16 and DLY32**
8	RW	Rst103	**Enable OSC counter result output to pin** 0: disable 1: enable (for debugging purpose only)
7	RO	-	**Clock frequency measurement compare result** 0 : Fail 1 : Pass This status flag is the result by comparing the output counter value at SCU320[29:16] with the upper and lower limit defined at SCU324, if the lower_limit <= counter <= upper_limit, then the compare result is Pass.
6	RO	-	**Clock frequency measurement finished** 0 : Not finished 1 : Finished This status flag can be cleared by setting SCU320[1] to '0'

to next page

from previous page

5:2	RW	Rst103	**Clock source selection for clock frequency measurement** 0000: Select delay cell (4 stages) based ring oscillator (with 1/16 clock divider) 0001: Select NAND gate (41 stages) based ring oscillator (with 1/16 clock divider) 0010: Select DLY16 (with 1/4 clock divider) 0011: Select DLY32 (with 1/4 clock divider) 0100: Select D-PLL 0101: Select E-PLL/4 0110: Select XPCLK of PCIe Device 0111: Select XPCLK of PCIe RC 1000: Select MCLK 1001: Select HCLK 1010: Select UTMICLK of USB Port A 1011: Select UTMICLK of USB Port B 1100: Select DP_PLL of DP PHY 1101: Select 48MCLK of USB Port A 1110: Select DACCLK 1111: Select RGMIICLK This register is designed to select the clock source for clock frequency measurement.
1	RW	Rst103	**Oscillator Counter Enable** 0 : Reset frequency measurement counter 1 : Enable frequency measurement counter
0	RW	Rst103	**Enable Ring Oscillator** 0 : Disable ring oscillators 1 : Enable ring oscillators Before enabling the measurement of ring oscillator frequency, SW must enable this bit and wait for 1ms to make sure the ring oscillators are stable. After finished the measurement, SW must disable ring oscillators to reduce power consumption.

Note :
The procedure to start counter:

1. Set SCU320 = 0x24
2. Wait until SCU320[29:16] = 0
3. Set SCU320[0] = 1 and SCU320[5:2] = clock for measurement
4. Delay 1ms
5. Set SCU320[1] = 1
6. Wait until SCU320[6] = 1
7. Read SCU320[29:16] and calculate the result frequency using following equation

Oscillator Counter Algorithm :
When the reference clock CLK25M count from 0 to 512, measure the OSCCLK counting value, then

OSCCLK frequency = CLK25M / 512 * (SCU320[29:16] + 1)

Offset: 324h **SCU324: Frequency counter comparison range** **Init = 0**

Bit	R/W	Reset	Description
31:30	RO	Rst104	**Reserved (0)**
29:16	RW	Rst104	**Lower Limit**
15:14	RO	Rst104	**Reserved (0)**
13:0	RW	Rst104	**Upper Limit**

SCU32C: Reserve			
Offset: 32Ch			**Init = 0x00011320**
Bit	**Attr.**	**Reset**	**Description**
31:18	RO	Rst105	**Reserved**
17:0	RW	Rst105	**Reserved**

Offset: 330h		**SCU330: Frequency Counter Control Register**	**Init = 0**
Bit	**R/W**	**Reset**	**Description**
31:30	RO	-	**Reserved (0)**
29:16	RO	Rst106	**Value of frequency measurement counter** Reset to 0 automatically when starts counting Algorithm of frequency measurement: Frequency = (25MHz / 512) * (Value + 1)
15:9	RW	Rst106	**Delay ring stage control**
8	RW	Rst106	**Enable OSC counter result output to pin** 0: disable 1: enable (for debugging purpose only)
7	RO	Rst106	**Clock frequency measurement compare result** 0 : Fail 1 : Pass This status flag is the result by comparing the output counter value at SCU330[29:16] with the upper and lower limit defined at SCU334, if the lower_limit <= counter <= upper_limit, then the compare result is Pass.
6	RO	Rst106	**Clock frequency measurement finished** 0 : Not finished 1 : Finished This status flag can be cleared by setting SCU10[1] to '0'
5:2	RW	Rst106	**Clock source selection for clock frequency measurement** 0000: APLL / 8 0001: HCLK 0010: PCLK 0011: LCLK 0100: delay cell (4 stages) based ring oscillator (with 1/16 clock divider) 0101: NAND gate (41 stages) based ring oscillator (with 1/16 clock divider) 0110: DLY16 (with 1/4 clock divider) 0111: DLY32 (with 1/4 clock divider) This register is designed to select the clock source for clock frequency measurement.
1	RW	Rst106	**Oscillator Counter Enable** 0 : Reset frequency measurement counter 1 : Enable frequency measurement counter
0	RW	Rst106	**Enable Ring Oscillator** 0 : Disable ring oscillators 1 : Enable ring oscillators Before enabling the measurement of ring oscillator frequency, SW must enable this bit and wait for 1ms to make sure the ring oscillators are stable. After finished the measurement, SW must disable ring oscillators to reduce power consumption.

to next page

from previous page

Note :
The procedure to start counter:

1. Set SCU330 = 0x04
2. Wait until SCU330[29:16] = 0
3. Set SCU330[0] = 1 and SCU330[5:2] = clock for measurement
4. Delay 1ms
5. Set SCU330[1] = 1
6. Wait until SCU330[6] = 1
7. Read SCU330[29:16] and calculate the result frequency using following equation

Oscillator Counter Algorithm :
When the reference clock CLK25M count from 0 to 512, measure the OSCCLK counting value, then

OSCCLK frequency = CLK25M / 512 * (SCU330[29:16] + 1)

Offset: 334h **SCU334: Frequency counter comparison range** **Init = X**

Bit	R/W	Reset	Description
31:30	RO	Rst107	**Reserved (0)**
29:16	RW	Rst107	**Upper Limit**
15:14	RO	Rst107	**Reserved (0)**
13:0	RW	Rst107	**Lower Limit**

SCU338: Generate UARTCLK from UXCLK

Offset: 338h **Init = 0x00014506**

Bit	Attr.	Reset	Description
31:18	RO	Rst108	**Reserved**
17:8	RW	Rst108	**N-Value**
7:0	RW	Rst108	**R-Value**

Note :
UARTCLK = UXCLK * R / (N * 2)
Need to change the value when UXCLK is not 200MHz. Recommend to keep UARTCLK frequency equal to default 24/13 MHz.

SCU33C: Generate HUARTCLK from HUXCLK

Offset: 33Ch **Init = 0x000145C0**

Bit	Attr.	Reset	Description
31:18	RO	Rst109	**Reserved**
17:8	RW	Rst109	**HN-Value**
7:0	RW	Rst109	**HR-Value**

Note :
HUARTCLK = HUXCLK * HR / (HN * 2)
Recommend to program HUARTCLK frequency for higher UART baud rate. The default frequency is 768/13 MHz.

Offset: 340h	SCU340: MAC 12 Interface Clock Delay Setting	Init = 0

Bit	R/W	Reset	Description
31	RW	Rst110	**RGMII 125MHz clock source selection** 0: PAD_RGMIICK 1: Internal PLL
30	RW	Rst110	**RMII2 50MHz RCLK output enable** 0: Disable 1: Enable
29	RW	Rst110	**RMII1 50MHz RCLK output enable** 0: Disable 1: Enable
28	RW	Rst110	**RGMIICK pad direcetion** 0: Input mode 1: Output mode
27	RW	Rst110	**MAC#2 RMII transmit data at clock falling edge**
26	RW	Rst110	**MAC#1 RMII transmit data at clock falling edge**
25	RW	Rst110	**MAC#2 RXCLK clock tree invere phase**
24	RW	Rst110	**MAC#1 RXCLK clock tree invere phase**
23:18	RW	Rst110	**MAC#2 RMII_RCLK/RGMII_RXCLK (1G) clock input delay**
17:12	RW	Rst110	**MAC#1 RMII_RCLK/RGMII_RXCLK (1G) clock input delay**
11:6	RW	Rst110	**MAC#2 RGMII_TXCLK (1G) clock output delay**
5:0	RW	Rst110	**MAC#1 RGMII_TXCLK (1G) clock output delay**

Note :
The timing control block diagram are shown as Figure 46 and 45. This register can only be modified when MAC controller is at reset state (SCU040)

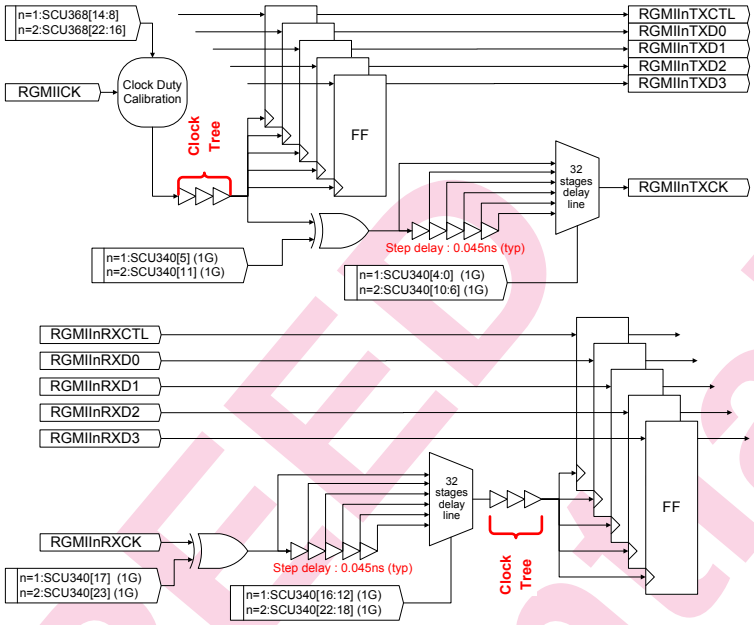


Figure 45: MAC1 and MAC2 RGMII timing control block diagram

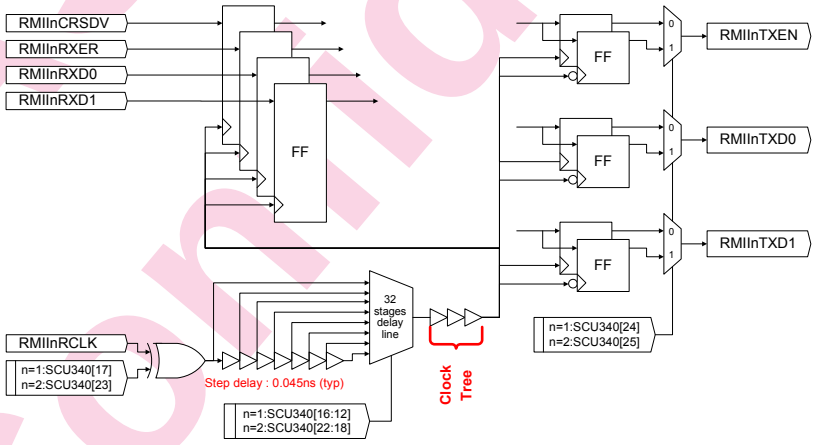


Figure 46: MAC1 and MAC2 RMII timing control block diagram

Offset: 348h	SCU348: MAC 12 Interface Clock Delay 100M Setting		Init = 0
Bit	**R/W**	**Reset**	**Description**
31:26	RO	-	**Reserved(0)**
25	RW	Rst111	**MAC#2 RXCLK 100M clock tree invere phase**
24	RW	Rst111	**MAC#1 RXCLK 100M clock tree invere phase**
23:18	RW	Rst111	**MAC#2 RGMII_RXCLK 100M clock input delay**
17:12	RW	Rst111	**MAC#1 RGMII_RXCLK 100M clock input delay**
11:6	RW	Rst111	**MAC#2 RGMII_TXCLK 100M clock output delay**
5:0	RW	Rst111	**MAC#1 RGMII_TXCLK 100M clock output delay**

Note :
This setting is a different option for RGMII 100M speed. Used when it requires different timing setting than SCU340 when working at 100M speed.

Offset: 34Ch	SCU34C: MAC 12 Interface Clock Delay 10M Setting		Init = 0
Bit	**R/W**	**Reset**	**Description**
31:26	RO	-	**Reserved(0)**
25	RW	Rst112	**MAC#2 RXCLK 10M clock tree invere phase**
24	RW	Rst112	**MAC#1 RXCLK 10M clock tree invere phase**
23:18	RW	Rst112	**MAC#2 RGMII_RXCLK 10M clock input delay**
17:12	RW	Rst112	**MAC#1 RGMII_RXCLK 10M clock input delay**
11:6	RW	Rst112	**MAC#2 RGMII_TXCLK 10M clock output delay**
5:0	RW	Rst112	**MAC#1 RGMII_TXCLK 10M clock output delay**

Note :
This setting is a different option for RGMII 10M speed. Used when it requires different timing setting than SCU340 when working at 10M speed.

Offset: 350h	SCU350: MAC 34 Interface Clock Delay Setting		Init = 0
Bit	**R/W**	**Reset**	**Description**
31	RW	Rst113	**RGMII 125MHz clock source selection** 0: PAD_RGMIICK 1: Internal PLL
30	RW	Rst113	**RMII4 50MHz RCLK output enable** 0: Disable 1: Enable
29	RW	Rst113	**RMII3 50MHz RCLK output enable** 0: Disable 1: Enable
28	RW	Rst113	**Reserved**
27	RW	Rst113	**MAC#4 RMII transmit data at clock falling edge**
26	RW	Rst113	**MAC#3 RMII transmit data at clock falling edge**
25	RW	Rst113	**MAC#4 RXCLK clock tree invere phase**
24	RW	Rst113	**MAC#3 RXCLK clock tree invere phase**
23:18	RW	Rst113	**MAC#4 RMII_RCLK/RGMII_RXCLK (1G) clock input delay**
17:12	RW	Rst113	**MAC#3 RMII_RCLK/RGMII_RXCLK (1G) clock input delay**
11:6	RW	Rst113	**MAC#4 RGMII_TXCLK (1G) clock output delay**

to next page

from previous page

5:0	RW	Rst113	**MAC#3 RGMII_TXCLK (1G) clock output delay**

Note :
The timing control block diagram are shown as Figure 48 and 47. This register can only be modified when MAC controller is at reset state (SCU040)

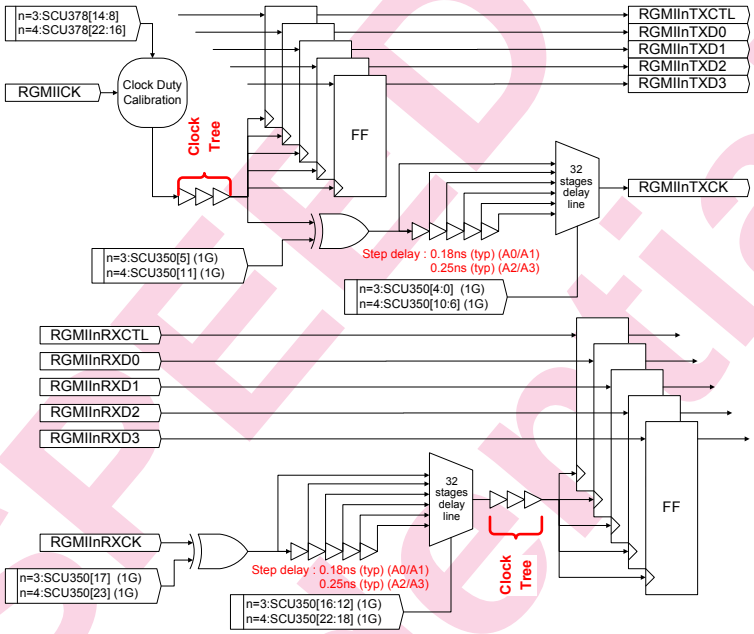


Figure 47: MAC3 and MAC4 RGMII timing control block diagram

Offset: 358h	SCU358: MAC 34 Interface Clock Delay 100M Setting	Init = 0

Bit	R/W	Reset	Description
31:26	RO	-	**Reserved(0)**
25	RW	Rst114	**MAC#4 RXCLK 100M clock tree invere phase**
24	RW	Rst114	**MAC#3 RXCLK 100M clock tree invere phase**
23:18	RW	Rst114	**MAC#4 RGMII_RXCLK 100M clock input delay**
17:12	RW	Rst114	**MAC#3 RGMII_RXCLK 100M clock input delay**
11:6	RW	Rst114	**MAC#4 RGMII_TXCLK 100M clock output delay**
5:0	RW	Rst114	**MAC#3 RGMII_TXCLK 100M clock output delay**

Note :
This setting is a different option for RGMII 100M speed. Used when it requires different timing setting than SCU48 when working at 100M speed.

Offset: 35Ch	SCU35C: MAC 34 Interface Clock Delay 10M Setting	Init = 0

Bit	R/W	Reset	Description
31:26	RO	-	**Reserved(0)**
25	RW	Rst115	**MAC#4 RXCLK 10M clock tree invere phase**

to next page

from previous page

24	RW	Rst115	**MAC#3 RXCLK 10M clock tree invere phase**
23:18	RW	Rst115	**MAC#4 RGMII_RXCLK 10M clock input delay**
17:12	RW	Rst115	**MAC#3 RGMII_RXCLK 10M clock input delay**
11:6	RW	Rst115	**MAC#4 RGMII_TXCLK 10M clock output delay**
5:0	RW	Rst115	**MAC#3 RGMII_TXCLK 10M clock output delay**

Note :
This setting is a different option for RGMII 10M speed. Used when it requires different timing setting than SCU48 when working at 10M speed.

Offset: 360h **SCU360: Clock Duty Measurement Control** **Init = 0**

Bit	R/W	Reset	Description
31:19	RO	-	**Reserved**
18	RO	-	**VPOCLK Duty count done status**
17	RO	-	**RGMII2TXCK Duty count done status**
16	RO	-	**RGMII1TXCK Duty count done status**
15:7	RO	-	**Reserved**
6:4	RW	Rst113	**Measurement result read select** 000: RGMII1TXCK 001: RGMII2TXCK 010: VPOCLK
3	RW	Rst113	**Start calculate duty**
2:1	RW	Rst113	**Ring clock select**
0	RW	Rst113	**Enable duty calculating ring clock**

Note :
The procedure for calculating duty:

1. Select duty control stage
2. Select ring clock bit[2:1]
3. Enable ring clock bit[0]=1
4. Wait 1 us for ring stable
5. Start calculate duty bit[3]=1
6. Wait duty count done
7. Stop calculate duty bit[3]=0
8. Repeat step 2 ~ step 7 at least 2 times and get the average duty value
9. Repeat step 1 ~ step 8 and get the maximum duty stage
10. Set the maximum duty stage to the duty control register

Before doing the duty calibration, the clock output delay must be fixed and can not modify after duty calibrated. If output delay is changed, duty must be calibrated again.

Offset: 364h **SCU364: Clock Duty Selection** **Init = 0**

Bit	R/W	Reset	Description
31:7	RW	Rst114	**Reserved**
6:0	RW	Rst114	**DCLK** bit[6]: enable duty calibration bit[5]: invert source clock bit[4:0]: select duty calibration stage

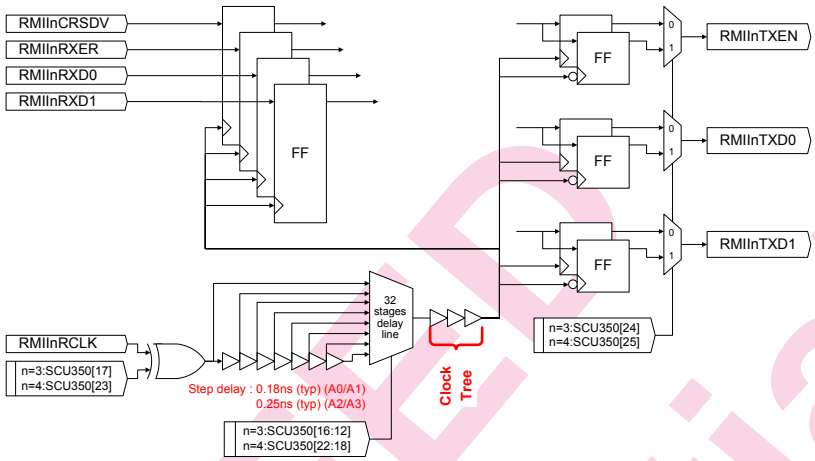


Figure 48: MAC3 and MAC4 RMII timing control block diagram

Offset: 368h	SCU368: Clock Duty Selection	Init = 0

Bit	R/W	Reset	Description
31:24	RO	-	**Reserved**
23	RW	Rst115	**Reserved**
22:16	RW	Rst115	**RGMII2TXCK**
15	RW	Rst115	**Reserved**
14:8	RW	Rst115	**RGMII1TXCK**
7	RW	Rst115	**Reserved**
6:0	RW	Rst115	**DCLK** bit[6]: enable duty calibration bit[5]: invert source clock bit[4:0]: select duty calibration stage

Offset: 36Ch	SCU36C: Clock Duty Measurement Result	Init = X

Bit	R/W	Reset	Description
29:16	RO	-	**N-phase counting value**
13:0	RO	-	**P-phase counting value**

Offset: 370h	SCU370: Clock Duty Measurement Control 2	Init = 0

Bit	R/W	Reset	Description
31:18	RO	-	**Reserved**
17	RO	-	**RGMII4TXCK Duty count done status**
16	RO	-	**RGMII3TXCK Duty count done status**
15:7	RO	-	**Reserved**
6:4	RW	Rst116	**Measurement result read select** 000: RGMII3TXCK 001: RGMII4TXCK
3	RW	Rst116	**Start calculate duty**

Offset: 374h	SCU374: Clock Duty Selection 2		Init = 0
Bit	**R/W**	**Reset**	**Description**
31:24	RO	-	**Reserved**
23	RW	Rst117	**Reserved**
22:16	RW	Rst117	**RGMII4TXCK**
15	RW	Rst117	**Reserved**
14:8	RW	Rst117	**RGMII3TXCK**
7:0	RW	Rst117	**Reserved**

Offset: 37Ch	SCU37C: Clock Duty Measurement Result 2		Init = X
Bit	**R/W**	**Reset**	**Description**
29:16	RO	-	**N-phase counting value for RGMII3TXCK or RGMII4TXCK**
13:0	RO	-	**P-phase counting value for RGMII3TXCK or RGMII4TXCK**

Offset: 400h	SCU400: Multi-function Pin Control #1		Init = 0
Bit	**R/W**	**Reset**	**Description**
31	RW	Rst124	**Enable EMMCWP# function pin**
30	RW	Rst124	**Enable EMMCCD# function pin**
29	RW	Rst124	**Enable EMMCDAT3 function pin**
28	RW	Rst124	**Enable EMMCDAT2 function pin**
27	RW	Rst124	**Enable EMMCDAT1 function pin**
26	RW	Rst124	**Enable EMMCDAT0 function pin**
25	RW	Rst124	**Enable EMMCCMD function pin**
24	RW	Rst124	**Enable EMMCCLK function pin**
23	RW	Rst124	**Enable RGMII2RXD3 function pin**
22	RW	Rst124	**Enable RGMII2RXD2 function pin**
21	RW	Rst124	**Enable RGMII2RXD1 function pin**
20	RW	Rst124	**Enable RGMII2RXD0 function pin**
19	RW	Rst124	**Enable RGMII2RXCTL function pin**
18	RW	Rst124	**Enable RGMII2RXCK function pin**
17	RW	Rst124	**Enable RGMII2TXD3 function pin**
16	RW	Rst124	**Enable RGMII2TXD2 function pin**
15	RW	Rst124	**Enable RGMII2TXD1 function pin**
14	RW	Rst124	**Enable RGMII2TXD0 function pin**
13	RW	Rst124	**Enable RGMII2TXCTL function pin**
12	RW	Rst124	**Enable RGMII2TXCK function pin**
11	RW	Rst124	**Enable RGMII1RXD3 function pin**
10	RW	Rst124	**Enable RGMII1RXD2 function pin**
9	RW	Rst124	**Enable RGMII1RXD1 function pin**
8	RW	Rst124	**Enable RGMII1RXD0 function pin**
7	RW	Rst124	**Enable RGMII1RXCTL function pin**
6	RW	Rst124	**Enable RGMII1RXCKRMII1RCLKI function pin**
5	RW	Rst124	**Enable RGMII1TXD3 function pin**

to next page

from previous page

4	RW	Rst124	**Enable RGMII1TXD2 function pin**
3	RW	Rst124	**Enable RGMII1TXD1RMII1TXD1 function pin**
2	RW	Rst124	**Enable RGMII1TXD0RMII1TXD0 function pin**
1	RW	Rst124	**Enable RGMII1TXCTLRMII1TXEN function pin**
0	RW	Rst124	**Enable RGMII1TXCKRMII1RCLKO function pin**

Offset: 404h **SCU404: Multi-function Pin Control #2** **Init = 0**

Bit	R/W	Reset	Description
31:7	RO	-	**Reserved**
6	RW	Rst125	**Enable GPIO18E3 as DP debug pin**
5	RW	Rst125	**Enable DDCDAT as DP debug pin**
4	RW	Rst125	**Enable DDCCLK as DP debug pin**
3	RW	Rst125	**Enable EMMCDAT7 function pin**
2	RW	Rst125	**Enable EMMCDAT6 function pin**
1	RW	Rst125	**Enable EMMCDAT5 function pin**
0	RW	Rst125	**Enable EMMCDAT4 function pin**

Offset: 40Ch **SCU40C: Multi-function Pin Control #3** **Init = 0**

Bit	R/W	Reset	Description
31:5	RO	-	**Reserved**
4	RW	Rst126	**Disable GPIO18E internal pull down resistor**
3	RW	Rst126	**Disable GPIO18D internal pull down resistor**
2	RW	Rst126	**Disable GPIO18C internal pull down resistor**
1	RW	Rst126	**Disable GPIO18B internal pull down resistor**
0	RW	Rst126	**Disable GPIO18A internal pull down resistor**

Offset: 410h **SCU410: Multi-function Pin Control #4** **Init = 0**

Bit	R/W	Reset	Description
31	RW	Rst127	**Enable NRI3 function pin**
30	RW	Rst127	**Enable NDSR3 function pin**
29	RW	Rst127	**Enable NDCD3 function pin**
28	RW	Rst127	**Enable NCTS3 function pin**
27	RW	Rst127	**Enable RGMII3RXD3 RMII3RXER function pin**
26	RW	Rst127	**Enable RGMII3RXD2 RMII3CRSDV function pin**
25	RW	Rst127	**Enable RGMII3RXD1 RMII3RXD1 function pin**
24	RW	Rst127	**Enable RGMII3RXD0 RMII3RXD0 function pin**
23	RW	Rst127	**Enable RGMII3RXCTL function pin**
22	RW	Rst127	**Enable RGMII3RXCK RMII3RCLKI function pin**
21	RW	Rst127	**Enable RGMII3TXD3 function pin**
20	RW	Rst127	**Enable RGMII3TXD2 function pin**
19	RW	Rst127	**Enable RGMII3TXD1 RMII3TXD1 function pin**
18	RW	Rst127	**Enable RGMII3TXD0 RMII3TXD0 function pin**

to next page

from previous page

17	RW	Rst127	**Enable RGMII3TXCTLRMII3TXEN function pin**
16	RW	Rst127	**Enable RGMII3TXCK RMII3RCLKO function pin**
15	RW	Rst127	**Enable RXD4 function pin**
14	RW	Rst127	**Enable TXD4 function pin**
13	RW	Rst127	**Enable MDIO2 function pin**
12	RW	Rst127	**Enable MDC2 function pin**
11	RW	Rst127	**Enable SALT4 function pin**
10	RW	Rst127	**Enable SALT3 function pin**
9	RW	Rst127	**Enable SALT2 function pin**
8	RW	Rst127	**Enable SALT1 function pin**
7	RW	Rst127	**Enable MAC4LINK function pin**
6	RW	Rst127	**Enable MAC3LINK function pin**
5	RW	Rst127	**Enable MAC2LINK function pin**
4	RW	Rst127	**Enable MAC1LINK function pin**
3	RW	Rst127	**Enable MDIO4 function pin**
2	RW	Rst127	**Enable MDC4 function pin**
1	RW	Rst127	**Enable MDIO3 function pin**
0	RW	Rst127	**Enable MDC3 function pin**

Offset: 414h **SCU414: Multi-function Pin Control #5** **Init = 0**

Bit	R/W	Reset	Description
31	RW	Rst128	**Enable SGPS1I1 function pin**
30	RW	Rst128	**Enable SGPS1I0 function pin**
29	RW	Rst128	**Enable SGPS1LD function pin**
28	RW	Rst128	**Enable SGPS1CK function pin**
27	RW	Rst128	**Enable SGPMI function pin**
26	RW	Rst128	**Enable SGPMO function pin**
25	RW	Rst128	**Enable SGPMLD function pin**
24	RW	Rst128	**Enable SGPMCK function pin**
23	RW	Rst128	**Enable RXD9 function pin**
22	RW	Rst128	**Enable TXD9 function pin**
21	RW	Rst128	**Enable RXD8 function pin**
20	RW	Rst128	**Enable TXD8 function pin**
19	RW	Rst128	**Enable RXD7 function pin**
18	RW	Rst128	**Enable TXD7 function pin**
17	RW	Rst128	**Enable RXD6 function pin**
16	RW	Rst128	**Enable TXD6 function pin**
15	RW	Rst128	**Enable SD1WP# function pin**
14	RW	Rst128	**Enable SD1CD# function pin**
13	RW	Rst128	**Enable SD1DAT3 function pin**
12	RW	Rst128	**Enable SD1DAT2 function pin**
11	RW	Rst128	**Enable SD1DAT1 function pin**

to next page

from previous page

10	RW	Rst128	**Enable SD1DAT0 function pin**
9	RW	Rst128	**Enable SD1CMD function pin**
8	RW	Rst128	**Enable SD1CLK function pin**
7	RW	Rst128	**Enable NRTS4 function pin**
6	RW	Rst128	**Enable NDTR4 function pin**
5	RW	Rst128	**Enable NRI4 function pin**
4	RW	Rst128	**Enable NDSR4 function pin**
3	RW	Rst128	**Enable NDCD4 function pin**
2	RW	Rst128	**Enable NCTS4 function pin**
1	RW	Rst128	**Enable NRTS3 function pin**
0	RW	Rst128	**Enable NDTR3 function pin**

Offset: 418h **SCU418: Multi-function Pin Control #6** **Init = 0xC0000000**

Bit	R/W	Reset	Description
31	RW	Rst129	**Enable VGAVS function pin**
30	RW	Rst129	**Enable VGAHS function pin**
29	RW	Rst129	**Enable RXD3 function pin**
28	RW	Rst129	**Enable TXD3 function pin**
27	RW	Rst129	**Enable I2C SDA10 function pin**
26	RW	Rst129	**Enable I2C SCL10 function pin**
25	RW	Rst129	**Enable I2C SDA9 function pin**
24	RW	Rst129	**Enable I2C SCL9 function pin**
23	RW	Rst129	**Enable I2C SDA8 function pin**
22	RW	Rst129	**Enable I2C SCL8 function pin**
21	RW	Rst129	**Enable I2C SDA7 function pin**
20	RW	Rst129	**Enable I2C SCL7 function pin**
19	RW	Rst129	**Enable I2C SDA6 function pin**
18	RW	Rst129	**Enable I2C SCL6 function pin**
17	RW	Rst129	**Enable I2C SDA5 function pin**
16	RW	Rst129	**Enable I2C SCL5 function pin**
15	RW	Rst129	**Enable HV6I3CSDA function pin**
14	RW	Rst129	**Enable HV6I3CSCL function pin**
13	RW	Rst129	**Enable HV5I3CSDA function pin**
12	RW	Rst129	**Enable HV5I3CSCL function pin**
11	RW	Rst129	**Enable HV4I3CSDA function pin**
10	RW	Rst129	**Enable HV4I3CSCL function pin**
9	RW	Rst129	**Enable HV3I3CSDA function pin**
8	RW	Rst129	**Enable HV3I3CSCL function pin**
7	RW	Rst129	**Enable BMC Interrupt function pin**
6	RW	Rst129	**Enable PBI# function pin**
5	RW	Rst129	**Enable PBO# function pin**
4	RW	Rst129	**Enable JTAG Master TDI function pin**

to next page

from previous page

3	RW	Rst129	**Enable JTAG Master TMS function pin**
2	RW	Rst129	**Enable JTAG Master TCK function pin**
1	RW	Rst129	**Enable JTAG Master TDO function pin**
0	RW	Rst129	**Enable JTAG Master TRSTN function pin**

Offset: 41Ch		**SCU41C: Multi-function Pin Control #7**	**Init = 0xC0**
Bit	**R/W**	**Reset**	**Description**
31	RW	Rst130	**Enable PWM15 function pin**
30	RW	Rst130	**Enable PWM14 function pin**
29	RW	Rst130	**Enable PWM13 function pin**
28	RW	Rst130	**Enable PWM12 function pin**
27	RW	Rst130	**Enable PWM11 function pin**
26	RW	Rst130	**Enable PWM10 function pin**
25	RW	Rst130	**Enable PWM9 function pin**
24	RW	Rst130	**Enable PWM8 function pin**
23	RW	Rst130	**Enable PWM7 function pin**
22	RW	Rst130	**Enable PWM6 function pin**
21	RW	Rst130	**Enable PWM5 function pin**
20	RW	Rst130	**Enable PWM4 function pin**
19	RW	Rst130	**Enable PWM3 function pin**
18	RW	Rst130	**Enable PWM2 function pin**
17	RW	Rst130	**Enable PWM1 function pin**
16	RW	Rst130	**Enable PWM0 function pin**
15	RW	Rst130	**Enable RXD2 function pin**
14	RW	Rst130	**Enable TXD2 function pin**
13	RW	Rst130	**Enable NRTS2 function pin**
12	RW	Rst130	**Enable NDTR2 function pin**
11	RW	Rst130	**Enable NRI2 function pin**
10	RW	Rst130	**Enable NDSR2 function pin**
9	RW	Rst130	**Enable NDCD2 function pin**
8	RW	Rst130	**Enable NCTS2 function pin**
7	RW	Rst130	**Enable RXD1 function pin**
6	RW	Rst130	**Enable TXD1 function pin**
5	RW	Rst130	**Enable NRTS1 function pin**
4	RW	Rst130	**Enable NDTR1 function pin**
3	RW	Rst130	**Enable NRI1 function pin**
2	RW	Rst130	**Enable NDSR1 function pin**
1	RW	Rst130	**Enable NDCD1 function pin**
0	RW	Rst130	**Enable NCTS1 function pin**

Offset: 430h	SCU430: Multi-function Pin Control #8	Init = 0

Bit	R/W	Reset	Description
31	RW	Rst131	**Enable GPIT7 function pin**
30	RW	Rst131	**Enable GPIT6 function pin**
29	RW	Rst131	**Enable GPIT5 function pin**
28	RW	Rst131	**Enable GPIT4 function pin**
27	RW	Rst131	**Enable GPIT3 function pin**
26	RW	Rst131	**Enable GPIT2 function pin**
25	RW	Rst131	**Enable GPIT1 function pin**
24	RW	Rst131	**Enable GPIT0 function pin**
23	RW	Rst131	**Enable RXD11 function pin**
22	RW	Rst131	**Enable TXD11 function pin**
21	RW	Rst131	**Enable RXD10 function pin**
20	RW	Rst131	**Enable TXD10 function pin**
19	RW	Rst131	**Enable OSCCLK function pin**
18	RW	Rst131	**Enable PEWAKE# function pin**
17	RW	Rst131	**Enable MDIO function pin**
16	RW	Rst131	**Enable MDC function pin**
15	RW	Rst131	**Enable TACH15 function pin**
14	RW	Rst131	**Enable TACH14 function pin**
13	RW	Rst131	**Enable TACH13 function pin**
12	RW	Rst131	**Enable TACH12 function pin**
11	RW	Rst131	**Enable TACH11 function pin**
10	RW	Rst131	**Enable TACH10 function pin**
9	RW	Rst131	**Enable TACH9 function pin**
8	RW	Rst131	**Enable TACH8 function pin**
7	RW	Rst131	**Enable TACH7 function pin**
6	RW	Rst131	**Enable TACH6 function pin**
5	RW	Rst131	**Enable TACH5 function pin**
4	RW	Rst131	**Enable TACH4 function pin**
3	RW	Rst131	**Enable TACH3 function pin**
2	RW	Rst131	**Enable TACH2 function pin**
1	RW	Rst131	**Enable TACH1 function pin**
0	RW	Rst131	**Enable TACH0 function pin**

Offset: 434h	SCU434: Multi-function Pin Control #9	Init = 0xFF0000

Bit	R/W	Reset	Description
31	RW	Rst132	**Enable SPI2DQ3 function pin**
30	RW	Rst132	**Enable SPI2DQ2 function pin**
29	RW	Rst132	**Enable SPI2MISO function pin**
28	RW	Rst132	**Enable SPI2MOSI function pin**
27	RW	Rst132	**Enable SPI2CK function pin**
26	RW	Rst132	**Enable SPI2CS2# function pin**

to next page

from previous page

25	RW	Rst132	**Enable SPI2CS1# function pin**
24	RW	Rst132	**Enable SPI2CS0# function pin**
23	RW	Rst132	**Enable LPCRSTN/ESPIRSTN function pin**
22	RW	Rst132	**Enable LPCSIRQ/ESPIALERT function pin**
21	RW	Rst132	**Enable LFRAMEN/ESPICS# function pin**
20	RW	Rst132	**Enable LPCCLK/ESPICLK function pin**
19	RW	Rst132	**Enable LAD3/ESPID3 function pin**
18	RW	Rst132	**Enable LAD2/ESPID2 function pin**
17	RW	Rst132	**Enable LAD1/ESPID1 function pin**
16	RW	Rst132	**Enable LAD0/ESPID0 function pin**
15	RW	Rst132	**Enable LPCSMI# function pin**
14	RW	Rst132	**Enable LPCPME# function pin**
13	RW	Rst132	**Enable LPCPD# function pin**
12	RW	Rst132	**Enable SIOPWRGD function pin**
11	RW	Rst132	**Enable SIOONCTRL# function pin**
10	RW	Rst132	**Enable SIOPWREQ# function pin**
9	RW	Rst132	**Enable SIOS5# function pin**
8	RW	Rst132	**Enable SIOS3# function pin**
7	RW	Rst132	**Enable GPIU7/SALT16(AC17) function pin**
6	RW	Rst132	**Enable GPIU6/SALT15(AD16) function pin**
5	RW	Rst132	**Enable GPIU5/SALT14(AA16) function pin**
4	RW	Rst132	**Enable GPIU4/SALT13(AC16) function pin**
3	RW	Rst132	**Enable GPIU3/SALT12(AE16) function pin**
2	RW	Rst132	**Enable GPIU2/SALT11(AB17) function pin**
1	RW	Rst132	**Enable GPIU1/SALT10(AA17) function pin**
0	RW	Rst132	**Enable GPIU0/SALT9(AB16) function pin**

Offset: 438h **SCU438: Multi-function Pin Control #10** **Init = 0x3800**

Bit	R/W	Reset	Description
31:28	RO	-	**Reserved**
27	RW	-	**Reserved**
26	RW	Rst133	**Enable GPIO Passthrough 2 de-bounce**
25	RW	Rst133	**Enable GPIO Passthrough 1 de-bounce**
24	RW	Rst133	**Enable GPIO Passthrough 0 de-bounce**
23	RW	Rst133	**Enable I3C4SDA function pin**
22	RW	Rst133	**Enable I3C4SCL function pin**
21	RW	Rst133	**Enable I3C3SDA function pin**
20	RW	Rst133	**Enable I3C3SCL function pin**
19	RW	Rst133	**Enable I3C2SDA function pin**
18	RW	Rst133	**Enable I3C2SCL function pin**
17	RW	Rst133	**Enable I3C1SDA function pin**
16	RW	Rst133	**Enable I3C1SCL function pin**

to next page

from previous page

15	RW	Rst133	**Enable SPI1DQ3 function pin**
14	RW	Rst133	**Enable SPI1DQ2 function pin**
13	RW	Rst133	**Enable SPI1MISO function pin**
12	RW	Rst133	**Enable SPI1MOSI function pin**
11	RW	Rst133	**Enable SPI1CK function pin**
10	RW	Rst133	**Enable SPI1WP# function pin**
9	RW	Rst133	**Enable SPI1ABR function pin**
8	RW	Rst133	**Enable SPI1CS1# function pin**
7	RW	Rst133	**Enable FWSPIWP# function pin**
6	RW	Rst133	**Enable FWSPIABR function pin**
5	RW	Rst133	**Enable FWSPIDQ3 function pin**
4	RW	Rst133	**Enable FWSPIDQ2 function pin**
3	RW	Rst133	**Enable SALT8 function pin**
2	RW	Rst133	**Enable SALT7 function pin**
1	RW	Rst133	**Enable SALT6 function pin**
0	RW	Rst133	**Enable SALT5 function pin**

Offset: 440h **SCU440: USB Multi-function Pin Control #12** **Init = 0**

Bit	R/W	Reset	Description
31:30	RW	Rst135	**Rederved (0)**
29:28	RW	Rst135	**Select USB Port2 mode** 00: USB1.1 HID controller 01: USB2.0 Device controller 10: BMC EHCI port2
27:26	RW	Rst135	**Select USB Port1 PCIe EHCI to Hub path transfer rate** 00: 240 MHz 01: 120 MHz 10: 80 MHz 11: 60 MHz
25:24	RW	Rst135	**Select USB Port1 mode** 00: PCIe EHCI to Hub 01: Hub to PHY 10: BMC EHCI to PHY 11: PCIe EHCI to PHY
23:0	RW	Rst135	**Rederved (0)**

Offset: 450h **SCU450: Multi-function Pin Control #14** **Init = 0xA000**

Bit	R/W	Reset	Description
31	RW	Rst136	**Enable I2C16**
30	RW	Rst136	**Enable I2C15**
29	RW	Rst136	**Enable I2C14**
28	RW	Rst136	**Enable I2C13**
27	RW	Rst136	**Enable I2C12**
26	RW	Rst136	**Enable I2C11**

to next page

from previous page

25	RW	Rst136	**Enable I2C10**
24	RW	Rst136	**Enable I2C9**
23	RW	Rst136	**Enable I2C8**
22	RW	Rst136	**Enable I2C7**
21	RW	Rst136	**Enable I2C6**
20	RW	Rst136	**Enable I2C5**
19	RW	Rst136	**Enable I2C4**
18	RW	Rst136	**Enable I2C3**
17	RW	Rst136	**Enable I2C2**
16	RW	Rst136	**Enable I2C1**
15:4	RW	Rst136	**Reserved**
3	RW	Rst136	**Enable SD1 8-bit mode**
2	RW	Rst136	**Reserved**
1	RW	Rst136	**Enable SD2 function**
0	RW	Rst136	**Enable SD1 8-bit mode** To enable SD1 8-bit mode, set both bit 0 and 3 to 1.

Offset: 454h **SCU454: Multi-function Pin Control #15** **Init = 0**

Bit	R/W	Reset	Description
31:30	RW	Rst137	**LAD3/ESPID3 Driving Strength**
29:28	RW	Rst137	**LAD2/ESPID2 Driving Strength**
27:26	RW	Rst137	**LAD1/ESPID1 Driving Strength**
25:24	RW	Rst137	**LAD0/ESPID0 Driving Strength** 3 : strongest 0 : weakest.
23:4	RW	Rst137	**Reserved**
3	RW	Rst137	**Enable UART4**
2	RW	Rst137	**Enable UART3**
1:0	RW	Rst137	**Reserved**

Offset: 458h **SCU458: Multi-function Pin Control #16** **Init = 0x000500**

Bit	R/W	Reset	Description
31:21	RO	-	**Reserved**
20	RW	Rst138	**Voltage of GPIOW except SCU510[6] is 0.** 0 : 3.3v 1 : 1.8v
19:18	RW	Rst137	**LPCRSTN/ESPIRSTN Driving Strength**
17:16	RW	Rst137	**LADSIRQ/ESPIALERT Driving Strength**
15:14	RW	Rst137	**LPCFRAMEN/ESPICSN Driving Strength**
13:12	RW	Rst137	**LPCCLK/ESPICLK Driving Strength** 3 : strongest 0 : weakest.
11:10	RW	Rst138	**Driving Strength of GPIOG except neither SCU450[1:0] nor SCU450[3] is set.** 0 : weakest 3 : strongest

to next page

from previous page

9: 8	RW	Rst138	**Driving Strength of GPIOF except any of SCU414[15:8] is set.** 0 : weakest 3 : strongest
7	RW	Rst138	**Voltage of GPIOG except neither SCU450[1:0] nor SCU450[3] is set.** 0 : 3.3v 1 : 1.8v
6	RW	Rst138	**Voltage of GPIOF except any of SCU414[15:8] is set.** 0 : 3.3v 1 : 1.8v
5	RW	Rst138	**MAC#4 Voltage**
4	RW	Rst138	**MAC#3 Voltage** 1: 1.8V 0: 3.3V
3:2	RW	Rst138	**MAC#4 Driving Strength** 0 : weakest 3 : strongest
1:0	RW	Rst138	**MAC#3 Driving Strength** 0 : weakest 3 : strongest

Offset: 470h **SCU470: Multi-function Pin Control** **Init = 0**

Bit	R/W	Reset	Description
31:24	RW	Rst139	**Reserved**
23	RW	Rst139	**Disable NRTS4RGMII4RXD3RMII4RXER function pin**
22	RW	Rst139	**Disable NDTR4RGMII4RXD2RMII4CRSDV function pin**
21	RW	Rst139	**Disable NRI4RGMII4RXD1RMII4RXD1 function pin**
20	RW	Rst139	**Disable NDSR4RGMII4RXD0RMII4RXD0 function pin**
19	RW	Rst139	**Disable NDCD4RGMII4RXCTL function pin**
18	RW	Rst139	**Disable NCTS4RGMII4RXCKRMII4RCLKI function pin**
17	RW	Rst139	**Disable NRTS3RGMII4TXD3 function pin**
16	RW	Rst139	**Disable NDTR3RGMII4TXD2 function pin**
15: 0	RW	Rst139	**Reserved**

Offset: 4B0h **SCU4B0: Multi-function Pin Control #17** **Init = 0**

Bit	R/W	Reset	Description
31	RW	Rst140	**Enable RGMII4TXD1 RMII4TXD1 function pin**
30	RW	Rst140	**Enable RGMII4TXD0 RMII4TXD0 function pin**
29	RW	Rst140	**Enable RGMII4TXCTLRMII4TXEN function pin**
28	RW	Rst140	**Enable RGMII4TXCK RMII4RCLKO function pin**
27:8	RW	Rst140	**Reserved**
7	RW	Rst140	**Enable SDA14 function pin**
6	RW	Rst140	**Enable SCL14 function pin**
5	RW	Rst140	**Enable SDA13 function pin**
4	RW	Rst140	**Enable SCL13 function pin**
3	RW	Rst140	**Enable SDA12 function pin**

to next page

from previous page

2	RW	Rst140	**Enable SCL12 function pin**
1	RW	Rst140	**Enable SDA11 function pin**
0	RW	Rst140	**Enable SCL11 function pin**

Offset: 4B4h		**SCU4B4: Multi-function Pin Control #18**	**Init = 0**
Bit	**R/W**	**Reset**	**Description**
31	RW	Rst141	**Enable SDA16 function pin**
30	RW	Rst141	**Enable SCL16 function pin**
29	RW	Rst141	**Enable SDA15 function pin**
28	RW	Rst141	**Enable SCL15 function pin**
27:24	RW	Rst141	**Reserved**
23	RW	Rst141	**Enable SD2WP# function pin**
22	RW	Rst141	**Enable SD2CD# function pin**
21	RW	Rst141	**Enable SD2DAT3 function pin**
20	RW	Rst141	**Enable SD2DAT2 function pin**
19	RW	Rst141	**Enable SD2DAT1 function pin**
18	RW	Rst141	**Enable SD2DAT0 function pin**
17	RW	Rst141	**Enable SD2CMD function pin**
16	RW	Rst141	**Enable SD2CLK function pin**
15	RW	Rst141	**Enable PWM15 function pin**
14	RW	Rst141	**Enable PWM14 function pin**
13	RW	Rst141	**Enable PWM13 function pin**
12	RW	Rst141	**Enable PWM12 function pin**
11	RW	Rst141	**Enable PWM11 function pin**
10	RW	Rst141	**Enable PWM10 function pin**
9	RW	Rst141	**Enable PWM9 function pin**
8	RW	Rst141	**Enable PWM8 function pin**
7	RW	Rst141	**Enable RGMII4RXD3 RMII4RXER function pin**
6	RW	Rst141	**Enable RGMII4RXD2 RMII4CRSDV function pin**
5	RW	Rst141	**Enable RGMII4RXD1 RMII4RXD1 function pin**
4	RW	Rst141	**Enable RGMII4RXD0 RMII4RXD0 function pin**
3	RW	Rst141	**Enable RGMII4RXCTL function pin**
2	RW	Rst141	**Enable RGMII4RXCK RMII4RCLKI function pin**
1	RW	Rst141	**Enable RGMII4TXD3 function pin**
0	RW	Rst141	**Enable RGMII4TXD2 function pin**

Offset: 4B8h		**SCU4B8: Multi-function Pin Control #19**	**Init = 0**
Bit	**R/W**	**Reset**	**Description**
31:28	RW	Rst142	**Reserved**
27	RW	Rst142	**Enable I2CS SDAS2 to GPIOL[3]**
26	RW	Rst142	**Enable I2CS SCLS2 to GPIOL[2]**
25	RW	Rst142	**Enable I2CS SDAS1 to GPIOL[1]**

to next page

from previous page

24	RW	Rst142	**Enable I2CS SCLS1 to GPIOL[0]**
23	RW	Rst142	**Enable I2CS SDAS4 to GPIOK[7]**
22	RW	Rst142	**Enable I2CS SCLS4 to GPIOK[6]**
21	RW	Rst142	**Enable I2CS SDAS3 to GPIOK[5]**
20	RW	Rst142	**Enable I2CS SCLS3 to GPIOK[4]**
19	RW	Rst142	**Enable I2CS SDAS2 to GPIOK[3]**
18	RW	Rst142	**Enable I2CS SCLS2 to GPIOK[2]**
17	RW	Rst142	**Enable I2CS SDAS1 to GPIOK[1]**
16	RW	Rst142	**Enable I2CS SCLS1 to GPIOK[0]**
15	RW	Rst142	**Enable I2C SDA4 function pin**
14	RW	Rst142	**Enable I2C SCL4 function pin**
13	RW	Rst142	**Enable I2C SDA3 function pin**
12	RW	Rst142	**Enable I2C SCL3 function pin**
11	RW	Rst142	**Enable I2C SDA2 function pin**
10	RW	Rst142	**Enable I2C SCL2 function pin**
9	RW	Rst142	**Enable I2C SDA1 function pin**
8	RW	Rst142	**Enable I2C SCL1 function pin**
7:4	RW	Rst142	**Reserved**
3	RW	Rst142	**Enable RXD13 function pin**
2	RW	Rst142	**Enable TXD13 function pin**
1	RW	Rst142	**Enable RXD12 function pin**
0	RW	Rst142	**Enable TXD12 function pin**

Offset: 4BCh **SCU4BC: Multi-function Pin Control #20** **Init = 0**

Bit	R/W	Reset	Description
31	RW	-	**Reserved**
30	RW	-	**Reserved**
29	RW	Rst143	**Enable GPIO Passthrough 2 Output (GPIOP5) function pin**
28	RW	Rst143	**Enable GPIO Passthrough 2 Input (GPIOP4) function pin**
27	RW	Rst143	**Enable GPIO Passthrough 1 Output (GPIOP3) function pin**
26	RW	Rst143	**Enable GPIO Passthrough 1 Input (GPIOP2) function pin**
25	RW	Rst143	**Enable GPIO Passthrough 0 Output (GPIOP1) function pin**
24	RW	Rst143	**Enable GPIO Passthrough 0 Input (GPIOP0) function pin**
23:0	RW	Rst143	**Reserved**

Offset: 4D4h **SCU4D4: Multi-function Pin Control #22** **Init = 0**

Bit	R/W	Reset	Description
31	RW	Rst144	**Enable RXD12 function pin**
30	RW	Rst144	**Enable TXD12 function pin**
29:8	RW	Rst144	**Reserved**
7	RW	Rst144	**Select GPIU7 or SALT16(AC17) function pin**
6	RW	Rst144	**Select GPIU6 or SALT15(AD16) function pin**

to next page

from previous page

5	RW	Rst144	**Select GPIU5 or SALT14(AA16) function pin**
4	RW	Rst144	**Select GPIU4 or SALT13(AC16) function pin**
3	RW	Rst144	**Select GPIU3 or SALT12(AE16) function pin**
2	RW	Rst144	**Select GPIU2 or SALT11(AB17) function pin**
1	RW	Rst144	**Select GPIU1 or SALT10(AA17) function pin**
0	RW	Rst144	**Select GPIU0 or SALT9(AB16) function pin**

Offset: 4D8h **SCU4D8: Multi-function Pin Control #23** **Init = 0**

Bit	R/W	Reset	Description
31:28	RO	Rst145	**Reserved**
27:24	RW	Rst145	**Reserved**
23	RW	Rst145	**Enable FSI2DATA function pin**
22	RW	Rst145	**Enable FSI2CLK function pin**
21	RW	Rst145	**Enable FSI1DATA function pin**
20	RW	Rst145	**Enable FSI1CLK function pin**
19:16	RW	Rst145	**Reserved**
15	RW	Rst145	**Enable RXD13 function pin**
14	RW	Rst145	**Enable TXD13 function pin**
13:4	RW	Rst145	**Reserved**
3	RW	Rst145	**Enable WDT 4 Reset Output function pin**
2	RW	Rst145	**Enable WDT 3 Reset Output function pin**
1	RW	Rst145	**Enable WDT 2 Reset Output function pin**
0	RW	Rst145	**Enable WDT 1 Reset Output function pin**

Offset: 690h **SCU690: Multi-function Pin Control #24** **Init = 0**

Bit	R/W	Reset	Description
31:8	RW	Rst147	**Reserved**
7	RW	Rst147	**Enable SGPS2I1 function pin**
6	RW	Rst147	**Enable SGPS2I0 function pin**
5	RW	Rst147	**Enable SGPS2LD function pin**
4	RW	Rst147	**Enable SGPS2CK function pin**
3:0	RW	Rst147	**Reserved**
2	RW	Rst147	**Enable SCL12 function pin**
1	RW	Rst147	**Enable SDA11 function pin**
0	RW	Rst147	**Enable SCL11 function pin**

Offset: 694h **SCU694: Multi-function Pin Control #25** **Init = 0**

Bit	R/W	Reset	Description
31	RW	Rst147	**Enable I2CS SDAS4 to GPIOH[7]**
30	RW	Rst147	**Enable I2CS SCLS4 to GPIOH[6]**
29	RW	Rst147	**Enable I2CS SDAS3 to GPIOH[5]**
28	RW	Rst147	**Enable I2CS SCLS3 to GPIOH[4]**

to next page

from previous page

27:24	RW	Rst148	**Reserved**
23	RW	Rst147	**Select SALT16 ball assign** 1: B21 0: AC17
22	RW	Rst147	**Select SALT15 ball assign** 1: D21 0: AD16
21	RW	Rst147	**Select SALT14 ball assign** 1: E20 0: AA16
20	RW	Rst147	**Select SALT13 ball assign** 1: A21 0: AC16
19	RW	Rst147	**Select SALT12 ball assign** 1: A22 0: AE16
18	RW	Rst147	**Select SALT11 ball assign** 1: C21 0: AB17
17	RW	Rst147	**Select SALT10 ball assign** 1: B22 0: AA17
16	RW	Rst147	**Select SALT9 ball assign** 1: E21 0: AB16
15:0	RW	Rst147	**Reserved**
Note : Besides selecting ball assign, other pinmux conditions must also be met. Please refer to chapter of Multifunction.			

Offset: 698h		**SCU698: Multi-function Pin Control #26**	**Init = 0**
Bit	**R/W**	**Reset**	**Description**
31:16	RW	Rst149	**Reserved**
15	RW	Rst149	**Enable I2CS SDAS4 to GPIOJ[7]**
14	RW	Rst149	**Enable I2CS SCLS4 to GPIOJ[6]**
13	RW	Rst149	**Enable I2CS SDAS3 to GPIOJ[5]**
12	RW	Rst149	**Enable I2CS SCLS3 to GPIOJ[4]**
11	RW	Rst149	**Enable I2CS SDAS2 to GPIOJ[3]**
10	RW	Rst149	**Enable I2CS SCLS2 to GPIOJ[2]**
9	RW	Rst149	**Enable I2CS SDAS1 to GPIOJ[1]**
8	RW	Rst149	**Enable I2CS SCLS1 to GPIOJ[0]**
7: 0	RW	Rst149	**Reserved**

Offset: 69Ch	SCU69C: Multi-function Pin Control #27	Init = 0x80000000

Bit	R/W	Reset	Description
31	RW	Rst150	**Enable HeartBeat LED**
30: 0	RO	-	**Reserved(0)**

Offset: 6B8h	SCU6B8: Multi-function Pin Control #28	Init = 0x0

Bit	R/W	Reset	Description
31: 4	RO	-	**Reserved(0)**
3	RW	Rst231	**Enable SMBusS Alert 3 to GPIOY[3]**
2	RW	Rst231	**Enable SMBusS Alert 2 to GPIOY[2]**
1	RW	Rst231	**Enable SMBusS Alert 1 to GPIOY[1]**
0	RW	Rst231	**Enable SMBusS Alert 0 to GPIOY[0]**

Offset: 6D0h	SCU6D0: Multi-function Pin Control #29	Init = 0

Bit	R/W	Reset	Description
31:8	RO	-	**Reserved**
6: 4	RW	Rst151	**Enable SGPIO to GPIOA[7:4]**
3: 0	RO	-	**Reserved(0)**

Offset: 6D4h	SCU6D4: Multi-function Pin Control #30	Init = 0x0

Bit	R/W	Reset	Description
31:24	RO	-	**Reserved(0)**
23	RW	Rst232	**Enable SMBusS Alert 3 to GPIOG[7]**
22	RW	Rst232	**Enable SMBusS Alert 2 to GPIOG[6]**
21	RW	Rst232	**Enable SMBusS Alert 1 to GPIOG[5]**
20	RW	Rst232	**Enable SMBusS Alert 0 to GPIOG[4]**
19	RW	Rst232	**Enable SMBusS Alert 3 to GPIOG[3]**
18	RW	Rst232	**Enable SMBusS Alert 2 to GPIOG[2]**
17	RW	Rst232	**Enable SMBusS Alert 1 to GPIOG[1]**
16	RW	Rst232	**Enable SMBusS Alert 0 to GPIOG[0]**
15: 0	RO	-	**Reserved(0)**

Offset: 710h	SCU710: Multi-function Pin Control #31	Init = 0x0

Bit	R/W	Reset	Description
31: 8	RO	-	**Reserved(0)**
7	RW	Rst233	**Enable I2CS SDAS2 to GPIOA[7]**
6	RW	Rst233	**Enable I2CS SCLS2 to GPIOA[6]**
5	RW	Rst233	**Enable I2CS SDAS1 to GPIOA[5]**
4	RW	Rst233	**Enable I2CS SCLS1 to GPIOA[4]**
3	RW	Rst233	**Enable I2CS SDAS4 to GPIOA[3]**
2	RW	Rst233	**Enable I2CS SCLS4 to GPIOA[2]**
1	RW	Rst233	**Enable I2CS SDAS3 to GPIOA[1]**
0	RW	Rst233	**Enable I2CS SCLS3 to GPIOA[0]**

Offset: 4F4h	SCU4F4: UART Debug interface Baud Rate Control	Init = 0x00600001

Bit	R/W	Reset	Description
31:16	RW	Rst146	**Baud Rate divisor of password phase**
15: 0	RW	Rst146	**Baud Rate divisor of normal phase**

Offset: 500h	SCU500: Hardware Strap1 Register	Init = PIN/OTP

Bit	R/W	Reset	Description
31	RW1S	RstPwr	**Enable boot SPI auxiliary control pins (FWSPIDQ2, FWSPIDQ3, FWSPIABR, FWSPIWP#) (OTPSTRAP[54])** 0: Disable (default) 1: Enable This is mirror register of SCU510[22]. This strap bit will enable pins FWSPIDQ2, FWSPIDQ3, FWSPIABR, FWSPIWP# and related functions automatically.
30	RW1S	RstPwr	**Disable RVAS function (OTPSTRAP[29])** 0: Enable RVAS (default) 1: Disable RVAS
29:28	RW1S	RstPwr	**Select Reset Source of eMMC part** 00b : GPIOY3. x1b : GPIO18A2. 10b : GPIO18B6. When booting from eMMC, SBMCU will base on this selection to toggle corresponding GPIO to reset eMMC part. If BMC FW hopes to reset eMMC part after booting, selecting them as GPIO and control pin state directly.
27:26	RW1S	RstPwr	**Internal Bridge Speed Selection (OTPSTRAP[26:25])** 00: 1x (default) 01: 1/2x 10: 1/4x 11: 1/8x Recommend to keep the default setting. Other settings may cause system unstable.
25	RW1S	RstPwr	**Disable watchdog to reset full chip (OTPSTRAP[24])** 0: Full watchdog mode (default) 1: Disable watchdog system reset signal The system reset signal of WDT controller is used to reset full chip. Sometimes it will cause system unstable or security issue due to all registers are back to power on reset state. It is used when software confirms to assert a whole chip reset event and back to state as power on reset. Set this bit will disable it and prevent unpredictable reset results.
24	RW1S	RstPwr	**Dedicate PCIe root complex reset (OTPSTRAP[23])** 0: SSPRST# pin is for secondary processor dedicated reset pin (default) 1: SSPRST# pin is for PCIe root complex dedicated reset pin Sometimes for PCIe root complex application, it needs a dedicated reset signal (RCRST#) to PCIe devices. Set this bit will change SSPRST# to RCRST#.
23	RW1S	RstPwr	**Disable Dedicated BMC function (OTPSTRAP[22])** 0: Normal BMC mode (default) 1: Disable dedicated BMC functions for non-BMC application AST2600 can be used on non-BMC application. When this bit is set, the major effect is to cause VGA output failure.

to next page

from previous page

22	RW1S	RstPwr	**Disable ARM JTAG trust world debug (OTPSTRAP[21])** 0: Enable ARM JTAG trust world debug (default) 1: Disable ARM JTAG trust world debug This bit will disable ARM CA7 trust world debug through JTAG interface. ARM CA7 non-trust world and CM3 debug JTAG interfacse still can work when SCU500[16] is not set.
21	RW1S	RstPwr	**Enable PCIe EHCI (OTPSTRAP[20])** 0: Disable EHCI device (default) 1: Enable EHCI device The EHCI device can be enabled by this bit or by SCUC20[16].
20	RW1S	RstPwr	**Secure Boot or Boot from eMMC speed mode (OTPSTRAP[19])** 0: Normal speed (default) 1: High speed The SPI secure boot frequency of normal speed is 25MHz. The SPI secure boot frequency of high speed is 100MHz. The eMMC boot frequency of normal speed is 25MHz. The eMMC boot frequency of high speed is 50MHz.
19	RW1S	RstPwr	**Disable Debug Interfaces (OTPSTRAP[18])** 0: Keep normal states for debug interfaces (default) 1: Disable debug interfaces This bit will disable all debug mode paths that are listed in SCU0C8 and is not possible to enable by other registers after power on. When this bit is not set, user still can program SCU0C8 to disable each one of the debug interfaces. It is not recommended during development period because it will cause developer almost impossible to diagnose hardware issues.
18	RW1S	RstPwr	**VGA Class Code selection (OTPSTRAP[17])** 0: Select the Class Code for VGA device (default) 1: Select the Class Code for video device
17	RW1S	RstPwr	**Disable ARM JTAG debug (OTPSTRAP[16])** 0: Enable ARM JTAG debug (default) 1: Disable ARM JTAG debug This bit will disable ARM JTAG debug interface for CA7 and CM3. It is not recommended during development period.
16	RW1S	RstPwr	**CPU/AXI clock frequency ratio selection (OTPSTRAP[15])** 0: Select CPU:AXI = 2:1 (default) 1: Select CPU:AXI = 1:1 Recommend to keep the default setting. Other settings may cause system unstable.
15	RW1S	RstPwr	**Reserved (0) (OTPSTRAP[14])**

to next page

from previous page

14:13	RW1S	RstPwr	**VGA memory size selection (OTPSTRAP[13:12])** 00: Select 8 MB VGA memory (default) 01: Select 16 MB VGA memory 10: Select 32 MB VGA memory 11: Select 64 MB VGA memory Defined the VGA memory size that will share with SOC memory. The minimum memory size required for the VGA high resolution mode shows as below: 1280x1024x16bpp = 8MB 1600x1200x16bpp = 8MB 1680x1050x16bpp = 8MB 1920x1080x16bpp = 8MB 1920x1200x16bpp = 8MB 1280x1024x32bpp = 8MB 1600x1200x32bpp = 8MB 1680x1050x32bpp = 8MB 1920x1080x32bpp = 16MB 1920x1200x32bpp = 16MB Here recommend to set the VGA memory size to 16MB otherwise VGA resolution modes will be limited. ASPEED BSP will detect and set the VGA size to 16MB when the VGA memory size is default value, 8MB.
12:11	RW1S	RstPwr	**AXI/AHB clock frequency ratio selection (OTPSTRAP[11:10])** When SCU500[16]=0 00: Select AXI:AHB = default that make HCLK = 200MHz(default) 01: Select AXI:AHB = 2:1 10: Select AXI:AHB = 3:1 11: Select AXI:AHB = 4:1 When SCU500[16]=1 00: Select AXI:AHB = default that make HCLK = 200MHz 01: Select AXI:AHB = 4:1 10: Select AXI:AHB = 6:1 11: Select AXI:AHB = 8:1 Recommend to keep the default setting. Other setting may cause system unstable.
10:8	RW1S	RstPwr	**CPU frequency selection (OTPSTRAP[9:7])** 000: 1.2GHz (SCU200, SCU204)(default) 001: 1.6GHz 010: 1.2GHz 011: 1.6GHz 100: 800MHz 101: 800MHz 110: 800MHz 111: 800MHz Recommend to keep the default setting. Other setting may cause system unstable. The CPU frequency is defined by SCU200 and SCU204 when it is default value.
7	RW1S	RstPwr	**Define MAC#2 interface (OTPSTRAP[6])** 0: RMII (default) 1: RGMII This bit will set MAC#2 interface into RGMII mode.

to next page

from previous page

6	RW1S	RstPwr	**Define MAC#1 interface (OTPSTRAP[5])** 0: RMII (default) 1: RGMII This bit will set MAC#1 interface into RGMII mode.
5	RW1S	RstPwr	**Enable dedicated VGA BIOS ROM (OTPSTRAP[4])** 0: No VGA BIOS ROM, VGA BIOS is merged in the system BIOS (default) 1: Enable dedicated VGA BIOS ROM VGA BIOS is used for VGA initialization and display mode settings. Most BMC integrate VGA device and the VGA BIOS are integrated into system BIOS. It is not necessary to have a stand along SPI flash for VGA BIOS. For add-on VGA device, it needs a VGA BIOS ROM for VGA device. The VGA BIOS ROM is an SPI flash attached on FWSPI18* pins. This bit will enabled dedicate VGA BIOS ROM attached on FWSPI18* pins.
4	RW1S	RstPwr	**Secondary processor Cortex M3 (OTPSTRAP[3])** 0: Enable CM3 (default) 1: Disable CM3 This bit will disable CM3. CM3 will enter hardware reset mode.
3	RW1S	RstPwr	**Boot from debug SPI (OTPSTRAP[2])** 0: Disable (default) 1: Enable Enable this bit will set CPU to boot from SPI that is attached on pins FWSPI18*. This strap will not work when secure boot or boot from Uart5 is enabled. This bit is for verification and testing only. Please do NOT enable the OTPSTRAP[2] and protect it by setting OTPCFG30[2]=1 and OTPCFG28[2]=1 if there are security concerns.
2	RW1S	RstPwr	**Enable Boot from eMMC (OTPSTRAP[1])** 0: BMC boot from SPI (default) 1: BMC boot from eMMC BMC boot code can be in eMMC boot area partition instead of SPI when this bit is set. The image will be copied into SRAM then CPU start to fetch and execure codes from SRAM after ARMRST# de-asserted. The boot code size is limited to 64KB because the SRAM size.
1	RW1S	RstPwr	**Reserved (1)** This bit is reserved since A2 revision. Please refer to SEC14[6] and SCU510[8] to get the original information.
0	RW1S	RstPwr	**Disable ARM CA7 CPU boot (TXD5)** 0: Enable ARM CA7 CPU boot (default) 1: Disable ARM CA7 CPU boot When set this bit, hardware will stop CPU to boot up. Typically, there is an jumper with pull low resistor on TXD5. When user want to program SPI flash by using SPI programmer, short the jumper on TXD5 and power on BMC then user can start to program SPI flash without interference of CPU on SPI interface.

Note :
SCU500[i] can be set by write SCU500[i]=1 and be cleared by write SCU504[i]=1. After power on reset, the strap source information will be strapped into this register. It is programmable by writing data into SCU500 and SCU504 when SCU508 is 0. Set corresponding bits in OTPCFG28 can load the default value into SCU508 to enable write protection after power on or set SCU508 to enable write protection by software.

Offset: 504h	SCU504: Hardware Strap1 Clear Register	Init = 0

Bit	R/W	Reset	Description
31:0	W1C	-	**SCU500 Hardware Strap1 Clear Register**

Offset: 508h	SCU508: Hardware Strap1 Protect Register	Init = 0x40080000

Bit	R/W	Reset	Description
31:0	RW1S	Rst152	**SCU500 Write Protection Bit Control** SCU508[i]=0: SCU500[i] can be set by write SCU500[i]=1 and be cleared by write SCU504[i]=1 SCU508[i]=1: SCU500[i] is read only This regiter can be written to 1 only.

Offset: 510h	SCU510: Hardware Strap2 Register	Init = PIN/OTP

Bit	R/W	Reset	Description
31	RW1S	RstPwr	**Secure Boot Enable Pin Strap (FWSPIMOSI) Status** 0: Disables 1: Secure Boot Enabled This register is showing the latched value of the FWSPIMOSI pin strap. Please note that the Secure Boot is enabled only when the (OTPCFG0[6]=0. If (OTPCFG0[6]=1 it will ignore the FWSPIMOSI pin strap value. Refer to Straps for Secure Boot Enable for more details.
30	RW1S	RstPwr	**Enable Dedicate GPIO Strap Pins (OTPSTRAP[62])** 0: Disable (default) 1: Enable Set this bit will enable the strap sources of SCU510[15:13], SCU510[21:19] and SCU510[26:25] with physical hardware pins instead of from OTPSTRAP. Please reference to SCU510 related register description and 2.3 for more details.
29	RW1S	RstPwr	**Reserved (OTPSTRAP[61])**
28	RW1S	RstPwr	**Enable GPIO Pass Through (OTPSTRAP[60])** 0: Disable, pass through. (default) 1: Enable pass-through at power on. Pass-through pins set: GPIOP0 → GPIOP1 GPIOP2 → GPIOP3 GPIOP4 → GPIOP5 When OTPSTRAP[60]=1, The strap value will be 1 when the strap source OTPSTRAP[60]=1 or GPIOZ6 is high. When OTPSTRAP[60]=0, The strap value will be 1 when the strap source OTPSTRAP[60]=1. Set this bit will enable passthrough function from GPIOP0 to GPIOP1, GPIOP2 to GPIOP3 and GPIOP4 to GPIOP5.
27	RW1S	RstPwr	**Enable host SPI auxiliary control pins (trap_en_hspi_auxpin) (OTPSTRAP[59])** 0: Disable (default) 1: Enable Enable host SPI auxiliary control pins. Please reference to 15 SPI1 for more detail.

to next page

from previous page

26:25	RW1S	RstPwr	**Host SPI CRTM size (trap_hspi_crtmsize[1:0]) (OTPSTRAP[58:57])** 00: disable CRTM (default) 01: 1MB 10: 2MB 11: 4MB When SCU510[30]=0, the strap source is OTPSTRAP[58:57]. When SCU510[30]=1, the strap source OTPSTRAP[58] is replaced by hardware strap pin GPIOY5 and OTPSTRAP[57] is replaced by hardware strap pin GPIOY4. Set CRTM region size in host SPI flash device. Please reference to 15 SPI1 for more detail.
24:23	RW1S	RstPwr	**Boot SPI CRTM size (trap_bspi_crtmsize[1:0]) (OTPSTRAP[56:55])** 00: disable CRTM (default) 01: 256KB 10: 512KB 11: 1MB Set CRTM region size in boot SPI flash device. Please reference to 14 BSPI for more detail.
22	RW1S	RstPwr	**Enable boot SPI auxiliary control pins FWSPIDQ2, FWSPIDQ3, FWSPIABR, FWSPIWP# (OTPSTRAP[54])** 0: Disable (default) 1: Enable This is mirror register of SCU500[31]. This strap bit will enable pins FWSPIDQ2, FWSPIDQ3, FWSPIABR, FWSPIWP# and related functions automatically.
21:19	RW1S	RstPwr	**Host SPI flash size (trap_hspi_size[2:0]) (OTPSTRAP[53:51], GPIOZ3, GPIOZ4, GPIOZ5)** 000: no define size (default) 001: 2MB 010: 4MB 011: 8MB 100: 16MB 101: 32MB 110: 64MB 111: 128MB When SCU510[30]=0, the strap source is OTPSTRAP[53:51]. When SCU510[30]=1, the strap source OTPSTRAP[53] is replaced by hardware strap pin GPIOZ5, OTPSTRAP[52] is replaced by hardware strap pin GPIOZ4 and OTPSTRAP[51] is replaced by hardware strap pin GPIOZ3. When the value is 0, firmware can program SPIR30 to configure the maximum direct fetch address space. When the value is not 0, BSPI will automatically configure the register values of SPIR04[0], SPIR04[4], SPIR30[31:16] and SPIR34[31:00]. Please reference to 15 SPI1 for more detail.
18	RW1S	RstPwr	**Host SPI ABR Mode (trap_hspi_abrmode) (OTPSTRAP[50]** 0: dual SPI flash (default) 1: single SPI flash Alternative host SPI region mode selection. Please reference to 15 SPI1 for more detail.
17	RW1S	RstPwr	**Enable host SPI ABR mode select pin (trap_en_hspiabr_selpin) (OTPSTRAP[49])** 0: Disable (default) 1: Enable Alternative host SPI region source selection by pin SPI1ABR. Please reference to 15 SPI1 for more detail.

to next page

from previous page

16	RW1S	RstPwr	**Enable host SPI ABR (second boot)(trap_en_hspiabr) (OTPSTRAP[48])** 0: Disable (default) 1: Enable Alternative host SPI region mode enable. Please reference to 15 SPI1 for more detail.
15:13	RW1S	RstPwr	**Boot SPI flash size (trap_bspi_size[2:0]) (OTPSTRAP[47:45], GPIOZ6, GPIOZ7, FWSPICK)** 000: no define size (default) 001: 2MB 010: 4MB 011: 8MB 100: 16MB 101: 32MB 110: 64MB 111: 128MB When SCU510[30]=0, the strap source is OTPSTRAP[47:45]. When SCU510[30]=1, the strap source OTPSTRAP[47] is replaced by hardware strap pin FWSPICK, OTPSTRAP[46] is replaced by hardware strap pin GPIOZ7 and OTPSTRAP[45] is replaced by hardware strap pin GPIOZ6. When the value is 0, firmware can program FMC30 to configure the maximum direct fetch address space. When the value is not 0, BSPI will automatically configure the register values of FMC04[0], FMC04[4], FMC30[31:16] and FMC34[31:00]. Please reference to 14 BSPI for more detail.
12	RW1S	RstPwr	**Boot SPI ABR Mode (trap_bspi_abrmode) (OTPSTRAP[44])** 0: dual SPI flash (default) 1: single SPI flash Alternative boot region mode selection. Please reference to 14 BSPI for more detail.
11	RW1S	RstPwr	**Enable boot SPI or eMMC ABR (second boot)(trap_en_bspiabr) (OTPSTRAP[43])** 0: Disable (default) 1: Enable Alternative boot region mode enable. Please reference to 14 BSPI for more detail.
10	RW1S	RstPwr	**Enable boot SPI 3B/4B address mode auto detection (trap_en_bspiadrswap) (OTPSTRAP[42])** 0: Disable (default) 1: Enable When the BMC boot up, the boot SPI flash device could be possible in 3-byte or 4-byte address mode. Set this bit will be useful for such kind of unpredictable case. If the boot SPI flash device is 3-byte address mode only or always 4-byte address mode only and device size is fixed, set SCU510[15:13] is recommended.
9	RW1S	RstPwr	**Enable boot SPI 3B address mode auto-clear (OTPSTRAP[41])** 0: Disable (default) 1: Enable When SPI ABR mode is enabled and the SPI devices are 3B/4B capible, the SPI controller is expected to reset to 3B mode after watchdog reset, set this bit can automatically reset the SPI controller to 3B mode after watchdog reset without further software settings.

to next page

from previous page

8	RW1S	RstPwr	**Enable Boot from Uart by Pin Strap (OTPSTRAP[40] or FWSPICK)** 0: Disable(default) 1: Enable boot from Uart1 or Uart5 When SCU510[30]=0, the strap source is from pin FWSPICK. When SCU510[30]=1, the strap source is from OTPSTRAP[40]. When this bit is set, BMC will load code from Uart5 instead of eMMC or SPI. Its priority is higher than boot from eMMC or SPI. When secure boot is enabled, the code loaded from Uart5 is necessary to be signed correctly for security measurement.
7	RW1S	RstPwr	**Enable SAFS mode (OTPSTRAP[39])** 0: Disable SAFS (default) 1: Enable SAFS Set this bit to enable eSPI bus SAFS function.
6	RW1S	RstPwr	**Enable LPC mode (OTPSTRAP[38])** 0: eSPI mode (default) 1: LPC mode This bit is to enable LPC mode. This bit will strap to 1 when the strap source OTPSTRAP[38]=1. When SCU510[30]=0, this bit is 1 or SCU51C[7] will enable LPC mode. When SCU510[30]=1, only SCU510[6] can enable LPC mode.
5	RW1S	RstPwr	**Enable ACPI function (OTPSTRAP[37])** 0: Disable ACPI (default) 1: Enable ACPI Set this be will enable hardware ACPI and also enable pins SIOS3#, SIOS5#, SIOPWREQ#, SIOONCTRL#, SIOPWRGD, SIOPBO#, SIOPBO# and SIOSCI#
4	RW1S	RstPwr	**Disable Debug Interfaces (OTPSTRAP[36])** 0: Enable debug interfaces (default) 1: Disable debug interfaces This bit will disable all debug mode paths that are listed in SCU0D8 and is not possible to enable by other registers after power on. When this bit is not set, user still can program SCU0D8 to disable each one of the debug interfaces. It is not recommended during development period because it will cause developer almost impossible to diagnose hardware issues.
3	RW1S	RstPwr	**Disable LPC to decode SuperIO 0x2E/0x4E address (OTPSTRAP[35])** 0: Enable address decoding (default) 1: Disable address decoding This bit is used to disable SuperIO.
2	RW1S	RstPwr	**SuperIO configuration address selection (OTPSTRAP[34])** 0: Decode 0x2E (default) 1: Decode 0x4E SuperIO configuration register can be 0x2E or 0x4E. This bit is used for decode address slection.
1	RW1S	RstPwr	**Define MAC#4 interface (OTPSTRAP[33])** 0: RMII/NCSI (default) 1: RGMII This bit will set MAC#4 interface into RGMII mode.
0	RW1S	RstPwr	**Define MAC#3 interface (OTPSTRAP[32])** 0: RMII/NCSI (default) 1: RGMII This bit will set MAC#3 interface into RGMII mode.

to next page

from previous page

Note :
SCU510[i] can be set by write SCU510[i]=1 and be cleared by write SCU514[i]=1. After power on reset, the strap source information will be strapped into this register. It is programmable by writing data into SCU510 and SCU514 when SCU508 is 0. Set corresponding bits in OTPCFG29 can load the default value into SCU518 to enable write protection after power on or set SCU518 to enable write protection by software.

Offset: 514h	SCU514: Hardware Strap2 Clear Register	Init = 0

Bit	R/W	Reset	Description
31:0	W1C	-	**SCU510 Hardware Strap2 Clear Register**

Offset: 518h	SCU518: Hardware Strap2 Protect Register	Init = 0x00000010

Bit	R/W	Reset	Description
31:0	RW1S	Rst153	**SCU510 Write Protection Bit Control** SCU518[N]=0: SCU510[N] can be set by write SCU510[N]=1 and be cleared by write SCU514[N]=1 SCU518[N]=1: SCU510[N] is read only This regiter can be written to 1 only.

Offset: 51Ch	SCU51C: Hardware Strap3 Register	Init = PIN/OTP

Bit	R/W	Reset	Description
31:8	RW	-	**Reserved**
10	RW	-	**Enable ACPI function pins(GPIOZ7) when SCU510[30]=0** 0: Disable ACPI (default) 1: Enable ACPI This bit will be strapped to 1 when the strap source GPIOZ7 is high. It is used to enable hardware ACPI and also enable pins SIOS3#, SIOS5#, SIOPWREQ#, SIOONCTRL#, SIOPWRGD, SIOPBO#, SIOPBO# and SIOSCI#. When SCU510[30]=0, this bit is 1 or SCU510[5] will enable ACPI function pins. When SCU510[30]=1, only SCU510[5] can enable ACPI function pins.
9	RW	-	**Enable GPIO Pass Through (GPIOZ6) when SCU510[30]=0** 0: Disable, pass through. (default) 1: Enable pass through at power on. Pass through pins set: GPIOP0 → GPIOP1 GPIOP2 → GPIOP3 GPIOP4 → GPIOP5 This bit will be strapped to 1 when the strap source GPIOZ6 is high. It is used enable pass through function from GPIOP0 to GPIOP1, GPIOP2 to GPIOP3 and GPIOP4 to GPIOP5. When SCU510[30]=0, this bit is 1 or SCU510[28] will enable GPIO Pass Through. When SCU510[30]=1, only SCU510[28] can enable GPIO Pass Through.
8	RW	-	**Disable HBLED# (GPIOZ5) when SCU510[30]=0** This bit will be strapped to 1 when the strap source GPIOZ5 is high. It is used to disable HBLED# pin function. When SCU510[30]=0, this bit is used to disable HBLED# pin function. When SCU510[30]=1, this bit will not disable HBLED#.

to next page

from previous page

7	RW	-	**Enable LPC mode (GPIOZ4) when SCU510[30]=0** 0: eSPI mode (default) 1: LPC mode This bit will strap to 1 when the strap source GPIOZ4 is high. It is used to enable LPC mode when SCU510[30]=0. When SCU510[30]=0, this bit or SCU510[6] will enable LPC mode. When SCU510[30]=1, only SCU510[6] can enable LPC mode.
6	RW	-	**Disable VGA device(PCIe device 0) (GPIOZ3) when SCU510[30]=0** This bit will be strapped to 1 when the strap source GPIOZ3 is high. It is used to disable VGA device(PCIe device 0) only when SCU510[30]=0. When SCU510[30]=1, this bit will not disable VGA device.
5	RW	-	**Mirror Bit of ORPSTRAP[26]**
4	RW	-	**Mirror Bit of ORPSTRAP[25]**
3	RW	-	**Mirror Bit of ORPSTRAP[24]**
2	RW	-	**Mirror Bit of ORPSTRAP[22]**
1	RW	-	**Secure Boot using OTPSTRAP[0]** 0: Disabled 1: Secure Boot Enabled This register shows the content of the (OTPCFG0[0]. Please note that the Secure Boot is enabled using (OTPCFG0[0] only when the (OTPCFG0[6]=0. If (OTPCFG0[6]=1 it will ignore the (OTPCFG0[0] value. Refer to Straps for Secure Boot Enable for more details.
0	RW	-	**Enable Low Security Secure Boot Key by Pin Strap (FWSPIMISO)** 0: Disable 1: Enable Low Security Secure Boot Key by Pin Strap

Offset: 520h **SCU520: Random Number Generator Control** **Init = 0xE**

Bit	R/W	Reset	Description
31:4	RO	Rst154	**Reserved(0)**
3:1	RW	Rst154	**Random number generator mode selection** There are 8 types of random number generate methods.
0	RW	Rst154	**Random number generator disable control** 0: Enable 1: Disable

Offset: 524h **SCU524: Random Number Generator Data Output** **Init = X**

Bit	R/W	Reset	Description
31:0	RO	-	**Random number data** The random number is generated for each 1 us. Software can read the data by 1 us interval, or get 1 data for each 32 read command cycle.

Offset: 530h **SCU530: Random Number Generator 2 Control** **Init = 0xE**

Bit	R/W	Reset	Description
31	RO	RstFull	**Random Data is valid.** **This bit is valid only when bit 4 is high.** **Read SCU534 will clear this bit**
30:6	RO	Rst154	**Reserved(0)**

to next page

from previous page

5:1	RW	Rst154	**Random number generator mode selection** There are 9 types of random number generate methods.
0	RW	Rst154	**Random number generator disable control** 0: Enable 1: Disable

Offset: 534h **SCU534: Random Number Generator 2 Data Output** **Init = X**

Bit	R/W	Reset	Description
31:0	RO	-	**Random number data** The random number is generated for each 1 us. Software can read the data by 1 us interval, or get 1 data for each 32 read command cycle.

Offset: 544h **SCU544: Power Saving Wakeup Control Register (Reserved)** **Init = 0**

Bit	R/W	Reset	Description
31:9	RO	-	**Reserved(0)**
8:0	RW	Rst155	**Reserved**

Offset: 550h **SCU550: Power Saving Wakeup Enable Register (Reserved)** **Init = 0**

Bit	R/W	Reset	Description
31:10	RO	-	**Reserved(0)**
8:0	RW	Rst156	**Reserved**

Offset: 560h **SCU560: Interrupt Control and Status Register** **Init = 0**

Bit	R/W	Reset	Description
31:29	RO	-	**Reserved (0)**
28	RW1C	Rst01	**SRAM parity error interrupt status** 0 : No interrupt occurs 1 : Interrupt occurs The status flag can be cleared by writing '1' to this bit.
27	RW1C	Rst01	**ARM CA7 L2 cache parity error interrupt status** 0 : No interrupt occurs 1 : Interrupt occurs The status flag can be cleared by writing '1' to this bit.
26	RW1C	Rst01	**ARM CA7 L1 data cache parity error interrupt status** 0 : No interrupt occurs 1 : Interrupt occurs The status flag can be cleared by writing '1' to this bit.
25	RW1C	Rst01	**ARM CA7 L1 instruction cache parity error interrupt status** 0 : No interrupt occurs 1 : Interrupt occurs The status flag can be cleared by writing '1' to this bit.
24:22	RW1C	Rst01	**Reserved (0)**
21	RW1C	Rst01	**PCI-E RCRST# high-to-low interrupt status** 0 : No interrupt occurs 1 : Interrupt occurs The status flag can be cleared by writing '1' to this bit.

to next page

from previous page

20	RW1C	Rst01	**PCI-E RCRST# low-to-high interrupt status** 0 : No interrupt occurs 1 : Interrupt occurs The status flag can be cleared by writing '1' to this bit.
19	RW1C	Rst01	**PCI-E PERST# high-to-low interrupt status** 0 : No interrupt occurs 1 : Interrupt occurs The status flag can be cleared by writing '1' to this bit.
18	RW1C	Rst01	**PCI-E PERST# low-to-high interrupt status** 0 : No interrupt occurs 1 : Interrupt occurs The status flag can be cleared by writing '1' to this bit.
17	RW1C	Rst01	**VGA scratch register change Interrupt and status** 0 : No interrupt occurs 1 : Interrupt occurs The status flag can be cleared by writing '1' to this bit.
16	RW1C	Rst01	**VGA cursor change interrupt and status** 0 : No interrupt occurs 1 : Interrupt occurs The status flag can be cleared by writing '1' to this bit.
15	RW	Rst157	**BMC interrupt to HOST** 0 : Disable interrupt 1 : Enable interrupt generation
14:13	RO	-	**Reserved (0)**
12	RW	Rst157	**Enable SRAM parity error interrupt** 0 : Disable interrupt 1 : Enable interrupt generation
11	RW	Rst157	**Enable ARM CA7 L2 cache parity error interrupt** 0 : Disable interrupt 1 : Enable interrupt generation
10	RW	Rst157	**Enable ARM CA7 L1 data cache parity error interrupt** 0 : Disable interrupt 1 : Enable interrupt generation
9	RW	Rst157	**Enable ARM CA7 L1 instruction cache parity error interrupt** 0 : Disable interrupt 1 : Enable interrupt generation
8:6	RW	Rst157	**Reserved (0)**
5	RW	Rst157	**Enable PCI-E RCRST# high-to-low interrupt** 0 : Disable interrupt 1 : Enable interrupt generation
4	RW	Rst157	**Enable PCI-E RCRST# low-to-high interrupt** 0 : Disable interrupt 1 : Enable interrupt generation
3	RW	Rst157	**Enable PCI-E PERST# high-to-low interrupt** 0 : Disable interrupt 1 : Enable interrupt generation
2	RW	Rst157	**Enable PCI-E PERST# low-to-high interrupt** 0 : Disable interrupt 1 : Enable interrupt generation

to next page

from previous page

1	RW	Rst157	**Enable VGA scratch register change interrupt** 0 : Disable interrupt 1 : Enable interrupt generation
0	RW	Rst157	**Enable VGA cursor change interrupt** 0 : Disable interrupt 1 : Enable interrupt generation

Offset: 564h **SCU564: Interrupt Control and Status Register 1** **Init = 0**

Bit	R/W	Reset	Description
31:28	RO	Rst01	**Event Counter of PCI-E reset**
27: 0	RO	-	**Reserved (0)**

Offset: 570h **SCU570: Interrupt Control and Status Register 2** **Init = 0**

Bit	R/W	Reset	Description
31:28	RO	Rst01	**Event Counter of LPC reset**
27:22	RO	-	**Reserved (0)**
21	RW	Rst01	**LPC reset high-to-low interrupt status** 0 : No interrupt occurs 1 : Interrupt occurs The status flag can be cleared by writing '1' to this bit.
20	RW	Rst01	**LPC reset low-to-high interrupt status** 0 : No interrupt occurs 1 : Interrupt occurs The status flag can be cleared by writing '1' to this bit.
19:6	RO	-	**Reserved (0)**
6	RW	Rst157	**Reserved (0)**
5	RW	Rst157	**Enable LPC reset high-to-low interrupt (INT#41)** 0 : Disable interrupt 1 : Enable interrupt generation
4	RW	Rst157	**Enable LPC reset low-to-high interrupt (INT#41)** 0 : Disable interrupt 1 : Enable interrupt generation
3:0	RW	Rst157	**Reserved (0)**

Offset: 584h **SCU584: EFUSE Data Register (Mirror)** **Init = EFUSE**

Bit	R/W	Reset	Description
31:22	RO	-	**Reserved (0)**
21:0	RW	Rst159	**Mirror register of SCU594[21:0]**

Offset: 590h **SCU590: EFUSE Control** **Init = 0xFF3FA**

Bit	R/W	Reset	Description
31:30	RO	-	**Reserved**
29:28	RO	-	**EFUSE Controls**
27:0	RW	Rst160	**EFUSE Controls**

Offset: 594h	SCU594: EFUSE Data Register	Init = EFUSE

Bit	R/W	Reset	Description
31:18	RO	-	**Reserved (0)**
17	RO	RstFull	**Disable DP**
16	RO	RstFull	**Reserved**
15	RO	RstFull	**Disable EHCI**
14	RO	RstFull	**Disable VGA**
13	RO	RstFull	**Disable PCIe EP mode**
12	RO	RstFull	**Disable USB2.0 Host**
11	RO	RstFull	**Reserved**
10	RO	RstFull	**Disable RVAS**
9	RO	RstFull	**Disable Video decode engine**
8	RO	RstFull	**Disable Video engine**
7	RO	RstFull	**Disable PCIe RC mode**
6	RO	RstFull	**Disable ARM Cortex M3**
5	RO	RstFull	**Disable ARM Cortex A7**
4:2	RO	RstFull	**Chip ID** 000: AST2600 001: AST2620 010: Reserved 011: Reserved 100: Reserved 101: Reserved 110: Reserved 111: Reserved
1:0	RO	RstFull	**Reserved**

Offset: 5A0h	SCU5A0: SCU Free Run Counter Read Back	Init = 0
Offset: 5A4h	SCU5A4: SCU Free Run Counter Extended Read Back	Init = 0

Bit	Attr.	Reset	Description
63:60	RO	-	**Reserved (0)**
59:0	RO	RstPwr	**SCU free run counter bit[59:0] read back** The SCU free run counter is a 48-bit couter. Its value is reset by SRST# signal. The counter tick is 25MHz. The counter read back is for reference since last SRST#. It is not guaranteed to be glitch free when readback.

Offset: 5B0h	SCU5B0: Chip Unique ID 0	Init = EFUSE
Offset: 5B4h	SCU5B4: Chip Unique ID 1	Init = EFUSE

Bit	Attr.	Reset	Description
63:0	RO	RstFull	**Chip Unique ID bit[63:0] read back** Every chip has a chip unique ID. ASPEED will have production records for all of them. This ID also can be used by software for security purpose.

Offset: 5B8h	SCU5B8: Reserved Read Only ID 0	Init = EFUSE
Offset: 5BCh	SCU5BC: Reserved Read Only ID 1	Init = EFUSE
Offset: 5D0h	SCU5D0: Reserved Read Only ID 2	Init = EFUSE
Offset: 5D4h	SCU5D4: Reserved Read Only ID 3	Init = EFUSE

Bit	Attr.	Reset	Description
127:0	RO	RstFull	**Reserved Read ID bit[191:64] read back** There regiters are read only and not unique.

Offset: 610h	SCU610: Disable GPIO Internal Pull-Down #0	Init = 0

Bit	R/W	Reset	Description
31:16	RW	Rst161	**Reserved**
15	RW	Rst161	**Disable GPIOB7 Internal Pull-Down**
14	RW	Rst161	**Disable GPIOB6 Internal Pull-Down**
13	RW	Rst161	**Disable GPIOB5 Internal Pull-Down**
12	RW	Rst161	**Disable GPIOB4 Internal Pull-Down**
11	RW	Rst161	**Disable GPIOB3 Internal Pull-Down**
10	RW	Rst161	**Disable GPIOB2 Internal Pull-Down**
9	RW	Rst161	**Disable GPIOB1 Internal Pull-Down**
8	RW	Rst161	**Disable GPIOB0 Internal Pull-Down**
7:0	RW	Rst161	**Reserved**

Offset: 614h	SCU614: Disable GPIO Internal Pull-Down #1	Init = 0

Bit	R/W	Reset	Description
31:28	RW	Rst162	**Reserved**
27	RW	Rst162	**Disable GPIOH3 Internal Pull-Down**
26	RW	Rst162	**Disable GPIOH2 Internal Pull-Down**
25	RW	Rst162	**Disable GPIOH1 Internal Pull-Down**
24	RW	Rst162	**Disable GPIOH0 Internal Pull-Down**
23:0	RW	Rst162	**Reserved**

Offset: 618h	SCU618: Disable GPIO Internal Pull-Down #2	Init = 0

Bit	R/W	Reset	Description
31	RW	Rst163	**Disable GPIOL7 Internal Pull-Down**
30	RW	Rst163	**Disable GPIOL6 Internal Pull-Down**
29	RW	Rst163	**Disable GPIOL5 Internal Pull-Down**
28	RW	Rst163	**Disable GPIOL4 Internal Pull-Down**
27:16	RW	Rst163	**Reserved**
15	RW	Rst163	**Disable GPIOJ7 Internal Pull-Up**
14	RW	Rst163	**Disable GPIOJ6 Internal Pull-Up**
13	RW	Rst163	**Disable GPIOJ5 Internal Pull-Up**
12	RW	Rst163	**Disable GPIOJ4 Internal Pull-Up**
11	RW	Rst163	**Disable GPIOJ3 Internal Pull-Up**
10	RW	Rst163	**Disable GPIOJ2 Internal Pull-Up**

to next page

from previous page

9	RW	Rst163	**Disable GPIOJ1 Internal Pull-Up**
8	RW	Rst163	**Disable GPIOJ0 Internal Pull-Up**
7	RW	Rst163	**Disable GPIOI7 Internal Pull-Down**
6	RW	Rst163	**Disable GPIOI6 Internal Pull-Down**
5	RW	Rst163	**Disable GPIOI5 Internal Pull-Down**
4	RW	Rst163	**Disable GPIOI4 Internal Pull-Down**
3	RW	Rst163	**Disable GPIOI3 Internal Pull-Down**
2	RW	Rst163	**Disable GPIOI2 Internal Pull-Down**
1	RW	Rst163	**Disable GPIOI1 Internal Pull-Down**
0	RW	Rst163	**Disable GPIOI0 Internal Pull-Down**

Offset: 61Ch **SCU61C: Disable GPIO Internal Pull-Down #3** **Init = 0**

Bit	R/W	Reset	Description
31	RW	Rst164	**Disable GPIOP7 Internal Pull-Down**
30	RW	Rst164	**Disable GPIOP6 Internal Pull-Down**
29	RW	Rst164	**Disable GPIOP5 Internal Pull-Down**
28	RW	Rst164	**Disable GPIOP4 Internal Pull-Down**
27	RW	Rst164	**Disable GPIOP3 Internal Pull-Down**
26	RW	Rst164	**Disable GPIOP2 Internal Pull-Down**
25	RW	Rst164	**Disable GPIOP1 Internal Pull-Down**
24	RW	Rst164	**Disable GPIOP0 Internal Pull-Down**
23	RW	Rst164	**Disable GPIOO7 Internal Pull-Down**
22	RW	Rst164	**Disable GPIOO6 Internal Pull-Down**
21	RW	Rst164	**Disable GPIOO5 Internal Pull-Down**
20	RW	Rst164	**Disable GPIOO4 Internal Pull-Down**
19	RW	Rst164	**Disable GPIOO3 Internal Pull-Down**
18	RW	Rst164	**Disable GPIOO2 Internal Pull-Down**
17	RW	Rst164	**Disable GPIOO1 Internal Pull-Down**
16	RW	Rst164	**Disable GPIOO0 Internal Pull-Down**
15	RW	Rst164	**Disable GPION7 Internal Pull-Down**
14	RW	Rst164	**Disable GPION6 Internal Pull-Down**
13	RW	Rst164	**Disable GPION5 Internal Pull-Down**
12	RW	Rst164	**Disable GPION4 Internal Pull-Down**
11	RW	Rst164	**Disable GPION3 Internal Pull-Down**
10	RW	Rst164	**Disable GPION2 Internal Pull-Down**
9	RW	Rst164	**Disable GPION1 Internal Pull-Down**
8	RW	Rst164	**Disable GPION0 Internal Pull-Down**
7	RW	Rst164	**Disable GPIOM7 Internal Pull-Down**
6	RW	Rst164	**Disable GPIOM6 Internal Pull-Down**
5	RW	Rst164	**Disable GPIOM5 Internal Pull-Down**
4	RW	Rst164	**Disable GPIOM4 Internal Pull-Down**
3	RW	Rst164	**Disable GPIOM3 Internal Pull-Down**

to next page

from previous page

2	RW	Rst164	**Disable GPIOM2 Internal Pull-Down**
1	RW	Rst164	**Disable GPIOM1 Internal Pull-Down**
0	RW	Rst164	**Disable GPIOM0 Internal Pull-Down**

Offset: 630h **SCU630: Disable GPIO Internal Pull-Down #4** **Init = 0**

Bit	R/W	Reset	Description
31	RW	Rst165	**Disable ADC7/GPIT7 Internal Pull-Down**
30	RW	Rst165	**Disable ADC6/GPIT6 Internal Pull-Down**
29	RW	Rst165	**Disable ADC5/GPIT5 Internal Pull-Down**
28	RW	Rst165	**Disable ADC4/GPIT4 Internal Pull-Down**
27	RW	Rst165	**Disable ADC3/GPIT3 Internal Pull-Down**
26	RW	Rst165	**Disable ADC2/GPIT2 Internal Pull-Down**
25	RW	Rst165	**Disable ADC1/GPIT1 Internal Pull-Down**
24	RW	Rst165	**Disable ADC0/GPIT0 Internal Pull-Down**
23	RW	Rst165	**Disable GPIOS7 Internal Pull-Down**
22	RW	Rst165	**Disable GPIOS6 Internal Pull-Down**
21	RW	Rst165	**Disable GPIOS5 Internal Pull-Down**
20	RW	Rst165	**Disable GPIOS4 Internal Pull-Down**
19	RW	Rst165	**Disable GPIOS3 Internal Pull-Down**
18	RW	Rst165	**Disable GPIOS2 Internal Pull-Down**
17	RW	Rst165	**Disable GPIOS1 Internal Pull-Down**
16	RW	Rst165	**Disable GPIOS0 Internal Pull-Down**
15	RW	Rst165	**Disable GPIOR7 Internal Pull-Down**
14	RW	Rst165	**Disable GPIOR6 Internal Pull-Down**
13	RW	Rst165	**Disable GPIOR5 Internal Pull-Down**
12	RW	Rst165	**Disable GPIOR4 Internal Pull-Down**
11	RW	Rst165	**Disable GPIOR3 Internal Pull-Down**
10	RW	Rst165	**Disable GPIOR2 Internal Pull-Down**
9	RW	Rst165	**Disable GPIOR1 Internal Pull-Down**
8	RW	Rst165	**Disable GPIOR0 Internal Pull-Down**
7:0	RW	Rst165	**Reserved**

Offset: 634h **SCU634: Disable GPIO Internal Pull-Down #5** **Init = 0**

Bit	R/W	Reset	Description
31	RW	Rst166	**Disable GPIOX7 Internal Pull-Down**
30	RW	Rst166	**Disable GPIOX6 Internal Pull-Down**
29	RW	Rst166	**Disable GPIOX5 Internal Pull-Down**
28	RW	Rst166	**Disable GPIOX4 Internal Pull-Down**
27	RW	Rst166	**Disable GPIOX3 Internal Pull-Down**
26	RW	Rst166	**Disable GPIOX2 Internal Pull-Down**
25	RW	Rst166	**Disable GPIOX1 Internal Pull-Down**
24	RW	Rst166	**Disable GPIOX0 Internal Pull-Down**

to next page

from previous page

23:16	RW	Rst166	**Reserved**
15	RW	Rst166	**Disable GPIOV7 Internal Pull-Down**
14	RW	Rst166	**Disable GPIOV6 Internal Pull-Down**
13	RW	Rst166	**Disable GPIOV5 Internal Pull-Down**
12	RW	Rst166	**Disable GPIOV4 Internal Pull-Down**
11	RW	Rst166	**Disable GPIOV3 Internal Pull-Down**
10	RW	Rst166	**Disable GPIOV2 Internal Pull-Down**
9	RW	Rst166	**Disable GPIOV1 Internal Pull-Down**
8	RW	Rst166	**Disable GPIOV0 Internal Pull-Down**
7	RW	Rst165	**Disable ADC15/GPIU7 Internal Pull-Down**
6	RW	Rst165	**Disable ADC14/GPIU6 Internal Pull-Down**
5	RW	Rst165	**Disable ADC13/GPIU5 Internal Pull-Down**
4	RW	Rst165	**Disable ADC12/GPIU4 Internal Pull-Down**
3	RW	Rst165	**Disable ADC11/GPIU3 Internal Pull-Down**
2	RW	Rst165	**Disable ADC10/GPIU2 Internal Pull-Down**
1	RW	Rst165	**Disable ADC9/GPIU1 Internal Pull-Down**
0	RW	Rst165	**Disable ADC8/GPIU0 Internal Pull-Down**

Offset: 638h **SCU638: Disable GPIO Internal Pull-Down #6** **Init = 0xF0000**

Bit	R/W	Reset	Description
31:16	RW	Rst167	**Reserved**
15	RW	Rst167	**Disable GPIOZ7 Internal Pull-Down**
14	RW	Rst167	**Disable GPIOZ6 Internal Pull-Down**
13	RW	Rst167	**Disable GPIOZ5 Internal Pull-Down**
12	RW	Rst167	**Disable GPIOZ4 Internal Pull-Down**
11	RW	Rst167	**Disable GPIOZ3 Internal Pull-Down**
10	RW	Rst167	**Reserved**
9	RW	Rst167	**Disable GPIOZ1 Internal Pull-Down**
8	RW	Rst167	**Disable GPIOZ0 Internal Pull-Down**
7	RW	Rst167	**Reserved**
6	RW	Rst167	**Disable GPIOY6 Internal Pull-Down**
5	RW	Rst167	**Disable GPIOY5 Internal Pull-Down**
4	RW	Rst167	**Disable GPIOY4 Internal Pull-Down**
3	RW	Rst167	**Disable GPIOY3 Internal Pull-Down**
2	RW	Rst167	**Disable GPIOY2 Internal Pull-Down**
1	RW	Rst167	**Disable GPIOY1 Internal Pull-Down**
0	RW	Rst167	**Disable GPIOY0 Internal Pull-Down**

Offset: 650h **SCU650: IO Driving Strength** **Init = 0x11**

Bit	R/W	Reset	Description
31:29	RW	Rst168	**FSI 1 DATA delay stage**
28:26	RW	Rst168	**FSI 1 CLK delay stage**

to next page

from previous page

25:23	RW	Rst168	**FSI 0 DATA delay stage**
22:20	RW	Rst168	**FSI 0 CLK delay stage**
19:16	RW	Rst168	**Reserved**
15	RW	Rst168	**HVI3C6 Driving Strength**
14	RW	Rst168	**HVI3C5 Driving Strength**
13	RW	Rst168	**HVI3C4 Driving Strength**
12	RW	Rst168	**HVI3C3 Driving Strength** 1b : 16mA. 0b : 12mA.
11	RW	Rst168	**Reserved**
10	RW	Rst168	**SLI VP Slew Rate control**
9: 8	RW	Rst168	**SLI VP Driving Strength**
7	RW	Rst168	**Reserved**
6	RW	Rst168	**SLI UP Slew Rate control**
5: 4	RW	Rst168	**SLI UP Driving Strength**
3	RW	Rst168	**Reserved**
2	RW	Rst168	**SLI DOWN Slew Rate control**
1: 0	RW	Rst168	**SLI DOWN Driving Strength**
Note : Slew Rate control and Driving Strength of SLI relating setting is HW parameter and not suggesting to change it			

Offset: 800h **SCU800: CA7 Processor Control Register** **Init = 0x3FCF**

Bit	R/W	Reset	Description
31:16	RO	-	**Reserved(0)**
15	RW	Rst169	**CA7 AXI Prefetch Enable** 0: Disable 1: Enable
14	RW	Rst169	**Reserved**
13	RW	Rst169	**ARM CM3 NIDEN** ARM CM3 non-invasive debug enable. Default value 1
12	RW	Rst169	**ARM CM3 DBGEN** ARM CM3 debug enable. Default value 1
11	RW	Rst169	**Coresight NIDEN** Coresight non-invasive debug enable. Default value 1
10	RW	Rst169	**Coresight DBGEN** Coresight enable. Default value 1
9	RW	Rst169	**ARM CA7 DBGEN** ARM CA7 debug enable. Default value 1
8	RW	Rst169	**ARM CA7 SPIDEN** ARM CA7 secure privileged debug ENable Default value 1
7	RW	Rst169	**ARM CA7 NIDEN** ARM CA7 non-invasive debug enable. Default value 1
6	RW	Rst169	**ARM CA7 SPNIDEN** ARM CA7 secure privileged non-invasive debug enable. Default value 1
5	RW	Rst169	**ARM CA7 CP15SDISABLE** Default value 0

to next page

from previous page

4	RW	Rst169	**ARM CA7 CFGSDISABLE** Default value 0
3:0	RW	Rst169	**ARM CA7 CLUSTERID** Default value 0xF

Offset: 808h	**SCU808: CA7 Processor Control Register**	**Init = 0xFFFFFFFF**

Bit	R/W	Reset	Description
31:0	RW	Rst170	**CA7 AXI Prefetch Range Start Address**

Offset: 80Ch	**SCU80C: CA7 Processor Control Register**	**Init = 0**

Bit	R/W	Reset	Description
31:0	RW	Rst171	**CA7 AXI Prefetch Range End Address**

Note :
The AXI prefetch is disabled when the value of SCU80C is less than SCU808

Offset: 820h	**SCU820: CA7 Processor Parity Check Control Register**	**Init = 0**

Bit	R/W	Reset	Description
31: 5	RO	-	**Reserved(0)**
4	RW	Rst172	**SRAM Parity Check Enable**
3: 1	RO	-	**Reserved(0)**
0	RW	Rst172	**CPU Cache Parity Check Enable**

Offset: 824h	**SCU824: CA7 Processor Parity Clear Register**	**Init = 0x1**

Bit	R/W	Reset	Description
31: 1	RO	-	**Reserved(0)**
0	RW	Rst173	**CA7 Parity Clear**

Offset: 830h	**SCU830: Watchdog Controller Protection Control 1**	**Init = 0**

Bit	R/W	Reset	Description
31: 2	RO	-	**Reserved(0)**
1	RW	-	**Reserved(0)**
0	RW	Rst234	**Enable Watchdog Control Protection Control** 0: disable 1: enable

Offset: 834h	**SCU834: Security Protection Control**	**Init = 0**

Bit	R/W	Reset	Description
31: 1	RO	-	**Reserved(0)**
15:11	RW	-	**Reserved(0)**
10	RW	Rst235	**Enable I2CS Control SSP Protection** 0: I2CS Control can be access by ARM CA7 or SSP both 1: I2CS Control can be access by SSP only.

to next page

from previous page

9	RW	Rst235	**Enable I3C DMA Control Protection** 0: Disable 1: Protection of I3C DMA to access DRAM memory space only.
8 :7	RW	-	**Reserved(0)**
6	RW	Rst235	**Enable UART DMA Control Protection** 0: Disable 1: Protection of UART DMA to access DRAM memory space only.
5	RW	Rst235	**Enable eSPI DMA Control Protection** 0: Disable 1: Protection of eSPI DMA to access DRAM memory space only.
4	RW	Rst235	**Enable Post Code DMA Control Protection** 0: Disable 1: Protection of Post Code DMA to access DRAM memory space only.
3	RW	Rst235	**Enable SPI DMA Control Protection** 0: Disable 1: Protection of SPI DMA to access DRAM memory space only.
2	RW	Rst235	**Enable Boot SPI DMA Control Protection** 0: Disable 1: Protection of Boot SPI DMA to access DRAM memory space only.
1	RW	Rst235	**Enable LPC DMA Control Protection** 0: Disable 1: Protection of LPC DMA to access DRAM memory space only.
0	RW	Rst235	**Enable SD DMA Control Protection** 0: Disable 1: Protection of SD DMA to access DRAM memory space only.

Offset: 850h **SCU850: Watchdog Controller Protection Control 2** **Init = 0**

Bit	R/W	Reset	Description
31: 8	RO	-	**Reserved(0)**
7	RW	Rst236	**Watchdog Control WDT8 SSP Protection Control** 0: normal 1: WDT8 can be access by SSP only.
6	RW	Rst236	**Watchdog Control WDT7 SSP Protection Control** 0: normal 1: WDT7 can be access by SSP only.
5	RW	Rst236	**Watchdog Control WDT6 SSP Protection Control** 0: normal 1: WDT6 can be access by SSP only.
4	RW	Rst236	**Watchdog Control WDT5 SSP Protection Control** 0: normal 1: WDT5 can be access by SSP only.
3	RW	Rst236	**Watchdog Control WDT4 SSP Protection Control** 0: normal 1: WDT4 can be access by SSP only.
2	RW	Rst236	**Watchdog Control WDT3 SSP Protection Control** 0: normal 1: WDT3 can be access by SSP only.
1	RW	Rst236	**Watchdog Control WDT2 SSP Protection Control** 0: normal 1: WDT2 can be access by SSP only.

to next page

from previous page

0	RW	Rst236	**Watchdog Control WDT1 SSP Protection Control** 0: normal 1: WDT1 can be access by SSP only.

Offset: 854h **SCU854: Watchdog Controller Protection Control 3** **Init = 0**

Bit	R/W	Reset	Description
31: 8	RO	-	**Reserved(0)**
7	RW	Rst237	**Watchdog Control WDT8 ARM CA7 Protection Control** 0: normal 1: WDT8 can be access by ARM CA7 only.
6	RW	Rst237	**Watchdog Control WDT7 ARM CA7 Protection Control** 0: normal 1: WDT7 can be access by ARM CA7 only.
5	RW	Rst237	**Watchdog Control WDT6 ARM CA7 Protection Control** 0: normal 1: WDT6 can be access by ARM CA7 only.
4	RW	Rst237	**Watchdog Control WDT5 ARM CA7 Protection Control** 0: normal 1: WDT5 can be access by ARM CA7 only.
3	RW	Rst237	**Watchdog Control WDT4 ARM CA7 Protection Control** 0: normal 1: WDT4 can be access by ARM CA7 only.
2	RW	Rst237	**Watchdog Control WDT3 ARM CA7 Protection Control** 0: normal 1: WDT3 can be access by ARM CA7 only.
1	RW	Rst237	**Watchdog Control WDT2 ARM CA7 Protection Control** 0: normal 1: WDT2 can be access by ARM CA7 only.
0	RW	Rst237	**Watchdog Control WDT1 ARM CA7 Protection Control** 0: normal 1: WDT1 can be access by ARM CA7 only.

Offset: 858h **SCU858: Watchdog Controller Protection Control 4** **Init = 0**

Bit	R/W	Reset	Description
31: 8	RO	-	**Reserved(0)**
7	RW	Rst238	**Watchdog Control WDT8 Write Protection Control** 0: normal 1: WDT8 related registers are write protected.
6	RW	Rst238	**Watchdog Control WDT7 Write Protection Control** 0: normal 1: WDT7 related registers are write protected.
5	RW	Rst238	**Watchdog Control WDT6 Write Protection Control** 0: normal 1: WDT6 related registers are write protected.
4	RW	Rst238	**Watchdog Control WDT5 Write Protection Control** 0: normal 1: WDT5 related registers are write protected.

to next page

from previous page

3	RW	Rst238	**Watchdog Control WDT4 Write Protection Control** 0: normal 1: WDT4 related registers are write protected.
2	RW	Rst238	**Watchdog Control WDT3 Write Protection Control** 0: normal 1: WDT3 related registers are write protected.
1	RW	Rst238	**Watchdog Control WDT2 Write Protection Control** 0: normal 1: WDT2 related registers are write protected.
0	RW	Rst238	**Watchdog Control WDT1 Write Protection Control** 0: normal 1: WDT1 related registers are write protected.
Note : The above WDT write protection will apply to all of WDT registers except WDT08 and WDT24.			

Offset: 85Ch **SCU85C: Watchdog Controller Protection Control 5** **Init = 0**

Bit	R/W	Reset	Description
31: 8	RO	-	**Reserved(0)**
7	RW	Rst239	**Watchdog Control WDT8 Mask Registers Protection Control** 0: normal 1: WDT8 mask registers are write protected.
6	RW	Rst239	**Watchdog Control WDT7 Mask Registers Protection Control** 0: normal 1: WDT7 mask registers are write protected.
5	RW	Rst239	**Watchdog Control WDT6 Mask Registers Protection Control** 0: normal 1: WDT6 mask registers are write protected.
4	RW	Rst239	**Watchdog Control WDT5 Mask Registers Protection Control** 0: normal 1: WDT5 mask registers are write protected.
3	RW	Rst239	**Watchdog Control WDT4 Mask Registers Protection Control** 0: normal 1: WDT4 mask registers are write protected.
2	RW	Rst239	**Watchdog Control WDT3 Mask Registers Protection Control** 0: normal 1: WDT3 mask registers are write protected.
1	RW	Rst239	**Watchdog Control WDT2 Mask Registers Protection Control** 0: normal 1: WDT2 mask registers are write protected.
0	RW	Rst239	**Watchdog Control WDT1 Mask Registers Protection Control** 0: normal 1: WDT1 mask registers are write protected.
Note : The above WDT mask register protection will apply to WDT mask registers WDT1C, WDT20, WDT28 and WDT2C.			

Offset: 870h	SCU870: Watchdog Controller Protection Control 6	Init = 0

Bit	R/W	Reset	Description
31: 0	RW	Rst240	**WDT1C and WDT28 Mask Register for ARM CA7 Write Access** If the bits in this register are set to 1, the write data from ARM CA7 to WDT1C and WDT28 will be masked to 0. The others will keep the same. The masked write data from ARM CA7 to WDT1C and WDT20 is: Masked Data = Write Data & inverse SCU870

Offset: 874h	SCU874: Watchdog Controller Protection Control 7	Init = 0

Bit	R/W	Reset	Description
31: 0	RW	Rst241	**WDT20 and WDT2C Mask Register for ARM CA7 Write Access** If the bits in this register are set to 1, the write data from ARM CA7 to WDT20 and WDT2C will be masked to 0. The others will keep the same. The masked write data from ARM CA7 to WDT20 and WDT2C is: Masked Data = Write Data & inverse SCU874

Offset: 878h	SCU878: Watchdog Controller Protection Control 8	Init = 0

Bit	R/W	Reset	Description
31: 0	RW	Rst242	**WDT1C and WDT28 Mask Register for SSP Write Access** If the bits in this register are set to 1, the write data from SSP to WDT1C and WDT28 will be masked to 0. The others will keep the same. The masked write data from SSP to WDT1C and WDT28 is: Masked Data = Write Data & inverse SCU878

Offset: 87Ch	SCU87C: Watchdog Controller Protection Control 9	Init = 0

Bit	R/W	Reset	Description
31: 0	RW	Rst243	**WDT20 and WDT2C Mask Register for SSP Write Access** If the bits in this register are set to 1, the write data from SSP to WDT20 and WDT2C will be masked to 0. The others will keep the same. The masked write data from SSP to WDT20 and WDT2C is: Masked Data = Write Data & inverse SCU87C

Offset: A00h	SCUA00: CM3 Coprocessor Control Register	Init = 0

Bit	R/W	Reset	Description
31:3	RO	-	**Reserved(0)**
2	RW	Rst174	**Reserved**
1	RW	Rst174	**CM3 Coprocessor Reset** 0: Normal operation 1: Enable reset When turned ON, it must keep at least 1 us. When Coprocessor disabled, it is also reset
0	RW	Rst174	**CM3 Coprocessor Enable** 0: Disable 1: Enable When turned OFF, it must keep at least 1 us.

Offset: A04h	SCUA04: CM3 Memory Base Address Register	Init = 0

Bit	R/W	Reset	Description
31:20	RW	Rst175	**Bit 31 20 of the DRAM base address for CM3**
19:0	RO	-	**Reserved(0)**

Note :
This register can be only changed while CM3 is held in reset
Or CM3 will get wrong instruction and data

Offset: A08h	SCUA08: CM3 Instruction Memory Address Limit Register	Init = 0

Bit	R/W	Reset	Description
31:20	RW	Rst176	**Bit 31 20 of the upper bound DRAM base address for CM3 instruction**
19:0	RO	-	**Reserved(0)**

Note :
This register can be only changed while CM3 is held in reset

Offset: A0Ch	SCUA0C: CM3 Data Memory Address Limit Register	Init = 0

Bit	R/W	Reset	Description
31:20	RW	Rst176	**Bit 31 20 of the upper bound DRAM base address for CM3 data**
19:0	RO	-	**Reserved(0)**

Note :
This register can be only changed while CM3 is held in reset

Offset: A40	SCUA40: CM3 Cacheable Area Declaration	Init = 0xFFFFFFFF

Bit	R/W	Reset	Description
31	RW	Rst178	**cacheable for CM3 0x1F000000 0x1FFFFFFF (the 32nd 16MB)**
30	RW	Rst178	**cacheable for CM3 0x1E000000 0x1EFFFFFF (the 31st 16MB)**
29	RW	Rst178	**cacheable for CM3 0x1D000000 0x1DFFFFFF (the 30th 16MB)**
28	RW	Rst178	**cacheable for CM3 0x1C000000 0x1CFFFFFF (the 29th 16MB)**
27	RW	Rst178	**cacheable for CM3 0x1B000000 0x1BFFFFFF (the 28th 16MB)**
26	RW	Rst178	**cacheable for CM3 0x1A000000 0x1AFFFFFF (the 27th 16MB)**
25	RW	Rst178	**cacheable for CM3 0x19000000 0x19FFFFFF (the 26th 16MB)**
24	RW	Rst178	**cacheable for CM3 0x18000000 0x18FFFFFF (the 25th 16MB)**
23	RW	Rst178	**cacheable for CM3 0x17000000 0x17FFFFFF (the 24th 16MB)**
22	RW	Rst178	**cacheable for CM3 0x16000000 0x16FFFFFF (the 23rd 16MB)**
21	RW	Rst178	**cacheable for CM3 0x15000000 0x15FFFFFF (the 22nd 16MB)**
20	RW	Rst178	**cacheable for CM3 0x14000000 0x14FFFFFF (the 21st 16MB)**
19	RW	Rst178	**cacheable for CM3 0x13000000 0x13FFFFFF (the 20th 16MB)**
18	RW	Rst178	**cacheable for CM3 0x12000000 0x12FFFFFF (the 19th 16MB)**
17	RW	Rst178	**cacheable for CM3 0x11000000 0x11FFFFFF (the 18th 16MB)**
16	RW	Rst178	**cacheable for CM3 0x10000000 0x10FFFFFF (the 17th 16MB)**
15	RW	Rst178	**cacheable for CM3 0x0F000000 0x0FFFFFFF (the 16th 16MB)**
14	RW	Rst178	**cacheable for CM3 0x0E000000 0x0EFFFFFF (the 15th 16MB)**
13	RW	Rst178	**cacheable for CM3 0x0D000000 0x0DFFFFFF (the 14th 16MB)**

to next page

from previous page

12	RW	Rst178	**cacheable for CM3 0x0C000000 0x0CFFFFFF (the 13rd 16MB)**
11	RW	Rst178	**cacheable for CM3 0x0B000000 0x0BFFFFFF (the 12nd 16MB)**
10	RW	Rst178	**cacheable for CM3 0x0A000000 0x0AFFFFFF (the 11st 16MB)**
9	RW	Rst178	**cacheable for CM3 0x09000000 0x09FFFFFF (the 10th 16MB)**
8	RW	Rst178	**cacheable for CM3 0x08000000 0x08FFFFFF (the 9th 16MB)**
7	RW	Rst178	**cacheable for CM3 0x07000000 0x07FFFFFF (the 8th 16MB)**
6	RW	Rst178	**cacheable for CM3 0x06000000 0x06FFFFFF (the 7th 16MB)**
5	RW	Rst178	**cacheable for CM3 0x05000000 0x05FFFFFF (the 6th 16MB)**
4	RW	Rst178	**cacheable for CM3 0x04000000 0x04FFFFFF (the 5th 16MB)**
3	RW	Rst178	**cacheable for CM3 0x03000000 0x03FFFFFF (the 4th 16MB)**
2	RW	Rst178	**cacheable for CM3 0x02000000 0x02FFFFFF (the 3rd 16MB)**
1	RW	Rst178	**cacheable for CM3 0x01000000 0x01FFFFFF (the 2nd 16MB)**
0	RW	Rst178	**cacheable for CM3 0x00000000 0x00FFFFFF (the 1st 16MB)**

Note :
CM3 instruction and data access only cover 512MB
SCUA04 has already defined the physical DRAM base as CM3 address 0x00000000
SCUA40 is used to define 32 areas of 16MB inside that 512MB are cacheable or not
This register can be only changed while CM3 is held in reset

Offset: A44h **SCUA44: CM3 Cache Invalidation Control Register** **Init = 0**

Bit	R/W	Reset	Description
31	W1T	-	**CM3 Data Cache Invalid** Write with 1 to invalid each entry of data cache corresponding to SCUA44[26:16]
26:16	RW	RstFull	**CM3 Data Cache Invalid Address** Each entry of data cache corresponding to this 11-bit lsb byte address to be invalidated
15	W1T	-	**CM3 Instruction Cache Invalid** Write with 1 to invalid each entry of instruction cache corresponding to SCUA44[10:0]
10:0	RW	RstFull	**CM3 Instruction Cache Invalid Address** Each entry of instruction cache corresponding to this 11-bit lsb byte address to be invalidated

Note :
In AST2600 , from CM3 point of view, all its write will be issued directly to DRAM since the cache uses write through strategy
If CA7 and CM3 share a region of DRAM and exchange data there, CM3 shall program SCUA44 before reading. Or CA7 shall program SCUA44 after writing
For example, if CA7 writes 0x86000280 and CM3 DRAM base address is 0x83000000, CM3 shall write SCUA44 with 0x8280 before reading 0x03000280, which corresponds to DRAM 0x86000280
For example, if CA7 writes 0x86000000 and CM3 DRAM base address is 0x83000000, CM3 shall write SCUA44 with 0x8000 before reading 0x03000000, which corresponds to DRAM 0x86000000

Offset: A48h **SCUA48: CM3 Cache Function Control Register** **Init = 0**

Bit	R/W	Reset	Description
31	RO	-	**Reserved(0)**

to next page

from previous page

30	RO	-	**Masked interrupt status for CM3 system bus address monitoring** SCUA48[30] = SCUA48[26] & SCUA48[13]; In the beginning, this signal is intended to connect these interrupt signals to CA7, but at the end this is for debugging only
29	RO	-	**Masked interrupt status for CM3 data bus address monitoring** SCUA48[29] = SCUA48[25] & SCUA48[13]; In the beginning, this signal is intended to connect these interrupt signals to CA7, but at the end this is for debugging only
28	RO	-	**Masked interrupt status for CM3 instruction bus address monitoring** SCUA48[28] = SCUA48[24] & SCUA48[13]; In the beginning, this signal is intended to connect these interrupt signals to CA7, but at the end this is for debugging only
27	RO	-	**Reserved(0)**
26	RO	-	**Raw interrupt status for CM3 system bus address monitoring** For debugging to see if CM3 firmware tries to access undefined memory allocation to itself this bit is to indicate if CM3 S-bus access is not within 0x60000000 and 0xE0000000
25	RO	-	**Raw interrupt status for CM3 data bus address monitoring** For debugging to see if CM3 firmware tries to access undefined memory allocation to itself this bit is to indicate if CM3 D-bus access is not within SCUA04 and SCUA0C
24	RO	-	**Raw interrupt status for CM3 instruction bus address monitoring** For debugging to see if CM3 firmware tries to access undefined memory allocation to itself this bit is to indicate if CM3 I-bus access is not within SCUA04 and SCUA08
23:16	RO	-	**Reserved(0)**
15	RW	Rst179	**Reserved**
14	RW	Rst179	**Interrupt clear for CM3 instruction/data/system bus address monitoring** 0: disable 1: enable
13	RW	Rst179	**Interrupt mask for CM3 instruction/data/system bus address monitoring this bit decides if SCUA48[30:28] will be copies of SCUA48[26:24]** 0: disable 1: enable
12	RW	Rst179	**Interrupt enable for CM3 instruction/data/system bus address monitoring** 0: disable 1: enable
11	RW	Rst179	**Selection of master for closely coupled memory** This register can be only changed while CM3 is held in reset. Or CM3 shall be reset after any changes of this register. This bit shall be kept in 0 0: CM3 instruction and data bus; CM3 owns the access permission to closely coupled memory for CM3 1: System bus; CA7 and any other bus masters own the access permission to closely coupled memory for CM3

to next page

from previous page

10	RW	Rst179	**Closely coupled memory mode** Enabling this function will not access DRAM When a certain light version application is preferred without DRAM access since prompt reaction without DRAM latency is expected internal SRAM of 4KB * 2 is offered for I-bus and D-bus Before enabling this function, CA7 shall write SCUA48[11] with 1 and prepare CM3 instruction and initial data within 0x1A000000 0x1A03FFFF After CA7 finishes programming instruction and code for CM3, CA7 shall write SCUA48[11] with 0 This register can be only changed while CM3 is held in reset. Or CM3 shall be reset after any changes of this register. This bit shall be kept in 0 0: disable 1: enable
9 :3	RW	Rst179	**Reserved**
2	RW	Rst179	**Clear instruction cache** Before writing this bit with 1, it shall be 0 0: disable 1: enable clearing instruction cache for one time
1	RW	Rst179	**Clear data cache data** Before writing this bit with 1, it shall be 0 0: disable 1: enable clearing data cache for one time
0	RW	Rst179	**Cache Enable** 0: disable cache 1: enable cache

Offset: C00h **SCUC00: PCI Configuration Setting Register #1** **Init = 0x20001A03**

Bit	R/W	Reset	Description
31:16	RW	Rst180	**PCI Device ID** The register can support firmware code the flexibility to change PCI Device ID of AST2600 . But changing the ID is usually not recommended.
15:0	RW	Rst180	**PCI Vendor ID** The register can support firmware code the flexibility to change PCI Vendor ID of AST2600 . But changing the ID is usually not recommended.

Offset: C04h **SCUC04: PCI Configuration Setting Register #2** **Init = 0x20001A03**

Bit	R/W	Reset	Description
31:16	RW	Rst181	**PCI Sub-System ID** The register can support firmware code the flexibility to change PCI Sub-System ID. Customers may change the ID according to their requirements.
15:0	RW	Rst181	**PCI Sub-Vendor ID** The register can support firmware code the flexibility to change PCI Sub-Vendor ID. Customers may change the ID according to their requirements.

Offset: C08h	SCUC08: PCI Configuration Setting Register #3	Init = 0x04000052

Bit	R/W	Reset	Description
31:8	RW	Rst182	**Class Code** The register can support firmware code the flexibility to change PCI Class Code ID of AST2600 . But changing the ID is usually not recommended.
7:0	RW	Rst182	**PCI Revision ID** The register can support firmware code the flexibility to change PCI Revision ID of AST2600 . But changing the ID is usually not recommended. 50: A0 51: A1 and A2 52: A3

Offset: C20h	SCUC20: PCI Express Configuration Setting Contol Register	Init = 0x000C007B

Bit	R/W	Reset	Description
31:22	RO	-	**Reserved(0)**
21	RW	Rst183	**Enable memory mapped LPC decode on BMC device**
20	RW	Rst183	**Enable memory mapped LPC decode on VGA device**
19	RW	Rst183	**Reserved**
18	RW	Rst183	**Enable MSI on EHCI device**
17	RW	Rst183	**Reserved**
16	RW	Rst183	**Enable EHCI device (device 2)**
15	RW	Rst183	**Enable E2L**
14	RW	Rst183	**Enable PCI Express Bus Master on BMC device**
13	RW	Rst183	**Enable interrupt on BMC device**
12	RW	Rst183	**Reserved**
11	RW	Rst183	**Enable MSI on BMC device**
10	RW	Rst183	**Enable relocate LPC IO on BMC device**
9	RW	Rst183	**Enable BMC MMIO on BMC device**
8	RW	Rst183	**Enable BMC device (device 1)**
7	RW	Rst183	**Disable MSI function on VGA device or BMC device**
6	RW	Rst183	**Enable PCI Express Bus Master on VGA device**
5	RW	Rst183	**Enable interrupt on VGA device**
4	RW	Rst183	**Reserved**
3	RW	Rst183	**Enable MSI on VGA device**
2	RW	Rst183	**Enable relocate LPC IO decode on VGA device**
1	RW	Rst183	**Enable BMC MMIO decode on VGA device**
0	RW	Rst183	**Enable VGA device (PCI device 0 configuration registers)**

Note :
This register must be initialized before PCI Express reset (PERST#) deasserted.

Offset: C24h	SCUC24: BMC MMIO Decode Setting Register	Init = 0

Bit	R/W	Reset	Description
31:24	RO	-	**Reserved(0)**
23	RW	Rst184	**Enable RC-L device DMA decoding**
22:20	RW	Rst184	**Reserved**

to next page

from previous page

19	RW	Rst184	**Enable RC-L device DMA mode**
18	RW	Rst184	**Enable PCI device 1 INTx/MSI from Host-to-BMC controller**
17	RW	Rst184	**Enable PCI device 1 INTx/MSI from SCU560[15]**
16:14	RW	Rst184	**Reserved**
13	RW	Rst184	**Enable MMIO decode area 256KB**
12	RW	Rst184	**Set MSI 32-bit mode**
11:10	RW	Rst184	**MSI Routing of PCIe-to-LPC** 00: through PCIe-to-PCI bridge 01: through PCI device 0 10: through PCI device 1 11: Reserved
9	RW	Rst184	**Enable E2L INTx/MSI in AST2600A3**
8	RW	Rst184	**Enable MMIO offset 0x0e000 to shared DRAM area location decode**
7	RW	Rst184	**Enable MMIO offset 0x0d000 to shared SRAM area location decode**
6	RW	Rst184	**Enable MMIO offset 0x0c000 to mailbox location decode**
5	RW	Rst184	**Enable MMIO offset 0x07000 to LPC or relocated controller decode**
4	RW	Rst184	**Enable MMIO offset 0x06000 to PECI or relocated controller decode**
3	RW	Rst184	**Enable MMIO offset 0x05000 to I2C controller decode**
2	RW	Rst184	**Enable MMIO offset 0x04000 to GPIO controller decode**
1	RW	Rst184	**Enable MMIO offset 0x03000 to PWM/TACH controller decode**
0	RW	Rst184	**Enable MMIO offset 0x02000 to ADC controller decode**

Offset: C28h	**SCUC28: First relocated controller decode area location**	**Init = 0**

Bit	R/W	Reset	Description
31:12	RW	Rst185	**First relocated controller decode area location**
11:0	RO	-	**Reserved(0)**

Offset: C2Ch	**SCUC2C: Second relocated controller decode area location**	**Init = 0**

Bit	R/W	Reset	Description
31:12	RW	Rst186	**Second relocated controller decode area location**
11:0	RO	-	**Reserved(0)**

Offset: C40h	**SCUC40: Mailbox decode area location**	**Init = 0**

Bit	R/W	Reset	Description
31:12	RW	Rst187	**Mailbox decode area location**
11:0	RO	-	**Reserved(0)**

Offset: C44h	**SCUC44: Shared memory area decode location 1**	**Init = 0**

Bit	R/W	Reset	Description
31:12	RW	Rst188	**Shared memory area decode location**
11:0	RO	-	**Reserved(0)**

Offset: C48h	SCUC48: Shared memory area decode location 2	Init = 0

Bit	R/W	Reset	Description
31:12	RW	Rst189	**Shared memory area decode location**
11:0	RO	-	**Reserved(0)**

Offset: C60h	SCUC60: BMC Device ID	Init = 0x24021A03

Bit	R/W	Reset	Description
31:16	RW	Rst191	**PCI BMC Device ID** The register can support firmware code the flexibility to change PCI Device ID of AST2600 . But changing the ID is usually not recommended.
15:0	RW	Rst191	**PCI BMC Vendor ID** The register can support firmware code the flexibility to change PCI Vendor ID of AST2600 . But changing the ID is usually not recommended.

Offset: C68h	SCUC68: BMC device class code and revision ID	Init = 0x0C070100

Bit	R/W	Reset	Description
31:0	Rst192	RW	**BMC device class code and revision ID**

Offset: C80h	SCUC80: EHCI Device ID	Init = 0x2503

Bit	R/W	Reset	Description
31:16	RO	-	**Reserved(0)**
15:0	RW	Rst193	**PCI EHCI Device ID** The register can support firmware code the flexibility to change PCI Vendor ID of AST2600 . But changing the ID is usually not recommended.

Offset: E00h–E2Ch	SCUE00 ~ SCUE2C: VGA Scratch Register	Init = 0

Offset	Bit	Attr.	Description
E00h	31:0	RO	**VGA scratch register bit[31:0]**
E04h	31:0	RO	**VGA scratch register bit[63:32]**
E08h	31:0	RO	**VGA scratch register bit[95:64]**
E0Ch	31:0	RO	**VGA scratch register bit[127:96]**
E20h	31:0	RO	**VGA scratch register bit[159:128]**
E24h	31:0	RO	**VGA scratch register bit[191:160]**
E28h	31:0	RO	**VGA scratch register bit[223:192]**
E2Ch	31:0	RO	**VGA scratch register bit[255:224]**

Note :
VGA scratch registers are designed for Host CPU to pass the necessary information to ARM CPU, especially for the needs of embedded firmware. All these registers can be read back by ARM CPU. The meaning of each bit is defined by software.

Offset: F00h	SCUF00: Write Protect Register #1	Init = 0

Bit	R/W	Reset	Description
31:27	RO	-	**Reserved(0)**
26	RW1S	Rst194	**Enable Write Protection of SCU0C8**

to next page

from previous page

25	RW1S	Rst194	**Enable Write Protection of SCU0C4**
24	RW1S	Rst194	**Enable Write Protection of SCU0C0**
23:17	RO	-	**Reserved(0)**
16	RW1S	Rst194	**Enable Write Protection of SCU080**
15:13	RO	-	**Reserved(0)**
12	RW1S	Rst194	**Enable Write Protection of SCU060**
11: 9	RO	-	**Reserved(0)**
8	RW1S	Rst194	**Enable Write Protection of SCU040**
7: 0	RO	-	**Reserved(0)**

Note :
Each bit can only be set. It needs reset to clear.

Offset: F04h **SCUF04: Write Protect Register #2** **Init = 0**

Bit	R/W	Reset	Description
31:24	RO	-	**Reserved(0)**
23	RW1S	Rst195	**Enable Write Protection of SCU1F0-SCU1FC**
22	RW1S	Rst195	**Enable Write Protection of SCU1E0-SCU1EC**
21	RW1S	Rst195	**Enable Write Protection of SCU1D0-SCU1DC**
20	RW1S	Rst195	**Enable Write Protection of SCU1C0-SCU1CC**
19	RW1S	Rst195	**Enable Write Protection of SCU1B0-SCU1BC**
18	RW1S	Rst195	**Enable Write Protection of SCU1A0-SCU1AC**
17	RW1S	Rst195	**Enable Write Protection of SCU190-SCU19C**
16	RW1S	Rst195	**Enable Write Protection of SCU180-SCU18C**
15: 4	RO	-	**Reserved(0)**
3	RW1S	Rst195	**Enable Write Protection of SCU10C**
2	RW1S	Rst195	**Enable Write Protection of SCU108**
1	RW1S	Rst195	**Enable Write Protection of SCU104**
0	RW1S	Rst195	**Enable Write Protection of SCU100**

Note :
Each bit can only be set. It needs reset to clear.

Offset: F08h **SCUF08: Write Protect Register #3** **Init = 0**

Bit	R/W	Reset	Description
31:14	RO	-	**Reserved(0)**
13	RW1S	Rst196	**Enable Write Protection of SCU264**
12	RW1S	Rst196	**Enable Write Protection of SCU260**
11:10	RO	-	**Reserved(0)**
9	RW1S	Rst196	**Enable Write Protection of SCU244**
8	RW1S	Rst196	**Enable Write Protection of SCU240**
7: 6	RO	-	**Reserved(0)**
5	RW1S	Rst196	**Enable Write Protection of SCU224**
4	RW1S	Rst196	**Enable Write Protection of SCU220**

to next page

from previous page

3: 2	RO	-	**Reserved(0)**
1	RW1S	Rst196	**Enable Write Protection of SCU204**
0	RW1S	Rst196	**Enable Write Protection of SCU200**

Note :
Each bit can only be set. It needs reset to clear.

Offset: F0Ch	**SCUF0C: Write Protect Register #4**		**Init = 0**
Bit	**R/W**	**Reset**	**Description**
31:30	RO	-	**Reserved(0)**
29	RW1S	Rst197	**Enable Write Protection of SCU3E4**
28	RW1S	Rst197	**Enable Write Protection of SCU3E0**
27	RW1S	Rst197	**Enable Write Protection of SCU3CC**
26	RW1S	Rst197	**Enable Write Protection of SCU3C8**
25	RW1S	Rst197	**Enable Write Protection of SCU3C4**
24	RW1S	Rst197	**Enable Write Protection of SCU3C0**
23:15	RO	-	**Reserved(0)**
14	RW1S	Rst197	**Enable Write Protection of SCU368**
13	RW1S	Rst197	**Enable Write Protection of SCU364**
12	RW1S	Rst197	**Enable Write Protection of SCU360**
11	RW1S	Rst197	**Enable Write Protection of SCU34C**
10	RW1S	Rst197	**Enable Write Protection of SCU348**
9	RO	-	**Reserved(0)**
8	RW1S	Rst197	**Enable Write Protection of SCU340**
7	RW1S	Rst197	**Enable Write Protection of SCU32C**
6	RO	-	**Reserved(0)**
5	RW1S	Rst197	**Enable Write Protection of SCU324**
4	RW1S	Rst197	**Enable Write Protection of SCU320**
3	RO	-	**Reserved(0)**
2	RW1S	Rst197	**Enable Write Protection of SCU308**
1	RW1S	Rst197	**Enable Write Protection of SCU304**
0	RW1S	Rst197	**Enable Write Protection of SCU300**

Note :
Each bit can only be set. It needs reset to clear.

Offset: F10h	**SCUF10: Write Protect Register #5**		**Init = 0**
Bit	**R/W**	**Reset**	**Description**
31	RW1S	Rst198	**Enable Write Protection of SCU0FC**
30:27	RO	-	**Reserved(0)**
26	RW1S	Rst198	**Enable Write Protection of SCU0D8**
25	RW1S	Rst198	**Enable Write Protection of SCU0D4**
24	RW1S	Rst198	**Enable Write Protection of SCU0D0**
23:18	RO	-	**Reserved(0)**

to next page

from previous page

17	RW1S	Rst198	**Enable Write Protection of SCU094**
16	RW1S	Rst198	**Enable Write Protection of SCU090**
15:14	RO	-	**Reserved(0)**
13	RW1S	Rst198	**Enable Write Protection of SCU074**
12	RW1S	Rst198	**Enable Write Protection of SCU070**
11: 9	RO	-	**Reserved(0)**
8	RW1S	Rst198	**Enable Write Protection of SCU050**
7: 0	RO	-	**Reserved(0)**

Note :
Each bit can only be set. It needs reset to clear.

Offset: F18h **SCUF18: Write Protect Register #6** **Init = 0**

Bit	R/W	Reset	Description
31: 2	RO	-	**Reserved(0)**
1	RW1S	Rst199	**Enable Write Protection of SCU214**
0	RW1S	Rst199	**Enable Write Protection of SCU210**

Note :
Each bit can only be set. It needs reset to clear.

Offset: F1Ch **SCUF1C: Write Protect Register #7** **Init = 0**

Bit	R/W	Reset	Description
31:15	RO	-	**Reserved(0)**
13	RW1S	Rst200	**Enable Write Protection of SCU374**
12	RW1S	Rst200	**Enable Write Protection of SCU370**
11	RW1S	Rst200	**Enable Write Protection of SCU35C**
10	RW1S	Rst200	**Enable Write Protection of SCU358**
9	RO	-	**Reserved(0)**
8	RW1S	Rst200	**Enable Write Protection of SCU350**
7	RW1S	Rst200	**Enable Write Protection of SCU33C**
6	RW1S	Rst200	**Enable Write Protection of SCU338**
5	RW1S	Rst200	**Enable Write Protection of SCU334**
4	RW1S	Rst200	**Enable Write Protection of SCU330**
3: 2	RO	-	**Reserved(0)**
1	RW1S	Rst200	**Enable Write Protection of SCU314**
0	RW1S	Rst200	**Enable Write Protection of SCU310**

Note :
Each bit can only be set. It needs reset to clear.

Offset: F20h **SCUF20: Write Protect Register #8** **Init = 0**

Bit	R/W	Reset	Description
31:30	RO	-	**Reserved(0)**
29	RW1S	Rst201	**Enable Write Protection of SCU4E4**

to next page

from previous page

28	RW1S	Rst201	**Enable Write Protection of SCU4E0**
27	RO	-	**Reserved(0)**
26	RW1S	Rst201	**Enable Write Protection of SCU4C8**
25	RW1S	Rst201	**Enable Write Protection of SCU4C4**
24	RW1S	Rst201	**Enable Write Protection of SCU4C0**
23: 9	RO	-	**Reserved(0)**
8	RW1S	Rst201	**Enable Write Protection of SCU440**
7: 4	RO	-	**Reserved(0)**
3	RW1S	Rst201	**Enable Write Protection of SCU40C**
2	RO	-	**Reserved(0)**
1	RW1S	Rst201	**Enable Write Protection of SCU404**
0	RW1S	Rst201	**Enable Write Protection of SCU400**
Note : Each bit can only be set. It needs reset to clear.			

Offset: F24h **SCUF24: Write Protect Register #9** **Init = 0**

Bit	R/W	Reset	Description
31:18	RO	-	**Reserved(0)**
17	RW1S	Rst202	**Enable Write Protection of SCU584**
16:13	RO	-	**Reserved(0)**
12	RW1S	Rst202	**Enable Write Protection of SCU560**
11:10	RO	-	**Reserved(0)**
9	RW1S	Rst202	**Enable Write Protection of SCU544**
8: 5	RO	-	**Reserved(0)**
4	RW1S	Rst202	**Enable Write Protection of SCU520**
3	RO	-	**Reserved(0)**
2	RW1S	Rst202	**Enable Write Protection of SCU508**
1:0	RO	-	**Reserved(0)**
Note : Each bit can only be set. It needs reset to clear.			

Offset: F30h **SCUF30: Write Protect Register #10** **Init = 0**

Bit	R/W	Reset	Description
31:27	RO	-	**Reserved(0)**
26	RW1S	Rst203	**Enable Write Protection of SCU4D8**
25	RW1S	Rst203	**Enable Write Protection of SCU4D4**
24	RW1S	Rst203	**Enable Write Protection of SCU4D0**
23	RW1S	Rst203	**Enable Write Protection of SCU4BC**
22	RW1S	Rst203	**Enable Write Protection of SCU4B8**
21	RW1S	Rst203	**Enable Write Protection of SCU4B4**
20	RW1S	Rst203	**Enable Write Protection of SCU4B0**
19:11	RO	-	**Reserved(0)**

to next page

from previous page

10	RW1S	Rst203	**Enable Write Protection of SCU458**
9	RW1S	Rst203	**Enable Write Protection of SCU454**
8	RW1S	Rst203	**Enable Write Protection of SCU450**
7	RO	-	**Reserved(0)**
6	RW1S	Rst203	**Enable Write Protection of SCU438**
5	RW1S	Rst203	**Enable Write Protection of SCU434**
4	RW1S	Rst203	**Enable Write Protection of SCU430**
3	RW1S	Rst203	**Enable Write Protection of SCU41C**
2	RW1S	Rst203	**Enable Write Protection of SCU418**
1	RW1S	Rst203	**Enable Write Protection of SCU414**
0	RW1S	Rst203	**Enable Write Protection of SCU410**

Note :
Each bit can only be set. It needs reset to clear.

Offset: F34h **SCUF34: Write Protect Register #11** **Init = 0**

Bit	R/W	Reset	Description
31:17	RO	-	**Reserved(0)**
16	RW1S	Rst204	**Enable Write Protection of SCU590**
15:13	RO	-	**Reserved(0)**
12	RW1S	Rst204	**Enable Write Protection of SCU570**
11: 9	RO	-	**Reserved(0)**
8	RW1S	Rst204	**Enable Write Protection of SCU550**
7: 5	RO	-	**Reserved(0)**
4	RW1S	Rst204	**Enable Write Protection of SCU530**
3	RO	-	**Reserved(0)**
2	RW1S	Rst204	**Enable Write Protection of SCU518**
1	RW1S	Rst204	**Enable Write Protection of SCU514**
0	RW1S	Rst204	**Enable Write Protection of SCU510**

Note :
Each bit can only be set. It needs reset to clear.

Offset: F38h **SCUF38: Write Protect Register #12** **Init = 0**

Bit	R/W	Reset	Description
31	RO	-	**Reserved(0)**
30	RW1S	Rst205	**Enable Write Protection of SCU6F8**
29:26	RO	-	**Reserved(0)**
25	RW1S	Rst205	**Enable Write Protection of SCU6D4**
24	RW1S	Rst205	**Enable Write Protection of SCU6D0**
23	RO	-	**Reserved(0)**
22	RW1S	Rst205	**Enable Write Protection of SCU6B8**
21:20	RO	-	**Reserved(0)**
19	RW1S	Rst205	**Enable Write Protection of SCU69C**

to next page

from previous page

18	RW1S	Rst205	**Enable Write Protection of SCU698**
17	RW1S	Rst205	**Enable Write Protection of SCU694**
16	RW1S	Rst205	**Enable Write Protection of SCU690**
15: 9	RO	-	**Reserved(0)**
8	RW1S	Rst205	**Enable Write Protection of SCU650**
7	RO	-	**Reserved(0)**
6	RW1S	Rst205	**Enable Write Protection of SCU638**
5	RW1S	Rst205	**Enable Write Protection of SCU634**
4	RW1S	Rst205	**Enable Write Protection of SCU630**
3	RW1S	Rst205	**Enable Write Protection of SCU61C**
2	RW1S	Rst205	**Enable Write Protection of SCU618**
1	RW1S	Rst205	**Enable Write Protection of SCU614**
0	RW1S	Rst205	**Enable Write Protection of SCU610**

Note :
Each bit can only be set. It needs reset to clear.

Offset: F3Ch **SCUF3C: Write Protect Register #13** **Init = 0**

Bit	R/W	Reset	Description
31: 1	RO	-	**Reserved(0)**
0	RW1S	Rst205	**Enable Write Protection of SCU710**

Note :
Each bit can only be set. It needs reset to clear.

Offset: F40h **SCUF40: Write Protect Register #14** **Init = 0**

Bit	R/W	Reset	Description
31:6	RO	-	**Reserved(0)**
5	RW1S	Rst206	**Enable Write Protection of SCU824**
4	RW1S	Rst206	**Enable Write Protection of SCU820**
3	RW1S	Rst206	**Enable Write Protection of SCU80C**
2	RW1S	Rst206	**Enable Write Protection of SCU808**
1	RO	-	**Reserved(0)**
0	RW1S	Rst206	**Enable Write Protection of SCU800**

Note :
Each bit can only be set. It needs reset to clear.

Offset: F48h **SCUF48: Write Protect Register #15** **Init = 0**

Bit	R/W	Reset	Description
31:11	RO	-	**Reserved(0)**
10	RW1S	Rst207	**Enable Write Protection of SCUA48**
9	RW1S	Rst207	**Enable Write Protection of SCUA44**
8	RW1S	Rst207	**Enable Write Protection of SCUA40**
7:4	RO	-	**Reserved(0)**

to next page

from previous page

3	RW1S	Rst207	**Enable Write Protection of SCUA0C**
2	RW1S	Rst207	**Enable Write Protection of SCUA08**
1	RW1S	Rst207	**Enable Write Protection of SCUA04**
0	RW1S	Rst207	**Enable Write Protection of SCUA00**

Note :
Each bit can only be set. It needs reset to clear.

Offset: F50h **SCUF50: Write Protect Register #16** **Init = 0**

Bit	R/W	Reset	Description
31:16	RO	-	**Reserved(0)**
15	RW1S	Rst208	**Enable Write Protection of SCU87C**
14	RW1S	Rst208	**Enable Write Protection of SCU878**
13	RW1S	Rst208	**Enable Write Protection of SCU874**
12	RW1S	Rst208	**Enable Write Protection of SCU870**
11	RW1S	Rst208	**Enable Write Protection of SCU85C**
10	RW1S	Rst208	**Enable Write Protection of SCU858**
9	RW1S	Rst208	**Enable Write Protection of SCU854**
8	RW1S	Rst208	**Enable Write Protection of SCU850**
7: 6	RO	-	**Reserved(0)**
Enable Write Pro-tection of SCUC285	RW1S	Rst208	**Enable Write Protection of SCU834**
4	RW1S	Rst208	**Enable Write Protection of SCU830**
3: 0	RO	-	**Reserved(0)**

Note :
Each bit can only be set. It needs reset to clear.

Offset: F60h **SCUF60: Write Protect Register #17** **Init = 0**

Bit	R/W	Reset	Description
31:14	RO	-	**Reserved(0)**
13	RW1S	Rst208	**Enable Write Protection of SCUC64**
12	RO	-	**Reserved(0)**
11	RW1S	Rst208	**Enable Write Protection of SCUC4C**
10	RW1S	Rst208	**Enable Write Protection of SCUC48**
9	RW1S	Rst208	**Enable Write Protection of SCUC44**
8	RW1S	Rst208	**Enable Write Protection of SCUC40**
7	RW1S	Rst208	**Enable Write Protection of SCUC2C**
6	RW1S	Rst208	**Enable Write Protection of SCUC28**
5	RW1S	Rst208	**Enable Write Protection of SCUC24**

to next page

from previous page

4	RW1S	Rst208	**Enable Write Protection of SCUC20**
3	RO	-	**Reserved(0)**
2	RW1S	Rst208	**Enable Write Protection of SCUC08**
1	RW1S	Rst208	**Enable Write Protection of SCUC04**
0	RW1S	Rst208	**Enable Write Protection of SCUC00**

Note :
Each bit can only be set. It needs reset to clear.

Offset: F7Ch **SCUF7C: Write Protect Register #18** **Init = 0**

Bit	R/W	Reset	Description
31	RW1S	Rst208	**Enable Write Protection of SCUFFC**
30:25	RO	-	**Reserved(0)**
24	RW1S	Rst208	**Enable Write Protection of SCUFD0**
23	RW1S	Rst208	**Enable Write Protection of SCUFBC**
22	RW1S	Rst208	**Enable Write Protection of SCUFB8**
21	RW1S	Rst208	**Enable Write Protection of SCUFB4**
20	RW1S	Rst208	**Enable Write Protection of SCUFB0**
19	RW1S	Rst208	**Enable Write Protection of SCUF9C**
18	RW1S	Rst208	**Enable Write Protection of SCUF98**
17	RO	-	**Reserved(0)**
16	RW1S	Rst208	**Enable Write Protection of SCUF90**
15:0	RO	-	**Reserved(0)**

Note :
Each bit can only be set. It needs reset to clear.

Offset: F80h **SCUF80: Reset Source Control Register #1** **Init = 0**

Bit	R/W	Reset	Description
31:27	RO	-	**Reserved(0)**
26	RW1S	RstFull	**Enable RstARM as reset source of SCU0C8**
25	RW1S	RstFull	**Enable RstARM as reset source of SCU0C4**
24	RW1S	RstFull	**Enable RstARM as reset source of SCU0C0**
23:17	RO	-	**Reserved(0)**
16	RW1S	RstFull	**Enable RstARM as reset source of SCU080**
15:13	RO	-	**Reserved(0)**
12	RW1S	RstFull	**Enable RstARM as reset source of SCU060**
11: 9	RO	-	**Reserved(0)**
8	RW1S	RstFull	**Enable RstARM as reset source of SCU040**
7: 0	RO	-	**Reserved(0)**

Note :
Each bit can only be set. It needs reset to clear.

Offset: F84h	SCUF84: Reset Source Control Register #2	Init = 0

Bit	R/W	Reset	Description
31:24	RO	-	**Reserved(0)**
23	RW1S	RstFull	**Enable RstARM as reset source of SCU1F0-SCU1FC**
22	RW1S	RstFull	**Enable RstARM as reset source of SCU1E0-SCU1EC**
21	RW1S	RstFull	**Enable RstARM as reset source of SCU1D0-SCU1DC**
20	RW1S	RstFull	**Enable RstARM as reset source of SCU1C0-SCU1CC**
19	RW1S	RstFull	**Enable RstARM as reset source of SCU1B0-SCU1BC**
18	RW1S	RstFull	**Enable RstARM as reset source of SCU1A0-SCU1AC**
17	RW1S	RstFull	**Enable RstARM as reset source of SCU190-SCU19C**
16	RW1S	RstFull	**Enable RstARM as reset source of SCU180-SCU18C**
15: 4	RO	-	**Reserved(0)**
3	RW1S	RstFull	**Enable RstARM as reset source of SCU10C**
2	RW1S	RstFull	**Enable RstARM as reset source of SCU108**
1	RW1S	RstFull	**Enable RstARM as reset source of SCU104**
0	RW1S	RstFull	**Enable RstARM as reset source of SCU100**

Note :
Each bit can only be set. It needs reset to clear.

Offset: F88h	SCUF88: Reset Source Control Register #3	Init = 0

Bit	R/W	Reset	Description
31:14	RO	-	**Reserved(0)**
13	RW1S	RstFull	**Enable RstARM as reset source of SCU264**
12	RW1S	RstFull	**Enable RstARM as reset source of SCU260**
11:10	RO	-	**Reserved(0)**
9	RW1S	RstFull	**Enable RstARM as reset source of SCU244**
8	RW1S	RstFull	**Enable RstARM as reset source of SCU240**
7: 6	RO	-	**Reserved(0)**
5	RW1S	RstFull	**Enable RstARM as reset source of SCU224**
4	RW1S	RstFull	**Enable RstARM as reset source of SCU220**
3: 2	RO	-	**Reserved(0)**
1	RW1S	RstFull	**Enable RstARM as reset source of SCU204**
0	RW1S	RstFull	**Enable RstARM as reset source of SCU200**

Note :
Each bit can only be set. It needs reset to clear.

Offset: F8Ch	SCUF8C: Reset Source Control Register #4	Init = 0

Bit	R/W	Reset	Description
31:30	RO	-	**Reserved(0)**
29	RW1S	RstFull	**Enable RstARM as reset source of SCU3E4**
28	RW1S	RstFull	**Enable RstARM as reset source of SCU3E0**
27	RW1S	RstFull	**Enable RstARM as reset source of SCU3CC**
26	RW1S	RstFull	**Enable RstARM as reset source of SCU3C8**

to next page

from previous page

25	RW1S	RstFull	**Enable RstARM as reset source of SCU3C4**
24	RW1S	RstFull	**Enable RstARM as reset source of SCU3C0**
23:15	RO	-	**Reserved(0)**
14	RW1S	RstFull	**Enable RstARM as reset source of SCU368**
13	RW1S	RstFull	**Enable RstARM as reset source of SCU364**
12	RW1S	RstFull	**Enable RstARM as reset source of SCU360**
11	RW1S	RstFull	**Enable RstARM as reset source of SCU34C**
10	RW1S	RstFull	**Enable RstARM as reset source of SCU348**
9	RO	-	**Reserved(0)**
8	RW1S	RstFull	**Enable RstARM as reset source of SCU340**
7	RW1S	RstFull	**Enable RstARM as reset source of SCU32C**
6	RO	-	**Reserved(0)**
5	RW1S	RstFull	**Enable RstARM as reset source of SCU324**
4	RW1S	RstFull	**Enable RstARM as reset source of SCU320**
3	RO	-	**Reserved(0)**
2	RW1S	RstFull	**Enable RstARM as reset source of SCU308**
1	RW1S	RstFull	**Enable RstARM as reset source of SCU304**
0	RW1S	RstFull	**Enable RstARM as reset source of SCU300**

Note :
Each bit can only be set. It needs reset to clear.

Offset: F90h **SCUF90: Reset Source Control Register #5** **Init = 0**

Bit	R/W	Reset	Description
31	RW1S	RstFull	**Enable RstARM as reset source of SCU0FC**
30:27	RO	-	**Reserved(0)**
26	RW1S	RstFull	**Enable RstARM as reset source of SCU0D8**
25	RW1S	RstFull	**Enable RstARM as reset source of SCU0D4**
24	RW1S	RstFull	**Enable RstARM as reset source of SCU0D0**
23:18	RO	-	**Reserved(0)**
17	RW1S	RstFull	**Enable RstARM as reset source of SCU094**
16	RW1S	RstFull	**Enable RstARM as reset source of SCU090**
15:14	RO	-	**Reserved(0)**
13	RW1S	RstFull	**Enable RstARM as reset source of SCU074**
12	RW1S	RstFull	**Enable RstARM as reset source of SCU070**
11: 9	RO	-	**Reserved(0)**
8	RW1S	RstFull	**Enable RstARM as reset source of SCU050**
7: 0	RO	-	**Reserved(0)**

Note :
Each bit can only be set. It needs reset to clear.

Offset: F98h	SCUF98: Reset Source Control Register #6	Init = 0

Bit	R/W	Reset	Description
31: 2	RO	-	**Reserved(0)**
1	RW1S	RstFull	**Enable RstARM as reset source of SCU214**
0	RW1S	RstFull	**Enable RstARM as reset source of SCU210**

Note :
Each bit can only be set. It needs reset to clear.

Offset: F9Ch	SCUF9C: Reset Source Control Register #7	Init = 0

Bit	R/W	Reset	Description
31:15	RO	-	**Reserved(0)**
13	RW1S	RstFull	**Enable RstARM as reset source of SCU374**
12	RW1S	RstFull	**Enable RstARM as reset source of SCU370**
11	RW1S	RstFull	**Enable RstARM as reset source of SCU35C**
10	RW1S	RstFull	**Enable RstARM as reset source of SCU358**
9	RO	-	**Reserved(0)**
8	RW1S	RstFull	**Enable RstARM as reset source of SCU350**
7	RW1S	RstFull	**Enable RstARM as reset source of SCU33C**
6	RW1S	RstFull	**Enable RstARM as reset source of SCU338**
5	RW1S	RstFull	**Enable RstARM as reset source of SCU334**
4	RW1S	RstFull	**Enable RstARM as reset source of SCU330**
3: 2	RO	-	**Reserved(0)**
1	RW1S	RstFull	**Enable RstARM as reset source of SCU314**
0	RW1S	RstFull	**Enable RstARM as reset source of SCU310**

Note :
Each bit can only be set. It needs reset to clear.

Offset: FA0h	SCUFA0: Reset Source Control Register #8	Init = 0

Bit	R/W	Reset	Description
31:30	RO	-	**Reserved(0)**
29	RW1S	RstFull	**Enable RstARM as reset source of SCU4E4**
28	RW1S	RstFull	**Enable RstARM as reset source of SCU4E0**
27	RO	-	**Reserved(0)**
26	RW1S	RstFull	**Enable RstARM as reset source of SCU4C8**
25	RW1S	RstFull	**Enable RstARM as reset source of SCU4C4**
24	RW1S	RstFull	**Enable RstARM as reset source of SCU4C0**
23: 9	RO	-	**Reserved(0)**
8	RW1S	RstFull	**Enable RstARM as reset source of SCU440**
7: 4	RO	-	**Reserved(0)**
3	RW1S	RstFull	**Enable RstARM as reset source of SCU40C**
2	RO	-	**Reserved(0)**
1	RW1S	RstFull	**Enable RstARM as reset source of SCU404**
0	RW1S	RstFull	**Enable RstARM as reset source of SCU400**

to next page

from previous page

Note :
Each bit can only be set. It needs reset to clear.

Offset: FA4h			**SCUFA4: Reset Source Control Register #9**	**Init = 0**
Bit	**R/W**	**Reset**	**Description**	
31:18	RO	-	**Reserved(0)**	
17	RW1S	RstFull	**Enable RstARM as reset source of SCU584**	
16:13	RO	-	**Reserved(0)**	
12	RW1S	RstFull	**Enable RstARM as reset source of SCU560**	
11:10	RO	-	**Reserved(0)**	
9	RW1S	RstFull	**Enable RstARM as reset source of SCU544**	
8: 5	RO	-	**Reserved(0)**	
4	RW1S	RstFull	**Enable RstARM as reset source of SCU520**	
3	RO	-	**Reserved(0)**	
2	RW1S	RstFull	**Enable RstARM as reset source of SCU508**	
1:0	RO	-	**Reserved(0)**	

Note :
Each bit can only be set. It needs reset to clear.

Offset: FB0h			**SCUFB0: Reset Source Control Register #10**	**Init = 0**
Bit	**R/W**	**Reset**	**Description**	
31:27	RO	-	**Reserved(0)**	
26	RW1S	RstFull	**Enable RstARM as reset source of SCU4D8**	
25	RW1S	RstFull	**Enable RstARM as reset source of SCU4D4**	
24	RW1S	RstFull	**Enable RstARM as reset source of SCU4D0**	
23	RW1S	RstFull	**Enable RstARM as reset source of SCU4BC**	
22	RW1S	RstFull	**Enable RstARM as reset source of SCU4B8**	
21	RW1S	RstFull	**Enable RstARM as reset source of SCU4B4**	
20	RW1S	RstFull	**Enable RstARM as reset source of SCU4B0**	
19:11	RO	-	**Reserved(0)**	
10	RW1S	RstFull	**Enable RstARM as reset source of SCU458**	
9	RW1S	RstFull	**Enable RstARM as reset source of SCU454**	
8	RW1S	RstFull	**Enable RstARM as reset source of SCU450**	
7	RO	-	**Reserved(0)**	
6	RW1S	RstFull	**Enable RstARM as reset source of SCU438**	
5	RW1S	RstFull	**Enable RstARM as reset source of SCU434**	
4	RW1S	RstFull	**Enable RstARM as reset source of SCU430**	
3	RW1S	RstFull	**Enable RstARM as reset source of SCU41C**	
2	RW1S	RstFull	**Enable RstARM as reset source of SCU418**	
1	RW1S	RstFull	**Enable RstARM as reset source of SCU414**	
0	RW1S	RstFull	**Enable RstARM as reset source of SCU410**	

Note :
Each bit can only be set. It needs reset to clear.

Offset: FB4h	SCUFB4: Reset Source Control Register #11	Init = 0

Bit	R/W	Reset	Description
31:17	RO	-	**Reserved(0)**
16	RW1S	RstFull	**Enable RstARM as reset source of SCU590**
15:13	RO	-	**Reserved(0)**
12	RW1S	RstFull	**Enable RstARM as reset source of SCU570**
11: 9	RO	-	**Reserved(0)**
8	RW1S	RstFull	**Enable RstARM as reset source of SCU550**
7: 5	RO	-	**Reserved(0)**
4	RW1S	RstFull	**Enable RstARM as reset source of SCU530**
3	RO	-	**Reserved(0)**
2	RW1S	RstFull	**Enable RstARM as reset source of SCU518**
1	RW1S	RstFull	**Enable RstARM as reset source of SCU514**
0	RW1S	RstFull	**Enable RstARM as reset source of SCU510**

Note :
Each bit can only be set. It needs reset to clear.

Offset: FB8h	SCUFB8: Reset Source Control Register #12	Init = 0

Bit	R/W	Reset	Description
31	RO	-	**Reserved(0)**
30	RW1S	RstFull	**Enable RstARM as reset source of SCU6F8**
29:26	RO	-	**Reserved(0)**
25	RW1S	RstFull	**Enable RstARM as reset source of SCU6D4**
24	RW1S	RstFull	**Enable RstARM as reset source of SCU6D0**
23	RO	-	**Reserved(0)**
22	RW1S	RstFull	**Enable RstARM as reset source of SCU6B8**
21:20	RO	-	**Reserved(0)**
19	RW1S	RstFull	**Enable RstARM as reset source of SCU69C**
18	RW1S	RstFull	**Enable RstARM as reset source of SCU698**
17	RW1S	RstFull	**Enable RstARM as reset source of SCU694**
16	RW1S	RstFull	**Enable RstARM as reset source of SCU690**
15: 9	RO	-	**Reserved(0)**
8	RW1S	RstFull	**Enable RstARM as reset source of SCU650**
7	RO	-	**Reserved(0)**
6	RW1S	RstFull	**Enable RstARM as reset source of SCU638**
5	RW1S	RstFull	**Enable RstARM as reset source of SCU634**
4	RW1S	RstFull	**Enable RstARM as reset source of SCU630**
3	RW1S	RstFull	**Enable RstARM as reset source of SCU61C**
2	RW1S	RstFull	**Enable RstARM as reset source of SCU618**
1	RW1S	RstFull	**Enable RstARM as reset source of SCU614**
0	RW1S	RstFull	**Enable RstARM as reset source of SCU610**

Note :
Each bit can only be set. It needs reset to clear.

Offset: FBCh	SCUFB8: Reset Source Control Register #13	Init = 0

Bit	R/W	Reset	Description
31: 1	RO	-	**Reserved(0)**
0	RW1S	RstFull	**Enable RstARM as reset source of SCU710**

Offset: FC0h	SCUFC0: Reset Source Control Register #13	Init = 0

Bit	R/W	Reset	Description
31:6	RO	-	**Reserved(0)**
5	RW1S	RstFull	**Enable RstARM as reset source of SCU824**
4	RW1S	RstFull	**Enable RstARM as reset source of SCU820**
3	RW1S	RstFull	**Enable RstARM as reset source of SCU80C**
2	RW1S	RstFull	**Enable RstARM as reset source of SCU808**
1	RO	-	**Reserved(0)**
0	RW1S	RstFull	**Enable RstARM as reset source of SCU800**

Note :
Each bit can only be set. It needs reset to clear.

Offset: FC8h	SCUFC8: Reset Source Control Register #14	Init = 0

Bit	R/W	Reset	Description
31:11	RO	-	**Reserved(0)**
10	RW1S	RstFull	**Enable RstARM as reset source of SCUA48**
9	RW1S	RstFull	**Enable RstARM as reset source of SCUA44**
8	RW1S	RstFull	**Enable RstARM as reset source of SCUA40**
7:4	RO	-	**Reserved(0)**
3	RW1S	RstFull	**Enable RstARM as reset source of SCUA0C**
2	RW1S	RstFull	**Enable RstARM as reset source of SCUA08**
1	RW1S	RstFull	**Enable RstARM as reset source of SCUA04**
0	RW1S	RstFull	**Enable RstARM as reset source of SCUA00**

Note :
Each bit can only be set. It needs reset to clear.

Offset: FD0h	SCUFD0: Reset Source Control Register #18	Init = 0

Bit	R/W	Reset	Description
31:16	RO	-	**Reserved(0)**
15	RW1S	RstFull	**Enable RstARM as reset source of SCU87C**
14	RW1S	RstFull	**Enable RstARM as reset source of SCU878**
13	RW1S	RstFull	**Enable RstARM as reset source of SCU874**
12	RW1S	RstFull	**Enable RstARM as reset source of SCU870**
11	RW1S	RstFull	**Enable RstARM as reset source of SCU85C**
10	RW1S	RstFull	**Enable RstARM as reset source of SCU858**
9	RW1S	RstFull	**Enable RstARM as reset source of SCU854**
8	RW1S	RstFull	**Enable RstARM as reset source of SCU850**
7: 6	RO	-	**Reserved(0)**

to next page

from previous page

5	RW1S	RstFull	**Enable RstARM as reset source of SCU834**
4	RW1S	RstFull	**Enable RstARM as reset source of SCU830**
3: 0	RO	-	**Reserved(0)**

Note :
Each bit can only be set. It needs reset to clear.

Offset: FE0h **SCUFE0: Reset Source Control Register #15** **Init = 0**

Bit	R/W	Reset	Description
31:17	RO	-	**Reserved(0)**
16	RW1S	RstFull	**Enable RstARM as reset source of SCUC80**
15	RO	-	**Reserved(0)**
14	RW1S	RstFull	**Enable RstARM as reset source of SCUC68**
13	RO	-	**Reserved(0)**
12	RW1S	RstFull	**Enable RstARM as reset source of SCUC60**
11	RW1S	RstFull	**Enable RstARM as reset source of SCUC4C**
10	RW1S	RstFull	**Enable RstARM as reset source of SCUC48**
9	RW1S	RstFull	**Enable RstARM as reset source of SCUC44**
8	RW1S	RstFull	**Enable RstARM as reset source of SCUC40**
7	RW1S	RstFull	**Enable RstARM as reset source of SCUC2C**
6	RW1S	RstFull	**Enable RstARM as reset source of SCUC28**
5	RW1S	RstFull	**Enable RstARM as reset source of SCUC24**
4	RW1S	RstFull	**Enable RstARM as reset source of SCUC20**
3	RO	-	**Reserved(0)**
2	RW1S	RstFull	**Enable RstARM as reset source of SCUC08**
1	RW1S	RstFull	**Enable RstARM as reset source of SCUC04**
0	RW1S	RstFull	**Enable RstARM as reset source of SCUC00**

Note :
Each bit can only be set. It needs reset to clear.

Offset: FECh **SCUFEC: Reset Source Control Register #16** **Init = 0**

Bit	R/W	Reset	Description
31:16	RO	-	**Reserved(0)**
15	RW1S	RstFull	**Enable RstARM as reset source of SCUF6C**
14:13	RO	-	**Reserved(0)**
12	RW1S	RstFull	**Enable RstARM as reset source of SCUF60**
11	RO	-	**Reserved(0)**
10	RW1S	RstFull	**Enable RstARM as reset source of SCUF48**
9	RO	-	**Reserved(0)**
8	RW1S	RstFull	**Enable RstARM as reset source of SCUF40**
7:6	RO	-	**Reserved(0)**
5	RW1S	RstFull	**Enable RstARM as reset source of SCUF24**
4	RW1S	RstFull	**Enable RstARM as reset source of SCUF20**

to next page

from previous page

3	RW1S	RstFull	**Enable RstARM as reset source of SCUF0C**
2	RW1S	RstFull	**Enable RstARM as reset source of SCUF08**
1	RW1S	RstFull	**Enable RstARM as reset source of SCUF04**
0	RW1S	RstFull	**Enable RstARM as reset source of SCUF00**
Note : Each bit can only be set. It needs reset to clear.			

Offset: FFCh	**SCUFFC: Reset Source Control Register #17**		**Init = 0**
Bit	**R/W**	**Reset**	**Description**
31:16	RO	-	**Reserved(0)**
15	RW1S	RstFull	**Enable RstARM as reset source of SCUF7C**
14:7	RO	-	**Reserved(0)**
6	RW1S	RstFull	**Enable RstARM as reset source of SCUF38**
5	RW1S	RstFull	**Enable RstARM as reset source of SCUF34**
4	RW1S	RstFull	**Enable RstARM as reset source of SCUF30**
3	RW1S	RstFull	**Enable RstARM as reset source of SCUF1C**
2	RW1S	RstFull	**Enable RstARM as reset source of SCUF18**
1	RO	-	**Reserved(0)**
0	RW1S	RstFull	**Enable RstARM as reset source of SCUF10**
Note : Each bit can only be set. It needs reset to clear.			

25 Hash & Crypto Engine (HACE)

25.1 Overview

Hash and Crypto Engine (HACE) is designed to accelerate the throughput of **hash data digest**, **encryption**, and **decryption**. Basically, HACE can be divided into two independently engines — Hash Engine and Crypto Engine. Each of which can work independently. The two engines can also be programmed to work at cascaded mode, either hash first or crypto first. Working at cascaded mode can significantly reduce memory bandwidth requirement. HACE can directly fetch data through memory bus. Therefore, HACE will not result in AHB bus congestions.

HACE only implements 17 sets of 32-bit registers to program the various supported functions. The physical address of these registers can be derived as the following:

Base address of HACE = 0x1E6D_0000
Physical address = (Base address of HACE) + Offset

HACE00: Crypto Data Source Base Address Register
HACE04: Crypto Data Destination Base Address Register
HACE08: Crypto Context Buffer Base Address Register
HACE0C: Crypto Data Length Register
HACE10: Crypto Engine Command Register
HACE14: Crypto AES-GCM Additional Data Length Register
HACE18: Crypto AES-GCM Tag Write Buffer Base Address Register
HACE1C: HAC Engine Status Register
HACE20: Hash Data Source Base Address Register
HACE24: Hash Digest Write Buffer Base Address Register
HACE28: Hash HMAC Key Buffer Base Address Register
HACE2C: Hash Data Length Register
HACE30: Hash Engine Command Register
HACE34: Hash Data Padding Length Register
HACE50: Command Queue Base Address
HACE54: Command Queue End Pointer
HACE58: Command Queue Write Pointer
HACE5C: Command Queue Read Pointer
HACE60: HAC Engine Feature Register
HACE64: HAC Software Tag Register

25.2 Features

- Directly connected to AHB bus
- Register programming through AHB bus interface
- Supports Advanced Encryption Standard (AES) with options:
  - Electronic Code Book (ECB)
  - Cipher Block Chaining (CBC)
  - Cipher Feedback (CFB)
  - Output Feedback (OFB)
  - Counter (CTR)
  - Galois/Counter (GCM)
  - Support Three Different Key Sizes : 128, 192 or 256 bits
- Supports Data Encryption Standard (DES) with options:

    - Electronic Code Book (ECB)
    - Cipher Block Chaining (CBC)
    - Cipher Feedback (CFB)
    - Output Feedback (OFB)
    - Counter (CTR)
- Support RC4 Encryption Standard
- Support AES/RC4 key expansion by hardware and pre-loading into DRAM memory
- Support AES/RC4 key expansion by software and pre-loading into DRAM memory
- Support DES key expansion by hardware and pre-loading into DRAM memory
- Support multiple message digest standards:
    - MD5, SHA-1
    - SHA-224, SHA-256
    - SHA-384, SHA-512, SHA-512/224, SHA-512/256
    - HMAC-MD5, HMAC-SHA-1
    - HMAC-SHA-224, HMAC-SHA-256
    - HMAC-SHA-384, HMAC-SHA-512, HMAC-SHA-512/224, HMAC-SHA-512/256
- Hash function support length up to 256 MByte.
- Support 4 types of engine trigger modes:
    - Encryption/decryption only
    - Message digest only
    - Encryption/decryption first, message digest second
    - Message digest first, encryption/decryption second
- Crypto and hash engine support independent and cascaded mode.
- Engine fired by directly writing command into command register
- Support CPU Interrupt option
- Programmable DES/AES/RC4 encryption or decryption mode with programmable context buffer location for fast context switching.
- Internal Key context memory.
- Programmable key context management.
- Programmable address of source buffer & destination buffer
- Programmable address of expanded key buffer
- Direct DRAM memory access for:
    - Expanded key loading
    - Hash input data read-in
    - Hash digest write-back
    - Plaintext/Ciphertext read-in
    - Ciphertext/Plaintext write-back

25.3 Registers : Base Address = 0x1E6D:0000

Offset: 00h	HACE00: Crypto Data Source Base Address Register	Init = X
Bit	**R/W**	**Description**
Scatter-Gather Mode		
31	RO	**Reserved (0)**
30:3	RW	**Base address of scatter-gather list for crypto source data[30:3] (8-byte aligned)**
2 :0	RO	**Reserved (0)**
Direct Access Mode		
31	RO	**Reserved (0)**
30:0	RW	**Base address of crypto source data[30:0] (byte aligned)**
Note : When crypto engine works in cascaded mode (Hash first, crypto second), HACE00 MUST equal to HACE20.		

Offset: 04h	HACE04: Crypto Data Destination Base Address Register	Init = X
Bit	**R/W**	**Description**
Scatter-Gather Mode		
31	RO	**Reserved (0)**
30:3	RW	**Base address of scatter-gather list for crypto destination data[30:3] (8-byte aligned)**
2 :0	RO	**Reserved (0)**
Direct Access Mode		
31	RO	**Reserved (0)**
30:0	RW	**Base address of crypto destination data[30:0] (byte aligned)**
Note : When crypto engine works in cascaded mode (Crypto first, hash second), HACE04 MUST equal to HACE20.		

Offset: 08h	HACE08: Crypto Context Buffer Base Address Registerr	Init = X
Bit	**R/W**	**Description**
31	RO	**Reserved (0)**
30:3	RW	**Base address of crypto context buffer[30:3] (8-byte aligned)**
2 :0	RO	**Reserved (0)**

Offset: 0Ch	HACE0C: Crypto Data Length Register	Init = X
Bit	**R/W**	**Description**
31:28	RO	**Reserved (0)**

to next page

from previous page

27:0	RW	**Crypto data length (bytes)** 0: **Invalid** (ONLY **AES-GCM** can support 0 byte confidential data) 1: 1 byte 2: 2 bytes ... In AES-GCM mode, this register value is the lenght of the confidential data. The maximum data length is up to **(256MB-1) bytes** for a **RC4/AES/DES** crypto command.

Function	HACE10[8]	HACE10[16]	HACE10[6:4]	Alignment	Minimum
RC4	1	-	0	1-byte aligned	1h
AES (GCM)	0	0	5	1-byte aligned	0h
AES (others)	0	0	0,1,2,3,4	16-byte aligned	10h
DES	0	1	0,1,2,3,4	8-byte aligned	8h

Note :
When crypto engine works in cascaded mode, HACE0C MUST equal to HACE2C.

Offset: 10h **HACE10: Crypto Engine Command Register** **Init = 0**

Bit	R/W	Description
31:25	RO	**Reserved (0)**
24	RW	**AES Key source selection** 0: AES Key from DRAM (Crypto Context Buffer) 1: AES Key from Secure Valut Key SEC80C When setting HACE10[24] = 1, the HACE10[13] must be value 1
23	RW	**GHASH Tag output XOR control bit** 0: Enable GHASH Tag XOR with the output of GCTR 1: Disable GHASH Tag XOR with the output of GCTR
22	RW	**GHASH padding block length field selection** 0: Length field is normal order(len(A) || len(D)) 1: Length field is inverse order(len(D) || len(A))
21	RW	**AES-GCM Tag write address selection** 0: Attaching the AES-GCM Tag to the end of the crypto destination data 1: Writing the AES-GCM Tag to the Tag write buffer base address(HACE18)
20	RW	**M-Bus request synchronization for Crypto Engine Idle** 0: Disable M-Bus request synchronization 1: Enable M-Bus request synchronization
19	RW	**Crypto destination data scatter-gather control** 0: Direct access mode 1: Scatter-gather mode
18	RW	**Crypto source data scatter-gather control** 0: Direct access mode 1: Scatter-gather mode
17	RW	**Enable triple DES** 0: Single DES 1: Triple DES
16	RW	**AES/DES engine selection** 0: AES Engine 1: DES Engine

to next page

from previous page

15:14	RW	**AES/DES CTR mode control** HACE10[15:14] / AES / DES 0 / 128 bits / 64 bits 1 / 96 bits / 32 bits 2 / 64 bits / reserved 3 / 32 bits / reserved
13	RW	**AES key expansion selection** 0: Software key expansion 1: Hardware key expansion
12	RW	**Enable crypto interrupt** 0: Disable crypto interrupt 1: Enable crypto interrupt when crypto command finished
11	RW	~~**Disable crypto engine read-in & write-out data control**~~ ~~0: Enable crypto engine read-in & write-out datat~~ ~~1: Disable crypto engine read-in & write-out data~~
10	RW	~~**Disable loading context data from context buffer**~~ ~~0: Enable loading context data from context buffer before running crypto algorithm~~ ~~1: Disable loading context data from context buffer before running crypto algorithm~~
9	RW	**Disable saving context data into context buffer** 0: Enable saving context data into context buffer when finished crypto command 1: Disable saving context data into context buffer when finished crypto command
8	RW	**Crypto algorithm selection** 0: Select AES/DES crypto algorithm 1: Select RC4 crypto algorithm
7	RW	**Crypto mode selection** 0: Decryption mode (ciphertext in, plaintext out) 1: Encryption mode (plaintext in, ciphertext out)
6 :4	RW	**AES/DES operation mode selection** 000: AES/DES ECB mode (Initial Vector is **NOT** required) 001: AES/DES CBC mode (Initial Vector is required) 010: AES/DES CFB mode (Initial Vector is required) 011: AES/DES OFB mode (Initial Vector is required) 100: AES/DES CTR mode (Initial Vector is required) 101: AES-GCM mode (Initial Vector is required) 110: **Invalid** 111: **Invalid** Initial Vector is presented in the context buffer. This register in only applied to AES/DES crypto algorithm. When RC4 crypto algorithm is selected, this register will be ignored.
3 :2	RW	**Key length of AES crypto algorithm** 00: 128-bit key length 01: 192-bit key length 10: 256-bit key length 11: **Invalid** This register is only applied to AES crypto algorithm. When RC4 crypto algorithm is selected, this register will be ignored.

to next page

from previous page

1 :0	RW	**Crypto engine operation mode control** 00: Crypto engine works in independent mode 01: Crypto engine works in independent mode 10: Crypto engine works in cascaded mode (Crypto first, hash second) 11: Crypto engine works in cascaded mode (Hash first, crypto second) In cascaded mode, the programming of HACE10 [1:0] MUST be consistent with the programming of HACE30 [1:0]. Otherwise, HACE may be trapped in a dead lock.

Offset: 14h **HACE14: Crypto AES-GCM Additional Data Length Register** **Init = X**

Bit	R/W	Description
31:28	RO	**Reserved (0)**
27:0	RW	**Crypto AES-GCM additional data length (bytes)** 0: 0 byte 1: 1 byte 2: 2 bytes ... In AES-GCM mode, this register value is the lenght of the additional authenticated data. The data length is from **0 byte** to **(256MB-1) bytes** for an **AES-GCM** crypto command.

Note :
When setting HACE10[8] = 0, and HACE10[16] = 0, and HACE10[6:4] = 5 (**AES-GCM mode**), the crypto engine would use this register.

Offset: 18h **HACE18: Crypto AES-GCM Tag Write Buffer Base Address Register** **Init = X**

Bit	R/W	Description
31	RO	**Reserved (0)**
30:0	RW	**Base address of crypto AES-GCM Tag write buffer[30:0] (byte aligned)** The authentication tag of AES-GCM is **16 bytes**.

Note :
When setting HACE10[8] = 0, and HACE10[16] = 0, and HACE10[6:4] = 5 (**AES-GCM mode**), and HACE10[21] = 0, the crypto engine would use this register.

Offset: 1Ch **HACE1C: HAC Engine Status Register** **Init = 0**

Bit	R/W	Description
31	RO	**CQ data bffer format error** **(for debugging purpose only)**
30	RO	**Crypto output data scatter and gather list buffer overflow** **(for debugging purpose only)**
29	RO	**Crypto input data scatter and gather list buffer overflow** **(for debugging purpose only)**
28	RO	**Hash input data scatter and gather list buffer overflow** **(for debugging purpose only)**
27	RO	**CQ low data buffer overflow** **(for debugging purpose only)**
26	RO	**CQ high data buffer overflow** **(for debugging purpose only)**
25	RO	**Crypto input data buffer overflow** **(for debugging purpose only)**
24	RO	**Crypto input rorate data overflow** **(for debugging purpose only)**

to next page

from previous page

23	RO	**Hash input data buffer overflow** **(for debugging purpose only)**
22	RO	**Hash input rorate data overflow** **(for debugging purpose only)**
21	RO	**Crypto output data scatter and gather list total length mismatch** **(for debugging purpose only)**
20	RO	**Crypto input data scatter and gather list total length mismatch** **(for debugging purpose only)**
19	RO	**Hash input data scatter and gather list total length mismatch** **(for debugging purpose only)**
18:16	RO	**Reserved (0)**
15	RW	**Write software tag interrupt flag** 0: No interrupt 1: Interrupt is pending When write software tag interrupt is enabled, this bit will be set to ”1” when write value to HAC software tag register. Writing ”1” to this bit will clear this register.
14:13	RO	**Reserved (0)**
12	RW	**Crypto interrupt flag** 0: No interrupt 1: Interrupt is pending When crypto command has been finished, this bit will be set to ”1”. Writing ”1” to this bit will clear this register. The HAC10[12] dsiable(”0”) or enable(”1”) Crypto interrupt event.
11:10	RO	**Reserved (0)**
9	RW	**Hash interrupt flag** 0: No interrupt 1: Interrupt is pending When hash command has been finished, this bit will be set to ”1”. Writing ”1” to this bit will clear this register. The HAC30[9] dsiable(”0”) or enable(”1”) hash interrupt event.
8 :4	RO	**Reserved (0)**
3	RO	**Command queue status flag** 0: Command queue is idle 1: Command queue is busy
2	RO	**Reserved (0)**
1	RO	**Crypto engine status flag** 0: Crypto engine is idle 1: Crypto engine is busy
0	RO	**Hash engine status flag** 0: Hash engine is idle 1: Hash engine is busy

Offset: 20h	**HACE20: Hash Data Source Base Address Register**	**Init = X**
Bit	**R/W**	**Description**
	Scatter-Gather Mode	
31	RO	**Reserved (0)**

to next page

from previous page

30:3	RW	**Base address of scatter-gather list for hash source data[30:3] (8-byte aligned)**
2 :0	RO	**Reserved (0)**
Direct Access Mode		
31	RO	**Reserved (0)**
30:0	RW	**Base address of hash source data[30:0] (byte aligned)**

Note :
When hash engine works in cascaded mode (Crypto first, hash second), HACE20 MUST equal to HACE04.
When hash engine works in cascaded mode (Hash first, crypto second), HACE20 MUST equal to HACE00.

Offset: 24h **HACE24: Hash Digest Write Buffer Base Address Register** **Init = X**

Bit	R/W	Description
31	RO	**Reserved (0)**
30:3	RW	**Base address of hash digest write buffer[30:3] (8-byte aligned)**
2 :0	RO	**Reserved (0)**

Algorithm	Digest	Digest write buffer
MD5	16 bytes	16 bytes
SHA-1	20 bytes	20 bytes
SHA-224	28 bytes	32 bytes
SHA-256	32 bytes	32 bytes
SHA-384	48 bytes	64 bytes
SHA-512	64 bytes	64 bytes
SHA-512/224	28 bytes	64 bytes
SHA-512/256	32 bytes	64 bytes

Offset: 28h **HACE28: Hash HMAC Key Buffer Base Address Register** **Init = X**

Bit	R/W	Description
31	RO	**Reserved (0)**
30:3	RW	**Base address of HMAC key buffer[30:2] (8-byte aligned)**
2 :0	RO	**Reserved (0)**

Note :
HMAC Key Buffer store the result of calculate HMAC key command (HACE30 [8:7] = 0x3).
See ”Hash Function Programming Sequence” for detail information.

Offset: 2Ch **HACE2C: Hash Data Length Register** **Init = X**

Bit	R/W	Description
31:28	RO	**Reserved (0)**

to next page

from previous page

27:0	RW	**Hash data length** 0: 0 byte 1: 1 byte 2: 2 bytes ... The register determines the data length to be hashed. The maximum data length is up to **(256MB-1) bytes** for a hash command.

HACE30[8:7]	HACE30[6:4]	HACE30[14:13]	Alignment	Minimum
0,1	-	-	1-byte aligned	0h
2	SHA-512 series	0,1	128-byte aligned	80h
2	others	0,1	64-byte aligned	40h
2	-	2,3	1-byte aligned	0h
3	SHA-512 series	-	80h(fixed value)	80h
3	others	-	40h(fixed value)	40h

Note :
When hash engine works in cascaded mode, HACE2C MUST equal to HACE0C.

Offset: 30h **HACE30: Hash Engine Command Register** **Init = 0**

Bit	R/W	Description
31:21	RO	**Reserved (0)**
20	RW	**M-Bus request synchronization for Hash Engine Idle** 0: Disable M-Bus request synchronization 1: Enable M-Bus request synchronization
19	RO	**Reserved (0)**
18	RW	**Hash source data scatter-gather control** 0: Direct access mode 1: Scatter-gather mode When the HACE30 [8:7] = 3, the setting of HACE30 [18] must be in direct access mode.
17:15	RO	**Reserved (0)**
14	RW	**Last block for accumulative mode** 0: This command DON'T the last block for the accumulative mode 1: This command include the last block for the accumulative mode Before setting this bit to value 1, the value of HACE34 should be ready.
13	RW	**First block for the accumulative mode** 0: This command DON'T the first block for the accumulative mode 1: This command include the first block for the accumulative mode
12:10	RW	**SHA-512 series algorithm selection** 000: Select SHA-512 algorithm 001: Select SHA-384 algorithm 010: Select SHA-512/256 algorithm 011: Select SHA-512/224 algorithm 100: **Invalid** 101: **Invalid** 110: **Invalid** 111: **Invalid**

to next page

from previous page

9	RW	**Enable hash interrupt** 0: Disable hash interrupt 1: Enable hash interrupt when hash command finished
8 :7	RW	**HMAC engine command mode** 00: Calculate digest without HMAC 01: Calculate digest with HMAC 10: Calculate digest with Accumulative Mode 11: Calculate HMAC key (Hash engine must be programmed to be working at the independent mode) When the HACE30 [8:7] = 2 or 3, the setting of HACE30 [1:0] must be in independent mode.
6 :4	RW	**Hash algorithm selection** 000: Select MD5 algorithm 001: **Invalid** 010: Select SHA-1 algorithm 011: **Invalid** 100: Select SHA-224 algorithm 101: Select SHA-256 algorithm 110: Select SHA-512 series algorithm 111: **Invalid** When HACE30 [6:4] = 6, the HACE30 [12:10] must be setting to select the SHA-512 series algorithm.
3 :2	RW	**Byte swapping control** 00: **Invalid** 01: Byte swapping control for all MD5 hash commands (little-endian) 10: Byte swapping control for all SHA-1,SHA-224,SHA-256,SHA-384,SHA-512,SHA-512/224,SHA-512/256 hash commands (big-endian) 11: **Invalid**
1 :0	RW	**Hash engine operation mode control** 00: Hash engine works in independent mode 01: Hash engine works in independent mode 10: Hash engine works in cascaded mode (Crypto first, hash second) 11: Hash engine works in cascaded mode (Hash first, crypto second) In cascaded mode, the programming of HACE10 [1:0] MUST be consistent with the programming of HACE30 [1:0]. Otherwise, HACE may be trapped in a dead lock.

Offset: 34h **HACE34: Hash Data Padding Length Register** **Init = X**

Bit	R/W	Description
31:28	RO	**Reserved (0)**
27:0	RW	**Hash data padding length**

Note :
When setting HACE30 [8:7] = 2, and HACE30 [14] = 1, this register would be used in the padding block.

Offset: 50h **HACE50: Command Queue Base Address** **Init = X**

Bit	R/W	Description
31	RO	**Reserved (0)**

to next page

from previous page

30: 4	RW	**Command queue base address[30:4] (16-byte aligned)**
3: 0	RO	**Reserved (0)**
Note : Writing initial value to this register before turn on the DRAM Command Queue Buffer.		

Offset: 54h	**HACE54: Command Queue End Pointer**	**Init = X**
Bit	**R/W**	**Description**
31:12	RO	**Reserved (0)**
11: 1	RW	**Command queue end pointer[11:1] (16-byte aligned)**
0	RO	**Reserved (0)**
Note : Writing initial value to this register before turn on the DRAM Command Queue Buffer.		

Offset: 58h	**HACE58: Command Queue Write Pointer**	**Init = X**
Bit	**R/W**	**Description**
31:12	RO	**Reserved (0)**
11: 0	RW	**Command queue write pointer[11:0] (8-byte aligned)**
Note : Writing initial value to this register before turn on the DRAM Command Queue Buffer.		

Offset: 5Ch	**HACE5C: Command Queue Read Pointer**	**Init = 0**
Bit	**R/W**	**Description**
31:12	RO	**Reserved (0)**
11: 0	RO	**Command queue read pointer[11:0] (8-byte aligned)**

Offset: 60h	**HACE60: HAC Engine Feature Register**	**Init = 0**
Bit	**R/W**	**Description**
31	RW	**Enable DRAM Command Queue Buffer** 0: Disable DRAM Command Queue Buffer 1: Enable DRAM Command Queue Buffer
30	RW	**Enable Write Register Data from Command Queue Data** 0: Write HACE00 ~ HACE18, HACE20 ~ HACE34, HACE64 register data from AHB bus. 1: Write HACE00 ~ HACE18, HACE20 ~ HACE34, HACE64 register data from command queue data.
29	RW	~~**Enable AES/DES CTR Mode with 64-bit Counter**~~ ~~0: AES/DES CTR mode with 32-bit counter.~~ ~~1: AES/DES CTR mode with 64-bit counter.~~
28	RW	**Command Queue Data Format** 0: Single Command queue data. 1: Burst Command queue data.
27: 0	RO	**Reserved (0)**

Offset: 64h	HACE64: HAC Software Tag Register	Init = X

Bit	R/W	Description
31	RW	**Enable write software tag interrupt** 0: Disable write software tag interrupt 1: Enable write software tag interrupt when write value to HAC engine tag register HACE64
30	RO	**Reserved (0)**
29	RW	**Wait Crypto Engine IDLE Control** 0: Update the software tag value without checking whether the crypto engine is IDLE 1: Update the software tag value with checking the crypto engine is IDLE
28	RW	**Wait Hash Engine IDLE Control** 0: Update the software tag value without checking whether the hash engine is IDLE 1: Update the software tag value with checking the hash engine is IDLE
27: 0	RW	**HAC software tag register[27:0]**

25.4 Crypto Context Buffer Format

25.4.1 RC4 (272 Bytes)

Byte Range	Description
000 - 007	Reserved (0)
008	Index I (With initial value 1)
009	Index J (With initial value 0)
00A - 00F	Reserved (0)
010 - 10F	RC4 Key Byte 0 ~ 255

25.4.2 [Hardware key expansion] AES-128 (32/48 Bytes)

Byte Range	Description
000 - 00F	Initial Vector Byte 0 ~ 15 (Initial Vector is NOT required in **ECB** mode)
010 - 01F	AES Key Byte 0 ~ 15
020 - 02F	GHASH subKey Byte 0 ~ 15 (**GCM** mode)

25.4.3 [Software key expansion] AES-128 (192/208 Bytes)

Byte Range	Description
000 - 00F	Initial Vector Byte 0 ~ 15 (Initial Vector is NOT required in **ECB** mode)
010 - 0BF	AES SW Expanded Key Byte 0 ~ 175
0C0 - 0CF	GHASH subKey Byte 0 ~ 15 (**GCM** mode)

25.4.4 [Hardware key expansion] AES-192 (40/64 Bytes)

Byte Range	Description
000 - 00F	Initial Vector Byte 0 ∼ 15 (Initial Vector is NOT required in **ECB** mode)
010 - 027	AES Key Byte 0 ∼ 23
028 - 02F	Reserved (0)
030 - 03F	GHASH subKey Byte 0 ∼ 15 (**GCM** mode)

25.4.5 [Software key expansion] AES-192 (224/240 Bytes)

Byte Range	Description
000 - 00F	Initial Vector Byte 0 ∼ 15 (Initial Vector is NOT required in **ECB** mode)
010 - 0DF	AES SW Expanded Key Byte 0 ∼ 207
0E0 - 0EF	GHASH subKey Byte 0 ∼ 15 (**GCM** mode)

25.4.6 [Hardware key expansion] AES-256 (48/64 Bytes)

Byte Range	Description
000 - 00F	Initial Vector Byte 0 ∼ 15 (Initial Vector is NOT required in **ECB** mode)
010 - 02F	AES Key Byte 0 ∼ 31
030 - 03F	GHASH subKey Byte 0 ∼ 15 (**GCM** mode)

25.4.7 [Software key expansion] AES-256 (256/272 Bytes)

Byte Range	Description
000 - 00F	Initial Vector Byte 0 ∼ 15 (Initial Vector is NOT required in **ECB** mode)
010 - 0FF	AES SW Expanded Key Byte 0 ∼ 239
100 - 10F	GHASH subKey Byte 0 ∼ 15 (**GCM** mode)

25.4.8 DES (40 Bytes)

Byte Range	Description
000 - 007	Reserved (0)
008 - 00F	Initial Vector Byte 0 ∼ 7 (Initial Vector is NOT required in **ECB** mode)
010 - 017	DES Key 1 Byte 0 ∼ 7
018 - 01F	DES Key 2 Byte 0 ∼ 7
020 - 027	DES Key 3 Byte 0 ∼ 7

25.5 Scatter-gather List Format

- Scatter-gather list size: 8 Bytes

Offset: 0h	HACSG#0: Scatter-gather List Data[31:0]	Init = X

Bit	R/W	Description
31	RW	**Last scatter-gather list** When set, it indicates that this is the last scatter-gather list of the data
30:28	RO	**Reserved (0)**
27:0	RW	**scatter-gather data length (bytes)**

Offset: 4h	HACSG#1: Scatter-gather List Data[64:32]	Init = X

Bit	R/W	Description
31	RO	**Reserved (0)**
30:0	RW	**Base address of scatter-gather data[30:0] (byte aligned)**

25.6 Hash Function Programming Sequence

25.6.1 Parameter Definition

- $Hash\_Input\_Data\_Base\_Adr$ (byte aligned): Base address of data buffer which want to calculate hash digest.

- $Hash\_Digest\_Base\_Adr$ (8-byte aligned): Base address of hash digest write buffer.
  - MD5 : Digest is 16 bytes, digest write buffer is 16 bytes
  - SHA-1 : Digest is 20 bytes, digest write buffer is 20 bytes
  - SHA-224 : Digest is 28 bytes, digest write buffer is 32 bytes
  - SHA-256 : Digest is 32 bytes, digest write buffer is 32 bytes
  - SHA-384 : Digest is 48 bytes, digest write buffer is 64 bytes
  - SHA-512 : Digest is 64 bytes, digest write buffer is 64 bytes
  - SHA-512/224 : Digest is 28 bytes, digest write buffer is 64 bytes
  - SHA-512/256 : Digest is 32 bytes, digest write buffer is 64 bytes
- $Hash\_Acc\_Digest\_Write\_Buffer$ (16-byte aligned): Accumulative hash digest write buffer.
  - MD5 : Accumulative digest is 16 bytes, accumulative digest write buffer is 16 bytes
  - SHA-1 : Accumulative digest is 20 bytes, accumulative digest write buffer is 20 bytes
  - SHA-224 : Accumulative digest is 28 bytes, accumulative digest write buffer is 32 bytes
  - SHA-256 : Accumulative digest is 32 bytes, accumulative digest write buffer is 32 bytes
  - SHA-384 : Accumulative digest is 48 bytes, accumulative digest write buffer is 64 bytes
  - SHA-512 : Accumulative digest is 64 bytes, accumulative digest write buffer is 64 bytes
  - SHA-512/224 : Accumulative digest is 28 bytes, accumulative digest write buffer is 64 bytes
  - SHA-512/256 : Accumulative digest is 32 bytes, accumulative digest write buffer is 64 bytes
- $Hash\_Acc\_Digest\_Base\_Adr$ (16-byte aligned): Base address of accumulative hash digest write buffer ($Hash\_Acc\_Digest\_Write\_Buffer$).
- $Hash\_Input\_Size$ (byte aligned): Byte size of data buffer which want to calculate hash digest.
- $Hash\_Acc\_Input\_Size$ (64-byte aligned)(128-byte aligned for SHA-512 series): Byte size of accumulative data buffer which want to calculate accumulative hash digest.

- $K_0\_Buffer\_Base\_Adr$ (8-byte aligned): Base address of 64 byte **(128 byte for SHA-512 series)** K_0 buffer.
- $HMAC\_Key\_Buffer\_Base\_Adr$ (16-byte aligned): Base address of 64 byte **(128 byte for SHA-512 series)** buffer which is used to store the result of calculate HMAC key command (HACE30 [8:7] = 3).

25.6.2 MD5,SHA-1,SHA-224,SHA-256,SHA-384,SHA-512,SHA-512/224,SHA-512/256

- $Hash\_Input\_Data\_Base\_Adr$ (byte aligned)
- $Hash\_Digest\_Base\_Adr$ (8-byte aligned)
- $Hash\_Input\_Size$ (byte aligned)

1. Calculating Hash Digest:
   (a) HACE20 = $Hash\_Input\_Data\_Base\_Adr$ (byte aligned)
   (b) HACE24 = $Hash\_Digest\_Base\_Adr$ (8-byte aligned)
   (c) HACE2C = $Hash\_Input\_Size$ (byte aligned)
   (d) HACE30:
      - MD5 : 04h or 204h
      - SHA-1 : 28h or 228h

- SHA-224 : 48h or 248h
- SHA-256 : 58h or 258h
- SHA-384 : 468h or 668h
- SHA-512 : 68h or 268h
- SHA-512/224 : C68h or E68h
- SHA-512/256 : 868h or A68h

25.6.3 HMAC MD5,SHA-1,SHA-224,SHA-256,SHA-384,SHA-512,SHA-512/224,SHA-512/256

- $K_0\_Buffer\_Base\_Adr$ (8-byte aligned)
- $HMAC\_Key\_Buffer\_Base\_Adr$ **(16-byte aligned)**
- $Hash\_Input\_Data\_Base\_Adr$ (byte aligned)
- $Hash\_Digest\_Base\_Adr$ (8-byte aligned)
- $Hash\_Input\_Size$ (byte aligned)

Programming sequence "Preparing K_0 Buffer" & "Calculating HMAC Key Buffer" are needed ONLY when secret key are changed.

1. Preparing K_0 Buffer:
   Software need to prepare 64 byte **(128 byte for SHA-512 series)** K_0 buffer from original secret key. This sequence equal to step 1 $\sim$ 3 in Table 1 of "FIPS PUB 198: The Keyed-Hash Message Authentication Code (HMAC)", and the APPENDIX A has some examples.

2. Calculating HMAC Key Buffer:
   (a) HACE20 = $K_0\_Buffer\_Base\_Adr$ (8-byte aligned)
   (b) HACE28 = $HMAC\_Key\_Buffer\_Base\_Adr$ **(16-byte aligned)**
   (c) HACE2C = 40h or **80h (for SHA-512 series)**
   (d) HACE30:
   - HMAC-MD5 : 184h or 384h
   - HMAC-SHA-1 : 1A8h or 3A8h
   - HMAC-SHA-224 : 1C8h or 3C8h
   - HMAC-SHA-256 : 1D8h or 3D8h
   - HMAC-SHA-384 : 5E8h or 7E8h
   - HMAC-SHA-512 : 1E8h or 3E8h
   - HMAC-SHA-512/224 : DE8h or FE8h
   - HMAC-SHA-512/256 : 9E8h or BE8h
3. Calculating HMAC Hash Digest:
   (a) HACE20 = $Hash\_Input\_Data\_Base\_Adr$ (byte aligned)
   (b) HACE24 = $Hash\_Digest\_Base\_Adr$ (8-byte aligned)
   (c) HACE28 = $HMAC\_Key\_Buffer\_Base\_Adr$ **(16-byte aligned)**
   (d) HACE2C = $Hash\_Input\_Size$ (byte aligned)
   (e) HACE30:
   - HMAC-MD5 : 84h or 284h
   - HMAC-SHA-1 : A8h or 2A8h
   - HMAC-SHA-224 : C8h or 2C8h
   - HMAC-SHA-256 : D8h or 2D8h
   - HMAC-SHA-384 : 4E8h or 6E8h
   - HMAC-SHA-512 : E8h or 2E8h
   - HMAC-SHA-512/224 : CE8h or EE8h
   - HMAC-SHA-512/256 : 8E8h or AE8h

25.6.4 Accumulative Mode

- $HMAC\_Key\_Buffer\_Base\_Adr$ **(16-byte aligned)**
- $Hash\_Input\_Data\_Base\_Adr$ (byte aligned)
- $Hash\_Acc\_Digest\_Base\_Adr$ **(16-byte aligned)**
- $Hash\_Acc\_Input\_Size$ **(64-byte aligned)(128-byte aligned for SHA-512 series)**

1. Allocating & Initiating Accumulative Hash Digest Write Buffer ($Hash\_Acc\_Digest\_Write\_Buffer$): This sequence is neeed ONLY before processing first accumulative data.

– MD5,SHA-1,SHA-224,SHA-256 ($Hash\_Acc\_Digest\_Write\_Buffer$):
This table is little-endian format.

Byte Range	MD5	SHA-1	SHA-224	SHA-256
00 - 03	67452301h	01234567h	D89E05C1h	67E6096Ah
04 - 07	EFCDAB89h	89ABCDEFh	07D57C36h	85AE67BBh
08 - 0B	98BADCFEh	FEDCBA98h	17DD7030h	72F36E3Ch
0C - 0F	10325476h	76543210h	39590EF7h	3AF54FA5h
10 - 13	Reserved	F0E1D2C3h	310BC0FFh	7F520E51h
14 - 17	Reserved	Reserved	11155868h	8C68059Bh
18 - 1B	Reserved	Reserved	A78FF964h	ABD9831Fh
1C - 1F	Reserved	Reserved	A44FFABEh	19CDE05Bh

– SHA-384,SHA-512,SHA-512/224,SHA-512/256 ($Hash\_Acc\_Digest\_Write\_Buffer$):
This table is little-endian format.

Byte Range	SHA-384	SHA-512	SHA-512/224	SHA-512/256
00 - 03	5D9DBBCBh	67E6096Ah	C8373D8Ch	94213122h
04 - 07	D89E05C1h	08C9BCF3h	A24D5419h	2CF72BFCh
08 - 0B	2A299A62h	85AE67BBh	6699E173h	A35F559Fh
0C - 0F	07D57C36h	3BA7CA84h	D6D4DC89h	C2644CC8h
10 - 13	5A015991h	72F36E3Ch	AEB7FA1Dh	6BB89323h
14 - 17	17DD7030h	2BF894FEh	829CFF32h	51B1536Fh
18 - 1B	D8EC2F15h	3AF54FA5h	14D59D67h	19773896h
1C - 1F	39590EF7h	F1361D5Fh	CF9F2F58h	BDEA4059h
20 - 23	67263367h	7F520E51h	692B6D0Fh	E23E2896h
24 - 27	310BC0FFh	D182E6ADh	A84DD47Bh	E3FF8EA8h
28 - 2B	874AB48Eh	8C68059Bh	736FE377h	251E5EBEh
2C - 2F	11155868h	1F6C3E2Bh	4289C404h	92398653h
30 - 33	0D2E0CDBh	ABD9831Fh	A8859D3Fh	FC99012Bh
34 - 37	A78FF964h	6BBD41FBh	C8361D6Ah	AAB8852Ch
38 - 3B	1D48B547h	19CDE05Bh	ADE61211h	DC2DB70Eh
3C - 3F	A44FFABEh	79217E13h	A192D691h	A22CC581h

– Phase 1 of HMAC MD5,SHA-1,SHA-224,SHA-256,SHA-384,SHA-512,SHA-512/224,SHA-512/256 ($Hash\_Acc\_Digest\_Write\_Buffer$):
Phase 1 equal to step 5 $\sim$ 6 in Table 1 of ”FIPS PUB 198: The Keyed-Hash Message Authentication Code (HMAC)”.

HMAC_Key_Buffer store the outcome of programming sequence ”Calculating HMAC Key Buffer”. The HMAC_Key_Buffer must be ready before running this sequence.

Byte Range	HMAC MD5,SHA-1,SHA-224,SHA-256
00 - 03	HMAC_Key_Buffer[00:03] (MD5, SHA-1, SHA-224, SHA-256)
04 - 07	HMAC_Key_Buffer[04:07] (MD5, SHA-1, SHA-224, SHA-256)
08 - 0B	HMAC_Key_Buffer[08:0B] (MD5, SHA-1, SHA-224, SHA-256)
0C - 0F	HMAC_Key_Buffer[0C:0F] (MD5, SHA-1, SHA-224, SHA-256)
10 - 13	HMAC_Key_Buffer[10:13] (SHA-1, SHA-224, SHA-256)
14 - 17	HMAC_Key_Buffer[14:17] (SHA-224, SHA-256)
18 - 1B	HMAC_Key_Buffer[18:1B] (SHA-224, SHA-256)
1C - 1F	HMAC_Key_Buffer[1C:1F] (SHA-224, SHA-256)

Byte Range	HMAC SHA-384,SHA-512,SHA-512/224,SHA-512/256
00 - 03	HMAC_Key_Buffer[00:03] (SHA-384, SHA-512, SHA-512/224, SHA-512/256)
04 - 07	HMAC_Key_Buffer[04:07] (SHA-384, SHA-512, SHA-512/224, SHA-512/256)
08 - 0B	HMAC_Key_Buffer[08:0B] (SHA-384, SHA-512, SHA-512/224, SHA-512/256)
0C - 0F	HMAC_Key_Buffer[0C:0F] (SHA-384, SHA-512, SHA-512/224, SHA-512/256)
10 - 13	HMAC_Key_Buffer[10:13] (SHA-384, SHA-512, SHA-512/224, SHA-512/256)
14 - 17	HMAC_Key_Buffer[14:17] (SHA-384, SHA-512, SHA-512/224, SHA-512/256)
18 - 1B	HMAC_Key_Buffer[18:1B] (SHA-384, SHA-512, SHA-512/224, SHA-512/256)
1C - 1F	HMAC_Key_Buffer[1C:1F] (SHA-384, SHA-512, SHA-512/224, SHA-512/256)
20 - 23	HMAC_Key_Buffer[20:23] (SHA-384, SHA-512, SHA-512/224, SHA-512/256)
24 - 27	HMAC_Key_Buffer[24:27] (SHA-384, SHA-512, SHA-512/224, SHA-512/256)
28 - 2B	HMAC_Key_Buffer[28:2B] (SHA-384, SHA-512, SHA-512/224, SHA-512/256)
2C - 2F	HMAC_Key_Buffer[2C:2F] (SHA-384, SHA-512, SHA-512/224, SHA-512/256)
30 - 33	HMAC_Key_Buffer[30:33] (SHA-384, SHA-512, SHA-512/224, SHA-512/256)
34 - 37	HMAC_Key_Buffer[34:37] (SHA-384, SHA-512, SHA-512/224, SHA-512/256)
38 - 3B	HMAC_Key_Buffer[38:3B] (SHA-384, SHA-512, SHA-512/224, SHA-512/256)
3C - 3F	HMAC_Key_Buffer[3C:3F] (SHA-384, SHA-512, SHA-512/224, SHA-512/256)

– Phase 2 of HMAC MD5,SHA-1,SHA-224,SHA-256,SHA-384,SHA-512,SHA-512/224,SHA-512/256 ($Hash\_Acc\_Digest\_Write\_Buffer$):
Phase 2 equal to step 8 $\sim$ 9 in Table 1 of ”FIPS PUB 198: The Keyed-Hash Message Authentication Code (HMAC)”

HMAC_Key_Buffer store the outcome of programming sequence ”Calculating HMAC Key Buffer”. The HMAC_Key_Buffer must be ready before running this sequence.

Byte Range	HMAC MD5,SHA-1,SHA-224,SHA-256
00 - 03	HMAC_Key_Buffer[20:23] (MD5, SHA-1, SHA-224, SHA-256)
04 - 07	HMAC_Key_Buffer[24:27] (MD5, SHA-1, SHA-224, SHA-256)
08 - 0B	HMAC_Key_Buffer[28:2B] (MD5, SHA-1, SHA-224, SHA-256)
0C - 0F	HMAC_Key_Buffer[2C:2F] (MD5, SHA-1, SHA-224, SHA-256)
10 - 13	HMAC_Key_Buffer[30:33] (SHA-1, SHA-224, SHA-256)
14 - 17	HMAC_Key_Buffer[34:37] (SHA-224, SHA-256)
18 - 1B	HMAC_Key_Buffer[38:3B] (SHA-224, SHA-256)
1C - 1F	HMAC_Key_Buffer[3C:3F] (SHA-224, SHA-256)

Byte Range	HMAC SHA-384,SHA-512,SHA-512/224,SHA-512/256
00 - 03	HMAC_Key_Buffer[40:43] (SHA-384, SHA-512, SHA-512/224, SHA-512/256)
04 - 07	HMAC_Key_Buffer[44:47] (SHA-384, SHA-512, SHA-512/224, SHA-512/256)
08 - 0B	HMAC_Key_Buffer[48:4B] (SHA-384, SHA-512, SHA-512/224, SHA-512/256)
0C - 0F	HMAC_Key_Buffer[4C:4F] (SHA-384, SHA-512, SHA-512/224, SHA-512/256)
10 - 13	HMAC_Key_Buffer[50:53] (SHA-384, SHA-512, SHA-512/224, SHA-512/256)
14 - 17	HMAC_Key_Buffer[54:57] (SHA-384, SHA-512, SHA-512/224, SHA-512/256)
18 - 1B	HMAC_Key_Buffer[58:5B] (SHA-384, SHA-512, SHA-512/224, SHA-512/256)
1C - 1F	HMAC_Key_Buffer[5C:5F] (SHA-384, SHA-512, SHA-512/224, SHA-512/256)
20 - 23	HMAC_Key_Buffer[60:63] (SHA-384, SHA-512, SHA-512/224, SHA-512/256)
24 - 27	HMAC_Key_Buffer[64:67] (SHA-384, SHA-512, SHA-512/224, SHA-512/256)
28 - 2B	HMAC_Key_Buffer[68:6B] (SHA-384, SHA-512, SHA-512/224, SHA-512/256)
2C - 2F	HMAC_Key_Buffer[6C:6F] (SHA-384, SHA-512, SHA-512/224, SHA-512/256)
30 - 33	HMAC_Key_Buffer[70:73] (SHA-384, SHA-512, SHA-512/224, SHA-512/256)
34 - 37	HMAC_Key_Buffer[74:77] (SHA-384, SHA-512, SHA-512/224, SHA-512/256)
38 - 3B	HMAC_Key_Buffer[78:7B] (SHA-384, SHA-512, SHA-512/224, SHA-512/256)
3C - 3F	HMAC_Key_Buffer[7C:7F] (SHA-384, SHA-512, SHA-512/224, SHA-512/256)

2. Calculating Accumulative Hash Digest:
   Running this sequence repeatedly until receiving the last accumulative data.

   (a) When receiving the last accumulative data, software need to add Padding Message at the end of the accumulative data. Padding Message is described in the specific of MD5,SHA-1,SHA-224,SHA-256,SHA-384,SHA-512,SHA-512/224,SHA-512/256.

   Let N be the totally byte size of accumulative data, the 64 bit length-column of Padding Message is:

Hash Algorithm	Non-HMAC	HMAC Phase 1	HMAC Phase 2
MD5	N * 8	(64 + N) * 8	(64 + 16) * 8
SHA-1	N * 8	(64 + N) * 8	(64 + 20) * 8
SHA-224	N * 8	(64 + N) * 8	(64 + 28) * 8
SHA-256	N * 8	(64 + N) * 8	(64 + 32) * 8
SHA-384	N * 8	(128 + N) * 8	(128 + 48) * 8
SHA-512	N * 8	(128 + N) * 8	(128 + 64) * 8
SHA-512/224	N * 8	(128 + N) * 8	(128 + 28) * 8
SHA-512/256	N * 8	(128 + N) * 8	(128 + 32) * 8

   (b) HACE20 = $Hash\_Input\_Data\_Base\_Adr$ (byte aligned)
   (c) HACE24 = $Hash\_Acc\_Digest\_Base\_Adr$ **(16-byte aligned)**
   (d) HACE28 = $Hash\_Acc\_Digest\_Base\_Adr$ **(16-byte aligned)**
   (e) HACE2C = $Hash\_Acc\_Input\_Size$ **(64-byte aligned)(128-byte aligned for SHA-512 series)**
   (f) HACE30:
   - MD5 or HMAC-MD5 : 104h or 304h
   - SHA-1 or HMAC-SHA-1 : 128h or 328h
   - SHA-224 or HMAC-SHA-224 : 148h or 348h
   - SHA-256 or HMAC-SHA-256 : 158h or 358h
   - SHA-384 or HMAC-SHA-384 : 568h or 768h
   - SHA-512 or HMAC-SHA-512 : 168h or 368h
   - SHA-512/224 or HMAC-SHA-512/224 : D68h or F68h
   - SHA-512/256 or HMAC-SHA-512/256 : 968h or B68h

25.7 Command Queue Data Format

- Single Command queue data size: 8 Bytes
- Burst Command queue data size: 8, 16, 24, 32 Bytes
- Maximum capacity of Command queue buffer: 32 KBytes

Offset: 0h	**HACSinCQ#0: Single Command Queue Data[31:0]**	**Init = X**
Bit	**R/W**	**Description**
31:0	RO	**Register wirte data[31:0]**

Offset: 4h	**HACSinCQ#1: Single Command Queue Data[64:32]**	**Init = X**
Bit	**R/W**	**Description**
31:8	RW	**Single Command Queue Data ID[23:0] (0x665599)**
7 :5	RO	**Reserved (0)**
4 :0	RW	**Register wirte adddress[4:0]**

Offset: 0h	**HACBurCQ#0: Burst Command Queue Data[31:0]**	**Init = X**
Bit	**R/W**	**Description**
31:12	RW	**Burst Command Queue Data ID[19:0] (0x65965)**
11	RO	**Reserved (0)**
10:8	RW	**Burst length[2:0]** 0: Register wirte data #0 is valid 1: Register wirte data #0 is valid 2: Register wirte data #0 ~ #1 are valid, and register wirte data #2 is padding data 3: Register wirte data #0 ~ #2 are valid 4: Register wirte data #0 ~ #3 are valid, and register wirte data #4 is padding data 5: Register wirte data #0 ~ #4 are valid 6: Register wirte data #0 ~ #5 are valid, and register wirte data #6 is padding data 7: Register wirte data #0 ~ #6 are valid
7 :5	RO	**Reserved (0)**
4 :0	RW	**Register wirte start adddress[4:0]**

Offset: 04h	**HACBurCQ#1: Burst Command Queue Data[63:32]**	**Init = X**
Offset: 08h	**HACBurCQ#2: Burst Command Queue Data[95:64]**	**Init = X**
Offset: 0Ch	**HACBurCQ#3: Burst Command Queue Data[127:96]**	**Init = X**
Offset: 10h	**HACBurCQ#4: Burst Command Queue Data[159:128]**	**Init = X**
Offset: 14h	**HACBurCQ#5: Burst Command Queue Data[191:160]**	**Init = X**
Offset: 18h	**HACBurCQ#6: Burst Command Queue Data[223:192]**	**Init = X**
Offset: 1Ch	**HACBurCQ#7: Burst Command Queue Data[255:224]**	**Init = X**
Bit	**Attr.**	**Description**
31:0	RO	**Register wirte data #0 ~ #6[31:0]**

26 ECC/RSA Engine (ACRY)

26.1 Overview

ECC Engine is designed to accelerate ECDSA Digital Signature Generation and Verification. The supported NIST Elliptic Curves over Prime Fields are P-192, P-224, P-256 and P-384.

RSA Engine supports 256 bits to 4096 bits RSA signature and verification.

Base address of ECCRSA = 0x1E6F_A000
Physical address = (Base address of ECCRSA) + Offset

26.2 Registers : Base Address = 0x1E6F:A000

Offset: 00h	ECCRSA_CTRL00: ECC and RSA Engine Control	Init = 0x00000000
Bit	**R/W**	**Description**
31:7	RW	**Reserved**
6	RW	**ECC_DMA_DATA : ECC Data DMA**
5	RW	**ECC_DMA_PROG : ECC Program DMA**
4	RW	**ECC_START : Trigger ECC Engine**
3:2	RW	**Reserved**
1	RW	**RSA_DMA_DATA : RSA Data DMA**
0	RW	**RSA_START : Trigger RSA Engine**

Offset: 04h	ECCRSA_CTRL01: ECC Program Index	Init = 0x00000000
Bit	**R/W**	**Description**
31:16	RW	**ECC_PIDX_01 : ECC Program #1 Index**
15:0	RW	**ECC_PIDX_00 : ECC Program #0 Index**

Offset: 08h	ECCRSA_CTRL02: ECC Program Index	Init = 0x00000000
Bit	**R/W**	**Description**
31:16	RW	**ECC_PIDX_03 : ECC Program #3 Index**
15:0	RW	**ECC_PIDX_02 : ECC Program #2 Index**

Offset: 0Ch	ECCRSA_CTRL03: ECC Program Index	Init = 0x00000000
Bit	**R/W**	**Description**
31:16	RW	**ECC_PIDX_05 : ECC Program #5 Index**
15:0	RW	**ECC_PIDX_04 : ECC Program #4 Index**

Offset: 10h	ECCRSA_CTRL04: ECC Program Index	Init = 0x00000000
Bit	**R/W**	**Description**
31:16	RW	**ECC_PIDX_07 : ECC Program #7 Index**
15:0	RW	**ECC_PIDX_06 : ECC Program #6 Index**

Offset: 14h	ECCRSA_CTRL05: ECC Program Index	Init = 0x00000000
Bit	**R/W**	**Description**
31:16	RW	**ECC_PIDX_09 : ECC Program #9 Index**
15:0	RW	**ECC_PIDX_08 : ECC Program #8 Index**

Offset: 18h	ECCRSA_CTRL06: ECC Program Index	Init = 0x00000000
Bit	**R/W**	**Description**
31:16	RW	**ECC_PIDX_11 : ECC Program #11 Index**
15:0	RW	**ECC_PIDX_10 : ECC Program #10 Index**

Offset: 1Ch	ECCRSA_CTRL07: ECC Program Index	Init = 0x00000000
Bit	**R/W**	**Description**
31:16	RW	**ECC_PIDX_13 : ECC Program #13 Index**
15:0	RW	**ECC_PIDX_12 : ECC Program #12 Index**

Offset: 20h	ECCRSA_CTRL08: ECC Program Index	Init = 0x00000000
Bit	**R/W**	**Description**
31:16	RW	**ECC_PIDX_15 : ECC Program #15 Index**
15:0	RW	**ECC_PIDX_14 : ECC Program #14 Index**

Offset: 24h	ECCRSA_CTRL09: ECC Parameters	Init = 0x00000000
Bit	**R/W**	**Description**
31:16	RW	**ECC_WREG_01 : ECC Parameters #1**
15:0	RW	**ECC_WREG_00 : ECC Parameters #0**

Offset: 28h	ECCRSA_CTRL10: ECC Parameters	Init = 0x00000000
Bit	**R/W**	**Description**
31:16	RW	**ECC_WREG_03 : ECC Parameters #3**
15:0	RW	**ECC_WREG_02 : ECC Parameters #2**

Offset: 2Ch	ECCRSA_CTRL11: ECC Parameters	Init = 0x00000000
Bit	**R/W**	**Description**
31:16	RW	**ECC_WREG_05 : ECC Parameters #5**
15:0	RW	**ECC_WREG_04 : ECC Parameters #4**

Offset: 30h	ECCRSA_CTRL12: ECC Parameters	Init = 0x00000000
Bit	**R/W**	**Description**
31:16	RW	**ECC_WREG_07 : ECC Parameters #7**
15:0	RW	**ECC_WREG_06 : ECC Parameters #6**

Offset: 34h	ECCRSA_CTRL13: ECC Parameters	Init = 0x00000000
Bit	**R/W**	**Description**
31:16	RW	**ECC_WREG_09 : ECC Parameters #9**
15:0	RW	**ECC_WREG_08 : ECC Parameters #8**

Offset: 38h	ECCRSA_CTRL14: ECC Parameters	Init = 0x00000000
Bit	**R/W**	**Description**
31:16	RW	**ECC_WREG_11 : ECC Parameters #11**
15:0	RW	**ECC_WREG_10 : ECC Parameters #10**

Offset: 3Ch	ECCRSA_CTRL15: ECC Parameters	Init = 0x00000000
Bit	**R/W**	**Description**
31:16	RW	**ECC_WREG_13 : ECC Parameters #13**
15:0	RW	**ECC_WREG_12 : ECC Parameters #12**

Offset: 40h	ECCRSA_CTRL16: ECC Parameters	Init = 0x00000000
Bit	**R/W**	**Description**
31:16	RW	**ECC_WREG_15 : ECC Parameters #15**
15:0	RW	**ECC_WREG_14 : ECC Parameters #14**

Offset: 44h	ECCRSA_CTRL17: ECC and RSA Engine Control	Init = 0x00000000
Bit	**R/W**	**Description**
31:17	RW	**Reserved**
16	RW	**ECC_MUL_LONG : Long Multiplier**
15:13	RW	**Reserved**
12	RW	**ECC_FLAG_F : FLAG field of F**
11:9	RW	**Reserved**
8	RW	**ECC_MUL_P : Parallel Multiplier**
7:6	RW	**Reserved**
5:4	RW	**ECC_BITS : Prime Field Selection** 00: NIST P-192 (default) 01: NIST P-224 10: NIST P-256 11: NIST P-384
3:0	RW	**ECC_PROG_IDX : ECC Program Entry Index**

Offset: 48h	ECCRSA_CTRL18: ECC and RSA Engine Control	Init = 0x00000000
Bit	**R/W**	**Description**
31:19	RW	**Reserved**
18	RW	**RSA_DMEM_PD : RSA Data Memory Power Down**
17	RW	**ECC_DMEM_PD : ECC Data Memory Power Down**
16	RW	**ECC_PMEM_PD : ECC Program Memory Power Down**
15:9	RW	**Reserved**

to next page

from previous page

8	RW	**ECC_DMEM_AHB : Set Data Memory for AHB Access Mode**
7:6	RW	**Reserved**
5:4	RW	**ECC_RSA_MODE : ECC or RSA Mode Selection** 10: ECC Mode (default) 11: RSA Mode
3:2	RW	**Reserved**
1	RW	**ECC_DISABLE : Disable ECC Engine**
0	RW	**RSA_DISABLE : Disable RSA Engine**

Offset: 4Ch **ECCRSA_CTRL19: ECC Data Control** **Init = 0x00000000**

Bit	R/W	Description
31:0	RW	**ECC_RSA_DRAMBASE : DRAM Base Address of DMA**

Offset: 50h **ECCRSA_CTRL20: ECC Data Control** **Init = 0x00000000**

Bit	R/W	Description
31:16	RW	**ECC_RSA_DEST : Data Destination Address of DMA**
15:0	RW	**ECC_RSA_DATALEN : Data Length of DMA**

Offset: 54h **ECCRSA_CTRL21: ECC Data Control** **Init = 0x00000000**

Bit	R/W	Description
31:4	RW	**Reserved**
3:0	RW	**ECC_DMA_BURST : DRAM Read Burst Length of DMA**

Offset: 58h **ECCRSA_CTRL22: RSA Data Control** **Init = 0x00000000**

Bit	R/W	Description
31:16	RW	**RSA_E_BITS : RSA Exponent Bits**
15:0	RW	**RSA_M_BITS : RSA Modulus Bits**

Offset: 3F8h **ECCRSA_CTRL23: RSA INTERRUPT Control** **Init = 0x00000000**

Bit	R/W	Description
31:3	RW	**Reserved**
2	RW	**RSA_DMA_MASK : Interrupt Enable when RSA DATA Prepare Completed**
1	RW	**RSA_ENG_MASK : Interrupt Enable when RSA Process Completed**
0	RW	**Reserved**

Offset: 3FCh **ECCRSA_CTRL24: RSA INTERRUPT Status** **Init = 0x00000000**

Bit	R/W	Description
31:3	RW	**Reserved**
2	RW	**RSA_DMA_DONE : RSA DATA Prepare Completed**
1	RW	**RSA_ENG_DONE : RSA Process Completed**
0	RW	**Reserved**

27 JTAG Master Controller (JTAGM)

27.1 Overview

AST2600 support 2 identical JTAG Master controllers. First one is pin-multiplexing with JTAG slave interface which is 3.3v.
The other is pin-multiplexing with GPIOI which is 3.3v. JTAG Master follows Test Access Port(TAP) and state diagram in section 6.1 of IEEE 1149-1. It has flexible value and length of data and instuction respectively. JTAG Master Controller includes a new control mode. This mode can reduce numbers of register access to transfer big data. And it's also easier to control device's state.

Base Address of JTAG Master 1 = 0x1E6E_4000
Base Address of JTAG Master 2 = 0x1E6E_4100
Register Address = Base Address + Offset

JTAG00: Data register.
JTAG04: Instruction register.
JTAG08: Engine control.
JTAG0C: Interrupt status and enable.
JTAG10: Test mode and status.
JTAG14: TCK control.
JTAG20: Shift In/Out Data register 0.
JTAG24: Shift In/Out Data register 1.
JTAG28: Padding control 0.
JTAG2C: Padding control 1.
JTAG30: Shift control.
JTAG34: Global control.
JTAG38: Interrupt control.
JTAG3C: Status.

27.2 Features

- Following state diagram in section 6.1 of IEEE 1149-1.
- Flexible instruction/data combination.
- Internal FIFO capability of 512 bits.
- Interrupt when data or instruction transmission complete or pause.
- Software mode to direct control TCK, TMS and TDI through APB register.

27.3 Registers

JTAG_RSTN is Rst26 when JTAG Master 1 is using or Rst43 when JTAG Master 2 is using.

Offset: 00h	**JTAG00: Data Port register**	**Init = 0**
Offset: 04h	**JTAG04: Data Port register**	**Init = 0**

Bit	Attr.	Reset	Description
31: 0	RW	-	**Data Port register** Write: value of data to be sent. Read: value of data received.

Offset: 08h			JTAG08: Engine control	Init = 0
Bit	**R/W**	**Reset**	**Description**	
31	RW	JTAG_RST#	**Engine enable.**	
30	RW	JTAG_RST#	**Engine output enable.** 1: output enable. 0: output disable.	
29	RW	JTAG_RST#	**Force controller and slave into Reset state by TMS.**	
28:22	RO	-	**Reserved**	
21	RW	JTAG_RST#	**Reset Internal FIFO** Write 1 to Reset and auto clear	
20	RO	JTAG_RST#	**Internal FIFO mode** 0: CPU mode 1: Controller mode	
19:18	RO	-	**Reserved**	
17: 8	RW	JTAG_RST#	**Length in one transmission** 0: forbidden 1: 1 bit 512: 512 bits 513 and more is not supported.	
7	RO	-	**Reserved**	
6	RW	JTAG_RST#	**MSB first.** 1: MSB first. 0: LSB first.	
5	RW	JTAG_RST#	**Terminating transmission** 1: terminate. 0: normal operation.	
4	RW	JTAG_RST#	**Last transmission** 1: last transmission. 0: more data waited.	
1	RW	JTAG_RST#	**Enable transmission of instruction.**	
0	RW	JTAG_RST#	**Enable transmission of data.**	

Offset: 0Ch			JTAG0C: Interrupt status and enable	Init = 0
Bit	**R/W**	**Reset**	**Description**	
31:20	RO	-	**Reserved**	
19	RW	JTAG_RST#	**Interrupt status of instruction transmission pause.** Read 0: No interrupt pending Read 1: interrupt pending Write 0: No operation Write 1: Clear interrupt status flag	
18	RW	JTAG_RST#	**Instruction transmission completed.** Read 0: No interrupt pending Read 1: interrupt pending Write 0: No operation Write 1: Clear interrupt status flag	
17	RW	JTAG_RST#	**Data transmission pause.** Read 0: No interrupt pending Read 1: interrupt pending Write 0: No operation Write 1: Clear interrupt status flag	

to next page

from previous page

16	RW	JTAG_RST#	**Data transmission completed.** Read 0: No interrupt pending Read 1: interrupt pending Write 0: No operation Write 1: Clear interrupt status flag
15: 4	RO	-	**Reserved**
3	RW	JTAG_RST#	**Enable of Instruction transmission pause.** 0: Disable interrupt 1: Enable interrupt
2	RW	JTAG_RST#	**Enable of Instruction transmission completed.** 0: Disable interrupt 1: Enable interrupt
1	RW	JTAG_RST#	**Enable of Data transmission pause.** 0: Disable interrupt 1: Enable interrupt
0	RW	JTAG_RST#	**Enable of Data transmission completed.** 0: Disable interrupt 1: Enable interrupt

Offset: 10h **JTAG10: Software mode and status** **Init = 0**

Bit	R/W	Reset	Description
31:20	RO	-	**Reserved**
19	RW	JTAG_RST#	**Software mode enable.**
18	RW	JTAG_RST#	**Software TCK.** Direct connect to TCK when software mode enabled.
17	RW	JTAG_RST#	**Software TMS.** Direct connect to TMS when software mode enabled.
16	RW	JTAG_RST#	**Software TDI and TDO.** Write: Direct connect to TDI when software mode enabled. Read : Direct connect to TDO when software mode enabled
15: 3	RO	-	**Reserved**
2	RO	JTAG_RST#	**Instuction transmission pause.**
1	RO	JTAG_RST#	**Data transmission pause.**
0	RO	JTAG_RST#	**Engine idle.**

Offset: 14h **JTAG14: TCK Control** **Init = 0x0000_0007**

Bit	R/W	Reset	Description
31	RW	JTAG_RST#	**TCK inverse.** 1: Inverse TCK. 0: Not inverse TCK.
30:11	RO	-	**Reserved**
10: 0	RW	JTAG_RST#	**TCK divisor.** TCK period = Period of HCLK * (JTAG14[10:0] + 1)

Offset: 18h	JTAG18: Engine Control 1	Init = 0x8000_0000

Bit	R/W	Reset	Description
31	RW	JTAG_RST#	**Control of TRSTn.** 1: TRSTn is high. 0: TRSTn is low.
30: 0	RO	-	**Reserved**

Offset: 20h	JTAG20: Data Port register	Init = 0
Offset: 24h	JTAG24: Data Port register	Init = 0

Bit	Attr.	Reset	Description
31: 0	RW	-	**Shift Data [31:00]** Write: data to shift out. Read : data from shift in.

Offset: 28h	JTAG28: Padding control 0	Init = 0

Bit	R/W	Reset	Description
31:25	RW	-	**Reserved**
24	RW	JTAG_RST#	**Padding Data**
23:21	RW	-	**Reserved**
20:12	RW	JTAG_RST#	**Post Padding Number** 0 : No post padding. 1 : Post padding 1 bit. 2 : Post padding 2 bits. ... 511 : Post padding 511 bits.
11: 9	RW	-	**Reserved**
8: 0	RW	JTAG_RST#	**Pre Padding Number** 0 : No Pre padding. 1 : Pre padding 1 bit. 2 : Pre padding 2 bits. ... 511 : Pre padding 511 bits.

Offset: 2Ch	JTAG2C: Padding control 1	Init = 0

Bit	R/W	Reset	Description
31:25	RW	-	**Reserved**
24	RW	JTAG_RST#	**Padding Data**
23:21	RW	-	**Reserved**
20:12	RW	JTAG_RST#	**Post Padding Number** 0 : No post padding. 1 : Post padding 1 bit. 2 : Post padding 2 bits. ... 511 : Post padding 511 bits.
11: 9	RW	-	**Reserved**
8: 0	RW	JTAG_RST#	**Pre Padding Number** 0 : No pre padding. 1 : Pre padding 1 bit. 2 : Pre padding 2 bits. ... 511 : Pre padding 511 bits.

Offset: 30h			JTAG30: Shift control	Init = 0
Bit	**R/W**	**Reset**	**Description**	
31	RW	JTAG_RST#	**Enable Free Run TCK** 1 : TCK is free running. 0 : TCK toggles only when necessary.	
30	RW	JTAG_RST#	**Enable Static Shift**	
29:16	RW	JTAG_RST#	**TMS Value.**	
15:13	RW	JTAG_RST#	**Post TMS Shift Number.** If JTAG30 bit 8 is 0, this field must be also 0. 0 : No post TMS shift. 1 : Post TMS shift 1 cycle. 2 : Post TMS shift 2 cycles. ... 7 : Post TMS shift 7 cycles.	
12:10	RW	JTAG_RST#	**Pre TMS Shift Number.** If JTAG30 bit 7 is 0, this field must be also 0. 0 : No pre TMS shift. 1 : Pre TMS shift 1 cycle. 2 : Pre TMS shift 2 cycles. ... 7 : Pre TMS shift 7 cycles.	
9	RW	JTAG_RST#	**Paddign selection** 0 : Use JTAG28 for padding setting. 1 : Use JTAG2C for padding setting.	
8	RW	JTAG_RST#	**End of shift**	
7	RW	JTAG_RST#	**Start of shift**	
6: 0	RW	JTAG_RST#	**Lower Data Shift Number.** Actual Data shift number is cascading Upper and Lower Data Shift Number. 0 : No data shift. 1 : Data shift 1 bit. 2 : Data shift 2 bits. ... 512: Data shift 512 bit. Others: forbiddened	

Offset: 34h			JTAG34: Global control	Init = 0x8007
Bit	**R/W**	**Reset**	**Description**	
31	RW	JTAG_RST#	**Engine and Mode Enable**	
30	RW	JTAG_RST#	**Engine output enable.** Only takes effect when JTAG34[31] is 1. 1: output enable. 0: output disable.	
29	RW	JTAG_RST#	**Force controller and slave into Reset state by TMS.** Valid only when JTAG34[31] is 1.	
28:26	RO	-	**Reserved**	
25	RW	JTAG_RST#	**Reset Internal FIFO** Write 1 to Reset and auto clear	
24	RO	-	**Internal FIFO mode** 0: CPU mode 1: Controller mode	
23	RW	-	**Reserved**	

to next page

from previous page

22:20	RW	JTAG_RSTN	**Upper Data Shift Number.** Actual Data shift number is cascading Upper and Lower Data Shift Number. 0 : No data shift. 1 : Data shift 1 bit. 2 : Data shift 2 bits. ... 512: Data shift 512 bit. Others: forbiddened
19:17	RW	-	**Reserved**
16	RW	JTAG_RSTN	**Static Shift Value** When JTAG30[30] is 1, shift this bit out.
15	RW	JTAG_RSTN	**TRST value**
14:12	RW	-	**Reserved**
11: 0	RW	JTAG_RSTN	**Clock divisor** TCK period = Period of HCLK * (JTAG34[10:0] + 1)

Offset: 38h			**JTAG38: Interrupt control**	**Init = 0**
Bit	**R/W**	**Reset**	**Description**	
31:17	RW	-	**Reserved**	
16	RW	JTAG_RSTN	**Shift Complete Interrupt Enable**	
15: 1	RW	-	**Reserved**	
0	RW	JTAG_RSTN	**Shift Complete Interrupt Status** Read 0: No interrupt pending Read 1: interrupt pending Write 0: No operation Write 1: Clear interrupt status flag	

Offset: 3Ch			**JTAG3C: Status**	**Init = 0**
Bit	**R/W**	**Reset**	**Description**	
31: 1	RO	-	**Reserved**	
0	RO	-	**Engine Idle**	

27.4 Operation

27.4.1 Mode of JTAG Master Controller

- Software mode. FW control TCK/TMC/TDI and monitor TDO directly by JTAG10
- Hardware mode 1. FW just needs to program data and length. HW will take care the rest. JTAG00 JTAG14
- Hardware mode 2. It's similar to Hardware mode 1 except Hardware mode 2 is more efficient to daisy chain devices. JTAG20 JTAG3C

27.4.2 Reset State Machine

- Set JTAG08[31:30] or JTAG34[31:30] to 0x3.
- Set JTAG08[29] or JTAG34[29]. This will assert TMS for at least 5 cycles.
- Wait JTAG08[29] or JTAG34[29] to be 0.

27.4.3 Instruction Transmission

If instruction length is not larger than 512 bits.

- Set JTAG08[21] to reset internal FIFO.
- Write all instruction to JTAG04 32 bits by 32 bits.
- Write instruction length to JTAG08[17:8].
- Set JTAG08[4] to indicate that it is last transmission.
- Set JTAG08[1] to enable transmission.
- Wait JTAG0C[18] or JTAG10[0] to be 1.

If insturction length is larger than 512 bits.

- Set JTAG08[21] to reset internal FIFO.
- Write 512 bits of instruction to JTAG04.
- Write instruction length as 0x200 to JTAG08[17:8].
- Set JTAG08[1] to enable transmission.
- Wait JTAG0C[19] or JTAG10[2] to be 1.
- repeat above steps until last transmission.
- Set JTAG08[21] to reset internal FIFO.
- Write rest instruction to JTAG04.
- Write rest instruction length to JTAG08[17:8].
- Set JTAG08[4] to indicate that it is last transmission.
- Set JTAG08[1] to enable transmission.
- Wait JTAG0C[18] or JTAG10[0] to be 1.

27.4.4 Data Transmission

If Data length is not larger than 512 bits.

- Set JTAG08[21] to reset internal FIFO.
- Write all Data to JTAG00 32 bits by 32 bits.
- Write Data length to JTAG08[17:8].
- Set JTAG08[4] to indicate that it is last transmission.
- Set JTAG08[0] to enable transmission.
- Wait JTAG0C[16] or JTAG10[0] to be 1.

If insturction length is larger than 512 bits.

- Set JTAG08[21] to reset internal FIFO.
- Write 512 bits of Data to JTAG00.
- Write Data length as 0x200 to JTAG08[17:8].
- Set JTAG08[0] to enable transmission.
- Wait JTAG0C[17] or JTAG10[1] to be 1.
- repeat above steps until last transmission.
- Set JTAG08[21] to reset internal FIFO.
- Write rest Data to JTAG00.
- Write rest Data length to JTAG08[17:8].
- Set JTAG08[4] to indicate that it is last transmission.
- Set JTAG08[0] to enable transmission.
- Wait JTAG0C[16] or JTAG10[0] to be 1.

27.5 Application Note

27.5.1 Introduction

There are 3 operation modes in AST2600 JTAG master controller.
First two is hardware mode (or accelerated mode), the rest is software mode.

In hardware mode, JTAG interface is controlled by internal state machine.
When JTAG master wants to program device's instruction or data register, just put instruction or data to controller and fire command.
Then all action will be taken care by hardware.

In software mode, the interface is controlled by software directly.

27.5.2 Hardware Mode

After enabled JTAG master controller, it's in hardware mode by default.
To get controller ready, an initiation sequence by F/W is required.

1. Set expected TCK frequency.
2. Set output enable bit to switch pin mux to JTAG master.
3. Reset device's state machine.

Then put instruction or data to register and fire a command.
Controller will send an interrupt if enabled when command is completed or paused. Polling status register is an alternative way.

For HW1 mode, each bit has to be programmed into JTAG00 and/or JTAG04. And TCK is a free-running clock. So transfer is only able to stop at PAUSE-IR, PAUSE-DR, or Run-Test/Idle.
For HW2 mode, it is for a complex JTAG chain operation, like following figure.

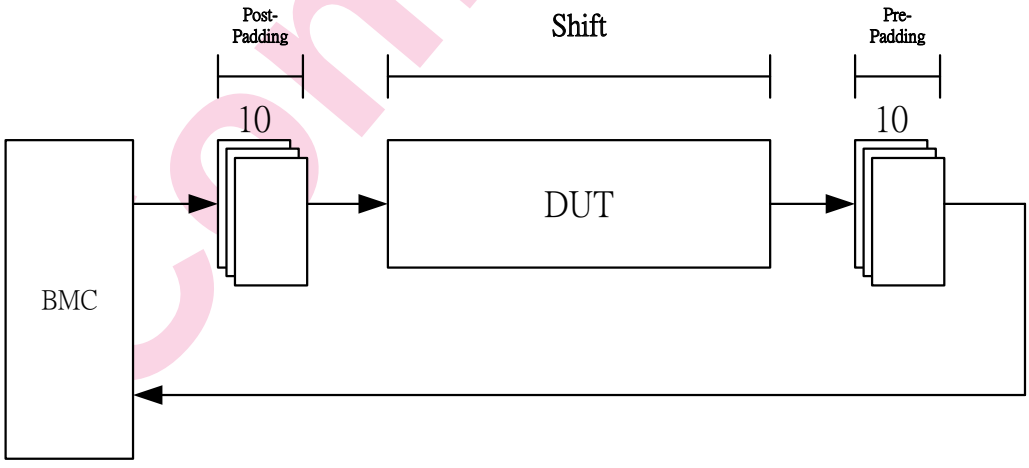


Figure 49: JTAG chain example

There are 21 devices chained together and target device is 11th in chain. To save register access in HW1 for those 20 devices, HW2 uses Pre-/Post-padding for them. Only bits consumed by DUT have to be programmed to JTAG20 and/or JTAG24. TCK is NOT a free-running clock. So state of JTAG interface can be stopped at any state. Pre-/Post-TMS is used to changing state before and after shift and padding. For detail initiation sequence, please contact ASPEED for the sample code.

27.5.3 Software Mode

To use software mode, software mode enable bit must be set.
Then TCK, TMS, and TDI are controlled by software directly.
And TDO is read back from register.

28 SOC Display Controller (GFX)

28.1 Overview

Base Address of SOC Display Controller = 0x1E6E_6000
Physical address of register = (Base address of SOC Display Controller) + Offset

GFX060: CRT Control Register I
GFX064: CRT Control Register II
GFX068: CRT Status Register
GFX070: CRT Horizontal Total & Display Enable End Register
GFX074: CRT Horizontal Retrace Start & End Register
GFX078: CRT Vertical Total & Display Enable End Register
GFX07C: CRT Vertical Retrace Start & End Register
GFX080: CRT Display Starting Address Register
GFX084: CRT Display Offset & Terminal Count Register
GFX088: CRT Threshold Register
GFX090: CRT Hardware Cursor X & Y Offset Register
GFX094: CRT Hardware Cursor X & Y Position Register
GFX098: CRT Hardware Cursor Pattern Address Register
GFX0B8: CRT Status Register V
GFX0BC: Scratchpad Register

28.2 Features

- Support dynamic switching the triple DAC display output and VESA DisplayPort between VGA and SOC Display Controller
- Support RGB565 and XRGB8888 graphics display mode
- Support hardware mono/color cursor
- Maximum display resolution without dedicated PLL: 1600x1200 32bpp@60Hz
- Maximum display resolution with dedicated PLL: 1920x1080 32bpp@60Hz

28.3 SOC Display Controller Registers

Offset: 60h		**GFX060: CRT Control Register I**	**Init = 30000000h**
Bit	**R/W**	**Description**	
0	RW	**Enable CRT graphics display** 0: disable 1: enable	
1	RW	**Enable CRT hardware cursor display** 0: disable 1: enable	
2	RW	**Reserved**	
3	RW	**Enable CRT interlace timing** 0: disable 1: enable	
6:4	RW	**Reserved**	

to next page

from previous page

9:7	RW	**CRT color format selection** 000 : RGB565 (RGB output) 010 : XRGB8888(RGB output) other: Reserved
10	RW	**CRT hardware cursor pattern format** 0: XRGB4444 1: ARGB4444
15:11	RW	**Reserved**
16	RW	**CRT horizontal sync polarity selection** 1: negative polarity 0: positive polarity
17	RW	**CRT vertical sync polarity selection** 1: negative polarity 0: positive polarity
18	RW	**Enable CRT horizontal sync off** 0: disable 1: enable
19	RW	**Enable CRT vertical sync off** 0: disable 1: enable
20	RW	**Enable CRT screen off** 0: disable 1: enable
22:21	RW	**Reserved**
23	RW	**Enable CRT desk off** 0: disable 1: enable
27:24	RW	**Reserved**
29:28	RO	**Reserved**
30	RW	**Enable CRT vertical interrupt to ARM** 0: disable 1: enable
31	RO	**CRT vertical interrupt status** 0: no interrupt 1: interrupt is pending Writing GFX068[31] ”1” will clear this status

Offset: 68h **GFX068: CRT Status Register** **Init = xxxxxxxxh**

Bit	R/W	Description
0	RO	**CRT vertical retrace signal read back**
1	RO	**CRT vertical display enable read back**
2	RO	**CRT horizontal retrace signal read back**
3	RO	**CRT horizontal display enable signal read back**
4	RO	**OddField: Odd field frame status**
5	RO	**OddFieldSync: Odd field sync status**
7 :6	RW	**Reserved**
15:8	RO	**CRT blue signature read back bit[7:0]**

to next page

from previous page

23:16	RO	**CRT green signature read back bit[7:0]**
31:24	RO	**CRT red signature read back bit[7:0]**
31	W1C	**Writing ”1” will clear the status GFX060[31]**

Offset: 70h GFX070: CRT Horizontal Total & Display Enable End Register Init = xxxxxxxxh

Bit	R/W	Description
12:0	RW	**CRT horizontal total bit[12:0] (-1)**
15:13	RO	**Reserved**
28:16	RW	**CRT horizontal display enable end bit[12:0] (-1)**
31:29	RO	**Reserved**

Offset: 74h GFX074: CRT Horizontal Retrace Start & End Register Init = xxxxxxxxh

Bit	R/W	Description
12:0	RW	**CRT horizontal retrace start bit[12:0] (-1)**
15:13	RO	**Reserved**
28:16	RW	**CRT horizontal retrace end bit[12:0] (-1)**
31:29	RO	**Reserved**

Offset: 78h GFX078: CRT Vertical Total & Display Enable End Register Init = 0xxx0xxxh

Bit	R/W	Description
11:0	RW	**CRT vertical total bit[11:0] (-1)**
15:12	RO	**Reserved**
27:16	RW	**CRT vertical display enable end bit[11:0] (-1)**
31:28	RO	**Reserved**

Offset: 7Ch GFX07C: CRT Vertical Retrace Start & End Register Init = 0xxx0xxxh

Bit	R/W	Description
11:0	RW	**CRT vertical retrace start bit[11:0] (-1)**
15:12	RO	**Reserved**
27:16	RW	**CRT vertical retrace end bit[11:0] (-1)**
31:28	RO	**Reserved**

Offset: 80h GFX080: CRT Display Starting Address Register Init = xxxxxxxxh

Bit	R/W	Description
0	RO	**Reserved**
1	RW	**EnIntlAg: Enable interlace address display mode**
2	RO	**Reserved**
3	RO	**Reserved**
30:4	RW	**CRT display starting address bit[30:4]**
31	RO	**Reserved**

Offset: 84h	GFX084: CRT Display Offset & Terminal Count Register	Init = 0xxxxxxxh
Bit	**R/W**	**Description**
3 :0	RO	**Reserved**
15:4	RW	**CRT display offset bit[15:4]**
28:16	RW	**CRT terminal count[12:0] (-0)**
31:29	RO	**Reserved**

Offset: 88h	GFX088: CRT Threshold Register	Init = 00xxxxxxh
Bit	**R/W**	**Description**
6:0	RW	**CRT threshold low bit[6:0]**
7	RO	**Reserved**
14:8	RW	**CRT threshold high bit[6:0]**
31:15	RO	**Reserved**

Offset: 90h	GFX090: CRT Hardware Cursor X & Y Offset Register	Init = 0000xxxxh
Bit	**R/W**	**Description**
5:0	RW	**CRT hardware cursor X offset bit[5:0]**
7:6	RO	**Reserved**
13:8	RW	**CRT hardware cursor Y offset bit[5:0]**
31:14	RO	**Reserved**

Offset: 94h	GFX094: CRT Hardware Cursor X & Y Position Register	Init = 0xxxxxxxh
Bit	**R/W**	**Description**
12:0	RW	**CRT hardware cursor X position bit[12:0]**
15:13	RO	**Reserved**
27:16	RW	**CRT hardware cursor Y position bit[11:0]** Update cursor after writing this register
31:28	RO	**Reserved**

Offset: 98h	GFX098: CRT Hardware Cursor Pattern Address Register	Init = xxxxxxxxh
Bit	**R/W**	**Description**
0	RO	**Reserved**
1	RW	**EnIntlAc: Enable interlace address cursor mode**
2	RO	**Reserved**
3	RO	**Reserved**
30:4	RW	**CRT hardware cursor pattern address bit[30:4]**
31	RO	**Reserved**

Offset: B8h	GFX0B8: CRT Status Register V	Init = 00000xxxh
Bit	**R/W**	**Description**
11:0	RO	**VCnt: Vertical count status**
31:12	RO	**Reserved**

Offset: BCh		GFX0BC: Scratchpad Register	Init = 00000000h
Bit	**R/W**	**Description**	
31:0	RW	**Scratchpad bit[31:0]**	

29 X-DMA Controller (XDMA)

29.1 Overview

For HOST: Base address of XDMA Engine = (MMIO Base Address) + 0x1000
Register address of XDMA Engine = (Base address of XDMA Engine) + Offset

For BMC: Base address of XDMA Engine = 0x1e6e_7000
Register address of XDMA Engine = (Base address of XDMA Engine) + Offset

XDMA00: HOST Command Queue Base Address [31:3]
XDMA04: HOST Command Queue Base Address [63:32]
XDMA08: HOST Command Queue End Pointer
XDMA0C: HOST Command Queue Write Pointer
XDMA10: HOST Command Queue Read Pointer
XDMA14: BMC Command Queue Base Address [30:4]
XDMA18: BMC Command Queue End Pointer
XDMA1C: BMC Command Queue Write Pointer
XDMA20: BMC Command Queue Read Pointer
XDMA24: VGA Command Queue Base Address [31:3]
XDMA28: VGA Command Queue Base Address [63:32]
XDMA2C: VGA Command Queue End Pointer
XDMA30: VGA Command Queue Write Pointer
XDMA34: VGA Command Queue Read Pointer
XDMA38: Interrupt Enable and Engine Control
XDMA3C: Interrupt Flag and Engine Status
XDMA40: In Processing Down-stream Command #0[31:00]
XDMA44: In Processing Down-stream Command #0[63:32]
XDMA48: In Processing Down-stream Command #1[31:00]
XDMA4c: In Processing Down-stream Command #1[63:32]
XDMA50: In Processing Down-stream Command #2[31:00]
XDMA54: In Processing Down-stream Command #2[63:32]
XDMA60: In Processing Up-stream Command #0[31:00]
XDMA64: In Processing Up-stream Command #0[63:32]
XDMA68: In Processing Up-stream Command #1[31:00]
XDMA6c: In Processing Up-stream Command #1[63:32]
XDMA70: In Processing Up-stream Command #2[31:00]
XDMA74: In Processing Up-stream Command #2[63:32]

29.2 Features

- Independent descriptor space for BMC and HOST.
- Both BMC and HOST can transfer data upward or downward.
- Flexible interrupt target.
- Interrupt for complete and receive non-successful completion.

29.3 Registers

Offset: 00h	XDMA00: HOST Command Queue Base Address [31:3]		Init = 0
Bit	**R/W**	**Reset**	**Description**
31:3	RW	RstPE	**Base address of HOST command queue [31:3].**
2 :1	RO	-	**Reserved (0)**
0	RW	RstPE	**Enable HOST Command Queue 64 bit address mode**

Offset: 04h	XDMA04: HOST Command Queue Base Address [63:32]		Init = X
Bit	**R/W**	**Reset**	**Description**
31:0	RW	RstPE	**Base address of HOST command queue [63:32].**

Offset: 08h	XDMA08: HOST Command Queue End Pointer		Init = 0
Bit	**R/W**	**Reset**	**Description**
31:18	RO	-	**Reserved (0)**
17:0	RW	RstPE	**End pointer of HOST command queue [17:0], in unit of 8 bytes.**

Offset: 0Ch	XDMA0C: HOST Command Queue Write Pointer		Init = 0
Bit	**R/W**	**Reset**	**Description**
31:18	RO	-	**Reserved (0)**
17:0	RW	RstPE	**Write pointer of HOST command queue [17:0], in unit of 8 bytes.**

Offset: 10h	XDMA10: HOST Command Queue Read Pointer		Init = 0
Bit	**R/W**	**Reset**	**Description**
31	RO	Rst29	**Valid for setting read/write pointer of HOST command queue**
30:18	RO	-	**Reserved (0)**
17:0	RW	Rst29	**Read pointer of HOST command queue [17:0], in unit of 8 bytes.**

Offset: 14h	XDMA14: BMC Command Queue Base Address		Init = 0
Bit	**R/W**	**Reset**	**Description**
31	RO	-	**Reserved (0)**
30:4	RW	Rst29	**Base address of BMC command queue [30:4].**
3 :0	RO	-	**Reserved (0)**

Offset: 18h	XDMA18: BMC Command Queue End Pointer		Init = 0
Bit	**R/W**	**Reset**	**Description**
31:18	RO	-	**Reserved (0)**
17:0	RW	Rst29	**End pointer of BMC command queue [17:0], in unit of 16 bytes.**

Offset: 1Ch	XDMA1C: BMC Command Queue Write Pointer		Init = 0
Bit	**R/W**	**Reset**	**Description**
31:18	RO	-	**Reserved (0)**
17:0	RW	Rst29	**Write pointer of BMC command queue [17:0], in unit of 16 bytes.**

Offset: 20h	XDMA20: BMC Command Queue Read Pointer	Init = X

Bit	R/W	Reset	Description
31	RO	Rst29	**Valid for setting read/write pointer of BMC command queue**
30:18	RO	-	**Reserved (0)**
17:0	RW	Rst29	**Read pointer of BMC command queue [17:0], in unit of 8 bytes.**

Offset: 24h	XDMA24: VGA Command Queue Base Address [31:3]	Init = 0

Bit	R/W	Reset	Description
31:3	RW	RstPE	**Base address of HOST command queue [31:3].**
2 :1	RO	-	**Reserved (0)**
0	RW	RstPE	**Enable VGA Command Queue 64 bit address mode**

Offset: 28h	XDMA28: VGA Command Queue Base Address [63:32]	Init = X

Bit	R/W	Reset	Description
31:0	RW	RstPE	**Base address of HOST command queue [63:32].**

Offset: 2Ch	XDMA2C: VGA Command Queue End Pointer	Init = 0

Bit	R/W	Reset	Description
31:18	RO	-	**Reserved (0)**
17:0	RW	RstPE	**End pointer of HOST command queue [17:0], in unit of 8 bytes.**

Offset: 30h	XDMA30: VGA Command Queue Write Pointer	Init = 0

Bit	R/W	Reset	Description
31:18	RO	-	**Reserved (0)**
17:0	RW	RstPE	**Write pointer of HOST command queue [17:0], in unit of 8 bytes.**

Offset: 34h	XDMA34: VGA Command Queue Read Pointer	Init = 0

Bit	R/W	Reset	Description
31	RO	Rst29	**Valid for setting read/write pointer of HOST command queue**
30:18	RO	-	**Reserved (0)**
17:0	RW	Rst29	**Read pointer of HOST command queue [17:0], in unit of 8 bytes.**

Access from HOST

Offset: 38h	XDMA38: Interrupt Enable and Engine Control	Init = 0

Bit	R/W	Reset	Description
31:25	RO	-	**Reserved (0)**
24	RW	RstPE	**Interrupt Type to HOST** 1: Falling edge. 0: Low level.
23	RO	-	**Reserved (0)**

to next page

from previous page

22:20	RW	RstPE	**Down-stream PCIe Request Size Selection[2:0].** 0: 128 bytes. 1: 256 bytes. 2: 512 bytes. 3: 1K bytes. 4: 2K bytes. 5: 4K bytes. 6~7: Reserved.
19:10	RO	-	**Reserved (0)**
9	RW	RstPE	**Enable VGA command queue dirty frame INT.** **Only access by HOST**
8	RW	RstPE	**Enable HOST command queue dirty frame INT.** **Only access by HOST**
7	RW	RstPE	**Enable down-stream dirty frame INT from VGA CQ.** **Only access by HOST**
6	RW	RstPE	**Enable down-stream dirty frame INT from HOST CQ.** **Only access by HOST**
5	RW	RstPE	**Enable down-stream complete INT from BMC CQ.** **Only access by HOST**
4	RW	RstPE	**Enable up-stream complete INT from BMC CQ.** **Only access by HOST**
3	RW	RstPE	**Enable down-stream complete INT from VGA CQ.** **Only access by HOST**
2	RW	RstPE	**Enable up-stream complete INT from VGA CQ.** **Only access by HOST**
1	RW	RstPE	**Enable down-stream complete INT from HOST CQ.** **Only access by HOST**
0	RW	RstPE	**Enable up-stream complete INT from HOST CQ.** **Only access by HOST**

Access from BMC

Offset: 38h			**XDMA38: Interrupt Enable and Engine Control**	**Init = 0**
Bit	**R/W**	**Reset**	**Description**	
31:23	RO	-	**Reserved (0)**	
22:20	RW	Rst29	**Down-stream PCIe Request Size Selection[2:0].** 0: 128 bytes. 1: 256 bytes. 2: 512 bytes. 3: 1K bytes. 4: 2K bytes. 5: 4K bytes. 6~7: Reserved.	
19:12	RO	-	**Reserved (0)**	
18	RW	Rst29	**Enable down-stream dirty frame INT from BMC CQ.** **Only access by BMC**	
17	RW	Rst29	**Enable down-stream complete INT from BMC CQ.** **Only access by BMC**	
16	RW	Rst29	**Enable up-stream complete INT from BMC CQ.** **Only access by BMC**	

to next page

from previous page

15	RW	Rst29	**Enable down-stream complete INT from VGA CQ. Only access by BMC**
14	RW	Rst29	**Enable up-stream complete INT from VGA CQ. Only access by BMC**
13	RW	Rst29	**Enable down-stream complete INT from HOST CQ. Only access by BMC**
12	RW	Rst29	**Enable up-stream complete INT from HOST CQ. Only access by BMC**
11: 0	RO	-	**Reserved (0)**

Access from HOST

Offset: 3Ch	**XDMA3C: Interrupt Flag and Engine Status**		**Init = X**
Bit	**R/W**	**Reset**	**Description**
31	RO	Rst29	**XDMA all engine IDLE.**
30	RO	Rst29	**HOST command queue IDLE.**
29	RO	Rst29	**BMC command queue IDLE.**
28	RO	Rst29	**Down-stream engine IDLE.**
27	RO	Rst29	**Up-steram engine IDLE.**
26	RO	Rst29	**VGA command queue IDLE.**
25	RW	RstPE	**HOST command queue in Dirty Frame.**
24	RW	RstPE	**VGA command queue in Dirty Frame.**
23	RO	Rst29	**Down-stream in Dirty Frame from BMC CQ.**
22	RW	RstPE	**Down-stream in Dirty Frame from VGA CQ.**
21	RW	RstPE	**Down-stream in Dirty Frame from HOST CQ.**
20:19	RO	-	**Reserved (0)**
18	RO	Rst29	**down-stream dirty frame INT from BMC CQ.**
17	RO	Rst29	**down-stream complete INT from BMC CQ.**
16	RO	Rst29	**up-stream complete INT from BMC CQ.**
15	RO	Rst29	**down-stream complete INT from VGA CQ.**
14	RO	Rst29	**up-stream complete INT from VGA CQ.**
13	RO	Rst29	**down-stream complete INT from HOST CQ.**
12	RO	Rst29	**up-stream complete INT from HOST CQ.**
11:10	RO	-	**Reserved (0)**
9	RW	RstPE	**VGA command queue dirty frame INT.**
8	RW	RstPE	**HOST command queue dirty frame INT.**
7	RW	RstPE	**down-stream dirty frame INT from VGA CQ.**
6	RW	RstPE	**down-stream dirty frame INT from HOST CQ.**
5	RW	RstPE	**down-stream complete INT from BMC CQ.**
4	RW	RstPE	**up-stream complete INT from BMC CQ.**
3	RW	RstPE	**down-stream complete INT from VGA CQ.**
2	RW	RstPE	**up-stream complete INT from VGA CQ.**
1	RW	RstPE	**down-stream complete INT from HOST CQ.**
0	RW	RstPE	**up-stream complete INT from HOST CQ.**

Access from BMC

Offset: 3Ch	XDMA3C: Interrupt Flag and Engine Status	Init = X

Bit	R/W	Reset	Description
31	RO	Rst29	**XDMA all engine IDLE.**
30	RO	Rst29	**HOST command queue IDLE.**
29	RO	Rst29	**BMC command queue IDLE.**
28	RO	Rst29	**Down-stream engine IDLE.**
27	RO	Rst29	**Up-steram engine IDLE.**
26	RO	Rst29	**VGA command queue IDLE.**
25	RO	RstPE	**HOST command queue in Dirty Frame.**
24	RO	RstPE	**VGA command queue in Dirty Frame.**
23	RW	Rst29	**Down-stream in Dirty Frame from BMC CQ.**
22	RO	RstPE	**Down-stream in Dirty Frame from VGA CQ.**
21	RO	RstPE	**Down-stream in Dirty Frame from HOST CQ.**
20:19	RO	-	**Reserved (0)**
18	RW	Rst29	**down-stream dirty frame INT from BMC CQ.**
17	RW	Rst29	**down-stream complete INT from BMC CQ.**
16	RW	Rst29	**up-stream complete INT from BMC CQ.**
15	RW	Rst29	**down-stream complete INT from VGA CQ.**
14	RW	Rst29	**up-stream complete INT from VGA CQ.**
13	RW	Rst29	**down-stream complete INT from HOST CQ.**
12	RW	Rst29	**up-stream complete INT from HOST CQ.**
11:10	RO	-	**Reserved (0)**
9	RO	RstPE	**VGA command queue dirty frame INT.**
8	RO	RstPE	**HOST command queue dirty frame INT.**
7	RO	RstPE	**down-stream dirty frame INT from VGA CQ.**
6	RO	RstPE	**down-stream dirty frame INT from HOST CQ.**
5	RO	RstPE	**down-stream complete INT from BMC CQ.**
4	RO	RstPE	**up-stream complete INT from BMC CQ.**
3	RO	RstPE	**down-stream complete INT from VGA CQ.**
2	RO	RstPE	**up-stream complete INT from VGA CQ.**
1	RO	RstPE	**down-stream complete INT from HOST CQ.**
0	RO	RstPE	**up-stream complete INT from HOST CQ.**

Offset: 40h	XDMA40: In Processing Down-stream Command #0[31:00]	Init = X

Bit	R/W	Reset	Description
31:0	RO	RstPE	**In Processing Down-stream Command #0[31:00].**

Offset: 44h	XDMA44: In Processing Down-stream Command #0[63:32]	Init = X

Bit	R/W	Reset	Description
31:0	RO	RstPE	**In Processing Down-stream Command #0[63:32].**

Offset: 48h	XDMA48: In Processing Down-stream Command #1[31:00]		Init = X
Bit	**R/W**	**Reset**	**Description**
31:0	RO	RstPE	**In Processing Down-stream Command #1[31:00].**

Offset: 4Ch	XDMA4C: In Processing Down-stream Command #1[63:32]		Init = X
Bit	**R/W**	**Reset**	**Description**
31:0	RO	RstPE	**In Processing Down-stream Command #1[63:32].**

Offset: 50h	XDMA50: In Processing Down-stream Command #2[31:00]		Init = X
Bit	**R/W**	**Reset**	**Description**
31:0	RO	RstPE	**In Processing Down-stream Command #2[31:00].**

Offset: 54h	XDMA54: In Processing Down-stream Command #2[63:32]		Init = X
Bit	**R/W**	**Reset**	**Description**
31:0	RO	RstPE	**In Processing Down-stream Command #2[63:32].**

Offset: 60h	XDMA60: In Processing Up-stream Command #0[31:00]		Init = X
Bit	**R/W**	**Reset**	**Description**
31:0	RO	Rst29	**In Processing Up-stream Command #0[31:00].**

Offset: 64h	XDMA64: In Processing Up-stream Command #0[63:32]		Init = X
Bit	**R/W**	**Reset**	**Description**
31:0	RO	Rst29	**In Processing Up-stream Command #0[63:32].**

Offset: 68h	XDMA68: In Processing Up-stream Command #1[31:00]		Init = X
Bit	**R/W**	**Reset**	**Description**
31:0	RO	Rst29	**In Processing Up-stream Command #1[31:00].**

Offset: 6Ch	XDMA6C: In Processing Up-stream Command #1[63:32]		Init = X
Bit	**R/W**	**Reset**	**Description**
31:0	RO	Rst29	**In Processing Up-stream Command #1[63:32].**

Offset: 70h	XDMA70: In Processing Up-stream Command #2[31:00]		Init = X
Bit	**R/W**	**Reset**	**Description**
31:0	RO	Rst29	**In Processing Up-stream Command #2[31:00].**

Offset: 74h	XDMA74: In Processing Up-stream Command #2[63:32]		Init = X
Bit	**R/W**	**Reset**	**Description**
31:0	RO	Rst29	**In Processing Up-stream Command #2[63:32].**

29.4 Command Format

Offset: 00h	Comm#0: PCIe Data Base Address	Init = X
Bit	**R/W**	**Description**
63:0	RW	**Base address of PCIe data[63:00].**

Offset: 08h	Comm#1: Pitch and Base Address	Init = X
Bit	**R/W**	**Description**
63	RO	**Reserved (0)**
62:48	RW	**Pitch of MBus data[14:0].**
47	RO	**Reserved (0)**
46:32	RW	**Pitch of PCIe data[14:0].**
31	RO	**Reserved (0)**
30:00	RW	**Base address of MBus data[30:00].**

Offset: 10h	Comm#2: Fire Command	Init = X
Bit	**R/W**	**Description**
63:41	RO	**Reserved (0)**
40	RW	**Enable 64 bit address**
39:38	RO	**Reserved (0)**
37	RW	**Enable Interrupt to BMC**
36	RW	**Enable Interrupt to HOST**
35:34	RO	**Reserved (0)**
33	RW	**Command Direction.** 1: VGA 0: HOST or BMC
32	RW	**Transfer Direction.** 1: Up-stream. 0: Down-stream.
31:28	RO	**Reserved**
27:16	RW	**Frame line number[11:0].** Value 1h is 1 line, and value 2h is 2 lines.
15	RO	**Reserved**
14:0	RW	**Frame line byte size[14:00].** Value 0h is 0 byte, and value 1h is 1 byte.

Offset: 18h	Comm#3: Reserved Command	Init = X
Bit	**R/W**	**Description**
63: 0	RO	**Reserved (0)**

29.5 Clearing and Setting Procedure

29.5.1 In Dirty Frame Clearing Procedure

If a Read operation on PCIe gets a response of ”Un-Successful Completion”, it goes into ”Dirty Frame” state. In this state, Command Queue or Down-stream engine will stop until the flag is cleared.

Command Queue in Dirty Frame Clearing Precedure

- Write XDMA3C[25] or XDMA3C[8] to clear Dirty Frame state of HOST command queue.
- Write XDMA3C[24] or XDMA3C[9] to clear Dirty Frame state of VGA command queue.

Down-stream in Dirty Frame Clearing Procedure

- For HOST initiating operation, Write XDMA3C[21] or XDMA3C[8] to clear Dirty Frame state.
- For VGA initiating operation, Write XDMA3C[22] or XDMA3C[9] to clear Dirty Frame state.
- For BMC initiating operation, Write XDMA3C[23] or XDMA3C[18] to clear Dirty Frame state.

29.5.2 HOST Command Queue Read/Write Pointer Setting Procedure

1. Write XDMA0C with password.
2. Write XDMA08 with pointer value.

29.5.3 BMC Command Queue Read/Write Pointer Setting Procedure

1. Write XDMA1C with password.
2. Write XDMA18 with pointer value.

29.5.4 VGA Command Queue Read/Write Pointer Setting Procedure

1. Write XDMA70 with password.
2. Write XDMA6C with pointer value.

30 MCTP Controller (MCTP)

30.1 Overview

MCTP Engine receives Msg which carries MCTP packet through PCIe from Host or sends it to Host. Support transmission unit is 64/128/256/512 bytes.

Base Address of MCTP_EP = 0x1E6E_8000
Base Address of MCTP_RC = 0x1E6F_9000
Register Address = Base Address + Offset

MCTP00: Engine Status and Engine Control.
MCTP04: TX Command Address.
MCTP08: RX Buffer Address.
MCTP0C: Interrupt Status (MCTP_EP only)
MCTP10: Interrupt Enable (MCTP_EP only)
MCTP14: MCTP EID
MCTP18: OBFF Control
MCTP1C: Engine Control 1
MCTP20: RX Buffer Address
MCTP24: RX Buffer Size
MCTP28: RX Buffer Update Pointer
MCTP2C: RX Buffer Service Pointer
MCTP30: TX Command Address
MCTP34: TX Command Size
MCTP38: TX Command Read Pointer
MCTP3C: TX Command Write Pointer

30.2 Features

- Descriptor type DMA engine.
- Supports Maximum Payload Size of 512/256/128/64 bytes.
- Flexible for receiving all kinds of MsgD or MCTP only.
- Option of matching EID.
- MCTP_EP supports interrupt for receiving or sending complete. MCTP_RC does not.
- PCIe VDM Header is fully accessible.

30.3 Registers

Offset: 00h	MCTP00: Engine Status and Engine Control		Init = 0x00330000
Bit	**R/W**	**Reset**	**Description**
31:24	RO	Rst27	**Current command count**
23:22	RO	-	**Reserved**
21	RO	Rst27	**RX PCIe idle**
20	RO	Rst27	**RX DMA idle**
19:18	RO	-	**Reserved**
17	RO	Rst27	**TX PCIe idle**

to next page

from previous page

16	RO	Rst27	**TX DMA idle**
15	RW	Rst27	**Disable Patch** 0: Enable HW Patch for Errata #38 & #64 1: Disable HW Patch for Errata #38 & #64
14:10	RO	-	**Reserved**
9	RW	Rst27	**Matching MCTP EID** 1: Only accept packet whose MCTP Destination EID matches MCTP14[7:0]
8: 5	RO	-	**Reserved**
4	RW	Rst27	**RX Command is ready.** This bit will be cleared when RX last command is used.
3: 0	RO	-	**Reserved**
0	RW	Rst27	**Trigger TX operation.**

Offset: 04h **MCTP04: TX Command Address** **Init = 0**
Offset: 30h **MCTP30: TX Command Address** **Init = 0**

Bit	Attr.	Reset	Description
31:30	RO	-	**Reserved**
29: 3	RW	RstFull	**TX command address**
2: 0	RO	-	**Reserved**

Offset: 08h **MCTP08: RX Buffer Address** **Init = 0**
Offset: 20h **MCTP20: RX Buffer Address** **Init = 0**

Bit	Attr.	Reset	Description
31:30	RO	-	**Reserved**
29: 4	RW	RstFull	**RX Buffer address**
3: 0	RO	-	**Reserved**

Offset: 0Ch **MCTP0C: Interrupt Status (MCTP_EP only)** **Init = 0**

Bit	R/W	Reset	Description
31:28	RO	-	**Reserved**
27	RO	Rst27	**MSG obff state changes.**
26	RO	Rst27	**MSG obff in active state.**
25	RO	Rst27	**MSG obff in idle state.**
24	RO	Rst27	**MSG obff in obff state.**
23:20	RO	-	**Reserved**
19	RO	Rst27	**Wake obff in active state.**
18	RO	Rst27	**Wake obff in post-obff state.**
17	RO	Rst27	**Wake obff in obff state.**
16	RO	Rst27	**Wake obff in idle state.**
15:10	RO	-	**Reserved**
9	RO	Rst27	**No more RX command.**
8	RO	Rst27	**RX receiving complete**

to next page

from previous page

7: 3	RO	-	**Reserved**
2	RO	Rst27	**Set when TX command is wrong.**
1	RO	Rst27	**Set when all TX command is sent.**
0	RO	Rst27	**Set when CMD1[15] is set.**

Offset: 10h	**MCTP10: Interrupt Enable (MCTP_EP only)**		**Init = 0**
Bit	**R/W**	**Reset**	**Description**
31:28	RO	-	**Reserved**
27	RW	Rst27	**Enable interrupt of MSG obff state changes.**
26	RW	Rst27	**Enable interrupt of MSG obff in active state.**
25	RW	Rst27	**Enable interrupt of MSG obff in idle state.**
24	RW	Rst27	**Enable interrupt of MSG obff in obff state.**
23:20	RO	-	**Reserved**
19	RW	Rst27	**Enable interrupt of Wake obff in active state.**
18	RW	Rst27	**Enable interrupt of Wake obff in post-obff state.**
17	RW	Rst27	**Enable interrupt of Wake obff in obff state.**
16	RW	Rst27	**Enable interrupt of Wake obff in idle state.**
15:10	RO	-	**Reserved**
9	RW	Rst27	**Enable interrupt of No more RX command.**
8	RW	Rst27	**Enable interrupt of RX receiving complete**
7: 3	RO	-	**Reserved**
2	RW	Rst27	**Enalbe interrupt of TX command is wrong**
1	RW	Rst27	**Enalbe interrupt of TX last command is sent**
0	RW	Rst27	**Enalbe interrupt of TX sending complete**

Offset: 14h	**MCTP14: MCTP EID**		**Init = 0**
Bit	**R/W**	**Reset**	**Description**
31: 8	RO	-	**Reserved**
7: 0	RW	Rst27	**MCTP EID** Valid when MCTP00[9] = 1’b1 Compare with EID of incoming packet.

Offset: 18h	**MCTP18: OBFF Control**		**Init = 0**
Bit	**R/W**	**Reset**	**Description**
29:16	RO	Rst27	**WAKE OBFF status**
15	RO	-	**Reserved**
14	RO	Rst27	**MSG OBFF is in active state**
13	RO	Rst27	**MSG OBFF is in idle state**
12	RO	Rst27	**MSG OBFF is in obff state**
11: 6	RO	-	**Reserved**
5: 0	RW	Rst27	**Wake OBFF control**

Offset: 1Ch	MCTP1C: Engine Control 1	Init = 0

Bit	R/W	Reset	Description
31:20	RO	-	**Reserved**
19	RW	RstFull	**FIFO full condintion** 1: Incoming packet will be halted until FIFO is not full. 0: Incoming packet will be dropped.
18	RW	RstFull	**Enable OBFF DMA.**
17	RW	RstFull	**Enable OBFF monitoring.**
16:10	RO	-	**Reserved**
9: 8	RW	RstFull	**FIFO layout** 0: 6KB for TX; 2KB for RX. 1: 4KB for TX; 4KB for RX. 2: 2KB for TX; 6KB for RX. 3: forbidden.
7: 6	RO	-	**Reserved**
5: 4	RW	RstFull	**RX Maximum Payload Size** 0: 64 bytes. 1: 128 bytes. 2: 256 bytes. 3: 512 bytes.
3: 2	RO	-	**Reserved**
1: 0	RW	RstFull	**TX Maximum Payload Size** 0: 64 bytes. 1: 128 bytes. 2: 256 bytes. ~~3: 512 bytes.~~

Offset: 24h	MCTP24: RX Buffer Size	Init = 0

Bit	R/W	Reset	Description
31:13	RO	-	**Reserved**
12: 0	RW	Rst27	**RX Buffer Size** In unit of Maximum Payload Size (MCTP1C[5:4])

Offset: 28h	MCTP28: RX Buffer Update Pointer	Init = 0

Bit	R/W	Reset	Description
31	RW	Rst27	**Update RX Buffer Update Pointer**
30:12	RO	-	**Reserved**
11: 0	RW	Rst27	**RX Buffer Update Pointer** This pointer indicates which descriptor is going to be consumed for next incoming packet. This field updates only when bit 31 is set. It is to avoid getting transitional value.

Offset: 2Ch	MCTP2C: RX Buffer Service Pointer	Init = 0

Bit	R/W	Reset	Description
31:12	RO	-	**Reserved**
11: 0	RW	Rst27	**RX Buffer Service Pointer** This pointer indicates which descriptor is going to be service by driver.

Offset: 34h	MCTP34: TX Command Size	Init = 0

Bit	R/W	Reset	Description
31:13	RO	-	**Reserved**
12: 0	RW	Rst27	**TX Command Size** In unit of descriptor

Offset: 38h	MCTP38: TX Command Read Pointer	Init = 0

Bit	R/W	Reset	Description
31	RW	Rst27	**Update TX Command Read Pointer**
30:12	RO	-	**Reserved**
11: 0	RW	Rst27	**TX Command Read Pointer** This pointer indicates which descriptor is going to be sent. This field updates only when bit 31 is set. It is to avoid getting transitional value.

Offset: 3Ch	MCTP3C: TX Command Write Pointer	Init = 0

Bit	R/W	Reset	Description
31:12	RO	-	**Reserved**
11: 0	RW	Rst27	**TX Command Write Pointer** This pointer indicates the last prepared TX descriptors.

30.4 Command

30.4.1 TX command

Offset: 00h	TX Commnand	Init = 0

Bit	R/W	Description
63	RO	**Last command**
62:36	RO	**Data address[30:4]**
35:34	RO	**Reserved**
33:32	RO	**0x1**
31:17	RO	**Reserved**
16	RO	**Stop after this command**
15	RO	**Generate Interrupt after this command**
14:13	RO	**Reserved**
12: 2	RO	**Packet size in 4bytes**
1: 0	RO	**0x0**

30.4.2 RX buffer

Offset: 00h	RX Buffer	Init = 0

Bit	R/W	Description
31	RO	**Reserved**

to next page

from previous page

30: 4	RO	**RX address[30:4]**
3	RO	**Reserved**
2	RO	**Generate Interrput when this command is comsumed.**
1: 0	RO	**Reserved**

30.5 Operation

30.5.1 Send Packet

1. Set proper address to MCTP04 or MCTP30.
2. Prepare command to address set to MCTP04 or MCTP30 one by one.
   - Prepare PCIe Medium-Specific Header to Data Address.
   - Prepare following MCTP Transport Header
   - Prepare following MCTP Packet Payload
3. Program MCTP3C for the number of TX commands filled in previous step.
4. Set MCTP00[0] to 1 to trigger TX.
5. Waiting transaction complete.
   - MCTP0C[0] indicates one command who set TXCMD[15] is completed.
   - MCTP0C[1] indicates the last command is completed.
   - MCTP0C[2] indicates TX command syntax is wrong.

30.5.2 Receive Packet

1. Set proper buffer address to MCTP08 or MCTP20.
2. Set proper buffer size in unit of MCTP1C[5:4].
3. Set MCTP00[4] to enable receiving.
4. Waiting transaction complete.
   - MCTP0C[8] indicates at lease one command is completed.
   - MCTP0C[9] indicates buffer write pointer is going to equal to buffer read pointer.
5. Whole PCIe Medium-Specific Header, MCTP Transport Header, and MCTP Packet Payload are stored in buffer.

31 ADC Controller (ADC)

31.1 Overview

ADC Engine has 2 Analog-to-Digital Convertor. Each one has 8 voltage sensing channels.
One of the 8 channels is also for battery sensing. It has internal dividing circuit.
Each channel has upper and lower bound.
Larger or smaller than bound triggers interrupt. There are second set bound for hysteresis.
Build-in a compensating method.

ADC000: Engine Control.
ADC004: Interrupt Enable and Interrupt Status.
ADC008: ADC VGA Detect Control.
ADC00C: ADC Clock Control.
ADC010: Data of Channel 1 and 0.
ADC014: Data of Channel 3 and 2.
ADC018: Data of Channel 5 and 4.
ADC01C: Data of Channel 7 and 6.
ADC030: Upper and Lower bound 0 of Channel 0.
ADC034: Upper and Lower bound 0 of Channel 1.
ADC038: Upper and Lower bound 0 of Channel 2.
ADC03C: Upper and Lower bound 0 of Channel 3.
ADC040: Upper and Lower bound 0 of Channel 4.
ADC044: Upper and Lower bound 0 of Channel 5.
ADC048: Upper and Lower bound 0 of Channel 6.
ADC04C: Upper and Lower bound 0 of Channel 7.
ADC070: Hysteresis Control and bound of Channel 0.
ADC074: Hysteresis Control and bound of Channel 1.
ADC078: Hysteresis Control and bound of Channel 2.
ADC07C: Hysteresis Control and bound of Channel 3.
ADC080: Hysteresis Control and bound of Channel 4.
ADC084: Hysteresis Control and bound of Channel 5.
ADC088: Hysteresis Control and bound of Channel 6.
ADC08C: Hysteresis Control and bound of Channel 7.
ADC0C0: Interrupt Source Selection.
ADC0C4: Compensating and Trimming.
ADC0CC: Global Interrupt Status.

ADC100: Engine Control.
ADC104: Interrupt Enable and Interrupt Status.
ADC108: ADC VGA Detect Control.
ADC10C: ADC Clock Control.
ADC110: Data of Channel 9 and 8.
ADC114: Data of Channel 11 and 10.
ADC118: Data of Channel 13 and 12.
ADC11C: Data of Channel 15 and 14.
ADC130: Upper and Lower bound 0 of Channel 8.
ADC134: Upper and Lower bound 0 of Channel 9.
ADC138: Upper and Lower bound 0 of Channel 10.
ADC13C: Upper and Lower bound 0 of Channel 11.
ADC140: Upper and Lower bound 0 of Channel 12.
ADC144: Upper and Lower bound 0 of Channel 13.
ADC148: Upper and Lower bound 0 of Channel 14.
ADC14C: Upper and Lower bound 0 of Channel 15.
ADC170: Hysteresis Control and bound of Channel 8.
ADC174: Hysteresis Control and bound of Channel 9.

ADC178: Hysteresis Control and bound of Channel 10.
ADC17C: Hysteresis Control and bound of Channel 11.
ADC180: Hysteresis Control and bound of Channel 12.
ADC184: Hysteresis Control and bound of Channel 13.
ADC188: Hysteresis Control and bound of Channel 14.
ADC18C: Hysteresis Control and bound of Channel 15.
ADC1C0: Interrupt Source Selection.
ADC1C4: Compensating and Trimming.
ADC1CC: Global Interrupt Status.

31.2 Features

- 10-bits resolution for 8x2 voltage channels.
- Channel scanning can be non-continuous.
- Internal or External reference voltage.
- Support 2 Internal reference voltage: 1.2v or 2.5v.
- Integrate dividing circuit for battery sensing.
- Programmable ADC clock frequency.
- Programmable upper and lower bound for each channels.
- Interrupt when larger or less than bounds for each channels.
- Support hysteresis for each channels.
- Buildin a compensating method.

31.3 Registers : Base Address = 0x1E6E_9000

Offset: 000h			**ADC000: Engine Control**	**Init = 0**
Bit	**R/W**	**Reset**	**Description**	
31:24	RO	-	**Reserved**	
23:16	RW	Rst51	**Channel enable.** 1: Enable 0: Skip ADC00[23] for channel 7. ADC00[22] for channel 6. ADC00[21] for channel 5. ADC00[20] for channel 4. ADC00[19] for channel 3. ADC00[18] for channel 2. ADC00[17] for channel 1. ADC00[16] for channel 0.	
15:14	RO	-	**Reserved**	
13	RW	Rst51	**Enable Battery Sensing**	
12	RW	Rst51	**Channel 7 Selection** 0: Normal Voltage 1: Battery	
11: 9	RO	-	**Reserved**	
8	RO	Rst51	**Initial sequence complete**	

to next page

from previous page

7: 6	RW	Rst51	**Reference Voltage Selection** 00b : 2.5v 01b : 1.2v 10b : External Voltage (1.55v to 2.7v) 11b : External Voltage (0.9v to 1.65v)
5	RW	Rst51	**Auto Compensating sensing mode.** 1: Trigger compensating method. 0: Compensating method is done.
4	RW	Rst51	**Compensating sensing mode.**
3: 1	RW	Rst51	**ADC Operation mode.** 000b: Power down mode. 001b: Standby mode. 111b: Normal mode.
0	RW	Rst51	**Engine enable.**

Offset: 004h **ADC004: Interrupt Enable and Interrupt Status** **Init = 0**

Bit	R/W	Reset	Description
31:24	RO	-	**Reserved**
23:16	RW	Rst51	**Interrupt enable** 1: Enable 0: Disable ADC04[23] for channel 7. ADC04[22] for channel 6. ADC04[21] for channel 5. ADC04[20] for channel 4. ADC04[19] for channel 3. ADC04[18] for channel 2. ADC04[17] for channel 1. ADC04[16] for channel 0.
15: 8	RO	-	**Reserved**
7: 0	RW	Rst51	**Interrupt status** Write 1 to clear ADC04[07] for channel 7. ADC04[06] for channel 6. ADC04[05] for channel 5. ADC04[04] for channel 4. ADC04[03] for channel 3. ADC04[02] for channel 2. ADC04[01] for channel 1. ADC04[00] for channel 0.

Offset: 008h **ADC008: ADC VGA Detect Control** **Init = 0x0000_000F**

Bit	R/W	Reset	Description
31:17	RO	-	**Reserved**
16	RW	Rst51	**ADC VGA detect enable**
15: 0	RW	Rst51	**Divisor of ADC clock used for VGA detection** Period ADC clock = period of PCLK * 2 * (ADC008[15:0] + 1)

Offset: 00Ch	ADC00C: ADC Clock Control	Init = 0x0000_000F

Bit	R/W	Reset	Description
31:16	RO	-	**Reserved**
15: 0	RW	Rst51	**Divisor of ADC clock** Period of ADC clock = period of PCLK * 2 * (ADC0C[15:0] + 1) Less than 6MHz is recommand. Sample rate is Period of ADC clock * 12

Offset: 010h	ADC010: Data of Channel 1 and 0	Init = 0

Bit	R/W	Reset	Description
31:26	RO	-	**Reserved**
25:16	RO	Rst51	**Data of Channel 1**
15:10	RO	-	**Reserved**
9: 0	RO	Rst51	**Data of Channel 0**

Offset: 014h	ADC014: Data of Channel 3 and 2	Init = 0

Bit	R/W	Reset	Description
31:26	RO	-	**Reserved**
25:16	RO	Rst51	**Data of Channel 3**
15:10	RO	-	**Reserved**
9: 0	RO	Rst51	**Data of Channel 2**

Offset: 018h	ADC018: Data of Channel 5 and 4	Init = 0

Bit	R/W	Reset	Description
31:26	RO	-	**Reserved**
25:16	RO	Rst51	**Data of Channel 5**
15:10	RO	-	**Reserved**
9: 0	RO	Rst51	**Data of Channel 4**

Offset: 01Ch	ADC01C: Data of Channel 7 and 6	Init = 0

Bit	R/W	Reset	Description
31:26	RO	-	**Reserved**
25:16	RO	Rst51	**Data of Channel 7**
15:10	RO	-	**Reserved**
9: 0	RO	Rst51	**Data of Channel 6**

Offset: 030h	ADC030: Upper and Lower bound of Channel 0	Init = 0

Bit	R/W	Reset	Description
31:26	RO	-	**Reserved**
25:16	RW	Rst51	**Upper bound**
15:10	RO	-	**Reserved**
9: 0	RW	Rst51	**Lower bound**

Offset: 034h	ADC034: Upper and Lower bound of Channel 1		Init = 0
Bit	**R/W**	**Reset**	**Description**
31:26	RO	-	**Reserved**
25:16	RW	Rst51	**Upper bound**
15:10	RO	-	**Reserved**
9: 0	RW	Rst51	**Lower bound**

Offset: 038h	ADC038: Upper and Lower bound of Channel 2		Init = 0
Bit	**R/W**	**Reset**	**Description**
31:26	RO	-	**Reserved**
25:16	RW	Rst51	**Upper bound**
15:10	RO	-	**Reserved**
9: 0	RW	Rst51	**Lower bound**

Offset: 03Ch	ADC03C: Upper and Lower bound of Channel 3		Init = 0
Bit	**R/W**	**Reset**	**Description**
31:26	RO	-	**Reserved**
25:16	RW	Rst51	**Upper bound**
15:10	RO	-	**Reserved**
9: 0	RW	Rst51	**Lower bound**

Offset: 040h	ADC040: Upper and Lower bound of Channel 4		Init = 0
Bit	**R/W**	**Reset**	**Description**
31:26	RO	-	**Reserved**
25:16	RW	Rst51	**Upper bound**
15:10	RO	-	**Reserved**
9: 0	RW	Rst51	**Lower bound**

Offset: 044h	ADC044: Upper and Lower bound of Channel 5		Init = 0
Bit	**R/W**	**Reset**	**Description**
31:26	RO	-	**Reserved**
25:16	RW	Rst51	**Upper bound**
15:10	RO	-	**Reserved**
9: 0	RW	Rst51	**Lower bound**

Offset: 048h	ADC048: Upper and Lower bound of Channel 6		Init = 0
Bit	**R/W**	**Reset**	**Description**
31:26	RO	-	**Reserved**
25:16	RW	Rst51	**Upper bound**
15:10	RO	-	**Reserved**
9: 0	RW	Rst51	**Lower bound**

Offset: 04Ch	ADC04C: Upper and Lower bound of Channel 7	Init = 0

Bit	R/W	Reset	Description
31:26	RO	-	**Reserved**
25:16	RW	Rst51	**Upper bound**
15:10	RO	-	**Reserved**
9: 0	RW	Rst51	**Lower bound**

Offset: 070h	ADC070: Hysteresis Control and bound of Channel 0	Init = 0

Bit	R/W	Reset	Description
31	RW	Rst51	**Hysteresis Control** 1: Enable hysteresis. 0: Disable hysteresis.
30:26	RO	-	**Reserved**
25:16	RW	Rst51	**Upper bound**
15:10	RO	-	**Reserved**
9: 0	RW	Rst51	**Lower bound**

Offset: 074h	ADC074: Hysteresis Control and bound of Channel 1	Init = 0

Bit	R/W	Reset	Description
31	RW	Rst51	**Hysteresis Control** 1: Enable hysteresis. 0: Disable hysteresis.
30:26	RO	-	**Reserved**
25:16	RW	Rst51	**Upper bound**
15:10	RO	-	**Reserved**
9: 0	RW	Rst51	**Lower bound**

Offset: 078h	ADC078: Hysteresis Control and bound of Channel 2	Init = 0

Bit	R/W	Reset	Description
31	RW	Rst51	**Hysteresis Control** 1: Enable hysteresis. 0: Disable hysteresis.
30:26	RO	-	**Reserved**
25:16	RW	Rst51	**Upper bound**
15:10	RO	-	**Reserved**
9: 0	RW	Rst51	**Lower bound**

Offset: 07Ch	ADC07C: Hysteresis Control and bound of Channel 3	Init = 0

Bit	R/W	Reset	Description
31	RW	Rst51	**Hysteresis Control** 1: Enable hysteresis. 0: Disable hysteresis.
30:26	RO	-	**Reserved**
25:16	RW	Rst51	**Upper bound**

to next page

from previous page

15:10	RO	-	**Reserved**
9: 0	RW	Rst51	**Lower bound**

Offset: 080h	**ADC080: Hysteresis Control and bound of Channel 4**	**Init = 0**

Bit	R/W	Reset	Description
31	RW	Rst51	**Hysteresis Control** 1: Enable hysteresis. 0: Disable hysteresis.
30:26	RO	-	**Reserved**
25:16	RW	Rst51	**Upper bound**
15:10	RO	-	**Reserved**
9: 0	RW	Rst51	**Lower bound**

Offset: 084h	**ADC084: Hysteresis Control and bound of Channel 5**	**Init = 0**

Bit	R/W	Reset	Description
31	RW	Rst51	**Hysteresis Control** 1: Enable hysteresis. 0: Disable hysteresis.
30:26	RO	-	**Reserved**
25:16	RW	Rst51	**Upper bound**
15:10	RO	-	**Reserved**
9: 0	RW	Rst51	**Lower bound**

Offset: 088h	**ADC088: Hysteresis Control and bound of Channel 6**	**Init = 0**

Bit	R/W	Reset	Description
31	RW	Rst51	**Hysteresis Control** 1: Enable hysteresis. 0: Disable hysteresis.
30:26	RO	-	**Reserved**
25:16	RW	Rst51	**Upper bound**
15:10	RO	-	**Reserved**
9: 0	RW	Rst51	**Lower bound**

Offset: 08Ch	**ADC08C: Hysteresis Control and bound of Channel 7**	**Init = 0**

Bit	R/W	Reset	Description
31	RW	Rst51	**Hysteresis Control** 1: Enable hysteresis. 0: Disable hysteresis.
30:26	RO	-	**Reserved**
25:16	RW	Rst51	**Upper bound**
15:10	RO	-	**Reserved**
9: 0	RW	Rst51	**Lower bound**

Offset: 0C0h	ADC0C0: Interrupt Source	Init = 0

Bit	R/W	Reset	Description
31: 8	RO	-	**Reserved**
7: 0	RW	Rst51	**Interrupt Source.** 1: Primary Processor 0: Coprocessor ADCC0[7] for channel 7. ADCC0[6] for channel 6. ADCC0[5] for channel 5. ADCC0[4] for channel 4. ADCC0[3] for channel 3. ADCC0[2] for channel 2. ADCC0[1] for channel 1. ADCC0[0] for channel 0.

Offset: 0C4h	ADC0C4: Compensating and Trimming	Init = 0

Bit	R/W	Reset	Description
31:26	RO	-	**Reserved**
25:16	RO	Rst51	**Compensating value.**
15: 4	RO	-	**Reserved**
3: 0	RW	Rst51	**Trimming value.**

Offset: 100h	ADC100: Engine Control	Init = 0

Bit	R/W	Reset	Description
31:24	RO	-	**Reserved**
23:16	RW	Rst51	**Channel enable.** 1: Enable 0: Skip ADC00[23] for channel 15. ADC00[22] for channel 14. ADC00[21] for channel 13. ADC00[20] for channel 12. ADC00[19] for channel 11. ADC00[18] for channel 10. ADC00[17] for channel 9. ADC00[16] for channel 8.
15:14	RO	-	**Reserved**
13	RW	Rst51	**Enable Battery Sensing**
12	RW	Rst51	**Channel 15 Selection** 0: Normal Voltage 1: Battery
11: 9	RO	-	**Reserved**
8	RO	Rst51	**Initial sequence complete**
7: 6	RW	Rst51	**Reference Voltage Selection** 00b : 2.5v 01b : 1.2v 10b : External Voltage (1.55v to 2.7v) 11b : External Voltage (0.9v to 1.65v)

to next page

from previous page

5	RW	Rst51	**Auto Compensating sensing mode.** 1: Trigger compensating method. 0: Compensating method is done.
4	RW	Rst51	**Compensating sensing mode.**
3: 1	RW	Rst51	**ADC Operation mode.** 000b: Power down mode. 001b: Standby mode. 111b: Normal mode.
0	RW	Rst51	**Engine enable.**

Offset: 104h **ADC104: Interrupt Enable and Interrupt Status** **Init = 0**

Bit	R/W	Reset	Description
31:24	RO	-	**Reserved**
23:16	RW	Rst51	**Interrupt enable** 1: Enable 0: Disable ADC104[23] for channel 15. ADC104[22] for channel 14. ADC104[21] for channel 13. ADC104[20] for channel 12. ADC104[19] for channel 11. ADC104[18] for channel 10. ADC104[17] for channel 9. ADC104[16] for channel 8.
15: 8	RO	-	**Reserved**
7: 0	RW	Rst51	**Interrupt status** Write 1 to clear ADC104[07] for channel 15. ADC104[06] for channel 14. ADC104[05] for channel 13. ADC104[04] for channel 12. ADC104[03] for channel 11. ADC104[02] for channel 10. ADC104[01] for channel 9. ADC104[00] for channel 8.

Offset: 108h **ADC108: ADC VGA Detect Control** **Init = 0x0000_000F**

Bit	R/W	Reset	Description
31:17	RO	-	**Reserved**
16	RW	Rst51	**ADC VGA detect enable**
15: 0	RW	Rst51	**Divisor of ADC clock used for VGA detection** Period ADC clock = period of PCLK * 2 * (ADC108[15:0] + 1)

Offset: 10Ch **ADC10C: ADC Clock Control** **Init = 0x0000_000F**

Bit	R/W	Reset	Description
31:16	RO	-	**Reserved**

to next page

from previous page

15: 0	RW	Rst51	**Divisor of ADC clock** Period of ADC clock = period of PCLK * 2 * (ADC10C[15:0] + 1) Less than 6MHz is recommand. Sample rate is Period of ADC clock * 12

Offset: 110h	**ADC110: Data of Channel 9 and 8**		**Init = 0**
Bit	**R/W**	**Reset**	**Description**
31:26	RO	-	**Reserved**
25:16	RO	Rst51	**Data of Channel 9**
15:10	RO	-	**Reserved**
9: 0	RO	Rst51	**Data of Channel 8**

Offset: 114h	**ADC114: Data of Channel 11 and 10**		**Init = 0**
Bit	**R/W**	**Reset**	**Description**
31:26	RO	-	**Reserved**
25:16	RO	Rst51	**Data of Channel 11**
15:10	RO	-	**Reserved**
9: 0	RO	Rst51	**Data of Channel 10**

Offset: 118h	**ADC118: Data of Channel 13 and 12**		**Init = 0**
Bit	**R/W**	**Reset**	**Description**
31:26	RO	-	**Reserved**
25:16	RO	Rst51	**Data of Channel 13**
15:10	RO	-	**Reserved**
9: 0	RO	Rst51	**Data of Channel 12**

Offset: 11Ch	**ADC11C: Data of Channel 15 and 14**		**Init = 0**
Bit	**R/W**	**Reset**	**Description**
31:26	RO	-	**Reserved**
25:16	RO	Rst51	**Data of Channel 15**
15:10	RO	-	**Reserved**
9: 0	RO	Rst51	**Data of Channel 14**

Offset: 130h	**ADC130: Upper and Lower bound of Channel 8**		**Init = 0**
Bit	**R/W**	**Reset**	**Description**
31:26	RO	-	**Reserved**
25:16	RW	Rst51	**Upper bound**
15:10	RO	-	**Reserved**
9: 0	RW	Rst51	**Lower bound**

Offset: 134h	ADC134: Upper and Lower bound of Channel 9	Init = 0

Bit	R/W	Reset	Description
31:26	RO	-	**Reserved**
25:16	RW	Rst51	**Upper bound**
15:10	RO	-	**Reserved**
9: 0	RW	Rst51	**Lower bound**

Offset: 138h	ADC138: Upper and Lower bound of Channel 10	Init = 0

Bit	R/W	Reset	Description
31:26	RO	-	**Reserved**
25:16	RW	Rst51	**Upper bound**
15:10	RO	-	**Reserved**
9: 0	RW	Rst51	**Lower bound**

Offset: 13Ch	ADC13C: Upper and Lower bound of Channel 11	Init = 0

Bit	R/W	Reset	Description
31:26	RO	-	**Reserved**
25:16	RW	Rst51	**Upper bound**
15:10	RO	-	**Reserved**
9: 0	RW	Rst51	**Lower bound**

Offset: 140h	ADC140: Upper and Lower bound of Channel 12	Init = 0

Bit	R/W	Reset	Description
31:26	RO	-	**Reserved**
25:16	RW	Rst51	**Upper bound**
15:10	RO	-	**Reserved**
9: 0	RW	Rst51	**Lower bound**

Offset: 144h	ADC144: Upper and Lower bound of Channel 13	Init = 0

Bit	R/W	Reset	Description
31:26	RO	-	**Reserved**
25:16	RW	Rst51	**Upper bound**
15:10	RO	-	**Reserved**
9: 0	RW	Rst51	**Lower bound**

Offset: 148h	ADC148: Upper and Lower bound of Channel 14	Init = 0

Bit	R/W	Reset	Description
31:26	RO	-	**Reserved**
25:16	RW	Rst51	**Upper bound**
15:10	RO	-	**Reserved**
9: 0	RW	Rst51	**Lower bound**

Offset: 14Ch	ADC14C: Upper and Lower bound of Channel 15	Init = 0

Bit	R/W	Reset	Description
31:26	RO	-	**Reserved**
25:16	RW	Rst51	**Upper bound**
15:10	RO	-	**Reserved**
9: 0	RW	Rst51	**Lower bound**

Offset: 170h	ADC170: Hysteresis Control and bound of Channel 8	Init = 0

Bit	R/W	Reset	Description
31	RW	Rst51	**Hysteresis Control** 1: Enable hysteresis. 0: Disable hysteresis.
30:26	RO	-	**Reserved**
25:16	RW	Rst51	**Upper bound**
15:10	RO	-	**Reserved**
9: 0	RW	Rst51	**Lower bound**

Offset: 174h	ADC174: Hysteresis Control and bound of Channel 9	Init = 0

Bit	R/W	Reset	Description
31	RW	Rst51	**Hysteresis Control** 1: Enable hysteresis. 0: Disable hysteresis.
30:26	RO	-	**Reserved**
25:16	RW	Rst51	**Upper bound**
15:10	RO	-	**Reserved**
9: 0	RW	Rst51	**Lower bound**

Offset: 178h	ADC178: Hysteresis Control and bound of Channel 10	Init = 0

Bit	R/W	Reset	Description
31	RW	Rst51	**Hysteresis Control** 1: Enable hysteresis. 0: Disable hysteresis.
30:26	RO	-	**Reserved**
25:16	RW	Rst51	**Upper bound**
15:10	RO	-	**Reserved**
9: 0	RW	Rst51	**Lower bound**

Offset: 17Ch	ADC17C: Hysteresis Control and bound of Channel 11	Init = 0

Bit	R/W	Reset	Description
31	RW	Rst51	**Hysteresis Control** 1: Enable hysteresis. 0: Disable hysteresis.
30:26	RO	-	**Reserved**
25:16	RW	Rst51	**Upper bound**

to next page

from previous page

15:10	RO	-	**Reserved**
9: 0	RW	Rst51	**Lower bound**

Offset: 180h	**ADC180: Hysteresis Control and bound of Channel 12**	**Init = 0**

Bit	R/W	Reset	Description
31	RW	Rst51	**Hysteresis Control** 1: Enable hysteresis. 0: Disable hysteresis.
30:26	RO	-	**Reserved**
25:16	RW	Rst51	**Upper bound**
15:10	RO	-	**Reserved**
9: 0	RW	Rst51	**Lower bound**

Offset: 184h	**ADC184: Hysteresis Control and bound of Channel 13**	**Init = 0**

Bit	R/W	Reset	Description
31	RW	Rst51	**Hysteresis Control** 1: Enable hysteresis. 0: Disable hysteresis.
30:26	RO	-	**Reserved**
25:16	RW	Rst51	**Upper bound**
15:10	RO	-	**Reserved**
9: 0	RW	Rst51	**Lower bound**

Offset: 188h	**ADC188: Hysteresis Control and bound of Channel 14**	**Init = 0**

Bit	R/W	Reset	Description
31	RW	Rst51	**Hysteresis Control** 1: Enable hysteresis. 0: Disable hysteresis.
30:26	RO	-	**Reserved**
25:16	RW	Rst51	**Upper bound**
15:10	RO	-	**Reserved**
9: 0	RW	Rst51	**Lower bound**

Offset: 18Ch	**ADC18C: Hysteresis Control and bound of Channel 15**	**Init = 0**

Bit	R/W	Reset	Description
31	RW	Rst51	**Hysteresis Control** 1: Enable hysteresis. 0: Disable hysteresis.
30:26	RO	-	**Reserved**
25:16	RW	Rst51	**Upper bound**
15:10	RO	-	**Reserved**
9: 0	RW	Rst51	**Lower bound**

Offset: 1C0h	ADC1C0: Interrupt Source	Init = 0

Bit	R/W	Reset	Description
31: 8	RO	-	**Reserved**
7: 0	RW	Rst51	**Interrupt Source.** 1: Primary Processor 0: Coprocessor ADCC0[7] for channel 15. ADCC0[6] for channel 14. ADCC0[5] for channel 13. ADCC0[4] for channel 12. ADCC0[3] for channel 11. ADCC0[2] for channel 10. ADCC0[1] for channel 9. ADCC0[0] for channel 8.

Offset: 1C4h	ADC1C4: Compensating and Trimming	Init = 0

Bit	R/W	Reset	Description
31:26	RO	-	**Reserved**
25:16	RO	Rst51	**Compensating value.**
15: 4	RO	-	**Reserved**
3: 0	RW	Rst51	**Trimming value.**

31.4 Operation

31.4.1 Initialize Sequence

1. Set ADC000/ADC100 to 0x0000000F.
2. Wait bit 8 of ADC000/ADC100 to be set.

31.4.2 Compensating Sensing Method

- Compensating sensing mode
  1. Set ADC000/ADC100 to 0x0000001F
  2. Set ADC000/ADC100 to 0x0001001F
  3. Wait a sensing cycle.
     - Sensing cycle of ADC1 = 12 * period of PCLK * 2 * (ADC00C[31:17] + 1) * (ADC00C[9:0] + 1)
     - Sensing cycle of ADC2 = 12 * period of PCLK * 2 * (ADC10C[31:17] + 1) * (ADC10C[9:0] + 1)
  4. Storing a sample of Compensating Value(CV) = 0x200 - ADC010[9:0]/ADC110[9:0]
  5. It is recommanded to average at least 10 samples to get a final CV.
  6. Set ADC00 to 0x0000000F
  7. Adding CV to every channel reading(i.e. ADC10[9:0] or ADC10[25:16])

31.4.3 Channel 7 or 15 selection and operation

1. Set what voltage is connected to channel 7 or 15. ADC000[12]/ADC100[12].
2. If it connects to normal voltage, its operation is like others.
3. If it connects to battery:
   - Integrated dividing circuit ratio is decided by ADC000[6]/ADC100[6].
     - 0: Voltage under test is 2/3 of external voltage.
     - 1: Voltage under test is 1/3 of external voltage.
   - Set 1 to ADC000[13]/ADC100[13] to enable battery sensing. Current begins to flow through dividing circuit.
   - Set 0 to ADC000[13]/ADC100[13] to turn off battery sensing and stop current draining.

31.4.4 Voltage Sense Method

Value read from register(VR) = $(V2 + \frac{R2}{R1+R2}(V1 - V2)) * \frac{1024}{Vref} - 1$
Vref depends on ADC000/ADC100 setting.
V1 = $(\frac{Vref}{1024}\frac{R1+R2}{R2}(VR + 1)) - \frac{R1}{R2}V2$

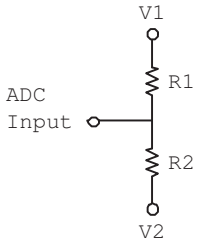


Figure 50: ADC Reference Circuit

32 AHB to PCIe Bus Bridge (H2X)

32.1 Overview

H2X bridge is a bridge between AHB bus and PCIe RC.
It also supports a direct mapping memory space. Base address and size is programmable

H2X00: AHB to PCIe RC Register 00
H2X04: AHB to PCIe RC Register 04
H2X08: AHB to PCIe RC Register 08
H2X0C: AHB to PCIe RC Register 0C
H2X10: AHB to PCIe RC Register 10
H2X14: AHB to PCIe RC Register 14
H2X18: AHB to PCIe RC Register 18
H2X1C: AHB to PCIe RC Register 1C
H2X20: AHB to PCIe RC Register 20
H2X24: AHB to PCIe RC Register 24
H2X28: AHB to PCIe RC Register 28
H2X2C: AHB to PCIe RC Register 2C
H2X30: AHB to PCIe RC Register 30
H2X34: AHB to PCIe RC Register 34
H2X38: AHB to PCIe RC Register 38
H2X3C: AHB to PCIe RC Register 3C
H2X60: AHB to PCIe RC Register 60
H2X64: AHB to PCIe RC Register 64
H2X68: AHB to PCIe RC Register 68
H2X70: AHB to PCIe RC Register 70
H2X74: AHB to PCIe RC Register 74
H2X78: AHB to PCIe RC Register 78
H2X7C: AHB to PCIe RC Register 7C
H2X80: AHB to PCIe RC Register 80
H2X84: AHB to PCIe RC Register 84
H2X88: AHB to PCIe RC Register 88
H2X8C: AHB to PCIe RC Register 8C
H2X90: AHB to PCIe RC Register 90
H2X94: AHB to PCIe RC Register 94
H2X98: AHB to PCIe RC Register 98
H2X9C: AHB to PCIe RC Register 9C
H2XA0: AHB to PCIe RC Register A0
H2XA4: AHB to PCIe RC Register A4
H2XA8: AHB to PCIe RC Register A8
H2XAC: AHB to PCIe RC Register AC
H2XBC: AHB to PCIe RC Register BC
H2XC0: AHB to PCIe RC Register C0
H2XC4: AHB to PCIe RC Register C4
H2XC8: AHB to PCIe RC Register C8
H2XCC: AHB to PCIe RC Register CC
H2XD0: AHB to PCIe RC Register D0
H2XD4: AHB to PCIe RC Register D4
H2XD8: AHB to PCIe RC Register D8
H2XDC: AHB to PCIe RC Register DC
H2XE0: AHB to PCIe RC Register E0
H2XE4: AHB to PCIe RC Register E4
H2XE8: AHB to PCIe RC Register E8
H2XEC: AHB to PCIe RC Register EC
H2XFC: AHB to PCIe RC Register FC

32.2 Features

- Connect to AHB bus and PCIe RC.
- Flexible packet type for sending and receiving.
- Interrupt for receiving or sending complete

32.3 Registers : Base Address = 0x1E77_0000

Offset: 00h	H2X00: AHB to PCIe RC Register 00	Init = 0x0
Bit	**R/W**	**Description**
31: 5	RW	**Reserved**
4	RW	**Clear RX buffer**
3: 1	RW	**Reserved**
0	RW	**Enable PCIe Host Bridge**

Offset: 04h	H2X04: AHB to PCIe RC Register 04	Init = 0x0
Bit	**R/W**	**Description**
31: 1	RW	**Reserved**
0	RW	**Enable interrupt**

Offset: 08h	H2X08: AHB to PCIe RC Register 08	Init = 0x0
Bit	**R/W**	**Description**
31: 1	RW	**Reserved**
0	RW	**TX idle status: write 1 clear**

Offset: 0Ch	H2X0C: AHB to PCIe RC Register 0C	Init = 0x0
Bit	**R/W**	**Description**
31: 0	RO	**Read data of PCIe Host Bridge**

Offset: 10h	H2X10: AHB to PCIe RC Register 10	Init = 0x0
Bit	**R/W**	**Description**
31: 0	RW	**TX descriptor[127:96]**

Offset: 14h	H2X14: AHB to PCIe RC Register 14	Init = 0x0
Bit	**R/W**	**Description**
31: 0	RW	**TX descriptor[95:64]**

Offset: 18h	H2X18: AHB to PCIe RC Register 18	Init = 0x0
Bit	**R/W**	**Description**
31: 0	RW	**TX descriptor[63:32]**

Offset: 1Ch	H2X1C: AHB to PCIe RC Register 1C	Init = 0x0
Bit	**R/W**	**Description**
31: 0	RW	**TX descriptor[31:0]**

Offset: 20h	H2X20: AHB to PCIe RC Register 20	Init = 0x0
Bit	**R/W**	**Description**
31: 0	W	**TX Data**

Offset: 24h	H2X24: AHB to PCIe RC Register 24	Init = 0x0
Bit	**R/W**	**Description**
31	RO	**TX idle**
30:28	RO	**Reserved**
27	RO	**RC-H RX done**
26	RO	**RC-L RX done**
25:24	RO	**Status of TX**
23: 1	RO	**Reserved**
0	W	**Trigger TX**

Offset: 28h	H2X28: AHB to PCIe RC Register 28	Init = 0x0
Bit	**R/W**	**Description**
31: 0	RW	**Reserved**

Offset: 2Ch	H2X2C: AHB to PCIe RC Register 2C	Init = 0x0
Bit	**R/W**	**Description**
31: 0	RW	**Reserved**

Offset: 30h	H2X30: AHB to PCIe RC Register 30	Init = 0x26001A03
Bit	**R/W**	**Description**
31: 0	RW	**DID:VID**

Offset: 34h	H2X34: AHB to PCIe RC Register 34	Init = 0x00000006
Bit	**R/W**	**Description**
31: 0	RW	**Status:Command**

Offset: 38h	H2X38: AHB to PCIe RC Register 38	Init = 0x06000000
Bit	**R/W**	**Description**
31: 0	RW	**Code:RID**

Offset: 3Ch	H2X3C: AHB to PCIe RC Register 3C	Init = 0x26001A03
Bit	**R/W**	**Description**
31: 0	RW	**SID:SVID**

Offset: 60h	H2X60: AHB to PCIe RC Register 60	Init = 0x0
Bit	**R/W**	**Description**
31:20	RW	**mask 32-bit AHB direct address bit[31:20]**
15: 4	RW	**remap 32-bit AHB direct address bit[31:20]**

Offset: 64h	H2X64: AHB to PCIe RC Register 64	Init = 0x0
Bit	**R/W**	**Description**
31: 0	RW	**remap 32-bit AHB direct address bit[63:32]**

Offset: 68h	H2X68: AHB to PCIe RC Register 68	Init = 0x0
Bit	**R/W**	**Description**
31: 0	RW	**mask 32-bit AHB direct address bit[63:32]**

Offset: 70h	H2X70: AHB to PCIe RC Register 70	Init = 0x0
Bit	**R/W**	**Description**
31:20	RW	**mask 40-bit AHB direct address bit[31:20]**
15: 4	RW	**remap 40-bit AHB direct address bit[31:20]**

Offset: 74h	H2X74: AHB to PCIe RC Register 74	Init = 0x0
Bit	**R/W**	**Description**
31: 0	RW	**remap 40-bit AHB direct address bit[63:32]**

Offset: 78h	H2X78: AHB to PCIe RC Register 78	Init = 0x0
Bit	**R/W**	**Description**
31: 0	RW	**mask 40-bit AHB direct address bit[63:32]**

Offset: 7Ch	H2X7C: AHB to PCIe RC Register 7C	Init = 0x0
Bit	**R/W**	**Description**
31: 4	RW	**Reserved**
3: 0	RW	**Direct address timeout selection bit[3:0]**

Offset: 80h	H2X80: AHB to PCIe RC Register 80	Init = 0x0
Bit	**R/W**	**Description**
31: 9	RW	**Reserved**
23:16	RW	**RX TAG**
15:10	RW	**Reserved**
9	RW	**RX DMA enable**
8	RW	**RX linear address selection (31)**
7	RW	**RX MSI selection (0)**
6	RW	**Enable RX MSI**
5	RW	**Enable legacy 1M address**
4	RW	**Unlock RX buffer**
3	RW	**RX TLP with TAG matched H2X80[23:16] only**
2	RW	**Wait until RX TLP cleaned by FW**
1	RW	**Enable PCIe RC-L RX**
0	RW	**Enable PCIe RC-L**

Offset: 84h	H2X84: AHB to PCIe RC Register 84	Init = 0x0
Bit	**R/W**	**Description**
31: 7	RW	**Reserved**
6	RW	**Enable RC-L RX IntCplCA**
5	RW	**Enable RC-L RX IntCplUR**
4	RW	**Enable RC-L RX IntDone**
3	RW	**Enable RC-L IntD**
2	RW	**Enable RC-L IntC**
1	RW	**Enable RC-L IntB**
0	RW	**Enable RC-L IntA**

Offset: 88h	H2X88: AHB to PCIe RC Register 88	Init = 0x0
Bit	**R/W**	**Description**
31: 7	RW	**Reserved**
6	RW	**RC-L RX IntDirCplCA status**
5	RW	**RC-L RX IntDirCplUR status**
4	RW	**RC-L RX IntDone status**
3	RO	**RC-L IntD status**
2	RO	**RC-L IntC status**
1	RO	**RC-L IntB status**
0	RO	**RC-L IntA status**

Offset: 8Ch	H2X8C: AHB to PCIe RC Register 8C	Init = 0x0
Bit	**R/W**	**Description**
31: 0	RO	**Read data of RC-L RX**

Offset: 90h	H2X90: AHB to PCIe RC Register 90	Init = 0x0
Bit	**R/W**	**Description**
31: 0	RO	**RC-L RX descriptor[127:96]**

Offset: 94h	H2X94: AHB to PCIe RC Register 94	Init = 0x0
Bit	**R/W**	**Description**
31: 0	RO	**RC-L RX descriptor[95:64]**

Offset: 98h	H2X98: AHB to PCIe RC Register 98	Init = 0x0
Bit	**R/W**	**Description**
31: 0	RO	**RC-L RX descriptor[63:32]**

Offset: 9Ch	H2X9C: AHB to PCIe RC Register 9C	Init = 0x0
Bit	**R/W**	**Description**
31: 0	RO	**RC-L RX descriptor[31:0]**

Offset: A0h	H2XA0: AHB to PCIe RC Register A0	Init = 0x0
Bit	**R/W**	**Description**
31: 0	RW	**Enable MSI bit[31:0]**

Offset: A4h	H2XA4: AHB to PCIe RC Register A4	Init = 0x0
Bit	**R/W**	**Description**
31: 0	RW	**Enable MSI bit[63:32]**

Offset: A8h	H2XA8: AHB to PCIe RC Register A8	Init = 0x0
Bit	**R/W**	**Description**
31: 0	RW	**MSI status bit[31:0]**

Offset: ACh	H2XAC: AHB to PCIe RC Register AC	Init = 0x0
Bit	**R/W**	**Description**
31: 0	RW	**MSI status bit[63:32]**

Offset: BCh	H2XBC: AHB to PCIe RC Register BC	Init = 0x0
Bit	**R/W**	**Description**
31: 8	RW	**Reserved**
7: 0	RW	**RC-L direct TX TAG**

Offset: C0h	H2XC0: AHB to PCIe RC Register C0	Init = 0x0
Bit	**R/W**	**Description**
31: 9	RW	**Reserved**

to next page

from previous page

23:16	RW	**RX TAG**
15:10	RW	**Reserved**
9	RW	**RX DMA enable**
8	RW	**RX linear address selection (31)**
7	RW	**RX MSI selection (0)**
6	RW	**Enable RX MSI**
5	RW	**Enable legacy 1M address**
4	RW	**Unlock RX buffer**
3	RW	**RX TLP with TAG matched H2X80[23:16] only**
2	RW	**Wait until RX TLP cleaned by FW**
1	RW	**Enable PCIe RC-H RX**
0	RW	**Enable PCIe RC-H**

Offset: C4h **H2XC4: AHB to PCIe RC Register C4** **Init = 0x0**

Bit	R/W	Description
31: 7	RW	**Reserved**
6	RW	**Enable RC-H RX IntCplCA**
5	RW	**Enable RC-H RX IntCplUR**
4	RW	**Enable RC-H RX IntDone**
3	RW	**Enable RC-H IntD**
2	RW	**Enable RC-H IntC**
1	RW	**Enable RC-H IntB**
0	RW	**Enable RC-H IntA**

Offset: C8h **H2XC8: AHB to PCIe RC Register C8** **Init = 0x0**

Bit	R/W	Description
31: 7	RW	**Reserved**
6	RW	**RC-H RX IntDirCplCA status**
5	RW	**RC-H RX IntDirCplUR status**
4	RW	**RC-H RX IntDone status**
3	RO	**RC-H IntD status**
2	RO	**RC-H IntC status**
1	RO	**RC-H IntB status**
0	RO	**RC-H IntA status**

Offset: CCh **H2XCC: AHB to PCIe RC Register CC** **Init = 0x0**

Bit	R/W	Description
31: 0	RO	**Read data of RC-H RX**

Offset: D0h	H2XD0: AHB to PCIe RC Register D0	Init = 0x0
Bit	**R/W**	**Description**
31: 0	RO	**RC-H RX descriptor[127:96]**

Offset: D4h	H2XD4: AHB to PCIe RC Register D4	Init = 0x0
Bit	**R/W**	**Description**
31: 0	RO	**RC-H RX descriptor[95:64]**

Offset: D8h	H2XD8: AHB to PCIe RC Register D8	Init = 0x0
Bit	**R/W**	**Description**
31: 0	RO	**RC-H RX descriptor[63:32]**

Offset: DCh	H2XDC: AHB to PCIe RC Register DC	Init = 0x0
Bit	**R/W**	**Description**
31: 0	RO	**RC-H RX descriptor[31:0]**

Offset: E0h	H2XE0: AHB to PCIe RC Register E0	Init = 0x0
Bit	**R/W**	**Description**
31: 0	RW	**Enable MSI bit[31:0]**

Offset: E4h	H2XE4: AHB to PCIe RC Register E4	Init = 0x0
Bit	**R/W**	**Description**
31: 0	RW	**Enable MSI bit[63:32]**

Offset: E8h	H2XE8: AHB to PCIe RC Register E8	Init = 0x0
Bit	**R/W**	**Description**
31: 0	RW	**MSI status bit[31:0]**

Offset: ECh	H2XEC: AHB to PCIe RC Register EC	Init = 0x0
Bit	**R/W**	**Description**
31: 0	RW	**MSI status bit[63:32]**

Offset: FCh	H2XFC: AHB to PCIe RC Register FC	Init = 0x0
Bit	**R/W**	**Description**
31: 8	RW	**Reserved**
7: 0	RW	**RC-H direct TX TAG**

33 eSPI Controller (ESPI)

33.1 Overview

Enhanced Serial Peripheral Interface (eSPI) is an interface using pins of SPI, but runs different protocol.
It interface supports peripheral, virtual wire, out-of-band, and flash sharing channels.
This controller supports all 4 channels and operates at max frequency of 66MHz.
It also supports up to quad-io mode.

ESPI000: Engine Control.
ESPI004: Engine Status.
ESPI008: Interrupt Status.
ESPI00C: Interrupt Enable.
ESPI010: DMA Address of Peripheral Channel Posted Rx Packet.
ESPI014: Control of Peripheral Channel Posted Rx Packet.
ESPI018: Data port of Peripheral Channel Posted Rx Packet.
ESPI020: DMA Address of Peripheral Channel Posted Tx Packet.
ESPI024: Control of Peripheral Channel Posted Tx Packet.
ESPI028: Data port of Peripheral Channel Posted Tx Packet.
ESPI030: DMA Address of Peripheral Channel Non-Posted Tx Packet.
ESPI034: Control of Peripheral Channel Non-Posted Tx Packet.
ESPI038: Data port of Peripheral Channel Non-Posted Tx Packet.
ESPI040: DMA Address of OOB Channel Rx Packet.
ESPI044: Control of OOB Channel Rx Packet.
ESPI048: Data port of OOB Channel Rx Packet.
ESPI050: DMA Address of OOB Channel Tx Packet.
ESPI054: Control of OOB Channel Tx Packet.
ESPI058: Data port of OOB Channel Tx Packet.
ESPI060: DMA Address of Flash Channel Rx Packet.
ESPI064: Control of Flash Channel Rx Packet.
ESPI068: Data port of Flash Channel Rx Packet.
ESPI070: DMA Address of Flash Channel Tx Packet.
ESPI074: Control of Flash Channel Tx Packet.
ESPI078: Data port of Flash Channel Tx Packet.
ESPI080: Engine Control 2.
ESPI084: Mapping Source Address of Peripheral Channel Rx Packet.
ESPI088: Mapping Target Address of Peripheral Channel Rx Packet.
ESPI08C: Mapping Address Mask of Peripheral Channel Rx Packet.
ESPI090: Mapping Target Address and Mask of Flash Channel.
ESPI094: Interrupt enable of System Event from Master.
ESPI098: System Event from and to Master.
ESPI09C: GPIO through Virtual Wire Channel.
ESPI0A0: General Capabilities and Configurations.
ESPI0A4: Channel 0 Capabilities and Configurations.
ESPI0A8: Channel 1 Capabilities and Configurations.
ESPI0AC: Channel 2 Capabilities and Configurations.
ESPI0B0: Channel 3 Capabilities and Configurations.
ESPI0B4: Channel 3 Capabilities and Configurations 2.
ESPI0C0: GPIO Direction of Virtual Wire Channel.
ESPI0C4: GPIO Selection of Virtual Wire Channel.
ESPI0C8: GPIO Reset Selection of Virtual Wire Channel.
ESPI0CC: Virtual Wire Channel GPIO Control.
ESPI0D0: GP50 Direction of Virtual Wire Channel 0.
ESPI0D4: GP50 Direction of Virtual Wire Channel 1.
ESPI0D8: GP50 Value of Virtual Wire Channel 0.
ESPI0DC: GP50 Value of Virtual Wire Channel 1.

ESPI0F0: Channel 3 Capabilities Control.
ESPI0F4: Virtual Wire Channel SW Interrupt.
ESPI0FC: Interrupt Enable Clear.
ESPI100: Interrupt enable of System Event 1 from Master.
ESPI104: System Event 1 from and to Master.
ESPI110: Interrupt type 0 of System Event from Master.
ESPI114: Interrupt type 1 of System Event from Master.
ESPI118: Interrupt type 2 of System Event from Master.
ESPI11C: Interrupt status of System Event from Master.
ESPI120: Interrupt type 0 of System Event 1 from Master.
ESPI124: Interrupt type 1 of System Event 1 from Master.
ESPI128: Interrupt type 2 of System Event 1 from Master.
ESPI12C: Interrupt status of System Event 1 from Master.
ESPI130: OOB Channel RX DMA Descriptor End Pointer.
ESPI134: OOB Channel RX DMA Descriptor Read Pointer.
ESPI138: OOB Channel RX DMA Descriptor Write Pointer.
ESPI140: OOB Channel TX DMA Descriptor End Pointer.
ESPI144: OOB Channel TX DMA Descriptor Read Pointer.
ESPI148: OOB Channel TX DMA Descriptor Write Pointer.
ESPI170: OOB Channel RX DMA Timeout Control.
ESPI180: Master owner Register 0.
ESPI184: Master owner Register 0.
ESPI188: Master owner Register 0.
ESPI1B0: Interrupt owner Register.

Following registers are only accessible from eSPI master.

ESPICFG004 : Device Identification.
ESPICFG008 : General Capabilities and Configurations.
ESPICFG010 : Channel 0 Capabilities and Configurations.
ESPICFG020 : Channel 1 Capabilities and Configurations.
ESPICFG030 : Channel 2 Capabilities and Configurations.
ESPICFG040 : Channel 3 Capabilities and Configurations.
ESPICFG044 : Channel 3 Capabilities and Configurations 2.
ESPICFG800 : GPIO Direction of Virtual Wire Channel.
ESPICFG804 : GPIO Selection of Virtual Wire Channel.
ESPICFG808 : GPIO Reset Selection of Virtual Wire Channel.
ESPICFG810 : Mapping Source Address of Peripheral Channel Rx Packet.
ESPICFG814 : Mapping Target Address of Peripheral Channel Rx Packet.
ESPICFG818 : Mapping Address Mask of Peripheral Channel Rx Packet.
ESPICFG820 : GP50 Direction of Virtual Wire Channel 0.
ESPICFG824 : GP50 Direction of Virtual Wire Channel 1.

33.2 Features

- Supports 66, 50, 33, 25, and 20MHz of eSPI clock frequency.
- Supports Quad-, Dual-, and Single-IO mode.
- Supports both alert mode: shared IO or dedicated.
- Supports 4 channels.
- Peripheral Channel:
  - Maximum payload size is 64 bytes.
- Virtual Wire Channel:

- Maximum virtual wire count is 8.
- Support 32 interrupts. 16 of them are controlled by BMC FW.
- Support 32 GPIOs (idx: 0x80 0x87). These GPIOs can be connected to physical pins or software mode.
- Support 64 GP50s (idx: 0x50 0x5f). These indexes are for controlibility of GPIOs in PCH.

– Out-Of-Band Channel:

- Maximum payload size is 64 bytes.
- Support descriptor type DMA.

– Run-time Flash Sharing Channel:

- Maximum payload size is 64 bytes.
- Supports both master- and slave-attached flash sharing.
- For SAFS, it supports 3 modes: HW mode, Mix mode, or SW mode. And the flash it goes to is CS0# of SPI1 interface.

33.3 Registers : Base Address = 0x1E6E_E000

Offset: 000h	ESPI000: Engine Control	Init = 0xff00b800
Bit	**R/W**	**Description**
31	RW	**Flash Tx SW Reset** 0: Reset. This bit is to reset Flash Channel Tx queue.
30	RW	**Flash Rx SW Reset** 0: Reset. This bit is to reset Flash Channel Rx queue.
29	RW	**OOB Tx SW Reset** 0: Reset. This bit is to reset OOB Channel Tx queue.
28	RW	**OOB Rx SW Reset** 0: Reset. This bit is to reset OOB Channel Rx queue.
27	RW	**Non-Posted Tx SW Reset** 0: Reset. This bit is to reset Peripheral Channel Non-Posted Tx queue.
26	RW	**Non-Posted Rx SW Reset** 0: Reset. This bit is to reset Peripheral Channel Non-Posted Rx queue.
25	RW	**Posted Tx SW Reset** 0: Reset. This bit is to reset Peripheral Channel Posted Tx queue.
24	RW	**Posted Rx SW Reset** 0: Reset. This bit is to reset Peripheral Channel Posted Rx queue.
23	RW	**Flash Channel Tx DMA Enable**
22	RW	**Flash Channel Rx DMA Enable**
21	RW	**OOB Channel Tx DMA Enable**
20	RW	**OOB Channel Rx DMA Enable**
19	RW	**Peripheral Channel Non-Posted Tx DMA Enable**

to next page

from previous page

18	RO	**Reserved**
17	RW	**Peripheral Channel Posted Tx DMA Enable**
16	RW	**Peripheral Channel Posted Rx DMA Enable**
15	RW	**Software reset for SAFS HW mode. Active low.**
14	RO	**Reserved**
13	RW	**Direction of eSPI_Reset#** 1b: Input. 0b: Output.
12	RW	**Value of eSPI_Reset#**
11:10	RW	**Software Mode of Flash Read operation** 11 : Do NOT support 10 : Hardware Mode. All read/write/erase commands are processed by HW. 01 : Software Mode. All read/write/erase commands are processed by SW. 00 : Mix Mode. Read commands are processed by HW. WriteErase commands are processed by SW.
9	RW	**Software Mode of GPIO through Virturl Wire Channel.**
8	RO	**Reserved**
7	RW	**Flash Channel Software Ready.** This bit must be set to Tx/Rx Flash Channel Write/Erase cycles.
6	RO	**Flash Channel Ready.**
5	RO	**Reserved**
4	RW	**OOB Channel Ready.** This bit must be set to Tx/Rx OOB Channel. OOB_FREE will set automatically. And it will clear when a OOB packet is received and set again when the packet is serviced.
3	RW	**Virtual Wire Channel Software Ready.** This bit must be set to Tx/Rx Virtual Wire Channel.
2	RO	**Virtual Wire Channel Ready.**
1	RW	**Peripheral Channel Software Ready.** This bit must be set to Tx/Rx Peripheral Channel Massage cycle.
0	RO	**Peripheral Channel Ready.**

Note :
Default value of ESPI000[4] is 0. When this bit is 0 the OOB channel is not ready to process incoming packet and the status response will not say OOB_FREE. However we observed that Intel PCH was expecting the OOB_FREE status right after OOB channel is enabled which is before BMC FW is ready. This needs to be programmed to 1 by the BMC FW in order to process OOB Channel correctly. OOB_FREE will be set automatically in A3.

Offset: 004h **ESPI004: Engine Status** **Init = 0x00001001**

Bit	R/W	Description
31:28	RO	**Command Abort Counter**
27:24	RO	**Reserved**
23	RO	**Flash Channel Tx Busy**
22	RO	**Flash Channel Rx Busy**
21	RO	**OOB Channel Tx Busy**
20	RO	**OOB Channel Rx Busy**
19	RO	**Peripheral Channel Non-Posted Tx Busy**

to next page

from previous page

18	RO	**Peripheral Channel Non-Posted Rx Busy**
17	RO	**Peripheral Channel Posted Tx Busy**
16	RO	**Peripheral Channel Posted Rx Busy**
15	RO	**Pin status of ESPI_RSTN**
14:12	RO	**Erase Size of Flash Channel** 000b: Reserved. 001b: 4KB. 010b: 64KB. 011b: Both 4KB and 64KB. 100b: 128KB. 101b: 256KB. 110b: Reserved. 111b: Reserved.
11:10	RO	**Reserved**
9	RO	**Virtual Wire Channel Tx Unlocked** Tx is unlocked after receiving any Rx.
8	RO	**Peripheral Channel Bus Master Enable**
7	RO	**Flash Channel Enable**
6	RO	**OOB Channel Enable**
5	RO	**Virtual Wire Channel Enable**
4	RO	**Peripheral Channel Enable**
3: 1	RO	**Reserved**
0	RO	**eSPI Controller Enable**

Offset: 008h **ESPI008: Interrupt Status** **Init = 0**

Bit	R/W	Description
31	RW	**Hardware Reset Event** Set when Hardware Reset de-asserted and asserted. Write 1 Clear
30:24	RO	**Reserved**
23	RW	**OOB RX DMA Timeout** Set when OOB RX DMA Timeout. Write 1 Clear
22	RW	**Virtual Wire System Event 1** Set when System Event 1 from Master is trigged. Write 1 Clear
21	RW	**Flash Channel Tx Error** Set when attemp Flash Channel Tx with Flash Channel disabled. Write 1 Clear
20	RW	**OOB Channel Tx Error** Set when attemp OOB Channel Tx with OOB Channel disabled. Write 1 Clear
19	RW	**Flash Channel Tx Abort** Set when Flash Channel Tx is aborted. Write 1 Clear
18	RW	**OOB Channel Tx Abort** Set when OOB Channel Tx is aborted. Write 1 Clear

to next page

from previous page

17	RW	**Peripheral Channel Non-Posted Tx Abort** Set when Peripheral Channel Non-Posted Tx is aborted. Write 1 Clear
16	RW	**Peripheral Channel Posted Tx Abort** Set when Peripheral Channel Posted Tx is aborted. Write 1 Clear
15	RW	**Flash Channel Rx Abort** Set when Flash Channel Rx is aborted. Write 1 Clear
14	RW	**OOB Channel Rx Abort** Set when OOB Channel Rx is aborted. Write 1 Clear
13	RW	**Peripheral Channel Non-Posted Rx Abort** Set when Peripheral Channel Non-Posted Rx is aborted. Write 1 Clear
12	RW	**Peripheral Channel Posted Rx Abort** Set when Peripheral Channel Posted Rx is aborted. Write 1 Clear
11	RW	**Peripheral Channel Non-Posted Tx Error** Set when attemp Peripheral Channel Non-Posted Tx with Peripheral Channel Bus Master disabled. Write 1 Clear
10	RW	**Peripheral Channel Posted Tx Error** Set when attemp Peripheral Channel Posted Tx with Peripheral Channel Bus Master disabled. Write 1 Clear
9	RW	**Virtual Wire GPIO Event** Set when input GPIO changing. Write 1 Clear
8	RW	**Virtual Wire System Event** Set when System Event from Master is trigged. Write 1 Clear
7	RW	**Flash Channel Tx Complete** Set when Flash Channel Tx Complete. Write 1 Clear
6	RW	**Flash Channel Rx Complete** Set when Flash Channel Rx Complete. Write 1 Clear
5	RW	**OOB Channel Tx Complete** Set when OOB Channel Tx Complete. Write 1 Clear
4	RW	**OOB Channel Rx Complete** Set when OOB Channel Rx Complete. Write 1 Clear
3	RW	**Peripheral Channel Non-Posted Tx Complete** Set when Peripheral Channel Non-Posted Tx Complete. Write 1 Clear
2	RO	**Reserved**

to next page

from previous page

1	RW	**Peripheral Channel Posted Tx Complete** Set when Peripheral Channel Posted Tx Complete. Write 1 Clear
0	RW	**Peripheral Channel Posted Rx Complete** Set when Peripheral Channel Posted Rx Complete. Write 1 Clear

Offset: 00Ch **ESPI00C: Interrupt Enable** **Init = 0**

Bit	R/W	Description
31	W1S	**Enable of Hardware Reset Event**
30:24	RO	**Reserved**
23	W1S	**Enable of OOB RX DMA Timeout**
22	W1S	**Enable of Virtual Wire System Event 1**
21	W1S	**Enable of Flash Channel Tx Error**
20	W1S	**Enable of OOB Channel Tx Error**
19	W1S	**Enable of Flash Channel Tx Abort**
18	W1S	**Enable of OOB Channel Tx Abort**
17	W1S	**Enable of Peripheral Channel Non-Posted Tx Abort**
16	W1S	**Enable of Peripheral Channel Posted Tx Abort**
15	W1S	**Enable of Flash Channel Rx Abort**
14	W1S	**Enable of OOB Channel Rx Abort**
13	W1S	**Enable of Peripheral Channel Non-Posted Rx Abort**
12	W1S	**Enable of Peripheral Channel Posted Rx Abort**
11	W1S	**Enable of Peripheral Channel Non-Posted Tx Error**
10	W1S	**Enable of Peripheral Channel Posted Tx Error**
9	W1S	**Enable of Virtual Wire GPIO Event**
8	W1S	**Enable of Virtual Wire System Event**
7	W1S	**Enable of Flash Channel Tx Complete**
6	W1S	**Enable of Flash Channel Rx Complete**
5	W1S	**Enable of OOB Channel Tx Complete**
4	W1S	**Enable of OOB Channel Rx Complete**
3	W1S	**Enable of Peripheral Channel Non-Posted Tx Complete**
2	RO	**Reserved**
1	W1S	**Enable of Peripheral Channel Posted Tx Complete**
0	W1S	**Enable of Peripheral Channel Posted Rx Complete**

Note :
ESPI00C is W1S. ESPI0FC is W1C.

Offset: 010h **ESPI010: DMA Address of Peripheral Channel Posted Rx Packet** **Init = 0**

Bit	R/W	Description
31: 2	RW	**DMA Address of Peripheral Channel Posted Rx Packet**
1: 0	RO	**Reserved**

Offset: 014h	ESPI014: Control of Peripheral Channel Posted Rx Packet	Init = 0
Bit	**R/W**	**Description**
31	RW	**Pending and Serviced** Read 1 : There is one receiving packet waiting to be serviced. Write 1 : Current packet is serviced.
30:24	RO	**Reserved**
23:12	RO	**Length**
11: 8	RO	**Tag**
7: 0	RO	**Cycle Type**

Offset: 018h	ESPI018: Data port of Peripheral Channel Posted Rx Packet	Init = 0
Bit	**R/W**	**Description**
31: 8	RO	**Reserved**
7: 0	RO	**Data**

Offset: 020h	ESPI020: DMA Address of Peripheral Channel Posted Tx Packet	Init = 0
Bit	**R/W**	**Description**
31: 2	RW	**DMA Address of Peripheral Channel Posted Tx Packet**
1: 0	RO	**Reserved**

Offset: 024h	ESPI024: Control of Peripheral Channel Posted Tx Packet	Init = 0
Bit	**R/W**	**Description**
31	RW	**Trigger** Write 1 : Trigger transaction. Read 1 : The TX packet is waiting eSPI master to get. Auto-clear when the packet is collected by eSPI master.
30:24	RO	**Reserved**
23:12	RW	**Length**
11: 8	RW	**Tag**
7: 0	RW	**Cycle Type** 0x01 : Memory Write 32. 0x03 : Memory Write 64. 0x06 : Successful Completion without Data. 0x0B : First Successful Completion with Data. 0x09 : Middle Successful Completion with Data. 0x0D : Last Successful Completion with Data. 0x0F : The only one successful Completion with Data. 0x40 : Message. 0x41 : Message with Data. Others are forbidden.

Offset: 028h	ESPI028: Data port of Peripheral Channel Posted Tx Packet	Init = 0
Bit	**R/W**	**Description**
31: 8	-	**Reserved**
7: 0	WO	**Data**

Offset: 030h	ESPI030: DMA Address of Peripheral Channel Non-Posted Tx Packet	Init = 0
Bit	**R/W**	**Description**
31: 2	RW	**DMA Address of Peripheral Channel Non-Posted Tx Packet**
1: 0	RO	**Reserved**

Offset: 034h	ESPI034: Control of Peripheral Channel Non-Posted Tx Packet	Init = 0
Bit	**R/W**	**Description**
31	RW	**Trigger** Write 1 : Trigger transaction. Read 1 : The TX packet is waiting eSPI master to get. Auto-clear when the packet is collected by eSPI master.
30:24	RO	**Reserved**
23:12	RW	**Length**
11: 8	RW	**Tag**
7: 0	RW	**Cycle Type** 0x00 : Memory Read 32. 0x02 : Memory Read 64.

Offset: 038h	ESPI038: Data port of Peripheral Channel Non-Posted Tx Packet	Init = 0
Bit	**R/W**	**Description**
31: 8	-	**Reserved**
7: 0	WO	**Data**

Offset: 040h	ESPI040: DMA Address of OOB Channel Rx Packet	Init = 0
Bit	**R/W**	**Description**
31: 2	RW	**DMA Address of OOB Channel Rx Packet**
1: 0	RO	**Reserved**

Offset: 044h	ESPI044: Control of OOB Channel Rx Packet	Init = 0
Bit	**R/W**	**Description**
31	RW	**Pending and Serviced** Read 1 : There is one receiving packet waiting to be serviced. Write 1 : Current packet is serviced.
30:24	RO	**Reserved**
23:12	RO	**Length**
11: 8	RO	**Tag**
7: 0	RO	**Cycle Type**

Offset: 048h	ESPI048: Data port of OOB Channel Rx Packet	Init = 0
Bit	**R/W**	**Description**
31: 8	RO	**Reserved**
7: 0	RO	**Data**

Offset: 050h	ESPI050: DMA Address of OOB Channel Tx Packet	Init = 0

Bit	R/W	Description
31: 2	RW	**DMA Address of OOB Channel Tx Packet**
1: 0	RO	**Reserved**

Offset: 054h	ESPI054: Control of OOB Channel Tx Packet	Init = 0

Bit	R/W	Description
31	W1T	**Trigger** Write 1 : Trigger transaction. Read 1 : The TX packet is waiting eSPI master to get. Auto-clear when the packet is collected by eSPI master.
30:24	RO	**Reserved**
30:24	RO	**Reserved**
23:12	RW	**Length**
11: 8	RW	**Tag**
7: 0	RW	**Cycle Type** 0x21 : OOB others are forbidden.

Offset: 058h	ESPI058: Data port of OOB Channel Tx Packet	Init = 0

Bit	R/W	Description
31: 8	-	**Reserved**
7: 0	WO	**Data**

Offset: 060h	ESPI060: DMA Address of Flash Channel Rx Packet	Init = 0

Bit	R/W	Description
31: 2	RW	**DMA Address of Flash Channel Rx Packet**
1: 0	RO	**Reserved**

Offset: 064h	ESPI064: Control of Flash Channel Rx Packet	Init = 0

Bit	R/W	Description
31	RW	**Serviced** Read 1 : There is one receiving packet waiting to be serviced. Write 1 : Current packet is serviced.
30:24	RO	**Reserved**
23:12	RO	**Length**
11: 8	RO	**Tag**
7: 0	RO	**Cycle Type**

Offset: 068h	ESPI068: Data port of Flash Channel Rx Packet	Init = 0

Bit	R/W	Description
31: 8	RO	**Reserved**
7: 0	RO	**Data**

Offset: 070h	ESPI070: DMA Address of Flash Channel Tx Packet	Init = 0

Bit	R/W	Description
31: 2	RW	**DMA Address of Flash Channel Tx Packet**
1: 0	RO	**Reserved**

Offset: 074h	ESPI074: Control of Flash Channel Tx Packet	Init = 0

Bit	R/W	Description
31	RW	**Trigger** Write 1 : Trigger transaction. Read 1 : The TX packet is waiting eSPI master to get. Auto-clear when the packet is collected by eSPI master.
30:24	RO	**Reserved**
23:12	RW	**Length**
11: 8	RW	**Tag**
7: 0	RW	**Cycle Type** 0x00 : Flash Read. 0x01 : Flash Write. 0x02 : Flash Erase. 0x06 : Successful Completion without Data. 0x0B : First Successful Completion with Data. 0x09 : Middle Successful Completion with Data. 0x0C : Last Unsuccessful Completion without Data. 0x0D : Last Successful Completion with Data. 0x0E : The only one Unsuccessful Completion without Data. 0x0F : The only one successful Completion with Data. others are forbidden.

Offset: 078h	ESPI078: Data port of Flash Channel Tx Packet	Init = 0

Bit	R/W	Description
31: 8	-	**Reserved**
7: 0	WO	**Data**

Offset: 080h	ESPI080: Engine Control 2	Init = 0x40000f57

Bit	R/W	Description
31	RW	**LPC Reset Source** 1: eSPI Reset Pin 0: Platform Reset from VW
30	RW	**VW TX Sort**
29	RW	**Reserved**
28	RW	**New Interrupt Mode**
27:20	RW	**eSPI_Reset# counter**
19:12	RW	**Reserved**
11	RW	**Disable Peripheral Channel Memory Read Cycle after WDT reset**
10	RW	**Disable Peripheral Channel Memory Write Cycle after WDT reset**
9	RW	**Enable automatic acknowledge of Slave Boot Done**
8	RW	**Enable automatic acknowledge of Slave Boot Sts**

to next page

from previous page

7	RW1S	**Enable write protection of ESPI080 bit 6** Make ESPI080 bit 6 not writable until RstFull.
6	RW	**Disable Peripheral Channel Memory Read Cycle.**
5	RW1S	**Enable write protection of ESPI080 bit 4** Make ESPI080 bit 4 not writable until RstFull.
4	RW	**Disable Peripheral Channel Memory Write Cycle.**
3	RO	**Reserved**
2	RW	**Enable automatic acknowledge of Host Reset Warn**
1	RW	**Enable automatic acknowledge of OOB Reset Warn**
0	RW	**Enable automatic acknowledge of Suspend Warn**

Offset: 084h ESPI084: Mapping Source Address of Peripheral Channel Rx Packet Init = 0

Bit	R/W	Description
31: 0	RW	**Mapping Source Address of Peripheral Channel Rx Packet**

Offset: 088h ESPI088: Mapping Target Address of Peripheral Channel Rx Packet Init = 0

Bit	R/W	Description
31: 0	RW	**Mapping Target Address of Peripheral Channel Rx Packet**

Offset: 08Ch ESPI08C: Mapping Address Mask of Peripheral Channel Rx Packet Init = 0xffff0001

Bit	R/W	Description
31:16	RW	**Mapping Address Mask of Peripheral Channel Rx Packet**
15: 2	RO	**Reserved**
1	RW1S	**Enable write protection of ESPI084, ESPI088, and ESPI08C** Make ESPI084, ESPI088, and ESPI08C not writable until RstFull.
0	RW	**Enable write protection of ESPICFG810, ESPICFG814, and ESPICFG818** Make ESPICFG810, ESPICFG814, and ESPICFG818 not writable until RstFull.

Offset: 090h ESPI090: Mapping Target Address and Mask of Flash Channel Init = 0x3000ff00

Bit	R/W	Description
31:24	RW	**Mapping Target Address of Flash Channel**
23:16	RO	**Reserved**
15: 8	RW	**Mapping Mask of Flash Channel**
9: 1	RO	**Reserved**
0	RW	**Enable write protection of ESPI090** Make ESPI090 not writable until PWRSTN.

Offset: 094h ESPI094: Interrupt enable of System Event from Master Init = 0

Bit	R/W	Description
31:11	RO	**Reserved**
10	RW	**Enable Interrupt of NMI Out**
9	RW	**Enable Interrupt of SMI Out**

to next page

from previous page

8	RW	**Enable Interrupt of Host Reset Warn**
7	RO	**Reserved**
6	RW	**Enable Interrupt of OOB Reset Warn**
5	RW	**Enable Interrupt of PLTRSTN**
4	RW	**Enable Interrupt of Suspend Status**
3	RO	**Reserved**
2	RW	**Enable Interrupt of S5 Sleep Control**
1	RW	**Enable Interrupt of S4 Sleep Control**
0	RW	**Enable Interrupt of S3 Sleep Control**

Offset: 098h **ESPI098: System Event from and to Master** **Init = 0x070c0600**

Bit	R/W	Description
31:28	RO	**Reserved**
27	RW	**Host Reset Acknowledge**
26	RW	**Reset CPU Init#**
25:24	RO	**Reserved**
23	RW	**Slave Boot Status**
22	RW	**Non-Fatal Error**
21	RW	**Fatal Error**
20	RW	**Slave Boot Done**
19:17	RO	**Reserved**
16	RW	**OOB Reset Acknowledge**
15:11	RO	**Reserved**
10	RO	**NMI Out**
9	RO	**SMI Out**
8	RO	**Host Reset Warn**
7	RO	**Reserved**
6	RO	**OOB Reset Warn**
5	RO	**PLTRSTN**
4	RO	**Suspend Status**
3	RO	**Reserved**
2	RO	**S5 Sleep Control**
1	RO	**S4 Sleep Control**
0	RO	**S3 Sleep Control**

Offset: 09Ch **ESPI09C: GPIO through Virtual Wire Channel** **Init = 0**

Bit	R/W	Description
31: 0	RW	**GPIO through Virtual Wire Channel**

Offset: 0A0h	ESPI0A0: General Capabilities and Configurations	Init = 0x030c000f

Bit	R/W	Description
31	RO	**CRC Check Enable**
30	RO	**Response Modifier Enable**
29	RO	**Reserved**
28	RO	**Alert Mode**
27:26	RO	**IO Mode Select** 00b Single IO. 01b Dual IO. 10b Quad IO. 11b Reserved
25:24	RO	**I/O Mode Support.** 00b: Single I/O. 01b: Single and Dual I/O. 10b: Single and Quad I/O. 11b: Single, Dual and Quad I/O.
23	RO	**Open Drain Alert# Select** 0b: Alert# pin is a driven output. 1b: Alert# pin is an open-drain output.
22:20	RO	**Operating Frequency.** 000b: 20MHz. 001b: 25MHz. 010b: 33MHz. 011b: 50MHz. 100b: 66MHz. Others: Reserved.
19	RO	**Alert# pin type status** 1b: Open-drain Alert# pin is supported.
18:16	RO	**Maximum Frequency Support** 000b 20 MHz 001b 25 MHz 010b 33 MHz 011b 50 MHz 100b 66 MHz Others Reserved
15:12	RO	**Maximum Wait State allowed**
11: 8	RO	**Reserved**
7: 0	RO	**Channel Supported** Bit 0: Peripheral Channel. Bit 1: Virtaul Wire Channel. Bit 2: Out-Of-Band Channel. Bit 3: Flash Access Channel. Others: Reserved

Offset: 0A4h	ESPI0A4: Channel 0 Capabilities and Configurations	Init = 0x00001110

Bit	R/W	Description
31:15	RO	**Reserved**

to next page

from previous page

14:12	RO	**Peripheral Channel Maximum Read Request Size** 000b: Reserved 001b: 64 bytes address aligned max read request size. 010b: 128 bytes address aligned max read request size. 011b: 256 bytes address aligned max read request size. 100b: 512 bytes address aligned max read request size. 101b: 1024 bytes address aligned max read request size. 110b: 2048 bytes address aligned max read request size. 111b: 4096 bytes address aligned max read request size.
11	RO	**Reserved**
10: 8	RO	**Peripheral Channel Maximum Payload Size Selected** 000b: Reserved 001b: 64 bytes address aligned max payload size. 010b: 128 bytes address aligned max payload size. 011b: 256 bytes address aligned max payload size. Others: Reserved
7	RO	**Reserved**
6: 4	RO	**Peripheral Channel Maximum Payload Size Supported** 000b: Reserved 001b: 64 bytes address aligned max payload size. 010b: 128 bytes address aligned max payload size. 011b: 256 bytes address aligned max payload size. Others: Reserved
3	RO	**Reserved**
2	RO	**Bus Master Enable**
1	RO	**Peripheral Channel Ready**
0	RO	**Peripheral Channel Enable**

Offset: 0A8h **ESPI0A8: Channel 1 Capabilities and Configurations** **Init = 0x00000700**

Bit	R/W	Description
31:22	RO	**Reserved**
21:16	RO	**Operating Maximum Virtual Wire Count**
15:14	RO	**Reserved**
13: 8	RO	**Maximum Maximum Virtual Wire Count Supported**
7: 2	RO	**Reserved**
1	RO	**Virtual Wire Channel Ready**
0	RO	**Virtual Wire Channel Enable**

Offset: 0ACh **ESPI0AC: Channel 2 Capabilities and Configurations** **Init = 0x00000110**

Bit	R/W	Description
31:11	RO	**Reserved**
10: 8	RO	**OOB Message Channel Maximum Payload Size Selected** 000b: Reserved 001b: 64 bytes max payload size. 010b: 128 bytes max payload size. 011b: 256 bytes max payload size. Others: Reserved

to next page

from previous page

7	RO	**Reserved**
6: 4	RO	**OOB Message Channel Maximum Payload Size Supported** 000b: Reserved 001b: 64 bytes max payload size. 010b: 128 bytes max payload size. 011b: 256 bytes max payload size. Others: Reserved
3: 2	RO	**Reserved**
1	RO	**OOB Message Channel Ready**
0	RO	**OOB Message Channel Enable**

Offset: 0B0h ESPI0B0: Channel 3 Capabilities and Configurations Init = 0x00031124

Bit	R/W	Description
31:15	RO	**Reserved**
14:12	RO	**Flash Access Channel Maximum Read Request Size** 000b: Reserved 001b: 64 bytes address aligned max read request size. 010b: 128 bytes address aligned max read request size. 011b: 256 bytes address aligned max read request size. 100b: 512 bytes address aligned max read request size. 101b: 1024 bytes address aligned max read request size. 110b: 2048 bytes address aligned max read request size. 111b: 4096 bytes address aligned max read request size.
11	RO	**Flash Sharing Mode** 0b: Master attached flash sharing. 1b: Slave attached flash sharing.
10: 8	RO	**Flash Access Channel Maximum Payload Size Selected** 000b: Reserved 001b: 64 bytes max payload size. 010b: 128 bytes max payload size. 011b: 256 bytes max payload size. Others: Reserved
7: 5	RO	**Flash Access Channel Maximum Payload Size Supported** 000b: Reserved 001b: 64 bytes max payload size. 010b: 128 bytes max payload size. 011b: 256 bytes max payload size. Others: Reserved
4: 2	RO	**Flash Block Erase Size** 000b: Reserved 001b: 4 KBytes. 010b: 64 KBytes. 011b: Both 4 KBytes and 64 KBytes. 100b: 128 KBytes. 101b: 256 KBytes. Others: Reserved
1	RO	**Flash Access Channel Ready**
0	RO	**Flash Access Channel Enable**

Offset: 0B4h	ESPI0B4: Channel 3 Capabilities and Configurations 2	Init = 0x00004401

Bit	R/W	Description
31:22	RO	**Reserved**
21:16	RO	**Target RPMC Supported** 0h : Slave does not support RPMC. 1h : Slave supports up to 1 RPMC. 2h : Slave supports up to 2 RPMC. ... 3Fh : Slave supports up to 63 RPMC.
15: 8	RO	**Target Flash Erase Block Size for Master's Regions** Bit 0 : 1 Kbytes EBS supported Bit 1 : 2 Kbytes EBS supported Bit 2 : 4 Kbytes EBS supported Bit 3 : 8 Kbytes EBS supported Bit 4 : 16 Kbytes EBS supported Bit 5 : 32 Kbytes EBS supported Bit 6 : 64 Kbytes EBS supported Bit 7 : 128 Kbytes EBS supported
7: 3	RO	**Reserved**
2: 0	RO	**Target Maximum Read Request Size Supported** 000b: 64 bytes max read request size 001b: 64 bytes max read request size 010b: 128 bytes max read request size. 011b: 256 bytes max read request size. 100b: 512 bytes max read request size 101b: 1024 bytes max read request size 110b: 2048 bytes max read request size. 111b: 4096 bytes max read request size.

Offset: 0C0h	ESPI0C0: GPIO Direction of Virtual Wire Channel	Init = 0

Bit	R/W	Description
31: 0	RO	**GPIO Direction of Virtual Wire Channel** 0b: Input 1b: Output

Offset: 0C4h	ESPI0C4: GPIO Selection of Virtual Wire Channel	Init = 0x03020100
Bit	**R/W**	**Description**
31:24	RO	**Select GPIO group for GPIO31 - GPIO24** 0x00: GPIOA 0x01: GPIOB 0x02: GPIOC 0x03: GPIOD 0x04: GPIOE 0x05: GPIOF 0x06: GPIOG 0x07: GPIOH 0x08: GPIOI 0x09: GPIOJ 0x0A: GPIOK 0x0B: GPIOL 0x0C: GPIOM 0x0D: GPION 0x0E: GPIOO 0x0F: GPIOP 0x11: GPIOQ 0x12: GPIOR 0x13: GPIOS 0x14: GPIOT 0x15: GPIOU 0x16: GPIOV 0x17: GPIOW 0x18: GPIOX 0x19: GPIOY 0x1A: GPIOZ 0x1B: GPIOAA 0x1C: GPIOAB 0x1D: GPIOAC
23:16	RO	**Select GPIO group for GPIO23 - GPIO16**
15: 8	RO	**Select GPIO group for GPIO15 - GPIO08**
7: 0	RO	**Select GPIO group for GPIO07 - GPIO00**

Offset: 0C8h	ESPI0C8: GPIO Reset Selection of Virtual Wire Channel	Init = 0
Bit	**R/W**	**Description**
31: 0	RO	**GPIO Reset Selection of Virtual Wire Channel** 0b: Reset by eSPI_Reset# 1b: Reset by PLTRSTN

Offset: 0CCh	ESPI0CC: Virtual Wire Channel GPIO Control	Init = 0
Bit	**R/W**	**Description**
31	RW	**Selection of Read value of ESPI0D8 and ESPI0DC** 1b: Previous written value 0b: Value from eSPI master
30:18	RO	**Reserved**
17	RW	**Enable write protection of ESPI0D0 and ESPI0D4**
16	RW	**Enable write protection of ESPICFG820 and ESPICFG824**

to next page

from previous page

15: 2	RO	**Reserved**
1	RW	**Enable write protection of ESPI0C0, ESPI0C4, and ESPI0C8**
0	RW	**Enable write protection of ESPICFG800, ESPICFG804, and ESPICFG808**

Offset: 0D0h **GP50 Direction of Virtual Wire Channel 0** **Init = 0**

Bit	R/W	Description
31: 0	RW	**GP50 Direction of Virtual Wire Channel** 0b: Slave to master 1b: master to slave

Offset: 0D4h **GP50 Direction of Virtual Wire Channel 1** **Init = 0**

Bit	R/W	Description
31: 0	RW	**GP50 Direction of Virtual Wire Channel** 0b: master to slave 1b: Slave to master

Offset: 0D8h **GP50 Value of Virtual Wire Channel 0** **Init = 0**

Bit	R/W	Description
31: 0	RW	**GP50 Value of Virtual Wire Channel** Write value of slave to master. Read if ESPI0CC bit 31 is: 1b: Previous written value 0b: Value from eSPI master

Offset: 0DCh **GP50 Value of Virtual Wire Channel 1** **Init = 0**

Bit	R/W	Description
31: 0	RW	**GP50 Value of Virtual Wire Channel** Write value of slave to master. Read if ESPI0CC bit 31 is: 1b: Previous written value 0b: Value from eSPI master

Offset: 0F0h **Channel 3 Capabilities Control** **Init = 0x00004400**

Bit	R/W	Description
31:22	RO	**Reserved**
21:16	RW	**Target RPMC Supported** 0h : Slave does not support RPMC. 1h : Slave supports up to 1 RPMC. 2h : Slave supports up to 2 RPMC. ... 3Fh : Slave supports up to 63 RPMC.

to next page

from previous page

15: 8	RW	**Target Flash Erase Block Size for Master's Regions** Bit 0 : 1 Kbytes EBS supported Bit 1 : 2 Kbytes EBS supported Bit 2 : 4 Kbytes EBS supported Bit 3 : 8 Kbytes EBS supported Bit 4 : 16 Kbytes EBS supported Bit 5 : 32 Kbytes EBS supported Bit 6 : 64 Kbytes EBS supported Bit 7 : 128 Kbytes EBS supported
7: 0	RO	**Reserved**

Offset: 0F4h	**Virtual Wire Channel SW Interrupt**	**Init = 0**
Bit	**R/W**	**Description**
31:16	RW	**Interrupt source selection of IRQ# 16 31** 1b : From ESPI0F4 0b : From HW
15: 0	RW	**Interrupt of IRQ# 16 31**

Offset: 0FCh	**ESPI0FC: Interrupt Enable Clear**	**Init = 0**
Bit	**R/W**	**Description**
31	W1C	**Enable of Hardware Reset Event**
30:24	RO	**Reserved**
23	W1C	**Enable of OOB RX DMA Timeout**
22	W1C	**Enable of Virtual Wire System Event 1**
21	W1C	**Enable of Flash Channel Tx Error**
20	W1C	**Enable of OOB Channel Tx Error**
19	W1C	**Enable of Flash Channel Tx Abort**
18	W1C	**Enable of OOB Channel Tx Abort**
17	W1C	**Enable of Peripheral Channel Non-Posted Tx Abort**
16	W1C	**Enable of Peripheral Channel Posted Tx Abort**
15	W1C	**Enable of Flash Channel Rx Abort**
14	W1C	**Enable of OOB Channel Rx Abort**
13	W1C	**Enable of Peripheral Channel Non-Posted Rx Abort**
12	W1C	**Enable of Peripheral Channel Posted Rx Abort**
11	W1C	**Enable of Peripheral Channel Non-Posted Tx Error**
10	W1C	**Enable of Peripheral Channel Posted Tx Error**
9	W1C	**Enable of Virtual Wire GPIO Event**
8	W1C	**Enable of Virtual Wire System Event**
7	W1C	**Enable of Flash Channel Tx Complete**
6	W1C	**Enable of Flash Channel Rx Complete**
5	W1C	**Enable of OOB Channel Tx Complete**
4	W1C	**Enable of OOB Channel Rx Complete**
3	W1C	**Enable of Peripheral Channel Non-Posted Tx Complete**
2	RO	**Reserved**

to next page

from previous page

1	W1C	**Enable of Peripheral Channel Posted Tx Complete**
0	W1C	**Enable of Peripheral Channel Posted Rx Complete**

Note :
ESPI00C is W1S. ESPI0FC is W1C.

Offset: 100h **ESPI100: Interrupt enable of System Event 1 from Master** **Init = 0**

Bit	R/W	Description
31:17	RO	**Reserved**
16	RW	**Enable Interrput of Host C10**
15: 8	RW	**Enable Interrput of PCH Generic**
7: 6	RO	**Reserved**
5	RW	**Enable Interrput of Wireless Lan Sleep Control**
4	RW	**Enable Interrput of Lan Sleep Control**
3	RW	**Enable Interrput of A# Sleep Control**
2	RO	**Reserved**
1	RW	**Enable Interrput of Suspend PowerDown Ack**
0	RW	**Enable Interrput of Suspend Warn**

Offset: 104h **ESPI104: System Event 1 from and to Master** **Init = 0**

Bit	R/W	Description
31:24	RO	**BMC Generic**
23:21	RO	**Reserved**
20	RW	**Suspend Ack**
19:17	RO	**Reserved**
16	RO	**Host C10**
15: 8	RO	**PCH Generic**
7: 6	RO	**Reserved**
5	RO	**Wireless Lan Sleep Control**
4	RO	**Lan Sleep Control**
3	RO	**A# Sleep Control**
2	RO	**Reserved**
1	RO	**Suspend PowerDown Ack**
0	RO	**Suspend Warn**

Offset: 110h **ESPI110: Interrupt type 0 of System Event from Master** **Init = 0**

Bit	R/W	Description
31:11	RO	**Reserved**
10	RW	**Interrupt type 0 of NMI Out**
9	RW	**Interrupt type 0 of SMI Out**
8	RW	**Interrupt type 0 of Host Reset Warn**
7	RO	**Reserved**
6	RW	**Interrupt type 0 of OOB Reset Warn**

to next page

from previous page

5	RW	**Interrupt type 0 of PLTRSTN**
4	RW	**Interrupt type 0 of Suspend Status**
3	RO	**Reserved**
2	RW	**Interrupt type 0 of S5 Sleep Control**
1	RW	**Interrupt type 0 of S4 Sleep Control**
0	RW	**Interrupt type 0 of S3 Sleep Control**

Offset: 114h **ESPI114: Interrupt type 1 of System Event from Master** **Init = 0**

Bit	R/W	Description
31:11	RO	**Reserved**
10	RW	**Interrupt type 1 of NMI Out**
9	RW	**Interrupt type 1 of SMI Out**
8	RW	**Interrupt type 1 of Host Reset Warn**
7	RO	**Reserved**
6	RW	**Interrupt type 1 of OOB Reset Warn**
5	RW	**Interrupt type 1 of PLTRSTN**
4	RW	**Interrupt type 1 of Suspend Status**
3	RO	**Reserved**
2	RW	**Interrupt type 1 of S5 Sleep Control**
1	RW	**Interrupt type 1 of S4 Sleep Control**
0	RW	**Interrupt type 1 of S3 Sleep Control**

Offset: 118h **ESPI118: Interrupt type 2 of System Event from Master** **Init = 0**

Bit	R/W	Description
31:11	RO	**Reserved**
10	RW	**Interrupt type 2 of NMI Out**
9	RW	**Interrupt type 2 of SMI Out**
8	RW	**Interrupt type 2 of Host Reset Warn**
7	RO	**Reserved**
6	RW	**Interrupt type 2 of OOB Reset Warn**
5	RW	**Interrupt type 2 of PLTRSTN**
4	RW	**Interrupt type 2 of Suspend Status**
3	RO	**Reserved**
2	RW	**Interrupt type 2 of S5 Sleep Control**
1	RW	**Interrupt type 2 of S4 Sleep Control**
0	RW	**Interrupt type 2 of S3 Sleep Control**

The definition of Interrupt Type 0, 1, and 2 are as follows :

Interrupt Type #2	Interrupt Type #1	Interrupt Type #0	Type
1	X	X	Dual Edge
0	1	1	Level-High
0	1	0	Level-Low
0	0	1	Rising-Edge
0	0	0	Falling-Edge

Offset: 11Ch **ESPI11C: Interrupt status of System Event from Master** **Init = 0**

Bit	R/W	Description
31:11	RO	**Reserved**
10	RW	**Interrupt status of NMI Out** Write 1 to clear.
9	RW	**Interrupt status of SMI Out** Write 1 to clear.
8	RW	**Interrupt status of Host Reset Warn** Write 1 to clear.
7	RO	**Reserved**
6	RW	**Interrupt status of OOB Reset Warn** Write 1 to clear.
5	RW	**Interrupt status of PLTRSTN** Write 1 to clear.
4	RW	**Interrupt status of Suspend Status** Write 1 to clear.
3	RO	**Reserved**
2	RW	**Interrupt status of S5 Sleep Control** Write 1 to clear.
1	RW	**Interrupt status of S4 Sleep Control** Write 1 to clear.
0	RW	**Interrupt status of S3 Sleep Control** Write 1 to clear.

Offset: 120h **ESPI120: Interrupt type 0 of System Event 1 from Master** **Init = 0**

Bit	R/W	Description
31:17	RO	**Reserved**
16	RW	**Interrupt type 0 of Host C10**
15: 8	RW	**Interrupt type 0 of PCH Generic**
7: 6	RO	**Reserved**
5	RW	**Interrupt type 0 of Wireless Lan Sleep Control**
4	RW	**Interrupt type 0 of Lan Sleep Control**
3	RW	**Interrupt type 0 of A# Sleep Control**
2	RO	**Reserved**
1	RW	**Interrupt type 0 of Suspend PowerDown Ack**
0	RW	**Interrupt type 0 of Suspend Warn**

Offset: 124h	ESPI124: Interrupt type 1 of System Event 1 from Master	Init = 0
Bit	**R/W**	**Description**
31:17	RO	**Reserved**
16	RW	**Interrupt type 1 of Host C10**
15: 8	RW	**Interrupt type 1 of PCH Generic**
7: 6	RO	**Reserved**
5	RW	**Interrupt type 1 of Wireless Lan Sleep Control**
4	RW	**Interrupt type 1 of Lan Sleep Control**
3	RW	**Interrupt type 1 of A# Sleep Control**
2	RO	**Reserved**
1	RW	**Interrupt type 1 of Suspend PowerDown Ack**
0	RW	**Interrupt type 1 of Suspend Warn**

Offset: 128h	ESPI128: Interrupt type 2 of System Event 1 from Master	Init = 0
Bit	**R/W**	**Description**
31:17	RO	**Reserved**
16	RW	**Interrupt type 2 of Host C10**
15: 8	RW	**Interrupt type 2 of PCH Generic**
7: 6	RO	**Reserved**
5	RW	**Interrupt type 2 of Wireless Lan Sleep Control**
4	RW	**Interrupt type 2 of Lan Sleep Control**
3	RW	**Interrupt type 2 of A# Sleep Control**
2	RO	**Reserved**
1	RW	**Interrupt type 2 of Suspend PowerDown Ack**
0	RW	**Interrupt type 2 of Suspend Warn**

Offset: 12Ch	ESPI12C: Interrupt status of System Event 1 from Master	Init = 0
Bit	**R/W**	**Description**
31:17	RO	**Reserved**
16	RW	**Interrupt status of Host C10**
15: 8	RW	**Interrupt status of PCH Generic**
7: 6	RO	**Reserved**
5	RW	**Interrupt status of Wireless Lan Sleep Control**
4	RW	**Interrupt status of Lan Sleep Control**
3	RW	**Interrupt status of A# Sleep Control**
2	RO	**Reserved**
1	RW	**Interrupt status of Suspend PowerDown Ack**
0	RW	**Interrupt status of Suspend Warn**

The definition of Interrupt Type 0, 1, and 2 are as follows :

Interrupt Type #2	Interrupt Type #1	Interrupt Type #0	Type
1	X	X	Dual Edge
0	1	1	Level-High
0	1	0	Level-Low
0	0	1	Rising-Edge
0	0	0	Falling-Edge

Offset: 130h **ESPI130: OOB Channel RX DMA Descriptor End Pointer** **Init = 0**

Bit	R/W	Description
31:13	R	**Reserved**
12: 0	RW	**DMA Descriptor End Pointer**

Offset: 134h **ESPI134: OOB Channel RX DMA Descriptor Read Pointer** **Init = 0**

Bit	R/W	Description
31	RW	**DMA Descriptor Read Pointer Update** Write 1 to update DMA Descriptor Read Pointer
30:12	R	**Reserved**
11: 0	RW	**DMA Descriptor Read Pointer**

Offset: 138h **ESPI138: OOB Channel RX DMA Descriptor Write Pointer** **Init = 0**

Bit	R/W	Description
31	RW	**RX DMA Descriptor valid**
30:28	R	**Reserved**
27:16	RW	**DMA Descriptor Service Pointer** This pointer points to the end of serviced descriptor of RX.
15:12	R	**Reserved**
11: 0	RW	**DMA Descriptor Write Pointer** This pointer points to the end of prepared descriptor of RX.

Offset: 140h **ESPI140: OOB Channel TX DMA Descriptor End Pointer** **Init = 0**

Bit	R/W	Description
31:13	R	**Reserved**
12: 0	RW	**DMA Descriptor End Pointer**

Offset: 144h **ESPI144: OOB Channel TX DMA Descriptor Read Pointer** **Init = 0**

Bit	R/W	Description
31	RW	**DMA Descriptor Read Pointer Update** Write 1 to update DMA Descriptor Read Pointer
30:12	R	**Reserved**
11: 0	RW	**DMA Descriptor Read Pointer**

Offset: 148h	ESPI148: OOB Channel TX DMA Descriptor Write Pointer	Init = 0
Bit	**R/W**	**Description**
31	W	**TX DMA Descriptor valid**
30:12	R	**Reserved**
11: 0	RW	**DMA Descriptor Write Pointer**

Offset: 170h	ESPI170: OOB Channel RX DMA Timeout Control	Init = 0
Bit	**R/W**	**Description**
31	RW	**Enable Timeout timer for OOB Channel RX DMA**
30: 0	RW	**Timeout timer for OOB Channel RX DMA**

Offset: 180h	ESPI180: Master owner Register 0	Init = ffff_ffff
Bit	**R/W**	**Description**
31	RW	**Reserved**
30	RW	**Owner of ESPI078**
29	RW	**Owner of ESPI074**
28	RW	**Owner of ESPI070**
27	RW	**Reserved**
26	RW	**Owner of ESPI068**
25	RW	**Owner of ESPI064**
24	RW	**Owner of ESPI060**
23	RW	**Reserved**
22	RW	**Owner of ESPI058**
21	RW	**Owner of ESPI054**
20	RW	**Owner of ESPI050**
19	RW	**Reserved**
18	RW	**Owner of ESPI048**
17	RW	**Owner of ESPI044**
16	RW	**Owner of ESPI040**
15	RW	**Reserved**
14	RW	**Owner of ESPI038**
13	RW	**Owner of ESPI034**
12	RW	**Owner of ESPI030**
11	RW	**Reserved**
10	RW	**Owner of ESPI028**
9	RW	**Owner of ESPI024**
8	RW	**Owner of ESPI020**
7	RW	**Reserved**
6	RW	**Owner of ESPI018**
5	RW	**Owner of ESPI014**
4	RW	**Owner of ESPI010**
3	RW	**Reserved**
2	RW	**Reserved**

to next page

from previous page

1	RW	**Owner of ESPI004**
0	RW	**Owner of ESPI000** 0: CM3 only. 1: All allowed.

Offset: 184h **ESPI184: Master owner Register 1** **Init = ffff_ffff**

Bit	R/W	Description
31	RW	**Reserved**
30	RW	**Reserved**
29	RW	**Owner of ESPI0F4**
28	RW	**Owner of ESPI0F0**
27	RW	**Owner of ESPI0EC**
26	RW	**Owner of ESPI0E8**
25	RW	**Owner of ESPI0E4**
24	RW	**Owner of ESPI0E0**
23	RW	**Owner of ESPI0DC**
22	RW	**Owner of ESPI0D8**
21	RW	**Owner of ESPI0D4**
20	RW	**Owner of ESPI0D0**
19	RW	**Owner of ESPI0CC**
18	RW	**Owner of ESPI0C8**
17	RW	**Owner of ESPI0C4**
16	RW	**Owner of ESPI0C0**
15	RW	**Reserved**
14	RW	**Reserved**
13	RW	**Owner of ESPI0B4**
12	RW	**Owner of ESPI0B0**
11	RW	**Owner of ESPI0AC**
10	RW	**Owner of ESPI0A8**
9	RW	**Owner of ESPI0A4**
8	RW	**Owner of ESPI0A0**
7	RW	**Owner of ESPI09C**
6	RW	**Owner of ESPI098**
5	RW	**Owner of ESPI094**
4	RW	**Owner of ESPI090**
3	RW	**Owner of ESPI08C**
2	RW	**Owner of ESPI088**
1	RW	**Owner of ESPI084**
0	RW	**Owner of ESPI080** 0: CM3 only. 1: All allowed.

Offset: 188h	ESPI188: Master owner Register 2	Init = ffff_ffff

Bit	R/W	Description
31	RW	**Reserved**
30	RW	**Reserved**
29	RW	**Reserved**
28	RW	**Owner of ESPI170**
27	RW	**Reserved**
26	RW	**Reserved**
25	RW	**Reserved**
24	RW	**Reserved**
23	RW	**Reserved**
22	RW	**Reserved**
21	RW	**Reserved**
20	RW	**Reserved**
19	RW	**Reserved**
18	RW	**Reserved**
17	RW	**Reserved**
16	RW	**Reserved**
15	RW	**Reserved**
14	RW	**Reserved**
13	RW	**Reserved**
12	RW	**Reserved**
11	RW	**Owner of ESPI12C**
10	RW	**Owner of ESPI128**
9	RW	**Owner of ESPI124**
8	RW	**Owner of ESPI120**
7	RW	**Owner of ESPI11C**
6	RW	**Owner of ESPI118**
5	RW	**Owner of ESPI114**
4	RW	**Owner of ESPI110**
3	RW	**Owner of ESPI10C**
2	RW	**Owner of ESPI108**
1	RW	**Owner of ESPI104**
0	RW	**Owner of ESPI100** 0: CM3 only. 1: All allowed.

Offset: 1B0h	ESPI1B0: Interrupt owner Register	Init = 0x55

Bit	R/W	Description
31	RW1S	**Write Protection of ESPI1B0**
7: 6	RW	**Owner of Flash Channel Interrupt**
5: 4	RW	**Owner of OOB Channel Interrupt**
3: 2	RW	**Owner of Virtual Wire Channel Interrupt**

to next page

from previous page

1: 0	RW	**Owner of Peripheral Channel Interrupt** 10b: CM3 only. x1b: All allowed. 00b: Forbidden. Only allowed masters are able to modify ESPI008 and ESPI00C corresponding bit.

Following registers which are named as CFG are only accessible from eSPI master side.

Offset: 004h	**ESPICFG004: Device Identification**	**Init = 0000_0001**
Bit	**R/W**	**Description**
31: 8	RO	**Reserved.**
31: 0	RO	**Version ID**

Offset: 008h	**ESPICFG008: General Capabilities and Configurations**	**Init = 030c_000f**
Bit	**R/W**	**Description**
31	RW	**CRC Checking Enable.**
30	RW	**Response Modifier Enable.**
29	RO	**Reserved.**
28	RW	**Alert Mode.** 0: I/O[1] is used to signal the Alert event. 1: A dedicated Alert# pin is used to signal the Alert event.
27:26	RW	**I/O Mode Select.** 00b: Single I/O. 01b: Dual I/O. 10b: Quad I/O. 11b: Reserved.
25:24	RO	**I/O Mode Support.** 00b: Single I/O. 01b: Single and Dual I/O. 10b: Single and Quad I/O. 11b: Single, Dual and Quad I/O.
23	RW	**Open Drain Alert# Select** 0b: Alert# pin is a driven output. 1b: Alert# pin is an open-drain output.
22:20	RW	**Operating Frequency.** 000b: 20MHz. 001b: 25MHz. 010b: 33MHz. 011b: 50MHz. 100b: 66MHz. Others: Reserved.
19	RO	**Alert# pin type status** 1b: Open-drain Alert# pin is supported.
18:16	RO	**Maximum Frequency Supported.** 000b: 20MHz. 001b: 25MHz. 010b: 33MHz. 011b: 50MHz. 100b: 66MHz. Others: Reserved.

to next page

from previous page

15:12	RW	**Maximum WAIT STATE Allowed.**
11: 4	RO	**Reserved.**
3: 0	RO	**Channel Supported.** Bit 0: Peripheral Channel. Bit 1: Virtual Wire Channel. Bit 2: OOB Message Channel. Bit 3: Flash Access Channel.

Offset: 010h ESPICFG010: Channel 0 Capabilities and Configurations Init = 0000_1110

Bit	R/W	Description
31:15	RO	**Reserved.**
14:12	RW	**Peripheral Channel Maximum Read Request Size.** 000b: Reserved. 001b: 64 bytes address aligned max read request size. 010b: 128 bytes address aligned max read request size. 011b: 256 bytes address aligned max read request size. 100b: 512 bytes address aligned max read request size. 101b: 1024 bytes address aligned max read request size. 110b: 2048 bytes address aligned max read request size. 110b: 4096 bytes address aligned max read request size.
11	RO	**Reserved.**
10: 8	RW	**Peripheral Channel Maximum Payload Size Selected.** 000b: Reserved. 001b: 64 bytes address aligned max payload size. 010b: 128 bytes address aligned max payload size. 011b: 256 bytes address aligned max payload size. 100b: Reserved. 101b: Reserved. 110b: Reserved. 110b: Reserved.
7	RO	**Reserved.**
6: 4	RO	**Peripheral Channel Maximum Payload Size Supported.** 000b: Reserved. 001b: 64 bytes address aligned max payload size. 010b: 128 bytes address aligned max payload size. 011b: 256 bytes address aligned max payload size. 100b: Reserved. 101b: Reserved. 110b: Reserved. 110b: Reserved.
3	RO	**Reserved.**
2	RW	**Bus Master Enable.**
1	RO	**Peripheral Channel Ready.**
0	RW	**Peripheral Channel Enable.**

Offset: 020h	ESPICFG020: Channel 1 Capabilities and Configurations	Init = 0000_0700

Bit	R/W	Description
31:22	RO	**Reserved.**
27:16	RW	**Operating Maximum Virtual Wire Count.**
15:14	RO	**Reserved.**
13: 8	RW	**Maximum Virtual Wire Count Supported.**
7: 2	RO	**Reserved.**
1	RO	**Virtual Wire Channel Ready.**
0	RW	**Virtual Wire Channel Enable.**

Offset: 030h	ESPICFG030: Channel 2 Capabilities and Configurations	Init = 0000_0110

Bit	R/W	Description
31:11	RO	**Reserved.**
10: 8	RW	**OOB Message Channel Maximum Payload Size Selected.** 000b: Reserved. 001b: 64 bytes max payload size. 010b: 128 bytes max payload size. 011b: 256 bytes max payload size. 100b: Reserved. 101b: Reserved. 110b: Reserved. 110b: Reserved.
7	RO	**Reserved.**
6: 4	RO	**OOB Message Channel Maximum Payload Size Supported.** 000b: Reserved. 001b: 64 bytes max payload size. 010b: 128 bytes max payload size. 011b: 256 bytes max payload size. 100b: Reserved. 101b: Reserved. 110b: Reserved. 110b: Reserved.
3: 2	RO	**Reserved.**
1	RO	**OOB Message Channel Ready.**
0	RW	**OOB Message Channel Enable.**

Offset: 040h	ESPICFG040: Channel 3 Capabilities and Configurations	Init = 0003_1124

Bit	R/W	Description
31:15	RO	**Reserved.**
14:12	RW	**Flash Access Channel Maximum Read Request Size.** 000b: Reserved. 001b: 64 bytes address aligned max read request size. 010b: 128 bytes address aligned max read request size. 011b: 256 bytes address aligned max read request size. 100b: 512 bytes address aligned max read request size. 101b: 1024 bytes address aligned max read request size. 110b: 2048 bytes address aligned max read request size. 110b: 4096 bytes address aligned max read request size.

to next page

from previous page

11	RW	**Flash Sharing Mode.** 0: Master attached flash sharing. 1: Slave attached flash sharing.
10: 8	RW	**Flash Access Channel Maximum Payload Size Selected.** 000b: Reserved. 001b: 64 bytes max payload size. 010b: 128 bytes max payload size. 011b: 256 bytes max payload size. 100b: Reserved. 101b: Reserved. 110b: Reserved. 110b: Reserved.
7: 5	RO	**Flash Access Channel Maximum Payload Size Supported.** 000b: Reserved. 001b: 64 bytes max payload size. 010b: 128 bytes max payload size. 011b: 256 bytes max payload size. 100b: Reserved. 101b: Reserved. 110b: Reserved. 110b: Reserved.
4: 2	RW	**Flash Block Erase Size.** 000b: Reserved. 001b: 4 Kbytes. 010b: 64 Kbytes. 011b: Both 4 Kbytes and 64 Kbytes are supported. 100b: 128 Kbytes. 101b: 256 Kbytes. 110b: Reserved. 111b: Reserved.
1	RO	**Flash Access Channel Ready.**
0	RW	**Flash Access Channel Enable.**

Offset: 044h ESPICFG044: Channel 3 Capabilities and Configurations 2 Init = 0000_4401

Bit	R/W	Description
31:22	RO	**Reserved.**
21:16	RO	**Target RPMC Supported.** 0h: Slave does not support RPMC. 1h: Slave supports up to 1 RPMC. 2h: Slave supports up to 2 RPMC. ... 3fh: Slave supports up to 63 RPMC.
15: 8	RO	**Target Flash Erase Block Size for Master's Regions.** Bit 0: 1 Kbytes EBS supported. Bit 1: 2 Kbytes EBS supported. Bit 2: 4 Kbytes EBS supported. Bit 3: 8 Kbytes EBS supported. Bit 4: 16 Kbytes EBS supported. Bit 5: 32 Kbytes EBS supported. Bit 6: 64 Kbytes EBS supported. Bit 7: 128 Kbytes EBS supported.

to next page

from previous page

7: 3	RO	**Reserved.**
2: 0	RO	**Target Maximum Read Request Size Supported.** 000b: 64 byts max read request size. 001b: 64 byts max read request size. 010b: 128 byts max read request size. 011b: 256 byts max read request size. 100b: 512 byts max read request size. 101b: 1024 byts max read request size. 110b: 2048 byts max read request size. 111b: 4096 byts max read request size.

Offset: 800h **ESPICFG800: GPIO Direction of Virtual Wire Channel** **Init = 0**

Bit	R/W	Description
31: 0	RW	**Direction of GPIO.** 0: Master to slave 1: Slave to Master

Offset: 804h **ESPICFG804: GPIO Selection of Virtual Wire Channel** **Init = 0**

Bit	R/W	Description
31:24	RW	**Selection of GPIO bit 31 to 24.**
23:16	RW	**Selection of GPIO bit 23 to 16.**
15: 8	RW	**Selection of GPIO bit 15 to 8.**
7: 0	RW	**Selection of GPIO bit 7 to 0.** Onle Effective when ESPI00 bit 9 is 0. 00h: GPIOA 01h: GPIOB 02h: GPIOC 03h: GPIOD 04h: GPIOE 05h: GPIOF 06h: GPIOG 07h: GPIOH 08h: GPIOI 09h: GPIOJ 0Ah: GPIOK 0Bh: GPIOL 0Ch: GPIOM 0Dh: GPION 0Eh: GPIOO 0Fh: GPIOP 10h: GPIOQ 11h: GPIOR 12h: GPIOS 13h: GPIOT 14h: GPIOU 15h: GPIOV 16h: GPIOW (Slave to Master Only) 17h: GPIOX (Slave to Master Only) 18h: GPIOY 19h: GPIOZ 1Ah: GPIOAA 1Bh: GPIOAB

Offset: 808h	ESPICFG808: GPIO Reset Selection of Virtual Wire Channel	Init = 0
Bit	**R/W**	**Description**
31: 0	RW	**Reset Seletion of GPIO.** 0: eSPI Reset# 1: Platform Reset

Offset: 810h	ESPICFG810: Mapping Source Address of Peripheral Channel Rx Packet	Init = 0
Bit	**R/W**	**Description**
31: 0	RW	**Mapping Source Address of Peripheral Channel Rx Packet.**

Offset: 814h	ESPICFG814: Mapping Target Address of Peripheral Channel Rx Packet	Init = 0
Bit	**R/W**	**Description**
31: 0	RW	**Mapping Target Address of Peripheral Channel Rx Packet.**

Offset: 818h	ESPICFG818: Mapping Address Mask of Peripheral Channel Rx Packet	Init = ffff_0000
Bit	**R/W**	**Description**
31: 0	RW	**Mapping Source Address of Peripheral Channel Rx Packet.**

Offset: 820h	ESPICFG820: GP50 Direction of Virtual Wire Channel 0	Init = 0
Bit	**R/W**	**Description**
31: 0	RW	**GP50 Direction of Virtual Wire Channel.**

Offset: 824h	ESPICFG824: GP50 Direction of Virtual Wire Channel 0	Init = 0
Bit	**R/W**	**Description**
31: 0	RW	**GP50 Direction of Virtual Wire Channel.**

33.4 Programming Guide

33.4.1 Address of Peripheral Channel Rx Packet.(MemWR and MemRD)

Description of registers

- ESPICFG810 and ESPI084 are defining same definition.
  - However ESPICFG810 is accessible from eSPI master only.
  - ESPI084 is accessible from AST2600 itself only.
  - Each ESPICFG810 in this section can be replaced with ESPI084.
  - And so on for ESPICFG814/ESPI088 and ESPICFG818/ESPI08C.
- Setting ESPI08C[0] makes ESPICFG810/ESPICFG814/ESPICFG818 read-only from eSPI master.
- Setting ESPI08C[1] makes ESPI084/ESPI088/ESPI08C read-only from AST2600 .

- When a memory write or memory read is received, the address in packet must follow rules below. Response non-fatal error if otherwise.
  - Addr in packet & ESPICFG818 == ESPICFG810 & ESPICFG818.
- The mapped address is like below:
  - Target Addr = (Addr in packet & ∼ESPICFG818) || (ESPICFG814 & ESPICFG818).

Example

1. Assuming ESPICFG810 = 0xfedc0000; ESPICFG814 = 0x93010000; ESPICFG818 = 0xffff0000;
2. An incoming packet is received and the address is 0xfedc0104. This pass the first check.
   - 0xfedc0104 & 0xffff0000 == 0xfedc0000 & 0xffff0000;
3. Then Target address will be 0x93010104.
   - Target address = (0xfedc0104 & ∼(0xffff0000)) || (0x93010000 & 0xffff0000)

33.4.2 Process of receiving packet

- Channel 0 and 3
  1. After receiving a packet of channel 0 or 3, the corresponding bit in ESPI08 will be set. If corresponding bit in ESPI0C is also set, an interrupt is generated.
  2. The received cycle type, tag, and length can be read from each ”Control” register.
  3. Data is depending on setting of ESPI00 bit 16 to 23.
     - If it is 0, which is FIFO mode, data is stored in ”Data port” register.
     - If it is 1, which is DMA mode, data is stored at address which is defined in ”DMA Address” register.
  4. After finished processing of the packet, write 1 to bit 31 of ”Control” register to release the channel.
  - In channel 0, only message, message with data, and all kinds of completion will follow this procedure.
    IO and memory access is automatically finished.
- Channel 1
  1. When received a packet of channel 1, bit 8 or 9 of ESPI00 will be set. If corresponding bit in ESPI0C is also set, an interrupt is generated.
  2. The system events will goes to ESPI98 and GPIO goes to ESPI9C.

33.4.3 Procedure to sending packet

- Channel 0 and 3
  1. Preparing the data if necessary.
     - If it is FIFO mode, put data to ”Data port” register one byte each time.
     - If it is DMA mode, put data to the address which is defined in ”DMA Address” register.
  2. Write cycle type, tag, and length to ”Control” register.
  3. Write 1 to bit 31 of ”Control” register to trigger.
  4. After the packet is gotten by master, the corresponding bit in ESPI08 is set. If corresponding bit in ESPI0C is also set, an interrupt is generated.
- Channel 1
  - Just write the event or GPIO stat to ESPI98 or ESPI9C

34 MMBI Controller (MMBI)

34.1 Overview

MMBI is a function to exchange data between PCH and BMC via eSPI Peripheral Channel Memory Cycle.

MMBI000: MMBI Control Register.
MMBI008: Interrupt Status.
MMBI00C: Interrupt Enable.
MMBI010: Host_RWP 0 of Instance #0
MMBI014: Host_RWP 1 of Instance #0
MMBI018: Host_RWP 0 of Instance #1
MMBI01C: Host_RWP 1 of Instance #1
MMBI020: Host_RWP 0 of Instance #2
MMBI024: Host_RWP 1 of Instance #2
MMBI028: Host_RWP 0 of Instance #3
MMBI02C: Host_RWP 1 of Instance #3
MMBI030: Host_RWP 0 of Instance #4
MMBI034: Host_RWP 1 of Instance #4
MMBI038: Host_RWP 0 of Instance #5
MMBI03C: Host_RWP 1 of Instance #5
MMBI040: Host_RWP 0 of Instance #6
MMBI044: Host_RWP 1 of Instance #6
MMBI048: Host_RWP 0 of Instance #7
MMBI04C: Host_RWP 1 of Instance #7

34.2 Features

- Supporting total size is from 64KB to 8MB.
- Supporting maximum instance is 8.
- Supporting instance size is from 8KB to 1MB.

34.3 Registers : Base Address = 0x1E6E_E800

Offset: 000h	MMBI000: MMBI Control Register	Init = 0
Bit	**R/W**	**Description**
31:11	RO	**Reserved**
10: 8	RW	**Instance size** 000b : 8KB 001b : 16KB 010b : 32KB 011b : 64KB 100b : 128KB 101b : 256KB 110b : 512KB 111b : 1024KB
7	RO	**Reserved**

to next page

from previous page

6: 4	RW	**Total MMBI size** 000b : 64KB 001b : 128KB 010b : 256KB 011b : 512KB 100b : 1MB 101b : 2MB 110b : 4MB 111b : 8MB
3: 2	RO	**Reserved**
1	RW	**Write Protection of MMBI000[0]**
0	RW	**Enable MMBI function**

Offset: 008h **MMBI008: Interrupt Status** **Init = 0**

Bit	R/W	Description
31:16	RO	**Reserved**
15	RW1C	**Host_RWP 1 of Instance #7** Set when Host_RWP 1 of Instance #7 is updated. Write 1 Clear
14	RW1C	**Host_RWP 0 of Instance #7** Set when Host_RWP 0 of Instance #7 is updated. Write 1 Clear
13	RW1C	**Host_RWP 1 of Instance #6** Set when Host_RWP 1 of Instance #6 is updated. Write 1 Clear
12	RW1C	**Host_RWP 0 of Instance #6** Set when Host_RWP 0 of Instance #6 is updated. Write 1 Clear
11	RW1C	**Host_RWP 1 of Instance #5** Set when Host_RWP 1 of Instance #5 is updated. Write 1 Clear
10	RW1C	**Host_RWP 0 of Instance #5** Set when Host_RWP 0 of Instance #5 is updated. Write 1 Clear
9	RW1C	**Host_RWP 1 of Instance #4** Set when Host_RWP 1 of Instance #4 is updated. Write 1 Clear
8	RW1C	**Host_RWP 0 of Instance #4** Set when Host_RWP 0 of Instance #4 is updated. Write 1 Clear
7	RW1C	**Host_RWP 1 of Instance #3** Set when Host_RWP 1 of Instance #3 is updated. Write 1 Clear
6	RW1C	**Host_RWP 0 of Instance #3** Set when Host_RWP 0 of Instance #3 is updated. Write 1 Clear
5	RW1C	**Host_RWP 1 of Instance #2** Set when Host_RWP 1 of Instance #2 is updated. Write 1 Clear

to next page

from previous page

4	RW1C	**Host_RWP 0 of Instance #2** Set when Host_RWP 0 of Instance #2 is updated. Write 1 Clear
3	RW1C	**Host_RWP 1 of Instance #1** Set when Host_RWP 1 of Instance #1 is updated. Write 1 Clear
2	RW1C	**Host_RWP 0 of Instance #1** Set when Host_RWP 0 of Instance #1 is updated. Write 1 Clear
1	RW1C	**Host_RWP 1 of Instance #0** Set when Host_RWP 1 of Instance #0 is updated. Write 1 Clear
0	RW1C	**Host_RWP 0 of Instance #0** Set when Host_RWP 0 of Instance #0 is updated. Write 1 Clear

Offset: 00Ch **MMBI00C: Interrupt Enable** **Init = 0**

Bit	R/W	Description
31:16	RO	**Reserved**
15	RW	**Enable interrupt of Host_RWP 1 of Instance #7**
14	RW	**Enable interrupt of Host_RWP 0 of Instance #7**
13	RW	**Enable interrupt of Host_RWP 1 of Instance #6**
12	RW	**Enable interrupt of Host_RWP 0 of Instance #6**
11	RW	**Enable interrupt of Host_RWP 1 of Instance #5**
10	RW	**Enable interrupt of Host_RWP 0 of Instance #5**
9	RW	**Enable interrupt of Host_RWP 1 of Instance #4**
8	RW	**Enable interrupt of Host_RWP 0 of Instance #4**
7	RW	**Enable interrupt of Host_RWP 1 of Instance #3**
6	RW	**Enable interrupt of Host_RWP 0 of Instance #3**
5	RW	**Enable interrupt of Host_RWP 1 of Instance #2**
4	RW	**Enable interrupt of Host_RWP 0 of Instance #2**
3	RW	**Enable interrupt of Host_RWP 1 of Instance #1**
2	RW	**Enable interrupt of Host_RWP 0 of Instance #1**
1	RW	**Enable interrupt of Host_RWP 1 of Instance #0**
0	RW	**Enable interrupt of Host_RWP 0 of Instance #0**

Offset: 010h	**MMBI010: Host_RWP 0 of Instance #0**	**Init = 0**
Offset: 014h	**MMBI014: Host_RWP 1 of Instance #0**	**Init = 0**
Offset: 018h	**MMBI018: Host_RWP 0 of Instance #1**	**Init = 0**
Offset: 01Ch	**MMBI01C: Host_RWP 1 of Instance #1**	**Init = 0**
Offset: 020h	**MMBI020: Host_RWP 0 of Instance #2**	**Init = 0**
Offset: 024h	**MMBI024: Host_RWP 1 of Instance #2**	**Init = 0**
Offset: 028h	**MMBI028: Host_RWP 0 of Instance #3**	**Init = 0**
Offset: 02Ch	**MMBI02C: Host_RWP 1 of Instance #3**	**Init = 0**
Offset: 030h	**MMBI030: Host_RWP 0 of Instance #4**	**Init = 0**
Offset: 034h	**MMBI034: Host_RWP 1 of Instance #4**	**Init = 0**
Offset: 038h	**MMBI038: Host_RWP 0 of Instance #5**	**Init = 0**
Offset: 03Ch	**MMBI03C: Host_RWP 1 of Instance #5**	**Init = 0**
Offset: 040h	**MMBI040: Host_RWP 0 of Instance #6**	**Init = 0**
Offset: 044h	**MMBI044: Host_RWP 1 of Instance #6**	**Init = 0**
Offset: 048h	**MMBI048: Host_RWP 0 of Instance #7**	**Init = 0**
Offset: 04Ch	**MMBI04C: Host_RWP 1 of Instance #7**	**Init = 0**

Bit	Attr.	Description
63:32	RO	**Host_RWP 1**
31: 0	RO	**Host_RWP 0**

34.4 Programming Guide

34.4.1 Memory Layout

	Name	Size (Bytes)	Host	BMC
Instance #0	BMC2Host	X	RO	RW
Instance #1	BMC2Host	X	RO	RW
...				
Instance #N	BMC2Host	X	RO	RW
Instance #0	Host2BMC	X	RO	RO/RW
Instance #1	Host2BMC	X	RO	RO/RW
...				
Instance #N	Host2BMC	X	RO	RO/RW

- Maximum N is 7
- X is from 4KB to 32KB.
- Each instance occupies 2*X bytes.
- First max 8 bytes of Host2BMC are RO from BMC. The rest is RW.

34.4.2 Single Instace Memory Layout example

	Name	Size (Bytes)	Host	BMC
BMC2Host X KB	MMBI_Config (if needed)	N	RO	RW
	Host_ROP	M	RO	RW
	B2H Circular Buffer	(X - N - M)	RO	RW
Host2BMC X KB	Host_RWP	Y	RW	RO
	H2B Circular Buffer	(X - Y)	RW	RW

- N and M are unlimited as long as (N + M) ¡= X.
- Maximum Y is 8 bytes.
- Host_RWP are Read-Only for BMC.

35 Battery Backed SRAM (BSRAM)

35.1 Overview

Battery Backed SRAM is a 128 bytes memory that is direct connected to APB bus.
Address space is from 0x1E6E_0000 to 0x1E6E_FFFF. The SRAM data will be backed by a chip external 3.0 - 3.5V battery when AST2600 chip core and I/O power is turned off. The data can be sustaned when the battery power is above 2.0V.

Base address of Battery Backed SRAM = 0x1E6E_F000
Physical address = (Base address of Battery Backed SRAM) + Offset

Offset: 000h	BSRAM00: Protection Key Register	Init = 0

Bit	R/W	Reset	Description
31:0	RW	Rst03	**Protection key** This register is designed to protect the battery backed SRAM from unpredictable updates. All the SRAM are still readable even if they are locked. The password for the protection key is **0xdba0_78e2** or **0xdba0_78e0**. **Unlock SRAM with slow timing**: Write 0xdba0_78e2 to this register (recommended) **Unlock SRAM with fast timing**: Write 0xdba0_78e0 to this register **Lock SRAM**: Write other values to this register When this SRAM is unlocked with slow timing, the read back value of this register is 0x0000_0003. When this SRAM is unlocked with fast timing, the read back value of this register is 0x0000_0001. When this SRAM is locked, the read back value of this register is 0x0000_0000. This register will be reset by power on reset, watch dog reset and SCU software reset. Software must wait minimum 1us to unlock the key after reset signal de-asserted.

Offset: 010h	CHAI10: Intrusion Control Register	Init = 0

Bit	R/W	Reset	Description
31:19	RO	-	**reserved (0)**
18	RO	CHAI10[16]	**I/O Power Good Status** 0: I/O power good has not been pulled low since last clear. 1: I/O power good has been pulled low since last clear.
17	RW	Rst03	**I/O Power Good Status Interrupt Enable** 0: disable interrupt 1: enable interrupt
16	RW	Rst03	**I/O Power Good Status Clear** 0: normal operation 1: clear I/O power status bit to 0 The I/O power status will be cleared when the value of this bit is 1. Set this bit to 0 to go to normal operation.
15:11	RO	-	**reserved (0)**
10	RO	CHAI10[8]	**Core Power Good Status** 0: Core power good has not been pulled low since last clear. 1: Core power good has been pulled low since last clear.

to next page

from previous page

9	RW	Rst03	**Core Power Good Status Interrupt Enable** 0: disable interrupt 1: enable interrupt
8	RW	Rst03	**Core Power Good Status Clear** 0: normal operation 1: clear core power status bit to 0 The core power status will be cleared when the value of this bit is 1. Set this bit to 0 to go to normal operation.
7:5	RO	-	**reserved (0)**
4	RO	-	**Raw Status of CHASI#** 0: CHASI# is in low state 1: CHASI# is in high state
3	RO	-	**Battery Power Good** 0: Battery power is not good 1: Battery power is good
2	RO	CHAI10[0]	**Intrusion Status** 0: CHASI# pin has not been pulled low since last clear. 1: CHASI# pin has been pulled low since last clear.
1	RW	Rst03	**Intrusion Interrupt Enable** 0: disable interrupt 1: enable interrupt
0	RW	Rst03	**Intrusion Status Clear** 0: normal operation 1: clear intrusion status bit to 0 The intrusion status will be cleared when the value of this bit is 1. Set this bit to 0 to go to normal operation.

BSRAM100 ~ BSRAM17C: Battery Backed SRAM #0 ~ #31

Offset: 100~17Fh **Init = X**

Bit	Attr.	Description
31:0	RW	**Battery Backed SRAM** The SRAM data backed by external battery.

36 Video Engine (VE)

36.1 Overview

Video Engine supports high performance video compressions with a wide range of video quality and compression ratio options. The adopted compressing algorithm is a modified JPEG algorithm. To enable video compression engine, the following memory buffers are required to be allocated from DRAM memory for each of them.

- Video Source Buffer #1
- Video Source Buffer #2
- JPEG Header Buffer
- Block Change Detection (BCD) Flag Buffer (for scene change detection buffer)
- Copy Buffer (for advance BCD detection mode)
- Compressed Video Stream Buffer

Video Engine implements many registers to program the various supported features. The physical address of these registers can be derived as the following:

Base address of Video Engine = 0x1E70_0000
Physical address = (Base address of Video Engine) + Offset

VR000: Protection Key Register
VR004: Video Engine Sequence Control Register
VR008: Video Control Register
VR00C: Video Timing Generation Setting Register (VR008[5]=0)
VR010: Video Timing Generation Setting Register (VR008[5]=0)
VR014: Video Scaling Factor Register
VR018: Video Scaling Filter Parameter Register #0
VR01C: Video Scaling Filter Parameter Register #1
VR020: Video Scaling Filter Parameter Register #2
VR024: Video Scaling Filter Parameter Register #3
VR02C: Video BCD Control Register
VR030: Video Capturing Window Setting Register
VR034: Video Compression Window Setting Register
VR038: Video Compression Stream Buffer Processing Offset Register
VR03C: Video Compression Stream Buffer Read Offset Register
VR040: Video Base Address of JPEG Header Buffer Register when VR004[13]=1
VR044: Video Based Address of Video Source Buffer #1 Register
VR048: Video Scan Line Offset of Video Source Buffer Register
VR04C: Video Base Address of Video Source Buffer #2 Register
VR050: Video Base Address of BCD Flag Buffer Register
VR054: Video Base Address of Compressed Video Stream Buffer Register
VR058: Video Stream Buffer Size Register
VR05C: Video Compression Stream Buffer Write Offset Read Back
VR060: Video Compression Control Register
VR064: Video JPEG effecive bit control Register
VR068: Video Quantization value
VR06C: Video Copy Buffer Base Address when VR02C[2]=1
VR070: Video Total Size of Compressed Video Stream Read Back Register
VR074: Video Total Number of Compressed Video Blocks Read Back Register
VR078: Video Frame-End Offset of Compressed Video Stream Buffer Read Back Register
VR07C: Video Compressed Frame Counter Read Back Registe
VR080: Video User Defined Header Parameter Setting Register when Compression

VR084: JPEG Compressed Size Read Back Register
VR090: Video Source Left/Right Edge Detection Read Back Register
VR094: Video Source Top/Bottom Edge Detection Read Back Register
VR098: Video Mode Detection Status Read Back Register
VR09C: Video Sync End Location Detection Read Back Register
VR0A0: Video Horizontal Total Pixel Count Read Back Register
VR0A4: Extended Mode Detection Control Register 1
VR0A8: Extended Video Control Register 1
VR0AC: Direct Access Mode Virtual Timing Setting Register 1
VR0B0: Direct Access Mode Virtual Timing Setting Register 2
VR0B4: Extended Video Control Register 2
VR0B8: Video Source Left/Right Edge Detection 4K2K Read Back Register (re-mapping from VR090)
VR0BC: Video Mode Detection Status 4K2K Read Back Register (re-mapping from VR098)
VR0C0: Video Engine Status 1 Read Back Register
VR0C4: Video Engine Status 2 Read Back Register
VR0C8: Video Engine Status 3 Read Back Register
VR0CC: Video Engine Status 4 Read Back Register
VR0D0: Extra Video Mode Detection Status Read Back Register
VR0D4: Video BCD Mode Bounding Box Detection Read Back Register 1
VR0D8: Video BCD Mode Bounding Box Detection Read Back Register 2
VR204: Video Management Sequence Control Register
VR208: Video Management Control Register
VR238: Video Management Compression Stream Buffer Processing Offset Register
VR23C: Video Management Compression Stream Buffer Read Offset Register
VR244: Video Management Based Address of Video Source Buffer Register
VR248: Video Management Scan Line Offset of Video Source Buffer Register
VR24C: Video Management Base Address of Video Source Buffer #2 Register
VR250: Video Management Base Address of BCD Flag Buffer Register
VR254: Video Management Base Address of Compressed Video Buffer Register
VR260: Video Management Compression or Video Profile 2-5 Decompression Control Register
VR268: Video Management Quantization value
VR270: Video Management Total Size of Compressed Video Stream Read Back Register
VR278: Video Management Frame-End Offset of Compressed Video Stream Buffer Read Back Register
VR280: Video Management User Defined Header Parameter Setting Register when Compression
VR300: Video Control Register
VR304: Video Interrupt Control Register
VR308: Video Interrupt Control Register
VR30C: Mode Detection Parameter Register
VR310: Video Memory Restriction Area Starting Address Register
VR314: Video Memory Restriction Area End Address Register
VR318: Video Memory Restriction Area Starting Address Register
VR328: Video Data Truncation Register
VR33C: Video Watch Dog Timer Read Back
VR340: VGA Scratch Remap Read Back Register
VR344: VGA Scratch Remap Read Back Register
VR348: VGA Scratch Remap Read Back Register
VR34C: VGA Scratch Remap Read Back Register
VR350: VGA Scratch Remap Read Back Register
VR354: VGA Scratch Remap Read Back Register
VR358: VGA Scratch Remap Read Back Register
VR35C: VGA Scratch Remap Read Back Register
VR360: VGA Scratch Remap Read Back Register
VR364: VGA Scratch Remap Read Back Register
VR3F0: Video Crypto Vector Register 1
VR3F4: Video Crypto Vector Register 2
VR3F8: Video Crypto Vector Register 3
VR3FC: Video Crypto Vector Register 4

VR400 ~ VR4FC: Video RC4/AES128 Encryption Key Register #0 ~ #63
VR400 ~ VR5FC: Video Quantization Table and Inverse Quantization Table #0 ~ #63
VR400 ~ VR55C: JPEG Huffman Table #0 ~ #63
VR400 ~ VR5FC: Multi-JPEG Data Buffer #0 ~ #31

36.2 Features

- Buiilt-in Hardware video compression engine that can reduce CPU loading
- Directly connected to AHB bus interface for register programming
- Directly accessible video data through M-Bus
- Video source can come from internal VGA or external DVO input
- Engine clock can be the same as CPU clock or memory clock
- Internal VGA mode:
  - Video capture mode: capture internal VGA RGB digital signals (Applied to legacy VGA display modes, like text modes or 16/256 color VGA modes)
  - Quick fetch mode: directly fetch RGB video data from VGA frame buffer (Applied to 16bpp and 32bpp VGA modes when memory bandwidth is limited.)
- Engine clock can be turned off when engine is idle
- Support two video compression quality modes
  - YUV420: for lower video quality but higher compression ratio
  - YUV444: for higher video quality but lower compression ratio
- Support two video compression formats
  - ASPEED proprietary compression mode: for multi-frame and differentical compression
  - JPEG JFIF standard mode: for single frame and management compression
- Support high resolution video compression up to 1920x1200x32bpp@60Hz
- Target frame rate: 30 frame/sec for 1280x1024@60Hz under YUV420 compression format
- Support independent management view capturing for special purpose like last frame recording.
- Support Quick fetch for video compression
  - Significantly reduces memory bandwidth requirements for video compression
  - Only enabled for high resolution modes (high color and true color modes)
  - Quick Cursor must be enabled (cursor overlay will be done in client site)
  - Regular VGA display refreshes can be turned off to save power and to reduce DRAM utilization rate
- Support arbitrary video down scaling with horizontal & vertical video filtering option (4x2 spatial filter)
- Integrate AES, RC4 encryption engine for video stream encryption
  - 1 set of loadable 256x8 SRAM for expanded key buffers
  - Key expansion is done by firmware
  - Provide enable/disable option
- Support two-pass (high quality) video compression scheme (Patent pending by ASPEED)
  - Applied for both YUV444 and YUV420 video format
  - Provide visually lossless video compression quality or to reduce the network average loading under intranet KVM applications
- Support smart video mode detection functions
- Support video mode watch dog interrupt when source video mode changes
- Support programmable bit resolution truncation to input video data
- Support 12 selectable pre-defined JPEG quality levels

- Support programmable selectable JPEG quality levels
- Support video auto stream mode and single frame trigger mode
- Support ring buffer mode and descriptor DMA buffer mode
- Support multi-JPEG compression mode with single trigger

36.3 Registers : Base Address = 0x1E70:0000

VR000: Protection Key Register			
Offset: 000h			**Init = 0**
Bit	**Attr.**	**Reset**	**Description**
31:0	RW	Rst17	**Protection key** This register is designed to protect all the registers associated with Video Engine from unpredictable updates. All the registers are still readable even if they are locked. The password for the protection key is **0x1A03_8AA8**. **Unlock registers**: Write 0x1A03_8AA8 to this register **Lock registers**: Write other values to this register When this register is unlocked, the read back value of this register is 0x0000_0001. When this register is locked, the read back value of this register is 0x0000_0000. This register will be reset by power on reset, watch dog reset and SCU software reset. Software must wait minimum 1us to unlock the key after reset signal de-asserted.

VR004: Video Engine Sequence Control Register			
Offset: 004h			**Init = 0**
Bit	**Attr.**	**Reset**	**Description**
31:24	RO	-	**Reserved (0)**
23	RO	Rst17	**Video engine JPEG JFIF compression status read back** 0: Video JPEG JFIF engine is in idle state 1: Video JPEG JFIF engine is in busy state
22	RO	Rst17	**Video engine insert full frame status read back** 0: Video compression engine insert full frame is in idle state 1: Video compression engine insert full frame is in busy state
21	RO	Rst17	**Video halt engine read back** 0: Video halt engine mode is not triggered 1: Video halt engine mode is triggered
20	RO	Rst17	**Video engine profile change status read back** 0: Video engine profile change is in idle state 1: Video engine profile change is in busy state
19	RO	-	**Reserved (0)**
18	RO	Rst17	**Video compression engine status** 0: Video compression engine is in busy state 1: Video compression engine is in idle state
17	RO	-	**Reserved (0)**
16	RO	Rst17	**Video capture engine status** 0: Video capture engine is in busy state 1: Video capture engine is in idle state
15	RW	Rst17	**Trigger Video to compress a JPEG JFIF compatible frame** Write 0: No operation. Write 1: Trigger engine to insert a JPEG JFIF compatible frame compression. This is use only when multiple frames mode and stream buffer mode.
14	RO	-	**Reserved (0)**

to next page

from previous page

13	RW	Rst17	**Enable JPEG JFIF compatible mode compression/decompression** 0: ASPEED proprietary compression mode 1: Enable JPEG compatible mode
12	RW	Rst17	**Video engine status** Write 0: Exit video engine halt state Write 1: Trig video engine halt state This register can be used to halt video engine in stream buffer mode. The video engine will halt when it finishes one full frame compression after this bit is triggered. Read 0: Video engine is in normal state Read 1: Video engine is in halt state
11:10	RW	Rst17	**Video data format conversion for video compression** 00: YUV444 01: YUV420 10: Reserved 11: Reserved Video capture engine always captures the input RGB data stream into the YUV444 data format and write out to the video memory. Changing the setting to 1 converts video data format to YUV420 format before doing any video compression. When YUV420 mode is enabled, the setting of VR300[13:10] will be ignored.
9	RW	Rst17	**Reserved** This register must always be ”0”
8	RW	Rst17	**Enable YUV to RGB transform dither** 0: Disable dither 1: Enable dither to reduce quantization error effects
7	RW	Rst17	**Enable watchdog for detecting input video resolution mode change** 0: Disable watchdog 1: Enable watchdog The display resolution of an input video source may change anytime. Therefore, building a watchdog is necessary for the dedicated monitor to change and generate an interrupt to notice S/W handler when a mode change does happen. This register works only when VR004[0] is ”1” and a stable video resolution has been detected. Set mode detection trigger first and then set this bit when the mode detection is ready.
6	RW	Rst17	**Trigger video to insert full frame compression** Write 0: No operation. Write 1: Trigger engine to insert signle full frame compression for stream mode encode. Read 0: The insertion is completed. Read 1: The insertion is not completed.
5	RW	Rst17	**Enable automatic video compression** 0: Compress a single frame for each trigger command 1: Compress multiple frames for each trigger command When this register is enabled, video capturing engine and video compression engine can work together to automatically capture and compress continuous frames without the control of S/W. Allocating double buffer is required before enabling this register. Software must set this bit first before setting the trigger register VR004[4].

to next page

from previous page

4	RW1T	Rst17	**Enable or trigger video compression** 0: No operation 0→1: Trigger video compression when compressing the single frame mode 1: Enable video compression Setting this register from 0 to 1 will trigger the video compression engine to compress the captured video data stored in the frame buffer. Video compression can compress single or multiple frames depending on the setting of the register VR004[5]. Software must insert at least one read cycle or 1us delay time between the continuous trigger register setting. Software must make sure that the VR004[18] is '1' before triggering this bit.
3	RW	Rst17	**Enable capturing multiple frames** 0: Capturing single frame 1: Capture multiple frames AST2600 can allocate double buffers for the video capture engine to continuously capture multiple frames. Capturing multiple frames can improve video performance. Allocating double buffers is required before enabling this register. Software must set this bit first before setting the trigger register VR004[1].
2	RW	Rst17	**Force video compression engine Idle** 0: No operation 1: Force compression engine to enter idle state This register is used by software to force compression engine to enter idle state only when capture engine is idle and compression engine hangs up.
1	RW1T	Rst17	**Enable or trigger video capture** 0: When video capture is idle: no operation 0: When video capture is not idle and in capture multiple frames mode (VR004[3]=1): Capture engine will stop when the last video frame in completely captured. 0→1: Trigger video capture when capture single frame mode Setting this register from 0 to 1 will trigger video capture engine to capture either single or multiple video frames, depending on the setting of VR004[3]. Video capture engine will stop capturing video at the end of a frame whenever this register is reset to 0. Software must insert at least one read cycle or 1us delay time between continuous triggering register setting. Software must make sure that the VR004[16] is '1' before trigger this bit.
0	RW1T	Rst17	**Trigger video mode detection hardware** 0: No operation 0→1: Trigger video mode detection 1: Enable mode dectection hardware Setting this register from 0 to 1 will trigger the video mode detection hardware to detect a video mode based on the input video source. When a stable video mode has been detected, the hardware will set the corresponding flag for status read back. And the related video parameters generated by the hardware can also be read back from the related registers. An optional interrupt is also available. Software must insert at least one read cycle or 1us delay time between continuous trigger register setting.

to next page

from previous page

Note :
About auto and trigger mode, please reference following table:

VR004[3]	VR004[5]	Capture	Compression	Buffer mode
0	0	Single frame	Software trigger	Frame buffer mode
0	1	Single frame	Hardware auto trigger	Frame buffer mode
1	0	NA	Multi-JPEG trigger	Multi-JPEG mode
1	1	Multiple frames	Hardware auto trigger	Stream buffer mode

VR008: Video Control Register

Offset: 008h **Init = 0**

Bit	Attr.	Reset	Description
31	RW	Rst17	**Enable compare only flag mode** 0: Disable 1: Enable This mode is used by setting VR008[31]=1 and VR02C[0]=1, then trigger a compression by VR004[5]=1. Other registers are same as compression mode. For example VR044 is the current frame and VR04C is the previous frame. In the result of flag buffer, 0x0 means the block is different and 0xf means the block is equal.
30	RW	Rst17	**Multi-JPEG Mode** 0: Disable 1: Enable For multi-JPEG Mode should also need to set VR004[14]=1, VR004[3]=1, VR004[1]=0 and VR004[5]=0
29:24	RW	Rst17	**Multi-JPEG Number** Multi-JPEG mode JPEG number setting (JPEG number = VR008[29:24]+1. The maximum JPEG number is 32)
23:16	RW	Rst17	**Maximum frame rate control for the video capture** When this register is reset to 0x00, capture engine will try to capture all the input frames, if memory and network bandwidth is sufficient for doing that. When this register is set to a non-zero value, video capture engine will skip some frames to reduce memory and network bandwidth. The maximum frame rate will be: Maximum frame rate = (VR008[23:16]) * (Source frame rate) / 60.
15	RW	Rst17	**HSYNC polarity control for mode detection** 0: Normal HSYNC 1: Inverted HSYNC polarity for mode detection
14	RW	Rst17	**Video source interlace mode** 0: Capture source is in the progressive mode 1: Capture source is in the interlace mode
13:12	RW	Rst17	**Reserved** This register must be always ”0”

to next page

from previous page

11:9	RW	Rst17	**Clock delay control for digital video input port** 000: No delay 001: Delay by 1 ns 010: Delay by 2 ns 011: Delay by 3 ns 100: Inversed clock but no delay 101: Inversed clock and delay by 1 ns 110: Inversed clock and delay by 2 ns 111: Inversed clock and delay by 3 ns This register can adjust the delay of the input video clock for the video capture engine to precisely capture the video data.
8	RW	Rst17	**Disable hardware cursor overlay for internal VGA (VR008[5]=0)** 0: With VGA hardware cursor overlay image 1: Without VGA hardware cursor overlay image This register can be set by the ARM CPU to inform internal VGA controller to generate video data without hardware cursor overlay image. When this register is enabled, the hardware cursor overlay has to be done in clients by Quick Cursor algorithm. The DAC output of internal VGA controller is, if necessary, with hardware cursor overlay image even this register is set to 1.
8	RW	Rst17	**Auto mode for direct fetch mode for VGA frame buffer (VR008[5]=1)** 0: normal operation 1: auto mode for register VR008[3], VR008[4], VR00C and VR010 This bit is used only for direct fetch mode for VGA frame buffer when VR004[5]=1. When this register is set to '1', the software dose't require it to set the registers VR008[3], VR008[4], VR00C and VR010. The video hardware will automatically refer to the VGA hardware setting for these registers.
7 :6	RW	Rst17	**Data format for video capture** 00: CCIR601-2 compliant YUV format for ASPEED proprietary compression mode JPEG compliant YUV format for JPEG mode 01: JPEG JFIF compliant YUV format for both JPEG mode and ASPEED proprietary compression mode 10: RGB format for ASPEED proprietary compression mode only 11: Gray color mode for both JPEG mode and ASPEED proprietary compression mode AST2600 supports two kinds of YUV formats. One is CCIR601-2 compatible; the other is in a proprietary format to support higher video quality. RGB format is for debugging purpose only.
5	RW	Rst17	**Fetch video data directly from VGA frame buffer (internal VGA only)** 0: From internal VGA input or external digital video input 1: Directive fetch from VGA display frame buffer This register is designed for capturing high resolution video modes under low memory bandwidth requirement. It only supports high color and true color display modes. And there is no hardware cursor overlay from the captured data. Therefore, hardware cursor overlay has to be done by clients by Quick Cursor algorithm.
4	RW	Rst17	**Use the internal timing generator to generate Display Enable (DE) signal (VR008[5]=0)** 0: Use the external DE signal (DVI only) 1: Use the internal DE signal DE signal is used for video capture engine to precisely sampling effective video data.

to next page

from previous page

4	RW	Rst17	**VGA frame buffer bpp mode (VR008[5]=1)** 0: 32 bpp mode 1: 16 bpp mode This register sets the number of bits per pixel in VGA frame buffer.
3	RW	Rst17	**Attribute of the external video source (VR008[5]=0)** 0: Form a pure digital video source 1: From an external Analog-to-Digital Converter (ADC) If a digital video source is from an external video ADC, the internal timing generator will be automatically enabled no matter the register setting of VR004[4].
3	RW	Rst17	**VGA frame buffer 16 bpp color mode (VR008[5]=1)** 0: RGB565 1: RGB555 This register sets the 16 bpp color format in VGA frame buffer
2	RW	Rst17	**Video source selection** 0: Video source is from the integrated VGA controller 1: Video source is from an external video source
1	RW	Rst17	**Video source VSYNC polarity selection** 0: Internal VSYNC polarity is same as source VSYNC 1: Internal VSYNC polarity is inversed of source VSYNC Note: This register should be reset to 0 before triggering video mode detection hardware.
0	RW	Rst17	**Video source HSYNC polarity selection** 0: Internal HSYNC polarity is same as source HSYNC 1: Internal HSYNC polarity is inversed of source HSYNC Note: This register should be reset to 0 before triggering video mode detection hardware.

VR00C: Video Timing Generation Setting Register (VR008[5]=0)

Offset: 00Ch **Init = X**

Bit	Attr.	Reset	Description
31:29	RO	-	**Reserved (0)**
28:16	RW	-	**Number of pixels to the first active pixel** This register defines the number of pixels from the rising edge of HSYNC to the first active pixel
15:13	RW	-	**Reserved (0)**
12:0	RW	-	**Number of pixels to the last active pixel** This register defines the number of pixels from the rising edge of HSYNC to the last active pixel.
Note : Timing generator is primarily designed for video source, like the one form ADC, going without Horizontal Display Enable (HDE) signal. Timing generator will generate one for video capture engine to precisely capture active pixels for the first active pixel to the last active pixel in a scan line.			
VR00C: Video Direct Frame Buffer Mode Control Register (VR008[5]=1)			
31:28	RW	-	**Reserved (0)**
28:3	RW	-	**Direct frame buffer fetch mode base address bit [28:3]** When direct frame buffer fetch mode is enabled (VR008[5]=1), this register set the base address of source video data.
2 :0	RW	-	**Reserved (0)**

VR010: Video Timing Generation Setting Register (VR008[5]=0)			
Offset: 010h			**Init = X**
Bit	**Attr.**	**Reset**	**Description**
31:28	RW	-	**Reserved (0)**
28:16	RW	-	**Number of scan lines to the first active scan line** This register defines that number scan lines from the rising edge of VSYNC to the first active scan line.
15:12	RW	-	**Reserved (0)**
12:0	RW	-	**Number of scan lines to the last active scan line** This register defines the number scan lines from the rising edge of VSYNC to the last active scan line.
Note : Timing generator is primarily designed for video source, like the one form ADC, going without Vertical Display Enable (VDE) signal. Timing generator will try to generate one for video capture engine to precisely capture scan lines from the first active scan line to the last active scan line in a frame.			
VR010: Video Direct Frame Buffer Mode Control Register (VR008[5]=1)			
31:16	RW	-	**Direct frame buffer fetch timing control bit[15:0]** When direct frame buffer fetch mode is enabled (VR008[5]=1), it controls the 64-pixel segment minimum fetch time = VR010[31:15] * MCLK_cycle_time
13:3	RW	-	**Direct frame buffer fetch mode line offset bit [27:3]** When direct frame buffer fetch mode is enabled (VR008[5]=1), this register sets the line offset of source video data.
2 :0	RW	-	**Reserved (0)**

VR014: Video Scaling Factor Register			
Offset: 014h			**Init = X**
Bit	**Attr.**	**Reset**	**Description**
31:16	RW	-	**Vertical down scaling factor** The setting value of this register must be equal to or larger than 4096. All the active scan lines will be kept. When the setting value is larger, the output video window size will be smaller. The formula is as the fowling: (Vertical window size of output video) = (Veridical window size of input video) * 4096 / (Vertical scaling factor)
15:0	RW	-	**Horizontal down scaling factor** The setting value of this register must be equal to or larger than 4096. All the active pixels will be kept. When the setting value is larger, the output video window size will be smaller. The formula is as the fowling: (Horizontal window size of output video) = (Horizontal window size of input video) * 4096 / (Horizontal scaling factor)

VR018: Video Scaling Filter Parameter Register #0

Offset: 018h **Init = X**

Bit	Attr.	Reset	Description
31:0	RW	-	**Scaling parameters F00, F01, F02, F03** F03: Bit[31:24] F02: Bit[23:24] F01: Bit[15:8] F00: Bit[7:0] The data format of the scaling parameters is 2'comlement format by S2.5.

Note :
The video scalar integrated in AST2600 can support a 2x4 scaling filter. It means that the video output of each down-scaled pixel will be generated by the weighting sum of the surrounding 8 pixels, avoiding graphics information loss. There are three alowed parameter settings.
when scaling down factor = 1.0, VR018, VR01C, VR020, VR024 = 00200000h,
when 0.5 <= scaling down factor < 1.0, VR018, VR01C, VR020, VR024 = 00101000h,
when scaling down factor < 0.5, VR018, VR01C, VR020, VR024 = 08080808h

VR01C: Video Scaling Filter Parameter Register #1

Offset: 01Ch **Init = X**

Bit	Attr.	Reset	Description
31:0	RW	-	**Scaling parameters F10, F11, F12, F13** F13: Bit[31:24] F12: Bit[23:24] F11: Bit[15:8] F10: Bit[7:0] The data format of the scaling parameters is 2'complement format by S2.5.

VR020: Video Scaling Filter Parameter Register #2

Offset: 020h **Init = X**

Bit	Attr.	Reset	Description
31:0	RW	-	**Scaling parameters F20, F21, F22, F23** F23: Bit[31:24] F22: Bit[23:24] F21: Bit[15:8] F20: Bit[7:0] The data format of the scaling parameters is 2'complement format by S2.5.

VR024: Video Scaling Filter Parameter Register #3

Offset: 024h **Init = X**

Bit	Attr.	Reset	Description
31:0	RW	-	**Scaling parameters F30, F31, F32, F33** F33: Bit[31:24] F32: Bit[23:24] F31: Bit[15:8] F30: Bit[7:0] The data format of the scaling parameters is 2'complement format by S2.5.

VR02C: Video BCD Control Register			
Offset: 02Ch			**Init = 0xXXXX0000**
Bit	**Attr.**	**Reset**	**Description**
31:24	RW	-	**ABCD tolerance value** This register defines the tolerance for block change detection (ABCD). Any blocks with maximum changing difference in its own block that is more than this tolerance value will be taken as a non-changing blocks. This function is designed to reduce memory and network bandwidth, especially for the noisy video source from the ADC or dithered source and avoid gradually chaning data detection error.
23:16	RW	-	**BCD tolerance value** This register defines the tolerance for block change detection (BCD). Any blocks with maximum changing difference in its own block more than this tolerance value will be taken as a non-changing blocks. This function is designed to reduce memory and network bandwidth, especially for the noisy video source from ADC or dithered source.
15:8	RW	Rst17	**Reserved (0)**
7:5	RW	Rst17	**Best quality video delay frames for block change update** 000: No delay 001: Delay 1 frame ... 111: Delay 7 frames When this register is set, the BCD can purposely delay the block change update for each changed block by a few frames. The changed block will be updated in the next frame. This function is designed to reduce memory and network bandwidth.
4:3	RW	Rst17	**High quality video delay frames for block change update** 00: No delay 01: Delay 1 frame 10: Delay 2 frame 11: Delay 3 frame When this register is set, BCD can purposely delay the block change update for each changed block by a few frames. The changed block will be updated in the next frame. This function is designed to reduce memory and network bandwidth.
2	RW	Rst17	**Enable copy buffer mode** 0: Normal Mode 1: Enable copy buffer (VR06C) for accurate block change detection when encode mode. Enable double buffer mode during encode mode. It can cause more memory copy and avoid display artifact when encode and decode video is not synchronized.
1	RW	Rst17	**Enable analog or dither source change detection (ABCD)** 0: Disable 1: Enable during ASPEED proprietary compression mode only The BCD must also be enabled for enabling ABCD. This function is designed for the noisy video source from ADC or dithered source and to avoid gradually chaning the data detection error.
0	RW	Rst17	**Enable block change detection (BCD)** 0: Disable 1: Enable during ASPEED proprietary compression mode only When BCD is disabled, video compression engine will compress all the input frames. When BCD is enabled, video compression engine will detect and compress changing blocks only. That will significantly reduce memory and network bandwidth. Higher video frame rate can be achieved as well.

VR030: Video Capturing Window Setting Register			
Offset: 030h			**Init = X**
Bit	**Attr.**	**Reset**	**Description**
31:29	RO	-	**Reserved (0)**
28:16	RW	-	**Horizontal total pixels to be captured**
15:13	RW	-	**Reserved (0)**
12:0	RW	-	**Vertical total scan lines to be captured**
Note : The register provides the possibility of doing video capture only for a partial display window in a frame.			

VR034: Video Compression Window Setting Register			
Offset: 034h			**Init = X**
Bit	**Attr.**	**Reset**	**Description**
31:29	RO	-	**Reserved (0)**
28:16	RW	-	**Horizontal total pixels to be compressed**
15:13	RO	-	**Reserved (0)**
12:0	RW	-	**Vertical total scan lines to be compressed**
Note : The register provides the possibility of doing video compression only for a partial display window in a frame.			

VR038: Video Compression Stream Buffer Processing Offset Register			
Offset: 038h			**Init = 0**
Bit	**Attr.**	**Reset**	**Description**
31:24	RO	-	**Reserved (0)**
23:7	RW	Rst17	**Video compression stream buffer process offset** The video compression stream data processing address which tell the stream interrupt controller the data have been accepted by ISR, but is still occupied the buffer.
6 :0	RO	-	**Reserved (0)**

VR03C: Video Compression Stream Buffer Read Offset Register			
Offset: 03Ch			**Init = 0**
Bit	**Attr.**	**Reset**	**Description**
31:24	RO	-	**Reserved (0)**
23:7	RW	Rst17	**Video compression stream buffer read offset** The video compression stream data read pointer which will notify the stream buffer controller the current software read pointer.
6 :0	RO	-	**Reserved (0)**

VR040: Video Base Address of JPEG Header Buffer Register when VR004[13]=1			
Offset: 040h			**Init = X**
Bit	**Attr.**	**Reset**	**Description**
31:30	RO	-	**Reserved (0)**

to next page

from previous page

29:4	RW	-	**JPEG header buffer base address Bit [29:4] when JPEG mode enabled** The software should prepare a JPEG header buffer which contains 12 JPEG headers. The 12 headers corresponds to the DCT luminance quantization table VR060[15:11]. The offset between two headers is 1KB. The real header size is defined in the header, and the hardware will check the header size automatically. The address bit [3:0] should always be 0.
3 :0	RO	-	**Reserved (0)**

VR044: Video Based Address of Video Source Buffer #1 Register

Offset: 044h **Init = X**

Bit	Attr.	Reset	Description
31	RO	-	**Reserved (0)**
31:8	RW	-	**Base address of video source buffer #1 Bit [29:8]** In order to support the double-buffer mode for video capturing, two video source buffers, allocated from SDRAM memory, are required. This register determines the base address of the first video source buffer. The address bit [7:0] should be 0.
7 :0	RO	-	**Reserved (0)**

VR048: Video Scan Line Offset of Video Source Buffer Register

Offset: 048h **Init = X**

Bit	Attr.	Reset	Description
31:15	RO	-	**Reserved (0)**
14:4	RW	-	**Scan line offset of the video source buffer Bit [14:4]** This register determines the scan line offset (memory address distance) of video source buffer #1 and buffer #2 from one scan line to the next scan line. The address bit [3:0] should be 0.
3 :0	RO	-	**Reserved (0)**

Note :
This is the offset address from line to line. The address bit [3:0] should be 0. The buffer offset 0 can be calculated from the horizontal total pixel number * 4bpp. The horizontal pixel number must be a multiplier of 8. If the real pixel number is not a multiplier of 8, please select the smallest number which is a multiplier of 8 and greater than the pixel number.

VR04C: Video Base Address of Video Source Buffer #2 Register

Offset: 04Ch **Init = X**

Bit	Attr.	Reset	Description
31	RO	-	**Reserved (0)**
30:8	RW	-	**Base address of video source buffer #2 Bit [29:8]** In order to support the double-buffer mode for video capturing, two video source buffers, allocated from SDRAM memory, are required. This register determines the base address of the second video source buffer. The address bit [7:0] should be 0.
7 :0	RO	-	**Reserved (0)**

VR050: Video Base Address of BCD Flag Buffer Register			
Offset: 050h			**Init = X**
Bit	**Attr.**	**Reset**	**Description**
31	RO	-	**Reserved (0)**
30:4	RW	-	**Base address of BCD flag buffer Bit [29:4]** BCD flag buffer, allocated from SDRAM memory, records the BCD flag for each block. It requires 4 bits per block to store the necessary status information. BCD flag buffer needs to be initialized for the first time. This register determines the base address of BCD flag buffer. The address bit [3:0] should be 0.
3 :0	RO	-	**Reserved (0)**

VR054: Video Base Address of Compressed Video Stream Buffer Register			
Offset: 054h			**Init = X**
Bit	**Attr.**	**Reset**	**Description**
31	RO	-	**Reserved (0)**
29:7	RW	-	**Base address of compressed video stream buffer Bit [29:7]** The register determines the base address of the video buffer for storing compressed video stream. The address bit [6:0] should be 0.
6 :0	RO	-	**Reserved (0)**

VR058: Video Stream Buffer Size Register			
Offset: 058h			**Init = X**
Bit	**Attr.**	**Reset**	**Description**
31:25	RO	-	**Reserved (0)**
24:16	RW	-	**Reserved**
15:11	RO	-	**Reserved (0)**
10:6	RW	-	**Reserved**
5 :3	RW	-	**Stream buffer packet number** 000: 4 packets 001: 8 packets 010: 16 packets 011: 32 packets 100: 64 packets 101: 128 packets others: Reserved
2 :0	RW	-	**Stream buffer packet size** 000: 1 KB 001: 2 KB 010: 4 KB 011: 8 KB 100: 16 KB 101: 32 KB 110: 64 KB 111: 128 KB

VR05C: Video Compression Stream Buffer Write Offset Read Back			
Offset: 05Ch			**Init = 0**
Bit	**Attr.**	**Reset**	**Description**
31:24	RO	-	**Reserved (0)**
23:7	RO	Rst17	**Video compression stream buffer write offset** This read back register can return the offset address of current video write offset address for compressed data in the stream buffer.
6 :0	RO	-	**Reserved (0)**

VR060: Video Compression Control Register			
Offset: 060h			**Init = 0**
Bit	**Attr.**	**Reset**	**Description**
31:27	RW	Rst17	**High quality DCT luminance quantization table (VR060[2]=0)** This register determines how the DCT engine executes DCT quantization for the luminance DCT coefficients of each block. There are 12 DCT luminance quantization tables and 12 DCT chrominance quantization tables implemented in the DCT engine. The first bit of this register (VR060[31]) determines whether the 12 DCT luminance quantization tables or the 12 DCT chrominance quantization tables will be referred. 0: Select one of the 12 DCT luminance quantization tables (Table #0 ~ Table #11) 1: Select one of the 12 DCT chrominance quantization tables (Table #0 ~ Table #11) The left 4 bits of this register (VR060[30:27]) will determine which one of the selected 12 tables will be used for quantizing the luminance DCT coefficients. 0000: Table #0 0001: Table #1 ... 1011: Table #11 Others: Invalid
26:22	RW	Rst17	**High quality DCT chrominance quantization table (VR060[2]=0)** This register determines how the DCT engine executes DCT quantization for the luminance DCT coefficients of each block. There are 12 DCT luminance quantization tables and 12 DCT chrominance quantization tables implemented in the DCT engine. The first bit of this register (VR060[26]) determines whether the 12 DCT luminance quantization tables or the 12 DCT chrominance quantization tables will be referred. 0: Select one of the 12 DCT luminance quantization tables (Table #0 ~ Table #11) 1: Select one of the 12 DCT chrominance quantization tables (Table #0 ~ Table #11) The left 4 bits of this register (VR060[25:22]) will determine which one of the selected 12 tables will be used for quantizing the luminance DCT coefficients. 0000: Table #0 0001: Table #1 ... 1011: Table #11 Others: Invalid
21:20	RW	Rst17	**DCT Huffman encoding table selection** 00: Select Y and UV tables 01: Select Y table only 1x: Select UV table only In most of the cases, ”00” is recommended.

to next page

from previous page

19	RW	Rst17	**Reserved(1)** This register must be always ”1”.
18:17	RW	Rst17	**DCT engine hardware test control** For testing hardware only. Writing 0 for a normal condition.
16	RW	Rst17	**Enable video high quality mode** 0: Disable 1: Enable when ASPEED proprietary compression mode only This register can be enabled for both YUV420 and YUV444 mode.
15:11	RW	Rst17	**DCT luminance quantization table selection** This register determines how DCT engine executes DCT quantization for the luminance DCT coefficients of each block. There are 12 DCT luminance quantization tables and 12 DCT chrominance quantization tables implemented in DCT engine. The first bit of this register (VR060[15]) determines whether the 12 DCT luminance quantization tables or the 12 DCT chrominance quantization tables will be referenced. 0: Select one of the 12 DCT luminance quantization tables (Table #0 ∼ Table #11) 1: Select one of the 12 DCT chrominance quantization tables (Table #0 ∼ Table #11) The left 4 bits of this register (VR060[14:11]) will determine which one of the selected 12 tables will be used for quantizing the luminance DCT coefficients. 0000: Table #0 0001: Table #1 ... 1011: Table #11 Others: Invalid These tables can be applied to both YUV420 and YUV444 video compression. VR060[15:11] must be 0 when VR300[18]=1
10:6	RW	Rst17	**DCT chrominance quantization table selection** This register determines how DCT engine executes DCT quantization for the chrominance DCT coefficients of each block. There are 12 DCT luminance quantization tables and 12 DCT chrominance quantization tables implemented in DCT engine. The first bit of this register (VR060[10]) determines whether the 12 DCT luminance quantization tables or the 12 DCT chrominance quantization tables will be referenced. 0: Select one of the 12 DCT luminance quantization tables (Table #0 ∼ Table #11) 1: Select one of the 12 DCT chrominance quantization tables (Table #0 ∼ Table #11) The left 4 bits of this register (VR060[9:6]) will determine which one of the selected 12 tables will be used for quantizing the chrominance DCT coefficients. 0000: Table #0 0001: Table #1 ... 1011: Table #11 Others: Invalid These tables can be applied to both YUV420 and YUV444 video compression.

to next page

from previous page

5	RW	Rst17	**Enable RC4/AES128 encryption** 0: Disable 1: Enable This register will determine if the compressed video stream will be RC4/AES128-encrypted or not. When enabled, video engine will encrypt the compressed video stream before writing into compressed video stream buffer.
4	RW	Rst17	**Enable video best quality mode** 0: Disable 1: Enable when ASPEED proprietary compression mode only This register can be enabled for both YUV420 and YUV444 mode.
3	RW	Rst17	**Reserved(0)** This register must be always ”0”.
2	RW	Rst17	**Compression/decompression mode setting for high video quality** 0: DCT mode 1: Quantization mode
1	RW	Rst17	**Reserved(0)** This register must be always ”0”.
0	RW	Rst17	**Reserved(1)** This register can be ”0” or ”1”.

VR064: Video JPEG effecive bit control Register

Offset: 064h **Init = 0**

Bit	Attr.	Reset	Description
31:7	RO	-	**Reserved (0)**
6	RW	Rst17	**JPEG chrominance dither mode** 0 Disable dither 1 Enable dithe
5	RW	Rst17	**JPEG luminance dither mode** 0 Disable dither 1 Enable dithe
4	RW	Rst17	**Reserved (0)**
3 :2	RW	Rst17	**Effective JPEG chrominance bits** 00: 8-bit 01: 7-bit 10: 6-bit 11: 5-bit
1 :0	RW	Rst17	**Effective JPEG luminance bits** 00: 8-bit 01: 7-bit 10: 6-bit 11: 5-bit
1 :0	RW	Rst17	**Reserved (0)**

VR068: Video Quantization value

Offset: 068h **Init = X**

Bit	Attr.	Reset	Description
31	RO	-	**Reserved (0)**

to next page

from previous page

30:16	RW	-	**High quality mode quantization value** This register determines the compression engine quantize video data before the video data are compressed. Its format is 1.14. One bit integer and 14 bits fraction parts. This register value must be the reciprocal of VR060[26:22] in decompression side.
15	RO	-	**Reserved (0)**
14:0	RW	-	**Best quality mode quantization value** This register determines the compression engine quantize video data before the video data are compressed. Its format is 1.14. One bit integer and 14 bits fraction parts. This register value must be the reciprocal of VR060[31:27] in decompression side.

VR06C: Video Copy Buffer Base Address when VR02C[2]=1

Offset: 06Ch **Init = X**

Bit	Attr.	Reset	Description
31	RO	-	**Reserved (0)**
30:4	RW	-	**Video copy buffer base address[29:4]** The copy buffer can be used to store the first compressed data. It can be used for more accurate block change detection when encode mode.
3 :0	RO	-	**Reserved (0)**

VR070: Video Total Size of Compressed Video Stream Read Back Register

Offset: 070h **Init = 0**

Bit	Attr.	Reset	Description
31:22	RO	-	**Reserved (0)**
21:0	RO	Rst17	**Total size of compressed video stream** This register reports the total length of compressed video stream already stored in the compressed video stream buffer for a video frame. The unit is one double word.

VR074: Video Total Number of Compressed Video Blocks Read Back Register

Offset: 074h **Init = X**

Bit	Attr.	Reset	Description
31:16	RO	-	**Compressed block counter read back (number of blocks)** This register reports the number of video blocks have been compressed into the video compressed stream buffer for a video frame.
15:0	RO	-	**Processed total block counter read back (number of blocks)** This register reports the total number of video blocks have been processed by video engine
Note : This register is applicable at YUV420 mode is 16 pixels x16 pixels block size.			

VR078: Video Frame-End Offset of Compressed Video Stream Buffer Read Back Register			
Offset: 078h			**Init = 0**
Bit	**Attr.**	**Reset**	**Description**
31:24	RO	-	**Reserved (0)**
23:4	RO	Rst17	**Frame-end offset of compressed video stream buffer Bit [23:4]** This register reports the frame-end offset of the compressed video stream buffer. Adding the value of this register with the base address of the compressed video stream buffer (VR054) can derive the last address of the last video stream data for the last compressed frame. The bit [3:0] should be 0.
3 :0	RO	-	**Reserved (0)**

VR07C: Video Compressed Frame Counter Read Back Register			
Offset: 07Ch			**Init = 0**
Bit	**Attr.**	**Reset**	**Description**
31:0	RO	Rst17	**Compressed frame counter Bit [31:0]** This register reports the value of the frame counter which is designed to count the number of compressed frames up to the read back moment. This register is supported during multi-frame mode only.

VR080: Video User Defined Header Parameter Setting Register when Compression			
Offset: 080h			**Init = X**
Bit	**Attr.**	**Reset**	**Description**
31:16	RO	-	**Reserved (0)**
15:0	RW	-	**User defined header parameter[15:0]** This register data will be inserted into every stream frame header and is supported for ASPEED proprietary compression mode only.

VR084: JPEG Compressed Size Read Back Register			
Offset: 084h			**Init = 0**
Bit	**Attr.**	**Reset**	**Description**
31:0	RO	Rst17	**JPEG Compressed Size Read Back[31:0]** This register reports the size of the JPEG compression result. The size is accurate to byte.

Offset: 088h	**VR088: SCU Free Run Counter Read Back**		**Init = 0**
Offset: 08Ch	**VR08C: SCU Free Run Counter Extended Read Back**		**Init = 0**
Bit	**Attr.**	**Reset**	**Description**
63:48	RO	-	**Reserved (0)**
47:0	RO	RstPwr	**SCU free run counter bit[47:0] read back** The SCU free run counter is a 48-bit couter. Its value is reset by SRST# signal. The counter tick is 25MHz. The counter read back is for reference since last SRST#. It is not guaranteed to be glitch free when readback.

VR090: Video Source Left/Right Edge Detection Read Back Register

Offset: 090h **Init = X**

Bit	Attr.	Reset	Description
31	RO	-	**Video interlace mode detected** 0: Progressive mode detected 1: Interlace mode detected
30:28	RO	-	**Reserved (0)**
27:16	RO	-	**Video source right edge location from the rising edge of HSYNC Bit [11:0]** The unit of this register is one pixel.
15	RO	-	**No display clock detected** 0: No display clock detected 1: Display clock detected
14	RO	-	**No active display detected** 0: No active display detected 1: Active display detected
13	RO	-	**No HSYNC detected** 0: HSYNC detected 1: No HSYNC detected
12	RO	-	**No VSYNC detected** 0: VSYNC detected 1: No VSYNC detected
11:0	RO	-	**Video source left edge location from the rising edge of HSYNC Bit [11:0]** The unit of this register is one pixel.

Note :
This register is primarily designed to detect the video timing of an external video source. If the video source is from the internal VGA controller, then the video timing can directly be from the information, recorded in VGA scratch registers, provided by the VGA BIOS.

VR094: Video Source Top/Bottom Edge Detection Read Back Register

Offset: 094h **Init = X**

Bit	Attr.	Reset	Description
31:29	RO	-	**Reserved (0)**
28:16	RO	-	**Video source bottom edge location from the rising edge of VSYNC Bit [12:0]** The unit of this register is one scan line.
15:13	RO	-	**Reserved (0)**
12:0	RO	-	**Video source top edge location from the rising edge of VSYNC Bit [12:0]** The unit of this register is one scan line.

Note :
This register is primarily designed to detect the video timing of an external video source. If the video source is from the internal VGA controller, then the video timing can directly be from the information, recorded in VGA scratch registers, provided by the VGA BIOS.

VR098: Video Mode Detection Status Read Back Register			
Offset: 098h			**Init = X**
Bit	**Attr.**	**Reset**	**Description**
31	RO	-	**Mode detection HSYNC ready** 0: HSYNC is not yet ready 1: HSYNC is ready HSYNC being ready only means that the signal has been detected, but not necessary stable.
30	RO	-	**Mode detection VSYNC ready** 0: VSYNC is not yet ready 1: VSYNC is ready VSYNC being ready only means that the signal has been detected, but not necessary stable.
29	RO	-	**Mode detection HSYNC polarity** 0: Source HSYNC polarity is positive 1: Source HSYNC polarity is negative
28	RO	-	**Mode detection VSYNC polarity** 0: Source VSYNC polarity is positive 1: Source VSYNC polarity is negative
27:16	RO	-	**Mode detection vertical scan lines Bit [11:0]** This register reports the number of scan lines detected between two continuous VSYNC. When there is no VSYNC signal detected, all the bits of this register will be ”1”.
15	RO	-	**Video source is out of synchronization** 0: Video source is still stable 1: Video source is out of synchronization This status register, designed to report any mode changes, is effective only when mode detection watchdog is enabled (VR004[7] = 1). Whenever video source is out of synchronization, S/W needs to trig mode detection again.
14	RO	-	**Mode detection vertical signal stable** 0: Vertical signal detection is not stable 1: Vertical signal detection is stable
13	RO	-	**Mode detection horizontal signal stable** 0: Horizontal signal detection is not stable 1: Horizontal signal detection is stable
12	RO	-	**Auto detection of external digital video source type** 0: Video source is from DVI receiver 1: Video source is from ADC output The major difference is the video source from DVI receiver goes with Display Enable signal, the video source from ADC output goes without Display Enable signal.
11:0	RO	-	**Mode detection horizontal period Bit [11:0]** This register reports the period of the detected HSYNC signal after horizontal mode detection is stable (VR098[13] = 1). If there is no HSYNC signal detected, all bits of this register will be 1. The measurement clock is 25 MHz.

VR09C: Video Sync End Location Detection Read Back Register			
Offset: 09Ch			**Init = X**
Bit	**Attr.**	**Reset**	**Description**
31:28	RO	-	**Reserved (0)**

to next page

from previous page

27:16	RO	-	**Video falling edge location of VSYNC from the rising edge of VSYNC Bit [11:0]** The unit of this register is scan line.
15:12	RO	-	**Reserved (0)**
11:0	RO	-	**Video falling edge location of HSYNC from the rising edge of HSYNC Bit [11:0]** The unit of this register is pixel.
Note : This register is primarily designed to detect the video timing of an external video source. If the video source is from the internal VGA controller, then the video timing can directly be from the information, recorded in VGA scratch registers, provided by the VGA BIOS.			

VR0A0: Video Horizontal Total Pixel Count Read Back Register

Offset: 0A0h **Init = X**

Bit	Attr.	Reset	Description
31:12	RO	-	**Reserved (0)**
11:0	RO	-	**Video horizontal total pixel counts Bit [11:0]** The unit of this register is pixel.
Note : This register is primarily designed to detect the video timing of an external video source. If the video source is from the internal VGA controller, then the video timing can directly be from the information, recorded in VGA scratch registers, provided by the VGA BIOS.			

VR0A4: Extended Mode Detection Control Register 1

Offset: 0A4h **Init = 0**

Bit	Attr.	Reset	Description
31	RW	Rst17	**Video Interlace Signal Generated by Internal Timing Generator** 0: Disable 1: Enable
30	RW	Rst17	**Video source ODD/EVEN polarity selection** 0: Internal ODD/EVEN polarity is same as source ODD/EVEN 1: Internal ODD/EVEN polarity is inversed of source ODD/EVEN
29	RW	Rst17	**Long Horizontal Stable Time Detecion Mode** 0: Disable 1: Enable for more stable mode detection
28:24	RW	Rst17	**Internal Timing Generator Horizontal Sync Width Setting** The unit of this register is in unit of pixel clock cycle.
23:16	RW	Rst17	**Internal Timing Generator Horizontal Sync Delay Setting** The unit of this register is in unit of display clock cycle.
15:8	RW	Rst17	**Mode Detection Right Margin Setting** The unit of this register is in unit of display clock cycle.
7 :0	RW	Rst17	**Mode Detection Left Margin Setting** The unit of this register is in unit of display clock cycle.

VR0A8: Extended Video Control Register 1			
Offset: 0A8h			**Init = 0**
Bit	**Attr.**	**Reset**	**Description**
31	RW	Rst17	**Pop a frame from flip queue** 0 to 1: Pop a frame from flip queue when value changes from 0 to 1 1 to 0: Pop a frame from flip queue when value changes from 1 to 0
30	RW	Rst17	**Reserved (0)**
29	RW	Rst17	**Disable flipping deflicker during switching between YUV444 and YUV420** 0: Enable 1: Disable
28	RW	Rst17	**Video input port YUV dual edge mode** 0: Disable 1: Enable
27	RW	Rst17	**Video input port U/V pin swap** 0: Disable 1: Enable
26	RW	Rst17	**Decode Double Buffer mode.** 0: Disable 1: Enable Enable double buffer mode when decode mode. It can cause more memory copy and avoid display artifact when encode and decode video is not synchronized. The base address of second buffer is assigned in VR04C
25			**Reserved**
24	RW	Rst17	**Mask Video Compression Data Output.** 0: normal 1: all compressed data are masked For VGA auto-phase detection, this bit can avoid unwanted data writing to stream buffer.
21	RW	Rst17	**Enable Deflicker for Scaled CRT display (Client)** 0: Disable 1: Enable The setting include following registers: VR010[9:0]: 512 / (CRT scaling factor) VR010[28:16]: Target display starting line number.
20:14	RW	Rst17	**Reserved (0)**
13	RW	Rst17	**Disable Direct Mode Frame Control** 0: Enable 1: Disable The setting may let the direct mode frame rate to be higher than 60.
12	RW	Rst17	**Enable Direct Mode Virtual Timing Generator Programmable** 0: Disable 1: Enable This bit allows frame mode capturing to reach 60 fps.
11	RW	Rst17	**Enable Fast video capture ready mode** 0: Disable 1: Enable register VR0AC and VR0B0
10:5	RW	Rst17	**Mode Detection HSYNC De-glitch Setting** The unit of this register is in unit of display clock cycle.

to next page

from previous page

4 :0	RW	Rst17	**Internal Timing Generator Vertical Sync Delay Setting** The unit of this register is in unit of line.

VR0AC: Direct Access Mode Virtual Timing Setting Register 1

Offset: 0ACh **Init = 0**

Bit	Attr.	Reset	Description
31:24	RW	Rst17	**Direct Access Mode Virtual Vertical Back Porch Width Timing** The unit of this register is in unit of display clock cycle.
23:16	RW	Rst17	**Direct Access Mode Virtual Horizontal Back Porch Width Timing** The unit of this register is in unit of display clock cycle.
15:8	RW	Rst17	**Direct Access Mode Virtual Horizontal Front Porch Width Timing** The unit of this register is in unit of display clock cycle.
7 :0	RW	Rst17	**Direct Access Mode Virtual Horizontal Sync Width Timing** The unit of this register is in unit of line.

VR0B0: Direct Access Mode Virtual Timing Setting Register 2

Offset: 0B0h **Init = 0**

Bit	Attr.	Reset	Description
31:24	RW	Rst17	**Direct Access Mode Frame Time Control** The unit of this register is in unit of 64us.
23:16	RW	Rst17	**Reserved (0)**
15:8	RW	Rst17	**Direct Access Mode Virtual Vertical Front Porch Width Timing** The unit of this register is in unit of line.
7 :0	RW	Rst17	**Direct Access Mode Virtual Vertical Sync Width Timing** The unit of this register is in unit of line.

VR0B4: Extended Video Control Register 2

Offset: 0B4h **Init = 0x288**

Bit	Attr.	Reset	Description
31:24	RO	-	**Reserved (0)**
23:12	RW	Rst17	**Reserved (0)**
11:7	RW	Rst17	**JPEG JFIF header size** The size is in unit of 128-byte. The JPEG header located in VR040. This field must set the same value as VR0B4[6:2] when using desc-mode+VR004[15].
6 :2	RW	Rst17	**Big software frame header size** The size is in unit of 128-byte. The software header located in VR040. The header MUST reserve first 16 bytes for VE hardware frame header.
1	RW	Rst17	**Enable big software frame header** 0: Disable 1: Enable It is used for software header and not compatible to old stream.
0	RW	Rst17	**Enable long block header** 0: Disable 1: Enable It is used for 4Kx2K mode and not compatible to old stream.

VR0B8: Video Source Left/Right Edge Detection 4K2K Read Back Register (re-mapping from VR090)			
Offset: 0B8h			**Init = X**
Bit	**Attr.**	**Reset**	**Description**
31:29	RO	-	**Reserved (0)**
28:16	RO	-	**Video source right edge location from the rising edge of HSYNC Bit [12:0]**
15:13	RO	-	**Reserved (0)**
12:0	RO	-	**Video source left edge location from the rising edge of HSYNC Bit [12:0]**

VR0BC: Video Mode Detection Status 4K2K Read Back Register (re-mapping from VR098)			
Offset: 0BCh			**Init = X**
Bit	**Attr.**	**Reset**	**Description**
31:29	RO	-	**Reserved (0)**
28:16	RW	-	**Video mode detection vertical counter Bit [12:0]** read: After mode detection is ready, the read back value is the detected VSYNC period which is in unit of line. write: The write value will initialize the mode detection watch dog VSYNC period which is in unit of line. Only double word write access is allowed. After the value is written, the software need to read the register and make sure the value is written successfully.
15:13	RO	-	**Reserved (0)**
12:0	RW	-	**Video mode detection horizontal counter Bit [12:0]** read: After mode detection is ready, the read back value is the detected VSYNC period which is in unit of line. write: The write value will initialize the mode detection watch dog VSYNC period which is in unit of line. Only double word write access is allowed. After the value is written, the software need to read the register and make sure the value is written successfully.

VR0C0: Video Engine Status 1 Read Back Register			
Offset: 0C0h			**Init = X**
Bit	**Attr.**	**Reset**	**Description**
31:10	RO	Rst17	**Video Engine Status** Video engine status read back. Reserved for debugging.
9:0	RO	Rst17	**Video Engine State Bits [9:0]** Video engine state read back. Reserved for debugging.

VR0C4: Video Engine Status 2 Read Back Register			
Offset: 0C4h			**Init = X**
Bit	**Attr.**	**Reset**	**Description**
31:0	RO	Rst17	**Video Engine Status** Video engine status read back. Reserved for debugging.

VR0C8: Video Engine Status 3 Read Back Register			
Offset: 0C8h			**Init = X**
Bit	**Attr.**	**Reset**	**Description**
31:0	RO	Rst17	**Video Engine Status** Video engine status read back. Reserved for debugging.

VR0CC: Video Engine Status 4 Read Back Register			
Offset: 0CCh			**Init = X**
Bit	**Attr.**	**Reset**	**Description**
31:0	RO	Rst17	**Video Engine Status** Video engine status read back. Reserved for debugging.

VR0D0: Extra Video Mode Detection Status Read Back Register			
Offset: 0D0h			**Init = X**
Bit	**Attr.**	**Reset**	**Description**
31:22	RO	-	**Reserved (0)**
21:0	RO	-	**Mode detection horizontal period Bit [21:0]** This register reports the period of the detected VSYNC signal after vertical mode detection is stable (VR098[14] = 1). If there is no VSYNC signal detected, all bits of this register will be 1. The measurement clock is 25 MHz. This is useful to distinguish 59.94Hz and 60Hz VSYNC sources.

VR0D4: Video BCD Mode Bounding Box Detection Read Back Register 1			
Offset: 0D4h			**Init = X**
Bit	**Attr.**	**Reset**	**Description**
31:26	RO	Rst17	**Reserved (0)**
25:16	RO	Rst17	**Block change bounding box right edge** This register reports the right edge of bounding box of the result of block change detection (BCD). It is the coordinate of macro block.
15:10	RO	-	**Reserved (0)**
9:0	RO	Rst17	**Block change bounding box left edge** This register reports the left edge of bounding box of the result of block change detection (BCD). It is the coordinate of macro block.

VR0D8: Video BCD Mode Bounding Box Detection Read Back Register 2			
Offset: 0D4h			**Init = X**
Bit	**Attr.**	**Reset**	**Description**
31:26	RO	Rst17	**Reserved (0)**
25:16	RO	Rst17	**Block change bounding box bottom edge** This register reports the bottom edge of bounding box of the result of block change detection (BCD). It is the coordinate of macro block.
15:10	RO	-	**Reserved (0)**
9:0	RO	Rst17	**Block change bounding box top edge** This register reports the top edge of bounding box of the result of block change detection (BCD). It is the coordinate of macro block.

VR204: Video Management Sequence Control Register			
Offset: 204h			**Init = 0**
Bit	**Attr.**	**Reset**	**Description**
31:19	RO	-	**Reserved (0)**
18	RO	Rst17	**Video management compression status** 0: Video management compression is in busy state 1: Video management compression is in idle state
17	RO	-	**Reserved (0)**
16	RO	Rst17	**Video management capture status** 0: Video management capture is in busy state 1: Video management capture is in idle state
15:12	RW	Rst17	**Reserved** Thess registers must always be ”0”
11:10	RW	Rst17	**Video data format conversion for video management compression** 00: YUV444 01: YUV420 10: Reserved 11: Reserved Video capture engine always converts the input RGB data stream to YUV444 data format and write out to the video memory, and this register can select the expected video data format that the video compression engine has to convert first before doing video compression.
9	RW	Rst17	**Reserved** This register must be always ”0”
8	RW	Rst17	**Enable JPEG compatible mode management compression** 0: ASPEED proprietary compression mode 1: Enable JPEG compatible mode
7:5	RW	Rst17	**Reserved** This register must be always ”0”
4	RW	Rst17	**Enable or trigger video management compression** 0: No operation 0→1: Trigger management video compression 1: Enable video compression Setting this register from 0 to 1 will trigger the video compression engine to compress the captured management video data stored in the frame buffer. Video management compression can compress one single frame.
3:2	RW	Rst17	**Reserved** This register must be always ”0”
1	RW	Rst17	**Enable or trigger video management capture** 0: No operation 0→1: Trigger management video capture Setting this register from 0 to 1 will trigger the video capture engine to capture either single management video frame. The software must insert at least one read cycle or 1us delay time between continuous trigger register setting.
0	RW	Rst17	**Reserved** This register must be always ”0”

VR208: Video Management Control Register			
Offset: 208h			**Init = 0**
Bit	**Attr.**	**Reset**	**Description**
31:8	RO	-	**Reserved (0)**
7 :6	RW	Rst17	**Data format for video capture** 00: CCIR601-2 compliant YUV format for ASPEED proprietary compression mode JPEG compliant YUV format for JPEG mode 01: JPEG compliant YUV format for both JPEG mode and ASPEED proprietary compression mode 10: RGB format for ASPEED proprietary compression mode only 11: Gray color mode for both JPEG mode and ASPEED proprietary compression mode AST2600 supports two kinds of YUV formats. One is CCIR601-2 compatible; the other is in a proprietary format to support higher video quality. RGB format is for debugging purpose only.
5 :1	RO	-	**Reserved (0)**
0	RW	Rst17	**Video management auto source register** 0: VR244 is required for management compression 1: VR244 is not required for management compression

VR238: Video Management Compression Stream Buffer Processing Offset Register			
Offset: 238h			**Init = 0**
Bit	**Attr.**	**Reset**	**Description**
31:24	RO	-	**Reserved (0)**
23:7	RW	Rst17	**Video compression stream buffer process offset** The video compression stream data processing address which tell the stream interrupt controller the data have been accepted by ISR, but is still occupied the buffer.
6 :0	RO	-	**Reserved (0)**

VR23C: Video Management Compression Stream Buffer Read Offset Register			
Offset: 23Ch			**Init = 0**
Bit	**Attr.**	**Reset**	**Description**
31:24	RO	-	**Reserved (0)**
23:7	RW	Rst17	**Video compression stream buffer read offset** The video compression stream data read pointer which will notify the stream buffer controller the current software read pointer.
6 :0	RO	-	**Reserved (0)**

VR244: Video Management Based Address of Video Source Buffer Register			
Offset: 244h			**Init = X**
Bit	**Attr.**	**Reset**	**Description**
31:28	RO	-	**Reserved (0)**
27:8	RW	-	**Base address of video management source buffer #1 Bit [27:8]** In order to support the management video compression, this address is the base address for video data which will be compressed. Normally this register will be the same as the base address in VR044 or VR04C. The address bit [7:0] should be 0.

to next page

from previous page

7 :0	RO	-	**Reserved (0)**

VR248: Video Management Scan Line Offset of Video Source Buffer Register

Offset: 248h **Init = X**

Bit	Attr.	Reset	Description
31:15	RO	-	**Reserved (0)**
14:4	RW	-	**Scan line offset of the video source buffer Bit [14:4]** This register determines the scan line offset (memory address distance) of video source buffer #1 and buffer #2 from one scan line to the next scan line. The address bit [3:0] should be 0.
3 :0	RO	-	**Reserved (0)**

Note :
This is the offset address from line to line. The address bit [3:0] should be 0. The buffer offset 0 can be calculated from the horizontal total pixel number * 4bpp. The horizontal pixel number must be a multiplier of 8. If the real pixel number is not a multiplier of 8, please select the smallest number which is a multiplier of 8 and greater than the pixel number.

VR24C: Video Management Base Address of Video Source Buffer #2 Register

Offset: 24Ch **Init = X**

Bit	Attr.	Reset	Description
31	RO	-	**Reserved (0)**
30:8	RW	-	**Base address of video source buffer #2 Bit [29:8]** In order to support the double-buffer mode for video capturing, two video source buffers, allocated from SDRAM memory, are required. This register determines the base address of the second video source buffer. The address bit [7:0] should be 0.
7 :0	RO	-	**Reserved (0)**

VR250: Video Management Base Address of BCD Flag Buffer Register

Offset: 250h **Init = X**

Bit	Attr.	Reset	Description
31	RO	-	**Reserved (0)**
30:4	RW	-	**Base address of BCD flag buffer Bit [29:4]** BCD flag buffer, allocated from SDRAM memory, records the BCD flag for each block. It requires 4 bits per block to store the necessary status information. BCD flag buffer needs to be initialized for the first time. This register determines the base address of BCD flag buffer. The address bit [3:0] should be 0.
3 :0	RO	-	**Reserved (0)**

VR254: Video Management Base Address of Compressed Video Buffer Register

Offset: 254h **Init = X**

Bit	Attr.	Reset	Description
31:28	RO	-	**Reserved (0)**
27:7	RW	-	**Base address of compressed management video stream buffer Bit [27:3]** The register determines the base address of the video buffer for storing compressed management video stream. The address bit [6:0] should be 0.

to next page

from previous page

6 :0	RO	-	**Reserved (0)**

VR260: Video Management Compression or Video Profile 2-5 Decompression Control Register

Offset: 260h **Init = 0**

Bit	Attr.	Reset	Description
31:16	RW	Rst17	**Reserved (0)**
15:11	RW	Rst17	**DCT luminance quantization table selection** This register determines how the DCT engine executes DCT quantization for the luminance DCT coefficients of each block. There are 12 DCT luminance quantization tables and 12 DCT chrominance quantization tables implemented in DCT engine. The first bit of this register (VR260[15]) determines whether the 12 DCT luminance quantization tables or the 12 DCT chrominance quantization tables will be referred. 0: Select one of the 12 DCT luminance quantization tables (Table #0 ~ Table #11) 1: Select one of the 12 DCT chrominance quantization tables (Table #0 ~ Table #11) The left 4 bits of this register (VR260[14:11]) will determine which one of the selected 12 tables will be used for quantizing the luminance DCT coefficients. 0000: Table #0 0001: Table #1 ... 1011: Table #11 Others: Invalid These tables can be applied to both YUV420 and YUV444 video compression.
10:6	RW	Rst17	**DCT chrominance quantization table selection** This register determines how the DCT engine executes DCT quantization for the chrominance DCT coefficients of each block. There are 12 DCT luminance quantization tables and 12 DCT chrominance quantization tables implemented in DCT engine. The first bit of this register (VR260[10]) determines whether the 12 DCT luminance quantization tables or the 12 DCT chrominance quantization tables will be referred. 0: Select one of the 12 DCT luminance quantization tables (Table #0 ~ Table #11) 1: Select one of the 12 DCT chrominance quantization tables (Table #0 ~ Table #11) The left 4 bits of this register (VR260[9:6]) will determine which one of the selected 12 tables will be used for quantizing the chrominance DCT coefficients. 0000: Table #0 0001: Table #1 ... 1011: Table #11 Others: Invalid These tables can be applied to both YUV420 and YUV444 video compression.
5 :2	RW	Rst17	**Reserved** This register must be always ”0”.
1	RW	Rst17	**Reserved** This register must be always ”0”.
0	RW	Rst17	**Reserved** This register can be ”0” or ”1”.

VR268: Video Management Quantization value			
Offset: 268h			**Init = X**
Bit	**Attr.**	**Reset**	**Description**
31	RO	-	**Reserved (0)**
30:16	RW	-	**High quality mode quantization value** This register determines the compression engine quantize video data before the video data are compressed. Its format is 1.14. One bit integer and 14 bits fraction parts. This register value must be the reciprocal of VR060[26:22] in decompression side.
15	RO	-	**Reserved (0)**
14:0	RW	-	**Best quality mode quantization value** This register determines the compression engine quantize video data before the video data are compressed. Its format is 1.14. One bit integer and 14 bits fraction parts. This register value must be the reciprocal of VR060[31:27] in decompression side.

VR270: Video Management Total Size of Compressed Video Stream Read Back Register			
Offset: 270h			**Init = 0**
Bit	**Attr.**	**Reset**	**Description**
31:22	RO	-	**Reserved (0)**
21:0	RO	Rst17	**Total size of compressed video stream** This register reports the total length of compressed video stream already stored in the compressed video stream buffer for a video frame. The unit is one double word.

VR278: Video Management Frame-End Offset of Compressed Video Stream Buffer Read Back Register			
Offset: 278h			**Init = X**
Bit	**Attr.**	**Reset**	**Description**
31:22	RO	-	**Reserved (0)**
21:3	RO	Rst17	**Frame-end offset of compressed video stream buffer Bit [21:3]** This register reports the frame-end offset of the compressed video stream buffer. Adding the value of this register with the base address of the compressed video stream buffer (VR054) can derive the last address of the last video stream data for the last compressed frame. The bit [2:0] should be 0.
2 :0	RO	-	**Reserved (0)**

VR280: Video Management User Defined Header Parameter Setting Register when Compression			
Offset: 280h			**Init = X**
Bit	**Attr.**	**Reset**	**Description**
31:16	RO	-	**Reserved (0)**
15:0	RW	-	**User defined header parameter[15:0]** This register data will be inserted into every stream frame header and is supported for ASPEED proprietary compression mode only.

VR300: Video Control Register			
Offset: 300h			**Init = 0**
Bit	**Attr.**	**Reset**	**Description**
31:30	RW	Rst17	**VR400-VR5FC Address mapping selection** 00: Mapping to crypro table 01: Mapping to quantization/inverse quantization table 10: Mapping to Huffman table 11: Mapping to Multi-JPEG data buffer
29:28	RW	Rst17	**Capture linear RGB format (can't be used for compression)** 00: Normal YUV tile mode 01: Linear RGB 555 mode 10: Linear RGB 888 mode 11: Reserved
27	RW	Rst17	**Decode output write RGB mode** 0: Decode write is tile YUV mode 1: Decode write is linear RGB mode
26:24	RW	Rst17	**Decode output rotate selection** 000: Normal 001: Vertical flip 010: Horizontal flip 011: Clockwise rotate 180 degree 100: Backslash 101: Clockwise rotate 90 degree 110: Clockwise rotate 270 degree 111: Slash
23	RW	Rst17	**Vertical border auto mask mode** 0: Normal operation 1: Mask the data which are outside the bottom border. This can avoid some noises on bottom size when the bottom side don't align to macro block boundary.
22:21	RW	Rst17	**Profile selection register (when VR300[19]=0)** 00: VR104-VR160 are Profile 2 registers 01: VR104-VR160 are Profile 3 registers 10: VR104-VR160 are Profile 4 registers 11: VR104-VR160 are Profile 5 registers
22:21	RW	Rst17	**Huffman code length selection (when VR300[19]=1)** 00: VR160-VR16C are luminance AC Huffman list of code length registers 01: VR160-VR16C are chrominance AC Huffman list of code length registers 10: VR160-VR16C are luminance DC Huffman list of code length registers 11: VR160-VR16C are chrominance DC Huffman list of code length registers
20	RW	Rst17	**Enable profile mode** 0: Disable 1: Enable
19	RW	Rst17	**Enable programmable Huffman table** 0: Use default Huffman table 1: Use programmable Huffman table. Can't set when encode mode.
18	RW	Rst17	**Enable programmable quantization table** 0: Use default quantization tables 1: Use programmable quantization tables. Can't set when the high quality video mode is enabled. The DCT luminance quantization table VR060[15:11] register must be 0 for correct JPEG header buffer selection.

to next page

from previous page

17	RW	Rst17	**Crypto mode selection** 0: RC4 crypto mode 1: AES crypto mode
16	RW	Rst17	**Fast crypto mode** This register is recommended to be always ”0”
15	RW	Rst17	**Crypto status software reset** 0: Normal operation mode 1: Reset crypto state
14	RW	Rst17	**RC4 vector read mode selection** 0: Normal read for register VR3F0, VR3F4, VR3F8 and VR3FC 1: Read AES vector value from register VR3F0, VR3F4, VR3F8 and VR3FC Read RC4 I, J value from register VR3F8
13:12	RW	Rst17	**Video capture frame buffer address mode** 00: 32 bpp YUV444 tile mode 01: 24 bpp YUV444 tile mode 10: Reserved 11: 16 bpp YUV422 tile mode
11:10	RW	Rst17	**Video compression frame buffer address mode** 00: 32 bpp YUV444 tile mode 01: 24 bpp YUV444 tile mode 10: Reserved 11: 16 bpp YUV422 tile mode
9	RW	Rst17	**RC4 test mode** This register is recommended to be always ”0”
8	RW	Rst17	**RC4 initial reset** This register can be set only when engine is idle. Setting it to ’1’ can reset RC4 states, then setting it to ’0’ to go back normal state.
7 :4	RW	Rst17	**Reserved** This register must be always ”0”
3 :2	RW	Rst17	**Delay internal VSYNC** 00: No delay 01: Delay internal VSYNC by 12 periods of HSYNC cycle time 10: Auto optimized delay mode 11: Reserved This is used for video capture auto mode and anti-flicter enabled to avoid frame dropped.
1 :0	RW	Rst17	**Reserved (0)**

VR304: Video Interrupt Control Register

Offset: 304h **Init = 0**

Bit	Attr.	Reset	Description
31:18	RO	-	**Reserved (0)**
17	RW	Rst17	**Enable video management compression complete interrupt** 0: Disable 1: Enable
16	RW	Rst17	**Enable video management frame capture complete interrupt** 0: Disable 1: Enable
15:12	RO	-	**Reserved (0)**

to next page

from previous page

11	RW	Rst17	**Enable video VSYNC descriptor interrupt** 0: Disable 1: Enable
10	RW	Rst17	**Enable video stream descriptor interrupt** 0: Disable 1: Enable
9	RW	Rst17	**Enable video hang watchdog interrupt** 0: Disable 1: Enable
8	RW	Rst17	**Enable video halt ready interrupt** 0: Disable 1: Enable
7	RW	Rst17	**Reserved (0)**
6	RW	Rst17	**Enable video decode stream error interrupt** 0: Disable 1: Enable
5	RW	Rst17	**Enable video frame complete interrupt** 0: Disable 1: Enable
4	RW	Rst17	**Enable video mode detection ready interrupt** 0: Disable 1: Enable
3	RW	Rst17	**Enable video compression complete interrupt** 0: Disable 1: Enable
2	RW	Rst17	**Enable video compression packet ready interrupt** 0: Disable 1: Enable
1	RW	Rst17	**Enable video frame capture complete interrupt** 0: Disable 1: Enable
0	RW	Rst17	**Enable video mode detection watchdog out of lock interrupt** 0: Disable 1: Enable

VR308: Video Interrupt Control Register

Offset: 308h **Init = 0**

Bit	Attr.	Reset	Description
31:10	RO	-	**Reserved (0)**
17	RW	Rst17	**Video management compression complete interrupt status** 0: No interrupt 1: Interrupt is pending. Clear this register by writing 1.
16	RW	Rst17	**Video management capture complete interrupt status** 0: No interrupt 1: Interrupt is pending Clear this register by writing 1.
15:12	RO	-	**Reserved (0)**

to next page

from previous page

11	RW	Rst17	**Video VSYNC descriptor interrupt status** 0: No interrupt 1: Interrupt is pending Clear this register by writing 1.
10	RW	Rst17	**Video stream descriptor interrupt status** 0: No interrupt 1: Interrupt is pending Clear this register by writing 1.
9	RW	Rst17	**Video hang watchdog interrupt status** 0: No interrupt 1: Interrupt is pending Clear this register by writing 1.
8	RW	Rst17	**Video halt ready interrupt status** 0: No interrupt 1: Interrupt is pending Clear this register by writing 1.
7	RO	-	**Reserved (0)**
6	RW	Rst17	**Video decode stream error interrupt status** 0: No interrupt 1: Interrupt is pending Clear this register by writing 1.
5	RW	Rst17	**Video frame complete interrupt status** 0: No interrupt 1: Interrupt is pending Clear this register by writing 1.
4	RW	Rst17	**Video mode detection ready interrupt status** 0: No interrupt 1: Interrupt is pending Clear this register by writing 1.
3	RW	Rst17	**Video compression complete interrupt status** 0: No interrupt 1: Interrupt is pending. Clear this register by writing 1.
2	RW	Rst17	**Video compression packet ready interrupt status** 0: No interrupt 1: Interrupt is pending Clear this register by writing 1.
1	RW	Rst17	**Video capture complete interrupt status** 0: No interrupt 1: Interrupt is pending Clear this register by writing 1.
0	RW	Rst17	**Video mode detection watchdog out of lock interrupt status** 0: No interrupt 1: Interrupt is pending Clear this register by writing 1.

VR30C: Mode Detection Parameter Register			
Offset: 30Ch			**Init = X**
Bit	**Attr.**	**Reset**	**Description**
31:28	RW	-	**Tolerance value [3:0] in mode detection for stable horizontal signal** This register defines the tolerance in detecting for stable horizontal signal. The sampling clock to detect horizontal signal is 25 MHz. The unit of this register is one clock period of 25 MHz clock source.
27:24	RW	-	**Tolerance value [3:0] in mode detection for stable vertical signal** This register defines the tolerance in detecting for stable vertical signal. The unit of this register is one period of a scan line.
23:20	RW	-	**Minimum required count [3:0] in mode detection for stable horizontal signal** This register defines the required minimum count in detecting stable HSYNC signal to set mode detection horizontal signal stable. The minimum acceptable value of this register is 3.
19:16	RW	-	**Minimum required count [3:0] in mode detection for stable vertical signal** This register defines the required minimum count in detecting stable VSYNC signal to set mode detection vertical signal stable. The minimum acceptable value of this register is 3.
15:8	RW	-	**Edge pixel value threshold [7:0] in mode detection** This register defines the minimum RGB value of effective pixels in detecting left or right edge. Due to video source from ADC signals are very easy to be coupled with analog noises, the setting of this register can help the mode detector to ignore the noise when the noise values are smaller then the threshold.
7 :0	RW	-	**Reserved (0)**

VR310: Video Memory Restriction Area Starting Address Register			
Offset: 310h			**Init = 0x0000_0000**
Bit	**Attr.**	**Reset**	**Description**
31:28	RO	-	**Reserved (0)**
27:16	RW	Rst17	**Video memory restriction area start address** The address must be 64 KB alignment.
15:0	RO	-	**Reserved (0)**
Note : Any video memory write access whose address is outside the restriction area will be discarded.			

VR314: Video Memory Restriction Area End Address Register			
Offset: 314h			**Init = 0x0FFF_0000**
Bit	**Attr.**	**Reset**	**Description**
31:28	RO	-	**Reserved (0)**
27:16	RW	Rst17	**Video memory restriction area end address** The address must be 64 KB alignment.
15:0	RO	-	**Reserved (0)**
Note : Any video memory write access whose address is outside the restriction area will be discarded.			

VR318: Video Memory Restriction Area Starting Address Register			
Offset: 318h			**Init = 0x0000_0000**
Bit	**Attr.**	**Reset**	**Description**
31:28	RO	-	**Reserved (0)**
27:16	RW	Rst17	**Video memory restriction area starting address** The address must be 64 KB alignment.
15:0	RO	-	**Reserved (0)**
Note : Any video memory write access whose address is outside the restriction area will be discarded.			

VR328: Video Data Truncation Register			
Offset: 328h			**Init = X**
Bit	**Attr.**	**Reset**	**Description**
31:16	RO	-	**Reserved (0)**
8 :6	RW	-	**R channel reduction bit number** 000: no reduction 001: recude 1 bit 010: recude 2 bits 011: recude 3 bits 100: recude 4 bits 101: recude 5 bits 110: recude 6 bits 111: recude 7 bits
5 :3	RW	-	**G channel reduction bit number** 000: no reduction 001: recude 1 bit 010: recude 2 bits 011: recude 3 bits 100: recude 4 bits 101: recude 5 bits 110: recude 6 bits 111: recude 7 bits
2 :0	RW	-	**B channel reduction bit number** 000: no reduction 001: recude 1 bit 010: recude 2 bits 011: recude 3 bits 100: recude 4 bits 101: recude 5 bits 110: recude 6 bits 111: recude 7 bits

VR33C: Video Watch Dog Timer Read Back			
Offset: 33Ch			**Init = X**
Bit	**Attr.**	**Reset**	**Description**
31:24	RO	-	**Reserved (0)**
23:0	RO	-	**Video watch dog timer read back** This register is in unit of us

VR340: VGA Scratch Remap Read Back Register			
Offset: 340h			**Init = X**
Bit	**Attr.**	**Reset**	**Description**
31:30	RO	-	**Reserved (0)**
29:24	RO	-	**VGA hardware cursor X position offset bit[5:0]**
23:22	RO	-	**Reserved (0)**
21:16	RO	-	**VGA hardware cursor Y position offset bit[5:0]**
15:10	RO	-	**Reserved (0)**
9	RO	-	**VGA hardware cursor type** 0: Monochrome cursor type. 1: Color cursor type.
8	RO	-	**VGA hardware cursor is enabled** 0: VGA hardware cursor is disabled. 1: VGA hardware cursor is enabled.
7 :0	RO	-	**Remap to VGA CR80 register**

VR344: VGA Scratch Remap Read Back Register			
Offset: 344h			**Init = X**
Bit	**Attr.**	**Reset**	**Description**
31:27	RO	-	**Reserved (0)**
26:16	RO	-	**Hardware cursor Y position bit[10:0]**
15:12	RO	-	**Reserved (0)**
11:0	RO	v	**Hardware cursor X position bit[11:0]**

VR348: VGA Scratch Remap Read Back Register			
Offset: 348h			**Init = X**
Bit	**Attr.**	**Reset**	**Description**
31:26	RO	-	**Reserved (0)**
25:3	RO	-	**Hardware cursor pattern memory address bit [25:3]**
2 :0	RO	-	**Reserved (0)**

VR34C: VGA Scratch Remap Read Back Register			
Offset: 34Ch			**Init = X**
Bit	**Attr.**	**Reset**	**Description**
31:24	RO	-	**Remap to VGACR8F register**
23:16	RO	-	**Remap to VGACR8E register**
15:8	RO	-	**Remap to VGACR8D register**
7 :0	RO	-	**Remap to VGACR8C register**

VR350: VGA Scratch Remap Read Back Register			
Offset: 350h			**Init = X**
Bit	**Attr.**	**Reset**	**Description**
31:24	RO	-	**Remap to VGACR93 register**
23:16	RO	-	**Remap to VGACR92 register**
15:8	RO	-	**Remap to VGACR91 register**
7 :0	RO	-	**Remap to VGACR90 register**

VR354: VGA Scratch Remap Read Back Register			
Offset: 354h			**Init = X**
Bit	**Attr.**	**Reset**	**Description**
31:24	RO	-	**Remap to VGACR97 register**
23:16	RO	-	**Remap to VGACR96 register**
15:8	RO	-	**Remap to VGACR95 register**
7 :0	RO	-	**Remap to VGACR94 register**

VR358: VGA Scratch Remap Read Back Register			
Offset: 358h			**Init = X**
Bit	**Attr.**	**Reset**	**Description**
31:24	RO	-	**Remap to VGACR9B register**
23:16	RO	-	**Remap to VGACR9A register**
15:8	RO	-	**Remap to VGACR99 register**
7 :0	RO	-	**Remap to VGACR98 register**

VR35C: VGA Scratch Remap Read Back Register			
Offset: 35Ch			**Init = X**
Bit	**Attr.**	**Reset**	**Description**
31:30	RO	-	**VGA power state (PCIS44[1:0])**
29	RO	-	**VGA attribute index register bit 5**
28	RO	-	**VGA mask register not zero**
27	RO	-	**VGA CRT reset**
26	RO	-	**VGA screen off**
25	RO	-	**VGA reset**
24	RO	-	**VGA enable**
23:16	RO	-	**Remap to VGACR9E register**
15:8	RO	-	**Remap to VGACR9D register**
7 :0	RO	-	**Remap to VGACR9C register**

VR360: VGA Scratch Remap Read Back Register			
Offset: 360h			**Init = X**
Bit	**Attr.**	**Reset**	**Description**
31:25	RO	-	**Reserved (0)**

to next page

from previous page

24:16	RO	-	**Remap to VGA horzontal total register**
15:9	RO	-	**Reserved (0)**
8 :0	RO	-	**Remap to VGA horzontal end register**

VR364: VGA Scratch Remap Read Back Register

Offset: 364h **Init = X**

Bit	Attr.	Reset	Description
31:27	RO	-	**Reserved (0)**
26:16	RO	-	**Remap to VGA vertical total register**
15:11	RO	-	**Reserved (0)**
10:0	RO	-	**Remap to VGA vertical end register**

VR3F0: Video Crypto Vector Register 1

Offset: 3F0h **Init = X**

Bit	Attr.	Reset	Description
31:0	RW	-	**Video crypto vector registers** RC4 mode: reserved. AES128 mode: initial vector [31:0].

VR3F4: Video Crypto Vector Register 2

Offset: 3F4h **Init = X**

Bit	Attr.	Reset	Description
31:0	RW	-	**Video crypto vector registers** RC4 mode: reserved. AES128 mode: initial vector [63:32].

VR3F8: Video Crypto Vector Register 3

Offset: 3F8h **Init = X**

Bit	Attr.	Reset	Description
31:0	RW	-	**Video crypto vector registers** RC4 mode: i initial value [7:0]. j initial value [15:8]. AES128 mode: initial vector [95:64].

VR3FC: Video Crypto Vector Register 4

Offset: 3FCh **Init = X**

Bit	Attr.	Reset	Description
31:0	RW	-	**Video crypto vector registers** RC4 mode: reserved. AES128 mode: initial vector [127:96].

VR400 ~ VR4FC: Video RC4/AES128 Encryption Key Register #0 ~ #63

Offset: 400~4FC **Init = X**

Bit	Attr.	Reset	Description
31:0	RW	-	**RC4 key data SRAM** The address map to encryption key table whn VR300[30]=0 and VR300[31]=0. There are 256 bytes of embedded SRAM (64 double words in total) designed to store RC4/AES128 encryption keys. AES128 mode uses 176 bytes only. Initializing the SRAM is necessary when enabling RC4/AES128 encryption for compressed video stream.

VR400 ~ VR5FC: Video Quantization Table and Inverse Quantization Table #0 ~ #63

Offset: 400~5FCh **Init = X**

Bit	Attr.	Reset	Description
31:0	RW	-	**Quantization table SRAM** The address map to quantization table when VR300[30]=1 and VR300[31]=0

VR400 ~ VR55C: JPEG Huffman Table #0 ~ #63

Offset: 400~55Ch **Init = X**

Bit	Attr.	Reset	Description
31:0	RW	-	**JPEG Huffman table SRAM** The address map to quantization table when VR300[30]=0 and VR300[31]=1

VR400 ~ VR4F8: Multi-JPEG Data Buffer (Even) #0 ~ #31

Offset: 400~4F8 **Init = X**

Bit	Attr.	Reset	Description
31:26	W	-	**Reserved**
25:13	W	-	**Multi-JPEG JPEG Width**
12:0	W	-	**Multi-JPEG JPEG Height**
31:24	RO	-	**Reserved**
23:0	RO	-	**Multi-JPEG JPEG data size after compression in unit of byte**

Note :
Note: Here is the programming guide of multi-JPEG mode:
1. Set VR008[30]=1 to enable multi-JPEG mode.
2. Set VR008[29:24] with multi-JPEG frame number.
3. Set VR004[3]=1 and VR004[5]=0 for multi-JPEG trigger mode.
4. Set VR300[30]=1 and VR300[31]=1 for multi-JPEG VR400-VR5FC address mapping selection
5. Set VR400, VR404 for JPEG frame #1 and VR408, VR40C for JPEG frame #2 etc. for their location and size.
6. The other registers are same as normal JPEG compression.
7. Trigger multi-JPEG compression by setting VR004[4]=1.
8. Wait engine idle and check VR400-VR5FC for compression status and frame buffer for compression results.

VR404 ~ VR4FC: Multi-JPEG Data Buffer (Odd) #0 ~ #31

Offset: 404~4FC **Init = X**

Bit	Attr.	Reset	Description
31	W	-	**Reserved**

to next page

from previous page

30:7	W	-	**Multi-JPEG Source Start Address** The purpose of multi-JPEG is used to compress multiple partial images inside captured image. The captured image are arranged in block (tile) mode and the partial JPEG boundary can be block aligned only. The block size of YUV444 mode is 8 by 8 and YUV420 mode is 16 by 16. When the top left coordinate of partial image is (x, y) then the block coordinate (bx, by) is (x/8, y/8) for YUV444 mode and (x/16, y/16) for YUV420 mode. The partial image start address of (bx, by): YUV444: = VR044 + VR048*8*by + 256*bx YUV420: = VR044 + VR048*16*by + 256*bx
6:0	W	-	**Reserved**
31	RO	-	**Reserved**
25:8	RO	-	**Multi-JPEG Next Frame Compressed Start Address Offset from VR054** The start address of next frame N can calculate by: VR054 + ((VR(404+(N-1)*8) & 0x01FFFF00) >> 8) << 7.
7	RO	-	**Multi-JPEG Current Frame Compression Valid** This bit means the frame is compressed succesfully.
6:0	RO	-	**Reserved**

36.4 ASPEED Graphics Scratched Registers Definition

VGACR8C: (VIDEO:1E70:034C)

- D[7:4]
  - Current Color Mode
- D[3:0]
  - New Color Mode
- Color Mode Definition:
  - 000b: EGA Mode
  - 001b: VGA Mode
  - 010b: 15bpp Mode
  - 011b: 16bpp Mode
  - 100b: 32bpp Mode
  - 1111b: CGA Mode
  - 1110b: Text Mode

VGACR8D: (VIDEO:1E70:034D)

- D[3:0]
  - Refresh Rate Index
- Refresh Rate Index Definition:

Resolution	Refresh rate	Index	Note
640x480	60	1	
	72	2	
	75	3	
	85	4	
800x600	56	1	
	60	2	
	72	3	
	75	4	
	85	5	
1024x768	60	1	
	70	2	
	75	3	
	85	4	
1280x1024	60	1	
	75	2	
	85	3	
1600x1200	60	1	
	65	2	
	70	3	
	75	3	
	85	3	
1920x1200	60	1	AST2100/2200 only

VGACR8E: (VIDEO:1E70:034E)

- D[7:0]: MODE ID

Mode ID	Resolution	Note
0x2E	640x480	
0x30	800x600	
0x31	1024x768	
0x32	1280x1024	
0x33	1600x1200	
0x34	1920x1200	
0x35	1280x800	
0x36	1440x900	
0x37	1680x1050	
0x38	1920x1080	
0x39	1366x768	
0x3A	1600x900	
0x3B	1152x864	
0x50	320x240	
0x51	400x300	
0x52	512x384	
0x6A	800x600	

VGACR8F: (VIDEO:1E70:034F)

- D[7:6]: Alphanumeric scan lines (map to 0040:0089 D[7] D[4])
  - 00b: 350 lines
  - 01b: 400 lines
  - 10b: 200 lines
- D[5]: Customized VSync Control
- D[4]:
  - 0b: disable VSync
  - 1b: enable VSync
- D[3]: V-PLL Reference Clock
  - 0b: 24MHz
  - 1b: 25MHz
- D[2]: Inhibit to access through P2A (1)
- D[1]: AST2100 A2 (1)
- D[0]: SOC RTC Initial Status (Have effects only if VBIOS init. SOC RTC)
  - 0b: not ready
  - 1b: ready

VGACR90: (VIDEO:1E70:0350) (VGA only Scratched)

- D[7]: Display On control by driver (1)
- D[6]: In DPMS State (1)
- D[5]: Bank/Linear Mode (0/1)

- D[4]:Two Clone VGA Support (1)
- D[3]:UEFI Graphics Driver (1)
- D[2]:Detect 3rd party TX Chip (1)
- D[1:0]: Chip Bounding Options (for AST1100/2050/2100/2200)
  - 01b: AST2200
  - 10b: AST1100/AST2050
  - 11b: AST2100

VGACR98: (VIDEO:1E70:0358) (VGA only Scratched)

- D[7:0]: Available Bandwidth / 8

VGACR99: (VIDEO:1E70:0359) (VGA only Scratched)

- D[7]: Monitor Support Prefer Mode(1)
- D[6]: Monitor Support Wide Screen Mode (1)
- D[5]: Monitor Support Reduce Blank(1)
- D[1:0]: Reserved Video Buffer from Last
  - 00b: 0MB
  - 01b: 1MB
  - 10b: 2MB
  - 11b: 4MB

VGACR9C: (VIDEO:1E70:035C) (VGA only Scratched)

- D[7:0]: HDE D[7:0]

VGACR9D: (VIDEO:1E70:035D)(VGA only Scratched)

- D[7:0]: VDE D[7:0]

VGACR9E: (VIDEO:1E70:035E) (VGA only Scratched)

- D[7:4]: VDE D[11:8]
- D[3:0]: HDE D[11:8]

VGACR91: New Mode Info Header (VIDEO:1E70:0351) (New Mode Info Scratched)

- D[7:0]: 0xA8

VGACR92: (VIDEO:1E70:0352) (New Mode Info Scratched)

- D[7:0]: Color Depth (set to 0 if the color depth is below then 8bpp)

VGACR93: (VIDEO:1E70:0353) (New Mode Info Scratched)

- D[7:0]: Pixel Clock (MHz)

VGACR94: (VIDEO:1E70:0354) (New Mode Info Scratched)

- D[7:0]: HDE D[7:0]

VGACR95: (VIDEO:1E70:0355) (New Mode Info Scratched)

- D[7:0]: HDE D[15:8]

VGACR96: (VIDEO:1E70:0356) (New Mode Info Scratched)

- D[7:0]: VDE D[7:0]

VGACR97: (VIDEO:1E70:0357) (New Mode Info Scratched)

- D[7:0]: VDE D[15:8]

VGACR9A: (VIDEO:1E70:035A) (CMD Interface with Embedded Firmware)

- D[7:0]: CMD/Data

VGACR9B: (VIDEO:1E70:035B) (CMD Interface with Embedded Firmware)

- D[7]: VGA ACK

37 SRAM Memory Buffer (SRAM)

37.1 Overview

Embedded SRAM provide 3 portion of address space, which are all directly connect to AHB bus.

1. 64KB. Address range is from 0x1000_0000 to 0x1000_FFFF.
   - Available at boot.
   - With Parity Check. Parity Check only support Double-Word access.
2. 24KB. Address range is from 0x1001_0000 to 0x1001_5FFF.
   - Available at boot.
   - Without Parity Check.
3. 1KB. Address range is from 0x1001_6000 to 0x1001_63FF.
   - Available at boot.
   - Without Parity Check.
   - Each byte is write once. Unlock until RstARM.
   - One global control bit to lock all at once. Unlock until RstPwr

Besides storage part, there are also some registers for understanding write protection status.

Base Address of Embedded SRAM = 0x1E72_0000
Register Address = Base Address + Offset

SRAMWP00: Write Protect Status #0
SRAMWP04: Write Protect Status #1
SRAMWP08: Write Protect Status #2
SRAMWP0C: Write Protect Status #3
SRAMWP10: Write Protect Status #4
SRAMWP14: Write Protect Status #5
SRAMWP18: Write Protect Status #6
SRAMWP1C: Write Protect Status #7
SRAMWP20: Write Protect Status #8
SRAMWP24: Write Protect Status #9
SRAMWP28: Write Protect Status #10
SRAMWP2C: Write Protect Status #11
SRAMWP30: Write Protect Status #12
SRAMWP34: Write Protect Status #13
SRAMWP38: Write Protect Status #14
SRAMWP3C: Write Protect Status #15
SRAMWP40: Write Protect Status #16
SRAMWP44: Write Protect Status #17
SRAMWP48: Write Protect Status #18
SRAMWP4C: Write Protect Status #19
SRAMWP50: Write Protect Status #20
SRAMWP54: Write Protect Status #21
SRAMWP58: Write Protect Status #22
SRAMWP5C: Write Protect Status #23
SRAMWP60: Write Protect Status #24
SRAMWP64: Write Protect Status #25
SRAMWP68: Write Protect Status #26
SRAMWP6C: Write Protect Status #27
SRAMWP70: Write Protect Status #28
SRAMWP74: Write Protect Status #29
SRAMWP78: Write Protect Status #30

SRAMWP7C: Write Protect Status #31

37.2 Registers

Offset: 00h-7Ch	SRAMWP00-SRAMWP7C: Write Protection Status #0-#31	Init = 0

Bit	R/W	Reset	Description
31:0	RO	RstARM	**Write Protection Status for SRAM address range from 0x1001_6000 to 0x1001_63FF** Each bit in the register means: 1: Write Protected 0: Write allowed

Note :
The SRAM address range from 0x1001_6000 to 0x1001_63FF is write once only. When the bit in the register is 1 means the corrseponding SRAM byte has been written and write protection is enabled automatically. When the bit in the register is 1 means the corrseponding SRAM byte hasn't been written.
The calculation of SRAM offset for bit #M in Write Protection Status #N register N*32 + M.

38 SD/SDIO Host Controller (SDC)

38.1 Overview

Base Address of SDIO Controller = 0x1E74_0000
Physical address of register = (Base address of SDIO Controller) + Offset

SDIO000: General Information Register
SDIO004: Debounce Settings Register
SDIO008: Bus Settings Register
SDIO010: HWInit SDIO140 Configuration for slot #0
SDIO014: HWInit SDIO144 Configuration for slot #0
SDIO018: HWInit SDIO148 Configuration for slot #0
SDIO020: HWInit SDIO240 Configuration for slot #0
SDIO024: HWInit SDIO244 Configuration for slot #0
SDIO028: HWInit SDIO248 Configuration for slot #0
SDIO0F0: Card Detect Control
SDIO0F4: Clock Phase Control
SDIO0FC: Interrupt Status

Slot Register Set for slot 0

SDIO100: SDMA System Address Register
SDIO104: Block Count and Size Register
SDIO108: Argument Register
SDIO10C: Command and Transfer Mode Register
SDIO110: Response #0
SDIO114: Response #1
SDIO118: Response #2
SDIO11C: Response #3
SDIO120: Buffer Data Port Register
SDIO124: Present State Register
SDIO128: Host Control Settings Register #0
SDIO12C: Host Control Settings Register #1
SDIO130: Interrupt Status Register
SDIO134: Interrupt Status Enable Register
SDIO138: Interrupt Signal Enable Register
SDIO13C: Auto CMD12 Error Status Register
SDIO140: Capabilities Register #1
SDIO144: Capabilities Register #2
SDIO148: Maximum Current Capabilities Register
SDIO150: Force Event Register
SDIO154: ADMA Error Status Register
SDIO158: ADMA System Address Register

Slot Register Set for slot 1

SDIO200: SDMA System Address Register
SDIO204: Block Count and Size Register
SDIO208: Argument Register
SDIO20C: Command and Transfer Mode Register
SDIO210: Response #0
SDIO214: Response #1
SDIO218: Response #2
SDIO21C: Response #3

SDIO220: Buffer Data Port Register
SDIO224: Present State Register
SDIO228: Host Control Settings Register #0
SDIO22C: Host Control Settings Register #1
SDIO230: Interrupt Status Register
SDIO234: Interrupt Status Enable Register
SDIO238: Interrupt Signal Enable Register
SDIO23C: Auto CMD12 Error Status Register
SDIO240: Capabilities Register #1
SDIO244: Capabilities Register #2
SDIO248: Maximum Current Capabilities Register
SDIO250: Force Event Register
SDIO254: ADMA Error Status Register
SDIO258: ADMA System Address Register

38.2 Features

- Each slots is compatible to
  - SD Memory Card Version 3.00
  - EMMC Version 3.00
  - eMMC Version 5.1
- Support 2 slots.
- Slot 0 supports SD1/SD4 or eMMC1/eMMC4/eMMC8 modes of operation
- Slot 1 supports SD1/SD4 or eMMC1/eMMC4 modes of operation
- Indepent clock/configuration for each slot.
- Indepent register set for each slot.
- Integrated ADMA2 controllers.
- Master device side clock and each slot's clock can be switched off.

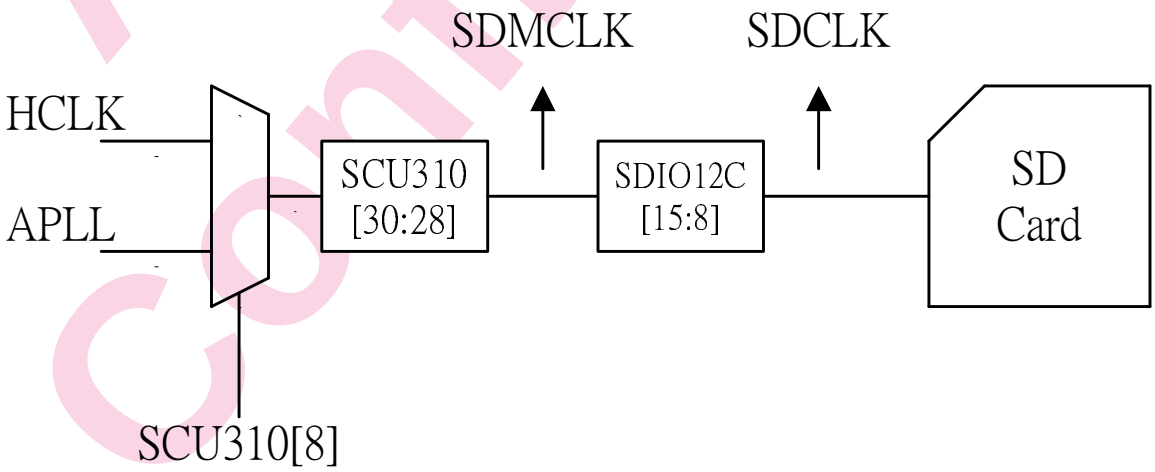


Figure 51: Clock structure of SD interface

38.3 Registers : Base Address = 0x1E74:0000

Offset: 000h	SDIO000: General Information Register	Init = 0x0003_0000

Bit	R/W	Reset	Description
31:26	RO	-	**Reserved**
25	RW	Rst42	**S1MMC8** 1: 8-bit mode 0: 1 or 4 bit mode (depends on SDIO228[1])
24	RW	Rst42	**S0MMC8** 1: 8-bit mode 0: 1 or 4 bit mode (depends on SDIO128[1])
23:18	RO	-	**Reserved**
17	RO	-	**Slot #1 available** 1: slot is available 0: slot is not available
16	RO	-	**Slot #0 available** 1: slot is available 0: slot is not available
15: 1	RO	-	**Reserved**
0	RW	Rst42	**Software Reset** Set to reset all flip-flops in every slot. This bit will automatically clear. After set this bit, software should wait it is cleared to continue.

Offset: 004h	SDIO004: Debounce Settings Register	Init = 0x0000_0005

Bit	R/W	Reset	Description
31:24	RO	-	**Reserved**
23: 0	RW	Rst42	**Debounce Period** Debounce period = SDIO004[23:0] * period of HCLK

Offset: 008h	SDIO008: Bus Settings Register	Init = 0

Bit	R/W	Reset	Description
31: 4	RO	-	**Reserved**
3: 0	RW	Rst42	**Programmable Burst Length** Maximum number of beats within single DMA burst. Value Number of beats 0001b 1 0010b 2 0011b 4 0100b 8 0101b 16 0110b 32 0111b 64 1000b 128 1001b 256 1010b 512 1011b 1024 others 2048

Offset: 010h	SDIO010: HWInit SDIO140 Configuration for slot #0	Init = 0

Bit	R/W	Reset	Description
31: 0	RW	Rst42	**Mirror of SDIO140** Write to update corresponding SDIO140 value

Offset: 014h	SDIO014: HWInit SDIO144 Configuration for slot #0	Init = 0

Bit	R/W	Reset	Description
31: 0	RW	Rst42	**Mirror of SDIO144** Write to update corresponding SDIO144 value

Offset: 018h	SDIO018: HWInit SDIO148 Configuration for slot #0	Init = 0

Bit	R/W	Reset	Description
31: 0	RW	Rst42	**Mirror of SDIO148** Write to update corresponding SDIO148 value

Offset: 020h	SDIO020: HWInit SDIO240 Configuration for slot #0	Init = 0

Bit	R/W	Reset	Description
31: 0	RW	Rst42	**Mirror of SDIO240** Write to update corresponding SDIO240 value

Offset: 024h	SDIO014: HWInit SDIO244 Configuration for slot #0	Init = 0

Bit	R/W	Reset	Description
31: 0	RW	Rst42	**Mirror of SDIO244** Write to update corresponding SDIO244 value

Offset: 028h	SDIO028: HWInit SDIO248 Configuration for slot #0	Init = 0

Bit	R/W	Reset	Description
31: 0	RW	Rst42	**Mirror of SDIO248** Write to update corresponding SDIO248 value

Offset: 0F0h	SDIO0F0: Card Detect Control	Init = 0xc0000000

Bit	R/W	Reset	Description
31: 2	RO	-	**Reserved**
1	RW	Rst42	**Slot 1 Write Protection Inverse bit** 1: Inverse. 0: No operation.
0	RW	Rst42	**Slot 0 Write Protection Inverse bit** 1: Inverse. 0: No operation.

Offset: 0F4h	SDIO0F4: Clock Phase Control	Init = 0x101

Bit	R/W	Reset	Description
31:26	RO	-	**Reserved**
25:21	RW	Rst42	**Slot 1 Input Clock Phase**
20:16	RW	Rst42	**Slot 0 Input Clock Phase**
15:11	RW	Rst42	**Slot 1 Output Clock Phase**
10	RW	Rst42	**Slot 1 Input Clock Phase Selection** 1: Enable. 0: Disable.
9: 8	RW	Rst42	**Slot 1 Output Clock Phase Selection** 11b : Enable. Others: Disable.
7: 3	RW	Rst42	**Slot 0 Output Clock Phase**
2	RW	Rst42	**Slot 0 Input Clock Phase Selection** 1: Enable. 0: Disable.
1: 0	RW	Rst42	**Slot 0 Output Clock Phase Selection** 11b : Enable. Others: Disable.

Offset: 0FCh	SDIO0FC: Interrupt Status	Init = 0x00010000

Bit	R/W	Reset	Description
31: 2	RO	-	**Reserved**
1	RW	Rst42	**Interrupt Status** Bit 1: Interrput pending in Slot1.
0	RW	Rst42	**Interrupt Status** Bit 0: Interrput pending in Slot0.

Offset: 100h	SDIO100: SDMA System Address Register	Init = 0

Bit	R/W	Reset	Description
31: 0	RW	Rst216	**SDMA System Address**

Offset: 104h	SDIO104: Block Count and Size Register	Init = 0

Bit	R/W	Reset	Description
31:16	RW	Rst216	**Block Count for Current Transfer** 0000h: 0 0001h: 1 ... FFFFh: 65535
15	RO	-	**Reserved**

to next page

from previous page

14:12	RW	Rst216	**Host DMA Buffer Boundary** 000b 4kB 001b 8kB 010b 16kB 011b 32kB 100b 64kB 101b 128kB 110b 256kB 111b 512kB
11: 0	RW	Rst216	**Transfer Block Size** 800h 2048 Bytes 799h 2047 Bytes .. 002h 2 Bytes 001h 1 Byte 000h no data transfer

Offset: 108h **SDIO108: Argument Register** **Init = 0**

Bit	R/W	Reset	Description
31: 0	RW	Rst216	**Command Argument** Contain bit [39:8] of command argument

Offset: 10Ch **SDIO10C: Command and Transfer Mode Register** **Init = 0**

Bit	R/W	Reset	Description
31:30	RO	-	**Reserved**
29:24	RW	Rst216	**Command Index** 00h: CMD0 / ACMD0 01h: CMD1 / ACMD1 ... 3Fh: CMD63 / ACMD63
23:22	RW	Rst216	**Command Type** 11b Abort Command 10b Suspend Command 01b Resume Command 00b Normal Command
21	RW	Rst216	**Data Present Select** 1: Command which transfer data 0: all other commands
20	RW	Rst216	**Command Index Check Enable** 1: Check if command index field of the response equal to SDIO10C[29:24].
19	RW	Rst216	**Command CRC Check Enable** 1: Check if CRC field of the response is valid.
18	RO	-	**Reserved**
17:16	RW	Rst216	**Response Type Select** 00b no response 01b 136-bit response 10b 48-bit response 11b 48-bit response with BUSY
15: 6	RO	-	**Reserved**

to next page

from previous page

5	RW	Rst216	**Multi/Single Block Select** 1: Multi block data transfer. 0: Single block data transfer.
4	RW	Rst216	**Data Transfer Direction Select** 1: Data Read. Data from the card to the host. 0: Data Write. Data from the host to the card.
3	RO	-	**Reserved**
2	RW	Rst216	**Auto CMD12 Enable** 1: Enable. 0: Disable.
1	RW	Rst216	**Block Count Enable** 1: Enable. 0: Disable.
0	RW	Rst216	**DMA Enable** 1: Enable. 0: Disable.

Offset: 110h **SDIO110: Response #0** **Init = 0**

Bit	R/W	Reset	Description
31: 0	RO	Rst216	**Command response[31: 0]**

Offset: 114h **SDIO114: Response #1** **Init = 0**

Bit	R/W	Reset	Description
31: 0	RO	Rst216	**Command response[63: 32]**

Offset: 118h **SDIO118: Response #2** **Init = 0**

Bit	R/W	Reset	Description
31: 0	RO	Rst216	**Command response[95: 64]**

Offset: 11Ch **SDIO11C: Response #3** **Init = 0**

Bit	R/W	Reset	Description
31: 0	RO	Rst216	**Command response[127: 96]**

Kind of Response	Meaning of Response	Response Field	Response Register
R1, R1b(Normal Response)	Card Status	R[39: 8]	RESP[31: 0]
R1b(Auto CMD12 Response)	Card Status for Auto CMD12	R[39: 8]	RESP[127:96]
R2(CID, CSD Register)	CID or CSD register	R[127: 8]	RESP[119: 0]
R3(OCR Register)	OCR register for memory	R[39: 8]	RESP[31: 0]
R4(OCR Register)	OCR register for I/O etc	R[39: 8]	RESP[31: 0]
R5,R5b	SDIO response	R[39: 8]	RESP[31: 0]
R6(Published RCA Response)	New Published RCA[31:16] etc	R[39: 8]	RESP[31: 0]

Offset: 120h	SDIO120: Buffer Data Port Register	Init = 0

Bit	R/W	Reset	Description
31: 0	RW	-	**Buffer Data Port**

Offset: 124h	SDIO124: Present State Register	Init = 0

Bit	R/W	Reset	Description
31:25	RO	-	**Reserved**
24	RO	-	**CMD Line Signal Level** Equal to the actual signal level on CMD line of the interface.
23:20	RO	-	**DAT[3:0] Line Signal Level** Equal to the actual signal level on DAT line of the interface. SDIO124[23]: DAT[3] SDIO124[22]: DAT[2] SDIO124[21]: DAT[1] SDIO124[20]: DAT[0]
19	RO	-	**Write Protect Switch Pin Level** Equal to the actual signal level on Write Protect line of the interface.
18	RO	-	**Card Detect Switch Pin Level** Equal to the actual signal level on Card Detect line of the interface.
17	RO	-	**Card State Stable** Indicate if SDIO124[18] is stable. 1: SDIO124[18] is stable. 0: SDIO124[18] is unstable.
16	RO	-	**Card Inserted** Indicate if the card is inserted inside the slot. 1: the card is inserted. 0: no card is inside the slot. This bit is guaranteed to be stable.
15:12	RO	-	**Reserved**
11	RO	-	**Buffer Read Enable** Shows the current data buffer state during non-DMA data read transfers. 1: valid data can be read from the data buffer. 0: no valid data inside the data buffer.
10	RO	-	**Buffer Write Enable** Shows the current data buffer state during non-DMA data write transfers. 1: data can be written to the data buffer. 0: data cna not be written.
9	RO	-	**Read Transfer Active** Indicate the status of the read data transfer. 1: data read transfer is in progress. 0: no read transfer is in progress.
8	RO	-	**Write Transfer Active** Indicate the status of the write data transfer. 1: data write transfer is in progress. 0: no write transfer is in progress.
7: 4	RO	-	**Reserved**
3	RW	-	**Re-Tining Request**
2	RO	-	**DAT Line Active** Indicate if the DAT line of the interface are currently in use. 1: DAT line is active(in use). 0: DAT line is released(not in use).

to next page

from previous page

1	RO	-	**Command Inhibit DAT** Indicate if SDIO-HOST can issue a command which use DAT line. 1: command using DAT line can not be sent. 0: command using DAT line can be sent.
0	RO	-	**Command Inhibit CMD** Indicate if SDIO-HOST can issue a command. 1: command can not be sent. 0: command can be sent.

Offset: 128h **SDIO128: Host Control Settings #0 Register** **Init = 0**

Bit	R/W	Reset	Description
31:27	RO	-	**Reserved**
26	RW	Rst216	**Wakeup Event Enable on SD Card Removal** 1: Enable. 0: Disable.
25	RW	Rst216	**Wakeup Event Enable on SD Card Inserted** 1: Enable. 0: Disable.
24	RW	Rst216	**Wakeup Event Enable on SD Card Interrupt** 1: Enable. 0: Disable.
23:20	RO	-	**Reserved**
19	RW	Rst216	**Interrupt at Block Gap** 1: Enable interrupt detection at the block gap for a multiple block transfer. This bit is only valid in SD4 mode. If the SD card can not signal an interrupt during a multiple block transfer, this bit should be set to 0.
18	RW	Rst216	**Read Wait Control** 1: Enable. 0: Disable.
17	RW	Rst216	**Continue Request** 1: Restart transfer stopped previously using SDIO128[16]. This bit is cleared automatically.
16	RW	Rst216	**Stop at Block Gap Request** 1: Stop excuting read and write transaction at the next block gap. This bit should be cleared before continuing request.
15:12	RO	-	**Reserved**
11: 9	RW	Rst216	**SD Bus Voltage Select** 111b : 3.3V 110b : 3.0V 101b : 1.8V
8	RW	Rst216	**SD Bus Power** 1: SD Device is powered. 0: SDIO-Host stops driving CMD, DAT and SDCLK. When card is removal from the slot, SDIO-Host automatically set this bit to 0.
7	RW	Rst216	**Card Detect Signal Selection** 1: SDIO128[6]. 0: From slot.

to next page

from previous page

6	RW	Rst216	**Card Detect Test Level** 1: Card inserted. 0: No card.
5	RW	Rst216	**Extended Data Transfer Width** 1: 8-bit. (Same effect with SDIO000[24]) 0: Depends on SDIO128[1]
4: 3	RW	Rst216	**DMA Select** 00b: SDMA 01b: ADMA1 10b: ADMA2 11b: Reserved
2	RW	Rst216	**Hish Speed Enable** 1: Enable 0: Disable
1	RW	Rst216	**Data Transfer Width** 1: SD4 (4-bits) mode. 0: SD1 (1-bit) mode.
0	RO	-	**Reserved**

Offset: 12Ch **SDIO12C: Host Control Settings #1 Register** **Init = 0**

Bit	R/W	Reset	Description
31:27	RO	-	**Reserved**
26	RW	Rst42	**Software Reset for DAT Line** 1: Reset data path, including data buffer and DMA logics. This bit is cleared automatically.
25	RW	Rst42	**Software Reset for CMD Line** 1: Reset command/response generation and checking. This bit is cleared automatically.
24	RW	Rst42	**Software Reset for All** 1: Entire SDIO-Host is reset. This bit is cleared automatically.
23:20	RO	-	**Reserved**
19:16	RW	Rst216	**Data Timeout Counter Value** 1111b Reserved 1110b Period of SDMCLK * 2^27 1101b Period of SDMCLK * 2^26 ... 0001b Period of SDMCLK * 2^14 0000b Period of SDMCLK * 2^13
15: 8	RW	Rst216	**SDCLK Frequency Select** Period of SDMCLK * 2 * (SDIO12C[7:6], SDIO12C[15:8])
7: 6	RW	Rst216	**SDCLK Frequency Select (upper)**
5	RW	Rst216	**SD Generator Select** 1: Programmable Clock Mode. 0: 10-bit Divider Clock Mode.
4: 3	RO	-	**Reserved**
2	RW	Rst216	**SD Clock Enable** 1: Enable SD slot clock. 0: Disable SD slot clock.

to next page

from previous page

1	RO	Rst216	**Internal Clock Stable** 1: Indicates clock pin of SDMCLK is stable. 0: Indicates clock pin of SDMCLK is not stable.
0	RW	Rst216	**Internal Clock Enable** 1: Enable SDMCLK.

Offset: 130h **SDIO130: Interrupt Status Register** **Init = 0**

Bit	R/W	Reset	Description
31:27	RO	-	**Reserved**
26	RW1C	Rst216	**Tuning Error** Read 1: Error occurs if tuning block is wrong. Write 1 to clear.
25	RW1C	Rst218	**ADMA Error** Read 1: Error occurs during ADMA read or write transfer. Write 1 to clear.
24	RW1C	Rst216	**Auto CMD12 Error** Read 1: Error occurs during Auto CMD12 command transmition. Write 1 to clear.
23	RW1C	Rst216	**Current Limit Error** Read 1: SD Card is not powered due to some failure. Write 1 to clear.
22	RW1C	Rst216	**Data End Bit Error** Read 1: Indicate detecting 0 at the end bit position of read data which uses the DAT line, or at the end bit position of the Write CRC Status. Write 1 to clear.
21	RW1C	Rst216	**Data CRC Error** Read 1: indicate transferring read data which uses the DAT line, or when detecting the Write CRC status having a value of other than ”010”. Write 1 to clear.
20	RW1C	Rst216	**Data Timeout Error** Read 1: indicate detecting one of the following timeout conditions: 1. Busy timeout for the response with busy. 2. Busy timeout after Write CRC status. 3. Write CRC Status timeout. 4. Read data timeout. Write 1 to clear.
19	RW1C	Rst216	**Command Index Error** Read 1: Index error occurs in the command response. Write 1 to clear.
18	RW1C	Rst216	**Command End Bit Error** Read 1: Indicate detecting the end bit of a command response is 0. Write 1 to clear.
17	RW1C	Rst216	**Command CRC Error** Read 1: Command CRC error occurs. Write 1 to clear.
16	RW1C	Rst216	**Command Timeout Error** Read 1: Indicate no response was returned within 64 SDCLK cycles from the end bit of the command. Write 1 to clear.

to next page

from previous page

15	RO	-	**Error Interrupt** 1: One of SDIO130[26:16] is set.
14:13	RO	-	**Reserved**
12	RO	Rst216	**Re-Tuning Event** 1: Indicate SDIO124[3] is changed from 0 to 1.
11: 9	RO	-	**Reserved**
8	RO	Rst216	**Card Interrupt** 1: Indicate the card interrupt.
7	RW1C	Rst216	**Card Removal** Read 1: Indicate card was removal from slot. Read 0: Indicate card is still inserted or removal or the debounce is in progress. Write 1 to clear.
6	RW1C	Rst216	**Card Inserted** Read 1: Indicate card was inserted to slot. Read 0: Indicate card is still inserted or removal or the debounce is in progress. Write 1 to clear.
5	RW1C	Rst218	**Buffer Read Ready** Read 1: Indicate data buffer can be read. Write 1 to clear.
4	RW1C	Rst218	**Buffer Write Ready** Read 1: Indicate data buffer can be written. Write 1 to clear.
3	RW1C	Rst218	**DMA Interrupt** In SDMA mode, DMA interrupt is generated when the Host Controller detects the Host SDMA Buffer boundary. In ADMA mode, DMA interrupt is generated when the INT flag is set in a currently serviced ADMA descriptor. Write 1 to clear.
2	RW1C	Rst218	**Block Gap Event** Read 1: Indicate read/write transaction is stopped at a block gap. Write 1 to clear.
1	RW1C	Rst218	**Transfer Complete** Read 1: Indicate transfer using DAT line is completed. Write 1 to clear.
0	RW1C	Rst217	**Command Complete** Read 1: Indicate the end bit of the response is received, except the response for Auto CMD12 command. Write 1 to clear.

Offset: 134h **SDIO134: Interrupt Status Enable Register** **Init = 0**

Bit	R/W	Reset	Description
31:26	RO	-	**Reserved**
26	RW	Rst216	**Tuning Error Status Enable** 1: Enable. 0: Masked.
25	RW	Rst216	**ADMA Error Status Enable** 1: Enable. 0: Masked.

to next page

from previous page

24	RW	Rst216	**Auto CMD12 Error Status Enable** 1: Enable. 0: Masked.
23	RW	Rst216	**Current Limit Error Status Enable** 1: Enable. 0: Masked.
22	RW	Rst216	**Data End Bit Error Status Enable** 1: Enable. 0: Masked.
21	RW	Rst216	**Data CRC Error Status Enable** 1: Enable. 0: Masked.
20	RW	Rst216	**Data Timeout Error Status Enable** 1: Enable. 0: Masked.
19	RW	Rst216	**Command Index Error Status Enable** 1: Enable. 0: Masked.
18	RW	Rst216	**Command End Bit Error Status Enable** 1: Enable. 0: Masked.
17	RW	Rst216	**Command CRC Error Status Enable** 1: Enable. 0: Masked.
16	RW	Rst216	**Command Timeout Error Status Enable** 1: Enable. 0: Masked.
15:13	RO	-	**Reserved**
12	RW	Rst216	**Re-Tuning Event Status Enable** 1: Enable. 0: Masked.
11: 9	RO	-	**Reserved**
8	RW	Rst216	**Card Interrupt Status Enable** 1: Enable. 0: Masked.
7	RW	Rst216	**Card Removal Status Enable** 1: Enable. 0: Masked.
6	RW	Rst216	**Card Inserted Status Enable** 1: Enable. 0: Masked.
5	RW	Rst216	**Buffer Read Ready Status Enable** 1: Enable. 0: Masked.
4	RW	Rst216	**Buffer Write Ready Status Enable** 1: Enable. 0: Masked.

to next page

from previous page

3	RW	Rst216	**DMA Interrupt Status Enable** 1: Enable. 0: Masked.
2	RW	Rst216	**Block Gap Event Status Enable** 1: Enable. 0: Masked.
1	RW	Rst216	**Transfer Complete Status Enable** 1: Enable. 0: Masked.
0	RW	Rst216	**Command Complete Status Enable** 1: Enable. 0: Masked.

Offset: 138h **SDIO138: Interrupt Enable Register** **Init = 0**

Bit	R/W	Reset	Description
31:27	RO	-	**Reserved**
26	RW	Rst216	**Tuning Error Interrupt Enable** 1: Enable. 0: Masked.
25	RW	Rst216	**ADMA Error Interrupt Enable** 1: Enable. 0: Masked.
24	RW	Rst216	**Auto CMD12 Error Interrupt Enable** 1: Enable. 0: Masked.
23	RW	Rst216	**Current Limit Error Interrupt Enable** 1: Enable. 0: Masked.
22	RW	Rst216	**Data End Bit Error Interrupt Enable** 1: Enable. 0: Masked.
21	RW	Rst216	**Data CRC Error Interrupt Enable** 1: Enable. 0: Masked.
20	RW	Rst216	**Data Timeout Error Interrupt Enable** 1: Enable. 0: Masked.
19	RW	Rst216	**Command Index Error Interrupt Enable** 1: Enable. 0: Masked.
18	RW	Rst216	**Command End Bit Error Interrupt Enable** 1: Enable. 0: Masked.
17	RW	Rst216	**Command CRC Error Interrupt Enable** 1: Enable. 0: Masked.
16	RW	Rst216	**Command Timeout Error Interrupt Enable** 1: Enable. 0: Masked.

to next page

from previous page

15:13	RO	-	**Reserved**
12	RW	Rst216	**Re-Tuning Event Interrupt Enable** 1: Enable. 0: Masked.
11: 9	RO	-	**Reserved**
8	RO	Rst216	**Card Interrupt Interrupt Enable** 1: Enable. 0: Masked.
7	RW	Rst216	**Card Removal Interrupt Enable** 1: Enable. 0: Masked.
6	RW	Rst216	**Card Inserted Interrupt Enable** 1: Enable. 0: Masked.
5	RW	Rst216	**Buffer Read Ready Interrupt Enable** 1: Enable. 0: Masked.
4	RW	Rst216	**Buffer Write Ready Interrupt Enable** 1: Enable. 0: Masked.
3	RW	Rst216	**DMA Interrupt Interrupt Enable** 1: Enable. 0: Masked.
2	RW	Rst216	**Block Gap Event Interrupt Enable** 1: Enable. 0: Masked.
1	RW	Rst216	**Transfer Complete Interrupt Enable** 1: Enable. 0: Masked.
0	RW	Rst216	**Command Complete Interrupt Enable** 1: Enable. 0: Masked.

Offset: 13Ch **SDIO13C: Auto CMD12 Error Status Register** **Init = 0**

Bit	R/W	Reset	Description
31	RW	Rst216	**Preset Value Enable**
30	RW	Rst216	**Asynchronous Interrupt Enable**
29:24	RO	-	**Reserved**
23	RW	Rst216	**Sampling Clock Select** Write 1: no operation. Write 0: reset and disable tuning block. Read 1: tuning procedure completes.
22	RW	Rst216	**Execute Tuning**
21:20	RW	Rst216	**Driver Strength Select**
19	RW	Rst216	**1.8V Signaling Enable**

to next page

from previous page

18:16	RW	Rst216	**UHS Mode Select** 000b : SDR12 001b : SDR25 010b : SDR50 011b : Reserved 100b : DDR50 other: reserved
15: 8	RO	-	**Reserved**
7	RO	Rst217	**Command Not Issue by Auto CMD12 Error** 1: Command was not executed by SDIO-Host due to previous Auto CMD12 error.
6: 5	RO	-	**Reserved**
4	RO	Rst217	**Auto CMD12 Index Error** 1: Command Index error occurs in Auto CMD12 response.
3	RO	Rst217	**Auto CMD12 End Bit Error** 1: End bit of Auto CMD12 response is 0.
2	RO	Rst217	**Auto CMD12 CRC Error** 1: CRC error in Auto CMD12 response.
1	RO	Rst217	**Auto CMD12 Timeout Error** 1: No response is returned within 64 SDCLK cycle from the end bit of Auto CMD12.
0	RW	Rst217	**Auto CMD12 Not Executed** 1: Auto CMD12 can not be issued due to some error.

Offset: 140h **SDIO140: Capability Register #1** **Init = 0**

Bit	R/W	Reset	Description
31:30	RO	-	**Slot Type** 00b : Removable Card Slot 01b : Embedded Slot for One Device 10b : Shared Bus Slot 11b : Reserved
29	RO	-	**Asynchronous Interrupt Support** 1: Support 0: Not supprot
28	RO	-	**64-bits System Bus Support** Always set to 0.
27	RO	-	**Reserved**
26	RO	-	**Voltage Support 1.8V** 1: 1.8V supported. 0: 1.8V not supported.
25	RO	-	**Voltage Support 3.0V** 1: 3.0V supported. 0: 3.0V not supported.
24	RO	-	**Voltage Support 3.3V** 1: 3.3V supported. 0: 3.3V not supported.
23	RO	-	**Suspend/Resume Support** 1: Suspend/Resume enable. 0: Suspend/Resume disable.

to next page

from previous page

22	RO	-	**SDMA Support** 1: SDMA supported. 0: SDMA not supported.
21	RO	-	**High Speed Support** 1: High Speed supported. 0: High Speed not supported.
20	RO	-	**ADMA1 Support** 1: ADMA1 supported. 0: ADMA1 not supported.
19	RO	-	**ADMA2 Support** 1: ADMA2 supported. 0: ADMA2 not supported.
18	RO	-	**8-bit Embedded Device Support** 1: Support 0: Not Support
17:16	RO	-	**Max Block Length** 00b: 512 Bytes 01b: 1024 Bytes 10b: 2048 Bytes 11b: Reserved
15:14	RO	-	**Reserved**
13: 8	RO	-	**Base Clock Frequency for SD Clock** 00h: Obtain clock information via another method. 01h: 1MHz 02h: 2MHz 03h: 3MHz ... 3Fh: 63MHz
7	RO	-	**Timeout Clock Unit** Unit of SDIO140[5:0] 0: kHz 1: MHz
6	RO	-	**Reserved**
5: 0	RO	-	**Timeout Clock Frequency** 00h: Obtain clock information via another method. 01h: 1 02h: 2 03h: 3 ... 3Fh: 63

Offset: 144h **SDIO144: Capability Register #2** **Init = 0**

Bit	R/W	Reset	Description
31:24	RO	-	**Reserved**
23:16	RO	-	**Clock Multiplier**
15:14	RO	-	**Re-Tuning Mode**
13	RO	-	**Use Tuning for SDR50**
12	RO	-	**Reserved**

to next page

from previous page

11: 8	RO	-	**Timer Count for Re-Tuning** 0h : Re-Tuning Timer disabled 1h : 1 second ... n : 2^(n-1) seconds ... Bh : 1024 seconds Ch : Reserved Dh : Reserved Eh : Reserved Fh : Obtain this info in other way
7	RO	-	**Reserved**
6	RO	-	**1.8V Line Driver Type D Supported** 1: Support 0: Not support
5	RO	-	**1.8V Line Driver Type C Supported** 1: Support 0: Not support
4	RO	-	**1.8V Line Driver Type A Supported** 1: Support 0: Not support
3	RO	-	**Reserved**
2	RO	-	**DDR50 Supported** 1: Support 0: Not support
1	RO	-	**Reserved**
0	RO	-	**SDR50 Supported** 1: Support 0: Not support

Offset: 148h **SDIO148: Maximum Current Capabilities Register** **Init = 0**

Bit	R/W	Reset	Description
31:24	RO	-	**Reserved**
23:16	RO	-	**Maximum Current for 1.8V** 00h: Obtain current information via another method. 01h: 4mA 02h: 8mA 03h: 12mA ... FFh: 1020mA
15: 8	RO	-	**Maximum Current for 3.0V** 00h: Obtain current information via another method. 01h: 4mA 02h: 8mA 03h: 12mA ... FFh: 1020mA

to next page

from previous page

7: 0	RO	-	**Maximum Current for 3.3V** 00h: Obtain current information via another method. 01h: 4mA 02h: 8mA 03h: 12mA ... FFh: 1020mA

Offset: 150h	**SDIO150: Event Trigger Register**		**Init = 0**
Bit	**R/W**	**Reset**	**Description**
31:26	RO	-	**Reserved**
25	W1T	-	**Force ADMA Error Event**
24	W1T	-	**Force Auto CMD12 Error Event**
23	W1T	-	**Force Current Limit Error Event**
22	W1T	-	**Force Data End Bit Error Event**
21	W1T	-	**Force Data CRC Error Event**
20	W1T	-	**Force Data Timeout Error Event**
19	W1T	-	**Force Command Index Error Event**
18	W1T	-	**Force Command End Bit Error Event**
17	W1T	-	**Force Command CRC Error Event**
16	W1T	-	**Force Command Timeout Error Event**
15: 8	RO	-	**Reserved**
7	W1T	-	**Force Card Removal Event**
6: 5	RO	-	**Reserved**
4	W1T	-	**Force Buffer Write Ready Event**
3	W1T	-	**Force DMA Interrupt Event**
2	W1T	-	**Force Block Gap Event Event**
1	W1T	-	**Force Transfer Complete Event**
0	W1T	-	**Force Command Complete Event**

Offset: 154h	**SDIO154: ADMA Error Status Register**		**Init = 0**
Bit	**R/W**	**Reset**	**Description**
31: 3	RO	-	**Reserved**
2	RW	Rst216	**ADMA Length Mismatch Error** This bit is set when: 1. total data length specified in ADMA descriptors is different from that specified by the Block Count and Block Length fields (if Block Count Enable is set). 2. total data length cannot be divided by the block length (if Block Count Enable is not set).
1: 0	RW	Rst216	**ADMA Error State** 00b: ADMA Stop. 01b: Fetching descriptor. 10b: not used. 11b: Transfer data.

Offset: 158h	SDIO158: ADMA Error Status Register	Init = 0

Bit	R/W	Reset	Description
31: 0	RW	Rst216	**ADMA System Address** Write: descriptor table base address. Read: 1. no error occurs: next descriptor to be fetched. 2. error occurs: SDIO154[1:0] 00b: next of the error decriptor. 01b: error descriptor 10b: not used. 11b: next of the error decriptor.

Offset: 200h	SDIO200: SDMA System Address Register	Init = 0

Bit	R/W	Reset	Description
31: 0	RW	Rst219	**SDMA System Address**

Offset: 204h	SDIO204: Block Count and Size Register	Init = 0

Bit	R/W	Reset	Description
31:16	RW	Rst219	**Block Count for Current Transfer** 0000h: 0 0001h: 1 ... FFFFh: 65535
15	RO	-	**Reserved**
14:12	RW	Rst219	**Host DMA Buffer Boundary** 000b 4kB 001b 8kB 010b 16kB 011b 32kB 100b 64kB 101b 128kB 110b 256kB 111b 512kB
11: 0	RW	Rst219	**Transfer Block Size** 800h 2048 Bytes 799h 2047 Bytes .. 002h 2 Bytes 001h 1 Byte 000h no data transfer

Offset: 208h	SDIO208: Argument Register	Init = 0

Bit	R/W	Reset	Description
31: 0	RW	Rst219	**Command Argument** Contain bit [39:8] of command argument

Offset: 20Ch	SDIO20C: Command and Transfer Mode Register	Init = 0

Bit	R/W	Reset	Description
31:30	RO	-	**Reserved**

to next page

from previous page

29:24	RW	Rst219	**Command Index** 00h: CMD0 / ACMD0 01h: CMD1 / ACMD1 ... 3Fh: CMD63 / ACMD63
23:22	RW	Rst219	**Command Type** 11b Abort Command 10b Suspend Command 01b Resume Command 00b Normal Command
21	RW	Rst219	**Data Present Select** 1: Command which transfer data 0: all other commands
20	RW	Rst219	**Command Index Check Enable** 1: Check if command index field of the response equal to SDIO20C[29:24].
19	RW	Rst219	**Command CRC Check Enable** 1: Check if CRC field of the response is valid.
18	RO	-	**Reserved**
17:16	RW	Rst219	**Response Type Select** 00b no response 01b 136-bit response 10b 48-bit response 11b 48-bit response with BUSY
15: 6	RO	-	**Reserved**
5	RW	Rst219	**Multi/Single Block Select** 1: Multi block data transfer. 0: Single block data transfer.
4	RW	Rst219	**Data Transfer Direction Select** 1: Data Read. Data from the card to the host. 0: Data Write. Data from the host to the card.
3	RO	-	**Reserved**
2	RW	Rst219	**Auto CMD12 Enable** 1: Enable. 0: Disable.
1	RW	Rst219	**Block Count Enable** 1: Enable. 0: Disable.
0	RW	Rst219	**DMA Enable** 1: Enable. 0: Disable.

Offset: 210h		**SDIO210: Response #0**	**Init = 0**
Bit	**R/W**	**Reset**	**Description**
31: 0	RO	Rst219	**Command response[31: 0]**

Offset: 214h	SDIO214: Response #1	Init = 0

Bit	R/W	Reset	Description
31: 0	RO	Rst219	**Command response[63: 32]**

Offset: 218h	SDIO218: Response #2	Init = 0

Bit	R/W	Reset	Description
31: 0	RO	Rst219	**Command response[95: 64]**

Offset: 21Ch	SDIO21C: Response #3	Init = 0

Bit	R/W	Reset	Description
31: 0	RO	Rst219	**Command response[127: 96]**

Kind of Response	Meaning of Response	Response Field	Response Register
R1, R1b(Normal Response)	Card Status	R[39: 8]	RESP[31: 0]
R1b(Auto CMD12 Response)	Card Status for Auto CMD12	R[39: 8]	RESP[127:96]
R2(CID, CSD Register)	CID or CSD register	R[127: 8]	RESP[119: 0]
R3(OCR Register)	OCR register for memory	R[39: 8]	RESP[31: 0]
R4(OCR Register)	OCR register for I/O etc	R[39: 8]	RESP[31: 0]
R5,R5b	SDIO response	R[39: 8]	RESP[31: 0]
R6(Published RCA Response)	New Published RCA[31:16] etc	R[39: 8]	RESP[31: 0]

Offset: 220h	SDIO220: Buffer Data Port Register	Init = 0

Bit	R/W	Reset	Description
31: 0	RW	-	**Buffer Data Port**

Offset: 224h	SDIO224: Present State Register	Init = 0

Bit	R/W	Reset	Description
31:25	RO	-	**Reserved**
24	RO	-	**CMD Line Signal Level** Equal to the actual signal level on CMD line of the interface.
23:20	RO	-	**DAT[3:0] Line Signal Level** Equal to the actual signal level on DAT line of the interface. SDIO224[23]: DAT[3] SDIO224[22]: DAT[2] SDIO224[21]: DAT[1] SDIO224[20]: DAT[0]
19	RO	-	**Write Protect Switch Pin Level** Equal to the actual signal level on Write Protect line of the interface.
18	RO	-	**Card Detect Switch Pin Level** Equal to the actual signal level on Card Detect line of the interface.
17	RO	-	**Card State Stable** Indicate if SDIO224[18] is stable. 1: SDIO224[18] is stable. 0: SDIO224[18] is unstable.

to next page

from previous page

16	RO	-	**Card Inserted** Indicate if the card is inserted inside the slot. 1: the card is inserted. 0: no card is inside the slot. This bit is guaranteed to be stable.
15:12	RO	-	**Reserved**
11	RO	-	**Buffer Read Enable** Shows the current data buffer state during non-DMA data read transfers. 1: valid data can be read from the data buffer. 0: no valid data inside the data buffer.
10	RO	-	**Buffer Write Enable** Shows the current data buffer state during non-DMA data write transfers. 1: data can be written to the data buffer. 0: data cna not be written.
9	RO	-	**Read Transfer Active** Indicate the status of the read data transfer. 1: data read transfer is in progress. 0: no read transfer is in progress.
8	RO	-	**Write Transfer Active** Indicate the status of the write data transfer. 1: data write transfer is in progress. 0: no write transfer is in progress.
7: 4	RO	-	**Reserved**
3	RW	-	**Re-Tining Request**
2	RO	-	**DAT Line Active** Indicate if the DAT line of the interface are currently in use. 1: DAT line is active(in use). 0: DAT line is released(not in use).
1	RO	-	**Command Inhibit DAT** Indicate if SDIO-HOST can issue a command which use DAT line. 1: command using DAT line can not be sent. 0: command using DAT line can be sent.
0	RO	-	**Command Inhibit CMD** Indicate if SDIO-HOST can issue a command. 1: command can not be sent. 0: command can be sent.

Offset: 228h **SDIO228: Host Control Settings #0 Register** **Init = 0**

Bit	R/W	Reset	Description
31:27	RO	-	**Reserved**
26	RW	Rst219	**Wakeup Event Enable on SD Card Removal** 1: Enable. 0: Disable.
25	RW	Rst219	**Wakeup Event Enable on SD Card Inserted** 1: Enable. 0: Disable.
24	RW	Rst219	**Wakeup Event Enable on SD Card Interrupt** 1: Enable. 0: Disable.

to next page

from previous page

23:20	RO	-	**Reserved**
19	RW	Rst219	**Interrupt at Block Gap** 1: Enable interrupt detection at the block gap for a multiple block transfer. This bit is only valid in SD4 mode. If the SD card can not signal an interrupt during a multiple block transfer, this bit should be set to 0.
18	RW	Rst219	**Read Wait Control** 1: Enable. 0: Disable.
17	RW	Rst219	**Continue Request** 1: Restart transfer stopped previously using SDIO228[16]. This bit is cleared automatically.
16	RW	Rst219	**Stop at Block Gap Request** 1: Stop excuting read and write transaction at the next block gap. This bit should be cleared before continuing request.
15:12	RO	-	**Reserved**
11: 9	RW	Rst219	**SD Bus Voltage Select** 111b : 3.3V 110b : 3.0V 101b : 1.8V
8	RW	Rst219	**SD Bus Power** 1: SD Device is powered. 0: SDIO-Host stops driving CMD, DAT and SDCLK. When card is removal from the slot, SDIO-Host automatically set this bit to 0.
7	RW	Rst219	**Card Detect Signal Selection** 1: SDIO228[6]. 0: From slot.
6	RW	Rst219	**Card Detect Test Level** 1: Card inserted. 0: No card.
5	RW	Rst219	**Extended Data Transfer Width** 1: 8-bit. (Same effect with SDIO000[24]) 0: Depends on SDIO228[1]
4: 3	RW	Rst219	**DMA Select** 00b: SDMA 01b: ADMA1 10b: ADMA2 11b: Reserved
2	RW	Rst219	**Hish Speed Enable** 1: Enable 0: Disable
1	RW	Rst219	**Data Transfer Width** 1: SD4 (4-bits) mode. 0: SD1 (1-bit) mode.
0	RO	-	**Reserved**

Offset: 22Ch	SDIO22C: Host Control Settings #1 Register	Init = 0

Bit	R/W	Reset	Description
31:27	RO	-	**Reserved**
26	RW	Rst42	**Software Reset for DAT Line** 1: Reset data path, including data buffer and DMA logics. This bit is cleared automatically.
25	RW	Rst42	**Software Reset for CMD Line** 1: Reset command/response generation and checking. This bit is cleared automatically.
24	RW	Rst42	**Software Reset for All** 1: Entire SDIO-Host is reset. This bit is cleared automatically.
23:20	RO	-	**Reserved**
19:16	RW	Rst219	**Data Timeout Counter Value** 1111b Reserved 1110b Period of SDMCLK * 2^27 1101b Period of SDMCLK * 2^26 ... 0001b Period of SDMCLK * 2^14 0000b Period of SDMCLK * 2^13
15: 8	RW	Rst219	**SDCLK Frequency Select** Period of SDMCLK * 2 * (SDIO22C[7:6], SDIO22C[15:8])
7: 6	RW	Rst219	**SDCLK Frequency Select (upper)**
5	RW	Rst219	**SD Generator Select** 1: Programmable Clock Mode. 0: 10-bit Divider Clock Mode.
4: 3	RO	-	**Reserved**
2	RW	Rst219	**SD Clock Enable** 1: Enable SD slot clock. 0: Disable SD slot clock.
1	RO	Rst219	**Internal Clock Stable** 1: Indicates clock pin of SDMCLK is stable. 0: Indicates clock pin of SDMCLK is not stable.
0	RW	Rst219	**Internal Clock Enable** 1: Enable SDMCLK.

Offset: 230h	SDIO230: Interrupt Status Register	Init = 0

Bit	R/W	Reset	Description
31:27	RO	-	**Reserved**
26	RW	Rst219	**Tuning Error** Read 1: Error occurs if tuning block is wrong. Write 1 to clear.
25	RW	Rst221	**ADMA Error** Read 1: Error occurs during ADMA read or write transfer. Write 1 to clear.
24	RW	Rst219	**Auto CMD12 Error** Read 1: Error occurs during Auto CMD12 command transmition. Write 1 to clear.

to next page

from previous page

23	RW	Rst219	**Current Limit Error** Read 1: SD Card is not powered due to some failure. Write 1 to clear.
22	RW	Rst219	**Data End Bit Error** Read 1: Indicate detecting 0 at the end bit position of read data which uses the DAT line, or at the end bit position of the Write CRC Status. Write 1 to clear.
21	RW	Rst219	**Data CRC Error** Read 1: indicate transferring read data which uses the DAT line, or when detecting the Write CRC status having a value of other than ”010”. Write 1 to clear.
20	RW	Rst219	**Data Timeout Error** Read 1: indicate detecting one of the following timeout conditions: 1. Busy timeout for the response with busy. 2. Busy timeout after Write CRC status. 3. Write CRC Status timeout. 4. Read data timeout. Write 1 to clear.
19	RW	Rst219	**Command Index Error** Read 1: Index error occurs in the command response. Write 1 to clear.
18	RW	Rst219	**Command End Bit Error** Read 1: Indicate detecting the end bit of a command response is 0. Write 1 to clear.
17	RW	Rst219	**Command CRC Error** Read 1: Command CRC error occurs. Write 1 to clear.
16	RW	Rst219	**Command Timeout Error** Read 1: Indicate no response was returned within 64 SDCLK cycles from the end bit of the command. Write 1 to clear.
15	RO	-	**Error Interrupt** 1: One of SDIO230[26:16] is set.
14:13	RO	-	**Reserved**
12	RO	Rst219	**Re-Tuning Event** 1: Indicate SDIO224[3] is changed from 0 to 1.
11: 9	RO	-	**Reserved**
8	RO	Rst219	**Card Interrupt** 1: Indicate the card interrupt.
7	RW	Rst219	**Card Removal** Read 1: Indicate card was removal from slot. Read 0: Indicate card is still inserted or removal or the debounce is in progress. Write 1 to clear.
6	RW	Rst219	**Card Inserted** Read 1: Indicate card was inserted to slot. Read 0: Indicate card is still inserted or removal or the debounce is in progress. Write 1 to clear.
5	RW	Rst221	**Buffer Read Ready** Read 1: Indicate data buffer can be read. Write 1 to clear.

to next page

from previous page

4	RW	Rst221	**Buffer Write Ready** Read 1: Indicate data buffer can be written. Write 1 to clear.
3	RW	Rst221	**DMA Interrupt** In SDMA mode, DMA interrupt is generated when the Host Controller detects the Host SDMA Buffer boundary. In ADMA mode, DMA interrupt is generated when the INT flag is set in a currently serviced ADMA descriptor. Write 1 to clear.
2	RW	Rst221	**Block Gap Event** Read 1: Indicate read/write transaction is stopped at a block gap. Write 1 to clear.
1	RW	Rst221	**Transfer Complete** Read 1: Indicate transfer using DAT line is completed. Write 1 to clear.
0	RW	Rst220	**Command Complete** Read 1: Indicate the end bit of the response is received, except the response for Auto CMD12 command. Write 1 to clear.

Offset: 234h **SDIO234: Interrupt Status Enable Register** **Init = 0**

Bit	R/W	Reset	Description
31:26	RO	-	**Reserved**
26	RW	Rst219	**Tuning Error Status Enable** 1: Enable. 0: Masked.
25	RW	Rst219	**ADMA Error Status Enable** 1: Enable. 0: Masked.
24	RW	Rst219	**Auto CMD12 Error Status Enable** 1: Enable. 0: Masked.
23	RW	Rst219	**Current Limit Error Status Enable** 1: Enable. 0: Masked.
22	RW	Rst219	**Data End Bit Error Status Enable** 1: Enable. 0: Masked.
21	RW	Rst219	**Data CRC Error Status Enable** 1: Enable. 0: Masked.
20	RW	Rst219	**Data Timeout Error Status Enable** 1: Enable. 0: Masked.
19	RW	Rst219	**Command Index Error Status Enable** 1: Enable. 0: Masked.
18	RW	Rst219	**Command End Bit Error Status Enable** 1: Enable. 0: Masked.

to next page

from previous page

17	RW	Rst219	**Command CRC Error Status Enable** 1: Enable. 0: Masked.
16	RW	Rst219	**Command Timeout Error Status Enable** 1: Enable. 0: Masked.
15:13	RO	-	**Reserved**
12	RW	Rst219	**Re-Tuning Event Status Enable** 1: Enable. 0: Masked.
11: 9	RO	-	**Reserved**
8	RW	Rst219	**Card Interrupt Status Enable** 1: Enable. 0: Masked.
7	RW	Rst219	**Card Removal Status Enable** 1: Enable. 0: Masked.
6	RW	Rst219	**Card Inserted Status Enable** 1: Enable. 0: Masked.
5	RW	Rst219	**Buffer Read Ready Status Enable** 1: Enable. 0: Masked.
4	RW	Rst219	**Buffer Write Ready Status Enable** 1: Enable. 0: Masked.
3	RW	Rst219	**DMA Interrupt Status Enable** 1: Enable. 0: Masked.
2	RW	Rst219	**Block Gap Event Status Enable** 1: Enable. 0: Masked.
1	RW	Rst219	**Transfer Complete Status Enable** 1: Enable. 0: Masked.
0	RW	Rst219	**Command Complete Status Enable** 1: Enable. 0: Masked.

Offset: 238h **SDIO238: Interrupt Enable Register** **Init = 0**

Bit	R/W	Reset	Description
31:27	RO	-	**Reserved**
26	RW	Rst219	**Tuning Error Interrupt Enable** 1: Enable. 0: Masked.
25	RW	Rst219	**ADMA Error Interrupt Enable** 1: Enable. 0: Masked.

to next page

from previous page

24	RW	Rst219	**Auto CMD12 Error Interrupt Enable** 1: Enable. 0: Masked.
23	RW	Rst219	**Current Limit Error Interrupt Enable** 1: Enable. 0: Masked.
22	RW	Rst219	**Data End Bit Error Interrupt Enable** 1: Enable. 0: Masked.
21	RW	Rst219	**Data CRC Error Interrupt Enable** 1: Enable. 0: Masked.
20	RW	Rst219	**Data Timeout Error Interrupt Enable** 1: Enable. 0: Masked.
19	RW	Rst219	**Command Index Error Interrupt Enable** 1: Enable. 0: Masked.
18	RW	Rst219	**Command End Bit Error Interrupt Enable** 1: Enable. 0: Masked.
17	RW	Rst219	**Command CRC Error Interrupt Enable** 1: Enable. 0: Masked.
16	RW	Rst219	**Command Timeout Error Interrupt Enable** 1: Enable. 0: Masked.
15:13	RO	-	**Reserved**
12	RW	Rst219	**Re-Tuning Event Interrupt Enable** 1: Enable. 0: Masked.
11: 9	RO	-	**Reserved**
8	RO	Rst219	**Card Interrupt Interrupt Enable** 1: Enable. 0: Masked.
7	RW	Rst219	**Card Removal Interrupt Enable** 1: Enable. 0: Masked.
6	RW	Rst219	**Card Inserted Interrupt Enable** 1: Enable. 0: Masked.
5	RW	Rst219	**Buffer Read Ready Interrupt Enable** 1: Enable. 0: Masked.
4	RW	Rst219	**Buffer Write Ready Interrupt Enable** 1: Enable. 0: Masked.

to next page

from previous page

3	RW	Rst219	**DMA Interrupt Interrupt Enable** 1: Enable. 0: Masked.
2	RW	Rst219	**Block Gap Event Interrupt Enable** 1: Enable. 0: Masked.
1	RW	Rst219	**Transfer Complete Interrupt Enable** 1: Enable. 0: Masked.
0	RW	Rst219	**Command Complete Interrupt Enable** 1: Enable. 0: Masked.

Offset: 23Ch **SDIO23C: Auto CMD12 Error Status Register** **Init = 0**

Bit	R/W	Reset	Description
31	RW	Rst219	**Preset Value Enable**
30	RW	Rst219	**Asynchronous Interrupt Enable**
29:24	RO	-	**Reserved**
23	RW	Rst219	**Sampling Clock Select** Write 1: no operation. Write 0: reset and disable tuning block. Read 1: tuning procedure completes.
22	RW	Rst219	**Execute Tuning**
21:20	RW	Rst219	**Driver Strength Select**
19	RW	Rst219	**1.8V Signaling Enable**
18:16	RW	Rst219	**UHS Mode Select** 000b : SDR12 001b : SDR25 010b : SDR50 011b : Reserved 100b : DDR50 other: reserved
15: 8	RO	-	**Reserved**
7	RO	Rst220	**Command Not Issue by Auto CMD12 Error** 1: Command was not executed by SDIO-Host due to previous Auto CMD12 error.
6: 5	RO	-	**Reserved**
4	RO	Rst220	**Auto CMD12 Index Error** 1: Command Index error occurs in Auto CMD12 response.
3	RO	Rst220	**Auto CMD12 End Bit Error** 1: End bit of Auto CMD12 response is 0.
2	RO	Rst220	**Auto CMD12 CRC Error** 1: CRC error in Auto CMD12 response.
1	RO	Rst220	**Auto CMD12 Timeout Error** 1: No response is returned within 64 SDCLK cycle from the end bit of Auto CMD12.
0	RW	Rst220	**Auto CMD12 Not Executed** 1: Auto CMD12 can not be issued due to some error.

Offset: 240h			SDIO240: Capability Register #1	Init = 0
Bit	**R/W**	**Reset**	**Description**	
31:30	RO	-	**Slot Type** 00b : Removable Card Slot 01b : Embedded Slot for One Device 10b : Shared Bus Slot 11b : Reserved	
29	RO	-	**Asynchronous Interrupt Support** 1: Support 0: Not supprot	
28	RO	-	**64-bits System Bus Support** Always set to 0.	
27	RO	-	**Reserved**	
26	RO	-	**Voltage Support 1.8V** 1: 1.8V supported. 0: 1.8V not supported.	
25	RO	-	**Voltage Support 3.0V** 1: 3.0V supported. 0: 3.0V not supported.	
24	RO	-	**Voltage Support 3.3V** 1: 3.3V supported. 0: 3.3V not supported.	
23	RO	-	**Suspend/Resume Support** 1: Suspend/Resume enable. 0: Suspend/Resume disable.	
22	RO	-	**SDMA Support** 1: SDMA supported. 0: SDMA not supported.	
21	RO	-	**High Speed Support** 1: High Speed supported. 0: High Speed not supported.	
20	RO	-	**ADMA1 Support** 1: ADMA1 supported. 0: ADMA1 not supported.	
19	RO	-	**ADMA2 Support** 1: ADMA2 supported. 0: ADMA2 not supported.	
18	RO	-	**8-bit Embedded Device Support** 1: Support 0: Not Support	
17:16	RO	-	**Max Block Length** 00b: 512 Bytes 01b: 1024 Bytes 10b: 2048 Bytes 11b: Reserved	
15:14	RO	-	**Reserved**	

to next page

from previous page

13: 8	RO	-	**Base Clock Frequency for SD Clock** 00h: Obtain clock information via another method. 01h: 1MHz 02h: 2MHz 03h: 3MHz ... 3Fh: 63MHz
7	RO	-	**Timeout Clock Unit** Unit of SDIO240[5:0] 0: kHz 1: MHz
6	RO	-	**Reserved**
5: 0	RO	-	**Timeout Clock Frequency** 00h: Obtain clock information via another method. 01h: 1 02h: 2 03h: 3 ... 3Fh: 63

Offset: 244h **SDIO244: Capability Register #2** **Init = 0**

Bit	R/W	Reset	Description
31:24	RO	-	**Reserved**
23:16	RO	-	**Clock Multiplier**
15:14	RO	-	**Re-Tuning Mode**
13	RO	-	**Use Tuning for SDR50**
12	RO	-	**Reserved**
11: 8	RO	-	**Timer Count for Re-Tuning** 0h : Re-Tuning Timer disabled 1h : 1 second ... n : 2^(n-1) seconds ... Bh : 1024 seconds Ch : Reserved Dh : Reserved Eh : Reserved Fh : Obtain this info in other way
7	RO	-	**Reserved**
6	RO	-	**1.8V Line Driver Type D Supported** 1: Support 0: Not support
5	RO	-	**1.8V Line Driver Type C Supported** 1: Support 0: Not support
4	RO	-	**1.8V Line Driver Type A Supported** 1: Support 0: Not support
3	RO	-	**Reserved**
2	RO	-	**DDR50 Supported** 1: Support 0: Not support

to next page

from previous page

1	RO	-	**Reserved**
0	RO	-	**SDR50 Supported** 1: Support 0: Not support

Offset: 248h **SDIO248: Maximum Current Capabilities Register** **Init = 0**

Bit	R/W	Reset	Description
31:24	RO	-	**Reserved**
23:16	RO	-	**Maximum Current for 1.8V** 00h: Obtain current information via another method. 01h: 4mA 02h: 8mA 03h: 12mA ... FFh: 1020mA
15: 8	RO	-	**Maximum Current for 3.0V** 00h: Obtain current information via another method. 01h: 4mA 02h: 8mA 03h: 12mA ... FFh: 1020mA
7: 0	RO	-	**Maximum Current for 3.3V** 00h: Obtain current information via another method. 01h: 4mA 02h: 8mA 03h: 12mA ... FFh: 1020mA

Offset: 250h **SDIO250: Event Trigger Register** **Init = 0**

Bit	R/W	Reset	Description
31:26	RO	-	**Reserved**
25	W1T	-	**Force ADMA Error Event**
24	W1T	-	**Force Auto CMD12 Error Event**
23	W1T	-	**Force Current Limit Error Event**
22	W1T	-	**Force Data End Bit Error Event**
21	W1T	-	**Force Data CRC Error Event**
20	W1T	-	**Force Data Timeout Error Event**
19	W1T	-	**Force Command Index Error Event**
18	W1T	-	**Force Command End Bit Error Event**
17	W1T	-	**Force Command CRC Error Event**
16	W1T	-	**Force Command Timeout Error Event**
15: 8	RO	-	**Reserved**
7	W1T	-	**Force Card Removal Event**
6: 5	RO	-	**Reserved**

to next page

from previous page

4	W1T	-	**Force Buffer Write Ready Event**
3	W1T	-	**Force DMA Interrupt Event**
2	W1T	-	**Force Block Gap Event Event**
1	W1T	-	**Force Transfer Complete Event**
0	W1T	-	**Force Command Complete Event**

Offset: 254h **SDIO254: ADMA Error Status Register** **Init = 0**

Bit	R/W	Reset	Description
31: 3	RO	-	**Reserved**
2	RW	Rst219	**ADMA Length Mismatch Error** This bit is set when: 1. total data length specified in ADMA descriptors is different from that specified by the Block Count and Block Length fields (if Block Count Enable is set). 2. total data length cannot be divided by the block length (if Block Count Enable is not set).
1: 0	RW	Rst219	**ADMA Error State** 00b: ADMA Stop. 01b: Fetching descriptor. 10b: not used. 11b: Transfer data.

Offset: 258h **SDIO258: ADMA Error Status Register** **Init = 0**

Bit	R/W	Reset	Description
31: 0	RW	Rst219	**ADMA System Address** Write: descriptor table base address. Read: 1. no error occurs: next descriptor to be fetched. 2. error occurs: SDIO254[1:0] 00b: next of the error decriptor. 01b: error descriptor 10b: not used. 11b: next of the error decriptor.

39 eMMC Controller (EMMC)

39.1 Overview

Base Address of eMMC/SDIO Controller = 0x1E75_0000
Physical address of register = (Base address of EMMC Controller) + Offset

EMMC000: General Information Register
EMMC004: Debounce Settings Register
EMMC008: Bus Settings Register
EMMC010: HWInit EMMC140 Configuration for slot #0
EMMC014: HWInit EMMC144 Configuration for slot #0
EMMC018: HWInit EMMC148 Configuration for slot #0
EMMC0F0: Card Detect Control
EMMC0F4: Clock Phase Control
EMMC0FC: Interrupt Status

Slot Register Set for slot 0

EMMC100: SDMA System Address Register
EMMC104: Block Count and Size Register
EMMC108: Argument Register
EMMC10C: Command and Transfer Mode Register
EMMC110: Response #0
EMMC114: Response #1
EMMC118: Response #2
EMMC11C: Response #3
EMMC120: Buffer Data Port Register
EMMC124: Present State Register
EMMC128: Host Control Settings Register #0
EMMC12C: Host Control Settings Register #1
EMMC130: Interrupt Status Register
EMMC134: Interrupt Status Enable Register
EMMC138: Interrupt Signal Enable Register
EMMC13C: Auto CMD12 Error Status Register
EMMC140: Capabilities Register #1
EMMC144: Capabilities Register #2
EMMC148: Maximum Current Capabilities Register
EMMC150: Force Event Register
EMMC154: ADMA Error Status Register
EMMC158: ADMA System Address Register

39.2 Features

- Compatibility
  - SD Memory Card Version 3.00
  - EMMC Version 3.00
  - eMMC Version 5.1
- Support 1 slot.
- SD1/SD4 or eMMC1/eMMC4/eMMC8 modes of operation
- Integrated ADMA2 controllers.
- Master device side clock and slot's clock can be switched off.

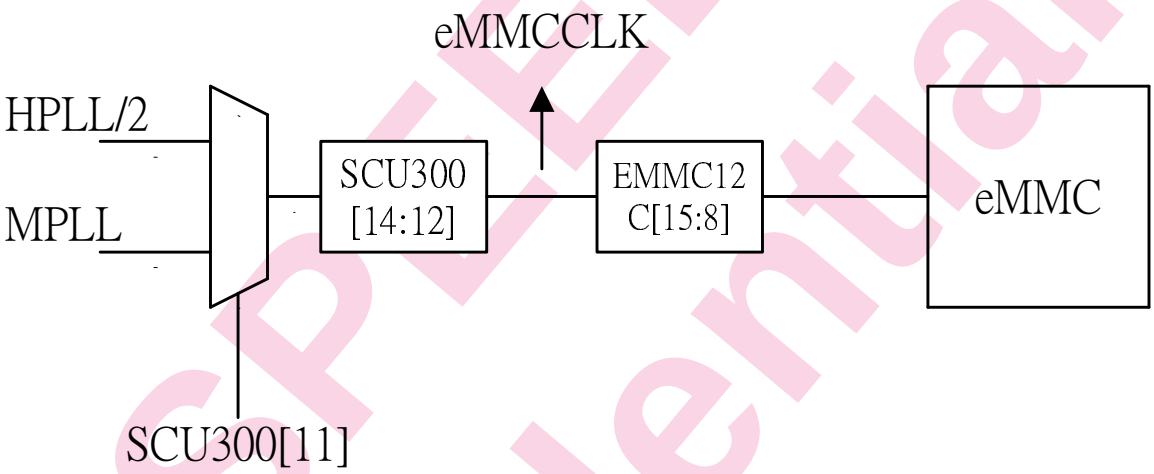


Figure 52: Clock structure of eMMC interface

39.3 Registers : Base Address = 0x1E75:0000

Offset: 000h	EMMC000: General Information Register	Init = 0x0001_0000

Bit	R/W	Reset	Description
31:25	RO	-	**Reserved**
24	RW	Rst25	**S0MMC8** 1: 8-bit mode 0: 1 or 4 bit mode (depends on EMMC128[1])
23:17	RO	-	**Reserved**
16	RO	-	**Slot #0 available** 1: slot is available 0: slot is not available
15: 1	RO	-	**Reserved**
0	RW1S	Rst25	**Software Reset** Set to reset all flip-flops in every slot. This bit will automatically clear. After set this bit, software should wait it is cleared to continue.

Offset: 004h	EMMC004: Debounce Settings Register	Init = 0x0000_0005

Bit	R/W	Reset	Description
31:24	RO	-	**Reserved**
23: 0	RW	Rst25	**Debounce Period** Debounce period = EMMC004[23:0] * period of HCLK

Offset: 008h	EMMC008: Bus Settings Register	Init = 0

Bit	R/W	Reset	Description
31: 4	RO	-	**Reserved**
3: 0	RW	Rst25	**Programmable Burst Length** Maximum number of beats within single DMA burst. Value Number of beats 0001b 1 0010b 2 0011b 4 0100b 8 0101b 16 0110b 32 0111b 64 1000b 128 1001b 256 1010b 512 1011b 1024 others 2048

Offset: 010h	EMMC010: HWInit EMMC140 Configuration for slot #0	Init = 0

Bit	R/W	Reset	Description
31: 0	RW	Rst25	**Mirror of EMMC140** Write to update corresponding EMMC140 value

Offset: 014h	EMMC014: HWInit EMMC144 Configuration for slot #0	Init = 0

Bit	R/W	Reset	Description
31: 0	RW	Rst25	**Mirror of EMMC144** Write to update corresponding EMMC144 value

Offset: 018h	EMMC018: HWInit EMMC148 Configuration for slot #0	Init = 0

Bit	R/W	Reset	Description
31: 0	RW	Rst25	**Mirror of EMMC148** Write to update corresponding EMMC148 value

Offset: 0F0h	EMMC0F0: Card Detect Control	Init = 0

Bit	R/W	Reset	Description
31: 1	RO	-	**Reserved**
0	RW	Rst25	**Slot 0 Write Protection Inverse bit** 1: Inverse. 0: No operation.

Offset: 0F4h	EMMC0F4: Clock Phase Control	Init = 0x101

Bit	R/W	Reset	Description
31:26	RO	-	**Reserved**
25:21	RW	Rst42	**Slot 1 Input Clock Phase**
20:16	RW	Rst42	**Slot 0 Input Clock Phase**
15:11	RW	Rst42	**Slot 1 Output Clock Phase**
10	RW	Rst42	**Slot 1 Input Clock Phase Selection** 1: Enable. 0: Disable.
9: 8	RW	Rst42	**Slot 1 Output Clock Phase Selection** 11b : Enable. Others: Disable.
7: 3	RW	Rst42	**Slot 0 Output Clock Phase**
2	RW	Rst42	**Slot 0 Input Clock Phase Selection** 1: Enable. 0: Disable.
1: 0	RW	Rst42	**Slot 0 Output Clock Phase Selection** 11b : Enable. Others: Disable.

Offset: 0FCh	EMMC0FC: Interrupt Status	Init = 0x00010000

Bit	R/W	Reset	Description
31: 1	RO	-	**Reserved**
0	RW	Rst25	**Interrupt Status** Bit 0: Interrput pending in Slot0.

Offset: 100h	EMMC100: SDMA System Address Register	Init = 0

Bit	R/W	Reset	Description
31: 0	RW	Rst213	**SDMA System Address**

Offset: 104h	EMMC104: Block Count and Size Register	Init = 0

Bit	R/W	Reset	Description
31:16	RW	Rst213	**Block Count for Current Transfer** 0000h: 0 0001h: 1 ... FFFFh: 65535
15	RO	-	**Reserved**
14:12	RW	Rst213	**Host DMA Buffer Boundary** 000b 4kB 001b 8kB 010b 16kB 011b 32kB 100b 64kB 101b 128kB 110b 256kB 111b 512kB
11: 0	RW	Rst213	**Transfer Block Size** 800h 2048 Bytes 799h 2047 Bytes .. 002h 2 Bytes 001h 1 Byte 000h no data transfer

Offset: 108h	EMMC108: Argument Register	Init = 0

Bit	R/W	Reset	Description
31: 0	RW	Rst213	**Command Argument** Contain bit [39:8] of command argument

Offset: 10Ch	EMMC10C: Command and Transfer Mode Register	Init = 0

Bit	R/W	Reset	Description
31:30	RO	-	**Reserved**
29:24	RW	Rst213	**Command Index** 00h: CMD0 / ACMD0 01h: CMD1 / ACMD1 ... 3Fh: CMD63 / ACMD63

to next page

from previous page

23:22	RW	Rst213	**Command Type** 11b Abort Command 10b Suspend Command 01b Resume Command 00b Normal Command
21	RW	Rst213	**Data Present Select** 1: Command which transfer data 0: all other commands
20	RW	Rst213	**Command Index Check Enable** 1: Check if command index field of the response equal to EMMC10C[29:24].
19	RW	Rst213	**Command CRC Check Enable** 1: Check if CRC field of the response is valid.
18	RO	-	**Reserved**
17:16	RW	Rst213	**Response Type Select** 00b no response 01b 136-bit response 10b 48-bit response 11b 48-bit response with BUSY
15: 6	RO	-	**Reserved**
5	RW	Rst213	**Multi/Single Block Select** 1: Multi block data transfer. 0: Single block data transfer.
4	RW	Rst213	**Data Transfer Direction Select** 1: Data Read. Data from the card to the host. 0: Data Write. Data from the host to the card.
3	RO	-	**Reserved**
2	RW	Rst213	**Auto CMD12 Enable** 1: Enable. 0: Disable.
1	RW	Rst213	**Block Count Enable** 1: Enable. 0: Disable.
0	RW	Rst213	**DMA Enable** 1: Enable. 0: Disable.

Offset: 110h		**EMMC110: Response #0**	**Init = 0**
Bit	**R/W**	**Reset**	**Description**
31: 0	RO	Rst213	**Command response[31: 0]**

Offset: 114h		**EMMC114: Response #1**	**Init = 0**
Bit	**R/W**	**Reset**	**Description**
31: 0	RO	Rst213	**Command response[63: 32]**

Offset: 118h	EMMC118: Response #2	Init = 0

Bit	R/W	Reset	Description
31: 0	RO	Rst213	**Command response[95: 64]**

Offset: 11Ch	EMMC11C: Response #3	Init = 0

Bit	R/W	Reset	Description
31: 0	RO	Rst213	**Command response[127: 96]**

Kind of Response	Meaning of Response	Response Field	Response Register
R1, R1b(Normal Response)	Card Status	R[39: 8]	RESP[31: 0]
R1b(Auto CMD12 Response)	Card Status for Auto CMD12	R[39: 8]	RESP[127:96]
R2(CID, CSD Register)	CID or CSD register	R[127: 8]	RESP[119: 0]
R3(OCR Register)	OCR register for memory	R[39: 8]	RESP[31: 0]
R4(OCR Register)	OCR register for I/O etc	R[39: 8]	RESP[31: 0]
R5,R5b	EMMC response	R[39: 8]	RESP[31: 0]
R6(Published RCA Response)	New Published RCA[31:16] etc	R[39: 8]	RESP[31: 0]

Offset: 120h	EMMC120: Buffer Data Port Register	Init = 0

Bit	R/W	Reset	Description
31: 0	RW	Rst213	**Buffer Data Port**

Offset: 124h	EMMC124: Present State Register	Init = 0

Bit	R/W	Reset	Description
31:25	RO	-	**Reserved**
24	RO	-	**CMD Line Signal Level** Equal to the actual signal level on CMD line of the interface.
23:20	RO	-	**DAT[3:0] Line Signal Level** Equal to the actual signal level on DAT line of the interface. EMMC124[23]: DAT[3] EMMC124[22]: DAT[2] EMMC124[21]: DAT[1] EMMC124[20]: DAT[0]
19	RO	-	**Write Protect Switch Pin Level** Equal to the actual signal level on Write Protect line of the interface.
18	RO	-	**Card Detect Switch Pin Level** Equal to the actual signal level on Card Detect line of the interface.
17	RO	-	**Card State Stable** Indicate if EMMC124[18] is stable. 1: EMMC124[18] is stable. 0: EMMC124[18] is unstable.
16	RO	-	**Card Inserted** Indicate if the card is inserted inside the slot. 1: the card is inserted. 0: no card is inside the slot. This bit is guaranteed to be stable.
15:12	RO	-	**Reserved**

to next page

from previous page

11	RO	-	**Buffer Read Enable** Shows the current data buffer state during non-DMA data read transfers. 1: valid data can be read from the data buffer. 0: no valid data inside the data buffer.
10	RO	-	**Buffer Write Enable** Shows the current data buffer state during non-DMA data write transfers. 1: data can be written to the data buffer. 0: data cna not be written.
9	RO	-	**Read Transfer Active** Indicate the status of the read data transfer. 1: data read transfer is in progress. 0: no read transfer is in progress.
8	RO	-	**Write Transfer Active** Indicate the status of the write data transfer. 1: data write transfer is in progress. 0: no write transfer is in progress.
7: 4	RO	-	**Reserved**
3	RW	-	**Re-Tining Request**
2	RO	-	**DAT Line Active** Indicate if the DAT line of the interface are currently in use. 1: DAT line is active(in use). 0: DAT line is released(not in use).
1	RO	-	**Command Inhibit DAT** Indicate if EMMC-HOST can issue a command which use DAT line. 1: command using DAT line can not be sent. 0: command using DAT line can be sent.
0	RO	-	**Command Inhibit CMD** Indicate if EMMC-HOST can issue a command. 1: command can not be sent. 0: command can be sent.

Offset: 128h **EMMC128: Host Control Settings #0 Register** **Init = 0**

Bit	R/W	Reset	Description
31:27	RO	-	**Reserved**
26	RW	Rst213	**Wakeup Event Enable on SD Card Removal** 1: Enable. 0: Disable.
25	RW	Rst213	**Wakeup Event Enable on SD Card Inserted** 1: Enable. 0: Disable.
24	RW	Rst213	**Wakeup Event Enable on SD Card Interrupt** 1: Enable. 0: Disable.
23:20	RO	-	**Reserved**
19	RW	Rst213	**Interrupt at Block Gap** 1: Enable interrupt detection at the block gap for a multiple block transfer. This bit is only valid in SD4 mode. If the SD card can not signal an interrupt during a multiple block transfer, this bit should be set to 0.

to next page

from previous page

18	RW	Rst213	**Read Wait Control** 1: Enable. 0: Disable.
17	RW	Rst213	**Continue Request** 1: Restart transfer stopped previously using EMMC128[16]. This bit is cleared automatically.
16	RW	Rst213	**Stop at Block Gap Request** 1: Stop excuting read and write transaction at the next block gap. This bit should be cleared before continuing request.
15:12	RO	-	**Reserved**
11: 9	RW	Rst213	**SD Bus Voltage Select** 111b : 3.3V 110b : 3.0V 101b : 1.8V
8	RW	Rst213	**SD Bus Power** 1: SD Device is powered. 0: EMMC-Host stops driving CMD, DAT and SDCLK. When card is removal from the slot, EMMC-Host automatically set this bit to 0.
7	RW	Rst213	**Card Detect Signal Selection** 1: EMMC128[6]. 0: From slot.
6	RW	Rst213	**Card Detect Test Level** 1: Card inserted. 0: No card.
5	RW	Rst213	**Extended Data Transfer Width** 1: 8-bit. (Same effect with EMMC000[24]) 0: Depends on EMMC128[1]
4: 3	RW	Rst213	**DMA Select** 00b: SDMA 01b: ADMA1 10b: ADMA2 11b: Reserved
2	RW	Rst213	**Hish Speed Enable** 1: Enable 0: Disable
1	RW	Rst213	**Data Transfer Width** 1: SD4 (4-bits) mode. 0: SD1 (1-bit) mode.
0	RO	-	**Reserved**

Offset: 12Ch **EMMC12C: Host Control Settings #1 Register** **Init = 0**

Bit	R/W	Reset	Description
31:27	RO	-	**Reserved**
26	RW	Rst25	**Software Reset for DAT Line** 1: Reset data path, including data buffer and DMA logics. This bit is NOT cleared automatically.
25	RW	Rst25	**Software Reset for CMD Line** 1: Reset command/response generation and checking. This bit is cleared automatically.

to next page

from previous page

24	RW	Rst25	**Software Reset for All** 1: Entire EMMC-Host is reset. This bit is NOT cleared automatically.
23:20	RO	-	**Reserved**
19:16	RW	Rst213	**Data Timeout Counter Value** 1111b Reserved 1110b Period of SDMCLK * 2^27 1101b Period of SDMCLK * 2^26 ... 0001b Period of SDMCLK * 2^14 0000b Period of SDMCLK * 2^13
15: 8	RW	Rst213	**SDCLK Frequency Select** Period of SDMCLK * 2 * (EMMC12C[7:6], EMMC12C[15:8])
7: 6	RW	Rst213	**SDCLK Frequency Select (upper)**
5	RW	Rst213	**SD Generator Select** 1: Programmable Clock Mode. 0: 10-bit Divider Clock Mode.
4: 3	RO	-	**Reserved**
2	RW	Rst213	**SD Clock Enable** 1: Enable SD slot clock. 0: Disable SD slot clock.
1	RO	-	**Internal Clock Stable** 1: Indicates clock pin of SDMCLK is stable. 0: Indicates clock pin of SDMCLK is not stable.
0	RW	Rst213	**Internal Clock Enable** 1: Enable SDMCLK.

Offset: 130h **EMMC130: Interrupt Status Register** **Init = 0**

Bit	R/W	Reset	Description
31:27	RO	-	**Reserved**
26	RW1C	Rst213	**Tuning Error** Read 1: Error occurs if tuning block is wrong. Write 1 to clear.
25	RW1C	Rst215	**ADMA Error** Read 1: Error occurs during ADMA read or write transfer. Write 1 to clear.
24	RW1C	Rst213	**Auto CMD12 Error** Read 1: Error occurs during Auto CMD12 command transmition. Write 1 to clear.
23	RW1C	Rst213	**Current Limit Error** Read 1: SD Card is not powered due to some failure. Write 1 to clear.
22	RW1C	Rst213	**Data End Bit Error** Read 1: Indicate detecting 0 at the end bit position of read data which uses the DAT line, or at the end bit position of the Write CRC Status. Write 1 to clear.
21	RW1C	Rst213	**Data CRC Error** Read 1: indicate transferring read data which uses the DAT line, or when detecting the Write CRC status having a value of other than ”010”. Write 1 to clear.

to next page

from previous page

20	RW1C	Rst213	**Data Timeout Error** Read 1: indicate detecting one of the following timeout conditions: 1. Busy timeout for the response with busy. 2. Busy timeout after Write CRC status. 3. Write CRC Status timeout. 4. Read data timeout. Write 1 to clear.
19	RW1C	Rst213	**Command Index Error** Read 1: Index error occurs in the command response. Write 1 to clear.
18	RW1C	Rst213	**Command End Bit Error** Read 1: Indicate detecting the end bit of a command response is 0. Write 1 to clear.
17	RW1C	Rst213	**Command CRC Error** Read 1: Command CRC error occurs. Write 1 to clear.
16	RW1C	Rst213	**Command Timeout Error** Read 1: Indicate no response was returned within 64 SDCLK cycles from the end bit of the command. Write 1 to clear.
15	RO	-	**Error Interrupt** 1: One of EMMC130[26:16] is set.
14:13	RO	-	**Reserved**
12	RW1C	Rst213	**Re-Tuning Event** 1: Indicate EMMC124[3] is changed from 0 to 1.
11: 9	RO	-	**Reserved**
8	RO	-	**Card Interrupt** 1: Indicate the card interrupt.
7	RW1C	Rst213	**Card Removal** Read 1: Indicate card was removal from slot. Read 0: Indicate card is still inserted or removal or the debounce is in progress. Write 1 to clear.
6	RW1C	Rst213	**Card Inserted** Read 1: Indicate card was inserted to slot. Read 0: Indicate card is still inserted or removal or the debounce is in progress. Write 1 to clear.
5	RW1C	Rst215	**Buffer Read Ready** Read 1: Indicate data buffer can be read. Write 1 to clear.
4	RW1C	Rst215	**Buffer Write Ready** Read 1: Indicate data buffer can be written. Write 1 to clear.
3	RW1C	Rst215	**DMA Interrupt** In SDMA mode, DMA interrupt is generated when the Host Controller detects the Host SDMA Buffer boundary. In ADMA mode, DMA interrupt is generated when the INT flag is set in a currently serviced ADMA descriptor. Write 1 to clear.

to next page

from previous page

2	RW1C	Rst215	**Block Gap Event** Read 1: Indicate read/write transaction is stopped at a block gap. Write 1 to clear.
1	RW1C	Rst215	**Transfer Complete** Read 1: Indicate transfer using DAT line is completed. Write 1 to clear.
0	RW1C	Rst214	**Command Complete** Read 1: Indicate the end bit of the response is received, except the response for Auto CMD12 command. Write 1 to clear.

Offset: 134h **EMMC134: Interrupt Status Enable Register** **Init = 0**

Bit	R/W	Reset	Description
31:27	RO	-	**Reserved**
26	RW	Rst213	**Tuning Error Status Enable** 1: Enable. 0: Masked.
25	RW	Rst213	**ADMA Error Status Enable** 1: Enable. 0: Masked.
24	RW	Rst213	**Auto CMD12 Error Status Enable** 1: Enable. 0: Masked.
23	RW	Rst213	**Current Limit Error Status Enable** 1: Enable. 0: Masked.
22	RW	Rst213	**Data End Bit Error Status Enable** 1: Enable. 0: Masked.
21	RW	Rst213	**Data CRC Error Status Enable** 1: Enable. 0: Masked.
20	RW	Rst213	**Data Timeout Error Status Enable** 1: Enable. 0: Masked.
19	RW	Rst213	**Command Index Error Status Enable** 1: Enable. 0: Masked.
18	RW	Rst213	**Command End Bit Error Status Enable** 1: Enable. 0: Masked.
17	RW	Rst213	**Command CRC Error Status Enable** 1: Enable. 0: Masked.
16	RW	Rst213	**Command Timeout Error Status Enable** 1: Enable. 0: Masked.
15:13	RO	-	**Reserved**

to next page

from previous page

12	RW	Rst213	**Re-Tuning Event Status Enable** 1: Enable. 0: Masked.
11: 9	RO	-	**Reserved**
8	RW	Rst213	**Card Interrupt Status Enable** 1: Enable. 0: Masked.
7	RW	Rst213	**Card Removal Status Enable** 1: Enable. 0: Masked.
6	RW	Rst213	**Card Inserted Status Enable** 1: Enable. 0: Masked.
5	RW	Rst213	**Buffer Read Ready Status Enable** 1: Enable. 0: Masked.
4	RW	Rst213	**Buffer Write Ready Status Enable** 1: Enable. 0: Masked.
3	RW	Rst213	**DMA Interrupt Status Enable** 1: Enable. 0: Masked.
2	RW	Rst213	**Block Gap Event Status Enable** 1: Enable. 0: Masked.
1	RW	Rst213	**Transfer Complete Status Enable** 1: Enable. 0: Masked.
0	RW	Rst213	**Command Complete Status Enable** 1: Enable. 0: Masked.

Offset: 138h **EMMC138: Interrupt Enable Register** **Init = 0**

Bit	R/W	Reset	Description
31:27	RO	-	**Reserved**
26	RW	Rst213	**Tuning Error Interrupt Enable** 1: Enable. 0: Masked.
25	RW	Rst213	**ADMA Error Interrupt Enable** 1: Enable. 0: Masked.
24	RW	Rst213	**Auto CMD12 Error Interrupt Enable** 1: Enable. 0: Masked.
23	RW	Rst213	**Current Limit Error Interrupt Enable** 1: Enable. 0: Masked.
22	RW	Rst213	**Data End Bit Error Interrupt Enable** 1: Enable. 0: Masked.

to next page

from previous page

21	RW	Rst213	**Data CRC Error Interrupt Enable** 1: Enable. 0: Masked.
20	RW	Rst213	**Data Timeout Error Interrupt Enable** 1: Enable. 0: Masked.
19	RW	Rst213	**Command Index Error Interrupt Enable** 1: Enable. 0: Masked.
18	RW	Rst213	**Command End Bit Error Interrupt Enable** 1: Enable. 0: Masked.
17	RW	Rst213	**Command CRC Error Interrupt Enable** 1: Enable. 0: Masked.
16	RW	Rst213	**Command Timeout Error Interrupt Enable** 1: Enable. 0: Masked.
15:13	RO	-	**Reserved**
12	RW	Rst213	**Re-Tuning Event Interrupt Enable** 1: Enable. 0: Masked.
11: 9	RO	-	**Reserved**
8	RO	Rst213	**Card Interrupt Interrupt Enable** 1: Enable. 0: Masked.
7	RW	Rst213	**Card Removal Interrupt Enable** 1: Enable. 0: Masked.
6	RW	Rst213	**Card Inserted Interrupt Enable** 1: Enable. 0: Masked.
5	RW	Rst213	**Buffer Read Ready Interrupt Enable** 1: Enable. 0: Masked.
4	RW	Rst213	**Buffer Write Ready Interrupt Enable** 1: Enable. 0: Masked.
3	RW	Rst213	**DMA Interrupt Interrupt Enable** 1: Enable. 0: Masked.
2	RW	Rst213	**Block Gap Event Interrupt Enable** 1: Enable. 0: Masked.
1	RW	Rst213	**Transfer Complete Interrupt Enable** 1: Enable. 0: Masked.

to next page

from previous page

0	RW	Rst213	**Command Complete Interrupt Enable** 1: Enable. 0: Masked.

Offset: 13Ch		**EMMC13C: Auto CMD12 Error Status Register**	**Init = 0**
Bit	**R/W**	**Reset**	**Description**
31	RW	Rst213	**Preset Value Enable**
30	RW	Rst213	**Asynchronous Interrupt Enable**
29:24	RO	-	**Reserved**
23	RW	Rst213	**Sampling Clock Select** Write 1: no operation. Write 0: reset and disable tuning block. Read 1: tuning procedure completes.
22	RW	Rst213	**Execute Tuning**
21:20	RW	Rst213	**Driver Strength Select**
19	RW	Rst213	**1.8V Signaling Enable**
18:16	RW	Rst213	**UHS Mode Select** 000b : SDR12 001b : SDR25 010b : SDR50 011b : SDR104 100b : DDR50 other: reserved
15: 8	RO	-	**Reserved**
7	RO	Rst214	**Command Not Issue by Auto CMD12 Error** 1: Command was not executed by EMMC-Host due to previous Auto CMD12 error.
6: 5	RO	-	**Reserved**
4	RO	Rst214	**Auto CMD12 Index Error** 1: Command Index error occurs in Auto CMD12 response.
3	RO	Rst214	**Auto CMD12 End Bit Error** 1: End bit of Auto CMD12 response is 0.
2	RO	Rst214	**Auto CMD12 CRC Error** 1: CRC error in Auto CMD12 response.
1	RO	Rst214	**Auto CMD12 Timeout Error** 1: No response is returned within 64 SDCLK cycle from the end bit of Auto CMD12.
0	RW	Rst214	**Auto CMD12 Not Executed** 1: Auto CMD12 can not be issued due to some error.

Offset: 140h		**EMMC140: Capability Register #1**	**Init = 0**
Bit	**R/W**	**Reset**	**Description**
31:30	RO	-	**Slot Type** 00b : Removable Card Slot 01b : Embedded Slot for One Device 10b : Shared Bus Slot 11b : Reserved

to next page

from previous page

29	RO	-	**Asynchronous Interrupt Support** 1: Support 0: Not supprot
28	RO	-	**64-bits System Bus Support** Always set to 0.
27	RO	-	**Reserved**
26	RO	-	**Voltage Support 1.8V** 1: 1.8V supported. 0: 1.8V not supported.
25	RO	-	**Voltage Support 3.0V** 1: 3.0V supported. 0: 3.0V not supported.
24	RO	-	**Voltage Support 3.3V** 1: 3.3V supported. 0: 3.3V not supported.
23	RO	-	**Suspend/Resume Support** 1: Suspend/Resume enable. 0: Suspend/Resume disable.
22	RO	-	**SDMA Support** 1: SDMA supported. 0: SDMA not supported.
21	RO	-	**High Speed Support** 1: High Speed supported. 0: High Speed not supported.
20	RO	-	**ADMA1 Support** 1: ADMA1 supported. 0: ADMA1 not supported.
19	RO	-	**ADMA2 Support** 1: ADMA2 supported. 0: ADMA2 not supported.
18	RO	-	**8-bit Embedded Device Support** 1: Support 0: Not Support
17:16	RO	-	**Max Block Length** 00b: 512 Bytes 01b: 1024 Bytes 10b: 2048 Bytes 11b: Reserved
15:14	RO	-	**Reserved**
13: 8	RO	-	**Base Clock Frequency for SD Clock** 00h: Obtain clock information via another method. 01h: 1MHz 02h: 2MHz 03h: 3MHz ... 3Fh: 63MHz
7	RO	-	**Timeout Clock Unit** Unit of EMMC140[5:0] 0: kHz 1: MHz

to next page

from previous page

6	RO	-	**Reserved**
5: 0	RO	-	**Timeout Clock Frequency** 00h: Obtain clock information via another method. 01h: 1 02h: 2 03h: 3 ... 3Fh: 63

Offset: 144h **EMMC144: Capability Register #2** **Init = 0**

Bit	R/W	Reset	Description
31:24	RO	-	**Reserved**
23:16	RO	-	**Clock Multiplier**
15:14	RO	-	**Re-Tuning Mode**
13	RO	-	**Use Tuning for SDR50**
12	RO	-	**Reserved**
11: 8	RO	-	**Timer Count for Re-Tuning** 0h : Re-Tuning Timer disabled 1h : 1 second ... n : 2^(n-1) seconds ... Bh : 1024 seconds Ch : Reserved Dh : Reserved Eh : Reserved Fh : Obtain this info in other way
7	RO	-	**Reserved**
6	RO	-	**1.8V Line Driver Type D Supported** 1: Support 0: Not support
5	RO	-	**1.8V Line Driver Type C Supported** 1: Support 0: Not support
4	RO	-	**1.8V Line Driver Type A Supported** 1: Support 0: Not support
3	RO	-	**Reserved**
2	RO	-	**DDR50 Supported** 1: Support 0: Not support
1	RO	-	**SDR104 Supported** 1: Support 0: Not support
0	RO	-	**SDR50 Supported** 1: Support 0: Not support

Offset: 148h	EMMC148: Maximum Current Capabilities Register		Init = 0
Bit	**R/W**	**Reset**	**Description**
31:24	RO	-	**Reserved**
23:16	RO	-	**Maximum Current for 1.8V** 00h: Obtain current information via another method. 01h: 4mA 02h: 8mA 03h: 12mA ... FFh: 1020mA
15: 8	RO	-	**Maximum Current for 3.0V** 00h: Obtain current information via another method. 01h: 4mA 02h: 8mA 03h: 12mA ... FFh: 1020mA
7: 0	RO	-	**Maximum Current for 3.3V** 00h: Obtain current information via another method. 01h: 4mA 02h: 8mA 03h: 12mA ... FFh: 1020mA

Offset: 150h	EMMC150: Event Trigger Register		Init = 0
Bit	**R/W**	**Reset**	**Description**
31:26	RO	-	**Reserved**
25	W1T	-	**Force ADMA Error Event**
24	W1T	-	**Force Auto CMD12 Error Event**
23	W1T	-	**Force Current Limit Error Event**
22	W1T	-	**Force Data End Bit Error Event**
21	W1T	-	**Force Data CRC Error Event**
20	W1T	-	**Force Data Timeout Error Event**
19	W1T	-	**Force Command Index Error Event**
18	W1T	-	**Force Command End Bit Error Event**
17	W1T	-	**Force Command CRC Error Event**
16	W1T	-	**Force Command Timeout Error Event**
15: 8	RO	-	**Reserved**
7	W1T	-	**Force Card Removal Event**
6: 5	RO	-	**Reserved**
4	W1T	-	**Force Buffer Write Ready Event**
3	W1T	-	**Force DMA Interrupt Event**
2	W1T	-	**Force Block Gap Event Event**
1	W1T	-	**Force Transfer Complete Event**
0	W1T	-	**Force Command Complete Event**

Offset: 154h	EMMC154: ADMA Error Status Register	Init = 0

Bit	R/W	Reset	Description
31: 3	RO	-	**Reserved**
2	RW	Rst213	**ADMA Length Mismatch Error** This bit is set when: 1. total data length specified in ADMA descriptors is different from that specified by the Block Count and Block Length fields (if Block Count Enable is set). 2. total data length cannot be divided by the block length (if Block Count Enable is not set).
1: 0	RW	Rst213	**ADMA Error State** 00b: ADMA Stop. 01b: Fetching descriptor. 10b: not used. 11b: Transfer data.

Offset: 158h	EMMC158: ADMA System Address	Init = 0

Bit	R/W	Reset	Description
31: 0	RW	Rst213	**ADMA System Address** Write: descriptor table base address. Read: 1. no error occurs: next descriptor to be fetched. 2. error occurs: EMMC154[1:0] 00b: next of the error decriptor. 01b: error descriptor 10b: not used. 11b: next of the error decriptor.

40 eMMC Boot Controller (EB)

40.1 Overview

EMMC Boot controller supports moving eMMC into pre-IDLE state which is ready to access boot area.
It also supports DMA to move data from eMMC to a pre-programmed area.
DMA transfer length is in unit of 512 bytes.

Base Address of EMMC Boot Controller = 0x1E6F_5000
Physical address of register = (Base address of EMMC Boot Controller) + Offset

EB00: DMA Address
EB04: DMA Length
EB08: Interrupt Enable and Status
EB0C: Control Register

40.2 Registers : Base Address = 0x1E6F:5000

Offset: 00h	EB00: DMA_Address	Init = 0

Bit	R/W	Reset	Description
31: 2	RW	RstARM	**DMA Address**
1: 0	RO	-	**Reserved**

Offset: 04h	EB04: DMA Length	Init = 0

Bit	R/W	Reset	Description
31:18	RO	-	**Reserved**
17: 9	RW	RstARM	**DMA Length** 1 : 512 bytes 2 : 1024 bytes and so on
8: 0	RO	-	**Reserved**

Offset: 08h	EB08: Interrupt Enable and Status	Init = 0

Bit	R/W	Reset	Description
31:18	RO	-	**Reserved**
17	RW	RstARM	**Enable Interrupt of Error**
16	RW	RstARM	**Enable Interrupt of DMA Complete**
15: 2	RO	-	**Reserved**
1	RW	RstARM	**Interrupt status of Error**
0	RW	RstARM	**Interrupt status of DMA Complete**

Offset: 0Ch			EB0C: Control Register	Init = 0
Bit	**R/W**	**Reset**	**Description**	
31	RW	RstARM	**Engine Enable**	
30	RW	RstARM	**Write protection of EB0C.31**	
29	RO	-	**Reserved**	
28	RW	RstARM	**Switch eMMC multi-function pin to EMMC Boot function**	
27	RO	-	**Reserved**	
26	RW	RstARM	**Send CMD0 with argument 0xF0F0F0F0**	
25	RW	RstARM	**Send CMD0 with argument 0x00000000**	
24	RW	RstARM	**Trigger DMA**	
23:19	RO	-	**Reserved**	
18	RO	RstARM	**CRC Error**	
17	RO	RstARM	**Data does not appear within 1 second after triggering**	
16	RO	RstARM	**Acknowledge does not appear within 50 mini-second after triggering**	
15:12	RO	-	**Reserved**	
11: 8	RO	RstARM	**Data State**	
7: 0	RO	RstARM	**Control State**	

41 GPIO Controller (GPIO)

41.1 Overview

AST2600 Integrates one set of Parallel GPIO Controller with maximum 208 control pins, which are 26 sets, to provide general-purpose input/output functions, one set of Serial GPIO Controller with maximum 80 serial input and 80 serial output , and one set of Serial GPIO slave controller which follows SFF-8485. Please reference Section 2.4.

Parallel GPIO

Each GPIO sets can be programmed to accept command from Coprocessor CPU, LPC or ARM.

- All GPIO pins can be programmed to support the following options:
  - Input or output option (input mode or output mode)
  - Interrupt generation option (enabled or disabled interrupt generation)
  - Interrupt sensitivity option (level-high, level-low, rising-edge, falling-edge or both-edge trigger mode)
  - Interrupt direction option (ARM or LPC)
  - WDT reset tolerance (for non-interrupted related registers only)
  - De-bouncing option (0ms, 1ms, 5ms or 10ms de-bouncing)
  - Input mask

1.8V Parallel GPIO

Each GPIO sets can be programmed to accept command from Coprocessor CPU, LPC or ARM.

- All GPIO pins can be programmed to support the following options:
  - Input or output option (input mode or output mode)
  - Interrupt generation option (enabled or disabled interrupt generation)
  - Interrupt sensitivity option (level-high, level-low, rising-edge, falling-edge or both-edge trigger mode)
  - Interrupt direction option (ARM or LPC)
  - WDT reset tolerance (for non-interrupted related registers only)
  - De-bouncing option (0ms, 1ms, 5ms or 10ms de-bouncing)
  - Input mask

Serial GPIO Master

Each Serial GPIO input pins can be programmed to support the following options:

- Directly connected to APB bus
- Co-work with external serial-chained TTL components (74LV165/74LV595)
- Support 2 master. One is up to 128 SGPIO input ports and 128 output ports concurrently and Second one is up to 80.
- Each of them is only at the cost of 4 control pins
- Shift clock is from APB bus clock divided by a programmable value.
- Programmable shift-load clock length (8/16/24/32/40/48/56/64/72/80 clocks)
- Support interrupt option for each input port
- Support interrupt sensitivity option: Level-High, Level-Low, Edge-High, Edge-Low
- Support reset tolerance option for each output port

Serial GPIO Slave Monitor

Serial GPIO Slave monitors SGPIO bus between Initiator and Target that follows SFF-8485.

- Serial GPIO Slave Monitor is monitoring SGPIO bus between Initiator and Target that follows SFF-8485.
- Support 2 monitors. Each one supports maximum 32 drives.

Parallel GPIO implements 59 sets of 32-bit registers, which are listed below, to program the various supported functions including input/output mode, interrupt sensitivity, WDT tolerance, and de-bouncing options. Each register has its own specific offset value, ranging from 0x00 to 0x128h to derive its physical address location.

Base Address of GPIO = 0x1E78_0000
Register Address of GPIO = (Base Address of GPIO) + Offset

GPIO000: GPIO_A/B/C/D Data Value Register
GPIO004: GPIO_A/B/C/D Direction Register
GPIO008: GPIO_A/B/C/D Interrupt Enable Register
GPIO00C: GPIO_A/B/C/D Interrupt Sensitivity Type 0 Register
GPIO010: GPIO_A/B/C/D Interrupt Sensitivity Type 1 Register
GPIO014: GPIO_A/B/C/D Interrupt Sensitivity Type 2 Register
GPIO018: GPIO_A/B/C/D Interrupt Status Register
GPIO01C: GPIO_A/B/C/D Reset Tolerant Register
GPIO020: GPIO_E/F/G/H Data Value Register
GPIO024: GPIO_E/F/G/H Direction Register
GPIO028: GPIO_E/F/G/H Interrupt Enable Register
GPIO02C: GPIO_E/F/G/H Interrupt Sensitivity Type 0 Register
GPIO030: GPIO_E/F/G/H Interrupt Sensitivity Type 1 Register
GPIO034: GPIO_E/F/G/H Interrupt Sensitivity Type 2 Register
GPIO038: GPIO_E/F/G/H Interrupt Status Register
GPIO03C: GPIO_E/F/G/H Reset Tolerant Register
GPIO040: GPIO_A/B/C/D Debounce Setting #1
GPIO044: GPIO_A/B/C/D Debounce Setting #2
GPIO048: GPIO_E/F/G/H Debounce Setting #1
GPIO04C: GPIO_E/F/G/H Debounce Setting #2
GPIO050: Debounce Time Setting #1
GPIO054: Debounce Time Setting #2
GPIO058: Debounce Time Setting #3
GPIO060: GPIO_A/B/C/D Command Source 0
GPIO064: GPIO_A/B/C/D Command Source 1
GPIO068: GPIO_E/F/G/H Command Source 0
GPIO06C: GPIO_E/F/G/H Command Source 1
GPIO070: GPIO_I/J/K/L Data Value Register
GPIO074: GPIO_I/J/K/L Direction Register
GPIO078: GPIO_M/N/O/P Data Value Register
GPIO07C: GPIO_M/N/O/P Direction Register
GPIO080: GPIO_Q/R/S/T Data Value Register
GPIO084: GPIO_Q/R/S/T Direction Register
GPIO088: GPIO_U/V/W/X Data Value Register
GPIO08C: GPIO_U/V/W/X Direction Register
GPIO090: GPIO_I/J/K/L Command Source 0
GPIO094: GPIO_I/J/K/L Command Source 1
GPIO098: GPIO_I/J/K/L Interrupt Enable Register
GPIO09C: GPIO_I/J/K/L Interrupt Sensitivity Type 0 Register
GPIO0A0: GPIO_I/J/K/L Interrupt Sensitivity Type 1 Register
GPIO0A4: GPIO_I/J/K/L Interrupt Sensitivity Type 2 Register
GPIO0A8: GPIO_I/J/K/L Interrupt Status Register
GPIO0AC: GPIO_I/J/K/L Reset Tolerant Register

GPIO0B0: GPIO_I/J/K/L Debounce Setting #1
GPIO0B4: GPIO_I/J/K/L Debounce Setting #2
GPIO0B8: GPIO_I/J/K/L Input Mask
GPIO0C0: GPIO_A/B/C/D Data Read Register
GPIO0C4: GPIO_E/F/G/H Data Read Register
GPIO0C8: GPIO_I/J/K/L Data Read Register
GPIO0CC: GPIO_M/N/O/P Data Read Register
GPIO0D0: GPIO_Q/R/S/T Data Read Register
GPIO0D4: GPIO_U/V/W/X Data Read Register
GPIO0D8: GPIO_Y/Z Data Read Register
GPIO0E0: GPIO_M/N/O/P Command Source 0
GPIO0E4: GPIO_M/N/O/P Command Source 1
GPIO0E8: GPIO_M/N/O/P Interrupt Enable Register
GPIO0EC: GPIO_M/N/O/P Interrupt Sensitivity Type 0 Register
GPIO0F0: GPIO_M/N/O/P Interrupt Sensitivity Type 1 Register
GPIO0F4: GPIO_M/N/O/P Interrupt Sensitivity Type 2 Register
GPIO0F8: GPIO_M/N/O/P Interrupt Status Register
GPIO0FC: GPIO_M/N/O/P Reset Tolerant Register
GPIO100: GPIO_M/N/O/P Debounce Setting #1
GPIO104: GPIO_M/N/O/P Debounce Setting #2
GPIO108: GPIO_M/N/O/P Input Mask
GPIO110: GPIO_Q/R/S/T Command Source 0
GPIO114: GPIO_Q/R/S/T Command Source 1
GPIO118: GPIO_Q/R/S/T Interrupt Enable Register
GPIO11C: GPIO_Q/R/S/T Interrupt Sensitivity Type 0 Register
GPIO120: GPIO_Q/R/S/T Interrupt Sensitivity Type 1 Register
GPIO124: GPIO_Q/R/S/T Interrupt Sensitivity Type 2 Register
GPIO128: GPIO_Q/R/S/T Interrupt Status Register
GPIO12C: GPIO_Q/R/S/T Reset Tolerant Register
GPIO130: GPIO_Q/R/S/T Debounce Setting #1
GPIO134: GPIO_Q/R/S/T Debounce Setting #2
GPIO138: GPIO_Q/R/S/T Input Mask
GPIO140: GPIO_U/V/W/X Command Source 0
GPIO144: GPIO_U/V/W/X Command Source 1
GPIO148: GPIO_U/V/W/X Interrupt Enable Register
GPIO14C: GPIO_U/V/W/X Interrupt Sensitivity Type 0 Register
GPIO150: GPIO_U/V/W/X Interrupt Sensitivity Type 1 Register
GPIO154: GPIO_U/V/W/X Interrupt Sensitivity Type 2 Register
GPIO158: GPIO_U/V/W/X Interrupt Status Register
GPIO15C: GPIO_U/V/W/X Reset Tolerant Register
GPIO160: GPIO_U/V/W/X Debounce Setting #1
GPIO164: GPIO_U/V/W/X Debounce Setting #2
GPIO168: GPIO_U/V/W/X Input Mask
GPIO170: GPIO_Y/Z Command Source 0
GPIO174: GPIO_Y/Z Command Source 1
GPIO178: GPIO_Y/Z Interrupt Enable Register
GPIO17C: GPIO_Y/Z Interrupt Sensitivity Type 0 Register
GPIO180: GPIO_Y/Z Interrupt Sensitivity Type 1 Register
GPIO184: GPIO_Y/Z Interrupt Sensitivity Type 2 Register
GPIO188: GPIO_Y/Z Interrupt Status Register
GPIO18C: GPIO_Y/Z Reset Tolerant Register
GPIO190: GPIO_Y/Z Debounce Setting #1
GPIO194: GPIO_Y/Z Debounce Setting #2
GPIO198: GPIO_Y/Z Input Mask
GPIO1D0: GPIO_A/B/C/D Input Mask
GPIO1D4: GPIO_E/F/G/H Input Mask
GPIO1E0: GPIO_Y/Z Data Value Register

GPIO1E4: GPIO_Y/Z Direction Register
GPIO22C: GPIO_A/B/C/D New Command Source For Write
GPIO230: GPIO_A/B/C/D New Command Source For Read
GPIO238: GPIO_A/B/C/D Blink Counter Selection #1
GPIO23C: GPIO_A/B/C/D Blink Counter Selection #2
GPIO27C: GPIO_E/F/G/H New Command Source For Write
GPIO280: GPIO_E/F/G/H New Command Source For Read
GPIO288: GPIO_E/F/G/H Blink Counter Selection #1
GPIO28C: GPIO_E/F/G/H Blink Counter Selection #2
GPIO2B0: GPIO Blink Source Clock Division
GPIO2B4: GPIO Blink Control
GPIO2B8: GPIO Blink Counter #1 Configuration
GPIO2BC: GPIO Blink Counter #2 Configuration
GPIO2C0: GPIO Blink Counter #3 Configuration
GPIO2D0: GPIO Command Source Selection
GPIO32C: GPIO_I/J/K/L New Command Source For Write
GPIO330: GPIO_I/J/K/L New Command Source For Read
GPIO338: GPIO_I/J/K/L Blink Counter Selection #1
GPIO33C: GPIO_I/J/K/L Blink Counter Selection #2
GPIO37C: GPIO_M/N/O/P New Command Source For Write
GPIO380: GPIO_M/N/O/P New Command Source For Read
GPIO388: GPIO_M/N/O/P Blink Counter Selection #1
GPIO38C: GPIO_M/N/O/P Blink Counter Selection #2
GPIO3CC: GPIO_Q/R/S/T New Command Source For Write
GPIO3D0: GPIO_Q/R/S/T New Command Source For Read
GPIO3D8: GPIO_Q/R/S/T Blink Counter Selection #1
GPIO3DC: GPIO_Q/R/S/T Blink Counter Selection #2
GPIO41C: GPIO_U/V/W/X New Command Source For Write
GPIO420: GPIO_U/V/W/X New Command Source For Read
GPIO428: GPIO_U/V/W/X Blink Counter Selection #1
GPIO42C: GPIO_U/V/W/X Blink Counter Selection #2
GPIO46C: GPIO_Y/Z New Command Source For Write
GPIO470: GPIO_Y/Z New Command Source For Read
GPIO478: GPIO_Y/Z Blink Counter Selection #1
GPIO47C: GPIO_Y/Z Blink Counter Selection #2
GPIO2AC: GPIO Index Register

Base Address of GPIO = 0x1E78_0000
Register Address of GPIO = (Base Address of GPIO) + Offset
GPIO800: GPIO18_A/B/C/D Data Value Register
GPIO804: GPIO18_A/B/C/D Direction Register
GPIO808: GPIO18_A/B/C/D Interrupt Enable Register
GPIO80C: GPIO18_A/B/C/D Interrupt Sensitivity Type 0 Register
GPIO810: GPIO18_A/B/C/D Interrupt Sensitivity Type 1 Register
GPIO814: GPIO18_A/B/C/D Interrupt Sensitivity Type 2 Register
GPIO818: GPIO18_A/B/C/D Interrupt Status Register
GPIO81C: GPIO18_A/B/C/D Reset Tolerant Register
GPIO820: GPIO18_E Data Value Register
GPIO824: GPIO18_E Direction Register
GPIO828: GPIO18_E Interrupt Enable Register
GPIO82C: GPIO18_E Interrupt Sensitivity Type 0 Register
GPIO830: GPIO18_E Interrupt Sensitivity Type 1 Register
GPIO834: GPIO18_E Interrupt Sensitivity Type 2 Register
GPIO838: GPIO18_E Interrupt Status Register
GPIO83C: GPIO18_E Reset Tolerant Register
GPIO840: GPIO18_A/B/C/D Debounce Setting #1

GPIO844: GPIO18_A/B/C/D Debounce Setting #2
GPIO848: GPIO18_E Debounce Setting #1
GPIO84C: GPIO18_E Debounce Setting #2
GPIO850: Debounce Time Setting #1
GPIO854: Debounce Time Setting #2
GPIO858: Debounce Time Setting #3
GPIO860: GPIO18_A/B/C/D Command Source 0
GPIO864: GPIO18_A/B/C/D Command Source 1
GPIO868: GPIO18_E Command Source 0
GPIO86C: GPIO18_E Command Source 1
GPIO8C0: GPIO18_A/B/C/D Data Read Register
GPIO8C4: GPIO18_E Data Read Register
GPIO9D0: GPIO18_A/B/C/D Input Mask
GPIO9D4: GPIO18_E Input Mask
GPIOAAC: GPIO18 Index Register

Serial GPIO Master 1
Serial GPIO implements 25 sets of 32-bit registers, which are listed below, to program the various supported functions including interrupt sensitivity, WDT tolerance

Base Address of GPIO = 0x1E78_0000
Register Address of GPIO = (Base Address of GPIO) + Offset

GPIO500: Serial GPIO_A/B/C/D 1 Data Value Register
GPIO504: Serial GPIO_A/B/C/D 1 Interrupt Enable Register
GPIO508: Serial GPIO_A/B/C/D 1 Interrupt Sensitivity Type 0 Register
GPIO50C: Serial GPIO_A/B/C/D 1 Interrupt Sensitivity Type 1 Register
GPIO510: Serial GPIO_A/B/C/D 1 Interrupt Sensitivity Type 2 Register
GPIO514: Serial GPIO_A/B/C/D 1 Interrupt Status Register
GPIO518: Serial GPIO_A/B/C/D 1 Reset Tolerant Register
GPIO51C: Serial GPIO_E/F/G/H 1 Data Value Register
GPIO520: Serial GPIO_E/F/G/H 1 Interrupt Enable Register
GPIO524: Serial GPIO_E/F/G/H 1 Interrupt Sensitivity Type 0 Register
GPIO528: Serial GPIO_E/F/G/H 1 Interrupt Sensitivity Type 1 Register
GPIO52C: Serial GPIO_E/F/G/H 1 Interrupt Sensitivity Type 2 Register
GPIO530: Serial GPIO_E/F/G/H 1 Interrupt Status Register
GPIO534: Serial GPIO_E/F/G/H 1 Reset Tolerant Register
GPIO538: Serial GPIO_I/J/K/L 1 Data Value Register
GPIO53C: Serial GPIO_I/J/K/L 1 Interrupt Enable Register
GPIO540: Serial GPIO_I/J/K/L 1 Interrupt Sensitivity Type 0 Register
GPIO544: Serial GPIO_I/J/K/L 1 Interrupt Sensitivity Type 1 Register
GPIO548: Serial GPIO_I/J/K/L 1 Interrupt Sensitivity Type 2 Register
GPIO54C: Serial GPIO_I/J/K/L 1 Interrupt Status Register
GPIO550: Serial GPIO_I/J/K/L 1 Reset Tolerant Register
GPIO554: Serial GPIO 1 Contral Register
GPIO558: Serail GPIO_A/B/C/D 1 Input Mask
GPIO558: Serail GPIO_E/F/G/H 1 Input Mask
GPIO558: Serail GPIO_I/J/K/L 1 Input Mask
GPIO558: Serail GPIO_M/N/O/P 1 Input Mask
GPIO570: Serial GPIO_A/B/C/D 1 Data Read Register
GPIO574: Serial GPIO_E/F/G/H 1 Data Read Register
GPIO578: Serial GPIO_I/J/K/L 1 Data Read Register
GPIO57C: Serial GPIO_M/N/O/P 1 Data Read Register
GPIO590: Serial GPIO_M/N/O/P 1 Data Value Register
GPIO594: Serial GPIO_M/N/O/P 1 Interrupt Enable Register
GPIO598: Serial GPIO_M/N/O/P 1 Interrupt Sensitivity Type 0 Register

GPIO59C: Serial GPIO_M/N/O/P 1 Interrupt Sensitivity Type 1 Register
GPIO5A0: Serial GPIO_M/N/O/P 1 Interrupt Sensitivity Type 2 Register
GPIO5A4: Serial GPIO_M/N/O/P 1 Interrupt Status Register
GPIO5A8: Serial GPIO_M/N/O/P 1 Reset Tolerant Register

Serial GPIO Master 2

GPIO600: Serial GPIO_A/B/C/D 2 Data Value Register
GPIO604: Serial GPIO_A/B/C/D 2 Interrupt Enable Register
GPIO608: Serial GPIO_A/B/C/D 2 Interrupt Sensitivity Type 0 Register
GPIO60C: Serial GPIO_A/B/C/D 2 Interrupt Sensitivity Type 1 Register
GPIO610: Serial GPIO_A/B/C/D 2 Interrupt Sensitivity Type 2 Register
GPIO614: Serial GPIO_A/B/C/D 2 Interrupt Status Register
GPIO618: Serial GPIO_A/B/C/D 2 Reset Tolerant Register
GPIO61C: Serial GPIO_E/F/G/H 2 Data Value Register
GPIO620: Serial GPIO_E/F/G/H 2 Interrupt Enable Register
GPIO624: Serial GPIO_E/F/G/H 2 Interrupt Sensitivity Type 0 Register
GPIO628: Serial GPIO_E/F/G/H 2 Interrupt Sensitivity Type 1 Register
GPIO62C: Serial GPIO_E/F/G/H 2 Interrupt Sensitivity Type 2 Register
GPIO630: Serial GPIO_E/F/G/H 2 Interrupt Status Register
GPIO634: Serial GPIO_E/F/G/H 2 Reset Tolerant Register
GPIO638: Serial GPIO_I/J 2 Data Value Register
GPIO63C: Serial GPIO_I/J 2 Interrupt Enable Register
GPIO640: Serial GPIO_I/J 2 Interrupt Sensitivity Type 0 Register
GPIO644: Serial GPIO_I/J 2 Interrupt Sensitivity Type 1 Register
GPIO648: Serial GPIO_I/J 2 Interrupt Sensitivity Type 2 Register
GPIO64C: Serial GPIO_I/J 2 Interrupt Status Register
GPIO650: Serial GPIO_I/J 2 Reset Tolerant Register
GPIO654: Serial GPIO 2 Contral Register
GPIO658: Serail GPIO_A/B/C/D 2 Input Mask
GPIO658: Serail GPIO_E/F/G/H 2 Input Mask
GPIO658: Serail GPIO_I/J 2 Input Mask
GPIO670: Serial GPIO_A/B/C/D 2 Data Read Register
GPIO674: Serial GPIO_E/F/G/H 2 Data Read Register
GPIO678: Serial GPIO_I/J 2 Data Read Register

Serial GPIO Slave 1

Serial GPIO slave monitor implements 9 sets of 32-bit registers, which are listed below, to monitering behavior betwenn Initaitor and Target, which follows SPF-8485. This monitor reads SData_I and SLoad from Initiator and SData_T from Target. Each register has its own specific offset value, ranging from 0x00 to 0x20h, to derive its physical address location.

Base Address of GPIO = 0x1E78_0000
Register Address of GPIO = (Base Address of GPIO) + Offset

GPIO700: SGPIO Slave 1 Data from Initiator #1
GPIO704: SGPIO Slave 1 Data from Initiator #2
GPIO708: SGPIO Slave 1 Data from Initiator #3
GPIO70C: SGPIO Slave 1 Data from Target #1
GPIO710: SGPIO Slave 1 Data from Target #2
GPIO714: SGPIO Slave 1 Data from Target #3
GPIO718: SGPIO Slave 1 Status
GPIO71C: SGPIO Slave 1 Interrupt Enable and Status

Serial GPIO Slave 2

GPIO740: SGPIO Slave 2 Data from Initiator #1
GPIO744: SGPIO Slave 2 Data from Initiator #2
GPIO748: SGPIO Slave 2 Data from Initiator #3
GPIO74C: SGPIO Slave 2 Data from Target #1
GPIO750: SGPIO Slave 2 Data from Target #2
GPIO754: SGPIO Slave 2 Data from Target #3
GPIO758: SGPIO Slave 2 Status
GPIO75C: SGPIO Slave 2 Interrupt Enable and Status

41.2 Features

41.2.1 Parallel GPIO

- Directly connected to APB bus
- Programmable reset tolerance option for all GIPO pin.
- Support interrupt triggered by all GPIO pins.
- All input pin is with de-bouncing logic option.

41.2.2 Serial GPIO

- Directly connected to APB bus
- Programmable reset tolerance option for 128 and 80 Serial GIPO pin.

41.2.3 SGPIO Slave Monitor

- Monitor Bus behavior between Initiator and Target, which follows SPF-8485.
- Maximum 32 drives.

41.3 Registers : Base Address = 0x1E78:0000

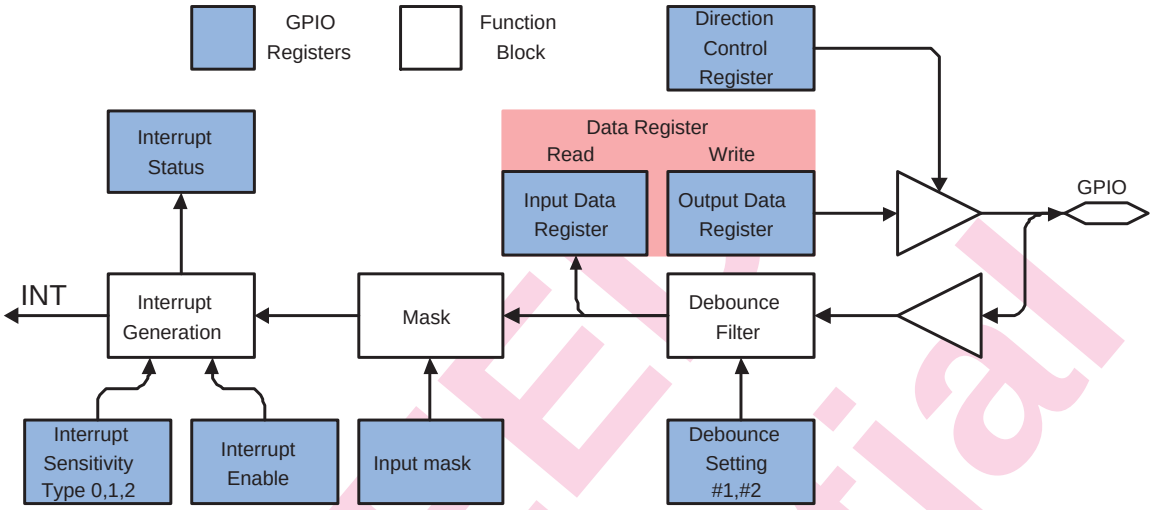


Figure 53: Parallel GPIO Function Block Diagram

41.3.1 Parallel GPIO

Offset: 000h	GPIO000: GPIO_A/B/C/D Data Value Register	Init = 0
Bit	**R/W**	**Description**
31:24	RW	**Port GPIOD[7:0] data register**
23:16	RW	**Port GPIOC[7:0] data register**
15:8	RW	**Port GPIOB[7:0] data register**
7 :0	RW	**Port GPIOA[7:0] data register**

Offset: 004h	GPIO004: GPIO_A/B/C/D Direction Register	Init = 0
Bit	**R/W**	**Description**
31:24	RW	**Port GPIOD[7:0] direction control** 0: Select input mode 1: Select output mode
23:16	RW	**Port GPIOC[7:0] direction control** 0: Select input mode 1: Select output mode
15:8	RW	**Port GPIOB[7:0] direction control** 0: Select input mode 1: Select output mode
7 :0	RW	**Port GPIOA[7:0] direction control** 0: Select input mode 1: Select output mode

Offset: 008h	GPIO008: GPIO_A/B/C/D Interrupt Enable Register	Init = 0
Bit	**R/W**	**Description**
31:24	RW	**Port GPIOD[7:0] interrupt enable** 0: Disable interrupt 1: Enable interrupt
23:16	RW	**Port GPIOC[7:0] interrupt enable** 0: Disable interrupt 1: Enable interrupt
15:8	RW	**Port GPIOB[7:0] interrupt enable** 0: Disable interrupt 1: Enable interrupt
7 :0	RW	**Port GPIOA[7:0] interrupt enable** 0: Disable interrupt 1: Enable interrupt

Offset: 00Ch	GPIO00C: GPIO_A/B/C/D Interrupt Sensitivity Type 0 Register	Init = 0
Bit	**R/W**	**Description**
31:24	RW	**Port GPIOD[7:0] interrupt sensitivity type 0 selection** 0: Select falling-edge or level-low trigger mode 1: Select rising-edge or level-high trigger mode
23:16	RW	**Port GPIOC[7:0] interrupt sensitivity type 0 selection** 0: Select falling-edge or level-low trigger mode 1: Select rising-edge or level-high trigger mode
15:8	RW	**Port GPIOB[7:0] interrupt sensitivity type 0 selection** 0: Select falling-edge or level-low trigger mode 1: Select rising-edge or level-high trigger mode
7 :0	RW	**Port GPIOA[7:0] interrupt sensitivity type 0 selection** 0: Select falling-edge or level-low trigger mode 1: Select rising-edge or level-high trigger mode

Offset: 010h	GPIO010: GPIO_A/B/C/D Interrupt Sensitivity Type 1 Register	Init = 0
Bit	**R/W**	**Description**
31:24	RW	**Port GPIOD[7:0] interrupt sensitivity type 1 selection** 0: Select edge trigger mode 1: Select level trigger mode
23:16	RW	**Port GPIOC[7:0] interrupt sensitivity type 1 selection** 0: Select edge trigger mode 1: Select level trigger mode
15:8	RW	**Port GPIOB[7:0] interrupt sensitivity type 1 selection** 0: Select edge trigger mode 1: Select level trigger mode
7 :0	RW	**Port GPIOA[7:0] interrupt sensitivity type 1 selection** 0: Select edge trigger mode 1: Select level trigger mode

Offset: 014h	GPIO014: GPIO_A/B/C/D Interrupt Sensitivity Type 2 Register	Init = 0

Bit	R/W	Description
31:24	RW	**Port GPIOD[7:0] interrupt sensitivity type 2 selection** 0: Select edge or level trigger mode 1: Select dual-edge trigger mode
23:16	RW	**Port GPIOC[7:0] interrupt sensitivity type 2 selection** 0: Select edge or level trigger mode 1: Select dual-edge trigger mode
15:8	RW	**Port GPIOB[7:0] interrupt sensitivity type 2 selection** 0: Select edge or level trigger mode 1: Select dual-edge trigger mode
7 :0	RW	**Port GPIOA[7:0] interrupt sensitivity type 2 selection** 0: Select edge or level trigger mode 1: Select dual-edge trigger mode

Offset: 018h	GPIO018: GPIO_A/B/C/D Interrupt Status Register	Init = 0

Bit	R/W	Description
31:24	RW	**Port GPIOD[7:0] interrupt status register** Read 0: No interrupt pending Read 1: interrupt pending Write 0: No operation Write 1: Clear interrupt status flag
23:16	RW	**Port GPIOC[7:0] interrupt status register** Read 0: No interrupt pending Read 1: interrupt pending Write 0: No operation Write 1: Clear interrupt status flag
15:8	RW	**Port GPIOB[7:0] interrupt status register** Read 0: No interrupt pending Read 1: interrupt pending Write 0: No operation Write 1: Clear interrupt status flag
7 :0	RW	**Port GPIOA[7:0] interrupt status register** Read 0: No interrupt pending Read 1: interrupt pending Write 0: No operation Write 1: Clear interrupt status flag

Offset: 01Ch	GPIO01C: GPIO_A/B/C/D Reset Tolerant Register	Init = 0

Bit	R/W	Description
31:24	RW	**Port GPIOD[7:0] WDT reset tolerance enable** 0: GPIO00 and GPIO04 registers will be reset by WDT reset 1: GPIO00 and GPIO04 registers will not be reset by WDT reset Each GPIO pin can be individually programmed to be WDT reset tolerant or not.
23:16	RW	**Port GPIOC[7:0] WDT reset tolerance enable** 0: GPIO00 and GPIO04 registers will be reset by WDT reset 1: GPIO00 and GPIO04 registers will not be reset by WDT reset Each GPIO pin can be individually programmed to be WDT reset tolerant or not.

to next page

from previous page

15:8	RW	**Port GPIOB[7:0] WDT reset tolerance enable** 0: GPIO00 and GPIO04 registers will be reset by WDT reset 1: GPIO00 and GPIO04 registers will not be reset by WDT reset Each GPIO pin can be individually programmed to be WDT reset tolerant or not.
7 :0	RW	**Port GPIOA[7:0] WDT reset tolerance enable** 0: GPIO00 and GPIO04 registers will be reset by WDT reset 1: GPIO00 and GPIO04 registers will not be reset by WDT reset Each GPIO pin can be individually programmed to be WDT reset tolerant or not.

Offset: 020h **GPIO020: GPIO_E/F/G/H Data Value Register** **Init = 0**

Bit	R/W	Description
31:24	RW	**Port GPIOH[7:0] data register**
23:16	RW	**Port GPIOG[7:0] data register**
15:8	RW	**Port GPIOF[7:0] data register**
7 :0	RW	**Port GPIOE[7:0] data register**

Offset: 024h **GPIO024: GPIO_E/F/G/H Direction Register** **Init = 0**

Bit	R/W	Description
31:24	RW	**Port GPIOH[7:0] direction control** 0: Select input mode 1: Select output mode
23:16	RW	**Port GPIOG[7:0] direction control** 0: Select input mode 1: Select output mode
15:8	RW	**Port GPIOF[7:0] direction control** 0: Select input mode 1: Select output mode
7 :0	RW	**Port GPIOE[7:0] direction control** 0: Select input mode 1: Select output mode

Offset: 028h **GPIO028: GPIO_E/F/G/H Interrupt Enable Register** **Init = 0**

Bit	R/W	Description
31:24	RW	**Port GPIOH[7:0] interrupt enable** 0: Disable interrupt 1: Enable interrupt
23:16	RW	**Port GPIOG[7:0] interrupt enable** 0: Disable interrupt 1: Enable interrupt
15:8	RW	**Port GPIOF[7:0] interrupt enable** 0: Disable interrupt 1: Enable interrupt
7 :0	RW	**Port GPIOE[7:0] interrupt enable** 0: Disable interrupt 1: Enable interrupt

Offset: 02Ch	GPIO02C: GPIO_E/F/G/H Interrupt Sensitivity Type 0 Register	Init = 0
Bit	**R/W**	**Description**
31:24	RW	**Port GPIOH[7:0] interrupt sensitivity type 0 selection** 0: Select falling-edge or level-low trigger mode 1: Select rising-edge or level-high trigger mode
23:16	RW	**Port GPIOG[7:0] interrupt sensitivity type 0 selection** 0: Select falling-edge or level-low trigger mode 1: Select rising-edge or level-high trigger mode
15:8	RW	**Port GPIOF[7:0] interrupt sensitivity type 0 selection** 0: Select falling-edge or level-low trigger mode 1: Select rising-edge or level-high trigger mode
7 :0	RW	**Port GPIOE[7:0] interrupt sensitivity type 0 selection** 0: Select falling-edge or level-low trigger mode 1: Select rising-edge or level-high trigger mode

Offset: 030h	GPIO030: GPIO_E/F/G/H Interrupt Sensitivity Type 1 Register	Init = 0
Bit	**R/W**	**Description**
31:24	RW	**Port GPIOH[7:0] interrupt sensitivity type 1 selection** 0: Select edge trigger mode 1: Select level trigger mode
23:16	RW	**Port GPIOG[7:0] interrupt sensitivity type 1 selection** 0: Select edge trigger mode 1: Select level trigger mode
15:8	RW	**Port GPIOF[7:0] interrupt sensitivity type 1 selection** 0: Select edge trigger mode 1: Select level trigger mode
7 :0	RW	**Port GPIOE[7:0] interrupt sensitivity type 1 selection** 0: Select edge trigger mode 1: Select level trigger mode

Offset: 034h	GPIO034: GPIO_E/F/G/H Interrupt Sensitivity Type 2 Register	Init = 0
Bit	**R/W**	**Description**
31:24	RW	**Port GPIOH[7:0] interrupt sensitivity type 2 selection** 0: Select edge or level trigger mode 1: Select dual-edge trigger mode
23:16	RW	**Port GPIOG[7:0] interrupt sensitivity type 2 selection** 0: Select edge or level trigger mode 1: Select dual-edge trigger mode
15:8	RW	**Port GPIOF[7:0] interrupt sensitivity type 2 selection** 0: Select edge or level trigger mode 1: Select dual-edge trigger mode
7 :0	RW	**Port GPIOE[7:0] interrupt sensitivity type 2 selection** 0: Select edge or level trigger mode 1: Select dual-edge trigger mode

Offset: 038h	GPIO038: GPIO_E/F/G/H Interrupt Status Register	Init = 0

Bit	R/W	Description
31:24	RW	**Port GPIOH[7:0] interrupt status register** Read 0: No interrupt pending Read 1: interrupt pending Write 0: No operation Write 1: Clear interrupt status flag
23:16	RW	**Port GPIOG[7:0] interrupt status register** Read 0: No interrupt pending Read 1: interrupt pending Write 0: No operation Write 1: Clear interrupt status flag
15:8	RW	**Port GPIOF[7:0] interrupt status register** Read 0: No interrupt pending Read 1: interrupt pending Write 0: No operation Write 1: Clear interrupt status flag
7 :0	RW	**Port GPIOE[7:0] interrupt status register** Read 0: No interrupt pending Read 1: interrupt pending Write 0: No operation Write 1: Clear interrupt status flag

Offset: 03Ch	GPIO03C: GPIO_E/F/G/H Reset Tolerant Register	Init = 0

Bit	R/W	Description
31:24	RW	**Port GPIOH[7:0] WDT reset tolerance enable** 0: GPIO20 and GPIO24 registers will be reset by WDT reset 1: GPIO20 and GPIO24 registers will not be reset by WDT reset Each GPIO pin can be individually programmed to be WDT reset tolerant or not.
23:16	RW	**Port GPIOG[7:0] WDT reset tolerance enable** 0: GPIO20 and GPIO24 registers will be reset by WDT reset 1: GPIO20 and GPIO24 registers will not be reset by WDT reset Each GPIO pin can be individually programmed to be WDT reset tolerant or not.
15:8	RW	**Port GPIOF[7:0] WDT reset tolerance enable** 0: GPIO20 and GPIO24 registers will be reset by WDT reset 1: GPIO20 and GPIO24 registers will not be reset by WDT reset Each GPIO pin can be individually programmed to be WDT reset tolerant or not.
7 :0	RW	**Port GPIOE[7:0] WDT reset tolerance enable** 0: GPIO20 and GPIO24 registers will be reset by WDT reset 1: GPIO20 and GPIO24 registers will not be reset by WDT reset Each GPIO pin can be individually programmed to be WDT reset tolerant or not.

Offset: 040h	GPIO040: GPIO_A/B/C/D Debounce Setting Register #1	Init = 0

Bit	R/W	Description
31:24	RW	**Port GPIOD[7:0] debounce setting register #1**
23:16	RW	**Port GPIOC[7:0] debounce setting register #1**
15:8	RW	**Port GPIOB[7:0] debounce setting register #1**
7 :0	RW	**Port GPIOA[7:0] debounce setting register #1**

Offset: 044h	GPIO044: GPIO_A/B/C/D Debounce Setting Register #2	Init = 0
Bit	**R/W**	**Description**
31:24	RW	**Port GPIOD[7:0] debounce setting register #2**
23:16	RW	**Port GPIOC[7:0] debounce setting register #2**
15:8	RW	**Port GPIOB[7:0] debounce setting register #2**
7 :0	RW	**Port GPIOA[7:0] debounce setting register #2**

Offset: 048h	GPIO048: GPIO_E/F/G/H Debounce Setting Register #1	Init = 0
Bit	**R/W**	**Description**
31:24	RW	**Port GPIOH[7:0] debounce setting register #1**
23:16	RW	**Port GPIOG[7:0] debounce setting register #1**
15:8	RW	**Port GPIOF[7:0] debounce setting register #1**
7 :0	RW	**Port GPIOE[7:0] debounce setting register #1**

Offset: 04Ch	GPIO04C: GPIO_E/F/G/H Debounce Setting Register #2	Init = 0
Bit	**R/W**	**Description**
31:24	RW	**Port GPIOH[7:0] debounce setting register #2**
23:16	RW	**Port GPIOG[7:0] debounce setting register #2**
15:8	RW	**Port GPIOF[7:0] debounce setting register #2**
7 :0	RW	**Port GPIOE[7:0] debounce setting register #2**

The definition of interrupt trigger mode registers:
GPIO00C ~ GPIO014, GPIO02C ~ GPIO034 are as follows :

Type 2	Type 1	Type 0	Interrupt Trigger Mode
(14/34h)	(10/30h)	(0C/2Ch)	
0	0	0	falling-edge trigger mode
0	0	1	rising-edge trigger mode
0	1	0	level-low trigger mode
0	1	1	level-high trigger mode
1	x	x	dual-edge trigger mode

The definition of debounce setting registers GPIO040 ~ GPIO04C are as follows :

Debounce Setting #2	Debounce Setting #1	Function
0	0	No Debounce
1	0	Select GPIO050 as debounce timer
0	1	Select GPIO054 as debounce timer
1	1	Select GPIO058 as debounce timer

Offset: 050h	GPIO050: Debounce Timer Setting Register #1	Init = 0
Offset: 054h	GPIO054: Debounce Timer Setting Register #2	Init = 0
Offset: 058h	GPIO058: Debounce Timer Setting Register #3	Init = 0

Bit	Attr.	Description
31:24		**Reserved**
23:0	RW	**Debounce Timer Value** This register defines the timer period for GPIO input sampling. The sampling timer period is : Debounce time = PCLK cycle time * Debounce timer value Latency teim = Debounce time * 2

Offset: 060h	GPIO060: GPIO_A/B/C/D Command Source 0	Init = 0

Bit	R/W	Description
31:25		**Reserved**
24	RW	**Port GPIOD[7:0] Command Source 0**
23:17		**Reserved**
16	RW	**Port GPIOC[7:0] Command Source 0**
15: 9		**Reserved**
8	RW	**Port GPIOB[7:0] Command Source 0**
7: 1		**Reserved**
0	RW	**Port GPIOA[7:0] Command Source 0**

Note :

Command Source #1	Command Source #0	Source
0	0	ARM (default)
0	1	LPC
1	0	Coprocessor CPU
1	1	Reserved

Offset: 064h	GPIO064: GPIO_A/B/C/D Command Source 1	Init = 0

Bit	R/W	Description
31:25		**Reserved**
24	RW	**Port GPIOD[7:0] Command Source 1**
23:17		**Reserved**
16	RW	**Port GPIOC[7:0] Command Source 1**
15: 9		**Reserved**
8	RW	**Port GPIOB[7:0] Command Source 1**
7: 1		**Reserved**
0	RW	**Port GPIOA[7:0] Command Source 1**

Note :

Command Source #1	Command Source #0	Source
0	0	ARM (default)
0	1	LPC
1	0	Coprocessor CPU
1	1	Reserved

Offset: 068h	GPIO068: GPIO_E/F/G/H Command Source 0	Init = 0

Bit	R/W	Description
31:25		**Reserved**
24	RW	**Port GPIOH[7:0] Command Source 0**
23:17		**Reserved**
16	RW	**Port GPIOG[7:0] Command Source 0**
15: 9		**Reserved**
8	RW	**Port GPIOF[7:0] Command Source 0**
7: 1		**Reserved**
0	RW	**Port GPIOE[7:0] Command Source 0**

Note :

Command Source #1	Command Source #0	Source
0	0	ARM (default)
0	1	LPC
1	0	Coprocessor CPU
1	1	Reserved

Offset: 06Ch	GPIO06C: GPIO_E/F/G/H Command Source 1	Init = 0

Bit	R/W	Description
31:25		**Reserved**
24	RW	**Port GPIOH[7:0] Command Source 1**
23:17		**Reserved**
16	RW	**Port GPIOG[7:0] Command Source 1**
15: 9		**Reserved**
8	RW	**Port GPIOF[7:0] Command Source 1**
7: 1		**Reserved**
0	RW	**Port GPIOE[7:0] Command Source 1**

Note :

Command Source #1	Command Source #0	Source
0	0	ARM (default)
0	1	LPC
1	0	Coprocessor CPU
1	1	Reserved

Offset: 070h	GPIO070: GPIO_I/J/K/L Data Value Register	Init = 0

Bit	R/W	Description
31:24	RW	**Port GPIOL[7:0] data register**
23:16	RW	**Port GPIOK[7:0] data register**
15:8	RW	**Port GPIOJ[7:0] data register**
7 :0	RW	**Port GPIOI[7:0] data register**

Offset: 074h	GPIO074: GPIO_I/J/K/L Direction Register	Init = 0

Bit	R/W	Description
31:24	RW	**Port GPIOL[7:0] direction control** 0: Select input mode 1: Select output mode
23:16	RW	**Port GPIOK[7:0] direction control** 0: Select input mode 1: Select output mode
15:8	RW	**Port GPIOJ[7:0] direction control** 0: Select input mode 1: Select output mode
7 :0	RW	**Port GPIOI[7:0] direction control** 0: Select input mode 1: Select output mode

Offset: 078h	GPIO078: GPIO_M/N/O/P Data Value Register	Init = 0

Bit	R/W	Description
31:24	RW	**Port GPIOP[7:0] data register**
23:16	RW	**Port GPIOO[7:0] data register**
15:8	RW	**Port GPION[7:0] data register**
7 :0	RW	**Port GPIOM[7:0] data register**

Offset: 07Ch	GPIO07C: GPIO_M/N/O/P Direction Register	Init = 0

Bit	R/W	Description
31:24	RW	**Port GPIOP[7:0] direction control** 0: Select input mode 1: Select output mode
23:16	RW	**Port GPIOO[7:0] direction control** 0: Select input mode 1: Select output mode
15:8	RW	**Port GPION[7:0] direction control** 0: Select input mode 1: Select output mode
7 :0	RW	**Port GPIOM[7:0] direction control** 0: Select input mode 1: Select output mode

Offset: 080h	GPIO080: GPIO_Q/R/S/T Data Value Register	Init = 0

Bit	R/W	Description
31:24	R	**Port GPIOT[7:0] data register**
23:16	RW	**Port GPIOS[7:0] data register**
15:8	RW	**Port GPIOR[7:0] data register**
7 :0	RW	**Port GPIOQ[7:0] data register**

Offset: 084h	GPIO084: GPIO_Q/R/S/T Direction Register	Init = 0
Bit	**R/W**	**Description**
31:24	R	**Reserved** GPIOT is input only
23:16	RW	**Port GPIOS[7:0] direction control** 0: Select input mode 1: Select output mode
15:8	RW	**Port GPIOR[7:0] direction control** 0: Select input mode 1: Select output mode
7 :0	RW	**Port GPIOQ[7:0] direction control** 0: Select input mode 1: Select output mode

Offset: 088h	GPIO088: GPIO_U/V/W/X Data Value Register	Init = 0
Bit	**R/W**	**Description**
31:24	RW	**Port GPIOX[7:0] data register**
23:16	RW	**Port GPIOW[7:0] data register**
15:8	RW	**Port GPIOV[7:0] data register**
7 :0	R	**Port GPIU[7:0] data register**

Offset: 08Ch	GPIO08C: GPIO_U/V/W/X Direction Register	Init = 0
Bit	**R/W**	**Description**
31:24	RW	**Port GPIOX[7:0] direction control** 0: Select input mode 1: Select output mode
23:16	RW	**Port GPIOW[7:0] direction control** 0: Select input mode 1: Select output mode
15:8	RW	**Port GPIOV[7:0] direction control** 0: Select input mode 1: Select output mode
7 :0	R	**Reserved** GPIU is input only

Offset: 090h	GPIO090: GPIO_I/J/K/L Command Source 0	Init = 0
Bit	**R/W**	**Description**
31:25		**Reserved**
24	RW	**Port GPIOL[7:0] Command Source 0**
23:17		**Reserved**
16	RW	**Port GPIOK[7:0] Command Source 0**
15: 9		**Reserved**
8	RW	**Port GPIOJ[7:0] Command Source 0**
7: 1		**Reserved**
0	RW	**Port GPIOI[7:0] Command Source 0**

to next page

from previous page

Note :

Command Source #1	Command Source #0	Source
0	0	ARM (default)
0	1	LPC
1	0	Coprocessor CPU
1	1	Reserved

Offset: 094h **GPIO094: GPIO_I/J/K/L Command Source 1** **Init = 0**

Bit	R/W	Description
31:25		**Reserved**
24	RW	**Port GPIOL[7:0] Command Source 1**
23:17		**Reserved**
16	RW	**Port GPIOK[7:0] Command Source 1**
15: 9		**Reserved**
8	RW	**Port GPIOJ[7:0] Command Source 1**
7: 1		**Reserved**
0	RW	**Port GPIOI[7:0] Command Source 1**

Note :

Command Source #1	Command Source #0	Source
0	0	ARM (default)
0	1	LPC
1	0	Coprocessor CPU
1	1	Reserved

Offset: 098h **GPIO098: GPIO_I/J/K/L Interrupt Enable Register** **Init = 0**

Bit	R/W	Description
31:24	RW	**Port GPIOL[7:0] interrupt enable** 0: Disable interrupt 1: Enable interrupt
23:16	RW	**Port GPIOK[7:0] interrupt enable** 0: Disable interrupt 1: Enable interrupt
15:8	RW	**Port GPIOJ[7:0] interrupt enable** 0: Disable interrupt 1: Enable interrupt
7 :0	RW	**Port GPIOI[7:0] interrupt enable** 0: Disable interrupt 1: Enable interrupt

Offset: 09Ch	GPIO09C: GPIO_I/J/K/L Interrupt Sensitivity Type 0 Register	Init = 0
Bit	**R/W**	**Description**
31:24	RW	**Port GPIOL[7:0] interrupt sensitivity type 0 selection** 0: Select falling-edge or level-low trigger mode 1: Select rising-edge or level-high trigger mode
23:16	RW	**Port GPIOK[7:0] interrupt sensitivity type 0 selection** 0: Select falling-edge or level-low trigger mode 1: Select rising-edge or level-high trigger mode
15:8	RW	**Port GPIOJ[7:0] interrupt sensitivity type 0 selection** 0: Select falling-edge or level-low trigger mode 1: Select rising-edge or level-high trigger mode
7 :0	RW	**Port GPIOI[7:0] interrupt sensitivity type 0 selection** 0: Select falling-edge or level-low trigger mode 1: Select rising-edge or level-high trigger mode

Offset: 0A0h	GPIO0A0: GPIO_I/J/K/L Interrupt Sensitivity Type 1 Register	Init = 0
Bit	**R/W**	**Description**
31:24	RW	**Port GPIOL[7:0] interrupt sensitivity type 1 selection** 0: Select edge trigger mode 1: Select level trigger mode
23:16	RW	**Port GPIOK[7:0] interrupt sensitivity type 1 selection** 0: Select edge trigger mode 1: Select level trigger mode
15:8	RW	**Port GPIOJ[7:0] interrupt sensitivity type 1 selection** 0: Select edge trigger mode 1: Select level trigger mode
7 :0	RW	**Port GPIOI[7:0] interrupt sensitivity type 1 selection** 0: Select edge trigger mode 1: Select level trigger mode

Offset: 0A4h	GPIO0A4: GPIO_I/J/K/L Interrupt Sensitivity Type 2 Register	Init = 0
Bit	**R/W**	**Description**
31:24	RW	**Port GPIOL[7:0] interrupt sensitivity type 2 selection** 0: Select edge or level trigger mode 1: Select dual-edge trigger mode
23:16	RW	**Port GPIOK[7:0] interrupt sensitivity type 2 selection** 0: Select edge or level trigger mode 1: Select dual-edge trigger mode
15:8	RW	**Port GPIOJ[7:0] interrupt sensitivity type 2 selection** 0: Select edge or level trigger mode 1: Select dual-edge trigger mode
7 :0	RW	**Port GPIOI[7:0] interrupt sensitivity type 2 selection** 0: Select edge or level trigger mode 1: Select dual-edge trigger mode

Offset: 0A8h	GPIO0A8: GPIO_I/J/K/L Interrupt Status Register	Init = 0

Bit	R/W	Description
31:24	RW	**Port GPIOL[7:0] interrupt status register** Read 0: No interrupt pending Read 1: interrupt pending Write 0: No operation Write 1: Clear interrupt status flag
23:16	RW	**Port GPIOK[7:0] interrupt status register** Read 0: No interrupt pending Read 1: interrupt pending Write 0: No operation Write 1: Clear interrupt status flag
15:8	RW	**Port GPIOJ[7:0] interrupt status register** Read 0: No interrupt pending Read 1: interrupt pending Write 0: No operation Write 1: Clear interrupt status flag
7 :0	RW	**Port GPIOI[7:0] interrupt status register** Read 0: No interrupt pending Read 1: interrupt pending Write 0: No operation Write 1: Clear interrupt status flag

Offset: 0ACh	GPIO0AC: GPIO_I/J/K/L Reset Tolerant Register	Init = 0

Bit	R/W	Description
31:24	RW	**Port GPIOL[7:0] WDT reset tolerance enable** 0: GPIO70 and GPIO74 registers will be reset by WDT reset 1: GPIO70 and GPIO74 registers will not be reset by WDT reset Each GPIO pin can be individually programmed to be WDT reset tolerant or not.
23:16	RW	**Port GPIOK[7:0] WDT reset tolerance enable** 0: GPIO70 and GPIO74 registers will be reset by WDT reset 1: GPIO70 and GPIO74 registers will not be reset by WDT reset Each GPIO pin can be individually programmed to be WDT reset tolerant or not.
15:8	RW	**Port GPIOJ[7:0] WDT reset tolerance enable** 0: GPIO70 and GPIO74 registers will be reset by WDT reset 1: GPIO70 and GPIO74 registers will not be reset by WDT reset Each GPIO pin can be individually programmed to be WDT reset tolerant or not.
7 :0	RW	**Port GPIOI[7:0] WDT reset tolerance enable** 0: GPIO70 and GPIO74 registers will be reset by WDT reset 1: GPIO70 and GPIO74 registers will not be reset by WDT reset Each GPIO pin can be individually programmed to be WDT reset tolerant or not.

Offset: 0B0h	GPIO0B0: GPIO_I/J/K/L Debounce Setting Register #1	Init = 0

Bit	R/W	Description
31:24	RW	**Port GPIOL[7:0] debounce setting register #1**
23:16	RW	**Port GPIOK[7:0] debounce setting register #1**
15:8	RW	**Port GPIOJ[7:0] debounce setting register #1**
7 :0	RW	**Port GPIOI[7:0] debounce setting register #1**

Offset: 0B4h	GPIO0B4: GPIO_I/J/K/L Debounce Setting Register #2	Init = 0
Bit	**R/W**	**Description**
31:24	RW	**Port GPIOL[7:0] debounce setting register #2**
23:16	RW	**Port GPIOK[7:0] debounce setting register #2**
15:8	RW	**Port GPIOJ[7:0] debounce setting register #2**
7 :0	RW	**Port GPIOI[7:0] debounce setting register #2**

Offset: 0B8h	GPIO0B8: GPIO_I/J/K/L Input Mask Register	Init = 0
Bit	**R/W**	**Description**
31:24	RW	**Port GPIOL[7:0] input mask** 0: Read from GPIO070 will be updated. 1: Read from GPIO070 will not be updated.
23:16	RW	**Port GPIOK[7:0] input mask** 0: Read from GPIO070 will be updated. 1: Read from GPIO070 will not be updated.
15:8	RW	**Port GPIOJ[7:0] input mask** 0: Read from GPIO070 will be updated. 1: Read from GPIO070 will not be updated.
7 :0	RW	**Port GPIOI[7:0] input mask** 0: Read from GPIO070 will be updated. 1: Read from GPIO070 will not be updated.

Offset: 0C0h	GPIO0C0: GPIO_A/B/C/D Data Read Register	Init = 0
Bit	**R/W**	**Description**
31:0	R	**Data written to GPIO000.(GPIOD/GPIOC/GPIOB/GPIOA)**

Offset: 0C4h	GPIO0C4: GPIO_E/F/G/H Data Read Register	Init = 0
Bit	**R/W**	**Description**
31:0	R	**Data written to GPIO020.(GPIOH/GPIOG/GPIOF/GPIOE)**

Offset: 0C8h	GPIO0C8: GPIO_I/J/K/L Data Read Register	Init = 0
Bit	**R/W**	**Description**
31:0	R	**Data written to GPIO070.(GPIOL/GPIOK/GPIOJ/GPIOI)**

Offset: 0CCh	GPIO0CC: GPIO_M/N/O/P Data Read Register	Init = 0
Bit	**R/W**	**Description**
31:0	R	**Data written to GPIO078.(GPIOP/GPIOO/GPION/GPIOM)**

Offset: 0D0h	GPIO0D0: GPIO_Q/R/S/T Data Read Register	Init = 0
Bit	**R/W**	**Description**
31:0	R	**Data written to GPIO080.(GPIOT/GPIOS/GPIOR/GPIOQ)**

Offset: 0D4h	GPIO0D4: GPIO_U/V/W/X Data Read Register	Init = 0
Bit	**R/W**	**Description**
31:16		**Reserved**
15:0	R	**Data written to GPIO088.(GPIOV/GPIOU)**

Offset: 0D8h	GPIO0D8: GPIO_Y/Z Data Read Register	Init = 0
Bit	**R/W**	**Description**
31:28		**Reserved**
15:0	R	**Data written to GPIO1E0.(GPIOZ/GPIOY)**

Offset: 0E0h	GPIO0E0: GPIO_M/N/O/P Command Source 0	Init = 0
Bit	**R/W**	**Description**
31:25		**Reserved**
24	RW	**Port GPIOP[7:0] Command Source 0**
23:17		**Reserved**
16	RW	**Port GPIOO[7:0] Command Source 0**
15: 9		**Reserved**
8	RW	**Port GPION[7:0] Command Source 0**
7: 1		**Reserved**
0	RW	**Port GPIOM[7:0] Command Source 0**

Note :

Command Source #1	Command Source #0	Source
0	0	ARM (default)
0	1	LPC
1	0	Coprocessor CPU
1	1	Reserved

Offset: 0E4h	GPIO0E4: GPIO_M/N/O/P Command Source 1	Init = 0
Bit	**R/W**	**Description**
31:25		**Reserved**
24	RW	**Port GPIOP[7:0] Command Source 1**
23:17		**Reserved**
16	RW	**Port GPIOO[7:0] Command Source 1**
15: 9		**Reserved**
8	RW	**Port GPION[7:0] Command Source 1**
7: 1		**Reserved**
0	RW	**Port GPIOM[7:0] Command Source 1**

Note :

Command Source #1	Command Source #0	Source
0	0	ARM (default)
0	1	LPC
1	0	Coprocessor CPU
1	1	Reserved

Offset: 0E8h	GPIO0E8: GPIO_M/N/O/P Interrupt Enable Register	Init = 0
Bit	**R/W**	**Description**
31:24	RW	**Port GPIOP[7:0] interrupt enable** 0: Disable interrupt 1: Enable interrupt
23:16	RW	**Port GPIOO[7:0] interrupt enable** 0: Disable interrupt 1: Enable interrupt
15:8	RW	**Port GPION[7:0] interrupt enable** 0: Disable interrupt 1: Enable interrupt
7 :0	RW	**Port GPIOM[7:0] interrupt enable** 0: Disable interrupt 1: Enable interrupt

Offset: 0ECh	GPIO0EC: GPIO_M/N/O/P Interrupt Sensitivity Type 0 Register	Init = 0
Bit	**R/W**	**Description**
31:24	RW	**Port GPIOP[7:0] interrupt sensitivity type 0 selection** 0: Select falling-edge or level-low trigger mode 1: Select rising-edge or level-high trigger mode
23:16	RW	**Port GPIOO[7:0] interrupt sensitivity type 0 selection** 0: Select falling-edge or level-low trigger mode 1: Select rising-edge or level-high trigger mode
15:8	RW	**Port GPION[7:0] interrupt sensitivity type 0 selection** 0: Select falling-edge or level-low trigger mode 1: Select rising-edge or level-high trigger mode
7 :0	RW	**Port GPIOM[7:0] interrupt sensitivity type 0 selection** 0: Select falling-edge or level-low trigger mode 1: Select rising-edge or level-high trigger mode

Offset: 0F0h	GPIO0F0: GPIO_M/N/O/P Interrupt Sensitivity Type 1 Register	Init = 0
Bit	**R/W**	**Description**
31:24	RW	**Port GPIOP[7:0] interrupt sensitivity type 1 selection** 0: Select edge trigger mode 1: Select level trigger mode
23:16	RW	**Port GPIOO[7:0] interrupt sensitivity type 1 selection** 0: Select edge trigger mode 1: Select level trigger mode
15:8	RW	**Port GPION[7:0] interrupt sensitivity type 1 selection** 0: Select edge trigger mode 1: Select level trigger mode
7 :0	RW	**Port GPIOM[7:0] interrupt sensitivity type 1 selection** 0: Select edge trigger mode 1: Select level trigger mode

Offset: 0F4h	GPIO0F4: GPIO_M/N/O/P Interrupt Sensitivity Type 2 Register	Init = 0

Bit	R/W	Description
31:24	RW	**Port GPIOP[7:0] interrupt sensitivity type 2 selection** 0: Select edge or level trigger mode 1: Select dual-edge trigger mode
23:16	RW	**Port GPIOO[7:0] interrupt sensitivity type 2 selection** 0: Select edge or level trigger mode 1: Select dual-edge trigger mode
15:8	RW	**Port GPION[7:0] interrupt sensitivity type 2 selection** 0: Select edge or level trigger mode 1: Select dual-edge trigger mode
7 :0	RW	**Port GPIOM[7:0] interrupt sensitivity type 2 selection** 0: Select edge or level trigger mode 1: Select dual-edge trigger mode

Offset: 0F8h	GPIO0F8: GPIO_M/N/O/P Interrupt Status Register	Init = 0

Bit	R/W	Description
31:24	RW	**Port GPIOP[7:0] interrupt status register** Read 0: No interrupt pending Read 1: interrupt pending Write 0: No operation Write 1: Clear interrupt status flag
23:16	RW	**Port GPIOO[7:0] interrupt status register** Read 0: No interrupt pending Read 1: interrupt pending Write 0: No operation Write 1: Clear interrupt status flag
15:8	RW	**Port GPION[7:0] interrupt status register** Read 0: No interrupt pending Read 1: interrupt pending Write 0: No operation Write 1: Clear interrupt status flag
7 :0	RW	**Port GPIOM[7:0] interrupt status register** Read 0: No interrupt pending Read 1: interrupt pending Write 0: No operation Write 1: Clear interrupt status flag

Offset: 0FCh	GPIO0FC: GPIO_M/N/O/P Reset Tolerant Register	Init = 0

Bit	R/W	Description
31:24	RW	**Port GPIOP[7:0] WDT reset tolerance enable** 0: GPIO78 and GPIO7C registers will be reset by WDT reset 1: GPIO78 and GPIO7C registers will not be reset by WDT reset Each GPIO pin can be individually programmed to be WDT reset tolerant or not.
23:16	RW	**Port GPIOO[7:0] WDT reset tolerance enable** 0: GPIO78 and GPIO7C registers will be reset by WDT reset 1: GPIO78 and GPIO7C registers will not be reset by WDT reset Each GPIO pin can be individually programmed to be WDT reset tolerant or not.

to next page

from previous page

15:8	RW	**Port GPION[7:0] WDT reset tolerance enable** 0: GPIO78 and GPIO7C registers will be reset by WDT reset 1: GPIO78 and GPIO7C registers will not be reset by WDT reset Each GPIO pin can be individually programmed to be WDT reset tolerant or not.
7 :0	RW	**Port GPIOM[7:0] WDT reset tolerance enable** 0: GPIO78 and GPIO7C registers will be reset by WDT reset 1: GPIO78 and GPIO7C registers will not be reset by WDT reset Each GPIO pin can be individually programmed to be WDT reset tolerant or not.

Offset: 100h **GPIO100: GPIO_M/N/O/P Debounce Setting Register #1** **Init = 0**

Bit	R/W	Description
31:24	RW	**Port GPIOP[7:0] debounce setting register #1**
23:16	RW	**Port GPIOO[7:0] debounce setting register #1**
15:8	RW	**Port GPION[7:0] debounce setting register #1**
7 :0	RW	**Port GPIOM[7:0] debounce setting register #1**

Offset: 104h **GPIO104: GPIO_M/N/O/P Debounce Setting Register #2** **Init = 0**

Bit	R/W	Description
31:24	RW	**Port GPIOP[7:0] debounce setting register #2**
23:16	RW	**Port GPIOO[7:0] debounce setting register #2**
15:8	RW	**Port GPION[7:0] debounce setting register #2**
7 :0	RW	**Port GPIOM[7:0] debounce setting register #2**

Offset: 108h **GPIO108: GPIO_M/N/O/P Input Mask Register** **Init = 0**

Bit	R/W	Description
31:24	RW	**Port GPIOP[7:0] input mask** 0: Read from GPIO078 will be updated. 1: Read from GPIO078 will not be updated.
23:16	RW	**Port GPIOO[7:0] input mask** 0: Read from GPIO078 will be updated. 1: Read from GPIO078 will not be updated.
15:8	RW	**Port GPION[7:0] input mask** 0: Read from GPIO078 will be updated. 1: Read from GPIO078 will not be updated.
7 :0	RW	**Port GPIOM[7:0] input mask** 0: Read from GPIO078 will be updated. 1: Read from GPIO078 will not be updated.

Offset: 110h **GPIO110: GPIO_Q/R/S/T Command Source 0** **Init = 0**

Bit	R/W	Description
31:25		**Reserved**
24	RW	**Port GPIT[7:0] Command Source 0**
23:17		**Reserved**
16	RW	**Port GPIOS[7:0] Command Source 0**

to next page

from previous page

15: 9		**Reserved**
8	RW	**Port GPIOR[7:0] Command Source 0**
7: 1		**Reserved**
0	RW	**Port GPIOQ[7:0] Command Source 0**

Note :

Command Source #1	Command Source #0	Source
0	0	ARM (default)
0	1	LPC
1	0	Coprocessor CPU
1	1	Reserved

Offset: 114h **GPIO114: GPIO_Q/R/S/T Command Source 1** **Init = 0**

Bit	R/W	Description
31:25		**Reserved**
24	RW	**Port GPIT[7:0] Command Source 1**
23:17		**Reserved**
16	RW	**Port GPIOS[7:0] Command Source 1**
15: 9		**Reserved**
8	RW	**Port GPIOR[7:0] Command Source 1**
7: 1		**Reserved**
0	RW	**Port GPIOQ[7:0] Command Source 1**

Note :

Command Source #1	Command Source #0	Source
0	0	ARM (default)
0	1	LPC
1	0	Coprocessor CPU
1	1	Reserved

Offset: 118h **GPIO118: GPIO_Q/R/S/T Interrupt Enable Register** **Init = 0**

Bit	R/W	Description
31:24	RW	**Port GPIT[7:0] interrupt enable** 0: Disable interrupt 1: Enable interrupt
23:16	RW	**Port GPIOS[7:0] interrupt enable** 0: Disable interrupt 1: Enable interrupt
15:8	RW	**Port GPIOR[7:0] interrupt enable** 0: Disable interrupt 1: Enable interrupt
7 :0	RW	**Port GPIOQ[7:0] interrupt enable** 0: Disable interrupt 1: Enable interrupt

Offset: 11Ch	GPIO11C: GPIO_Q/R/S/T Interrupt Sensitivity Type 0 Register	Init = 0
Bit	**R/W**	**Description**
31:24	RW	**Port GPIT[7:0] interrupt sensitivity type 0 selection** 0: Select falling-edge or level-low trigger mode 1: Select rising-edge or level-high trigger mode
23:16	RW	**Port GPIOS[7:0] interrupt sensitivity type 0 selection** 0: Select falling-edge or level-low trigger mode 1: Select rising-edge or level-high trigger mode
15:8	RW	**Port GPIOR[7:0] interrupt sensitivity type 0 selection** 0: Select falling-edge or level-low trigger mode 1: Select rising-edge or level-high trigger mode
7 :0	RW	**Port GPIOQ[7:0] interrupt sensitivity type 0 selection** 0: Select falling-edge or level-low trigger mode 1: Select rising-edge or level-high trigger mode

Offset: 120h	GPIO120: GPIO_Q/R/S/T Interrupt Sensitivity Type 1 Register	Init = 0
Bit	**R/W**	**Description**
31:24	RW	**Port GPIT[7:0] interrupt sensitivity type 1 selection** 0: Select edge trigger mode 1: Select level trigger mode
23:16	RW	**Port GPIOS[7:0] interrupt sensitivity type 1 selection** 0: Select edge trigger mode 1: Select level trigger mode
15:8	RW	**Port GPIOR[7:0] interrupt sensitivity type 1 selection** 0: Select edge trigger mode 1: Select level trigger mode
7 :0	RW	**Port GPIOQ[7:0] interrupt sensitivity type 1 selection** 0: Select edge trigger mode 1: Select level trigger mode

Offset: 124h	GPIO124: GPIO_Q/R/S/T Interrupt Sensitivity Type 2 Register	Init = 0
Bit	**R/W**	**Description**
31:24	RW	**Port GPIT[7:0] interrupt sensitivity type 2 selection** 0: Select edge or level trigger mode 1: Select dual-edge trigger mode
23:16	RW	**Port GPIOS[7:0] interrupt sensitivity type 2 selection** 0: Select edge or level trigger mode 1: Select dual-edge trigger mode
15:8	RW	**Port GPIOR[7:0] interrupt sensitivity type 2 selection** 0: Select edge or level trigger mode 1: Select dual-edge trigger mode
7 :0	RW	**Port GPIOQ[7:0] interrupt sensitivity type 2 selection** 0: Select edge or level trigger mode 1: Select dual-edge trigger mode

Offset: 128h	GPIO128: GPIO_Q/R/S/T Interrupt Status Register	Init = 0

Bit	R/W	Description
31:24	RW	**Port GPIT[7:0] interrupt status register** Read 0: No interrupt pending Read 1: interrupt pending Write 0: No operation Write 1: Clear interrupt status flag
23:16	RW	**Port GPIOS[7:0] interrupt status register** Read 0: No interrupt pending Read 1: interrupt pending Write 0: No operation Write 1: Clear interrupt status flag
15:8	RW	**Port GPIOR[7:0] interrupt status register** Read 0: No interrupt pending Read 1: interrupt pending Write 0: No operation Write 1: Clear interrupt status flag
7 :0	RW	**Port GPIOQ[7:0] interrupt status register** Read 0: No interrupt pending Read 1: interrupt pending Write 0: No operation Write 1: Clear interrupt status flag

Offset: 12Ch	GPIO12C: GPIO_Q/R/S/T Reset Tolerant Register	Init = 0

Bit	R/W	Description
31:24	RW	**Port GPIT[7:0] WDT reset tolerance enable** 0: GPIO80 and GPIO84 registers will be reset by WDT reset 1: GPIO80 and GPIO84 registers will not be reset by WDT reset Each GPIO pin can be individually programmed to be WDT reset tolerant or not.
23:16	RW	**Port GPIOS[7:0] WDT reset tolerance enable** 0: GPIO80 and GPIO84 registers will be reset by WDT reset 1: GPIO80 and GPIO84 registers will not be reset by WDT reset Each GPIO pin can be individually programmed to be WDT reset tolerant or not.
15:8	RW	**Port GPIOR[7:0] WDT reset tolerance enable** 0: GPIO80 and GPIO84 registers will be reset by WDT reset 1: GPIO80 and GPIO84 registers will not be reset by WDT reset Each GPIO pin can be individually programmed to be WDT reset tolerant or not.
7 :0	RW	**Port GPIOQ[7:0] WDT reset tolerance enable** 0: GPIO80 and GPIO84 registers will be reset by WDT reset 1: GPIO80 and GPIO84 registers will not be reset by WDT reset Each GPIO pin can be individually programmed to be WDT reset tolerant or not.

Offset: 130h	GPIO130: GPIO_Q/R/S/T Debounce Setting Register #1	Init = 0

Bit	R/W	Description
31:24	RW	**Port GPIT[7:0] debounce setting register #1**
23:16	RW	**Port GPIOS[7:0] debounce setting register #1**
15:8	RW	**Port GPIOR[7:0] debounce setting register #1**
7 :0	RW	**Port GPIOQ[7:0] debounce setting register #1**

Offset: 134h	GPIO134: GPIO_Q/R/S/T Debounce Setting Register #2	Init = 0
Bit	**R/W**	**Description**
31:24	RW	**Port GPIT[7:0] debounce setting register #2**
23:16	RW	**Port GPIOS[7:0] debounce setting register #2**
15:8	RW	**Port GPIOR[7:0] debounce setting register #2**
7 :0	RW	**Port GPIOQ[7:0] debounce setting register #2**

Offset: 138h	GPIO138: GPIO_Q/R/S/T Input Mask Register	Init = 0
Bit	**R/W**	**Description**
31:24	RW	**Port GPIT[7:0] input mask** 0: Read from GPIO080 will be updated. 1: Read from GPIO080 will not be updated.
23:16	RW	**Port GPIOS[7:0] input mask** 0: Read from GPIO080 will be updated. 1: Read from GPIO080 will not be updated.
15:8	RW	**Port GPIOR[7:0] input mask** 0: Read from GPIO080 will be updated. 1: Read from GPIO080 will not be updated.
7 :0	RW	**Port GPIOQ[7:0] input mask** 0: Read from GPIO080 will be updated. 1: Read from GPIO080 will not be updated.

Offset: 140h	GPIO140: GPIO_U/V/W/X Command Source 0	Init = 0
Bit	**R/W**	**Description**
31:25		**Reserved**
24	RW	**Port GPIOX[7:0] Command Source 0**
23:17		**Reserved**
16	RW	**Port GPIOW[7:0] Command Source 0**
15: 9		**Reserved**
8	RW	**Port GPIOV[7:0] Command Source 0**
7: 1		**Reserved**
0	RW	**Port GPIU[7:0] Command Source 0**

Note :

Command Source #1	Command Source #0	Source
0	0	ARM (default)
0	1	LPC
1	0	Coprocessor CPU
1	1	Reserved

Offset: 144h	GPIO144: GPIO_U/V/W/X Command Source 1	Init = 0
Bit	**R/W**	**Description**
31:25		**Reserved**
24	RW	**Port GPIOX[7:0] Command Source 1**
23:17		**Reserved**

to next page

from previous page

16	RW	**Port GPIOW[7:0] Command Source 1**
15: 9		**Reserved**
8	RW	**Port GPIOV[7:0] Command Source 1**
7: 1		**Reserved**
0	RW	**Port GPIU[7:0] Command Source 1**

Note :

Command Source #1	Command Source #0	Source
0	0	ARM (default)
0	1	LPC
1	0	Coprocessor CPU
1	1	Reserved

Offset: 148h **GPIO148: GPIO_U/V/W/X Interrupt Enable Register** **Init = 0**

Bit	R/W	Description
31:24	RW	**Port GPIOX[7:0] interrupt enable** 0: Disable interrupt 1: Enable interrupt
23:16	RW	**Port GPIOW[7:0] interrupt enable** 0: Disable interrupt 1: Enable interrupt
15:8	RW	**Port GPIOV[7:0] interrupt enable** 0: Disable interrupt 1: Enable interrupt
7 :0	RW	**Port GPIU[7:0] interrupt enable** 0: Disable interrupt 1: Enable interrupt

Offset: 14Ch **GPIO14C: GPIO_U/V/W/X Interrupt Sensitivity Type 0 Register** **Init = 0**

Bit	R/W	Description
31:24	RW	**Port GPIOX[7:0] interrupt sensitivity type 0 selection** 0: Select falling-edge or level-low trigger mode 1: Select rising-edge or level-high trigger mode
23:16	RW	**Port GPIOW[7:0] interrupt sensitivity type 0 selection** 0: Select falling-edge or level-low trigger mode 1: Select rising-edge or level-high trigger mode
15:8	RW	**Port GPIOV[7:0] interrupt sensitivity type 0 selection** 0: Select falling-edge or level-low trigger mode 1: Select rising-edge or level-high trigger mode
7 :0	RW	**Port GPIU[7:0] interrupt sensitivity type 0 selection** 0: Select falling-edge or level-low trigger mode 1: Select rising-edge or level-high trigger mode

Offset: 150h	GPIO150: GPIO_U/V/W/X Interrupt Sensitivity Type 1 Register	Init = 0
Bit	**R/W**	**Description**
31:24	RW	**Port GPIOX[7:0] interrupt sensitivity type 1 selection** 0: Select edge trigger mode 1: Select level trigger mode
23:16	RW	**Port GPIOW[7:0] interrupt sensitivity type 1 selection** 0: Select edge trigger mode 1: Select level trigger mode
15:8	RW	**Port GPIOV[7:0] interrupt sensitivity type 1 selection** 0: Select edge trigger mode 1: Select level trigger mode
7 :0	RW	**Port GPIU[7:0] interrupt sensitivity type 1 selection** 0: Select edge trigger mode 1: Select level trigger mode

Offset: 154h	GPIO154: GPIO_U/V/W/X Interrupt Sensitivity Type 2 Register	Init = 0
Bit	**R/W**	**Description**
31:24	RW	**Port GPIOX[7:0] interrupt sensitivity type 2 selection** 0: Select edge or level trigger mode 1: Select dual-edge trigger mode
23:16	RW	**Port GPIOW[7:0] interrupt sensitivity type 2 selection** 0: Select edge or level trigger mode 1: Select dual-edge trigger mode
15:8	RW	**Port GPIOV[7:0] interrupt sensitivity type 2 selection** 0: Select edge or level trigger mode 1: Select dual-edge trigger mode
7 :0	RW	**Port GPIU[7:0] interrupt sensitivity type 2 selection** 0: Select edge or level trigger mode 1: Select dual-edge trigger mode

Offset: 158h	GPIO158: GPIO_U/V/W/X Interrupt Status Register	Init = 0
Bit	**R/W**	**Description**
31:24	RW	**Port GPIOX[7:0] interrupt status register** Read 0: No interrupt pending Read 1: interrupt pending Write 0: No operation Write 1: Clear interrupt status flag
23:16	RW	**Port GPIOW[7:0] interrupt status register** Read 0: No interrupt pending Read 1: interrupt pending Write 0: No operation Write 1: Clear interrupt status flag
15:8	RW	**Port GPIOV[7:0] interrupt status register** Read 0: No interrupt pending Read 1: interrupt pending Write 0: No operation Write 1: Clear interrupt status flag

to next page

from previous page

7 :0	RW	**Port GPIU[7:0] interrupt status register** Read 0: No interrupt pending Read 1: interrupt pending Write 0: No operation Write 1: Clear interrupt status flag

Offset: 15Ch **GPIO15C: GPIO_U/V/W/X Reset Tolerant Register** **Init = 0**

Bit	R/W	Description
31:24	RW	**Port GPIOX[7:0] WDT reset tolerance enable** 0: GPIO80 and GPIO84 registers will be reset by WDT reset 1: GPIO80 and GPIO84 registers will not be reset by WDT reset Each GPIO pin can be individually programmed to be WDT reset tolerant or not.
23:16	RW	**Port GPIOW[7:0] WDT reset tolerance enable** 0: GPIO80 and GPIO84 registers will be reset by WDT reset 1: GPIO80 and GPIO84 registers will not be reset by WDT reset Each GPIO pin can be individually programmed to be WDT reset tolerant or not.
15:8	RW	**Port GPIOV[7:0] WDT reset tolerance enable** 0: GPIO80 and GPIO84 registers will be reset by WDT reset 1: GPIO80 and GPIO84 registers will not be reset by WDT reset Each GPIO pin can be individually programmed to be WDT reset tolerant or not.
7 :0	RW	**Port GPIU[7:0] WDT reset tolerance enable** 0: GPIO80 and GPIO84 registers will be reset by WDT reset 1: GPIO80 and GPIO84 registers will not be reset by WDT reset Each GPIO pin can be individually programmed to be WDT reset tolerant or not.

Offset: 160h **GPIO160: GPIO_U/V/W/X Debounce Setting Register #1** **Init = 0**

Bit	R/W	Description
31:24	RW	**Port GPIOX[7:0] debounce setting register #1**
23:16	RW	**Port GPIOW[7:0] debounce setting register #1**
15:8	RW	**Port GPIOV[7:0] debounce setting register #1**
7 :0	RW	**Port GPIU[7:0] debounce setting register #1**

Offset: 164h **GPIO164: GPIO_U/V/W/X Debounce Setting Register #2** **Init = 0**

Bit	R/W	Description
31:24	RW	**Port GPIOX[7:0] debounce setting register #2**
23:16	RW	**Port GPIOW[7:0] debounce setting register #2**
15:8	RW	**Port GPIOV[7:0] debounce setting register #2**
7 :0	RW	**Port GPIU[7:0] debounce setting register #2**

Offset: 168h **GPIO168: GPIO_U/V/W/X Input Mask Register** **Init = 0**

Bit	R/W	Description
31:24	RW	**Port GPIOX[7:0] input mask** 0: Read from GPIO080 will be updated. 1: Read from GPIO080 will not be updated.

to next page

from previous page

23:16	RW	**Port GPIOW[7:0] input mask** 0: Read from GPIO080 will be updated. 1: Read from GPIO080 will not be updated.
15:8	RW	**Port GPIOV[7:0] input mask** 0: Read from GPIO080 will be updated. 1: Read from GPIO080 will not be updated.
7 :0	RW	**Port GPIU[7:0] input mask** 0: Read from GPIO080 will be updated. 1: Read from GPIO080 will not be updated.

Offset: 170h **GPIO170: GPIO_Y/Z Command Source 0** **Init = 0**

Bit	R/W	Description
31: 9		**Reserved**
8	RW	**Port GPIOZ[7:0] Command Source 0**
7: 1		**Reserved**
0	RW	**Port GPIOY[7:0] Command Source 0**

Note :

Command Source #1	Command Source #0	Source
0	0	ARM (default)
0	1	LPC
1	0	Coprocessor CPU
1	1	Reserved

Offset: 174h **GPIO174: GPIO_Y/Z Command Source 1** **Init = 0**

Bit	R/W	Description
31: 9		**Reserved**
8	RW	**Port GPIOZ[7:0] Command Source 1**
7: 1		**Reserved**
0	RW	**Port GPIOY[7:0] Command Source 1**

Note :

Command Source #1	Command Source #0	Source
0	0	ARM (default)
0	1	LPC
1	0	Coprocessor CPU
1	1	Reserved

Offset: 178h **GPIO178: GPIO_Y/Z Interrupt Enable Register** **Init = 0**

Bit	R/W	Description
31:16		**Reserved**
15:8	RW	**Port GPIOZ[7:0] interrupt enable** 0: Disable interrupt 1: Enable interrupt

to next page

from previous page

7 :0	RW	**Port GPIOY[7:0] interrupt enable** 0: Disable interrupt 1: Enable interrupt

Offset: 17Ch **GPIO17C: GPIO_Y/Z Interrupt Sensitivity Type 0 Register** **Init = 0**

Bit	R/W	Description
31:16		**Reserved**
15:8	RW	**Port GPIOZ[7:0] interrupt sensitivity type 0 selection** 0: Select falling-edge or level-low trigger mode 1: Select rising-edge or level-high trigger mode
7 :0	RW	**Port GPIOY[7:0] interrupt sensitivity type 0 selection** 0: Select falling-edge or level-low trigger mode 1: Select rising-edge or level-high trigger mode

Offset: 180h **GPIO180: GPIO_Y/Z Interrupt Sensitivity Type 1 Register** **Init = 0**

Bit	R/W	Description
31:16		**Reserved**
15:8	RW	**Port GPIOZ[7:0] interrupt sensitivity type 1 selection** 0: Select edge trigger mode 1: Select level trigger mode
7 :0	RW	**Port GPIOY[7:0] interrupt sensitivity type 1 selection** 0: Select edge trigger mode 1: Select level trigger mode

Offset: 184h **GPIO184: GPIO_Y/Z Interrupt Sensitivity Type 2 Register** **Init = 0**

Bit	R/W	Description
31:16		**Reserved**
15:8	RW	**Port GPIOZ[7:0] interrupt sensitivity type 2 selection** 0: Select edge or level trigger mode 1: Select dual-edge trigger mode
7 :0	RW	**Port GPIOY[7:0] interrupt sensitivity type 2 selection** 0: Select edge or level trigger mode 1: Select dual-edge trigger mode

Offset: 188h **GPIO188: GPIO_Y/Z Interrupt Status Register** **Init = 0**

Bit	R/W	Description
31:16		**Reserved**
15:8	RW	**Port GPIOZ[7:0] interrupt status register** Read 0: No interrupt pending Read 1: interrupt pending Write 0: No operation Write 1: Clear interrupt status flag
7 :0	RW	**Port GPIOY[7:0] interrupt status register** Read 0: No interrupt pending Read 1: interrupt pending Write 0: No operation Write 1: Clear interrupt status flag

Offset: 18Ch	GPIO18C: GPIO_Y/Z Reset Tolerant Register	Init = 0
Bit	**R/W**	**Description**
31:16		**Reserved**
15:8	RW	**Port GPIOZ[7:0] WDT reset tolerance enable** 0: GPIO1E0 and GPIO1E4 registers will be reset by WDT reset 1: GPIO1E0 and GPIO1E4 registers will not be reset by WDT reset Each GPIO pin can be individually programmed to be WDT reset tolerant or not.
7 :0	RW	**Port GPIOY[7:0] WDT reset tolerance enable** 0: GPIO1E0 and GPIO1E4 registers will be reset by WDT reset 1: GPIO1E0 and GPIO1E4 registers will not be reset by WDT reset Each GPIO pin can be individually programmed to be WDT reset tolerant or not.

Offset: 190h	GPIO190: GPIO_Y/Z Debounce Setting Register #1	Init = 0
Bit	**R/W**	**Description**
31:16		**Reserved**
15:8	RW	**Port GPIOZ[7:0] debounce setting register #1**
7 :0	RW	**Port GPIOY[7:0] debounce setting register #1**

Offset: 194h	GPIO194: GPIO_Y/Z Debounce Setting Register #2	Init = 0
Bit	**R/W**	**Description**
31:16		**Reserved**
15:8	RW	**Port GPIOZ[7:0] debounce setting register #2**
7 :0	RW	**Port GPIOY[7:0] debounce setting register #2**

Offset: 198h	GPIO198: GPIO_Y/Z Input Mask Register	Init = 0
Bit	**R/W**	**Description**
31:16		**Reserved**
15:0	RW	**Port GPIOZ[7:0] input mask** 0: Read from GPIO080 will be updated. 1: Read from GPIO080 will not be updated.
7 :0	RW	**Port GPIOY[7:0] input mask** 0: Read from GPIO080 will be updated. 1: Read from GPIO080 will not be updated.

Offset: 1D0h	GPIO1D0: GPIO_A/B/C/D Input Mask Register	Init = 0
Bit	**R/W**	**Description**
31:24	RW	**Port GPIOD[7:0] input mask** 0: Read from GPIO080 will be updated. 1: Read from GPIO080 will not be updated.
23:16	RW	**Port GPIOC[7:0] input mask** 0: Read from GPIO080 will be updated. 1: Read from GPIO080 will not be updated.
15:8	RW	**Port GPIOB[7:0] input mask** 0: Read from GPIO080 will be updated. 1: Read from GPIO080 will not be updated.

to next page

from previous page

7 :0	RW	**Port GPIOA[7:0] input mask** 0: Read from GPIO080 will be updated. 1: Read from GPIO080 will not be updated.

Offset: 1D4h **GPIO1D4: GPIO_E/F/G/H Input Mask Register** **Init = 0**

Bit	R/W	Description
31:24	RW	**Port GPIOH[7:0] input mask** 0: Read from GPIO080 will be updated. 1: Read from GPIO080 will not be updated.
23:16	RW	**Port GPIOG[7:0] input mask** 0: Read from GPIO080 will be updated. 1: Read from GPIO080 will not be updated.
15:8	RW	**Port GPIOF[7:0] input mask** 0: Read from GPIO080 will be updated. 1: Read from GPIO080 will not be updated.
7 :0	RW	**Port GPIOE[7:0] input mask** 0: Read from GPIO080 will be updated. 1: Read from GPIO080 will not be updated.

Offset: 1E0h **GPIO1E0: GPIO_Y/Z Data Value Register 0** **Init = 0**

Bit	R/W	Description
31:16		**Reserved**
15:8	RW	**Port GPIOZ[7:0] data register**
7 :0	RW	**Port GPIOY[7:0] data register**

Offset: 1E4h **GPIO1E4: GPIO_Y/Z Direction Register 0** **Init = 0**

Bit	R/W	Description
31:16		**Reserved**
15:8	RW	**Port GPIOZ[7:0] direction control** 0: Select input mode 1: Select output mode
7 :0	RW	**Port GPIOY[7:0] direction control** 0: Select input mode 1: Select output mode

Offset: 22Ch **GPIO22C: GPIO_A/B/C/D New Write Command Source Register** **Init = 0**

Bit	R/W	Description
31	RW1S	**Write Protection of GPIO22C[31:24] until RstFull**
30	RW	**GPIOD Command Source Mode selection** 0: Legacy mode. Command Source works as GPIO060 and GPIO064 1: New mode. Command Source works by GPIO22C
29	RW	**GPIOD Write Permission for Others**
28	RW	**GPIOD Write Permission for MST5 GPIO2D0[24:20]**
27	RW	**GPIOD Write Permission for MST4 GPIO2D0[19:15]**
26	RW	**GPIOD Write Permission for MST3 GPIO2D0[14:10]**

to next page

from previous page

25	RW	**GPIOD Write Permission for MST2 GPIO2D0[9: 5]**
24	RW	**GPIOD Write Permission for MST1 GPIO2D0[4: 0]**
23	RW1S	**Write Protection of GPIO22C[23:16] until RstFull**
22	RW	**GPIOC Command Source Mode selection** 0: Legacy mode. Command Source works as GPIO060 and GPIO064 1: New mode. Command Source works by GPIO22C
21	RW	**GPIOC Write Permission for Others**
20	RW	**GPIOC Write Permission for MST5 GPIO2D0[24:20]**
19	RW	**GPIOC Write Permission for MST4 GPIO2D0[19:15]**
18	RW	**GPIOC Write Permission for MST3 GPIO2D0[14:10]**
17	RW	**GPIOC Write Permission for MST2 GPIO2D0[9: 5]**
16	RW	**GPIOC Write Permission for MST1 GPIO2D0[4: 0]**
15	RW1S	**Write Protection of GPIO22C[15:08] until RstFull**
14	RW	**GPIOB Command Source Mode selection** 0: Legacy mode. Command Source works as GPIO060 and GPIO064 1: New mode. Command Source works by GPIO22C
13	RW	**GPIOB Write Permission for Others**
12	RW	**GPIOB Write Permission for MST5 GPIO2D0[24:20]**
11	RW	**GPIOB Write Permission for MST4 GPIO2D0[19:15]**
10	RW	**GPIOB Write Permission for MST3 GPIO2D0[14:10]**
9	RW	**GPIOB Write Permission for MST2 GPIO2D0[9: 5]**
8	RW	**GPIOB Write Permission for MST1 GPIO2D0[4: 0]**
7	RW1S	**Write Protection of GPIO22C[07:00] until RstFull**
6	RW	**GPIOA Command Source Mode selection** 0: Legacy mode. Command Source works as GPIO060 and GPIO064 1: New mode. Command Source works by GPIO22C
5	RW	**GPIOA Write Permission for Others**
4	RW	**GPIOA Write Permission for MST5 GPIO2D0[24:20]**
3	RW	**GPIOA Write Permission for MST4 GPIO2D0[19:15]**
2	RW	**GPIOA Write Permission for MST3 GPIO2D0[14:10]**
1	RW	**GPIOA Write Permission for MST2 GPIO2D0[9: 5]**
0	RW	**GPIOA Write Permission for MST1 GPIO2D0[4: 0]**

Offset: 230h **GPIO230: GPIO_A/B/C/D New Read Command Source Register** **Init = 0**

Bit	R/W	Description
31	RW1S	**Write Protection of GPIO230[31:24] until RstFull**
30	RW	**GPIOD Command Source Mode selection** 0: Legacy mode. Command Source works as GPIO060 and GPIO064 1: New mode. Command Source works by GPIO230
29	RW	**GPIOD Read Permission for Others**
28	RW	**GPIOD Read Permission for MST5 GPIO2D0[24:20]**
27	RW	**GPIOD Read Permission for MST4 GPIO2D0[19:15]**
26	RW	**GPIOD Read Permission for MST3 GPIO2D0[14:10]**
25	RW	**GPIOD Read Permission for MST2 GPIO2D0[9: 5]**

to next page

from previous page

24	RW	**GPIOD Read Permission for MST1 GPIO2D0[4: 0]**
23	RW1S	**Write Protection of GPIO230[23:16] until RstFull**
22	RW	**GPIOC Command Source Mode selection** 0: Legacy mode. Command Source works as GPIO060 and GPIO064 1: New mode. Command Source works by GPIO230
21	RW	**GPIOC Read Permission for Others**
20	RW	**GPIOC Read Permission for MST5 GPIO2D0[24:20]**
19	RW	**GPIOC Read Permission for MST4 GPIO2D0[19:15]**
18	RW	**GPIOC Read Permission for MST3 GPIO2D0[14:10]**
17	RW	**GPIOC Read Permission for MST2 GPIO2D0[9: 5]**
16	RW	**GPIOC Read Permission for MST1 GPIO2D0[4: 0]**
15	RW1S	**Write Protection of GPIO230[15:08] until RstFull**
14	RW	**GPIOB Command Source Mode selection** 0: Legacy mode. Command Source works as GPIO060 and GPIO064 1: New mode. Command Source works by GPIO230
13	RW	**GPIOB Read Permission for Others**
12	RW	**GPIOB Read Permission for MST5 GPIO2D0[24:20]**
11	RW	**GPIOB Read Permission for MST4 GPIO2D0[19:15]**
10	RW	**GPIOB Read Permission for MST3 GPIO2D0[14:10]**
9	RW	**GPIOB Read Permission for MST2 GPIO2D0[9: 5]**
8	RW	**GPIOB Read Permission for MST1 GPIO2D0[4: 0]**
7	RW1S	**Write Protection of GPIO230[07:00] until RstFull**
6	RW	**GPIOA Command Source Mode selection** 0: Legacy mode. Command Source works as GPIO060 and GPIO064 1: New mode. Command Source works by GPIO230
5	RW	**GPIOA Read Permission for Others**
4	RW	**GPIOA Read Permission for MST5 GPIO2D0[24:20]**
3	RW	**GPIOA Read Permission for MST4 GPIO2D0[19:15]**
2	RW	**GPIOA Read Permission for MST3 GPIO2D0[14:10]**
1	RW	**GPIOA Read Permission for MST2 GPIO2D0[9: 5]**
0	RW	**GPIOA Read Permission for MST1 GPIO2D0[4: 0]**

Offset: 238h **GPIO238: GPIO_A/B/C/D Blink Counter Selection #1** **Init = 0**

Bit	R/W	Description
31:24	RW	**GPIOD Blink Counter Selection #1**
23:16	RW	**GPIOC Blink Counter Selection #1**
15: 8	RW	**GPIOB Blink Counter Selection #1**
7: 0	RW	**GPIOA Blink Counter Selection #1**

Offset: 23Ch **GPIO23C: GPIO_A/B/C/D Blink Counter Selection #2** **Init = 0**

Bit	R/W	Description
31:24	RW	**GPIOD Blink Counter Selection #2**
23:16	RW	**GPIOC Blink Counter Selection #2**

to next page

from previous page

15: 8	RW	**GPIOB Blink Counter Selection #2**
7: 0	RW	**GPIOA Blink Counter Selection #2**

Offset: 27Ch **GPIO27C: GPIO_E/F/G/H New Write Command Source Register** **Init = 0**

Bit	R/W	Description
31	RW1S	**Write Protection of GPIO27C[31:24] until RstFull**
30	RW	**GPIOH Command Source Mode selection** 0: Legacy mode. Command Source works as GPIO068 and GPIO06C 1: New mode. Command Source works by GPIO27C
29	RW	**GPIOH Write Permission for Others**
28	RW	**GPIOH Write Permission for MST5 GPIO2D0[24:20]**
27	RW	**GPIOH Write Permission for MST4 GPIO2D0[19:15]**
26	RW	**GPIOH Write Permission for MST3 GPIO2D0[14:10]**
25	RW	**GPIOH Write Permission for MST2 GPIO2D0[9: 5]**
24	RW	**GPIOH Write Permission for MST1 GPIO2D0[4: 0]**
23	RW1S	**Write Protection of GPIO27C[23:16] until RstFull**
22	RW	**GPIOG Command Source Mode selection** 0: Legacy mode. Command Source works as GPIO068 and GPIO06C 1: New mode. Command Source works by GPIO27C
21	RW	**GPIOG Write Permission for Others**
20	RW	**GPIOG Write Permission for MST5 GPIO2D0[24:20]**
19	RW	**GPIOG Write Permission for MST4 GPIO2D0[19:15]**
18	RW	**GPIOG Write Permission for MST3 GPIO2D0[14:10]**
17	RW	**GPIOG Write Permission for MST2 GPIO2D0[9: 5]**
16	RW	**GPIOG Write Permission for MST1 GPIO2D0[4: 0]**
15	RW1S	**Write Protection of GPIO27C[15:08] until RstFull**
14	RW	**GPIOF Command Source Mode selection** 0: Legacy mode. Command Source works as GPIO068 and GPIO06C 1: New mode. Command Source works by GPIO27C
13	RW	**GPIOF Write Permission for Others**
12	RW	**GPIOF Write Permission for MST5 GPIO2D0[24:20]**
11	RW	**GPIOF Write Permission for MST4 GPIO2D0[19:15]**
10	RW	**GPIOF Write Permission for MST3 GPIO2D0[14:10]**
9	RW	**GPIOF Write Permission for MST2 GPIO2D0[9: 5]**
8	RW	**GPIOF Write Permission for MST1 GPIO2D0[4: 0]**
7	RW1S	**Write Protection of GPIO27C[07:00] until RstFull**
6	RW	**GPIOE Command Source Mode selection** 0: Legacy mode. Command Source works as GPIO068 and GPIO06C 1: New mode. Command Source works by GPIO27C
5	RW	**GPIOE Write Permission for Others**
4	RW	**GPIOE Write Permission for MST5 GPIO2D0[24:20]**
3	RW	**GPIOE Write Permission for MST4 GPIO2D0[19:15]**
2	RW	**GPIOE Write Permission for MST3 GPIO2D0[14:10]**
1	RW	**GPIOE Write Permission for MST2 GPIO2D0[9: 5]**
0	RW	**GPIOE Write Permission for MST1 GPIO2D0[4: 0]**

Offset: 280h	GPIO280: GPIO_E/F/G/H New Read Command Source Register	Init = 0
Bit	**R/W**	**Description**
31	RW1S	**Write Protection of GPIO280[31:24] until RstFull**
30	RW	**GPIOH Command Source Mode selection** 0: Legacy mode. Command Source works as GPIO068 and GPIO06C 1: New mode. Command Source works by GPIO280
29	RW	**GPIOH Read Permission for Others**
28	RW	**GPIOH Read Permission for MST5 GPIO2D0[24:20]**
27	RW	**GPIOH Read Permission for MST4 GPIO2D0[19:15]**
26	RW	**GPIOH Read Permission for MST3 GPIO2D0[14:10]**
25	RW	**GPIOH Read Permission for MST2 GPIO2D0[9: 5]**
24	RW	**GPIOH Read Permission for MST1 GPIO2D0[4: 0]**
23	RW1S	**Write Protection of GPIO280[23:16] until RstFull**
22	RW	**GPIOG Command Source Mode selection** 0: Legacy mode. Command Source works as GPIO068 and GPIO06C 1: New mode. Command Source works by GPIO280
21	RW	**GPIOG Read Permission for Others**
20	RW	**GPIOG Read Permission for MST5 GPIO2D0[24:20]**
19	RW	**GPIOG Read Permission for MST4 GPIO2D0[19:15]**
18	RW	**GPIOG Read Permission for MST3 GPIO2D0[14:10]**
17	RW	**GPIOG Read Permission for MST2 GPIO2D0[9: 5]**
16	RW	**GPIOG Read Permission for MST1 GPIO2D0[4: 0]**
15	RW1S	**Write Protection of GPIO280[15:08] until RstFull**
14	RW	**GPIOF Command Source Mode selection** 0: Legacy mode. Command Source works as GPIO068 and GPIO06C 1: New mode. Command Source works by GPIO280
13	RW	**GPIOF Read Permission for Others**
12	RW	**GPIOF Read Permission for MST5 GPIO2D0[24:20]**
11	RW	**GPIOF Read Permission for MST4 GPIO2D0[19:15]**
10	RW	**GPIOF Read Permission for MST3 GPIO2D0[14:10]**
9	RW	**GPIOF Read Permission for MST2 GPIO2D0[9: 5]**
8	RW	**GPIOF Read Permission for MST1 GPIO2D0[4: 0]**
7	RW1S	**Write Protection of GPIO280[07:00] until RstFull**
6	RW	**GPIOE Command Source Mode selection** 0: Legacy mode. Command Source works as GPIO068 and GPIO06C 1: New mode. Command Source works by GPIO280
5	RW	**GPIOE Read Permission for Others**
4	RW	**GPIOE Read Permission for MST5 GPIO2D0[24:20]**
3	RW	**GPIOE Read Permission for MST4 GPIO2D0[19:15]**
2	RW	**GPIOE Read Permission for MST3 GPIO2D0[14:10]**
1	RW	**GPIOE Read Permission for MST2 GPIO2D0[9: 5]**
0	RW	**GPIOE Read Permission for MST1 GPIO2D0[4: 0]**

Offset: 288h	GPIO288: GPIO_E/F/G/H Blink Counter Selection #1	Init = 0
Bit	**R/W**	**Description**
31:24	RW	**GPIOH Blink Counter Selection #1**
23:16	RW	**GPIOG Blink Counter Selection #1**
15: 8	RW	**GPIOF Blink Counter Selection #1**
7: 0	RW	**GPIOE Blink Counter Selection #1**

Offset: 28Ch	GPIO28C: GPIO_E/F/G/H Blink Counter Selection #2	Init = 0
Bit	**R/W**	**Description**
31:24	RW	**GPIOH Blink Counter Selection #2**
23:16	RW	**GPIOG Blink Counter Selection #2**
15: 8	RW	**GPIOF Blink Counter Selection #2**
7: 0	RW	**GPIOE Blink Counter Selection #2**

Offset: 2B0h	GPIO2B0: GPIO Blink Source Clock Division	Init = 0
Bit	**R/W**	**Description**
31:24	R	**Reserved**
23: 0	RW	**Blink Source Clock Division. Only effective to PCLK as source clock**

Offset: 2B4h	GPIO2B4: GPIO Blink Control	Init = 0
Bit	**R/W**	**Description**
31:10	R	**Reserved**
9	RW	**Clock Source of Counter #3 Selection** 0: Choose PCLK/(GPIO2B0+1) as source clock. 1: Choose CLK1M as source clock.
8	RW	**Enable Blink Counter #3**
7: 6	R	**Reserved**
5	RW	**Clock Source of Counter #2 Selection** 0: Choose PCLK/(GPIO2B0+1) as source clock. 1: Choose CLK1M as source clock.
4	RW	**Enable Blink Counter #2**
3: 2	R	**Reserved**
1	RW	**Clock Source of Counter #1 Selection** 0: Choose PCLK/(GPIO2B0+1) as source clock. 1: Choose CLK1M as source clock.
0	RW	**Enable Blink Counter #1**

Offset: 2B8h	GPIO2B8: GPIO Blink Counter #1 Configuration	Init = 0
Bit	**R/W**	**Description**
31:16	RW	**Number of Source Clock cycles for Low** 0: 1 cycle. 1: 2 cycles. ... n: n+1 cycles

to next page

from previous page

15: 0	RW	**Number of Source Clock cycles for High** 0: 1 cycle. 1: 2 cycles. ... n: n+1 cycles

Offset: 2BCh **GPIO2BC: GPIO Blink Counter #2 Configuration** **Init = 0**

Bit	R/W	Description
31:16	RW	**Number of Source Clock cycles for Low** 0: 1 cycle. 1: 2 cycles. ... n: n+1 cycles
15: 0	RW	**Number of Source Clock cycles for High** 0: 1 cycle. 1: 2 cycles. ... n: n+1 cycles

Offset: 2C0h **GPIO2C0: GPIO Blink Counter #3 Configuration** **Init = 0**

Bit	R/W	Description
31:16	RW	**Number of Source Clock cycles for Low** 0: 1 cycle. 1: 2 cycles. ... n: n+1 cycles
15: 0	RW	**Number of Source Clock cycles for High** 0: 1 cycle. 1: 2 cycles. ... n: n+1 cycles

Blink Counter Selection #2	Blink Counter Selection #1	Function
0	0	Diable Blink
0	1	Select Counter #1 as Blink Counter
1	0	Select Counter #2 as Blink Counter
1	1	Select Counter #3 as Blink Counter

Offset: 2D0h **GPIO2D0: GPIO Command Source Selection** **Init = 0x109A61**

Bit	R/W	Description
31	RW1S	**Write Protection of GPIO2D0**
30:25		**Reserved**
24:20	RW	**Master 5**
19:15	RW	**Master 4**
14:10	RW	**Master 3**
9: 5	RW	**Master 2**

to next page

from previous page

4: 0	RW	**Master 1** 0x01 : Primary Processor 0x06 : Co-processor 0x13 : LPC

Offset: 32Ch **GPIO32C: GPIO_I/J/K/L New Write Command Source Register** **Init = 0**

Bit	R/W	Description
31	RW1S	**Write Protection of GPIO32C[31:24] until RstFull**
30	RW	**GPIOL Command Source Mode selection** 0: Legacy mode. Command Source works as GPIO090 and GPIO094 1: New mode. Command Source works by GPIO32C
29	RW	**GPIOL Write Permission for Others**
28	RW	**GPIOL Write Permission for MST5 GPIO2D0[24:20]**
27	RW	**GPIOL Write Permission for MST4 GPIO2D0[19:15]**
26	RW	**GPIOL Write Permission for MST3 GPIO2D0[14:10]**
25	RW	**GPIOL Write Permission for MST2 GPIO2D0[9: 5]**
24	RW	**GPIOL Write Permission for MST1 GPIO2D0[4: 0]**
23	RW1S	**Write Protection of GPIO32C[23:16] until RstFull**
22	RW	**GPIOK Command Source Mode selection** 0: Legacy mode. Command Source works as GPIO090 and GPIO094 1: New mode. Command Source works by GPIO32C
21	RW	**GPIOK Write Permission for Others**
20	RW	**GPIOK Write Permission for MST5 GPIO2D0[24:20]**
19	RW	**GPIOK Write Permission for MST4 GPIO2D0[19:15]**
18	RW	**GPIOK Write Permission for MST3 GPIO2D0[14:10]**
17	RW	**GPIOK Write Permission for MST2 GPIO2D0[9: 5]**
16	RW	**GPIOK Write Permission for MST1 GPIO2D0[4: 0]**
15	RW1S	**Write Protection of GPIO32C[15:08] until RstFull**
14	RW	**GPIOJ Command Source Mode selection** 0: Legacy mode. Command Source works as GPIO090 and GPIO094 1: New mode. Command Source works by GPIO32C
13	RW	**GPIOJ Write Permission for Others**
12	RW	**GPIOJ Write Permission for MST5 GPIO2D0[24:20]**
11	RW	**GPIOJ Write Permission for MST4 GPIO2D0[19:15]**
10	RW	**GPIOJ Write Permission for MST3 GPIO2D0[14:10]**
9	RW	**GPIOJ Write Permission for MST2 GPIO2D0[9: 5]**
8	RW	**GPIOJ Write Permission for MST1 GPIO2D0[4: 0]**
7	RW1S	**Write Protection of GPIO32C[07:00] until RstFull**
6	RW	**GPIOI Command Source Mode selection** 0: Legacy mode. Command Source works as GPIO090 and GPIO094 1: New mode. Command Source works by GPIO32C
5	RW	**GPIOI Write Permission for Others**
4	RW	**GPIOI Write Permission for MST5 GPIO2D0[24:20]**
3	RW	**GPIOI Write Permission for MST4 GPIO2D0[19:15]**
2	RW	**GPIOI Write Permission for MST3 GPIO2D0[14:10]**

to next page

from previous page

1	RW	**GPIOI Write Permission for MST2 GPIO2D0[9: 5]**
0	RW	**GPIOI Write Permission for MST1 GPIO2D0[4: 0]**

Offset: 330h **GPIO330: GPIO_I/J/K/L New Read Command Source Register** **Init = 0**

Bit	R/W	Description
31	RW1S	**Write Protection of GPIO330[31:24] until RstFull**
30	RW	**GPIOL Command Source Mode selection** 0: Legacy mode. Command Source works as GPIO090 and GPIO094 1: New mode. Command Source works by GPIO330
29	RW	**GPIOL Read Permission for Others**
28	RW	**GPIOL Read Permission for MST5 GPIO2D0[24:20]**
27	RW	**GPIOL Read Permission for MST4 GPIO2D0[19:15]**
26	RW	**GPIOL Read Permission for MST3 GPIO2D0[14:10]**
25	RW	**GPIOL Read Permission for MST2 GPIO2D0[9: 5]**
24	RW	**GPIOL Read Permission for MST1 GPIO2D0[4: 0]**
23	RW1S	**Write Protection of GPIO330[23:16] until RstFull**
22	RW	**GPIOK Command Source Mode selection** 0: Legacy mode. Command Source works as GPIO090 and GPIO094 1: New mode. Command Source works by GPIO330
21	RW	**GPIOK Read Permission for Others**
20	RW	**GPIOK Read Permission for MST5 GPIO2D0[24:20]**
19	RW	**GPIOK Read Permission for MST4 GPIO2D0[19:15]**
18	RW	**GPIOK Read Permission for MST3 GPIO2D0[14:10]**
17	RW	**GPIOK Read Permission for MST2 GPIO2D0[9: 5]**
16	RW	**GPIOK Read Permission for MST1 GPIO2D0[4: 0]**
15	RW1S	**Write Protection of GPIO330[15:08] until RstFull**
14	RW	**GPIOJ Command Source Mode selection** 0: Legacy mode. Command Source works as GPIO090 and GPIO094 1: New mode. Command Source works by GPIO330
13	RW	**GPIOJ Read Permission for Others**
12	RW	**GPIOJ Read Permission for MST5 GPIO2D0[24:20]**
11	RW	**GPIOJ Read Permission for MST4 GPIO2D0[19:15]**
10	RW	**GPIOJ Read Permission for MST3 GPIO2D0[14:10]**
9	RW	**GPIOJ Read Permission for MST2 GPIO2D0[9: 5]**
8	RW	**GPIOJ Read Permission for MST1 GPIO2D0[4: 0]**
7	RW1S	**Write Protection of GPIO330[07:00] until RstFull**
6	RW	**GPIOI Command Source Mode selection** 0: Legacy mode. Command Source works as GPIO090 and GPIO094 1: New mode. Command Source works by GPIO330
5	RW	**GPIOI Read Permission for Others**
4	RW	**GPIOI Read Permission for MST5 GPIO2D0[24:20]**
4	RW	**GPIOI Read Permission for MST4 GPIO2D0[19:15]**
2	RW	**GPIOI Read Permission for MST3 GPIO2D0[14:10]**
1	RW	**GPIOI Read Permission for MST2 GPIO2D0[9: 5]**
0	RW	**GPIOI Read Permission for MST1 GPIO2D0[4: 0]**

Offset: 338h	GPIO338: GPIO_I/J/K/L Blink Counter Selection #1	Init = 0
Bit	**R/W**	**Description**
31:24	RW	**GPIOL Blink Counter Selection #1**
23:16	RW	**GPIOK Blink Counter Selection #1**
15: 8	RW	**GPIOJ Blink Counter Selection #1**
7: 0	RW	**GPIOI Blink Counter Selection #1**

Offset: 33Ch	GPIO33C: GPIO_I/J/K/L Blink Counter Selection #2	Init = 0
Bit	**R/W**	**Description**
31:24	RW	**GPIOL Blink Counter Selection #2**
23:16	RW	**GPIOK Blink Counter Selection #2**
15: 8	RW	**GPIOJ Blink Counter Selection #2**
7: 0	RW	**GPIOI Blink Counter Selection #2**

Offset: 37Ch	GPIO37C: GPIO_M/N/O/P New Write Command Source Register	Init = 0
Bit	**R/W**	**Description**
31	RW1S	**Write Protection of GPIO37C[31:24] until RstFull**
30	RW	**GPIOP Command Source Mode selection** 0: Legacy mode. Command Source works as GPIO0E0 and GPIO0E4 1: New mode. Command Source works by GPIO37C
29	RW	**GPIOP Write Permission for Others**
28	RW	**GPIOP Write Permission for MST5 GPIO2D0[24:20]**
27	RW	**GPIOP Write Permission for MST4 GPIO2D0[19:15]**
26	RW	**GPIOP Write Permission for MST3 GPIO2D0[14:10]**
25	RW	**GPIOP Write Permission for MST2 GPIO2D0[9: 5]**
24	RW	**GPIOP Write Permission for MST1 GPIO2D0[4: 0]**
23	RW1S	**Write Protection of GPIO37C[23:16] until RstFull**
22	RW	**GPIOO Command Source Mode selection** 0: Legacy mode. Command Source works as GPIO0E0 and GPIO0E4 1: New mode. Command Source works by GPIO37C
21	RW	**GPIOO Write Permission for Others**
20	RW	**GPIOO Write Permission for MST5 GPIO2D0[24:20]**
19	RW	**GPIOO Write Permission for MST4 GPIO2D0[19:15]**
18	RW	**GPIOO Write Permission for MST3 GPIO2D0[14:10]**
17	RW	**GPIOO Write Permission for MST2 GPIO2D0[9: 5]**
16	RW	**GPIOO Write Permission for MST1 GPIO2D0[4: 0]**
15	RW1S	**Write Protection of GPIO37C[15:08] until RstFull**
14	RW	**GPION Command Source Mode selection** 0: Legacy mode. Command Source works as GPIO0E0 and GPIO0E4 1: New mode. Command Source works by GPIO37C
13	RW	**GPION Write Permission for Others**
12	RW	**GPION Write Permission for MST5 GPIO2D0[24:20]**
11	RW	**GPION Write Permission for MST4 GPIO2D0[19:15]**
10	RW	**GPION Write Permission for MST3 GPIO2D0[14:10]**

to next page

from previous page

9	RW	**GPION Write Permission for MST2 GPIO2D0[9: 5]**
8	RW	**GPION Write Permission for MST1 GPIO2D0[4: 0]**
7	RW1S	**Write Protection of GPIO37C[07:00] until RstFull**
6	RW	**GPIOM Command Source Mode selection** 0: Legacy mode. Command Source works as GPIO0E0 and GPIO0E4 1: New mode. Command Source works by GPIO37C
5	RW	**GPIOM Write Permission for Others**
4	RW	**GPIOM Write Permission for MST5 GPIO2D0[24:20]**
3	RW	**GPIOM Write Permission for MST4 GPIO2D0[19:15]**
2	RW	**GPIOM Write Permission for MST3 GPIO2D0[14:10]**
1	RW	**GPIOM Write Permission for MST2 GPIO2D0[9: 5]**
0	RW	**GPIOM Write Permission for MST1 GPIO2D0[4: 0]**

Offset: 380h **GPIO380: GPIO_M/N/O/P New Read Command Source Register** **Init = 0**

Bit	R/W	Description
31	RW1S	**Write Protection of GPIO380[31:24] until RstFull**
30	RW	**GPIOP Command Source Mode selection** 0: Legacy mode. Command Source works as GPIO0E0 and GPIO0E4 1: New mode. Command Source works by GPIO380
29	RW	**GPIOP Read Permission for Others**
28	RW	**GPIOP Read Permission for MST5 GPIO2D0[24:20]**
27	RW	**GPIOP Read Permission for MST4 GPIO2D0[19:15]**
26	RW	**GPIOP Read Permission for MST3 GPIO2D0[14:10]**
25	RW	**GPIOP Read Permission for MST2 GPIO2D0[9: 5]**
24	RW	**GPIOP Read Permission for MST1 GPIO2D0[4: 0]**
23	RW1S	**Write Protection of GPIO380[23:16] until RstFull**
22	RW	**GPIOO Command Source Mode selection** 0: Legacy mode. Command Source works as GPIO0E0 and GPIO0E4 1: New mode. Command Source works by GPIO380
21	RW	**GPIOO Read Permission for Others**
20	RW	**GPIOO Read Permission for MST5 GPIO2D0[24:20]**
19	RW	**GPIOO Read Permission for MST4 GPIO2D0[19:15]**
18	RW	**GPIOO Read Permission for MST3 GPIO2D0[14:10]**
17	RW	**GPIOO Read Permission for MST2 GPIO2D0[9: 5]**
16	RW	**GPIOO Read Permission for MST1 GPIO2D0[4: 0]**
15	RW1S	**Write Protection of GPIO380[15:08] until RstFull**
14	RW	**GPION Command Source Mode selection** 0: Legacy mode. Command Source works as GPIO0E0 and GPIO0E4 1: New mode. Command Source works by GPIO380
13	RW	**GPION Read Permission for Others**
12	RW	**GPION Read Permission for MST5 GPIO2D0[24:20]**
11	RW	**GPION Read Permission for MST4 GPIO2D0[19:15]**
10	RW	**GPION Read Permission for MST3 GPIO2D0[14:10]**
9	RW	**GPION Read Permission for MST2 GPIO2D0[9: 5]**

to next page

from previous page

8	RW	**GPION Read Permission for MST1 GPIO2D0[4: 0]**
7	RW1S	**Write Protection of GPIO380[07:00] until RstFull**
6	RW	**GPIOM Command Source Mode selection** 0: Legacy mode. Command Source works as GPIO0E0 and GPIO0E4 1: New mode. Command Source works by GPIO380
5	RW	**GPIOM Read Permission for Others**
4	RW	**GPIOM Read Permission for MST5 GPIO2D0[24:20]**
3	RW	**GPIOM Read Permission for MST4 GPIO2D0[19:15]**
2	RW	**GPIOM Read Permission for MST3 GPIO2D0[14:10]**
1	RW	**GPIOM Read Permission for MST2 GPIO2D0[9: 5]**
0	RW	**GPIOM Read Permission for MST1 GPIO2D0[4: 0]**

Offset: 388h **GPIO388: GPIO_M/N/O/P Blink Counter Selection #1** **Init = 0**

Bit	R/W	Description
31:24	RW	**GPIOP Blink Counter Selection #1**
23:16	RW	**GPIOO Blink Counter Selection #1**
15: 8	RW	**GPION Blink Counter Selection #1**
7: 0	RW	**GPIOM Blink Counter Selection #1**

Offset: 38Ch **GPIO38C: GPIO_M/N/O/P Blink Counter Selection #2** **Init = 0**

Bit	R/W	Description
31:24	RW	**GPIOP Blink Counter Selection #2**
23:16	RW	**GPIOO Blink Counter Selection #2**
15: 8	RW	**GPION Blink Counter Selection #2**
7: 0	RW	**GPIOM Blink Counter Selection #2**

Offset: 3CCh **GPIO3CC: GPIO_Q/R/S/T New Write Command Source Register** **Init = 0**

Bit	R/W	Description
31	RW1S	**Write Protection of GPIO3CC[31:24] until RstFull**
30	RW	**GPIT Command Source Mode selection** 0: Legacy mode. Command Source works as GPIO110 and GPIO114 1: New mode. Command Source works by GPIO3CC
29	RW	**GPIT Write Permission for Others**
28	RW	**GPIT Write Permission for MST5 GPIO2D0[24:20]**
27	RW	**GPIT Write Permission for MST4 GPIO2D0[19:15]**
26	RW	**GPIT Write Permission for MST3 GPIO2D0[14:10]**
25	RW	**GPIT Write Permission for MST2 GPIO2D0[9: 5]**
24	RW	**GPIT Write Permission for MST1 GPIO2D0[4: 0]**
23	RW1S	**Write Protection of GPIO3CC[23:16] until RstFull**
22	RW	**GPIOS Command Source Mode selection** 0: Legacy mode. Command Source works as GPIO110 and GPIO114 1: New mode. Command Source works by GPIO3CC
21	RW	**GPIOS Write Permission for Others**

to next page

from previous page

20	RW	**GPIOS Write Permission for MST5 GPIO2D0[24:20]**
19	RW	**GPIOS Write Permission for MST4 GPIO2D0[19:15]**
18	RW	**GPIOS Write Permission for MST3 GPIO2D0[14:10]**
17	RW	**GPIOS Write Permission for MST2 GPIO2D0[9: 5]**
16	RW	**GPIOS Write Permission for MST1 GPIO2D0[4: 0]**
15	RW1S	**Write Protection of GPIO3CC[15:08] until RstFull**
14	RW	**GPIOR Command Source Mode selection** 0: Legacy mode. Command Source works as GPIO110 and GPIO114 1: New mode. Command Source works by GPIO3CC
13	RW	**GPIOR Write Permission for Others**
12	RW	**GPIOR Write Permission for MST5 GPIO2D0[24:20]**
11	RW	**GPIOR Write Permission for MST4 GPIO2D0[19:15]**
10	RW	**GPIOR Write Permission for MST3 GPIO2D0[14:10]**
9	RW	**GPIOR Write Permission for MST2 GPIO2D0[9: 5]**
8	RW	**GPIOR Write Permission for MST1 GPIO2D0[4: 0]**
7	RW1S	**Write Protection of GPIO3CC[07:00] until RstFull**
6	RW	**GPIOQ Command Source Mode selection** 0: Legacy mode. Command Source works as GPIO110 and GPIO114 1: New mode. Command Source works by GPIO3CC
5	RW	**GPIOQ Write Permission for Others**
4	RW	**GPIOQ Write Permission for MST5 GPIO2D0[24:20]**
3	RW	**GPIOQ Write Permission for MST4 GPIO2D0[19:15]**
2	RW	**GPIOQ Write Permission for MST3 GPIO2D0[14:10]**
1	RW	**GPIOQ Write Permission for MST2 GPIO2D0[9: 5]**
0	RW	**GPIOQ Write Permission for MST1 GPIO2D0[4: 0]**

Offset: 3D0h **GPIO3D0: GPIO_Q/R/S/T New Read Command Source Register** **Init = 0**

Bit	R/W	Description
31	RW1S	**Write Protection of GPIO3D0[31:24] until RstFull**
30	RW	**GPIT Command Source Mode selection** 0: Legacy mode. Command Source works as GPIO110 and GPIO114 1: New mode. Command Source works by GPIO3D0
29	RW	**GPIT Read Permission for Others**
28	RW	**GPIT Read Permission for MST5 GPIO2D0[24:20]**
27	RW	**GPIT Read Permission for MST4 GPIO2D0[19:15]**
26	RW	**GPIT Read Permission for MST3 GPIO2D0[14:10]**
25	RW	**GPIT Read Permission for MST2 GPIO2D0[9: 5]**
24	RW	**GPIT Read Permission for MST1 GPIO2D0[4: 0]**
23	RW1S	**Write Protection of GPIO3D0[23:16] until RstFull**
22	RW	**GPIOS Command Source Mode selection** 0: Legacy mode. Command Source works as GPIO110 and GPIO114 1: New mode. Command Source works by GPIO3D0
21	RW	**GPIOS Read Permission for Others**
20	RW	**GPIOS Read Permission for MST5 GPIO2D0[24:20]**

to next page

from previous page

19	RW	**GPIOS Read Permission for MST4 GPIO2D0[19:15]**
18	RW	**GPIOS Read Permission for MST3 GPIO2D0[14:10]**
17	RW	**GPIOS Read Permission for MST2 GPIO2D0[9: 5]**
16	RW	**GPIOS Read Permission for MST1 GPIO2D0[4: 0]**
15	RW1S	**Write Protection of GPIO3D0[15:08] until RstFull**
14	RW	**GPIOR Command Source Mode selection** 0: Legacy mode. Command Source works as GPIO110 and GPIO114 1: New mode. Command Source works by GPIO3D0
13	RW	**GPIOR Read Permission for Others**
12	RW	**GPIOR Read Permission for MST5 GPIO2D0[24:20]**
11	RW	**GPIOR Read Permission for MST4 GPIO2D0[19:15]**
10	RW	**GPIOR Read Permission for MST3 GPIO2D0[14:10]**
9	RW	**GPIOR Read Permission for MST2 GPIO2D0[9: 5]**
8	RW	**GPIOR Read Permission for MST1 GPIO2D0[4: 0]**
7	RW1S	**Write Protection of GPIO3D0[07:00] until RstFull**
6	RW	**GPIOQ Command Source Mode selection** 0: Legacy mode. Command Source works as GPIO110 and GPIO114 1: New mode. Command Source works by GPIO3D0
5	RW	**GPIOQ Read Permission for Others**
4	RW	**GPIOQ Read Permission for MST5 GPIO2D0[24:20]**
3	RW	**GPIOQ Read Permission for MST4 GPIO2D0[19:15]**
2	RW	**GPIOQ Read Permission for MST3 GPIO2D0[14:10]**
1	RW	**GPIOQ Read Permission for MST2 GPIO2D0[9: 5]**
0	RW	**GPIOQ Read Permission for MST1 GPIO2D0[4: 0]**

Offset: 3D8h **GPIO3D8: GPIO_Q/R/S/T Blink Counter Selection #1** **Init = 0**

Bit	R/W	Description
31:24	R	**Reserved (GPIOT is input only)**
23:16	RW	**GPIOS Blink Counter Selection #1**
15: 8	RW	**GPIOR Blink Counter Selection #1**
7: 0	RW	**GPIOQ Blink Counter Selection #1**

Offset: 3DCh **GPIO3DC: GPIO_Q/R/S/T Blink Counter Selection #2** **Init = 0**

Bit	R/W	Description
31:24	R	**Reserved (GPIOT is input only)**
23:16	RW	**GPIOS Blink Counter Selection #2**
15: 8	RW	**GPIOR Blink Counter Selection #2**
7: 0	RW	**GPIOQ Blink Counter Selection #2**

Offset: 41Ch **GPIO41C: GPIO_U/V/W/X New Write Command Source Register** **Init = 0**

Bit	R/W	Description
31	RW1S	**Write Protection of GPIO41C[31:24] until RstFull**

to next page

from previous page

30	RW	**GPIOX Command Source Mode selection** 0: Legacy mode. Command Source works as GPIO140 and GPIO144 1: New mode. Command Source works by GPIO41C
29	RW	**GPIOX Write Permission for Others**
28	RW	**GPIOX Write Permission for MST5 GPIO2D0[24:20]**
27	RW	**GPIOX Write Permission for MST4 GPIO2D0[19:15]**
26	RW	**GPIOX Write Permission for MST3 GPIO2D0[14:10]**
25	RW	**GPIOX Write Permission for MST2 GPIO2D0[9: 5]**
24	RW	**GPIOX Write Permission for MST1 GPIO2D0[4: 0]**
23	RW1S	**Write Protection of GPIO41C[23:16] until RstFull**
22	RW	**GPIOW Command Source Mode selection** 0: Legacy mode. Command Source works as GPIO140 and GPIO144 1: New mode. Command Source works by GPIO41C
21	RW	**GPIOW Write Permission for Others**
20	RW	**GPIOW Write Permission for MST5 GPIO2D0[24:20]**
19	RW	**GPIOW Write Permission for MST4 GPIO2D0[19:15]**
18	RW	**GPIOW Write Permission for MST3 GPIO2D0[14:10]**
17	RW	**GPIOW Write Permission for MST2 GPIO2D0[9: 5]**
16	RW	**GPIOW Write Permission for MST1 GPIO2D0[4: 0]**
15	RW1S	**Write Protection of GPIO41C[15:08] until RstFull**
14	RW	**GPIOV Command Source Mode selection** 0: Legacy mode. Command Source works as GPIO140 and GPIO144 1: New mode. Command Source works by GPIO41C
13	RW	**GPIOV Write Permission for Others**
12	RW	**GPIOV Write Permission for MST5 GPIO2D0[24:20]**
11	RW	**GPIOV Write Permission for MST4 GPIO2D0[19:15]**
10	RW	**GPIOV Write Permission for MST3 GPIO2D0[14:10]**
9	RW	**GPIOV Write Permission for MST2 GPIO2D0[9: 5]**
8	RW	**GPIOV Write Permission for MST1 GPIO2D0[4: 0]**
7	RW1S	**Write Protection of GPIO41C[07:00] until RstFull**
6	RW	**GPIU Command Source Mode selection** 0: Legacy mode. Command Source works as GPIO140 and GPIO144 1: New mode. Command Source works by GPIO41C
5	RW	**GPIU Write Permission for Others**
4	RW	**GPIU Write Permission for MST5 GPIO2D0[24:20]**
3	RW	**GPIU Write Permission for MST4 GPIO2D0[19:15]**
2	RW	**GPIU Write Permission for MST3 GPIO2D0[14:10]**
1	RW	**GPIU Write Permission for MST2 GPIO2D0[9: 5]**
0	RW	**GPIU Write Permission for MST1 GPIO2D0[4: 0]**

Offset: 420h **GPIO420: GPIO_U/V/W/X New Read Command Source Register** **Init = 0**

Bit	R/W	Description
31	RW1S	**Write Protection of GPIO420[31:24] until RstFull**

to next page

from previous page

30	RW	**GPIOX Command Source Mode selection** 0: Legacy mode. Command Source works as GPIO140 and GPIO144 1: New mode. Command Source works by GPIO420
29	RW	**GPIOX Read Permission for Others**
28	RW	**GPIOX Read Permission for MST5 GPIO2D0[24:20]**
27	RW	**GPIOX Read Permission for MST4 GPIO2D0[19:15]**
26	RW	**GPIOX Read Permission for MST3 GPIO2D0[14:10]**
25	RW	**GPIOX Read Permission for MST2 GPIO2D0[9: 5]**
24	RW	**GPIOX Read Permission for MST1 GPIO2D0[4: 0]**
23	RW1S	**Write Protection of GPIO420[23:16] until RstFull**
22	RW	**GPIOW Command Source Mode selection** 0: Legacy mode. Command Source works as GPIO140 and GPIO144 1: New mode. Command Source works by GPIO420
21	RW	**GPIOW Read Permission for Others**
20	RW	**GPIOW Read Permission for MST5 GPIO2D0[24:20]**
19	RW	**GPIOW Read Permission for MST4 GPIO2D0[19:15]**
18	RW	**GPIOW Read Permission for MST3 GPIO2D0[14:10]**
17	RW	**GPIOW Read Permission for MST2 GPIO2D0[9: 5]**
16	RW	**GPIOW Read Permission for MST1 GPIO2D0[4: 0]**
15	RW1S	**Write Protection of GPIO420[15:08] until RstFull**
14	RW	**GPIOV Command Source Mode selection** 0: Legacy mode. Command Source works as GPIO140 and GPIO144 1: New mode. Command Source works by GPIO420
13	RW	**GPIOV Read Permission for Others**
12	RW	**GPIOV Read Permission for MST5 GPIO2D0[24:20]**
11	RW	**GPIOV Read Permission for MST4 GPIO2D0[19:15]**
10	RW	**GPIOV Read Permission for MST3 GPIO2D0[14:10]**
9	RW	**GPIOV Read Permission for MST2 GPIO2D0[9: 5]**
8	RW	**GPIOV Read Permission for MST1 GPIO2D0[4: 0]**
7	RW1S	**Write Protection of GPIO420[07:00] until RstFull**
6	RW	**GPIU Command Source Mode selection** 0: Legacy mode. Command Source works as GPIO140 and GPIO144 1: New mode. Command Source works by GPIO420
5	RW	**GPIU Read Permission for Others**
4	RW	**GPIU Read Permission for MST5 GPIO2D0[24:20]**
3	RW	**GPIU Read Permission for MST4 GPIO2D0[19:15]**
2	RW	**GPIU Read Permission for MST3 GPIO2D0[14:10]**
1	RW	**GPIU Read Permission for MST2 GPIO2D0[9: 5]**
0	RW	**GPIU Read Permission for MST1 GPIO2D0[4: 0]**

Offset: 428h **GPIO428: GPIO_U/V/W/X Blink Counter Selection #1** **Init = 0**

Bit	R/W	Description
31:24	RW	**GPIOX Blink Counter Selection #1**
23:16	RW	**GPIOW Blink Counter Selection #1**

to next page

from previous page

15: 8	RW	**GPIOV Blink Counter Selection #1**
7: 0	R	**Reserved (GPIOU is input only)**

Offset: 42Ch **GPIO42C: GPIO_U/V/W/X Blink Counter Selection #2** **Init = 0**

Bit	R/W	Description
31:24	RW	**GPIOX Blink Counter Selection #2**
23:16	RW	**GPIOW Blink Counter Selection #2**
15: 8	RW	**GPIOV Blink Counter Selection #2**
7: 0	RW	**Reserved (GPIOU is input only)**

Offset: 46Ch **GPIO46C: GPIO_Y/Z New Write Command Source Register** **Init = 0**

Bit	R/W	Description
31:16		**Reserved**
15	RW1S	**Write Protection of GPIO46C[15:08] until RstFull**
14	RW	**GPIOZ Command Source Mode selection** 0: Legacy mode. Command Source works as GPIO170 and GPIO174 1: New mode. Command Source works by GPIO46C
13	RW	**GPIOZ Write Permission for Others**
12	RW	**GPIOZ Write Permission for MST5 GPIO2D0[24:20]**
11	RW	**GPIOZ Write Permission for MST4 GPIO2D0[19:15]**
10	RW	**GPIOZ Write Permission for MST3 GPIO2D0[14:10]**
9	RW	**GPIOZ Write Permission for MST2 GPIO2D0[9: 5]**
8	RW	**GPIOZ Write Permission for MST1 GPIO2D0[4: 0]**
7	RW1S	**Write Protection of GPIO46C[07:00] until RstFull**
6	RW	**GPIOY Command Source Mode selection** 0: Legacy mode. Command Source works as GPIO170 and GPIO174 1: New mode. Command Source works by GPIO46C
5	RW	**GPIOY Write Permission for Others**
4	RW	**GPIOY Write Permission for MST5 GPIO2D0[24:20]**
3	RW	**GPIOY Write Permission for MST4 GPIO2D0[19:15]**
2	RW	**GPIOY Write Permission for MST3 GPIO2D0[14:10]**
1	RW	**GPIOY Write Permission for MST2 GPIO2D0[9: 5]**
0	RW	**GPIOY Write Permission for MST1 GPIO2D0[4: 0]**

Offset: 470h **GPIO470: GPIO_Y/Z New Read Command Source Register** **Init = 0**

Bit	R/W	Description
31:16		**Reserved**
15	RW1S	**Write Protection of GPIO470[15:08] until RstFull**
14	RW	**GPIOZ Command Source Mode selection** 0: Legacy mode. Command Source works as GPIO170 and GPIO174 1: New mode. Command Source works by GPIO470
13	RW	**GPIOZ Read Permission for Others**
12	RW	**GPIOZ Read Permission for MST5 GPIO2D0[24:20]**

to next page

from previous page

11	RW	**GPIOZ Read Permission for MST4 GPIO2D0[19:15]**
10	RW	**GPIOZ Read Permission for MST3 GPIO2D0[14:10]**
9	RW	**GPIOZ Read Permission for MST2 GPIO2D0[9: 5]**
8	RW	**GPIOZ Read Permission for MST1 GPIO2D0[4: 0]**
7	RW1S	**Write Protection of GPIO470[07:00] until RstFull**
6	RW	**GPIOY Command Source Mode selection** 0: Legacy mode. Command Source works as GPIO170 and GPIO174 1: New mode. Command Source works by GPIO470
5	RW	**GPIOY Read Permission for Others**
4	RW	**GPIOY Read Permission for MST5 GPIO2D0[24:20]**
3	RW	**GPIOY Read Permission for MST4 GPIO2D0[19:15]**
2	RW	**GPIOY Read Permission for MST3 GPIO2D0[14:10]**
1	RW	**GPIOY Read Permission for MST2 GPIO2D0[9: 5]**
0	RW	**GPIOY Read Permission for MST1 GPIO2D0[4: 0]**

Offset: 478h **GPIO478: GPIO_Y/Z Blink Counter Selection #1** **Init = 0**

Bit	R/W	Description
31:16	R	**Reserved**
15: 8	RW	**GPIOZ Blink Counter Selection #1**
7: 0	RW	**GPIOY Blink Counter Selection #1**

Offset: 47Ch **GPIO47C: GPIO_Y/Z Blink Counter Selection #2** **Init = 0**

Bit	R/W	Description
31:16	RW	**Reserved**
15: 8	RW	**GPIOZ Blink Counter Selection #2**
7: 0	RW	**GPIOY Blink Counter Selection #2**

Offset: 2ACh **GPIO2AC: GPIO Index Register** **Init = 0**

Bit	R/W	Description
31:20	RW	**Index Data**
19:16	RW	**Index Type** 0: Data Register 1: Direction 2: Interrupt 3: Debounce 4: Tolerance 5: Command Source 6: Input Mask 7: Reserved 8: New Write Command Source 9: New Read Command Source
15:13		**Reserved**
12	RW	**Index Command** 0: Write 1: Read

to next page

from previous page

11:8		**Reserved**
7 :0	RW	**Index Number** 00h: GPIOA0 01h: GPIOA1 02h: GPIOA2 ... CEh: GPIOZ6 CFh: GPIOZ7

Note :
Index Data:

	Bit 20	Bit 21	Bit 22	Bit 23	Bit 24
Data	Data	x	x	x	x
Direction	Direction	x	x	x	x
Interrupt	Interrupt Enable	Interrupt Type 0	Interrupt Type 1	Interrupt Type 2	Interrupt Status
Debounce	Debounce Setting 1	Debounce Setting 2	x	x	x
Tolerance	Tolerance	x	x	x	x
Command Source	Command Source #0	Command Source #1	x	x	x
Input Mask	Input Mask	x	x	x	x
New Write Cmd Src	Input Mask	x	x	x	x

The definition of Command Source 0 and 1 are as follows :

Command Source #1	Command Source #0	Source
0	0	ARM (default)
0	1	LPC
1	0	Coprocessor CPU
1	1	Reserved

Once source of one set is programmed, corresponding bit of register below only can be access by designated source.

- Data Value
- Direction
- Interrupt Enable
- Interrupt Sensitivity Type 0
- Interrupt Sensitivity Type 1
- Interrupt Sensitivity Type 2
- Reset Tolerant (if exists)
- Debounce Setting #1 (if exists)
- Debounce Setting #2 (if exists)

Features of each group of GPIO : :

Group	Interrupt	Reset Tolerant	Debounce
GPIOA	O	O	O
GPIOB	O	O	O
GPIOC	O	O	O
GPIOD	O	O	O
GPIOE	O	O	O
GPIOF	O	O	O
GPIOG	O	O	O
GPIOH	O	O	O
GPIOI	O	O	O
GPIOJ	O	O	O
GPIOK	O	O	O
GPIOL	O	O	O
GPIOM	O	O	O
GPION	O	O	O
GPIOO	O	O	O
GPIOP	O	O	O
GPIOQ	O	O	O
GPIOR	O	O	O
GPIOS	O	O	O
GPIT	O	O	O
GPIU	O	O	O
GPIOV	O	O	O
GPIOW	O	O	O
GPIOX	O	O	O
GPIOY	O	O	O
GPIOZ	O	O	O

Relation between GPIO Registers and Reset:

Register	SRST# or Watchdog Reset Full	Watchdog Reset SOC
GPIO000	O	O (If GPIO01C is set)
GPIO004	O	O (If GPIO01C is set)
GPIO008	O	O
GPIO00C	O	O
GPIO010	O	O
GPIO014	O	O
GPIO018	O	O
GPIO01C	O	X
GPIO020	O	O (If GPIO03C is set)
GPIO024	O	O (If GPIO03C is set)
GPIO028	O	O

to next page

from previous page

GPIO02C	O	O
GPIO030	O	O
GPIO034	O	O
GPIO038	O	O
GPIO03C	O	X
GPIO040	O	O
GPIO044	O	O
GPIO048	O	O
GPIO04C	O	O
GPIO050	O	O
GPIO054	O	O
GPIO058	O	O
GPIO060	O	O
GPIO064	O	O
GPIO068	O	O
GPIO06C	O	O
GPIO070	O	O (If GPIO0AC is set)
GPIO074	O	O (If GPIO0AC is set)
GPIO078	O	O (If GPIO0FC is set)
GPIO07C	O	O (If GPIO0FC is set)
GPIO080	O	O (If GPIO12C is set)
GPIO084	O	O (If GPIO12C is set)
GPIO088	O	O (If GPIO15C is set)
GPIO08C	O	O (If GPIO15C is set)
GPIO090	O	O
GPIO094	O	O
GPIO098	O	O
GPIO09C	O	O
GPIO0A0	O	O
GPIO0A4	O	O
GPIO0A8	O	O
GPIO0AC	O	X
GPIO0B0	O	O
GPIO0B4	O	O
GPIO0B8	O	O
GPIO0C0	O	O (If GPIO01C is set)
GPIO0C4	O	O (If GPIO03C is set)
GPIO0C8	O	O (If GPIO0AC is set)
GPIO0CC	O	O (If GPIO0FC is set)
GPIO0D0	O	O (If GPIO12C is set)
GPIO0D4	O	O (If GPIO15C is set)
GPIO0D8	O	O (If GPIO18C is set)
GPIO0E0	O	O
GPIO0E4	O	O
GPIO0E8	O	O

to next page

from previous page

GPIO0EC	O	O
GPIO0F0	O	O
GPIO0F4	O	O
GPIO0F8	O	O
GPIO0FC	O	X
GPIO100	O	O
GPIO104	O	O
GPIO108	O	O
GPIO110	O	O
GPIO114	O	O
GPIO118	O	O
GPIO11C	O	O
GPIO120	O	O
GPIO124	O	O
GPIO128	O	O
GPIO12C	O	X
GPIO130	O	O
GPIO134	O	O
GPIO138	O	O
GPIO140	O	O
GPIO144	O	O
GPIO148	O	O
GPIO14C	O	O
GPIO150	O	O
GPIO154	O	O
GPIO158	O	O
GPIO15C	O	X
GPIO160	O	O
GPIO164	O	O
GPIO168	O	O
GPIO170	O	O
GPIO174	O	O
GPIO178	O	O
GPIO17C	O	O
GPIO180	O	O
GPIO184	O	O
GPIO188	O	O
GPIO18C	O	X
GPIO190	O	O
GPIO194	O	O
GPIO198	O	O
GPIO1D0	O	O
GPIO1D4	O	O
GPIO1E0	O	O (If GPIO18C is set)
GPIO1E4	O	O (If GPIO18C is set)

Group	Input	Output
GPIOA	O	O
GPIOB	O	O
GPIOC	O	O
GPIOD	O	O
GPIOE	O	O
GPIOF	O	O
GPIOG	O	O
GPIOH	O	O
GPIOI	O	O
GPIOJ	O	O
GPIOK	O	O
GPIOL	O	O
GPIOM	O	O
GPION	O	O
GPIOO	O	O
GPIOP	O	O
GPIOQ	O	O
GPIOR	O	O
GPIOS	O	O
GPIT	O	X
GPIU	O	X
GPIOV	O	O
GPIOW	O	O
GPIOX	O	O
GPIOY	O	O
GPIOZ	O	O

41.3.2 1.8V Parallel GPIO

Offset: 800h	GPIO800: GPIO18_A/B/C/D Data Value Register	Init = 0
Bit	**R/W**	**Description**
31:24	RW	**Port GPIOD[7:0] data register**
23:16	RW	**Port GPIOC[7:0] data register**
15:8	RW	**Port GPIOB[7:0] data register**
7 :0	RW	**Port GPIOA[7:0] data register**

Offset: 804h	GPIO804: GPIO18_A/B/C/D Direction Register	Init = 0
Bit	**R/W**	**Description**
31:24	RW	**Port GPIOD[7:0] direction control** 0: Select input mode 1: Select output mode
23:16	RW	**Port GPIOC[7:0] direction control** 0: Select input mode 1: Select output mode
15:8	RW	**Port GPIOB[7:0] direction control** 0: Select input mode 1: Select output mode
7 :0	RW	**Port GPIOA[7:0] direction control** 0: Select input mode 1: Select output mode

Offset: 808h	GPIO808: GPIO18_A/B/C/D Interrupt Enable Register	Init = 0
Bit	**R/W**	**Description**
31:24	RW	**Port GPIOD[7:0] interrupt enable** 0: Disable interrupt 1: Enable interrupt
23:16	RW	**Port GPIOC[7:0] interrupt enable** 0: Disable interrupt 1: Enable interrupt
15:8	RW	**Port GPIOB[7:0] interrupt enable** 0: Disable interrupt 1: Enable interrupt
7 :0	RW	**Port GPIOA[7:0] interrupt enable** 0: Disable interrupt 1: Enable interrupt

Offset: 80Ch	GPIO80C: GPIO18_A/B/C/D Interrupt Sensitivity Type 0 Register	Init = 0
Bit	**R/W**	**Description**
31:24	RW	**Port GPIOD[7:0] interrupt sensitivity type 0 selection** 0: Select falling-edge or level-low trigger mode 1: Select rising-edge or level-high trigger mode
23:16	RW	**Port GPIOC[7:0] interrupt sensitivity type 0 selection** 0: Select falling-edge or level-low trigger mode 1: Select rising-edge or level-high trigger mode
15:8	RW	**Port GPIOB[7:0] interrupt sensitivity type 0 selection** 0: Select falling-edge or level-low trigger mode 1: Select rising-edge or level-high trigger mode
7 :0	RW	**Port GPIOA[7:0] interrupt sensitivity type 0 selection** 0: Select falling-edge or level-low trigger mode 1: Select rising-edge or level-high trigger mode

Offset: 810h	GPIO810: GPIO18_A/B/C/D Interrupt Sensitivity Type 1 Register	Init = 0
Bit	**R/W**	**Description**
31:24	RW	**Port GPIOD[7:0] interrupt sensitivity type 1 selection** 0: Select edge trigger mode 1: Select level trigger mode
23:16	RW	**Port GPIOC[7:0] interrupt sensitivity type 1 selection** 0: Select edge trigger mode 1: Select level trigger mode
15:8	RW	**Port GPIOB[7:0] interrupt sensitivity type 1 selection** 0: Select edge trigger mode 1: Select level trigger mode
7 :0	RW	**Port GPIOA[7:0] interrupt sensitivity type 1 selection** 0: Select edge trigger mode 1: Select level trigger mode

Offset: 814h	GPIO814: GPIO18_A/B/C/D Interrupt Sensitivity Type 2 Register	Init = 0
Bit	**R/W**	**Description**
31:24	RW	**Port GPIOD[7:0] interrupt sensitivity type 2 selection** 0: Select edge or level trigger mode 1: Select dual-edge trigger mode
23:16	RW	**Port GPIOC[7:0] interrupt sensitivity type 2 selection** 0: Select edge or level trigger mode 1: Select dual-edge trigger mode
15:8	RW	**Port GPIOB[7:0] interrupt sensitivity type 2 selection** 0: Select edge or level trigger mode 1: Select dual-edge trigger mode
7 :0	RW	**Port GPIOA[7:0] interrupt sensitivity type 2 selection** 0: Select edge or level trigger mode 1: Select dual-edge trigger mode

Offset: 818h	GPIO818: GPIO18_A/B/C/D Interrupt Status Register	Init = 0
Bit	**R/W**	**Description**
31:24	RW	**Port GPIOD[7:0] interrupt status register** Read 0: No interrupt pending Read 1: interrupt pending Write 0: No operation Write 1: Clear interrupt status flag
23:16	RW	**Port GPIOC[7:0] interrupt status register** Read 0: No interrupt pending Read 1: interrupt pending Write 0: No operation Write 1: Clear interrupt status flag
15:8	RW	**Port GPIOB[7:0] interrupt status register** Read 0: No interrupt pending Read 1: interrupt pending Write 0: No operation Write 1: Clear interrupt status flag

to next page

from previous page

7 :0	RW	**Port GPIOA[7:0] interrupt status register** Read 0: No interrupt pending Read 1: interrupt pending Write 0: No operation Write 1: Clear interrupt status flag

Offset: 81Ch **GPIO81C: GPIO18_A/B/C/D Reset Tolerant Register** **Init = 0**

Bit	R/W	Description
31:24	RW	**Port GPIOD[7:0] WDT reset tolerance enable** 0: GPIO00 and GPIO04 registers will be reset by WDT reset 1: GPIO00 and GPIO04 registers will not be reset by WDT reset Each GPIO pin can be individually programmed to be WDT reset tolerant or not.
23:16	RW	**Port GPIOC[7:0] WDT reset tolerance enable** 0: GPIO00 and GPIO04 registers will be reset by WDT reset 1: GPIO00 and GPIO04 registers will not be reset by WDT reset Each GPIO pin can be individually programmed to be WDT reset tolerant or not.
15:8	RW	**Port GPIOB[7:0] WDT reset tolerance enable** 0: GPIO00 and GPIO04 registers will be reset by WDT reset 1: GPIO00 and GPIO04 registers will not be reset by WDT reset Each GPIO pin can be individually programmed to be WDT reset tolerant or not.
7 :0	RW	**Port GPIOA[7:0] WDT reset tolerance enable** 0: GPIO00 and GPIO04 registers will be reset by WDT reset 1: GPIO00 and GPIO04 registers will not be reset by WDT reset Each GPIO pin can be individually programmed to be WDT reset tolerant or not.

Offset: 820h **GPIO820: GPIO18_E Data Value Register** **Init = 0**

Bit	R/W	Description
31:4	R	**Reserved**
3 :0	RW	**Port GPIOE[7:0] data register**

Offset: 824h **GPIO824: GPIO18_E Direction Register** **Init = 0**

Bit	R/W	Description
31:4	R	**Reserved**
3 :0	RW	**Port GPIOE[7:0] direction control** 0: Select input mode 1: Select output mode

Offset: 828h **GPIO828: GPIO18_E Interrupt Enable Register** **Init = 0**

Bit	R/W	Description
31:4	R	**Reserved**
3 :0	RW	**Port GPIOE[7:0] interrupt enable** 0: Disable interrupt 1: Enable interrupt

Offset: 82Ch	GPIO82C: GPIO18_E Interrupt Sensitivity Type 0 Register	Init = 0
Bit	**R/W**	**Description**
31:4	R	**Reserved**
3 :0	RW	**Port GPIOE[7:0] interrupt sensitivity type 0 selection** 0: Select falling-edge or level-low trigger mode 1: Select rising-edge or level-high trigger mode

Offset: 830h	GPIO830: GPIO18_E Interrupt Sensitivity Type 1 Register	Init = 0
Bit	**R/W**	**Description**
31:4	R	**Reserved**
3 :0	RW	**Port GPIOE[7:0] interrupt sensitivity type 1 selection** 0: Select edge trigger mode 1: Select level trigger mode

Offset: 834h	GPIO834: GPIO18_E Interrupt Sensitivity Type 2 Register	Init = 0
Bit	**R/W**	**Description**
31:4	R	**Reserved**
3 :0	RW	**Port GPIOE[7:0] interrupt sensitivity type 2 selection** 0: Select edge or level trigger mode 1: Select dual-edge trigger mode

Offset: 838h	GPIO838: GPIO18_E Interrupt Status Register	Init = 0
Bit	**R/W**	**Description**
31:4	R	**Reserved**
3 :0	RW	**Port GPIOE[3:0] interrupt status register** Read 0: No interrupt pending Read 1: interrupt pending Write 0: No operation Write 1: Clear interrupt status flag

Offset: 83Ch	GPIO83C: GPIO18_E Reset Tolerant Register	Init = 0
Bit	**R/W**	**Description**
31:4	R	**Reserved**
3 :0	RW	**Port GPIOE[3:0] WDT reset tolerance enable** 0: GPIO20 and GPIO24 registers will be reset by WDT reset 1: GPIO20 and GPIO24 registers will not be reset by WDT reset Each GPIO pin can be individually programmed to be WDT reset tolerant or not.

Offset: 840h	GPIO040: GPIO18_A/B/C/D Debounce Setting Register #1	Init = 0
Bit	**R/W**	**Description**
31:24	RW	**Port GPIOD[7:0] debounce setting register #1**
23:16	RW	**Port GPIOC[7:0] debounce setting register #1**
15:8	RW	**Port GPIOB[7:0] debounce setting register #1**
7 :0	RW	**Port GPIOA[7:0] debounce setting register #1**

Offset: 844h	GPIO844: GPIO18_A/B/C/D Debounce Setting Register #2	Init = 0
Bit	**R/W**	**Description**
31:4	R	**Reserved**
3 :0	RW	**Port GPIOA[7:0] debounce setting register #2**

Offset: 848h	GPIO848: GPIO18_E Debounce Setting Register #1	Init = 0
Bit	**R/W**	**Description**
31:4	R	**Reserved**
3 :0	RW	**Port GPIOE[7:0] debounce setting register #1**

Offset: 84Ch	GPIO84C: GPIO18_E/F/G/H Debounce Setting Register #2	Init = 0
Bit	**R/W**	**Description**
31:4	R	**Reserved**
3 :0	RW	**Port GPIOE[3:0] debounce setting register #2**

The definition of interrupt trigger mode registers:
GPIO80C ~ GPIO814, GPIO82C ~ GPIO834 are as follows :

Type 2	Type 1	Type 0	Interrupt Trigger Mode
(14/34h)	(10/30h)	(0C/2Ch)	
0	0	0	falling-edge trigger mode
0	0	1	rising-edge trigger mode
0	1	0	level-low trigger mode
0	1	1	level-high trigger mode
1	x	x	dual-edge trigger mode

The definition of debounce setting registers GPIO040 ~ GPIO04C are as follows :

Debounce Setting #2	Debounce Setting #1	Function
0	0	No Debounce
1	0	Select GPIO850 as debounce timer
0	1	Select GPIO854 as debounce timer
1	1	Select GPIO858 as debounce timer

Offset: 850h	GPIO850: Debounce Timer Setting Register #1	Init = 0
Offset: 854h	GPIO854: Debounce Timer Setting Register #2	Init = 0
Offset: 858h	GPIO858: Debounce Timer Setting Register #3	Init = 0
Bit	**Attr.**	**Description**
31:24		**Reserved**

to next page

from previous page

23:0	RW	**Debounce Timer Value** This register defines the timer period for GPIO input sampling. The sampling timer period is : Debounce time = PCLK cycle time * Debounce timer value Latency teim = Debounce time * 2

Offset: 860h **GPIO860: GPIO18_A/B/C/D Command Source 0** **Init = 0**

Bit	R/W	Description
31:25		**Reserved**
24	RW	**Port GPIOD[7:0] Command Source 0**
23:17		**Reserved**
16	RW	**Port GPIOC[7:0] Command Source 0**
15: 9		**Reserved**
8	RW	**Port GPIOB[7:0] Command Source 0**
7: 1		**Reserved**
0	RW	**Port GPIOA[7:0] Command Source 0**

Note :

Command Source #1	Command Source #0	Source
0	0	ARM (default)
0	1	LPC
1	0	Coprocessor CPU
1	1	Reserved

Offset: 864h **GPIO864: GPIO18_A/B/C/D Command Source 1** **Init = 0**

Bit	R/W	Description
31:25		**Reserved**
24	RW	**Port GPIOD[7:0] Command Source 1**
23:17		**Reserved**
16	RW	**Port GPIOC[7:0] Command Source 1**
15: 9		**Reserved**
8	RW	**Port GPIOB[7:0] Command Source 1**
7: 1		**Reserved**
0	RW	**Port GPIOA[7:0] Command Source 1**

Note :

Command Source #1	Command Source #0	Source
0	0	ARM (default)
0	1	LPC
1	0	Coprocessor CPU
1	1	Reserved

Offset: 868h	GPIO868: GPIO18_E Command Source 0	Init = 0

Bit	R/W	Description
31:1	R	**Reserved**
0	RW	**Port GPIOE[3:0] Command Source 0**

Note :

Command Source #1	Command Source #0	Source
0	0	ARM (default)
0	1	LPC
1	0	Coprocessor CPU
1	1	Reserved

Offset: 86Ch	GPIO86C: GPIO18_E Command Source 1	Init = 0

Bit	R/W	Description
31:1	R	**Reserved**
0	RW	**Port GPIOE[3:0] Command Source 1**

Note :

Command Source #1	Command Source #0	Source
0	0	ARM (default)
0	1	LPC
1	0	Coprocessor CPU
1	1	Reserved

Offset: 8C0h	GPIO8C0: GPIO18_A/B/C/D Data Read Register	Init = 0

Bit	R/W	Description
31:0	R	**Data written to GPIO800.(GPIOD/GPIOC/GPIOB/GPIOA)**

Offset: 8C4h	GPIO8C4: GPIO18_E Data Read Register	Init = 0

Bit	R/W	Description
31:4	R	**Reserved**
3:0	R	**Data written to GPIO820.(GPIOE)**

Offset: 9D0h	GPIO9D0: GPIO18_A/B/C/D Input Mask Register	Init = 0

Bit	R/W	Description
31:24	RW	**Port GPIOD[7:0] input mask** 0: Read from GPIO080 will be updated. 1: Read from GPIO080 will not be updated.
23:16	RW	**Port GPIOC[7:0] input mask** 0: Read from GPIO080 will be updated. 1: Read from GPIO080 will not be updated.
15:8	RW	**Port GPIOB[7:0] input mask** 0: Read from GPIO080 will be updated. 1: Read from GPIO080 will not be updated.
7 :0	RW	**Port GPIOA[7:0] input mask** 0: Read from GPIO080 will be updated. 1: Read from GPIO080 will not be updated.

Offset: 9D4h	GPIO9D4: GPIO18_E Input Mask Register	Init = 0

Bit	R/W	Description
31:4	R	**Reserved**
3 :0	RW	**Port GPIOE[3:0] input mask** 0: Read from GPIO080 will be updated. 1: Read from GPIO080 will not be updated.

Offset: AACh	GPIOAAC: GPIO18 Index Register	Init = 0

Bit	R/W	Description
31:25		**Reserved**
24:20	RW	**Index Data**
18:16	RW	**Index Type** 0: Data Register 1: Direction 2: Interrupt 3: Debounce 4: Tolerance 5: Command Source 6: Input Mask
15:13		**Reserved**
12	RW	**Index Command** 0: Write 1: Read
11:8		**Reserved**
7 :0	RW	**Index Number** 00h: GPIO18A0 01h: GPIO18A1 02h: GPIO18A2 ... 26h: GPIO18E6 27h: GPIO18E7

Note :
Index Data:

	Bit 20	Bit 21	Bit 22	Bit 23	Bit 24
Data	Data	x	x	x	x
Direction	Direction	x	x	x	x
Interrupt	Interrupt Enable	Interrupt Type 0	Interrupt Type 1	Interrupt Type 2	Interrupt Status
Debounce	Debounce Setting 1	Debounce Setting 2	x	x	x
Tolerance	Tolerance	x	x	x	x
Command Source	Command Source #0	Command Source #1	x	x	x
Input Mask	Input Mask	x	x	x	x

41.3.3 Serial GPIO Master 1

Offset: 500h	GPIO500: Serial GPIO_A/B/C/D 1 Data Value Register	Init = 0
Bit	**R/W**	**Description**
31:24	RW	**Port Serial GPIOD[7:0] data register**
23:16	RW	**Port Serial GPIOC[7:0] data register**
15:8	RW	**Port Serial GPIOB[7:0] data register**
7 :0	RW	**Port Serial GPIOA[7:0] data register**

Offset: 504h	GPIO504: Serial GPIO_A/B/C/D 1 Interrupt Enable Register	Init = 0
Bit	**R/W**	**Description**
31:24	RW	**Port Serial GPIOD[7:0] interrupt enable** 0: Disable interrupt 1: Enable interrupt
23:16	RW	**Port Serial GPIOC[7:0] interrupt enable** 0: Disable interrupt 1: Enable interrupt
15:8	RW	**Port Serial GPIOB[7:0] interrupt enable** 0: Disable interrupt 1: Enable interrupt
7 :0	RW	**Port Serial GPIOA[7:0] interrupt enable** 0: Disable interrupt 1: Enable interrupt

Offset: 508h	GPIO508: Serial GPIO_A/B/C/D 1 Interrupt Sensitivity Type 0 Register	Init = 0
Bit	**R/W**	**Description**
31:24	RW	**Port Serial GPIOD[7:0] interrupt sensitivity type 0 selection** 0: Select falling-edge or level-low trigger mode 1: Select rising-edge or level-high trigger mode
23:16	RW	**Port Serial GPIOC[7:0] interrupt sensitivity type 0 selection** 0: Select falling-edge or level-low trigger mode 1: Select rising-edge or level-high trigger mode
15:8	RW	**Port Serial GPIOB[7:0] interrupt sensitivity type 0 selection** 0: Select falling-edge or level-low trigger mode 1: Select rising-edge or level-high trigger mode
7 :0	RW	**Port Serial GPIOA[7:0] interrupt sensitivity type 0 selection** 0: Select falling-edge or level-low trigger mode 1: Select rising-edge or level-high trigger mode

Offset: 50Ch	GPIO50C: Serial GPIO_A/B/C/D 1 Interrupt Sensitivity Type 1 Register	Init = 0
Bit	**R/W**	**Description**
31:24	RW	**Port Serial GPIOD[7:0] interrupt sensitivity type 1 selection** 0: Select edge trigger mode 1: Select level trigger mode
23:16	RW	**Port Serial GPIOC[7:0] interrupt sensitivity type 1 selection** 0: Select edge trigger mode 1: Select level trigger mode

to next page

from previous page

15:8	RW	**Port Serial GPIOB[7:0] interrupt sensitivity type 1 selection** 0: Select edge trigger mode 1: Select level trigger mode
7 :0	RW	**Port Serial GPIOA[7:0] interrupt sensitivity type 1 selection** 0: Select edge trigger mode 1: Select level trigger mode

Offset: 510h GPIO510: Serial GPIO_A/B/C/D 1 Interrupt Sensitivity Type 2 Register Init = 0

Bit	R/W	Description
31:24	RW	**Port Serial GPIOD[7:0] interrupt sensitivity type 2 selection** 0: Select edge or level trigger mode 1: Select dual-edge trigger mode
23:16	RW	**Port Serial GPIOC[7:0] interrupt sensitivity type 2 selection** 0: Select edge or level trigger mode 1: Select dual-edge trigger mode
15:8	RW	**Port Serial GPIOB[7:0] interrupt sensitivity type 2 selection** 0: Select edge or level trigger mode 1: Select dual-edge trigger mode
7 :0	RW	**Port Serial GPIOA[7:0] interrupt sensitivity type 2 selection** 0: Select edge or level trigger mode 1: Select dual-edge trigger mode

Offset: 514h GPIO514: Serial GPIO_A/B/C/D 1 Interrupt Status Register Init = 0

Bit	R/W	Description
31:24	RW	**Port Serial GPIOD[7:0] interrupt status register** Read 0: No interrupt pending Read 1: interrupt pending Write 0: No operation Write 1: Clear interrupt status flag
23:16	RW	**Port Serial GPIOC[7:0] interrupt status register** Read 0: No interrupt pending Read 1: interrupt pending Write 0: No operation Write 1: Clear interrupt status flag
15:8	RW	**Port Serial GPIOB[7:0] interrupt status register** Read 0: No interrupt pending Read 1: interrupt pending Write 0: No operation Write 1: Clear interrupt status flag
7 :0	RW	**Port Serial GPIOA[7:0] interrupt status register** Read 0: No interrupt pending Read 1: interrupt pending Write 0: No operation Write 1: Clear interrupt status flag

Offset: 518h	GPIO518: Serial GPIO_A/B/C/D 1 Reset Tolerant Register	Init = 0
Bit	**R/W**	**Description**
31:24	RW	**Port Serial GPIOD[7:0] WDT reset tolerance enable** 0: GPIO500 registers will be reset by WDT reset 1: GPIO500 registers will not be reset by WDT reset Each Serial GPIO pin can be individually programmed to be WDT reset tolerant or not.
23:16	RW	**Port Serial GPIOC[7:0] WDT reset tolerance enable** 0: GPIO500 registers will be reset by WDT reset 1: GPIO500 registers will not be reset by WDT reset Each GPIO pin can be individually programmed to be WDT reset tolerant or not.
15:8	RW	**Port Serial GPIOB[7:0] WDT reset tolerance enable** 0: GPIO500 registers will be reset by WDT reset 1: GPIO500 registers will not be reset by WDT reset Each Serial GPIO pin can be individually programmed to be WDT reset tolerant or not.
7 :0	RW	**Port Serial GPIOA[7:0] WDT reset tolerance enable** 0: GPIO500 registers will be reset by WDT reset 1: GPIO500 registers will not be reset by WDT reset Each Serial GPIO pin can be individually programmed to be WDT reset tolerant or not.

Offset: 51Ch	GPIO51C: Serial GPIO_E/F/G/H 1 Data Value Register	Init = 0
Bit	**R/W**	**Description**
31:24	RW	**Port Serial GPIOH[7:0] data register**
23:16	RW	**Port Serial GPIOG[7:0] data register**
15:8	RW	**Port Serial GPIOF[7:0] data register**
7 :0	RW	**Port Serial GPIOE[7:0] data register**

Offset: 520h	GPIO520: Serial GPIO_E/F/G/H 1 Interrupt Enable Register	Init = 0
Bit	**R/W**	**Description**
31:24	RW	**Port Serial GPIOH[7:0] interrupt enable** 0: Disable interrupt 1: Enable interrupt
23:16	RW	**Port Serial GPIOG[7:0] interrupt enable** 0: Disable interrupt 1: Enable interrupt
15:8	RW	**Port Serial GPIOF[7:0] interrupt enable** 0: Disable interrupt 1: Enable interrupt
7 :0	RW	**Port Serial GPIOE[7:0] interrupt enable** 0: Disable interrupt 1: Enable interrupt

Offset: 524h	GPIO524: Serial GPIO_E/F/G/H 1 Interrupt Sensitivity Type 0 Register	Init = 0
Bit	**R/W**	**Description**
31:24	RW	**Port Serial GPIOH[7:0] interrupt sensitivity type 0 selection** 0: Select falling-edge or level-low trigger mode 1: Select rising-edge or level-high trigger mode
23:16	RW	**Port Serial GPIOG[7:0] interrupt sensitivity type 0 selection** 0: Select falling-edge or level-low trigger mode 1: Select rising-edge or level-high trigger mode

to next page

from previous page

15:8	RW	**Port Serial GPIOF[7:0] interrupt sensitivity type 0 selection** 0: Select falling-edge or level-low trigger mode 1: Select rising-edge or level-high trigger mode
7 :0	RW	**Port Serial GPIOE[7:0] interrupt sensitivity type 0 selection** 0: Select falling-edge or level-low trigger mode 1: Select rising-edge or level-high trigger mode

Offset: 528h GPIO528: Serial GPIO_E/F/G/H 1 Interrupt Sensitivity Type 1 Register Init = 0

Bit	R/W	Description
31:24	RW	**Port Serial GPIOH[7:0] interrupt sensitivity type 1 selection** 0: Select edge trigger mode 1: Select level trigger mode
23:16	RW	**Port Serial GPIOG[7:0] interrupt sensitivity type 1 selection** 0: Select edge trigger mode 1: Select level trigger mode
15:8	RW	**Port Serial GPIOF[7:0] interrupt sensitivity type 1 selection** 0: Select edge trigger mode 1: Select level trigger mode
7 :0	RW	**Port Serial GPIOE[7:0] interrupt sensitivity type 1 selection** 0: Select edge trigger mode 1: Select level trigger mode

Offset: 52Ch GPIO52C: Serial GPIO_E/F/G/H 1 Interrupt Sensitivity Type 2 Register Init = 0

Bit	R/W	Description
31:24	RW	**Port Serial GPIOH[7:0] interrupt sensitivity type 2 selection** 0: Select edge or level trigger mode 1: Select dual-edge trigger mode
23:16	RW	**Port Serial GPIOG[7:0] interrupt sensitivity type 2 selection** 0: Select edge or level trigger mode 1: Select dual-edge trigger mode
15:8	RW	**Port Serial GPIOF[7:0] interrupt sensitivity type 2 selection** 0: Select edge or level trigger mode 1: Select dual-edge trigger mode
7 :0	RW	**Port Serial GPIOE[7:0] interrupt sensitivity type 2 selection** 0: Select edge or level trigger mode 1: Select dual-edge trigger mode

Offset: 530h GPIO530: Serial GPIO_E/F/G/H 1 Interrupt Status Register Init = 0

Bit	R/W	Description
31:24	RW	**Port Serial GPIOH[7:0] interrupt status register** Read 0: No interrupt pending Read 1: interrupt pending Write 0: No operation Write 1: Clear interrupt status flag

to next page

from previous page

23:16	RW	**Port Serial GPIOG[7:0] interrupt status register** Read 0: No interrupt pending Read 1: interrupt pending Write 0: No operation Write 1: Clear interrupt status flag
15:8	RW	**Port Serial GPIOF[7:0] interrupt status register** Read 0: No interrupt pending Read 1: interrupt pending Write 0: No operation Write 1: Clear interrupt status flag
7 :0	RW	**Port Serial GPIOE[7:0] interrupt status register** Read 0: No interrupt pending Read 1: interrupt pending Write 0: No operation Write 1: Clear interrupt status flag

Offset: 534h **GPIO534: Serial GPIO_E/F/G/H 1 Reset Tolerant Register** **Init = 0**

Bit	R/W	Description
31:24	RW	**Port Serial GPIOH[7:0] WDT reset tolerance enable** 0: GPIO51C registers will be reset by WDT reset 1: GPIO51C registers will not be reset by WDT reset Each Serial GPIO pin can be individually programmed to be WDT reset tolerant or not.
23:16	RW	**Port Serial GPIOG[7:0] WDT reset tolerance enable** 0: GPIO51C registers will be reset by WDT reset 1: GPIO51C registers will not be reset by WDT reset Each Serial GPIO pin can be individually programmed to be WDT reset tolerant or not.
15:8	RW	**Port Serial GPIOF[7:0] WDT reset tolerance enable** 0: GPIO51C registers will be reset by WDT reset 1: GPIO51C registers will not be reset by WDT reset Each Serial GPIO pin can be individually programmed to be WDT reset tolerant or not.
7 :0	RW	**Port Serial GPIOE[7:0] WDT reset tolerance enable** 0: GPIO51C registers will be reset by WDT reset 1: GPIO51C registers will not be reset by WDT reset Each Serial GPIO pin can be individually programmed to be WDT reset tolerant or not.

Offset: 538h **GPIO538: Serial GPIO_I/J/K/L 1 Data Value Register** **Init = 0**

Bit	R/W	Description
31:24	RW	**Port Serial GPIOL[7:0] data register**
23:16	RW	**Port Serial GPIOK[7:0] data register**
15:8	RW	**Port Serial GPIOJ[7:0] data register**
7 :0	RW	**Port Serial GPIOI[7:0] data register**

Offset: 53Ch **GPIO53C: Serial GPIO_I/J/K/L 1 Interrupt Enable Register** **Init = 0**

Bit	R/W	Description
31:24	RW	**Port Serial GPIOL[7:0] interrupt enable** 0: Disable interrupt 1: Enable interrupt

to next page

from previous page

23:16	RW	**Port Serial GPIOK[7:0] interrupt enable** 0: Disable interrupt 1: Enable interrupt
15:8	RW	**Port Serial GPIOJ[7:0] interrupt enable** 0: Disable interrupt 1: Enable interrupt
7 :0	RW	**Port Serial GPIOI[7:0] interrupt enable** 0: Disable interrupt 1: Enable interrupt

Offset: 540h GPIO540: Serial GPIO_I/J/K/L 1 Interrupt Sensitivity Type 0 Register Init = 0

Bit	R/W	Description
31:24	RW	**Port Serial GPIOL[7:0] interrupt sensitivity type 0 selection** 0: Select falling-edge or level-low trigger mode 1: Select rising-edge or level-high trigger mode
23:16	RW	**Port Serial GPIOK[7:0] interrupt sensitivity type 0 selection** 0: Select falling-edge or level-low trigger mode 1: Select rising-edge or level-high trigger mode
15:8	RW	**Port Serial GPIOJ[7:0] interrupt sensitivity type 0 selection** 0: Select falling-edge or level-low trigger mode 1: Select rising-edge or level-high trigger mode
7 :0	RW	**Port Serial GPIOI[7:0] interrupt sensitivity type 0 selection** 0: Select falling-edge or level-low trigger mode 1: Select rising-edge or level-high trigger mode

Offset: 544h GPIO544: Serial GPIO_I/J/K/L 1 Interrupt Sensitivity Type 1 Register Init = 0

Bit	R/W	Description
31:24	RW	**Port Serial GPIOL[7:0] interrupt sensitivity type 1 selection** 0: Select edge trigger mode 1: Select level trigger mode
23:16	RW	**Port Serial GPIOK[7:0] interrupt sensitivity type 1 selection** 0: Select edge trigger mode 1: Select level trigger mode
15:8	RW	**Port Serial GPIOJ[7:0] interrupt sensitivity type 1 selection** 0: Select edge trigger mode 1: Select level trigger mode
7 :0	RW	**Port Serial GPIOI[7:0] interrupt sensitivity type 1 selection** 0: Select edge trigger mode 1: Select level trigger mode

Offset: 548h GPIO548: Serial GPIO_I/J/K/L 1 Interrupt Sensitivity Type 2 Register Init = 0

Bit	R/W	Description
31:24	RW	**Port Serial GPIOL[7:0] interrupt sensitivity type 2 selection** 0: Select edge or level trigger mode 1: Select dual-edge trigger mode
23:16	RW	**Port Serial GPIOK[7:0] interrupt sensitivity type 2 selection** 0: Select edge or level trigger mode 1: Select dual-edge trigger mode

to next page

from previous page

15:8	RW	**Port Serial GPIOJ[7:0] interrupt sensitivity type 2 selection** 0: Select edge or level trigger mode 1: Select dual-edge trigger mode
7 :0	RW	**Port Serial GPIOI[7:0] interrupt sensitivity type 2 selection** 0: Select edge or level trigger mode 1: Select dual-edge trigger mode

Offset: 54Ch **GPIO54C: Serial GPIO_I/J/K/L 1 Interrupt Status Register** **Init = 0**

Bit	R/W	Description
31:24	RW	**Port Serial GPIOL[7:0] interrupt status register** Read 0: No interrupt pending Read 1: interrupt pending Write 0: No operation Write 1: Clear interrupt status flag
23:16	RW	**Port Serial GPIOK[7:0] interrupt status register** Read 0: No interrupt pending Read 1: interrupt pending Write 0: No operation Write 1: Clear interrupt status flag
15:8	RW	**Port Serial GPIOJ[7:0] interrupt status register** Read 0: No interrupt pending Read 1: interrupt pending Write 0: No operation Write 1: Clear interrupt status flag
7 :0	RW	**Port Serial GPIOI[7:0] interrupt status register** Read 0: No interrupt pending Read 1: interrupt pending Write 0: No operation Write 1: Clear interrupt status flag

Offset: 550h **GPIO550: Serial GPIO_I/J/K/L 1 Reset Tolerant Register** **Init = 0**

Bit	R/W	Description
31:24	RW	**Port Serial GPIOL[7:0] WDT reset tolerance enable** 0: GPIO538 registers will be reset by WDT reset 1: GPIO538 registers will not be reset by WDT reset Each Serial GPIO pin can be individually programmed to be WDT reset tolerant or not.
23:16	RW	**Port Serial GPIOK[7:0] WDT reset tolerance enable** 0: GPIO538 registers will be reset by WDT reset 1: GPIO538 registers will not be reset by WDT reset Each Serial GPIO pin can be individually programmed to be WDT reset tolerant or not.
15:8	RW	**Port Serial GPIOJ[7:0] WDT reset tolerance enable** 0: GPIO538 registers will be reset by WDT reset 1: GPIO538 registers will not be reset by WDT reset Each Serial GPIO pin can be individually programmed to be WDT reset tolerant or not.
7 :0	RW	**Port Serial GPIOI[7:0] WDT reset tolerance enable** 0: GPIO538 registers will be reset by WDT reset 1: GPIO538 registers will not be reset by WDT reset Each Serial GPIO pin can be individually programmed to be WDT reset tolerant or not.

Offset: 554h	GPIO554: Serial GPIO 1 Configuration Register	Init = 0

Bit	R/W	Description
31:16	RW	**Serial GPIO clock division** Serial GPIO clock period = period of PCLK * 2 * (GPIO554[31:16] + 1)
15:11		**Reserved**
10:6	RW	**Numbers of Serial GPIO pins** 1: 1 byte 2: 2 bytes 3: 3 bytes ... 8: 8 bytes 9: 9 bytes 10: 10 bytes ... 16: 16 bytes others are forbidden.
5 :3		**Reserved**
2		**Reserved**
1		**Reserved**
0	RW	**Enable of Serial GPIO** 0: Disable 1: Enable

Offset: 558h	GPIO558: Serial GPIO_A/B/C/D Input Mask Register	Init = 0

Bit	R/W	Description
31:24	RW	**Port GPIOD[7:0] input mask** 0: Read from GPIO500 will be updated. 1: Read from GPIO500 will not be updated.
23:16	RW	**Port GPIOC[7:0] input mask** 0: Read from GPIO500 will be updated. 1: Read from GPIO500 will not be updated.
15:8	RW	**Port GPIOB[7:0] input mask** 0: Read from GPIO500 will be updated. 1: Read from GPIO500 will not be updated.
7 :0	RW	**Port GPIOA[7:0] input mask** 0: Read from GPIO500 will be updated. 1: Read from GPIO500 will not be updated.

Offset: 55Ch	GPIO55C: Serial GPIO_E/F/G/H Input Mask Register	Init = 0

Bit	R/W	Description
31:24	RW	**Port GPIOH[7:0] input mask** 0: Read from GPIO01C will be updated. 1: Read from GPIO01C will not be updated.
23:16	RW	**Port GPIOG[7:0] input mask** 0: Read from GPIO01C will be updated. 1: Read from GPIO01C will not be updated.
15:8	RW	**Port GPIOF[7:0] input mask** 0: Read from GPIO01C will be updated. 1: Read from GPIO01C will not be updated.

to next page

from previous page

7 :0	RW	**Port GPIOE[7:0] input mask** 0: Read from GPIO01C will be updated. 1: Read from GPIO01C will not be updated.

Offset: 560h	**GPIO560: Serial GPIO_I/J/K/L Input Mask Register**	**Init = 0**
Bit	**R/W**	**Description**
31:24	RW	**Port GPIOL[7:0] input mask** 0: Read from GPIO538 will be updated. 1: Read from GPIO538 will not be updated.
23:16	RW	**Port GPIOK[7:0] input mask** 0: Read from GPIO538 will be updated. 1: Read from GPIO538 will not be updated.
15:8	RW	**Port GPIOJ[7:0] input mask** 0: Read from GPIO538 will be updated. 1: Read from GPIO538 will not be updated.
7 :0	RW	**Port GPIOI[7:0] input mask** 0: Read from GPIO538 will be updated. 1: Read from GPIO538 will not be updated.

Offset: 564h	**GPIO564: Serial GPIO_M/N/O/P Input Mask Register**	**Init = 0**
Bit	**R/W**	**Description**
31:24	RW	**Port GPIOP[7:0] input mask** 0: Read from GPIO590 will be updated. 1: Read from GPIO590 will not be updated.
23:16	RW	**Port GPIOO[7:0] input mask** 0: Read from GPIO590 will be updated. 1: Read from GPIO590 will not be updated.
15:8	RW	**Port GPION[7:0] input mask** 0: Read from GPIO590 will be updated. 1: Read from GPIO590 will not be updated.
7 :0	RW	**Port GPIOM[7:0] input mask** 0: Read from GPIO590 will be updated. 1: Read from GPIO590 will not be updated.

Offset: 570h	**GPIO570: Serial GPIO_A/B/C/D 1 Data Read Register**	**Init = 0**
Bit	**R/W**	**Description**
31: 0	R	**Data written to GPIO500**

Offset: 574h	**GPIO574: Serial GPIO_E/F/G/H 1 Data Read Register**	**Init = 0**
Bit	**R/W**	**Description**
31: 0	R	**Data written to GPIO51C**

Offset: 578h	GPIO578: Serial GPIO_I/J/K/L 1 Data Read Register	Init = 0
Bit	**R/W**	**Description**
31: 0	R	**Data written to GPIO538**

Offset: 590h	GPIO590: Serial GPIO_M/N/O/P 1 Data Value Register	Init = 0
Bit	**R/W**	**Description**
31:24	RW	**Port Serial GPIOP[7:0] data register**
23:16	RW	**Port Serial GPIOO[7:0] data register**
15:8	RW	**Port Serial GPION[7:0] data register**
7 :0	RW	**Port Serial GPIOM[7:0] data register**

Offset: 594h	GPIO594: Serial GPIO_M/N/O/P 1 Interrupt Enable Register	Init = 0
Bit	**R/W**	**Description**
31:24	RW	**Port Serial GPIOP[7:0] interrupt enable** 0: Disable interrupt 1: Enable interrupt
23:16	RW	**Port Serial GPIOO[7:0] interrupt enable** 0: Disable interrupt 1: Enable interrupt
15:8	RW	**Port Serial GPION[7:0] interrupt enable** 0: Disable interrupt 1: Enable interrupt
7 :0	RW	**Port Serial GPIOM[7:0] interrupt enable** 0: Disable interrupt 1: Enable interrupt

Offset: 598h	GPIO598: Serial GPIO_M/N/O/P 1 Interrupt Sensitivity Type 0 Register	Init = 0
Bit	**R/W**	**Description**
31:24	RW	**Port Serial GPIOP[7:0] interrupt sensitivity type 0 selection** 0: Select falling-edge or level-low trigger mode 1: Select rising-edge or level-high trigger mode
23:16	RW	**Port Serial GPIOO[7:0] interrupt sensitivity type 0 selection** 0: Select falling-edge or level-low trigger mode 1: Select rising-edge or level-high trigger mode
15:8	RW	**Port Serial GPION[7:0] interrupt sensitivity type 0 selection** 0: Select falling-edge or level-low trigger mode 1: Select rising-edge or level-high trigger mode
7 :0	RW	**Port Serial GPIOM[7:0] interrupt sensitivity type 0 selection** 0: Select falling-edge or level-low trigger mode 1: Select rising-edge or level-high trigger mode

Offset: 59Ch	GPIO59C: Serial GPIO_M/N/O/P 1 Interrupt Sensitivity Type 1 Register	Init = 0
Bit	**R/W**	**Description**
31:24	RW	**Port Serial GPIOP[7:0] interrupt sensitivity type 1 selection** 0: Select edge trigger mode 1: Select level trigger mode
23:16	RW	**Port Serial GPIOO[7:0] interrupt sensitivity type 1 selection** 0: Select edge trigger mode 1: Select level trigger mode
15:8	RW	**Port Serial GPION[7:0] interrupt sensitivity type 1 selection** 0: Select edge trigger mode 1: Select level trigger mode
7 :0	RW	**Port Serial GPIOM[7:0] interrupt sensitivity type 1 selection** 0: Select edge trigger mode 1: Select level trigger mode

Offset: 5A0h	GPIO5A0: Serial GPIO_M/N/O/P 1 Interrupt Sensitivity Type 2 Register	Init = 0
Bit	**R/W**	**Description**
31:24	RW	**Port Serial GPIOP[7:0] interrupt sensitivity type 2 selection** 0: Select edge or level trigger mode 1: Select dual-edge trigger mode
23:16	RW	**Port Serial GPIOO[7:0] interrupt sensitivity type 2 selection** 0: Select edge or level trigger mode 1: Select dual-edge trigger mode
15:8	RW	**Port Serial GPION[7:0] interrupt sensitivity type 2 selection** 0: Select edge or level trigger mode 1: Select dual-edge trigger mode
7 :0	RW	**Port Serial GPIOM[7:0] interrupt sensitivity type 2 selection** 0: Select edge or level trigger mode 1: Select dual-edge trigger mode

Offset: 5A4h	GPIO5A4: Serial GPIO_M/N/O/P 1 Interrupt Status Register	Init = 0
Bit	**R/W**	**Description**
31:24	RW	**Port Serial GPIOP[7:0] interrupt status register** Read 0: No interrupt pending Read 1: interrupt pending Write 0: No operation Write 1: Clear interrupt status flag
23:16	RW	**Port Serial GPIOO[7:0] interrupt status register** Read 0: No interrupt pending Read 1: interrupt pending Write 0: No operation Write 1: Clear interrupt status flag
15:8	RW	**Port Serial GPION[7:0] interrupt status register** Read 0: No interrupt pending Read 1: interrupt pending Write 0: No operation Write 1: Clear interrupt status flag

to next page

from previous page

7 :0	RW	**Port Serial GPIOM[7:0] interrupt status register** Read 0: No interrupt pending Read 1: interrupt pending Write 0: No operation Write 1: Clear interrupt status flag

Offset: 5A8h **GPIO5A8: Serial GPIO_M/N/O/P 1 Reset Tolerant Register** **Init = 0**

Bit	R/W	Description
31:24	RW	**Port Serial GPIOP[7:0] WDT reset tolerance enable** 0: GPIO500 registers will be reset by WDT reset 1: GPIO500 registers will not be reset by WDT reset Each Serial GPIO pin can be individually programmed to be WDT reset tolerant or not.
23:16	RW	**Port Serial GPIOO[7:0] WDT reset tolerance enable** 0: GPIO500 registers will be reset by WDT reset 1: GPIO500 registers will not be reset by WDT reset Each GPIO pin can be individually programmed to be WDT reset tolerant or not.
15:8	RW	**Port Serial GPION[7:0] WDT reset tolerance enable** 0: GPIO500 registers will be reset by WDT reset 1: GPIO500 registers will not be reset by WDT reset Each Serial GPIO pin can be individually programmed to be WDT reset tolerant or not.
7 :0	RW	**Port Serial GPIOM[7:0] WDT reset tolerance enable** 0: GPIO500 registers will be reset by WDT reset 1: GPIO500 registers will not be reset by WDT reset Each Serial GPIO pin can be individually programmed to be WDT reset tolerant or not.

41.3.4 Serial GPIO Master 2

Offset: 600h	GPIO600: Serial GPIO_A/B/C/D 2 Data Value Register	Init = 0
Bit	**R/W**	**Description**
31:24	RW	**Port Serial GPIOD[7:0] data register**
23:16	RW	**Port Serial GPIOC[7:0] data register**
15:8	RW	**Port Serial GPIOB[7:0] data register**
7 :0	RW	**Port Serial GPIOA[7:0] data register**

Offset: 604h	GPIO604: Serial GPIO_A/B/C/D 2 Interrupt Enable Register	Init = 0
Bit	**R/W**	**Description**
31:24	RW	**Port Serial GPIOD[7:0] interrupt enable** 0: Disable interrupt 1: Enable interrupt
23:16	RW	**Port Serial GPIOC[7:0] interrupt enable** 0: Disable interrupt 1: Enable interrupt
15:8	RW	**Port Serial GPIOB[7:0] interrupt enable** 0: Disable interrupt 1: Enable interrupt
7 :0	RW	**Port Serial GPIOA[7:0] interrupt enable** 0: Disable interrupt 1: Enable interrupt

Offset: 608h	GPIO608: Serial GPIO_A/B/C/D 2 Interrupt Sensitivity Type 0 Register	Init = 0
Bit	**R/W**	**Description**
31:24	RW	**Port Serial GPIOD[7:0] interrupt sensitivity type 0 selection** 0: Select falling-edge or level-low trigger mode 1: Select rising-edge or level-high trigger mode
23:16	RW	**Port Serial GPIOC[7:0] interrupt sensitivity type 0 selection** 0: Select falling-edge or level-low trigger mode 1: Select rising-edge or level-high trigger mode
15:8	RW	**Port Serial GPIOB[7:0] interrupt sensitivity type 0 selection** 0: Select falling-edge or level-low trigger mode 1: Select rising-edge or level-high trigger mode
7 :0	RW	**Port Serial GPIOA[7:0] interrupt sensitivity type 0 selection** 0: Select falling-edge or level-low trigger mode 1: Select rising-edge or level-high trigger mode

Offset: 60Ch	GPIO60C: Serial GPIO_A/B/C/D 2 Interrupt Sensitivity Type 1 Register	Init = 0
Bit	**R/W**	**Description**
31:24	RW	**Port Serial GPIOD[7:0] interrupt sensitivity type 1 selection** 0: Select edge trigger mode 1: Select level trigger mode
23:16	RW	**Port Serial GPIOC[7:0] interrupt sensitivity type 1 selection** 0: Select edge trigger mode 1: Select level trigger mode

to next page

from previous page

15:8	RW	**Port Serial GPIOB[7:0] interrupt sensitivity type 1 selection** 0: Select edge trigger mode 1: Select level trigger mode
7 :0	RW	**Port Serial GPIOA[7:0] interrupt sensitivity type 1 selection** 0: Select edge trigger mode 1: Select level trigger mode

Offset: 610h GPIO610: Serial GPIO_A/B/C/D 2 Interrupt Sensitivity Type 2 Register Init = 0

Bit	R/W	Description
31:24	RW	**Port Serial GPIOD[7:0] interrupt sensitivity type 2 selection** 0: Select edge or level trigger mode 1: Select dual-edge trigger mode
23:16	RW	**Port Serial GPIOC[7:0] interrupt sensitivity type 2 selection** 0: Select edge or level trigger mode 1: Select dual-edge trigger mode
15:8	RW	**Port Serial GPIOB[7:0] interrupt sensitivity type 2 selection** 0: Select edge or level trigger mode 1: Select dual-edge trigger mode
7 :0	RW	**Port Serial GPIOA[7:0] interrupt sensitivity type 2 selection** 0: Select edge or level trigger mode 1: Select dual-edge trigger mode

Offset: 614h GPIO614: Serial GPIO_A/B/C/D 2 Interrupt Status Register Init = 0

Bit	R/W	Description
31:24	RW	**Port Serial GPIOD[7:0] interrupt status register** Read 0: No interrupt pending Read 1: interrupt pending Write 0: No operation Write 1: Clear interrupt status flag
23:16	RW	**Port Serial GPIOC[7:0] interrupt status register** Read 0: No interrupt pending Read 1: interrupt pending Write 0: No operation Write 1: Clear interrupt status flag
15:8	RW	**Port Serial GPIOB[7:0] interrupt status register** Read 0: No interrupt pending Read 1: interrupt pending Write 0: No operation Write 1: Clear interrupt status flag
7 :0	RW	**Port Serial GPIOA[7:0] interrupt status register** Read 0: No interrupt pending Read 1: interrupt pending Write 0: No operation Write 1: Clear interrupt status flag

Offset: 618h	GPIO618: Serial GPIO_A/B/C/D 2 Reset Tolerant Register	Init = 0
Bit	**R/W**	**Description**
31:24	RW	**Port Serial GPIOD[7:0] WDT reset tolerance enable** 0: GPIO600 registers will be reset by WDT reset 1: GPIO600 registers will not be reset by WDT reset Each Serial GPIO pin can be individually programmed to be WDT reset tolerant or not.
23:16	RW	**Port Serial GPIOC[7:0] WDT reset tolerance enable** 0: GPIO600 registers will be reset by WDT reset 1: GPIO600 registers will not be reset by WDT reset Each GPIO pin can be individually programmed to be WDT reset tolerant or not.
15:8	RW	**Port Serial GPIOB[7:0] WDT reset tolerance enable** 0: GPIO600 registers will be reset by WDT reset 1: GPIO600 registers will not be reset by WDT reset Each Serial GPIO pin can be individually programmed to be WDT reset tolerant or not.
7 :0	RW	**Port Serial GPIOA[7:0] WDT reset tolerance enable** 0: GPIO600 registers will be reset by WDT reset 1: GPIO600 registers will not be reset by WDT reset Each Serial GPIO pin can be individually programmed to be WDT reset tolerant or not.

Offset: 61Ch	GPIO61C: Serial GPIO_E/F/G/H 2 Data Value Register	Init = 0
Bit	**R/W**	**Description**
31:24	RW	**Port Serial GPIOH[7:0] data register**
23:16	RW	**Port Serial GPIOG[7:0] data register**
15:8	RW	**Port Serial GPIOF[7:0] data register**
7 :0	RW	**Port Serial GPIOE[7:0] data register**

Offset: 620h	GPIO620: Serial GPIO_E/F/G/H 2 Interrupt Enable Register	Init = 0
Bit	**R/W**	**Description**
31:24	RW	**Port Serial GPIOH[7:0] interrupt enable** 0: Disable interrupt 1: Enable interrupt
23:16	RW	**Port Serial GPIOG[7:0] interrupt enable** 0: Disable interrupt 1: Enable interrupt
15:8	RW	**Port Serial GPIOF[7:0] interrupt enable** 0: Disable interrupt 1: Enable interrupt
7 :0	RW	**Port Serial GPIOE[7:0] interrupt enable** 0: Disable interrupt 1: Enable interrupt

Offset: 624h	GPIO624: Serial GPIO_E/F/G/H 2 Interrupt Sensitivity Type 0 Register	Init = 0
Bit	**R/W**	**Description**
31:24	RW	**Port Serial GPIOH[7:0] interrupt sensitivity type 0 selection** 0: Select falling-edge or level-low trigger mode 1: Select rising-edge or level-high trigger mode
23:16	RW	**Port Serial GPIOG[7:0] interrupt sensitivity type 0 selection** 0: Select falling-edge or level-low trigger mode 1: Select rising-edge or level-high trigger mode

to next page

from previous page

15:8	RW	**Port Serial GPIOF[7:0] interrupt sensitivity type 0 selection** 0: Select falling-edge or level-low trigger mode 1: Select rising-edge or level-high trigger mode
7 :0	RW	**Port Serial GPIOE[7:0] interrupt sensitivity type 0 selection** 0: Select falling-edge or level-low trigger mode 1: Select rising-edge or level-high trigger mode

Offset: 628h GPIO628: Serial GPIO_E/F/G/H 2 Interrupt Sensitivity Type 1 Register Init = 0

Bit	R/W	Description
31:24	RW	**Port Serial GPIOH[7:0] interrupt sensitivity type 1 selection** 0: Select edge trigger mode 1: Select level trigger mode
23:16	RW	**Port Serial GPIOG[7:0] interrupt sensitivity type 1 selection** 0: Select edge trigger mode 1: Select level trigger mode
15:8	RW	**Port Serial GPIOF[7:0] interrupt sensitivity type 1 selection** 0: Select edge trigger mode 1: Select level trigger mode
7 :0	RW	**Port Serial GPIOE[7:0] interrupt sensitivity type 1 selection** 0: Select edge trigger mode 1: Select level trigger mode

Offset: 62Ch GPIO62C: Serial GPIO_E/F/G/H 2 Interrupt Sensitivity Type 2 Register Init = 0

Bit	R/W	Description
31:24	RW	**Port Serial GPIOH[7:0] interrupt sensitivity type 2 selection** 0: Select edge or level trigger mode 1: Select dual-edge trigger mode
23:16	RW	**Port Serial GPIOG[7:0] interrupt sensitivity type 2 selection** 0: Select edge or level trigger mode 1: Select dual-edge trigger mode
15:8	RW	**Port Serial GPIOF[7:0] interrupt sensitivity type 2 selection** 0: Select edge or level trigger mode 1: Select dual-edge trigger mode
7 :0	RW	**Port Serial GPIOE[7:0] interrupt sensitivity type 2 selection** 0: Select edge or level trigger mode 1: Select dual-edge trigger mode

Offset: 630h GPIO630: Serial GPIO_E/F/G/H 2 Interrupt Status Register Init = 0

Bit	R/W	Description
31:24	RW	**Port Serial GPIOH[7:0] interrupt status register** Read 0: No interrupt pending Read 1: interrupt pending Write 0: No operation Write 1: Clear interrupt status flag

to next page

from previous page

23:16	RW	**Port Serial GPIOG[7:0] interrupt status register** Read 0: No interrupt pending Read 1: interrupt pending Write 0: No operation Write 1: Clear interrupt status flag
15:8	RW	**Port Serial GPIOF[7:0] interrupt status register** Read 0: No interrupt pending Read 1: interrupt pending Write 0: No operation Write 1: Clear interrupt status flag
7 :0	RW	**Port Serial GPIOE[7:0] interrupt status register** Read 0: No interrupt pending Read 1: interrupt pending Write 0: No operation Write 1: Clear interrupt status flag

Offset: 634h **GPIO634: Serial GPIO_E/F/G/H 2 Reset Tolerant Register** **Init = 0**

Bit	R/W	Description
31:24	RW	**Port Serial GPIOH[7:0] WDT reset tolerance enable** 0: GPIO61C registers will be reset by WDT reset 1: GPIO61C registers will not be reset by WDT reset Each Serial GPIO pin can be individually programmed to be WDT reset tolerant or not.
23:16	RW	**Port Serial GPIOG[7:0] WDT reset tolerance enable** 0: GPIO61C registers will be reset by WDT reset 1: GPIO61C registers will not be reset by WDT reset Each Serial GPIO pin can be individually programmed to be WDT reset tolerant or not.
15:8	RW	**Port Serial GPIOF[7:0] WDT reset tolerance enable** 0: GPIO61C registers will be reset by WDT reset 1: GPIO61C registers will not be reset by WDT reset Each Serial GPIO pin can be individually programmed to be WDT reset tolerant or not.
7 :0	RW	**Port Serial GPIOE[7:0] WDT reset tolerance enable** 0: GPIO61C registers will be reset by WDT reset 1: GPIO61C registers will not be reset by WDT reset Each Serial GPIO pin can be individually programmed to be WDT reset tolerant or not.

Offset: 638h **GPIO638: Serial GPIO_I/J 2 Data Value Register** **Init = 0**

Bit	R/W	Description
31:16	R	**Reserved**
15:8	RW	**Port Serial GPIOJ[7:0] data register**
7 :0	RW	**Port Serial GPIOI[7:0] data register**

Offset: 63Ch **GPIO63C: Serial GPIO_I/J 2 Interrupt Enable Register** **Init = 0**

Bit	R/W	Description
31:16	R	**Reserved**
15:8	RW	**Port Serial GPIOJ[7:0] interrupt enable** 0: Disable interrupt 1: Enable interrupt

to next page

from previous page

7 :0	RW	**Port Serial GPIOI[7:0] interrupt enable** 0: Disable interrupt 1: Enable interrupt

Offset: 640h **GPIO640: Serial GPIO_I/J 2 Interrupt Sensitivity Type 0 Register** **Init = 0**

Bit	R/W	Description
31:16	R	**Reserved**
15:8	RW	**Port Serial GPIOJ[7:0] interrupt sensitivity type 0 selection** 0: Select falling-edge or level-low trigger mode 1: Select rising-edge or level-high trigger mode
7 :0	RW	**Port Serial GPIOI[7:0] interrupt sensitivity type 0 selection** 0: Select falling-edge or level-low trigger mode 1: Select rising-edge or level-high trigger mode

Offset: 644h **GPIO644: Serial GPIO_I/J 2 Interrupt Sensitivity Type 1 Register** **Init = 0**

Bit	R/W	Description
31:16	R	**Reserved**
15:8	RW	**Port Serial GPIOJ[7:0] interrupt sensitivity type 1 selection** 0: Select edge trigger mode 1: Select level trigger mode
7 :0	RW	**Port Serial GPIOI[7:0] interrupt sensitivity type 1 selection** 0: Select edge trigger mode 1: Select level trigger mode

Offset: 648h **GPIO648: Serial GPIO_I/J 2 Interrupt Sensitivity Type 2 Register** **Init = 0**

Bit	R/W	Description
31:16	R	**Reserved**
15:8	RW	**Port Serial GPIOJ[7:0] interrupt sensitivity type 2 selection** 0: Select edge or level trigger mode 1: Select dual-edge trigger mode
7 :0	RW	**Port Serial GPIOI[7:0] interrupt sensitivity type 2 selection** 0: Select edge or level trigger mode 1: Select dual-edge trigger mode

Offset: 64Ch **GPIO64C: Serial GPIO_I/J 2 Interrupt Status Register** **Init = 0**

Bit	R/W	Description
31:16	R	**Reserved**
15:8	RW	**Port Serial GPIOJ[7:0] interrupt status register** Read 0: No interrupt pending Read 1: interrupt pending Write 0: No operation Write 1: Clear interrupt status flag
7 :0	RW	**Port Serial GPIOI[7:0] interrupt status register** Read 0: No interrupt pending Read 1: interrupt pending Write 0: No operation Write 1: Clear interrupt status flag

Offset: 650h	GPIO650: Serial GPIO_I/J 2 Reset Tolerant Register	Init = 0
Bit	**R/W**	**Description**
31:16	R	**Reserved**
15:8	RW	**Port Serial GPIOJ[7:0] WDT reset tolerance enable** 0: GPIO638 registers will be reset by WDT reset 1: GPIO638 registers will not be reset by WDT reset Each Serial GPIO pin can be individually programmed to be WDT reset tolerant or not.
7 :0	RW	**Port Serial GPIOI[7:0] WDT reset tolerance enable** 0: GPIO638 registers will be reset by WDT reset 1: GPIO638 registers will not be reset by WDT reset Each Serial GPIO pin can be individually programmed to be WDT reset tolerant or not.

Offset: 654h	GPIO654: Serial GPIO 2 Configuration Register	Init = 0
Bit	**R/W**	**Description**
31:16	RW	**Serial GPIO clock division** Serial GPIO clock period = period of PCLK * 2 * (GPIO654[31:16] + 1)
15:11		**Reserved**
10 :6	RW	**Numbers of Serial GPIO pins** 1: 1 byte 2: 2 bytes 3: 3 bytes ... 8: 8 bytes 9: 9 bytes 10: 10 bytes others are forbidden.
5 :3		**Reserved**
2		**Reserved**
1		**Reserved**
0	RW	**Enable of Serial GPIO** 0: Disable 1: Enable

Offset: 670h	GPIO670: Serial GPIO_A/B/C/D 2 Data Read Register	Init = 0
Bit	**R/W**	**Description**
31: 0	R	**Data written to GPIO600**

Offset: 674h	GPIO674: Serial GPIO_E/F/G/H 2 Data Read Register	Init = 0
Bit	**R/W**	**Description**
31: 0	R	**Data written to GPIO61C**

Offset: 678h	GPIO678: Serial GPIO_I/J 2 Data Read Register	Init = 0
Bit	**R/W**	**Description**
31: 0	R	**Data written to GPIO638**

41.3.5 SGPIO Slave 1

Offset: 700h	GPIO700: SGPIO Slave Initiator Data Register #1	Init = 0
Offset: 704h	GPIO704: SGPIO Slave Initiator Data Register #2	Init = 0
Offset: 708h	GPIO708: SGPIO Slave Initiator Data Register #3	Init = 0

Bit	Attr.	Description
31:0	R	SGPIO Slave Initiator Data Register

Offset: 70Ch	GPIO70C: SGPIO Slave Target Data Register #1	Init = 0
Offset: 710h	GPIO710: SGPIO Slave Target Data Register #2	Init = 0
Offset: 714h	GPIO714: SGPIO Slave Target Data Register #3	Init = 0

Bit	Attr.	Description
31:0	R	SGPIO Slave Target Data Register

Offset: 718h	GPIO718: SGPIO Slave Status	Init = 0

Bit	R/W	Description
31:20		Reserved
19:16	RW	SLoad Pattern
15: 1		Reserved
0	R	SGPIO Slave Valie Frame

Offset: 71Ch	GPIO71C: SGPIO Slave Interrupt Enable and Status	Init = 0

Bit	R/W	Description
31:17		Reserved
16	RW	Interrupt Enable
15: 1		Reserved
0	RW1C	Set when a new frame comes

41.3.6 SGPIO Slave 2

Offset: 740h	GPIO740: SGPIO Slave Initiator Data Register #1	Init = 0
Offset: 744h	GPIO744: SGPIO Slave Initiator Data Register #2	Init = 0
Offset: 748h	GPIO748: SGPIO Slave Initiator Data Register #3	Init = 0

Bit	Attr.	Description
31:0	R	SGPIO Slave Initiator Data Register

Offset: 74Ch	GPIO74C: SGPIO Slave Target Data Register #1	Init = 0
Offset: 750h	GPIO750: SGPIO Slave Target Data Register #2	Init = 0
Offset: 754h	GPIO754: SGPIO Slave Target Data Register #3	Init = 0

Bit	Attr.	Description
31:0	R	SGPIO Slave Target Data Register

Offset: 758h	GPIO758: SGPIO Slave Status	Init = 0
Bit	**R/W**	**Description**
31:20		**Reserved**
19:16	RW	**SLoad Pattern**
15: 1		**Reserved**
0	R	**SGPIO Slave Valie Frame**

Offset: 75Ch	GPIO75C: SGPIO Slave Interrupt Enable and Status	Init = 0
Bit	**R/W**	**Description**
31:17		**Reserved**
16	RW	**Interrupt Enable**
15: 1		**Reserved**
0	RW1C	**Set when a new frame comes**

41.4 Programming Guide

41.4.1 LPC port80h direct to GPIO

In AST2600 GPIO, it supports output data from 80h. Taking GPIOA as example.

1. Configure LPC snoop function.
   (a) Set SNPWADR(0x1e789090)[15:0] to 0x80.
   (b) Set HICR5(0x1e789080)[0] to 1 to enable snoop.
2. Configure GPIOA
   (a) Set GPIO004[7:0] to 0xff.
   (b) Set GPIO060[0] to 1 and GPIO064[0] to 0.
3. Set SuperIO
   (a) Set SIOR7_30h to 0x80.
   (b) Set SIOR7_38h to 0x0. 0x0 for GPIOA, 0x1 for GPIOB, and last group is 0x1B for GPIOAB. Please make sure all selected GPIOs are outputable.

41.4.2 LPC port80h direct to SGPIO

In AST2600 SGPIO, it supports output data from 80h. It always uses SGPIOA.

1. Configure LPC snoop function.
   (a) Set SNPWADR(0x1e789090)[15:0] to 0x80.
   (b) Set HICR5(0x1e789080)[0] to 1 to enable snoop.
2. Configure SGPIO
   (a) Set GPIO554[9:6] to larger than or equal to 0x1.
   (b) Set GPIO554[0] to 1 to enable SGPIO.
3. Set SuperIO
   (a) Set SIOR7_30h to 0x40.

41.4.3 Parallel GPIO output driving mode

For all parallel GPIO pins of AST2600, it can support both push-pull or open-drain driving mode.

- Push-pull output mode
  1. Program the direction control register as output mode
  2. Program the data register to 0/1 to drive low/high output
  3. The external pin voltage level can be read from the data register
  4. The programmed output value can be read from the ”GPIO Data Read Register”
- Open-drain output mode
  1. Program the data register to the value of output active level, ’0’ for active-low and ’1’ for active-high
  2. Program the direction control register to output mode for driving active-level output, and program to input mode for inactive-level output

41.4.4 GPIO Index guide

- Write
  1. Put index number to Index Number.
  2. Put 0 to Index Command.
  3. Put correponding register number to Index Type.
  4. Put proper value to Indes Data.
  5. Write to GPIO2AC or GPIOAAC
- Read
  1. Put index number to Index Number.
  2. Put 1 to Index Command.
  3. Put correponding register number to Index Type.
  4. Write to GPIO2AC or GPIOAAC
  5. Read from GPIO2AC or GPIOAAC
  6. Value is presented in Index Data.

42 Real Time Clock (RTC)

42.1 Overview

Real Time Clock (RTC) provides separated second, minute, hour, day, month and year counters. The second counter is toggled once every second, the minute counter is toggled once every minute, the hour counter is toggled once every hour, and so on. The separated counter mechanism reduces the complexity of software. The software only needs to read counter values and gets the current time.

RTC provides second, minute, hour and day and clock alarm function. When turned on the second alarm function, the RTC will auto trigger an interrupt each second. Also, the auto minute, hour and day alarm can be turned on. The function is useful for implementing a clock.

RTC totally implements 5 sets of 32-bit registers, which are listed below, to program the various supported functions. Each register has its own specific offset value, ranging from 0x00 to 0x14h, to derive its physical address location.

Base address of RTC = 0x1E78_1000
Physical address = (Base address of RTC) + Offset

RTC00: Counter Status Register #1
RTC04: Counter Status Register #2
RTC08: Clock Alarm Register
RTC10: Control Register
RTC14: Alarm Status Register

42.2 Features

- Directly connected to APB bus
- Clock source is divided from 25MHz clock input
- Precision ≈ 50ppm (25MHz input precision), approximately 1 second deviation for each 12 hours. So it is recommended to sync the time with the time server for couple days or couple weeks.
- Support Full Calendar function with correct leap years
- Clock mode for calculating:
  - seconds (0-59)
  - minutes (0-59)
  - hours (0-23)
  - days of month (1-28,29,30,31)
  - month (1-12)
  - year (0-99)
  - century (0-31) (hundred digits of year, ex. year 2013 = 20)
- Programmable alarm with interrupt generation
  - Periodic alarm for second, minute, hour or day setting seperately
  - Periodic alarm for a specified day/hour/minute/second time within a month
- Maskable interrupt
- No battery backup supported

42.3 Registers : Base Address = 0x1E78:1000

Offset: 00h	RTC00: Counter Status Register #1	Init = X

Bit	R/W	Description
31:29	RO	**Reserved (0)**
28:24	RW	**DayCnt: Status of Day Counter.** The DayCnt register is the RTC day counter register. After RTC is enabled, the DayCnt value increases by day. When the DayCnt value exceeds the day of the month, the value is reset to one. The DayCnt range is 1 ∼ 28,29,30 or 31. If the RTC is disabled, the DayCnt will hold the value.
23:21	RO	**Reserved (0)**
20:16	RW	**HourCnt: Status of Hour Counter.** The HourCnt register is the RTC hour counter register. After RTC is enabled, the HourCnt value increases by hour. When the HourCnt value exceeds 23, the value is reset to zero. The HourCnt range is 0 ∼ 23. If the RTC is disabled, the HourCnt will hold the value.
15:14	RO	**Reserved (0)**
13:8	RW	**MinuCnt: Status of Minute Counter.** The MinuCnt register is the RTC minute counter register. After RTC is enabled, the MinuCnt value increases by minute. When the MinuCnt value exceeds 59, the value is reset to zero. The MinuCnt range is 0 ∼ 59. If the RTC is disabled, the MinuCnt will hold the value.
7 :6	RO	**Reserved (0)**
5 :0	RW	**SecCnt: Status of Second Counter.** The SecCnt register is the RTC second counter register. After RTC is enabled, the SecCnt value increases by second. When the SecCnt value exceeds 59, the value is reset to zero. The SecCnt range is 0 ∼ 59. If the RTC is disabled, the SecCnt will hold the value.

Note :
The RTC counter value can be updated directly whenever the clock lock was disabled. The new value will be updated into RTC counter within about 4 us.

Offset: 04h	RTC04: Counter Status Register #2	Init = X

Bit	R/W	Description
31:21	RO	**Reserved (0)**
20:16	RW	**CentCnt: Status of Century Counter.** The CentCnt register is the RTC century counter register. After RTC is enabled, the CentCnt value increases by century. The CentCnt range is 0 ∼ 31. If the RTC is disabled, the CentCnt will hold the value.
15	RO	**Reserved (0)**
14:8	RW	**YearCnt: Status of Year Counter.** The YearCnt register is the RTC year counter register. After RTC is enabled, the YearCnt value increases by year. When the YearCnt value exceeds 99, the value is reset to zero. The YearCnt range is 0 ∼ 99. If the RTC is disabled, the YearCnt will hold the value.
7 :4	RO	**Reserved (0)**
3 :0	RW	**MonCnt: Status of Month Counter.** The MonCnt register is the RTC month counter register. After RTC is enabled, the MonCnt value increases by month. When the MonCnt value exceeds 12, the value is reset to one. The MonCnt range is 1 ∼ 12. If the RTC is disabled, the MonCnt will hold the value.

Note :
The RTC counter value can be updated directly whenever the clock lock was disabled. The new value will be updated into RTC counter within about 4 us.

Offset: 08h	RTC08: Clock Alarm Register	Init = X

Bit	R/W	Description
28:24	RW	**Day alarm**
20:16	RW	**Hour alarm**
13:8	RW	**Minute alarm**
5 :0	RW	**Second alarm**

Offset: 10h	RTC10: Control Register	Init = 0

Bit	R/W	Description
31:9	RO	**Reserved (0)**
8	RW	**Enable wakeup alarm** 1: enable 0: disable The time for wakup alarm is the same as alarm interrupt defined at bit[6:2]. The wakeup alarm can work when system enters power saving mode and no bus clock. It only requires the 1MHz clock source to work.
7	RW	**Enable second interrupt** 1: enable 0: disable This interrupt differs from second alarm, it is triggerred for each second count.
6	RW	**Enable day alarm** 1: enable 0: disable
5	RW	**Enable hour alarm** 1: enable 0: disable
4	RW	**Enable minute alarm** 1: enable 0: disable
3	RW	**Enable second alarm** 1: enable 0: disable
2	RW	**Alarm mode selection** 1: Combination mode, alarm is issued when all enabled alarm were met. 0: Individual mode, alarm is issued depending on which alarm was set.
1	RW	**RTC Lock** 1: enable update 0: disable update
0	RW	**RTC enable** 1: enable 0: disable Default setting is disabled

Offset: 14h	RTC14: Alarm Status Register	Init = 0

Bit	R/W	Description
31:6	RO	**Reserved (0)**

to next page

from previous page

5	RW	**Wakeup alarm status** 1: Alarmed 0: Idle To clear this bit, set RTC10[8] = 0 for at least 3 us.
4	RW	**Second interrupt status** 1: Alarmed 0: Idle
3	RW	**Day alarm status** 1: Alarmed 0: Idle
2	RW	**Hour alarm status** 1: Alarmed 0: Idle
1	RW	**Minute alarm status** 1: Alarmed 0: Idle
0	RW	**Second or Combination alarm status** 1: Alarmed 0: Idle
Note : Write '1' to the specific bits to clear the alarm status.		

42.4 Operation

42.4.1 Initialize Sequence

Software must execute 1 time Initial sequence whenever power on reset happened.

1. Set RTC10[1:0] = "11"
2. Update Day, Hour, Minute and Second at RTC00
3. Update Century, Year and Month at RTC04
4. Lock RTC timer by RTC10[1] = 0
5. Set alarm time value
6. Enable alarm interrupt if necessary

42.4.2 Alarm Mode

Individual Mode

Alarm will be triggerred periodically whenever the set time and enabled alarm mode was met. For example, set the alarm time = Day:20 15:30:20

- Enable the Second alarm, Second alarm will be triggerred periodically when the second counter meets 20 for every minutes.
- Enable the Minute alarm, Minute alarm will be triggerred periodically when the minute counter meets 30 for every hours.
- Enable the Hour alarm, Hour alarm will be triggerred periodically when the hour counter meets 15 for every days.
- When multiple alarms are enabled at the same time, then different alarm will be triggerred seperately when the enabled alarm value meets the counter value.

Combination Mode

Alarm will be triggerred periodically whenever the set time and enabled alarm modes all were met. For example, set the alarm time = Day:20 15:30:20

- Enable Minute and Second alarm, alarm will be triggerred at min=30 and sec=20 for every hours.
- Enable Hour and Minute alarm, alarm will be triggerred at hour=15 and minute=30 for every days.

43 Timer Controller (TIMER)

43.1 Overview

Timer Controller (TMC) includes 8 sets of 32-bit decrement counters, based on either PCLK or 1 MHz clock. Each counter is equipped with two sets of matching registers. When any one of the Match registers equal to the corresponding counter value, a timer interrupt will be triggered. Each counter also will trigger an interrupt whenever overflow occurs. Furthermore, all the counter values can be read back at any time.

Below are the change items compared to AST2500.

- Remove pulse generation function.
- Add watchdog reset tolerance for SOC reset.
- Change interrupt type from edge trigger to level trigger.
- Change software programming method for TMC30.

Base address of Timer = 0x1E78_2000
Physical address = (Base address of Timer) + Offset

TMC00: Counter #1 Status Register
TMC10: Counter #2 Status Register
TMC20: Counter #3 Status Register
TMC40: Counter #4 Status Register
TMC50: Counter #5 Status Register
TMC60: Counter #6 Status Register
TMC70: Counter #7 Status Register
TMC80: Counter #8 Status Register

TMC04: Counter #1 Reload Value Register
TMC14: Counter #2 Reload Value Register
TMC24: Counter #3 Reload Value Register
TMC44: Counter #4 Reload Value Register
TMC54: Counter #5 Reload Value Register
TMC64: Counter #6 Reload Value Register
TMC74: Counter #7 Reload Value Register
TMC84: Counter #8 Reload Value Register

TMC08: Counter #1 First Matching Register
TMC18: Counter #2 First Matching Register
TMC28: Counter #3 First Matching Register
TMC48: Counter #4 First Matching Register
TMC58: Counter #5 First Matching Register
TMC68: Counter #6 First Matching Register
TMC78: Counter #7 First Matching Register
TMC88: Counter #8 First Matching Register

TMC0C: Counter #1 Second Matching Register
TMC1C: Counter #2 Second Matching Register
TMC2C: Counter #3 Second Matching Register
TMC4C: Counter #4 Second Matching Register
TMC5C: Counter #5 Second Matching Register
TMC6C: Counter #6 Second Matching Register
TMC7C: Counter #7 Second Matching Register

TMC8C: Counter #8 Second Matching Register

TMC30: Control Register
TMC34: Interrupt Status Register
TMC3C: TMC30 Clear Register

43.2 Features

- Directly connected to APB Bus
- Built-in 8 sets of 32-bit timer modules
- Free-running or periodic mode
- Maskable interrupts
- ~~4 of 8 sets timer can generate a programmable period and duty cycle pulse output.~~
- ~~6 of 8 sets timer can generate a programmable sequence signal output, it can be used to control power sequence.~~

43.3 Registers : Base Address = 0x1E78:2000

Offset: 00h	**TMC00: Counter #1 Status Register**	**Init = 0**
Offset: 10h	**TMC10: Counter #2 Status Register**	**Init = 0**
Offset: 20h	**TMC20: Counter #3 Status Register**	**Init = 0**
Offset: 40h	**TMC40: Counter #4 Status Register**	**Init = 0**
Offset: 50h	**TMC50: Counter #5 Status Register**	**Init = 0**
Offset: 60h	**TMC60: Counter #6 Status Register**	**Init = 0**
Offset: 70h	**TMC70: Counter #7 Status Register**	**Init = 0**
Offset: 80h	**TMC80: Counter #8 Status Register**	**Init = 0**

Bit	Attr.	Description
31:0	RW	**Counter status** This register stores the current status of counter. When timer enable bit TMC30 [0] is disabled, the reload register will be loaded into this counter. When timer enable bit TMC30 [0] is set, the counter will start to decrement. CPU can update this register value when enable bit is set.

Offset: 04h	**TMC04: Counter #1 Reload Value Register**	**Init = 0**
Offset: 14h	**TMC14: Counter #2 Reload Value Register**	**Init = 0**
Offset: 24h	**TMC24: Counter #3 Reload Value Register**	**Init = 0**
Offset: 44h	**TMC44: Counter #4 Reload Value Register**	**Init = 0**
Offset: 54h	**TMC54: Counter #5 Reload Value Register**	**Init = 0**
Offset: 64h	**TMC64: Counter #6 Reload Value Register**	**Init = 0**
Offset: 74h	**TMC74: Counter #7 Reload Value Register**	**Init = 0**
Offset: 84h	**TMC84: Counter #8 Reload Value Register**	**Init = 0**

Bit	Attr.	Description
31:0	RW	**Counter reload value register** When counter decrease to zero, the reload value will be loaded to counter automatically.

Offset: 08h	**TMC08: Counter #1 First Matching Register**	**Init = 0**
Offset: 18h	**TMC18: Counter #2 First Matching Register**	**Init = 0**
Offset: 28h	**TMC28: Counter #3 First Matching Register**	**Init = 0**
Offset: 48h	**TMC48: Counter #4 First Matching Register**	**Init = 0**
Offset: 58h	**TMC58: Counter #5 First Matching Register**	**Init = 0**
Offset: 68h	**TMC68: Counter #6 First Matching Register**	**Init = 0**
Offset: 78h	**TMC78: Counter #7 First Matching Register**	**Init = 0**
Offset: 88h	**TMC88: Counter #8 First Matching Register**	**Init = 0**

Bit	Attr.	Description
31:0	RW	**First set match register** When counter match this register, the timer will generate an edge triggered interrupt to CPU.

Offset: 0Ch	**TMC0C: Counter #1 Second Matching Register**	**Init = 0**
Offset: 1Ch	**TMC1C: Counter #2 Second Matching Register**	**Init = 0**
Offset: 2Ch	**TMC2C: Counter #3 Second Matching Register**	**Init = 0**
Offset: 4Ch	**TMC4C: Counter #4 Second Matching Register**	**Init = 0**
Offset: 5Ch	**TMC5C: Counter #5 Second Matching Register**	**Init = 0**
Offset: 6Ch	**TMC6C: Counter #6 Second Matching Register**	**Init = 0**
Offset: 7Ch	**TMC7C: Counter #7 Second Matching Register**	**Init = 0**
Offset: 8Ch	**TMC8C: Counter #8 Second Matching Register**	**Init = 0**

Bit	Attr.	Description
31:0	RW	**Secondary match register** When counter match this register, the timer will generate an edge triggered interrupt to CPU.

Offset: 30h	**TMC30: Control Register**	**Init = 0**

Bit	R/W	Description
31	RW	**Enable Timer #8 can be reset by watchdog**
30	RW	**Enable Overflow Interrupt for Timer/Counter #8** 0: disable. 1: enable. when timer overflow (count to 0) occurred, interrupt will be generated
29	RW	**Clock selection for Timer/Counter #8** Counter is base on the selected clock to count down 0: APB clock (PCLK) 1: 1 MHz clock
28	RW	**Timer enable for Timer/Counter #8** 0: disable 1: enable When timer is disabled, all action for counter, reload, and interrupt will be gated.
27	RW	**Enable Timer #7 can be reset by watchdog**
26	RW	**Enable Overflow Interrupt for Timer/Counter #7** 0: disable. 1: enable. when timer overflow (count to 0) occurred, interrupt will be generated
25	RW	**Clock selection for Timer/Counter #7** Counter is base on the selected clock to count down 0: APB clock (PCLK) 1: 1 MHz clock

to next page

from previous page

24	RW	**Timer enable for Timer/Counter #7** 0: disable 1: enable When timer is disabled, all action for counter, reload, and interrupt will be gated.
23	RW	**Enable Timer #6 can be reset by watchdog**
22	RW	**Enable Overflow Interrupt for Timer/Counter #6** 0: disable. 1: enable. when timer overflow (count to 0) occurred, interrupt will be generated
21	RW	**Clock selection for Timer/Counter #6** Counter is base on the selected clock to count down 0: APB clock (PCLK) 1: 1 MHz clock
20	RW	**Timer enable for Timer/Counter #6** 0: disable 1: enable When timer is disabled, all action for counter, reload, and interrupt will be gated.
19	RW	**Enable Timer #5 can be reset by watchdog**
18	RW	**Enable Overflow Interrupt for Timer/Counter #5** 0: disable. 1: enable. when timer overflow (count to 0) occurred, interrupt will be generated
17	RW	**Clock selection for Timer/Counter #5** Counter is base on the selected clock to count down 0: APB clock (PCLK) 1: 1 MHz clock
16	RW	**Timer enable for Timer/Counter #5** 0: disable 1: enable When timer is disabled, all action for counter, reload, and interrupt will be gated.
15	RW	**Enable Timer #4 can be reset by watchdog**
14	RW	**Enable Overflow Interrupt for Timer/Counter #4** 0: disable. 1: enable. when timer overflow (count to 0) occurred, interrupt will be generated
13	RW	**Clock selection for Timer/Counter #4** Counter is base on the selected clock to count down 0: APB clock (PCLK) 1: 1 MHz clock
12	RW	**Timer enable for Timer/Counter #4** 0: disable 1: enable When timer is disabled, all action for counter, reload, and interrupt will be gated.
11	RW	**Enable Timer #3 can be reset by watchdog**
10	RW	**Enable Overflow Interrupt for Timer/Counter #3** 0: disable. 1: enable. when timer overflow (count to 0) occurred, interrupt will be generated
9	RW	**Clock selection for Timer/Counter #3** Counter is base on the selected clock to count down 0: APB clock (PCLK) 1: 1 MHz clock

to next page

from previous page

8	RW	**Timer enable for Timer/Counter #3** 0: disable 1: enable When timer is disabled, all action for counter, reload, and interrupt will be gated.
7	RW	**Enable Timer #2 can be reset by watchdog**
6	RW	**Enable Overflow Interrupt for Timer/Counter #2** 0: disable. 1: enable. when timer overflow (count to 0) occurred, interrupt will be generated
5	RW	**Clock selection for Timer/Counter #2** Counter is base on the selected clock to count down 0: APB clock (PCLK) 1: 1 MHz clock
4	RW	**Timer enable for Timer/Counter #2** 0: disable 1: enable When timer is disabled, all action for counter, reload, and interrupt will be gated.
3	RW	**Enable Timer #1 can be reset by watchdog**
2	RW	**Enable Overflow Interrupt for Timer/Counter #1** 0: disable. 1: enable. when timer overflow (count to 0) occurred, interrupt will be generated
1	RW	**Clock selection for Timer/Counter #1** Counter is base on the selected clock to count down 0: APB clock (PCLK) 1: 1 MHz clock
0	RW	**Timer enable for Timer/Counter #1** 0: disable 1: enable When timer is disabled, all action for counter, reload, and interrupt will be gated.

Note :
Write ’1’ to TMC30 will set the specific bits to ’1’. To clear the specific bits to ’0’, it should write ’1’ to TMC3C on the same bit position.

Offset: 34h **TMC34: Interrupt Status Register** **Init = 0**

Bit	R/W	Description
31:8	RO	**Reserved (0)**
7	RW	**Timer8 Interrupt Event**
6	RW	**Timer7 Interrupt Event**
5	RW	**Timer6 Interrupt Event**
4	RW	**Timer5 Interrupt Event**
3	RW	**Timer4 Interrupt Event**
2	RW	**Timer3 Interrupt Event**
1	RW	**Timer2 Interrupt Event**
0	RW	**Timer1 Interrupt Event**

Note :
Interrupt status set by HW, and clear to ’0’ by software writing ’1’ on the specific bit.

Offset: 3Ch	TMC3C: TMC30 Clear Register	Init = 0

Bit	R/W	Description
31:0	W1C	**Clear TMC30** Write specific = '1', clear specific bit of TMC30 to 0 Write specific = '0', no effect

43.4 Operation

Reload, Match1, Match2 and Control[Interrupt] must be set when timer is used. Reload controls the period between twice overflow. For example, if 0x02 value be set to Reload and then enable timer Control[Enable], the sequence of counter is 2,1,0,2,1,....
An interrupt can be generated when timer counter reach zero, if Control[Interrupt] was set.

Initial Sequence :

1. Set Reload
2. Set Control[Interrupt]
3. Enable Timer, Control[Enable]

Update Timer Value Sequence :

- Method 1
  1. Disable Timer
  2. Set Reload
  3. Set Control[Interrupt]
  4. Enable Timer, Control[Enable]
- Method 2
  1. Set new counting value to status register

43.5 Programming Note

43.5.1 Interrupt Generation

The timer interrupt event will be generated by any one of the following 3 conditions:

1. Counter count to 0 (overflow) and overflow interrupt enabled.
2. Counter value match the first matching register value.
3. Counter value match the second matching register value.

So, if you don't want to use the match register function, please set the match register values to 0xFFFFFFFF or 0x0.

44 UART Controller (16550)

44.1 Overview

AST2600 integrates 13 sets of UART (Universal Asynchronous Receiver/Transmitter) providing serial communication capabilities with other external devices, like another computer using a serial cable based on RS232 protocol. This core is designed to be compatible with the industry standard — 16550 UART. The 13 sets of UART are equipped with a 16x8 FIFO that can be programmed to be enabled or disabled. The supported baud rates are also programmable.

Each unit of UART totally implements 12 sets of 32-bit registers, which are listed below, to program the various supported functions including character length selection, baud rate selection, interrupt generation, and parity generation/checking. Each register has its own specific offset value, ranging from 0x00 to 0x14h, to derive its physical address location.

Base Address of UART1 = 0x1E78_3000
Base Address of UART2 = 0x1E78_D000
Base Address of UART3 = 0x1E78_E000
Base Address of UART4 = 0x1E78_F000
Base Address of UART5 = 0x1E78_4000
Base Address of UART6 = 0x1E79_0000
Base Address of UART7 = 0x1E79_0100
Base Address of UART8 = 0x1E79_0200
Base Address of UART9 = 0x1E79_0300
Base Address of UART10 = 0x1E79_0400
Base Address of UART11 = 0x1E79_0500
Base Address of UART12 = 0x1E79_0600
Base Address of UART13 = 0x1E79_0700
Register Address of UART = (Base Address of UART) + Offset

UART_RBR: Receiving Buffer Register (DLAB = 0)
UART_THR: Transmit Holding Register (DLAB = 0)
UART_IER: Interrupt Enable Register (DLAB = 0)
UART_IIR: Interrupt Identity Register
UART_FCR: FIFO Control Register
UART_LCR: Line Control Register
UART_MCR: Modem Control Register
UART_LSR: Line Status Register
UART_MSR: Modem Status Register
UART_SCR: Scratch Register
UART_DLL: Divisor Latch Low Register : (DLAB = 1)
UART_DLH: Divisor Latch High Register : (DLAB = 1)

The UART packet frame format is shown as Figure 54.

44.2 Features

- Directly connected to APB bus
- Support two UART with full flow control pins (one is with dedicated flow control pins, the other is shared with GPIO pins)
- Separate transmit & receive FIFO buffer (16x8) to reduce CPU interrupts
- Support up to 3686.4K baud-rate except UART5 921.6K baud-rate

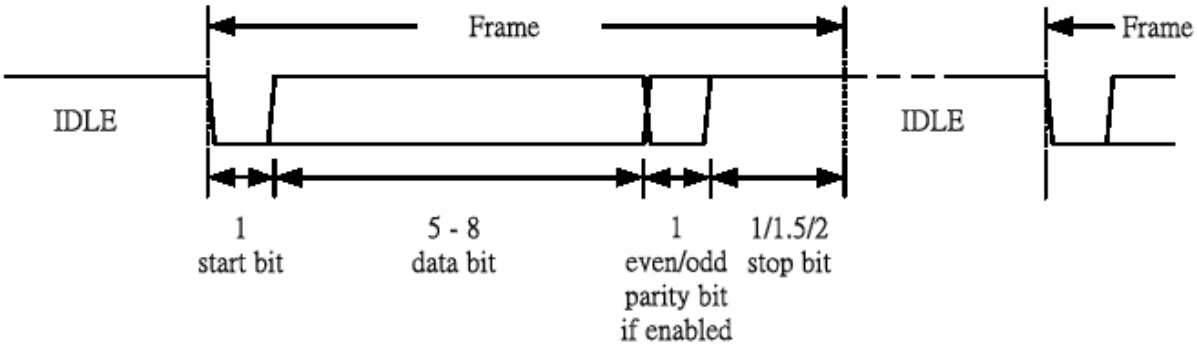


Figure 54: UART Packet Frame

- Programmable baud rate generator
- Standard asynchronous communication bits — Stat/Stop/Parity
- Independent masking of transmit FIFO, receive FIFO, receiving timeout and error condition Interrupts
- False start-bit detection
- Line break generation and detection
- Fully programmable serial interface characteristics:
  - 5/6/7/8 data length
  - Even, odd and none parity generation and detection
  - 1/2 stop-bit generation
- Extended diagnostic Loopback Mode allows testing more Modem Control and Auto Flow Control features

44.3 Registers

Offset: 00h	**UART_RBR: Receiving Buffer Register (DLAB = 0)**	**Init = 0**
Offset: 00h	**UART_THR: Transmit Holding Register (DLAB = 0)**	**Init = 0**

Bit	Attr.	Description
31:8	RO	**Reserved (0)**
7 :0	RO	**UART_RBR: Receiving Buffer Register** The UART_RBR is a read-only register that contains the data byte received on the serial input port. The data in register is valid only if the Data Ready bit in the Line status Register (UART_LSR) is set. When the **FIFOs are programmed OFF**, the data in the UART_RBR must be read before the next data arrives; otherwise it will be overwritten, resulting in an overrun error. When the **FIFOs are programmed ON**, this register accesses the head of the receive FIFO. If the receive FIFO is full and this register is not read before the next data character arrives, then the data already in the FIFO will be preserved but any incoming data will be lost. An overrun error will also occur.

to next page

from previous page

7:0	WO	**UART_THR: Transmit Holding Register** The UART_THR is a write-only register that contains data to be transmitted on the serial output port. Data can be written to the UART_THR any time that the THR Empty (**THRE**) bit of the Line Status Register (UART_LSR) is set. When the **FIFOs are programmed OFF**, writing a single character to the UART_THR clears the THRE. Any additional writes to the UART_THR before the THRE is set again causes the UART_THR data to be overwritten. When the **FIFOs are programmed ON**, 16 bytes of data may be written to the UART_THR before the FIFO is full. Any attempt to write data when the FIFO is full results in the write data being lost.

Offset: 04h **UART_IER: Interrupt Enable Register (DLAB = 0)** **Init = 0**

Bit	R/W	Description
31:8	RO	**Reserved (0)**
7 :4	RO	**Reserved (0)**
3	RW	**EDSSI: Enable Modem Status Interrupt** 0: Disable interrupt 1: Enable interrupt
2	RW	**ELSI: Enable Receiver Line Status Interrupt** 0: Disable interrupt 1: Enable interrupt
1	RW	**ETBEI: Enable Transmitter Holding Register Empty Interrupt** 0: Disable interrupt 1: Enable interrupt
0	RW	**ERBFI: Enable Received Data Available Interrupt** 0: Disable interrupt 1: Enable interrupt

Offset: 08h **UART_IIR: Interrupt Identity Register** **Init = 0x01**

Bit	R/W	Description
31:8	RO	**Reserved (0)**
7:6	RO	**FIFO-Enabled Bits** 00: FIFOs disabled 11: FIFOs enabled
5:4	RO	**Reserved (0)**
3:1	RO	**Interrupt Decoding Table** The content of this register can be used to identify the source of the current interrupt based on the following: 000: Modem Status Changed 001: UART_THR empty 010: Received Data Available 011: Receiver Status 110: Character Time Out For more information about Interrupt Identity, see the following Table for detailed description.

to next page

from previous page

0	RO	Indicates that an interrupt is pending when its logic ”0”. When its ”1”, no interrupt is pending.

Note :
The IIR enables the programmer to retrieve what is the current highest priority pending interrupt.

UART Interrupt Type Decoding

Bit [3:1]	Priority	Interrupt Type	Interrupt Source	Interrupt Clear Control
011	Highest	Receiver line status	Parity, Overrun, Framing errors or Break Interrupt.	Reading the Line Status Register.
010	2nd	Received Data available	FIFO OFF: Receiver data available FIFO ON: RX FIFO trigger level reached	FIFO OFF: Reading the RBR FIFO ON: FIFO drops below the trigger level
110	3rd	Character Timeout indication	Theres at least 1 character in the FIFO but no character has been input to the FIFO or read from it for the last 4 Character times.	Reading the RBR.
001	4th	Transmitter Holding register empty	THR Empty	Reading the IIR or writing into THR
000	5th	Modem status	nCTS (Clear to send), nDSR (data set ready), nRI (Ring indicator), nDCD (Data center detect)	Reading the UART_MSR.

Offset: 08h **UART_FCR: FIFO Control Register** **Init = 0**

Bit	R/W	Description
31:8	RO	**Reserved (0)**
7:6	WO	**Define the Receiver FIFO Interrupt trigger level.** 00: 1 byte received 01: 4 bytes received 10: 8 bytes received 11: 14 bytes received
5:3	RO	**Reserved (0)**
2	WO	**Transmit FIFO Reset** Writing ”1” to this bit clears the Transmitter FIFO and resets its logic.
1	WO	**Receive FIFO Reset** Writing ”1” to this bit clears the Receiver FIFO and resets its logic.
0	WO	**Enable UART FIFO** 0: Disable FIFO 1: Enable FIFO Changing the value of this register will always reset UART FIFO immediately.

Note :
The FCR allows selection of the FIFO trigger level (the number of bytes in FIFO required to enable the Received Data Available interrupt). In addition, the FIFOs can be cleared using this register.

Offset: 0Ch	UART_LCR: Line Control Register	Init = 0

Bit	R/W	Description
31:8	RO	**Reserved (0)**
7	RW	**DLAB: Divisor latch access bit** 0: The normal registers are accessed. 1: The divisor latches can be accessed. Setting this bit will enable the reading and writing of the Divisor Latch register (UART_DLL and UART_DLH) to set the baud rate of the UART. This bit must be cleared after initial baud rate setup in order to access other registers.
6	RW	**Break Control bit.** 0: break is disabled. 1: When not in Loopback Mode, the serial out is forced into logic '0' (break state). When in Loopback Mode, the break condition is internally looped back to the receiver.
5	RO	**Reserved (0)**
4	RW	**EPS: Parity mode selection** 0: Select odd parity mode (odd number of "1" for data and parity combined) 1: Select even parity mode (even number of "1" for data and parity combined)
3	RW	**PEN: Enable parity bit** 0: Disable parity bit 1: Enable parity bit
2	RW	**STOP: Number of stop bits transmitted** 0: 1 stop bit. 1: 1.5 stop bits when 5-bit character length selected and 2 bits otherwise Note that the receiver always checks the first stop bit only.
1:0	RW	**CLS: Select number of bits per character** 00: 5 bits. 01: 6 bits. 10: 7 bits. 11: 8 bits.

Offset: 10h	UART_MCR: Modem Control Register	Init = 0

Bit	R/W	Description
31:5	RO	**Reserved (0)**
4	RW	**Loopback mode.** 0: normal operation. 1: loopback mode. When in loopback mode, the Serial Output Signal (TXD) is set to logic '1'. The signal of the transmitter shift register is internally connected to the input of the receiver shift register. The following connections are made: nDTR $\rightarrow$ nDSR nRTS $\rightarrow$ nCTS Out1 $\rightarrow$ nRI Out2 $\rightarrow$ nDCD
3	RW	**Out2.** In loopback mode, connected to Data Carrier Detect(nDCD) input.
2	RW	**Out1.** In loopback mode, connected to Ring Indicator (nRI)signal input.
1	RW	**Request To Send (nRTS) signal control.** 0: nRTS is '1' 1: nRTS is '0'

to next page

from previous page

0	RW	**Data Terminal Ready (nDTR) signal control.** 0: nDTR is '1' 1: nDTR is '0'

Offset: 14h **UART_LSR: Line Status Register** **Init = 0x60**

Bit	R/W	Description
31:8	RO	**Reserved (0)**
7	RO	**Error in Receiver FIFO (Read clear)** 1: There is at least one parity error, framing error, or break indication in the FIFO. This bit is only active when FIFOs are enabled. This bit is cleared when the UART_LSR is read. 0: Otherwise.
6	RO	**Transmitter empty** 1: When FIFO enabled, the Transmitter Shift Register and FIFO are both empty. When FIFO disabled, the Transmitter Shift Register and UART_THR are both empty. 0: Otherwise.
5	RO	**THRE: Transmitter holding register empty** 1: This bit is set whenever data is transferred from UART_THR or TX FIFO to the transmitter shift register and no new data has been written to the UART_THR or TX FIFO. This also causes a THRE Interrupt to occur, if THRE Interrupt is enabled. 0: Otherwise.
4	RO	**BI: Break interrupt (Read clear)** 1: The serial input is held in a logic "0" state for longer than the sum of start time + data bits + parity + stop bits. A break condition on serial input causes one and only one character, consisting of all zeros, to be received by the UART. 0: No break condition in the current character.
3	RO	**FE: Framing error (Read clear)** 1: There is a framing error in the receiver. A framing error occurs when the receiver does not detect a valid STOP bit in the received data. 0: No framing error in the current character.
2	RO	**PE: Parity error (Read clear)** 1: There is a parity error in the receiver if the Parity Enable is set. 0: No parity error in the current character.
1	RO	**OE: Overrun error (Read clear)** 1: An overrun error has occurred because a new data character was received before the previous data was read. In the FIFO mode, an overrun error occurs when the FIFO is full and a new character arrives at the receiver. 0: No overrun state.
0	RO	**DR: Data ready** 1: The receiver contains at least one character in the UART_RBR or the receiver FIFO. 0: The UART_RBR is read or the receiver FIFO is empty.

Offset: 18h **UART_MSR: Modem Status Register** **Init = 0**

Bit	R/W	Description
31:8	RO	**Reserved (0)**
7	RO	**Complement of the nDCD input or equals to Out2(MCR[3]) in loopback mode.**
6	RO	**Complement of the nRI input or equals to Out1(MCR[2]) in loopback mode.**
5	RO	**Complement of the nDSR input or equals to nDTR(MCR[0]) in loopback mode.**
4	RO	**Complement of the nCTS input or equals to nRTS(MCR[1]) in loopback mode.**

to next page

from previous page

3	RO	**Delta Data Carrier Detect (DDCD) indicator (Read clear)** 1: The nDCD line has changed its state since the last time the CPU read the MSR. In loopback mode, DDCD reflects changes on MCR bit 3 (Out2)
2	RO	**Trailing Edge of Ring Indicator (TERI) detector (Read clear)** The nRI line has changed its state from low to high state since the last time the CPU read the MSR. In loopback mode, TERI reflects when MCR bit 2 (Out1) has changed state from a high to a low.
1	RO	**Delta Data Set Ready (DDSR) indicator (Read clear)** 1: The nDSR line has changed its state since the last time the CPU read the MSR. In loopback mode, DDSR reflects changes on MCR bit 0 (DTR)
0	RO	**Delta Clear To Send (DCTS) indicator (Read clear)** 1: The nCTS line has changed its state since the last time the CPU read the MSR. In loopback mode, DCTS reflects changes on MCR bit 1 (RTS)
Note : The register displays the current state of the modem control lines. Also, four bits also provide an indication in the state of one of the modem status lines. These bits are set to '1' when a change in corresponding line has been detected and they are reset when the register is being read.		

Offset: 1Ch	**UART_SCR: Scratch Register**	**Init = 0**
Bit	**R/W**	**Description**
31:8	RO	**Reserved (0)**
7 :0	RW	**Scratch bits** This register can be used as a temporary storage, no specific definition.

44.3.1 UART_DLL/UART_DLH

In addition, there are 2 Clock Divisor registers that together to form one 16-bits, read/write, Divisor Latch register that contains the baud rate.
The registers can be accessed when the 7th (DLAB) bit of the Line Control Register idivisor for the UARTs set to '1'. At this time the above registers at addresses 0x0 & 0x4 can't be accessed.

Setting the 7th bit of UART_LCR to "1" can access the divisor latches. You should restore this bit to "0" after setting the divisor latches in order to restore access to the other registers that occupy the same addresses.

The 2 bytes form one 16-bit register, which is internally accessed as a single number. You should therefore set all 2 bytes of the register to ensure normal operation. The register is set to the default value of 0 on reset, which disables all serial I/O operations in order to ensure explicit setup of the register in the software. The value set should be equal to

Baud rate (by default) = (24MHz / 13) / (16 * divisor).

The internal counter starts to work when the LSB of DL is written, so when setting the divisor, write the MSB first and the LSB last.

Offset: 00h	UART_DLL: Divisor Latch Low Register : (DLAB = 1)	Init = 0
Bit	**R/W**	**Description**
31:8	RO	**Reserved.**
7:0	RW	**The LSB of the baud-rate divisor latch.**

Offset: 04h	UART_DLH: Divisor Latch High Register : (DLAB = 1)	Init = 0
Bit	**R/W**	**Description**
31:8	RO	**Reserved.**
7:0	RW	**The MSB of the baud-rate divisor latch.**

45 UART DMA (UDMA)

45.1 Overview

Uart DMA Controller (UDMA) includes 14 sets of Uart transimition and receiver DMA controllers. Each DMA controller can transmit or receive data from or to Uart controller automatically. They also include read and write pointer to their corresponding buffer for software to access the data buffer.

Base address of Uart DMA = 0x1E79_E000
Physical address = (Base address of Uart DMA) + Offset

UDMA000: UART TX DMA enable
UDMA004: UART RX DMA enable
UDMA008: Misc, buffer size
UDMA00C: UART DMA time out timer
UDMA020: UART TX DMA reset
UDMA024: UART RX DMA reset
UDMA030: UART TX DMA interrrupt enable
UDMA034: UART TX DMA interrrupt status
UDMA038: UART RX DMA interrrupt enable
UDMA03C: UART RX DMA interrrupt status
UDMA040: UART1 TX read pointer
UDMA044: UART1 TX write pointer
UDMA048: UART1 TX buffer base address
UDMA04C: UART1 TX control register
UDMA050: UART1 RX read pointer
UDMA054: UART1 RX write pointer
UDMA058: UART1 RX buffer base address
UDMA05C: UART1 RX control register
UDMA060: UART2 TX read pointer
UDMA064: UART2 TX write pointer
UDMA068: UART2 TX buffer base address
UDMA06C: UART2 TX control register
UDMA070: UART2 RX read pointer
UDMA074: UART2 RX write pointer
UDMA078: UART2 RX buffer base address
UDMA07C: UART2 RX control register
UDMA080: UART3 TX read pointer
UDMA084: UART3 TX write pointer
UDMA088: UART3 TX buffer base address
UDMA08C: UART3 TX control register
UDMA090: UART3 RX read pointer
UDMA094: UART3 RX write pointer
UDMA098: UART3 RX buffer base address
UDMA09C: UART3 RX control register
UDMA0A0: UART4 TX read pointer
UDMA0A4: UART4 TX write pointer
UDMA0A8: UART4 TX buffer base address
UDMA0AC: UART4 TX control register
UDMA0B0: UART4 RX read pointer
UDMA0B4: UART4 RX write pointer
UDMA0B8: UART4 RX buffer base address
UDMA0BC: UART4 RX control register
UDMA0C0: UART6 TX read pointer
UDMA0C4: UART6 TX write pointer
UDMA0C8: UART6 TX buffer base address
UDMA0CC: UART6 TX control register

UDMA0D0: UART6 RX read pointer
UDMA0D4: UART6 RX write pointer
UDMA0D8: UART6 RX buffer base address
UDMA0DC: UART6 RX control register
UDMA0E0: UART7 TX read pointer
UDMA0E4: UART7 TX write pointer
UDMA0E8: UART7 TX buffer base address
UDMA0EC: UART7 TX control register
UDMA0F0: UART7 RX read pointer
UDMA0F4: UART7 RX write pointer
UDMA0F8: UART7 RX buffer base address
UDMA0FC: UART7 RX control register
UDMA100: UART8 TX read pointer
UDMA104: UART8 TX write pointer
UDMA108: UART8 TX buffer base address
UDMA10C: UART8 TX control register
UDMA110: UART8 RX read pointer
UDMA114: UART8 RX write pointer
UDMA118: UART8 RX buffer base address
UDMA11C: UART8 RX control register
UDMA120: UART9 TX read pointer
UDMA124: UART9 TX write pointer
UDMA128: UART9 TX buffer base address
UDMA12C: UART9 TX control register
UDMA130: UART9 RX read pointer
UDMA134: UART9 RX write pointer
UDMA138: UART9 RX buffer base address
UDMA13C: UART9 RX control register
UDMA140: UART10 TX read pointer
UDMA144: UART10 TX write pointer
UDMA148: UART10 TX buffer base address
UDMA14C: UART10 TX control register
UDMA150: UART10 RX read pointer
UDMA154: UART10 RX write pointer
UDMA158: UART10 RX buffer base address
UDMA15C: UART10 RX control register
UDMA160: UART11 TX read pointer
UDMA164: UART11 TX write pointer
UDMA168: UART11 TX buffer base address
UDMA16C: UART11 TX control register
UDMA170: UART11 RX read pointer
UDMA174: UART11 RX write pointer
UDMA178: UART11 RX buffer base address
UDMA17C: UART11 RX control register
UDMA180: UART12 TX read pointer
UDMA184: UART12 TX write pointer
UDMA188: UART12 TX buffer base address
UDMA18C: UART12 TX control register
UDMA190: UART12 RX read pointer
UDMA194: UART12 RX write pointer
UDMA198: UART12 RX buffer base address
UDMA19C: UART12 RX control register
UDMA1A0: UART13 TX read pointer
UDMA1A4: UART13 TX write pointer
UDMA1A8: UART13 TX buffer base address
UDMA1AC: UART13 TX control register
UDMA1B0: UART13 RX read pointer

UDMA1B4: UART13 RX write pointer
UDMA1B8: UART13 RX buffer base address
UDMA1BC: UART13 RX control register
UDMA1C0: VUART1 TX read pointer
UDMA1C4: VUART1 TX write pointer
UDMA1C8: VUART1 TX buffer base address
UDMA1CC: VUART1 TX control register
UDMA1D0: VUART1 RX read pointer
UDMA1D4: VUART1 RX write pointer
UDMA1D8: VUART1 RX buffer base address
UDMA1DC: VUART2 RX control register
UDMA1E0: VUART2 TX read pointer
UDMA1E4: VUART2 TX write pointer
UDMA1E8: VUART2 TX buffer base address
UDMA1EC: VUART2 TX control register
UDMA1F0: VUART2 RX read pointer
UDMA1F4: VUART2 RX write pointer
UDMA1F8: VUART2 RX buffer base address
UDMA1FC: VUART2 RX control register

45.2 Features

- Directly connected to Uart Controller
- Built-in time out timer
- 1KB, 4KB, 16KB and 64KB programmable buffer size
- Buffer full interrupt for receiver
- Buffer empty interrupt for transmiter

45.3 Registers : Base Address = 0x1E78:2000

Offset: 000h	UDMA000: UART TX DMA enable	Init = 0
Bit	**R/W**	**Description**
31:14	RO	**reserved(0)**
13	RW	**VUART2 TX DMA enable register**
12	RW	**VUART1 TX DMA enable register**
11	RW	**UART13 TX DMA enable register**
10	RW	**UART12 TX DMA enable register**
9	RW	**UART11 TX DMA enable register**
8	RW	**UART10 TX DMA enable register**
7	RW	**UART9 TX DMA enable register**
6	RW	**UART8 TX DMA enable register**
5	RW	**UART7 TX DMA enable register**
4	RW	**UART6 TX DMA enable register**
3	RW	**UART4 TX DMA enable register**
2	RW	**UART3 TX DMA enable register**
1	RW	**UART2 TX DMA enable register**
0	RW	**UART1 TX DMA enable register**

Offset: 004h	UDMA004: UART RX DMA enable	Init = 0
Bit	**R/W**	**Description**
31:14	RO	**reserved(0)**
13	RW	**VUART2 RX DMA enable register**
12	RW	**VUART1 RX DMA enable register**
11	RW	**UART13 RX DMA enable register**
10	RW	**UART12 RX DMA enable register**
9	RW	**UART11 RX DMA enable register**
8	RW	**UART10 RX DMA enable register**
7	RW	**UART9 RX DMA enable register**
6	RW	**UART8 RX DMA enable register**
5	RW	**UART7 RX DMA enable register**
4	RW	**UART6 RX DMA enable register**
3	RW	**UART4 RX DMA enable register**
2	RW	**UART3 RX DMA enable register**
1	RW	**UART2 RX DMA enable register**
0	RW	**UART1 RX DMA enable register**

Offset: 008h	UDMA008: Misc, buffer size	Init = x
Bit	**R/W**	**Description**
31:5	RO	**reserved(0)**
4:0	RO	**reserved(0)**

Offset: 00Ch	UDMA00C: UART DMA time out timer	Init = x
Bit	**R/W**	**Description**
31:16	RO	**reserved(0)**
15:0	RW	**UART DMA time out timer** The time out is in unit of PCLK_cycle * 14400

Offset: 020h	UDMA020: UART TX DMA reset	Init = 0
Bit	**R/W**	**Description**
31:14	RO	**reserved(0)**
13	RW	**VUART2 TX DMA reset**
12	RW	**VUART1 TX DMA reset**
11	RW	**UART13 TX DMA reset**
10	RW	**UART12 TX DMA reset**
9	RW	**UART11 TX DMA reset**
8	RW	**UART10 TX DMA reset**
7	RW	**UART9 TX DMA reset**
6	RW	**UART8 TX DMA reset**
5	RW	**UART7 TX DMA reset**
4	RW	**UART6 TX DMA reset**
3	RW	**UART4 TX DMA reset**

to next page

from previous page

2	RW	**UART3 TX DMA reset**
1	RW	**UART2 TX DMA reset**
0	RW	**UART1 TX DMA reset**

Offset: 024h **UDMA024: UART RX DMA reset** **Init = 0**

Bit	R/W	Description
31:14	RO	**reserved(0)**
13	RW	**VUART2 RX DMA reset**
12	RW	**VUART1 RX DMA reset**
11	RW	**UART13 RX DMA reset**
10	RW	**UART12 RX DMA reset**
9	RW	**UART11 RX DMA reset**
8	RW	**UART10 RX DMA reset**
7	RW	**UART9 RX DMA reset**
6	RW	**UART8 RX DMA reset**
5	RW	**UART7 RX DMA reset**
4	RW	**UART6 RX DMA reset**
3	RW	**UART4 RX DMA reset**
2	RW	**UART3 RX DMA reset**
1	RW	**UART2 RX DMA reset**
0	RW	**UART1 RX DMA reset**

Offset: 030h **UDMA030: UART TX DMA interrrupt enable** **Init = 0**

Bit	R/W	Description
31:14	RO	**reserved(0)**
13	RW	**VUART2 TX DMA interrupt enable**
12	RW	**VUART1 TX DMA interrupt enable**
11	RW	**UART13 TX DMA interrupt enable**
10	RW	**UART12 TX DMA interrupt enable**
9	RW	**UART11 TX DMA interrupt enable**
8	RW	**UART10 TX DMA interrupt enable**
7	RW	**UART9 TX DMA interrupt enable**
6	RW	**UART8 TX DMA interrupt enable**
5	RW	**UART7 TX DMA interrupt enable**
4	RW	**UART6 TX DMA interrupt enable**
3	RW	**UART4 TX DMA interrupt enable**
2	RW	**UART3 TX DMA interrupt enable**
1	RW	**UART2 TX DMA interrupt enable**
0	RW	**UART1 TX DMA interrupt enable**

Note :
The TX DMA interrept will be asserted when the TX DMA buffer status changed from not empty to empty. The TX buffer status is full when TX read pointer is equal to write pointer.

Offset: 034h	UDMA034: UART TX DMA interrrupt status	Init = 0

Bit	R/W	Description
31:14	RO	**reserved(0)**
13	RW1C	**VUART2 TX DMA interrupt status** This register is interrupt status of TX controller. Write 1 clear.
12	RW1C	**VUART1 TX DMA interrupt status** This register is interrupt status of TX controller. Write 1 clear.
11	RW1C	**UART13 TX DMA interrupt status** This register is interrupt status of TX controller. Write 1 clear.
10	RW1C	**UART12 TX DMA interrupt status** This register is interrupt status of TX controller. Write 1 clear.
9	RW1C	**UART11 TX DMA interrupt status** This register is interrupt status of TX controller. Write 1 clear.
8	RW1C	**UART10 TX DMA interrupt status** This register is interrupt status of TX controller. Write 1 clear.
7	RW1C	**UART9 TX DMA interrupt status** This register is interrupt status of TX controller. Write 1 clear.
6	RW1C	**UART8 TX DMA interrupt status** This register is interrupt status of TX controller. Write 1 clear.
5	RW1C	**UART7 TX DMA interrupt status** This register is interrupt status of TX controller. Write 1 clear.
4	RW1C	**UART6 TX DMA interrupt status** This register is interrupt status of TX controller. Write 1 clear.
3	RW1C	**UART4 TX DMA interrupt status** This register is interrupt status of TX controller. Write 1 clear.
2	RW1C	**UART3 TX DMA interrupt status** This register is interrupt status of TX controller. Write 1 clear.
1	RW1C	**UART2 TX DMA interrupt status** This register is interrupt status of TX controller. Write 1 clear.
0	RW1C	**UART1 TX DMA interrupt status** This register is interrupt status of TX controller. Write 1 clear.

Offset: 038h	UDMA038: UART RX DMA interrrupt enable	Init = 0

Bit	R/W	Description
31:14	RO	**reserved(0)**
13	RW	**VUART2 RX DMA interrupt enable**
12	RW	**VUART1 RX DMA interrupt enable**
11	RW	**UART13 RX DMA interrupt enable**
10	RW	**UART12 RX DMA interrupt enable**
9	RW	**UART11 RX DMA interrupt enable**
8	RW	**UART10 RX DMA interrupt enable**
7	RW	**UART9 RX DMA interrupt enable**
6	RW	**UART8 RX DMA interrupt enable**
5	RW	**UART7 RX DMA interrupt enable**
4	RW	**UART6 RX DMA interrupt enable**
3	RW	**UART4 RX DMA interrupt enable**

to next page

from previous page

2	RW	**UART3 RX DMA interrupt enable**
1	RW	**UART2 RX DMA interrupt enable**
0	RW	**UART1 RX DMA interrupt enable**

Note :
The RX DMA interrept will be asserted when the RX DMA buffer status changed from not full to full and becoming full or RX DMA buffer is not empty and time out is enabled.

Offset: 03Ch **UDM0A3C: UART RX DMA interrrupt status** **Init = 0**

Bit	R/W	Description
31:14	RO	**reserved(0)**
13	RW1C	**VUART2 RX DMA interrupt status** This register is interrupt status of RX controller. Write 1 clear.
12	RW1C	**VUART1 RX DMA interrupt status** This register is interrupt status of RX controller. Write 1 clear.
11	RW1C	**UART13 RX DMA interrupt status** This register is interrupt status of RX controller. Write 1 clear.
10	RW1C	**UART12 RX DMA interrupt status** This register is interrupt status of RX controller. Write 1 clear.
9	RW1C	**UART11 RX DMA interrupt status** This register is interrupt status of RX controller. Write 1 clear.
8	RW1C	**UART10 RX DMA interrupt status** This register is interrupt status of RX controller. Write 1 clear.
7	RW1C	**UART9 RX DMA interrupt status** This register is interrupt status of RX controller. Write 1 clear.
6	RW1C	**UART8 RX DMA interrupt status** This register is interrupt status of RX controller. Write 1 clear.
5	RW1C	**UART7 RX DMA interrupt status** This register is interrupt status of RX controller. Write 1 clear.
4	RW1C	**UART6 RX DMA interrupt status** This register is interrupt status of RX controller. Write 1 clear.
3	RW1C	**UART4 RX DMA interrupt status** This register is interrupt status of RX controller. Write 1 clear.
2	RW1C	**UART3 RX DMA interrupt status** This register is interrupt status of RX controller. Write 1 clear.
1	RW1C	**UART2 RX DMA interrupt status** This register is interrupt status of RX controller. Write 1 clear.
0	RW1C	**UART1 RX DMA interrupt status** This register is interrupt status of RX controller. Write 1 clear.

Offset: 040h **UDMA040: UART1 TX read pointer** **Init = x**

Bit	R/W	Description
31:16	RO	**reserved(0)**
15:0	RO	**UART1 TX read pointer** This is a read only register. The pointer is in unit of byte.

Offset: 044h	UDMA044: UART1 TX write pointer	Init = x

Bit	R/W	Description
31:16	RO	**reserved(0)**
15:0	RW	**UART1 TX write pointer** This pointer is controllered by software. The pointer is in unit of byte.

Offset: 048h	UDMA048: UART1 TX buffer base address	Init = x

Bit	R/W	Description
31:2	RW	**UART1 TX buffer base address** This base address of UART1 TX buffer.
1:0	RO	**reserved(0)**

Offset: 04Ch	UDMA04C: UART1 TX control register	Init = 0

Bit	R/W	Description
31:5	RW	**reserved(0)**
4	RW	**UART1 TX DMA time out disable** 0: The TX DMA will not start to transmit until at least a full double word data in TX buffer or time out. 1: Time out is disabled
3:2	RW	**reserved(0)**
1:0	RW	**UART1 TX buffer size** 00: 1KB 01: 4KB 10: 16KB 11: 64KB

Offset: 050h	UDMA050: UART1 RX read pointer	Init = x

Bit	R/W	Description
31:16	RO	**reserved(0)**
15:0	RW	**UART1 RX read pointer** This pointer is controllered by software. The pointer is in unit of byte.

Offset: 054h	UDMA054: UART1 RX write pointer	Init = x

Bit	R/W	Description
31:16	RO	**reserved(0)**
15:0	RO	**UART1 RX write pointer** This is a read only register. The pointer is in unit of byte.

Offset: 058h	UDMA058: UART1 RX buffer base address	Init = x

Bit	R/W	Description
31:2	RW	**UART1 RX buffer base address** This base address of UART1 RX buffer.
1:0	RO	**reserved(0)**

Offset: 05Ch	UDMA05C: UART1 RX control register	Init = 0

Bit	R/W	Description
31:6	RW	**reserved(0)**
5	RW	**UART1 RX DMA full mode** 0: RX buffer is full when RX read pointer equals to RX write pointer minus 1. This is used for ring buffer mode. 1: RX buffer is full when RX write pointer equals to RX write pointer. This is used for single buffer mode.
4	RW	**UART1 RX DMA time out disable** 0: The RX DMA will assert interrupt when RX buffer is full or RX buffer is not empty and time out. 1: Time out is disabled. The RX DMA will assert interrupt only when RX buffer is full.
3:2	RW	**reserved(0)**
1:0	RW	**UART1 RX buffer size** 00: 1KB 01: 4KB 10: 16KB 11: 64KB

Offset: 060h	UDMA060: UART2 TX read pointer	Init = x

Bit	R/W	Description
31:16	RO	**reserved(0)**
15:0	RO	**UART2 TX read pointer** This is a read only register. The pointer is in unit of byte.

Offset: 064h	UDMA064: UART2 TX write pointer	Init = x

Bit	R/W	Description
31:16	RO	**reserved(0)**
15:0	RW	**UART2 TX write pointer** This pointer is controllered by software. The pointer is in unit of byte.

Offset: 068h	UDMA068: UART2 TX buffer base address	Init = x

Bit	R/W	Description
31:2	RW	**UART2 TX buffer base address** This base address of UART2 TX buffer.
1:0	RO	**reserved(0)**

Offset: 06Ch	UDMA06C: UART2 TX control register	Init = 0

Bit	R/W	Description
31:5	RW	**reserved(0)**
4	RW	**UART2 TX DMA time out disable** 0: The TX DMA will not start to transmit until at least a full double word data in TX buffer or time out. 1: Time out is disabled
3:2	RW	**reserved(0)**

to next page

from previous page

1:0	RW	**UART2 TX buffer size** 00: 1KB 01: 4KB 10: 16KB 11: 64KB

Offset: 070h **UDMA070: UART2 RX read pointer** **Init = x**

Bit	R/W	Description
31:16	RO	**reserved(0)**
15:0	RW	**UART2 RX read pointer** This pointer is controllered by software. The pointer is in unit of byte.

Offset: 074h **UDMA074: UART2 RX write pointer** **Init = x**

Bit	R/W	Description
31:16	RO	**reserved(0)**
15:0	RO	**UART2 RX write pointer** This is a read only register. The pointer is in unit of byte.

Offset: 078h **UDMA078: UART2 RX buffer base address** **Init = x**

Bit	R/W	Description
31:2	RW	**UART2 RX buffer base address** This base address of UART2 RX buffer.
1:0	RO	**reserved(0)**

Offset: 07Ch **UDMA07C: UART2 RX control register** **Init = 0**

Bit	R/W	Description
31:6	RW	**reserved(0)**
5	RW	**UART2 RX DMA full mode** 0: RX buffer is full when RX read pointer equals to RX write pointer minus 1. This is used for ring buffer mode. 1: RX buffer is full when RX write pointer equals to RX write pointer. This is used for single buffer mode.
4	RW	**UART2 RX DMA time out disable** 0: The RX DMA will assert interrupt when RX buffer is full or RX buffer is not empty and time out. 1: Time out is disabled. The RX DMA will assert interrupt only when RX buffer is full.
3:2	RW	**reserved(0)**
1:0	RW	**UART2 RX buffer size** 00: 1KB 01: 4KB 10: 16KB 11: 64KB

Offset: 080h	UDMA080: UART3 TX read pointer	Init = x

Bit	R/W	Description
31:16	RO	**reserved(0)**
15:0	RO	**UART3 TX read pointer** This is a read only register. The pointer is in unit of byte.

Offset: 084h	UDMA084: UART3 TX write pointer	Init = x

Bit	R/W	Description
31:16	RO	**reserved(0)**
15:0	RW	**UART3 TX write pointer** This pointer is controllered by software. The pointer is in unit of byte.

Offset: 088h	UDMA088: UART3 TX buffer base address	Init = x

Bit	R/W	Description
31:2	RW	**UART3 TX buffer base address** This base address of UART3 TX buffer.
1:0	RO	**reserved(0)**

Offset: 08Ch	UDMA08C: UART3 TX control register	Init = 0

Bit	R/W	Description
31:5	RW	**reserved(0)**
4	RW	**UART3 TX DMA time out disable** 0: The TX DMA will not start to transmit until at least a full double word data in TX buffer or time out. 1: Time out is disabled
3:2	RW	**reserved(0)**
1:0	RW	**UART3 TX buffer size** 00: 1KB 01: 4KB 10: 16KB 11: 64KB

Offset: 090h	UDMA090: UART3 RX read pointer	Init = x

Bit	R/W	Description
31:16	RO	**reserved(0)**
15:0	RW	**UART3 RX read pointer** This pointer is controllered by software. The pointer is in unit of byte.

Offset: 094h	UDMA094: UART3 RX write pointer	Init = x

Bit	R/W	Description
31:16	RO	**reserved(0)**
15:0	RO	**UART3 RX write pointer** This is a read only register. The pointer is in unit of byte.

Offset: 098h	UDMA098: UART3 RX buffer base address	Init = x

Bit	R/W	Description
31:2	RW	**UART3 RX buffer base address** This base address of UART3 RX buffer.
1:0	RO	**reserved(0)**

Offset: 09Ch	UDMA09C: UART3 RX control register	Init = 0

Bit	R/W	Description
31:6	RW	**reserved(0)**
5	RW	**UART3 RX DMA full mode** 0: RX buffer is full when RX read pointer equals to RX write pointer minus 1. This is used for ring buffer mode. 1: RX buffer is full when RX write pointer equals to RX write pointer. This is used for single buffer mode.
4	RW	**UART3 RX DMA time out disable** 0: The RX DMA will assert interrupt when RX buffer is full or RX buffer is not empty and time out. 1: Time out is disabled. The RX DMA will assert interrupt only when RX buffer is full.
3:2	RW	**reserved(0)**
1:0	RW	**UART3 RX buffer size** 00: 1KB 01: 4KB 10: 16KB 11: 64KB

Offset: 0A0h	UDMA0A0: UART4 TX read pointer	Init = x

Bit	R/W	Description
31:16	RO	**reserved(0)**
15:0	RO	**UART4 TX read pointer** This is a read only register. The pointer is in unit of byte.

Offset: 0A4h	UDMA0A4: UART4 TX write pointer	Init = x

Bit	R/W	Description
31:16	RO	**reserved(0)**
15:0	RW	**UART4 TX write pointer** This pointer is controllered by software. The pointer is in unit of byte.

Offset: 0A8h	UDMA0A8: UART4 TX buffer base address	Init = x

Bit	R/W	Description
31:2	RW	**UART4 TX buffer base address** This base address of UART4 TX buffer.
1:0	RO	**reserved(0)**

Offset: 0ACh	UDMA0AC: UART4 TX control register	Init = 0
Bit	**R/W**	**Description**
31:5	RW	**reserved(0)**
4	RW	**UART4 TX DMA time out disable** 0: The TX DMA will not start to transmit until at least a full double word data in TX buffer or time out. 1: Time out is disabled
3:2	RW	**reserved(0)**
1:0	RW	**UART4 TX buffer size** 00: 1KB 01: 4KB 10: 16KB 11: 64KB

Offset: 0B0h	UDMA0B0: UART4 RX read pointer	Init = x
Bit	**R/W**	**Description**
31:16	RO	**reserved(0)**
15:0	RW	**UART4 RX read pointer** This pointer is controllered by software. The pointer is in unit of byte.

Offset: 0B4h	UDMA0B4: UART4 RX write pointer	Init = x
Bit	**R/W**	**Description**
31:16	RO	**reserved(0)**
15:0	RO	**UART4 RX write pointer** This is a read only register. The pointer is in unit of byte.

Offset: 0B8h	UDMA0B8: UART4 RX buffer base address	Init = x
Bit	**R/W**	**Description**
31:2	RW	**UART4 RX buffer base address** This base address of UART4 RX buffer.
1:0	RO	**reserved(0)**

Offset: 0BCh	UDMA0BC: UART4 RX control register	Init = 0
Bit	**R/W**	**Description**
31:6	RW	**reserved(0)**
5	RW	**UART4 RX DMA full mode** 0: RX buffer is full when RX read pointer equals to RX write pointer minus 1. This is used for ring buffer mode. 1: RX buffer is full when RX write pointer equals to RX write pointer. This is used for single buffer mode.
4	RW	**UART4 RX DMA time out disable** 0: The RX DMA will assert interrupt when RX buffer is full or RX buffer is not empty and time out. 1: Time out is disabled. The RX DMA will assert interrupt only when RX buffer is full.
3:2	RW	**reserved(0)**

to next page

from previous page

1:0	RW	**UART4 RX buffer size** 00: 1KB 01: 4KB 10: 16KB 11: 64KB

Offset: 0C0h **UDMA0C0: UART6 TX read pointer** **Init = x**

Bit	R/W	Description
31:16	RO	**reserved(0)**
15:0	RO	**UART6 TX read pointer** This is a read only register. The pointer is in unit of byte.

Offset: 04Ch **UDMA0C4: UART6 TX write pointer** **Init = x**

Bit	R/W	Description
31:16	RO	**reserved(0)**
15:0	RW	**UART6 TX write pointer** This pointer is controllered by software. The pointer is in unit of byte.

Offset: 0C8h **UDMA0C8: UART6 TX buffer base address** **Init = x**

Bit	R/W	Description
31:2	RW	**UART6 TX buffer base address** This base address of UART6 TX buffer.
1:0	RO	**reserved(0)**

Offset: 0CCh **Init = UDMA0CC: UART6 TX control register**

Bit	R/W	Description
0 31:5	RW	**reserved(0)**
4	RW	**UART6 TX DMA time out disable** 0: The TX DMA will not start to transmit until at least a full double word data in TX buffer or time out. 1: Time out is disabled
3:2	RW	**reserved(0)**
1:0	RW	**UART6 TX buffer size** 00: 1KB 01: 4KB 10: 16KB 11: 64KB

Offset: 0D0h **UDMA0D0: UART6 RX read pointer** **Init = x**

Bit	R/W	Description
31:16	RO	**reserved(0)**
15:0	RW	**UART6 RX read pointer** This pointer is controllered by software. The pointer is in unit of byte.

Offset: 0D4h	UDMA0D4: UART6 RX write pointer	Init = x

Bit	R/W	Description
31:16	RO	**reserved(0)**
15:0	RO	**UART6 RX write pointer** This is a read only register. The pointer is in unit of byte.

Offset: 0D8h	UDMA0D8: UART6 RX buffer base address	Init = x

Bit	R/W	Description
31:2	RW	**UART6 RX buffer base address** This base address of UART6 RX buffer.
1:0	RO	**reserved(0)**

Offset: 0DCh	UDMA0DC: UART6 RX control register	Init = 0

Bit	R/W	Description
31:6	RW	**reserved(0)**
5	RW	**UART6 RX DMA full mode** 0: RX buffer is full when RX read pointer equals to RX write pointer minus 1. This is used for ring buffer mode. 1: RX buffer is full when RX write pointer equals to RX write pointer. This is used for single buffer mode.
4	RW	**UART6 RX DMA time out disable** 0: The RX DMA will assert interrupt when RX buffer is full or RX buffer is not empty and time out. 1: Time out is disabled. The RX DMA will assert interrupt only when RX buffer is full.
3:2	RW	**reserved(0)**
1:0	RW	**UART6 RX buffer size** 00: 1KB 01: 4KB 10: 16KB 11: 64KB

Offset: 0E0h	UDMA0E0: UART7 TX read pointer	Init = x

Bit	R/W	Description
31:16	RO	**reserved(0)**
15:0	RO	**UART7 TX read pointer** This is a read only register. The pointer is in unit of byte.

Offset: 0E4h	UDMA0E4: UART7 TX write pointer	Init = x

Bit	R/W	Description
31:16	RO	**reserved(0)**
15:0	RW	**UART7 TX write pointer** This pointer is controllered by software. The pointer is in unit of byte.

Offset: 0E8h	UDMA0E8: UART7 TX buffer base address	Init = x

Bit	R/W	Description
31:2	RW	**UART7 TX buffer base address** This base address of UART7 TX buffer.
1:0	RO	**reserved(0)**

Offset: 0ECh	UDMA0EC: UART7 TX control register	Init = 0

Bit	R/W	Description
31:5	RW	**reserved(0)**
4	RW	**UART6 TX DMA time out disable** 0: The TX DMA will not start to transmit until at least a full double word data in TX buffer or time out. 1: Time out is disabled
3:2	RW	**reserved(0)**
1:0	RW	**UART6 TX buffer size** 00: 1KB 01: 4KB 10: 16KB 11: 64KB

Offset: 0F0h	UDMA0F0: UART7 RX read pointer	Init = x

Bit	R/W	Description
31:16	RO	**reserved(0)**
15:0	RW	**UART7 RX read pointer** This pointer is controllered by software. The pointer is in unit of byte.

Offset: 0F4h	UDMA0F4: UART7 RX write pointer	Init = x

Bit	R/W	Description
31:16	RO	**reserved(0)**
15:0	RO	**UART7 RX write pointer** This is a read only register. The pointer is in unit of byte.

Offset: 0F8h	UDMA0F8: UART7 RX buffer base address	Init = x

Bit	R/W	Description
31:2	RW	**UART7 RX buffer base address** This base address of UART7 RX buffer.
1:0	RO	**reserved(0)**

Offset: 0FCh	UDMA0FC: UART7 RX control register	Init = 0

Bit	R/W	Description
31:6	RW	**reserved(0)**
5	RW	**UART7 RX DMA full mode** 0: RX buffer is full when RX read pointer equals to RX write pointer minus 1. This is used for ring buffer mode. 1: RX buffer is full when RX write pointer equals to RX write pointer. This is used for single buffer mode.

to next page

from previous page

4	RW	**UART7 RX DMA time out disable** 0: The RX DMA will assert interrupt when RX buffer is full or RX buffer is not empty and time out. 1: Time out is disabled. The RX DMA will assert interrupt only when RX buffer is full.
3:2	RW	**reserved(0)**
1:0	RW	**UART7 RX buffer size** 00: 1KB 01: 4KB 10: 16KB 11: 64KB

Offset: 100h **UDMA100: UART8 TX read pointer** **Init = x**

Bit	R/W	Description
31:16	RO	**reserved(0)**
15:0	RO	**UART8 TX read pointer** This is a read only register. The pointer is in unit of byte.

Offset: 104h **UDMA104: UART8 TX write pointer** **Init = x**

Bit	R/W	Description
31:16	RO	**reserved(0)**
15:0	RW	**UART8 TX write pointer** This pointer is controllered by software. The pointer is in unit of byte.

Offset: 108h **UDMA108: UART8 TX buffer base address** **Init = x**

Bit	R/W	Description
31:2	RW	**UART8 TX buffer base address** This base address of UART8 TX buffer.
1:0	RO	**reserved(0)**

Offset: 10Ch **UDMA10C: UART8 TX control register** **Init = 0**

Bit	R/W	Description
31:5	RW	**reserved(0)**
4	RW	**UART6 TX DMA time out disable** 0: The TX DMA will not start to transmit until at least a full double word data in TX buffer or time out. 1: Time out is disabled
3:2	RW	**reserved(0)**
1:0	RW	**UART6 TX buffer size** 00: 1KB 01: 4KB 10: 16KB 11: 64KB

Offset: 110h	UDMA110: UART8 RX read pointer	Init = x

Bit	R/W	Description
31:16	RO	**reserved(0)**
15:0	RW	**UART8 RX read pointer** This pointer is controllered by software. The pointer is in unit of byte.

Offset: 114h	UDMA114: UART8 RX write pointer	Init = x

Bit	R/W	Description
31:16	RO	**reserved(0)**
15:0	RO	**UART8 RX write pointer** This is a read only register. The pointer is in unit of byte.

Offset: 118h	UDMA118: UART8 RX buffer base address	Init = x

Bit	R/W	Description
31:2	RW	**UART8 RX buffer base address** This base address of UART8 RX buffer.
1:0	RO	**reserved(0)**

Offset: 11Ch	Init = UDMA11C: UART8 RX control register

Bit	R/W	Description
0 31:6	RW	**reserved(0)**
5	RW	**UART8 RX DMA full mode** 0: RX buffer is full when RX read pointer equals to RX write pointer minus 1. This is used for ring buffer mode. 1: RX buffer is full when RX write pointer equals to RX write pointer. This is used for single buffer mode.
4	RW	**UART8 RX DMA time out disable** 0: The RX DMA will assert interrupt when RX buffer is full or RX buffer is not empty and time out. 1: Time out is disabled. The RX DMA will assert interrupt only when RX buffer is full.
3:2	RW	**reserved(0)**
1:0	RW	**UART8 RX buffer size** 00: 1KB 01: 4KB 10: 16KB 11: 64KB

Offset: 120h	UDMA120: UART9 TX read pointer	Init = x

Bit	R/W	Description
31:16	RO	**reserved(0)**
15:0	RO	**UART9 TX read pointer** This is a read only register. The pointer is in unit of byte.

Offset: 124h	UDMA124: UART9 TX write pointer	Init = x

Bit	R/W	Description
31:16	RO	**reserved(0)**
15:0	RW	**UART9 TX write pointer** This pointer is controllered by software. The pointer is in unit of byte.

Offset: 128h	UDMA128: UART9 TX buffer base address	Init = x

Bit	R/W	Description
31:2	RW	**UART9 TX buffer base address** This base address of UART9 TX buffer.
1:0	RO	**reserved(0)**

Offset: 12Ch	UDMA12C: UART9 TX control register	Init = 0

Bit	R/W	Description
31:5	RW	**reserved(0)**
4	RW	**UART6 TX DMA time out disable** 0: The TX DMA will not start to transmit until at least a full double word data in TX buffer or time out. 1: Time out is disabled
3:2	RW	**reserved(0)**
1:0	RW	**UART6 TX buffer size** 00: 1KB 01: 4KB 10: 16KB 11: 64KB

Offset: 130h	UDMA130: UART9 RX read pointer	Init = x

Bit	R/W	Description
31:16	RO	**reserved(0)**
15:0	RW	**UART9 RX read pointer** This pointer is controllered by software. The pointer is in unit of byte.

Offset: 134h	UDMA134: UART9 RX write pointer	Init = x

Bit	R/W	Description
31:16	RO	**reserved(0)**
15:0	RO	**UART9 RX write pointer** This is a read only register. The pointer is in unit of byte.

Offset: 138h	UDMA138: UART9 RX buffer base address	Init = x

Bit	R/W	Description
31:2	RW	**UART9 RX buffer base address** This base address of UART9 RX buffer.
1:0	RO	**reserved(0)**

Offset: 13Ch		Init = UDMA13C: UART9 RX control register
Bit	**R/W**	**Description**
0 31:6	RW	**reserved(0)**
5	RW	**UART9 RX DMA full mode** 0: RX buffer is full when RX read pointer equals to RX write pointer minus 1. This is used for ring buffer mode. 1: RX buffer is full when RX write pointer equals to RX write pointer. This is used for single buffer mode.
4	RW	**UART9 RX DMA time out disable** 0: The RX DMA will assert interrupt when RX buffer is full or RX buffer is not empty and time out. 1: Time out is disabled. The RX DMA will assert interrupt only when RX buffer is full.
3:2	RW	**reserved(0)**
1:0	RW	**UART9 RX buffer size** 00: 1KB 01: 4KB 10: 16KB 11: 64KB

Offset: 140h	UDMA140: UART10 TX read pointer	Init = x
Bit	**R/W**	**Description**
31:16	RO	**reserved(0)**
15:0	RO	**UART10 TX read pointer** This is a read only register. The pointer is in unit of byte.

Offset: 144h	UDMA144: UART10 TX write pointer	Init = x
Bit	**R/W**	**Description**
31:16	RO	**reserved(0)**
15:0	RW	**UART10 TX write pointer** This pointer is controllered by software. The pointer is in unit of byte.

Offset: 148h	UDMA148: UART10 TX buffer base address	Init = x
Bit	**R/W**	**Description**
31:2	RW	**UART10 TX buffer base address** This base address of UART10 TX buffer.
1:0	RO	**reserved(0)**

Offset: 14Ch	UDMA14C: UART10 TX control register	Init = 0
Bit	**R/W**	**Description**
31:5	RW	**reserved(0)**
4	RW	**UART6 TX DMA time out disable** 0: The TX DMA will not start to transmit until at least a full double word data in TX buffer or time out. 1: Time out is disabled
3:2	RW	**reserved(0)**

to next page

from previous page

1:0	RW	**UART6 TX buffer size** 00: 1KB 01: 4KB 10: 16KB 11: 64KB

Offset: 150h **UDMA150: UART10 RX read pointer** **Init = x**

Bit	R/W	Description
31:16	RO	**reserved(0)**
15:0	RW	**UART10 RX read pointer** This pointer is controllered by software. The pointer is in unit of byte.

Offset: 154h **UDMA154: UART10 RX write pointer** **Init = x**

Bit	R/W	Description
31:16	RO	**reserved(0)**
15:0	RO	**UART10 RX write pointer** This is a read only register. The pointer is in unit of byte.

Offset: 158h **UDMA158: UART10 RX buffer base address** **Init = x**

Bit	R/W	Description
31:2	RW	**UART10 RX buffer base address** This base address of UART10 RX buffer.
1:0	RO	**reserved(0)**

Offset: 15Ch **UDMA15C: UART10 RX control register** **Init = 0**

Bit	R/W	Description
31:6	RW	**reserved(0)**
5	RW	**UART10 RX DMA full mode** 0: RX buffer is full when RX read pointer equals to RX write pointer minus 1. This is used for ring buffer mode. 1: RX buffer is full when RX write pointer equals to RX write pointer. This is used for single buffer mode.
4	RW	**UART10 RX DMA time out disable** 0: The RX DMA will assert interrupt when RX buffer is full or RX buffer is not empty and time out. 1: Time out is disabled. The RX DMA will assert interrupt only when RX buffer is full.
3:2	RW	**reserved(0)**
1:0	RW	**UART10 RX buffer size** 00: 1KB 01: 4KB 10: 16KB 11: 64KB

Offset: 160h	UDMA160: UART11 TX read pointer	Init = x

Bit	R/W	Description
31:16	RO	**reserved(0)**
15:0	RO	**UART11 TX read pointer** This is a read only register. The pointer is in unit of byte.

Offset: 164h	UDMA164: UART11 TX write pointer	Init = x

Bit	R/W	Description
31:16	RO	**reserved(0)**
15:0	RW	**UART11 TX write pointer** This pointer is controllered by software. The pointer is in unit of byte.

Offset: 168h	UDMA168: UART11 TX buffer base address	Init = x

Bit	R/W	Description
31:2	RW	**UART11 TX buffer base address** This base address of UART11 TX buffer.
1:0	RO	**reserved(0)**

Offset: 16Ch	UDMA16C: UART11 TX control register	Init = 0

Bit	R/W	Description
31:5	RW	**reserved(0)**
4	RW	**UART6 TX DMA time out disable** 0: The TX DMA will not start to transmit until at least a full double word data in TX buffer or time out. 1: Time out is disabled
3:2	RW	**reserved(0)**
1:0	RW	**UART6 TX buffer size** 00: 1KB 01: 4KB 10: 16KB 11: 64KB

Offset: 170h	UDMA170: UART11 RX read pointer	Init = x

Bit	R/W	Description
31:16	RO	**reserved(0)**
15:0	RW	**UART11 RX read pointer** This pointer is controllered by software. The pointer is in unit of byte.

Offset: 174h	UDMA174: UART11 RX write pointer	Init = x

Bit	R/W	Description
31:16	RO	**reserved(0)**
15:0	RO	**UART11 RX write pointer** This is a read only register. The pointer is in unit of byte.

Offset: 178h	UDMA178: UART11 RX buffer base address	Init = x

Bit	R/W	Description
31:2	RW	**UART11 RX buffer base address** This base address of UART11 RX buffer.
1:0	RO	**reserved(0)**

Offset: 17Ch	UDMA17C: UART11 RX control register	Init = 0

Bit	R/W	Description
31:6	RW	**reserved(0)**
5	RW	**UART11 RX DMA full mode** 0: RX buffer is full when RX read pointer equals to RX write pointer minus 1. This is used for ring buffer mode. 1: RX buffer is full when RX write pointer equals to RX write pointer. This is used for single buffer mode.
4	RW	**UART11 RX DMA time out disable** 0: The RX DMA will assert interrupt when RX buffer is full or RX buffer is not empty and time out. 1: Time out is disabled. The RX DMA will assert interrupt only when RX buffer is full.
3:2	RW	**reserved(0)**
1:0	RW	**UART11 RX buffer size** 00: 1KB 01: 4KB 10: 16KB 11: 64KB

Offset: 180h	UDMA180: UART12 TX read pointer	Init = x

Bit	R/W	Description
31:16	RO	**reserved(0)**
15:0	RO	**UART12 TX read pointer** This is a read only register. The pointer is in unit of byte.

Offset: 184h	UDMA184: UART12 TX write pointer	Init = x

Bit	R/W	Description
31:16	RO	**reserved(0)**
15:0	RW	**UART12 TX write pointer** This pointer is controllered by software. The pointer is in unit of byte.

Offset: 188h	UDMA188: UART12 TX buffer base address	Init = x

Bit	R/W	Description
31:2	RW	**UART12 TX buffer base address** This base address of UART12 TX buffer.
1:0	RO	**reserved(0)**

Offset: 18Ch	UDMA18C: UART12 TX control register	Init = 0

Bit	R/W	Description
31:5	RW	**reserved(0)**
4	RW	**UART6 TX DMA time out disable** 0: The TX DMA will not start to transmit until at least a full double word data in TX buffer or time out. 1: Time out is disabled
3:2	RW	**reserved(0)**
1:0	RW	**UART6 TX buffer size** 00: 1KB 01: 4KB 10: 16KB 11: 64KB

Offset: 190h	UDMA190: UART12 RX read pointer	Init = x

Bit	R/W	Description
31:16	RO	**reserved(0)**
15:0	RW	**UART12 RX read pointer** This pointer is controllered by software. The pointer is in unit of byte.

Offset: 194h	UDMA194: UART12 RX write pointer	Init = x

Bit	R/W	Description
31:16	RO	**reserved(0)**
15:0	RO	**UART12 RX write pointer** This is a read only register. The pointer is in unit of byte.

Offset: 198h	UDMA198: UART12 RX buffer base address	Init = x

Bit	R/W	Description
31:2	RW	**UART12 RX buffer base address** This base address of UART12 RX buffer.
1:0	RO	**reserved(0)**

Offset: 19Ch	UDMA19C: UART12 RX control register	Init = 0

Bit	R/W	Description
31:6	RW	**reserved(0)**
5	RW	**UART12 RX DMA full mode** 0: RX buffer is full when RX read pointer equals to RX write pointer minus 1. This is used for ring buffer mode. 1: RX buffer is full when RX write pointer equals to RX write pointer. This is used for single buffer mode.
4	RW	**UART12 RX DMA time out disable** 0: The RX DMA will assert interrupt when RX buffer is full or RX buffer is not empty and time out. 1: Time out is disabled. The RX DMA will assert interrupt only when RX buffer is full.
3:2	RW	**reserved(0)**

to next page

from previous page

1:0	RW	**UART12 RX buffer size** 00: 1KB 01: 4KB 10: 16KB 11: 64KB

Offset: 1A0h **UDMA1A0: UART13 TX read pointer** **Init = x**

Bit	R/W	Description
31:16	RO	**reserved(0)**
15:0	RO	**UART13 TX read pointer** This is a read only register. The pointer is in unit of byte.

Offset: 1A4h **UDMA1A4: UART13 TX write pointer** **Init = x**

Bit	R/W	Description
31:16	RO	**reserved(0)**
15:0	RW	**UART13 TX write pointer** This pointer is controllered by software. The pointer is in unit of byte.

Offset: 1A8h **UDMA1A8: UART13 TX buffer base address** **Init = x**

Bit	R/W	Description
31:2	RW	**UART13 TX buffer base address** This base address of UART13 TX buffer.
1:0	RO	**reserved(0)**

Offset: 1ACh **UDMA1AC: UART13 TX control register** **Init = 0**

Bit	R/W	Description
31:5	RW	**reserved(0)**
4	RW	**UART6 TX DMA time out disable** 0: The TX DMA will not start to transmit until at least a full double word data in TX buffer or time out. 1: Time out is disabled
3:2	RW	**reserved(0)**
1:0	RW	**UART6 TX buffer size** 00: 1KB 01: 4KB 10: 16KB 11: 64KB

Offset: 1B0h **UDMA1B0: UART13 RX read pointer** **Init = x**

Bit	R/W	Description
31:16	RO	**reserved(0)**
15:0	RW	**UART13 RX read pointer** This pointer is controllered by software. The pointer is in unit of byte.

Offset: 1B4h	UDMA1B4: UART13 RX write pointer	Init = x

Bit	R/W	Description
31:16	RO	**reserved(0)**
15:0	RO	**UART13 RX write pointer** This is a read only register. The pointer is in unit of byte.

Offset: 1B8h	UDMA1B8: UART13 RX buffer base address	Init = x

Bit	R/W	Description
31:2	RW	**UART13 RX buffer base address** This base address of UART13 RX buffer.
1:0	RO	**reserved(0)**

Offset: 1BCh	UDMA1BC: UART13 RX control register	Init = 0

Bit	R/W	Description
31:6	RW	**reserved(0)**
5	RW	**UART13 RX DMA full mode** 0: RX buffer is full when RX read pointer equals to RX write pointer minus 1. This is used for ring buffer mode. 1: RX buffer is full when RX write pointer equals to RX write pointer. This is used for single buffer mode.
4	RW	**UART13 RX DMA time out disable** 0: The RX DMA will assert interrupt when RX buffer is full or RX buffer is not empty and time out. 1: Time out is disabled. The RX DMA will assert interrupt only when RX buffer is full.
3:2	RW	**reserved(0)**
1:0	RW	**UART13 RX buffer size** 00: 1KB 01: 4KB 10: 16KB 11: 64KB

Offset: 1C0h	UDMA1C0: VUART1 TX read pointer	Init = x

Bit	R/W	Description
31:24	RO	**reserved(0)**
23:0	RO	**VUART1 TX read pointer** This is a read only register. The pointer is in unit of byte.

Offset: 1C4h	UDMA1C4: VUART1 TX write pointer	Init = x

Bit	R/W	Description
31:24	RO	**reserved(0)**
23:0	RW	**VUART1 TX write pointer** This pointer is controllered by software. The pointer is in unit of byte.

Offset: 1C8h	UDMA1C8: VUART1 TX buffer base address	Init = x

Bit	R/W	Description
31:2	RW	**VUART1 TX buffer base address** This base address of VUART1 TX buffer.
1:0	RO	**reserved(0)**

Offset: 1CCh	UDMA1CC: VUART1 TX control register	Init = 0

Bit	R/W	Description
31:5	RW	**reserved(0)**
4	RW	**VUART1 TX DMA time out disable** 0: The TX DMA will not start to transmit until at least a full double word data in TX buffer or time out. 1: Time out is disabled
3:0	RW	**VUART1 TX buffer size** 0000: 1KB 0001: 4KB 0010: 16KB 0011: 64KB 0100: 128KB 0101: 256KB 0110: 512KB 0111: 1024KB 1000: 2048KB 1001: 4096KB 1010: 8192KB 1011: 16384KB others: reserved

Offset: 1D0h	UDMA1D0: VUART1 RX read pointer	Init = x

Bit	R/W	Description
31:16	RO	**reserved(0)**
15:0	RW	**VUART1 RX read pointer** This pointer is controllered by software. The pointer is in unit of byte.

Offset: 1D4h	UDMA1D4: VUART1 RX write pointer	Init = x

Bit	R/W	Description
31:16	RO	**reserved(0)**
15:0	RO	**VUART1 RX write pointer** This is a read only register. The pointer is in unit of byte.

Offset: 1D8h	UDMA1D8: VUART1 RX buffer base address	Init = x

Bit	R/W	Description
31:2	RW	**VUART1 RX buffer base address** This base address of VUART1 RX buffer.
1:0	RO	**reserved(0)**

Offset: 1DCh	UDMA1DC: VUART1 RX control register	Init = 0

Bit	R/W	Description
31:6	RW	**reserved(0)**
5	RW	**VUART1 RX DMA full mode** 0: RX buffer is full when RX read pointer equals to RX write pointer minus 1. This is used for ring buffer mode. 1: RX buffer is full when RX write pointer equals to RX write pointer. This is used for single buffer mode.
4	RW	**VUART1 RX DMA time out disable** 0: The RX DMA will assert interrupt when RX buffer is full or RX buffer is not empty and time out. 1: Time out is disabled. The RX DMA will assert interrupt only when RX buffer is full.
3:0	RW	**VUART1 RX buffer size** 0000: 1KB 0001: 4KB 0010: 16KB 0011: 64KB 0100: 128KB 0101: 256KB 0110: 512KB 0111: 1024KB 1000: 2048KB 1001: 4096KB 1010: 8192KB 1011: 16384KB others: reserved

Offset: 1E0h	UDMA1E0: VUART2 TX read pointer	Init = x

Bit	R/W	Description
31:24	RO	**reserved(0)**
23:0	RO	**VUART2 TX read pointer** This is a read only register. The pointer is in unit of byte.

Offset: 1E4h	UDMA1E4: VUART2 TX write pointer	Init = x

Bit	R/W	Description
31:24	RO	**reserved(0)**
23:0	RW	**VUART2 TX write pointer** This pointer is controllered by software. The pointer is in unit of byte.

Offset: 1E8h	UDMA1E8: VUART2 TX buffer base address	Init = x

Bit	R/W	Description
31:2	RW	**VUART2 TX buffer base address** This base address of VUART2 TX buffer.
1:0	RO	**reserved(0)**

Offset: 1ECh	UDMA1EC: VUART2 TX control register	Init = 0

Bit	R/W	Description
31:5	RW	**reserved(0)**
4	RW	**VUART2 TX DMA time out disable** 0: The TX DMA will not start to transmit until at least a full double word data in TX buffer or time out. 1: Time out is disabled
3:0	RW	**VUART2 TX buffer size** 0000: 1KB 0001: 4KB 0010: 16KB 0011: 64KB 0100: 128KB 0101: 256KB 0110: 512KB 0111: 1024KB 1000: 2048KB 1001: 4096KB 1010: 8192KB 1011: 16384KB others: reserved

Offset: 1F0h	UDMA1F0: VUART2 RX read pointer	Init = x

Bit	R/W	Description
31:24	RO	**reserved(0)**
23:0	RW	**VUART2 RX read pointer** This pointer is controllered by software. The pointer is in unit of byte.

Offset: 1F4h	UDMA1F4: VUART2 RX write pointer	Init = x

Bit	R/W	Description
31:24	RO	**reserved(0)**
23:0	RO	**VUART2 RX write pointer** This is a read only register. The pointer is in unit of byte.

Offset: 1F8h	UDMA1F8: VUART2 RX buffer base address	Init = x

Bit	R/W	Description
31:2	RW	**VUART2 RX buffer base address** This base address of VUART2 RX buffer.
1:0	RO	**reserved(0)**

Offset: 1FCh	UDMA1FC: VUART2 RX control register	Init = 0

Bit	R/W	Description
31:6	RW	**reserved(0)**
5	RW	**VUART2 RX DMA full mode** 0: RX buffer is full when RX read pointer equals to RX write pointer minus 1. This is used for ring buffer mode. 1: RX buffer is full when RX write pointer equals to RX write pointer. This is used for single buffer mode.

to next page

from previous page

4	RW	**VUART2 RX DMA time out disable** 0: The RX DMA will assert interrupt when RX buffer is full or RX buffer is not empty and time out. 1: Time out is disabled. The RX DMA will assert interrupt only when RX buffer is full.
3:0	RW	**VUART2 RX buffer size** 0000: 1KB 0001: 4KB 0010: 16KB 0011: 64KB 0100: 128KB 0101: 256KB 0110: 512KB 0111: 1024KB 1000: 2048KB 1001: 4096KB 1010: 8192KB 1011: 16384KB others: reserved

46 Watchdog Timer (WDT)

46.1 Overview

Watchdog Timer (WDT) includes 8 sets of 32-bit decrement counters, based on 1 MHz clock.
Watchdog Timer (WDT) is designed to prevent system deadlock. In general, the WDT must be restarted before WDT timeout. Whenever timeout occurs, WDT can program to generate 5 types of signals:

- **ARM reset signal**: to reset ARM CPU only
- **SOC reset signal**: to reset SOC part function
- **System reset signal**: to reset full chip
- **Interrupt signal**: to interrupt CPU
- **External signal**: to reset external controller (Only WDT1 - WDT4 supports)

Below are the change items compared to AST2500.

- Add 5 set watchdog counter.
- Remove 2nd boot and SPI address toggle control, move into flash controller.
- Add watchdog reset tolerance for SOC reset.
- Reset mask is re-defined.
- Add Software mode reset control in each set of watchdog controller.

Base address of WDT1 = 0x1E78_5000
Base address of WDT2 = 0x1E78_5040
Base address of WDT3 = 0x1E78_5080
Base address of WDT4 = 0x1E78_50C0
Base address of WDT5 = 0x1E78_5100
Base address of WDT6 = 0x1E78_5140
Base address of WDT7 = 0x1E78_5180
Base address of WDT8 = 0x1E78_51C0
Physical address = (Base address of WDT) + Offset

WDT00: WDTn Counter Status Register
WDT04: WDTn Counter Reload Value Register
WDT08: WDTn Counter Restart Register
WDT0C: WDTn Control Register
WDT10: WDTn Timeout Status Register
WDT14: WDTn Clear Timeout Status Register
WDT18: WDTn Reset Width Register
WDT1C: WDTn Reset Mask Register #1
WDT20: WDTn Reset Mask Register #2
WDT24: WDTn Software Mode Reset Control Register
WDT28: WDTn Software Mode Reset Mask Register #1
WDT2C: WDTn Software Mode Reset Mask Register #2
WDT30: WDTn Funtion Disable Control Register

46.2 Features

- Directly connected to APB bus
- Watchdog function
- Built-in 8 sets of 32-bit WDT modules

- Generate either interrupt or reset after counting down to zero (programmable)
- Generate 3 types of reset pulse (programmable) to reset SOC part or full chip.
- Support Software mode reset control.

46.3 Registers

Offset: 00h	WDT00: WDTn Counter Status Register	Init = 0x014FB180

Bit	R/W	Description
31:0	RO	**Counter status** This register stores the current status of counter. After power on reset or timeout, this register is loaded with value 0x14FB180 (22 seconds). When the programmer writes 0x4755 to Restart register, Reload register will be loaded into this register. Counter starts to decrease count once WDT0C[0] enable bit is set. If watchdog timer is disabled, it will hold the value.

Offset: 04h	WDT04: WDTn Counter Reload Value Register	Init = 0x014FB180

Bit	R/W	Description
31:0	RW	**Counter reload value register** Reload register contains value which will be loaded into WDT00 register. When reset or restart, Reload value will be automatically loaded into WDT00 register. The timing unit is 1us.

Offset: 08h	WDT08: WDTn Counter Restart Register	Init = 0

Bit	R/W	Description
31:16	RO	**Reserved (0)**
15:0	WT	**Restart register** Restart register is used to avoid system deadlock. If the 0x4755 value is written into this register, the Reload register will be loaded into WDT00 register and WDT00 register restarts to decrease if WDT0C[0] register is set.

Offset: 0Ch	WDT0C: WDTn Control Register	Init = 0x0010

Bit	R/W	Description
31:10	RW	**Pre-timeout value for interrupt generation** When watchdog timer match below equation, issue the interrupt. WDT00[31:10] == pre-timeout && WDT00[9:0] == 0
9:7	RO	**Reserved (0)**
6:5	RW	**Reset system mode** 00: SOC system (gated by reset mask registers) 01: Full chip 1x: CPU/FMC only, just reboot firmware, no any other IPs will be reset.
4	RW	**Enable WDT(n) to be reset by SOC reset** 0: WDT(n) can not be reset by SOC reset, only SRST# or Full chip reset can reset it. 1: WDT(n) can be reset by SOC reset.

to next page

from previous page

3	RW	**wdt_ext: External signal enable after timeout** This signal is connected to an external output pin. If timeout occurs, and this bit is enabled, an active high output pulse will be generated. The generated pulse width is controller by WDT18. 0: disable 1: enable
2	RW	**wdt_intr: Generate interrupt when timer count matches to the pre-timeout setting** 0: disable 1: enable
1	RW	**Reset system after timeout** 0: disable 1: enable At this mode, WDT will be cleared and disabled after timeout occurs. WDT10[0] must be cleared before enable this bit.
0	RW	**WDT enable signal** 0: disable 1: enable

Offset: 10h **WDT10: WDTn Timeout Status Register** **Init = 0**

Bit	R/W	Description
31:24	RO	**Reserved (0)**
23:20	RO	**Watchdog event counter : ARM reset** This field reports the count of the WDT timeout event happened.
19:16	RO	**Watchdog event counter : Full reset** This field reports the count of the WDT timeout event happened.
15:8	RO	**Watchdog event counter : SOC reset** This field reports the count of the WDT timeout event happened.
7:3	RO	**Reserved (0)**
2	RO	**Indicate interrupt** Interrupt register is a record when watchdog counter match the pre-timeout setting.
1	RO	**Reserved (0)**
0	RO	**Indicate timeout** Timeout register is a record. If the WDT had ever occurred timeout condition, this bit will be set to '1' when timeout occurred, and clear to '0' if WDT0C[1] = 1. 0: timeout never occur 1: timeout occur

Offset: 14h **WDT14: WDTn Clear Timeout Status Register** **Init = 0**

Bit	R/W	Description
31:8	RO	**Reserved (0)**
7:1	WT	**Clear timeout counter status** Write 0x3B value into this field to clear WDT counter register.
0	W1T	**Clear timeout, boot code selection and interrupt status** Write '1' value into this bit to clear Timeout and Boot Code Source selection status register.

Offset: 18h	WDT18: WDTn Reset Width Register	Init = 0xFF

Bit	R/W	Description
31	RW	**Reset pulse polarity selection** 0: active low output 1: active high output To set this bit to value '1', write bit[31:24] = 0xA5. To set this bit to value '0', write bit[31:24] = 0x5A. For others value, this bit will keep old value without change.
30	RW	**Reset pulse output driving type** 0: open-drain 1: push-pull To set this bit to value '1', write bit[31:24] = 0xA8. To set this bit to value '0', write bit[31:24] = 0x8A. For others value, this bit will keep old value without change.
29	RW	**Reset pulse generation trigger selection** 0: trigger by timeout 1: trigger by pre-timeout To set this bit to value '1', write bit[31:24] = 0xA3. To set this bit to value '0', write bit[31:24] = 0x3A. For others value, this bit will keep old value without change.
28:20	RO	**Reserved (0)**
19:0	RW	**Reset width** This register decides the asserting duration of wdt_ext and wdt_rstarm signal. The default value is 0xFF. It means the default asserting duration of wdt_ext and wdt_rstarm is 256 us. The value will be decremented by hardware when pulse generating, and return to default value after pulse end.

Offset: 1Ch	WDT1C: WDTn Reset Mask Register #1	Init = 0x030F1FF1

Bit	R/W	Description
31:26	RO	**Reserved**
25	RW	**Enable reset RVAS controller**
24	RW	**Enable reset GPIO #1 controller**
23	RW	**Enable reset XDMA #2 controller**
22	RW	**Enable reset XDMA #1 controller**
21	RW	**Enable reset MCTP #2 controller**
20	RW	**Enable reset MCTP #1 controller**
19	RW	**Enable reset JTAG #1 master controller**
18	RW	**Enable reset SD/SDIO #1 controller**
17	RW	**Enable reset MAC#2 controller**
16	RW	**Enable reset MAC#1 controller**
15	RW	**Enable reset GP MCU controller**
14	RW	**Enable reset DP MCU controller**
13	RW	**Enable reset DP controller**
12	RW	**Enable reset HAC engine**
11	RW	**Enable reset Video engine**
10	RW	**Enable reset CRT mode 2D engine**
9	RW	**Enable reset Graphics CRT controller**

to next page

from previous page

8	RW	**Enable reset USB1.1 UHCI Host controller**
7	RW	**Enable reset USB portB Host/Dev controller**
6	RW	**Enable reset USB portA Host/Hub controller**
5	RW	**Enable reset Coprocessor**
4	RW	**Enable reset SOC controller** SOC controller includes: WDT, RTC, Timer, UART, SRAM.
3	RW	**Enable reset SLI bridge**
2	RW	**Enable reset AHB bridges**
1	RW	**Enable reset SDRAM controller**
0	RW	**Enable reset ARM and related controllers**
Note : This register controls the IPs to be reset by the watchdog event. The watchdog event indicates the Watchdog_reset_SOC generated by WDTn controller.		

Offset: 20h **WDT20: WDTn Reset Mask Register #2** **Init = 0x3FFFFF1**

Bit	R/W	Description
31:28	RO	**Reserved**
27	RW	**Enable reset I2CS controller**
26	RW	**Enable reset eSPI controller**
25:24	RO	**Reserved**
23	RW	**Enable reset I3C bus6 controller**
22	RW	**Enable reset I3C bus5 controller**
21	RW	**Enable reset I3C bus4 controller**
20	RW	**Enable reset I3C bus3 controller**
19	RW	**Enable reset I3C bus2 controller**
18	RW	**Enable reset I3C bus1 controller**
17	RW	**Enable reset I3C Global controller**
16	RW	**Enable reset I2C controller**
15	RW	**Enable reset FSI controller**
14	RW	**Enable reset ADC controller**
13	RW	**Enable reset PWM controller**
12	RW	**Enable reset PECI controller**
11	RW	**Enable reset LPC controller**
10	RW	**Enable reset MDC/MDIO controller**
9	RW	**Enable reset GPIO #2 controller**
8	RW	**Enable reset JTAG #2 master controller**
7	RW	**Enable reset SD/SDIO #2 controller**
6	RW	**Enable reset MAC#4 controller**
5	RW	**Enable reset MAC#3 controller**
4	RW	**Enable reset SOC controller** SOC controller includes: WDT, UART, BSRAM.
3	RW	**Enable reset SLI2 bridge**
2	RW	**Enable reset AHB2 bridges**

to next page

from previous page

1	RW	**Enable reset SPI1/SPI2 controller**
0	RW	**Enable reset ARM related controllers** This bit must be set together with WDT1C[0].

Note :
This register controls the IPs to be reset by the watchdog event. The watchdog event indicates the Watchdog_reset_SOC generated by WDTn controller.

Offset: 24h **WDT24: WDTn Software Mode Reset Control Register** **Init = 0x0**

Bit	R/W	Description
31:8	RO	**Reserved**
7:4	RWT	**Watchdog event counter** This field reports the count of the Software mode reset event. Write 0xDEADDEAD to clear this counter value.
3:1	RO	**Reserved**
0	RWT	**Enable trigger software mode reset** Write 0xAEEDF123 to fire the reset generation. After reset, this bit will be cleared to 0 automatically. Software mode reset only support SOC reset mode.

Offset: 28h **WDT28: WDTn Software Mode Reset Mask Register #1)** **Init = 0x0**

Bit	R/W	Description
31:28	RO	**Reserved**
27	RW	**Enable reset I2CS controller**
26	RW	**Enable reset eSPI controller**
25:24	RO	**Reserved**
25	RW	**Enable reset RVAS controller**
24	RW	**Enable reset GPIO #1 controller**
23	RW	**Enable reset XDMA #2 controller**
22	RW	**Enable reset XDMA #1 controller**
21	RW	**Enable reset MCTP #2 controller**
20	RW	**Enable reset MCTP #1 controller**
19	RW	**Enable reset JTAG #1 master controller**
18	RW	**Enable reset SD/SDIO #1 controller**
17	RW	**Enable reset MAC#2 controller**
16	RW	**Enable reset MAC#1 controller**
15	RW	**Enable reset GP MCU controller**
14	RW	**Enable reset DP MCU controller**
13	RW	**Enable reset DP controller**
12	RW	**Enable reset HAC engine**
11	RW	**Enable reset Video engine**
10	RW	**Enable reset CRT mode 2D engine**
9	RW	**Enable reset Graphics CRT controller**
8	RW	**Enable reset USB1.1 UHCI Host controller**
7	RW	**Enable reset USB portB Host/Dev controller**

to next page

from previous page

6	RW	**Enable reset USB portA Host/Hub controller**
5	RW	**Enable reset Coprocessor**
4	RW	**Enable reset SOC controller** SOC controller includes: WDT, RTC, Timer, UART, SRAM.
3	RW	**Enable reset SLI bridge**
2	RW	**Enable reset AHB bridges**
1	RW	**Enable reset SDRAM controller**
0	RW	**Enable reset ARM and related controllers**

Offset: 2Ch **WDT2C: WDTn Software Mode Reset Mask Register #2** **Init = 0x0**

Bit	R/W	Description
31:26	RO	**Reserved**
25	RW	**Enable reset I3C bus8 controller**
24	RW	**Enable reset I3C bus7 controller**
23	RW	**Enable reset I3C bus6 controller**
22	RW	**Enable reset I3C bus5 controller**
21	RW	**Enable reset I3C bus4 controller**
20	RW	**Enable reset I3C bus3 controller**
19	RW	**Enable reset I3C bus2 controller**
18	RW	**Enable reset I3C bus1 controller**
17	RW	**Enable reset I3C Global controller**
16	RW	**Enable reset I2C controller**
15	RW	**Enable reset FSI controller**
14	RW	**Enable reset ADC controller**
13	RW	**Enable reset PWM controller**
12	RW	**Enable reset PECI controller**
11	RW	**Enable reset LPC controller**
10	RW	**Enable reset MDC/MDIO controller**
9	RW	**Enable reset GPIO #2 controller**
8	RW	**Enable reset JTAG #2 master controller**
7	RW	**Enable reset SD/SDIO #2 controller**
6	RW	**Enable reset MAC#4 controller**
5	RW	**Enable reset MAC#3 controller**
4	RW	**Enable reset SOC controller** SOC controller includes: WDT, UART, BSRAM.
3	RW	**Enable reset SLI2 bridge**
2	RW	**Enable reset AHB2 bridges**
1	RW	**Enable reset SPI1/SPI2 controller**
0	RW	**Enable reset ARM related controllers** This bit must be set together with WDT28[0].

Offset: 30h	WDT30: WDTn Funtion Disable Control Register	Init = 0x0

Bit	R/W	Description
5	RW1S	**Disable watchdog Software mode reset**
4	RW1S	**Disable watchdog interrupt generation**
3	RW1S	**Disable watchdog CPU/FMC reset mode**
2	RW1S	**Disable watchdog Full reset mode**
1	RW1S	**Disable watchdog SOC reset mode**
0	RW1S	**Disable full watchdog function**

Note :
This register is write '1' possible only and can't be cleared until next SRST# reset.

46.4 Operation

46.4.1 Enable watchdog reset

1. disable watchdog timer
2. set Reload register
3. write 0x4755 to Restart register
4. set WDT0C[7:1] bit
5. enable watchdog timer

46.4.2 Enable watchdog pulse output

1. wdt_ext output can be programmed as open-drain or push-pull type
2. confirm the external pin state is pulled high or low at the inactive state (power up default state)
3. set watchdog reset pulse output active polarity at WDT18
   (a) set the output active level
   (b) set the output driving mode
4. enable multi-function pin selection to WDTRST pin mode
5. do below sequence to initialize watchdog reset pulse output
   (a) disable watchdog timer
   (b) clear WDT10 bit[0] by writing WDT14 = 0x1
   (c) set Reload register
   (d) write 0x4755 to Restart register
   (e) set WDT18 to the desired output pulse width
   (f) set WDT0C[3] = 1 and WDT0C[7:1] to the required value
   (g) enable watchdog timer

46.4.3 Enable watchdog interrupt output

1. disable watchdog timer
2. set Reload register
3. write 0x4755 to Restart register
4. set WDT0C[2:1] = "10" and WDT0C[31:10] to the required time for generating interrupt
5. enable watchdog timer
6. when interrupt triggered, clear WDT10 bit[2] by writing WDT14 = 0x1 to clear interrupt

47 PWM & Fan Tacho Controller (PWM/TACHO)

47.1 Overview

Base Address of PWM & Fan Tach Controller = 0x1E61_0000
Physical address of register = (Base address of PWM & Fan Tach Controller) + Offset

PTCR000: PWM0 General Register
PTCR004: PWM0 Duty Cycle Register
PTCR008: TACH0 General Register
PTCR00C: TACH0 Status Register
PTCR010: PWM1 General Register
PTCR014: PWM1 Duty Cycle Register
PTCR018: TACH1 General Register
PTCR01C: TACH1 Status Register
PTCR020: PWM2 General Register
PTCR024: PWM2 Duty Cycle Register
PTCR028: TACH2 General Register
PTCR02C: TACH2 Status Register
PTCR030: PWM3 General Register
PTCR034: PWM3 Duty Cycle Register
PTCR038: TACH3 General Register
PTCR03C: TACH3 Status Register
PTCR040: PWM4 General Register
PTCR044: PWM4 Duty Cycle Register
PTCR048: TACH4 General Register
PTCR04C: TACH4 Status Register
PTCR050: PWM5 General Register
PTCR054: PWM5 Duty Cycle Register
PTCR058: TACH5 General Register
PTCR05C: TACH5 Status Register
PTCR060: PWM6 General Register
PTCR064: PWM6 Duty Cycle Register
PTCR068: TACH6 General Register
PTCR06C: TACH6 Status Register
PTCR070: PWM7 General Register
PTCR074: PWM7 Duty Cycle Register
PTCR078: TACH7 General Register
PTCR07C: TACH7 Status Register
PTCR080: PWM8 General Register
PTCR084: PWM8 Duty Cycle Register
PTCR088: TACH8 General Register
PTCR08C: TACH8 Status Register
PTCR090: PWM9 General Register
PTCR094: PWM9 Duty Cycle Register
PTCR098: TACH9 General Register
PTCR09C: TACH9 Status Register
PTCR0A0: PWMA General Register
PTCR0A4: PWMA Duty Cycle Register
PTCR0A8: TACHA General Register
PTCR0AC: TACHA Status Register
PTCR0B0: PWMB General Register
PTCR0B4: PWMB Duty Cycle Register
PTCR0B8: TACHB General Register
PTCR0BC: TACHB Status Register
PTCR0C0: PWMC General Register

PTCR0C4: PWMC Duty Cycle Register
PTCR0C8: TACHC General Register
PTCR0CC: TACHC Status Register
PTCR0D0: PWMD General Register
PTCR0D4: PWMD Duty Cycle Register
PTCR0D8: TACHD General Register
PTCR0DC: TACHD Status Register
PTCR0E0: PWME General Register
PTCR0E4: PWME Duty Cycle Register
PTCR0E8: TACHE General Register
PTCR0EC: TACHE Status Register
PTCR0F0: PWMF General Register
PTCR0F4: PWMF Duty Cycle Register
PTCR0F8: TACHF General Register
PTCR0FC: TACHF Status Register
PTCR100: PWM G100 Register
PTCR104: PWM G104 Register
PTCR108: PWM G108 Register
PTCR10C: TACH Status Register

47.2 Features

PWM and Fan Tachometer Controller

- Support 16 PWM outputs and 16 fan tachometer inputs
- Support PWM frequency range from 780KHz to 24Hz
- Duty cycle from 0 to 100% with 1/256 resolution incremental
- Support fan tachometer frequency range from 1 RPM to 20K (180K) RPM
- Shared with GPIO pins

47.3 Registers : Base Address = 0x1E61:0000

Offset: 000h	PTCR000: PWM0 General Register	Init = 0x00000000
Bit	**R/W**	**Description**
31:20	RW	**reserved**
19	RW	**load selection of duty as WDT** 0: falling 1: rising
18	RW	**enable PWM duty load as WDT**
17	RW	**disable PWM duty instant change**
16	RW	**enable PWM clock**
15	RW	**output PWM level**
14	RW	**inverse PWM pin**
13	RW	**enable open-drain**
12	RW	**enable PWM pin**

to next page

from previous page

11:8	RW	**PWM clock division H bit [3:0]** 0: divide 1 1: divide 2 2: divide 4 3: divide 8 ... F: divide 32768
7 :0	RW	**PWM clock division L bit [7:0]** 00: divide 1 01: divide 2 02: divide 3 03: divide 4 ... FF: divide 256

Offset: 004h **PTCR004: PWM0 Duty Cycle Register** **Init = 0x00000000**

Bit	R/W	Description
31:24	RW	**PWM period bit [7:0]**
23:16	RW	**PWM rising/falling point bit [7:0] as WDT**
15:8	RW	**PWM falling point bit [7:0]**
7 :0	RW	**PWM rising point bit [7:0]**

Offset: 008h **PTCR008: TACH0 General Register** **Init = 0x00000000**

Bit	R/W	Description
31	RW	**enable tacho interrupt**
30	RW	**reserved; inverse tacho limit comparison**
29	RW	**enable tacho loopback mode**
28	RW	**enable tacho**
27:26	RW	**tacho de-bounce** 0: 3 tacho clock 1: 2 tacho clock 2: 1 tacho clock 3: none
25:24	RW	**tacho edge** 0: F-to-F (default) 1: R-to-R 2: both 3: reserved
23:20	RW	**tacho clock division T bit [3:0]** 0: divide 1 1: divide 4 2: divide 16 3: divide 64 ... B: divide 4194304 others: reserved
19 :0	RW	**tacho threshold bit [19:0]**

Offset: 00Ch	PTCR00C: TACH0 Status Register	Init = 0x00000000

Bit	R/W	Description
31	RW1C	**interrupt status and clear**
30:26	RO	**reserved**
25	RO	**pwm_oeN (output control) status**
24	RO	**pwm_out (output) status**
23	RO	**tacho debounce input**
22	RO	**tacho raw input**
21	RO	**tacho value updated since last read**
20	RO	**tacho full measurement**
19 :0	RO	**tacho value bit [19:0]**

Offset: 010h	PTCR010: PWM1 General Register	Init = 0x00000000

Bit	R/W	Description
31:20	RW	**reserved**
19	RW	**load selection of duty as WDT** 0: falling 1: rising
18	RW	**enable PWM duty load as WDT**
17	RW	**disable PWM duty instant change**
16	RW	**enable PWM clock**
15	RW	**output PWM level**
14	RW	**inverse PWM pin**
13	RW	**enable open-drain**
12	RW	**enable PWM pin**
11:8	RW	**PWM clock division H bit [3:0]** 0: divide 1 1: divide 2 2: divide 4 3: divide 8 ... F: divide 32768
7 :0	RW	**PWM clock division L bit [7:0]** 00: divide 1 01: divide 2 02: divide 3 03: divide 4 ... FF: divide 256

Offset: 0E0h	PTCR0E0: PWME General Register	Init = 0x00000000

Bit	R/W	Description
31:20	RW	**reserved**
19	RW	**load selection of duty as WDT** 0: falling 1: rising

to next page

from previous page

18	RW	**enable PWM duty load as WDT**
17	RW	**disable PWM duty instant change**
16	RW	**enable PWM clock**
15	RW	**output PWM level**
14	RW	**inverse PWM pin**
13	RW	**enable open-drain**
12	RW	**enable PWM pin**
11:8	RW	**PWM clock division H bit [3:0]** 0: divide 1 1: divide 2 2: divide 4 3: divide 8 ... F: divide 32768
7 :0	RW	**PWM clock division L bit [7:0]** 00: divide 1 01: divide 2 02: divide 3 03: divide 4 ... FF: divide 256

Offset: 0F0h **PTCR0F0: PWMF General Register** **Init = 0x00000000**

Bit	R/W	Description
31:21	RW	**reserved**
20	RW	**disable PWM clock instant change, e.g., heart beat rate**
19	RW	**load selection of duty as WDT** 0: falling 1: rising
18	RW	**enable PWM duty load as WDT**
17	RW	**disable PWM duty instant change**
16	RW	**enable PWM clock**
15	RW	**output PWM level**
14	RW	**inverse PWM pin**
13	RW	**enable open-drain**
12	RW	**enable PWM pin**
11:8	RW	**PWM clock division H bit [3:0]** 0: divide 1 1: divide 2 2: divide 4 3: divide 8 ... F: divide 32768

to next page

from previous page

7 :0	RW	**PWM clock division L bit [7:0]** 00: divide 1 01: divide 2 02: divide 3 03: divide 4 ... FF: divide 256

Offset: 100h	**PTCR100: PWM G100 Register**	**Init = 0x00000000**
Bit	**R/W**	**Description**
31:16	RO	**reserved**
15:0	RW	**PWM clock gating bit [15:0]**

Offset: 104h	**PTCR104: PWM G104 Register**	**Init = 0x00000000**
Bit	**R/W**	**Description**
31:16	RO	**reserved**
15:0	RW	**PWM duty gating bit [15:0]**

Offset: 108h	**PTCR108: PWM G108 Register**	**Init = 0x00000000**
Bit	**R/W**	**Description**
31:0	RO	**reserved**

Offset: 10Ch	**PTCR10C: TACH Status Register**	**Init = 0x00000000**
Bit	**R/W**	**Description**
31:16	RO	**reserved**
15: 0	RO	**interrupt status bit [15:0]**

†† RPM = (AHB_CLK*60) / [PulsePR * (TachoValue + 1) * TachoClk-Div]

where PulsePR means the number of tacho Pulse Per Revolution
PulsePR is fan dependent. A general parameter of PulsePR is 2.

48 Virtual UART (VUART)

48.1 Overview

AST2600 integrates two Virtual UART modules providing virtual serial communication capabilities between host CPU and ARM CPU. Virtual UART is equipped with two sets of registers compatible with the industry defector standard - 16550 UART.

One set is for host CPU; the other set is for ARM CPU. Host CPU and ARM CPU can communicate with each other like there is a physical UART link between them, but the related data transfer actually is just through pure register read/write transfers in the chip. The base address for host CPU to access UART registers through LPC bus can be programmed by ARM CPU by the extended related registers (VxUART28 and VxUART2C)

In other words, Virtual UART is a single module with dual heads. One side is exposed to Host over the eSPI/LPC (see the Host side registers below), the other side is exposed to the BMC (see the Slave side register below). When the host writes to V1UART00 (Host), the data goes into an internal 16 byte FIFO, the same FIFO can be read from the V1UART00 (Slave) from the BMC side. Similarly, BMC writes goes to another FIFO that can be read from the host side by reading the V1UART00 (Host). The rest of the control registers are only required when you take the Byte data and shift it out serially to a connector but in this implementation we never do serial shift internally. The data stays as byte, so whatever programming in host/BMC side to other UART control registers are just ignored.

Base Address of Virtual 1 UART = 0x1E78_7000
Register Address of Virtual 1 UART = (Base Address of V1UART) + Offset

The following registers can be access by host CPU through LPC bus.

V1UART00 (Host): Receiving Buffer Register (Read, DLAB = 0)
V1UART00 (Host): Transmit Holding Register (Write, DLAB = 0)
V1UART00 (Host): Divisor Latch Low Register (Read/Write, DLAB = 1)
V1UART04 (Host): Interrupt Enable Register (Read/Write, DLAB = 0)
V1UART04 (Host): Divisor Latch High Register (Read/Write, DLAB = 1)
V1UART08 (Host): FIFO Control Register
V1UART0C (Host): Line Control Register
V1UART10 (Host): Modem Control Register
V1UART14 (Host): Line Status Register
V1UART18 (Host): Modem Status Register
V1UART1C (Host): Scratch Register

The following registers can be access by ARM CPU through APB bus.

V1UART00 (Slave): Receiving Buffer Register (Read, DLAB = 0)
V1UART00 (Slave): Transmit Holding Register (Write, DLAB = 0)
V1UART00 (Slave): Divisor Latch Low Register (Read/Write, DLAB = 1)
V1UART04 (Slave): Interrupt Enable Register (Read/Write, DLAB = 0)
V1UART04 (Slave): Divisor Latch High Register (Read/Write, DLAB = 1)
V1UART08 (Slave): FIFO Control Register
V1UART0C (Slave): Line Control Register
V1UART10 (Slave): Modem Control Register
V1UART14 (Slave): Line Status Register
V1UART18 (Slave): Modem Status Register
V1UART1C (Slave): Scratch Register
V1UART20 (Slave): General Control Register A

V1UART24 (Slave): General Control Register B
V1UART28 (Slave): VUART Address Register L
V1UART2C (Slave): VUART Address Register H
V1UART30 (Slave): General Control Register E
V1UART34 (Slave): General Control Register F
V1UART38 (Slave): General Control Register G
V1UART3C (Slave): General Control Register H

Base Address of Virtual 2 UART = 0x1E78_8000
Register Address of Virtual 2 UART = (Base Address of V2UART) + Offset

The following registers can be access by host CPU through LPC bus.

V2UART00 (Host): Receiving Buffer Register (Read, DLAB = 0)
V2UART00 (Host): Transmit Holding Register (Write, DLAB = 0)
V2UART00 (Host): Divisor Latch Low Register (Read/Write, DLAB = 1)
V2UART04 (Host): Interrupt Enable Register (Read/Write, DLAB = 0)
V2UART04 (Host): Divisor Latch High Register (Read/Write, DLAB = 1)
V2UART08 (Host): FIFO Control Register
V2UART0C (Host): Line Control Register
V2UART10 (Host): Modem Control Register
V2UART14 (Host): Line Status Register
V2UART18 (Host): Modem Status Register
V2UART1C (Host): Scratch Register

The following registers can be access by ARM CPU through APB bus.

V2UART00 (Slave): Receiving Buffer Register (Read, DLAB = 0)
V2UART00 (Slave): Transmit Holding Register (Write, DLAB = 0)
V2UART00 (Slave): Divisor Latch Low Register (Read/Write, DLAB = 1)
V2UART04 (Slave): Interrupt Enable Register (Read/Write, DLAB = 0)
V2UART04 (Slave): Divisor Latch High Register (Read/Write, DLAB = 1)
V2UART08 (Slave): FIFO Control Register
V2UART0C (Slave): Line Control Register
V2UART10 (Slave): Modem Control Register
V2UART14 (Slave): Line Status Register
V2UART18 (Slave): Modem Status Register
V2UART1C (Slave): Scratch Register
V2UART20 (Slave): General Control Register A
V2UART24 (Slave): General Control Register B
V2UART28 (Slave): VUART Address Register L
V2UART2C (Slave): VUART Address Register H
V2UART30 (Slave): General Control Register E
V2UART34 (Slave): General Control Register F
V2UART38 (Slave): General Control Register G
V2UART3C (Slave): General Control Register H

48.2 Features

- Directly connected to both APB bus and LPC Bus
- Support two Virtual UART interfaces
- Separate transmit & receive FIFO buffer (16x8) to reduce CPU overhead
- Programmable base address for host CPU to access UART registers through LPC bus

48.3 V1UART Registers : Base Address = 0x1E78:7000

Offset: 00h	V1UART00 (Host)	Init = X
Bit	**R/W**	**Description**
31:8	RO	**Reserved (0)**
RBR: Receiving Buffer Register (DLAB = 0)		
7 :0	RO	**Receiving Buffer Register** The RBR contains the data byte received on the serial input port (sin). The data in this register is valid only if the Data Ready (DR) bit in the Line status Register (UARTLSR) is set. When the FIFOs are programmed ON, this register accesses the head of the receive FIFO. If the receive FIFO is full and this register is not read before the next data character arrives, then the data already in the FIFO will be preserved but any incoming data will be lost. An overrun error will also occur.
THR: Transmit Holding Register (DLAB = 0)		
7:0	WO	**Transmit Holding Register** The THR contains data to be transmitted on the serial output port (sout). Data can be written to the THR any time that the THR Empty (THRE) bit of the Line Status Register (UARTLSR) is set. If FIFOs are enabled and THRE is set, 16 number of characters of data may be written to the THR before the FIFO is full. Any attempt to write data when the FIFO is full results in the write data being lost.
DLL: Divisor Latch Low Register (DLAB = 1)		
7 :0	RW	**Divisor Latch Low Register** This DLL register is designed for software compatible. The actual output baud rate is equal to LPC clock frequency, 33MHz.

Offset: 00h	V1UART00 (Slave)	Init = X
Bit	**R/W**	**Description**
31:8	RO	**Reserved (0)**
RBR: Receiving Buffer Register (DLAB = 0)		
7 :0	RO	**Receiving Buffer Register** The RBR contains the data byte received on the serial input port (sin). The data in this register is valid only if the Data Ready (DR) bit in the Line status Register (UARTLSR) is set. When the FIFOs are programmed ON, this register accesses the head of the receive FIFO. If the receive FIFO is full and this register is not read before the next data character arrives, then the data already in the FIFO will be preserved but any incoming data will be lost. An overrun error will also occur.
THR: Transmit Holding Register (DLAB = 0)		
7:0	WO	**Transmit Holding Register** The THR contains data to be transmitted on the serial output port (sout). Data can be written to the THR any time that the THR Empty (THRE) bit of the Line Status Register (UARTLSR) is set. If FIFOs are enabled and THRE is set, 16 number of characters of data may be written to the THR before the FIFO is full. Any attempt to write data when the FIFO is full results in the write data being lost.
DLL: Divisor Latch Low Register (DLAB = 1 and V1UART34[2] = 0)		

to next page

from previous page

7 :0	RW	**Divisor Latch Low Register** This DLL register is designed for software compatible. The actual output baud rate is equal to LPC clock frequency, 33MHz.
DLL: Divisor Latch Low Register (DLAB = 1 and V1UART34[2] = 1)		
7 :0	RO	**Divisor Latch Low Register (Host)** The slave (BMC) can read the DLL of the Host by this register.

Offset: 04h	**V1UART04 (Host)**	**Init = 0**
Bit	**R/W**	**Description**
31:8	RO	**Reserved (0)**
IER: Interrupt Enable Register (DLAB = 0)		
7	RW	**Enable FIFO 1/2 full THRE Interrupt Mode if V1UART34[6]=1** 0: Disable FIFO 1/2 full THRE interrupt mode 1: Enable FIFO 1/2 full THRE interrupt mode if V1UART34[6]=1
6 :4	RO	**Reserved (0)**
3	RW	**Enable Modem Status Interrupt** 0: Disable interrupt 1: Enable interrupt
2	RW	**Enable Receiver Line Status Interrupt** 0: Disable interrupt 1: Enable interrupt
1	RW	**Enable Transmitter Holding Register Empty Interrupt** 0: Disable interrupt 1: Enable interrupt
0	RW	**Enable Received Data Available Interrupt** 0: Disable interrupt 1: Enable interrupt
DLH: Divisor Latch High Register (DLAB = 1)		
7 :0	RW	**Divisor latch (High)** This DLH register is designed for software compatible. The actual output baud rate is equal to LPC clock frequency, 33MHz.

Offset: 04h	**V1UART04 (Slave)**	**Init = 0**
Bit	**R/W**	**Description**
31:8	RO	**Reserved (0)**
IER: Interrupt Enable Register (DLAB = 0)		
7 :4	RO	**Reserved (0)**
3	RW	**Enable Modem Status Interrupt** 0: Disable interrupt 1: Enable interrupt
2	RW	**Enable Receiver Line Status Interrupt** 0: Disable interrupt 1: Enable interrupt
1	RW	**Enable Transmitter Holding Register Empty Interrupt** 0: Disable interrupt 1: Enable interrupt

to next page

from previous page

0	RW	**Enable Received Data Available Interrupt** 0: Disable interrupt 1: Enable interrupt
DLH: Divisor Latch High Register (DLAB = 1 and V1UART34[2] = 0)		
7 :0	RW	**Divisor latch (High)** This DLH register is designed for software compatible. The actual output baud rate is equal to LPC clock frequency, 33MHz.
DLH: Divisor Latch High Register (DLAB = 1 and V1UART34[2] = 1)		
7 :0	RO	**Divisor latch (High) (Host)** The slave (BMC) can read the DLH of the Host by this register.

Offset: 08h **V1UART08 (Host): (IIR) Interrupt Identity Register** **Init = 0xC1**

Bit	R/W	Description
31:4	RO	**Reserved (0)**
3:1	RO	**Interrupt Decoding Table** The content of this register can be used to identify the source of the current interrupt based on the following: 000: Modem Status Changed 001: THR empty 010: Received Data Available 011: Receiver Status 110: Character Time Out For more information about Interrupt Identity, see the following Table for detailed description.
0	RO	Indicates that an interrupt is pending when its logic 0. When its 1, no interrupt is pending.
Note : The IIR enables the programmer to retrieve what is the current highest priority pending interrupt.		

V1UART Interrupt Type Decoding

Bit [3:1]	Priority	Interrupt Type	Interrupt Source	Interrupt Clear Control
011	Highest	Receiver line status	Overrun or Break Interrupt.	Reading the Line Status Register. IER[2](Host)=0
010	2nd	Received Data available	FIFO ON: RX FIFO trigger level reached	FIFO ON: FIFO drops below the trigger level IER[0](Host)=0 FCR[1](Host)=1
110	3rd	Character Timeout indication	Theres at least 1 character in the FIFO but no character has been input to the FIFO or read from it for 1/512, 1/256, 1/128 or 1/64 second.	Reading the RBR. IER[0](Host)=0 FCR[1](Host)=1 VUART34[1](Slave)=1
001	4th	Transmitter Holding register empty	Transmitter Holding register empty	Reading the IIR or writing into THR
000	5th	Modem status	nCTS (Clear to send), nDSR (data set ready)	Reading the MSR.

Offset: 08h	V1UART08 (Host): (FCR) FIFO Control Register	Init = 0x01

Bit	R/W	Description
31:8	RO	**Reserved (0)**
7:6	W	**Define the Receiver FIFO Interrupt trigger level.** 00: 1 byte received 01: 4 bytes received 10: 8 bytes received 11: 14 bytes received
5:3	RO	**Reserved (0)**
2	W	**Transmit FIFO Reset** Writing 1 to this bit clears the Transmitter FIFO and resets its logic.
1	W	**Receive FIFO Reset** Writing 1 to this bit clears the Receiver FIFO and resets its logic.
0	W	**Enable UART FIFO** 0: Disable FIFO 1: Enable FIFO The value of this register is always logic '1'.

Note :
The FCR allows selection of the FIFO trigger level (the number of bytes in FIFO required to enable the Received Data Available interrupt). In addition, the FIFOs can be cleared using this register.

Offset: 08h	V1UART08 (Slave): (IIR) Interrupt Identity Register	Init = 0xC1

Bit	R/W	Description
31:4	RO	**Reserved (0)**
3:1	RO	**Interrupt Decoding Table** The content of this register can be used to identify the source of the current interrupt based on the following: 000: Modem Status Changed 001: THR empty 010: Received Data Available 011: Receiver Status 110: Character Time Out For more information about Interrupt Identity, see the following Table for detailed description.
0	RO	Indicates that an interrupt is pending when its logic 0. When its 1, no interrupt is pending.

Note :
The IIR enables the programmer to retrieve what is the current highest priority pending interrupt.

V1UART Interrupt Type Decoding				
Bit [3:1]	Priority	Interrupt Type	Interrupt Source	Interrupt Clear Control
011	Highest	Receiver line status	Overrun or Break Interrupt.	Reading the Line Status Register. IER[2](Slave)=0 FCR[1](Slave)=1 if Overrun
010	2nd	Received Data available	FIFO ON: RX FIFO trigger level reached	FIFO ON: FIFO drops below the trigger level IER[0](Slave)=0 FCR[1](Slave)=1
110	3rd	Character Timeout indication	Theres at least 1 character in the FIFO but no character has been input to the FIFO or read from it for 1/512, 1/256, 1/128 or 1/64 second.	Reading the RBR. IER[0](Slave)=0 FCR[1](Slave)=1 V1UART34[0](Slave)=1
001	4th	Transmitter Holding register empty	Transmitter Holding register empty	Reading the IIR or writing into THR FCR[2](Slave)=1
000	5th	Modem status	nCTS (Clear to send), nDSR (data set ready)	Reading the MSR.

Offset: 08h	V1UART08 (Slave): (FCR) FIFO Control Register	Init = 0x01
Bit	**R/W**	**Description**
31:8	RO	**Reserved (0)**
7:6	W	**Define the Receiver FIFO Interrupt trigger level.** 00: 1 byte received 01: 4 bytes received 10: 8 bytes received 11: 14 bytes received
5:3	RO	**Reserved (0)**
2	W	**Transmit FIFO Reset** Writing 1 to this bit clears the Transmitter FIFO and resets its logic.
1	W	**Receive FIFO Reset** Writing 1 to this bit clears the Receiver FIFO and resets its logic.
0	W	**Enable UART FIFO** 0: Disable FIFO 1: Enable FIFO The value of this register is always logic '1'.
Note : The FCR allows selection of the FIFO trigger level (the number of bytes in FIFO required to enable the Received Data Available interrupt). In addition, the FIFOs can be cleared using this register.		

Offset: 0Ch	V1UART0C (Host): (LCR) Line Control Register	Init = 0x03
Bit	**R/W**	**Description**
31:8	RO	**Reserved (0)**

to next page

from previous page

7	RW	**DLAB: Divisor latch access bit** 0: The normal registers are accessed. 1: The divisor latches can be accessed. Setting this bit will enable the reading and writing of the Divisor Latch register (DLL and DLH) to set the baud rate of the UART. This bit must be cleared after initial baud rate setup in order to access other registers.
6	RW	**Break Control bit.** 0: break is disabled. 1: break event is transmitted to the Slave side.
5:2	RW	**Reserved**
1:0	RW	**Select number of bits per character** 00: 5 bits. 01: 6 bits. 10: 7 bits. 11: 8 bits.

Offset: 0Ch **V1UART0C (Slave): (LCR) Line Control Register** **Init = 0x03**

Bit	R/W	Description
31:8	RO	**Reserved (0)**
7	RW	**DLAB: Divisor latch access bit** 0: The normal registers are accessed. 1: The divisor latches can be accessed. Setting this bit will enable the reading and writing of the Divisor Latch register (DLL and DLH) to set the baud rate of the UART. This bit must be cleared after initial baud rate setup in order to access other registers.
6	RW	**Break Control bit.** 0: break is disabled. 1: break event is transmitted to the Host side.
5:2	RW	**Reserved**
1:0	RW	**Select number of bits per character** 00: 5 bits. 01: 6 bits. 10: 7 bits. 11: 8 bits.

Offset: 10h **V1UART10 (Host): (MCR) Modem Control Register** **Init = 0**

Bit	R/W	Description
31:8	RO	**Reserved (0)**
7	RO	**Transmit FIFO full.** 0: transmit FIFO not full. 1: transmit FIFO full.
6:5	RO	**Reserved (0)**

to next page

from previous page

4	RW	**Loopback mode.** 0: normal operation. 1: loopback mode. The signal of the transmitter shift register is internally connected to the input of the receiver shift register. The following connections are made: nDTR → nDSR nRTS → nCTS Out1 → nRI Out2 → nDCD
3	RW	**Out2.** In loopback mode, connected to Data Carrier Detect(nDCD) input.
2	RW	**Out1.** In loopback mode, connected to Ring Indicator (nRI)signal input.
1	RW	**Request To Send (nRTS) signal control.** 0: nRTS is '1' 1: nRTS is '0'
0	RW	**Data Terminal Ready (nDTR) signal control.** 0: nDTR is '1' 1: nDTR is '0'

Offset: 10h **V1UART10 (Slave): (MCR) Modem Control Register** **Init = 0**

Bit	R/W	Description
31:5	RO	**Reserved (0)**
4	RW	**Loopback mode.** 0: normal operation. 1: loopback mode. The signal of the transmitter shift register is internally connected to the input of the receiver shift register. The following connections are made: nDTR → nDSR nRTS → nCTS Out1 → nRI Out2 → nDCD
3	RW	**Out2.** In loopback mode, connected to Data Carrier Detect(nDCD) input.
2	RW	**Out1.** In loopback mode, connected to Ring Indicator (nRI)signal input.
1	RW	**Request To Send (nRTS) signal control.** 0: nRTS is '1' 1: nRTS is '0'
0	RW	**Data Terminal Ready (nDTR) signal control.** 0: nDTR is '1' 1: nDTR is '0'

Offset: 14h **V1UART14 (Host): (LSR) Line Status Register** **Init = 0x60**

Bit	R/W	Description
31:7	RO	**Reserved (0)**
6	RO	**Transmitter empty** 1: The THR or FIFO and the Transmitter Shift Register are both empty. 0: Otherwise.

to next page

from previous page

5	RO	**Transmitter holding register empty** 1: The THR or FIFO and the Transmitter Shift Register are both empty. This also causes a THRE Interrupt to occur, if the THRE Interrupt is enabled. 0: Otherwise.
4	RO	**Break interrupt** 1: The Break Control bit of Line Control Register on the opposite side is set. 0: No break condition in the current character.
3:2	RO	**Reserved (0)**
1	RO	**Overrun error** 1: In the FIFO mode, an overrun error occurs when the FIFO is full and a new character arrives at the receiver. 0: No overrun state.
0	RO	**Data ready** 1: The receiver contains at least one character in the RBR or the receiver FIFO. 0: The RBR is read or the receiver FIFO is empty.

Offset: 14h **V1UART14 (Slave): (LSR) Line Status Register** **Init = 0x60**

Bit	R/W	Description
31:7	RO	**Reserved (0)**
6	RO	**Transmitter empty** 1: The THR or FIFO and the Transmitter Shift Register are both empty. 0: Otherwise.
5	RO	**Transmitter holding register empty** 1: The THR or FIFO and the Transmitter Shift Register are both empty. This also causes a THRE Interrupt to occur, if the THRE Interrupt is enabled. 0: Otherwise.
4	RO	**Break interrupt** 1: The Break Control bit of Line Control Register on the opposite side is set. 0: No break condition in the current character.
3:2	RO	**Reserved (0)**
1	RO	**Overrun error** 1: In the FIFO mode, an overrun error occurs when the FIFO is full and a new character arrives at the receiver. 0: No overrun state.
0	RO	**Data ready** 1: The receiver contains at least one character in the RBR or the receiver FIFO. 0: The RBR is read or the receiver FIFO is empty.

Offset: 18h **V1UART18 (Host): (MSR) Modem Status Register** **Init = X**

Bit	R/W	Description
31:8	RO	**Reserved (0)**
7	RO	**Complement of the nDSR input or equals to Out2(MCR[3]) in loopback mode.**
6	RO	**Out1(MCR[2]) in loopback mode.**
5	RO	**Complement of the nDSR input or equals to nDTR(MCR[0]) in loopback mode.**
4	RO	**Complement of the nCTS input or equals to nRTS(MCR[1]) in loopback mode.**
3	RO	**Delta Data Carrier Detect (DDCD) indicator** 1: The nDSR line has changed its state since the last time the CPU read the MSR. In loopback mode, DDCD reflects changes on MCR bit 3 (Out2)

to next page

from previous page

2	RO	**Trailing Edge of Ring Indicator (TERI) detector.** In loopback mode, TERI reflects when MCR bit 2 (Out1) has changed state from a high to a low.
1	RO	**Delta Data Set Ready (DDSR) indicator** 1: The nDSR line has changed its state since the last time the CPU read the MSR. In loopback mode, DDSR reflects changes on MCR bit 0 (DTR)
0	RO	**Delta Clear To Send (DCTS) indicator** 1: The nCTS line has changed its state since the last time the CPU read the MSR. In loopback mode, DCTS reflects changes on MCR bit 1 (RTS)

Offset: 18h **V1UART18 (Slave): (MSR) Modem Status Register** **Init = X**

Bit	R/W	Description
31:8	RO	**Reserved (0)**
7	RO	**Complement of the nDSR input or equals to Out2(MCR[3]) in loopback mode.**
6	RO	**Out1(MCR[2]) in loopback mode.**
5	RO	**Complement of the nDSR input or equals to nDTR(MCR[0]) in loopback mode.**
4	RO	**Complement of the nCTS input or equals to nRTS(MCR[1]) in loopback mode.**
3	RO	**Delta Data Carrier Detect (DDCD) indicator** 1: The nDSR line has changed its state since the last time the CPU read the MSR.
2	RO	**Reserved (0)**
1	RO	**Delta Data Set Ready (DDSR) indicator** 1: The nDSR line has changed its state since the last time the CPU read the MSR.
0	RO	**Delta Clear To Send (DCTS) indicator** 1: The nCTS line has changed its state since the last time the CPU read the MSR.

Offset: 1Ch **V1UART1C (Host): (SCR) Scratch Register** **Init = X**

Bit	R/W	Description
31:8	RO	**Reserved (0)**
7 :0	RW	**Scratch bits** This register can be used as a temporary storage, no specific definition.

Offset: 1Ch **V1UART1C (Slave): (SCR) Scratch Register** **Init = X**

Bit	R/W	Description
31:8	RO	**Reserved (0)**
7 :0	RW	**Scratch bits (V1UART34[2] = 0)** This register can be used as a temporary storage, no specific definition.
7 :0	RO	**Scratch bits (V1UART34[2] = 1)** The slave (BMC) can read the SCR of the Host by this register.

Offset: 20h **V1UART20 (Slave): General Control Register A** **Init = 0b00x0_xx00**

Bit	R/W	Description
31:8	RO	**Reserved (0)**
7 :6	RO	**Status of host-side Receiver FIFO Trigger**

to next page

from previous page

5	RW	**Disable Host-Tx-discard mode** 0: Enable Host-Tx-discard mode Slave HW (BMC) will throw data away automatically when Host Tx FIFO is not empty. Background: If Host Tx FIFO is full, and BMC FW or DMA engine cant read data from BMC Rx FIFO (e.g., BMC FW is busy and can't handle BMC Rx FIFO interrupts, V1UART08 (Slave) [3:1] = 010 or 110), Host SW will not see V1UART14 (Host) [6:5] = 11, and that may cause Host SW stops the next Host Tx data transit. 1: Disable Host-Tx-discard mode
4	RO	**Status of host-side loopback mode**
3 :2	RW	**Slave-side timeout time width selection bit [1:0]** 00: 1/3600 second if V1UART38[1]=0 01: 1/7200 second if V1UART38[1]=0 10: 1/14400 second if V1UART38[1]=0 11: 1/28800 second if V1UART38[1]=0 00: 512*LCLK if V1UART38[1]=1 01: 256*LCLK if V1UART38[1]=1 10: 128*LCLK if V1UART38[1]=1 11: 64*LCLK if V1UART38[1]=1
1	RW	**SIRQ polarity** 0: The output is high impedance when host interrupt has been cleared (or eSPI mode). The output is low level when host interrupt has been set. 1: The output is low level when host interrupt has been cleared. The output is high impedance when host interrupt has been set. This bit should be properly assigned even if V1UART20[0] is low.
0	RW	**Enable virtual UART** 0: Disable 1: Enable
Note : This register is defined for ARM CPU only.		

Offset: 24h **V1UART24 (Slave): General Control Register B** **Init = 0bxxxx_xx11**

Bit	R/W	Description
31:8	RO	**Reserved (0)**
7 :4	RW	**SIRQ number selection bit [3:0]** 0000: IRQ0 0001: IRQ1 0010: SMI ... 1111: IRQ15
3 :2	RW	**Host-side timeout period selection** 00: 1/3600 second if V1UART38[1]=0 01: 1/7200 second if V1UART38[1]=0 10: 1/14400 second if V1UART38[1]=0 11: 1/28800 second if V1UART38[1]=0 00: 512*PCLK if V1UART38[1]=1 01: 256*PCLK if V1UART38[1]=1 10: 128*PCLK if V1UART38[1]=1 11: 64*PCLK if V1UART38[1]=1
1 :0	RO	**Number of bits per character (host-side)**
Note : This register is defined for ARM CPU only.		

Offset: 28h	V1UART28 (Slave): Virtual 1 UART Address Register L	Init = 0bxxxx_x001

Bit	R/W	Description
31:8	RO	**Reserved (0)**
7 :3	RW	**Virtual 1 UART address bit [7:3]** This register defines the base address (the low bytes) for host CPU to access virtual UART registers through LPC bus.
2	RW	**Halt Tx DMA**
1	RW	**Halt Rx DMA**
0	RO	**Version ID**

Note :
This register is defined for ARM CPU only.

Offset: 2Ch	V1UART2C (Slave): Virtual 1 UART Address Register H	Init = X

Bit	R/W	Description
31:8	RO	**Reserved (0)**
7 :0	RW	**Virtual 1 UART address bit [15:8]** This register defines the base address (the high bytes) for host CPU to access virtual UART registers through LPC bus.

Note :
This register is defined for ARM CPU only.

Offset: 30h	V1UART30 (Slave): General Control Register E	Init = 0b0000_1110

Bit	R/W	Description
31:8	RO	**Reserved (0)**
7	RO	**Transmit FIFO full. (slave-side)** 0: transmit FIFO not full. 1: transmit FIFO full.
6 :4	RO	**THR read pointer bit [2:0] (slave-side)**
3 :0	RO	**Complement of IIR status bit [3:0] (host-side)**

Note :
This register is defined for ARM CPU only.

Offset: 34h	V1UART34 (Slave): General Control Register F	Init = X

Bit	R/W	Description
31:8	RO	**Reserved (0)**
7	RW	**Enable FIFO 1/2 full THRE Interrupt Mode (slave-side)** 0: Disable FIFO 1/2 full THRE interrupt mode 1: Enable FIFO 1/2 full THRE interrupt mode
6	RW	**Enable FIFO 1/2 full THRE Interrupt Mode Control (host-side)** 0: Disable FIFO 1/2 full THRE interrupt mode control 1: Enable FIFO 1/2 full THRE interrupt mode control
5	RW	**Trig a THRE interrupt on host side even through it has been empty already**
4 :3	RW	**Reserved**
2	RW	**Enable the Slave (BMC) to monitor DL and SCR on the Host side.**
1	RW	**Disable character time out interrupt (slave-side)**

to next page

from previous page

0	RW	**Disable character time out interrupt (host-side)**

Note :
This register is defined for ARM CPU only.

Offset: 38h **V1UART38 (Slave): General Control Register G** **Init = X**

Bit	R/W	Description
31:8	RO	**Reserved (0)**
7 :0	RO	**THR read back data bit [7:0] (slave-side) if V1UART34[2]=0**
7 :2	RO	**Reserved if V1UART34[2]=1**
1	RO	**V1UART timeout time width control bit (slave-side) if V1UART34[2]=1**
0	RO	**V1UART timeout time width control bit (host-side) if V1UART34[2]=1**
7 :2	W	**Reserved**
1	W	**V1UART timeout time width control bit (slave-side)**
0	W	**V1UART timeout time width control bit (host-side)**

Note :
This register is defined for ARM CPU only.

Offset: 3Ch **V1UART3C (Slave): General Control Register H** **Init = X**

Bit	R/W	Description
31:8	RO	**Reserved (0)**
7 :6	RO	**Read back data bit [1:0] of receiver FIFO interrupt trigger level (slave-side)**
5	RO	**Interrupt Enable Register bit [7] on the Host side**
4	RO	**Enable UART FIFO on the Host side**
3 :0	RO	**Interrupt Enable Register bit [3:0] on the Host side**

Note :
This register is defined for ARM CPU only.

49 LPC Controller (LPC)

49.1 Overview

AST2600 integrates LPC Slave Controller, which includes IPMI 2.0/1.1 compliant BMC controller. There are totally 81 registers, which is listed below, to control the various functions supported by AST2600 . Each register has its own specific offset value to derive its physical address location.

Base Address of LPC Controller = 0x1E78_9000
Physical address of register = (Base address of LPC Controller) + Offset

HICR0 : Host Interface Control Register 0
HICR1 : Host Interface Control Register 1
HICR2 : Host Interface Control Register 2
HICR3 : Host Interface Control Register 3
HICR4 : Host Interface Control Register 4
LADR3H : LPC Channel #3 Address register H
LADR3L : LPC Channel #3 Address Register L
LADR12H : LPC Channel #1/#2 Address Register H
LADR12L : LPC Channel #1/#2 Address Register L
IDR1 : Input Data Register 1
IDR2 : Input Data Register 2
IDR3 : Input Data Register 3
ODR1 : Output Data Register 1
ODR2 : Output Data Register 2
ODR3 : Output Data Register 3
STR1 : Status Register 1
STR2 : Status Register 2
STR3 : Status Register 3
BTR0 : BT Status Register 0
BTR1 : BT Status Register 1
BTCSR0 : BT Control Status Register 0
BTCSR1 : BT Control Status Register 1
BTCR : BT Control Register
BTDTR : BT Data Buffer
BTIMSR : BT Interrupt Mask Register
BTFVSR0 : BT FIFO Valid Size Register 0
BTFVSR1 : BT FIFO Valid Size Register 1
SIRQCR0 : SERIRQ Control Register 0
SIRQCR1 : SERIRQ Control Register 1
SIRQCR2 : SERIRQ Control Register 2
SIRQCR3 : SERIRQ Control Register 3
HICR5 : Host Interface Control Register 5
HICR6 : Host Interface Control Register 6
HICR7 : Host Interface Control Register 7
HICR8 : Host Interface Control Register 8
SNPWADR : LPC Snoop Address Register
SNPWDR : LPC Snoop Data Register
HICR9 : Host Interface Control Register 9
HICRA : Host Interface Control Register A
LHCR0 : LPC Host Control Register 0
LHCR1 : LPC Host Control Register 1
LHCR2 : LPC Host Control Register 2
LHCR3 : LPC Host Control Register 3
LHCR4 : LPC Host Control Register 4

LHCR5 : LPC Host Control Register 5
LHCR6 : LPC Host Control Register 6
LHCR7 : LPC Host Control Register 7
LHCR8 : LPC Host Control Register 8
PCCR6 : Post Code Control Register 6
LHCRA : LPC Host Control Register A
LHCRB : LPC Host Control Register B
PCCR4 : Post Code Control Register 4
PCCR5 : Post Code Control Register 5
HICRB : Host Interface Control Register B
HICRC : Host Interface Control Register C
HISR0 : Host Interface Scratch Register 0
HISR1 : Host Interface Scratch Register 1
LADR4 : LPC Channel #4 Address register
IDR4 : Input Data Register 4
ODR4 : Output Data Register 4
STR4 : Status Register 4
LSADR12 : LPC Channel #1/#2 Status Address Register
PCCR0 : Post Code Control Register 0
PCCR1 : Post Code Control Register 1
PCCR2 : Post Code Control Register 2
PCCR3 : Post Code Control Register 3
iBTCR0 : iBT Control Register 0
iBTCR1 : iBT Control Register 1
iBTCR2 : iBT Control Register 2
iBTCR3 : iBT Control Register 3
iBTCR4 : iBT Control Register 4
iBTCR5 : iBT Control Register 5
iBTCR6 : iBT Control Register 6
SRUART1 : Status Register of UART1
SRUART2 : Status Register of UART2
SRUART3 : Status Register of UART3
SRUART4 : Status Register of UART4
SCR0SIO : Scratch Register 0 of SIO
SCR1SIO : Scratch Register 1 of SIO
SCR2SIO : Scratch Register 2 of SIO
SCR3SIO : Scratch Register 3 of SIO
SWCR_03_00 : SWC Register 03_00
SWCR_07_04 : SWC Register 07_04
SWCR_0B_08 : SWC Register 0B_08
SWCR_0F_0C : SWC Register 0F_0C
SWCR_13_10 : SWC Register 13_10
SWCR_17_14 : SWC Register 17_14
SWCR_1B_18 : SWC Register 1B_18
SWCR_1F_1C : SWC Register 1F_1C
ACPI_E3_E0 : ACPI Register E3_E0
ACPI_C1_C0 : ACPI Register C1_C0
ACPI_B3_B0 : ACPI Register B3_B0
ACPI_B7_B4 : ACPI Register B7_B4
MBXDAT_0 : MailBox Data Register 0
MBXDAT_1 : MailBox Data Register 1
MBXDAT_2 : MailBox Data Register 2
MBXDAT_3 : MailBox Data Register 3
MBXDAT_4 : MailBox Data Register 4
MBXDAT_5 : MailBox Data Register 5
MBXDAT_6 : MailBox Data Register 6
MBXDAT_7 : MailBox Data Register 7

MBXDAT_8 : MailBox Data Register 8
MBXDAT_9 : MailBox Data Register 9
MBXDAT_A : MailBox Data Register A
MBXDAT_B : MailBox Data Register B
MBXDAT_C : MailBox Data Register C
MBXDAT_D : MailBox Data Register D
MBXDAT_E : MailBox Data Register E
MBXDAT_F : MailBox Data Register F
MBXDAT_10 : MailBox Data Register 10
MBXDAT_11 : MailBox Data Register 11
MBXDAT_12 : MailBox Data Register 12
MBXDAT_13 : MailBox Data Register 13
MBXDAT_14 : MailBox Data Register 14
MBXDAT_15 : MailBox Data Register 15
MBXDAT_16 : MailBox Data Register 16
MBXDAT_17 : MailBox Data Register 17
MBXDAT_18 : MailBox Data Register 18
MBXDAT_19 : MailBox Data Register 19
MBXDAT_1A : MailBox Data Register 1A
MBXDAT_1B : MailBox Data Register 1B
MBXDAT_1C : MailBox Data Register 1C
MBXDAT_1D : MailBox Data Register 1D
MBXDAT_1E : MailBox Data Register 1E
MBXDAT_1F : MailBox Data Register 1F
MBXSTS_0 : MailBox Status Register 0
MBXSTS_1 : MailBox Status Register 1
MBXSTS_2 : MailBox Status Register 2
MBXSTS_3 : MailBox Status Register 3
MBXBCR : MailBox BMC Control Register
MBXHCR : MailBox Host Control Register
MBXBIE_0 : MailBox BMC Interrupt Enable Register 0
MBXBIE_1 : MailBox BMC Interrupt Enable Register 1
MBXBIE_2 : MailBox BMC Interrupt Enable Register 2
MBXBIE_3 : MailBox BMC Interrupt Enable Register 3
MBXHIE_0 : MailBox Host Interrupt Enable Register 0
MBXHIE_1 : MailBox Host Interrupt Enable Register 1
MBXHIE_2 : MailBox Host Interrupt Enable Register 2
MBXHIE_3 : MailBox Host Interrupt Enable Register 3

The definition of BMC related registers, from offset 0x00 to offset 0x7C, are basically compatible with the popular BMC controller - H8S/2168. Therefore, the software code developed for the chip can be easily ported to AST2600 .

49.2 Features

- Directly connected to APB bus interface
- Operation mode
  - Salve mode: designed for BMC functions (I/O read write cycles) and SBIOS boot (memory or firmware read write cycles)
- Support Serial IRQ (reduce polling time)
- Support port 80H/81H (programmable address) snooping registers with interrupt option
- Support two set of Virtual UART (16550) (SIRQ#)
- Compliant with IPMI version 2.0 KCS mode and BT mode

  - Channel #1 supports KCS interface
  - Channel #2 supports KCS interface
  - Channel #3 supports KCS or BT (H8S/2168 compliant) interface
  - Channel #4 supports KCS interface
  - Channel #5 supports iBT (IPMI compliant) interface
- Three register sets to support four programmable I/O channels. Each register set includes:
  - Input data register (IDR1-IDR4)
  - Output data register (ODR1-ODR4)
  - Status register (STR1-STR4)
- H8S/2168 compliant register definition and programming sequence
- Two sets of 256x8 Embedded SRAM for BT mode support
- Two sets of 256x8 Embedded SRAM for iBT mode support
- Support LPC S/W & H/W power down function
- Support LPC Abort monitoring function
- Support LPC bus debug function

49.3 Registers : Base Address = 0x1E78:9000

Attribute Definition:

Attribute	:	Description
R	:	Readable
W	:	Writable
RO	:	Read Only
WO	:	Write Only
W0C	:	Write '0' to clear value to 0
W1C	:	Write '1' to clear value to 0
W1T	:	Write '1' to toggle value
W0S	:	Write '0' to set value to 1
W1S	:	Write '1' to set value to 1
U	:	Unknown value
P	:	Initialized by PWRST_N
H	:	Initialized by LPC_RST_N
L	:	Initialized by LPC_LRST_N

HICR0: Host Interface Control Register 0 **Offset: 00h**

Bit	Name	Initial	Slave	Host	Description
7	**LPC3E**	0H	RW	-	**Enable LPC Channel #3**
6	**LPC2E**	0H	RW	-	**Enable LPC Channel #2**
5	**LPC1E**	0H	RW	-	**Enable LPC Channel #1**
4		0	RO	-	**Reserved**
3	**SDWNE**	0H	RW	-	**Enable LPC software shutdown**
2	**PMEE**	0H	RW	-	**Enable PME output**
1		0	RO	-	**Reserved**
0		0	RO	-	**Reserved**

HICR1: Host Interface Control Register 1					Offset: 04h
Bit	**Name**	**Initial**	**Slave**	**Host**	**Description**
7	**LPCBSY**	0L	RO	-	**LPC busy flag**
6		0	RO	-	**Reserved**
5	**IRQBSY**	0L	RO	-	**SERIRQ busy flag**
4	**LRSTB**	0H	RW	-	**LPC software reset bit**
3	**SDWNB**	0H	RW	-	**LPC software shutdown bit**
2	**PMEB**	0H	RW	-	**PME output bit**
1		0	RO	-	**Reserved**
0		0	RO	-	**Reserved**

HICR2: Host Interface Control Register 2					Offset: 08h
Bit	**Name**	**Initial**	**Slave**	**Host**	**Description**
7		0	RO	-	**Reserved**
6	**LRST**	0P	RW0C	-	**LPC reset interrupt status (INT#35)**
5	**SDWN**	0P	RW0C	-	**LPC shutdown interrupt status (INT#35)**
4	**ABRT**	0P	RW0C	-	**LPC Abort Interrupt status (INT#35)**
3	**IBFIF3**	0H	RW	-	**Enable IBFI3 interrupt**
2	**IBFIF2**	0H	RW	-	**Enable IDR2 receive completion interrupt**
1	**IBFIE1**	0H	RW	-	**Enable IDR1 receive completion interrupt**
0	**ERRIE**	0H	RW	-	**Enable error interrupt of bit[6:4]**

HICR3: Host Interface Control Register 3					Offset: 0Ch
Bit	**Name**	**Initial**	**Slave**	**Host**	**Description**
7	**LFRAME**	U	RO	-	**LFRAME pin monitoring**
6		0	RO	-	**Reserved**
5	**SERIRQ**	U	RO	-	**SERIRQ pin monitoring**
4	**LRESET**	U	RO	-	**LRESET pin monitoring**
3	**LPCPD**	U	RO	-	**LPCPD pin monitoring**
2	**PME**	U	RO	-	**PME pin monitoring**
1		0	RO	-	**Reserved**
0		0	RO	-	**Reserved**

HICR4: Host Interface Control Register 4					Offset: 10h
Bit	**Name**	**Initial**	**Slave**	**Host**	**Description**
7	**LADR12AS**	0H	RW	-	**Channel address selection (LADR12H or LADRL)**
6	**ClrIntLRstR**	0	RW1C	-	**Write one clear HICR4[5]**
5	**StsIntLRstR**	0P	RO	-	**Status of LRESET rising interrupt (INT#35)**
4	**EnIntLRstR**	0H	RW	-	**Enable LRESET rising interrupt**
3		0	RO	-	**Reserved**
2	**KCSENBL**	0H	RW	-	**Enable KCS interface in Channel #3**
1		0	RO	-	**Reserved**
0	**BTENBL**	0H	RW	-	**Enable BT interface in Channel #3**

LADR3H: LPC Channel #3 Address register H **Offset: 14h**

Bit	Name	Initial	Slave	Host	Description
7	**Bit15**	0H	RW	-	**Channel #3 address Bit[15]**
6	**Bit14**	0H	RW	-	**Channel #3 address Bit[14]**
5	**Bit13**	0H	RW	-	**Channel #3 address Bit[13]**
4	**Bit12**	0H	RW	-	**Channel #3 address Bit[12]**
3	**Bit11**	0H	RW	-	**Channel #3 address Bit[11]**
2	**Bit10**	0H	RW	-	**Channel #3 address Bit[10]**
1	**Bit9**	0H	RW	-	**Channel #3 address Bit[9]**
0	**Bit8**	0H	RW	-	**Channel #3 address Bit[8]**

LADR3L: LPC Channel #3 Address Register L **Offset: 18h**

Bit	Name	Initial	Slave	Host	Description
7	**Bit7**	0H	RW	-	**Channel #3 address Bit[7]**
6	**Bit6**	0H	RW	-	**Channel #3 address Bit[6]**
5	**Bit5**	0H	RW	-	**Channel #3 address Bit[5]**
4	**Bit4**	0H	RW	-	**Channel #3 address Bit[4]**
3		0H	RW	-	**Reserved; Channel #3 KCS address Bit[3]=0; Channel #3 BT address Bit[3]=0**
2		0	RO	-	**Reserved; Channel #3 KCS address Bit[2]=0; Channel #3 BT address Bit[2]=1**
1		0H	RW	-	**Reserved; Channel #3 KCS address Bit[1]=1**
0		0	RO	-	**Reserved**

LADR12H: LPC Channel #1/#2 Address Register H **Offset: 1Ch**

Bit	Name	Initial	Slave	Host	Description
31:16		62hH	RO	-	**Channel #2 address**
7:0	**Bit[15:8]**	0H	RW	-	**Channel #1/#2 address Bit[15:8]** (Selected by LADRSEL)

LADR12L: LPC Channel #1/#2 Address Register L **Offset: 20h**

Bit	Name	Initial	Slave	Host	Description
31:16		60hH	RO	-	**Channel #1 address**
7:0	**Bit[7:0]**	60hH /62hH	RW	-	**Channel #1/#2 address bit[7:0]** (Selected by LADRSEL)

IDR1: Input Data Register 1 **Offset: 24h**

Bit	Name	Initial	Slave	Host	Description
7:0	**Bit[7:0]**	U	RO	WO	**Channel #1 input data Bit[7:0]**

IDR2: Input Data Register 2 **Offset: 28h**

Bit	Name	Initial	Slave	Host	Description
7:0	**Bit[7:0]**	U	RO	WO	**Channel #2 input data Bit[7:0]**

IDR3: Input Data Register 3 **Offset: 2Ch**

Bit	Name	Initial	Slave	Host	Description
7:0	**Bit[7:0]**	U	RO	WO	**Channel #3 input data Bit[7:0]**

ODR1: Output Data Register 1 **Offset: 30h**

Bit	Name	Initial	Slave	Host	Description
7:0	**Bit[7:0]**	U	RW	RO	**Channel #1 output data Bit[7:0]**

ODR2: Output Data Register 2 **Offset: 34h**

Bit	Name	Initial	Slave	Host	Description
7:0	**Bit[7:0]**	U	RW	RO	**Channel #2 output data Bit[7:0]**

ODR3: Output Data Register 3 **Offset: 38h**

Bit	Name	Initial	Slave	Host	Description
7:0	**Bit[7:0]**	U	RW	RO	**Channel #3 output data Bit[7:0]**

STR1: Status Register 1 **Offset: 3Ch**

Bit	Name	Initial	Slave	Host	Description
7	**DBU17**	0H	RW	RO	**Defined by user**
6	**DBU16**	0H	RW	RO	**Defined by user**
5	**DBU15**	0H	RW	RO	**Defined by user**
4	**DBU14**	0H	RW	RO	**Defined by user**
3	**C/D1**	0P	RO	RO	**Command/Data**
2	**DBU12**	0H	RW	RO	**Defined by user**
1	**IBF1**	0P	RO	RO	**Input data register full**
0	**OBF1**	0P	RW0C	RO	**Output data register full**

STR2: Status Register 2 **Offset: 40h**

Bit	Name	Initial	Slave	Host	Description
7	**DBU27**	0H	RW	RO	**Defined by user**
6	**DBU26**	0H	RW	RO	**Defined by user**
5	**DBU25**	0H	RW	RO	**Defined by user**
4	**DBU24**	0H	RW	RO	**Defined by user**
3	**C/D2**	0P	RO	RO	**Command/Data**
2	**DBU22**	0H	RW	RO	**Defined by user**
1	**IBF2**	0P	RO	RO	**Input data register full**
0	**OBF2**	0P	RW0C	RO	**Output data register full**

STR3: Status Register 3					Offset: 44h
Bit	**Name**	**Initial**	**Slave**	**Host**	**Description**
7	**DBU37**	0H	RW	RO	**Defined by user**
6	**DBU36**	0H	RW	RO	**Defined by user**
5	**DBU35**	0H	RW	RO	**Defined by user**
4	**DBU34**	0H	RW	RO	**Defined by user**
3	**C/D3**	0P	RO	RO	**Command/Data**
2	**DBU32**	0H	RW	RO	**Defined by user**
1	**IBF3**	0P	RO	RO	**Input data register full**
0	**OBF3**	0P	RW0C	RO	**Output data register full**

BTR0: BT Status Register 0					Offset: 48h
Bit	**Name**	**Initial**	**Slave**	**Host**	**Description**
31:28		1	RO	-	**Version ID**
7		0	RO	-	**Reserved**
6		0	RO	-	**Reserved**
5		0	RO	-	**Reserved**
4	**FRDI**	0H	RW0C	-	**FIFO read request interrupt**
3	**HRDI**	0H	RW0C	-	**BT host read interrupt**
2	**HWRI**	0H	RW0C	-	**BT host write interrupt**
1	**HBTWI**	0H	RW0C	-	**BTDTR host write start interrupt**
0	**HBTRI**	0H	RW0C	-	**BTDTR host read end interrupt**

BTR1: BT Status Register 1					Offset: 4Ch
Bit	**Name**	**Initial**	**Slave**	**Host**	**Description**
7		0	RO	-	**Reserved**
6	**HRSTI**	0H	RW0C	-	**BT reset interrupt**
5		0	RO	-	**Reserved**
4	**BEVTI**	0H	RW0C	-	**BEVT_ATN clear interrupt**
3	**B2HI**	0H	RW0C	-	**Read-end interrupt**
2	**H2BI**	0H	RW0C	-	**Write-end interrupt**
1	**CRRPI**	0H	RW0C	-	**Read pointer clear interrupt**
0	**CRWPI**	0H	RW0C	-	**Write pointer clear interrupt**

BTCSR0: BT Control Status Register 0					Offset: 50h
Bit	**Name**	**Initial**	**Slave**	**Host**	**Description**
7		0	RO	-	**Reserved**
6	**FSEL1**	0H	RW	-	**BT Transfer FIFO selection bit 1**
5	**FSEL0**	0H	RW	-	**BT Transfer FIFO selection bit 0**
4	**FRDIE**	0H	RW	-	**Enable FIFO read request interrupt**
3	**HRDIE**	0H	RW	-	**Enable BT host read interrupt**
2	**HWRIE**	0H	RW	-	**Enable BT host write interrupt**

to next page

from previous page

1	**HBTWIE**	0H	RW	-	**Enable BTDTR host write start interrupt**
0	**HBTRIE**	0H	RW	-	**Enable BTDTR host read end interrupt**

BTCSR1: BT Control Status Register 1 **Offset: 54h**

Bit	Name	Initial	Slave	Host	Description
7	**RSTREN**	0H	RW	-	**Enable slave reset read**
6	**HRSTIE**	0H	RW	-	**Enable BT reset interrupt**
5		0	RO	-	**Reserved**
4	**BEVTIE**	0H	RW	-	**Enable BEVT_ATN clear interrupt**
3	**B2HIE**	0H	RW	-	**Enable read end interrupt**
2	**H2BIE**	0H	RW	-	**Enable write end interrupt**
1	**CRRPIE**	0H	RW	-	**Enable read pointer clear interrupt**
0	**CRWPIE**	0H	RW	-	**Enable write pointer clear interrupt**

BTCR: BT Control Register **Offset: 58h**

Bit	Name	Initial	Slave	Host	Description
7	**B_BUSY**	0H	RW	RO	**BT write transfer busy flag**
6	**H_BUSY**	0H	RO	W1T	**BT read transfer busy flag**
5	**OEM0**	0H	RW	RW0S	**User defined bit**
4	**BEVT_ATN**	0H	RW1S	RW1C	**Event interrupt**
3	**B2H_ATN**	0H	RW1S	RW1C	**Slave buffer write end indication flag**
2	**H2B_ATN**	0H	RW0C	RW1S	**Host buffer write end indication flag**
1	**CLR_RD_PTR**	0H	RW0C	W1S	**Read pointer clear**
0	**CLR_WR_PTR**	0H	RW0C	W1S	**Write pointer clear**

BTDTR: BT Data Buffer **Offset: 5Ch**

Bit	Name	Initial	Slave	Host	Description
7:0	**Bit [7:0]**	U	RW	RW	**BT mode buffer read/write data**

BTIMSR: BT Interrupt Mask Register **Offset: 60h**

Bit	Name	Initial	Slave	Host	Description
7	**BMC_HWRST**	0H	RW0C	RW1S	**Slave reset**
6		0	RO	RO	**Reserved**
5		0	RO	RO	**Reserved**
4	**OEM3**	0H	RW	RW0S	**User defined bit**
3	**OEM2**	0H	RW	RW0S	**User defined bit**
2	**OEM1**	0H	RW	RW0S	**User defined bit**
1		0	RO	RO	**Reserved**
0		0	RO	RO	**Reserved**

BTFVSR0: BT FIFO Valid Size Register 0 **Offset: 64h**

Bit	Name	Initial	Slave	Host	Description
8:0	**N8 to N0**	0H	RO	-	**Valid bytes in the FIFO for host write transfer; FIFO size is 256 bytes**

BTFVSR1: BT FIFO Valid Size Register 1 **Offset: 68h**

Bit	Name	Initial	Slave	Host	Description
8:0	**N8 to N0**	0H	RO	-	**Valid bytes in the FIFO for host read transfer; FIFO size is 256 bytes**

SIRQCR0: SERIRQ Control Register 0 **Offset: 70h**

Bit	Name	Initial	Slave	Host	Description
31:28		5L	RO	-	**Status of SIORC_70[3:0]**
27:24		6L	RO	-	**Status of SIORB_70[3:0]**
23:20		3L	RO	-	**Status of SIOR3_70[3:0]**
19:16		4L	RO	-	**Status of SIOR2_70[3:0]**
15		0L	RO	-	**Status of inverse SIORC_71[0]**
14		0L	RO	-	**Status of inverse SIORB_71[0]**
13		0L	RO	-	**Status of inverse SIOR3_71[0]**
12		0L	RO	-	**Status of inverse SIOR2_71[0]**
7	**Q_C**	0L	RO	-	**Quiet/Continuous Mode Flag**
6		0H	RW	-	**Reserved**
5	**IEDIR**	0H	RW	-	**Interrupt Enable Direct Mode**
4		0	RO	-	**Reserved**
3	**SMIE3A**	0H	RW	-	**Host SMI Interrupt Enable 3A**
2	**SMIE2**	0H	RW	-	**Host SMI Interrupt Enable 2**
1	**IRQ12E1**	0H	RW	-	**Host IRQ12 Interrupt Enable 1**
0	**IRQ1E1**	0H	RW	-	**Host IRQ1 Interrupt Enable 1**

SIRQCR1: SERIRQ Control Register 1 **Offset: 74h**

Bit	Name	Initial	Slave	Host	Description
13		0P	RW	-	**Write protect HICRB**
12		0P	RW	-	**Write protect PCCR5**
11		0P	RW	-	**Write protect PCCR4**
10		0P	RW	-	**Write protect HICR8**
9		0P	RW	-	**Write protect HICR7**
8		0P	RW	-	**Write protect HICR5**
7	**IRQ11E3**	0H	RW	-	**Host IRQ11 Interrupt Enable 3**
6	**IRQ10E**	0H	RW	-	**Host IRQ10 Interrupt Enable 3**
5	**IRQ9E3**	0H	RW	-	**Host IRQ9 Interrupt Enable 3**
4	**IRQ6E3**	0H	RW	-	**Host IRQ6 Interrupt Enable 3**
3	**IRQ11E2**	0H	RW	-	**Host IRQ11 Interrupt Enable 2**
2	**IRQ10E2**	0H	RW	-	**Host IRQ10 Interrupt Enable 2**

to next page

from previous page

1	**IRQ9E2**	0H	RW	-	**Host IRQ9 Interrupt Enable 2**
0	**IRQ6E2**	0H	RW	-	**Host IRQ6 Interrupt Enable 2**

SIRQCR2: SERIRQ Control Register 2 **Offset: 78h**

Bit	Name	Initial	Slave	Host	Description
15		0H	RW	-	**Interrupt Enable Direct Mode 1**
14		0H	RW	-	**Disable KCS1 serial IRQ low stretcher**
13		0H	RW	-	**Select SERIRQX output for channel #1**
12		0L	RW	-	**Host IRQX interrupt enable for channel #1**
11: 8		1H	RW	-	**Select ID bit[3:0] of IRQX for channel #1**
7	**IEDIR3**	0H	RW	-	**Interrupt Enable Direct Mode 3**
6:0		0	RO	-	**Reserved**

SIRQCR3: SERIRQ Control Register 3 **Offset: 7Ch**

Bit	Name	Initial	Slave	Host	Description
28: 8		U	RO	-	**HW debug status**
7		0	RO	-	**Reserved**
6	**SELIRQ11**	0H	RW	-	**Select SERIRQ11 Output**
5	**SELIRQ10**	0H	RW	-	**Select SERIRQ10 Output**
4	**SELIRQ9**	0H	RW	-	**Select SERIRQ9 Output**
3	**SELIRQ6**	0H	RW	-	**Select SERIRQ6 Output**
2	**SELSMI**	0H	RW	-	**Select SERSMI Output**
1	**SELIRQ12**	0H	RW	-	**Select SERIRQ12 Output**
0	**SELIRQ1**	0H	RW	-	**Select SERIRQ1 Output**

HICR5: Host Interface Control Register 5 **Offset: 80h**

Bit	Name	Initial	Slave	Host	Description
31		0P	RW	-	**EnSIO80hGIO: Enable post code 80h GPIO controls from host side**
30		0P	RW	-	**En80hGIO: Enable post code 80h GPIO (BMC side)**
29		0P	RW	-	**EnInv80hGIO: Enable post code 80h polarity inversed GPIO (BMC side)**

to next page

from previous page

28:24		U	RW	-	**Sel80hGIO: Group selection of post code 80h GPIO (BMC side)** 00000 : GPIOA 00001 : GPIOB 00010 : GPIOC ... 10010 : GPIOS 10011 : reserved 10100 : reserved 10101 : GPIOV 10010 : GPIOW 10011 : GPIOX 10100 : GPIOY 10101 : GPIOZ others: reserved
23:20	**ID3IRQX**	U	RW	-	**Select ID bit[3:0] of IRQX for channel #3**
19:16	**ID2IRQX**	U	RW	-	**Select ID bit[3:0] of IRQX for channel #2**
15	**SEL3IRQX**	0P	RW	-	**Select SERIRQX output for channel #3**
14	**IRQXE3**	0H	RW	-	**Host IRQX interrupt enable for channel #3**
13	**SEL2IRQX**	0P	RW	-	**Select SERIRQX output for channel #2**
12	**IRQXE2**	0H	RW	-	**Host IRQX interrupt enable for channel #2**
11		0P	RW	-	**Enable interrupt of UART1/2/3/4 baudrate touch**
10	**ENFWH**	1P	RW	-	**Enable LPC FWH cycles**
9	**ENINT_PME**	0H	RW	-	**Enable PME# interrupt** 0: Disable 1: Enable
8	**ENL2H**	0P	RW	-	**Enable LPC to AHB bridge**
7		0P	RW	-	**Enable interrupt of UART1/2/3/4 RX pending but data not ready**
6		0P	RW	-	**Disable LPCPD function when pin is defined as SMI**
5	**ENSET_SF**	0H	RW	-	**Enable the capability to issue SIRQ start frame cycles** 0: No operation 1: Enable the capability This register is designed to enable LCP Slave Controller to be able to trig chipset to issue SIRQ start frame cycles.
4		0H	RW	-	**Reserved**
3	**ENINT_SNP1W**	0P	RW	-	**Enable snooping address #1 interrupt** 0: Disable 1: Enable This bit will enable LPC Slave Controller to issue interrupt for LPC bus-write cycles (snooping address #1).

to next page

from previous page

2	**EN_SNP1W**	0P	RW	-	**Enable snooping address #1** 0: Disable 1: Enable This bit will enable LPC Slave Controller to snoop LPC bus cycle regarding to the snooping address #1.
1	**ENINT_SNP0W**	0P	RW	-	**Enable snooping address #0 interrupt** 0: Disable 1: Enable This bit will enable LPC Slave Controller to issue interrupt for all LPC bus-write cycles matched with the snooping address #0.
0	**EN_SNP0W**	0P	RW	-	**Enable snooping address #0** 0: Disable 1: Enable This bit will enable LPC Slave Controller to snoop LPC bus cycles regarding to the snooping address #0.

HICR6: Host Interface Control Register 6 **Offset: 84h**

Bit	Name	Initial	Slave	Host	Description
31		0P	RW	-	**EnSIO81hGIO: Enable post code 81h GPIO controls from host side**
30		0P	RW	-	**En81hGIO: Enable post code 81h GPIO (BMC side)**
29		0P	RW	-	**EnInv81hGIO: Enable post code 81h polarity inversed GPIO (BMC side)**
28:24		U	RW	-	**Sel81hGIO: Group selection of post code 81h GPIO (BMC side)** 00000 : GPIOA 00001 : GPIOB 00010 : GPIOC ... 10010 : GPIOS 10011 : reserved 10100 : reserved 10101 : GPIOV 10010 : GPIOW 10011 : GPIOX 10100 : GPIOY 10101 : GPIOZ others: reserved
23		0P	RW	-	**En81hSGIO: Enable POST code 81h to SGPIO function when HICR6[31]=1. Will replace serial data GPIO500[15:8].**
22:20		U	RW	-	**Reserved**
19		0P	RW	-	**Enable 2-byte mode of snoop functions. When enabled and SNPWADR[0]=0 and SNPWADR[16]=0, SNPWDR[15:8] is Snooping address #0 data Bit[15:8] and SNPWDR[31:24] is Snooping address #1 data Bit[15:8].**

to next page

from previous page

18		U	RW	-	**Reserved**
17	**SW_FWH2AHB**	0P	RW	-	**Switch to FWH2AHB mode**
16		0P	RW1C	-	**Status of UART1/2/3/4 interrupt of RX pending but data not ready** 0: No interrupt 1: Interrupt is pending Writing '1' to this bit will clear the status.
15		0P	RW1C	-	**Status of UART4 LCR touch interrupt**
14		0P	RW1C	-	**Status of UART3 LCR touch interrupt**
13		0P	RW1C	-	**Status of UART2 LCR touch interrupt**
12		0P	RW1C	-	**Status of UART1 LCR touch interrupt**
11		0P	RW1C	-	**Status of UART4 baudrate touch interrupt**
10		0P	RW1C	-	**Status of UART3 baudrate touch interrupt**
9		0P	RW1C	-	**Status of UART2 baudrate touch interrupt**
8		0P	RW1C	-	**Status of UART1 baudrate touch interrupt**
7	**SIRQSTOP**	0L	RO	-	**SerIRQ frame stop status**
6		0P	RO	-	**Reserved**
5		0	RO	-	**Reserved**
4		0	RO	-	**Reserved**
3	**STR_Baud**	0P	RW1C	-	**Status of UART1/2/3/4 baudrate touch interrupt** 0: No interrupt 1: Interrupt is pending Writing '1' to this bit will clear the status.
2	**STR_PME**	0P	RW1C	-	**PME# interrupt status** 0: No interrupt 1: Interrupt is pending Writing '1' to this bit will clear the status.
1	**STR_SNP1W**	0P	RW1C	-	**Snooping address #1 interrupt status** 0: No interrupt 1: Interrupt is pending Writing '1' to this bit will clear the status.
0	**STR_SNP0W**	0P	RW1C	-	**Snooping address #0 interrupt status** 0: No interrupt 1: Interrupt is pending Writing '1' to this bit will clear the status.

HICR7: Host Interface Control Register 7 **Offset: 88h**

Bit	Name	Initial	Slave	Host	Description
31:16	**ADRBASE**	3000P	RW	-	**Remapping address base bit [31:16] of LPC to AHB bridge**
15:0	**HWMBASE**	FFF8P	RW	-	**LPC to AHB bridge address decoding base bit [31:16]**

HICR8: Host Interface Control Register 8					Offset: 8Ch
Bit	**Name**	**Initial**	**Slave**	**Host**	**Description**
31:16	**ADRMASK**	FFF8P	RW	-	**Remapping address mask bit [31:16] of LPC to AHB bridge**
15:0	**HWNCARE**	F007P	RW	-	**Address decoding range control bit [31:16] of LPC to AHB bridge**

SNPWADR: LPC Snoop Address Register					Offset: 90h
Bit	**Name**	**Initial**	**Slave**	**Host**	**Description**
31:16	**Bit [15:0]**	81P	RW	-	**Snooping address #1** This register is designed to set the snooping address #1 (Bit [15:0]) for monitoring LPC bus I/O-write cycles. Setting snooping address like 80h or 81h is the typical applications to monitor system BIOS activities.
15:0	**Bit [15:0]**	80P	RW	-	**Snooping address #0** This register is designed to set the snooping address #0 Bit [15:0] for monitoring LPC bus I/O-write cycles. Setting snooping address like 80h or 81h is the typical applications to monitor system BIOS activities.

SNPWDR: LPC Snoop Data Register					Offset: 94h
Bit	**Name**	**Initial**	**Slave**	**Host**	**Description**
31:24	**Bit [7:0]**	FFP	RO	-	**Snooping address #1 old data Bit[7:0]** This register will always record the second last data of LPC bus write cycles with address matched with SNPWADR [31:16]. This register will be automatically updated whenever a new write cycle with matched address. Firmware code needs to take care of the potential issue of reading transition data.
23:16	**Bit [7:0]**	FFP	RO	-	**Snooping address #0 old data Bit[7:0]** This register will always record the second last data of LPC bus write cycles with address matched with SNPWADR [15:0]. This register will be automatically updated whenever a new write cycle with matched address. Firmware code needs to take care of the potential issue of reading transition data.
15:8	**Bit [7:0]**	FFP	RO	-	**Snooping address #1 data Bit[7:0]** This register will always record the last data of LPC bus write cycles with address matched with SNPWADR [31:16]. This register will be automatically updated whenever a new write cycle with matched address. Firmware code needs to take care of the potential issue of reading transition data.
7:0	**Bit [7:0]**	FFP	RO	-	**Snooping address #0 data Bit[7:0]** This register will always record the last data of the LPC bus write cycles with address matched with SNPWADR [15:0]. This register will be automatically updated whenever a new write cycle with matched address. Firmware code needs to take care of the potential issue of reading transition data.

HICR9: Host Interface Control Register 9					Offset: 98h
Bit	**Name**	**Initial**	**Slave**	**Host**	**Description**
31:24		0	RO	-	**Reserved**
23		0P	RW	-	**Enable interrupt of UART4 LCR touch**
22		0P	RW	-	**Enable interrupt of UART3 LCR touch**
21		0P	RW	-	**Enable interrupt of UART2 LCR touch**
20		0P	RW	-	**Enable interrupt of UART1 LCR touch**
19		0P	RW	-	**Enable interrupt of UART4 baudrate touch**
18		0P	RW	-	**Enable interrupt of UART3 baudrate touch**
17		0P	RW	-	**Enable interrupt of UART2 baudrate touch**
16		0P	RW	-	**Enable interrupt of UART1 baudrate touch**
15:12	**sel6DW**	0P	RW	-	**0000 : Route IO6 to UART6 (A1 silicon)** **0001 : Route IO1 to UART6 (A1 silicon)** **0010 : Route IO2 to UART6 (A1 silicon)** **0011 : Route IO3 to UART6 (A1 silicon)** **0100 : Route IO4 to UART6 (A1 silicon)** **0110 : Route UART1 to UART6 (A1 silicon)** **0111 : Route UART2 to UART6 (A1 silicon)** **1000 : Route UART3 to UART6 (A1 silicon)** **1001 : Route UART4 to UART6 (A1 silicon)** **others: Reserved (A1 silicon)**
15:12	**sel10DW**	0P	RW	-	**0000 : Route IO10 to UART10 (A2 silicon)** **0001 : Route IO1 to UART10 (A2 silicon)** **0010 : Route IO2 to UART10 (A2 silicon)** **0011 : Route IO3 to UART10 (A2 silicon)** **0100 : Route IO4 to UART10 (A2 silicon)** **0110 : Route UART1 to UART10 (A2 silicon)** **0111 : Route UART2 to UART10 (A2 silicon)** **1000 : Route UART3 to UART10 (A2 silicon)** **1001 : Route UART4 to UART10 (A2 silicon)** **others: Reserved (A2 silicon)**
11:8	**sel6IO**	AP	RW	-	**0000 : Route UART1 to IO6 (A1 silicon)** **0001 : Route UART2 to IO6 (A1 silicon)** **0010 : Route UART3 to IO6 (A1 silicon)** **0011 : Route UART4 to IO6 (A1 silicon)** **0101 : Route IO1 to IO6 (A1 silicon)** **0110 : Route IO2 to IO6 (A1 silicon)** **0111 : Route IO3 to IO6 (A1 silicon)** **1000 : Route IO4 to IO6 (A1 silicon)** **1010 : Route UART6 to IO6 (A1 silicon)** **others: Reserved (A1 silicon)**
11:8	**sel10IO**	AP	RW	-	**0000 : Route UART1 to IO10 (A2 silicon)** **0001 : Route UART2 to IO10 (A2 silicon)** **0010 : Route UART3 to IO10 (A2 silicon)** **0011 : Route UART4 to IO10 (A2 silicon)** **0101 : Route IO1 to IO10 (A2 silicon)** **0110 : Route IO2 to IO10 (A2 silicon)** **0111 : Route IO3 to IO10 (A2 silicon)** **1000 : Route IO4 to IO10 (A2 silicon)** **1010 : Route UART10 to IO10 (A2 silicon)** **others: Reserved (A2 silicon)**

to next page

from previous page

7		0P	RW	-	**Enable UART4 reset source from LPC**
6		0P	RW	-	**Enable UART3 reset source from LPC**
5		1P	RW	-	**Enable UART2 reset source from LPC**
4		1P	RW	-	**Enable UART1 reset source from LPC**
3		0	RO	-	**Status of SIORC_F0[2] AND SIORC_30[0], reserved for HW (A2 silicon)**
2		0	RO	-	**Status of SIORB_F0[2] AND SIORB_30[0], reserved for HW (A2 silicon)**
1		0	RO	-	**Status of SIOR3_F0[2] AND SIOR3_30[0], reserved for HW (A2 silicon)**
0		0	RO	-	**Status of SIOR2_F0[2] AND SIOR2_30[0], reserved for HW (A2 silicon)**

HICRA: Host Interface Control Register A **Offset: 9Ch**

Bit	Name	Initial	Slave	Host	Description
31		0P	RW	-	**Enable UART4 to automatically adopt baud rate and line control registers bits of the source UART. The source UART is determined by HICRA Bit[27:25]. The possible values of HICRA Bit[27:25] are 100, 101 and 110. (A2 silicon)**
30		0P	RW	-	**Enable UART3 to automatically adopt baud rate and line control registers bits of the source UART. The source UART is determined by HICRA Bit[24:22]. The possible values of HICRA Bit[24:22] are 100, 101 and 110. (A2 silicon)**
29		0P	RW	-	**Enable UART2 to automatically adopt baud rate and line control registers bits of the source UART. The source UART is determined by HICRA Bit[21:19]. The possible values of HICRA Bit[21:19] are 100, 101 and 110. (A2 silicon)**
28		0P	RW	-	**Enable UART1 to automatically adopt baud rate and line control registers bits of the source UART. The source UART is determined by HICRA Bit[18:16]. The possible values of HICRA Bit[18:16] are 100, 101 and 110. (A2 silicon)**
27:25	**sel4DW**	0P	RW	-	**000 : Route IO4 to UART4** **001 : Route IO1 to UART4** **010 : Route IO2 to UART4** **011 : Route IO3 to UART4** **100 : Route UART1 to UART4** **101 : Route UART2 to UART4** **110 : Route UART3 to UART4** **111 : Route IO6 to UART4 (A1 silicon)** **111 : Route IO10 to UART4 (A2 silicon)**

to next page

from previous page

24:22	**sel3DW**	0P	RW	-	**000 : Route IO3 to UART3** **001 : Route IO4 to UART3** **010 : Route IO1 to UART3** **011 : Route IO2 to UART3** **100 : Route UART4 to UART3** **101 : Route UART1 to UART3** **110 : Route UART2 to UART3** **111 : Route IO6 to UART3 (A1 silicon)** **111 : Route IO10 to UART3 (A2 silicon)**
21:19	**sel2DW**	0P	RW	-	**000 : Route IO2 to UART2** **001 : Route IO3 to UART2** **010 : Route IO4 to UART2** **011 : Route IO1 to UART2** **100 : Route UART3 to UART2** **101 : Route UART4 to UART2** **110 : Route UART1 to UART2** **111 : Route IO6 to UART2 (A1 silicon)** **111 : Route IO10 to UART2 (A2 silicon)**
18:16	**sel1DW**	0P	RW	-	**000 : Route IO1 to UART1** **001 : Route IO2 to UART1** **010 : Route IO3 to UART1** **011 : Route IO4 to UART1** **100 : Route UART2 to UART1** **101 : Route UART3 to UART1** **110 : Route UART4 to UART1** **111 : Route IO6 to UART1 (A1 silicon)** **111 : Route IO10 to UART1 (A2 silicon)**
15		0P	RW	-	**Route Idle to UART4 when there is no proper source (A2 silicon)**
14		0P	RW	-	**Route Idle to UART3 when there is no proper source (A2 silicon)**
13		0P	RW	-	**Route Idle to UART2 when there is no proper source (A2 silicon)**
12		0P	RW	-	**Route Idle to UART1 when there is no proper source (A2 silicon)**
11: 9	**sel4IO**	0P	RW	-	**000 : Route UART4 to IO4** **001 : Route UART10 to IO4 (A2 silicon)** **010 : Route UART1 to IO4** **011 : Route UART2 to IO4** **100 : Route UART3 to IO4** **101 : Route IO1 to IO4** **110 : Route IO2 to IO4** **111 : Route IO6 to IO4 (A1 silicon)** **111 : Route IO10 to IO4 (A2 silicon)**

to next page

from previous page

8: 6	**sel3IO**	0P	RW	-	**000 : Route UART3 to IO3** **001 : Route UART4 to IO3** **010 : Route UART10 to IO3 (A2 silicon)** **011 : Route UART1 to IO3** **100 : Route UART2 to IO3** **101 : Route IO1 to IO3** **110 : Route IO2 to IO3** **111 : Route IO6 to IO3 (A1 silicon)** **111 : Route IO10 to IO3 (A2 silicon)**
5: 3	**sel2IO**	0P	RW	-	**000 : Route UART2 to IO2** **001 : Route UART3 to IO2** **010 : Route UART4 to IO2** **011 : Route UART10 to IO2 (A2 silicon)** **100 : Route UART1 to IO2** **101 : Route IO3 to IO2** **110 : Route IO4 to IO2** **111 : Route IO6 to IO2 (A1 silicon)** **111 : Route IO10 to IO2 (A2 silicon)**
2: 0	**sel1IO**	0P	RW	-	**000 : Route UART1 to IO1** **001 : Route UART2 to IO1** **010 : Route UART3 to IO1** **011 : Route UART4 to IO1** **100 : Route UART10 to IO1 (A2 silicon)** **101 : Route IO3 to IO1** **110 : Route IO4 to IO1** **111 : Route IO6 to IO1 (A1 silicon)** **111 : Route IO10 to IO1 (A2 silicon)**

LHCR0: LPC Host Control Register 0 **Offset: A0h**

Bit	Name	Initial	Slave	Host	Description
31		U	RO	-	**Inputs of LPC SIRQ pin**
30		U	RO	-	**Inputs of LPC LPCPDN pin**
29		U	RO	-	**Inputs of LPC LCLK pin**
28		U	RO	-	**Inputs of LPC FrameN pin**
27:24		U	RO	-	**Inputs of LPC AD [3:0] pins**
23	**LRSTNO**	U	RW	-	**LPC reset pin output** 0: output low 1: output high
22		1H	RW	-	**LPC Host LPCPDN pin output signal** **Also, it is defined as the firmware driven bit for SMI if both SIORD_30[1] and SIORD_30[3] are set.**
21:16		U	RW	-	**Reserved**
15	**LRSTNOEN**	1H	RW	-	**LPC reset pin output control** 0: output mode 1: input mode
14		1H	RW	-	**LPC Host LPCPDN pin output control**
13		1H	RW	-	**Reserved**

to next page

from previous page

12		1P	RW	-	**Disable vector interrupt output connected to host serial IRQ** It provide an option for host to use GPIO when ARM is disabled. 0: enable 1: disable
11		1P	RW	-	**Disable KCS3 serial IRQ low stretcher** 0: enable 1: disable
10		1P	RW	-	**Disable KCS2 serial IRQ low stretcher** 0: enable 1: disable
9		1P	RW	-	**Disable PUART serial IRQ low stretcher** 0: enable 1: disable
8		1P	RW	-	**Disable VUART serial IRQ low stretcher** 0: enable 1: disable
7	**SIRQLONG**	U	RW	-	**Extend LPC Host SIRQ start frame 8/6 LPCCLK**
6	**EnLhSirqF**	0H	RW	-	**Extend LPC Host SIRQ start frame**
5		0	RO	-	**Reserved**
4	**ENP2L**	0H	RW	-	**Enable APB to LPC bridge**
3		0	RO	-	**Reserved**
2	**ENLHSIRQ**	0H	RW	-	**Enable LPC Host SIRQ** 0: Disable LPC host SIRQ 1: Enable LPC host SIRQ When this bit is enabled, LPC Host Controller will be able to receive SIRQ from LPC slave devices. This bit can be enabled only when LPC Host Controller is enabled.
1	**ENLHTM-OUT**	0H	RW	-	**Enable LPC host time-out function** 0: Disable LPC host time-out function 1: Enable LPC host time-out function When this bit has been enabled, LPC Host Controller will count the period of waiting cycles. If the period of waiting cycles is over LPC host time-out limit (LHCR1 Bit [31:16]), LPC Host Controller will automatically abort.
0	**ENLPC-HOST**	0P	RW	-	**Enable LPC Host Controller** 0: Disable LPC Host Controller 1: Enable LPC Host Controller This function is designed to support BMC controller to update system flash BIOS through LPC bus. Since LPC bus protocol doesnt support multi-master mode, LPC Host Controller can only be enabled when host platform has been full shutdown; otherwise, it will cause serious LPC bus conflictions between chipset and AST2600 .

LHCR1: LPC Host Control Register 1 **Offset: A4h**

Bit	Name	Initial	Slave	Host	Description
31:16	**LHTMOUTLMT**	U	RW	-	**LPC host time-out limit Bit[15:0]** This register sets the maximum number of cycles that LPC Host Controller can wait for Sync Ready from LPC slave devices. If the number of waiting cycles is over the limit, LPC Host Controller will automatically abort the cycle.
15		0P	RW	-	**TrigErrExit: Trigger an event to force LPC SYNC exit. HW debug only**
14:2		0	RO	-	**Reserved**
1	**LHS-ABORT**	0L	RW	-	**Force LPC Host Controller to abort** 0: No operation 1: Force LPC Host Controller to abort This bit will force LPC Host Controller to stop the current bus cycle and return to its initial state. S/W needs to reset this bit to resume LPC Host Controller.
0	**LHFIRE**	0L	RW	-	**Fire LPC host bus cycle** 0: No operation 1: Fire a LPC bus cycle Writing '1' to this register will force LPC Host Controller to issue one LPC bus cycle based on the information provided by LHCR4 and LHCR5. The write data will be from LHCR6. The read back data will be latched in LHCR7. S/W needs to write '0' to this bit before firing the next bus cycle by writing '1' to this bit again.

LHCR2: LPC Host Control Register 2 **Offset: A8h**

Bit	Name	Initial	Slave	Host	Description
31		0	RO	-	**Reserved**
30	**LSIRQCLR**	0H	RW	-	**Reserved**
29	**ENLSIRQW**	0H	RW	-	**Reserved**
28:8	**ENLHSR-INT**	0H	RW	-	**Enable LPC Host SIRQ Interrupt Bit [20:0]** Each bit of this register represents one of the interrupt enable bit for the IRQ sources listed below. (0: Disable, 1: Enable) Bit[0]: IRQ0 Bit[1]: IRQ1 .... Bit[15]: IQR15 Bit[16]: IOCK Bit[17]: INTA Bit[18]: INTB Bit[19]: INTC Bit[20]: INTD
7:4		0	RO	-	**Reserved**
3	**ENLHTO-INT**	0H	RW	-	**Enable LPC host time-out interrupt** 0: Disable 1: Enable

to next page

from previous page

2	**ENLHES-INT**	0H	RW	-	**Enable LPC host sync error interrupt** 0: Disable 1: Enable
1	**NLHNS-INT**	0H	RW	-	**Enable LPC host no-sync error interrupt** 0: Disable 1: Enable
0	**ENLHDN-INT**	0H	RW	-	**Enable LPC host cycle done interrupt** 0: Disable 1: Enable

LHCR3: LPC Host Control Register 3 **Offset: ACh**

Bit	Name	Initial	Slave	Host	Description
31	**LHBUSY**	0L	RO	-	**LPC host busy cycle** 0: LPC host is in idle cycle 1: LPC host is in busy cycle This is read only register reflecting the status of LPC host controller.
30	**LHWAIT**	0L	RW1C	-	**LPC host waiting cycle** 0: LPC host is not in waiting cycles 1: LPC host is in waiting cycles This is read only register reflecting the status of LPC host controller.
29	**STLSIRQW**	0H	RW1C	-	**Reserved**
28:8	**STR_LHSRINT**	0H	RW1C	-	**LPC host SIRQ interrupt status Bit [20:0]** Each bit of this register represents the interrupt status of the 20 IRQ sources listed below (0: no interrupt, 1: interrupt pending) Bit[0]: IRQ0 Bit[1]: IRQ1 .... Bit[15]: IQR15 Bit[16]: IOCK Bit[17]: INTA Bit[18]: INTB Bit[19]: INTC Bit[20]: INTD Writing '1' to this each bit will clear the status of the corresponding interrupt.
7:4		0	RO	-	**Reserved**
3	**STR_LHTOINT**	0H	RW1C	-	**LPC host time-out error interrupt status** 0: No interrupt 1: Time-out error interrupt is pending Writing '1' to this bit will clear the status. Time-out means the period of waiting sync is longer than time-out limit (LHCR1 [31:16]).
2	**STR_LHESINT**	0H	RW1C	-	**LPC host sync error interrupt status** 0: No interrupt 1: Sync error interrupt is pending Writing '1' to this bit will clear the status. Sync error means the LPC target device needs to aid higher layers with more robust error recovery.

to next page

from previous page

1	**STR_LHNSINT**	0H	RW1C	-	**LPC host no-sync interrupt status** 0: No interrupt 1: No-sync interrupt is pending Writing '1' to this bit will clear the status. No-sync means no LPC device acknowledge.
0	**STR_LHDNINT**	0H	RW1C	-	**LPC host cycle done interrupt status** 0: No interrupt 1: Cycle done interrupt is pending Writing '1' to this bit will clear the status.

LHCR4: LPC Host Control Register 4 **Offset: B0h**

Bit	Name	Initial	Slave	Host	Description
31:25	**P2LBASE**	U	RW	-	**Remapping address base bit [31:25] of APB to LPC bridge**
24:8		0	RO	-	**Reserved**
7:4	**LHCMD**	U	RW	-	**LPC host command Bit[3:0]** LPC host cycles will issue LPC bus cycles with commands regarding to the content of this register. S/W takes the responsibility to provide valid commands.
3:0	**LHHDR**	U	RW	-	**LPC host start header Bit[3:0]** LPC host controller will issue LPC bus cycles with start headers regarding to the content of this register. S/W takes the responsibility to provide valid headers.

LHCR5: LPC Host Control Register 5 **Offset: B4h**

Bit	Name	Initial	Slave	Host	Description
31:0	**LHADR**	U	RW	-	**LPC host address Bit[31:0]** LPC Host Controller will issue LPC bus cycles with address regarding the content of this register. The valid address bits, which can be either 16 bits or 32 bits, are automatically determined by LPC host command (LHCR4[7:4]) and LPC host start header (LHCR4[3:0]). Also, this register is defined for LPC I/O pattern search A Bit[31:24]: pattern A_IV Bit[23:16]: pattern A_III Bit[15: 8]: pattern A_II Bit[7: 0]: pattern A_I

LHCR6: LPC Host Control Register 6 **Offset: B8h**

Bit	Name	Initial	Slave	Host	Description
31:0	**LHTXD**	U	RW	-	**LPC host write data Bit[31:0]** LPC host write cycles will write out data based on the content of this register. The valid bytes of the write data is determined by LHCR4 [7:4]. S/W needs to check the valid bytes for each write cycle. Also, this register is defined for LPC I/O pattern search B Bit[31:24]: pattern B_IV Bit[23:16]: pattern B_III Bit[15: 8]: pattern B_II Bit[7: 0]: pattern B_I

LHCR7: LPC Host Control Register 7 **Offset: BCh**

Bit	Name	Initial	Slave	Host	Description
31:0	**LHRXD**	U	RO	-	**LPC host read data Bit[31:0]** This register will record the data latched from the last LPC host read cycle. It will be updated whenever a new LPC host read cycle has been issued. The valid bytes of the read data is automatically determined by LPC host command (LHCR4 [7:4]). S/W needs to check the valid bytes for each read cycle.

LHCR8: LPC Host Control Register 8 **Offset: C0h**

Bit	Name	Initial	Slave	Host	Description
31:27		1DhL	RO	-	**LPC host SIRQ frame number**
26:24		0L	RO	-	**LPC host SIRQ sub-frame number**
23:21		0	RO	-	**Reserved**
20: 0		0L	RO	-	**LPC host SIRQ state bit [20:0]**

PCCR6: Post Code Control Register 6 **Offset: C4h**

Bit	Name	Initial	Slave	Host	Description
31:30		0	RO	-	**Reserved**
29:28		0	RO	-	**Post Code DMA current write pointer bit[1:0]**
27: 0		0	RO	-	**Post Code DMA current address bit[27:0]**

LHCRA: LPC Host Control Register A **Offset: C8h**

Bit	Name	Initial	Slave	Host	Description
31:21		0	RO	-	**Reserved**
20: 0	**LSIRQEG**	U	RW	-	**LPC host SIRQ Bit[20:0] edge trigger mode** 0: level trigger 1: edge trigger Also, this register is defined for LPC I/O pattern search A Bit[18:17]: length or times Bit[16]: write cycle Bit[15: 0]: address

LHCRB: LPC Host Control Register B **Offset: CCh**

Bit	Name	Initial	Slave	Host	Description
31:21		0	RO	-	**Reserved**
20: 0	**LSIRQHV**	U	RW	-	**LPC host SIRQ Bit[20:0] high/rising trigger mode** 0: low level or falling edge trigger 1: high level or rising edge trigger Also, this register is defined for LPC I/O pattern search B Bit[18:17]: length or times Bit[16]: write cycle Bit[15: 0]: address

PCCR4: Post Code Control Register 4 **Offset: D0h**

Bit	Name	Initial	Slave	Host	Description
31: 2		U	RW	-	**Post Code DMA address bit[31:2]**
1: 0		0	RO	-	**Reserved**

PCCR5: Post Code Control Register 5 **Offset: D4h**

Bit	Name	Initial	Slave	Host	Description
31:28		1	RO	-	**Version ID**
27:24		0	RW	-	**Reserved**
23: 0		U	RW	-	**Post Code DMA length bit[23:0] (unit: 4-byte)**

HICRB: Host Interface Control Register B **Offset: 100h**

Bit	Name	Initial	Slave	Host	Description
31		0P	RW	-	**DisSIOLDN15: Disables the SIO LDN #15, SIO LSAFS. Refer section 47.1 for list of SIO LDNs. Default is Enabled.**
30		0P	RW	-	**DisSIOLDN14: Disables the SIO LDN #14, SIO Mailbox. Refer section 47.1 for list of SIO LDNs. Default is Enabled.**
29		1P	RW	-	**DisSIOLDN13: Disables the SIO LDN #13, SIO iLPC2AHB. Refer section 47.1 for list of SIO LDNs. Default is Enabled.** **Programming a 1 in this register disables SIO iLPC2AHB read/write function.** **Note: The iLPC2AHB is feature in the AST BMC allowing the host side to be able to access the BMC registers over LPC/eSPI.** **The SocFlash utility uses this feature.** **Some customers use the SocFlash utility during manufacturing, debug, testing phases.** **If you do not want this functionality in your production systems, this bit should be set to 1.** **In addition to this bit refer to HICRB[6] below to disable the write only capability through iLPC2AHB.**

to next page

from previous page

28		0P	RW	-	**DisSIOLDN12: Disables the SIO LDN #12, SIO UART4. Refer section 47.1 for list of SIO LDNs. Default is Enabled.**
27		0P	RW	-	**DisSIOLDN11: Disables the SIO LDN #11, SIO UART3. Refer section 47.1 for list of SIO LDNs. Default is Enabled.**
26:24		0P	RW	-	**Reserved**
23		0P	RW	-	**DisSIOLDN7: Disables the SIO LDN #7, SIO GPIO. Refer section 47.1 for list of SIO LDNs. Default is Enabled.**
22:21		0P	RW	-	**Reserved**
20		0P	RW	-	**DisSIOLDN4: Disables the SIO LDN #4, SIO SWC. Refer section 47.1 for list of SIO LDNs. Default is Enabled.**
19		0P	RW	-	**DisSIOLDN3: Disables the SIO LDN #3, SIO UART2. Refer section 47.1 for list of SIO LDNs. Default is Enabled.**
18		0P	RW	-	**DisSIOLDN2: Disables the SIO LDN #2, SIO UART1. Refer section 47.1 for list of SIO LDNs. Default is Enabled.**
17:16		0P	RW	-	**Reserved**
15		0P	RW	-	**EnSNP1D: Enable ACCEPT response code for snoop #1 commands, defined in HICR5[2], in eSPI mode.** **By default the eSPI controller in AST2600 provides a NO_RESPONSE command for the POST CODE cycles on the eSPI interface.** **Programming this bit to 1 enables ACCEPT response code for POST CODE cycles when matching the snoop #1, defined in HICR9[31:16], in eSPI mode.**
14		0P	RW	-	**EnSNP0D: Enable ACCEPT response code for snoop #0 commands, defined in HICR5[0], in eSPI mode.** **By default the eSPI controller in AST2600 provides a NO_RESPONSE command for the POST CODE cycles on the eSPI interface.** **Programming this bit to 1 enables ACCEPT response code for POST CODE cycles when matching the snoop #0, defined in HICR9[15:0], in eSPI mode.**
13		0P	RW	-	**En80hSGIO: Enable POST code 80h to SGPIO function when HICR5[31]=1. Will replace serial data GPIO500[7:0].**
12		0P	RW	-	**Enable SIO to BMC interrupt by one of HISR1[31:24] changing**
11: 8		0P	RW	-	**Enable SIO to BMC interrupt by SIORx_20[3:0]**
7		0P	RW	-	**A16E2L: Enable SIO full 16-bit address decoding in IPMI over PCIe mode. No effect in eSPI/LPC mode**

to next page

from previous page

6		1P	RW	-	**Disable SIO iLPC2AHB write function.** **Programming a 1 in this register disables SIO iLPC2AHB write function.** **Note: The iLPC2AHB is feature in the AST BMC allowing the host side to be able to access the BMC registers over LPC/eSPI.** **The SocFlash utility uses this feature.** **Some customers use the SocFlash utility during manufacturing, debug, testing phases.** **If you do not want this functionality in your production systems, this bit should be set to 1.** **In addition to this bit refer to HICRB[29] above to disable the read/write capability through iLPC2AHB.**
5		0P	RW	-	**En16LADR2: Relax KCS2 I/O address limitation of base2+0x0 (defined in LADR12H/LADR12L) and base2+0x4.** **By default the KCS2 I/O decoder decodes only the address base2+0x0 (defined in LADR12H/LADR12L) and base2+0x4.** **For example, the I/O locations 62h and 66h.** **Programming this bit to 1 enables the KCS2 I/O address decode for base2A (defined in LADR12H/LADR12L) and base2B (defined in LSADR12[31:16]).**
4		0P	RW	-	**En16LADR1: Relax KCS1 I/O address limitation of base1+0x0 (defined in LADR12H/LADR12L) and base1+0x4.** **By default the KCS1 I/O decoder decodes only the address base1+0x0 (defined in LADR12H/LADR12L) and base1+0x4.** **For example, the I/O locations 60h and 64h.** **Programming this bit to 1 enables the KCS1 I/O address decode for base1A (defined in LADR12H/LADR12L) and base1B (defined in LSADR12[15:0]).**
3: 2		0	RO	-	**Reserved**
1		0P	RW	-	**Enable KCS channel #4 receive completion interrupt**
0		0P	RW	-	**Enable KCS channel #4**

HICRC: Host Interface Control Register C **Offset: 104h**

Bit	Name	Initial	Slave	Host	Description
31:8		0	RO	-	**Reserved**
7: 4		U	RW	-	**Select ID bit[3:0] of SerIRQ for KCS channel #4**
3: 2		0P	RW	-	**Host SerIRQ interrupt type for KCS channel #4** 00: low level trig 01: high level trig 10: reserved 11: rising edge trig

to next page

from previous page

1		0P	RW	-	**Disable Host SerIRQ interrupt of KCS channel #4 auto-clear by OBF4**
0		0P	RW	-	**Host SerIRQ interrupt enable for KCS channel #4**

HISR0: Host Interface Scratch Register 0 **Offset: 108h**

Bit	Name	Initial	Slave	Host	Description
31:16		0P	RW	-	**Reserved**
15		0P	RW	-	**Inverse UART4 NRTS4 polarity in RS485**
14		0P	RW	-	**Enable UART4 NRTS4 as RS485 DE/nRE**
13		0P	RW	-	**Inverse UART3 NRTS3 polarity in RS485**
12		0P	RW	-	**Enable UART3 NRTS3 as RS485 DE/nRE**
11		0P	RW	-	**Inverse UART2 NRTS2 polarity in RST485**
10		0P	RW	-	**Enable UART2 NRTS2 as RS485 DE/nRE**
9		0P	RW	-	**Inverse UART1 NRTS1 polarity in RS485**
8		0P	RW	-	**Enable UART1 NRTS1 as RS485 DE/nRE**
7: 4		0P	RW	-	**Reserved**
3: 0		0P	RW	-	**BMC to SIO interrupt bit[3:0] by SIORx_21[7:4]**

HISR1: Host Interface Scratch Register 1 **Offset: 10Ch**

Bit	Name	Initial	Slave	Host	Description
31:10		0P	RW	-	**Reserved**
9		0P	RW	-	**clr_str4_IBF: Mask IBF4**
8		0P	RW1C	-	**Status of SIO to BMC interrupt by one of HISR1[31:24] changing**
7: 4		0P	RW1C	-	**Status of SIO to BMC interrupt by SIORx_21[3:0]**
3: 0		0P	RO	-	**SIORx_21[3:0]: SIO SIORx_21[3:0] status**

LADR4: LPC Channel #4 Address register **Offset: 110h**

Bit	Name	Initial	Slave	Host	Description
31:16		U	RW	-	**KCS channel #4 second (command/status) base register address bit[15:0]**
15: 0		U	RW	-	**KCS channel #4 first (data) base register address bit[15:0]**

IDR4: Input Data Register 4 **Offset: 114h**

Bit	Name	Initial	Slave	Host	Description
7: 0		U	RO	WO	**KCS channel #4 input data bit[7:0]**

ODR4: Output Data Register 4 **Offset: 118h**

Bit	Name	Initial	Slave	Host	Description
7: 0		U	RW	RO	**KCS channel #4 output data bit[7:0]**

STR4: Status Register 4 **Offset: 11Ch**

Bit	Name	Initial	Slave	Host	Description
7	**DBU47**	1H	RW	RO	**Defined by user**
6	**DBU46**	1H	RW	RO	**Defined by user**
5	**DBU45**	0H	RW	RO	**Defined by user**
4	**DBU44**	0H	RW	RO	**Defined by user**
3	**C/D4**	0P	RO	RO	**Command/Data**
2	**DBU42**	0H	RW	RO	**Defined by user**
1	**IBF4**	0P	RO	RO	**Input data register full**
0	**OBF4**	0P	RW0C	RO	**Output data register full**

LSADR12: LPC Channel #1/#2 Status Address Register **Offset: 120h**

Bit	Name	Initial	Slave	Host	Description
31:16		U	RW	-	**KCS channel #2 status address bit[15:0]**
15: 0		U	RW	-	**KCS channel #1 status address bit[15:0]**

PCCR0: Post Code Control Register 0 **Offset: 130h**

Bit	Name	Initial	Slave	Host	Description
31		0P	RW	-	**Enable post code DMA interrupt**
30:29		3P	RW	-	**DMA timeout selection** 00: 16 APB clocks 01: 240 APB clocks 10: 3840 APB clocks 11: 61440 APB clocks
28:24		U	RW	-	**SerIRQ monitoring selection bit[4:0]**
23		0P	RW	-	**Enable pattern search B interrupt**
22		0P	RW	-	**Enable pattern search B**
21		0P	RW	-	**Enable pattern search A interrupt**
20		0P	RW	-	**Enable pattern search A**
19:16		U	RW	-	**Time out period selection bit[3:0] of RxTO** 0000: 16 LPC clocks 0001: 32 LPC clocks 0010: 64 LPC clocks ... 1111: 524288 LPC clocks
15		0	RO	-	**Reserved**
14		0P	RW	-	**Enable post code DMA mode**
13:12		U	RW	-	**Address selection bit[1:0] of the post code function** 00: record SerIRQ; not high address 01: address bit[5:4] 10: address bit[7:6] 11: address bit[9:8] Also, it controls the parser of read/write cycles for I/O 1B mode. 00: parse write only 01: parse read only 1x: parse read/write

to next page

from previous page

11		0	RO	-	**Reserved**
10: 8		0P	RW	-	**Trigger level bit[2:0] of Rx FIFO** 000: 1 byte 001: 1/8 FIFO 010: 2/8 FIFO 011: 3/8 FIFO 100: 4/8 FIFO 101: 5/8 FIFO 110: 6/8 FIFO 111: 7/8 FIFO
7		0	RW	-	**Write 1 clear Rx FIFO**
6		0P	RW	-	**Reserved**
5: 4		0P	RW	-	**Mode selection bit[1:0] of the post code function** 00: I/O 1B mode 01: I/O 2B mode 10: I/O 4B mode 11: full mode
3		0P	RW	-	**Enable post code RxOvr interrupt**
2		0P	RW	-	**Enable post code RxTO interrupt**
1		0P	RW	-	**Enable post code RxAva interrupt**
0		0P	RW	-	**Enable post code function**

PCCR1: Post Code Control Register 1 **Offset: 134h**

Bit	Name	Initial	Slave	Host	Description
31:24		0P	RW	-	**FIFO read pointer bit[7:0]**
23		0	RW1S	-	**Enable write of FIFO read pointer bit[7:0]**
22		0	RW	-	**Reserved**
21:16		U	RW	-	**LPC dont care address bit[9:8]/bit[7:6]/bit[5:4] and bit[3:0]**
15: 0		U	RW	-	**LPC captured base address bit[15:0]**

PCCR2: Post Code Control Register 2 **Offset: 138h**

Bit	Name	Initial	Slave	Host	Description
31:24		0P	RO	-	**FIFO write pointer bit[7:0]**
23:18		0P	RO	-	**State of the pattern search B**
17		0	RW1S	-	**Reset the state of the pattern search B**
16		0P	RW1C	-	**Interrupt status of pattern search B**
15:10		0P	RO	-	**State of the pattern search A**
9		0	RW1S	-	**Reset the state of the pattern search A**
8		0P	RW1C	-	**Interrupt status of pattern search A**
7		0	RO	-	**Reserved**
6		0P	RO	-	**FIFO full indicator**
5		0P	RO	-	**Rx data count bit[8]**
4		0P	RW1C	-	**Data ready indicator (read only) if non-DMA mode DMA done indicator (read and write 1 clear) if DMA mode**

to next page

from previous page

3		0P	RW1C	-	**Interrupt status of RxOvr**
2		0P	RO	-	**Interrupt status of RxTO**
1		0P	RO	-	**Interrupt status of RxAva**
0		0P	RO	-	**Interrupt status of the post code function**

PCCR3: Post Code Control Register 3 **Offset: 13Ch**

Bit	Name	Initial	Slave	Host	Description
31:24		0P	RO	-	**FIFO write pointer bit[7:0]**
23:16		0P	RO	-	**FIFO read pointer bit[7:0]**
15: 8		0P	RO	-	**Rx data count bit[7:0]**
7: 0		U	RO	-	**FIFO data bit[7:0]**

Note :
The data structure of the FIFO is mode dependent.

For storing 1B mode:
dataX[7:0], dataY[7:0], ...

For storing 2B mode:
dataX[7:0], abort_write_sirqTa_sirqTb_addrX[3:0],
dataY[7:0], abort_write_sirqTa_sirqTb_addrY[3:0], ..., if PCCR0[13:12]=00
dataX[7:0], abort_write_addrX[5:4]_addrX[3:0],
dataY[7:0], abort_write_addrY[5:4]_addrY[3:0], ..., if PCCR0[13:12]=01
dataX[7:0], abort_write_addrX[7:6]_addrX[3:0],
dataY[7:0], abort_write_addrY[7:6]_addrY[3:0], ..., if PCCR0[13:12]=10
dataX[7:0], abort_write_addrX[9:8]_addrX[3:0],
dataY[7:0], abort_write_addrY[9:8]_addrY[3:0], ..., if PCCR0[13:12]=11

For storing 4B mode:
dataX[7:0], abort_write_sirqTa_sirqTb_ioCycN_fwhCyc_ln4679[1:0],
addrX[7:0], addrX[15:8],
dataY[7:0], abort_write_sirqTa_sirqTb_ioCycN_fwhCyc_ln4679[1:0],
addrY[7:0], addrY[15:8], ...

For fully storing mode:
dataX[7:0], abort_write_sirqTa_sirqTb_ioCycN_fwhCyc_ln4679[1:0],
addrX[7:0], addrX[15:8], addrX[23:16], addrX[31:24],
dataX[15:8], dataX[23:16], dataX[31:24],
dataY[7:0], abort_write_sirqTa_sirqTb_ioCycN_fwhCyc_ln4679[1:0],
addrY[7:0], addrY[15:8], addrY[23:16], addrY[31:24],
dataY[15:8], dataY[23:16], dataY[31:24], ...

iBTCR0: iBT Control Register 0 **Offset: 140h**

Bit	Name	Initial	Slave	Host	Description
31:18		U	RW	-	**iBT LPC base address bit[15:2]**
17:16		0	RO	-	**Reserved**
15:12		AP	RW	-	**Select ID bit[3:0] of SerIRQ for iBT**
11:10		0P	RW	-	**Host SerIRQ interrupt type for iBT** 00: low level trig 01: high level trig 10: reserved 11: rising edge trig

to next page

from previous page

9: 4		0P	RW	-	**Reserved**
3	**EnClrSvRdP**	0P	RW	-	**Enable iBT clear slave read pointer by host write pointer clear**
2	**EnClrSvWrP**	0P	RW	-	**Enable iBT clear slave write pointer by H_BUSY falling**
1		0P	RW	-	**Enable iBT slave reset read**
0		0P	RW	-	**Enable iBT**

iBTCR1: iBT Control Register 1 **Offset: 144h**

Bit	Name	Initial	Slave	Host	Description
31:28		1	RO	-	**Version ID**
27		0	RO	-	**FIFO read pointer bit[8] of BMC to Host**
26		0	RO	-	**FIFO write pointer bit[8] of BMC to Host**
25		0	RO	-	**FIFO read pointer bit[8] of Host to BMC**
24		0	RO	-	**FIFO write pointer bit[8] of Host to BMC**
23:18		0	RO	-	**Reserved**
17:16	**EnSIRQOEM**	0H	RW	-	**Reserved**
15		0H	RW	-	**Enable interrupt of iBT BMC_HWRST**
14		0H	RW	-	**Enable interrupt of iBT OEM3**
13		0H	RW	-	**Enable interrupt of iBT OEM2**
12		0H	RW	-	**Enable interrupt of iBT OEM1**
11		0H	RW	-	**Enable interrupt of iBT slave read overrun**
10		0H	RW	-	**Enable interrupt of iBT slave write overrun**
9		0H	RW	-	**Enable interrupt of iBT host read overrun**
8		0H	RW	-	**Enable interrupt of iBT host write overrun**
7		0H	RW	-	**Reserved**
6		0H	RW	-	**Enable interrupt of iBT HBusy falling**
5		0H	RW	-	**Enable interrupt of iBT SMS falling**
4		0H	RW	-	**Enable interrupt of iBT B2H falling**
3: 2		0H	RW	-	**Reserved**
1		0H	RW	-	**Enable interrupt of iBT OEM0 rising**
0		0H	RW	-	**Enable interrupt of iBT H2B rising**

iBTCR2: iBT Control Register 2 **Offset: 148h**

Bit	Name	Initial	Slave	Host	Description
31:24		U	RO	-	**FIFO data bit[7:0] of BMC to Host**
23:16		U	RO	-	**FIFO data bit[7:0] of Host to BMC**
15		0P	RO	-	**Interrupt status of iBT BMC_HWRST**
14		0H	RO	-	**Interrupt status of iBT OEM3**
13		0H	RO	-	**Interrupt status of iBT OEM2**
12		0H	RO	-	**Interrupt status of iBT OEM1**
11		0H	RW1C	-	**Interrupt status of iBT slave read overrun**

to next page

from previous page

10		0H	RW1C	-	**Interrupt status of iBT slave write overrun**
9		0H	RW1C	-	**Interrupt status of iBT host read overrun**
8		0H	RW1C	-	**Interrupt status of iBT host write overrun**
7		0H	RW	-	**Reserved**
6		0H	RW1C	-	**Interrupt status of iBT HBusy falling**
5		0H	RW1C	-	**Interrupt status of iBT SMS falling**
4		0H	RW1C	-	**Interrupt status of iBT B2H falling**
3: 2		0H	RW	-	**Reserved**
1		0H	RW1C	-	**Interrupt status of iBT OEM0 rising**
0		0H	RW1C	-	**Interrupt status of iBT H2B rising**

iBTCR3: iBT Control Register 3 **Offset: 14Ch**

Bit	Name	Initial	Slave	Host	Description
31:24		0L	RO	-	**FIFO read pointer bit[7:0] of BMC to Host**
23:16		0H	RO	-	**FIFO write pointer bit[7:0] of BMC to Host**
15: 8		0H	RO	-	**FIFO read pointer bit[7:0] of Host to BMC**
7: 0		0L	RO	-	**FIFO write pointer bit[7:0] of Host to BMC**

iBTCR4: iBT Control Register 4 **Offset: 150h**

Bit	Name	Initial	Slave	Host	Description
31: 8		0	RO	-	**Reserved**
7		1H	RW1T	RO	**B_BUSY**
6		0L	RO	RW1T	**H_BUSY**
5		0H	RW1C	RW1S	**OEM0**
4		0H	RW1S	RW1C	**SMS_ATN**
3		0H	RW1S	RW1C	**B2H_ATN**
2		0H	RW1C	RW1S	**H2B_ATN**
1		0	RW1C	RW1C	**CLR_RD_PTR**
0		0	RW1C	RW1C	**CLR_WR_PTR**

iBTCR5: iBT Control Register 5 **Offset: 154h**

Bit	Name	Initial	Slave	Host	Description
31: 8		0	RO	-	**Reserved**
7: 0		U	RW	RW	**FIFO data bit[7:0]; FIFO size is 256 bytes**

iBTCR6: iBT Control Register 6 **Offset: 158h**

Bit	Name	Initial	Slave	Host	Description
31: 8		0	RO	-	**Reserved**
7		0P	RW1C	RW1S	**BMC_HWRST**
6: 5		0H	RW	RW	**Reserved**
4		0H	RW	RW	**OEM3**

to next page

from previous page

3		0H	RW	RW	**OEM2**
2		0H	RW	RW	**OEM1**
1		0L	RO	RW	**B2H_IRQ**
0		0L	RO	RW	**B2H_IRQ_EN**

SRUART1: Status Register of UART1 **Offset: 160h**

Bit	Name	Initial	Slave	Host	Description
31:30		0	RO	-	**RxFIFO trigger level bit [1:0]**
29		0	RO	-	**Buadrate write**
28		0	RO	-	**EnFIFO status**
27:24		0	RO	-	**MSR bit [3:0]**
23:16		0	RO	-	**LSR bit [7:0]**
15:12		1	RO	-	**IIR bit [3:0]**
11		0	RO	-	**Buadrate bit [11] OR non-16550 LCR error**
10		0	RO	-	**Buadrate bit [10] OR non-16550 LCR mode**
9		0	RO	-	**Buadrate bit [9] OR non-16550 reset mode**
8: 0		0	RO	-	**Buadrate bit [8:0]**

SRUART2: Status Register of UART2 **Offset: 164h**

Bit	Name	Initial	Slave	Host	Description
31:30		0	RO	-	**RxFIFO trigger level bit [1:0]**
29		0	RO	-	**Buadrate write**
28		0	RO	-	**EnFIFO status**
27:24		0	RO	-	**MSR bit [3:0]**
23:16		0	RO	-	**LSR bit [7:0]**
15:12		1	RO	-	**IIR bit [3:0]**
11		0	RO	-	**Buadrate bit [11] OR non-16550 LCR error**
10		0	RO	-	**Buadrate bit [10] OR non-16550 LCR mode**
9		0	RO	-	**Buadrate bit [9] OR non-16550 reset mode**
8: 0		0	RO	-	**Buadrate bit [8:0]**

SRUART3: Status Register of UART3 **Offset: 168h**

Bit	Name	Initial	Slave	Host	Description
31:30		0	RO	-	**RxFIFO trigger level bit [1:0]**
29		0	RO	-	**Buadrate write**
28		0	RO	-	**EnFIFO status**
27:24		0	RO	-	**MSR bit [3:0]**
23:16		0	RO	-	**LSR bit [7:0]**
15:12		1	RO	-	**IIR bit [3:0]**
11		0	RO	-	**Buadrate bit [11] OR non-16550 LCR error**
10		0	RO	-	**Buadrate bit [10] OR non-16550 LCR mode**
9		0	RO	-	**Buadrate bit [9] OR non-16550 reset mode**

to next page

from previous page

8: 0		0	RO	-	**Buadrate bit [8:0]**

SRUART4: Status Register of UART4 **Offset: 16Ch**

Bit	Name	Initial	Slave	Host	Description
31:30		0	RO	-	**RxFIFO trigger level bit [1:0]**
29		0	RO	-	**Buadrate write**
28		0	RO	-	**EnFIFO status**
27:24		0	RO	-	**MSR bit [3:0]**
23:16		0	RO	-	**LSR bit [7:0]**
15:12		1	RO	-	**IIR bit [3:0]**
11		0	RO	-	**Buadrate bit [11] OR non-16550 LCR error**
10		0	RO	-	**Buadrate bit [10] OR non-16550 LCR mode**
9		0	RO	-	**Buadrate bit [9] OR non-16550 reset mode**
8: 0		0	RO	-	**Buadrate bit [8:0]**

SCR0SIO: Scratch Register 0 of SIO **Offset: 170h**

Bit	Name	Initial	Slave	Host	Description
31:24		0P	RW	-	**SIORx_2B**
23:16		0P	RW	-	**SIORx_2A**
15: 8		0P	RW	-	**SIORx_29**
7: 0		A8hP	RW	-	**SIORx_28**

SCR1SIO: Scratch Register 1 of SIO **Offset: 174h**

Bit	Name	Initial	Slave	Host	Description
31:24		0P	RW	-	**SIORx_2F**
23:16		0P	RW	-	**SIORx_2E**
15: 8		0P	RW	-	**SIORx_2D**
7: 0		0P	RW	-	**SIORx_2C**

SCR2SIO: Scratch Register 2 of SIO **Offset: 178h**

Bit	Name	Initial	Slave	Host	Description
31:24		0P	RO	-	**SIORx_24**
23:16		0P	RO	-	**SIORx_23**
15: 8		0P	RO	-	**SIORx_22**
7: 0		0P	RO	-	**SIORx_21**

SCR3SIO: Scratch Register 3 of SIO **Offset: 17Ch**

Bit	Name	Initial	Slave	Host	Description
31		0P	RO	-	**Status of SIO SMI enable**
30		0P	RO	-	**Status of SIO iLPC2AHB enable**
29		0P	RO	-	**Status of SIO GPIO enable**

to next page

from previous page

28		0P	RO	-	**Status of SIO KBC enable**
27		0P	RO	-	**Status of SIO SUART2 enable**
26		0P	RO	-	**Status of SIO SUART1 enable**
25		0P	RO	-	**Status of SIO SUART4 enable**
24		0P	RO	-	**Status of SIO SUART3 enable**
23:16		0P	RO	-	**SIORx_27**
15: 8		0P	RO	-	**SIORx_26**
7: 0		0P	RO	-	**SIORx_25**

SWCR_03_00: SWC Register 03_00 **Offset: 180h**

Bit	Name	Initial	Slave	Host	Description
31		0P	RW1C	-	**Setting sleep state event status (INT#45)**
30		0P	RO	-	**PWRGD status after debounce**
29		0P	RW1C	-	**PWRGD rising event status (INT#45)**
28		0P	RW1C	-	**PWRGD falling event status (INT#45)**
27		1P	RO	-	**PWRBTN status after debounce**
26		0P	RW1C	-	**PWRBTN rising event status (INT#45)**
25		0P	RW1C	-	**PWRBTN falling event status (INT#45)**
24		1P	RO	-	**RI status after debounce**
23		0P	RW1C	-	**RI rising event status (INT#45)**
22		0P	RW1C	-	**RI falling event status (INT#45)**
21		0P	RO	-	**S5n status after debounce**
20		0P	RW1C	-	**S5n rising event status (INT#45)**
19		0P	RW1C	-	**S5n falling event status (INT#45)**
18		0P	RO	-	**S3n status after debounce**
17		0P	RW1C	-	**S3n rising event status (INT#45)**
16		0P	RW1C	-	**S3n falling event status (INT#45)**
15		1P	RO	-	**PWBTO raw status**
14		0P	RO	-	**Last_ONCTL status**
13		1P	RW	-	**Was_pfail status**
12		0P	RW1C	-	**Crowbar status (INT#45)**
11		0P	RW1C	-	**PWRBTN Override status (INT#45)**
10			RO	-	**Reserved**
9		0P	RW1C	-	**RI wake-up event status**
8		0P	RW1C	-	**BMC trigger wake-up event status**
7: 0		0P	RW1C	-	**GPIO wake-up event status bit[7:0]**

SWCR_07_04: SWC Register 07_04 **Offset: 184h**

Bit	Name	Initial	Slave	Host	Description
31		0P	RW	-	**Enable BMC interrupt as setting sleep state event**
30			RO	-	**Reserved**
29		0P	RW	-	**Enable BMC interrupt as PWRGD rising event**

to next page

from previous page

28		0P	RW	-	**Enable BMC interrupt as PWRGD falling event**
27			RO	-	**Reserved**
26		0P	RW	-	**Enable BMC interrupt as PWRBTN rising event**
25		0P	RW	-	**Enable BMC interrupt as PWRBTN falling event**
24			RO	-	**Reserved**
23		0P	RW	-	**Enable BMC interrupt as RI rising event**
22		0P	RW	-	**Enable BMC interrupt as RI falling event**
21			RO	-	**Reserved**
20		0P	RW	-	**Enable BMC interrupt as S5n rising event**
19		0P	RW	-	**Enable BMC interrupt as S5n falling event**
18			RO	-	**Reserved**
17		0P	RW	-	**Enable BMC interrupt as S3n rising event**
16		0P	RW	-	**Enable BMC interrupt as S3n falling event**
15		0P	RW	-	**Force PWBTO output**
14		0P	RW	-	**Enable PWREQ output**
13		0P	RW	-	**Select RI event from UART2 or UART1**
12		0P	RW	-	**Disable sleep entering as Crowbar**
11		0P	RW	-	**Enable BMC interrupt as PWRBTN Override**
10			RO	-	**Reserved**
9		0P	RW	-	**RI wake-up event enable**
8		0P	RW	-	**BMC trigger wake-up event enable**
7: 0		0P	RW	-	**GPIO wake-up event enable bit[7:0]**

SWCR_0B_08: SWC Register 0B_08 **Offset: 188h**

Bit	Name	Initial	Slave	Host	Description
31:28		0P	RW	-	**time (unit 100ms) of filtering S5n, S3n and ONCTL as AC on**
27		0P	RW	-	**Reserved**
26		0P	RW	-	**Enable 4s PWRBTN Override**
25		0P	RW	-	**PWREQ output level**
24		0P	RW	-	**PWBTO output level**
23		0P	RW	-	**Enable PWBTO from RI wake-up event in S45**
22		0P	RW	-	**Enable PWBTO from RI wake-up event in S3I**
21		0P	RW	-	**Enable ONCTL from RI wake-up event in S45**
20		0P	RW	-	**Enable ONCTL from RI wake-up event in S3I**
19		0P	RW	-	**Enable PWBTO from BMC trigger wake-up event in S45**
18		0P	RW	-	**Enable PWBTO from BMC trigger wake-up event in S3I**
17		0P	RW	-	**Enable ONCTL from BMC trigger wake-up event in S45**
16		0P	RW	-	**Enable ONCTL from BMC trigger wake-up event in S3I**

to next page

from previous page

15		1P	RO	-	**ONCTL status after debounce**
14		0P	RW	-	**GPIO level on ONCTL**
13		0P	RW	-	**Enable GPIO output on ONCTL**
12		0P	RW	-	**Enable GPIO mode on ONCTL**
11		0P	RW	-	**Force internal vale of S5n**
10		0P	RW	-	**Force internal vale of S3n**
9		0P	RW	-	**Enable debounce of S3n and S5n**
8		0P	RW	-	**Disable external wake-up from S3n**
7		0P	RW	-	**Enable software mode of S3n and S5n**
6		0P	W1S	-	**BMC trigger wake-up event**
5		0P	RW	-	**Force sleep state S45 entering**
4		0P	RW	-	**Force sleep state S12 entering**
3		0P	RW	-	**Enable SerIRQ interrupt as RI wake-up event**
2		0P	RW	-	**Enable SMI interrupt as RI wake-up event**
1		0P	RW	-	**Enable SerIRQ interrupt as BMC trigger wake-up event**
0		0P	RW	-	**Enable SMI interrupt as BMC trigger wake-up event**

SWCR_0F_0C: SWC Register 0F_0C **Offset: 18Ch**

Bit	Name	Initial	Slave	Host	Description
31:24		0P	RW	-	**Enable PWBTO from GPIO wake-up event in S45**
23:16		0P	RW	-	**Enable PWBTO from GPIO wake-up event in S3I**
15: 8		0P	RW	-	**Enable ONCTL from GPIO wake-up event in S45**
7: 0		0P	RW	-	**Enable ONCTL from GPIO wake-up event in S3I**

SWCR_13_10: SWC Register 13_10 **Offset: 190h**

Bit	Name	Initial	Slave	Host	Description
31:27			RO	-	**Reserved**
26:24		5P	RW	-	**Sleep type encoding of S3**
23:19			RO	-	**Reserved**
18:16		2P	RW	-	**Sleep type encoding of S2**
15:11			RO	-	**Reserved**
10: 8		1P	RW	-	**Sleep type encoding of S1**
7: 3			RO	-	**Reserved**
2: 0		0P	RW	-	**Sleep type encoding of S0**

SWCR_17_14: SWC Register 17_14 **Offset: 194h**

Bit	Name	Initial	Slave	Host	Description
31:29		0P	RW	-	**Group selection of GPIO wake-up event**
28		0P	RW	-	**Enable BMC interrupt as Crowbar**
27:24		7P	RW	-	**Power up timeout second bit[3:0]**

to next page

from previous page

23			RO	-	**Reserved**
22:21		0P	RO	-	**slp_now_st[1:0]**
20		0P	RW	-	**Lock PWRBTN** 0: PWRBTN is directly output on PWBTO 1: PWBTO is pulsed only if PWRBTN is held low for more than 4 seconds
19		0P	RW	-	**Enable pulse on PWBTO as wake-up event**
18		0P	RW	-	**Enable PWRBTN press wake-up**
17		0P	RW	-	**Enable Host power in S3I**
16		0P	RW	-	**Select (S3I = S3 or S4) mode or (S45 = S4 or S5) mode**
15:11			RO	-	**Reserved**
10: 8		7P	RW	-	**Sleep type encoding of S5**
7: 3			RO	-	**Reserved**
2: 0		6P	RW	-	**Sleep type encoding of S4**

SWCR_1B_18: SWC Register 1B_18 **Offset: 198h**

Bit	Name	Initial	Slave	Host	Description
31:24		0P	RO	-	**SMI current minute count bit[7:0] (unit 1 min)**
23:16		0P	RO	-	**SMI current second count bit[7:0] (unit 125 ms)**
15: 8		0P	RW	-	**SMI minute count bit[7:0] (unit 1 min)**
7: 0		0P	RW	-	**SMI second count bit[7:0] (unit 125 ms)**

SWCR_1F_1C: SWC Register 1F_1C **Offset: 19Ch**

Bit	Name	Initial	Slave	Host	Description
31:24		0P	RW	-	**SWC scratch bit[7:0]**
23:16			RO	-	**Reserved**
15: 8			RO	-	**Reserved**
7: 6			RO	-	**Reserved**
5		0P	RW1C	-	**SMI minute interrupt status**
4		0P	RW1C	-	**SMI second interrupt status**
3		0P	RW	-	**Enable SMI interrupt of minute timer**
2		0P	RW	-	**Enable SMI interrupt of second timer**
1		0P	RW	-	**Enable SMI minute timer**
0		0P	RW	-	**Enable SMI second timer**

ACPI_E3_E0: ACPI Register E3_E0 **Offset: 1A0h**

Bit	Name	Initial	Slave	Host	Description
31		0P	RO	-	**Enable SCI from wake-up event**
30:25			RO	-	**Reserved**
24		0P	RW	-	**Enable SCI from PWRBTN event**
23:16			RO	-	**Reserved**

to next page

from previous page

15		0P	RW1C	-	**Wake-up event status**
14: 9			RO	-	**Reserved**
8		0P	RW1C	-	**PWRBTN event status**
7: 0			RO	-	**Reserved**

ACPI_C1_C0: ACPI Register C1_C0 **Offset: 1A4h**

Bit	Name	Initial	Slave	Host	Description
31:16			RO	-	**Reserved**
15:14			RO	-	**Reserved**
13		0P	RW	-	**Update Sleep Type value**
12:10		0P	RW	-	**Sleep Type bit[2:0]**
9: 8			RO	-	**Reserved**
7: 0			RO	-	**Reserved**

ACPI_B3_B0: ACPI Register B3_B0 **Offset: 1A8h**

Bit	Name	Initial	Slave	Host	Description
31:16			RO	-	**Reserved**
15:10			RO	-	**Reserved**
9			RW1C	-	**RI SCI event status**
8			RW1C	-	**BMC trigger SCI event status**
7: 0			RW1C	-	**GPIO SCI event status bit[7:0]**

ACPI_B7_B4: ACPI Register B7_B4 **Offset: 1ACh**

Bit	Name	Initial	Slave	Host	Description
31:16			RO	-	**Reserved**
15:10			RO	-	**Reserved**
9			RW	-	**RI SCI event enable**
8			RW	-	**BMC trigger SCI event enable**
7: 0			RW	-	**GPIO SCI event enable bit[7:0]**

MBXDAT_0: MailBox Data Register 0 **Offset: 200h**

Bit	Name	Initial	Slave	Host	Description
31:24		0P	RO	-	**MailBox Data Register 3 bit[7:0]**
23:16		0P	RO	-	**MailBox Data Register 2 bit[7:0]**
15: 8		0P	RO	-	**MailBox Data Register 1 bit[7:0]**
7: 0		0P	RW	RW	**MailBox Data Register 0 bit[7:0]**

MBXDAT_1: MailBox Data Register 1 **Offset: 204h**

Bit	Name	Initial	Slave	Host	Description
31: 8		0	RO	-	**Reserved**
7: 0		0P	RW	RW	**MailBox Data Register 1 bit[7:0]**

MBXDAT_2: MailBox Data Register 2 **Offset: 208h**

Bit	Name	Initial	Slave	Host	Description
31: 8		0	RO	-	**Reserved**
7: 0		0P	RW	RW	**MailBox Data Register 2 bit[7:0]**

MBXDAT_3: MailBox Data Register 3 **Offset: 20Ch**

Bit	Name	Initial	Slave	Host	Description
31: 8		0	RO	-	**Reserved**
7: 0		0P	RW	RW	**MailBox Data Register 3 bit[7:0]**

MBXDAT_4: MailBox Data Register 4 **Offset: 210h**

Bit	Name	Initial	Slave	Host	Description
31:24		0P	RO	-	**MailBox Data Register 7 bit[7:0]**
23:16		0P	RO	-	**MailBox Data Register 6 bit[7:0]**
15: 8		0P	RO	-	**MailBox Data Register 5 bit[7:0]**
7: 0		0P	RW	RW	**MailBox Data Register 4 bit[7:0]**

MBXDAT_5: MailBox Data Register 5 **Offset: 214h**

Bit	Name	Initial	Slave	Host	Description
31: 8		0	RO	-	**Reserved**
7: 0		0P	RW	RW	**MailBox Data Register 5 bit[7:0]**

MBXDAT_6: MailBox Data Register 6 **Offset: 218h**

Bit	Name	Initial	Slave	Host	Description
31: 8		0	RO	-	**Reserved**
7: 0		0P	RW	RW	**MailBox Data Register 6 bit[7:0]**

MBXDAT_7: MailBox Data Register 7 **Offset: 21Ch**

Bit	Name	Initial	Slave	Host	Description
31: 8		0	RO	-	**Reserved**
7: 0		0P	RW	RW	**MailBox Data Register 7 bit[7:0]**

MBXDAT_8: MailBox Data Register 8 **Offset: 220h**

Bit	Name	Initial	Slave	Host	Description
31:24		0P	RO	-	**MailBox Data Register B bit[7:0]**
23:16		0P	RO	-	**MailBox Data Register A bit[7:0]**
15: 8		0P	RO	-	**MailBox Data Register 9 bit[7:0]**
7: 0		0P	RW	RW	**MailBox Data Register 8 bit[7:0]**

MBXDAT_9: MailBox Data Register 9 **Offset: 224h**

Bit	Name	Initial	Slave	Host	Description
31: 8		0	RO	-	**Reserved**
7: 0		0P	RW	RW	**MailBox Data Register 9 bit[7:0]**

MBXDAT_A: MailBox Data Register A **Offset: 228h**

Bit	Name	Initial	Slave	Host	Description
31: 8		0	RO	-	**Reserved**
7: 0		0P	RW	RW	**MailBox Data Register A bit[7:0]**

MBXDAT_B: MailBox Data Register B **Offset: 22Ch**

Bit	Name	Initial	Slave	Host	Description
31: 8		0	RO	-	**Reserved**
7: 0		0P	RW	RW	**MailBox Data Register B bit[7:0]**

MBXDAT_C: MailBox Data Register C **Offset: 230h**

Bit	Name	Initial	Slave	Host	Description
31:24		0P	RO	-	**MailBox Data Register F bit[7:0]**
23:16		0P	RO	-	**MailBox Data Register E bit[7:0]**
15: 8		0P	RO	-	**MailBox Data Register D bit[7:0]**
7: 0		0P	RW	RW	**MailBox Data Register C bit[7:0]**

MBXDAT_D: MailBox Data Register D **Offset: 234h**

Bit	Name	Initial	Slave	Host	Description
31: 8		0	RO	-	**Reserved**
7: 0		0P	RW	RW	**MailBox Data Register D bit[7:0]**

MBXDAT_E: MailBox Data Register E **Offset: 238h**

Bit	Name	Initial	Slave	Host	Description
31: 8		0	RO	-	**Reserved**
7: 0		0P	RW	RW	**MailBox Data Register E bit[7:0]**

MBXDAT_F: MailBox Data Register F **Offset: 23Ch**

Bit	Name	Initial	Slave	Host	Description
31: 8		0	RO	-	**Reserved**
7: 0		0P	RW	RW	**MailBox Data Register F bit[7:0]**

MBXDAT_10: MailBox Data Register 10 **Offset: 240h**

Bit	Name	Initial	Slave	Host	Description
31:24		0P	RO	-	**MailBox Data Register 13 bit[7:0]**
23:16		0P	RO	-	**MailBox Data Register 12 bit[7:0]**

to next page

from previous page

15: 8		0P	RO	-	**MailBox Data Register 11 bit[7:0]**
7: 0		0P	RW	RW	**MailBox Data Register 10 bit[7:0]**

MBXDAT_11: MailBox Data Register 11 **Offset: 244h**

Bit	Name	Initial	Slave	Host	Description
31: 8		0	RO	-	**Reserved**
7: 0		0P	RW	RW	**MailBox Data Register 11 bit[7:0]**

MBXDAT_12: MailBox Data Register 12 **Offset: 248h**

Bit	Name	Initial	Slave	Host	Description
31: 8		0	RO	-	**Reserved**
7: 0		0P	RW	RW	**MailBox Data Register 12 bit[7:0]**

MBXDAT_13: MailBox Data Register 13 **Offset: 24Ch**

Bit	Name	Initial	Slave	Host	Description
31: 8		0	RO	-	**Reserved**
7: 0		0P	RW	RW	**MailBox Data Register 13 bit[7:0]**

MBXDAT_14: MailBox Data Register 14 **Offset: 250h**

Bit	Name	Initial	Slave	Host	Description
31:24		0P	RO	-	**MailBox Data Register 17 bit[7:0]**
23:16		0P	RO	-	**MailBox Data Register 16 bit[7:0]**
15: 8		0P	RO	-	**MailBox Data Register 15 bit[7:0]**
7: 0		0P	RW	RW	**MailBox Data Register 14 bit[7:0]**

MBXDAT_15: MailBox Data Register 15 **Offset: 254h**

Bit	Name	Initial	Slave	Host	Description
31: 8		0	RO	-	**Reserved**
7: 0		0P	RW	RW	**MailBox Data Register 15 bit[7:0]**

MBXDAT_16: MailBox Data Register 16 **Offset: 258h**

Bit	Name	Initial	Slave	Host	Description
31: 8		0	RO	-	**Reserved**
7: 0		0P	RW	RW	**MailBox Data Register 16 bit[7:0]**

MBXDAT_17: MailBox Data Register 17 **Offset: 25Ch**

Bit	Name	Initial	Slave	Host	Description
31: 8		0	RO	-	**Reserved**
7: 0		0P	RW	RW	**MailBox Data Register 17 bit[7:0]**

MBXDAT_18: MailBox Data Register 18 **Offset: 260h**

Bit	Name	Initial	Slave	Host	Description
31:24		0P	RO	-	**MailBox Data Register 1B bit[7:0]**
23:16		0P	RO	-	**MailBox Data Register 1A bit[7:0]**
15: 8		0P	RO	-	**MailBox Data Register 19 bit[7:0]**
7: 0		0P	RW	RW	**MailBox Data Register 18 bit[7:0]**

MBXDAT_19: MailBox Data Register 19 **Offset: 264h**

Bit	Name	Initial	Slave	Host	Description
31: 8		0	RO	-	**Reserved**
7: 0		0P	RW	RW	**MailBox Data Register 19 bit[7:0]**

MBXDAT_1A: MailBox Data Register 1A **Offset: 268h**

Bit	Name	Initial	Slave	Host	Description
31: 8		0	RO	-	**Reserved**
7: 0		0P	RW	RW	**MailBox Data Register 1A bit[7:0]**

MBXDAT_1B: MailBox Data Register 1B **Offset: 26Ch**

Bit	Name	Initial	Slave	Host	Description
31: 8		0	RO	-	**Reserved**
7: 0		0P	RW	RW	**MailBox Data Register 1B bit[7:0]**

MBXDAT_1C: MailBox Data Register 1C **Offset: 270h**

Bit	Name	Initial	Slave	Host	Description
31:24		0P	RO	-	**MailBox Data Register 1F bit[7:0]**
23:16		0P	RO	-	**MailBox Data Register 1E bit[7:0]**
15: 8		0P	RO	-	**MailBox Data Register 1D bit[7:0]**
7: 0		0P	RW	RW	**MailBox Data Register 1C bit[7:0]**

MBXDAT_1D: MailBox Data Register 1D **Offset: 274h**

Bit	Name	Initial	Slave	Host	Description
31: 8		0	RO	-	**Reserved**
7: 0		0P	RW	RW	**MailBox Data Register 1D bit[7:0]**

MBXDAT_1E: MailBox Data Register 1E **Offset: 278h**

Bit	Name	Initial	Slave	Host	Description
31: 8		0	RO	-	**Reserved**
7: 0		0P	RW	RW	**MailBox Data Register 1E bit[7:0]**

MBXDAT_1F: MailBox Data Register 1F **Offset: 27Ch**

Bit	Name	Initial	Slave	Host	Description
31: 8		0	RO	-	**Reserved**
7: 0		0P	RW	RW	**MailBox Data Register 1F bit[7:0]**

MBXSTS_0: MailBox Status Register 0 **Offset: 280h**

Bit	Name	Initial	Slave	Host	Description
31:24		0	RO	-	**MailBox Status Register bit[31:24]**
23:16		0	RO	-	**MailBox Status Register bit[23:16]**
15: 8		0	RO	-	**MailBox Status Register bit[15: 8]**
7: 0		0P	RW1C	RW1C	**MailBox Status Register bit[7: 0]**

MBXSTS_1: MailBox Status Register 1 **Offset: 284h**

Bit	Name	Initial	Slave	Host	Description
31: 8		0	RO	-	**Reserved**
7: 0		0P	RW1C	RW1C	**MailBox Status Register bit[15: 8]**

MBXSTS_2: MailBox Status Register 2 **Offset: 288h**

Bit	Name	Initial	Slave	Host	Description
31: 8		0	RO	-	**Reserved**
7: 0		0P	RW1C	RW1C	**MailBox Status Register bit[23:16]**

MBXSTS_3: MailBox Status Register 3 **Offset: 28Ch**

Bit	Name	Initial	Slave	Host	Description
31: 8		0	RO	-	**Reserved**
7: 0		0P	RW1C	RW1C	**MailBox Status Register bit[31:24]**

MBXBCR: MailBox BMC Control Register **Offset: 290h**

Bit	Name	Initial	Slave	Host	Description
15		0P	RO	-	**Interrupt status from the BMC to the Host**
14:10		0	RO	-	**Reserved**
9		0P	RO	-	**Mask interrupt to the Host from the status bit, MBXHCR[7]**
8		0P	RO	-	**Generate interrupt to the BMC; set the status bit, MBXBCR[7]**
7		0P	RW1C	RO	**Interrupt status from the Host to the BMC**
6: 2		0	RO	RO	**Reserved**
1		0P	RW	RO	**Mask interrupt to the BMC from the status bit, MBXBCR[7]**
0		0P	W1T	RO	**Generate interrupt to the Host; set the status bit, MBXHCR[7]**

MBXHCR: MailBox Host Control Register **Offset: 294h**

Bit	Name	Initial	Slave	Host	Description
31: 8		0	RO	-	**Reserved**
7		0P	RO	RW1C	**Interrupt status from the BMC to the Host**
6: 2		0	RO		**Reserved**
1		0P	RO	RW	**Mask interrupt to the Host from the status bit, MBXHCR[7]**
0		0P	RO	RW	**Generate interrupt to the BMC; set the status bit, MBXBCR[7]**

MBXFCR_1: MailBox Flow Control Register 1 **Offset: 298h**

Bit	Name	Initial	Slave	Host	Description
31: 8		0	RO	-	**Reserved**
7: 0		0P	RW	RO	**Enable flow control register bit[7:0]. Host can't update data until BMC acknowledges.** bit[0]: SMBXDAT0 bit[1]: SMBXDAT1 ... bit[7]: SMBXDAT7

MBXFCR_2: MailBox Flow Control Register 2 **Offset: 29Ch**

Bit	Name	Initial	Slave	Host	Description
31: 8		0	RO	-	**Reserved**
7: 0		0P	W1T	RO	**BMC acknowleges flow control register bit[7:0].** bit[0]: SMBXDAT0 bit[1]: SMBXDAT1 ... bit[7]: SMBXDAT7

MBXBIE_0: MailBox BMC Interrupt Enable Register 0 **Offset: 2A0h**

Bit	Name	Initial	Slave	Host	Description
31:24		0P	RO	-	**MailBox BMC Interrupt Enable bit[31:24]**
23:16		0P	RO	-	**MailBox BMC Interrupt Enable bit[23:16]**
15: 8		0P	RO	-	**MailBox BMC Interrupt Enable bit[15:8]**
7: 0		0P	RW	RO	**MailBox BMC Interrupt Enable bit[7:0]**

MBXBIE_1: MailBox BMC Interrupt Enable Register 1 **Offset: 2A4h**

Bit	Name	Initial	Slave	Host	Description
31: 8		0	RO	-	**Reserved**
7: 0		0P	RW	RO	**MailBox BMC Interrupt Enable bit[15:8]**

MBXBIE_2: MailBox BMC Interrupt Enable Register 2					Offset: 2A8h
Bit	**Name**	**Initial**	**Slave**	**Host**	**Description**
31: 8		0	RO	-	**Reserved**
7: 0		0P	RW	RO	**MailBox BMC Interrupt Enable bit[23:16]**

MBXBIE_3: MailBox BMC Interrupt Enable Register 3					Offset: 2ACh
Bit	**Name**	**Initial**	**Slave**	**Host**	**Description**
31: 8		0	RO	-	**Reserved**
7: 0		0P	RW	RO	**MailBox BMC Interrupt Enable bit[31:24]**

MBXHIE_0: MailBox Host Interrupt Enable Register 0					Offset: 2B0h
Bit	**Name**	**Initial**	**Slave**	**Host**	**Description**
31:24		0P	RO	-	**MailBox Host Interrupt Enable bit[31:24]**
23:16		0P	RO	-	**MailBox Host Interrupt Enable bit[23:16]**
15: 8		0P	RO	-	**MailBox Host Interrupt Enable bit[15:8]**
7: 0		0P	RO	RW	**MailBox Host Interrupt Enable bit[7:0]**

MBXHIE_1: MailBox Host Interrupt Enable Register 1					Offset: 2B4h
Bit	**Name**	**Initial**	**Slave**	**Host**	**Description**
31: 8		0	RO	-	**Reserved**
7: 0		0P	RO	RW	**MailBox Host Interrupt Enable bit[15:8]**

MBXHIE_2: MailBox Host Interrupt Enable Register 2					Offset: 2B8h
Bit	**Name**	**Initial**	**Slave**	**Host**	**Description**
31: 8		0	RO	-	**Reserved**
7: 0		0P	RO	RW	**MailBox Host Interrupt Enable bit[23:16]**

MBXHIE_3: MailBox Host Interrupt Enable Register 3					Offset: 2BCh
Bit	**Name**	**Initial**	**Slave**	**Host**	**Description**
31: 8		0		-	**Reserved**
7: 0		0P	R	RW	**MailBox Host Interrupt Enable bit[31:24]**

49.4 Block Diagram

zS5#	zS3#	nxState	nxSIOONCTRL# (SCU510[5]=1, w/ SIOPWRGD, w/o BMC FW)
1	1	S12	Drive 0
1	0	S3I	Pull 1
0	X	S45	Pull 1

SCU510[5]	Enable ACPI pins, S3, S5, PWREQ, ONCTRL, PWRGD, PBO, PBI and SCI
SCU510[6]	Enable LPC mode 0:eSPI, 1:LPC (used to determine the source of S3/S5 signals)
SCU434[11]	Enable SIOONCTRL#function pin
SWCR_0B_08[12]	Enable GPIO mode on ONCTL
SWCR_0B_08[13]	Enable GPIO output on ONCTL
SWCR_0B_08[14]	GPIO level on ONCTL
SWCR_17_14[17]	Enable Host Power in S3I

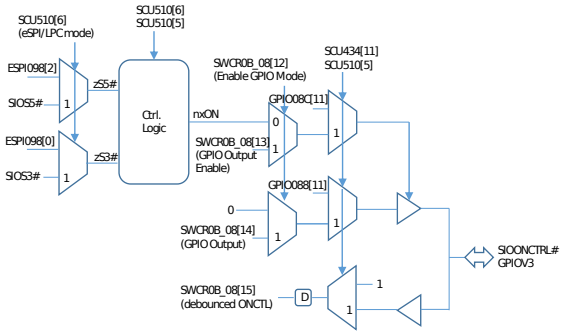


Figure 55: Block Diagram of SIOONCTRL#

SWCR_17_14[19]	Enable pulse on PWBTO as wake-up event
SWCR_17_14[20]	Lock PWRBTN
SWCR_07_04[15]	Force PWBTO output
SCU418[5]	Enable PBO#Pin
SCU510[5]/OTPSTRAP[37]	Enable ACPI pins, S3, S5, PWREQ, ONCTRL, PWRGD, PBO, PBI and SCI
SWCR_0B_08[24]	PWBTO output level

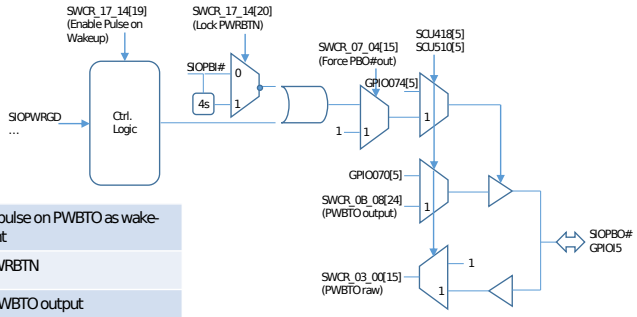


Figure 56: Block Diagram of SIOPBO#

SWCR_03_00[30]	PWRGD status after debounce
SWCR_03_00[29]	PWRGD rising event status
SWCR_03_00[28]	PWRGD falling event status
SWCR_07_04[29]	Enable BMC interrupt as rising event
SWCR_07_04[28]	Enable BMC interrupt as falling event
SCU434[12]	Enable SIOPWRGD function pin
SCU510[5]/OTPSTRAP[37]	Enable ACPI pins, S3, S5, PWREQ, ONCTRL, PWRGD, PBO, PBI and SCI

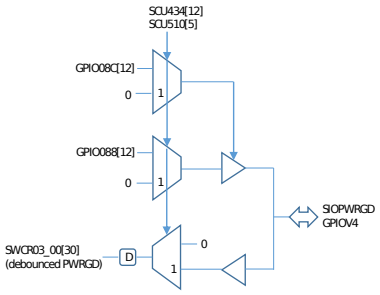


Figure 57: Block Diagram of SIOPWRGD

50 SuperIO Controller (SIO)

50.1 Overview

AST2600 integrates a Super I/O module with LPC protocol (I/O cycle 0x2E/0x2F or 0x4E/0x4F). There are 9 logical device corresponding to 9 individual functinos:

1. SUART1 (logical device 2)
2. SUART2 (logical device 3)
3. System Wake-Up Control (logical device 4)
4. GPIO (logical device 7)
5. SUART3 (logical device B)
6. SUART4 (logical device C)
7. iLPC2AHB (logical device D)
8. Mailbox (logical device E)
9. LSAFS (logical device F)

Each logical device has its own configuration register above index 0x30. The following software programming example is written in Intel assembly language for reference.

```
; password
MOV DX, 2EH
MOV AL, A5H
OUT DX, AL
OUT DX, AL
; select logical device 2
MOV DX, 2EH
MOV AL, 07H
OUT DX, AL
MOV DX, 2FH
MOV AL, 02H
OUT DX, AL
; set SIOR2_30 01H
MOV DX, 2EH
MOV AL, 30H
OUT DX, AL
MOV DX, 2FH
MOV AL, 01H
OUT DX, AL
; exit
MOV DX, 2EH
MOV AL, AAH
OUT DX, AL
```

The following registers can be access by host CPU through LPC bus.

SIORP : Super IO Password Register
SIORx_07: Super IO Register x_07
SIORx_20: Super IO Register x_20
SIORx_21: Super IO Register x_21
SIORx_22: Super IO Register x_22
SIORx_23: Super IO Register x_23

SIORx_24: Super IO Register x_24
SIORx_25: Super IO Register x_25
SIORx_26: Super IO Register x_26
SIORx_27: Super IO Register x_27
SIORx_28: Super IO Register x_28
SIORx_29: Super IO Register x_29
SIORx_2A: Super IO Register x_2A
SIORx_2B: Super IO Register x_2B
SIORx_2C: Super IO Register x_2C
SIORx_2D: Super IO Register x_2D
SIORx_2E: Super IO Register x_2E
SIORx_2F: Super IO Register x_2F

SIOR2_30: Super IO Register 2_30
SIOR2_60: Super IO Register 2_60
SIOR2_61: Super IO Register 2_61
SIOR2_70: Super IO Register 2_70
SIOR2_71: Super IO Register 2_71
SIOR2_F0: Super IO Register 2_F0

SIOR3_30: Super IO Register 3_30
SIOR3_60: Super IO Register 3_60
SIOR3_61: Super IO Register 3_61
SIOR3_70: Super IO Register 3_70
SIOR3_71: Super IO Register 3_71
SIOR3_F0: Super IO Register 3_F0

SIOR4_30: Super IO Register 4_30
SIOR4_60: Super IO Register 4_60
SIOR4_61: Super IO Register 4_61
SIOR4_62: Super IO Register 4_62
SIOR4_63: Super IO Register 4_63
SIOR4_64: Super IO Register 4_64
SIOR4_65: Super IO Register 4_65
SIOR4_66: Super IO Register 4_66
SIOR4_67: Super IO Register 4_67
SIOR4_70: Super IO Register 4_70
SIOR4_71: Super IO Register 4_71

SIOR7_30: Super IO Register 7_30
SIOR7_31: Super IO Register 7_31
SIOR7_38: Super IO Register 7_38
SIOR7_39: Super IO Register 7_39
SIOR7_70: Super IO Register 7_70
SIOR7_71: Super IO Register 7_71
SIOR7_72: Super IO Register 7_72
SIOR7_73: Super IO Register 7_73

SIORB_30: Super IO Register B_30
SIORB_60: Super IO Register B_60
SIORB_61: Super IO Register B_61
SIORB_70: Super IO Register B_70
SIORB_71: Super IO Register B_71
SIORB_F0: Super IO Register B_F0

SIORC_30: Super IO Register C_30
SIORC_60: Super IO Register C_60
SIORC_61: Super IO Register C_61
SIORC_70: Super IO Register C_70
SIORC_71: Super IO Register C_71
SIORC_F0: Super IO Register C_F0

SIORD_30: Super IO Register D_30
SIORD_70: Super IO Register D_70
SIORD_71: Super IO Register D_71
SIORD_F0: Super IO Register D_F0
SIORD_F1: Super IO Register D_F1
SIORD_F2: Super IO Register D_F2
SIORD_F3: Super IO Register D_F3
SIORD_F4: Super IO Register D_F4
SIORD_F5: Super IO Register D_F5
SIORD_F6: Super IO Register D_F6
SIORD_F7: Super IO Register D_F7
SIORD_F8: Super IO Register D_F8
SIORD_FE: Super IO Register D_FE

SIORE_30: Super IO Register E_30
SIORE_60: Super IO Register E_60
SIORE_61: Super IO Register E_61
SIORE_70: Super IO Register E_70
SIORE_71: Super IO Register E_71

SIORF_30: Super IO Register F_30
SIORF_F0: Super IO Register F_F0
SIORF_F1: Super IO Register F_F1
SIORF_F2: Super IO Register F_F2
SIORF_F3: Super IO Register F_F3
SIORF_F4: Super IO Register F_F4
SIORF_F5: Super IO Register F_F5
SIORF_F6: Super IO Register F_F6
SIORF_F7: Super IO Register F_F7
SIORF_F8: Super IO Register F_F8
SIORF_F9: Super IO Register F_F9
SIORF_FE: Super IO Register F_FE

Attribute Definition:

Attribute	: Description
R	: Readable
W	: Writable
W1C	: Write '1' to clear value to 0
X	: Unknown value
P	: Initialized by PWRST_N

Offset: 2E/4Eh	SIORP: Super IO Password Register	Init = 0
Bit	**R/W**	**Description**
7:0	WT	**Enable by writing 0xA5 twice; Disable by writing 0xAA once**

Offset: 07h	SIORx_07: Super IO Register x_07	Init = 0
Bit	**R/W**	**Description**
7:4	RO	**Reserved**
3:0	RW	**Logical device number**

Offset: 20h	SIORx_20: Super IO Register x_20	Init = 26P
Bit	**R/W**	**Description**
7:0	RO	**SIO identification bit[7:0]**

Offset: 21h	SIORx_21: Super IO Register x_21	Init = 0P
Bit	**R/W**	**Description**
3:0	RW	**SIO to BMC software interrupt bit[3:0] by rising trigger**
7:4	RW1C	**BMC to SIO software interrupt status bit[3:0] by writing one clear**

Offset: 22h	SIORx_22: Super IO Register x_22	Init = 0P
Bit	**R/W**	**Description**
7:0	RW	**SIO to BMC scratch register 2 bit[7:0]**

Offset: 23h	SIORx_23: Super IO Register x_23	Init = 0P
Bit	**R/W**	**Description**
7:0	RW	**SIO to BMC scratch register 3 bit[7:0]**

Offset: 24h	SIORx_24: Super IO Register x_24	Init = 0P
Bit	**R/W**	**Description**
7:0	RW	**SIO to BMC scratch register 4 bit[7:0]**

Offset: 25h	SIORx_25: Super IO Register x_25	Init = 0P
Bit	**R/W**	**Description**
7:0	RW	**SIO to BMC scratch register 5 bit[7:0]**

Offset: 26h	SIORx_26: Super IO Register x_26	Init = 0P
Bit	**R/W**	**Description**
7:0	RW	**SIO to BMC scratch register 6 bit[7:0]**

Offset: 27h	SIORx_27: Super IO Register x_27	Init = 0P
Bit	**R/W**	**Description**
7:0	RW	**SIO to BMC scratch register 7 bit[7:0]**

Offset: 28h	SIORx_28: Super IO Register x_28	Init = A8P
Bit	**R/W**	**Description**
7:0	RO	**BMC to SIO scratch register 0 bit[7:0]**

Offset: 29h	SIORx_29: Super IO Register x_29	Init = 0P
Bit	**R/W**	**Description**
7:0	RO	**BMC to SIO scratch register 1 bit[7:0]**

Offset: 2Ah	SIORx_2A: Super IO Register x_2A	Init = 0P
Bit	**R/W**	**Description**
7:0	RO	**BMC to SIO scratch register 2 bit[7:0]**

Offset: 2Bh	SIORx_2B: Super IO Register x_2B	Init = 0P
Bit	**R/W**	**Description**
7:0	RO	**BMC to SIO scratch register 3 bit[7:0]**

Offset: 2Ch	SIORx_2C: Super IO Register x_2C	Init = 0P
Bit	**R/W**	**Description**
7:0	RO	**BMC to SIO scratch register 4 bit[7:0]**

Offset: 2Dh	SIORx_2D: Super IO Register x_2D	Init = 0P
Bit	**R/W**	**Description**
7:0	RO	**BMC to SIO scratch register 5 bit[7:0]**

Offset: 2Eh	SIORx_2E: Super IO Register x_2E	Init = 0P
Bit	**R/W**	**Description**
7:0	RO	**BMC to SIO scratch register 6 bit[7:0]**

Offset: 2Fh	SIORx_2F: Super IO Register x_2F	Init = 0P
Bit	**R/W**	**Description**
7:0	RO	**BMC to SIO scratch register 7 bit[7:0]**

Offset: 30h	SIOR2_30: Super IO Register 2_30	Init = 0
Bit	**R/W**	**Description**
7:1	RW	**Reserved**
0	RW	**Enable SUART1**

Offset: 60h	SIOR2_60: Super IO Register 2_60	Init = 03
Bit	**R/W**	**Description**
7:0	RW	**SUART1 base address bit[15:8]**

Offset: 61h	SIOR2_61: Super IO Register 2_61	Init = F8
Bit	**R/W**	**Description**
7:3	RW	**SUART1 base address bit[7:3]**
2:0	RW	**Reserved**

Offset: 70h	SIOR2_70: Super IO Register 2_70	Init = 04h
Bit	**R/W**	**Description**
7:4	RO	**Reserved**
3:0	RW	**LPC: Select ID bit[3:0] of SerIRQ for SUART1**

Offset: 71h	SIOR2_71: Super IO Register 2_71	Init = 02
Bit	**R/W**	**Description**
7:2	RO	**Reserved**
1:0	RW	**Host SerIRQ interrupt type for SUART1** 00: BIOS setting in eSPI mode 01: low level trig 10: rising edge trig 11: high level trig

Offset: F0h	SIOR2_F0: Super IO Register 2_F0	Init = 0
Bit	**R/W**	**Description**
7:3	RO	**Reserved**
2	RW	**SUART1 clock source** 0: 1.846MHz (24MHz / 13) to generate baud-rates up to 115200 using DLL/DLH registers 1: 59.08MHz (24MHz * 32 / 13) to generate baud-rates greater than 115200 using DLL/DLH registers
1:0	RW	**Reserved**

Offset: 30h	SIOR3_30: Super IO Register 3_30	Init = 0
Bit	**R/W**	**Description**
7:1	RW	**Reserved**
0	RW	**Enable SUART2**

Offset: 60h	SIOR3_60: Super IO Register 3_60	Init = 02
Bit	**R/W**	**Description**
7:0	RW	**SUART2 base address bit[15:8]**

Offset: 61h	SIOR3_61: Super IO Register 3_61	Init = F8
Bit	**R/W**	**Description**
7:3	RW	**SUART2 base address bit[7:3]**
2:0	RW	**Reserved**

Offset: 70h	SIOR3_70: Super IO Register 3_70	Init = 03
Bit	**R/W**	**Description**
7:4	RO	**Reserved**
3:0	RW	**LPC: Select ID bit[3:0] of SerIRQ for SUART2**

Offset: 71h	SIOR3_71: Super IO Register 3_71	Init = 02
Bit	**R/W**	**Description**
7:2	RO	**Reserved**
1:0	RW	**Host SerIRQ interrupt type for SUART2** 00: BIOS setting in eSPI mode 01: low level trig 10: rising edge trig 11: high level trig

Offset: F0h	SIOR3_F0: Super IO Register 3_F0	Init = 0
Bit	**R/W**	**Description**
7:3	RO	**Reserved**
2	RW	**SUART2 clock source** 0: 1.846MHz (24MHz / 13) to generate baud-rates up to 115200 using DLL/DLH registers 1: 59.08MHz (24MHz * 32 / 13) to generate baud-rates greater than 115200 using DLL/DLH registers
1:0	RW	**Reserved**

Offset: 30h	SIOR4_30: Super IO Register 4_30	Init = 0
Bit	**R/W**	**Description**
7:1	RW	**Reserved**
0	RW	**Enable SWC**

Offset: 60h	SIOR4_60: Super IO Register 4_60	Init = 8P
Bit	**R/W**	**Description**
7:0	RW	**SWC base address bit[15:8]**

Offset: 61h	SIOR4_61: Super IO Register 4_61	Init = E6P
Bit	**R/W**	**Description**
7:1	RW	**SWC base address bit[7:1]**
0	RW	**Reserved**

Offset: 62h	SIOR4_62: Super IO Register 4_62	Init = 8P
Bit	**R/W**	**Description**
7:0	RW	**PM1b_EVT base address bit[15:8]**

Offset: 63h	SIOR4_63: Super IO Register 4_63	Init = E0P
Bit	**R/W**	**Description**
7:2	RW	**PM1b_EVT base address bit[7:2]**
1:0	RW	**Reserved**

Offset: 64h	SIOR4_64: Super IO Register 4_64	Init = 8P
Bit	**R/W**	**Description**
7:0	RW	**PM1b_CNT base address bit[15:8]**

Offset: 65h	SIOR4_65: Super IO Register 4_65	Init = E4P
Bit	**R/W**	**Description**
7:1	RW	**PM1b_CNT base address bit[7:1]**
0	RW	**Reserved**

Offset: 66h	SIOR4_66: Super IO Register 4_66	Init = 8P
Bit	**R/W**	**Description**
7:0	RW	**GPE1_BLK base address bit[15:8]**

Offset: 67h	SIOR4_67: Super IO Register 4_67	Init = E8P
Bit	**R/W**	**Description**
7:3	RW	**GPE1_BLK base address bit[7:3]**
2:0	RW	**Reserved**

Offset: 70h	SIOR4_70: Super IO Register 4_70	Init = 09P
Bit	**R/W**	**Description**
7:4	RO	**Reserved**
3:0	RW	**LPC: Select ID bit[3:0] of SerIRQ for SWC**

Offset: 71h	SIOR4_71: Super IO Register 4_71	Init = 01P
Bit	**R/W**	**Description**
7:2	RO	**Reserved**
1:0	RW	**Host SerIRQ interrupt type for SWC** 00: BIOS setting in eSPI mode 01: low level trig 10: rising edge trig 11: high level trig

Offset: 30h	SIOR7_30: Super IO Register 7_30	Init = 0
Bit	**R/W**	**Description**
7	RW	**Enable port 80h GPIO**
6	RW	**Enable port 80h SGPIO**
5	RW	**Reserved**
4	RW	**Enable USB SerIRQ**
3	RW	**Reserved**
2	RW	**Enable inverse polarity of SMI**
1	RW	**Enable SMI**
0	RW	**Enable SIO GPIO SerIRQ**

Offset: 31h	SIOR7_31: Super IO Register 7_31	Init = 0
Bit	**R/W**	**Description**
7	RW	**Enable port 81h GPIO**
6	RW	**Enable port 81h SGPIO**
5:0	RW	**Reserved**

Offset: 32h	SIOR7_32: Super IO Register 7_32	Init = 0
Bit	**R/W**	**Description**
7:2	RW	**Reserved**
1	RW1C	**Status of SIO GPIO interrupt**
0	RW1C	**Status of USB interrupt**

Offset: 38h	SIOR7_38: Super IO Register 7_38	Init = 0
Bit	**R/W**	**Description**
7	RW	**Enable inverse polarity of port 80h GPIO**
6:5	RW	**Reserved**
4:0	RW	**IO band Selection bit[4:0] for port 80h GPIO**

Offset: 39h	SIOR7_39: Super IO Register 7_39	Init = 0
Bit	**R/W**	**Description**
7	RW	**Enable inverse polarity of port 81h GPIO**
6:5	RW	**Reserved**
4:0	RW	**IO band Selection bit[4:0] for port 81h GPIO**

Offset: 70h	SIOR7_70: Super IO Register 7_70	Init = 0B
Bit	**R/W**	**Description**
7:4	RO	**Reserved**
3:0	RW	**LPC: Select ID bit[3:0] of SerIRQ for SIO GPIO**

Offset: 71h	SIOR7_71: Super IO Register 7_71	Init = 01
Bit	**R/W**	**Description**
7:2	RO	**Reserved**
1:0	RW	**Host SerIRQ interrupt type for SIO GPIO** 00: BIOS setting in eSPI mode 01: low level trig 10: rising edge trig 11: high level trig

Offset: 72h	SIOR7_72: Super IO Register 7_72	Init = 02
Bit	**R/W**	**Description**
7:4	RO	**Reserved**
3:0	RW	**LPC: Select ID bit[3:0] of SerIRQ for USB**

Offset: 73h	SIOR7_73: Super IO Register 7_73	Init = 01
Bit	**R/W**	**Description**
7:2	RO	**Reserved**
1:0	RW	**Host SerIRQ interrupt type for USB** 00: BIOS setting in eSPI mode 01: low level trig 10: rising edge trig 11: high level trig

Offset: 30h	SIORB_30: Super IO Register B_30	Init = 0
Bit	**R/W**	**Description**
7:1	RW	**Reserved**
0	RW	**Enable SUART3**

Offset: 60h	SIORB_60: Super IO Register B_60	Init = 03
Bit	**R/W**	**Description**
7:0	RW	**SUART3 base address bit[15:8]**

Offset: 61h	SIORB_61: Super IO Register B_61	Init = E8
Bit	**R/W**	**Description**
7:3	RW	**SUART3 base address bit[7:3]**
2:0	RW	**Reserved**

Offset: 70h	SIORB_70: Super IO Register B_70	Init = 06h
Bit	**R/W**	**Description**
7:4	RO	**Reserved**
3:0	RW	**Select ID bit[3:0] of SerIRQ for SUART3**

Offset: 71h	SIORB_71: Super IO Register B_71	Init = 02
Bit	**R/W**	**Description**
7:2	RO	**Reserved**
1:0	RW	**Host SerIRQ interrupt type for SUART3** 00: reserved 01: low level trig 10: rising edge trig 11: high level trig

Offset: F0h	SIORB_F0: Super IO Register B_F0	Init = 0
Bit	**R/W**	**Description**
7:3	RO	**Reserved**
2	RW	**SUART3 clock source** 0: 1.846MHz (24MHz / 13) to generate baud-rates up to 115200 using DLL/DLH registers 1: 59.08MHz (24MHz * 32 / 13) to generate baud-rates greater than 115200 using DLL/DLH registers
1:0	RW	**Reserved**

Offset: 30h	SIORC_30: Super IO Register C_30	Init = 0
Bit	**R/W**	**Description**
7:1	RW	**Reserved**
0	RW	**Enable SUART4**

Offset: 60h	SIORC_60: Super IO Register C_60	Init = 02
Bit	**R/W**	**Description**
7:0	RW	**SUART4 base address bit[15:8]**

Offset: 61h	SIORC_61: Super IO Register C_61	Init = E8
Bit	**R/W**	**Description**
7:3	RW	**SUART4 base address bit[7:3]**
2:0	RW	**Reserved**

Offset: 70h	SIORC_70: Super IO Register C_70	Init = 05
Bit	**R/W**	**Description**
7:4	RO	**Reserved**
3:0	RW	**Select ID bit[3:0] of SerIRQ for SUART4**

Offset: 71h	SIORC_71: Super IO Register C_71	Init = 02
Bit	**R/W**	**Description**
7:2	RO	**Reserved**
1:0	RW	**Host SerIRQ interrupt type for SUART4** 00: reserved 01: low level trig 10: rising edge trig 11: high level trig

Offset: F0h	SIORC_F0: Super IO Register C_F0	Init = 0
Bit	**R/W**	**Description**
7:3	RO	**Reserved**
2	RW	**SUART4 clock source** 0: 1.846MHz (24MHz / 13) to generate baud-rates up to 115200 using DLL/DLH registers 1: 59.08MHz (24MHz * 32 / 13) to generate baud-rates greater than 115200 using DLL/DLH registers
1:0	RW	**Reserved**

Offset: 30h	SIORD_30: Super IO Register D_30	Init = 0
Bit	**R/W**	**Description**
7:5	RW	**Reserved**
4	RW	**Enable SerIRQ of BMC to SIO scratch**
3	RW	**Select firmware driven or SVIC mode of SMI; default mode: SVIC**
2	RW	**SMI polarity; default: active low**
1	RW	**Enable SMI**
0	RW	**Enable SIO iLPC2AHB**

Offset: 70h	SIORD_70: Super IO Register D_70	Init = 09
Bit	**R/W**	**Description**
7:4	RO	**Reserved**
3:0	RW	**LPC: Select ID bit[3:0] of SerIRQ for BMC to SIO scratch**

Offset: 71h	SIORD_71: Super IO Register D_71	Init = 01
Bit	**R/W**	**Description**
7:2	RO	**Reserved**
1:0	RW	**Host SerIRQ interrupt type for BMC to SIO scratch** 00: BIOS setting in eSPI mode 01: low level trig 10: rising edge trig 11: high level trig

Offset: F0h	SIORD_F0: Super IO Register D_F0	Init = X
Bit	**R/W**	**Description**
7:0	RW	**SIO iLPC2AHB address bit[31:24]**

Offset: F1h	SIORD_F1: Super IO Register D_F1	Init = X
Bit	**R/W**	**Description**
7:0	RW	**SIO iLPC2AHB address bit[23:16]**

Offset: F2h	SIORD_F2: Super IO Register D_F2	Init = X
Bit	**R/W**	**Description**
7:0	RW	**SIO iLPC2AHB address bit[15:8]**

Offset: F3h	SIORD_F3: Super IO Register D_F3	Init = X
Bit	**R/W**	**Description**
7:0	RW	**SIO iLPC2AHB address bit[7:0]**

Offset: F4h	SIORD_F4: Super IO Register D_F4	Init = X
Bit	**R/W**	**Description**
7:0	RW	**SIO iLPC2AHB data bit[31:24]**

Offset: F5h	SIORD_F5: Super IO Register D_F5	Init = X
Bit	**R/W**	**Description**
7:0	RW	**SIO iLPC2AHB data bit[23:16]**

Offset: F6h	SIORD_F6: Super IO Register D_F6	Init = X
Bit	**R/W**	**Description**
7:0	RW	**SIO iLPC2AHB data bit[15:8]**

Offset: F7h	SIORD_F7: Super IO Register D_F7	Init = X
Bit	**R/W**	**Description**
7:0	RW	**SIO iLPC2AHB data bit[7:0]**

Offset: F8h	SIORD_F8: Super IO Register D_F8	Init = X
Bit	**R/W**	**Description**
7	RO	**SIO iLPC2AHB address alignment valid**
6	RO	**Disable SIO iLPC2AHB**
5:2	RW	**Reserved**
1:0	RW	**SIO iLPC2AHB length** 00: 1 byte 01: 2 bytes 10: 4 bytes 11: reserved

Offset: FEh	SIORD_FE: Super IO Register D_FE	Init = 0
Bit	**R/W**	**Description**
7:0	RW	**Read/Write 0xCF trigger SIO iLPC2AHB read/write command**

Offset: 30h	SIORE_30: Super IO Register E_30	Init = 0
Bit	**R/W**	**Description**
7:1	RW	**Reserved**
0	RW	**Enable Mailbox**

Offset: 60h	SIORE_60: Super IO Register E_60	Init = 0CP
Bit	**R/W**	**Description**
7:0	RW	**Mailbox base address bit[15:8]**

Offset: 61h	SIORE_61: Super IO Register E_61	Init = C0P
Bit	**R/W**	**Description**
7:6	RW	**Mailbox base address bit[7:6]**
5:0	RW	**Reserved**

Offset: 70h	SIORE_70: Super IO Register E_70	Init = 09P
Bit	**R/W**	**Description**
7:4	RO	**Reserved**
3:0	RW	**LPC: Select ID bit[3:0] of SerIRQ for Mailbox**

Offset: 71h	SIORE_71: Super IO Register E_71	Init = 01P
Bit	**R/W**	**Description**
7:2	RO	**Reserved**
1:0	RW	**Host SerIRQ interrupt type for Mailbox** 00: BIOS setting in eSPI mode 01: low level trig 10: rising edge trig 11: high level trig

Offset: 30h	SIORF_30: Super IO Register F_30	Init = 0
Bit	**R/W**	**Description**
7:1	RW	**Reserved**
0	RW	**Enable SIO SAFS**

Offset: F0h	SIORF_F0: Super IO Register F_F0	Init = X
Bit	**R/W**	**Description**
7:0	RW	**SIO SAFS address bit[31:24]**

Offset: F1h	SIORF_F1: Super IO Register F_F1	Init = X
Bit	**R/W**	**Description**
7:0	RW	**SIO SAFS address bit[23:16]**

Offset: F2h	SIORF_F2: Super IO Register F_F2	Init = X
Bit	**R/W**	**Description**
7:0	RW	**SIO SAFS address bit[15:8]**

Offset: F3h	SIORF_F3: Super IO Register F_F3	Init = X
Bit	**R/W**	**Description**
7:0	RW	**SIO SAFS address bit[7:0]**

Offset: F4h	SIORF_F4: Super IO Register F_F4	Init = X
Bit	**R/W**	**Description**
7:0	RW	**SIO SAFS data bit[31:24]**

Offset: F5h	SIORF_F5: Super IO Register F_F5	Init = X
Bit	**R/W**	**Description**
7:0	RW	**SIO SAFS data bit[23:16]**

Offset: F6h	SIORF_F6: Super IO Register F_F6	Init = X
Bit	**R/W**	**Description**
7:0	RW	**SIO SAFS data bit[15:8]**

Offset: F7h	SIORF_F7: Super IO Register F_F7	Init = X
Bit	**R/W**	**Description**
7:0	RW	**SIO SAFS data bit[7:0]**

Offset: F8h	SIORF_F8: Super IO Register F_F8	Init = X
Bit	**R/W**	**Description**
7:4	RW	**SIO SAFS command** 0: MemWr 1: MemRd 2: RegWr 3: RegRd 4: Erase4KB 5: Erase32KB 6: Erase64KB others: reserved
3:0	RW	**data byte enable bit[3:0] (active low)**

Offset: F9h	SIORF_F9: Super IO Register F_F9	Init = 1
Bit	**R/W**	**Description**
7:1	RW	**Reserved**
0	RW	**HSpiCfgLockN: SPI configuration lock control**

Offset: FEh	SIORF_FE: Super IO Register F_FE	Init = 0

Bit	R/W	Description
7:0	RW	**Read/Write 0xAF trigger SIO SAFS Mem/Reg/Erase command**

51 System Wake-Up Control (SWC)

51.1 Overview

The System Wake-Up Control (SWC) is a logical device (logical device 4) in Super I/O module. The registers use the Index/Data register as defined at indexes 60h/61h. This logical device includes ACPI registers defined at indexes 62h/63h (PM1b_EVT), 64h/65h (PM1b_CNT), and 66h/67h (GPE1_BLK). These registers are also accessible by BMC through memory mapped I/O.

SWCR_00: SWC Wake-Up Event Status
SWCR_01: SWC Miscellaneous Event Status 0
SWCR_02: SWC Miscellaneous Event Status 1
SWCR_03: SWC Miscellaneous Event Status 2
SWCR_04: SWC Wake-Up Enable 0
SWCR_05: SWC Wake-Up Enable 1
SWCR_06: SWC BMC Interrupt Enable 0
SWCR_07: SWC BMC Interrupt Enable 1
SWCR_08: SWC Control 0
SWCR_09: SWC Control 1
SWCR_0A: SWC Control 2
SWCR_0B: SWC Control 3
SWCR_0C: SWC Control 4
SWCR_0D: SWC Control 5
SWCR_0E: SWC Control 6
SWCR_0F: SWC Control 7
SWCR_10: SWC Sleep Encoding 0
SWCR_11: SWC Sleep Encoding 1
SWCR_12: SWC Sleep Encoding 2
SWCR_13: SWC Sleep Encoding 3
SWCR_14: SWC Sleep Encoding 4
SWCR_15: SWC Sleep Encoding 5
SWCR_16: SWC Sleep Control
SWCR_17: SWC Control 8
SWCR_18: SWC SMI Timer Count 0
SWCR_19: SWC SMI Timer Count 1
SWCR_1A: SWC SMI Timer Count 2
SWCR_1B: SWC SMI Timer Count 3
SWCR_1C: SWC SMI Timer Control
SWCR_1D: Reserved
SWCR_1E: Reserved
SWCR_1F: SWC Scratch

ACPI_E0: PM1b_STS_LOW
ACPI_E1: PM1b_STS_HIGH
ACPI_E2: PM1b_EN_LOW
ACPI_E3: PM1b_EN_HIGH

ACPI_C0: PM1b_CNT_LOW
ACPI_C1: PM1b_CNT_HIGH

ACPI_B0: GPE1_STS_0
ACPI_B1: GPE1_STS_1
ACPI_B2: GPE1_STS_2
ACPI_B3: GPE1_STS_3
ACPI_B4: GPE1_EN_0

ACPI_B5: GPE1_EN_1
ACPI_B6: GPE1_EN_2
ACPI_B7: GPE1_EN_3

Attribute Definition:

Attribute	: Description
R	: Readable
W	: Writable
W1C	: Write '1' to clear value to 0
X	: Unknown value
P	: Initialized by PWRST_N

Offset: 00h	**SWCR_00: SWC Wake-Up Event Status**	**Init = 0P**
Bit	**R/W**	**Description**
7:0	RW1C	**GPIO wake-up event status bit[7:0]**

Offset: 01h	**SWCR_01: SWC Miscellaneous Event Status 0**	**Init = 1010_0000P**
Bit	**R/W**	**Description**
7	RO	**PWBTO raw status**
6	RO	**Last_ONCTL status**
5	RW1C	**Was_pfail status**
4	RO	**Crowbar status**
3	RO	**PWRBTN Override status**
2	RO	**Reserved**
1	RW1C	**RI wake-up event status**
0	RW1C	**BMC trigger wake-up event status**

Offset: 02h	**SWCR_02: SWC Miscellaneous Event Status 1**	**Init = 0P**
Bit	**R/W**	**Description**
7	RO	**RI rising event status**
6	RO	**RI falling event status**
5	RO	**S5n status after debounce**
4	RO	**S5n rising event status**
3	RO	**S5n falling event status**
2	RO	**S3n status after debounce**
1	RO	**S3n rising event status**
0	RO	**S3n falling event status**

Offset: 03h	**SWCR_03: SWC Miscellaneous Event Status 2**	**Init = 0000_1001P**
Bit	**R/W**	**Description**
7	RO	**Setting sleep state event status**
6	RO	**PWRGD status after debounce**

to next page

from previous page

5	RO	**PWRGD rising event status**
4	RO	**PWRGD falling event status**
3	RO	**PWRBTN status after debounce**
2	RO	**PWRBTN rising event status**
1	RO	**PWRBTN falling event status**
0	RO	**RI status after debounce**

Offset: 04h **SWCR_04: SWC Wake-Up Enable 0** **Init = 0P**

Bit	R/W	Description
7:0	RW	**GPIO wake-up event enable bit[7:0]**

Offset: 05h **SWCR_05: SWC Wake-Up Enable 1** **Init = 0P**

Bit	R/W	Description
7	RW	**Force PWBTO output**
6	RW	**Enable PWREQ output**
5	RW	**Select RI event from UART2 or UART1**
4	RW	**Disable sleep entering as Crowbar**
3	RW	**Enable BMC interrupt as PWRBTN Override**
2	RO	**Reserved**
1	RW	**RI wake-up event enable**
0	RW	**BMC trigger wake-up event enable**

Offset: 06h **SWCR_06: SWC BMC Interrupt Enable 0** **Init = 0P**

Bit	R/W	Description
7	RO	**Enable BMC interrupt as RI rising event**
6	RO	**Enable BMC interrupt as RI falling event**
5	RO	**Reserved**
4	RO	**Enable BMC interrupt as S5n rising event**
3	RO	**Enable BMC interrupt as S5n falling event**
2	RO	**Reserved**
1	RO	**Enable BMC interrupt as S3n rising event**
0	RO	**Enable BMC interrupt as S3n falling event**

Offset: 07h **SWCR_07: SWC BMC Interrupt Enable 1** **Init = 0P**

Bit	R/W	Description
7	RO	**Enable BMC interrupt as setting sleep state event**
6	RO	**Reserved**
5	RO	**Enable BMC interrupt as PWRGD rising event**
4	RO	**Enable BMC interrupt as PWRGD falling event**
3	RO	**Reserved**
2	RO	**Enable BMC interrupt as PWRBTN rising event**

to next page

from previous page

1	RO	**Enable BMC interrupt as PWRBTN falling event**
0	RO	**Reserved**

Offset: 08h **SWCR_08: SWC Control 0** **Init = 0P**

Bit	R/W	Description
7	RW	**Enable software mode of S3n and S5n**
6	RO	**Reserved**
5	RW	**Force sleep state S45 entering**
4	RW	**Force sleep state S12 entering**
3	RW	**Enable SerIRQ interrupt as RI wake-up event**
2	RW	**Enable SMI interrupt as RI wake-up event**
1	RW	**Enable SerIRQ interrupt as BMC trigger wake-up event**
0	RW	**Enable SMI interrupt as BMC trigger wake-up event**

Offset: 09h **SWCR_09: SWC Control 1** **Init = 0P**

Bit	R/W	Description
7	RO	**ONCTL status after debounce**
6	RW	**GPIO level on ONCTL**
5	RW	**Enable GPIO output on ONCTL**
4	RW	**Enable GPIO mode on ONCTL**
3	RW	**Force internal vale of S5n**
2	RW	**Force internal vale of S3n**
1	RW	**Enable debounce of S3n and S5n**
0	RW	**Disable external wake-up from S3n**

Offset: 0Ah **SWCR_0A: SWC Control 2** **Init = 0P**

Bit	R/W	Description
7	RW	**Enable PWBTO from RI wake-up event in S45**
6	RW	**Enable PWBTO from RI wake-up event in S3I**
5	RW	**Enable ONCTL from RI wake-up event in S45**
4	RW	**Enable ONCTL from RI wake-up event in S3I**
3	RW	**Enable PWBTO from BMC trigger wake-up event in S45**
2	RW	**Enable PWBTO from BMC trigger wake-up event in S3I**
1	RW	**Enable ONCTL from BMC trigger wake-up event in S45**
0	RW	**Enable ONCTL from BMC trigger wake-up event in S3I**

Offset: 0Bh **SWCR_0B: SWC Control 3** **Init = 0P**

Bit	R/W	Description
7:4	RW	**time (unit 100ms) of filtering S5n, S3n and ONCTL as AC on**
3	RO	**Reserved**
2	RW	**Enable 4s PWRBTN Override**

to next page

from previous page

1	RW	**PWREQ output level**
0	RW	**PWBTO output level**

Offset: 0Ch	**SWCR_0C: SWC Control 4**	**Init = 0P**
Bit	**R/W**	**Description**
7:0	RW	**Enable ONCTL from GPIO wake-up event in S3I**

Offset: 0Dh	**SWCR_0D: SWC Control 5**	**Init = 0P**
Bit	**R/W**	**Description**
7:0	RW	**Enable ONCTL from GPIO wake-up event in S45**

Offset: 0Eh	**SWCR_0E: SWC Control 6**	**Init = 0P**
Bit	**R/W**	**Description**
7:0	RW	**Enable PWBTO from GPIO wake-up event in S3I**

Offset: 0Fh	**SWCR_0F: SWC Control 7**	**Init = 0P**
Bit	**R/W**	**Description**
7:0	RW	**Enable PWBTO from GPIO wake-up event in S45**

Offset: 10h	**SWCR_10: SWC Sleep Encoding 0**	**Init = 0P**
Bit	**R/W**	**Description**
7:3	RO	**Reserved**
2:0	RW	**Sleep type encoding of S0**

Offset: 11h	**SWCR_11: SWC Sleep Encoding 1**	**Init = 1P**
Bit	**R/W**	**Description**
7:3	RO	**Reserved**
2:0	RW	**Sleep type encoding of S1**

Offset: 12h	**SWCR_12: SWC Sleep Encoding 2**	**Init = 2P**
Bit	**R/W**	**Description**
7:3	RO	**Reserved**
2:0	RW	**Sleep type encoding of S2**

Offset: 13h	**SWCR_13: SWC Sleep Encoding 3**	**Init = 5P**
Bit	**R/W**	**Description**
7:3	RO	**Reserved**
2:0	RW	**Sleep type encoding of S3**

Offset: 14h	SWCR_14: SWC Sleep Encoding 4	Init = 6P
Bit	**R/W**	**Description**
7:3	RO	**Reserved**
2:0	RW	**Sleep type encoding of S4**

Offset: 15h	SWCR_15: SWC Sleep Encoding 5	Init = 7P
Bit	**R/W**	**Description**
7:3	RO	**Reserved**
2:0	RW	**Sleep type encoding of S5**

Offset: 16h	SWCR_16: SWC Sleep Control	Init = 0P
Bit	**R/W**	**Description**
7:6	RO	**slp_now_st[1:0]: sleep state status**
5	RO	**Reserved**
4	RW	**Lock PWRBTN** 0: PWRBTN is directly output on PWBTO 1: PWBTO is pulsed only if PWRBTN is held low for more than 4 seconds
3	RW	**Enable pulse on PWBTO as wake-up event**
2	RW	**Enable PWRBTN press wake-up**
1	RW	**Enable Host power in S3I**
0	RW	**Select (S3I = S3 or S4) mode or (S45 = S4 or S5) mode**

Offset: 17h	SWCR_17: SWC Control 8	Init = 7P
Bit	**R/W**	**Description**
7:5	RW	**Group selection of GPIO wake-up event**
4	RW	**Enable BMC interrupt as Crowbar**
3:0	RW	**Power up timeout second bit[3:0]**

Offset: 18h	SWCR_18: SWC SMI Timer Count 0	Init = 0P
Bit	**R/W**	**Description**
7:0	RW	**SMI second count bit[7:0] (n-1 n: unit 125 ms)**

Offset: 19h	SWCR_19: SWC SMI Timer Count 1	Init = 0P
Bit	**R/W**	**Description**
7:0	RW	**SMI minute count bit[7:0] (n-1 n: unit 1 minute)**

Offset: 1Ah	SWCR_1A: SWC SMI Timer Count 2	Init = 0P
Bit	**R/W**	**Description**
7:0	RO	**SMI current second count bit[7:0] (unit 125 ms)**

Offset: 1Bh	SWCR_1B: SWC SMI Timer Count 3	Init = 0P
Bit	**R/W**	**Description**
7:0	RO	**SMI current minute count bit[7:0] (unit 1 minute)**

Offset: 1Ch	SWCR_1C: SWC SMI Timer Control	Init = 0P
Bit	**R/W**	**Description**
7:6	RO	**Reserved**
5	RW1C	**SMI minute interrupt status**
4	RW1C	**SMI second interrupt status**
3	RW	**Enable SMI interrupt of minute timer**
2	RW	**Enable SMI interrupt of second timer**
1	RW	**Enable SMI minute timer**
0	RW	**Enable SMI second timer**

Offset: 1Dh	SWCR_1D: SWC Control 9	Init = 0P
Bit	**R/W**	**Description**
7:0	RO	**Reserved**

Offset: 1Eh	SWCR_1E: SWC Control A	Init = 0P
Bit	**R/W**	**Description**
7:0	RO	**Reserved**

Offset: 1Fh	SWCR_1F: SWC Scratch	Init = 0P
Bit	**R/W**	**Description**
7:0	RW	**SWC scratch bit[7:0]**

Offset: 00h	ACPI_E0: PM1b_STS_LOW	Init = 0P
Bit	**R/W**	**Description**
7:0	RO	**Reserved**

Offset: 01h	ACPI_E1: PM1b_STS_HIGH	Init = 0P
Bit	**R/W**	**Description**
7	RW1C	**Wake-up event status**
6:1	RO	**Reserved**
0	RW1C	**PWRBTN event status**

Offset: 02h	ACPI_E2: PM1b_EN_LOW	Init = 0P
Bit	**R/W**	**Description**
7:0	RO	**Reserved**

Offset: 03h	ACPI_E3: PM1b_EN_HIGH	Init = 0P
Bit	**R/W**	**Description**
7	RW	**Enable SCI from wake-up event**
6:1	RO	**Reserved**
0	RW	**Enable SCI from PWRBTN event**

Offset: 00h	ACPI_C0: PM1b_CNT_LOW	Init = 0P
Bit	**R/W**	**Description**
7:0	RO	**Reserved**

Offset: 01h	ACPI_C1: PM1b_CNT_HIGH	Init = 0P
Bit	**R/W**	**Description**
7:6	RO	**Reserved**
5	RW	**Update Sleep Type value**
4:2	RW	**Sleep Type bit[2:0]**
1:0	RO	**Reserved**

Offset: 00h	ACPI_B0: GPE1_STS_0	Init = 0P
Bit	**R/W**	**Description**
7:0	RW1C	**GPIO SCI event status bit[7:0]**

Offset: 01h	ACPI_B1: GPE1_STS_1	Init = 0P
Bit	**R/W**	**Description**
7:2	RO	**Reserved**
1	RW1C	**RI SCI event status**
0	RW1C	**BMC trigger SCI event status**

Offset: 02h	ACPI_B2: GPE1_STS_2	Init = 0P
Bit	**R/W**	**Description**
7:0	RO	**Reserved**

Offset: 03h	ACPI_B3: GPE1_STS_3	Init = 0P
Bit	**R/W**	**Description**
7:0	RO	**Reserved**

Offset: 04h	ACPI_B4: GPE1_EN_0	Init = 0P
Bit	**R/W**	**Description**
7:0	RW	**GPIO SCI event enable bit[7:0]**

Offset: 05h	ACPI_B5: GPE1_EN_1	Init = 0P
Bit	**R/W**	**Description**
7:2	RO	**Reserved**
1	RW	**RI SCI event enable**
0	RW	**BMC trigger SCI event enable**

Offset: 06h	ACPI_B6: GPE1_EN_2	Init = 0P
Bit	**R/W**	**Description**
7:0	RO	**Reserved**

Offset: 07h	ACPI_B7: GPE1_EN_3	Init = 0P
Bit	**R/W**	**Description**
7:0	RO	**Reserved**

52 MailBox Controller (MBOX)

52.1 Overview

There are 32 general purpose mailbox registers implemented in each node. They can be used to communicate or send messages between the Host and the BMC. Also, the interrupts of signaling the opposite processor are introduced to ease the loading.

The following registers can be accessed by the Host and the BMC. Here shows the descriptions for the Host.

The host side MailBox control and base address registers are defined in below SuperIO registers:

SIORE_30
SIORE_60
SIORE_61
SIORE_70
SIORE_71
SIORE_F0

SMBXDAT_0: MailBox Data Register 0
SMBXDAT_1: MailBox Data Register 1
SMBXDAT_2: MailBox Data Register 2
SMBXDAT_3: MailBox Data Register 3
SMBXDAT_4: MailBox Data Register 4
SMBXDAT_5: MailBox Data Register 5
SMBXDAT_6: MailBox Data Register 6
SMBXDAT_7: MailBox Data Register 7
SMBXDAT_8: MailBox Data Register 8
SMBXDAT_9: MailBox Data Register 9
SMBXDAT_A: MailBox Data Register A
SMBXDAT_B: MailBox Data Register B
SMBXDAT_C: MailBox Data Register C
SMBXDAT_D: MailBox Data Register D
SMBXDAT_E: MailBox Data Register E
SMBXDAT_F: MailBox Data Register F

SMBXDAT_10: MailBox Data Register 10
SMBXDAT_11: MailBox Data Register 11
SMBXDAT_12: MailBox Data Register 12
SMBXDAT_13: MailBox Data Register 13
SMBXDAT_14: MailBox Data Register 14
SMBXDAT_15: MailBox Data Register 15
SMBXDAT_16: MailBox Data Register 16
SMBXDAT_17: MailBox Data Register 17
SMBXDAT_18: MailBox Data Register 18
SMBXDAT_19: MailBox Data Register 19
SMBXDAT_1A: MailBox Data Register 1A
SMBXDAT_1B: MailBox Data Register 1B
SMBXDAT_1C: MailBox Data Register 1C
SMBXDAT_1D: MailBox Data Register 1D
SMBXDAT_1E: MailBox Data Register 1E
SMBXDAT_1F: MailBox Data Register 1F

SMBXSTS_0: MailBox Status Register 0
SMBXSTS_1: MailBox Status Register 1
SMBXSTS_2: MailBox Status Register 2
SMBXSTS_3: MailBox Status Register 3
SMBXBCR : MailBox BMC Control Register
SMBXHCR : MailBox Host Control Register
SMBXBIE_0: MailBox BMC Interrupt Enable Register 0
SMBXBIE_1: MailBox BMC Interrupt Enable Register 1
SMBXBIE_2: MailBox BMC Interrupt Enable Register 2
SMBXBIE_3: MailBox BMC Interrupt Enable Register 3
SMBXHIE_0: MailBox Host Interrupt Enable Register 0
SMBXHIE_1: MailBox Host Interrupt Enable Register 1
SMBXHIE_2: MailBox Host Interrupt Enable Register 2
SMBXHIE_3: MailBox Host Interrupt Enable Register 3

Attribute Definition:

Attribute	: Description
R	: Readable
W	: Writable
W1C	: Write '1' to clear value to 0
X	: Unknown value
P	: Initialized by PWRST_N

SMBXDAT_0: MailBox Data Register 0 **Offset: 00**

Bit	Name	Initial	Slave	Host	Description
7:0		0P	RW	RW	**MailBox Data Register 0 bit[7:0]**

SMBXDAT_1: MailBox Data Register 1 **Offset: 01**

Bit	Name	Initial	Slave	Host	Description
7:0		0P	RW	RW	**MailBox Data Register 1 bit[7:0]**

SMBXDAT_2: MailBox Data Register 2 **Offset: 02**

Bit	Name	Initial	Slave	Host	Description
7:0		0P	RW	RW	**MailBox Data Register 2 bit[7:0]**

SMBXDAT_3: MailBox Data Register 3 **Offset: 03**

Bit	Name	Initial	Slave	Host	Description
7:0		0P	RW	RW	**MailBox Data Register 3 bit[7:0]**

SMBXDAT_4: MailBox Data Register 4 **Offset: 04**

Bit	Name	Initial	Slave	Host	Description
7:0		0P	RW	RW	**MailBox Data Register 4 bit[7:0]**

SMBXDAT_5: MailBox Data Register 5 **Offset: 05**

Bit	Name	Initial	Slave	Host	Description
7:0		0P	RW	RW	**MailBox Data Register 5 bit[7:0]**

SMBXDAT_6: MailBox Data Register 6 **Offset: 06**

Bit	Name	Initial	Slave	Host	Description
7:0		0P	RW	RW	**MailBox Data Register 6 bit[7:0]**

SMBXDAT_7: MailBox Data Register 7 **Offset: 07**

Bit	Name	Initial	Slave	Host	Description
7:0		0P	RW	RW	**MailBox Data Register 7 bit[7:0]**

SMBXDAT_8: MailBox Data Register 8 **Offset: 08**

Bit	Name	Initial	Slave	Host	Description
7:0		0P	RW	RW	**MailBox Data Register 8 bit[7:0]**

SMBXDAT_9: MailBox Data Register 9 **Offset: 09**

Bit	Name	Initial	Slave	Host	Description
7:0		0P	RW	RW	**MailBox Data Register 9 bit[7:0]**

SMBXDAT_A: MailBox Data Register A **Offset: 0A**

Bit	Name	Initial	Slave	Host	Description
7:0		0P	RW	RW	**MailBox Data Register A bit[7:0]**

SMBXDAT_B: MailBox Data Register B **Offset: 0B**

Bit	Name	Initial	Slave	Host	Description
7:0		0P	RW	RW	**MailBox Data Register B bit[7:0]**

SMBXDAT_C: MailBox Data Register C **Offset: 0C**

Bit	Name	Initial	Slave	Host	Description
7:0		0P	RW	RW	**MailBox Data Register C bit[7:0]**

SMBXDAT_D: MailBox Data Register D **Offset: 0D**

Bit	Name	Initial	Slave	Host	Description
7:0		0P	RW	RW	**MailBox Data Register D bit[7:0]**

SMBXDAT_E: MailBox Data Register E **Offset: 0E**

Bit	Name	Initial	Slave	Host	Description
7:0		0P	RW	RW	**MailBox Data Register E bit[7:0]**

SMBXDAT_F: MailBox Data Register F					Offset: 0F
Bit	**Name**	**Initial**	**Slave**	**Host**	**Description**
7:0		0P	RW	RW	**MailBox Data Register F bit[7:0]**

SMBXDAT_10: MailBox Data Register 10					Offset: 10
Bit	**Name**	**Initial**	**Slave**	**Host**	**Description**
7:0		0P	RW	RW	**MailBox Data Register 10 bit[7:0]**

SMBXDAT_11: MailBox Data Register 11					Offset: 11
Bit	**Name**	**Initial**	**Slave**	**Host**	**Description**
7:0		0P	RW	RW	**MailBox Data Register 11 bit[7:0]**

SMBXDAT_12: MailBox Data Register 12					Offset: 12
Bit	**Name**	**Initial**	**Slave**	**Host**	**Description**
7:0		0P	RW	RW	**MailBox Data Register 12 bit[7:0]**

SMBXDAT_13: MailBox Data Register 13					Offset: 13
Bit	**Name**	**Initial**	**Slave**	**Host**	**Description**
7:0		0P	RW	RW	**MailBox Data Register 13 bit[7:0]**

SMBXDAT_14: MailBox Data Register 14					Offset: 14
Bit	**Name**	**Initial**	**Slave**	**Host**	**Description**
7:0		0P	RW	RW	**MailBox Data Register 14 bit[7:0]**

SMBXDAT_15: MailBox Data Register 15					Offset: 15
Bit	**Name**	**Initial**	**Slave**	**Host**	**Description**
7:0		0P	RW	RW	**MailBox Data Register 15 bit[7:0]**

SMBXDAT_16: MailBox Data Register 16					Offset: 16
Bit	**Name**	**Initial**	**Slave**	**Host**	**Description**
7:0		0P	RW	RW	**MailBox Data Register 16 bit[7:0]**

SMBXDAT_17: MailBox Data Register 17					Offset: 17
Bit	**Name**	**Initial**	**Slave**	**Host**	**Description**
7:0		0P	RW	RW	**MailBox Data Register 17 bit[7:0]**

SMBXDAT_18: MailBox Data Register 18					Offset: 18
Bit	**Name**	**Initial**	**Slave**	**Host**	**Description**
7:0		0P	RW	RW	**MailBox Data Register 18 bit[7:0]**

SMBXDAT_19: MailBox Data Register 19					Offset: 19
Bit	**Name**	**Initial**	**Slave**	**Host**	**Description**
7:0		0P	RW	RW	**MailBox Data Register 19 bit[7:0]**

SMBXDAT_1A: MailBox Data Register 1A					Offset: 1A
Bit	**Name**	**Initial**	**Slave**	**Host**	**Description**
7:0		0P	RW	RW	**MailBox Data Register 1A bit[7:0]**

SMBXDAT_1B: MailBox Data Register 1B					Offset: 1B
Bit	**Name**	**Initial**	**Slave**	**Host**	**Description**
7:0		0P	RW	RW	**MailBox Data Register 1B bit[7:0]**

SMBXDAT_1C: MailBox Data Register 1C					Offset: 1C
Bit	**Name**	**Initial**	**Slave**	**Host**	**Description**
7:0		0P	RW	RW	**MailBox Data Register 1C bit[7:0]**

SMBXDAT_1D: MailBox Data Register 1D					Offset: 1D
Bit	**Name**	**Initial**	**Slave**	**Host**	**Description**
7:0		0P	RW	RW	**MailBox Data Register 1D bit[7:0]**

SMBXDAT_1E: MailBox Data Register 1E					Offset: 1E
Bit	**Name**	**Initial**	**Slave**	**Host**	**Description**
7:0		0P	RW	RW	**MailBox Data Register 1E bit[7:0]**

SMBXDAT_1F: MailBox Data Register 1F					Offset: 1F
Bit	**Name**	**Initial**	**Slave**	**Host**	**Description**
7:0		0P	RW	RW	**MailBox Data Register 1F bit[7:0]**

SMBXSTS_0: MailBox Status Register 0					Offset: 20
Bit	**Name**	**Initial**	**Slave**	**Host**	**Description**
7:0		0P	RW1C	RW1C	**MailBox Status Register 0 bit[7:0]**

SMBXSTS_1: MailBox Status Register 1					Offset: 21
Bit	**Name**	**Initial**	**Slave**	**Host**	**Description**
7:0		0P	RW1C	RW1C	**MailBox Status Register 1 bit[15:8]**

SMBXSTS_2: MailBox Status Register 2					Offset: 22
Bit	**Name**	**Initial**	**Slave**	**Host**	**Description**
7:0		0P	RW1C	RW1C	**MailBox Status Register 2 bit[23:16]**

SMBXSTS_3: MailBox Status Register 3 **Offset: 23**

Bit	Name	Initial	Slave	Host	Description
7:0		0P	RW1C	RW1C	**MailBox Status Register 3 bit[31:24]**

SMBXBCR: MailBox BMC Control Register **Offset: 24**

Bit	Name	Initial	Slave	Host	Description
7		0P	RW1C	RO	**Interrupt status from the Host to the BMC**
6:2		0P	RO	RO	**Reserved**
1		0P	RW	RO	**Mask interrupt to the BMC from the status bit, MBXBCR[7]**
0		0P	RW	RO	**Generate interrupt to the Host; set the status bit, MBXHCR[7]**

SMBXHCR: MailBox Host Control Register **Offset: 25**

Bit	Name	Initial	Slave	Host	Description
7		0P	RO	RW1C	**Interrupt status from the BMC to the Host**
6:2		0P	RO	RO	**Reserved**
1		0P	RO	RW	**Mask interrupt to the Host from the status bit, MBXHCR[7]**
0		0P	RO	W1T	**Generate interrupt to the BMC; set the status bit, MBXBCR[7]**

SMBXFCR_1: MailBox Flow Control Register 1 **Offset: 26**

Bit	Name	Initial	Slave	Host	Description
7:0		0P	RW	RO	**Enable flow control register bit[7:0]. Host can't update data until BMC acknowledges.** bit[0]: SMBXDAT0 bit[1]: SMBXDAT1 ... bit[7]: SMBXDAT7

SMBXFCR_2: MailBox Flow Control Register 2 **Offset: 27**

Bit	Name	Initial	Slave	Host	Description
7:0		0P	W1T	RO	**BMC acknowleges flow control register bit[7:0].** bit[0]: SMBXDAT0 bit[1]: SMBXDAT1 ... bit[7]: SMBXDAT7

SMBXBIE_0: MailBox BMC Interrupt Enable Register 0 **Offset: 28**

Bit	Name	Initial	Slave	Host	Description
7:0		0P	RW	RO	**MailBox BMC Interrupt Enable bit[7:0]**

SMBXBIE_1: MailBox BMC Interrupt Enable Register 1					Offset: 29
Bit	**Name**	**Initial**	**Slave**	**Host**	**Description**
7:0		0P	RW	RO	**MailBox BMC Interrupt Enable bit[15:8]**

SMBXBIE_2: MailBox BMC Interrupt Enable Register 2					Offset: 2A
Bit	**Name**	**Initial**	**Slave**	**Host**	**Description**
7:0		0P	RW	RO	**MailBox BMC Interrupt Enable bit[23:16]**

SMBXBIE_3: MailBox BMC Interrupt Enable Register 3					Offset: 2B
Bit	**Name**	**Initial**	**Slave**	**Host**	**Description**
7:0		0P	RW	RO	**MailBox BMC Interrupt Enable bit[31:24]**

SMBXHIE_0: MailBox Host Interrupt Enable Register 0					Offset: 2C
Bit	**Name**	**Initial**	**Slave**	**Host**	**Description**
7:0		0P	RO	RW	**MailBox Host Interrupt Enable bit[7:0]**

SMBXHIE_1: MailBox Host Interrupt Enable Register 1					Offset: 2D
Bit	**Name**	**Initial**	**Slave**	**Host**	**Description**
7:0		0P	RO	RW	**MailBox Host Interrupt Enable bit[15:8]**

SMBXHIE_2: MailBox Host Interrupt Enable Register 2					Offset: 2E
Bit	**Name**	**Initial**	**Slave**	**Host**	**Description**
7:0		0P	RO	RW	**MailBox Host Interrupt Enable bit[23:16]**

SMBXHIE_3: MailBox Host Interrupt Enable Register 3					Offset: 2F
Bit	**Name**	**Initial**	**Slave**	**Host**	**Description**
7:0		0P	RO	RW	**MailBox Host Interrupt Enable bit[31:24]**

53 I2C/SMBus Controller (I2C)

53.1 Overview

I2C/SMBus Controller implements one set of global registers and **16** sets of device registers to program the various functions supported by AST2600 . Each register has its own specific offset value to derive its physical address location.

Extra 4 Secure I2C controllers (I2CS) are built in A2 revision. They can be assigned to TrustZone or SSP for security usage. The Secure I2C controller can't support DMA function.

Base address of Global Register = 0x1E78_A000
Physical register address = (Base address) + Offset
Base address of Secure I2C (I2CS) Global Register = 0x1E7A_8000
Physical register address of Secure I2C (I2CS) = (Base address I2CS) + Offset

I2CG00: Device Master Mode Interrupt Status Register (I2CG0C[3]=1)
I2CG00: Device Master/Slave Mode Interrupt Status Register (I2CG0C[3]=0)
I2CG04: Device Slave Mode Interrupt Status Register
I2CG0C: Global Control Register
I2CG10: New Clock Divider Control Register (I2CG0C[1] = 1)

53.1.1 Old Register mode : I2CG0C[2] = 0

I2CD00: Function Control Register
I2CD04: Clock and AC Timing Control Register #1
I2CD08: Clock and AC Timing Control Register #2
I2CD0C: Interrupt Control Register
I2CD10: Interrupt Status Register
I2CD14: Command/Status Register
I2CD18: Slave Device Address Register
I2CD1C: Pool Buffer Control Register
I2CD20: Transmit/Receive Byte Buffer Register
I2CD24: DMA Mode Buffer Address Register
I2CD28: DMA Transfer Length Register
I2CD2C: Original DMA Mode Buffer Address Setting
I2CD30: Original DMA Transfer Length Setting and Final Status

53.1.2 New Register mode : I2CG0C[2] = 1

I2CC00: Master/Slave Function Control Register
I2CC04: Master/Slave Clock and AC Timing Control Register
I2CC08: Master/Slave Transmit/Receive Byte Buffer Register
I2CC0C: Master/Slave Pool Buffer Control Register
I2CM10: Master Interrupt Control Register
I2CM14: Master Interrupt Status Register
I2CM18: Master Command/Status Register
I2CM1C: Master DMA Buffer Length Register
I2CS20: Slave Interrupt Control Register
I2CS24: Slave Interrupt Status Register
I2CS28: Slave Command/Status Register
I2CS2C: Slave DMA Buffer Length Register

I2CM30: Master DMA Mode Tx Buffer Base Address
I2CM34: Master DMA Mode Rx Buffer Base Address
I2CS38: Slave DMA Mode Tx Buffer Base Address
I2CS3C: Slave DMA Mode Rx Buffer Base Address
I2CS40: Slave Device Address Register
I2CM48: Master DMA Length Status Register
I2CS4C: Slave DMA Length Status Register
I2CC50: Current DMA Operating Address Status
I2CC54: Current DMA Operating Length Status

I2CSG00: Device Master Mode Interrupt Status Register (I2CSG0C[3]=1)
I2CSG00: Device Master/Slave Mode Interrupt Status Register (I2CG0C[3]=0)
I2CSG04: Device Slave Mode Interrupt Status Register
I2CSG0C: Global Control Register
I2CSG10: New Clock Divider Control Register (I2CSG0C[1] = 1)

53.1.3 Old Register mode : I2CSG0C[2] = 0

I2CSD00: Function Control Register
I2CSD04: Clock and AC Timing Control Register #1
I2CSD08: Clock and AC Timing Control Register #2
I2CSD0C: Interrupt Control Register
I2CSD10: Interrupt Status Register
I2CSD14: Command/Status Register
I2CSD18: Slave Device Address Register
I2CSD1C: Pool Buffer Control Register
I2CSD20: Transmit/Receive Byte Buffer Register

53.1.4 New Register mode : I2CSG0C[2] = 1

I2CSC00: Master/Slave Function Control Register
I2CSC04: Master/Slave Clock and AC Timing Control Register
I2CSC08: Master/Slave Transmit/Receive Byte Buffer Register
I2CSC0C: Master/Slave Pool Buffer Control Register
I2CSM10: Master Interrupt Control Register
I2CSM14: Master Interrupt Status Register
I2CSM18: Master Command/Status Register
I2CSS20: Slave Interrupt Control Register
I2CSS24: Slave Interrupt Status Register
I2CSS28: Slave Command/Status Register
I2CSS40: Slave Device Address Register

53.2 Features

53.2.1 I2C Master - all 16 buses

- Compatible with Philips I2C-BUS Specification Version 2.1
- Multi Master Operation Supported
- Software programmable clock frequency
- Software programmable AC timing

- Support a wide range of transmission speed, 0.1Kbps - 5Mbps
- Clock Stretching and Wait state generation/detection
- Interrupt or bit-polling driven byte-by-byte data-transfers
- Arbitration lost interrupt, with automatic transfer cancellation
- Start/Stop/Repeated Start/Acknowledge generation
- Bus busy detection
- Automatic ACK/NACK Generation for Receive and Detection for Transmit
- Programmable Size ($\leq$ 32 Byte) of Pool Buffer Mode for improving performance
- Programmable Size ($\leq$ 4096 Byte) of DMA buffer for large amount of data transfer
- Support bus lock recovery function

53.2.2 I2C Slave - all 16 buses

- Start/Stop/Repeated Start detection
- Supports 7 bits addressing mode only
- ~~Controllable support for General Call Address~~
- Operates from a wide range of input clock frequencies as Master mode
- Clock Stretching and Wait state generation
- Automatic ACK/NACK response
- Support 3 programmable slave addresses, and range mode slave address.

53.2.3 SMBus - all 16 buses

- Compatible with SBS SMBus Specification Version 3.0
- Involved all features of I2C
- Support transmission speed from 0.1Kbps - 5Mbps if PCLK = 50MHz
- ~~Controllable support for ARP Default Host Address 0001 000~~
- ~~Controllable support for ARP Default Device Address 1100 001~~
- ~~Controllable support for Alert Response Address 0001 100~~
- Support dedicated alert pin for each set of SMBus/I2C controller
  - Support Master Alert interrupt
  - Support Slave Alert function

53.2.4 General

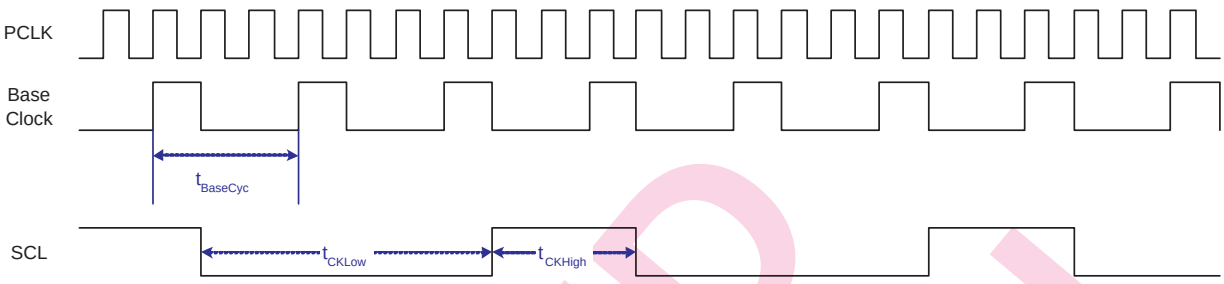
- Support totally **16** I2C/SMBus devices.
- All devices can be configured to Alertable SMBus device.
- Support 3 transfer modes:
  - **Byte mode**: 1 byte buffer
  - **Pool mode**: 32 bytes of internal SRAM for each device
  - **DMA mode**: Maximum 4096 bytes to or from SDRAM memory (New Regiser mode)
- Schmitt type of input data buffer and input clock buffers
- Anti-glitch data input filter
- Need external pull-up resistors

53.3 Migration from AST2500 to AST2600

1. Remove the supporting of 4 special slave addresses (0000_000, 0001_100, 0001_000, 1100_001), but add 1 more set of slave address3.
2. Add enable control for slave address1.
3. Add supporting range mode slave address, address range defined between address1 and address2.
4. Add new clock divider option for more flexible and accurate clock rate generation.
5. Add tCKHighMin timing to guarantee SCL high pulse width.
6. Increase pool buffer size of each device to 32 bytes.
7. Add support dual pool buffer mode, split 32 bytes pool buffer of each device into 2 x 16 bytes for Tx and Rx individually.
8. Increase DMA buffer size to 4096 bytes (new Register mode) and support byte alignment.
9. Re-define the base address of Device1 ∼ Device**16** and Pool buffer.
10. Re-define registers for separating master and slave mode control.
11. (New registers) Support 4 individual DMA buffers for master Tx and Rx, slave Tx and Rx.
12. (New registers) Support auto NACK response for slave mode if buffer is not ready.
13. (New registers) New Master packet operation mode: S → Aw → P[1].
14. (New registers) New Master packet operation mode: S → Aw → TxD → P
15. (New registers) New Master packet operation mode: S → Ar → RxD → P
16. (New registers) New Master packet operation mode: S → Aw → TxD → Sr → Ar → RxD → P
17. (New registers) New Slave packet operation mode: S → Aw → autoNACK → P.
18. (New registers) New Slave packet operation mode: S → Aw → RxD → Sr → Ar → autoNACK → P.
19. (New registers) New Slave packet operation mode: S → Aw → RxD → Sr → Ar → TxD → P.
20. (New registers) New packet operation mode for both Master and Slave only support Pool buffer and DMA modes.
21. (New registers) Byte mode will be disabled while slave packet operation mode is enabled.
22. (New registers) Bus SDA lock auto-release capability for new master DMA command mode.
23. (New registers) Bus auto timeout for new master/slave DMA mode.

[1] S: Start, Sr: Repeat Start, Aw/r: Address for write/read, P: Stop

53.4 Timing Definition

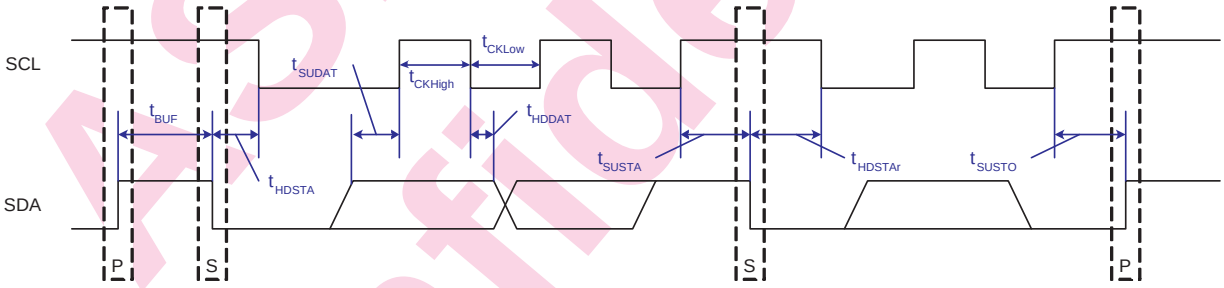


$Freq_{SCL} = Freq_{PCLK} / (t_{BaseCyc} * (t_{CKLow} + t_{CKHigh}))$

$t_{BaseCyc}$ = 1, 2, 4, 8, ... , 32768
t_{CKLow} = 1 ~ 16
t_{CKHigh} = 1 ~ 16

Because all AC timing definition are based on the Base Clock, so the clock divider setting is prefer that the value of t_{CKLow} and t_{CKHigh} as larger as possible for increasing AC timing resolution.

Figure 58: Clock Generation



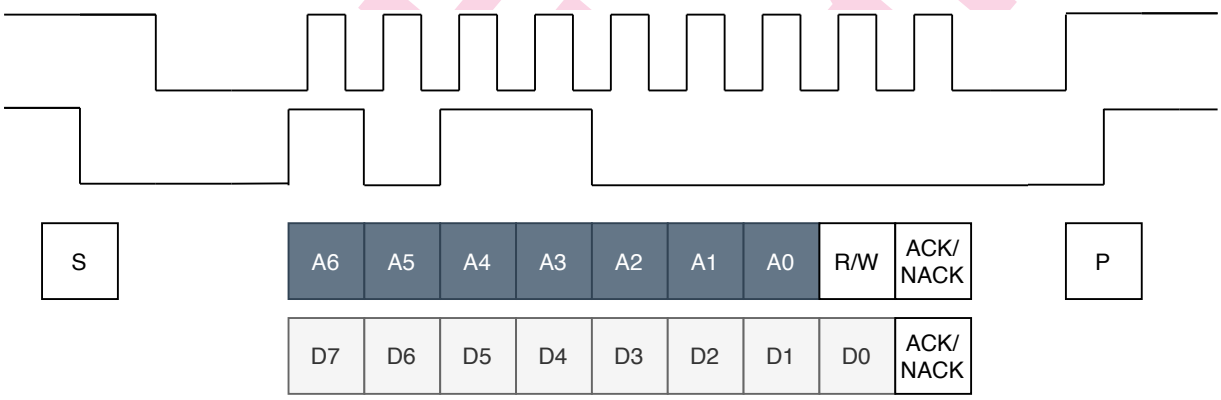
t_{SUSTA} : Repeated Start condition Setup time
t_{HDSTA} : Hold time after Start, After this period the first clock is generated
t_{HDSTAr} : Hold time after Repeated Start
t_{SUSTO} : Stop condition Setup time
t_{SUDAT} : Data Setup time
t_{HDDAT} : Data Hold time
t_{BUF} : Bus free time between Stop and Start condition

Relationship :
t_{SUDAT} = t_{CKLow} - t_{HDDAT}

Width $_{CKLow}$ = Max(t_{CKLow} , t_{HDDAT})

t_{SUSTA} , t_{HDSTA} , t_{HDSTAr} , t_{SUSTO} are default set at (16 * BaseCyc)

Figure 59: AC Timing



* Start(S) -> Address for read/write (Ar/Aw) -> Stop(P)
* { Bus Data (8bit) }

Figure 60: I2C PROTOCOL

53.4.1 Base clock selection rule

1. SCL clock rate = 100 KHz or smaller, base clock frequency $\geq$ 1 MHz
2. SCL clock rate = 400 KHz or smaller, base clock frequency $\geq$ 4 MHz
3. SCL clock rate = 1.0 MHz or smaller, base clock frequency $\geq$ 10 MHz
4. SCL clock rate = 3.4 MHz or smaller, base clock frequency $\geq$ 35 MHz
5. The selection rule is base clock frequency $\geq$ 10X SCL clock rate

53.4.2 Old divider mode clock rate calculation

1. H-CLK frequency : default 200MHZ
2. Get the PCLK frequency, SCU310[11:9]
   000 : PCLK = H-CLK/2
   001 : PCLK = H-CLK/4
   010 : PCLK = H-CLK/6
   011 : PCLK = H-CLK/8
   100 : PCLK = H-CLK/10
   101 : PCLK = H-CLK/12
   110 : PCLK = H-CLK/14
   111 : PCLK = H-CLK/16
3. Get the divider_ratio = PCLK / I2C_desired_clock_rate
4. Get the Base Clock Divisor I2CD04[3:0]

```
inc = 0;
div = 0;
while((divider_ratio + inc) > 32){
    inc |= divider_ratio & 0x1;
    divider_ratio >>= 1;
    div++;
}
divider_ratio += inc;
I2CD04[3:0] = div;
```

5. Get the SCL High/Low pulse width setting

   SCL_Low : I2CD04[15:12] = (divider_ratio >> 1) − 1;

   SCL_High : I2CD04[19:16] = divider_ratio − SCL_Low − 2;

53.4.3 New divider mode clock rate calculation

1. H-CLK frequency : default 200MHZ
2. Get the PCLK frequency, SCU310[11:9]
   000 : PCLK = H-CLK/2
   001 : PCLK = H-CLK/4
   010 : PCLK = H-CLK/6
   011 : PCLK = H-CLK/8
   100 : PCLK = H-CLK/10
   101 : PCLK = H-CLK/12
   110 : PCLK = H-CLK/14
   111 : PCLK = H-CLK/16
3. Set I2CG0C[1] = 1.
4. Properly define the base clock frequency of baseclk_1 ∼ baseclk_4 at I2CG10.
5. Get the Base Clock Divisor I2CC04[3:0]

```
if(PCLK / I2C_desired_clock_rate ≤ 32){
    div = 0;
```

```
        divider_ratio = PCLK / I2C_desired_clock_rate;
    }else if(baseclk_1 / I2C_desired_clock_rate ≤ 32){
        div = 1;
        divider_ratio = baseclk_1 / I2C_desired_clock_rate;
    }else if(baseclk_2 / I2C_desired_clock_rate ≤ 32){
        div = 2;
        divider_ratio = baseclk_2 / I2C_desired_clock_rate;
    }else if(baseclk_3 / I2C_desired_clock_rate ≤ 32){
        div = 3;
        divider_ratio = baseclk_3 / I2C_desired_clock_rate;
    }else{
        divider_ratio = baseclk_4 / I2C_desired_clock_rate;
        div = 4;
        inc = 0;
        while((divider_ratio + inc) > 32){
            inc |= divider_ratio & 0x1;
            divider_ratio >>= 1;
            div++;
        }
        divider_ratio += inc;
    }
    I2CC04[3:0] = div;
```

6. Get the SCL High/Low pulse width setting

   SCL_Low : I2CC04[15:12] = (divider_ratio >> 1) − 1;

   SCL_High : I2CC04[19:16] = divider_ratio − SCL_Low − 2;

   SCL_HighMin : I2CC04[23:20] = proper and smaller value than SCL_High;

7. SCL_HighMin is a feedback sampling of external SCL clock to guarantee the minimum valid high pulse width. The setting will affect the actual SCL high pulse cycle time if not properly set or the SCL rise time is slow.

53.4.4 Clock Setting Table for Quick Start

- set I2CG0C[1] = 1 (new divider mode)
- set I2CG10 = 0
- Divisor = APB Clock Frequency / Data Bit Rate
- find (n,x) where Divisor = 2^n * x and x $<$ 32
- set I2CC04[3:0] = 4 + n, tCKLow = (x/2)-1, tCKHigh = x - (tCKLow+1) - 1

Divisor	Base Div(n)	tCK High	tCK Low	Divisor	Base Div(n)	tCK High	tCK Low	Divisor	Base Div(n)	tCK High	tCK Low
				68	2	8	7	464	4	14	13
6	0	2	2	72	2	8	8	480	4	14	14
7	0	3	2	76	2	9	8	496	4	15	14
8	0	3	3	80	2	9	9	512	4	15	15
9	0	4	3	84	2	10	9	544	5	8	7
10	0	4	4	88	2	10	10	576	5	8	8
11	0	5	4	92	2	11	10	608	5	9	8
12	0	5	5	96	2	11	11	640	5	9	9
13	0	6	5	100	2	12	11	672	5	10	9
14	0	6	6	104	2	12	12	704	5	10	10
15	0	7	6	108	2	13	12	736	5	11	10
16	0	7	7	112	2	13	13	768	5	11	11
17	0	8	7	116	2	14	13	800	5	12	11
18	0	8	8	120	2	14	14	832	5	12	12
19	0	9	8	124	2	15	14	864	5	13	12
20	0	9	9	128	2	15	15	896	5	13	13
21	0	10	9	136	3	8	7	928	5	14	13
22	0	10	10	144	3	8	8	960	5	14	14
23	0	11	10	152	3	9	8	992	5	15	14
24	0	11	11	160	3	9	9	1024	5	15	15
25	0	12	11	168	3	10	9	1088	6	8	7
26	0	12	12	176	3	10	10	1152	6	8	8
27	0	13	12	184	3	11	10	1216	6	9	8
28	0	13	13	192	3	11	11	1280	6	9	9
29	0	14	13	200	3	12	11	1344	6	10	9
30	0	14	14	208	3	12	12	1408	6	10	10
31	0	15	14	216	3	13	12	1472	6	11	10
32	0	15	15	224	3	13	13	1536	6	11	11
34	1	8	7	232	3	14	13	1600	6	12	11
36	1	8	8	240	3	14	14	1664	6	12	12
38	1	9	8	248	3	15	14	1728	6	13	12
40	1	9	9	256	3	15	15	1792	6	13	13
42	1	10	9	272	4	8	7	1856	6	14	13
44	1	10	10	288	4	8	8	1920	6	14	14
46	1	11	10	304	4	9	8	1984	6	15	14
48	1	11	11	320	4	9	9	2048	6	15	15
50	1	12	11	336	4	10	9	2176	7	8	7
52	1	12	12	352	4	10	10	2304	7	8	8
54	1	13	12	368	4	11	10	2432	7	9	8
56	1	13	13	384	4	11	11	2560	7	9	9
58	1	14	13	400	4	12	11	2688	7	10	9
60	1	14	14	416	4	12	12	2816	7	10	10
62	1	15	14	432	4	13	12	2944	7	11	10
64	1	15	15	448	4	13	13	3072	7	11	11

53.5 Registers : Base Address = 0x1E78:A000

53.5.1 Address Definition

Offset	Size(Byte)	Description
07F-000	128	Global Register
0FF-080	128	Device 1
17F-100	128	Device 2
1FF-180	128	Device 3
27F-200	128	Device 4
2FF-280	128	Device 5
37F-300	128	Device 6
3FF-380	128	Device 7
47F-400	128	Device 8
4FF-480	128	Device 9
57F-500	128	Device 10
5FF-580	128	Device 11
67F-600	128	Device 12
6FF-680	128	Device 13
77F-700	128	Device 14
7FF-780	128	Device 15
87F-800	128	Device 16
C1F-C00	32	Device 1 buffer
C3F-C20	32	Device 2 buffer
C5F-C40	32	Device 3 buffer
C7F-C60	32	Device 4 buffer
C9F-C80	32	Device 5 buffer
CBF-CA0	32	Device 6 buffer
CDF-CC0	32	Device 7 buffer
CFF-CE0	32	Device 8 buffer
D1F-D00	32	Device 9 buffer
D3F-D20	32	Device 10 buffer
D5F-D40	32	Device 11 buffer
D7F-D60	32	Device 12 buffer
D9F-D80	32	Device 13 buffer
DBF-DA0	32	Device 14 buffer
DDF-DC0	32	Device 15 buffer
DFF-DE0	32	Device 16 buffer

53.5.2 Global Register Definition

Offset: 00h	**I2CG00: Device Master Mode Interrupt Status Register (I2CG0C[3] = 1)**	**Init = 0**
Offset: 00h	**I2CG00: Device Master/Slave Mode Interrupt Status Register (I2CG0C[3] = 0)**	**Init = 0**

Bit	Attr.	Description
31:16	RO	**Reserved (0)**
15	RO	**I2C/SMBus Device #16 Interrupt**
14	RO	**I2C/SMBus Device #15 Interrupt**
13	RO	**I2C/SMBus Device #14 Interrupt**
12	RO	**I2C/SMBus Device #13 Interrupt**
11	RO	**I2C/SMBus Device #12 Interrupt**
10	RO	**I2C/SMBus Device #11 Interrupt**
9	RO	**I2C/SMBus Device #10 Interrupt**
8	RO	**I2C/SMBus Device #9 Interrupt**
7	RO	**I2C/SMBus Device #8 Interrupt**
6	RO	**I2C/SMBus Device #7 Interrupt**
5	RO	**I2C/SMBus Device #6 Interrupt**
4	RO	**I2C/SMBus Device #5 Interrupt**
3	RO	**I2C/SMBus Device #4 Interrupt**
2	RO	**I2C/SMBus Device #3 Interrupt**
1	RO	**I2C/SMBus Device #2 Interrupt**
0	RO	**I2C/SMBus Device #1 Interrupt** 0 : No interrupt 1 : Interrupt occurs

Note :
This global register shows the summary report of the interrupt events from all of the **16** devices. There is no need to clear the interrupt status of this register.

Offset: 04h	**I2CG04: Device Slave Mode Interrupt Status Register**	**Init = 0**

Bit	R/W	Description
31:16	RO	**Reserved (0)**
15	RO	**I2C/SMBus Device #16 Interrupt**
14	RO	**I2C/SMBus Device #15 Interrupt**
13	RO	**I2C/SMBus Device #14 Interrupt**
12	RO	**I2C/SMBus Device #13 Interrupt**
11	RO	**I2C/SMBus Device #12 Interrupt**
10	RO	**I2C/SMBus Device #11 Interrupt**
9	RO	**I2C/SMBus Device #10 Interrupt**
8	RO	**I2C/SMBus Device #9 Interrupt**
7	RO	**I2C/SMBus Device #8 Interrupt**
6	RO	**I2C/SMBus Device #7 Interrupt**
5	RO	**I2C/SMBus Device #6 Interrupt**
4	RO	**I2C/SMBus Device #5 Interrupt**
3	RO	**I2C/SMBus Device #4 Interrupt**
2	RO	**I2C/SMBus Device #3 Interrupt**
1	RO	**I2C/SMBus Device #2 Interrupt**

to next page

from previous page

0	RO	**I2C/SMBus Device #1 Interrupt** 0 : No interrupt 1 : Interrupt occurs

Note :
This global register shows the summary report of the interrupt events from all of the **16** devices. There is no need to clear the interrupt status of this register.

Offset: 0Ch **I2CG0C: Global Control Register** **Init = 0**

Bit	R/W	Description
31:12	RO	**Reserved (0)**
11:8	RW	**Master transmit to receive turn-around delay** 0000: no delay 0001: delay 1 cycle of Base Clock 0010: delay 2 cycles of Base Clock .... 1111: delay 15 cycles of Base Clock This timing is only applied on master packet operation mode.
7:5	RO	**Reserved (0)**
4	RW	**Select the action when Slave packet mode RX buffer empty** 0: Issue packet done interrupt and pull SCL low to halt bus. (same as non-packet mode) 1: Expect to receive Repeat Start or Stop command, if more SRX data input then it will respond with NACK. Here the RX buffer empty means the received data bytes equals to the allocated buffer size. In non-packet operation mode, slave controller will stop reciving data and pull SCL low to halt bus, and then issue interrupt for this condition. In packet operation mode, it has another option to continue the operation when buffer empty. It expect to receive Repeat Start or Stop command after RX buffer empty, and will respond with NACK if the succeeding command is still RX.
3	RW	**Enable separating master and slave mode interrupts** 0: Master and slave mode interrupts are merged on I2CG00 1: Master interrupts is on I2CG00, and slave interrupts is on I2CG04
2	RW	**Register definition selection** 0: Old registers definition 1: New registers definition
1	RW	**Clock divider mode selection** 0: Old clock divider mode 1: New clock divider mode
0	RW	**Reserved (0)**

Offset: 10h **I2CG10: New Clock Divider Control Register (I2CG0C[1] = 1)** **Init = 0**

Bit	R/W	Description
31:24	RW	**Base clock 4 divisor, base_divider_4**
23:16	RW	**Base clock 3 divisor, base_divider_3**
15:8	RW	**Base clock 2 divisor, base_divider_2**

to next page

from previous page

7:0	RW	**Base clock 1 divisor, base_divider_1** 0x00: divided by 1 0x01: divided by 1.5 0x02: divided by 2 0x03: divided by 2.5 .... 0xFE: divided by 128 0xFF: divided by 128.5

53.5.3 Old Register Mode Definition

Offset: 00h		**I2CD00: Function Control Register**	**Init = 0x0**
Bit	**R/W**	**Description**	
31:18	RO	**Reserved (0)**	
17	RW	**Enable bus auto-release when SCL low, SDA low, or slave mode inactive timeout** 0: Disable 1: Enable The timeout timing setting is defined at I2CD08. To enable this feature, at least one bit of I2CD0C[6] or I2CD0C[15:14] must be enabled. This feature is valid only when none of master or slave Packet operation mode were enabled.	
16	RW	**Enable master auto SDA lock recovery capability (for single master case only)** 0: Disable 1: Enable When enabled, master will auto generate clock to recover SDA lock condition, if SDA was locked before issue START pattern command.	
15	RW	**Disable multi-master capability (for master function only)** 0: Enable 1: Disable (for single master application only) When disabled, device controller assume that there will have no arbitration lost possibility and ignore the clock stretch check.	
14	RW	**Enable SCL direct drive mode (for master function only)** 0: Disable (SCL output buffer is configured as an open-drain buffer with an external pull-up resistor) 1: Enable (SCL always drives output buffer, no tri-state) This is a special mode, it can be used only under single master condition and no clock stretching devices allowed on this bus. Slave mode can not be enabled when use this mode.	
13:10	RO	**Reserved (0)**	
9	RW	**Enable slave address range mode** 0: normal address mode for address1 and address2 1: address1 $\leq$ slave address $\leq$ address2 When select range mode, the address matching indicator is assigned to address1.	
8	RW	**Enable SDA signal to direct drive high for 1T** 0: Disable (SDA signal passively pulls high by an external pull-up resistor) 1: Enable (SDA signal actively drives high by current source for 1 PCLK cycle before entering tri-state) The function is designed to support a higher transfer rate.	

to next page

from previous page

7	RW	**Enable SCL signal to direct drive high for 1T (Master Only)** 0: Disable (SCL signal passively pulls high by an external pull-up resistor) 1: Enable (SCL signal actively drives high by current source for 1 PCLK cycle before entering tri-state) The function is designed to support a higher transfer rate. If high speed mode enabled, then it will drive SCL until it is really high and then release. Not limited by 1T. So no clock stretch allowed.
6	RW	**Enable High Speed master mode** 0 : normal speed mode 1 : high speed mode (3.4Mbps) High speed mode can only use buffer or DMA mode for transfer. And only master mode supports speed switching capability. At high speed mode, the clock rate is determined by I2CD04[19:12] with baseclk_0.
5	RW	~~Enable I2C/SMBus to respond the Default Address (1100_001)~~
4	RW	~~Enable I2C/SMBus to respond the Alert Address (0001_100)~~
3	RW	~~Enable I2C/SMBus to respond the ARP Host Address (0001_000)~~
2	RW	~~Enable I2C/SMBus to respond the General Call Address (0000_0000)~~
1	RW	**Enable slave function** 0 : Disable 1 : Enable Each device can be enabled to support either slave or master function or both functions.
0	RW	**Enable master function** 0 : Disable 1 : Enable Each device can be enabled to support either slave or master function or both functions.

Note :
The respective controller and the following registers of the respective controller will be reset whenever its master function and slave function are both disabled simultaneously.
1. I2CD10, I2CM14, I2CS24 : Interrupt Status Register
2. I2CD14, I2CM18, I2CS28 : Command Control Register

Offset: 04h **I2CD04: Clock and AC Timing Control Register #1** **Init = 0x0**

Bit	R/W	Description
31:24	RO	**Reserved (0)**
23:20	RO	**Reserved** It returns the value of I2CD04[19:17].
19:16	RW	**Cycles of master SCL clock-high pulse width (tCKHigh)** 000x: no guarantee pulse width 0011: 4 cycles of Base Clock 0100: 5 cycles of Base Clock .... 1111: 16 cycles of Base Clock
15:12	RW	**Cycles of master SCL clock-low pulse width (tCKLow)** 000x: no guarantee pulse width 0011: 4 cycles of Base Clock 0100: 5 cycles of Base Clock .... 1111: 16 cycles of Base Clock

to next page

from previous page

11:10	RW	**Hold time of master/slave data (tHDDAT)** Master Slave 00 1 0 01 2 1 10 3 2 11 4 3 The unit for the table is numbers of Base Clock
9:8	RW	**Timeout base clock divisor (toutBaseClk)** This divisor defines the Base Clock for Timeout Counter.
7:4	RW	**Reserved**
3:0	RW	**Base Clock divisor (tBaseClk)** The divisor defines the frequency of Base Clock which is divided from PCLK clock. This register defines the frequency of a free running counter which generates Base Clock for controlling the related AC timings. When switch to High Speed (HS) mode, the divisor will be switch to 0 by hardware automatically.

	Clock name	Old divider mode	New divider mode
00	tout_baseclk_0	PCLK / 16384	baseclk_4 / 256
01	tout_baseclk_1	PCLK / 65536	baseclk_4 / 1024
10	tout_baseclk_2	PCLK / 262144	baseclk_4 / 4096
11	tout_baseclk_3	PCLK / 1048576	baseclk_4 / 8192

	Clock name	Old divider mode	New divider mode
0000	baseclk_0	PCLK / 1	PCLK / 1
0001	baseclk_1	PCLK / 2	PCLK / base_divider_1
0010	baseclk_2	PCLK / 4	PCLK / base_divider_2
0011	baseclk_3	PCLK / 8	PCLK / base_divider_3
0100	baseclk_4	PCLK / 16	PCLK / base_divider_4 (1 MHz)
0101	baseclk_5	PCLK / 32	baseclk_4 / 2
0110	baseclk_6	PCLK / 64	baseclk_4 / 4
0111	baseclk_7	PCLK / 128	baseclk_4 / 8
1000	baseclk_8	PCLK / 256	baseclk_4 / 16
1001	baseclk_9	PCLK / 512	baseclk_4 / 32
1010	baseclk_10	PCLK / 1024	baseclk_4 / 64
1011	baseclk_11	PCLK / 2048	baseclk_4 / 128
1100	baseclk_12	PCLK / 4096	baseclk_4 / 256
1101	baseclk_13	PCLK / 8192	baseclk_4 / 512
1110	baseclk_14	PCLK / 16384	baseclk_4 / 1024
1111	baseclk_15	PCLK / 32768	baseclk_4 / 2048

Offset: 08h **I2CD08: Clock and AC Timing Control Register #2** **Init = 0x0**

Bit	R/W	Description
31:5	RO	**Reserved**

to next page

from previous page

4:0	RW	**Timeout timer for SCL low, SDA low, and slave active timeout (tTimeOut)** 0: No Timeout Control 1: 1 cycle of Timeout Base Clock 2: 2 cycles of Timeout Base Clock .... 31: 31 cycles of Timeout Base Clock Timeout Period = tTimeOut*(1/toutBaseClk) = tTimeOut*div / PCLK ex: Timeout Period = 6* 16384/PCLK = 0.983ms when PCLK = 100MHZ, div=16384 from I2CD04[9:8]

Offset: 0Ch **I2CD0C: Interrupt Control Register** **Init = 0x0**

Bit	R/W	Description
31:16	RO	**Reserved (0)**
15	RW	**Enable Slave mode inactive timeout interrupt** Alert for the state machine stays at Slave active mode for a long time (timeout counter). This maybe the condition that controller work slave mode, but suddenly the master stop to transfer, and this makes the state machine stuck at slave mode and can not quit.
14	RW	**Enable SDA data-low timeout interrupt**
13	RW	**Enable Bus Recover Done interrupt**
12	RW	**Enable SMBus Device Alert interrupt**
11:7	RO	**Reserved (0)**
6	RW	**Enable SCL clock-low timeout interrupt**
5	RW	**Enable abnormal Start/Stop condition detection interrupt** This interrupt is used to indicate a bus Start/Stop condition has been detected at an illegal transfer state.
4	RW	**Enable normal Stop condition detection interrupt** For master mode, this interrupt is used to report that a Stop pattern has been issued. For salve mode, this interrupt is used to report a Stop pattern has been detected
3	RW	**Enable master arbitration loss interrupt**
2	RW	**Enable Receive Done interrupt** Receive Done means : 1. Master : all the expected bytes have been received. 2. Slave : the buffer is full, or salve controller receives a termination signal.
1	RW	**Enable Transmit ended with NACK response interrupt**
0	RW	**Enable Transmit ended with ACK response interrupt** Transmit end means Byte, Pool or DMA mode Tx command end.

Note :
The definition of this register is :
0 : Disable
1 : Enable

Offset: 10h	I2CD10: Interrupt Status Register	Init = 0x0

Bit	R/W	Description
31:30	RO	**Slave address match indicator** 00: address1 01: address2 10: address3
29	RO	**Slave Address Received Pending** 0: none 1: A slave address match received interrupt is pending due to previous slave DMA receive done is not processed.
28:16	RO	**Refer to I2CS24[28:16]**
15	RW1C	**(WC) Slave mode inactive timeout interrupt status**
14	RW1C	**(WC) SDA data-low timeout interrupt status**
13	RW1C	**(WC) Bus Recover Done interrupt status**
12	RW1C	**(WC) SMBus Device Alert interrupt status**
11	RO	~~SMBus ARP Host Address Detection interrupt status~~
10	RO	~~SMBus Device Alert Response Address Detection interrupt status~~
9	RO	~~SMBus Device Default Address Detection interrupt status~~
8	RO	~~General Call Address Detection interrupt status~~
7	RW1C	**(WC) Slave Address Received Match interrupt status**
6	RW1C	**(WC) SCL clock-low timeout interrupt status**
5	RW1C	**(WC) Abnormal Start/Stop Condition Detection interrupt status**
4	RW1C	**(WC) Normal Stop Condition Detection interrupt status**
3	RW1C	**(WC) Master Arbitration Loss interrupt status**
2	RW1C	**(WC) Receive Done interrupt status** S/W needs to clear this status bit to enable next data receiving. And at (slave) byte buffer mode, this interrupt status may be set concurrently along with bit[11:7].
1	RW1C	**(WC) Transmit ended with NACK response interrupt status**
0	RW1C	**(WC) Transmit ended with ACK response interrupt status**

Note :
'WC' means this bit is cleared by writing '1'.

Offset: 14h	I2CD14: Command/Status Register	Init = 0x0A060000

Bit	R/W	Description
31:29	RO	**Transfer Data Direction** 000: IDLE 001: WAIT 100: MTX 101: MRX 110: STX 111: SRX
28	RO	**SDA_OE**
27	RO	**SDA_O**
26	RO	**SCL_OE**
25	RO	**SCL_O**

to next page

from previous page

24:23	RO	**Transfer Mode Timing Stage** 00: T0 01: T1 10: T2 11: T3
22:19	RO	**Transfer Mode State Machine** 0000: IDLE 1000: MACTIVE 1001: MSTART 1010: MSTARTR 1011: MSTOP 1100: MTXD 1101: MRXACK 1110: MRXD 1111: MTXACK 0001: SWAIT 0100: SRXD 0101: STXACK 0110: STXD 0111: SRXACK 0011: RECOVER
18	RO	**Sampled SCL Line State**
17	RO	**Sampled SDA Line State**
16	RO	**Bus Busy Status** 0: Bus is Idle 1: Bus is Busy or not meets Idle Timing Requirement
15	RW	**SDA_OE output direct control** Bit[15:12] is a GPIO function and only work when both Master and Slave function are disabled. When bus lock occurs, this GPIO function can help to recover the bus life. 0: output disable, tri-stated 1: output enable
14	RW	**SDA_O output direct control**
13	RW	**SCL_OE output direct control** 0: output disable, tri-stated 1: output enable
12	RW	**SCL_O output direct control**
11	RW	**Enable Bus Recover Command** 0: NOP 1: Issue Bus Recover Command This command can be applied only when SCL line status is High. When Bus Recovery Command is fired, 1 $\sim$ 8 SCL clock cycles will be issued to recover the bus. (Automatically stop issuing clock cycle whenever SDA has been released to high state) This command is used to recover the bus when a slave device locks SDA. When using this command, the Transfer Mode State Machine must be at the IDLE state; otherwise, it must be reset to the IDLE state firstly by disabling the device function (I2CD00). When recover operation has been done, the current bus state can be read back from Bit [17] of this register.

to next page

from previous page

10	RW	**Enable issuing I2C/SMBus Slave Alert signal** 0: NOP 1: Issuing Alert signal to a bus Master This command is valid only when slave function is enabled. This bit will be cleared by hardware automatically after packet received with slave address matched.
9	RW	**Enable Master/Slave Receive Data DMA Mode** 0: Disable 1: Enable HW will clear this register automatically when data receiving has been done. DMA transmit and receive can not be enabled at the same time. The result is un-predictable.
8	RW	**Enable Master/Slave Transmit Data DMA Mode** 0: Disable 1: Enable HW will clear this register automatically when data transmitting has been done. When set, DMA will start to fetch memory data, no matter I2C is start to active or not. So all the DMA related control registers must be set before enable this bit.
7	RW	**Enable Master/Slave Receive Data Buffer** 0: Disable 1: Enable This register will be automatically cleared by H/W when data receiving has been done.
6	RW	**Enable Master/Slave Transmit Data Buffer** 0: Disable 1: Enable This register will be automatically cleared by H/W when data receiving has been done.
5	RW	**Master Stop Command** 0: NOP 1: Issue Master Stop Command **4th** priority. This register will be automatically cleared by H/W when Stop Command has been issued. This command is valid only when master mode is enabled
4	RW	**Master/Slave Receive Command Last** 0: Receive Command can be continued by responding ACK 1: Receive Command will be ended by responding NACK When in buffer mode, the last control will acts after the latest byte is been received. When in Master mode and Stop Command activated, the last control must be set to ending transfer.
3	RW	**Master Receive Command** 0: NOP 1: Fire Master Receive Command **3rd** priority. HW will clear this register when RX buffer is full or receiving is terminated (Stop/Repeated Start). This command is valid only when Master mode is enabled
2	RW	**Slave Transmit Command** 0: NOP 1: Fire Slave Transmit Command HW will clear this register when TX buffer is empty or bus contention error has been detected.

to next page

from previous page

1	RW	**Master Transmit Command** 0: NOP 1: Fire Master Transmit Command **2nd** priority. HW will clear this register when TX buffer is empty or Bus Contention error has been detected.
0	RW	**Master Start Command** 0: NOP 1: Issue Master Start/Repeated Start Command **1st** priority. This register will be automatically cleared by HW when Master Start Command or Repeated Start Command has been issued. This command will be executed by HW only when master mode is enabled and the bus is in idle state.

Note :
When multiple commands in this register are fired simultaneously, Device controller will execute these commands according to the following sequence (priority):

(1) Master Start Command
(2) Master Transmit Command
(3) Master Receive Command
(4) Master Stop Command

HW will automatically clear each command (bit[11:0]) when it has been finished. Beside, HW will clear all the commands whenever Master Arbitration Lost or invalid Start/Stop conditions have been detected.
Attention: Master and Slave Command cannot be activated at the same time.

Offset: 18h **I2CD18: Slave Device Address Register** **Init = 0x80**

Bit	R/W	Description
31:24	RO	**Reserved (0)**
23	RW	**Enable Slave Device Address3** 0: disable 1: enable
22:16	RW	**Slave Device Address3**
15	RW	**Enable Slave Device Address2** 0: disable 1: enable
14:8	RW	**Slave Device Address2**
7	RW	**Enable Slave Device Address1 (only for New Register mode)** 0: disable 1: enable
6:0	RW	**Slave Device Address1**

Offset: 1Ch **I2CD1C: Pool Buffer Control Register** **Init = 0x0**

Bit	R/W	Description
31:30	RO	**Reserved**

to next page

from previous page

29:24	RW	**Actual Received Pool Buffer Size** 0 = 0 bytes 1 = 1 byte .... 32 = 32 bytes Write command can clear this field to 0.
23:21	RO	**Reserved**
20:16	RW	**Receive Pool Buffer Size** 0 = 1 byte space 1 = 2 bytes space .... 31 = 32 bytes space This value defines the maximum receive buffer size for Slave mode, or receive command data byte count for Master mode.
15:13	RO	**Reserved**
12:8	RW	**Transmit Data Byte Count** 0 = 1 byte 1 = 2 bytes .... 31 = 32 bytes
7:1	RO	**Reserved**
0	RW	**Buffer Organization** 0: 32 bytes for Tx or Rx 1: Lower 16 bytes for Tx and higher 16 bytes for Rx

Note :
The buffer is shared by Pool buffer mode and DMA mode of Master and Slave controllers. So there are some limitations on using the buffer.
1. The buffer can only be used by Master or Slave controller, but not both at a time.
2. When Master or Slave enables DMA mode, then Pool buffer mode can not be used.

Offset: 20h **I2CD20: Transmit/Receive Byte Buffer Register** **Init = 0x0**

Bit	R/W	Description
31:16	RO	**Reserved**
15:8	RO	**Receive Byte Buffer** This register is valid when DMA Buffer is not enabled.
7:0	RW	**Transmit Byte Buffer** This register is valid when DMA Buffer is not enabled.

Offset: 24h **I2CD24: DMA Mode Buffer Address Register** **Init = 0x80000000**

Bit	R/W	Description
31	RO	**Reserved (0x1)**
30:0	RW	**SDRAM DMA Buffer Base Address** The base address will be updated while DMA is in progress. To check the original address setting, it is at I2CD2C.

Offset: 28h	I2CD28: DMA Transfer Length Register	Init = 0x0
Bit	**R/W**	**Description**
31:12	RO	**Reserved**
11:0	RW	**DMA Transfer Length (Byte)** Define the maximum packet size for transmitting or receiving. 0: Not allowed, will cause unpredictable error 1: 1 bytes .... 4095: 4095 bytes The length will be decremented by hardware when transfer in progress. After transfer completed, the actual transmitted length equals total_buffer_length - current_buffer_length. For master mode, the length should be the actual value for transmitting and receiving. For slave mode, the length value can be larger than or equal to the actual transmitting/receiving length. Due to the transmit/receive job may be terminated by master at any length.

Offset: 2Ch	I2CD2C: Original DMA Mode Buffer Address Setting	Init = 0x0
Bit	**R/W**	**Description**
31:0	RO	**Original SDRAM DMA Buffer Base Address**

Offset: 30h	I2CD30: Original DMA Transfer Length Setting and Final Status	Init = 0x0
Bit	**R/W**	**Description**
31:29	RO	**Reserved**
28:16	RO	**The actual DMA transferred length**
15:12	RO	**Reserved**
11:0	RO	**Original DMA Transfer Length (Byte)**

53.5.4 New Register Mode Definition

Offset: 00h	I2CC00: Master/Slave Function Control Register	Init = 0x0
Bit	**R/W**	**Description**
21	RW	**DisableDMA AutoRecovery when residual Tx Count equals 0**
20	RW	**Enable save address byte into buffer for Slave Packet mode receive command** 0: No save address byte 1: Save address byte For Slave Packet operation mode, the slave address byte can be chosen to be saved into buffer or not. The slave address byte is default saved under stepping command mode.
19:18	RW	**Master packet operation retry count** 00: No retry 01: Retry 1 time 10: Retry 2 times 11: Retry 3 times

to next page

from previous page

17	RW	**Enable bus auto-release when SCL low, SDA low, or slave mode inactive timeout** 0: Disable 1: Enable The timeout timing setting is defined at I2CC04[28:24]. To enable this feature, at least one bit of I2CM14[6] or I2CM14[14] or I2CS20[15] must be enabled. This feature is valid only when none of master or slave Packet operation mode were enabled.
16	RW	**Enable master auto SDA lock recovery capability (for single master case only)** 0: Disable 1: Enable When enabled, master will auto generate clock to recover SDA lock condition, if SDA was locked before issue START pattern command.
15	RW	**Disable multi-master capability (for master function only)** 0: Enable 1: Disable (for single master application only) When disabled, device controller assume that there will have no arbitration lost possibility and ignore the clock stretch check.
14	RW	**Enable SCL direct drive mode (for master function only)** 0: Disable (SCL output buffer is configured as an open-drain buffer with an external pull-up resistor) 1: Enable (SCL always drives output buffer, no tri-state) This is a special mode, it can be used only under single master condition and no clock stretching devices allowed on this bus. Slave mode can not be enabled when use this mode.
13:10	RO	**Reserved (0)**
9	RW	**Enable slave address range mode** 0: normal address mode for address1 and address2 1: address1 $\leq$ slave address $\leq$ address2 When select range mode, the address matching indicator is assigned to address1.
8	RW	**Enable SDA signal to direct drive high for 1T** 0: Disable (SDA signal passively pulls high by an external pull-up resistor) 1: Enable (SDA signal actively drives high by current source for 1 PCLK cycle before entering tri-state) The function is designed to support a higher transfer rate.
7	RW	**Enable SCL signal to direct drive high for 1T (Master Only)** 0: Disable (SCL signal passively pulls high by an external pull-up resistor) 1: Enable (SCL signal actively drives high by current source for 1 PCLK cycle before entering tri-state) The function is designed to support a higher transfer rate. If high speed mode enabled, then it will drive SCL until it is really high and then release. Not limited by 1T. So no clock stretch allowed.
6	RW	**Enable High Speed master mode** 0 : normal speed mode 1 : high speed mode (3.4Mbps) High speed mode can only use buffer or DMA mode for transfer. And only master mode supports speed switching capability. At high speed mode, the clock rate is determined by I2CC04[19:12] with baseclk_0.
5	RW	~~Enable I2C/SMBus to respond the Default Address (1100_001)~~
4	RW	~~Enable I2C/SMBus to respond the Alert Address (0001_100)~~
3	RW	~~Enable I2C/SMBus to respond the ARP Host Address (0001_000)~~

to next page

from previous page

2	RW	~~Enable I2C/SMBus to respond the General Call Address (0000_0000)~~
1	RW	**Enable slave function** 0 : Disable 1 : Enable Each device can be enabled to support either slave or master function or both functions.
0	RW	**Enable master function** 0 : Disable 1 : Enable Each device can be enabled to support either slave or master function or both functions.
Note : The respective controller and the following registers of the respective controller will be reset whenever its master function and slave function are both disabled simultaneously. 1. I2CM14, I2CS24 : Interrupt Status Register 2. I2CM18, I2CS28 : Command Control Register		

Offset: 04h I2CC04: Master/Slave Clock and AC Timing Control Register #1 Init = 0x0

Bit	R/W	Description
31:29	RO	**Reserved (0)**
28:24	RW	**Timeout timer for SCL low, SDA low, and slave active timeout (tTimeOut)** 0: No Timeout Control 1: 1 cycle of Timeout Base Clock 2: 2 cycles of Timeout Base Clock .... 31: 31 cycles of Timeout Base Clock Timeout Period = tTimeOut*(1/toutBaseClk) = tTimeOut*div / PCLK ex: Timeout Period = 6* 16384/PCLK = 0.983ms when PCLK = 100MHZ, div=16384 from I2CC04[9:8]
23:20	RW	**Cycles of master SCL clock-high minimum pulse width (tCKHighMin)** 000x: no guarantee pulse width 0011: 4 cycles of Base Clock 0100: 5 cycles of Base Clock .... 1111: 16 cycles of Base Clock This register is used to guarantee the minimum SCL high pulse width by sampling the feedback SCL input. It should be set to be less than or equal to tCKHigh. The actual SCL high pulse width is determined by tCKHigh and tCKHighMin. This timing will lengthen the SCL high duty width, which is depending on the SCL rise time.
19:16	RW	**Cycles of master SCL clock-high pulse width (tCKHigh)** 000x: no guarantee pulse width 0011: 4 cycles of Base Clock 0100: 5 cycles of Base Clock .... 1111: 16 cycles of Base Clock
15:12	RW	**Cycles of master SCL clock-low pulse width (tCKLow)** 000x: no guarantee pulse width 0011: 4 cycles of Base Clock 0100: 5 cycles of Base Clock .... 1111: 16 cycles of Base Clock

to next page

from previous page

<table>
<tr><td>11:10</td><td>RW</td><td>Hold time of master/slave data (tHDDAT)
<table>
<tr><td></td><td>Master</td><td>Slave</td></tr>
<tr><td>00</td><td>1</td><td>0</td></tr>
<tr><td>01</td><td>2</td><td>1</td></tr>
<tr><td>10</td><td>3</td><td>2</td></tr>
<tr><td>11</td><td>4</td><td>3</td></tr>
</table>
The unit for the table is numbers of Base Clock</td></tr>
<tr><td>9:8</td><td>RW</td><td>Timeout base clock divisor (toutBaseClk)
<table>
<tr><td></td><td>Clock name</td><td>Old divider mode</td><td>New divider mode</td></tr>
<tr><td>00</td><td>tout_baseclk_0</td><td>PCLK / 16384</td><td>baseclk_4 / 256</td></tr>
<tr><td>01</td><td>tout_baseclk_1</td><td>PCLK / 65536</td><td>baseclk_4 / 1024</td></tr>
<tr><td>10</td><td>tout_baseclk_2</td><td>PCLK / 262144</td><td>baseclk_4 / 4096</td></tr>
<tr><td>11</td><td>tout_baseclk_3</td><td>PCLK / 1048576</td><td>baseclk_4 / 8192</td></tr>
</table>
This divisor defines the Base Clock for Timeout Counter.</td></tr>
<tr><td>7:4</td><td>RW</td><td>Reserved</td></tr>
<tr><td>3:0</td><td>RW</td><td>Base Clock divisor (tBaseClk)
The divisor defines the frequency of Base Clock which is divided from PCLK clock.
<table>
<tr><td></td><td>Clock name</td><td>Old divider mode</td><td>New divider mode</td></tr>
<tr><td>0000</td><td>baseclk_0</td><td>PCLK / 1</td><td>PCLK / 1</td></tr>
<tr><td>0001</td><td>baseclk_1</td><td>PCLK / 2</td><td>PCLK / base_divider_1</td></tr>
<tr><td>0010</td><td>baseclk_2</td><td>PCLK / 4</td><td>PCLK / base_divider_2</td></tr>
<tr><td>0011</td><td>baseclk_3</td><td>PCLK / 8</td><td>PCLK / base_divider_3</td></tr>
<tr><td>0100</td><td>baseclk_4</td><td>PCLK / 16</td><td>PCLK / base_divider_4 (1 MHz)</td></tr>
<tr><td>0101</td><td>baseclk_5</td><td>PCLK / 32</td><td>baseclk_4 / 2</td></tr>
<tr><td>0110</td><td>baseclk_6</td><td>PCLK / 64</td><td>baseclk_4 / 4</td></tr>
<tr><td>0111</td><td>baseclk_7</td><td>PCLK / 128</td><td>baseclk_4 / 8</td></tr>
<tr><td>1000</td><td>baseclk_8</td><td>PCLK / 256</td><td>baseclk_4 / 16</td></tr>
<tr><td>1001</td><td>baseclk_9</td><td>PCLK / 512</td><td>baseclk_4 / 32</td></tr>
<tr><td>1010</td><td>baseclk_10</td><td>PCLK / 1024</td><td>baseclk_4 / 64</td></tr>
<tr><td>1011</td><td>baseclk_11</td><td>PCLK / 2048</td><td>baseclk_4 / 128</td></tr>
<tr><td>1100</td><td>baseclk_12</td><td>PCLK / 4096</td><td>baseclk_4 / 256</td></tr>
<tr><td>1101</td><td>baseclk_13</td><td>PCLK / 8192</td><td>baseclk_4 / 512</td></tr>
<tr><td>1110</td><td>baseclk_14</td><td>PCLK / 16384</td><td>baseclk_4 / 1024</td></tr>
<tr><td>1111</td><td>baseclk_15</td><td>PCLK / 32768</td><td>baseclk_4 / 2048</td></tr>
</table>
This register defines the frequency of a free running counter which generates Base Clock for controlling the related AC timings.
When switch to High Speed (HS) mode, the divisor will be switch to 0 by hardware automatically.</td></tr>
</table>

Offset: 08h	I2CC08: Master/Slave Transmit/Receive Byte Buffer Register	Init = 0x0A060000
Bit	**R/W**	**Description**
31:29	RO	**Transfer Data Direction** 000: IDLE 001: WAIT 100: MTX 101: MRX 110: STX 111: SRX
28	RO	**SDA_OE : SDA output enable**
27	RO	**SDA_O : SDA output value**
26	RO	**SCL_OE : SCL output enable**
25	RO	**SCL_O : SCL output value**
24:23	RO	**Transfer Mode Timing Stage for each State of I2CC08[22:19]** 00: T0 01: T1 10: T2 11: T3
22:19	RO	**Transfer Mode State Machine** 0000: IDLE 1000: MACTIVE 1001: MSTART 1010: MSTARTR 1011: MSTOP 1100: MTXD 1101: MRXACK 1110: MRXD 1111: MTXACK 0001: SWAIT 0100: SRXD 0101: STXACK 0110: STXD 0111: SRXACK 0011: RECOVER
18	RO	**Sampled SCL Line State : SCL input value**
17	RO	**Sampled SDA Line State : SDA input value**
16	RO	**Bus Busy Status** 0: Bus is Idle 1: Bus is Busy or not meets Idle Timing Requirement
15:8	RO	**Receive Byte Buffer** This register is valid when DMA Buffer is not enabled.
7:0	RW	**Transmit Byte Buffer** This register is valid when DMA Buffer is not enabled.

Offset: 0Ch	I2CC0C: Master/Slave Pool Buffer Control Register	Init = 0x0
Bit	**R/W**	**Description**
31:30	RO	**Reserved**

to next page

from previous page

29:24	RW	**Actual Received Pool Buffer Size** 0 = 0 bytes 1 = 1 byte .... 32 = 32 bytes Write command can clear this field to 0.
23:21	RO	**Reserved**
20:16	RW	**Receive Pool Buffer Size** 0 = 1 byte space 1 = 2 bytes space .... 31 = 32 bytes space This value defines the maximum receive buffer size for Slave mode, or receive command data byte count for Master mode.
15:13	RO	**Reserved**
12:8	RW	**Transmit Data Byte Count** 0 = 1 byte 1 = 2 bytes .... 31 = 32 bytes
7:1	RO	**Reserved**
0	RW	**Buffer Organization** 0: 32 bytes for Tx or Rx 1: Lower 16 bytes for Tx and higher 16 bytes for Rx

Note :
The buffer is shared by Pool buffer mode and DMA mode of Master and Slave controllers. So there are some limitations on using the buffer.
1. The buffer can only be used by Master or Slave controller, but not both at a time.
2. When Master or Slave enables DMA mode, then Pool buffer mode can not be used.

Offset: 10h **I2CM10: Master Interrupt Control Register** **Init = 0**

Bit	R/W	Description
31:17	RO	**Reserved (0)**
16	RW	**Enable Packet command done interrupt** When in packet command mode, it only needs to enable this bit and/or bit12. All other bits have no effect.
15	RO	**Reserved (0)**
14	RW	**Enable SDA data-low timeout interrupt**
13	RW	**Enable Bus Recover Done interrupt**
12	RW	**Enable SMBus Device Alert interrupt** Alert interrupt is not included in Packet command interrupt, it should be enabled individually.
11:7	RO	**Reserved (0)**
6	RW	**Enable SCL clock-low timeout interrupt**
5	RW	**Enable abnormal Start/Stop condition detection interrupt** This interrupt is used to indicate a bus Start/Stop condition has been detected at an illegal transfer state.

to next page

from previous page

4	RW	**Enable normal Stop condition detection interrupt** This interrupt is used to report that a Stop pattern has been issued.
3	RW	**Enable master arbitration loss interrupt**
2	RW	**Enable Receive Done interrupt** Receive Done means Byte, Pool or DMA mode Rx command end.
1	RW	**Enable Transmit ended with NACK response interrupt**
0	RW	**Enable Transmit ended with ACK response interrupt** Transmit end means Byte, Pool or DMA mode Tx command end.

Note :
The bit position follows the old register I2CD0C definition.
When packet operation is enabled, the interrupt will be issued only after packet operation was completed.
The definition of this register is :
0 : Disable
1 : Enable

Offset: 14h **I2CM14: Master Interrupt Status Register** **Init = 0**

Bit	R/W	Description
31:28	RO	**Packet Operation State Machine** 0000: IDLE 0001: STARTH 0010: STARTW 0011: STARTR 0100: TXMCODE 0101: TXAW 0110: TXAR 1000: INIT 1001: TXD 1010: RXD 1011: STOP 1100: RETRY 1101: FAIL 1110: WAIT 1111: PASS
27:19	RO	**Reserved**
18	RW1C	**(WC) Packet command timeout interrupt status**
17	RW1C	**(WC) Packet command fail interrupt status** The detailed failure status can be checked from bit1, bit3, bit5, bit6, bit14, bit15, bit18.
16	RW1C	**(WC) Packet command done interrupt status** When in packet mode, clear this bit will also clear bit[6:0], bit[15:13], bit[18:17].
15	RW	**(WC) Bus Recover Fail status**
14	RW	**(WC) SDA data-low timeout interrupt status**
13	RW	**(WC) Bus Recover Done interrupt status**
12	RW	**(WC) SMBus Device Alert interrupt status** Alert interrupt is independent to Packet command interrupt status.
11:7	RO	**Reserved**
6	RW	**(WC) SCL clock-low timeout interrupt status**
5	RW	**(WC) Abnormal Start/Stop Condition Detection interrupt status**
4	RW	**(WC) Normal Stop Condition Detection interrupt status**

to next page

from previous page

3	RW	**(WC) Master Arbitration Loss interrupt status**
2	RW	**(WC) Receive Done interrupt status** S/W needs to clear this status bit to enable next data receiving.
1	RW	**(WC) Transmit ended with NACK response interrupt status**
0	RW	**(WC) Transmit ended with ACK response interrupt status**

Note :
The bit position follows the old register I2CD10 definition.
'WC' means this bit is cleared by writing '1'.

Offset: 18h **I2CM18: Master Command/Status Register** **Init = 0**

Bit	R/W	Description
31	W1T	**Write '1' control for I2CM18 for current write command** When write to this register, if write data bit31 = 1, then it is write '1' possible only. This is useful to update this register without read-modify-write required.
30:24	RW	**Target device address** The target device address is used under packet operation mode.
23:20	RO	**Reserved (0)**
19:17	RW	**High Speed mode master code LSB** The I2C HS mode master code is 0b00001xxx. This register defines the LSB 3 bits. This field is only applied to Packet operation mode.
16	RW	**Enable master Packet operation mode** 0: stepping operation type 1: packet operation type When enabled, the master controller will execute all commands automatically and handle some basic error conditions. Including SDA lock recovery, fail retry (4 times), and issue interrupt after all commands were completed. Besides, target device address is required for packet operation mode.
15	RW	**SDA_OE output direct control** Bit[15:12] is a GPIO function and only work when both Master and Slave function are disabled. When bus lock occurs, this GPIO function can help to recover the bus life. 0: output disable, tri-stated 1: output enable
14	RW	**SDA_O output direct control**
13	RW	**SCL_OE output direct control** 0: output disable, tri-stated 1: output enable
12	RW	**SCL_O output direct control**
11	RW	**Enable Bus Recover Command** 0: NOP 1: Issue Bus Recover Command This command can be applied only when SCL line status is High. When Bus Recovery Command is fired, 1 ~ 8 SCL clock cycles will be issued to recover the bus. (Automatically stop issuing clock cycle whenever SDA has been released to high state) This command is used to recover the bus when a slave device locks SDA. When using this command, the Transfer Mode State Machine must be at the IDLE state; otherwise, it must be reset to the IDLE state firstly by disabling the device function (I2CC00). When recover operation has been done, the current bus state can be checked from I2CC08[17] or I2CM14[15].
10	RO	**Reserved (0)**

to next page

from previous page

9	RW	**Enable Master Rx DMA Data buffer**
8	RW	**Enable Master Tx DMA Data buffer**
7	RW	**Enable Master Rx Pool Data buffer**
6	RW	**Enable Master Tx Pool Data buffer** 0: Disable 1: Enable
5	RW	**Master Stop Command (P)** 0: NOP 1: Issue Master Stop Command **4th** priority.
4	RW	**Master Receive Command Last** 0: Respond ACK after last RX byte 1: Respond NACK after last RX byte If Stop command is set, then it will also respond NACK after the last Rx byte.
3	RW	**Master Receive Command (RxD)** 0: NOP 1: Fire Master Receive Command **3rd** priority.
2	RO	**Reserved (0)**
1	RW	**Master Transmit Command (TxD)** 0: NOP 1: Fire Master Transmit Command **2nd** priority.
0	RW	**Master Start Command (S/Sr)** 0: NOP 1: Issue Master Start/Repeated Start Command **1st** priority.

Note :
The bit position follows the old register I2CD14 definition.
For old stepping command mode, it will execute the commands according to the following sequence (priority):

S → TxD → RxD → P

If packet operation mode is enabled, then the command sequence would become:

S → Aw → TxD → Sr → Ar → RxD → P

HW will clear each command (bit[11:0]) automatically when it is finished. Beside, HW will clear all the commands whenever Master Arbitration Lost or invalid Start/Stop conditions have been detected.

Offset: 1Ch **I2CM1C: Master DMA Transfer Length Register** **Init = 0**

Bit	R/W	Description
31	W1T	**DMA Rx buffer length write enable for current write command** Define the write enable is allowing firmware to program Tx and Rx DMA length separately without need to do read-modify-write. If both write enable control bit15 and bit31 are 0, then both Tx and Rx DMA length will be written.
27:16	RW	**DMA Rx buffer length (Byte)** Define the length (1 ~ 4096 bytes) for receiving. 0 = 1 byte .... 4095 = 4096 bytes

to next page

from previous page

15	W1T	**DMA Tx buffer length write enable for current write command** Define the write enable is allowing firmware to program Tx and Rx DMA length separately without need to do read-modify-write. If both write enable control bit15 and bit31 are 0, then both Tx and Rx DMA length will be written.
11:0	RW	**DMA Tx buffer length (Byte)** Define the length (1 ~ 4096 bytes) for transmitting. 0 = 1 byte .... 4095 = 4096 bytes

Offset: 20h **I2CS20: Slave Interrupt Control Register** **Init = 0**

Bit	R/W	Description
31:18	RO	**Reserved (0)**
17	RW	**Enable Slave address matched but NACKed interrupt** This interrupt is valid only when packet operation mode and auto NACK enabled. The NACK status can be checked from I2CS24[22:20]. There are 2 NACK conditions: 1. I2CS28[14] = 1 ⇒ Hit to active address but has no buffer ready. This condition also will trigger packet done interrupt. 2. I2CS28[15] = 1 ⇒ Hit to enabled but non-active address.
16	RW	**Enable Packet command done interrupt** When in packet command mode, it only needs to enable this bit and/or bit17. All other bits have no effect.
15	RW	**Enable Slave mode inactive timeout interrupt** Alert for the state machine stays at Slave active mode for a long time (timeout counter). This maybe the condition that controller work at slave mode, but suddenly the master stop to transfer, and this makes the state machine stuck at slave mode and can not quit.
14:6	RO	**Reserved (0)**
5	RW	**Enable abnormal Start/Stop condition detection interrupt** This interrupt is used to indicate a bus Start/Stop condition has been detected at an illegal transfer state.
4	RW	**Enable normal Stop condition detection interrupt** This interrupt is used to report a Stop pattern has been detected during receive data cycle, or a NACK was received during transmit data cycle.
3	RO	**Reserved (0)**
2	RW	**Enable Receive Done interrupt** Receive Done means the buffer is full, or salve controller receives a termination signal.
1	RO	**Reserved (0)**
0	RW	**Enable Transmit ended with ACK response interrupt** Transmit end means Byte, Pool or DMA mode Tx command end.

Note :
The bit position follows the old register I2CD0C definition.
When packet operation is enabled, the interrupt will be issued only after packet operation was completed.
The definition of this register is :
0 : Disable
1 : Enable

Offset: 24h	I2CS24: Slave Interrupt Status Register	Init = 0

Bit	R/W	Description
31:30	RO	**The Last Slave address match indicator** 00: address1 01: address2 10: address3
29	RO	**Slave Address Received Pending** 0: none 1: A slave address match received interrupt is pending due to previous slave DMA receive done is not processed.
28:26	RO	**Reserved**
25:24	RO	**Current Slave parking status** 00: Idle 01: Wait for RX buffer 10: Wait for Tx buffer 11: undefined
23	RO	**Reserved**
22	RW	**(WC) Slave address3 NACKed indicator**
21	RW	**(WC) Slave address2 NACKed indicator**
20	RW	**(WC) Slave address1 NACKed indicator**
19:18	RO	**The current active Slave address indicator** 00: address1 01: address2 10: address3 This field is valid only after at least 1 byte of data was received or transmitted, not including the slave address byte.
17	RW	**(WC) Packet command fail interrupt status** The detailed failure status can be checked from bit5, bit15.
16	RW	**(WC) Packet command done interrupt status** When in packet mode, clear this bit will also clear bit0, bit2, bit[5:4], bit7, bit15, bit17.
15	RW	**(WC) Slave mode inactive timeout interrupt status**
14:8	RO	**Reserved**
7	RW	**(WC) Slave Address Received Match interrupt status**
6	RO	**Reserved**
5	RW	**(WC) Abnormal Start/Stop Condition Detection interrupt status**
4	RW	**(WC) Normal Stop Condition Detection interrupt status** Normal Stop includes below conditions: 1. Received Stop during Slave RX cycle 2. Received NACK during Slave TX cycle 3. Respond NACK at end of Slave RX cycle
3	RO	**Receive Done ended with returning NACK** This bit is cleared at the same time while clearing bit4.
2	RW	**(WC) Receive Done interrupt status**
1	RW	**(WC) Transmit ended with NACK response interrupt status** In general, slave transmit cycle responded by NACK indicates transfer end. So bit4 Normal Stop status also will be raised.
0	RW	**(WC) Transmit ended with ACK response interrupt status**

to next page

from previous page

Note :
The bit position follows the old register I2CD10 definition.
'WC' means this bit is cleared by writing '1'.
SW should clear bit[2:0] after all new commands are set ready, it is a fire control for new commands.

Offset: 28h	I2CS28: Slave Command/Status Register	Init = 0

Bit	R/W	Description
31	W1T	**Write '1' control for I2CS28 for current command** When write to this register, if write data bit31 = 1, then it is write '1' possible only. This is useful to update this register without read-modify-write required.
30:20	RO	**Reserved (0)**
18:17	RW	**Active address selection for Packet operation** 00: address1 01: address2 10: address3 11: all enabled slave addresses
16	RW	**Enable slave Packet operation mode** When enabled, the slave controller will execute all commands automatically and handle some basic error conditions. Including timeout detection, and issue interrupt after all commands were completed. Besides, byte mode will be disabled under packet operation mode.
15	RW	**Enable Auto NACK to Non-Active address**
14	RW	**Enable Auto NACK to Active address** The auto NACK function only works under packet operation mode. When enabled, slave controller respond with NACK at address phase if address isn't active or buffer isn't ready. This bit is suggested to be enabled when the active address is not for "all". All slave addresses will be responded with ACK if autoNACK is not enabled.
13:11	RO	**Reserved (0)**
10	RW	**Enable issuing I2C/SMBus Slave Alert signal** 0: NOP 1: Issuing Alert signal to a bus Master This command is valid only when slave function is enabled. This bit will be cleared by hardware automatically after packet received with slave address matched.
9	RW	**Enable Slave Rx DMA Data buffer**
8	RW	**Enable Slave Tx DMA Data buffer**
7	RW	**Enable Slave Rx Pool Data buffer**
6	RW	**Enable Slave Tx Pool Data buffer** 0: Disable 1: Enable Since Packet mode don't support byte mode, so it can only set bit[9:6] for packet mode Tx/Rx, bit2 and bit4 are not used by packet mode.
4	RW	**Slave Receive Command Last** 0: Receive Command can be continued by responding ACK 1: Receive Command will be ended by responding NACK When in Pool/DMA mode, the NACK will be sent after the latest byte is received.

to next page

from previous page

2	RW	**Slave Transmit Command** 0: NOP 1: Fire Slave Transmit Command HW will clear this register when TX buffer is empty or bus contention error has been detected.
1:0	RO	**Reserved (0)**

Note :
The bit position follows the old register I2CD10 definition.
For old stepping command mode, it will execute the commands according to the following sequence (priority):
TxD → RxD(bit2=0)
If packet operation mode is enabled, then the command sequence is based on the $R/\overline{W}$ bit of the address byte.
HW will clear each command (bit[11:0]) automatically when it has been finished.

Offset: 2Ch **I2CS2C: Slave DMA Transfer Length Register** **Init = 0**

Bit	R/W	Description
31	W1T	**DMA Rx buffer length write enable for current command** Define the write enable is allowing firmware to program Tx and Rx DMA length separately without need to do read-modify-write. If both write enable control bit15 and bit31 are 0, then both Tx and Rx DMA length will be written.
27:16	RW	**DMA Rx buffer length (Byte)** Define the maximum buffer size (1 ∼ 4096 bytes) for receiving.
15	W1T	**DMA Tx buffer length write enable for current command** Define the write enable is allowing firmware to program Tx and Rx DMA length separately without need to do read-modify-write. If both write enable control bit15 and bit31 are 0, then both Tx and Rx DMA length will be written.
11:0	RW	**DMA Tx buffer length (Byte)** Define the length (1 ∼ 4096 bytes)for transmitting.

Offset: 30h **I2CM30: Master DMA Mode Tx Buffer Base Address** **Init = X**
Offset: 34h **I2CM34: Master DMA Mode Rx Buffer Base Address** **Init = X**
Offset: 38h **I2CS38: Slave DMA Mode Tx Buffer Base Address** **Init = X**
Offset: 3Ch **I2CS3C: Slave DMA Mode Rx Buffer Base Address** **Init = X**

Bit	Attr.	Description
31	RO	**Reserved (0x1)**
30:0	RW	**SDRAM DMA Buffer Base Address**

Offset: 40h **I2CS40: Slave Device Address Register** **Init = 0**

Bit	R/W	Description
31:24	RO	**Reserved (0)**
23	RW	**Enable Slave Device Address3** 0: disable 1: enable
22:16	RW	**Slave Device Address3**
15	RW	**Enable Slave Device Address2** 0: disable 1: enable

to next page

from previous page

14:8	RW	**Slave Device Address2**
7	RW	**Enable Slave Device Address1 (only for New Register mode)** 0: disable 1: enable
6:0	RW	**Slave Device Address1**

Offset: 48h **I2CM48: Master DMA Length Status Register** **Init = 0**

Bit	R/W	Description
28:16	RW	**DMA Rx actual length (Byte)**
12:0	RW	**DMA Tx actual length (Byte)** 0 = 0 byte 1 = 1 byte .... Write operation clear this register to 0.
Note : The DMA length status will be cleared under below conditions: 1. write I2CM48 2. Master Packet Operation Start 3. DMA operation start will clear TX or RX respectively		

Offset: 4Ch **I2CS4C: Slave DMA Length Status Register** **Init = 0**

Bit	R/W	Description
28:16	RW	**DMA Rx actual length (Byte)**
12:0	RW	**DMA Tx actual length (Byte)** 0 = 0 byte 1 = 1 byte .... Write operation clear this register to 0.
Note : The DMA length status will be cleared under below conditions: 1. write I2CS4C 2. DMA operation start will clear TX or RX respectively		

Offset: 50h **I2CC50: Current DMA Operating Address Status** **Init = 0**

Bit	R/W	Description
31:0	RO	**Current DMA Operating Address Counter** Increment counter

Offset: 54h **I2CC54: Current DMA Operating Length Status** **Init = 0**

Bit	R/W	Description
11:0	RO	**Current DMA Operating Length Counter** Decrement counter

53.5.5 Secure I2C Global Register Definition

Offset: 00h I2CSG00: Device Master Mode Interrupt Status Register (I2CSG0C[3] = 1) Init = 0
Offset: 00h I2CSG00: Device Master/Slave Mode Interrupt Status Register (I2CSG0C[3] = 0) Init = 0

Bit	Attr.	Description
31:4	RO	**Reserved (0)**
3	RO	**I2C/SMBus Device #4 Interrupt**
2	RO	**I2C/SMBus Device #3 Interrupt**
1	RO	**I2C/SMBus Device #2 Interrupt**
0	RO	**I2C/SMBus Device #1 Interrupt** 0 : No interrupt 1 : Interrupt occurs

Note :
This global register shows the summary report of the interrupt events from all of the **16** devices. There is no need to clear the interrupt status of this register.

Offset: 04h I2CSG04: Device Slave Mode Interrupt Status Register Init = 0

Bit	R/W	Description
31:4	RO	**Reserved (0)**
3	RO	**I2C/SMBus Device #4 Interrupt**
2	RO	**I2C/SMBus Device #3 Interrupt**
1	RO	**I2C/SMBus Device #2 Interrupt**
0	RO	**I2C/SMBus Device #1 Interrupt** 0 : No interrupt 1 : Interrupt occurs

Note :
This global register shows the summary report of the interrupt events from all of the **16** devices. There is no need to clear the interrupt status of this register.

Offset: 0Ch I2CSG0C: Global Control Register Init = 0

Bit	R/W	Description
31:12	RO	**Reserved (0)**
11:8	RW	**Master transmit to receive turn-around delay** 0000: no delay 0001: delay 1 cycle of Base Clock 0010: delay 2 cycles of Base Clock .... 1111: delay 15 cycles of Base Clock This timing is only applied on master packet operation mode.
7:5	RO	**Reserved (0)**
4	RW	**Select the action when Slave packet mode RX buffer empty** 0: Issue packet done interrupt and pull SCL low to halt bus. (same as non-packet mode) 1: Expect to receive Repeat Start or Stop command, if more SRX data input then it will respond with NACK. Here the RX buffer empty means the received data bytes equals to the allocated buffer size. In non-packet operation mode, slave controller will stop reciving data and pull SCL low to halt bus, and then issue interrupt for this condition. In packet operation mode, it has another option to continue the operation when buffer empty. It expect to receive Repeat Start or Stop command after RX buffer empty, and will respond with NACK if the succeeding command is still RX.

to next page

from previous page

3	RW	**Enable separating master and slave mode interrupts** 0: Master and slave mode interrupts are merged on I2CSG00 1: Master interrupts is on I2CSG00, and slave interrupts is on I2CSG04
2	RW	**Register definition selection** 0: Old registers definition 1: New registers definition
1	RW	**Clock divider mode selection** 0: Old clock divider mode 1: New clock divider mode
0	RW	**Reserved (0)**

Offset: 10h **I2CSG10: New Clock Divider Control Register (I2CSG0C[1] = 1)** **Init = 0**

Bit	R/W	Description
31:24	RW	**Base clock 4 divisor, base_divider_4**
23:16	RW	**Base clock 3 divisor, base_divider_3**
15:8	RW	**Base clock 2 divisor, base_divider_2**
7:0	RW	**Base clock 1 divisor, base_divider_1** 0x00: divided by 1 0x01: divided by 1.5 0x02: divided by 2 0x03: divided by 2.5 .... 0xFE: divided by 128 0xFF: divided by 128.5

53.5.6 Old Register Mode Definition

Offset: 00h **I2CSD00: Function Control Register** **Init = 0x0**

Bit	R/W	Description
31:18	RO	**Reserved (0)**
17	RW	**Enable bus auto-release when SCL low, SDA low, or slave mode inactive timeout** 0: Disable 1: Enable The timeout timing setting is defined at I2CSD08. To enable this feature, at least one bit of I2CSD0C[6] or I2CSD0C[15:14] must be enabled. This feature is valid only when none of master or slave Packet operation mode were enabled.
16	RW	**Enable master auto SDA lock recovery capability (for single master case only)** 0: Disable 1: Enable When enabled, master will auto generate clock to recover SDA lock condition, if SDA was locked before issue START pattern command.
15	RW	**Disable multi-master capability (for master function only)** 0: Enable 1: Disable (for single master application only) When disabled, device controller assume that there will have no arbitration lost possibility and ignore the clock stretch check.

to next page

from previous page

14	RW	**Enable SCL direct drive mode (for master function only)** 0: Disable (SCL output buffer is configured as an open-drain buffer with an external pull-up resistor) 1: Enable (SCL always drives output buffer, no tri-state) This is a special mode, it can be used only under single master condition and no clock stretching devices allowed on this bus. Slave mode can not be enabled when use this mode.
13:10	RO	**Reserved (0)**
9	RW	**Enable slave address range mode** 0: normal address mode for address1 and address2 1: address1 $\leq$ slave address $\leq$ address2 When select range mode, the address matching indicator is assigned to address1.
8	RW	**Enable SDA signal to direct drive high for 1T** 0: Disable (SDA signal passively pulls high by an external pull-up resistor) 1: Enable (SDA signal actively drives high by current source for 1 PCLK cycle before entering tri-state) The function is designed to support a higher transfer rate.
7	RW	**Enable SCL signal to direct drive high for 1T (Master Only)** 0: Disable (SCL signal passively pulls high by an external pull-up resistor) 1: Enable (SCL signal actively drives high by current source for 1 PCLK cycle before entering tri-state) The function is designed to support a higher transfer rate. If high speed mode enabled, then it will drive SCL until it is really high and then release. Not limited by 1T. So no clock stretch allowed.
6	RW	**Enable High Speed master mode** 0 : normal speed mode 1 : high speed mode (3.4Mbps) High speed mode can only use buffer or DMA mode for transfer. And only master mode supports speed switching capability. At high speed mode, the clock rate is determined by I2CSD04[19:12] with baseclk_0.
5	RW	~~Enable I2C/SMBus to respond the Default Address (1100_001)~~
4	RW	~~Enable I2C/SMBus to respond the Alert Address (0001_100)~~
3	RW	~~Enable I2C/SMBus to respond the ARP Host Address (0001_000)~~
2	RW	~~Enable I2C/SMBus to respond the General Call Address (0000_0000)~~
1	RW	**Enable slave function** 0 : Disable 1 : Enable Each device can be enabled to support either slave or master function or both functions.
0	RW	**Enable master function** 0 : Disable 1 : Enable Each device can be enabled to support either slave or master function or both functions.

Note :
The respective controller and the following registers of the respective controller will be reset whenever its master function and slave function are both disabled simultaneously.
1. I2CSD10, I2CSM14, I2CSS24 : Interrupt Status Register
2. I2CSD14, I2CSM18, I2CSS28 : Command Control Register

Offset: 04h **I2CSD04: Clock and AC Timing Control Register #1** **Init = 0x0**

Bit	R/W	Description
31:24	RO	**Reserved (0)**
23:20	RO	**Reserved** It returns the value of I2CD04[19:17].
19:16	RW	**Cycles of master SCL clock-high pulse width (tCKHigh)** 000x: no guarantee pulse width 0011: 4 cycles of Base Clock 0100: 5 cycles of Base Clock .... 1111: 16 cycles of Base Clock
15:12	RW	**Cycles of master SCL clock-low pulse width (tCKLow)** 000x: no guarantee pulse width 0011: 4 cycles of Base Clock 0100: 5 cycles of Base Clock .... 1111: 16 cycles of Base Clock
11:10	RW	**Hold time of master/slave data (tHDDAT)** Master Slave 00 1 0 01 2 1 10 3 2 11 4 3 The unit for the table is numbers of Base Clock
9:8	RW	**Timeout base clock divisor (toutBaseClk)** | | Clock name | Old divider mode | New divider mode | | 00 | tout_baseclk_0 | PCLK / 16384 | baseclk_4 / 256 | | 01 | tout_baseclk_1 | PCLK / 65536 | baseclk_4 / 1024 | | 10 | tout_baseclk_2 | PCLK / 262144 | baseclk_4 / 4096 | | 11 | tout_baseclk_3 | PCLK / 1048576 | baseclk_4 / 8192 | This divisor defines the Base Clock for Timeout Counter.
7:4	RW	**Reserved**

to next page

from previous page

3:0	RW	**Base Clock divisor (tBaseClk)** The divisor defines the frequency of Base Clock which is divided from PCLK clock.

	Clock name	Old divider mode	New divider mode
0000	baseclk_0	PCLK / 1	PCLK / 1
0001	baseclk_1	PCLK / 2	PCLK / base_divider_1
0010	baseclk_2	PCLK / 4	PCLK / base_divider_2
0011	baseclk_3	PCLK / 8	PCLK / base_divider_3
0100	baseclk_4	PCLK / 16	PCLK / base_divider_4 (1 MHz)
0101	baseclk_5	PCLK / 32	baseclk_4 / 2
0110	baseclk_6	PCLK / 64	baseclk_4 / 4
0111	baseclk_7	PCLK / 128	baseclk_4 / 8
1000	baseclk_8	PCLK / 256	baseclk_4 / 16
1001	baseclk_9	PCLK / 512	baseclk_4 / 32
1010	baseclk_10	PCLK / 1024	baseclk_4 / 64
1011	baseclk_11	PCLK / 2048	baseclk_4 / 128
1100	baseclk_12	PCLK / 4096	baseclk_4 / 256
1101	baseclk_13	PCLK / 8192	baseclk_4 / 512
1110	baseclk_14	PCLK / 16384	baseclk_4 / 1024
1111	baseclk_15	PCLK / 32768	baseclk_4 / 2048

This register defines the frequency of a free running counter which generates Base Clock for controlling the related AC timings.
When switch to High Speed (HS) mode, the divisor will be switch to 0 by hardware automatically.

Offset: 08h **I2CSD08: Clock and AC Timing Control Register #2** **Init = 0x0**

Bit	R/W	Description
31:5	RO	**Reserved**
4:0	RW	**Timeout timer for SCL low, SDA low, and slave active timeout (tTimeOut)** 0: No Timeout Control 1: 1 cycle of Timeout Base Clock 2: 2 cycles of Timeout Base Clock .... 31: 31 cycles of Timeout Base Clock Timeout Period = tTimeOut*(1/toutBaseClk) = tTimeOut*div / PCLK ex: Timeout Period = 6* 16384/PCLK = 0.983ms when PCLK = 100MHZ, div=16384 from I2CSD04[9:8]

Offset: 0Ch **I2CSD0C: Interrupt Control Register** **Init = 0x0**

Bit	R/W	Description
31:16	RO	**Reserved (0)**
15	RW	**Enable Slave mode inactive timeout interrupt** Alert for the state machine stays at Slave active mode for a long time (timeout counter). This maybe the condition that controller work slave mode, but suddenly the master stop to transfer, and this makes the state machine stuck at slave mode and can not quit.

to next page

from previous page

14	RW	**Enable SDA data-low timeout interrupt**
13	RW	**Enable Bus Recover Done interrupt**
12	RW	**Enable SMBus Device Alert interrupt**
11:7	RO	**Reserved (0)**
6	RW	**Enable SCL clock-low timeout interrupt**
5	RW	**Enable abnormal Start/Stop condition detection interrupt** This interrupt is used to indicate a bus Start/Stop condition has been detected at an illegal transfer state.
4	RW	**Enable normal Stop condition detection interrupt** For master mode, this interrupt is used to report that a Stop pattern has been issued. For salve mode, this interrupt is used to report a Stop pattern has been detected
3	RW	**Enable master arbitration loss interrupt**
2	RW	**Enable Receive Done interrupt** Receive Done means : 1. Master : all the expected bytes have been received. 2. Slave : the buffer is full, or salve controller receives a termination signal.
1	RW	**Enable Transmit ended with NACK response interrupt**
0	RW	**Enable Transmit ended with ACK response interrupt** Transmit end means Byte, Pool or DMA mode Tx command end.
Note : The definition of this register is : 0 : Disable 1 : Enable		

Offset: 10h **I2CSD10: Interrupt Status Register** **Init = 0x0**

Bit	R/W	Description
31:30	RO	**Slave address match indicator** 00: address1 01: address2 10: address3
29	RO	**Slave Address Received Pending** 0: none 1: A slave address match received interrupt is pending due to previous slave DMA receive done is not processed.
28:16	RO	**Refer to I2CSS24[28:16]**
15	RW1C	**(WC) Slave mode inactive timeout interrupt status**
14	RW1C	**(WC) SDA data-low timeout interrupt status**
13	RW1C	**(WC) Bus Recover Done interrupt status**
12	RW1C	**(WC) SMBus Device Alert interrupt status**
11	RO	~~SMBus ARP Host Address Detection interrupt status~~
10	RO	~~SMBus Device Alert Response Address Detection interrupt status~~
9	RO	~~SMBus Device Default Address Detection interrupt status~~
8	RO	~~General Call Address Detection interrupt status~~
7	RW1C	**(WC) Slave Address Received Match interrupt status**
6	RW1C	**(WC) SCL clock-low timeout interrupt status**
5	RW1C	**(WC) Abnormal Start/Stop Condition Detection interrupt status**

to next page

from previous page

4	RW1C	**(WC) Normal Stop Condition Detection interrupt status**
3	RW1C	**(WC) Master Arbitration Loss interrupt status**
2	RW1C	**(WC) Receive Done interrupt status** S/W needs to clear this status bit to enable next data receiving. And at (slave) byte buffer mode, this interrupt status may be set concurrently along with bit[11:7].
1	RW1C	**(WC) Transmit ended with NACK response interrupt status**
0	RW1C	**(WC) Transmit ended with ACK response interrupt status**
Note : 'WC' means this bit is cleared by writing '1'.		

Offset: 14h **I2CSD14: Command/Status Register** **Init = 0x0A060000**

Bit	R/W	Description
31:29	RO	**Transfer Data Direction** 000: IDLE 001: WAIT 100: MTX 101: MRX 110: STX 111: SRX
28	RO	**SDA_OE**
27	RO	**SDA_O**
26	RO	**SCL_OE**
25	RO	**SCL_O**
24:23	RO	**Transfer Mode Timing Stage** 00: T0 01: T1 10: T2 11: T3
22:19	RO	**Transfer Mode State Machine** 0000: IDLE 1000: MACTIVE 1001: MSTART 1010: MSTARTR 1011: MSTOP 1100: MTXD 1101: MRXACK 1110: MRXD 1111: MTXACK 0001: SWAIT 0100: SRXD 0101: STXACK 0110: STXD 0111: SRXACK 0011: RECOVER
18	RO	**Sampled SCL Line State**
17	RO	**Sampled SDA Line State**
16	RO	**Bus Busy Status** 0: Bus is Idle 1: Bus is Busy or not meets Idle Timing Requirement

to next page

from previous page

15	RW	**SDA_OE output direct control** Bit[15:12] is a GPIO function and only work when both Master and Slave function are disabled. When bus lock occurs, this GPIO function can help to recover the bus life. 0: output disable, tri-stated 1: output enable
14	RW	**SDA_O output direct control**
13	RW	**SCL_OE output direct control** 0: output disable, tri-stated 1: output enable
12	RW	**SCL_O output direct control**
11	RW	**Enable Bus Recover Command** 0: NOP 1: Issue Bus Recover Command This command can be applied only when SCL line status is High. When Bus Recovery Command is fired, 1 ∼ 8 SCL clock cycles will be issued to recover the bus. (Automatically stop issuing clock cycle whenever SDA has been released to high state) This command is used to recover the bus when a slave device locks SDA. When using this command, the Transfer Mode State Machine must be at the IDLE state; otherwise, it must be reset to the IDLE state firstly by disabling the device function (I2CSD00). When recover operation has been done, the current bus state can be read back from Bit [17] of this register.
10	RW	**Enable issuing I2C/SMBus Slave Alert signal** 0: NOP 1: Issuing Alert signal to a bus Master This command is valid only when slave function is enabled. This bit will be cleared by hardware automatically after packet received with slave address matched.
9	RW	**Enable Master/Slave Receive Data DMA Mode** 0: Disable 1: Enable HW will clear this register automatically when data receiving has been done. DMA transmit and receive can not be enabled at the same time. The result is unpredictable.
8	RW	**Enable Master/Slave Transmit Data DMA Mode** 0: Disable 1: Enable HW will clear this register automatically when data transmitting has been done. When set, DMA will start to fetch memory data, no matter I2C is start to active or not. So all the DMA related control registers must be set before enable this bit.
7	RW	**Enable Master/Slave Receive Data Buffer** 0: Disable 1: Enable This register will be automatically cleared by H/W when data receiving has been done.
6	RW	**Enable Master/Slave Transmit Data Buffer** 0: Disable 1: Enable This register will be automatically cleared by H/W when data receiving has been done.

to next page

from previous page

5	RW	**Master Stop Command** 0: NOP 1: Issue Master Stop Command **4th** priority. This register will be automatically cleared by H/W when Stop Command has been issued. This command is valid only when master mode is enabled
4	RW	**Master/Slave Receive Command Last** 0: Receive Command can be continued by responding ACK 1: Receive Command will be ended by responding NACK When in buffer mode, the last control will acts after the latest byte is been received. When in Master mode and Stop Command activated, the last control must be set to ending transfer.
3	RW	**Master Receive Command** 0: NOP 1: Fire Master Receive Command **3rd** priority. HW will clear this register when RX buffer is full or receiving is terminated (Stop/Repeated Start). This command is valid only when Master mode is enabled
2	RW	**Slave Transmit Command** 0: NOP 1: Fire Slave Transmit Command HW will clear this register when TX buffer is empty or bus contention error has been detected.
1	RW	**Master Transmit Command** 0: NOP 1: Fire Master Transmit Command **2nd** priority. HW will clear this register when TX buffer is empty or Bus Contention error has been detected.
0	RW	**Master Start Command** 0: NOP 1: Issue Master Start/Repeated Start Command **1st** priority. This register will be automatically cleared by HW when Master Start Command or Repeated Start Command has been issued. This command will be executed by HW only when master mode is enabled and the bus is in idle state.

Note :
When multiple commands in this register are fired simultaneously, Device controller will execute these commands according to the following sequence (priority):

(1) Master Start Command
(2) Master Transmit Command
(3) Master Receive Command
(4) Master Stop Command

HW will automatically clear each command (bit[11:0]) when it has been finished. Beside, HW will clear all the commands whenever Master Arbitration Lost or invalid Start/Stop conditions have been detected.
Attention: Master and Slave Command cannot be activated at the same time.

Offset: 18h	I2CSD18: Slave Device Address Register	Init = 0x80

Bit	R/W	Description
31:24	RO	**Reserved (0)**
23	RW	**Enable Slave Device Address3** 0: disable 1: enable
22:16	RW	**Slave Device Address3**
15	RW	**Enable Slave Device Address2** 0: disable 1: enable
14:8	RW	**Slave Device Address2**
7	RW	**Enable Slave Device Address1 (only for New Register mode)** 0: disable 1: enable
6:0	RW	**Slave Device Address1**

Offset: 1Ch	I2CSD1C: Pool Buffer Control Register	Init = 0x0

Bit	R/W	Description
31:30	RO	**Reserved**
29:24	RW	**Actual Received Pool Buffer Size** 0 = 0 bytes 1 = 1 byte .... 32 = 32 bytes Write command can clear this field to 0.
23:21	RO	**Reserved**
20:16	RW	**Receive Pool Buffer Size** 0 = 1 byte space 1 = 2 bytes space .... 31 = 32 bytes space This value defines the maximum receive buffer size for Slave mode, or receive command data byte count for Master mode.
15:13	RO	**Reserved**
12:8	RW	**Transmit Data Byte Count** 0 = 1 byte 1 = 2 bytes .... 31 = 32 bytes
7:1	RO	**Reserved**
0	RW	**Buffer Organization** 0: 32 bytes for Tx or Rx 1: Lower 16 bytes for Tx and higher 16 bytes for Rx

Note :
The buffer is shared by Pool buffer mode and DMA mode of Master and Slave controllers. So there are some limitations on using the buffer.
1. The buffer can only be used by Master or Slave controller, but not both at a time.
2. When Master or Slave enables DMA mode, then Pool buffer mode can not be used.

Offset: 20h	I2CSD20: Transmit/Receive Byte Buffer Register	Init = 0x0
Bit	**R/W**	**Description**
31:16	RO	**Reserved**
15:8	RO	**Receive Byte Buffer** This register is valid when DMA Buffer is not enabled.
7:0	RW	**Transmit Byte Buffer** This register is valid when DMA Buffer is not enabled.

53.5.7 New Register Mode Definition

Offset: 00h	I2CSC00: Master/Slave Function Control Register	Init = 0x0
Bit	**R/W**	**Description**
31:22	RO	**Reserved (0)**
21	RW	**DisableDMA AutoRecovery when residual Tx Count equals 0**
20	RW	**Enable save address byte into buffer for Slave Packet mode receive command** 0: No save address byte 1: Save address byte For Slave Packet operation mode, the slave address byte can be chosen to be saved into buffer or not. The slave address byte is default saved under stepping command mode.
19:18	RW	**Master packet operation retry count** 00: No retry 01: Retry 1 time 10: Retry 2 times 11: Retry 3 times
17:0	RW	**Same as I2CSD00[17:0]**

Offset: 04h	I2CSC04: Master/Slave Clock and AC Timing Control Register #1	Init = 0x0
Bit	**R/W**	**Description**
31:29	RO	**Reserved (0)**
28:24	RW	**Same as I2CSD08[4:0]**
23:20	RW	**Cycles of master SCL clock-high minimum pulse width (tCKHighMin)** 000x: no guarantee pulse width 0011: 4 cycles of Base Clock 0100: 5 cycles of Base Clock .... 1111: 16 cycles of Base Clock This register is used to guarantee the minimum SCL high pulse width by sampling the feedback SCL input. It should be set to be less than or equal to tCKHigh. The actual SCL high pulse width is determined by tCKHigh and tCKHighMin. This timing will lengthen the SCL high duty width, which is depending on the SCL rise time.
19:0	RW	**Same as I2CSD04[19:0]**

Offset: 08h	I2CSC08: Master/Slave Transmit/Receive Byte Buffer Register	Init = 0x0A060000
Bit	**R/W**	**Description**
31:16	RO	**Same as I2CSD14[31:16]**
15:0	RW	**Same as I2CSD20[15:0]**

Offset: 0Ch	I2CSC0C: Master/Slave Pool Buffer Control Register	Init = 0x0
Bit	**R/W**	**Description**
31:0	RW	**Same as I2CSD1C**

Offset: 10h	I2CSM10: Master Interrupt Control Register	Init = 0
Bit	**R/W**	**Description**
31:17	RO	**Reserved (0)**
16	RW	**Enable Packet command done interrupt** When in packet command mode, it only needs to enable this bit and/or bit12. All other bits have no effect.
15	RO	**Reserved (0)**
14	RW	**Enable SDA data-low timeout interrupt**
13	RW	**Enable Bus Recover Done interrupt**
12	RW	**Enable SMBus Device Alert interrupt** Alert interrupt is not included in Packet command interrupt, it should be enabled individually.
11:7	RO	**Reserved (0)**
6	RW	**Enable SCL clock-low timeout interrupt**
5	RW	**Enable abnormal Start/Stop condition detection interrupt** This interrupt is used to indicate a bus Start/Stop condition has been detected at an illegal transfer state.
4	RW	**Enable normal Stop condition detection interrupt** This interrupt is used to report that a Stop pattern has been issued.
3	RW	**Enable master arbitration loss interrupt**
2	RW	**Enable Receive Done interrupt** Receive Done means Byte, Pool or DMA mode Rx command end.
1	RW	**Enable Transmit ended with NACK response interrupt**
0	RW	**Enable Transmit ended with ACK response interrupt** Transmit end means Byte, Pool or DMA mode Tx command end.

Note :
The bit position follows the old register I2CSD0C definition.
When packet operation is enabled, the interrupt will be issued only after packet operation was completed.
The definition of this register is :
0 : Disable
1 : Enable

Offset: 14h	I2CSM14: Master Interrupt Status Register	Init = 0

Bit	R/W	Description
31:28	RO	**Packet Operation State Machine** 0000: IDLE 0001: STARTH 0010: STARTW 0011: STARTR 0100: TXMCODE 0101: TXAW 0110: TXAR 1000: INIT 1001: TXD 1010: RXD 1011: STOP 1100: RETRY 1101: FAIL 1110: WAIT 1111: PASS
27:19	RO	**Reserved**
18	RW1C	**(WC) Packet command timeout interrupt status**
17	RW1C	**(WC) Packet command fail interrupt status** The detailed failure status can be checked from bit1, bit3, bit5, bit6, bit14, bit15, bit18.
16	RW1C	**(WC) Packet command done interrupt status** When in packet mode, clear this bit will also clear bit[6:0], bit[15:13], bit[18:17].
15	RW	**(WC) Bus Recover Fail status**
14	RW	**(WC) SDA data-low timeout interrupt status**
13	RW	**(WC) Bus Recover Done interrupt status**
12	RW	**(WC) SMBus Device Alert interrupt status** Alert interrupt is independent to Packet command interrupt status.
11:7	RO	**Reserved**
6	RW	**(WC) SCL clock-low timeout interrupt status**
5	RW	**(WC) Abnormal Start/Stop Condition Detection interrupt status**
4	RW	**(WC) Normal Stop Condition Detection interrupt status**
3	RW	**(WC) Master Arbitration Loss interrupt status**
2	RW	**(WC) Receive Done interrupt status** S/W needs to clear this status bit to enable next data receiving.
1	RW	**(WC) Transmit ended with NACK response interrupt status**
0	RW	**(WC) Transmit ended with ACK response interrupt status**

Note :
The bit position follows the old register I2CSD10 definition.
'WC' means this bit is cleared by writing '1'.

Offset: 18h	I2CSM18: Master Command/Status Register	Init = 0

Bit	R/W	Description
31	W1T	**Write '1' control for I2CSM18 for current write command** When write to this register, if write data bit31 = 1, then it is write '1' possible only. This is useful to update this register without read-modify-write required.
30:24	RW	**Target device address** The target device address is used under packet operation mode.

to next page

from previous page

23:20	RO	**Reserved (0)**
19:17	RW	**High Speed mode master code LSB** The I2C HS mode master code is 0b00001xxx. This register defines the LSB 3 bits. This field is only applied to Packet operation mode.
16	RW	**Enable master Packet operation mode** 0: stepping operation type 1: packet operation type When enabled, the master controller will execute all commands automatically and handle some basic error conditions. Including SDA lock recovery, fail retry (4 times), and issue interrupt after all commands were completed. Besides, target device address is required for packet operation mode.
15	RW	**SDA_OE output direct control** Bit[15:12] is a GPIO function and only work when both Master and Slave function are disabled. When bus lock occurs, this GPIO function can help to recover the bus life. 0: output disable, tri-stated 1: output enable
14	RW	**SDA_O output direct control**
13	RW	**SCL_OE output direct control** 0: output disable, tri-stated 1: output enable
12	RW	**SCL_O output direct control**
11	RW	**Enable Bus Recover Command** 0: NOP 1: Issue Bus Recover Command This command can be applied only when SCL line status is High. When Bus Recovery Command is fired, 1 ∼ 8 SCL clock cycles will be issued to recover the bus. (Automatically stop issuing clock cycle whenever SDA has been released to high state) This command is used to recover the bus when a slave device locks SDA. When using this command, the Transfer Mode State Machine must be at the IDLE state; otherwise, it must be reset to the IDLE state firstly by disabling the device function (I2CSC00). When recover operation has been done, the current bus state can be checked from I2CSC08[17] or I2CSM14[15].
10	RO	**Reserved (0)**
9	RW	**Enable Master Rx DMA Data buffer**
8	RW	**Enable Master Tx DMA Data buffer**
7	RW	**Enable Master Rx Pool Data buffer**
6	RW	**Enable Master Tx Pool Data buffer** 0: Disable 1: Enable
5	RW	**Master Stop Command (P)** 0: NOP 1: Issue Master Stop Command **4th** priority.
4	RW	**Master Receive Command Last** 0: Respond ACK after last RX byte 1: Respond NACK after last RX byte If Stop command is set, then it will also respond NACK after the last Rx byte.

to next page

from previous page

3	RW	**Master Receive Command (RxD)** 0: NOP 1: Fire Master Receive Command **3rd** priority.
2	RO	**Reserved (0)**
1	RW	**Master Transmit Command (TxD)** 0: NOP 1: Fire Master Transmit Command **2nd** priority.
0	RW	**Master Start Command (S/Sr)** 0: NOP 1: Issue Master Start/Repeated Start Command **1st** priority.

Note :
The bit position follows the old register I2CSD14 definition.
For old stepping command mode, it will execute the commands according to the following sequence (priority):

S → TxD → RxD → P

If packet operation mode is enabled, then the command sequence would become:

S → Aw → TxD → Sr → Ar → RxD → P

HW will clear each command (bit[11:0]) automatically when it is finished. Beside, HW will clear all the commands whenever Master Arbitration Lost or invalid Start/Stop conditions have been detected.

Offset: 20h **I2CSS20: Slave Interrupt Control Register** **Init = 0**

Bit	R/W	Description
31:18	RO	**Reserved (0)**
17	RW	**Enable Slave address matched but NACKed interrupt** This interrupt is valid only when packet operation mode and auto NACK enabled. The NACK status can be checked from I2CSS24[22:20]. There are 2 NACK conditions: 1. I2CSS28[14] = 1 ⇒ Hit to active address but has no buffer ready. This condition also will trigger packet done interrupt. 2. I2CSS28[15] = 1 ⇒ Hit to enabled but non-active address.
16	RW	**Enable Packet command done interrupt** When in packet command mode, it only needs to enable this bit and/or bit17. All other bits have no effect.
15	RW	**Enable Slave mode inactive timeout interrupt** Alert for the state machine stays at Slave active mode for a long time (timeout counter). This maybe the condition that controller work at slave mode, but suddenly the master stop to transfer, and this makes the state machine stuck at slave mode and can not quit.
14:6	RO	**Reserved (0)**
5	RW	**Enable abnormal Start/Stop condition detection interrupt** This interrupt is used to indicate a bus Start/Stop condition has been detected at an illegal transfer state.
4	RW	**Enable normal Stop condition detection interrupt** This interrupt is used to report a Stop pattern has been detected during receive data cycle, or a NACK was received during transmit data cycle.
3	RO	**Reserved (0)**

to next page

from previous page

2	RW	**Enable Receive Done interrupt** Receive Done means the buffer is full, or salve controller receives a termination signal.
1	RO	**Reserved (0)**
0	RW	**Enable Transmit ended with ACK response interrupt** Transmit end means Byte, Pool or DMA mode Tx command end.
Note : The bit position follows the old register I2CSD0C definition. When packet operation is enabled, the interrupt will be issued only after packet operation was completed. The definition of this register is : 0 : Disable 1 : Enable		

Offset: 24h **I2CSS24: Slave Interrupt Status Register** **Init = 0**

Bit	R/W	Description
31:30	RO	**The Last Slave address match indicator** 00: address1 01: address2 10: address3
29	RO	**Slave Address Received Pending** 0: none 1: A slave address match received interrupt is pending due to previous slave DMA receive done is not processed.
28:26	RO	**Reserved**
25:24	RO	**Current Slave parking status** 00: Idle 01: Wait for RX buffer 10: Wait for Tx buffer 11: undefined
23	RO	**Reserved**
22	RW	**(WC) Slave address3 NACKed indicator**
21	RW	**(WC) Slave address2 NACKed indicator**
20	RW	**(WC) Slave address1 NACKed indicator**
19:18	RO	**The current active Slave address indicator** 00: address1 01: address2 10: address3 This field is valid only after at least 1 byte of data was received or transmitted, not including the slave address byte.
17	RW	**(WC) Packet command fail interrupt status** The detailed failure status can be checked from bit5, bit15.
16	RW	**(WC) Packet command done interrupt status** When in packet mode, clear this bit will also clear bit0, bit2, bit[5:4], bit7, bit15, bit17.
15	RW	**(WC) Slave mode inactive timeout interrupt status**
14:8	RO	**Reserved**
7	RW	**(WC) Slave Address Received Match interrupt status**
6	RO	**Reserved**
5	RW	**(WC) Abnormal Start/Stop Condition Detection interrupt status**

to next page

from previous page

4	RW	**(WC) Normal Stop Condition Detection interrupt status** Normal Stop includes below conditions: 1. Received Stop during Slave RX cycle 2. Received NACK during Slave TX cycle 3. Respond NACK at end of Slave RX cycle
3	RO	**Receive Done ended with returning NACK** This bit is cleared at the same time while clearing bit4.
2	RW	**(WC) Receive Done interrupt status**
1	RW	**(WC) Transmit ended with NACK response interrupt status** In general, slave transmit cycle responded by NACK indicates transfer end. So bit4 Normal Stop status also will be raised.
0	RW	**(WC) Transmit ended with ACK response interrupt status**
Note : The bit position follows the old register I2CSD10 definition. 'WC' means this bit is cleared by writing '1'. SW should clear bit[2:0] after all new commands are set ready, it is a fire control for new commands.		

Offset: 28h **I2CSS28: Slave Command/Status Register** **Init = 0**

Bit	R/W	Description
31	W1T	**Write '1' control for I2CSS28 for current command** When write to this register, if write data bit31 = 1, then it is write '1' possible only. This is useful to update this register without read-modify-write required.
30:20	RO	**Reserved (0)**
18:17	RW	**Active address selection for Packet operation** 00: address1 01: address2 10: address3 11: all enabled slave addresses
16	RW	**Enable slave Packet operation mode** When enabled, the slave controller will execute all commands automatically and handle some basic error conditions. Including timeout detection, and issue interrupt after all commands were completed. Besides, byte mode will be disabled under packet operation mode.
15	RW	**Enable Auto NACK to Non-Active address**
14	RW	**Enable Auto NACK to Active address** The auto NACK function only works under packet operation mode. When enabled, slave controller respond with NACK at address phase if address isn't active or buffer isn't ready. This bit is suggested to be enabled when the active address is not for "all". All slave addresses will be responded with ACK if autoNACK is not enabled.
13:11	RO	**Reserved (0)**
10	RW	**Enable issuing I2C/SMBus Slave Alert signal** 0: NOP 1: Issuing Alert signal to a bus Master This command is valid only when slave function is enabled. This bit will be cleared by hardware automatically after packet received with slave address matched.
9	RW	**Enable Slave Rx DMA Data buffer**
8	RW	**Enable Slave Tx DMA Data buffer**
7	RW	**Enable Slave Rx Pool Data buffer**

to next page

from previous page

6	RW	**Enable Slave Tx Pool Data buffer** 0: Disable 1: Enable Since Packet mode don't support byte mode, so it can only set bit[9:6] for packet mode Tx/Rx, bit2 and bit4 are not used by packet mode.
4	RW	**Slave Receive Command Last** 0: Receive Command can be continued by responding ACK 1: Receive Command will be ended by responding NACK When in Pool/DMA mode, the NACK will be sent after the latest byte is received.
2	RW	**Slave Transmit Command** 0: NOP 1: Fire Slave Transmit Command HW will clear this register when TX buffer is empty or bus contention error has been detected.
1:0	RO	**Reserved (0)**
Note : The bit position follows the old register I2CSD10 definition. For old stepping command mode, it will execute the commands according to the following sequence (priority): TxD → RxD(bit2=0) If packet operation mode is enabled, then the command sequence is based on the $R/\overline{W}$ bit of the address byte. HW will clear each command (bit[11:0]) automatically when it has been finished.		

Offset: 40h	**I2CSS40: Slave Device Address Register**	**Init = 0**
Bit	**R/W**	**Description**
31:0	RW	**Same as I2CSD18**

53.6 Software Programming Guide

53.6.1 Top Operation Commands

1. Set GLOB_CTL : Global top function (I2CG0C) for all devices. (I2CG00[2]=1 for new register mode)
2. Set GLOB_CLK : Global top Clock Divider(I2CG10) for all devices.

53.6.2 Master Operation Commands with Packet mode

1. Support Byte, Pool buffer, and DMA modes.
2. Set FUNC_CTL : Top function (I2CC00) for each device.
3. Set AC_Timing: Timing control (I2CC04): for tCKHigh, tCKLow and Timeout timing should be set.
4. Set INT_CLR : Clear all required interrupts status(I2CM14).
5. Set INT_CTL : Enable all required interrupts (I2CM10).
   Packet mode interrupt requires I2CM10 not equal to 0. Example : set I2CM10 = 0x407F
6. Set Data_CTL : Prepare Tx data and Rx Buffer / Address
   - Set I2CC08 if Byte mode used
   - Set I2CC0C if Pool buffer mode used
   - Set I2CM30 if DMA Tx buffer mode used
   - Set I2CM34 if DMA Rx buffer mode used
   - Set I2CM1C = Tx/Rx DMA length if DMA mode used
7. Set CMD_CTL : Prepare Tx data and Rx Command (I2CM18)

- S → Aw → P
  - Set I2CM18 = 0x10021 + (target address << 24)
- S → Aw → TxD → P
  - Set I2CM18 = 0x10023 + (target address << 24) + (pool buffer control or DMA buffer control)
- S → Aw → TxD → master_wait_for_FW
  - Set I2CM18 = 0x10003 + (target address << 24) + (pool buffer control or DMA buffer control)
- master_wait_for_FW → Sr → Ar → RxD → P
  - Set I2CM18 = 0x10039 + (target address << 24) + (pool buffer control or DMA buffer control)
- S → Ar → RxD → P
  - Set I2CM18 = 0x10039 + (target address << 24) + (pool buffer control or DMA buffer control)
- S → Aw → TxD → Sr → Ar → RxD → P
  - Set I2CM18 = 0x1003B + (target address << 24) + (pool buffer control or DMA buffer control)

8. Wait interrupt and check CMD_STA (I2CM18), INT_STA (I2CM14).
   Example : I2CM14[17:16] for INT_STA of package mode

53.6.3 Slave Operation Commands with Packet mode

1. Support Pool buffer and DMA modes, no support Byte mode.
2. AutoNACK is an option to NACK master at address phase if no buffer is ready.
3. Set FUNC_CTL : Top function (I2CC00) for each device.
4. Set SLV_ADDR : Slave address (I2CS40).
5. Set AC_Timing: Timing control (I2CC04): for tCKHigh, tCKLow and Timeout timing should be set.
6. Set INT_CLR : Clear all required interrupts status(I2CS24). For slave mode, interrupt status should be cleared after all new commands are ready.
7. Set INT_CTL : Enable all required interrupts (I2CS20).
   Packet mode interrupt requires I2CM10 not equal to 0.
   Example : set I2CS20 = 0x803F
8. Set Data_CTL : Prepare Tx data and Rx Buffer / Address
   - Set I2CC0C if Pool buffer mode used
   - Set I2CS38 if DMA Tx buffer mode used
   - Set I2CS3C if DMA Rx buffer mode used
   - Set I2CS2C = Tx/Rx DMA length if DMA mode used
9. Set CMD_CTL : Prepare Tx data and Rx Command (I2CS28)
   - Set I2CS28[18:17] = (active addresses)
   - Set I2CS28[16] = 1 (enable slave package mode)
   - Set I2CS28[15] = (NACK selection for non-active address input)
   - Set I2CS28[9:6]= DMA Buffer/ Pool Buffer, TX/RX Selection
10. Slave packet operation types:
    - No buffer prepared
      - I2CS28[14] = 0 : S → Aw/r → ACK → slave_wait_for_FW [2]
      - I2CS28[14] = 1 : S → Aw/r → autoNACK → P
    - Only Rx buffer

[2] slave_wait_for_FW means holding SCL at low state and waiting for FW next command input.

  - I2CS28[14] = 0 : S → Ar → ACK → slave_wait_for_FW
  - I2CS28[14] = 1 : S → Ar → autoNACK → P
  - I2CS28[14] = * : S → Aw → RxD → P
  - I2CS28[14] = 0 : S → Aw → RxD → Sr → Ar → ACK → slave_wait_for_FW
  - I2CS28[14] = 1 : S → Aw → RxD → Sr → Ar → autoNACK → P
  - I2CS28[14] = * : slave_wait_for_FW → RxD → P
  - I2CS28[14] = 0 : slave_wait_for_FW → RxD → Sr → Ar → ACK → slave_wait_for_FW
  - I2CS28[14] = 1 : slave_wait_for_FW → RxD → Sr → Ar → autoNACK → P
- Only Tx buffer
  - I2CS28[14] = 0 : S → Aw → ACK → slave_wait_for_FW
  - I2CS28[14] = 1 : S → Aw → autoNACK → P
  - I2CS28[14] = * : S → Ar → TxD → P
  - I2CS28[14] = * : slave_wait_for_FW → TxD → P
- Rx + Tx buffers
  - I2CS28[14] = * : S → Ar → TxD → P
  - I2CS28[14] = * : S → Aw → RxD → P
  - I2CS28[14] = * : S → Aw → RxD → Sr → Ar → TxD → P
  - I2CS28[14] = * : slave_wait_for_FW → TxD → P
  - I2CS28[14] = * : slave_wait_for_FW → RxD → P
  - I2CS28[14] = * : slave_wait_for_FW → RxD → Sr → Ar → TxD → P

11. Wait interrupt and check CMD_STA (I2CS28), INT_STA (I2CS24).
    Example : I2CS24[18:16] or any other required interrupt flags.
12. When in the state of slave_wait_for_FW, it can check I2CS24[25:24] for the last slave parking state.

53.6.4 Master Operation Commands with Step DMA mode

1. Support Byte, Pool buffer, and DMA modes.
2. Set FUNC_CTL : Top function (I2CD00) for each device.
3. Set AC_Timing: Timing control (I2CD04): for tCKHigh, tCKLow and Timeout timing should be set.
4. Set TimeOutCTL: TimeOut Setting(I2CD08) for each device.
5. Set INT_CLR : Clear all required interrupts status(I2CD10) before any Command.
6. Set INT_CTL : Enable all required interrupts (I2CD0C).
7. Set Data_CTL : Prepare Tx data and Rx Buffer / Address
   - Set I2CD24 = DMA Buffer Base Address if DMA Tx/Rx buffer mode used
   - Set I2CD28 = DMA Buffer Length = real_cnt or 0xfff if DMA Tx/Rx buffer mode used
8. Set CMD_CTL : Prepare Tx data and Rx Command (I2CD14)
   - S $\rightarrow$ Aw
     * Set I2CD20 = (Slave Address $<<$ 1) + 0x0
     * Set I2CD14 = 0x0003
     * Wait & Clear I2CD10 : TxACK (from bit0)
   - master_wait_for_FW $\rightarrow$ Data $\rightarrow$ P
     * Set I2CD14 = 0x0122 (M TX DMA, Tx Stop, Tx)
     * Wait & Clear I2CD10 : TxACK, Normal Stop (from bit0, bit1)
   - S $\rightarrow$ Ar
     * Set I2CD20 = (Slave Address $<<$ 1) + 0x1
     * Set I2CD14 = 0x0003
     * Wait & Clear I2CD10 : TxACK (from bit0)
   - slave_wait_for_FW $\rightarrow$ Data $\rightarrow$ P
     * Set I2CD14 = 0x0238 (MRX DMA Last Stop)
     * Wait & Clear I2CD10 : RxDone, Normal Stop (from bit2, bit4)

53.6.5 Slave Operation Commands with Step DMA mode

1. Support Pool buffer and DMA modes, no support Byte mode.
2. AutoNACK is an option to NACK master at address phase if no buffer is ready.
3. Set FUNC_CTL : Top function (I2CD00) for each device.
4. Set SLV_ADDR : Slave address (I2CD18).
5. Set AC_Timing: Timing control (I2CD04): for tCKHigh, tCKLow and Timeout timing should be set.
6. Set INT_CLR : Clear all required interrupts status(I2CD10). For slave mode, interrupt status should be cleared after all new commands are ready.
7. Set INT_CTL : Enable all required interrupts (I2CD0C).
8. Set Data_CTL : Prepare Tx data and Rx Buffer / Address
   - Set I2CD24 = DMA Buffer Base Address if DMA Tx/Rx buffer mode used
   - Set I2CD28 = DMA Buffer Length = real_cnt or 0xfff if DMA Tx/Rx buffer mode used
9. Set CMD_CTL : Prepare Tx data and Rx Command (I2CD14)
   - S $\rightarrow$ Aw for Slave
     * Wait & Clear I2CD10 : RxDone, SlaveMatch(from bit2, bit7) 0x0100_0084
     * Check Buffer Data I2CD20[8]=0 (from I2CPROTOCOL, bit0 of Received Byte Buffer will be used as Read/Write at Aw/Ar stage)
   - slave_wait_for_FW $\rightarrow$ Data $\rightarrow$ P for Slave RX DMA
     * Set I2CD14 = 0x0200 (enable RX DMA)

    - Wait & Clear I2CD10 : RxDone if(total length $<$ 0xfff), Normal Stop(from bit2, bit4)
  - S $\rightarrow$ Ar
    - Wait & Clear I2CD10 : RxDone, SlaveMatch(from bit2, bit7) 0x0200_0084
    - Check Buffer Data I2CD20[8]=1 (from I2CPROTOCOL, bit0 of Received Byte Buffer will be used as Read/Write at Aw/Ar stage)
  - slave_wait_for_FW $\rightarrow$ Data $\rightarrow$ P for Slave TX DMA
    - Set I2CD14 = 0x0104 (STX DMA)
    - Wait & Clear I2CD10 : TxNACK(from bit1)

53.7 Software Programming Note for Step mode

53.7.1 Initialization

1. Write I2CD00 = enable function setting
2. Write I2CD04
3. Write I2CD08
4. Write I2CD10 = FFFFFFFF
5. Write I2CD0C = interrupt enable setting
6. Master and Slave function mode can be open individually or concurrently.

53.7.2 Byte Buffer and Buffer Pool Usage

- This IP supports Tx/Rx Byte buffer and buffer pool options.
- SW can choose 1 type or mix 2 types when using.
- The Byte buffer is dedicated to each device controller, but buffer pool is shared.
- The buffer pool can be assigned for Tx and Rx at the same time in a single command.

53.7.3 Buffer Pool Allocation

- The buffer pool is available for all devices.
- Follows the steps to use buffer pool:
  1. Allocate a region, SW management the buffer allocation.
  2. Fill data in the buffer for transmitting command.
  3. Set buffer control register I2CD1C for base pointer and Tx/Rx end pointer.
  4. Set command/status register I2CD14, bit[6] : Tx_buffer_enable, bit[7] : Rx_buffer_enable. SW can fire Tx and Rx buffer command at the same time.

53.7.4 Master Mode Command

- In Master mode, there are 4 command types:
  1. 1st priority : Transmit Start pattern, when set, HW will generate Start pattern onto Bus when bus is idle.
  2. 2nd priority : Transmit Data, transmitting data until buffer is empty, arbitration lost or invalid bus condition occurs.
  3. 3rd priority : Receive Data, receiving data until buffer is full, transmission is stopped by NACK response or invalid bus condition occurs.
  4. 4th priority : Transmit Stop pattern, generate Stop pattern onto Bus.
- The 4 command types can be set at any combination follows the priority definition.
- The Receive Data Last command is part of Receive Data command for responding NACK state at the end of receiving cycle.

53.7.5 Slave Mode Command

- In Slave mode, controller is default at receiving mode and monitor the bus state to check if a matched address packet occurs, when such packet detected, HW will ACK this packet automatically and generate an interrupt to SW if interrupt is enabled.
- For the first byte when receiving, it is not recommend to use buffer pool mode, because HW do not clarify that the packet is for reading or writing, it will keep at receiving mode until the buffer full.
  Of course, SW can enable the buffer pool mode at start-up for the first packet receiving, if SW can confirm the next received packet is a write command.
- In Slave function mode, it does not support continuous command sequence like the Master function do.

53.7.6 Master/Slave Dual Mode Command

- The controller is designed with the capability to function as a Master and Slave concurrently.
- The usage is the same as Master/Slave individually.
- But if the Master function is accessing a device and another Master is accessing the Slave function at the same time, there will have a arbitration procedure, if the Slave function address is smaller than the device that the Master function access, then the Master function will lose the arbitration and the controller will terminate the Master command immediately and switch to Slave function mode.
- The Master command and Slave command can not be set at the same time.

53.7.7 Interrupt Handler

- When interrupt occurred, SW interrupt handler first read I2CG00 to decide which controller activating the interrupt.
- And then, read the controller interrupt status register (I2CD10) to see what happened.
- When processed the related interrupt operation, write '1' to clear the individual interrupt flag.

53.7.8 Resetting Device

- If SW wants to reset the device controller, it only needs to turn off both the Master and Slave function, then HW will reset all state machine and status (including I2CD0C, I2CD10 and I2CD14).
  - Write I2CD00 bit[1:0] = 0 , reset device.
  - The following status will be cleared when reset:
    - I2CD0C : The interrupt enable control will be cleared to 0.
    - I2CD10 : The interrupt status flag will be cleared.
    - I2CD14 : The command register will be cleared.
    - All internal control state machine and counter will be reset to initial state.
  - Write I2CD00 bit[1:0] = 3 , enable Master/Slave device (defined by user).
  - Write I2CD0C = interrupt mode needed.
- But if the I2C Bus was locked by another device, reset the controller can not recover the bus life. It needs to implement the recover operation.

53.7.9 DMA Buffer Mode Usage

- Device1 and Device2 support another DMA buffer mode. Which has a maximum 4Kbyte buffer allocated in the DRAM.
- The usage of DMA buffer mode is the same as Buffer Pool mode, except for the different data store area.

53.7.10 Command and Interrupt Processing Sequence

- Master : Clear interrupt status first and then active command.
- Slave : Active command first and then clear interrupt status. Because at Slave mode, there is no receive enable control, it is the default mode, so must set the command first to clarify the next command is transmit or receive cycle, then clear the interrupt status to fire the operation.

53.7.11 SDA Bus Lock Recover

When I2C bus lock occurred,

- Master function : Implement bus lock recover program.
- Slave function : Reset device to release bus lock. Don't implement bus lock recover program.

This controller supports 2 ways to solve SDA bus lock conditions. The first one is auto recover mode, which was implemented by hardware automatically. The second is GPIO mode, which was implemented by software and more flexibility.

Bus Lock Recognition

When command or SCL low timeout occurred, the bus operation must have something wrong occurred. At this time, it must to make sure what is the last condition on the bus.

1. First release the bus by disabling the bus function (I2CD00[1:0] = 0).
2. Check the bus state on I2CD14[18:17]:
   - bit[18] = 1 and bit[17] = 0, it is a SDA bus lock condition and mostly can be recovered.
   - bit[18] = 1 and bit[17] = 1, there is no bus lock, it may SW programming error.
   - other cases, difficultly to recover.

Auto Recover Mode

1. Enable Master function
2. Set Recover command I2CD14[11] = 1
3. Wait recover done interrupt
4. Check bus state, I2CD14[18:17] = "11".
   - If meet, go to next.
   - If fail and retry $<$ 2, go to step 2 again.
   - If fail and retry $>$= 2, bus cannot be recovered.
5. Set Start + Stop command to reset bus.

GPIO Mode

At this mode, SW can implement your own recover mechanism.

54 I3C Controller (I3C)

54.1 Overview

Base Address of Global Register = 0x1E7A_0000
Register Address of I3C Controller = (Base Address of Global Register) + Offset
I3C010: I3C1Reg0
I3C014: I3C1Reg1
I3C018: I3C1Dbg1 (Master)
I3C018: I3C1Dbg1 (Slave)
I3C01C: I3C1Dbg2 (Master)
I3C01C: I3C1Dbg2 (Slave)
I3C020: I3C2Reg0
I3C024: I3C2Reg1
I3C028: I3C2Dbg1 (Master)
I3C028: I3C2Dbg1 (Slave)
I3C02C: I3C2Dbg2 (Master)
I3C02C: I3C2Dbg2 (Slave)
I3C030: I3C3Reg0
I3C034: I3C3Reg1
I3C038: I3C3Dbg1 (Master)
I3C038: I3C3Dbg1 (Slave)
I3C03C: I3C3Dbg2 (Master)
I3C03C: I3C3Dbg2 (Slave)
I3C040: I3C4Reg0
I3C044: I3C4Reg1
I3C048: I3C4Dbg1 (Master)
I3C048: I3C4Dbg1 (Slave)
I3C04C: I3C4Dbg2 (Master)
I3C04C: I3C4Dbg2 (Slave)
I3C050: I3C5Reg0
I3C054: I3C5Reg1
I3C058: I3C5Dbg1 (Master)
I3C058: I3C5Dbg1 (Slave)
I3C05C: I3C5Dbg2 (Master)
I3C05C: I3C5Dbg2 (Slave)
I3C060: I3C6Reg0
I3C064: I3C6Reg1
I3C068: I3C6Dbg1 (Master)
I3C068: I3C6Dbg1 (Slave)
I3C06C: I3C6Dbg2 (Master)
I3C06C: I3C6Dbg2 (Slave)

Register Address of I3CX Device = (Base Address of Global Register) + (Offset of I3CX) + Offset
X = 0, 1, 2, 3, 4, 5
Offset of I3C0 = 0x2000
Offset of I3C1 = 0x3000
Offset of I3C2 = 0x4000
Offset of I3C3 = 0x5000
Offset of I3C4 = 0x6000
Offset of I3C5 = 0x7000
I3CD000: Device Control Register
I3CD004: Device Address Register
I3CD008: Hardware Capability register
I3CD00C: Command Queue Port
I3CD00CTC: Transfer Command Data Structure

I3CD00CTARG: Transfer Argument Data Structure
I3CD00CSDA: Short Data Argument Data Structure
I3CD00CAAC: Address Assignment Command Data Structure
I3CD010: Response Queue
I3CD014RX: Receive Data Port Register
I3CD014TX: Transmit Data Port Register
I3CD018: In-Band Interrupt Queue Data/Status Register
I3CD01C: Queue Threshold Control Register
I3CD020: Data Buffer Threshold Control Register
I3CD024: IBI Queue Control Register
I3CD02C: IBI MR Request Rejection Control Register
I3CD030: IBI SIR Request Rejection Control Register
I3CD034: Reset Control Register
I3CD038: Slave Event Control Register
I3CD03C: Interrupt Status Register
I3CD040: Interrupt Status Enable Register
I3CD044: Interrupt Signal Enable Register
I3CD048: Interrupt Force Enable Register
I3CD04C: QUEUE STATUS LEVEL Register
I3CD050: DATA BUFFER STATUS LEVEL Register
I3CD054: PRESENT STATE Register
I3CD058: Device Operating Status Register
I3CD05C: Pointer for Device Address Table Registers
I3CD060: Pointer for Device Characteristics Table
I3CD06C: Pointer for Vendor specific Registers
I3CD070: Provisional ID Register
I3CD074: Provisional ID Register
I3CD078: I3C Slave Characteristic Register
I3CD07C: I3C Max Write/Read Length Register
I3CD080: MXDS Maximum Read Turnaround Time Register
I3CD084: MXDS Maximum Data Speed Register
I3CD08C: Slave Interrupt Request Register
I3CD090: TSP/TSL Symbol Timing Register
I3CD0B0: Device Control Extended Register
I3CD0B4: SCL I3C Open Drain Timing Register
I3CD0B8: SCL I3C Push Pull Timing Register
I3CD0BC: SCL I2C Fast Mode Timing Register
I3CD0C0: SCL I2C Fast Mode Plus Timing Register
I3CD0C8: SCL Extended Low Count Timing Register
I3CD0CC: SCL Termination Bit Low count Timing Register
I3CD0D0: SDA Hold and Mode Switch Delay Timing Register
I3CD0D4: Bus Free Timing Register
I3CD0D8: Bus Idle Timing Register
I3CD0DC: SCL Low Master Extended Timeout Register
I3CD0E0: I3C Version ID Register
I3CD0E4: I3C Version TYPE Register
I3CD0E8: I3C Extended Capabilities Register
I3CD200: Device Characteristic Table Location-1 of Device1 (Master Mode) / Device Characteristic Table Location of Device1 (Secondary Master Mode)
I3CD204: Device Characteristic Table Location-2 of Device1 (Master Mode) / Device Characteristic Table Location of Device2 (Secondary Master Mode)
I3CD208: Device Characteristic Table Location-3 of Device1 (Master Mode) / Device Characteristic Table Location of Device3 (Secondary Master Mode)
I3CD20C: Device Characteristic Table Location-4 of Device1 (Master Mode) / Device Characteristic Table Location of Device4 (Secondary Master Mode)
I3CD210: Device Characteristic Table Location-1 of Device2 (Master Mode) / Device Characteristic Table Location of Device5 (Secondary Master Mode)

I3CD214: Device Characteristic Table Location-2 of Device2 (Master Mode) / Device Characteristic Table Location of Device6 (Secondary Master Mode)
I3CD218: Device Characteristic Table Location-3 of Device2 (Master Mode) / Device Characteristic Table Location of Device7 (Secondary Master Mode)
I3CD21C: Device Characteristic Table Location-4 of Device2 (Master Mode) / Device Characteristic Table Location of Device8 (Secondary Master Mode)
I3CD220: Device Characteristic Table Location-1 of Device3 (Master Mode) / Device Characteristic Table Location of Device9 (Secondary Master Mode)
I3CD224: Device Characteristic Table Location-2 of Device3 (Master Mode) / Device Characteristic Table Location of Device10 (Secondary Master Mode)
I3CD228: Device Characteristic Table Location-3 of Device3 (Master Mode) / Device Characteristic Table Location of Device11 (Secondary Master Mode)
I3CD22C: Device Characteristic Table Location-4 of Device3 (Master Mode) / Device Characteristic Table Location of Device12 (Secondary Master Mode)
I3CD230: Device Characteristic Table Location-1 of Device4 (Master Mode) / Device Characteristic Table Location of Device13 (Secondary Master Mode)
I3CD234: Device Characteristic Table Location-2 of Device4 (Master Mode) / Device Characteristic Table Location of Device14 (Secondary Master Mode)
I3CD238: Device Characteristic Table Location-3 of Device4 (Master Mode) / Device Characteristic Table Location of Device15 (Secondary Master Mode)
I3CD23C: Device Characteristic Table Location-4 of Device4 (Master Mode) / Device Characteristic Table Location of Device16 (Secondary Master Mode)
I3CD240: Device Characteristic Table Location-1 of Device5 (Master Mode) / Device Characteristic Table Location of Device17 (Secondary Master Mode)
I3CD244: Device Characteristic Table Location-2 of Device5 (Master Mode) / Device Characteristic Table Location of Device18 (Secondary Master Mode)
I3CD248: Device Characteristic Table Location-3 of Device5 (Master Mode) / Device Characteristic Table Location of Device19 (Secondary Master Mode)
I3CD24C: Device Characteristic Table Location-4 of Device5 (Master Mode) / Device Characteristic Table Location of Device20 (Secondary Master Mode)
I3CD250: Device Characteristic Table Location-1 of Device6 (Master Mode) / Device Characteristic Table Location of Device21 (Secondary Master Mode)
I3CD254: Device Characteristic Table Location-2 of Device6 (Master Mode) / Device Characteristic Table Location of Device22 (Secondary Master Mode)
I3CD258: Device Characteristic Table Location-3 of Device6 (Master Mode) / Device Characteristic Table Location of Device23 (Secondary Master Mode)
I3CD25C: Device Characteristic Table Location-4 of Device6 (Master Mode) / Device Characteristic Table Location of Device24 (Secondary Master Mode)
I3CD260: Device Characteristic Table Location-1 of Device7 (Master Mode) / Device Characteristic Table Location of Device25 (Secondary Master Mode)
I3CD264: Device Characteristic Table Location-2 of Device7 (Master Mode) / Device Characteristic Table Location of Device26 (Secondary Master Mode)
I3CD268: Device Characteristic Table Location-3 of Device7 (Master Mode) / Device Characteristic Table Location of Device27 (Secondary Master Mode)
I3CD26C: Device Characteristic Table Location-4 of Device7 (Master Mode) / Device Characteristic Table Location of Device28 (Secondary Master Mode)
I3CD270: Device Characteristic Table Location-1 of Device8 (Master Mode) / Device Characteristic Table Location of Device29 (Secondary Master Mode)
I3CD274: Device Characteristic Table Location-2 of Device8 (Master Mode) / Device Characteristic Table Location of Device30 (Secondary Master Mode)
I3CD278: Device Characteristic Table Location-3 of Device8 (Master Mode) / Device Characteristic Table Location of Device31 (Secondary Master Mode)
I3CD27C: Device Characteristic Table Location-4 of Device8 (Master Mode) / Device Characteristic Table Location of Device32 (Secondary Master Mode)
I3CD280: Device Address Table of Device1
I3CD284: Device Address Table of Device2
I3CD288: Device Address Table of Device3

I3CD28C: Device Address Table of Device4
I3CD290: Device Address Table of Device5
I3CD294: Device Address Table of Device6
I3CD298: Device Address Table of Device7
I3CD29C: Device Address Table of Device8

54.2 Registers

Offset: 10h		**I3C010: I3C1Reg0**	**Init = 0**
31:30	R	**RSVD**	
29:28	RW	**SDA_PULLUP_EN[1:0]** **2'b00: 2K ohm** **2'b01: RSVD** **2'b10: 750 ohm** **2'b11: 545 ohm**	
27:26	R	**RSVD**	
25	RW	**CDR_EN_MASK**	
24	RW	**CDR_EN_DG**	
23:16	RW	**CDR_DLYMAX**	
15:12	RW	**DG_SCL_RMAX**	
11:8	RW	**DG_SCL_FMAX**	
7 :4	RW	**DG_SDA_RMAX**	
3 :0	RW	**DG_SDA_FMAX**	

Offset: 14h		**I3C014: I3C1Reg1**	**Init = 0**
31:20	R	**RSVD**	
19:16	RW	**inst_id** Slave instance ID Applicable in only Slave mode of operation. **Active State:** High	
15	RW	**static_addr_en** Slave static address valid Applicable in only Slave mode of operation. **Active State:** High	
14:8	RW	**static_addr** Slave static address Applicable in only Slave mode of operation. **Active State:** High	
7 :4	RW	**pending_int** Pending interrupt info for GETSTATUS CCC Applicable in only Slave mode of operation. **Active State:** High	

3 :2	RW	**act_mode** Slave activity mode for GETSTATUS CCC Applicable in only Slave mode of operation. **Active State:** High
1	RW	**slv_test_mode** Slave Test Mode Applicable in only Slave mode of operation. **Active State:** High
0	RW	**mode_i2c** I2C or I3C mode select signal Applicable in only Slave mode of operation. **Active State:** High

Offset: 18h	**I3C018: I3C1Dbg1 (Master)**	**Init = 0**
31:0	R	**- Master Mode Debug Signals** **debug[0] indicates 1 if Master is the current master.** **debug[1] indicates 1 if SCL is in Push-Pull Mode.** **debug[2] indicates 1 if SDA is in Push-Pull Mode.** **debug[3] indicates SCL signal level.** **debug[4] indicates SDA signal level.** **debug[5] indicates 1 if Command Queue is full.** **debug[6] indicates 1 if Command Queue is empty.** **debug[7] indicates 1 if Response Queue is full.** **debug[8] indicates 1 if Response Queue is empty.** **debug[9] indicates 1 if IBI Queue is full.** **debug[10] indicates 1 if IBI Queue is empty.** **debug[11] indicates 1 if Transmit Buffer is full.** **debug[12] indicates 1 if Transmit Buffer is empty.** **debug[13] indicates Receive Buffer full.** **debug[14] indicates Receive Buffer empty.** **debug[20:15] indicates Current Master Transfer command.** **debug[26:21] indicates Current Master Transfer command state.** **debug[30:27] indicates Current Master Transfer command TID.** **debug[31] indicates 1 if the Master controller is in Idle state.**

Offset: 18h	**I3C018: I3C1Dbg1 (Slave)**	**Init = 0**
31:0	R	**- Slave Mode Debug Signals** **debug[4:0] Reserved** **debug[5] indicates 1 if Command Queue is full.** **debug[6] indicates 1 if Command Queue is empty.** **debug[7] indicates 1 if Response Queue is full.** **debug[8] indicates 1 if Response Queue is empty.** **debug[10:9] Reserved** **debug[11] indicates 1 if Transmit Buffer is full.** **debug[12] indicates 1 if Transmit Buffer is empty.** **debug[13] indicates 1 if Receive Buffer is full.** **debug[14] indicates 1 if Receive Buffer is empty.** **debug[31:15] Reserved**

Offset: 1Ch		I3C01C: I3C1Dbg2 (Master)	Init = 0
31:24	R	**RSVD**	
23	R	**sda_in: SDA input signal.**	
22	R	**sda_pullup_en: SDA pullup enable signal.**	
21	R	**sda_out: SDA output signal.**	
20	R	**sda_oe: SDA output enable signal.**	
19	R	**scl_in: SCL input signal.**	
18	R	**scl_pullup_en: SCL pullup enable signal.**	
17	R	**scl_out: SCL output signal.**	
16	R	**scl_oe: SCL output enable signal.**	
15:3	R	**RSVD**	
2	R	**debug[34] = Indicates Periodic Poll slot tick change pulse.**	
1	R	**debug[33] = Indicates if Periodic Poll command is in progress.**	
0	R	**debug[32] = Indicates the Enable status of the controller.**	

Offset: 1Ch		I3C01C: I3C1Dbg2 (Slave)	Init = 0
31:24	R	**RSVD**	
23	R	**sda_in: SDA input signal.**	
22	R	**sda_pullup_en: SDA pullup enable signal.**	
21	R	**sda_out: SDA output signal.**	
20	R	**sda_oe: SDA output enable signal.**	
19	R	**scl_in: SCL input signal.**	
18	R	**scl_pullup_en: SCL pullup enable signal.**	
17	R	**scl_out: SCL output signal.**	
16	R	**scl_oe: SCL output enable signal.**	
15:12	R	**Reserved**	
11:8	R	**debug[43:40] = Indicates Slave Command TID.**	
7:3	R	**debug[39:35] = Indicates Slave Transfer State.**	
2:1	R	**Reserved**	
0	R	**debug[32] = Indicates the Enable status of the controller.**	

Offset: 20h		I3C020: I3C2Reg0	Init = 0
31:30	R	**RSVD**	
29:28	RW	**SDA_PULLUP_EN[1:0]** **2'b00: 2K ohm** **2'b01: RSVD** **2'b10: 750 ohm** **2'b11: 545 ohm**	
27:26	R	**RSVD**	
25	RW	**CDR_EN_MASK**	
24	RW	**CDR_EN_DG**	
23:16	RW	**CDR_DLYMAX**	
15:12	RW	**DG_SCL_RMAX**	
11:8	RW	**DG_SCL_FMAX**	

7 :4	RW	**DG_SDA_RMAX**
3 :0	RW	**DG_SDA_FMAX**

Offset: 24h		**I3C024: I3C2Reg1** **Init = 0**
31:20	R	**RSVD**
19:16	RW	**inst_id** Slave instance ID Applicable in only Slave mode of operation. **Active State:** High
15	RW	**static_addr_en** Slave static address valid Applicable in only Slave mode of operation. **Active State:** High
14:8	RW	**static_addr** Slave static address Applicable in only Slave mode of operation. **Active State:** High
7 :4	RW	**pending_int** Pending interrupt info for GETSTATUS CCC Applicable in only Slave mode of operation. **Active State:** High
3 :2	RW	**act_mode** Slave activity mode for GETSTATUS CCC Applicable in only Slave mode of operation. **Active State:** High
1	RW	**slv_test_mode** Slave Test Mode Applicable in only Slave mode of operation. **Active State:** High
0	RW	**mode_i2c** I2C or I3C mode select signal Applicable in only Slave mode of operation. **Active State:** High

Offset: 28h		I3C028: I3C2Dbg1 (Master)	Init = 0
31:0	R	**- Master Mode Debug Signals** **debug[0] indicates 1 if Master is the current master.** **debug[1] indicates 1 if SCL is in Push-Pull Mode.** **debug[2] indicates 1 if SDA is in Push-Pull Mode.** **debug[3] indicates SCL signal level.** **debug[4] indicates SDA signal level.** **debug[5] indicates 1 if Command Queue is full.** **debug[6] indicates 1 if Command Queue is empty.** **debug[7] indicates 1 if Response Queue is full.** **debug[8] indicates 1 if Response Queue is empty.** **debug[9] indicates 1 if IBI Queue is full.** **debug[10] indicates 1 if IBI Queue is empty.** **debug[11] indicates 1 if Transmit Buffer is full.** **debug[12] indicates 1 if Transmit Buffer is empty.** **debug[13] indicates Receive Buffer full.** **debug[14] indicates Receive Buffer empty.** **debug[20:15] indicates Current Master Transfer command.** **debug[26:21] indicates Current Master Transfer command state.** **debug[30:27] indicates Current Master Transfer command TID.** **debug[31] indicates 1 if the Master controller is in Idle state.**	

Offset: 28h		I3C028: I3C2Dbg1 (Slave)	Init = 0
31:0	R	**- Slave Mode Debug Signals** **debug[4:0] Reserved** **debug[5] indicates 1 if Command Queue is full.** **debug[6] indicates 1 if Command Queue is empty.** **debug[7] indicates 1 if Response Queue is full.** **debug[8] indicates 1 if Response Queue is empty.** **debug[10:9] Reserved** **debug[11] indicates 1 if Transmit Buffer is full.** **debug[12] indicates 1 if Transmit Buffer is empty.** **debug[13] indicates 1 if Receive Buffer is full.** **debug[14] indicates 1 if Receive Buffer is empty.** **debug[31:15] Reserved**	

Offset: 2Ch		I3C02C: I3C2Dbg2 (Master)	Init = 0
31:24	R	**RSVD**	
23	R	**sda_in: SDA input signal.**	
22	R	**sda_pullup_en: SDA pullup enable signal.**	
21	R	**sda_out: SDA output signal.**	
20	R	**sda_oe: SDA output enable signal.**	
19	R	**scl_in: SCL input signal.**	
18	R	**scl_pullup_en: SCL pullup enable signal.**	
17	R	**scl_out: SCL output signal.**	
16	R	**scl_oe: SCL output enable signal.**	
15:3	R	**RSVD**	
2	R	**debug[34] = Indicates Periodic Poll slot tick change pulse.**	
1	R	**debug[33] = Indicates if Periodic Poll command is in progress.**	
0	R	**debug[32] = Indicates the Enable status of the controller.**	

Offset: 2Ch		I3C02C: I3C2Dbg2 (Slave)	Init = 0
31:24	R	**RSVD**	
23	R	**sda_in: SDA input signal.**	
22	R	**sda_pullup_en: SDA pullup enable signal.**	
21	R	**sda_out: SDA output signal.**	
20	R	**sda_oe: SDA output enable signal.**	
19	R	**scl_in: SCL input signal.**	
18	R	**scl_pullup_en: SCL pullup enable signal.**	
17	R	**scl_out: SCL output signal.**	
16	R	**scl_oe: SCL output enable signal.**	
15:12	R	**Reserved**	
11:8	R	**debug[43:40] = Indicates Slave Command TID.**	
7:3	R	**debug[39:35] = Indicates Slave Transfer State.**	
2:1	R	**Reserved**	
0	R	**debug[32] = Indicates the Enable status of the controller.**	

Offset: 30h		I3C030: I3C3Reg0	Init = 0
31:30	R	**RSVD**	
29:28	RW	**SDA_PULLUP_EN[1:0]** **2'b00: 2K ohm** **2'b01: RSVD** **2'b10: 750 ohm** **2'b11: 545 ohm**	
27:26	R	**RSVD**	
25	RW	**CDR_EN_MASK**	
24	RW	**CDR_EN_DG**	
23:16	RW	**CDR_DLYMAX**	
15:12	RW	**DG_SCL_RMAX**	
11:8	RW	**DG_SCL_FMAX**	
7 :4	RW	**DG_SDA_RMAX**	
3 :0	RW	**DG_SDA_FMAX**	

Offset: 34h		I3C034: I3C3Reg1	Init = 0
31:20	R	**RSVD**	
19:16	RW	**inst_id** Slave instance ID Applicable in only Slave mode of operation. **Active State:** High	
15	RW	**static_addr_en** Slave static address valid Applicable in only Slave mode of operation. **Active State:** High	

14:8	RW	**static_addr** Slave static address Applicable in only Slave mode of operation. **Active State:** High
7 :4	RW	**pending_int** Pending interrupt info for GETSTATUS CCC Applicable in only Slave mode of operation. **Active State:** High
3 :2	RW	**act_mode** Slave activity mode for GETSTATUS CCC Applicable in only Slave mode of operation. **Active State:** High
1	RW	**slv_test_mode** Slave Test Mode Applicable in only Slave mode of operation. **Active State:** High
0	RW	**mode_i2c** I2C or I3C mode select signal Applicable in only Slave mode of operation. **Active State:** High

Offset: 38h		**I3C038: I3C3Dbg1 (Master)**	**Init = 0**
31:0	R	**- Master Mode Debug Signals** **debug[0] indicates 1 if Master is the current master.** **debug[1] indicates 1 if SCL is in Push-Pull Mode.** **debug[2] indicates 1 if SDA is in Push-Pull Mode.** **debug[3] indicates SCL signal level.** **debug[4] indicates SDA signal level.** **debug[5] indicates 1 if Command Queue is full.** **debug[6] indicates 1 if Command Queue is empty.** **debug[7] indicates 1 if Response Queue is full.** **debug[8] indicates 1 if Response Queue is empty.** **debug[9] indicates 1 if IBI Queue is full.** **debug[10] indicates 1 if IBI Queue is empty.** **debug[11] indicates 1 if Transmit Buffer is full.** **debug[12] indicates 1 if Transmit Buffer is empty.** **debug[13] indicates Receive Buffer full.** **debug[14] indicates Receive Buffer empty.** **debug[20:15] indicates Current Master Transfer command.** **debug[26:21] indicates Current Master Transfer command state.** **debug[30:27] indicates Current Master Transfer command TID.** **debug[31] indicates 1 if the Master controller is in Idle state.**	

Offset: 38h		I3C038: I3C3Dbg1 (Slave)	Init = 0
31:0	R	**- Slave Mode Debug Signals** **debug[4:0] Reserved** **debug[5] indicates 1 if Command Queue is full.** **debug[6] indicates 1 if Command Queue is empty.** **debug[7] indicates 1 if Response Queue is full.** **debug[8] indicates 1 if Response Queue is empty.** **debug[10:9] Reserved** **debug[11] indicates 1 if Transmit Buffer is full.** **debug[12] indicates 1 if Transmit Buffer is empty.** **debug[13] indicates 1 if Receive Buffer is full.** **debug[14] indicates 1 if Receive Buffer is empty.** **debug[31:15] Reserved**	

Offset: 3Ch		I3C03C: I3C3Dbg2 (Master)	Init = 0
31:24	R	**RSVD**	
23	R	**sda_in: SDA input signal.**	
22	R	**sda_pullup_en: SDA pullup enable signal.**	
21	R	**sda_out: SDA output signal.**	
20	R	**sda_oe: SDA output enable signal.**	
19	R	**scl_in: SCL input signal.**	
18	R	**scl_pullup_en: SCL pullup enable signal.**	
17	R	**scl_out: SCL output signal.**	
16	R	**scl_oe: SCL output enable signal.**	
15:3	R	**RSVD**	
2	R	**debug[34] = Indicates Periodic Poll slot tick change pulse.**	
1	R	**debug[33] = Indicates if Periodic Poll command is in progress.**	
0	R	**debug[32] = Indicates the Enable status of the controller.**	

Offset: 3Ch		I3C03C: I3C3Dbg2 (Slave)	Init = 0
31:24	R	**RSVD**	
23	R	**sda_in: SDA input signal.**	
22	R	**sda_pullup_en: SDA pullup enable signal.**	
21	R	**sda_out: SDA output signal.**	
20	R	**sda_oe: SDA output enable signal.**	
19	R	**scl_in: SCL input signal.**	
18	R	**scl_pullup_en: SCL pullup enable signal.**	
17	R	**scl_out: SCL output signal.**	
16	R	**scl_oe: SCL output enable signal.**	
15:12	R	**Reserved**	
11:8	R	**debug[43:40] = Indicates Slave Command TID.**	
7:3	R	**debug[39:35] = Indicates Slave Transfer State.**	
2:1	R	**Reserved**	
0	R	**debug[32] = Indicates the Enable status of the controller.**	

Offset: 40h		I3C040: I3C4Reg0	Init = 0
31:30	R	**RSVD**	
29:28	RW	**SDA_PULLUP_EN[1:0]** **2'b00: 2K ohm** **2'b01: RSVD** **2'b10: 750 ohm** **2'b11: 545 ohm**	
27:26	R	**RSVD**	
25	RW	**CDR_EN_MASK**	
24	RW	**CDR_EN_DG**	
23:16	RW	**CDR_DLYMAX**	
15:12	RW	**DG_SCL_RMAX**	
11:8	RW	**DG_SCL_FMAX**	
7 :4	RW	**DG_SDA_RMAX**	
3 :0	RW	**DG_SDA_FMAX**	

Offset: 44h		I3C044: I3C4Reg1	Init = 0
31:20	R	**RSVD**	
19:16	RW	**inst_id** Slave instance ID Applicable in only Slave mode of operation. **Active State:** High	
15	RW	**static_addr_en** Slave static address valid Applicable in only Slave mode of operation. **Active State:** High	
14:8	RW	**static_addr** Slave static address Applicable in only Slave mode of operation. **Active State:** High	
7 :4	RW	**pending_int** Pending interrupt info for GETSTATUS CCC Applicable in only Slave mode of operation. **Active State:** High	
3 :2	RW	**act_mode** Slave activity mode for GETSTATUS CCC Applicable in only Slave mode of operation. **Active State:** High	
1	RW	**slv_test_mode** Slave Test Mode Applicable in only Slave mode of operation. **Active State:** High	

0	RW	**mode_i2c** I2C or I3C mode select signal Applicable in only Slave mode of operation. **Active State:** High

Offset: 48h	**I3C048: I3C4Dbg1 (Master)**	**Init = 0**
31:0	R	**- Master Mode Debug Signals** **debug[0] indicates 1 if Master is the current master.** **debug[1] indicates 1 if SCL is in Push-Pull Mode.** **debug[2] indicates 1 if SDA is in Push-Pull Mode.** **debug[3] indicates SCL signal level.** **debug[4] indicates SDA signal level.** **debug[5] indicates 1 if Command Queue is full.** **debug[6] indicates 1 if Command Queue is empty.** **debug[7] indicates 1 if Response Queue is full.** **debug[8] indicates 1 if Response Queue is empty.** **debug[9] indicates 1 if IBI Queue is full.** **debug[10] indicates 1 if IBI Queue is empty.** **debug[11] indicates 1 if Transmit Buffer is full.** **debug[12] indicates 1 if Transmit Buffer is empty.** **debug[13] indicates Receive Buffer full.** **debug[14] indicates Receive Buffer empty.** **debug[20:15] indicates Current Master Transfer command.** **debug[26:21] indicates Current Master Transfer command state.** **debug[30:27] indicates Current Master Transfer command TID.** **debug[31] indicates 1 if the Master controller is in Idle state.**

Offset: 48h	**I3C048: I3C4Dbg1 (Slave)**	**Init = 0**
31:0	R	**- Slave Mode Debug Signals** **debug[4:0] Reserved** **debug[5] indicates 1 if Command Queue is full.** **debug[6] indicates 1 if Command Queue is empty.** **debug[7] indicates 1 if Response Queue is full.** **debug[8] indicates 1 if Response Queue is empty.** **debug[10:9] Reserved** **debug[11] indicates 1 if Transmit Buffer is full.** **debug[12] indicates 1 if Transmit Buffer is empty.** **debug[13] indicates 1 if Receive Buffer is full.** **debug[14] indicates 1 if Receive Buffer is empty.** **debug[31:15] Reserved**

Offset: 4Ch	**I3C04C: I3C4Dbg2 (Master)**	**Init = 0**
31:24	R	**RSVD**
23	R	**sda_in: SDA input signal.**
22	R	**sda_pullup_en: SDA pullup enable signal.**
21	R	**sda_out: SDA output signal.**
20	R	**sda_oe: SDA output enable signal.**
19	R	**scl_in: SCL input signal.**

18	R	**scl_pullup_en: SCL pullup enable signal.**
17	R	**scl_out: SCL output signal.**
16	R	**scl_oe: SCL output enable signal.**
15:3	R	**RSVD**
2	R	**debug[34] = Indicates Periodic Poll slot tick change pulse.**
1	R	**debug[33] = Indicates if Periodic Poll command is in progress.**
0	R	**debug[32] = Indicates the Enable status of the controller.**

Offset: 4Ch		**I3C04C: I3C4Dbg2 (Slave)** **Init = 0**
31:24	R	**RSVD**
23	R	**sda_in: SDA input signal.**
22	R	**sda_pullup_en: SDA pullup enable signal.**
21	R	**sda_out: SDA output signal.**
20	R	**sda_oe: SDA output enable signal.**
19	R	**scl_in: SCL input signal.**
18	R	**scl_pullup_en: SCL pullup enable signal.**
17	R	**scl_out: SCL output signal.**
16	R	**scl_oe: SCL output enable signal.**
15:12	R	**Reserved**
11:8	R	**debug[43:40] = Indicates Slave Command TID.**
7:3	R	**debug[39:35] = Indicates Slave Transfer State.**
2:1	R	**Reserved**
0	R	**debug[32] = Indicates the Enable status of the controller.**

Offset: 50h		**I3C050: I3C5Reg0** **Init = 0**
31:30	R	**RSVD**
29:28	RW	**SDA_PULLUP_EN[1:0]** **2'b00: 2K ohm** **2'b01: RSVD** **2'b10: 750 ohm** **2'b11: 545 ohm**
27:26	R	**RSVD**
25	RW	**CDR_EN_MASK**
24	RW	**CDR_EN_DG**
23:16	RW	**CDR_DLYMAX**
15:12	RW	**DG_SCL_RMAX**
11:8	RW	**DG_SCL_FMAX**
7 :4	RW	**DG_SDA_RMAX**
3 :0	RW	**DG_SDA_FMAX**

Offset: 54h		**I3C054: I3C5Reg1** **Init = 0**
31:20	R	**RSVD**

19:16	RW	**inst_id** Slave instance ID Applicable in only Slave mode of operation. **Active State:** High
15	RW	**static_addr_en** Slave static address valid Applicable in only Slave mode of operation. **Active State:** High
14:8	RW	**static_addr** Slave static address Applicable in only Slave mode of operation. **Active State:** High
7 :4	RW	**pending_int** Pending interrupt info for GETSTATUS CCC Applicable in only Slave mode of operation. **Active State:** High
3 :2	RW	**act_mode** Slave activity mode for GETSTATUS CCC Applicable in only Slave mode of operation. **Active State:** High
1	RW	**slv_test_mode** Slave Test Mode Applicable in only Slave mode of operation. **Active State:** High
0	RW	**mode_i2c** I2C or I3C mode select signal Applicable in only Slave mode of operation. **Active State:** High

Offset: 58h		I3C058: I3C5Dbg1 (Master)	Init = 0
31:0	R	**- Master Mode Debug Signals** **debug[0] indicates 1 if Master is the current master.** **debug[1] indicates 1 if SCL is in Push-Pull Mode.** **debug[2] indicates 1 if SDA is in Push-Pull Mode.** **debug[3] indicates SCL signal level.** **debug[4] indicates SDA signal level.** **debug[5] indicates 1 if Command Queue is full.** **debug[6] indicates 1 if Command Queue is empty.** **debug[7] indicates 1 if Response Queue is full.** **debug[8] indicates 1 if Response Queue is empty.** **debug[9] indicates 1 if IBI Queue is full.** **debug[10] indicates 1 if IBI Queue is empty.** **debug[11] indicates 1 if Transmit Buffer is full.** **debug[12] indicates 1 if Transmit Buffer is empty.** **debug[13] indicates Receive Buffer full.** **debug[14] indicates Receive Buffer empty.** **debug[20:15] indicates Current Master Transfer command.** **debug[26:21] indicates Current Master Transfer command state.** **debug[30:27] indicates Current Master Transfer command TID.** **debug[31] indicates 1 if the Master controller is in Idle state.**	

Offset: 58h		I3C058: I3C5Dbg1 (Slave)	Init = 0
31:0	R	**- Slave Mode Debug Signals** **debug[4:0] Reserved** **debug[5] indicates 1 if Command Queue is full.** **debug[6] indicates 1 if Command Queue is empty.** **debug[7] indicates 1 if Response Queue is full.** **debug[8] indicates 1 if Response Queue is empty.** **debug[10:9] Reserved** **debug[11] indicates 1 if Transmit Buffer is full.** **debug[12] indicates 1 if Transmit Buffer is empty.** **debug[13] indicates 1 if Receive Buffer is full.** **debug[14] indicates 1 if Receive Buffer is empty.** **debug[31:15] Reserved**	

Offset: 5Ch		I3C05C: I3C5Dbg2 (Master)	Init = 0
31:24	R	**RSVD**	
23	R	**sda_in: SDA input signal.**	
22	R	**sda_pullup_en: SDA pullup enable signal.**	
21	R	**sda_out: SDA output signal.**	
20	R	**sda_oe: SDA output enable signal.**	
19	R	**scl_in: SCL input signal.**	
18	R	**scl_pullup_en: SCL pullup enable signal.**	
17	R	**scl_out: SCL output signal.**	
16	R	**scl_oe: SCL output enable signal.**	
15:3	R	**RSVD**	
2	R	**debug[34] = Indicates Periodic Poll slot tick change pulse.**	
1	R	**debug[33] = Indicates if Periodic Poll command is in progress.**	
0	R	**debug[32] = Indicates the Enable status of the controller.**	

Offset: 5Ch		I3C05C: I3C5Dbg2 (Slave)	Init = 0
31:24	R	**RSVD**	
23	R	**sda_in: SDA input signal.**	
22	R	**sda_pullup_en: SDA pullup enable signal.**	
21	R	**sda_out: SDA output signal.**	
20	R	**sda_oe: SDA output enable signal.**	
19	R	**scl_in: SCL input signal.**	
18	R	**scl_pullup_en: SCL pullup enable signal.**	
17	R	**scl_out: SCL output signal.**	
16	R	**scl_oe: SCL output enable signal.**	
15:12	R	**Reserved**	
11:8	R	**debug[43:40] = Indicates Slave Command TID.**	
7:3	R	**debug[39:35] = Indicates Slave Transfer State.**	
2:1	R	**Reserved**	
0	R	**debug[32] = Indicates the Enable status of the controller.**	

Offset: 60h		I3C060: I3C6Reg0	Init = 0
31:30	R	**RSVD**	
29:28	RW	**SDA_PULLUP_EN[1:0]** **2'b00: 2K ohm** **2'b01: RSVD** **2'b10: 750 ohm** **2'b11: 545 ohm**	
27:26	R	**RSVD**	
25	RW	**CDR_EN_MASK**	
24	RW	**CDR_EN_DG**	
23:16	RW	**CDR_DLYMAX**	
15:12	RW	**DG_SCL_RMAX**	
11:8	RW	**DG_SCL_FMAX**	
7 :4	RW	**DG_SDA_RMAX**	
3 :0	RW	**DG_SDA_FMAX**	

Offset: 64h		I3C064: I3C6Reg1	Init = 0
31:20	R	**RSVD**	
19:16	RW	**inst_id** Slave instance ID Applicable in only Slave mode of operation. **Active State:** High	
15	RW	**static_addr_en** Slave static address valid Applicable in only Slave mode of operation. **Active State:** High	

14:8	RW	**static_addr** Slave static address Applicable in only Slave mode of operation. **Active State:** High
7 :4	RW	**pending_int** Pending interrupt info for GETSTATUS CCC Applicable in only Slave mode of operation. **Active State:** High
3 :2	RW	**act_mode** Slave activity mode for GETSTATUS CCC Applicable in only Slave mode of operation. **Active State:** High
1	RW	**slv_test_mode** Slave Test Mode Applicable in only Slave mode of operation. **Active State:** High
0	RW	**mode_i2c** I2C or I3C mode select signal Applicable in only Slave mode of operation. **Active State:** High

Offset: 68h		**I3C068: I3C6Dbg1 (Master)**	**Init = 0**
31:0	R	**- Master Mode Debug Signals** **debug[0] indicates 1 if Master is the current master.** **debug[1] indicates 1 if SCL is in Push-Pull Mode.** **debug[2] indicates 1 if SDA is in Push-Pull Mode.** **debug[3] indicates SCL signal level.** **debug[4] indicates SDA signal level.** **debug[5] indicates 1 if Command Queue is full.** **debug[6] indicates 1 if Command Queue is empty.** **debug[7] indicates 1 if Response Queue is full.** **debug[8] indicates 1 if Response Queue is empty.** **debug[9] indicates 1 if IBI Queue is full.** **debug[10] indicates 1 if IBI Queue is empty.** **debug[11] indicates 1 if Transmit Buffer is full.** **debug[12] indicates 1 if Transmit Buffer is empty.** **debug[13] indicates Receive Buffer full.** **debug[14] indicates Receive Buffer empty.** **debug[20:15] indicates Current Master Transfer command.** **debug[26:21] indicates Current Master Transfer command state.** **debug[30:27] indicates Current Master Transfer command TID.** **debug[31] indicates 1 if the Master controller is in Idle state.**	

Offset: 68h		I3C068: I3C6Dbg1 (Slave)	Init = 0
31:0	R	**- Slave Mode Debug Signals** **debug[4:0] Reserved** **debug[5] indicates 1 if Command Queue is full.** **debug[6] indicates 1 if Command Queue is empty.** **debug[7] indicates 1 if Response Queue is full.** **debug[8] indicates 1 if Response Queue is empty.** **debug[10:9] Reserved** **debug[11] indicates 1 if Transmit Buffer is full.** **debug[12] indicates 1 if Transmit Buffer is empty.** **debug[13] indicates 1 if Receive Buffer is full.** **debug[14] indicates 1 if Receive Buffer is empty.** **debug[31:15] Reserved**	

Offset: 6Ch		I3C06C: I3C6Dbg2 (Master)	Init = 0
31:24	R	**RSVD**	
23	R	**sda_in: SDA input signal.**	
22	R	**sda_pullup_en: SDA pullup enable signal.**	
21	R	**sda_out: SDA output signal.**	
20	R	**sda_oe: SDA output enable signal.**	
19	R	**scl_in: SCL input signal.**	
18	R	**scl_pullup_en: SCL pullup enable signal.**	
17	R	**scl_out: SCL output signal.**	
16	R	**scl_oe: SCL output enable signal.**	
15:3	R	**RSVD**	
2	R	**debug[34] = Indicates Periodic Poll slot tick change pulse.**	
1	R	**debug[33] = Indicates if Periodic Poll command is in progress.**	
0	R	**debug[32] = Indicates the Enable status of the controller.**	

Offset: 6Ch		I3C06C: I3C6Dbg2 (Slave)	Init = 0
31:24	R	**RSVD**	
23	R	**sda_in: SDA input signal.**	
22	R	**sda_pullup_en: SDA pullup enable signal.**	
21	R	**sda_out: SDA output signal.**	
20	R	**sda_oe: SDA output enable signal.**	
19	R	**scl_in: SCL input signal.**	
18	R	**scl_pullup_en: SCL pullup enable signal.**	
17	R	**scl_out: SCL output signal.**	
16	R	**scl_oe: SCL output enable signal.**	
15:12	R	**Reserved**	
11:8	R	**debug[43:40] = Indicates Slave Command TID.**	
7:3	R	**debug[39:35] = Indicates Slave Transfer State.**	
2:1	R	**Reserved**	
0	R	**debug[32] = Indicates the Enable status of the controller.**	

Offset: 00h		DEVICE_CTRL: Devicee Control Register	Init = 0
31	RW	**Controls whether I3C is enabled in mastr modee of operation** 1: Enables the I3C controller. 0: Disables the I3C controller. Software can disable I3C while it is active. However, the disable happens after all the initiated commands are completed in the command queue and Master FSM is in IDLE state (as indicated in PRESENT_STATE Register). Software can read back 1'b0 from this field once disabling of I3C is completed. Controls whether I3C slave controller is enabled. 1: Enables the I3C slave controller. 0: Disables the I3C skave controller. Software can disable I3C while it is active. However, the disable happens after the ongoing transfer is completed on the I3C bus. Software can read back 1'b0 from this field once disabling of I3C is completed. After power on reset, the software can enable I3C slave controller by programming this bit to 1'b1. However the I3C bus interface of the controller, will respond to transfer on the bus, only after it observes Bus Available condition for BUS_FREE_TIMING*IDLE_CNT_mULTIPLIER counts of pclk period. The successful completion of Enable/Disable of the controller depends on availability of SCL to the controller at the time of performing this operation, and hence may not happen instantly. **Value After Reset:** 0x0 **Volatile:** true	
30	RW	**I3C Resume** This bit is used to resume the Controller after it goes to Halt state. In the master mode of operation the controller goes to the halt state (as indicated in PRESENT_STATE Register) due to any type of error in the transfer (the type of error is indicated by ERR_STATUS field in the RESPONSE_QUEUE_PORT register). After the controller has gone to halt state, the application has to write 1'b1 to this bit to resume the controller. This bit is auto-cleared once the controller resumes the transfers by initiating the next command. In the slave mode of operation the controller goes to the halt state due to following conditions 1 -¿ Any type of error in the transfer (the type of error is indicated by ERR_STATUS field in the RESPONSE_QUEUE_PORT rigister) 2 -¿ MRL Register updated by the master through SETMRL CCC. After the controller has gone to halt state, the application has to take necessary action to handle the error condition and then write 1'b1 to this bit to resume the controller. This bit is auto-cleared once the controller is ready to accept new transfers. **Value After Reset:** 0x0 **Volatile:** true	
29	RW	**I3C Abort.** This bit is used in master mode of operation. This bit allows the user to relinquish the I3C bus before completing the issued transfer. In response to an ABORT request, the controller issues the STOP condition after the complete data byte is transferred or recieved. This bit is auto-cleared once the transfer is aborted and controller issues a 'Transfer Abort' interrupt. **Value After Reset:** 0x0 **Volatile:** true	

28	RW	**DMA Handshake Interface Enable.** This bit is used to enable or disable the DMA Handshaking interface. 1: Enables the DMA handshake control to interact with external DMA. 0: The DMA handshake control has no significance. **Value After Reset:** 0x0
27	RW	**Control bit to enable the controller to adapt to I2C/I3C mode of operation.** This bit is used in slave mode of operation. If this bit is programmed to 1'b1, the controller will determine its mode of operation based on the transfers on the bus. If static address is received first, the mode will be I2C and if 7E is received at any point, the mode will change to I3C. This also means the controller will initiate a hot-join only after it confirms that it is connected to an I3C bus. This bit will be cleared automatically if controller determines the mode as I3C. If this bit is not set, the controller will initiate a hot-join even before determining the bus mode assuming itself to be on I3C bus. This bit should be set only if the slave application doesn't know to which bus the device is connected. **Value After Reset:** 0x0 **Volatile:** true
26	R	**RSVD26: These bits in Device Control Register is reserved. It will always return 0.** **Value After Reset:** 0x0
25:24	RW	**Idle count multiplier.** This bit is used in slave mode of operation. After power on reset, the slave controller will be enabled only after it sees both SDA and SCL lines are idle for a specified time. This idle time is calculated by multipliying IDLE_CNT_MULITPLIER with I3C_IBI_FREE field in the BUS_FREE_TIMING register. - 00 - I3C_IBI_FREE * 1 - 01 - I3C_IBI_FREE * 2 - 10 - I3C_IBI_FREE * 4 - 11 - I3C_IBI_FREE * 8
23:16	RW	**MDB: Mandatory Byte in Slave Mode** **Value After Reset:** 0x0
15:11	R	**RSVD11_15: These bits in Device Control Register is reserved. It will always return 0. These bits are reserved in slave mode of operation** **Value After Reset:** 0x0
10	RW	**SLAVE_PEC_EN: Enable Hardware PEC in Slave Mode** **Value After Reset:** 0x0
9	RW	**IBI_PAYLOAD_EN: Enable IBI with Data in Slave Mode** **Value After Reset:** 0x0

8	RW	**Hot-Join Ack/Nack Control** This bit is used in master mode of operation. This bit acts as global control to ACK/NACK the Hot-Join Request from the devices. The I3C Master will ACK/NACK the Hot-Join request from other devices connected on the I3C Bus, based on programming of this bit. 0: ACK the Hot-join request. 1: NACK and send broadcast CCC to disable Hot-join. **Values:** 0x0 (DISABLED): Ack Hot-Join requests 0x1 (ENABLED): Nack and auto-disable Hot-Join request **Value After Reset:** 0x0
7	RW	**I2C Slave Present** This bit is used in master mode of operation. This Bit indicates whether any Legacy I2C Devices are present in the system. When this bit is set, controller uses I2C_OD_LCNT as the minimum time to start a transfer after stop condition, else it uses I3C_MST_FREE for this. In HDR mode, this field is used to select TSL over TSP in mixed bus configuration. **Values:** 0x0 (DISABLED): I2C Slave not present 0x1 (ENABLED): I2C Slave present **Value After Reset:** 0x0
6 :1	RW	**Reserved Field: Yes**
0	RW	**I3C Broadcast Address include.** This bit is used in master mode of operation. This bit is used to include I3C broadcast address (0x7E) for private transfer. **Note:** If I3C broadcast address is not included for the private transfers, In-band Interrupts (IBI) driven from Slaves may not win address arbitration. Hence, the IBIs will get delayed. **Values:** 0x0 (NOT_INCLUDED): I3C Broadcast Address is not included for Private Transfers 0x1 (INCLUDED): I3C Broadcast Address is included for Private Transfers **Value After Reset:** 0x0

Offset: 04h **DEVICE_ADDR: Device Address Register** **Init = 0**

Bit	R/W	Description
31	RW	**Dynamic Address Valid** This bit is used to control whether the DYNAMIC_ADDR is valid or not. - In I3C Main Master mode, the user sets this bit to 1 as it self-assigns its dynamic address. - In all other operation modes, the Controller sets this bit to 1 when Main Master assigns the Dynamic address during ENTDAA or SETDASA mechanism. - In I3C Slave Mode the Controller sets this bit to 1 when Main Master assigns the Dynamic address during ENTDAA or SETDASA mechanism. **Value After Reset:** 0x0 **Volatile:** true
30:23	R	**RSVD_30_23: These bits in Device Address Register are reserved. It will always return 0.** **Value After Reset:** 0x0

to next page

from previous page

22:16	RW	**Device Dynamic Address** This field is used to program the Device Dynamic Address. The Controller uses this address for I3C transfers. - In Main Master mode, the user/application has to program the Dynamic Address through the Slave interface as it self-assigns its Dynamic Address. - In all other modes, the Main Master assigns this address during ENTDAA or SETDASA mechanism. - The Main Master assigns this address during ENTDAA or SETDASA mechanism. **Value After Reset:** 0x0 **Volatile:** true
15	RW	**Static Address Valid** In slave mode of operation this bit reflects the value of static_addr_en input port. The input port static_addr_en is expected to be driven to 1 only if the device supports I2C or I3C Static Address. **Values:** 0x0 (INVALID): Static Address is invalid 0x1 (VALID): Static Address is valid **Volatile:** true
14:7	R	**RSVD_14_7: These bits in Device Address Register are reserved. It will always return 0.** **Value After Reset:** 0x0
6 :0	RW	**Device Static Address.** In slave mode of operation this field reflects the value of static_addr input port. The controller uses this address to respond to SETDASA CCC Command to get the Dynamic Address if static address is valid (static_addr_en port is set to 1). **Value After Reset:** 0x0 **Volatile:** true

Offset: 08h **HW_CAPABILITY: Hardware Capability register** **Init = 0**

Bit	R/W	Description
31:20	R	**RSVD_31_20: These bits in Hardware Capabilities Register are reserved. It will always return 0.** **Value After Reset:** 0x0
19	R	**SLV_IBI_CAP: Specifies slave's capability to initiate slave interrupt requests.** **Value After Reset:** 0x1
18	R	**SLV_HJ_CAP: Specifies slave's capability to initiate Hot-join request** **Value After Reset:** 0x1
17	R	**DMA_EN: Specifies whether controller is configured to have DMA handshaking interface.** **Value After Reset:** 0x1

to next page

from previous page

16:11	R	**HDR_TX_CLOCK_PERIOD: Reflects the IC_HDR_TX_CLK_PERIOD Configurable Parameter.** **Value After Reset:** 0x40
10:5	R	**CLOCK_PERIOD: Reflects the IC_CLK_PERIOD Configurable Parameter** **Value After Reset:** 0x5
4	R	Specifies master's capability to initiate HDR-TS transfers. 0 : HDR-TS not supported 1 : HDR-TS supported **Value After Reset:** 0x1
3	R	Specifies master's capability to initiate HDR-DDR transfers. 0 : HDR-DDR not supported 1 : HDR-DDR supported **Value After Reset:** 0x1
2:0	R	Specifies the configured role of I3C controller - 1: Master Only - 2: Programmable Master-Slave - 3: Secondary Master - 4: Slave Only **Value After Reset:** 0x3

Offset: 0Ch **COMMAND_QUEUE_PORT** **Init = 0**

Bit	R/W	Description
31:0	W	**32 bit command** **Please reference I3CD00CTC, I3CD00CTARG, I3CD00CSDA and I3CD00CAAC for detail structure.**

Offset: 0Ch_TC **Transfer Command Data Structure** **Init = 0**

Bit	R/W	Description
2:0	W	**CMD_ATTR** Command Attribute -Defines the Command Type and its Bit-Field Format. -0: Transfer Command -1: Transfer Argument -2: Short Data Argument -3: Address Assignment Command -4-7: Reserved

to next page

from previous page

6:3	W	**TID** Transaction ID This Field is used as the identification tag for the commands. The I3C controller returns this ID along with the response upon completion or upon error. - 4'b0000 - 4'b0111 - User Defined TID - 4'b1000 - 4'b1111 - Reserved for I3C controller.
14:7	W	**CMD** Transfer Command This field is used to define the Transfer Command type. The field can be programmed to: 1. 8-bit Common Command Code for CCC transfers. 2. 7-bit Command Code for HDR-TS or HDR-DDR transfers (bit[14] is reserved).
15	W	**CP** Command Present This bit is used to control whether the transfer should be initiated with the Transfer Command represented in the 'CMD' filed or not. - 0 - CMD field is not valid - 1 - CMD field is valid This bit is applicable for CCC and HDR transfers.
20:16	W	**DEV_INDX** Device Index This field is used to refer the Device Address Table for getting the target address. DEV_INDX field points to the offset address of Device Address Table.
23:21	W	**SPEED** Speed This field is used to indicate the speed in which the transfer should be driven. - 0: SDR0 - 1: SDR1 - 2: SDR2 - 3: SDR3 - 4: SDR4 - 5: HDR-TS - 6: HDR-DDR - 7: RESERVED Note that HDR-DDR and HDR-TS are supported only if the Transfer Command Structure is set-up to use Transfer Argument and NOT Short Data Argument
25:24	W	**RESV** Reserved

to next page

from previous page

26	W	**ROC** Response On Completion This field indicates whether the Response Status is required or not, after the execution of this command for the successful transfer. - 1 - Response Status is required. - 0 - Response Status is not required. Note: 1) The exception to the above control is that the response status gets generated when the transfer has encountered an error condition. 2) It is recommended that the ROC bit is always set to 1 for the Read commands (RnW=1), so that the number of data received is indicated (through DATA_LENGTH field in Response port) if the Slave terminates early than the Master.
27	W	**SDAP** Short Data Argument Present This field indicates whether the command written prior to the Base command should be treated as Short Data Argument or the Transfer Argument. - 0 - Prior written command is Transfer Argument. - 1 - Prior written command is Short Data Argument.
28	W	**RnW** Read and Write This bit controls whether a Read or Write transfer is performed. - 0 - Write Transfer - 1 - Read Transfer Note: In HDR transfers, this bit is used to set the Read/Write flag of the HDR-TS/HDR-DDR Command Code.
29	W	**RESV** Reserved
30	W	**TOC** Termination On Completion This bit controls whether a STOP need to be issued after the completion of the transfer or not. - 1 - STOP issued after this transfer. - 0 - The next transfer starts with RESTART condition.
31	W	**PEC** Parity Error Check Enable This bit enables generation and validation of PEC byte for SDR CCC and private transfers. - 0: PEC check is disabled. - 1: PEC check is enabled. Note: This bit is valid only for SDR Transfers and not for HDR Transfers.

Offset: 0Ch_TARG	Transfer Argument Data Structure	Init = 0

Bit	R/W	Description
2:0	W	**CMD_ATTR** Command Attribute -Defines the Command Type and its Bit-Field Format. -0: Transfer Command -1: Transfer Argument -2: Short Data Argument -3: Address Assignment Command -4-7: Reserved
7:3	W	**RESV** Reserved
15:8	W	**DB** Defining Byte Value DB indicates the 8-bit defining byte to be transferred in the CCC transfer. This byte is valid only when both CP and DBP bits are enabled, otherwise the controller ignores this byte.
31:16	W	**DL** Data Length This field is used to indicate the Data length of the transfer.

Offset: 0Ch_SDA	Short Data Argument Data Structure	Init = 0

Bit	R/W	Description
2:0	W	**CMD_ATTR** Command Attribute -Defines the Command Type and its Bit-Field Format. -0: Transfer Command -1: Transfer Argument -2: Short Data Argument -3: Address Assignment Command -4-7: Reserved
5:3	W	**BYTE_STRB** Byte Strobe This Field is used to select the valid data bytes of the Short Data Argument. - BYTE_STRB[0] - Data Byte -0 Valid Qualifier - BYTE_STRB[1] - Data Byte -1 Valid Qualifier - BYTE_STRB[2] - Data Byte -2 Valid Qualifier Valid combinations = 3'b001, 3'b011 and 3'b111
7:6	W	**RESV** Reserved

to next page

from previous page

15:8	W	**DATA_BYTE_0/Defining Byte** Data Byte -0/Defining Byte This field is used for storing the Data Byte -0. DB indicates the 8-bit Defining Byte to be transferred in the CCC transfer. This byte is valid only when both CP and DBP bits are enabled, otherwise the controller ignores this byte. Note: When Defining Byte (DBP) is enabled in Short Data Argument, 1. The BYTE_STRB[0] bit is treated to be always enabled whether or not you set it. 2. The maximum possible payload bytes to be sent is 2 due the first byte being occupied by the Defining Byte.
23:16	W	**DATA_BYTE_1** Data Byte -1
31:24	W	**DATA_BYTE_2** Data Byte -2

Offset: 0Ch_AAC **Address Assignment Command Data Structure** **Init = 0**

Bit	R/W	Description
2:0	W	**CMD_ATTR** Command Attribute -Defines the Command Type and its Bit-Field Format. -0: Transfer Command -1: Transfer Argument -2: Short Data Argument -3: Address Assignment Command -4-7: Reserved
6:3	W	**TID** Transaction ID This Field is used as the identification tag for the commands. The I3C controller returns this ID along with the response upon completion or upon error. - 4'b0000 - 4'b0111 - User Defined TID - 4'b1000 - 4'b1111 - Reserved for I3C controller.
14:7	W	**CMD** Transfer Command This field is used to define the Transfer Command type. The field can be programmed to: 1. 8-bit Common Command Code for CCC transfers. 2. 7-bit Command Code for HDR-TS or HDR-DDR transfers (bit[14] is reserved).
15	W	**CP** Command Present This bit is used to control whether the transfer should be initiated with the Transfer Command represented in the 'CMD' filed or not. - 0 - CMD field is not valid - 1 - CMD field is valid This bit is applicable for CCC and HDR transfers.
20:16	W	**DEV_INDX** Device Index This field is used to refer the Device Address Table for getting the target address. DEV_INDX field points to the offset address of Device Address Table.

to next page

from previous page

23:21	W	**DEV_COUNT** Device Count This field is used to represent the number of devices to be assigned with the Dynamic Address.
25:24	W	**RESV** Reserved
26	W	**ROC** Response On Completion This field indicates whether the Response Status is required or not, after the execution of this command for the successful transfer. - 1 - Response Status is required. - 0 - Response Status is not required. Note: 1) The exception to the above control is that the response status gets generated when the transfer has encountered an error condition. 2) It is recommended that the ROC bit is always set to 1 for the Read commands (RnW=1), so that the number of data received is indicated (through DATA_LENGTH field in Response port) if the Slave terminates early than the Master.
29:27	W	**RESV** Reserved
30	W	**TOC** Termination On Completion This bit controls whether a STOP need to be issued after the completion of the transfer or not. - 1 - STOP issued after this transfer. - 0 - The next transfer starts with RESTART condition.
31	W	**RESV** Reserved

Offset: 10h	COMMAND_QUEUE_PORT : Response Queue	Init = 0
Bit	**R/W**	**Description**
31:28	R	**ERR_STATUS** Error Type Defines the Error Type of the processed command or received vendor extension CCC packet (Slave mode). - 0: No Error - 1: CRC Error - 2: Parity Error - 3: Frame Error - 4: I3C Broadcast Address NACK Error - 5: Address NACKed. This bit will be set in case the Slave NACKs for Dynamic Address Assignment during ENTDAA process. - 6: Receive Buffer Overflow/Transmit Buffer Underflow (Only for HDR Transfers) - 7: Reserved - 8: Transfer Aborted - 9: I2C Slave Write Data NACK Error - 10-15: Reserved
27:24	R	**TID** Transaction ID This Field is used as the identification tag for the commands. The I3C controller returns the ID received through commands. - 4'b0000 - 4'b0111 - User Defined TID (specified in the Transfer/Address Assignment Command) - 4'b1000 - Master Write Data Status (Slave only) - 4'b1111 - CCC Write Data Status. - 4'b1001 - 4'b1110 - Reserved for I3C controller.
23:16	R	**CCCT** CCC/HDR Header Type This field represents the CCC type of the received vendor extension CCC packet or the HDR header of the received HDR transaction. This field indicates the CCC type when the TID is set to 4'b1111 (reserved for all other transactions). During the master transactions and slave non-HDR transactions, this field returns 4'b0000 and can be considered as don't care.
15:0	R	**DL** Data Length For Write transfers, this field represents the remaining data length of the transfer if the transfer is terminated early (remaining data length = requested data length - transferred data length) For Read transfers, this field represents the actual amount of data received in bytes. For Address Assignment command, this field represents the remaining device count.

Offset: 14h	RX_DATA_PORT: Receive Data Port Register	Init = 0
Bit	**R/W**	**Description**
31:0	R	**Receive Data Port** The Receive data port is mapped to the Rx-Data Buffer. The Receive data is always packed in 4-byte aligned data words and stored in the Rx-Data Buffer. If the command length is not aligned to the 4-bytes, then the additional data bytes have to be ignored.

Offset: 14h	TX_DATA_PORT: Transmit Data Port Register	Init = 0
Bit	**R/W**	**Description**
31:0	W	**Transmit Data Port** The Transmit Data port is mapped to the Tx-Data Buffer. The transmit data should always be packed as 4-byte aligned data words and written to the Transmit Data Port register. If the Command length is not aligned to 4-bytes, then the additional bytes will be ignored.

Offset: 18h	IBI_QUEUE_DATA: In-Band Interrupt Queue Data Register	Init = 0
Bit	**R/W**	**Description**
31:0	R	**In-Band Interrupt Data** This register is mapped to the IBI Queue.The IBI Data is always packed in 4-byte aligned and put to the IBI Queue. If the incoming data is not aligned to the 4-bytes, then there will be unused bytes in the end location of the IBI transfer.

Offset: 18h	IBI_QUEUE_STATUS: In-Band Interrupt Queue Status Register	Init = 0
Bit	**R/W**	**Description**
31	R	**IBI Status** Indicates the status of the response returned for the received IBI. 1b0: The received IBI is responded with an ACK. Any non-zero value of the DATA_LEN field indicates the presence of data payload for the ACKed IBI. 1b1: The received IBI is responded with a NACK. An auto disable CCC command is issued if the received IBI address is valid and matching with the DAT entry. If an IBI is received from an unknown address (not a valid entry in DAT), the IBI_STS is set to 1
30	R	**ERROR** Indicates that during IBI Auto command, an error is encountered and the data from the Slave device is partially or fully discarded. The following errors are detected. - CRC/Parity Error (applicable for HDR transaction) - Slave Address NACK - 0x7E Address NACK - IBI buffer overflow (applicable for IBI HDR modes) 1b0: No Error. Transaction complete with no errors. 1b1: Error. Error encountered during IBI Auto command phase.
29:25	R	**These bits in IBI Queue Status Data Register are reserved. It will always return 0.**
24	R	**LAST_STATUS** When set, indicates that this status is the last for the received IBI. If the payload of the received SIR exceeds the programmed IBI data threshold, then the controller splits the IBI payload into multiple chunks of IBI_DATA_THLD size (max) which includes the timestamp bytes if enabled.
23:16	R	**These bits in IBI Queue Status Data Register are reserved. It will always return 0.**

to next page

from previous page

15:8	R	**IBI Identifier** The byte received after START which includes the address and the R/W bit. - Device address and R/W bit in case of Slave Interrupt or Master Request. - Hot-Join ID and R/W bit in case of Hot-Join IBI.
7 :0	R	**In-Band Interrupt data length.** **This field represents the length of data received along with the IBI, in bytes.**

Offset: 1Ch **QUEUE_THLD_CTRL: Queue Threshold Control Register** **Init = 0**

Bit	R/W	Description
31:24	RW	**In-Band Interrupt Status Threshold Value.** Every In Band Interrupt received (with or without data) by I3C controller generates an IBI status. This field controls the number of IBI status entries (or above) in the IBI queue that trigger the IBI_THLD_STAT interrupt. The valid range is 0 to 7. The software shall program only valid values. A value of 0 sets the threshold for 1 entry, and a value of N sets the threshold for N+1 entries. **Value After Reset:** 0x1
20:16	RW	**IBI Data Threshold Value** This field represents the IBI data segment size in Dwords (4 bytes). The minimum supported segment size is 1 (4 bytes) and the maximum supported size is 31. This field enables the slicing of the incoming IBI data and generate individual status and thereby promotes the cut-through operation in reading out the IBI data. **Value After Reset:** 0x0
15:8	RW	**Response Buffer Threshold Value.** Controls the number of entries (or above) in the Response Queue that trigger the RESP_READY_STAT_INTR interrupt. The valid range is 0 to 1. The software shall program only valid values. A value of 0 sets the threshold for 1 entry, and a value of N sets the threshold for N+1 entries. **Value After Reset:** 0x1
7 :0	RW	**Command Buffer Empty Threshold Value.** Controls the number of empty locations (or above) in the Command Queue that trigger CMD_QUEUE_READY_STAT interrupt. The valid range is 0 to 1. The software shall program only valid values. Value of N ranging from 1 to 1 sets the threshold to N empty locations and a value of 0 sets the threshold to indicate that the queue is completely empty. **Value After Reset:** 0x1

Offset: 20h **DATA_BUFFER_THLD_CTRL: Data Buffer Threshold Control Register** **Init = 0**

Bit	R/W	Description
31:28	R	**RSVD_31_28: These bits in Data Buffer Threshold Control register are reserved.** It will always return 0.
27	R	**RSVD_27: These bits in Data Buffer Threshold Control register are reserved.** It will always return 0.

to next page

from previous page

26:24	RW	**Receive Start Threshold Value.** In master mode of operation when the controller is set up to initiate a read transfer, it waits until the programmed number of empty locations(or more) are available in its receive buffer before it initiates the read transfer on the I3C Interface. The following configurable options are provided: - Store and Forward Mode: If the threshold value is set to buffer size, then the controller waits for one of the following to be true to initiate the read command: – Entire Receive FIFO to be empty if the data length is more than buffer size. – The data length number of locations to be empty in the Receive FIFO if data length is smaller than the buffer size. - Threshold Mode: In this case, if the threshold value is less than buffer size, then the controller will initiate the write command as soon as the programmed locations are filled in the Transmit FIFO. The supported values for RX_START_THLD are: - 000: 1 - 001: 4 - 010: 8 - 011: 16 - 100: 32 - 101: 64 **Value After Reset:** 0x1
23:20	R	**RSVD_23_20: These bits in Data Buffer Threshold Control Register are reserved.** It will always return 0.
19	R	**RSVD_19: These bits in Data Buffer Threshold Control Register are reserved.** It will always return 0.
18:16	RW	**Transfer Start Threshold Value.** In master mode of operation when the controller is set up to initiate a write transfer, it waits until the programmed number of entries (or more) are available in its transmit buffer before it initiates the write transfer on the I3C Interface. The following configurable options are provided: - Store and Forward Mode: If the threshold value is set to buffer size, then the controller waits for one of the following to be true to initiate the read command: – Entire Transmit FIFO to be empty if the data length is more than buffer size. – The data length number of locations to be empty in the Transmit FIFO if data length is smaller than the buffer size. - Threshold Mode: In this case, if the threshold value is less than buffer size, then the controller will initiate the write command as soon as the programmed locations are filled in the Transmit FIFO. The supported values for TX_START_THLD are: - 000: 1 - 001: 4 - 010: 8 - 011: 16 - 100: 32 - 101: 64 **Value After Reset:** 0x1

to next page

from previous page

15:12	R	**RSVD_15_12: These bits in Data Buffer Threshold Control Register are reserved.** It will always return 0.
11	R	**RSVD_11: These bits in Data Buffer Threshold Control register are reserved.** It will always return 0.
10:8	RW	**Receive Buffer Threshold Value.** In master mode of operation this field controls the number of entries (or above) in the Receive FIFO that trigger the RX_THLD_STAT interrupt. If the programmed value is greater than the buffer depth, then threshold will be set to 16. The supported values for 1 are - 000: 1 - 001: 4 - 010: 8 - 011: 16 - 100: 32 - 101: 64 **Value After Reset:** 0x1
7 :4	R	**RSVD_7_4: These bits in Data Buffer Threshold Control Register are reserved.** It will always return 0.
3	R	**RSVD_3: These bits in Data Buffer Threshold Control register are reserved.** It will always return 0.
2 :0	RW	**Transmit Buffer Threshold Value.** In master mode of operation this field controls the number of entries (or above) in the Transmit FIFO that trigger the RX_THLD_STAT interrupt. If the programmed value is greater than the buffer depth, then threshold will be set to 16. The supported values for 1 are - 000: 1 - 001: 4 - 010: 8 - 011: 16 - 100: 32 - 101: 64 **Value After Reset:** 0x1

Offset: 24h **IBI_QUEUE_CTRL: IBI Queue Control Register** **Init = 0**

Bit	R/W	Description
31:4	R	These bits in IBI queue control register are reserved. It will always return 0. **Value After Reset:** 0x0

to next page

from previous page

3	RW	**Notify Rejected Slave Interrupt Request Control** This bit is used to suppress the reporting to the application about individual SIR request rejected. - 0: Suppress passing the IBI Data to the IBI FIFO to intimate the application, if Individual Slave Interrupt Request is NACKed and auto-disabled based on the IBI_SIR_REQ_REJECT Register. - 1: Pass the IBI Data to the IBI FIFO to intimate the application, if Individual Slave Interrupt Request is NACKed and auto-disabled based on the IBI_SIR_REQ_REJECT Register. **Values:** - 0x0 (DISABLED): Notify SIR Rejected Disable - 0x1 (ENABLED): Notify SIR Rejected Enable **Value After Reset:** 0x0
2	R	**RSVD_2: These bits in IBI Queue Control register are reserved.** It will always return 0. **Value After Reset:** 0x0
1	RW	**Notify Rejected Master Request Control.** This bit is used to suppress the reporting to the application about individual Master request rejected. - 0: Suppress passing the IBI Data to the IBI FIFO to intimate the application, if Individual Master Request is NACKed and auto-disabled based on the IBI_MR_P2P_REQ_REJECT Register. - 1: Pass the IBI Data to the IBI FIFO to intimate the application, if Individual Master Request is NACKed and auto-disabled based on the IBI_MR_P2P_REQ_REJECT Register. NOTE: This register field is not used currently and is provided for future enhancements. **Valus:** - 0x0 (DISABLED): Notify Master Request Rejected Disable - 0x0 (DISABLED): Notify Master Request Rejected Disable **Value After Reset:** 0x0
0	RW	**Notify Rejected Hot-Join Control.** This bit is used to suppress the reporting to the application about Hot-Join request rejected (NACK and Auto Disable). - 0: Suppress passing the IBI Data to the IBI FIFO to intimate the application, if Hot-Join is NACKed and autodisabled based on the HJ_AUTO_DISABLE bit set to 1. - 1: Pass the IBI Data to the IBI FIFO to intimate the application, if Hot-Join is NACKed and auto-disabled based on the HJ_AUTO_DIABLE bit set to 1. **Valus:** - 0x0 (DISABLED): Notify Hot-Join Rejected Disable - 0x0 (ENABLED): Notify Hot-Join Rejected enable **Value After Reset:** 0x0

Offset: 2Ch	IBI_MR_REQ_REJECT: IBI MR Request Rejection Control Register	Init = 0
Bit	**R/W**	**Description**
31:0	RW	**In-band Master Request Reject.** The control bits of this field determines if the controller ACK's incoming Master Request or NACKs and Disables it. A device specific policy can be established by appropriately programming this register. - 0: ACK Master Request - 1: NACK and send Directed DISEC CCC to disable the interrupting slave. **Value After Reset:** 0x0

Offset: 30h	IBI_SIR_REQ_REJECT: IBI SIR Request Rejection Control Register	Init = 0
Bit	**R/W**	**Description**
31:0	RW	**In-band Slave Interrupt Request Reject.** The application of the DWC_mipi_i3c can decide whether to send ACK or NACK for a Slave request received from any I3C device. A device specific response control bit is provided to selectthe response option. Master will ACK/NACK the MasterRequest based on programming of control bit, correspondingto the interrupting device. - 0: ACK the SIR Request - 1: NACK and send directed auto disable CCC. **Value After Reset:** 0x0

Offset: 34h	RESET_CTRL: Reset Control Register	Init = 0
Bit	**R/W**	**Description**
31	RW	**Bus Reset.** This bit is only used in master mode of operation. Write 1'b1 to this bit to exercise Bus Reset Reset Pattern Generation based on Bus Reset Type selection. This bit is cleared automatically once the Bus Reset Pattern Generation is completed. **Value After Reset:** 0x0
30:29	RW	**Bus Reset type** Type of bus reset triggered by BUS_RESET field. Values: 2'b00: EXIT: Exit Pattern 2'b11: SCL_LOW_RESET Pattern Others: Reserved **Value After Reset:** 0x0
28:6	R	**RSVD_28_6: These bits in Reset Control Register are reserved.** It will always return 0. **Value After Reset:** 0x0
5	RW	**IBI Queue Software Reset** This bit is only used in master mode of operation. Write 1'b1 to this bit to exercise IBI Queue reset This bit will be cleared automatically once the IBI Queue reset is completed. **Value After Reset:** 0x0 **Volatile:** true

to next page

from previous page

4	RW	**Receive Buffer Software Reset.** Write 1'b1 to this bit to exercise Receive Buffer reset. This bit will be cleared automatically once the Receive buffer reset is completed. **Value After Reset:** 0x0 **Volatile:** true
3	RW	**Transmit Buffer Software Reset.** Write 1'b1 to this bit to exercise Transmit Buffer reset. This bit will be cleared automatically once the Transmit Buffer reset is completed. **Value After Reset:** 0x0 **Volatile:** true
2	RW	**Response Queue Software Reset.** Write 1'b1 to this bit to exercise Response Queue reset. This bit will be cleared automatically once the Response Queue reset is completed. **Value After Reset:** 0x0 **Volatile:** true
1	RW	**Command Queue Software Reset.** Write 1'b1 to this bit to exercise Command Queue reset. This bit will the cleared automatically once the Command Queue reset is completed. **Value After Reset:** 0x0 **Volatile:** true
0	RW	**Core Software Reset.** Write 1'b1 to this bit to exercise software reset. This will reset all Buffers - Receive, Transmit, Command and Response This bit will be cleared automatically once the core reset is completed. **Value After Reset:** 0x0 **Volatile:** true

Offset: 38h **SLV_EVENT_CTRL: Slave Event Control Register** **Init = 0**

Bit	R/W	Description
31:8	R	**RSVD_31_6: These bits in Slave Event Control Register are reserved.** It will always return 0. **Value After Reset:** 0x0
7	RW1C	**MWL Updated Status.** This bit indicates a SETMWL CCC is received by the slave. The updated MWL value can be read from SLV_MAX_LEN register. This status can be cleared by writing 1'b1 to this field after reading the updated MWL. **Value After Reset:** 0x0 **Volatile:** true
6	RW1C	**MRL Updated Status.** This bit indicates a SETMWL CCC is received by the slave. The updated MWL value can be read from SLV_MAX_LEN register. This status can be cleared by writing 1'b1 to this field after reading the updated MRL. **Value After Reset:** 0x0 **Volatile:** true

to next page

from previous page

5 :4	R	**Activity State Status.** - ENTAS0 - 00 - ENTAS1 - 01 - ENTAS2 - 10 - ENTAS3 - 11 This bit reflects the Activity State of slave set by the Master. **Value After Reset:** 0x0 **Volatile:** true
3	RW	**Hot-Join Interrupt Request.** This bit reflects whether the Hot-Join Request Interrupts are allowed on the I3C bus or not. Usually, this bit is set or cleared by the I3C Master through ENEC or DISEC CCC. **Value After Reset:** 0x1
2	R	**RSVD_2: These bits in Slave Event Control Register are reserved.** It will always return 0. **Value After Reset:** 0x0
1	R	**Master Interrupt Request.** In Slave mode of operation, this bit reflects whether the Master Interrupt Request are allowed on the I3C bus or not. Usually, this bit is set or cleared by the I3C Master through ENEC or DISEC CCC. **Value After Reset:** 0x1 **Volatile:** true
0	R	**Slave Interrupt Request.** This bit reflects whether the Slave Initiated Interrupts are allowed on the I3C bus or not. Usually, this bit is set or cleared by the I3C Master through ENEC or DISEC CCC. **Value After Reset:** 0x1

Offset: 3Ch **INTR_STATUS: Interrupt Status Register** **Init = 0**

Bit	R/W	Description
31:16	R	It will always return 0. **Value After Reset:** 0x0
15	RW1C	**BUS_RESET_DONE_STS** Bus Reset Pattern Generation Done Status. This field is used only in Master mode of operation. This interrupt is generated when the SCL Low Timeout Bus Reset Pattern Generation is completed. This bit can be cleared by writing 1’b1. **Value After Reset:** 0x0
14	R	It will always return 0. **Value After Reset:** 0x0

to next page

from previous page

13	RW1C	**BUSOWNER_UPDATED_STAT** This interrupt is set when the role of the controller changes from being a Master to Slave or vice versa. This bit can be cleared by writing 1'b1. **Value After Reset:** 0x0
12	RW1C	**IBI_UPDATED_STAT** IBI status is updated. This field is used only in slave mode of operation. The IBI requested to SIR request register is done and status is updated in the register. **Value After Reset:** 0x0
11	RW1C	**READ_REQ_RECV_STAT** Read Request Received. This field is used only in slave mode of operation. Read Request received from the current master when CMDQ is empty. This bit can be cleared by writing 1'b1. **Value After Reset:** 0x0
10	RW1C	**DEFSLV_STAT** Define Slave CCC Received Status. This interrupt is generated if any error occurs during transfer. This bit can be cleared by writing 1'b1. **Value After Reset:** 0x0
9	RW1C	**TRANSFER_ERR_STAT** Transfer Error Status. This interrupt is generated if DEFSLV CCC is received. The error type will be specified in the response packet associated with the command (in ERR_STATUS field of RESPONSE_QUEUE_PORT register). This bit can be cleared by writing 1'b1. **Value After Reset:** 0x0
8	RW1C	**DYN_ADDR_ASSGN_STAT** Dynamic Address Assigned Status. This field is used only in slave mode of operation. This interrupt is generated if the device's Dynamic Address is assigned through SETDASA or ENTDAA CCC. This bit can be cleared by writing 1'b1. **Value After Reset:** 0x0
7	R	**RSVD_7: This bit in Interrupt Status register is reserved.** It will always return 0. **Value After Reset:** 0x0
6	RW1C	**CCC_UPDATED_STAT** CCC Table Updated Status. This field is used only in slave mode of operation. This interrupt is generated if any of the CCC registers are updated by I3C Master through CCC commands. This interrupt can be cleared by writing 1'b1. **Value After Reset:** 0x0

to next page

from previous page

5	RW1C	**TRANSFER_ABORT_STAT** Transfer Abort Status. This field is used only in master mode of operation. This interrupt is generated if transfer is aborted. This interrupt can be cleared by writing 1'b1. **Value After Reset:** 0x0
4	RW	**RESP_READY_STAT_INTR** Response Queue Ready Status. This field is used only in master mode of operation. This interrupt is generated when number of entries in response queue is greater than or equal to threshold value specified by RESP_BUF_THLD field in QUEUE_THLD_CTRL register. This interrupt will be cleared automatically when number of entries in response buffer is less than threshold value specified. **Value After Reset:** 0x0
3	RW	**CMD_QUEUE_READY_STAT** Command Queue Ready. This interrupt is generated when number of empty locations in command queue is greater than or equal to threshold value specified by CMD_EMPTY_BUF_THLD field in QUEUE_THLD_CTRL register. This interrupt will be cleared automatically when number of empty locations in command buffer is less than threshold value specified. **Value After Reset:** 0x0
2	RW	**IBI_THLD_STAT** IBI Buffer Threshold Status. This field is used only in master mode of operation. This field is only used in master mode of operation This interrupt is generated when number of entries in IBI buffer is greater than or equal to threshold value specified by IBI_BUF_THLD field in QUEUE_THLD_CTRL register. This interrupt will be cleared automatically when number of entries in IBI buffer is less than threshold value specified. **Value After Reset:** 0x0
1	RW	**RX_THLD_STAT** Receive Buffer Threshold Status. This interrupt is generated when number of entries in receive buffer is greater than or equal to threshold value specified by RX_BUF_THLD field in DATA_BUFFER_THLD_CTRL register. This interrupt will be cleared automatically when number of entries in receive buffer is less than threshold value specified. **Value After Reset:** 0x0
0	RW	**TX_THLD_STAT** Transmit Buffer Threshold Status. This interrupt is generated when number of empty locations in transmit buffer is greater than or equal to threshold value specified by TX_EMPTY_BUF_THLD field in DATA_BUFFER_THLD_CTRL register. This interrupt will be cleared automatically when number of empty locations in transmit buffer is less than threshold value specified. **Value After Reset:** 0x0

Offset: 40h	INTR_STATUS_EN: Interrupt Status Enable Register	Init = 0

Bit	R/W	Description
31:16	R	These bits in Interrupt Status Register are reserved. It will always return 0. **Value After Reset:** 0x0
15	RW	**BUS_RESET_DONE_STS_EN** Bus Reset Pattern Generation Done Status Enable. This field is used only in Master mode of operation. **Value After Reset:** 0x0
14	R	These bit in Interrupt Status Register are reserved. It will always return 0. **Value After Reset:** 0x0
13	RW	**BUSOWNER_UPDATED_STAT_EN** Bus owner Updated Status Enable. **Value After Reset:** 0x0
12	RW	**IBI_UPDATED_STAT_EN** IBI Status is updated. IBI Updated Status Enable This field is used in slave mode of operation. **Value After Reset:** 0x0
11	RW	**READ_REQ_RECV_STAT_EN** Read Request Received Status Enable This field is used in slave mode of operation. **Value After Reset:** 0x0
10	RW	**DEFSLV_STAT_EN** Define Slave CCC Received Status Enable. **Value After Reset:** 0x0
9	RW	**TRANSFER_ERR_STAT_EN** Transfer Error Status Enable. **Value After Reset:** 0x0
8	RW	**DYN_ADDR_ASSGN_STAT_EN** Dynamic Address Assigned Status Enable This field is used in slave mode of operation. **Value After Reset:** 0x0
7	R	**RSVD_7: This bit in Interrupt Status Enable register is reserved.** It will always return 0. **Value After Reset:** 0x0
6	RW	**CCC_UPDATED_STAT_EN** CCC Table Updated Status Enable. This field is used only in slave mode of operation. **Value After Reset:** 0x0

to next page

from previous page

5	RW	**TRANSFER_ABORT_STAT_EN** Transfer Abort Status Enable. This field is used only in master mode of operation. **Value After Reset:** 0x0
4	RW	**RESP_READY_STAT_INTR_EN** Response Queue Ready Status Enable. **Value After Reset:** 0x0
3	RW	**CMD_QUEUE_READY_STAT_EN** Command Queue Ready Status Enable. **Value After Reset:** 0x0
2	RW	**IBI_THLD_STAT_EN** IBI Buffer Threshold Status. This field is used only in master mode of operation. **Value After Reset:** 0x0
1	RW	**RX_THLD_STAT_EN** Receive Buffer Threshold Status Enable. **Value After Reset:** 0x0
0	RW	**TX_THLD_STAT_EN** Transmit Buffer Threshold Status. **Value After Reset:** 0x0

Offset: 44h **INTR_SINGAL_EN: Interrupt Signal Enable Register** **Init = 0**

Bit	R/W	Description
31:16	R	These bits in Interrupt Status Register are reserved. It will always return 0. **Value After Reset:** 0x0
15	RW	**BUS_RESET_DONE_SIGNAL_EN** Bus Reset Pattern Generation Done Signal Enable. This field is used only in Master mode of operation. **Value After Reset:** 0x0
14	R	These bit in Interrupt Status Register are reserved. It will always return 0. **Value After Reset:** 0x0
13	RW	**BUSOWNER_UPDATED_SIGNAL_EN** Bus owner Updated Signal Enable. **Value After Reset:** 0x0
12	RW	**IBI_UPDATED_SIGNAL_EN** IBI Updated Signal Enable This field is used in slave mode of operation. **Value After Reset:** 0x0

to next page

from previous page

11	RW	**READ_REQ_RECV_SIGNAL_EN** Read Request Received Signal Enable This field is used in slave mode of operation. **Value After Reset:** 0x0
10	RW	**DEFSLV_SIGNAL_EN** Define Slave CCC Received Signal Enable. **Value After Reset:** 0x0
9	RW	**TRANSFER_ERR_SIGNAL_EN** Transfer Error Signal Enable. **Value After Reset:** 0x0
8	RW	**DYN_ADDR_ASSGN_SIGNAL_EN** Dynamic Address Assigned Signal Enable This field is used in slave mode of operation. **Value After Reset:** 0x0
7	R	**RSVD_7: This bit in Interrupt Signal Enable register is reserved.** It will always return 0. **Value After Reset:** 0x0
6	RW	**CCC_UPDATED_SIGNAL_EN** CCC Table Updated Signal Enable. This field is used only in slave mode of operation. **Value After Reset:** 0x0
5	RW	**TRANSFER_ABORT_SIGNAL_EN** Transfer Abort Signal Enable. This field is used only in master mode of operation. **Value After Reset:** 0x0
4	RW	**RESP_READY_SIGNAL_INTR_EN** Response Queue Ready Signal Enable. **Value After Reset:** 0x0
3	RW	**CMD_QUEUE_READY_SIGNAL_EN** Command Queue Ready Signal Enable. **Value After Reset:** 0x0
2	RW	**IBI_THLD_SIGNAL_EN** IBI Buffer Threshold Signal. This field is used only in master mode of operation. **Value After Reset:** 0x0
1	RW	**RX_THLD_SIGNAL_EN** Receive Buffer Threshold Signal Enable. **Value After Reset:** 0x0
0	RW	**TX_THLD_SIGNAL_EN** Transmit Buffer Threshold Signal Enable. **Value After Reset:** 0x0

Offset: 48h	INTR_FORCE: Interrupt Force Enable Register	Init = 0
Bit	**R/W**	**Description**
31:16	W	These bits in Interrupt Status Register are reserved. It will always return 0. **Value After Reset:** 0x0
15	W	**BUS_RESET_DONE_FORCE_EN** Bus Reset Pattern Generation Done Force Enable. This field is used only in Master mode of operation. **Value After Reset:** 0x0
14	W	These bits in Interrupt Status Register are reserved. It will always return 0. **Value After Reset:** 0x0
13	W	**BUSOWNER_UPDATED_FORCE_EN** Bus owner Updated Force Enable. **Value After Reset:** 0x0
12	W	**IBI_UPDATED_FORCE_EN** IBI Updated Force Enable This field is used in slave mode of operation. **Value After Reset:** 0x0
11	W	**READ_REQ_RECV_FORCE_EN** Read Request Received Force Enable This field is used in slave mode of operation. **Value After Reset:** 0x0
10	W	**DEFSLV_FORCE_EN** Define Slave CCC Received Force Enable. **Value After Reset:** 0x0
9	W	**TRANSFER_ERR_FORCE_EN** Transfer Error Force Enable. **Value After Reset:** 0x0
8	W	**DYN_ADDR_ASSGN_FORCE_EN** Dynamic Address Assigned Force Enable This field is used in slave mode of operation. **Value After Reset:** 0x0
7	W	**RSVD_7: This bit in Interrupt Force Enable register is reserved.** It will always return 0. **Value After Reset:** 0x0
6	W	**CCC_UPDATED_FORCE_EN** CCC Table Updated Force Enable. This field is used only in slave mode of operation. **Value After Reset:** 0x0

to next page

from previous page

5	W	**TRANSFER_ABORT_FORCE_EN** Transfer Abort Force Enable. This field is used only in master mode of operation. **Value After Reset:** 0x0
4	W	**RESP_READY_FORCE_INTR_EN** Response Queue Ready Force Enable. **Value After Reset:** 0x0
3	W	**CMD_QUEUE_READY_FORCE_EN** Command Queue Ready Force Enable. **Value After Reset:** 0x0
2	W	**IBI_THLD_FORCE_EN** IBI Buffer Threshold Force. This field is used only in master mode of operation. **Value After Reset:** 0x0
1	W	**RX_THLD_FORCE_EN** Receive Buffer Threshold Force Enable. **Value After Reset:** 0x0
0	W	**TX_THLD_FORCE_EN** Transmit Buffer Threshold Force Enable. **Value After Reset:** 0x0

Offset: 4Ch **QUEUE_STATUS_LEVEL: QUEUE STATUS LEVEL Register** **Init = 0**

Bit	R/W	Description
31:19	R	These bits in Queue Status Level Register are reserved. It will always return 0. This field is used in master mode of operation. **Value After Reset:** 0x0
28:24	R	**IBI_STATUS_CNT** IBI Buffer Status Count. Contains the number of IBI status entries in the IBI Buffer. This field is used in master mode of operation. **Value After Reset:** 0x0
23:16	R	**IBI_BUF_BLR** IBI Buffer Level Value. Contains the number of valid entries in the IBI Buffer. This field is used in master mode of operation. **Value After Reset:** 0x0
15:8	R	**RESP_BUF_BLR** Response Buffer Level Value. Contains the number of valid data entries in the response Buffer. **Value After Reset:** 0x0

to next page

from previous page

7 :0	R	**CMD_QUEUE_EMPTY_LOC** Command Queue Empty Locations. Contains the number of empty locations in the command Buffer **Value After Reset:** 0x10

Offset: 50h **DATA_BUFFER_STATUS_LEVEL: DATA BUFFER STATUS LEVEL Register** **Init = 0**

Bit	R/W	Description
31:24	R	**RSVD_31_24: These bits in Data Buffer Level Register are reserved.** It will always return 0. **Value After Reset:** 0x0
23:16	R	**RX_BUF_BLR** Receive Buffer Level Value. Contains the number of valid data entries in the receive Buffer. **Value After Reset:** 0x0
15:8	R	**RSVD_15_8: These bits in Data Buffer Level Register are reserved.** It will always return 0. **Value After Reset:** 0x0
7 :0	R	**TX_BUF_EMPTY_LOC** Transmit Buffer Empty Level Value. Contains the number of empty locations in the transmit Buffer. **Value After Reset:** 0x40

Offset: 54h **PRESENT_STATE: PRESENT STATE Register** **Init = 0**

Bit	R/W	Description
31:29	R	**RSVD_28_31: These bits in Present State Register are reserved.** It will always return 0. **Value After Reset:** 0x0
28	R	**MASTER_IDLE** This field reflects whether the Master Controller is in Idle state or not. This bit is set when all the Queues(Command , Response, IBI) and Buffers(Transmit and Receive) are empty along with the Master State machine is in Idle state. Values: - 0x0 (MST_NOT_IDLE): Master Controller is not in IDLE State - 0x1 (MST_IDLE): Master Controller is in IDLE State. **Value After Reset:** 0x1
27:24	R	**CMD_TID** This field reflects the Transaction-ID of the current executing command. **Value After Reset:** 0x0
22:23	R	**RSVD_22_23** These bits in Present State Register are reserved. It will always return 0. **Value After Reset:** 0x0

to next page

from previous page

21:16	R	**CM_TFR_ST_STATUS** Current Master Transfer State Status. Indicates the state of current transfer currently executing by the DWC_mipi_i3c controller. - 6'h0: IDLE (Controller is Idle state, waiting for commands from application or Slave initated In-band Interrupt) - 6'h1: START Generation State. - 6'h2: RESTART Generation State. - 6'h3: STOP Generation State. - 6'h4: START Hold Generation for the Slave Initiated START State. - 6'h5: Broadcast Write Address Header(7'h7E,W) Generation State. - 6'h6: Broadcast Read Address Header(7'h7E,R) Generation State. - 6'h7: Dynamic Address Assignment State. - 6'h8: Slave Address Generation State. - 6'hB: CCC Byte Generation State. - 6'hC: HDR Command Generation State. - 6'hD: Write Data Transfer State. - 6'hE: Read Data Transfer State. - 6'hF: In-Band Interrupt(SIR) Read Data State. - 6'h10: In-Band Interrupt Auto-Disable State. - 6'h11: HDR-DDR CRC Data Generation/Receive State. - 6'h12: Clock Extension State. - 6'h13: Halt State. **Value After Reset:** 0x0
14:15	R	**RSVD_14_15: These bits in Present State Register are reserved.** It will always return 0. **Value After Reset:** 0x0

to next page

from previous page

13:8	R	**CM_TFR_STATUS** Transfer Type Status. Indicates the type of transfer currently executing by the I3C controller. - 6'h0: IDLE (Controller is in Idle state, waiting for commands from application or Slave initated In-band Interrupt) - 6'h1: Broadcast CCC Write Transfer. - 6'h2: Directed CCC Write Transfer. - 6'h3: Directed CCC Read Transfer. - 6'h4: ENTDAA Address Assignment Transfer. - 6'h5: SETDASA Address Assignment Transfer. - 6'h6: Private I3C SDR Write Transfer. - 6'h7: Private I3C SDR Read Transfer. - 6'h8: Private I2C SDR Write Transfer. - 6'h9: Private I2C SDR Read Transfer. - 6'hA: Private HDR Ternary Symbol(TS) Write Transfer. - 6'hB: Private HDR Ternary Symbol(TS) Read Transfer. - 6'hC: Private HDR Double-Data Rate(DDR) Write Transfer. - 6'hD: Private HDR Double-Data Rate(DDR) Read Transfer. - 6'hE: Servicing In-Band Interrupt Transfer. - 6'hF: Halt state (Controller is in Halt State, waiting for the application to resume through DEVICE_CTRL Register) In Slave mode of operation : - 4'h0: IDLE (Controller is in Idle state) - 4'h1: Hot-Join transfer state - 4'h2: IBI transfer state - 4'h3: Master write transfer ongoing - 4'h4: Read data prefetch state - 4'h5: Master read transfer ongoing - 4'h6: Slave controller in Halt State waiting for resume from application **Value After Reset:** 0x0
7 :3	R	**RSVD_3_7: These bits in Present State Register are reserved.** It will always return 0. **Value After Reset:** 0x0
2	R	**CURRENT_MASTER** This Bit is used to check whether the Master is Current Master or not. The Current Master is the Master that owns the SCL line. If this bit is set to 0, the Master is not Current Master and requires to request and the ownership before initiating any transfer on the line. If this bit is set to 1, the Master is the Current Master and can initate the transfers on the line. - 0: Master is not Current Master - 1: Master is Current Master **Values:** - 0x0 (NOT_BUS_OWNER): Master is not a Current Master - 0x1 (BUS_OWNER): Master is Current Master **Value After Reset:** 0x0 **Volatile:** true

to next page

from previous page

1	R	**SDA_LINE_SIGNAL_LEVEL** This bit is used to check the SDA line level to recover from errors and for debugging. This bit reflects the value of synchronized sda_in_a signal. **Value After Reset:** 0x1 **Volatile:** true
0	R	**SCL_LINE_SIGNAL_LEVEL** This bit is used to check the SDA line level to recover from errors and for debugging. This bit reflects the value of synchronized scl_in_a signal. **Value After Reset:** 0x1 **Volatile:** true

Offset: 58h **I3CD058: Device Operating Status Register** **Init = 0**

Bit	R/W	Description
31:14	R	**RSVD** These bits in Device Operating Status Register are reserved. It will always return 0. **Value After Reset:** 0x0
13	R	**FRAME_ERROR** Buffer not available This bit is set when private write request from Master has frame error in HDR-DDR/HDR-TSP/TSL mode. This is cleared only after Master reads the device status through GET-STATUS_CCC. **Value After Reset:** 0x0 **Volatile:** true
12	R	**BUFFER_NOT_AVAIL** Buffer not available This bit is set when private write request from Master is NACKED because of RX buffer not having RX_BUF_THLD number of empty locations or Response buffer is full. In SDR mode of operation this will be cleared when the Master issues GET_STATUS CCC or **Value After Reset:** 0x0 **Volatile:** true
11	R	**DATA_NOT_READY** Data not ready This bit is set when private read request from Master is NACKED because of any of the following conditions - Command FIFO Empty. - Transmit FIFO threshold is not met. - Response FIFO Full. This will be cleared when the Master issues GET_STATUS CCC or upon successful completion of the subsequent read transfer. **Value After Reset:** 0x0 **Volatile:** true

to next page

from previous page

10	R	**OVERFLOW_ERR** Overflow Error Overflow error condition detected during master write transfer. This will be cleared only after master reads the Device Status through GETSTATUS CCC. **Value After Reset:** 0x0 **Volatile:** true
9	R	**SLAVE_BUSY** Slave Busy This bit is set if any change is made by the current master in to MRL register or occurance of any error. It is cleared after slave application resumes the slave operation by writing 1'b1 in RESUME field of Device Control Register. **Value After Reset:** 0x0 **Volatile:** true
10	R	**UNDER_ERR** Underflow Error Under Flow Error during private master read transfer. This bit is set if slave controller terminates a read transfer because of **Value After Reset:** 0x0 **Volatile:** true
7 :6	R	**ACTIVITY_MODE** Activity Mode This field reflects the input port signal act_mode. **Value After Reset:** 0x0 **Volatile:** true
7 :6	R	**PROTOCOL_ERR** Protocol Error This bit will be set when the slave controller encouters a Parity/CRC error during write data transfer. **Value After Reset:** 0x0 **Volatile:** true
4	R	**RSVD_4: This bit in Device Operating Status register is reserved.** It will always return 0. **Value After Reset:** 0x0 **Volatile:** true
3 :0	R	**PENDING_INTR** Pending Interrupt This field reflects the value driven on pending_int input port. **Value After Reset:** 0x0 **Volatile:** true

Offset: 5Ch DEVICE_ADDR_TABLE_POINTER: Pointer for Device Address Table Registers Init = 0

Bit	R/W	Description
31:16	R	**DEV_ADDR_TABLE_DEPTH** Depth of Device Address Table **Value After Reset:** 0x8

to next page

from previous page

15:0	R	**P_DEV_ADDR_TABLE_START_ADDR** Start Address of Device Address Table **Value After Reset:** 0x280

Offset: 60h **DEV_CHAR_TABLE_POINTER: Pointer for Device Characteristics Table** **Init = 0**

Bit	R/W	Description
31:22	R	**RSVD_31_24** These bits in Device Characteristics Table Pointer Register are reserved. It will always return 0. **Value After Reset:** 0x0
21:19	RW	**PRESENT_DEV_CHAR_TABLE_INDX** Current index of Device Characteristics Table. This field returns the current location of Device Characteristics Table index. Initially, this index points to 0. Once the complete characteristics information of a Slave device is written into Device Characteristics Table during ENTDAA, this index increments by 1. The first winning device information is stored in Device Characteristics Table index 0, the second winning device information in index 1, and so on. If required, this index can be used to override the location, where characteristic information of Slave devices on the I3C bus are written during ENTDAA. Hence, this field is useful only if the device is Current Master. During DEFSLV CCC, the index always starts from 0. In Non-current Master, this field is always read-only. **Value After Reset:** 0x0
18:12	R	**DEV_CHAR_TABLE_DEPTH** Depth of Device Characteristics Table **Value After Reset:** 0x20
11:0	R	**P_DEV_CHAR_TABLE_START_ADDR** Start Address of Device Characteristics Table. **Value After Reset:** 0x200

Offset: 6Ch **VENDOR_SPECIFIC_REG_POINTER: Pointer for Vendor specific Registers.** **Init = 0**

Bit	R/W	Description
31:16	R	**RSVD** These bits in Vendor Specific Pointer Register are reserved. It will always return 0.
15:0	R	**P_VENDOR_REG_START_ADDR** Start Address of Vendor specific registers. **Value After Reset:** 0xb0

Offset: 70h **SLV_MIPI_PID_VALUE:Provisional ID Register.** **Init = 0**

Bit	R/W	Description
31:16	R	**RSVD** **Value After Reset:** 0x0

to next page

from previous page

15:1	RW	**SLV_MIPI_MFG_ID** Specifies the MIPI Manufacturer ID. (PID[47:33]). **Value After Reset:** 0x0 **Volatile:** true
00	RW	**SLV_PID_DCR** Specifies the Provisional ID Type Selector (PID[32]). (1'b1: Random Value, 1'b0: Vendor Fixed Value) **Value After Reset:** 0x0

Offset: 74h **SLV_PID_VALUE:Provisional ID Register.** **Init = 0**

Bit	R/W	Description
31:16	RW	**SLV_PART_ID** Specifies the Part ID of I3C device (PID[31:16]) **Value After Reset:** 0x0
15:12	RW	**SLV_INST_ID** This field is used to program the instance ID of the Slave. The reset value of this register is taken from input port 'inst_id'. **Value After Reset:** 0x0 **Volatile:** true
11:0	RW	**SLV_PID_DCR** Specifies the additional 12-bit ID of I3C device (PID[11:0]). **Value After Reset:** 0x0

Offset: 78h **SLV_CHAR_CTRL: I3C Slave Characteristic Register** **Init = 0**

Bit	R/W	Description
31:24	R	**RSVD** These bits in I3C Slave Characteristic Register are reserved. User must ignore the values of these bits. **Value After Reset:** 0x0
23:16	R	**HDR_CAP** These bits in I3C Slave Characteristic Register are reserved. User must ignore the values of these bits. I3C Device HDR Capability Register Value. HDR_CAP[2] - HDR Mode 2 HDR_CAP[1] - HDR Mode 1 HDR_CAP[1] - HDR Mode 0 Others - Reserved **Value After Reset:** 0x7
15:8	R	**DCR** I3C Device Characteristic Value. **Value After Reset:** 0x0

to next page

from previous page

7 :0	R	**BCR** I3C Bus Characteristic Value. BCR[7] - Device Role [1] (0) BCR[6] - Device Role [2] (0) BCR[5] - HDR Capable BCR[4] - Bridge Identifier BCR[3] - Offline Capable BCR[2] - IBI Payload BCR[1] - IBI Request Capable BCR[0] - Max Data Speed Limitation **Value After Reset:** 0x66

Offset: 7Ch SLV_MAX_LEN: I3C Max Write/Read Length Register Init = 0

Bit	R/W	Description
31:16	R	**MRL** I3C Device Max Read Length. **Value After Reset:** 0xff
15:0	R	**MWL** I3C Device Max Read Length. **Value After Reset:** 0xff

Offset: 80h MAX_READ_TURNAROUND: MXDS Maximum Read Turnaround Time Register Init = 0

Bit	R/W	Description
31:24	R	These bits in Maximum Read Turnaround Time Register are reserved. User must ignore the values of these bits.
23:0	R	**MXDS_MAX_RD_TURN** Specifies the maximum read turnaround time (in microseconds (us)) of I3C Slave. These bits in Maximum Read Turnaround Time Register are reserved. User must ignore the values of these bits. **Value After Reset:** 0x0

Offset: 84h MAX_DATA_SPEED: MXDS Maximum Data Speed Register Init = 0

Bit	R/W	Description
31:19	R	These bits in Maximum Read Turnaround Time Register are reserved. User must ignore the values of these bits.

to next page

from previous page

18:16	RW	Specifies the clock to data turnaround time (Tsco parameter) of I3C device - 0 : 8ns - 1 : 9ns - 2 : 10ns - 3 : 11ns - 4 : 12ns **Value After Reset:** 0x0
15:11	R	RSVD_15_11: These bits in MXDS Maximum Data Speed Register are reserved. User must ignore the values of these bits
10:8	RW	**MXDS_MAX_RD_SPEED** Specifies the Maximum Sustained Data Rate for non-CCC messages sent by DWC_mipi_i3c Slave Device to Master Device - 0 : 12.5MHz - 1 : 8MHZ - 2 : 6MHz - 3 : 4MHz - 4 : 2MHz **Value After Reset:** 0x0
7 :3	R	**RSVD_7_3: These bits in MXDS Maximum Data Speed Register are reserved.** User must ignore the values of these bits.
2 :0	RW	**MXDS_MAX_WR_SPEED** Specifies the Maximum Sustained Data Rate for non-CCC messages sent by Master Device to DWC_mipi_i3c Slave device - 0 : 12.5MHz - 1 : 8MHZ - 2 : 6MHz - 3 : 4MHz - 4 : 2MHz **Value After Reset:** 0x0

Offset: 8Ch **SLV_INTR_REQ: Slave Interrupt Request Register** **Init = 0**

Bit	R/W	Description
31:24	R	**RSVD_31_24: These bits in Slave Interrupt Request Register are reserved.** It will always return 0. **Value After Reset:** 0x0
23:10	R	**RSVD_23_10: These bits in Slave Interrupt Request Register are reserved.** It will always return 0. **Value After Reset:** 0x0

to next page

from previous page

9 :8	R	**IBI_STS** IBI Completion Status - 2'b00 : Reserved - 2'b01 : Successful - 2'b10 : Reserved - 2'b11 : Not Attempted because of SIR event got disabled or dynamic address is not valid **Value After Reset:** 0x0
7 :4	R	**RSVD_7_4: These bits in Slave Interrupt Request Register are reserved.** It will always return 0. **Value After Reset:** 0x0
3	RW	**MIR** Master Interrupt Request When set, the Secondary Master controller initiates Master Request on I3C bus. Once the Master Request is ACK'ed by the current master the secondary master clears this bit automatically. If NACK is received for MIR, the controller will reattempt the MIR upon the next START or Bus Available time. **Value After Reset:** 0x0 **Volatile:** true
2 :1	RW	**SIR_CTRL** Slave Interrupt Request Control - 2'b00 : Initiate SIR-IBI - 2'b01 : Reserved - 2'b10 : Reserved - 2'b11 : Reserved **Value After Reset:** 0x0
0	RW	**SIR** Slave Interrupt Request When set, the slave controller initiates the SIR on the I3C bus based on the trigger control set in the SIR_CTRL field of the Slave Interrupt Request register. Once the SIR is successfully transferred to the current master (ACK received), the slave controller clears this bit automatically. If the NACK response is received for the SIR, the controller will reattempt the SIR in the next START or Bus Free time. The slave controller will clear this bit and will not attempt to initiate the SIR if the SIR_EN is not set in the Slave Event Control Register. **Value After Reset:** 0x0

Offset: 90h **SLV_TSX_SYMBL_TIMING: TSP/TSL Symbol Timing Register** **Init = 0**

Bit	R/W	Description
31:6	R	**RSVD_31_6: These bits in TSX symbol timing register is reserved.** It will always return 0.
5 :0	RW	**SLV_TSX_SYMBL_CNT** TSP/TSL Symbol Count Value. This field is used by the controller in Slave mode to calculate the symbol duration in TSP/TSL mode while sending data. THe symbol duration will be count times hdr_tx_clk period. **Value After Reset:** 0x3f

Offset: B0h	DEVICE_CTRL_EXTENDED: Device Control Extended Register	Init = 0
Bit	**R/W**	**Description**
31:4	R	RSVD: These bits in Device Control Extended Register is reserved. It will always return 0.
3	RW	**REQMST_ACK_CTRL** In Slave mode of operation, this bit serves as a control to ACK/NACK GETACCMST CCC from current master. - 0: ACK GETACCMST CCC - 1: NACK GETACCMST CCC **Values:** - 0x0 (ACK): ACK - 0x1 (NACK): NACK **Value After Reset:** 0x0
2	R	**RSVD_2: These bits in Device Control Extended Register is reserved.** It will always return 0.
1 :0	RW	This bit is used to select the Device Operation Mode. - 0: Master - 1: Slave - 2: Reserved - 3: Reserved NOTE: Only Master role is supported in this release. **Values:** - 0x0 (MASTER): Master - 0x1 (SLAVE): Slave **Value After Reset:** 0x0 **Volatile:** true

Offset: B4h	SCL_I3C_OD_TIMING: SCL I3C Open Drain Timing Register	Init = 0
Bit	**R/W**	**Description**
31:24	R	**RSVD_31_24: These bits in SCL I3C OD timing register are reserved.** It will always return 0.
23:16	RW	**I3C_OD_HCNT** I3C Open Drain High Count. SCL open-drain high count (I3C) for I3C transfers targeted to I3C devices. **Value After Reset:** 0xA
15:8	R	**RSVD_15_8: These bits in SCL I3C OD timing register are reserved.** It will always return 0.
7 :0	RW	**I3C_OD_LCNT** I3C Open Drain Low Count. SCL open-drain low count (I3C) for I3C transfers targeted to I3C devices. **Value After Reset:** 0x10

Offset: B8h	SCL_I3C_PP_TIMING: SCL I3C Push Pull Timing Register	Init = 0

Bit	R/W	Description
31:24	R	**RSVD_31_24: These bits in SCL I3C Push Pull Timing Register are reserved.** It will always return 0.
23:16	RW	**I3C_PP_HCNT** I3C Push Pull High Count. SCL push-pull high count (I3C) for I3C transfers targeted to I3C devices. **Value After Reset:** 0xA
15:8	R	**RSVD_15_8: These bits in SCL I3C Push Pull Timing Register are reserved.** It will always return 0.
7 :0	RW	**I3C_PP_LCNT** I3C Push Pull Low Count. SCL push-pull low count (I3C) for I3C transfers targeted to I3C devices. **Value After Reset:** 0xA

Offset: BCh	Init = SCL_I2C_FM_TIMING: SCL I2C Fast Mode Timing Register

Bit	R/W	Description
0 31:16	RW	**I2C_FM_HCNT** I2C Fast Mode High Count The SCL open-drain high count timing for I2C fast mode transfers. **Value After Reset:** 0x10
15:0	RW	**I2C_FM_LCNT** I2C Fast Mode Low Count The SCL open-drain high count timing for I2C fast mode transfers. **Value After Reset:** 0x10

Offset: C0h	SCL_I2C_FMP_TIMING: SCL I2C Fast Mode Plus Timing Register	Init = 0

Bit	R/W	Description
31:24	R	**RSVD_31_24: These bits in SCL I2C FM plus timing register are reserved.** It will always return 0.
23:16	RW	**I2C_FMP_HCNT** I2C Fast Mode Plus High Count The SCL open-drain high count timing for I2C fast mode plus transfers. **Value After Reset:** 0x10
15:0	RW	**I2C_FMP_LCNT** I2C Fast Mode Plus Low Count The SCL open-drain low count timing for I2C fast mode plus transfers. **Value After Reset:** 0x10

Offset: C8h	SCL_EXT_LCNT_TIMING: SCL Extended Low Count Timing Register	Init = 0

Bit	R/W	Description
31:24	RW	**I3C_EXT_LCNT_4** I3C Extended Low Count Register 4 SDR4 uses this register field for data transfer. **Value After Reset:** 0x20
23:16	RW	**I3C_EXT_LCNT_3** I3C Extended Low Count Register 3 SDR3 uses this register field for data transfer. **Value After Reset:** 0x20
15:8	RW	**I3C_EXT_LCNT_2** I3C Extended Low Count Register 2 SDR2 uses this register field for data transfer. **Value After Reset:** 0x20
7 :0	RW	**I3C_EXT_LCNT_1** I3C Extended Low Count Register 1 SDR1 uses this register field for data transfer. **Value After Reset:** 0x20

Offset: CCh SCL_EXT_TERMN_LCNT_TIMING: SCL Termination Bit Low count Timing Register Init = 0

Bit	R/W	Description
31:20	R	**RSVD** These bits in SCL Termination Bit Low count timing register is reserved. It will always return 0.
19:16	RW	**I3C_TS_SKEW_CNT** I3C HDR Ternary Skew Count. Ternary Skew Count in terms of core_clks which is used for HDR Ternary Bus Turn-Around Detection in Master Mode. **Value After Reset:** 0x3
15:4	R	**RSVD** These bits in SCL Termination Bit Low count timing register is reserved. It will always return 0.
3 :0	RW	**I3C_EXT_TERMN_LCNT** I3C Read Termination Bit Low count. Extended I3C Read Termination Bit low count for I3C Read transfers. Effective Termination-Bit Low Period is derived based on the SDR speed as shown below - SDR0 speed: I3C_PP_LCNT + I3C_EXT_TERMN_LCNT - SDR1 speed: I3C_EXT_LCNT_1 + I3C_EXT_TERMN_LCNT - SDR2 speed: I3C_EXT_LCNT_2 + I3C_EXT_TERMN_LCNT - SDR3 speed: I3C_EXT_LCNT_3 + I3C_EXT_TERMN_LCNT - SDR4 speed: I3C_EXT_LCNT_4 + I3C_EXT_TERMN_LCNT **Value After Reset:** 0x0

Offset: D0h SDA_HOLD_SWITCH_DLY_TIMING: SDA Hold and Mode Switch Delay Timing Register Init = 0

Bit	R/W	Description
31:19	RW	**RSVD** **Value After Reset:** 0x0
18:16	RW	**SDA_TX_HOLD** This field controls the hold time (in term of the core clock period) of the transmit data (SDA) with respect to the SCL edge in FM FM+ SDR and DDR speed mode of operations. This field is not applicable for the ternary speed modes. The valid values are 1 to 7. Others are Reserved. **Value After Reset:** 0x1
15: 0	RW	**RSVD** **Value After Reset:** 0x0

Offset: D4h BUS_FREE_TIMING: Bus Free Timing Register Init = 0

Bit	R/W	Description
31:16	RW	**I3C_IBI_FREE** This register field is used only in Slave mode of operation Bus Available Count Value. This field is used by the Slave/Non-current Master to initiate an IBI after STOP condition. **Value After Reset:** 0x20
15:0	RW	**I3C_MST_FREE** This register field is used only in Master mode of operation I3C Master Free Count Value. In Pure Bus System, this field represents tCAS parameter. In Mixed Bus system, this field is expected to be programmed to tLOW of I2C Timing. **Value After Reset:** 0x20

Offset: D8h BUS_IDLE_TIMING: Bus Idle Timing Register Init = 0

Bit	R/W	Description
31:20	R	**RSVD** These bits in Bus Idle Timing Register is reserved. It will always return 0.
19:0	RW	**BUS_IDLE_TIME** Bus Idle Count Value. This field is used by the controller in Slave or Non-Current Master mode to initiate Hot-Join request if the dynamic address is not valid. **Value After Reset:** 0x20

Offset: DCh SCL_LOW_MST_EXT_TIMEOUT: SCL Low Master Extended Timeout Register Init = 0

Bit	R/W	Description
31:26	R	**RSVD** These bits in Bus Idle Timing Register is reserved. It will always return 0.

to next page

from previous page

25:0	RW	**SCL_LOW_MST_TIMEOUT_COUNT** This count defines the number of core clock periods to count for generation of the SCL Low Bus Reset Pattern. **Value After Reset:** 0x3567e0

Offset: E0h **I3C_VER_ID: I3C Version ID Register** **Init = 0**

Bit	R/W	Description
31:0	R	Current release number This field indicates the Synopsys DesignWare Cores DWC_mipi_i3c current release number that is read by an application. For example, release number ”1.00a” is represented in ASCII as 0x313030. Lower 8 bits read from this register can be ignored by the application. An application reading this register along with the I3C_VER_TYPE register, gathers details of the current release. **Value After Reset:** 0x3130302a

Offset: E4h **I3C_VER_TYPE: I3C Version TYPE Register** **Init = 0**

Bit	R/W	Description
31:0	R	**I3C_VER_TYPE** Current release type This field indicates the Synopsys DesignWare Cores DWC_mipi_i3c current release type that is read by an application. For example, release type ”ga” is represented in ASCII as 0x6761 and ”ea” is represented as 0x6561. Lower 16 bits read from this register can be ignored by the application if release type is ”ga”. If release type is ”ea” the lower 16 bits represents the ”ea” release version. An application reading this register along with the I3C_VER_ID register, gathers details of the current release. **Value After Reset:** 0x6c633033

Offset: E8h **QUEUE_SIZE_CAPABILITY: I3C Queue Size Capability Register** **Init = 0**

Bit	R/W	Description
31:20	R	These bits in Component Parameter Register 1 are reserved. It will always return 0. **Value After Reset:** 0x0
19:16	R	**IBI_BUF_SIZE** IBI Queue Size This field reflects the configured IBI Queue size (in DWORDS) in Encoded Values. Values: - 0x0: 2 DWORDS - 0x1: 4 DWORDS - 0x2: 8 DWORDS - 0x3: 16 DWORDS Value After Reset: 0x3

to next page

from previous page

15:12	R	**RESP_BUF_SIZE** Response Queue Size This field reflects the configured Response Queue size (in DWORDS) in Encoded Values. Values: - 0x0: 2 DWORDS - 0x1: 4 DWORDS - 0x2: 8 DWORDS - 0x3: 16 DWORDS Value After Reset: 0x2
11:8	R	**CMD_BUF_SIZE** Command Queue Size This field reflects the configured Command Queue size (in DWORDS) in Encoded Values. Values: - 0x0: 2 DWORDS - 0x1: 4 DWORDS - 0x2: 8 DWORDS - 0x3: 16 DWORDS Value After Reset: 0x3
7 :4	R	**RX_BUF_SIZE** Recieve Data Buffer Size This field reflects the configured Recieve Buffer size (in DWORDS) in Encoded Values. Values: - 0x0: 2 DWORDS - 0x1: 4 DWORDS - 0x2: 8 DWORDS - 0x3: 16 DWORDS - 0x5: 64 DWORDS Value After Reset: 0x5
3 :0	R	**TX_BUF_SIZE** Transmit Data Buffer Size This field reflects the configured Transmit Buffer size (in DWORDS) in Encoded Values. Values: - 0x0: 2 DWORDS - 0x1: 4 DWORDS - 0x2: 8 DWORDS - 0x3: 16 DWORDS - 0x5: 64 DWORDS Value After Reset: 0x5

I3CD200: Device Characteristic Table Location-1 of Device1 (Master Mode)		
Offset: 200h		**Init = 0**
Bit	**Attr.**	**Description**
31:0	R	**LSB_PROVISIONAL_ID** The LSB 32-bit value of Provisional-ID **Value After Reset:** 0x0 **Testable:** untestable **Volatile:** true
I3CD200: Device Characteristic Table Location of Device1 (Secondary Master Mode))		
0 31:24	R	**STATIC_ADDR** The Static Addr of Device1 **Value After Reset:** 0x0 **Testable:** untestable **Volatile:** true
23:16	R	**BCR_TYPE** The BCR TYPE of Device1 **Value After Reset:** 0x0 **Testable:** untestable **Volatile:** true
15:8	R	**DCR_TYPE** The DCR TYPE of Device1 **Value After Reset:** 0x0 **Testable:** untestable **Volatile:** true
7 :0	R	**DYNAMIC_ADDR** The Dynamic Addr of Device1 **Value After Reset:** 0x0 **Testable:** untestable **Volatile:** true

I3CD204: Device Characteristic Table Location-2 of Device1 (Master Mode)		
Offset: 204h		**Init = 0**
Bit	**Attr.**	**Description**
31:16	R	**RSVD** These bits in Device Characteristic Register -2 are reserved. User must ignore the values of these bits. **Testable:** untestable **Volatile:** true
15:0	R	**MSB_PROVISIONAL_ID** The MSB 16-bit value of Provisional-ID **Value After Reset:** 0x0 **Testable:** untestable **Volatile:** true
I3CD204: Device Characteristic Table Location of Device2 (Secondary Master Mode))		

to next page

from previous page

0 31:24	R	**STATIC_ADDR** The Static Addr of Device2 **Testable:** untestable **Volatile:** true
23:16	R	**BCR_TYPE** The BCR TYPE of Device2 **Testable:** untestable **Volatile:** true
15:8	R	**DCR_TYPE** The DCR TYPE of Device2 **Testable:** untestable **Volatile:** true
7 :0	R	**DYNAMIC_ADDR** The Dynamic Addr of Device2 **Testable:** untestable **Volatile:** true

I3CD208: Device Characteristic Table Location-3 of Device1 (Master Mode)

Offset: 208h **Init = 0**

Bit	Attr.	Description
31:16	R	**RSVD** These bits in Device Characteristic Register -3 are reserved. User must ignore the values of these bits. **Testable:** untestable **Volatile:** true
15:8	R	**BCR** Bus Characteristic Value **Testable:** untestable **Volatile:** true
7 :0	R	**DCR** Device Characteristic Value **Testable:** untestable **Volatile:** true
I3CD208: Device Characteristic Table Location of Device3 (Secondary Master Mode))		
0 31:24	R	**STATIC_ADDR** The Static Addr of Device3 **Testable:** untestable **Volatile:** true
23:16	R	**BCR_TYPE** The BCR TYPE of Device3 **Testable:** untestable **Volatile:** true

to next page

from previous page

15:8	R	**DCR_TYPE** The DCR TYPE of Device3 **Testable:** untestable **Volatile:** true
7 :0	R	**DYNAMIC_ADDR** The Dynamic Addr of Device3 **Testable:** untestable **Volatile:** true

I3CD20C: Device Characteristic Table Location-4 of Device1 (Master Mode)

Offset: 20Ch **Init = 0**

Bit	Attr.	Description
31:8	R	**RSVD** These bits in Device Characteristic Register -4 are reserved. User must ignore the values of these bits. **Testable:** untestable **Volatile:** true
7 :0	R	**DEV_DYNAMIC_ADDR** Device Dynamic Address assigned. **Testable:** untestable **Volatile:** ture
I3CD20C: Device Characteristic Table Location of Device4 (Secondary Master Mode))		
0 31:24	R	**STATIC_ADDR** The Static Addr of Device4 **Testable:** untestable **Volatile:** ture
23:16	R	**BCR_TYPE** The BCR TYPE of Device4 **Testable:** untestable **Volatile:** ture
15:8	R	**DCR_TYPE** The DCR TYPE of Device4 **Testable:** untestable **Volatile:** ture
7 :0	R	**DYNAMIC_ADDR** The Dynamic Addr of Device4 **Testable:** untestable **Volatile:** ture

I3CD210: Device Characteristic Table Location-1 of Device2 (Master Mode)		
Offset: 210h		**Init = 0**
Bit	**Attr.**	**Description**
31:0	R	**LSB_PROVISIONAL_ID** The LSB 32-bit value of Provisional-ID **Testable:** untestable **Volatile:** true
I3CD210: Device Characteristic Table Location of Device5 (Secondary Master Mode))		
0 31:24	R	**STATIC_ADDR** The Static Addr of Device5 **Testable:** untestable **Volatile:** true
23:16	R	**BCR_TYPE** The BCR TYPE of Device5 **Testable:** untestable **Volatile:** true
15:8	R	**DCR_TYPE** The DCR TYPE of Device5 **Testable:** untestable **Volatile:** true
7 :0	R	**DYNAMIC_ADDR** The Dynamic Addr of Device5 **Testable:** untestable **Volatile:** true

I3CD214: Device Characteristic Table Location-2 of Device2 (Master Mode)		
Offset: 214h		**Init = 0**
Bit	**Attr.**	**Description**
31:16	R	**RSVD** These bits in Device Characteristic Register -2 are reserved. User must ignore the values of these bits. **Testable:** untestable **Volatile:** true
15:0	R	**MSB_PROVISIONAL_ID** The MSB 16-bit value of Provisional-ID **Testable:** untestable **Volatile:** true
I3CD214: Device Characteristic Table Location of Device6 (Secondary Master Mode))		
0 31:24	R	**STATIC_ADDR** The Static Addr of Device6 **Testable:** untestable **Volatile:** true

to next page

from previous page

23:16	R	**BCR_TYPE** The BCR TYPE of Device6 **Testable:** untestable **Volatile:** true
15:8	R	**DCR_TYPE** The DCR TYPE of Device6 **Testable:** untestable **Volatile:** true
7 :0	R	**DYNAMIC_ADDR** The Dynamic Addr of Device6 **Testable:** untestable **Volatile:** true

I3CD218: Device Characteristic Table Location-3 of Device2 (Master Mode)

Offset: 218h **Init = 0**

Bit	Attr.	Description
31:16	R	**RSVD** These bits in Device Characteristic Register -3 are reserved. User must ignore the values of these bits. **Testable:** untestable **Volatile:** true
15:8	R	**BCR** Bus Characteristic Value **Testable:** untestable **Volatile:** true
7 :0	R	**DCR** Device Characteristic Value **Testable:** untestable **Volatile:** true
I3CD218: Device Characteristic Table Location of Device7 (Secondary Master Mode))		
0 31:24	R	**STATIC_ADDR** The Static Addr of Device7 **Testable:** untestable **Volatile:** true
23:16	R	**BCR_TYPE** The BCR TYPE of Device7 **Testable:** untestable **Volatile:** true
15:8	R	**DCR_TYPE** The DCR TYPE of Device7 **Testable:** untestable **Volatile:** true

to next page

from previous page

7 :0	R	**DYNAMIC_ADDR** The Dynamic Addr of Device7 **Testable:** untestable **Volatile:** true

I3CD21C: Device Characteristic Table Location-4 of Device2 (Master Mode)

Offset: 21Ch **Init = 0**

Bit	Attr.	Description
31:8	R	**RSVD** These bits in Device Characteristic Register -4 are reserved. User must ignore the values of these bits. **Testable:** untestable **Volatile:** true
7 :0	R	**DEV_DYNAMIC_ADDR** Device Dynamic Address assigned. **Testable:** untestable **Volatile:** true
I3CD21C: Device Characteristic Table Location of Device8 (Secondary Master Mode))		
0 31:24	R	**STATIC_ADDR** The Static Addr of Device8 **Testable:** untestable **Volatile:** true
23:16	R	**BCR_TYPE** The BCR TYPE of Device8 **Testable:** untestable **Volatile:** true
15:8	R	**DCR_TYPE** The DCR TYPE of Device8 **Testable:** untestable **Volatile:** true
7 :0	R	**DYNAMIC_ADDR** The Dynamic Addr of Device8 **Testable:** untestable **Volatile:** true

I3CD220: Device Characteristic Table Location-1 of Device3 (Master Mode)		
Offset: 220h		**Init = 0**
Bit	**Attr.**	**Description**
31:0	R	**LSB_PROVISIONAL_ID** The LSB 32-bit value of Provisional-ID **Testable:** untestable **Volatile:** true
I3CD220: Device Characteristic Table Location of Device9 (Secondary Master Mode))		
0 31:24	R	**STATIC_ADDR** The Static Addr of Device9 **Testable:** untestable **Volatile:** true
23:16	R	**BCR_TYPE** The BCR TYPE of Device9 **Testable:** untestable **Volatile:** true
15:8	R	**DCR_TYPE** The DCR TYPE of Device9 **Testable:** untestable **Volatile:** true
7 :0	R	**DYNAMIC_ADDR** The Dynamic Addr of Device9 **Testable:** untestable **Volatile:** true

I3CD224: Device Characteristic Table Location-2 of Device3 (Master Mode)		
Offset: 224h		**Init = 0**
Bit	**Attr.**	**Description**
31:16	R	**RSVD** These bits in Device Characteristic Register -2 are reserved. User must ignore the values of these bits. **Testable:** untestable **Volatile:** true
15:0	R	**MSB_PROVISIONAL_ID** The MSB 16-bit value of Provisional-ID **Testable:** untestable **Volatile:** true
I3CD224: Device Characteristic Table Location of Device10 (Secondary Master Mode))		
0 31:24	R	**STATIC_ADDR** The Static Addr of Device10 **Testable:** untestable **Volatile:** true

to next page

from previous page

23:16	R	**BCR_TYPE** The BCR TYPE of Device10 **Testable:** untestable **Volatile:** true
15:8	R	**DCR_TYPE** The DCR TYPE of Device10 **Testable:** untestable **Volatile:** true
7 :0	R	**DYNAMIC_ADDR** The Dynamic Addr of Device10 **Testable:** untestable **Volatile:** true

I3CD228: Device Characteristic Table Location-3 of Device3 (Master Mode)

Offset: 228h **Init = 0**

Bit	Attr.	Description
31:16	R	**RSVD** These bits in Device Characteristic Register -3 are reserved. User must ignore the values of these bits. **Testable:** untestable **Volatile:** true
15:8	R	**BCR** Bus Characteristic Value **Testable:** untestable **Volatile:** true
7 :0	R	**DCR** Device Characteristic Value **Testable:** untestable **Volatile:** true
I3CD228: Device Characteristic Table Location of Device11 (Secondary Master Mode))		
0 31:24	R	**STATIC_ADDR** The Static Addr of Device11 **Testable:** untestable **Volatile:** true
23:16	R	**BCR_TYPE** The BCR TYPE of Device11 **Testable:** untestable **Volatile:** true
15:8	R	**DCR_TYPE** The DCR TYPE of Device11 **Testable:** untestable **Volatile:** true

to next page

from previous page

7 :0	R	**DYNAMIC_ADDR** The Dynamic Addr of Device11 **Testable:** untestable **Volatile:** true

I3CD22C: Device Characteristic Table Location-4 of Device3 (Master Mode)

Offset: 22Ch **Init = 0**

Bit	Attr.	Description
31:8	R	**RSVD** These bits in Device Characteristic Register -4 are reserved. User must ignore the values of these bits. **Testable:** untestable **Volatile:** true
7 :0	R	**DEV_DYNAMIC_ADDR** Device Dynamic Address assigned. **Testable:** untestable **Volatile:** true
I3CD22C: Device Characteristic Table Location of Device12 (Secondary Master Mode))		
0 31:24	R	**STATIC_ADDR** The Static Addr of Device12 **Testable:** untestable **Volatile:** true
23:16	R	**BCR_TYPE** The BCR TYPE of Device12 **Testable:** untestable **Volatile:** true
15:8	R	**DCR_TYPE** The DCR TYPE of Device12 **Testable:** untestable **Volatile:** true
7 :0	R	**DYNAMIC_ADDR** The Dynamic Addr of Device12 **Testable:** untestable **Volatile:** true

I3CD230: Device Characteristic Table Location-1 of Device4 (Master Mode)		
Offset: 230h		**Init = 0**
Bit	**Attr.**	**Description**
31:0	R	**LSB_PROVISIONAL_ID** The LSB 32-bit value of Provisional-ID **Testable:** untestable **Volatile:** true
I3CD230: Device Characteristic Table Location of Device13 (Secondary Master Mode))		
0 31:24	R	**STATIC_ADDR** The Static Addr of Device13 **Testable:** untestable **Volatile:** true
23:16	R	**BCR_TYPE** The BCR TYPE of Device13 **Testable:** untestable **Volatile:** true
15:8	R	**DCR_TYPE** The DCR TYPE of Device13 **Testable:** untestable **Volatile:** true
7 :0	R	**DYNAMIC_ADDR** The Dynamic Addr of Device13 **Testable:** untestable **Volatile:** true

I3CD234: Device Characteristic Table Location-2 of Device4 (Master Mode)		
Offset: 234h		**Init = 0**
Bit	**Attr.**	**Description**
31:16	R	**RSVD** These bits in Device Characteristic Register -2 are reserved. User must ignore the values of these bits. **Testable:** untestable **Volatile:** true
15:0	R	**MSB_PROVISIONAL_ID** The MSB 16-bit value of Provisional-ID **Testable:** untestable **Volatile:** true
I3CD234: Device Characteristic Table Location of Device14 (Secondary Master Mode))		
0 31:24	R	**STATIC_ADDR** The Static Addr of Device14 **Testable:** untestable **Volatile:** true

to next page

from previous page

23:16	R	**BCR_TYPE** The BCR TYPE of Device14 **Testable:** untestable **Volatile:** true
15:8	R	**DCR_TYPE** The DCR TYPE of Device14 **Testable:** untestable **Volatile:** true
7 :0	R	**DYNAMIC_ADDR** The Dynamic Addr of Device14 **Testable:** untestable **Volatile:** true

I3CD238: Device Characteristic Table Location-3 of Device4 (Master Mode)

Offset: 238h **Init = 0**

Bit	Attr.	Description
31:16	R	**RSVD** These bits in Device Characteristic Register -3 are reserved. User must ignore the values of these bits. **Testable:** untestable **Volatile:** true
15:8	R	**BCR** Bus Characteristic Value **Testable:** untestable **Volatile:** true
7 :0	R	**DCR** Device Characteristic Value **Testable:** untestable **Volatile:** true
I3CD238: Device Characteristic Table Location of Device15 (Secondary Master Mode))		
0 31:24	R	**STATIC_ADDR** The Static Addr of Device15 **Testable:** untestable **Volatile:** true
23:16	R	**BCR_TYPE** The BCR TYPE of Device15 **Testable:** untestable **Volatile:** true
15:8	R	**DCR_TYPE** The DCR TYPE of Device15 **Testable:** untestable **Volatile:** true

to next page

from previous page

7 :0	R	**DYNAMIC_ADDR** The Dynamic Addr of Device15 **Testable:** untestable **Volatile:** true

I3CD23C: Device Characteristic Table Location-4 of Device4 (Master Mode)

Offset: 23Ch **Init = 0**

Bit	Attr.	Description
31:8	R	**RSVD** These bits in Device Characteristic Register -4 are reserved. User must ignore the values of these bits. **Testable:** untestable **Volatile:** true
7 :0	R	**DEV_DYNAMIC_ADDR** Device Dynamic Address assigned. **Testable:** untestable **Volatile:** true
I3CD23C: Device Characteristic Table Location of Device16 (Secondary Master Mode))		
0 31:24	R	**STATIC_ADDR** The Static Addr of Device16 **Testable:** untestable **Volatile:** true
23:16	R	**BCR_TYPE** The BCR TYPE of Device16 **Testable:** untestable **Volatile:** true
15:8	R	**DCR_TYPE** The DCR TYPE of Device16 **Testable:** untestable **Volatile:** true
7 :0	R	**DYNAMIC_ADDR** The Dynamic Addr of Device16 **Testable:** untestable **Volatile:** true

I3CD240: Device Characteristic Table Location-1 of Device5 (Master Mode)		
Offset: 240h		**Init = 0**
Bit	**Attr.**	**Description**
31:0	R	**LSB_PROVISIONAL_ID** The LSB 32-bit value of Provisional-ID **Testable:** untestable **Volatile:** true
I3CD240: Device Characteristic Table Location of Device17 (Secondary Master Mode))		
0 31:24	R	**STATIC_ADDR** The Static Addr of Device17 **Testable:** untestable **Volatile:** true
23:16	R	**BCR_TYPE** The BCR TYPE of Device17 **Testable:** untestable **Volatile:** true
15:8	R	**DCR_TYPE** The DCR TYPE of Device17 **Testable:** untestable **Volatile:** true
7 :0	R	**DYNAMIC_ADDR** The Dynamic Addr of Device17 **Testable:** untestable **Volatile:** true

I3CD244: Device Characteristic Table Location-2 of Device5 (Master Mode)		
Offset: 244h		**Init = 0**
Bit	**Attr.**	**Description**
31:16	R	**RSVD** These bits in Device Characteristic Register -2 are reserved. User must ignore the values of these bits. **Testable:** untestable **Volatile:** true
15:0	R	**MSB_PROVISIONAL_ID** The MSB 16-bit value of Provisional-ID **Testable:** untestable **Volatile:** true
I3CD244: Device Characteristic Table Location of Device18 (Secondary Master Mode))		
0 31:24	R	**STATIC_ADDR** The Static Addr of Device18 **Testable:** untestable **Volatile:** true

to next page

from previous page

23:16	R	**BCR_TYPE** The BCR TYPE of Device18 **Testable:** untestable **Volatile:** true
15:8	R	**DCR_TYPE** The DCR TYPE of Device18 **Testable:** untestable **Volatile:** true
7 :0	R	**DYNAMIC_ADDR** The Dynamic Addr of Device18 **Testable:** untestable **Volatile:** true

I3CD248: Device Characteristic Table Location-3 of Device5 (Master Mode)

Offset: 248h **Init = 0**

Bit	Attr.	Description
31:16	R	**RSVD** These bits in Device Characteristic Register -3 are reserved. User must ignore the values of these bits. **Testable:** untestable **Volatile:** true
15:8	R	**BCR** Bus Characteristic Value **Testable:** untestable **Volatile:** true
7 :0	R	**DCR** Device Characteristic Value **Testable:** untestable **Volatile:** true
I3CD248: Device Characteristic Table Location of Device19 (Secondary Master Mode))		
0 31:24	R	**STATIC_ADDR** The Static Addr of Device19 **Testable:** untestable **Volatile:** true
23:16	R	**BCR_TYPE** The BCR TYPE of Device19 **Testable:** untestable **Volatile:** true
15:8	R	**DCR_TYPE** The DCR TYPE of Device19 **Testable:** untestable **Volatile:** true

to next page

from previous page

7 :0	R	**DYNAMIC_ADDR** The Dynamic Addr of Device19 **Testable:** untestable **Volatile:** true

I3CD24C: Device Characteristic Table Location-4 of Device5 (Master Mode)

Offset: 24Ch **Init = 0**

Bit	Attr.	Description
31:8	R	**RSVD** These bits in Device Characteristic Register -4 are reserved. User must ignore the values of these bits. **Testable:** untestable **Volatile:** true
7 :0	R	**DEV_DYNAMIC_ADDR** Device Dynamic Address assigned. **Testable:** untestable **Volatile:** true
I3CD24C: Device Characteristic Table Location of Device20 (Secondary Master Mode))		
0 31:24	R	**STATIC_ADDR** The Static Addr of Device20 **Testable:** untestable **Volatile:** true
23:16	R	**BCR_TYPE** The BCR TYPE of Device20 **Testable:** untestable **Volatile:** true
15:8	R	**DCR_TYPE** The DCR TYPE of Device20 **Testable:** untestable **Volatile:** true
7 :0	R	**DYNAMIC_ADDR** The Dynamic Addr of Device20 **Testable:** untestable **Volatile:** true

I3CD250: Device Characteristic Table Location-1 of Device6 (Master Mode)		
Offset: 250h		**Init = 0**
Bit	**Attr.**	**Description**
31:0	R	**LSB_PROVISIONAL_ID** The LSB 32-bit value of Provisional-ID **Testable:** untestable **Volatile:** true
I3CD250: Device Characteristic Table Location of Device21 (Secondary Master Mode))		
0 31:24	R	**STATIC_ADDR** The Static Addr of Device21 **Testable:** untestable **Volatile:** true
23:16	R	**BCR_TYPE** The BCR TYPE of Device21 **Testable:** untestable **Volatile:** true
15:8	R	**DCR_TYPE** The DCR TYPE of Device21 **Testable:** untestable **Volatile:** true
7 :0	R	**DYNAMIC_ADDR** The Dynamic Addr of Device21 **Testable:** untestable **Volatile:** true

I3CD254: Device Characteristic Table Location-2 of Device6 (Master Mode)		
Offset: 254h		**Init = 0**
Bit	**Attr.**	**Description**
31:16	R	**RSVD** These bits in Device Characteristic Register -2 are reserved. User must ignore the values of these bits. **Testable:** untestable **Volatile:** true
15:0	R	**MSB_PROVISIONAL_ID** The MSB 16-bit value of Provisional-ID **Testable:** untestable **Volatile:** true
I3CD254: Device Characteristic Table Location of Device22 (Secondary Master Mode))		
0 31:24	R	**STATIC_ADDR** The Static Addr of Device22 **Testable:** untestable **Volatile:** true

to next page

from previous page

23:16	R	**BCR_TYPE** The BCR TYPE of Device22 **Testable:** untestable **Volatile:** true
15:8	R	**DCR_TYPE** The DCR TYPE of Device22 **Testable:** untestable **Volatile:** true
7 :0	R	**DYNAMIC_ADDR** The Dynamic Addr of Device22 **Testable:** untestable **Volatile:** true

I3CD258: Device Characteristic Table Location-3 of Device6 (Master Mode)

Offset: 258h **Init = 0**

Bit	Attr.	Description
31:16	R	**RSVD** These bits in Device Characteristic Register -3 are reserved. User must ignore the values of these bits. **Testable:** untestable **Volatile:** true
15:8	R	**BCR** Bus Characteristic Value **Testable:** untestable **Volatile:** true
7 :0	R	**DCR** Device Characteristic Value **Testable:** untestable **Volatile:** true
I3CD258: Device Characteristic Table Location of Device23 (Secondary Master Mode))		
0 31:24	R	**STATIC_ADDR** The Static Addr of Device23 **Testable:** untestable **Volatile:** true
23:16	R	**BCR_TYPE** The BCR TYPE of Device23 **Testable:** untestable **Volatile:** true
15:8	R	**DCR_TYPE** The DCR TYPE of Device23 **Testable:** untestable **Volatile:** true

to next page

from previous page

7 :0	R	**DYNAMIC_ADDR** The Dynamic Addr of Device23 **Testable:** untestable **Volatile:** true

I3CD25C: Device Characteristic Table Location-4 of Device6 (Master Mode)

Offset: 25Ch **Init = 0**

Bit	Attr.	Description
31:8	R	**RSVD** These bits in Device Characteristic Register -4 are reserved. User must ignore the values of these bits. **Testable:** untestable **Volatile:** true
7 :0	R	**DEV_DYNAMIC_ADDR** Device Dynamic Address assigned. **Testable:** untestable **Volatile:** true
I3CD25C: Device Characteristic Table Location of Device24 (Secondary Master Mode))		
0 31:24	R	**STATIC_ADDR** The Static Addr of Device24 **Testable:** untestable **Volatile:** true
23:16	R	**BCR_TYPE** The BCR TYPE of Device24 **Testable:** untestable **Volatile:** true
15:8	R	**DCR_TYPE** The DCR TYPE of Device24 **Testable:** untestable **Volatile:** true
7 :0	R	**DYNAMIC_ADDR** The Dynamic Addr of Device24 **Testable:** untestable **Volatile:** true

I3CD260: Device Characteristic Table Location-1 of Device7 (Master Mode)		
Offset: 260h		**Init = 0**
Bit	**Attr.**	**Description**
31:0	R	**LSB_PROVISIONAL_ID** The LSB 32-bit value of Provisional-ID **Testable:** untestable **Volatile:** true
I3CD260: Device Characteristic Table Location of Device25 (Secondary Master Mode))		
0 31:24	R	**STATIC_ADDR** The Static Addr of Device25 **Testable:** untestable **Volatile:** true
23:16	R	**BCR_TYPE** The BCR TYPE of Device25 **Testable:** untestable **Volatile:** true
15:8	R	**DCR_TYPE** The DCR TYPE of Device25 **Testable:** untestable **Volatile:** true
7 :0	R	**DYNAMIC_ADDR** The Dynamic Addr of Device25 **Testable:** untestable **Volatile:** true

I3CD264: Device Characteristic Table Location-2 of Device7 (Master Mode)		
Offset: 264h		**Init = 0**
Bit	**Attr.**	**Description**
31:16	R	**RSVD** These bits in Device Characteristic Register -2 are reserved. User must ignore the values of these bits. **Testable:** untestable **Volatile:** true
15:0	R	**MSB_PROVISIONAL_ID** The MSB 16-bit value of Provisional-ID **Testable:** untestable **Volatile:** true
I3CD264: Device Characteristic Table Location of Device26 (Secondary Master Mode))		
0 31:24	R	**STATIC_ADDR** The Static Addr of Device26 **Testable:** untestable **Volatile:** true

to next page

from previous page

23:16	R	**BCR_TYPE** The BCR TYPE of Device26 **Testable:** untestable **Volatile:** true
15:8	R	**DCR_TYPE** The DCR TYPE of Device26 **Testable:** untestable **Volatile:** true
7 :0	R	**DYNAMIC_ADDR** The Dynamic Addr of Device26 **Testable:** untestable **Volatile:** true

I3CD268: Device Characteristic Table Location-3 of Device7 (Master Mode)

Offset: 268h **Init = 0**

Bit	Attr.	Description
31:16	R	**RSVD** These bits in Device Characteristic Register -3 are reserved. User must ignore the values of these bits. **Testable:** untestable **Volatile:** true
15:8	R	**BCR** Bus Characteristic Value **Testable:** untestable **Volatile:** true
7 :0	R	**DCR** Device Characteristic Value **Testable:** untestable **Volatile:** true
I3CD268: Device Characteristic Table Location of Device27 (Secondary Master Mode))		
0 31:24	R	**STATIC_ADDR** The Static Addr of Device27 **Testable:** untestable **Volatile:** true
23:16	R	**BCR_TYPE** The BCR TYPE of Device27 **Testable:** untestable **Volatile:** true
15:8	R	**DCR_TYPE** The DCR TYPE of Device27 **Testable:** untestable **Volatile:** true

to next page

from previous page

7 :0	R	**DYNAMIC_ADDR** The Dynamic Addr of Device27 **Testable:** untestable **Volatile:** true

I3CD26C: Device Characteristic Table Location-4 of Device7 (Master Mode)

Offset: 26Ch **Init = 0**

Bit	Attr.	Description
31:8	R	**RSVD** These bits in Device Characteristic Register -4 are reserved. User must ignore the values of these bits. **Testable:** untestable **Volatile:** true
7 :0	R	**DEV_DYNAMIC_ADDR** Device Dynamic Address assigned. **Testable:** untestable **Volatile:** true
I3CD26C: Device Characteristic Table Location of Device28 (Secondary Master Mode))		
0 31:24	R	**STATIC_ADDR** The Static Addr of Device28 **Testable:** untestable **Volatile:** true
23:16	R	**BCR_TYPE** The BCR TYPE of Device28 **Testable:** untestable **Volatile:** true
15:8	R	**DCR_TYPE** The DCR TYPE of Device28 **Testable:** untestable **Volatile:** true
7 :0	R	**DYNAMIC_ADDR** The Dynamic Addr of Device28 **Testable:** untestable **Volatile:** true

I3CD270: Device Characteristic Table Location-1 of Device8 (Master Mode)		
Offset: 270h		**Init = 0**
Bit	**Attr.**	**Description**
31:0	R	**LSB_PROVISIONAL_ID** The LSB 32-bit value of Provisional-ID **Testable:** untestable **Volatile:** true
I3CD270: Device Characteristic Table Location of Device29 (Secondary Master Mode))		
0 31:24	R	**STATIC_ADDR** The Static Addr of Device29 **Testable:** untestable **Volatile:** true
23:16	R	**BCR_TYPE** The BCR TYPE of Device29 **Testable:** untestable **Volatile:** true
15:8	R	**DCR_TYPE** The DCR TYPE of Device29 **Testable:** untestable **Volatile:** true
7 :0	R	**DYNAMIC_ADDR** The Dynamic Addr of Device29 **Testable:** untestable **Volatile:** true

I3CD274: Device Characteristic Table Location-2 of Device8 (Master Mode)		
Offset: 274h		**Init = 0**
Bit	**Attr.**	**Description**
31:16	R	**RSVD** These bits in Device Characteristic Register -2 are reserved. User must ignore the values of these bits. **Testable:** untestable **Volatile:** true
15:0	R	**MSB_PROVISIONAL_ID** The MSB 16-bit value of Provisional-ID **Testable:** untestable **Volatile:** true
I3CD274: Device Characteristic Table Location of Device30 (Secondary Master Mode))		
0 31:24	R	**STATIC_ADDR** The Static Addr of Device30 **Testable:** untestable **Volatile:** true

to next page

from previous page

23:16	R	**BCR_TYPE** The BCR TYPE of Device30 **Testable:** untestable **Volatile:** true
15:8	R	**DCR_TYPE** The DCR TYPE of Device30 **Testable:** untestable **Volatile:** true
7 :0	R	**DYNAMIC_ADDR** The Dynamic Addr of Device30 **Testable:** untestable **Volatile:** true

I3CD278: Device Characteristic Table Location-3 of Device8 (Master Mode)

Offset: 278h **Init = 0**

Bit	Attr.	Description
31:16	R	**RSVD** These bits in Device Characteristic Register -3 are reserved. User must ignore the values of these bits. **Testable:** untestable **Volatile:** true
15:8	R	**BCR** Bus Characteristic Value **Testable:** untestable **Volatile:** true
7 :0	R	**DCR** Device Characteristic Value **Testable:** untestable **Volatile:** true
I3CD278: Device Characteristic Table Location of Device31 (Secondary Master Mode))		
0 31:24	R	**STATIC_ADDR** The Static Addr of Device31 **Testable:** untestable **Volatile:** true
23:16	R	**BCR_TYPE** The BCR TYPE of Device31 **Testable:** untestable **Volatile:** true
15:8	R	**DCR_TYPE** The DCR TYPE of Device31 **Testable:** untestable **Volatile:** true

to next page

from previous page

7 :0	R	**DYNAMIC_ADDR** The Dynamic Addr of Device31 **Testable:** untestable **Volatile:** true

I3CD27C: Device Characteristic Table Location-4 of Device8 (Master Mode)

Offset: 27Ch **Init = 0**

Bit	Attr.	Description
31:8	R	**RSVD** These bits in Device Characteristic Register -4 are reserved. User must ignore the values of these bits. **Testable:** untestable **Volatile:** true
7 :0	R	**DEV_DYNAMIC_ADDR** Device Dynamic Address assigned. **Testable:** untestable **Volatile:** true
I3CD27C: Device Characteristic Table Location of Device32 (Secondary Master Mode))		
0 31:24	R	**STATIC_ADDR** The Static Addr of Device32 **Testable:** untestable **Volatile:** true
23:16	R	**BCR_TYPE** The BCR TYPE of Device32 **Testable:** untestable **Volatile:** true
15:8	R	**DCR_TYPE** The DCR TYPE of Device32 **Testable:** untestable **Volatile:** true
7 :0	R	**DYNAMIC_ADDR** The Dynamic Addr of Device32 **Testable:** untestable **Volatile:** true

Offset: 280h	DEV_ADDR_TABLE_LOC1: Device Address Table of Device1	Init = 0
Bit	**R/W**	**Description**
31	RW	**LEGACY_I2C_DEVICE** Legacy I2C device or not. This bit should be set to 1 if the device is a legacy I2C device. **Testable:** untestable
30:29	RW	**DEV_NACK_RETRY_CNT** This field is used to set the Device NACK Retry count for the particular device. If the Device NACK's for the device address, the controller automatically retries the same device until this count expires. If the Slave does not ACK for the mentioned number of retries, then Controller generates an error response and move to the Halt state. This feature is used for Retry Model for the following features mentioned in the I3C Specification: - Retry Model for Direct GET CCC Commands. - The incoming SIR-IBI matches with the slave address initated by the Master. **Testable:** untestable
28:26	R	**RSVD_28_26: These bits in Device Address Table Register are reserved.** User must ignore the values of these bits. **Testable:** untestable
25:24	RW	**ADDR_MASK: IBI Address Mask** - 2'b00: No address mask applied. - 2'b01: ignore DEV_DYNAMIC_ADDR[2:0] when receiving IBI. - 2'b10: ignore DEV_DYNAMIC_ADDR[3:0] when receiving IBI. - 2'b11: No address mask applied. **Testable:** untestable
23:16	RW	**DEV_DYNAMIC_ADDR** Device Dynamic Address with parity. The MSB, bit[23], should be programmed with parity of dynamic address. **Testable:** untestable
15	R	**RSVD_15: These bits in Device Address Table Register are reserved.** User must ignore the values of these bits. **Testable:** untestable
14	RW	**MR_REJECT: In-Band Master Request Reject field is used to control, per device, whether to accept Master request from Devices.** **- 0x0 - Accept: ACK the Master Request** **- 0x1 - Reject: NACK the Master Request and send auto disable CCC.** **Testable:** untestable
13	RW	**SIR_REJECT: In-Band Slave Interrupt Request Reject field is used to control, per device, whether to accept Master request from Devices.** **- 0x0 - Accept: ACK the Master Request** **- 0x1 - Reject: NACK the Master Request and send auto disable CCC.** **Testable:** untestable

to next page

from previous page

12	RW	**IBI_WITH_DATA:Mandatory one or more data bytes follow the accepted IBI from the device. Data byte continuation is indicated by T-Bit.** **- 0x0 - IBI Without Mandatory Byte** **- 0x1 - IBI with one or more Mandatory Bytes.** **Testable:** untestable
11	RW	**IBI_PEC_EN:Packet Error Check enabled for accepted IBI from the device. PEC byte is appended at the end of IBI data from the device. This bit controls whether PEC check should be performed for IBI data from device. This bit also controls whether PEC byte has to be send for auto disable CCC when controller NACKs the IBI.** **- 0x0 - Packet Error Check disabled for IBI** **- 0x1 - Packet Error Check enabled for IBI** **This field is applicable only if configuration parameter 'IC_HAS_PEC' is set to 1.** **Testable:** untestable
10:7	R	**RSVD_15_7: These bits in Device Address Table Register are reserved.** User must ignore the values of these bits. **Testable:** untestable
6 :0	RW	**DEV_STATIC_ADDR** Device Static Address. **Testable:** untestable

Offset: 284h **DEV_ADDR_TABLE_LOC2: Device Address Table of Device2** **Init = 0**

Bit	R/W	Description
31	RW	**LEGACY_I2C_DEVICE** Legacy I2C device or not. This bit should be set to 1 if the device is a legacy I2C device. **Testable:** untestable
30:29	RW	**DEV_NACK_RETRY_CNT** This field is used to set the Device NACK Retry count for the particular device. If the Device NACK's for the device address, the controller automatically retries the same device until this count expires. If the Slave does not ACK for the mentioned number of retries, then Controller generates an error response and move to the Halt state. This feature is used for Retry Model for the following features mentioned in the I3C Specification: - Retry Model for Direct GET CCC Commands. - The incoming SIR-IBI matches with the slave address initated by the Master. **Testable:** untestable
28:26	R	**RSVD_28_26: These bits in Device Address Table Register are reserved.** User must ignore the values of these bits. **Testable:** untestable

to next page

from previous page

25:24	RW	**ADDR_MASK: IBI Address Mask** - 2'b00: No address mask applied. - 2'b01: ignore DEV_DYNAMIC_ADDR[2:0] when receiving IBI. - 2'b10: ignore DEV_DYNAMIC_ADDR[3:0] when receiving IBI. - 2'b11: No address mask applied. **Testable:** untestable
23:16	RW	**DEV_DYNAMIC_ADDR** Device Dynamic Address with parity. The MSB, bit[23], should be programmed with parity of dynamic address. **Testable:** untestable
15	R	**RSVD_15: These bits in Device Address Table Register are reserved.** User must ignore the values of these bits. **Testable:** untestable
14	RW	**MR_REJECT: In-Band Master Request Reject field is used to control, per device, whether to accept Master request from Devices.** **- 0x0 - Accept: ACK the Master Request** **- 0x1 - Reject: NACK the Master Request and send auto disable CCC.** **Testable:** untestable
13	RW	**SIR_REJECT: In-Band Slave Interrupt Request Reject field is used to control, per device, whether to accept Master request from Devices.** **- 0x0 - Accept: ACK the Master Request** **- 0x1 - Reject: NACK the Master Request and send auto disable CCC.** **Testable:** untestable
12	RW	**IBI_WITH_DATA:Mandatory one or more data bytes follow the accepted IBI from the device. Data byte continuation is indicated by T-Bit.** **- 0x0 - IBI Without Mandatory Byte** **- 0x1 - IBI with one or more Mandatory Bytes.** **Testable:** untestable
11	RW	**IBI_PEC_EN:Packet Error Check enabled for accepted IBI from the device. PEC byte is appended at the end of IBI data from the device. This bit controls whether PEC check should be performed for IBI data from device. This bit also controls whether PEC byte has to be send for auto disable CCC when controller NACKs the IBI.** **- 0x0 - Packet Error Check disabled for IBI** **- 0x1 - Packet Error Check enabled for IBI** **This field is applicable only if configuration parameter 'IC_HAS_PEC' is set to 1.** **Testable:** untestable
10:7	R	**RSVD_15_7: These bits in Device Address Table Register are reserved.** User must ignore the values of these bits. **Testable:** untestable

to next page

from previous page

6 :0	RW	**DEV_STATIC_ADDR** Device Static Address. **Testable:** untestable

Offset: 288h **DEV_ADDR_TABLE_LOC3: Device Address Table of Device3** **Init = 0**

Bit	R/W	Description
31	RW	**LEGACY_I2C_DEVICE** Legacy I2C device or not. This bit should be set to 1 if the device is a legacy I2C device. **Testable:** untestable
30:29	RW	**DEV_NACK_RETRY_CNT** This field is used to set the Device NACK Retry count for the particular device. If the Device NACK's for the device address, the controller automatically retries the same device until this count expires. If the Slave does not ACK for the mentioned number of retries, then Controller generates an error response and move to the Halt state. This feature is used for Retry Model for the following features mentioned in the I3C Specification: - Retry Model for Direct GET CCC Commands. - The incoming SIR-IBI matches with the slave address initated by the Master. **Testable:** untestable
28:26	R	**RSVD_28_26: These bits in Device Address Table Register are reserved.** User must ignore the values of these bits. **Testable:** untestable
25:24	RW	**ADDR_MASK: IBI Address Mask** - 2'b00: No address mask applied. - 2'b01: ignore DEV_DYNAMIC_ADDR[2:0] when receiving IBI. - 2'b10: ignore DEV_DYNAMIC_ADDR[3:0] when receiving IBI. - 2'b11: No address mask applied. **Testable:** untestable
23:16	RW	**DEV_DYNAMIC_ADDR** Device Dynamic Address with parity. The MSB, bit[23], should be programmed with parity of dynamic address. **Testable:** untestable
15	R	**RSVD_15: These bits in Device Address Table Register are reserved.** User must ignore the values of these bits. **Testable:** untestable
14	RW	**MR_REJECT: In-Band Master Request Reject field is used to control, per device, whether to accept Master request from Devices.** **- 0x0 - Accept: ACK the Master Request** **- 0x1 - Reject: NACK the Master Request and send auto disable CCC.** **Testable:** untestable

to next page

from previous page

13	RW	**SIR_REJECT: In-Band Slave Interrupt Request Reject field is used to control, per device, whether to accept Master request from Devices.** **- 0x0 - Accept: ACK the Master Request** **- 0x1 - Reject: NACK the Master Request and send auto disable CCC.** **Testable:** untestable
12	RW	**IBI_WITH_DATA:Mandatory one or more data bytes follow the accepted IBI from the device. Data byte continuation is indicated by T-Bit.** **- 0x0 - IBI Without Mandatory Byte** **- 0x1 - IBI with one or more Mandatory Bytes.** **Testable:** untestable
11	RW	**IBI_PEC_EN:Packet Error Check enabled for accepted IBI from the device. PEC byte is appended at the end of IBI data from the device. This bit controls whether PEC check should be performed for IBI data from device. This bit also controls whether PEC byte has to be send for auto disable CCC when controller NACKs the IBI.** **- 0x0 - Packet Error Check disabled for IBI** **- 0x1 - Packet Error Check enabled for IBI** **This field is applicable only if configuration parameter 'IC_HAS_PEC' is set to 1.** **Testable:** untestable
10:7	R	**RSVD_15_7: These bits in Device Address Table Register are reserved.** User must ignore the values of these bits. **Testable:** untestable
6 :0	RW	**DEV_STATIC_ADDR** Device Static Address. **Testable:** untestable

Offset: 28Ch **DEV_ADDR_TABLE_LOC4: Device Address Table of Device4** **Init = 0**

Bit	R/W	Description
31	RW	**LEGACY_I2C_DEVICE** Legacy I2C device or not. This bit should be set to 1 if the device is a legacy I2C device. **Testable:** untestable
30:29	RW	**DEV_NACK_RETRY_CNT** This field is used to set the Device NACK Retry count for the particular device. If the Device NACK's for the device address, the controller automatically retries the same device until this count expires. If the Slave does not ACK for the mentioned number of retries, then Controller generates an error response and move to the Halt state. This feature is used for Retry Model for the following features mentioned in the I3C Specification: - Retry Model for Direct GET CCC Commands. - The incoming SIR-IBI matches with the slave address initated by the Master. **Testable:** untestable

to next page

from previous page

28:26	R	**RSVD_28_26: These bits in Device Address Table Register are reserved.** User must ignore the values of these bits. **Testable:** untestable
25:24	RW	**ADDR_MASK: IBI Address Mask** - 2'b00: No address mask applied. - 2'b01: ignore DEV_DYNAMIC_ADDR[2:0] when receiving IBI. - 2'b10: ignore DEV_DYNAMIC_ADDR[3:0] when receiving IBI. - 2'b11: No address mask applied. **Testable:** untestable
23:16	RW	**DEV_DYNAMIC_ADDR** Device Dynamic Address with parity. The MSB, bit[23], should be programmed with parity of dynamic address. **Testable:** untestable
15	R	**RSVD_15: These bits in Device Address Table Register are reserved.** User must ignore the values of these bits. **Testable:** untestable
14	RW	**MR_REJECT: In-Band Master Request Reject field is used to control, per device, whether to accept Master request from Devices.** **- 0x0 - Accept: ACK the Master Request** **- 0x1 - Reject: NACK the Master Request and send auto disable CCC.** **Testable:** untestable
13	RW	**SIR_REJECT: In-Band Slave Interrupt Request Reject field is used to control, per device, whether to accept Master request from Devices.** **- 0x0 - Accept: ACK the Master Request** **- 0x1 - Reject: NACK the Master Request and send auto disable CCC.** **Testable:** untestable
12	RW	**IBI_WITH_DATA:Mandatory one or more data bytes follow the accepted IBI from the device. Data byte continuation is indicated by T-Bit.** **- 0x0 - IBI Without Mandatory Byte** **- 0x1 - IBI with one or more Mandatory Bytes.** **Testable:** untestable
11	RW	**IBI_PEC_EN:Packet Error Check enabled for accepted IBI from the device. PEC byte is appended at the end of IBI data from the device. This bit controls whether PEC check should be performed for IBI data from device. This bit also controls whether PEC byte has to be send for auto disable CCC when controller NACKs the IBI.** **- 0x0 - Packet Error Check disabled for IBI** **- 0x1 - Packet Error Check enabled for IBI** **This field is applicable only if configuration parameter 'IC_HAS_PEC' is set to 1.** **Testable:** untestable

to next page

from previous page

10:7	R	**RSVD_15_7: These bits in Device Address Table Register are reserved.** User must ignore the values of these bits. **Testable:** untestable
6 :0	RW	**DEV_STATIC_ADDR** Device Static Address. **Testable:** untestable

Offset: 290h **DEV_ADDR_TABLE_LOC5: Device Address Table of Device5** **Init = 0**

Bit	R/W	Description
31	RW	**LEGACY_I2C_DEVICE** Legacy I2C device or not. This bit should be set to 1 if the device is a legacy I2C device. **Testable:** untestable
30:29	RW	**DEV_NACK_RETRY_CNT** This field is used to set the Device NACK Retry count for the particular device. If the Device NACK's for the device address, the controller automatically retries the same device until this count expires. If the Slave does not ACK for the mentioned number of retries, then Controller generates an error response and move to the Halt state. This feature is used for Retry Model for the following features mentioned in the I3C Specification: - Retry Model for Direct GET CCC Commands. - The incoming SIR-IBI matches with the slave address initated by the Master. **Testable:** untestable
28:26	R	**RSVD_28_26: These bits in Device Address Table Register are reserved.** User must ignore the values of these bits. **Testable:** untestable
25:24	RW	**ADDR_MASK: IBI Address Mask** - 2'b00: No address mask applied. - 2'b01: ignore DEV_DYNAMIC_ADDR[2:0] when receiving IBI. - 2'b10: ignore DEV_DYNAMIC_ADDR[3:0] when receiving IBI. - 2'b11: No address mask applied. **Testable:** untestable
23:16	RW	**DEV_DYNAMIC_ADDR** Device Dynamic Address with parity. The MSB, bit[23], should be programmed with parity of dynamic address. **Testable:** untestable
15	R	**RSVD_15: These bits in Device Address Table Register are reserved.** User must ignore the values of these bits. **Testable:** untestable

to next page

from previous page

14	RW	**MR_REJECT: In-Band Master Request Reject field is used to control, per device, whether to accept Master request from Devices.** **- 0x0 - Accept: ACK the Master Request** **- 0x1 - Reject: NACK the Master Request and send auto disable CCC.** **Testable:** untestable
13	RW	**SIR_REJECT: In-Band Slave Interrupt Request Reject field is used to control, per device, whether to accept Master request from Devices.** **- 0x0 - Accept: ACK the Master Request** **- 0x1 - Reject: NACK the Master Request and send auto disable CCC.** **Testable:** untestable
12	RW	**IBI_WITH_DATA:Mandatory one or more data bytes follow the accepted IBI from the device. Data byte continuation is indicated by T-Bit.** **- 0x0 - IBI Without Mandatory Byte** **- 0x1 - IBI with one or more Mandatory Bytes.** **Testable:** untestable
11	RW	**IBI_PEC_EN:Packet Error Check enabled for accepted IBI from the device. PEC byte is appended at the end of IBI data from the device. This bit controls whether PEC check should be performed for IBI data from device. This bit also controls whether PEC byte has to be send for auto disable CCC when controller NACKs the IBI.** **- 0x0 - Packet Error Check disabled for IBI** **- 0x1 - Packet Error Check enabled for IBI** **This field is applicable only if configuration parameter 'IC_HAS_PEC' is set to 1.** **Testable:** untestable
10:7	R	**RSVD_15_7: These bits in Device Address Table Register are reserved.** User must ignore the values of these bits. **Testable:** untestable
6 :0	RW	**DEV_STATIC_ADDR** Device Static Address. **Testable:** untestable

Offset: 294h **DEV_ADDR_TABLE_LOC6: Device Address Table of Device6** **Init = 0**

Bit	R/W	Description
31	RW	**LEGACY_I2C_DEVICE** Legacy I2C device or not. This bit should be set to 1 if the device is a legacy I2C device. **Testable:** untestable

to next page

from previous page

30:29	RW	**DEV_NACK_RETRY_CNT** This field is used to set the Device NACK Retry count for the particular device. If the Device NACK's for the device address, the controller automatically retries the same device until this count expires. If the Slave does not ACK for the mentioned number of retries, then Controller generates an error response and move to the Halt state. This feature is used for Retry Model for the following features mentioned in the I3C Specification: - Retry Model for Direct GET CCC Commands. - The incoming SIR-IBI matches with the slave address initated by the Master. **Testable:** untestable
28:26	R	**RSVD_28_26: These bits in Device Address Table Register are reserved.** User must ignore the values of these bits. **Testable:** untestable
25:24	RW	**ADDR_MASK: IBI Address Mask** - 2'b00: No address mask applied. - 2'b01: ignore DEV_DYNAMIC_ADDR[2:0] when receiving IBI. - 2'b10: ignore DEV_DYNAMIC_ADDR[3:0] when receiving IBI. - 2'b11: No address mask applied. **Testable:** untestable
23:16	RW	**DEV_DYNAMIC_ADDR** Device Dynamic Address with parity. The MSB, bit[23], should be programmed with parity of dynamic address. **Testable:** untestable
15	R	**RSVD_15: These bits in Device Address Table Register are reserved.** User must ignore the values of these bits. **Testable:** untestable
14	RW	**MR_REJECT: In-Band Master Request Reject field is used to control, per device, whether to accept Master request from Devices.** **- 0x0 - Accept: ACK the Master Request** **- 0x1 - Reject: NACK the Master Request and send auto disable CCC.** **Testable:** untestable
13	RW	**SIR_REJECT: In-Band Slave Interrupt Request Reject field is used to control, per device, whether to accept Master request from Devices.** **- 0x0 - Accept: ACK the Master Request** **- 0x1 - Reject: NACK the Master Request and send auto disable CCC.** **Testable:** untestable
12	RW	**IBI_WITH_DATA:Mandatory one or more data bytes follow the accepted IBI from the device. Data byte continuation is indicated by T-Bit.** **- 0x0 - IBI Without Mandatory Byte** **- 0x1 - IBI with one or more Mandatory Bytes.** **Testable:** untestable

to next page

from previous page

11	RW	**IBI_PEC_EN:Packet Error Check enabled for accepted IBI from the device. PEC byte is appended at the end of IBI data from the device. This bit controls whether PEC check should be performed for IBI data from device. This bit also controls whether PEC byte has to be send for auto disable CCC when controller NACKs the IBI.** **- 0x0 - Packet Error Check disabled for IBI** **- 0x1 - Packet Error Check enabled for IBI** **This field is applicable only if configuration parameter 'IC_HAS_PEC' is set to 1.** **Testable:** untestable
10:7	R	**RSVD_15_7: These bits in Device Address Table Register are reserved.** User must ignore the values of these bits. **Testable:** untestable
6 :0	RW	**DEV_STATIC_ADDR** Device Static Address. **Testable:** untestable

Offset: 298h **DEV_ADDR_TABLE_LOC7: Device Address Table of Device7** **Init = 0**

Bit	R/W	Description
31	RW	**LEGACY_I2C_DEVICE** Legacy I2C device or not. This bit should be set to 1 if the device is a legacy I2C device. **Testable:** untestable
30:29	RW	**DEV_NACK_RETRY_CNT** This field is used to set the Device NACK Retry count for the particular device. If the Device NACK's for the device address, the controller automatically retries the same device until this count expires. If the Slave does not ACK for the mentioned number of retries, then Controller generates an error response and move to the Halt state. This feature is used for Retry Model for the following features mentioned in the I3C Specification: - Retry Model for Direct GET CCC Commands. - The incoming SIR-IBI matches with the slave address initated by the Master. **Testable:** untestable
28:26	R	**RSVD_28_26: These bits in Device Address Table Register are reserved.** User must ignore the values of these bits. **Testable:** untestable
25:24	RW	**ADDR_MASK: IBI Address Mask** - 2'b00: No address mask applied. - 2'b01: ignore DEV_DYNAMIC_ADDR[2:0] when receiving IBI. - 2'b10: ignore DEV_DYNAMIC_ADDR[3:0] when receiving IBI. - 2'b11: No address mask applied. **Testable:** untestable

to next page

from previous page

23:16	RW	**DEV_DYNAMIC_ADDR** Device Dynamic Address with parity. The MSB, bit[23], should be programmed with parity of dynamic address. **Testable:** untestable
15	R	**RSVD_15: These bits in Device Address Table Register are reserved.** User must ignore the values of these bits. **Testable:** untestable
14	RW	**MR_REJECT: In-Band Master Request Reject field is used to control, per device, whether to accept Master request from Devices.** **- 0x0 - Accept: ACK the Master Request** **- 0x1 - Reject: NACK the Master Request and send auto disable CCC.** **Testable:** untestable
13	RW	**SIR_REJECT: In-Band Slave Interrupt Request Reject field is used to control, per device, whether to accept Master request from Devices.** **- 0x0 - Accept: ACK the Master Request** **- 0x1 - Reject: NACK the Master Request and send auto disable CCC.** **Testable:** untestable
12	RW	**IBI_WITH_DATA:Mandatory one or more data bytes follow the accepted IBI from the device. Data byte continuation is indicated by T-Bit.** **- 0x0 - IBI Without Mandatory Byte** **- 0x1 - IBI with one or more Mandatory Bytes.** **Testable:** untestable
11	RW	**IBI_PEC_EN:Packet Error Check enabled for accepted IBI from the device. PEC byte is appended at the end of IBI data from the device. This bit controls whether PEC check should be performed for IBI data from device. This bit also controls whether PEC byte has to be send for auto disable CCC when controller NACKs the IBI.** **- 0x0 - Packet Error Check disabled for IBI** **- 0x1 - Packet Error Check enabled for IBI** **This field is applicable only if configuration parameter 'IC_HAS_PEC' is set to 1.** **Testable:** untestable
10:7	R	**RSVD_15_7: These bits in Device Address Table Register are reserved.** User must ignore the values of these bits. **Testable:** untestable
6 :0	RW	**DEV_STATIC_ADDR** Device Static Address. **Testable:** untestable

Offset: 29Ch	DEV_ADDR_TABLE_LOC8: Device Address Table of Device8	Init = 0
Bit	**R/W**	**Description**
31	RW	**LEGACY_I2C_DEVICE** Legacy I2C device or not. This bit should be set to 1 if the device is a legacy I2C device. **Testable:** untestable
30:29	RW	**DEV_NACK_RETRY_CNT** This field is used to set the Device NACK Retry count for the particular device. If the Device NACK's for the device address, the controller automatically retries the same device until this count expires. If the Slave does not ACK for the mentioned number of retries, then Controller generates an error response and move to the Halt state. This feature is used for Retry Model for the following features mentioned in the I3C Specification: - Retry Model for Direct GET CCC Commands. - The incoming SIR-IBI matches with the slave address initated by the Master. **Testable:** untestable
28:26	R	**RSVD_28_26: These bits in Device Address Table Register are reserved.** User must ignore the values of these bits. **Testable:** untestable
25:24	RW	**ADDR_MASK: IBI Address Mask** - 2'b00: No address mask applied. - 2'b01: ignore DEV_DYNAMIC_ADDR[2:0] when receiving IBI. - 2'b10: ignore DEV_DYNAMIC_ADDR[3:0] when receiving IBI. - 2'b11: No address mask applied. **Testable:** untestable
23:16	RW	**DEV_DYNAMIC_ADDR** Device Dynamic Address with parity. The MSB, bit[23], should be programmed with parity of dynamic address. **Testable:** untestable
15	R	**RSVD_15: These bits in Device Address Table Register are reserved.** User must ignore the values of these bits. **Testable:** untestable
14	RW	**MR_REJECT: In-Band Master Request Reject field is used to control, per device, whether to accept Master request from Devices.** **- 0x0 - Accept: ACK the Master Request** **- 0x1 - Reject: NACK the Master Request and send auto disable CCC.** **Testable:** untestable
13	RW	**SIR_REJECT: In-Band Slave Interrupt Request Reject field is used to control, per device, whether to accept Master request from Devices.** **- 0x0 - Accept: ACK the Master Request** **- 0x1 - Reject: NACK the Master Request and send auto disable CCC.** **Testable:** untestable

to next page

from previous page

12	RW	**IBI_WITH_DATA:Mandatory one or more data bytes follow the accepted IBI from the device. Data byte continuation is indicated by T-Bit.** **- 0x0 - IBI Without Mandatory Byte** **- 0x1 - IBI with one or more Mandatory Bytes.** **Testable:** untestable
11	RW	**IBI_PEC_EN:Packet Error Check enabled for accepted IBI from the device. PEC byte is appended at the end of IBI data from the device. This bit controls whether PEC check should be performed for IBI data from device. This bit also controls whether PEC byte has to be send for auto disable CCC when controller NACKs the IBI.** **- 0x0 - Packet Error Check disabled for IBI** **- 0x1 - Packet Error Check enabled for IBI** **This field is applicable only if configuration parameter 'IC_HAS_PEC' is set to 1.** **Testable:** untestable
10:7	R	**RSVD_15_7: These bits in Device Address Table Register are reserved.** User must ignore the values of these bits. **Testable:** untestable
6 :0	RW	**DEV_STATIC_ADDR** Device Static Address. **Testable:** untestable

54.3 Register Change in AST2600A2 I3C

1. Add In-Band Interrupt (IBI) with data related register field: DEV_CTRL[23:16](0x00), DEV_CTRL[9](0x00), DEV_ADDR_TABLE[12](0x280-0x29C), IBI_QUEUE_STATUS(0x18), QUEUE_THLD_CTRL[23:16](0x1C)
2. Add Packet Error Check (PEC) features related register field PEC in Transfer Command Data Structure, DEV_ADDR_TABLE[31] (0x280-0x29C)
3. Add more capability information about AT2600 I3C in HW_CAPIBILITY(0x08) register
4. Add Transfer Command Data Structure, Transfer Argument Data Structure, Short Data Argument Data Structure and Address Assignment Command Data Structure to Command Queue Port(0x0c).
5. Add Bus Reset related register field: RESET_CTRL[31:29], INTR_STS[15](0x3C), INTR_STS_EN[15](0x40), INTR_SIGNAL_EN[15](0x44), INTR_FORCE[15](0x48) and SCL_LOW_MST_EXT_TIMEOUT(0xDC)
6. Add configurable Provisional ID related register: SLV_MIPI_PID_VALUE(0x70)
7. Add configurable delay/hold-time related register: SDA_HOLD_SWITCH_DLY_TIMING(0xD0)
8. Delete SLAVE_CONFIG (0xEC)

55 I3C HDMA Controller (I3CDMA)

55.1 Overview

Base Address of Global Register = 0x1E65_1000
Register Address of HDMA/Channel_X_Registers = (Base Address of Global Register) + Offset
X = 0, 1, 2, 3, 4, 5, 6, 7

HDMA000: SAR0
HDMA008: DAR0
HDMA010: LLP0
HDMA018: CTL0
HDMA020: SSTAT0
HDMA028: DSTAT0
HDMA030: SSTATAR0
HDMA038: DSTATAR0
HDMA040: CFG0
HDMA048: SGR0
HDMA050: DSR0
HDMA058: SAR1
HDMA060: DAR1
HDMA068: LLP1
HDMA070: CTL1
HDMA078: SSTAT1
HDMA080: DSTAT1
HDMA088: SSTATAR1
HDMA090: DSTATAR1
HDMA098: CFG1
HDMA0A0: SGR1
HDMA0A8: DSR1
HDMA0B0: SAR2
HDMA0B8: DAR2
HDMA0C0: LLP2
HDMA0C8: CTL2
HDMA0D0: SSTAT2
HDMA0D8: DSTAT2
HDMA0E0: SSTATAR2
HDMA0E8: DSTATAR2
HDMA0F0: CFG2
HDMA0F8: SGR2
HDMA100: DSR2
HDMA108: SAR3
HDMA110: DAR3
HDMA118: LLP3
HDMA120: CTL3
HDMA128: SSTAT3
HDMA130: DSTAT3
HDMA138: SSTATAR3
HDMA140: DSTATAR3
HDMA148: CFG3
HDMA150: SGR3
HDMA158: DSR3
HDMA160: SAR4
HDMA168: DAR4
HDMA170: LLP4
HDMA178: CTL4
HDMA180: SSTAT4

HDMA188: DSTAT4
HDMA190: SSTATAR4
HDMA198: DSTATAR4
HDMA1A0: CFG4
HDMA1A8: SGR4
HDMA1B0: DSR4
HDMA1B8: SAR5
HDMA1C0: DAR5
HDMA1C8: LLP5
HDMA1D0: CTL5
HDMA1D8: SSTAT5
HDMA1E0: DSTAT5
HDMA1E8: SSTATAR5
HDMA1F0: DSTATAR5
HDMA1F8: CFG5
HDMA200: SGR5
HDMA208: DSR5
HDMA210: SAR6
HDMA218: DAR6
HDMA220: LLP6
HDMA228: CTL6
HDMA230: SSTAT6
HDMA238: DSTAT6
HDMA240: SSTATAR6
HDMA248: DSTATAR6
HDMA250: CFG6
HDMA258: SGR6
HDMA260: DSR6
HDMA268: SAR7
HDMA270: DAR7
HDMA278: LLP7
HDMA280: CTL7
HDMA288: SSTAT7
HDMA290: DSTAT7
HDMA298: SSTATAR7
HDMA2A0: DSTATAR7
HDMA2A8: CFG7
HDMA2B0: SGR7
HDMA2B8: DSR7

Register Address of HDMA/Interrupt_Registers = (Base Address of Global Register) + Offset
HDMA2C0: RawTfr
HDMA2C8: RawBlock
HDMA2D0: RawSrcTran
HDMA2D8: RawDstTran
HDMA2E0: RawErr
HDMA2E8: StatusTfr
HDMA2F0: StatusBlock
HDMA2F8: StatusSrcTran
HDMA300: StatusDstTran
HDMA308: StatusErr
HDMA310: MaskTfr
HDMA318: MaskBlock
HDMA320: MaskSrcTran
HDMA328: MaskDstTran
HDMA330: MaskErr

HDMA338: ClearTfr
HDMA340: ClearBlock
HDMA348: ClearSrcTran
HDMA350: ClearDstTran
HDMA358: ClearErr
HDMA360: StatusInt

Register Address of HDMA/Software_Handshake_Registers = (Base Address of Global Register) + Offset
HDMA368: ReqSrcReg
HDMA370: ReqDstReg
HDMA378: SglRqSrcReg
HDMA380: SglRqDstReg
HDMA388: LstSrcReg
HDMA390: LstDstReg

Register Address of HDMA/Miscellaneous_Registers = (Base Address of Global Register) + Offset
HDMA398: DmaCfgReg
HDMA3A0: ChEnReg
HDMA3A8: DmaIdReg
HDMA3B0: DmaTestReg
HDMA3B8: DmaLpTimeoutReg
HDMA3C8: DMA_COMP_PARAMS_6
HDMA3D0: DMA_COMP_PARAMS_5
HDMA3D8: DMA_COMP_PARAMS_4
HDMA3E0: DMA_COMP_PARAMS_3
HDMA3E8: DMA_COMP_PARAMS_2
HDMA3F0: DMA_COMP_PARAMS_1
HDMA3F8: DmaCompsID

55.2 Registers

Offset: 000h		**SAR0: Source Address for Channel 0** **Init = 0**
63:32	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0 **Volatile:** true
31:0	RW	**SAR** Current Source Address of DMA transfer. Updated after each source transfer. The SINC field in the CTL0 register determines whether the address increments, decrements, or is left unchanged on every source transfer through the block transfer. **Value after reset:** 0x0 **Volatile:** true

Offset: 008h		DAR0: Destination Address for Channel 0	Init = 0
63:32	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0 **Volatile:** true	
31:0	RW	**DAR** Current Destination address of DMA transfer. Updated after each destination transfer. The DINC field in the CTL0 register determines whether the address increments, decrements, or is left unchanged on every destination transfer throughout the block transfer. **Value after reset:** 0x0 **Volatile:** true	

Offset: 010h		LLP0: Linked List Pointer Register for Channel 0	Init = 0
63:32	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0 **Volatile:** true	
31:2	RW	**LOC** Starting Address In Memory of next LLI if block chaining is enabled. Note that the two LSBs of the starting address are not stored because the address is assumed to be aligned to a 32-bit boundary. LLI accesses are always 32-bit accesses (Hsize=2) aligned to 32-bit boundaries and cannot be changed or programmed to anything other than 32-bit. **Value after reset:** 0x0 **Volatile:** true	
1 :0	RW	**LMS** List Master Select. Identifies the AHB layer/interface where the memory device that stores the next linked list item resides. This field does not exist if the configuration parameter is not set to NO_HARDCODE. In this case, the read-back value is always the hardcoded value. The maximum value of this field that can be read back is DMAH_NUM_MASTER_INT-1. **Values:** - 0x0 (LIST_MASTER_SELECT_1): The memory device stores the next linked list item on AHB master 1 - 0x1 (LIST_MASTER_SELECT_2): The memory device stores the next linked list item on AHB master 2 - 0x2 (LIST_MASTER_SELECT_3): The memory device stores the next linked list item on AHB master 3 - 0x3 (LIST_MASTER_SELECT_4): The memory device stores the next linked list item on AHB master 4 **Value after reset:** 0x0 **Volatile:** true	

Offset: 018h		CTL0: Control Register for Channel 0	Init = 0
63:45	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0 **Volatile:** true	
44	RW	**DONE** Done bit. If status write-back is enabled, the upper word of the control register, CTL0[63:32], is written to the control register location of the Linked List Item (LLI) in system memory at the end of the block transfer with the done bit set. Software can poll the LLI CTL0.DONE bit to see when a block transfer is complete. The LLI CTL0.DONE bit should be cleared when the linked lists are set up in memory prior to enabling the channel. LLI accesses are always 32-bit accesses (Hsize=2) aligned to 32-bit boundaries and cannot be changed or programmed to anything other than 32-bit. For more information, refer to ”Multi-Block Transfers”. **Values:** - 0x0 (DISABLED): DONE bit is deasserted the enb of block transfer - 0x1 (ENABLED): SET the DONE bit at the end of block transfer **Value after reset:** 0x0 **Volatile:** true	
43:37	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0 **Volatile:** true	
36:32	RW	**BLOCK_TS** Block Transfer Size. When the HDMA is the flow controller, the user writes this field before the channel is enabled in order to indicate the block size. The number programmed into BLOCK_TS indicates the total number of single transactions to perform for every block transfer; a single transaction is mapped to a single AMBA beat **Width:** The width of single transaction is determined by CTL0.SRC_TR_WIDTH. For further information on setting this field, refer to ”Transfer Operation”. Once the transfer starts, the read-back value is the total number of data items already read from the source peripheral, regardless of what is the flow controller. When the source or destination peripheral is assigned as the flow controller, then the maximum block size that can be read back saturates at DMAH_CH0_MAX_BLK_SIZE, but the actual block size can be greater. **Value after reset:** 0x2 **Volatile:** true	
31:29	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0 **Volatile:** true	

28	RW	**LLP_SRC_EN** Block chaining is enabled on the source side only if the LLP_SRC_EN field is high and LLP0.LOC is non-zero. For more information, see ”Block Chaining using Linked Lists”. **Values:** - 0x0 (LLP_SRC_DISABLE): Block chaining using Linked List is disabled on the Source side - 0x1 (LLP_SRC_ENABLE): Block chaining using Linked List is enabled on the Source side **Value after reset:** 0x0 **Volatile:** true
27	RW	**LLP_DST_EN** Block chaining is enabled on the destination side only if LLP_DST_EN field is high and LLP0.LOC is non-zero. For more information, see ”Block Chaining using Linked Lists” **Values:** - 0x0 (LLP_DST_DISABLE): Block chaining using Linked List is disabled on the Destination side - 0x1 (LLP_DST_ENABLE): Block chaining using Linked List is enabled on the Destination side **Value after reset:** 0x0 **Volatile:** true
26:25	RW	**SMS** Source Master Select. Identifies the Master Interface layer where the source device (peripheral or memory) resides. The maximum value of this field that can be read back is DMAH_NUM_MASTER_INT-1. **Values:** - 0x0 (SMS_0): Source device (peripheral or memory) is accessed from AHB master 1 - 0x1 (SMS_1): Source device (peripheral or memory) is accessed from AHB master 2 - 0x2 (SMS_2): Source device (peripheral or memory) is accessed from AHB master 3 - 0x3 (SMS_3): Source device (peripheral or memory) is accessed from AHB master 4 **Value after reset:** 0x0 **Volatile:** true
24:23	RW	**DMS** Destination Master Select. Identifies the Master Interface layer where the destination device (peripheral or memory) resides. The maximum value of this field that can be read back is DMAH_NUM_MASTER_INT-1. **Values:** - 0x0 (DMS_0): Destination device (peripheral or memory) is accessed from AHB master 1 - 0x1 (DMS_1): Destination device (peripheral or memory) is accessed from AHB master 2 - 0x2 (DMS_2): Destination device (peripheral or memory) is accessed from AHB master 3 - 0x3 (DMS_3): Destination device (peripheral or memory) is accessed from AHB master 4 **Value after reset:** 0x0 **Volatile:** true

22:20	RW	**TT_FC** Transfer Type and Flow Control. Flow control can be assigned to the HDMA, the source peripheral, or the destination peripheral. For more information on transfer types and flow control, refer to ”Setup/Operation of the HDMA Transfers”. Dependencies: If the configuration parameter DMAH_CH0_FC is set to DMA_FC_ONLY, then TT_FC[2] does not exist and TT_FC[2] always reads back 0. If DMAH_CH0_FC is set to SRC_FC_ONLY, then TT_FC[2:1] does not exist and TT_FC[2:1] always reads back 2’b10. If DMAH_CH0_FC is set to DST_FC_ONLY, then TT_FC[2:1] does not exist and TT_FC[2:1] always reads back 2’b11. For multi-block transfers using linked list operation, TT_FC must be constant for all blocks of this multi-block transfer. **Values:** - 0x0 (TT_FC_0): Transfer type is Memory to Memory and Flow Controller is HDMA - 0x1 (TT_FC_1): Transfer type is Memory to Peripheral and Flow Controller is HDMA - 0x2 (TT_FC_2): Transfer type is Peripheral to Memory and Flow Controller is HDMA - 0x3 (TT_FC_3): Transfer type is Peripheral to Peripheral and Flow Controller is HDMA - 0x4 (TT_FC_4): Transfer type is Peripheral to Memory and Flow Controller is Peripheral - 0x5 (TT_FC_5): Transfer type is Peripheral to Peripheral and Flow Controller is Source Peripheral - 0x6 (TT_FC_6): Transfer type is Memory to Peripheral and Flow Controller is Peripheral - 0x7 (TT_FC_7): Transfer type is Peripheral to Peripheral and Flow Controller is Destination Peripheral **Value after reset:** 0x3 **Volatile:** true
19	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0 **Volatile:** true
18	RW	**DST_SCATTER_EN** Destination scatter enable. Scatter on the destination side is applicable only when the CTL0.DINC bit indicates an incrementing or decrementing address control. **Values:** - 0x0 (DST_SCATTER_DISABLE): Destination Scatter is disabled - 0x1 (DST_SCATTER_ENABLE): Destination Scatter is enabled **Value after reset:** 0x0 **Volatile:** true
17	RW	**SRC_GATHER_EN** Destination gather enable. Gather on the destination side is applicable only when the CTL0.SINC bit indicates an incrementing or decrementing address control. **Values:** - 0x0 (SRC_SCATTER_DISABLE): Source gather is disabled - 0x1 (SRC_SCATTER_ENABLE): Source gather is enabled **Value after reset:** 0x0 **Volatile:** true

16:14	RW	**SRC_MSIZE** Source Burst Transaction Length. Number of data items, each of width CTL0.SRC_TR_WIDTH, to be read from the source every time a burst transferred request is made from either the corresponding hardware or software handshaking interface. **NOTE:** This value is not related to the AHB bus master HBURST bus. For information on the decoding for this field, see the ”Setting Up Transfers” section and for more inforamtion about this field, see the ”Choosing the Receive Watermark level” section in the HDMA Databook. **Values:** - 0x0 (SRC_MSIZE_0): Number of data items to be transferred is 1 - 0x1 (SRC_MSIZE_1): Number of data items to be transferred is 4 - 0x2 (SRC_MSIZE_2): Number of data items to be transferred is 8 - 0x3 (SRC_MSIZE_3): Number of data items to be transferred is 16 - 0x4 (SRC_MSIZE_4): Number of data items to be transferred is 32 - 0x5 (SRC_MSIZE_5): Number of data items to be transferred is 64 - 0x6 (SRC_MSIZE_6): Number of data items to be transferred is 128 - 0x7 (SRC_MSIZE_7): Number of data items to be transferred is 256 **Value after reset:** 0x1 **Volatile:** true
13:11	RW	**DEST_MSIZE** Destination Burst Transaction Length. Number of data items, each of width CTL0.DST_TR_WIDTH, to be written to the destination every time a destination burst transaction request is made from either the corresponding hardware or software handshaking interface. **NOTE:** This value is not related to the AHB bus master HBURST bus. **Values:** - 0x0 (DEST_MSIZE_0): Number of data items to be transferred is 1 - 0x1 (DEST_MSIZE_1): Number of data items to be transferred is 4 - 0x2 (DEST_MSIZE_2): Number of data items to be transferred is 8 - 0x3 (DEST_MSIZE_3): Number of data items to be transferred is 16 - 0x4 (DEST_MSIZE_4): Number of data items to be transferred is 32 - 0x5 (DEST_MSIZE_5): Number of data items to be transferred is 64 - 0x6 (DEST_MSIZE_6): Number of data items to be transferred is 128 - 0x7 (DEST_MSIZE_7): Number of data items to be transferred is 256 **Value after reset:** 0x1 **Volatile:** true
10:9	RW	**SINC** Source Address Increment. Indicates whether to increment or decrement the source address on every source transfer. If the device is fetching data from a source peripheral FIFO with a fixed address, then set this field to ”No change”. **Values:** - 0x0 (SINC_0): Increments the source address - 0x1 (SINC_1): Decrements the source address - 0x2 (SINC_2): No change in the source address - 0x3 (SINC_3): No change in the source address **Value after reset:** 0x0 **Volatile:** true

8 :7	RW	**DINC** Distination Address Increment. Indicates whether to increment or decrement the destination address on every destination transfer. If your device is writing data to a destination peripheral FIFO with a fixed address, then set this field to ”No Change”. **Values:** - 0x0 (DINC_0): Increments the distination address - 0x1 (DINC_1): Decrements the distination address - 0x2 (DINC_2): No change in the distination address - 0x3 (DINC_3): No change in the distination address **Value after reset:** 0x0 **Volatile:** true
6 :4	RW	**SRC_TR_WIDTH** Source Transfer Width. Mapped to AHB bus hsize. For a non-memory peripheral, typically the peripheral (source) FIFO width. This value must be less than or equl to DMAH_Mk_HDATA_WIDTH, where k is the AHB layer 1 to 4 where the source resides. **Values:** - 0x0 (SRC_TR_WIDTH_0): Source transfer width is 8 bits - 0x1 (SRC_TR_WIDTH_1): Source transfer width is 16 bits - 0x2 (SRC_TR_WIDTH_2): Source transfer width is 32 bits - 0x3 (SRC_TR_WIDTH_3): Source transfer width is 64 bits - 0x4 (SRC_TR_WIDTH_4): Source transfer width is 128 bits - 0x5 (SRC_TR_WIDTH_5): Source transfer width is 256 bits - 0x6 (SRC_TR_WIDTH_6): Source transfer width is 256 bits - 0x7 (SRC_TR_WIDTH_7): Source transfer width is 256 bits **Value after reset:** 0x2 **Volatile:** true
3 :1	RW	**DST_TR_WIDTH** Destination Transfer Width. Mapped to AHB bus hsize. For a non-memory peripheral, typically the peripheral (destination) FIFO width. This value must be less than or equal to DMAH_Mk_HDATA_WIDTH where k is the AHB layer 1 to 4 where the destination resides. For the decoding of this field, see the ”Setting Up Transfers” section in the HDMA Databook. **Values:** - 0x0 (DST_TR_WIDTH_0): Destination transfer width is 8 bits - 0x1 (DST_TR_WIDTH_1): Destination transfer width is 16 bits - 0x2 (DST_TR_WIDTH_2): Destination transfer width is 32 bits - 0x3 (DST_TR_WIDTH_3): Destination transfer width is 64 bits - 0x4 (DST_TR_WIDTH_4): Destination transfer width is 128 bits - 0x5 (DST_TR_WIDTH_5): Destination transfer width is 256 bits - 0x6 (DST_TR_WIDTH_6): Destination transfer width is 256 bits - 0x7 (DST_TR_WIDTH_7): Destination transfer width is 256 bits **Value after reset:** 0x2 **Volatile:** true

0	RW	**INT_EN** Interrupt Enable Bit. If set, then all interrupt-generating sources are enabled. Functions as a global mask bit for all interrupts for the channel; raw* interrupt registers still assert if CTL0.INT_EN=0. **Values:** - 0x0 (INTERRUPT_DISABLE): Interrupt is disabled - 0x1 (INTERRUPT_ENABLE): Interrupt is enabled **Value after reset:** 0x1 **Volatile:** true

Offset: 020h	**SSTAT0: Source Status Register for Channel 0**	**Init = 0**
63:32	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0 **Volatile:** true
31:0	RW	**SSTAT** Source status information retrieved by hardware from the address pointed to by the contents of the STATAR0 register **Value after reset:** 0x0 **Volatile:** true

Offset: 028h	**DSTAT0: Destination Status Register for Channel 0**	**Init = 0**
63:32	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0 **Volatile:** true
31:0	RW	**DSTAT** Destination status information retrieved by hardware from the address pointed to by the contents of DSTATAR0 register. **Value after reset:** 0x0 **Volatile:** true

Offset: 030h	**SSTATAR0: Source Status Address Register for Channel 0**	**Init = 0**
63:32	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0 **Volatile:** true
31:0	RW	**SSTATAR** Pointer from where hardware can fetch the source status information, which is registered in the SSTAT0 register and written out to the SSTAT0 register location of the LLI before the start of the next block. **Value after reset:** 0x0 **Volatile:** true

Offset: 038h		DSTATAR0: Source Status Address Register for Channel 0	Init = 0
63:32	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0	
31:0	RW	**DSTATAR** Pointer from where hardware can fetch the destination status information, which is registered in the DSTAT0 register and written out to the DSTAT0 register location of the LLI before the start of the next block. **Value after reset:** 0x0 **Volatile:** true	

Offset: 040h		CFG0: Configuration Register for Channel 0	Init = 0
63:47	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0 **Volatile:** true	
46:43	RW	**DEST_PER** Destination hardware interface. Assigns a hardware handshaking interface (0 : DMAH_NUM_HS_INT-1) to the destination of channel 0 if the CFG0.HS_SEL_DST field is 0; otherwise, this field is ignored. The channel can then communicate with the destination peripheral connected to that interface through the assigned hardware handshaking interface. **Note1:** For correct HDMA operation, only one peripheral (source or destination) should be assigned to the same handshaking interface. **Note2:** This field does not exist if the configuration parameter DMAH_NUM_HS_INT is set to 0. **Value after reset:** 0x0 **Volatile:** true	
42:39	RW	**SRC_PER** Source Hardware Interface. Assigns a hardware handshaking interface (0 : DMAH_NUM_HS_INT-1) to the source of channel 0 if the CFG0.HS_SEL_SRC field is 0; otherwise, this field is ignored. The channel can then communicate with the source peripheral connected to that interface through the assigned hardware handshaking interface. **Note1:** For correct HDMA operation, only one peripheral (source or destination) should be assigned to the same handshaking interface. **Note2:** This field does not exist if the configuration parameter DMAH_NUM_HS_INT is set to 0. **Value after reset:** 0x0 **Volatile:** true	

38	RW	**SS_UPD_EN** Source Status Update Enable. Source status information is fetched only from the location pointed to by the SSTATAR0 register, stored in the SSTAT0 register and written out to the SSTAT0 location of the LLI, if SS_UPD_EN is high. **Note:** This enalbe is applicable only if DMAH_CH0_STAT_SRC is set to True. This field does not exist if the configuration parameter DMAH_CH0_STAT_SRC is set to False; in this case, the read-back value is always 0. **Values:** - 0x0 (DISABLED): Source Status Update is disabled. - 0x1 (ENABLED): Source Status Update is enabled. **Value after reset:** 0x0 **Volatile:** true
37	RW	**DS_UPD_EN** Destination Status Update Enable. Destination status information is fetched only from the location pointed to by the DSTATAR0 register, stored in the DSTAT0 register and written out to the DSTAT0 location of the LLI, if DS_UPD_EN is high. This field does not exist if the configuration parameter DMAH_CH0_SATAT_DST is set to False; in this case, the read-back value is always 0. **Values:** - 0x0 (DISABLED): Destination Status Update is disabled. - 0x1 (ENABLED): Destination Status Update is enabled. **Value after reset:** 0x0 **Volatile:** true
36:34	RW	**PROTCTL** Protection Control bits used to drive the AHB HPROT[3:1] bus. The AMBA Specification recommends that the default of HPROT indicates a non-cached, non-buffered, privileged data access. The reset value is used to indicate such an access. HPROT[0] is tied high because all transfers are data accesses, as there are no opcode fetches. There is a one-to-one mapping of these register bits to the HPROT[3:1] master interface signals. Mapping of HPROT bus is as follows: - 1'b1 to HPROT[0] - CFG0.PROTCTL[1] to HPROT[1] - CFG0.PROTCTL[2] to HPROT[2] - CFG0.PROTCTL[3] to HPROT[3] **Value after reset:** 0x1 **Volatile:** true
33	RW	**FIFO_MODE** FIFO Mode Select. Determines how much space or data needs to be available in the FIFO before a burst transaction request is serviced. **Values:** - 0x0 (FIFO_MODE_0): Space/data available for single AHB transfer of the specified transfer width - 0x1 (FIFO_MODE_1): Data available is greater than or equal to half the FIFO depth for destination transfers and space available is greater than half the fifo depth for source transfers. The exceptions are at the end of a burst transaction request or at the end of a block transfer. **Value after reset:** 0x0 **Volatile:** true

32	RW	**FCMODE** Flow Control Mode. Determines when source transaction requests are serviced when the Destination Peripheral is the flow controller. **Values:** - 0x0 (FCMODE_0): Source transaction requests are serviced when they occur. Data pre-fetching is enabled - 0x1 (FCMODE_1): Source transaction requests are not serviced until a destination transaction request occurs. In this mode, the amount of data transferred from the source is limited so that it is guaranteed to be transferred to the destination prior to block termination by the destination. Data pre-fetching is disabled. **Value after reset:** 0x0 **Volatile:** true
31	RW	**RELOAD_DST** Automatic Destination Reload. The DAR0 register can be automatically reloaded from its initial value at the end of every block for multi-block transfers. A new block transfer is then initiated. This register does not exist if the configuration parameter DMAH_CH0_MULTI_BLK_EN is not selected; in this case, the read-back value is always 0. **Values:** - 0x0 (DISABLE): Destination Reload Disabled. - 0x1 (ENABLE): Destination Reload Enabled. **Value after reset:** 0x0 **Volatile:** true
30	RW	**RELOAD_SRC** Automatic Source Reload. The SAR0 register can be automatically reloaded from its initial value at the end of every block for multi-block transfers. A new block transfer is then initiated. This field does not exist if the configuration parameter DMAH_CH0_MULTI_BLK_EN is not selected; in this case, the read-back value is always 0. **Values:** - 0x0 (DISABLE): Source Reload Disabled. - 0x1 (ENABLE): Source Reload Enabled. **Value after reset:** 0x0 **Volatile:** true
29:20	R	**MAX_ABRST** Maximum AMBA Burst Length. Maximum AMBA burst length that is used for DMA transfers on this channel. A value of 0 indicates that software is not limiting the maximum AMBA burst length for DMA transfers on this channel. This field does not exist if the configuration parameter DMAH_MABRST is not selected; in this case, the read-back value is always 0, and the maximum AMBA burst length cannot be limited by software. **Value after reset:** 0x0 **Volatile:** true

19	RW	**SRC_HS_POL** Source Handshaking Interface Polarity. **Values:** - 0x0 (ACTIVE_HIGH): Source Handshaking Interface Polarity is Active high - 0x1 (ACTIVE_LOW): Source Handshaking Interface Polarity is Active low **Value after reset:** 0x0 **Volatile:** true
18	RW	**DST_HS_POL** Destination Handshaking Interface Polarity. **Values:** - 0x0 (ACTIVE_HIGH): Destination Handshaking Interface Polarity is Active high - 0x1 (ACTIVE_LOW): Destination Handshaking Interface Polarity is Active low **Value after reset:** 0x0 **Volatile:** true
17	RW	**LOCK_B** Bus Lock Bit. When active, the AHB bus master signal hlock is asserted for the duration specified in CFG0.LOCK_B_L. For more information, refer to ”Locked DMA Transfers”. This field does not exist if the configuration parameter DMAH_CH0_LOCK_EN is set to False; in this case, the read-back value is always 0. **Values:** - 0x0 (DISABLED): Bus lock bit is not enabled - 0x1 (ENABLED): Bus lock bit is enabled **Value after reset:** 0x0 **Volatile:** true
16	RW	**LOCK_CH** Channel Lock Bit. When the channel is granted control of the master bus interface and if the CFG0.LOCK_CH bit is asserted, then no other channels are granted control of the master bus interface for the duration specified in CFG0.LOCK_CH_L. Indicates to the master bus interface arbiter that this channel wants exclusive access to the master bus interface for the duration specified in CFG0.LOCK_CH_L. This field does not exist if the configuration parameter DMAH_CH0_LOCK_EN is set to False; in this case, the read-back value is always 0. **Values:** - 0x0 (DISABLED): Channel lock bit is not enabled - 0x1 (ENABLED): Channel lock bit is enabled **Value after reset:** 0x0 **Volatile:** true
14:15	RW	**LOCK_B_L** Bus lock level. Indicates the duration over which CFG0.LOCK_B bit applies. This field does not exist if the parameter DMAH_CH0_LOCK_EN is set to False; in this case, the read-back value is always 0. **Values:** - 0x0 (LOCK_B_L_0): Over complete DMA transfer - 0x1 (LOCK_B_L_1): Over complete DMA block transfer - 0x2 (LOCK_B_L_2): Over complete DMA transaction - 0x3 (LOCK_B_L_3): Over complete DMA transaction **Value after reset:** 0x0 **Volatile:** true

13:12	RW	**LOCK_CH_L** Channel Local Level. Indicates the duration over which CFG0.LOCK_CH applies. This field does not exist if the configuration parameter DMAH_CH0_LOCK_EN is set to False; in this case, the read-back value is always 0. **Values:** - 0x0 (LOCK_CH_L_0): Over complete DMA transfer - 0x1 (LOCK_CH_L_1): Over complete DMA block transfer - 0x2 (LOCK_CH_L_2): Over complete DMA transaction - 0x3 (LOCK_CH_L_3): Over complete DMA transaction **Value after reset:** 0x0 **Volatile:** true
11	RW	**HS_SEL_SRC** Source Software or Hardware Handshaking Select. This register selects which of the handshaking interfaces - hardware or software - is active for source requests on this channel. If the source peripheral is memory, then this bit is ignored. **Values:** - 0x0 (HARDWARE_HS): Hardware handshaking interface. Software initiated transaction requests are ignored. - 0x1 (SOFTWARE_HS): Software handshaking interface. Hardware initiated transaction requests are ignored. **Value after reset:** 0x1 **Volatile:** true
10	RW	**HS_SEL_DST** Destination Software or Hardware Handshaking Select. This register selects which of the handshaking interfaces - hardware or software - is active for destination requests on this channel. If the destination peripheral is memory, then this bit is ignored. **Values:** - 0x0 (HARDWARE_HS): Hardware handshaking interface. Software initiated transaction requests are ignored. - 0x1 (SOFTWARE_HS): Software handshaking interface. Hardware initiated transaction requests are ignored. **Value after reset:** 0x1 **Volatile:** true
9	R	**FIFO_EMPTY** Channel FIFO status. Indicates if there is data left in the channel FIFO. Can be used in conjunction with CFG0.CH_SUSP to cleanly disable a channel. For more information, refer to ”Disabling a Channel Prior to Transfer Completion”. **Values:** - 0x0 (NOT_EMPTY): Channel FIFO is not empty - 0x1 (EMPTY): Channel FIFO is empty **Value after reset:** 0x1 **Volatile:** true

8	R	**CH_SUSP** Channel Suspend. Suspends all DMA data transfers from the source until this bit is cleared. There is no guarantee that the current transaction will complete. Can also be used in conjunction with CFG0.FIFO_EMPTY to cleanly disable a channel without losing any data. For more information, refer to ”Disabling a Channel Prior to Transfer Completion”. **Values:** - 0x0 (NOT_SUSPENDED): DMA transfer from the source is not suspended - 0x1 (SUSPENDED): Suspend DMA transfer from the source **Value after reset:** 0x0 **Volatile:** true
7 :5	RW	**SRC_TR_WIDTH** Source Transfer Width. Mapped to AHB bus hsize. For a non-memory peripheral, typically the peripheral (source) FIFO width. This value must be less than or equl to DMAH_Mk_HDATA_WIDTH, where k is the AHB layer 1 to 4 where the source resides. **Values:** - 0x0 (SRC_TR_WIDTH_0): Source transfer width is 8 bits - 0x1 (SRC_TR_WIDTH_1): Source transfer width is 16 bits - 0x2 (SRC_TR_WIDTH_2): Source transfer width is 32 bits - 0x3 (SRC_TR_WIDTH_3): Source transfer width is 64 bits - 0x4 (SRC_TR_WIDTH_4): Source transfer width is 128 bits - 0x5 (SRC_TR_WIDTH_5): Source transfer width is 256 bits - 0x6 (SRC_TR_WIDTH_6): Source transfer width is 256 bits - 0x7 (SRC_TR_WIDTH_7): Source transfer width is 256 bits **Value after reset:** x **Volatile:** true
4 :0	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0

Offset: 048h	**SGR0: Source Gather Register for Channel 0**	**Init = 0**
63:32	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0
31:25	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0
24:19	RW	**SGC** Source Gather Count. Source contiguous tranfer count between successive gather boundaries. **Value after reset:** 0x0 **Volatile:** true

19:0	RW	**SGI** Source Gather Interval. **Value after reset:** 0x0 **Volatile:** true

Offset: 050h	**DSR0: Source Gather Register for Channel 0**	**Init = 0**
63:32	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0
31:25	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0
24:19	RW	**DSC** Destination Scatter Count. Destination contiguous transfer count between successive scatter boundaries. **Value after reset:** 0x0 **Volatile:** true
19:0	RW	**DSI** Destination Scatter Interval. **Value after reset:** 0x0 **Volatile:** true

Offset: 058h	**SAR1: Source Address for Channel 1**	**Init = 0**
63:32	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0 **Volatile:** true
31:0	RW	**SAR** Current Source Address of DMA transfer. Updated after each source transfer. The SINC field in the CTL1 register determines whether the address increments, decrements, or is left unchanged on every source transfer through the block transfer. **Value after reset:** 0x0 **Volatile:** true

Offset: 060h	**DAR1: Destination Address for Channel 1**	**Init = 0**
63:32	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0 **Volatile:** true

31:0	RW	**DAR** Current Destination address of DMA transfer. Updated after each destination transfer. The DINC field in the CTL1 register determines whether the address increments, decrements, or is left unchanged on every destination transfer throughout the block transfer. **Value after reset:** 0x0 **Volatile:** true

Offset: 068h	**LLP1: Linked List Pointer Register for Channel 1**	**Init = 0**
63:32	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0 **Volatile:** true
31:2	RW	**LOC** Starting Address In Memory of next LLI if block chaining is enabled. Note that the two LSBs of the starting address are not stored because the address is assumed to be aligned to a 32-bit boundary. LLI accesses are always 32-bit accesses (Hsize=2) aligned to 32-bit boundaries and cannot be changed or programmed to anything other than 32-bit. **Value after reset:** 0x0 **Volatile:** true
1 :0	RW	**LMS** List Master Select. Identifies the AHB layer/interface where the memory device that stores the next linked list item resides. This field does not exist if the configuration parameter is not set to NO_HARDCODE. In this case, the read-back value is always the hardcoded value. The maximum value of this field that can be read back is DMAH_NUM_MASTER_INT-1. **Values:** - 0x0 (LIST_MASTER_SELECT_1): The memory device stores the next linked list item on AHB master 1 - 0x1 (LIST_MASTER_SELECT_2): The memory device stores the next linked list item on AHB master 2 - 0x2 (LIST_MASTER_SELECT_3): The memory device stores the next linked list item on AHB master 3 - 0x3 (LIST_MASTER_SELECT_4): The memory device stores the next linked list item on AHB master 4 **Value after reset:** 0x0 **Volatile:** true

Offset: 070h	**CTL1: Control Register for Channel 1**	**Init = 0**
63:45	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0 **Volatile:** true

44	RW	**DONE** Done bit. If status write-back is enabled, the upper word of the control register, CTL1[63:32], is written to the control register location of the Linked List Item (LLI) in system memory at the end of the block transfer with the done bit set. Software can poll the LLI CTL1.DONE bit to see when a block transfer is complete. The LLI CTL1.DONE bit should be cleared when the linked lists are set up in memory prior to enabling the channel. LLI accesses are always 32-bit accesses (Hsize=2) aligned to 32-bit boundaries and cannot be changed or programmed to anything other than 32-bit. For more information, refer to ”Multi-Block Transfers”. **Values:** - 0x0 (DISABLED): DONE bit is deasserted the enb of block transfer - 0x1 (ENABLED): SET the DONE bit at the end of block transfer **Value after reset:** 0x0 **Volatile:** true
43:37	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0 **Volatile:** true
36:32	RW	**BLOCK_TS** Block Transfer Size. When the HDMA is the flow controller, the user writes this field before the channel is enabled in order to indicate the block size. The number programmed into BLOCK_TS indicates the total number of single transactions to perform for every block transfer; a single transaction is mapped to a single AMBA beat **Width:** The width of single transaction is determined by CTL1.SRC_TR_WIDTH. For further information on setting this field, refer to ”Transfer Operation”. Once the transfer starts, the read-back value is the total number of data items already read from the source peripheral, regardless of what is the flow controller. When the source or destination peripheral is assigned as the flow controller, then the maximum block size that can be read back saturates at DMAH_CH1_MAX_BLK_SIZE, but the actual block size can be greater. **Value after reset:** 0x2 **Volatile:** true
31:29	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0 **Volatile:** true
28	RW	**LLP_SRC_EN** Block chaining is enabled on the source side only if the LLP_SRC_EN field is high and LLP1.LOC is non-zero. For more information, see ”Block Chaining using Linked Lists”. **Values:** - 0x0 (LLP_SRC_DISABLE): Block chaining using Linked List is disabled on the Source side - 0x1 (LLP_SRC_ENABLE): Block chaining using Linked List is enabled on the Source side **Value after reset:** 0x0 **Volatile:** true

27	RW	**LLP_DST_EN** Block chaining is enabled on the destination side only if LLP_DST_EN field is high and LLP1.LOC is non-zero. For more information, see ”Block Chaining using Linked Lists” **Values:** - 0x0 (LLP_DST_DISABLE): Block chaining using Linked List is disabled on the Destination side - 0x1 (LLP_DST_ENABLE): Block chaining using Linked List is enabled on the Destination side **Value after reset:** 0x0 **Volatile:** true
26:25	RW	**SMS** Source Master Select. Identifies the Master Interface layer where the source device (peripheral or memory) resides. The maximum value of this field that can be read back is DMAH_NUM_MASTER_INT-1. **Values:** - 0x0 (SMS_0): Source device (peripheral or memory) is accessed from AHB master 1 - 0x1 (SMS_1): Source device (peripheral or memory) is accessed from AHB master 2 - 0x2 (SMS_2): Source device (peripheral or memory) is accessed from AHB master 3 - 0x3 (SMS_3): Source device (peripheral or memory) is accessed from AHB master 4 **Value after reset:** 0x0 **Volatile:** true
24:23	RW	**DMS** Destination Master Select. Identifies the Master Interface layer where the destination device (peripheral or memory) resides. The maximum value of this field that can be read back is DMAH_NUM_MASTER_INT-1. **Values:** - 0x0 (DMS_0): Destination device (peripheral or memory) is accessed from AHB master 1 - 0x1 (DMS_1): Destination device (peripheral or memory) is accessed from AHB master 2 - 0x2 (DMS_2): Destination device (peripheral or memory) is accessed from AHB master 3 - 0x3 (DMS_3): Destination device (peripheral or memory) is accessed from AHB master 4 **Value after reset:** 0x0 **Volatile:** true

22:20	RW	**TT_FC** Transfer Type and Flow Control. Flow control can be assigned to the HDMA, the source peripheral, or the destination peripheral. For more information on transfer types and flow control, refer to ”Setup/Operation of the HDMA Transfers”. Dependencies: If the configuration parameter DMAH_CH1_FC is set to DMA_FC_ONLY, then TT_FC[2] does not exist and TT_FC[2] always reads back 0. If DMAH_CH1_FC is set to SRC_FC_ONLY, then TT_FC[2:1] does not exist and TT_FC[2:1] always reads back 2’b10. If DMAH_CH1_FC is set to DST_FC_ONLY, then TT_FC[2:1] does not exist and TT_FC[2:1] always reads back 2’b11. For multi-block transfers using linked list operation, TT_FC must be constant for all blocks of this multi-block transfer. **Values:** - 0x0 (TT_FC_0): Transfer type is Memory to Memory and Flow Controller is HDMA - 0x1 (TT_FC_1): Transfer type is Memory to Peripheral and Flow Controller is HDMA - 0x2 (TT_FC_2): Transfer type is Peripheral to Memory and Flow Controller is HDMA - 0x3 (TT_FC_3): Transfer type is Peripheral to Peripheral and Flow Controller is HDMA - 0x4 (TT_FC_4): Transfer type is Peripheral to Memory and Flow Controller is Peripheral - 0x5 (TT_FC_5): Transfer type is Peripheral to Peripheral and Flow Controller is Source Peripheral - 0x6 (TT_FC_6): Transfer type is Memory to Peripheral and Flow Controller is Peripheral - 0x7 (TT_FC_7): Transfer type is Peripheral to Peripheral and Flow Controller is Destination Peripheral **Value after reset:** 0x3 **Volatile:** true
19	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0 **Volatile:** true
18	RW	**DST_SCATTER_EN** Destination scatter enable. Scatter on the destination side is applicable only when the CTL1.DINC bit indicates an incrementing or decrementing address control. **Values:** - 0x0 (DST_SCATTER_DISABLE): Destination Scatter is disabled - 0x1 (DST_SCATTER_ENABLE): Destination Scatter is enabled **Value after reset:** 0x0 **Volatile:** true
17	RW	**SRC_GATHER_EN** Destination gather enable. Gather on the destination side is applicable only when the CTL1.SINC bit indicates an incrementing or decrementing address control. **Values:** - 0x0 (SRC_SCATTER_DISABLE): Source gather is disabled - 0x1 (SRC_SCATTER_ENABLE): Source gather is enabled **Value after reset:** 0x0 **Volatile:** true

16:14	RW	**SRC_MSIZE** Source Burst Transaction Length. Number of data items, each of width CTL1.SRC_TR_WIDTH, to be read from the source every time a burst transferred request is made from either the corresponding hardware or software handshaking interface. **NOTE:** This value is not related to the AHB bus master HBURST bus. For information on the decoding for this field, see the ”Setting Up Transfers” section and for more inforamtion about this field, see the ”Choosing the Receive Watermark level” section in the HDMA Databook. **Values:** - 0x0 (SRC_MSIZE_0): Number of data items to be transferred is 1 - 0x1 (SRC_MSIZE_1): Number of data items to be transferred is 4 - 0x2 (SRC_MSIZE_2): Number of data items to be transferred is 8 - 0x3 (SRC_MSIZE_3): Number of data items to be transferred is 16 - 0x4 (SRC_MSIZE_4): Number of data items to be transferred is 32 - 0x5 (SRC_MSIZE_5): Number of data items to be transferred is 64 - 0x6 (SRC_MSIZE_6): Number of data items to be transferred is 128 - 0x7 (SRC_MSIZE_7): Number of data items to be transferred is 256 **Value after reset:** 0x1 **Volatile:** true
13:11	RW	**DEST_MSIZE** Destination Burst Transaction Length. Number of data items, each of width CTL1.DST_TR_WIDTH, to be written to the destination every time a destination burst transaction request is made from either the corresponding hardware or software handshaking interface. **NOTE:** This value is not related to the AHB bus master HBURST bus. **Values:** - 0x0 (DEST_MSIZE_0): Number of data items to be transferred is 1 - 0x1 (DEST_MSIZE_1): Number of data items to be transferred is 4 - 0x2 (DEST_MSIZE_2): Number of data items to be transferred is 8 - 0x3 (DEST_MSIZE_3): Number of data items to be transferred is 16 - 0x4 (DEST_MSIZE_4): Number of data items to be transferred is 32 - 0x5 (DEST_MSIZE_5): Number of data items to be transferred is 64 - 0x6 (DEST_MSIZE_6): Number of data items to be transferred is 128 - 0x7 (DEST_MSIZE_7): Number of data items to be transferred is 256 **Value after reset:** 0x1 **Volatile:** true
10:9	RW	**SINC** Source Address Increment. Indicates whether to increment or decrement the source address on every source transfer. If the device is fetching data from a source peripheral FIFO with a fixed address, then set this field to ”No change”. **Values:** - 0x0 (SINC_0): Increments the source address - 0x1 (SINC_1): Decrements the source address - 0x2 (SINC_2): No change in the source address - 0x3 (SINC_3): No change in the source address **Value after reset:** 0x0 **Volatile:** true

8 :7	RW	**DINC** Distination Address Increment. Indicates whether to increment or decrement the destination address on every destination transfer. If your device is writing data to a destination peripheral FIFO with a fixed address, then set this field to ”No Change”. **Values:** - 0x0 (DINC_0): Increments the distination address - 0x1 (DINC_1): Decrements the distination address - 0x2 (DINC_2): No change in the distination address - 0x3 (DINC_3): No change in the distination address **Value after reset:** 0x0 **Volatile:** true
6 :4	RW	**SRC_TR_WIDTH** Source Transfer Width. Mapped to AHB bus hsize. For a non-memory peripheral, typically the peripheral (source) FIFO width. This value must be less than or equl to DMAH_Mk_HDATA_WIDTH, where k is the AHB layer 1 to 4 where the source resides. **Values:** - 0x0 (SRC_TR_WIDTH_0): Source transfer width is 8 bits - 0x1 (SRC_TR_WIDTH_1): Source transfer width is 16 bits - 0x2 (SRC_TR_WIDTH_2): Source transfer width is 32 bits - 0x3 (SRC_TR_WIDTH_3): Source transfer width is 64 bits - 0x4 (SRC_TR_WIDTH_4): Source transfer width is 128 bits - 0x5 (SRC_TR_WIDTH_5): Source transfer width is 256 bits - 0x6 (SRC_TR_WIDTH_6): Source transfer width is 256 bits - 0x7 (SRC_TR_WIDTH_7): Source transfer width is 256 bits **Value after reset:** 0x2 **Volatile:** true
3 :1	RW	**DST_TR_WIDTH** Destination Transfer Width. Mapped to AHB bus hsize. For a non-memory peripheral, typically the peripheral (destination) FIFO width. This value must be less than or equal to DMAH_Mk_HDATA_WIDTH where k is the AHB layer 1 to 4 where the destination resides. For the decoding of this field, see the ”Setting Up Transfers” section in the HDMA Databook. **Values:** - 0x0 (DST_TR_WIDTH_0): Destination transfer width is 8 bits - 0x1 (DST_TR_WIDTH_1): Destination transfer width is 16 bits - 0x2 (DST_TR_WIDTH_2): Destination transfer width is 32 bits - 0x3 (DST_TR_WIDTH_3): Destination transfer width is 64 bits - 0x4 (DST_TR_WIDTH_4): Destination transfer width is 128 bits - 0x5 (DST_TR_WIDTH_5): Destination transfer width is 256 bits - 0x6 (DST_TR_WIDTH_6): Destination transfer width is 256 bits - 0x7 (DST_TR_WIDTH_7): Destination transfer width is 256 bits **Value after reset:** 0x2 **Volatile:** true

0	RW	**INT_EN** Interrupt Enable Bit. If set, then all interrupt-generating sources are enabled. Functions as a global mask bit for all interrupts for the channel; raw* interrupt registers still assert if CTL1.INT_EN=0. **Values:** - 0x0 (INTERRUPT_DISABLE): Interrupt is disabled - 0x1 (INTERRUPT_ENABLE): Interrupt is enabled **Value after reset:** 0x1 **Volatile:** true

Offset: 078h	**SSTAT1: Source Status Register for Channel 1**	**Init = 0**
63:32	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0 **Volatile:** true
31:0	RW	**SSTAT** Source status information retrieved by hardware from the address pointed to by the contents of the STATAR1 register **Value after reset:** 0x0 **Volatile:** true

Offset: 080h	**DSTAT1: Destination Status Register for Channel 1**	**Init = 0**
63:32	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0 **Volatile:** true
31:0	RW	**DSTAT** Destination status information retrieved by hardware from the address pointed to by the contents of DSTATAR1 register. **Value after reset:** 0x0 **Volatile:** true

Offset: 088h	**SSTATAR1: Source Status Address Register for Channel 1**	**Init = 0**
63:32	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0 **Volatile:** true
31:0	RW	**SSTATAR** Pointer from where hardware can fetch the source status information, which is registered in the SSTAT1 register and written out to the SSTAT1 register location of the LLI before the start of the next block. **Value after reset:** 0x0 **Volatile:** true

Offset: 090h		DSTATAR1: Source Status Address Register for Channel 1	Init = 0
63:32	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0	
31:0	RW	**DSTATAR** Pointer from where hardware can fetch the destination status information, which is registered in the DSTAT1 register and written out to the DSTAT1 register location of the LLI before the start of the next block. **Value after reset:** 0x0 **Volatile:** true	

Offset: 098h		CFG1: Configuration Register for Channel 1	Init = 0
63:47	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0 **Volatile:** true	
46:43	RW	**DEST_PER** Destination hardware interface. Assigns a hardware handshaking interface (0 : DMAH_NUM_HS_INT-1) to the destination of channel 1 if the CFG1.HS_SEL_DST field is 0; otherwise, this field is ignored. The channel can then communicate with the destination peripheral connected to that interface through the assigned hardware handshaking interface. **Note1:** For correct HDMA operation, only one peripheral (source or destination) should be assigned to the same handshaking interface. **Note2:** This field does not exist if the configuration parameter DMAH_NUM_HS_INT is set to 0. **Value after reset:** 0x0 **Volatile:** true	
42:39	RW	**SRC_PER** Source Hardware Interface. Assigns a hardware handshaking interface (0 : DMAH_NUM_HS_INT-1) to the source of channel 1 if the CFG1.HS_SEL_SRC field is 0; otherwise, this field is ignored. The channel can then communicate with the source peripheral connected to that interface through the assigned hardware handshaking interface. **Note1:** For correct HDMA operation, only one peripheral (source or destination) should be assigned to the same handshaking interface. **Note2:** This field does not exist if the configuration parameter DMAH_NUM_HS_INT is set to 0. **Value after reset:** 0x0 **Volatile:** true	

38	RW	**SS_UPD_EN** Source Status Update Enable. Source status information is fetched only from the location pointed to by the SSTATAR1 register, stored in the SSTAT1 register and written out to the SSTAT1 location of the LLI, if SS_UPD_EN is high. **Note:** This enalbe is applicable only if DMAH_CH0_STAT_SRC is set to True. This field does not exist if the configuration parameter DMAH_CH1_STAT_SRC is set to False; in this case, the read-back value is always 0. **Values:** - 0x0 (DISABLED): Source Status Update is disabled. - 0x1 (ENABLED): Source Status Update is enabled. **Value after reset:** 0x0 **Volatile:** true
37	RW	**DS_UPD_EN** Destination Status Update Enable. Destination status information is fetched only from the location pointed to by the DSTATAR0 register, stored in the DSTAT1 register and written out to the DSTAT0 location of the LLI, if DS_UPD_EN is high. This field does not exist if the configuration parameter DMAH_CH1_SATAT_DST is set to False; in this case, the read-back value is always 0. **Values:** - 0x0 (DISABLED): Destination Status Update is disabled. - 0x1 (ENABLED): Destination Status Update is enabled. **Value after reset:** 0x0 **Volatile:** true
36:34	RW	**PROTCTL** Protection Control bits used to drive the AHB HPROT[3:1] bus. The AMBA Specification recommends that the default of HPROT indicates a non-cached, non-buffered, privileged data access. The reset value is used to indicate such an access. HPROT[0] is tied high because all transfers are data accesses, as there are no opcode fetches. There is a one-to-one mapping of these register bits to the HPROT[3:1] master interface signals. Mapping of HPROT bus is as follows: - 1'b1 to HPROT[0] - CFG1.PROTCTL[1] to HPROT[1] - CFG1.PROTCTL[2] to HPROT[2] - CFG1.PROTCTL[3] to HPROT[3] **Value after reset:** 0x1 **Volatile:** true
33	RW	**FIFO_MODE** FIFO Mode Select. Determines how much space or data needs to be available in the FIFO before a burst transaction request is serviced. **Values:** - 0x0 (FIFO_MODE_0): Space/data available for single AHB transfer of the specified transfer width - 0x1 (FIFO_MODE_1): Data available is greater than or equal to half the FIFO depth for destination transfers and space available is greater than half the fifo depth for source transfers. The exceptions are at the end of a burst transaction request or at the end of a block transfer. **Value after reset:** 0x0 **Volatile:** true

32	RW	**FCMODE** Flow Control Mode. Determines when source transaction requests are serviced when the Destination Peripheral is the flow controller. **Values:** - 0x0 (FCMODE_0): Source transaction requests are serviced when they occur. Data pre-fetching is enabled - 0x1 (FCMODE_1): Source transaction requests are not serviced until a destination transaction request occurs. In this mode, the amount of data transferred from the source is limited so that it is guaranteed to be transferred to the destination prior to block termination by the destination. Data pre-fetching is disabled. **Value after reset:** 0x0 **Volatile:** true
31	RW	**RELOAD_DST** Automatic Destination Reload. The DAR1 register can be automatically reloaded from its initial value at the end of every block for multi-block transfers. A new block transfer is then initiated. This register does not exist if the configuration parameter DMAH_CH1_MULTI_BLK_EN is not selected; in this case, the read-back value is always 0. **Values:** - 0x0 (DISABLE): Destination Reload Disabled. - 0x1 (ENABLE): Destination Reload Enabled. **Value after reset:** 0x0 **Volatile:** true
30	RW	**RELOAD_SRC** Automatic Source Reload. The SAR1 register can be automatically reloaded from its initial value at the end of every block for multi-block transfers. A new block transfer is then initiated. This field does not exist if the configuration parameter DMAH_CH1_MULTI_BLK_EN is not selected; in this case, the read-back value is always 0. **Values:** - 0x0 (DISABLE): Source Reload Disabled. - 0x1 (ENABLE): Source Reload Enabled. **Value after reset:** 0x0 **Volatile:** true
29:20	R	**MAX_ABRST** Maximum AMBA Burst Length. Maximum AMBA burst length that is used for DMA transfers on this channel. A value of 0 indicates that software is not limiting the maximum AMBA burst length for DMA transfers on this channel. This field does not exist if the configuration parameter DMAH_MABRST is not selected; in this case, the read-back value is always 0, and the maximum AMBA burst length cannot be limited by software. **Value after reset:** 0x0 **Volatile:** true

19	RW	**SRC_HS_POL** Source Handshaking Interface Polarity. **Values:** - 0x0 (ACTIVE_HIGH): Source Handshaking Interface Polarity is Active high - 0x1 (ACTIVE_LOW): Source Handshaking Interface Polarity is Active low **Value after reset:** 0x0 **Volatile:** true
18	RW	**DST_HS_POL** Destination Handshaking Interface Polarity. **Values:** - 0x0 (ACTIVE_HIGH): Destination Handshaking Interface Polarity is Active high - 0x1 (ACTIVE_LOW): Destination Handshaking Interface Polarity is Active low **Value after reset:** 0x0 **Volatile:** true
17	RW	**LOCK_B** Bus Lock Bit. When active, the AHB bus master signal hlock is asserted for the duration specified in CFG0.LOCK_B_L. For more information, refer to ”Locked DMA Transfers”. This field does not exist if the configuration parameter DMAH_CH1_LOCK_EN is set to False; in this case, the read-back value is always 0. **Values:** - 0x0 (DISABLED): Bus lock bit is not enabled - 0x1 (ENABLED): Bus lock bit is enabled **Value after reset:** 0x0 **Volatile:** true
16	RW	**LOCK_CH** Channel Lock Bit. When the channel is granted control of the master bus interface and if the CFG1.LOCK_CH bit is asserted, then no other channels are granted control of the master bus interface for the duration specified in CFG1.LOCK_CH_L. Indicates to the master bus interface arbiter that this channel wants exclusive access to the master bus interface for the duration specified in CFG1.LOCK_CH_L. This field does not exist if the configuration parameter DMAH_CH1_LOCK_EN is set to False; in this case, the read-back value is always 0. **Values:** - 0x0 (DISABLED): Channel lock bit is not enabled - 0x1 (ENABLED): Channel lock bit is enabled **Value after reset:** 0x0 **Volatile:** true
14:15	RW	**LOCK_B_L** Bus lock level. Indicates the duration over which CFG1.LOCK_B bit applies. This field does not exist if the parameter DMAH_CH1_LOCK_EN is set to False; in this case, the read-back value is always 0. **Values:** - 0x0 (LOCK_B_L_0): Over complete DMA transfer - 0x1 (LOCK_B_L_1): Over complete DMA block transfer - 0x2 (LOCK_B_L_2): Over complete DMA transaction - 0x3 (LOCK_B_L_3): Over complete DMA transaction **Value after reset:** 0x0 **Volatile:** true

13:12	RW	**LOCK_CH_L** Channel Local Level. Indicates the duration over which CFG1.LOCK_CH applies. This field does not exist if the configuration parameter DMAH_CH1_LOCK_EN is set to False; in this case, the read-back value is always 0. **Values:** - 0x0 (LOCK_CH_L_0): Over complete DMA transfer - 0x1 (LOCK_CH_L_1): Over complete DMA block transfer - 0x2 (LOCK_CH_L_2): Over complete DMA transaction - 0x3 (LOCK_CH_L_3): Over complete DMA transaction **Value after reset:** 0x0 **Volatile:** true
11	RW	**HS_SEL_SRC** Source Software or Hardware Handshaking Select. This register selects which of the handshaking interfaces - hardware or software - is active for source requests on this channel. If the source peripheral is memory, then this bit is ignored. **Values:** - 0x0 (HARDWARE_HS): Hardware handshaking interface. Software initiated transaction requests are ignored. - 0x1 (SOFTWARE_HS): Software handshaking interface. Hardware initiated transaction requests are ignored. **Value after reset:** 0x1 **Volatile:** true
10	RW	**HS_SEL_DST** Destination Software or Hardware Handshaking Select. This register selects which of the handshaking interfaces - hardware or software - is active for destination requests on this channel. If the destination peripheral is memory, then this bit is ignored. **Values:** - 0x0 (HARDWARE_HS): Hardware handshaking interface. Software initiated transaction requests are ignored. - 0x1 (SOFTWARE_HS): Software handshaking interface. Hardware initiated transaction requests are ignored. **Value after reset:** 0x1 **Volatile:** true
9	R	**FIFO_EMPTY** Channel FIFO status. Indicates if there is data left in the channel FIFO. Can be used in conjunction with CFG1.CH_SUSP to cleanly disable a channel. For more information, refer to ”Disabling a Channel Prior to Transfer Completion”. **Values:** - 0x0 (NOT_EMPTY): Channel FIFO is not empty - 0x1 (EMPTY): Channel FIFO is empty **Value after reset:** 0x1 **Volatile:** true

8	R	**CH_SUSP** Channel Suspend. Suspends all DMA data transfers from the source until this bit is cleared. There is no guarantee that the current transaction will complete. Can also be used in conjunction with CFG1.FIFO_EMPTY to cleanly disable a channel without losing any data. For more information, refer to ”Disabling a Channel Prior to Transfer Completion”. **Values:** - 0x0 (NOT_SUSPENDED): DMA transfer from the source is not suspended - 0x1 (SUSPENDED): Suspend DMA transfer from the source **Value after reset:** 0x0 **Volatile:** true
7 :5	RW	**SRC_TR_WIDTH** Source Transfer Width. Mapped to AHB bus hsize. For a non-memory peripheral, typically the peripheral (source) FIFO width. This value must be less than or equl to DMAH_Mk_HDATA_WIDTH, where k is the AHB layer 1 to 4 where the source resides. **Values:** - 0x0 (SRC_TR_WIDTH_0): Source transfer width is 8 bits - 0x1 (SRC_TR_WIDTH_1): Source transfer width is 16 bits - 0x2 (SRC_TR_WIDTH_2): Source transfer width is 32 bits - 0x3 (SRC_TR_WIDTH_3): Source transfer width is 64 bits - 0x4 (SRC_TR_WIDTH_4): Source transfer width is 128 bits - 0x5 (SRC_TR_WIDTH_5): Source transfer width is 256 bits - 0x6 (SRC_TR_WIDTH_6): Source transfer width is 256 bits - 0x7 (SRC_TR_WIDTH_7): Source transfer width is 256 bits **Value after reset:** x **Volatile:** true
4 :0	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0

Offset: 0A0h	**SGR1: Source Gather Register for Channel 1**	**Init = 0**
63:32	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0
31:25	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0
24:19	RW	**SGC** Source Gather Count. Source contiguous tranfer count between successive gather boundaries. **Value after reset:** 0x0 **Volatile:** true

19:0	RW	**SGI** Source Gather Interval. **Value after reset:** 0x0 **Volatile:** true

Offset: 0A8h	**DSR1: Source Gather Register for Channel 1**	**Init = 0**
63:32	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0
31:25	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0
24:19	RW	**DSC** Destination Scatter Count. Destination contiguous transfer count between successive scatter boundaries. **Value after reset:** 0x0 **Volatile:** true
19:0	RW	**DSI** Destination Scatter Interval. **Value after reset:** 0x0 **Volatile:** true

Offset: 0B0h	**SAR2: Source Address for Channel 2**	**Init = 0**
63:32	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0 **Volatile:** true
31:0	RW	**SAR** Current Source Address of DMA transfer. Updated after each source transfer. The SINC field in the CTL2 register determines whether the address increments, decrements, or is left unchanged on every source transfer through the block transfer. **Value after reset:** 0x0 **Volatile:** true

Offset: 0B8h	**DAR2: Destination Address for Channel 2**	**Init = 0**
63:32	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0 **Volatile:** true

31:0	RW	**DAR** Current Destination address of DMA transfer. Updated after each destination transfer. The DINC field in the CTL2 register determines whether the address increments, decrements, or is left unchanged on every destination transfer throughout the block transfer. **Value after reset:** 0x0 **Volatile:** true

Offset: 0C0h	**LLP2: Linked List Pointer Register for Channel 2**	**Init = 0**
63:32	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0 **Volatile:** true
31:2	RW	**LOC** Starting Address In Memory of next LLI if block chaining is enabled. Note that the two LSBs of the starting address are not stored because the address is assumed to be aligned to a 32-bit boundary. LLI accesses are always 32-bit accesses (Hsize=2) aligned to 32-bit boundaries and cannot be changed or programmed to anything other than 32-bit. **Value after reset:** 0x0 **Volatile:** true
1 :0	RW	**LMS** List Master Select. Identifies the AHB layer/interface where the memory device that stores the next linked list item resides. This field does not exist if the configuration parameter is not set to NO_HARDCODE. In this case, the read-back value is always the hardcoded value. The maximum value of this field that can be read back is DMAH_NUM_MASTER_INT-1. **Values:** - 0x0 (LIST_MASTER_SELECT_1): The memory device stores the next linked list item on AHB master 1 - 0x1 (LIST_MASTER_SELECT_2): The memory device stores the next linked list item on AHB master 2 - 0x2 (LIST_MASTER_SELECT_3): The memory device stores the next linked list item on AHB master 3 - 0x3 (LIST_MASTER_SELECT_4): The memory device stores the next linked list item on AHB master 4 **Value after reset:** 0x0 **Volatile:** true

Offset: 0C8h	**CTL2: Control Register for Channel 2**	**Init = 0**
63:45	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0 **Volatile:** true

44	RW	**DONE** Done bit. If status write-back is enabled, the upper word of the control register, CTL2[63:32], is written to the control register location of the Linked List Item (LLI) in system memory at the end of the block transfer with the done bit set. Software can poll the LLI CTL2.DONE bit to see when a block transfer is complete. The LLI CTL2.DONE bit should be cleared when the linked lists are set up in memory prior to enabling the channel. LLI accesses are always 32-bit accesses (Hsize=2) aligned to 32-bit boundaries and cannot be changed or programmed to anything other than 32-bit. For more information, refer to ”Multi-Block Transfers”. **Values:** - 0x0 (DISABLED): DONE bit is deasserted the enb of block transfer - 0x1 (ENABLED): SET the DONE bit at the end of block transfer **Value after reset:** 0x0 **Volatile:** true
43:37	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0 **Volatile:** true
36:32	RW	**BLOCK_TS** Block Transfer Size. When the HDMA is the flow controller, the user writes this field before the channel is enabled in order to indicate the block size. The number programmed into BLOCK_TS indicates the total number of single transactions to perform for every block transfer; a single transaction is mapped to a single AMBA beat **Width:** The width of single transaction is determined by CTL2.SRC_TR_WIDTH. For further information on setting this field, refer to ”Transfer Operation”. Once the transfer starts, the read-back value is the total number of data items already read from the source peripheral, regardless of what is the flow controller. When the source or destination peripheral is assigned as the flow controller, then the maximum block size that can be read back saturates at DMAH_CH2_MAX_BLK_SIZE, but the actual block size can be greater. **Value after reset:** 0x2 **Volatile:** true
31:29	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0 **Volatile:** true
28	RW	**LLP_SRC_EN** Block chaining is enabled on the source side only if the LLP_SRC_EN field is high and LLP2.LOC is non-zero. For more information, see ”Block Chaining using Linked Lists”. **Values:** - 0x0 (LLP_SRC_DISABLE): Block chaining using Linked List is disabled on the Source side - 0x1 (LLP_SRC_ENABLE): Block chaining using Linked List is enabled on the Source side **Value after reset:** 0x0 **Volatile:** true

27	RW	**LLP_DST_EN** Block chaining is enabled on the destination side only if LLP_DST_EN field is high and LLP2.LOC is non-zero. For more information, see ”Block Chaining using Linked Lists” **Values:** - 0x0 (LLP_DST_DISABLE): Block chaining using Linked List is disabled on the Destination side - 0x1 (LLP_DST_ENABLE): Block chaining using Linked List is enabled on the Destination side **Value after reset:** 0x0 **Volatile:** true
26:25	RW	**SMS** Source Master Select. Identifies the Master Interface layer where the source device (peripheral or memory) resides. The maximum value of this field that can be read back is DMAH_NUM_MASTER_INT-1. **Values:** - 0x0 (SMS_0): Source device (peripheral or memory) is accessed from AHB master 1 - 0x1 (SMS_1): Source device (peripheral or memory) is accessed from AHB master 2 - 0x2 (SMS_2): Source device (peripheral or memory) is accessed from AHB master 3 - 0x3 (SMS_3): Source device (peripheral or memory) is accessed from AHB master 4 **Value after reset:** 0x0 **Volatile:** true
24:23	RW	**DMS** Destination Master Select. Identifies the Master Interface layer where the destination device (peripheral or memory) resides. The maximum value of this field that can be read back is DMAH_NUM_MASTER_INT-1. **Values:** - 0x0 (DMS_0): Destination device (peripheral or memory) is accessed from AHB master 1 - 0x1 (DMS_1): Destination device (peripheral or memory) is accessed from AHB master 2 - 0x2 (DMS_2): Destination device (peripheral or memory) is accessed from AHB master 3 - 0x3 (DMS_3): Destination device (peripheral or memory) is accessed from AHB master 4 **Value after reset:** 0x0 **Volatile:** true

22:20	RW	**TT_FC** Transfer Type and Flow Control. Flow control can be assigned to the HDMA, the source peripheral, or the destination peripheral. For more information on transfer types and flow control, refer to ”Setup/Operation of the HDMA Transfers”. Dependencies: If the configuration parameter DMAH_CH2_FC is set to DMA_FC_ONLY, then TT_FC[2] does not exist and TT_FC[2] always reads back 0. If DMAH_CH2_FC is set to SRC_FC_ONLY, then TT_FC[2:1] does not exist and TT_FC[2:1] always reads back 2’b10. If DMAH_CH2_FC is set to DST_FC_ONLY, then TT_FC[2:1] does not exist and TT_FC[2:1] always reads back 2’b11. For multi-block transfers using linked list operation, TT_FC must be constant for all blocks of this multi-block transfer. **Values:** - 0x0 (TT_FC_0): Transfer type is Memory to Memory and Flow Controller is HDMA - 0x1 (TT_FC_1): Transfer type is Memory to Peripheral and Flow Controller is HDMA - 0x2 (TT_FC_2): Transfer type is Peripheral to Memory and Flow Controller is HDMA - 0x3 (TT_FC_3): Transfer type is Peripheral to Peripheral and Flow Controller is HDMA - 0x4 (TT_FC_4): Transfer type is Peripheral to Memory and Flow Controller is Peripheral - 0x5 (TT_FC_5): Transfer type is Peripheral to Peripheral and Flow Controller is Source Peripheral - 0x6 (TT_FC_6): Transfer type is Memory to Peripheral and Flow Controller is Peripheral - 0x7 (TT_FC_7): Transfer type is Peripheral to Peripheral and Flow Controller is Destination Peripheral **Value after reset:** 0x3 **Volatile:** true
19	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0 **Volatile:** true
18	RW	**DST_SCATTER_EN** Destination scatter enable. Scatter on the destination side is applicable only when the CTL2.DINC bit indicates an incrementing or decrementing address control. **Values:** - 0x0 (DST_SCATTER_DISABLE): Destination Scatter is disabled - 0x1 (DST_SCATTER_ENABLE): Destination Scatter is enabled **Value after reset:** 0x0 **Volatile:** true
17	RW	**SRC_GATHER_EN** Destination gather enable. Gather on the destination side is applicable only when the CTL2.SINC bit indicates an incrementing or decrementing address control. **Values:** - 0x0 (SRC_SCATTER_DISABLE): Source gather is disabled - 0x1 (SRC_SCATTER_ENABLE): Source gather is enabled **Value after reset:** 0x0 **Volatile:** true

16:14	RW	**SRC_MSIZE** Source Burst Transaction Length. Number of data items, each of width CTL2.SRC_TR_WIDTH, to be read from the source every time a burst transferred request is made from either the corresponding hardware or software handshaking interface. **NOTE:** This value is not related to the AHB bus master HBURST bus. For information on the decoding for this field, see the ”Setting Up Transfers” section and for more inforamtion about this field, see the ”Choosing the Receive Watermark level” section in the HDMA Databook. **Values:** - 0x0 (SRC_MSIZE_0): Number of data items to be transferred is 1 - 0x1 (SRC_MSIZE_1): Number of data items to be transferred is 4 - 0x2 (SRC_MSIZE_2): Number of data items to be transferred is 8 - 0x3 (SRC_MSIZE_3): Number of data items to be transferred is 16 - 0x4 (SRC_MSIZE_4): Number of data items to be transferred is 32 - 0x5 (SRC_MSIZE_5): Number of data items to be transferred is 64 - 0x6 (SRC_MSIZE_6): Number of data items to be transferred is 128 - 0x7 (SRC_MSIZE_7): Number of data items to be transferred is 256 **Value after reset:** 0x1 **Volatile:** true
13:11	RW	**DEST_MSIZE** Destination Burst Transaction Length. Number of data items, each of width CTL2.DST_TR_WIDTH, to be written to the destination every time a destination burst transaction request is made from either the corresponding hardware or software handshaking interface. **NOTE:** This value is not related to the AHB bus master HBURST bus. **Values:** - 0x0 (DEST_MSIZE_0): Number of data items to be transferred is 1 - 0x1 (DEST_MSIZE_1): Number of data items to be transferred is 4 - 0x2 (DEST_MSIZE_2): Number of data items to be transferred is 8 - 0x3 (DEST_MSIZE_3): Number of data items to be transferred is 16 - 0x4 (DEST_MSIZE_4): Number of data items to be transferred is 32 - 0x5 (DEST_MSIZE_5): Number of data items to be transferred is 64 - 0x6 (DEST_MSIZE_6): Number of data items to be transferred is 128 - 0x7 (DEST_MSIZE_7): Number of data items to be transferred is 256 **Value after reset:** 0x1 **Volatile:** true
10:9	RW	**SINC** Source Address Increment. Indicates whether to increment or decrement the source address on every source transfer. If the device is fetching data from a source peripheral FIFO with a fixed address, then set this field to ”No change”. **Values:** - 0x0 (SINC_0): Increments the source address - 0x1 (SINC_1): Decrements the source address - 0x2 (SINC_2): No change in the source address - 0x3 (SINC_3): No change in the source address **Value after reset:** 0x0 **Volatile:** true

8 :7	RW	**DINC** Distination Address Increment. Indicates whether to increment or decrement the destination address on every destination transfer. If your device is writing data to a destination peripheral FIFO with a fixed address, then set this field to ”No Change”. **Values:** - 0x0 (DINC_0): Increments the distination address - 0x1 (DINC_1): Decrements the distination address - 0x2 (DINC_2): No change in the distination address - 0x3 (DINC_3): No change in the distination address **Value after reset:** 0x0 **Volatile:** true
6 :4	RW	**SRC_TR_WIDTH** Source Transfer Width. Mapped to AHB bus hsize. For a non-memory peripheral, typically the peripheral (source) FIFO width. This value must be less than or equl to DMAH_Mk_HDATA_WIDTH, where k is the AHB layer 1 to 4 where the source resides. **Values:** - 0x0 (SRC_TR_WIDTH_0): Source transfer width is 8 bits - 0x1 (SRC_TR_WIDTH_1): Source transfer width is 16 bits - 0x2 (SRC_TR_WIDTH_2): Source transfer width is 32 bits - 0x3 (SRC_TR_WIDTH_3): Source transfer width is 64 bits - 0x4 (SRC_TR_WIDTH_4): Source transfer width is 128 bits - 0x5 (SRC_TR_WIDTH_5): Source transfer width is 256 bits - 0x6 (SRC_TR_WIDTH_6): Source transfer width is 256 bits - 0x7 (SRC_TR_WIDTH_7): Source transfer width is 256 bits **Value after reset:** 0x2 **Volatile:** true
3 :1	RW	**DST_TR_WIDTH** Destination Transfer Width. Mapped to AHB bus hsize. For a non-memory peripheral, typically the peripheral (destination) FIFO width. This value must be less than or equal to DMAH_Mk_HDATA_WIDTH where k is the AHB layer 1 to 4 where the destination resides. For the decoding of this field, see the ”Setting Up Transfers” section in the HDMA Databook. **Values:** - 0x0 (DST_TR_WIDTH_0): Destination transfer width is 8 bits - 0x1 (DST_TR_WIDTH_1): Destination transfer width is 16 bits - 0x2 (DST_TR_WIDTH_2): Destination transfer width is 32 bits - 0x3 (DST_TR_WIDTH_3): Destination transfer width is 64 bits - 0x4 (DST_TR_WIDTH_4): Destination transfer width is 128 bits - 0x5 (DST_TR_WIDTH_5): Destination transfer width is 256 bits - 0x6 (DST_TR_WIDTH_6): Destination transfer width is 256 bits - 0x7 (DST_TR_WIDTH_7): Destination transfer width is 256 bits **Value after reset:** 0x2 **Volatile:** true

0	RW	**INT_EN** Interrupt Enable Bit. If set, then all interrupt-generating sources are enabled. Functions as a global mask bit for all interrupts for the channel; raw* interrupt registers still assert if CTL2.INT_EN=0. **Values:** - 0x0 (INTERRUPT_DISABLE): Interrupt is disabled - 0x1 (INTERRUPT_ENABLE): Interrupt is enabled **Value after reset:** 0x1 **Volatile:** true

Offset: 0D0h	**SSTAT2: Source Status Register for Channel 2**	**Init = 0**
63:32	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0 **Volatile:** true
31:0	RW	**SSTAT** Source status information retrieved by hardware from the address pointed to by the contents of the STATAR2 register **Value after reset:** 0x0 **Volatile:** true

Offset: 0D8h	**DSTAT2: Destination Status Register for Channel 2**	**Init = 0**
63:32	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0 **Volatile:** true
31:0	RW	**DSTAT** Destination status information retrieved by hardware from the address pointed to by the contents of DSTATAR2 register. **Value after reset:** 0x0 **Volatile:** true

Offset: 0E0h	**SSTATAR2: Source Status Address Register for Channel 2**	**Init = 0**
63:32	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0 **Volatile:** true
31:0	RW	**SSTATAR** Pointer from where hardware can fetch the source status information, which is registered in the SSTAT2 register and written out to the SSTAT2 register location of the LLI before the start of the next block. **Value after reset:** 0x0 **Volatile:** true

Offset: 0E8h		DSTATAR2: Source Status Address Register for Channel 2	Init = 0
63:32	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0	
31:0	RW	**DSTATAR** Pointer from where hardware can fetch the destination status information, which is registered in the DSTAT2 register and written out to the DSTAT2 register location of the LLI before the start of the next block. **Value after reset:** 0x0 **Volatile:** true	

Offset: 0F0h		CFG2: Configuration Register for Channel 2	Init = 0
63:47	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0 **Volatile:** true	
46:43	RW	**DEST_PER** Destination hardware interface. Assigns a hardware handshaking interface (0 : DMAH_NUM_HS_INT-1) to the destination of channel 2 if the CFG2.HS_SEL_DST field is 0; otherwise, this field is ignored. The channel can then communicate with the destination peripheral connected to that interface through the assigned hardware handshaking interface. **Note1:** For correct HDMA operation, only one peripheral (source or destination) should be assigned to the same handshaking interface. **Note2:** This field does not exist if the configuration parameter DMAH_NUM_HS_INT is set to 0. **Value after reset:** 0x0 **Volatile:** true	
42:39	RW	**SRC_PER** Source Hardware Interface. Assigns a hardware handshaking interface (0 : DMAH_NUM_HS_INT-1) to the source of channel 2 if the CFG2.HS_SEL_SRC field is 0; otherwise, this field is ignored. The channel can then communicate with the source peripheral connected to that interface through the assigned hardware handshaking interface. **Note1:** For correct HDMA operation, only one peripheral (source or destination) should be assigned to the same handshaking interface. **Note2:** This field does not exist if the configuration parameter DMAH_NUM_HS_INT is set to 0. **Value after reset:** 0x0 **Volatile:** true	

38	RW	**SS_UPD_EN** Source Status Update Enable. Source status information is fetched only from the location pointed to by the SSTATAR2 register, stored in the SSTAT2 register and written out to the SSTAT2 location of the LLI, if SS_UPD_EN is high. **Note:** This enalbe is applicable only if DMAH_CH2_STAT_SRC is set to True. This field does not exist if the configuration parameter DMAH_CH2_STAT_SRC is set to False; in this case, the read-back value is always 0. **Values:** - 0x0 (DISABLED): Source Status Update is disabled. - 0x1 (ENABLED): Source Status Update is enabled. **Value after reset:** 0x0 **Volatile:** true
37	RW	**DS_UPD_EN** Destination Status Update Enable. Destination status information is fetched only from the location pointed to by the DSTATAR0 register, stored in the DSTAT2 register and written out to the DSTAT2 location of the LLI, if DS_UPD_EN is high. This field does not exist if the configuration parameter DMAH_CH1_SATAT_DST is set to False; in this case, the read-back value is always 0. **Values:** - 0x0 (DISABLED): Destination Status Update is disabled. - 0x1 (ENABLED): Destination Status Update is enabled. **Value after reset:** 0x0 **Volatile:** true
36:34	RW	**PROTCTL** Protection Control bits used to drive the AHB HPROT[3:1] bus. The AMBA Specification recommends that the default of HPROT indicates a non-cached, non-buffered, privileged data access. The reset value is used to indicate such an access. HPROT[0] is tied high because all transfers are data accesses, as there are no opcode fetches. There is a one-to-one mapping of these register bits to the HPROT[3:1] master interface signals. Mapping of HPROT bus is as follows: - 1'b1 to HPROT[0] - CFG1.PROTCTL[1] to HPROT[1] - CFG1.PROTCTL[2] to HPROT[2] - CFG1.PROTCTL[3] to HPROT[3] **Value after reset:** 0x1 **Volatile:** true
33	RW	**FIFO_MODE** FIFO Mode Select. Determines how much space or data needs to be available in the FIFO before a burst transaction request is serviced. **Values:** - 0x0 (FIFO_MODE_0): Space/data available for single AHB transfer of the specified transfer width - 0x1 (FIFO_MODE_1): Data available is greater than or equal to half the FIFO depth for destination transfers and space available is greater than half the fifo depth for source transfers. The exceptions are at the end of a burst transaction request or at the end of a block transfer. **Value after reset:** 0x0 **Volatile:** true

32	RW	**FCMODE** Flow Control Mode. Determines when source transaction requests are serviced when the Destination Peripheral is the flow controller. **Values:** - 0x0 (FCMODE_0): Source transaction requests are serviced when they occur. Data pre-fetching is enabled - 0x1 (FCMODE_1): Source transaction requests are not serviced until a destination transaction request occurs. In this mode, the amount of data transferred from the source is limited so that it is guaranteed to be transferred to the destination prior to block termination by the destination. Data pre-fetching is disabled. **Value after reset:** 0x0 **Volatile:** true
31	RW	**RELOAD_DST** Automatic Destination Reload. The DAR2 register can be automatically reloaded from its initial value at the end of every block for multi-block transfers. A new block transfer is then initiated. This register does not exist if the configuration parameter DMAH_CH2_MULTI_BLK_EN is not selected; in this case, the read-back value is always 0. **Values:** - 0x0 (DISABLE): Destination Reload Disabled. - 0x1 (ENABLE): Destination Reload Enabled. **Value after reset:** 0x0 **Volatile:** true
30	RW	**RELOAD_SRC** Automatic Source Reload. The SAR2 register can be automatically reloaded from its initial value at the end of every block for multi-block transfers. A new block transfer is then initiated. This field does not exist if the configuration parameter DMAH_CH2_MULTI_BLK_EN is not selected; in this case, the read-back value is always 0. **Values:** - 0x0 (DISABLE): Source Reload Disabled. - 0x1 (ENABLE): Source Reload Enabled. **Value after reset:** 0x0 **Volatile:** true
29:20	R	**MAX_ABRST** Maximum AMBA Burst Length. Maximum AMBA burst length that is used for DMA transfers on this channel. A value of 0 indicates that software is not limiting the maximum AMBA burst length for DMA transfers on this channel. This field does not exist if the configuration parameter DMAH_MABRST is not selected; in this case, the read-back value is always 0, and the maximum AMBA burst length cannot be limited by software. **Value after reset:** 0x0 **Volatile:** true

19	RW	**SRC_HS_POL** Source Handshaking Interface Polarity. **Values:** - 0x0 (ACTIVE_HIGH): Source Handshaking Interface Polarity is Active high - 0x1 (ACTIVE_LOW): Source Handshaking Interface Polarity is Active low **Value after reset:** 0x0 **Volatile:** true
18	RW	**DST_HS_POL** Destination Handshaking Interface Polarity. **Values:** - 0x0 (ACTIVE_HIGH): Destination Handshaking Interface Polarity is Active high - 0x1 (ACTIVE_LOW): Destination Handshaking Interface Polarity is Active low **Value after reset:** 0x0 **Volatile:** true
17	RW	**LOCK_B** Bus Lock Bit. When active, the AHB bus master signal hlock is asserted for the duration specified in CFG0.LOCK_B_L. For more information, refer to ”Locked DMA Transfers”. This field does not exist if the configuration parameter DMAH_CH2_LOCK_EN is set to False; in this case, the read-back value is always 0. **Values:** - 0x0 (DISABLED): Bus lock bit is not enabled - 0x1 (ENABLED): Bus lock bit is enabled **Value after reset:** 0x0 **Volatile:** true
16	RW	**LOCK_CH** Channel Lock Bit. When the channel is granted control of the master bus interface and if the CFG2.LOCK_CH bit is asserted, then no other channels are granted control of the master bus interface for the duration specified in CFG2.LOCK_CH_L. Indicates to the master bus interface arbiter that this channel wants exclusive access to the master bus interface for the duration specified in CFG2.LOCK_CH_L. This field does not exist if the configuration parameter DMAH_CH2_LOCK_EN is set to False; in this case, the read-back value is always 0. **Values:** - 0x0 (DISABLED): Channel lock bit is not enabled - 0x1 (ENABLED): Channel lock bit is enabled **Value after reset:** 0x0 **Volatile:** true
14:15	RW	**LOCK_B_L** Bus lock level. Indicates the duration over which CFG2.LOCK_B bit applies. This field does not exist if the parameter DMAH_CH2_LOCK_EN is set to False; in this case, the read-back value is always 0. **Values:** - 0x0 (LOCK_B_L_0): Over complete DMA transfer - 0x1 (LOCK_B_L_1): Over complete DMA block transfer - 0x2 (LOCK_B_L_2): Over complete DMA transaction - 0x3 (LOCK_B_L_3): Over complete DMA transaction **Value after reset:** 0x0 **Volatile:** true

13:12	RW	**LOCK_CH_L** Channel Local Level. Indicates the duration over which CFG2.LOCK_CH applies. This field does not exist if the configuration parameter DMAH_CH2_LOCK_EN is set to False; in this case, the read-back value is always 0. **Values:** - 0x0 (LOCK_CH_L_0): Over complete DMA transfer - 0x1 (LOCK_CH_L_1): Over complete DMA block transfer - 0x2 (LOCK_CH_L_2): Over complete DMA transaction - 0x3 (LOCK_CH_L_3): Over complete DMA transaction **Value after reset:** 0x0 **Volatile:** true
11	RW	**HS_SEL_SRC** Source Software or Hardware Handshaking Select. This register selects which of the handshaking interfaces - hardware or software - is active for source requests on this channel. If the source peripheral is memory, then this bit is ignored. **Values:** - 0x0 (HARDWARE_HS): Hardware handshaking interface. Software initiated transaction requests are ignored. - 0x1 (SOFTWARE_HS): Software handshaking interface. Hardware initiated transaction requests are ignored. **Value after reset:** 0x1 **Volatile:** true
10	RW	**HS_SEL_DST** Destination Software or Hardware Handshaking Select. This register selects which of the handshaking interfaces - hardware or software - is active for destination requests on this channel. If the destination peripheral is memory, then this bit is ignored. **Values:** - 0x0 (HARDWARE_HS): Hardware handshaking interface. Software initiated transaction requests are ignored. - 0x1 (SOFTWARE_HS): Software handshaking interface. Hardware initiated transaction requests are ignored. **Value after reset:** 0x1 **Volatile:** true
9	R	**FIFO_EMPTY** Channel FIFO status. Indicates if there is data left in the channel FIFO. Can be used in conjunction with CFG2.CH_SUSP to cleanly disable a channel. For more information, refer to ”Disabling a Channel Prior to Transfer Completion”. **Values:** - 0x0 (NOT_EMPTY): Channel FIFO is not empty - 0x1 (EMPTY): Channel FIFO is empty **Value after reset:** 0x1 **Volatile:** true

8	R	**CH_SUSP** Channel Suspend. Suspends all DMA data transfers from the source until this bit is cleared. There is no guarantee that the current transaction will complete. Can also be used in conjunction with CFG2.FIFO_EMPTY to cleanly disable a channel without losing any data. For more information, refer to ”Disabling a Channel Prior to Transfer Completion”. **Values:** - 0x0 (NOT_SUSPENDED): DMA transfer from the source is not suspended - 0x1 (SUSPENDED): Suspend DMA transfer from the source **Value after reset:** 0x0 **Volatile:** true
7 :5	RW	**SRC_TR_WIDTH** Source Transfer Width. Mapped to AHB bus hsize. For a non-memory peripheral, typically the peripheral (source) FIFO width. This value must be less than or equl to DMAH_Mk_HDATA_WIDTH, where k is the AHB layer 1 to 4 where the source resides. **Values:** - 0x0 (SRC_TR_WIDTH_0): Source transfer width is 8 bits - 0x1 (SRC_TR_WIDTH_1): Source transfer width is 16 bits - 0x2 (SRC_TR_WIDTH_2): Source transfer width is 32 bits - 0x3 (SRC_TR_WIDTH_3): Source transfer width is 64 bits - 0x4 (SRC_TR_WIDTH_4): Source transfer width is 128 bits - 0x5 (SRC_TR_WIDTH_5): Source transfer width is 256 bits - 0x6 (SRC_TR_WIDTH_6): Source transfer width is 256 bits - 0x7 (SRC_TR_WIDTH_7): Source transfer width is 256 bits **Value after reset:** x **Volatile:** true
4 :0	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0

Offset: 0F8h	**SGR2: Source Gather Register for Channel 2**	**Init = 0**
63:32	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0
31:25	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0
24:19	RW	**SGC** Source Gather Count. Source contiguous tranfer count between successive gather boundaries. **Value after reset:** 0x0 **Volatile:** true

19:0	RW	**SGI** Source Gather Interval. **Value after reset:** 0x0 **Volatile:** true

Offset: 100h	**DSR2: Source Gather Register for Channel 2**	**Init = 0**
63:32	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0
31:25	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0
24:19	RW	**DSC** Destination Scatter Count. Destination contiguous transfer count between successive scatter boundaries. **Value after reset:** 0x0 **Volatile:** true
19:0	RW	**DSI** Destination Scatter Interval. **Value after reset:** 0x0 **Volatile:** true

Offset: 108h	**SAR3: Source Address for Channel 3**	**Init = 0**
63:32	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0 **Volatile:** true
31:0	RW	**SAR** Current Source Address of DMA transfer. Updated after each source transfer. The SINC field in the CTL3 register determines whether the address increments, decrements, or is left unchanged on every source transfer through the block transfer. **Value after reset:** 0x0 **Volatile:** true

Offset: 110h	**DAR3: Destination Address for Channel 3**	**Init = 0**
63:32	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0 **Volatile:** true

31:0	RW	**DAR** Current Destination address of DMA transfer. Updated after each destination transfer. The DINC field in the CTL3 register determines whether the address increments, decrements, or is left unchanged on every destination transfer throughout the block transfer. **Value after reset:** 0x0 **Volatile:** true

Offset: 118h	**LLP3: Linked List Pointer Register for Channel 3**	**Init = 0**
63:32	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0 **Volatile:** true
31:2	RW	**LOC** Starting Address In Memory of next LLI if block chaining is enabled. Note that the two LSBs of the starting address are not stored because the address is assumed to be aligned to a 32-bit boundary. LLI accesses are always 32-bit accesses (Hsize=2) aligned to 32-bit boundaries and cannot be changed or programmed to anything other than 32-bit. **Value after reset:** 0x0 **Volatile:** true
1 :0	RW	**LMS** List Master Select. Identifies the AHB layer/interface where the memory device that stores the next linked list item resides. This field does not exist if the configuration parameter is not set to NO_HARDCODE. In this case, the read-back value is always the hardcoded value. The maximum value of this field that can be read back is DMAH_NUM_MASTER_INT-1. **Values:** - 0x0 (LIST_MASTER_SELECT_1): The memory device stores the next linked list item on AHB master 1 - 0x1 (LIST_MASTER_SELECT_2): The memory device stores the next linked list item on AHB master 2 - 0x2 (LIST_MASTER_SELECT_3): The memory device stores the next linked list item on AHB master 3 - 0x3 (LIST_MASTER_SELECT_4): The memory device stores the next linked list item on AHB master 4 **Value after reset:** 0x0 **Volatile:** true

Offset: 120h	**CTL3: Control Register for Channel 3**	**Init = 0**
63:45	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0 **Volatile:** true

44	RW	**DONE** Done bit. If status write-back is enabled, the upper word of the control register, CTL3[63:32], is written to the control register location of the Linked List Item (LLI) in system memory at the end of the block transfer with the done bit set. Software can poll the LLI CTL3.DONE bit to see when a block transfer is complete. The LLI CTL3.DONE bit should be cleared when the linked lists are set up in memory prior to enabling the channel. LLI accesses are always 32-bit accesses (Hsize=2) aligned to 32-bit boundaries and cannot be changed or programmed to anything other than 32-bit. For more information, refer to ”Multi-Block Transfers”. **Values:** - 0x0 (DISABLED): DONE bit is deasserted the enb of block transfer - 0x1 (ENABLED): SET the DONE bit at the end of block transfer **Value after reset:** 0x0 **Volatile:** true
43:37	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0 **Volatile:** true
36:32	RW	**BLOCK_TS** Block Transfer Size. When the HDMA is the flow controller, the user writes this field before the channel is enabled in order to indicate the block size. The number programmed into BLOCK_TS indicates the total number of single transactions to perform for every block transfer; a single transaction is mapped to a single AMBA beat **Width:** The width of single transaction is determined by CTL3.SRC_TR_WIDTH. For further information on setting this field, refer to ”Transfer Operation”. Once the transfer starts, the read-back value is the total number of data items already read from the source peripheral, regardless of what is the flow controller. When the source or destination peripheral is assigned as the flow controller, then the maximum block size that can be read back saturates at DMAH_CH3_MAX_BLK_SIZE, but the actual block size can be greater. **Value after reset:** 0x2 **Volatile:** true
31:29	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0 **Volatile:** true
28	RW	**LLP_SRC_EN** Block chaining is enabled on the source side only if the LLP_SRC_EN field is high and LLP3.LOC is non-zero. For more information, see ”Block Chaining using Linked Lists”. **Values:** - 0x0 (LLP_SRC_DISABLE): Block chaining using Linked List is disabled on the Source side - 0x1 (LLP_SRC_ENABLE): Block chaining using Linked List is enabled on the Source side **Value after reset:** 0x0 **Volatile:** true

27	RW	**LLP_DST_EN** Block chaining is enabled on the destination side only if LLP_DST_EN field is high and LLP3.LOC is non-zero. For more information, see ”Block Chaining using Linked Lists” **Values:** - 0x0 (LLP_DST_DISABLE): Block chaining using Linked List is disabled on the Destination side - 0x1 (LLP_DST_ENABLE): Block chaining using Linked List is enabled on the Destination side **Value after reset:** 0x0 **Volatile:** true
26:25	RW	**SMS** Source Master Select. Identifies the Master Interface layer where the source device (peripheral or memory) resides. The maximum value of this field that can be read back is DMAH_NUM_MASTER_INT-1. **Values:** - 0x0 (SMS_0): Source device (peripheral or memory) is accessed from AHB master 1 - 0x1 (SMS_1): Source device (peripheral or memory) is accessed from AHB master 2 - 0x2 (SMS_2): Source device (peripheral or memory) is accessed from AHB master 3 - 0x3 (SMS_3): Source device (peripheral or memory) is accessed from AHB master 4 **Value after reset:** 0x0 **Volatile:** true
24:23	RW	**DMS** Destination Master Select. Identifies the Master Interface layer where the destination device (peripheral or memory) resides. The maximum value of this field that can be read back is DMAH_NUM_MASTER_INT-1. **Values:** - 0x0 (DMS_0): Destination device (peripheral or memory) is accessed from AHB master 1 - 0x1 (DMS_1): Destination device (peripheral or memory) is accessed from AHB master 2 - 0x2 (DMS_2): Destination device (peripheral or memory) is accessed from AHB master 3 - 0x3 (DMS_3): Destination device (peripheral or memory) is accessed from AHB master 4 **Value after reset:** 0x0 **Volatile:** true

22:20	RW	**TT_FC** Transfer Type and Flow Control. Flow control can be assigned to the HDMA, the source peripheral, or the destination peripheral. For more information on transfer types and flow control, refer to ”Setup/Operation of the HDMA Transfers”. Dependencies: If the configuration parameter DMAH_CH3_FC is set to DMA_FC_ONLY, then TT_FC[2] does not exist and TT_FC[2] always reads back 0. If DMAH_CH3_FC is set to SRC_FC_ONLY, then TT_FC[2:1] does not exist and TT_FC[2:1] always reads back 2’b10. If DMAH_CH3_FC is set to DST_FC_ONLY, then TT_FC[2:1] does not exist and TT_FC[2:1] always reads back 2’b11. For multi-block transfers using linked list operation, TT_FC must be constant for all blocks of this multi-block transfer. **Values:** - 0x0 (TT_FC_0): Transfer type is Memory to Memory and Flow Controller is HDMA - 0x1 (TT_FC_1): Transfer type is Memory to Peripheral and Flow Controller is HDMA - 0x2 (TT_FC_2): Transfer type is Peripheral to Memory and Flow Controller is HDMA - 0x3 (TT_FC_3): Transfer type is Peripheral to Peripheral and Flow Controller is HDMA - 0x4 (TT_FC_4): Transfer type is Peripheral to Memory and Flow Controller is Peripheral - 0x5 (TT_FC_5): Transfer type is Peripheral to Peripheral and Flow Controller is Source Peripheral - 0x6 (TT_FC_6): Transfer type is Memory to Peripheral and Flow Controller is Peripheral - 0x7 (TT_FC_7): Transfer type is Peripheral to Peripheral and Flow Controller is Destination Peripheral **Value after reset:** 0x3 **Volatile:** true
19	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0 **Volatile:** true
18	RW	**DST_SCATTER_EN** Destination scatter enable. Scatter on the destination side is applicable only when the CTL3.DINC bit indicates an incrementing or decrementing address control. **Values:** - 0x0 (DST_SCATTER_DISABLE): Destination Scatter is disabled - 0x1 (DST_SCATTER_ENABLE): Destination Scatter is enabled **Value after reset:** 0x0 **Volatile:** true
17	RW	**SRC_GATHER_EN** Destination gather enable. Gather on the destination side is applicable only when the CTL3.SINC bit indicates an incrementing or decrementing address control. **Values:** - 0x0 (SRC_SCATTER_DISABLE): Source gather is disabled - 0x1 (SRC_SCATTER_ENABLE): Source gather is enabled **Value after reset:** 0x0 **Volatile:** true

16:14	RW	**SRC_MSIZE** Source Burst Transaction Length. Number of data items, each of width CTL3.SRC_TR_WIDTH, to be read from the source every time a burst transferred request is made from either the corresponding hardware or software handshaking interface. **NOTE:** This value is not related to the AHB bus master HBURST bus. For information on the decoding for this field, see the ”Setting Up Transfers” section and for more inforamtion about this field, see the ”Choosing the Receive Watermark level” section in the HDMA Databook. **Values:** - 0x0 (SRC_MSIZE_0): Number of data items to be transferred is 1 - 0x1 (SRC_MSIZE_1): Number of data items to be transferred is 4 - 0x2 (SRC_MSIZE_2): Number of data items to be transferred is 8 - 0x3 (SRC_MSIZE_3): Number of data items to be transferred is 16 - 0x4 (SRC_MSIZE_4): Number of data items to be transferred is 32 - 0x5 (SRC_MSIZE_5): Number of data items to be transferred is 64 - 0x6 (SRC_MSIZE_6): Number of data items to be transferred is 128 - 0x7 (SRC_MSIZE_7): Number of data items to be transferred is 256 **Value after reset:** 0x1 **Volatile:** true
13:11	RW	**DEST_MSIZE** Destination Burst Transaction Length. Number of data items, each of width CTL3.DST_TR_WIDTH, to be written to the destination every time a destination burst transaction request is made from either the corresponding hardware or software handshaking interface. **NOTE:** This value is not related to the AHB bus master HBURST bus. **Values:** - 0x0 (DEST_MSIZE_0): Number of data items to be transferred is 1 - 0x1 (DEST_MSIZE_1): Number of data items to be transferred is 4 - 0x2 (DEST_MSIZE_2): Number of data items to be transferred is 8 - 0x3 (DEST_MSIZE_3): Number of data items to be transferred is 16 - 0x4 (DEST_MSIZE_4): Number of data items to be transferred is 32 - 0x5 (DEST_MSIZE_5): Number of data items to be transferred is 64 - 0x6 (DEST_MSIZE_6): Number of data items to be transferred is 128 - 0x7 (DEST_MSIZE_7): Number of data items to be transferred is 256 **Value after reset:** 0x1 **Volatile:** true
10:9	RW	**SINC** Source Address Increment. Indicates whether to increment or decrement the source address on every source transfer. If the device is fetching data from a source peripheral FIFO with a fixed address, then set this field to ”No change”. **Values:** - 0x0 (SINC_0): Increments the source address - 0x1 (SINC_1): Decrements the source address - 0x2 (SINC_2): No change in the source address - 0x3 (SINC_3): No change in the source address **Value after reset:** 0x0 **Volatile:** true

8 :7	RW	**DINC** Distination Address Increment. Indicates whether to increment or decrement the destination address on every destination transfer. If your device is writing data to a destination peripheral FIFO with a fixed address, then set this field to ”No Change”. **Values:** - 0x0 (DINC_0): Increments the distination address - 0x1 (DINC_1): Decrements the distination address - 0x2 (DINC_2): No change in the distination address - 0x3 (DINC_3): No change in the distination address **Value after reset:** 0x0 **Volatile:** true
6 :4	RW	**SRC_TR_WIDTH** Source Transfer Width. Mapped to AHB bus hsize. For a non-memory peripheral, typically the peripheral (source) FIFO width. This value must be less than or equl to DMAH_Mk_HDATA_WIDTH, where k is the AHB layer 1 to 4 where the source resides. **Values:** - 0x0 (SRC_TR_WIDTH_0): Source transfer width is 8 bits - 0x1 (SRC_TR_WIDTH_1): Source transfer width is 16 bits - 0x2 (SRC_TR_WIDTH_2): Source transfer width is 32 bits - 0x3 (SRC_TR_WIDTH_3): Source transfer width is 64 bits - 0x4 (SRC_TR_WIDTH_4): Source transfer width is 128 bits - 0x5 (SRC_TR_WIDTH_5): Source transfer width is 256 bits - 0x6 (SRC_TR_WIDTH_6): Source transfer width is 256 bits - 0x7 (SRC_TR_WIDTH_7): Source transfer width is 256 bits **Value after reset:** 0x2 **Volatile:** true
3 :1	RW	**DST_TR_WIDTH** Destination Transfer Width. Mapped to AHB bus hsize. For a non-memory peripheral, typically the peripheral (destination) FIFO width. This value must be less than or equal to DMAH_Mk_HDATA_WIDTH where k is the AHB layer 1 to 4 where the destination resides. For the decoding of this field, see the ”Setting Up Transfers” section in the HDMA Databook. **Values:** - 0x0 (DST_TR_WIDTH_0): Destination transfer width is 8 bits - 0x1 (DST_TR_WIDTH_1): Destination transfer width is 16 bits - 0x2 (DST_TR_WIDTH_2): Destination transfer width is 32 bits - 0x3 (DST_TR_WIDTH_3): Destination transfer width is 64 bits - 0x4 (DST_TR_WIDTH_4): Destination transfer width is 128 bits - 0x5 (DST_TR_WIDTH_5): Destination transfer width is 256 bits - 0x6 (DST_TR_WIDTH_6): Destination transfer width is 256 bits - 0x7 (DST_TR_WIDTH_7): Destination transfer width is 256 bits **Value after reset:** 0x2 **Volatile:** true

0	RW	**INT_EN** Interrupt Enable Bit. If set, then all interrupt-generating sources are enabled. Functions as a global mask bit for all interrupts for the channel; raw* interrupt registers still assert if CTL2.INT_EN=0. **Values:** - 0x0 (INTERRUPT_DISABLE): Interrupt is disabled - 0x1 (INTERRUPT_ENABLE): Interrupt is enabled **Value after reset:** 0x1 **Volatile:** true

Offset: 128h	**SSTAT3: Source Status Register for Channel 3**	**Init = 0**
63:32	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0 **Volatile:** true
31:0	RW	**SSTAT** Source status information retrieved by hardware from the address pointed to by the contents of the STATAR3 register **Value after reset:** 0x0 **Volatile:** true

Offset: 130h	**DSTAT3: Destination Status Register for Channel 3**	**Init = 0**
63:32	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0 **Volatile:** true
31:0	RW	**DSTAT** Destination status information retrieved by hardware from the address pointed to by the contents of DSTATAR3 register. **Value after reset:** 0x0 **Volatile:** true

Offset: 138h	**SSTATAR3: Source Status Address Register for Channel 3**	**Init = 0**
63:32	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0 **Volatile:** true
31:0	RW	**SSTATAR** Pointer from where hardware can fetch the source status information, which is registered in the SSTAT3 register and written out to the SSTAT3 register location of the LLI before the start of the next block. **Value after reset:** 0x0 **Volatile:** true

Offset: 140h		DSTATAR3: Source Status Address Register for Channel 3	Init = 0
63:32	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0	
31:0	RW	**DSTATAR** Pointer from where hardware can fetch the destination status information, which is registered in the DSTAT3 register and written out to the DSTAT3 register location of the LLI before the start of the next block. **Value after reset:** 0x0 **Volatile:** true	

Offset: 148h		CFG3: Configuration Register for Channel 3	Init = 0
63:47	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0 **Volatile:** true	
46:43	RW	**DEST_PER** Destination hardware interface. Assigns a hardware handshaking interface (0 : DMAH_NUM_HS_INT-1) to the destination of channel 3 if the CFG3.HS_SEL_DST field is 0; otherwise, this field is ignored. The channel can then communicate with the destination peripheral connected to that interface through the assigned hardware handshaking interface. **Note1:** For correct HDMA operation, only one peripheral (source or destination) should be assigned to the same handshaking interface. **Note2:** This field does not exist if the configuration parameter DMAH_NUM_HS_INT is set to 0. **Value after reset:** 0x0 **Volatile:** true	
42:39	RW	**SRC_PER** Source Hardware Interface. Assigns a hardware handshaking interface (0 : DMAH_NUM_HS_INT-1) to the source of channel 3 if the CFG3.HS_SEL_SRC field is 0; otherwise, this field is ignored. The channel can then communicate with the source peripheral connected to that interface through the assigned hardware handshaking interface. **Note1:** For correct HDMA operation, only one peripheral (source or destination) should be assigned to the same handshaking interface. **Note2:** This field does not exist if the configuration parameter DMAH_NUM_HS_INT is set to 0. **Value after reset:** 0x0 **Volatile:** true	

38	RW	**SS_UPD_EN** Source Status Update Enable. Source status information is fetched only from the location pointed to by the SSTATAR3 register, stored in the SSTAT3 register and written out to the SSTAT3 location of the LLI, if SS_UPD_EN is high. **Note:** This enalbe is applicable only if DMAH_CH3_STAT_SRC is set to True. This field does not exist if the configuration parameter DMAH_CH3_STAT_SRC is set to False; in this case, the read-back value is always 0. **Values:** - 0x0 (DISABLED): Source Status Update is disabled. - 0x1 (ENABLED): Source Status Update is enabled. **Value after reset:** 0x0 **Volatile:** true
37	RW	**DS_UPD_EN** Destination Status Update Enable. Destination status information is fetched only from the location pointed to by the DSTATAR0 register, stored in the DSTAT3 register and written out to the DSTAT3 location of the LLI, if DS_UPD_EN is high. This field does not exist if the configuration parameter DMAH_CH1_SATAT_DST is set to False; in this case, the read-back value is always 0. **Values:** - 0x0 (DISABLED): Destination Status Update is disabled. - 0x1 (ENABLED): Destination Status Update is enabled. **Value after reset:** 0x0 **Volatile:** true
36:34	RW	**PROTCTL** Protection Control bits used to drive the AHB HPROT[3:1] bus. The AMBA Specification recommends that the default of HPROT indicates a non-cached, non-buffered, privileged data access. The reset value is used to indicate such an access. HPROT[0] is tied high because all transfers are data accesses, as there are no opcode fetches. There is a one-to-one mapping of these register bits to the HPROT[3:1] master interface signals. Mapping of HPROT bus is as follows: - 1'b1 to HPROT[0] - CFG1.PROTCTL[1] to HPROT[1] - CFG1.PROTCTL[2] to HPROT[2] - CFG1.PROTCTL[3] to HPROT[3] **Value after reset:** 0x1 **Volatile:** true
33	RW	**FIFO_MODE** FIFO Mode Select. Determines how much space or data needs to be available in the FIFO before a burst transaction request is serviced. **Values:** - 0x0 (FIFO_MODE_0): Space/data available for single AHB transfer of the specified transfer width - 0x1 (FIFO_MODE_1): Data available is greater than or equal to half the FIFO depth for destination transfers and space available is greater than half the fifo depth for source transfers. The exceptions are at the end of a burst transaction request or at the end of a block transfer. **Value after reset:** 0x0 **Volatile:** true

32	RW	**FCMODE** Flow Control Mode. Determines when source transaction requests are serviced when the Destination Peripheral is the flow controller. **Values:** - 0x0 (FCMODE_0): Source transaction requests are serviced when they occur. Data pre-fetching is enabled - 0x1 (FCMODE_1): Source transaction requests are not serviced until a destination transaction request occurs. In this mode, the amount of data transferred from the source is limited so that it is guaranteed to be transferred to the destination prior to block termination by the destination. Data pre-fetching is disabled. **Value after reset:** 0x0 **Volatile:** true
31	RW	**RELOAD_DST** Automatic Destination Reload. The DAR2 register can be automatically reloaded from its initial value at the end of every block for multi-block transfers. A new block transfer is then initiated. This register does not exist if the configuration parameter DMAH_CH3_MULTI_BLK_EN is not selected; in this case, the read-back value is always 0. **Values:** - 0x0 (DISABLE): Destination Reload Disabled. - 0x1 (ENABLE): Destination Reload Enabled. **Value after reset:** 0x0 **Volatile:** true
30	RW	**RELOAD_SRC** Automatic Source Reload. The SAR3 register can be automatically reloaded from its initial value at the end of every block for multi-block transfers. A new block transfer is then initiated. This field does not exist if the configuration parameter DMAH_CH3_MULTI_BLK_EN is not selected; in this case, the read-back value is always 0. **Values:** - 0x0 (DISABLE): Source Reload Disabled. - 0x1 (ENABLE): Source Reload Enabled. **Value after reset:** 0x0 **Volatile:** true
29:20	R	**MAX_ABRST** Maximum AMBA Burst Length. Maximum AMBA burst length that is used for DMA transfers on this channel. A value of 0 indicates that software is not limiting the maximum AMBA burst length for DMA transfers on this channel. This field does not exist if the configuration parameter DMAH_MABRST is not selected; in this case, the read-back value is always 0, and the maximum AMBA burst length cannot be limited by software. **Value after reset:** 0x0 **Volatile:** true

19	RW	**SRC_HS_POL** Source Handshaking Interface Polarity. **Values:** - 0x0 (ACTIVE_HIGH): Source Handshaking Interface Polarity is Active high - 0x1 (ACTIVE_LOW): Source Handshaking Interface Polarity is Active low **Value after reset:** 0x0 **Volatile:** true
18	RW	**DST_HS_POL** Destination Handshaking Interface Polarity. **Values:** - 0x0 (ACTIVE_HIGH): Destination Handshaking Interface Polarity is Active high - 0x1 (ACTIVE_LOW): Destination Handshaking Interface Polarity is Active low **Value after reset:** 0x0 **Volatile:** true
17	RW	**LOCK_B** Bus Lock Bit. When active, the AHB bus master signal hlock is asserted for the duration specified in CFG0.LOCK_B_L. For more information, refer to ”Locked DMA Transfers”. This field does not exist if the configuration parameter DMAH_CH3_LOCK_EN is set to False; in this case, the read-back value is always 0. **Values:** - 0x0 (DISABLED): Bus lock bit is not enabled - 0x1 (ENABLED): Bus lock bit is enabled **Value after reset:** 0x0 **Volatile:** true
16	RW	**LOCK_CH** Channel Lock Bit. When the channel is granted control of the master bus interface and if the CFG3.LOCK_CH bit is asserted, then no other channels are granted control of the master bus interface for the duration specified in CFG3.LOCK_CH_L. Indicates to the master bus interface arbiter that this channel wants exclusive access to the master bus interface for the duration specified in CFG3.LOCK_CH_L. This field does not exist if the configuration parameter DMAH_CH3_LOCK_EN is set to False; in this case, the read-back value is always 0. **Values:** - 0x0 (DISABLED): Channel lock bit is not enabled - 0x1 (ENABLED): Channel lock bit is enabled **Value after reset:** 0x0 **Volatile:** true
14:15	RW	**LOCK_B_L** Bus lock level. Indicates the duration over which CFG3.LOCK_B bit applies. This field does not exist if the parameter DMAH_CH3_LOCK_EN is set to False; in this case, the read-back value is always 0. **Values:** - 0x0 (LOCK_B_L_0): Over complete DMA transfer - 0x1 (LOCK_B_L_1): Over complete DMA block transfer - 0x2 (LOCK_B_L_2): Over complete DMA transaction - 0x3 (LOCK_B_L_3): Over complete DMA transaction **Value after reset:** 0x0 **Volatile:** true

13:12	RW	**LOCK_CH_L** Channel Local Level. Indicates the duration over which CFG3.LOCK_CH applies. This field does not exist if the configuration parameter DMAH_CH3_LOCK_EN is set to False; in this case, the read-back value is always 0. **Values:** - 0x0 (LOCK_CH_L_0): Over complete DMA transfer - 0x1 (LOCK_CH_L_1): Over complete DMA block transfer - 0x2 (LOCK_CH_L_2): Over complete DMA transaction - 0x3 (LOCK_CH_L_3): Over complete DMA transaction **Value after reset:** 0x0 **Volatile:** true
11	RW	**HS_SEL_SRC** Source Software or Hardware Handshaking Select. This register selects which of the handshaking interfaces - hardware or software - is active for source requests on this channel. If the source peripheral is memory, then this bit is ignored. **Values:** - 0x0 (HARDWARE_HS): Hardware handshaking interface. Software initiated transaction requests are ignored. - 0x1 (SOFTWARE_HS): Software handshaking interface. Hardware initiated transaction requests are ignored. **Value after reset:** 0x1 **Volatile:** true
10	RW	**HS_SEL_DST** Destination Software or Hardware Handshaking Select. This register selects which of the handshaking interfaces - hardware or software - is active for destination requests on this channel. If the destination peripheral is memory, then this bit is ignored. **Values:** - 0x0 (HARDWARE_HS): Hardware handshaking interface. Software initiated transaction requests are ignored. - 0x1 (SOFTWARE_HS): Software handshaking interface. Hardware initiated transaction requests are ignored. **Value after reset:** 0x1 **Volatile:** true
9	R	**FIFO_EMPTY** Channel FIFO status. Indicates if there is data left in the channel FIFO. Can be used in conjunction with CFG3.CH_SUSP to cleanly disable a channel. For more information, refer to ”Disabling a Channel Prior to Transfer Completion”. **Values:** - 0x0 (NOT_EMPTY): Channel FIFO is not empty - 0x1 (EMPTY): Channel FIFO is empty **Value after reset:** 0x1 **Volatile:** true

8	R	**CH_SUSP** Channel Suspend. Suspends all DMA data transfers from the source until this bit is cleared. There is no guarantee that the current transaction will complete. Can also be used in conjunction with CFG3.FIFO_EMPTY to cleanly disable a channel without losing any data. For more information, refer to ”Disabling a Channel Prior to Transfer Completion”. **Values:** - 0x0 (NOT_SUSPENDED): DMA transfer from the source is not suspended - 0x1 (SUSPENDED): Suspend DMA transfer from the source **Value after reset:** 0x0 **Volatile:** true
7 :5	RW	**SRC_TR_WIDTH** Source Transfer Width. Mapped to AHB bus hsize. For a non-memory peripheral, typically the peripheral (source) FIFO width. This value must be less than or equl to DMAH_Mk_HDATA_WIDTH, where k is the AHB layer 1 to 4 where the source resides. **Values:** - 0x0 (SRC_TR_WIDTH_0): Source transfer width is 8 bits - 0x1 (SRC_TR_WIDTH_1): Source transfer width is 16 bits - 0x2 (SRC_TR_WIDTH_2): Source transfer width is 32 bits - 0x3 (SRC_TR_WIDTH_3): Source transfer width is 64 bits - 0x4 (SRC_TR_WIDTH_4): Source transfer width is 128 bits - 0x5 (SRC_TR_WIDTH_5): Source transfer width is 256 bits - 0x6 (SRC_TR_WIDTH_6): Source transfer width is 256 bits - 0x7 (SRC_TR_WIDTH_7): Source transfer width is 256 bits **Value after reset:** x **Volatile:** true
4 :0	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0

Offset: 150h	**SGR3: Source Gather Register for Channel 3**	**Init = 0**
63:32	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0
31:25	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0
24:19	RW	**SGC** Source Gather Count. Source contiguous tranfer count between successive gather boundaries. **Value after reset:** 0x0 **Volatile:** true

19:0	RW	**SGI** Source Gather Interval. **Value after reset:** 0x0 **Volatile:** true

Offset: 158h	**DSR3: Source Gather Register for Channel 3**	**Init = 0**
63:32	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0
31:25	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0
24:19	RW	**DSC** Destination Scatter Count. Destination contiguous transfer count between successive scatter boundaries. **Value after reset:** 0x0 **Volatile:** true
19:0	RW	**DSI** Destination Scatter Interval. **Value after reset:** 0x0 **Volatile:** true

Offset: 160h	**SAR4: Source Address for Channel 4**	**Init = 0**
63:32	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0 **Volatile:** true
31:0	RW	**SAR** Current Source Address of DMA transfer. Updated after each source transfer. The SINC field in the CTL4 register determines whether the address increments, decrements, or is left unchanged on every source transfer through the block transfer. **Value after reset:** 0x0 **Volatile:** true

Offset: 168h	**DAR4: Destination Address for Channel 4**	**Init = 0**
63:32	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0 **Volatile:** true

31:0	RW	**DAR** Current Destination address of DMA transfer. Updated after each destination transfer. The DINC field in the CTL4 register determines whether the address increments, decrements, or is left unchanged on every destination transfer throughout the block transfer. **Value after reset:** 0x0 **Volatile:** true

Offset: 170h	**LLP4: Linked List Pointer Register for Channel 4**	**Init = 0**
63:32	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0 **Volatile:** true
31:2	RW	**LOC** Starting Address In Memory of next LLI if block chaining is enabled. Note that the two LSBs of the starting address are not stored because the address is assumed to be aligned to a 32-bit boundary. LLI accesses are always 32-bit accesses (Hsize=2) aligned to 32-bit boundaries and cannot be changed or programmed to anything other than 32-bit. **Value after reset:** 0x0 **Volatile:** true
1 :0	RW	**LMS** List Master Select. Identifies the AHB layer/interface where the memory device that stores the next linked list item resides. This field does not exist if the configuration parameter is not set to NO_HARDCODE. In this case, the read-back value is always the hardcoded value. The maximum value of this field that can be read back is DMAH_NUM_MASTER_INT-1. **Values:** - 0x0 (LIST_MASTER_SELECT_1): The memory device stores the next linked list item on AHB master 1 - 0x1 (LIST_MASTER_SELECT_2): The memory device stores the next linked list item on AHB master 2 - 0x2 (LIST_MASTER_SELECT_3): The memory device stores the next linked list item on AHB master 3 - 0x3 (LIST_MASTER_SELECT_4): The memory device stores the next linked list item on AHB master 4 **Value after reset:** 0x0 **Volatile:** true

Offset: 178h	**CTL4: Control Register for Channel 4**	**Init = 0**
63:45	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0 **Volatile:** true

44	RW	**DONE** Done bit. If status write-back is enabled, the upper word of the control register, CTL4[63:32], is written to the control register location of the Linked List Item (LLI) in system memory at the end of the block transfer with the done bit set. Software can poll the LLI CTL4.DONE bit to see when a block transfer is complete. The LLI CTL4.DONE bit should be cleared when the linked lists are set up in memory prior to enabling the channel. LLI accesses are always 32-bit accesses (Hsize=2) aligned to 32-bit boundaries and cannot be changed or programmed to anything other than 32-bit. For more information, refer to ”Multi-Block Transfers”. **Values:** - 0x0 (DISABLED): DONE bit is deasserted the enb of block transfer - 0x1 (ENABLED): SET the DONE bit at the end of block transfer **Value after reset:** 0x0 **Volatile:** true
43:37	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0 **Volatile:** true
36:32	RW	**BLOCK_TS** Block Transfer Size. When the HDMA is the flow controller, the user writes this field before the channel is enabled in order to indicate the block size. The number programmed into BLOCK_TS indicates the total number of single transactions to perform for every block transfer; a single transaction is mapped to a single AMBA beat **Width:** The width of single transaction is determined by CTL4.SRC_TR_WIDTH. For further information on setting this field, refer to ”Transfer Operation”. Once the transfer starts, the read-back value is the total number of data items already read from the source peripheral, regardless of what is the flow controller. When the source or destination peripheral is assigned as the flow controller, then the maximum block size that can be read back saturates at DMAH_CH4_MAX_BLK_SIZE, but the actual block size can be greater. **Value after reset:** 0x2 **Volatile:** true
31:29	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0 **Volatile:** true
28	RW	**LLP_SRC_EN** Block chaining is enabled on the source side only if the LLP_SRC_EN field is high and LLP4.LOC is non-zero. For more information, see ”Block Chaining using Linked Lists”. **Values:** - 0x0 (LLP_SRC_DISABLE): Block chaining using Linked List is disabled on the Source side - 0x1 (LLP_SRC_ENABLE): Block chaining using Linked List is enabled on the Source side **Value after reset:** 0x0 **Volatile:** true

27	RW	**LLP_DST_EN** Block chaining is enabled on the destination side only if LLP_DST_EN field is high and LLP4.LOC is non-zero. For more information, see ”Block Chaining using Linked Lists” **Values:** - 0x0 (LLP_DST_DISABLE): Block chaining using Linked List is disabled on the Destination side - 0x1 (LLP_DST_ENABLE): Block chaining using Linked List is enabled on the Destination side **Value after reset:** 0x0 **Volatile:** true
26:25	RW	**SMS** Source Master Select. Identifies the Master Interface layer where the source device (peripheral or memory) resides. The maximum value of this field that can be read back is DMAH_NUM_MASTER_INT-1. **Values:** - 0x0 (SMS_0): Source device (peripheral or memory) is accessed from AHB master 1 - 0x1 (SMS_1): Source device (peripheral or memory) is accessed from AHB master 2 - 0x2 (SMS_2): Source device (peripheral or memory) is accessed from AHB master 3 - 0x3 (SMS_3): Source device (peripheral or memory) is accessed from AHB master 4 **Value after reset:** 0x0 **Volatile:** true
24:23	RW	**DMS** Destination Master Select. Identifies the Master Interface layer where the destination device (peripheral or memory) resides. The maximum value of this field that can be read back is DMAH_NUM_MASTER_INT-1. **Values:** - 0x0 (DMS_0): Destination device (peripheral or memory) is accessed from AHB master 1 - 0x1 (DMS_1): Destination device (peripheral or memory) is accessed from AHB master 2 - 0x2 (DMS_2): Destination device (peripheral or memory) is accessed from AHB master 3 - 0x3 (DMS_3): Destination device (peripheral or memory) is accessed from AHB master 4 **Value after reset:** 0x0 **Volatile:** true

22:20	RW	**TT_FC** Transfer Type and Flow Control. Flow control can be assigned to the HDMA, the source peripheral, or the destination peripheral. For more information on transfer types and flow control, refer to ”Setup/Operation of the HDMA Transfers”. Dependencies: If the configuration parameter DMAH_CH4_FC is set to DMA_FC_ONLY, then TT_FC[2] does not exist and TT_FC[2] always reads back 0. If DMAH_CH4_FC is set to SRC_FC_ONLY, then TT_FC[2:1] does not exist and TT_FC[2:1] always reads back 2’b10. If DMAH_CH4_FC is set to DST_FC_ONLY, then TT_FC[2:1] does not exist and TT_FC[2:1] always reads back 2’b11. For multi-block transfers using linked list operation, TT_FC must be constant for all blocks of this multi-block transfer. **Values:** - 0x0 (TT_FC_0): Transfer type is Memory to Memory and Flow Controller is HDMA - 0x1 (TT_FC_1): Transfer type is Memory to Peripheral and Flow Controller is HDMA - 0x2 (TT_FC_2): Transfer type is Peripheral to Memory and Flow Controller is HDMA - 0x3 (TT_FC_3): Transfer type is Peripheral to Peripheral and Flow Controller is HDMA - 0x4 (TT_FC_4): Transfer type is Peripheral to Memory and Flow Controller is Peripheral - 0x5 (TT_FC_5): Transfer type is Peripheral to Peripheral and Flow Controller is Source Peripheral - 0x6 (TT_FC_6): Transfer type is Memory to Peripheral and Flow Controller is Peripheral - 0x7 (TT_FC_7): Transfer type is Peripheral to Peripheral and Flow Controller is Destination Peripheral **Value after reset:** 0x3 **Volatile:** true
19	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0 **Volatile:** true
18	RW	**DST_SCATTER_EN** Destination scatter enable. Scatter on the destination side is applicable only when the CTL4.DINC bit indicates an incrementing or decrementing address control. **Values:** - 0x0 (DST_SCATTER_DISABLE): Destination Scatter is disabled - 0x1 (DST_SCATTER_ENABLE): Destination Scatter is enabled **Value after reset:** 0x0 **Volatile:** true
17	RW	**SRC_GATHER_EN** Destination gather enable. Gather on the destination side is applicable only when the CTL4.SINC bit indicates an incrementing or decrementing address control. **Values:** - 0x0 (SRC_SCATTER_DISABLE): Source gather is disabled - 0x1 (SRC_SCATTER_ENABLE): Source gather is enabled **Value after reset:** 0x0 **Volatile:** true

16:14	RW	**SRC_MSIZE** Source Burst Transaction Length. Number of data items, each of width CTL4.SRC_TR_WIDTH, to be read from the source every time a burst transferred request is made from either the corresponding hardware or software handshaking interface. **NOTE:** This value is not related to the AHB bus master HBURST bus. For information on the decoding for this field, see the ”Setting Up Transfers” section and for more inforamtion about this field, see the ”Choosing the Receive Watermark level” section in the HDMA Databook. **Values:** - 0x0 (SRC_MSIZE_0): Number of data items to be transferred is 1 - 0x1 (SRC_MSIZE_1): Number of data items to be transferred is 4 - 0x2 (SRC_MSIZE_2): Number of data items to be transferred is 8 - 0x3 (SRC_MSIZE_3): Number of data items to be transferred is 16 - 0x4 (SRC_MSIZE_4): Number of data items to be transferred is 32 - 0x5 (SRC_MSIZE_5): Number of data items to be transferred is 64 - 0x6 (SRC_MSIZE_6): Number of data items to be transferred is 128 - 0x7 (SRC_MSIZE_7): Number of data items to be transferred is 256 **Value after reset:** 0x1 **Volatile:** true
13:11	RW	**DEST_MSIZE** Destination Burst Transaction Length. Number of data items, each of width CTL4.DST_TR_WIDTH, to be written to the destination every time a destination burst transaction request is made from either the corresponding hardware or software handshaking interface. **NOTE:** This value is not related to the AHB bus master HBURST bus. **Values:** - 0x0 (DEST_MSIZE_0): Number of data items to be transferred is 1 - 0x1 (DEST_MSIZE_1): Number of data items to be transferred is 4 - 0x2 (DEST_MSIZE_2): Number of data items to be transferred is 8 - 0x3 (DEST_MSIZE_3): Number of data items to be transferred is 16 - 0x4 (DEST_MSIZE_4): Number of data items to be transferred is 32 - 0x5 (DEST_MSIZE_5): Number of data items to be transferred is 64 - 0x6 (DEST_MSIZE_6): Number of data items to be transferred is 128 - 0x7 (DEST_MSIZE_7): Number of data items to be transferred is 256 **Value after reset:** 0x1 **Volatile:** true
10:9	RW	**SINC** Source Address Increment. Indicates whether to increment or decrement the source address on every source transfer. If the device is fetching data from a source peripheral FIFO with a fixed address, then set this field to ”No change”. **Values:** - 0x0 (SINC_0): Increments the source address - 0x1 (SINC_1): Decrements the source address - 0x2 (SINC_2): No change in the source address - 0x3 (SINC_3): No change in the source address **Value after reset:** 0x0 **Volatile:** true

8 :7	RW	**DINC** Distination Address Increment. Indicates whether to increment or decrement the destination address on every destination transfer. If your device is writing data to a destination peripheral FIFO with a fixed address, then set this field to ”No Change”. **Values:** - 0x0 (DINC_0): Increments the distination address - 0x1 (DINC_1): Decrements the distination address - 0x2 (DINC_2): No change in the distination address - 0x3 (DINC_3): No change in the distination address **Value after reset:** 0x0 **Volatile:** true
6 :4	RW	**SRC_TR_WIDTH** Source Transfer Width. Mapped to AHB bus hsize. For a non-memory peripheral, typically the peripheral (source) FIFO width. This value must be less than or equl to DMAH_Mk_HDATA_WIDTH, where k is the AHB layer 1 to 4 where the source resides. **Values:** - 0x0 (SRC_TR_WIDTH_0): Source transfer width is 8 bits - 0x1 (SRC_TR_WIDTH_1): Source transfer width is 16 bits - 0x2 (SRC_TR_WIDTH_2): Source transfer width is 32 bits - 0x3 (SRC_TR_WIDTH_3): Source transfer width is 64 bits - 0x4 (SRC_TR_WIDTH_4): Source transfer width is 128 bits - 0x5 (SRC_TR_WIDTH_5): Source transfer width is 256 bits - 0x6 (SRC_TR_WIDTH_6): Source transfer width is 256 bits - 0x7 (SRC_TR_WIDTH_7): Source transfer width is 256 bits **Value after reset:** 0x2 **Volatile:** true
3 :1	RW	**DST_TR_WIDTH** Destination Transfer Width. Mapped to AHB bus hsize. For a non-memory peripheral, typically the peripheral (destination) FIFO width. This value must be less than or equal to DMAH_Mk_HDATA_WIDTH where k is the AHB layer 1 to 4 where the destination resides. For the decoding of this field, see the ”Setting Up Transfers” section in the HDMA Databook. **Values:** - 0x0 (DST_TR_WIDTH_0): Destination transfer width is 8 bits - 0x1 (DST_TR_WIDTH_1): Destination transfer width is 16 bits - 0x2 (DST_TR_WIDTH_2): Destination transfer width is 32 bits - 0x3 (DST_TR_WIDTH_3): Destination transfer width is 64 bits - 0x4 (DST_TR_WIDTH_4): Destination transfer width is 128 bits - 0x5 (DST_TR_WIDTH_5): Destination transfer width is 256 bits - 0x6 (DST_TR_WIDTH_6): Destination transfer width is 256 bits - 0x7 (DST_TR_WIDTH_7): Destination transfer width is 256 bits **Value after reset:** 0x2 **Volatile:** true

0	RW	**INT_EN** Interrupt Enable Bit. If set, then all interrupt-generating sources are enabled. Functions as a global mask bit for all interrupts for the channel; raw* interrupt registers still assert if CTL2.INT_EN=0. **Values:** - 0x0 (INTERRUPT_DISABLE): Interrupt is disabled - 0x1 (INTERRUPT_ENABLE): Interrupt is enabled **Value after reset:** 0x1 **Volatile:** true

Offset: 180h	**SSTAT4: Source Status Register for Channel 4**	**Init = 0**
63:32	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0 **Volatile:** true
31:0	RW	**SSTAT** Source status information retrieved by hardware from the address pointed to by the contents of the STATAR4 register **Value after reset:** 0x0 **Volatile:** true

Offset: 188h	**DSTAT4: Destination Status Register for Channel 4**	**Init = 0**
63:32	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0 **Volatile:** true
31:0	RW	**DSTAT** Destination status information retrieved by hardware from the address pointed to by the contents of DSTATAR4 register. **Value after reset:** 0x0 **Volatile:** true

Offset: 190h	**SSTATAR4: Source Status Address Register for Channel 4**	**Init = 0**
63:32	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0 **Volatile:** true
31:0	RW	**SSTATAR** Pointer from where hardware can fetch the source status information, which is registered in the SSTAT4 register and written out to the SSTAT4 register location of the LLI before the start of the next block. **Value after reset:** 0x0 **Volatile:** true

Offset: 198h		DSTATAR4: Source Status Address Register for Channel 4	Init = 0
63:32	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0	
31:0	RW	**DSTATAR** Pointer from where hardware can fetch the destination status information, which is registered in the DSTAT4 register and written out to the DSTAT4 register location of the LLI before the start of the next block. **Value after reset:** 0x0 **Volatile:** true	

Offset: 1A0h		CFG4: Configuration Register for Channel 4	Init = 0
63:47	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0 **Volatile:** true	
46:43	RW	**DEST_PER** Destination hardware interface. Assigns a hardware handshaking interface (0 : DMAH_NUM_HS_INT-1) to the destination of channel 4 if the CFG4.HS_SEL_DST field is 0; otherwise, this field is ignored. The channel can then communicate with the destination peripheral connected to that interface through the assigned hardware handshaking interface. **Note1:** For correct HDMA operation, only one peripheral (source or destination) should be assigned to the same handshaking interface. **Note2:** This field does not exist if the configuration parameter DMAH_NUM_HS_INT is set to 0. **Value after reset:** 0x0 **Volatile:** true	
42:39	RW	**SRC_PER** Source Hardware Interface. Assigns a hardware handshaking interface (0 : DMAH_NUM_HS_INT-1) to the source of channel 4 if the CFG4.HS_SEL_SRC field is 0; otherwise, this field is ignored. The channel can then communicate with the source peripheral connected to that interface through the assigned hardware handshaking interface. **Note1:** For correct HDMA operation, only one peripheral (source or destination) should be assigned to the same handshaking interface. **Note2:** This field does not exist if the configuration parameter DMAH_NUM_HS_INT is set to 0. **Value after reset:** 0x0 **Volatile:** true	

38	RW	**SS_UPD_EN** Source Status Update Enable. Source status information is fetched only from the location pointed to by the SSTATAR4 register, stored in the SSTAT4 register and written out to the SSTAT4 location of the LLI, if SS_UPD_EN is high. **Note:** This enalbe is applicable only if DMAH_CH4_STAT_SRC is set to True. This field does not exist if the configuration parameter DMAH_CH4_STAT_SRC is set to False; in this case, the read-back value is always 0. **Values:** - 0x0 (DISABLED): Source Status Update is disabled. - 0x1 (ENABLED): Source Status Update is enabled. **Value after reset:** 0x0 **Volatile:** true
37	RW	**DS_UPD_EN** Destination Status Update Enable. Destination status information is fetched only from the location pointed to by the DSTATAR0 register, stored in the DSTAT4 register and written out to the DSTAT4 location of the LLI, if DS_UPD_EN is high. This field does not exist if the configuration parameter DMAH_CH1_SATAT_DST is set to False; in this case, the read-back value is always 0. **Values:** - 0x0 (DISABLED): Destination Status Update is disabled. - 0x1 (ENABLED): Destination Status Update is enabled. **Value after reset:** 0x0 **Volatile:** true
36:34	RW	**PROTCTL** Protection Control bits used to drive the AHB HPROT[3:1] bus. The AMBA Specification recommends that the default of HPROT indicates a non-cached, non-buffered, privileged data access. The reset value is used to indicate such an access. HPROT[0] is tied high because all transfers are data accesses, as there are no opcode fetches. There is a one-to-one mapping of these register bits to the HPROT[3:1] master interface signals. Mapping of HPROT bus is as follows: - 1'b1 to HPROT[0] - CFG1.PROTCTL[1] to HPROT[1] - CFG1.PROTCTL[2] to HPROT[2] - CFG1.PROTCTL[3] to HPROT[3] **Value after reset:** 0x1 **Volatile:** true
33	RW	**FIFO_MODE** FIFO Mode Select. Determines how much space or data needs to be available in the FIFO before a burst transaction request is serviced. **Values:** - 0x0 (FIFO_MODE_0): Space/data available for single AHB transfer of the specified transfer width - 0x1 (FIFO_MODE_1): Data available is greater than or equal to half the FIFO depth for destination transfers and space available is greater than half the fifo depth for source transfers. The exceptions are at the end of a burst transaction request or at the end of a block transfer. **Value after reset:** 0x0 **Volatile:** true

32	RW	**FCMODE** Flow Control Mode. Determines when source transaction requests are serviced when the Destination Peripheral is the flow controller. **Values:** - 0x0 (FCMODE_0): Source transaction requests are serviced when they occur. Data pre-fetching is enabled - 0x1 (FCMODE_1): Source transaction requests are not serviced until a destination transaction request occurs. In this mode, the amount of data transferred from the source is limited so that it is guaranteed to be transferred to the destination prior to block termination by the destination. Data pre-fetching is disabled. **Value after reset:** 0x0 **Volatile:** true
31	RW	**RELOAD_DST** Automatic Destination Reload. The DAR2 register can be automatically reloaded from its initial value at the end of every block for multi-block transfers. A new block transfer is then initiated. This register does not exist if the configuration parameter DMAH_CH4_MULTI_BLK_EN is not selected; in this case, the read-back value is always 0. **Values:** - 0x0 (DISABLE): Destination Reload Disabled. - 0x1 (ENABLE): Destination Reload Enabled. **Value after reset:** 0x0 **Volatile:** true
30	RW	**RELOAD_SRC** Automatic Source Reload. The SAR4 register can be automatically reloaded from its initial value at the end of every block for multi-block transfers. A new block transfer is then initiated. This field does not exist if the configuration parameter DMAH_CH4_MULTI_BLK_EN is not selected; in this case, the read-back value is always 0. **Values:** - 0x0 (DISABLE): Source Reload Disabled. - 0x1 (ENABLE): Source Reload Enabled. **Value after reset:** 0x0 **Volatile:** true
29:20	R	**MAX_ABRST** Maximum AMBA Burst Length. Maximum AMBA burst length that is used for DMA transfers on this channel. A value of 0 indicates that software is not limiting the maximum AMBA burst length for DMA transfers on this channel. This field does not exist if the configuration parameter DMAH_MABRST is not selected; in this case, the read-back value is always 0, and the maximum AMBA burst length cannot be limited by software. **Value after reset:** 0x0 **Volatile:** true

19	RW	**SRC_HS_POL** Source Handshaking Interface Polarity. **Values:** - 0x0 (ACTIVE_HIGH): Source Handshaking Interface Polarity is Active high - 0x1 (ACTIVE_LOW): Source Handshaking Interface Polarity is Active low **Value after reset:** 0x0 **Volatile:** true
18	RW	**DST_HS_POL** Destination Handshaking Interface Polarity. **Values:** - 0x0 (ACTIVE_HIGH): Destination Handshaking Interface Polarity is Active high - 0x1 (ACTIVE_LOW): Destination Handshaking Interface Polarity is Active low **Value after reset:** 0x0 **Volatile:** true
17	RW	**LOCK_B** Bus Lock Bit. When active, the AHB bus master signal hlock is asserted for the duration specified in CFG0.LOCK_B_L. For more information, refer to ”Locked DMA Transfers”. This field does not exist if the configuration parameter DMAH_CH4_LOCK_EN is set to False; in this case, the read-back value is always 0. **Values:** - 0x0 (DISABLED): Bus lock bit is not enabled - 0x1 (ENABLED): Bus lock bit is enabled **Value after reset:** 0x0 **Volatile:** true
16	RW	**LOCK_CH** Channel Lock Bit. When the channel is granted control of the master bus interface and if the CFG4.LOCK_CH bit is asserted, then no other channels are granted control of the master bus interface for the duration specified in CFG4.LOCK_CH_L. Indicates to the master bus interface arbiter that this channel wants exclusive access to the master bus interface for the duration specified in CFG3.LOCK_CH_L. This field does not exist if the configuration parameter DMAH_CH4_LOCK_EN is set to False; in this case, the read-back value is always 0. **Values:** - 0x0 (DISABLED): Channel lock bit is not enabled - 0x1 (ENABLED): Channel lock bit is enabled **Value after reset:** 0x0 **Volatile:** true
14:15	RW	**LOCK_B_L** Bus lock level. Indicates the duration over which CFG4.LOCK_B bit applies. This field does not exist if the parameter DMAH_CH4_LOCK_EN is set to False; in this case, the read-back value is always 0. **Values:** - 0x0 (LOCK_B_L_0): Over complete DMA transfer - 0x1 (LOCK_B_L_1): Over complete DMA block transfer - 0x2 (LOCK_B_L_2): Over complete DMA transaction - 0x3 (LOCK_B_L_3): Over complete DMA transaction **Value after reset:** 0x0 **Volatile:** true

13:12	RW	**LOCK_CH_L** Channel Local Level. Indicates the duration over which CFG4.LOCK_CH applies. This field does not exist if the configuration parameter DMAH_CH4_LOCK_EN is set to False; in this case, the read-back value is always 0. **Values:** - 0x0 (LOCK_CH_L_0): Over complete DMA transfer - 0x1 (LOCK_CH_L_1): Over complete DMA block transfer - 0x2 (LOCK_CH_L_2): Over complete DMA transaction - 0x3 (LOCK_CH_L_3): Over complete DMA transaction **Value after reset:** 0x0 **Volatile:** true
11	RW	**HS_SEL_SRC** Source Software or Hardware Handshaking Select. This register selects which of the handshaking interfaces - hardware or software - is active for source requests on this channel. If the source peripheral is memory, then this bit is ignored. **Values:** - 0x0 (HARDWARE_HS): Hardware handshaking interface. Software initiated transaction requests are ignored. - 0x1 (SOFTWARE_HS): Software handshaking interface. Hardware initiated transaction requests are ignored. **Value after reset:** 0x1 **Volatile:** true
10	RW	**HS_SEL_DST** Destination Software or Hardware Handshaking Select. This register selects which of the handshaking interfaces - hardware or software - is active for destination requests on this channel. If the destination peripheral is memory, then this bit is ignored. **Values:** - 0x0 (HARDWARE_HS): Hardware handshaking interface. Software initiated transaction requests are ignored. - 0x1 (SOFTWARE_HS): Software handshaking interface. Hardware initiated transaction requests are ignored. **Value after reset:** 0x1 **Volatile:** true
9	R	**FIFO_EMPTY** Channel FIFO status. Indicates if there is data left in the channel FIFO. Can be used in conjunction with CFG4.CH_SUSP to cleanly disable a channel. For more information, refer to ”Disabling a Channel Prior to Transfer Completion”. **Values:** - 0x0 (NOT_EMPTY): Channel FIFO is not empty - 0x1 (EMPTY): Channel FIFO is empty **Value after reset:** 0x1 **Volatile:** true

8	R	**CH_SUSP** Channel Suspend. Suspends all DMA data transfers from the source until this bit is cleared. There is no guarantee that the current transaction will complete. Can also be used in conjunction with CFG4.FIFO_EMPTY to cleanly disable a channel without losing any data. For more information, refer to ”Disabling a Channel Prior to Transfer Completion”. **Values:** - 0x0 (NOT_SUSPENDED): DMA transfer from the source is not suspended - 0x1 (SUSPENDED): Suspend DMA transfer from the source **Value after reset:** 0x0 **Volatile:** true
7 :5	RW	**SRC_TR_WIDTH** Source Transfer Width. Mapped to AHB bus hsize. For a non-memory peripheral, typically the peripheral (source) FIFO width. This value must be less than or equl to DMAH_Mk_HDATA_WIDTH, where k is the AHB layer 1 to 4 where the source resides. **Values:** - 0x0 (SRC_TR_WIDTH_0): Source transfer width is 8 bits - 0x1 (SRC_TR_WIDTH_1): Source transfer width is 16 bits - 0x2 (SRC_TR_WIDTH_2): Source transfer width is 32 bits - 0x3 (SRC_TR_WIDTH_3): Source transfer width is 64 bits - 0x4 (SRC_TR_WIDTH_4): Source transfer width is 128 bits - 0x5 (SRC_TR_WIDTH_5): Source transfer width is 256 bits - 0x6 (SRC_TR_WIDTH_6): Source transfer width is 256 bits - 0x7 (SRC_TR_WIDTH_7): Source transfer width is 256 bits **Value after reset:** x **Volatile:** true
4 :0	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0

Offset: 1A8h	**SGR4: Source Gather Register for Channel 4**	**Init = 0**
63:32	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0
31:25	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0
24:19	RW	**SGC** Source Gather Count. Source contiguous tranfer count between successive gather boundaries. **Value after reset:** 0x0 **Volatile:** true

19:0	RW	**SGI** Source Gather Interval. **Value after reset:** 0x0 **Volatile:** true

Offset: 1B0h	**DSR4: Source Gather Register for Channel 4**	**Init = 0**
63:32	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0
31:25	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0
24:19	RW	**DSC** Destination Scatter Count. Destination contiguous transfer count between successive scatter boundaries. **Value after reset:** 0x0 **Volatile:** true
19:0	RW	**DSI** Destination Scatter Interval. **Value after reset:** 0x0 **Volatile:** true

Offset: 1B8h	**SAR5: Source Address for Channel 5**	**Init = 0**
63:32	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0 **Volatile:** true
31:0	RW	**SAR** Current Source Address of DMA transfer. Updated after each source transfer. The SINC field in the CTL5 register determines whether the address increments, decrements, or is left unchanged on every source transfer through the block transfer. **Value after reset:** 0x0 **Volatile:** true

Offset: 1C0h	**DAR5: Destination Address for Channel 5**	**Init = 0**
63:32	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0 **Volatile:** true

31:0	RW	**DAR** Current Destination address of DMA transfer. Updated after each destination transfer. The DINC field in the CTL5 register determines whether the address increments, decrements, or is left unchanged on every destination transfer throughout the block transfer. **Value after reset:** 0x0 **Volatile:** true

Offset: 1C8h	**LLP5: Linked List Pointer Register for Channel 5**	**Init = 0**
63:32	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0 **Volatile:** true
31:2	RW	**LOC** Starting Address In Memory of next LLI if block chaining is enabled. Note that the two LSBs of the starting address are not stored because the address is assumed to be aligned to a 32-bit boundary. LLI accesses are always 32-bit accesses (Hsize=2) aligned to 32-bit boundaries and cannot be changed or programmed to anything other than 32-bit. **Value after reset:** 0x0 **Volatile:** true
1 :0	RW	**LMS** List Master Select. Identifies the AHB layer/interface where the memory device that stores the next linked list item resides. This field does not exist if the configuration parameter is not set to NO_HARDCODE. In this case, the read-back value is always the hardcoded value. The maximum value of this field that can be read back is DMAH_NUM_MASTER_INT-1. **Values:** - 0x0 (LIST_MASTER_SELECT_1): The memory device stores the next linked list item on AHB master 1 - 0x1 (LIST_MASTER_SELECT_2): The memory device stores the next linked list item on AHB master 2 - 0x2 (LIST_MASTER_SELECT_3): The memory device stores the next linked list item on AHB master 3 - 0x3 (LIST_MASTER_SELECT_4): The memory device stores the next linked list item on AHB master 4 **Value after reset:** 0x0 **Volatile:** true

Offset: 1D0h	**CTL5: Control Register for Channel 5**	**Init = 0**
63:45	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0 **Volatile:** true

44	RW	**DONE** Done bit. If status write-back is enabled, the upper word of the control register, CTL5[63:32], is written to the control register location of the Linked List Item (LLI) in system memory at the end of the block transfer with the done bit set. Software can poll the LLI CTL5.DONE bit to see when a block transfer is complete. The LLI CTL5.DONE bit should be cleared when the linked lists are set up in memory prior to enabling the channel. LLI accesses are always 32-bit accesses (Hsize=2) aligned to 32-bit boundaries and cannot be changed or programmed to anything other than 32-bit. For more information, refer to ”Multi-Block Transfers”. **Values:** - 0x0 (DISABLED): DONE bit is deasserted the enb of block transfer - 0x1 (ENABLED): SET the DONE bit at the end of block transfer **Value after reset:** 0x0 **Volatile:** true
43:37	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0 **Volatile:** true
36:32	RW	**BLOCK_TS** Block Transfer Size. When the HDMA is the flow controller, the user writes this field before the channel is enabled in order to indicate the block size. The number programmed into BLOCK_TS indicates the total number of single transactions to perform for every block transfer; a single transaction is mapped to a single AMBA beat **Width:** The width of single transaction is determined by CTL5.SRC_TR_WIDTH. For further information on setting this field, refer to ”Transfer Operation”. Once the transfer starts, the read-back value is the total number of data items already read from the source peripheral, regardless of what is the flow controller. When the source or destination peripheral is assigned as the flow controller, then the maximum block size that can be read back saturates at DMAH_CH5_MAX_BLK_SIZE, but the actual block size can be greater. **Value after reset:** 0x2 **Volatile:** true
31:29	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0 **Volatile:** true
28	RW	**LLP_SRC_EN** Block chaining is enabled on the source side only if the LLP_SRC_EN field is high and LLP5.LOC is non-zero. For more information, see ”Block Chaining using Linked Lists”. **Values:** - 0x0 (LLP_SRC_DISABLE): Block chaining using Linked List is disabled on the Source side - 0x1 (LLP_SRC_ENABLE): Block chaining using Linked List is enabled on the Source side **Value after reset:** 0x0 **Volatile:** true

27	RW	**LLP_DST_EN** Block chaining is enabled on the destination side only if LLP_DST_EN field is high and LLP5.LOC is non-zero. For more information, see ”Block Chaining using Linked Lists” **Values:** - 0x0 (LLP_DST_DISABLE): Block chaining using Linked List is disabled on the Destination side - 0x1 (LLP_DST_ENABLE): Block chaining using Linked List is enabled on the Destination side **Value after reset:** 0x0 **Volatile:** true
26:25	RW	**SMS** Source Master Select. Identifies the Master Interface layer where the source device (peripheral or memory) resides. The maximum value of this field that can be read back is DMAH_NUM_MASTER_INT-1. **Values:** - 0x0 (SMS_0): Source device (peripheral or memory) is accessed from AHB master 1 - 0x1 (SMS_1): Source device (peripheral or memory) is accessed from AHB master 2 - 0x2 (SMS_2): Source device (peripheral or memory) is accessed from AHB master 3 - 0x3 (SMS_3): Source device (peripheral or memory) is accessed from AHB master 4 **Value after reset:** 0x0 **Volatile:** true
24:23	RW	**DMS** Destination Master Select. Identifies the Master Interface layer where the destination device (peripheral or memory) resides. The maximum value of this field that can be read back is DMAH_NUM_MASTER_INT-1. **Values:** - 0x0 (DMS_0): Destination device (peripheral or memory) is accessed from AHB master 1 - 0x1 (DMS_1): Destination device (peripheral or memory) is accessed from AHB master 2 - 0x2 (DMS_2): Destination device (peripheral or memory) is accessed from AHB master 3 - 0x3 (DMS_3): Destination device (peripheral or memory) is accessed from AHB master 4 **Value after reset:** 0x0 **Volatile:** true

22:20	RW	**TT_FC** Transfer Type and Flow Control. Flow control can be assigned to the HDMA, the source peripheral, or the destination peripheral. For more information on transfer types and flow control, refer to ”Setup/Operation of the HDMA Transfers”. Dependencies: If the configuration parameter DMAH_CH5_FC is set to DMA_FC_ONLY, then TT_FC[2] does not exist and TT_FC[2] always reads back 0. If DMAH_CH5_FC is set to SRC_FC_ONLY, then TT_FC[2:1] does not exist and TT_FC[2:1] always reads back 2’b10. If DMAH_CH5_FC is set to DST_FC_ONLY, then TT_FC[2:1] does not exist and TT_FC[2:1] always reads back 2’b11. For multi-block transfers using linked list operation, TT_FC must be constant for all blocks of this multi-block transfer. **Values:** - 0x0 (TT_FC_0): Transfer type is Memory to Memory and Flow Controller is HDMA - 0x1 (TT_FC_1): Transfer type is Memory to Peripheral and Flow Controller is HDMA - 0x2 (TT_FC_2): Transfer type is Peripheral to Memory and Flow Controller is HDMA - 0x3 (TT_FC_3): Transfer type is Peripheral to Peripheral and Flow Controller is HDMA - 0x4 (TT_FC_4): Transfer type is Peripheral to Memory and Flow Controller is Peripheral - 0x5 (TT_FC_5): Transfer type is Peripheral to Peripheral and Flow Controller is Source Peripheral - 0x6 (TT_FC_6): Transfer type is Memory to Peripheral and Flow Controller is Peripheral - 0x7 (TT_FC_7): Transfer type is Peripheral to Peripheral and Flow Controller is Destination Peripheral **Value after reset:** 0x3 **Volatile:** true
19	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0 **Volatile:** true
18	RW	**DST_SCATTER_EN** Destination scatter enable. Scatter on the destination side is applicable only when the CTL5.DINC bit indicates an incrementing or decrementing address control. **Values:** - 0x0 (DST_SCATTER_DISABLE): Destination Scatter is disabled - 0x1 (DST_SCATTER_ENABLE): Destination Scatter is enabled **Value after reset:** 0x0 **Volatile:** true
17	RW	**SRC_GATHER_EN** Destination gather enable. Gather on the destination side is applicable only when the CTL5.SINC bit indicates an incrementing or decrementing address control. **Values:** - 0x0 (SRC_SCATTER_DISABLE): Source gather is disabled - 0x1 (SRC_SCATTER_ENABLE): Source gather is enabled **Value after reset:** 0x0 **Volatile:** true

16:14	RW	**SRC_MSIZE** Source Burst Transaction Length. Number of data items, each of width CTL5.SRC_TR_WIDTH, to be read from the source every time a burst transferred request is made from either the corresponding hardware or software handshaking interface. **NOTE:** This value is not related to the AHB bus master HBURST bus. For information on the decoding for this field, see the ”Setting Up Transfers” section and for more inforamtion about this field, see the ”Choosing the Receive Watermark level” section in the HDMA Databook. **Values:** - 0x0 (SRC_MSIZE_0): Number of data items to be transferred is 1 - 0x1 (SRC_MSIZE_1): Number of data items to be transferred is 4 - 0x2 (SRC_MSIZE_2): Number of data items to be transferred is 8 - 0x3 (SRC_MSIZE_3): Number of data items to be transferred is 16 - 0x4 (SRC_MSIZE_4): Number of data items to be transferred is 32 - 0x5 (SRC_MSIZE_5): Number of data items to be transferred is 64 - 0x6 (SRC_MSIZE_6): Number of data items to be transferred is 128 - 0x7 (SRC_MSIZE_7): Number of data items to be transferred is 256 **Value after reset:** 0x1 **Volatile:** true
13:11	RW	**DEST_MSIZE** Destination Burst Transaction Length. Number of data items, each of width CTL5.DST_TR_WIDTH, to be written to the destination every time a destination burst transaction request is made from either the corresponding hardware or software handshaking interface. **NOTE:** This value is not related to the AHB bus master HBURST bus. **Values:** - 0x0 (DEST_MSIZE_0): Number of data items to be transferred is 1 - 0x1 (DEST_MSIZE_1): Number of data items to be transferred is 4 - 0x2 (DEST_MSIZE_2): Number of data items to be transferred is 8 - 0x3 (DEST_MSIZE_3): Number of data items to be transferred is 16 - 0x4 (DEST_MSIZE_4): Number of data items to be transferred is 32 - 0x5 (DEST_MSIZE_5): Number of data items to be transferred is 64 - 0x6 (DEST_MSIZE_6): Number of data items to be transferred is 128 - 0x7 (DEST_MSIZE_7): Number of data items to be transferred is 256 **Value after reset:** 0x1 **Volatile:** true
10:9	RW	**SINC** Source Address Increment. Indicates whether to increment or decrement the source address on every source transfer. If the device is fetching data from a source peripheral FIFO with a fixed address, then set this field to ”No change”. **Values:** - 0x0 (SINC_0): Increments the source address - 0x1 (SINC_1): Decrements the source address - 0x2 (SINC_2): No change in the source address - 0x3 (SINC_3): No change in the source address **Value after reset:** 0x0 **Volatile:** true

8 :7	RW	**DINC** Distination Address Increment. Indicates whether to increment or decrement the destination address on every destination transfer. If your device is writing data to a destination peripheral FIFO with a fixed address, then set this field to ”No Change”. **Values:** - 0x0 (DINC_0): Increments the distination address - 0x1 (DINC_1): Decrements the distination address - 0x2 (DINC_2): No change in the distination address - 0x3 (DINC_3): No change in the distination address **Value after reset:** 0x0 **Volatile:** true
6 :4	RW	**SRC_TR_WIDTH** Source Transfer Width. Mapped to AHB bus hsize. For a non-memory peripheral, typically the peripheral (source) FIFO width. This value must be less than or equl to DMAH_Mk_HDATA_WIDTH, where k is the AHB layer 1 to 4 where the source resides. **Values:** - 0x0 (SRC_TR_WIDTH_0): Source transfer width is 8 bits - 0x1 (SRC_TR_WIDTH_1): Source transfer width is 16 bits - 0x2 (SRC_TR_WIDTH_2): Source transfer width is 32 bits - 0x3 (SRC_TR_WIDTH_3): Source transfer width is 64 bits - 0x4 (SRC_TR_WIDTH_4): Source transfer width is 128 bits - 0x5 (SRC_TR_WIDTH_5): Source transfer width is 256 bits - 0x6 (SRC_TR_WIDTH_6): Source transfer width is 256 bits - 0x7 (SRC_TR_WIDTH_7): Source transfer width is 256 bits **Value after reset:** 0x2 **Volatile:** true
3 :1	RW	**DST_TR_WIDTH** Destination Transfer Width. Mapped to AHB bus hsize. For a non-memory peripheral, typically the peripheral (destination) FIFO width. This value must be less than or equal to DMAH_Mk_HDATA_WIDTH where k is the AHB layer 1 to 4 where the destination resides. For the decoding of this field, see the ”Setting Up Transfers” section in the HDMA Databook. **Values:** - 0x0 (DST_TR_WIDTH_0): Destination transfer width is 8 bits - 0x1 (DST_TR_WIDTH_1): Destination transfer width is 16 bits - 0x2 (DST_TR_WIDTH_2): Destination transfer width is 32 bits - 0x3 (DST_TR_WIDTH_3): Destination transfer width is 64 bits - 0x4 (DST_TR_WIDTH_4): Destination transfer width is 128 bits - 0x5 (DST_TR_WIDTH_5): Destination transfer width is 256 bits - 0x6 (DST_TR_WIDTH_6): Destination transfer width is 256 bits - 0x7 (DST_TR_WIDTH_7): Destination transfer width is 256 bits **Value after reset:** 0x2 **Volatile:** true

0	RW	**INT_EN** Interrupt Enable Bit. If set, then all interrupt-generating sources are enabled. Functions as a global mask bit for all interrupts for the channel; raw* interrupt registers still assert if CTL2.INT_EN=0. **Values:** - 0x0 (INTERRUPT_DISABLE): Interrupt is disabled - 0x1 (INTERRUPT_ENABLE): Interrupt is enabled **Value after reset:** 0x1 **Volatile:** true

Offset: 1D8h	**SSTAT5: Source Status Register for Channel 5**	**Init = 0**
63:32	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0 **Volatile:** true
31:0	RW	**SSTAT** Source status information retrieved by hardware from the address pointed to by the contents of the STATAR5 register **Value after reset:** 0x0 **Volatile:** true

Offset: 1E0h	**DSTAT5: Destination Status Register for Channel 5**	**Init = 0**
63:32	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0 **Volatile:** true
31:0	RW	**DSTAT** Destination status information retrieved by hardware from the address pointed to by the contents of DSTATAR5 register. **Value after reset:** 0x0 **Volatile:** true

Offset: 1E8h	**SSTATAR5: Source Status Address Register for Channel 5**	**Init = 0**
63:32	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0 **Volatile:** true
31:0	RW	**SSTATAR** Pointer from where hardware can fetch the source status information, which is registered in the SSTAT5 register and written out to the SSTAT5 register location of the LLI before the start of the next block. **Value after reset:** 0x0 **Volatile:** true

Offset: 1F0h		DSTATAR5: Source Status Address Register for Channel 5	Init = 0
63:32	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0	
31:0	RW	**DSTATAR** Pointer from where hardware can fetch the destination status information, which is registered in the DSTAT5 register and written out to the DSTAT4 register location of the LLI before the start of the next block. **Value after reset:** 0x0 **Volatile:** true	

Offset: 1F8h		CFG5: Configuration Register for Channel 5	Init = 0
63:47	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0 **Volatile:** true	
46:43	RW	**DEST_PER** Destination hardware interface. Assigns a hardware handshaking interface (0 : DMAH_NUM_HS_INT-1) to the destination of channel 5 if the CFG5.HS_SEL_DST field is 0; otherwise, this field is ignored. The channel can then communicate with the destination peripheral connected to that interface through the assigned hardware handshaking interface. **Note1:** For correct HDMA operation, only one peripheral (source or destination) should be assigned to the same handshaking interface. **Note2:** This field does not exist if the configuration parameter DMAH_NUM_HS_INT is set to 0. **Value after reset:** 0x0 **Volatile:** true	
42:39	RW	**SRC_PER** Source Hardware Interface. Assigns a hardware handshaking interface (0 : DMAH_NUM_HS_INT-1) to the source of channel 5 if the CFG5.HS_SEL_SRC field is 0; otherwise, this field is ignored. The channel can then communicate with the source peripheral connected to that interface through the assigned hardware handshaking interface. **Note1:** For correct HDMA operation, only one peripheral (source or destination) should be assigned to the same handshaking interface. **Note2:** This field does not exist if the configuration parameter DMAH_NUM_HS_INT is set to 0. **Value after reset:** 0x0 **Volatile:** true	

38	RW	**SS_UPD_EN** Source Status Update Enable. Source status information is fetched only from the location pointed to by the SSTATAR5 register, stored in the SSTAT5 register and written out to the SSTAT5 location of the LLI, if SS_UPD_EN is high. **Note:** This enalbe is applicable only if DMAH_CH5_STAT_SRC is set to True. This field does not exist if the configuration parameter DMAH_CH5_STAT_SRC is set to False; in this case, the read-back value is always 0. **Values:** - 0x0 (DISABLED): Source Status Update is disabled. - 0x1 (ENABLED): Source Status Update is enabled. **Value after reset:** 0x0 **Volatile:** true
37	RW	**DS_UPD_EN** Destination Status Update Enable. Destination status information is fetched only from the location pointed to by the DSTATAR0 register, stored in the DSTAT5 register and written out to the DSTAT5 location of the LLI, if DS_UPD_EN is high. This field does not exist if the configuration parameter DMAH_CH1_SATAT_DST is set to False; in this case, the read-back value is always 0. **Values:** - 0x0 (DISABLED): Destination Status Update is disabled. - 0x1 (ENABLED): Destination Status Update is enabled. **Value after reset:** 0x0 **Volatile:** true
36:34	RW	**PROTCTL** Protection Control bits used to drive the AHB HPROT[3:1] bus. The AMBA Specification recommends that the default of HPROT indicates a non-cached, non-buffered, privileged data access. The reset value is used to indicate such an access. HPROT[0] is tied high because all transfers are data accesses, as there are no opcode fetches. There is a one-to-one mapping of these register bits to the HPROT[3:1] master interface signals. Mapping of HPROT bus is as follows: - 1'b1 to HPROT[0] - CFG1.PROTCTL[1] to HPROT[1] - CFG1.PROTCTL[2] to HPROT[2] - CFG1.PROTCTL[3] to HPROT[3] **Value after reset:** 0x1 **Volatile:** true
33	RW	**FIFO_MODE** FIFO Mode Select. Determines how much space or data needs to be available in the FIFO before a burst transaction request is serviced. **Values:** - 0x0 (FIFO_MODE_0): Space/data available for single AHB transfer of the specified transfer width - 0x1 (FIFO_MODE_1): Data available is greater than or equal to half the FIFO depth for destination transfers and space available is greater than half the fifo depth for source transfers. The exceptions are at the end of a burst transaction request or at the end of a block transfer. **Value after reset:** 0x0 **Volatile:** true

32	RW	**FCMODE** Flow Control Mode. Determines when source transaction requests are serviced when the Destination Peripheral is the flow controller. **Values:** - 0x0 (FCMODE_0): Source transaction requests are serviced when they occur. Data pre-fetching is enabled - 0x1 (FCMODE_1): Source transaction requests are not serviced until a destination transaction request occurs. In this mode, the amount of data transferred from the source is limited so that it is guaranteed to be transferred to the destination prior to block termination by the destination. Data pre-fetching is disabled. **Value after reset:** 0x0 **Volatile:** true
31	RW	**RELOAD_DST** Automatic Destination Reload. The DAR2 register can be automatically reloaded from its initial value at the end of every block for multi-block transfers. A new block transfer is then initiated. This register does not exist if the configuration parameter DMAH_CH5_MULTI_BLK_EN is not selected; in this case, the read-back value is always 0. **Values:** - 0x0 (DISABLE): Destination Reload Disabled. - 0x1 (ENABLE): Destination Reload Enabled. **Value after reset:** 0x0 **Volatile:** true
30	RW	**RELOAD_SRC** Automatic Source Reload. The SAR5 register can be automatically reloaded from its initial value at the end of every block for multi-block transfers. A new block transfer is then initiated. This field does not exist if the configuration parameter DMAH_CH5_MULTI_BLK_EN is not selected; in this case, the read-back value is always 0. **Values:** - 0x0 (DISABLE): Source Reload Disabled. - 0x1 (ENABLE): Source Reload Enabled. **Value after reset:** 0x0 **Volatile:** true
29:20	R	**MAX_ABRST** Maximum AMBA Burst Length. Maximum AMBA burst length that is used for DMA transfers on this channel. A value of 0 indicates that software is not limiting the maximum AMBA burst length for DMA transfers on this channel. This field does not exist if the configuration parameter DMAH_MABRST is not selected; in this case, the read-back value is always 0, and the maximum AMBA burst length cannot be limited by software. **Value after reset:** 0x0 **Volatile:** true

19	RW	**SRC_HS_POL** Source Handshaking Interface Polarity. **Values:** - 0x0 (ACTIVE_HIGH): Source Handshaking Interface Polarity is Active high - 0x1 (ACTIVE_LOW): Source Handshaking Interface Polarity is Active low **Value after reset:** 0x0 **Volatile:** true
18	RW	**DST_HS_POL** Destination Handshaking Interface Polarity. **Values:** - 0x0 (ACTIVE_HIGH): Destination Handshaking Interface Polarity is Active high - 0x1 (ACTIVE_LOW): Destination Handshaking Interface Polarity is Active low **Value after reset:** 0x0 **Volatile:** true
17	RW	**LOCK_B** Bus Lock Bit. When active, the AHB bus master signal hlock is asserted for the duration specified in CFG0.LOCK_B_L. For more information, refer to ”Locked DMA Transfers”. This field does not exist if the configuration parameter DMAH_CH5_LOCK_EN is set to False; in this case, the read-back value is always 0. **Values:** - 0x0 (DISABLED): Bus lock bit is not enabled - 0x1 (ENABLED): Bus lock bit is enabled **Value after reset:** 0x0 **Volatile:** true
16	RW	**LOCK_CH** Channel Lock Bit. When the channel is granted control of the master bus interface and if the CFG5.LOCK_CH bit is asserted, then no other channels are granted control of the master bus interface for the duration specified in CFG5.LOCK_CH_L. Indicates to the master bus interface arbiter that this channel wants exclusive access to the master bus interface for the duration specified in CFG3.LOCK_CH_L. This field does not exist if the configuration parameter DMAH_CH5_LOCK_EN is set to False; in this case, the read-back value is always 0. **Values:** - 0x0 (DISABLED): Channel lock bit is not enabled - 0x1 (ENABLED): Channel lock bit is enabled **Value after reset:** 0x0 **Volatile:** true
14:15	RW	**LOCK_B_L** Bus lock level. Indicates the duration over which CFG5.LOCK_B bit applies. This field does not exist if the parameter DMAH_CH5_LOCK_EN is set to False; in this case, the read-back value is always 0. **Values:** - 0x0 (LOCK_B_L_0): Over complete DMA transfer - 0x1 (LOCK_B_L_1): Over complete DMA block transfer - 0x2 (LOCK_B_L_2): Over complete DMA transaction - 0x3 (LOCK_B_L_3): Over complete DMA transaction **Value after reset:** 0x0 **Volatile:** true

13:12	RW	**LOCK_CH_L** Channel Local Level. Indicates the duration over which CFG5.LOCK_CH applies. This field does not exist if the configuration parameter DMAH_CH5_LOCK_EN is set to False; in this case, the read-back value is always 0. **Values:** - 0x0 (LOCK_CH_L_0): Over complete DMA transfer - 0x1 (LOCK_CH_L_1): Over complete DMA block transfer - 0x2 (LOCK_CH_L_2): Over complete DMA transaction - 0x3 (LOCK_CH_L_3): Over complete DMA transaction **Value after reset:** 0x0 **Volatile:** true
11	RW	**HS_SEL_SRC** Source Software or Hardware Handshaking Select. This register selects which of the handshaking interfaces - hardware or software - is active for source requests on this channel. If the source peripheral is memory, then this bit is ignored. **Values:** - 0x0 (HARDWARE_HS): Hardware handshaking interface. Software initiated transaction requests are ignored. - 0x1 (SOFTWARE_HS): Software handshaking interface. Hardware initiated transaction requests are ignored. **Value after reset:** 0x1 **Volatile:** true
10	RW	**HS_SEL_DST** Destination Software or Hardware Handshaking Select. This register selects which of the handshaking interfaces - hardware or software - is active for destination requests on this channel. If the destination peripheral is memory, then this bit is ignored. **Values:** - 0x0 (HARDWARE_HS): Hardware handshaking interface. Software initiated transaction requests are ignored. - 0x1 (SOFTWARE_HS): Software handshaking interface. Hardware initiated transaction requests are ignored. **Value after reset:** 0x1 **Volatile:** true
9	R	**FIFO_EMPTY** Channel FIFO status. Indicates if there is data left in the channel FIFO. Can be used in conjunction with CFG5.CH_SUSP to cleanly disable a channel. For more information, refer to ”Disabling a Channel Prior to Transfer Completion”. **Values:** - 0x0 (NOT_EMPTY): Channel FIFO is not empty - 0x1 (EMPTY): Channel FIFO is empty **Value after reset:** 0x1 **Volatile:** true

8	R	**CH_SUSP** Channel Suspend. Suspends all DMA data transfers from the source until this bit is cleared. There is no guarantee that the current transaction will complete. Can also be used in conjunction with CFG5.FIFO_EMPTY to cleanly disable a channel without losing any data. For more information, refer to ”Disabling a Channel Prior to Transfer Completion”. **Values:** - 0x0 (NOT_SUSPENDED): DMA transfer from the source is not suspended - 0x1 (SUSPENDED): Suspend DMA transfer from the source **Value after reset:** 0x0 **Volatile:** true
7 :5	RW	**SRC_TR_WIDTH** Source Transfer Width. Mapped to AHB bus hsize. For a non-memory peripheral, typically the peripheral (source) FIFO width. This value must be less than or equl to DMAH_Mk_HDATA_WIDTH, where k is the AHB layer 1 to 4 where the source resides. **Values:** - 0x0 (SRC_TR_WIDTH_0): Source transfer width is 8 bits - 0x1 (SRC_TR_WIDTH_1): Source transfer width is 16 bits - 0x2 (SRC_TR_WIDTH_2): Source transfer width is 32 bits - 0x3 (SRC_TR_WIDTH_3): Source transfer width is 64 bits - 0x4 (SRC_TR_WIDTH_4): Source transfer width is 128 bits - 0x5 (SRC_TR_WIDTH_5): Source transfer width is 256 bits - 0x6 (SRC_TR_WIDTH_6): Source transfer width is 256 bits - 0x7 (SRC_TR_WIDTH_7): Source transfer width is 256 bits **Value after reset:** x **Volatile:** true
4 :0	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0

Offset: 200h	**SGR5: Source Gather Register for Channel 5**	**Init = 0**
63:32	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0
31:25	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0
24:19	RW	**SGC** Source Gather Count. Source contiguous tranfer count between successive gather boundaries. **Value after reset:** 0x0 **Volatile:** true

19:0	RW	**SGI** Source Gather Interval. **Value after reset:** 0x0 **Volatile:** true

Offset: 208h	**DSR5: Source Gather Register for Channel 5**	**Init = 0**
63:32	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0
31:25	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0
24:19	RW	**DSC** Destination Scatter Count. Destination contiguous transfer count between successive scatter boundaries. **Value after reset:** 0x0 **Volatile:** true
19:0	RW	**DSI** Destination Scatter Interval. **Value after reset:** 0x0 **Volatile:** true

Offset: 210h	**SAR6: Source Address for Channel 6**	**Init = 0**
63:32	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0 **Volatile:** true
31:0	RW	**SAR** Current Source Address of DMA transfer. Updated after each source transfer. The SINC field in the CTL6 register determines whether the address increments, decrements, or is left unchanged on every source transfer through the block transfer. **Value after reset:** 0x0 **Volatile:** true

Offset: 218h	**DAR6: Destination Address for Channel 6**	**Init = 0**
63:32	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0 **Volatile:** true

31:0	RW	**DAR** Current Destination address of DMA transfer. Updated after each destination transfer. The DINC field in the CTL6 register determines whether the address increments, decrements, or is left unchanged on every destination transfer throughout the block transfer. **Value after reset:** 0x0 **Volatile:** true

Offset: 220h	**LLP6: Linked List Pointer Register for Channel 6**	**Init = 0**
63:32	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0 **Volatile:** true
31:2	RW	**LOC** Starting Address In Memory of next LLI if block chaining is enabled. Note that the two LSBs of the starting address are not stored because the address is assumed to be aligned to a 32-bit boundary. LLI accesses are always 32-bit accesses (Hsize=2) aligned to 32-bit boundaries and cannot be changed or programmed to anything other than 32-bit. **Value after reset:** 0x0 **Volatile:** true
1 :0	RW	**LMS** List Master Select. Identifies the AHB layer/interface where the memory device that stores the next linked list item resides. This field does not exist if the configuration parameter is not set to NO_HARDCODE. In this case, the read-back value is always the hardcoded value. The maximum value of this field that can be read back is DMAH_NUM_MASTER_INT-1. **Values:** - 0x0 (LIST_MASTER_SELECT_1): The memory device stores the next linked list item on AHB master 1 - 0x1 (LIST_MASTER_SELECT_2): The memory device stores the next linked list item on AHB master 2 - 0x2 (LIST_MASTER_SELECT_3): The memory device stores the next linked list item on AHB master 3 - 0x3 (LIST_MASTER_SELECT_4): The memory device stores the next linked list item on AHB master 4 **Value after reset:** 0x0 **Volatile:** true

Offset: 228h	**CTL6: Control Register for Channel 6**	**Init = 0**
63:45	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0 **Volatile:** true

44	RW	**DONE** Done bit. If status write-back is enabled, the upper word of the control register, CTL6[63:32], is written to the control register location of the Linked List Item (LLI) in system memory at the end of the block transfer with the done bit set. Software can poll the LLI CTL6.DONE bit to see when a block transfer is complete. The LLI CTL6.DONE bit should be cleared when the linked lists are set up in memory prior to enabling the channel. LLI accesses are always 32-bit accesses (Hsize=2) aligned to 32-bit boundaries and cannot be changed or programmed to anything other than 32-bit. For more information, refer to ”Multi-Block Transfers”. **Values:** - 0x0 (DISABLED): DONE bit is deasserted the enb of block transfer - 0x1 (ENABLED): SET the DONE bit at the end of block transfer **Value after reset:** 0x0 **Volatile:** true
43:37	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0 **Volatile:** true
36:32	RW	**BLOCK_TS** Block Transfer Size. When the HDMA is the flow controller, the user writes this field before the channel is enabled in order to indicate the block size. The number programmed into BLOCK_TS indicates the total number of single transactions to perform for every block transfer; a single transaction is mapped to a single AMBA beat **Width:** The width of single transaction is determined by CTL6.SRC_TR_WIDTH. For further information on setting this field, refer to ”Transfer Operation”. Once the transfer starts, the read-back value is the total number of data items already read from the source peripheral, regardless of what is the flow controller. When the source or destination peripheral is assigned as the flow controller, then the maximum block size that can be read back saturates at DMAH_CH6_MAX_BLK_SIZE, but the actual block size can be greater. **Value after reset:** 0x2 **Volatile:** true
31:29	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0 **Volatile:** true
28	RW	**LLP_SRC_EN** Block chaining is enabled on the source side only if the LLP_SRC_EN field is high and LLP6.LOC is non-zero. For more information, see ”Block Chaining using Linked Lists”. **Values:** - 0x0 (LLP_SRC_DISABLE): Block chaining using Linked List is disabled on the Source side - 0x1 (LLP_SRC_ENABLE): Block chaining using Linked List is enabled on the Source side **Value after reset:** 0x0 **Volatile:** true

27	RW	**LLP_DST_EN** Block chaining is enabled on the destination side only if LLP_DST_EN field is high and LLP6.LOC is non-zero. For more information, see ”Block Chaining using Linked Lists” **Values:** - 0x0 (LLP_DST_DISABLE): Block chaining using Linked List is disabled on the Destination side - 0x1 (LLP_DST_ENABLE): Block chaining using Linked List is enabled on the Destination side **Value after reset:** 0x0 **Volatile:** true
26:25	RW	**SMS** Source Master Select. Identifies the Master Interface layer where the source device (peripheral or memory) resides. The maximum value of this field that can be read back is DMAH_NUM_MASTER_INT-1. **Values:** - 0x0 (SMS_0): Source device (peripheral or memory) is accessed from AHB master 1 - 0x1 (SMS_1): Source device (peripheral or memory) is accessed from AHB master 2 - 0x2 (SMS_2): Source device (peripheral or memory) is accessed from AHB master 3 - 0x3 (SMS_3): Source device (peripheral or memory) is accessed from AHB master 4 **Value after reset:** 0x0 **Volatile:** true
24:23	RW	**DMS** Destination Master Select. Identifies the Master Interface layer where the destination device (peripheral or memory) resides. The maximum value of this field that can be read back is DMAH_NUM_MASTER_INT-1. **Values:** - 0x0 (DMS_0): Destination device (peripheral or memory) is accessed from AHB master 1 - 0x1 (DMS_1): Destination device (peripheral or memory) is accessed from AHB master 2 - 0x2 (DMS_2): Destination device (peripheral or memory) is accessed from AHB master 3 - 0x3 (DMS_3): Destination device (peripheral or memory) is accessed from AHB master 4 **Value after reset:** 0x0 **Volatile:** true

22:20	RW	**TT_FC** Transfer Type and Flow Control. Flow control can be assigned to the HDMA, the source peripheral, or the destination peripheral. For more information on transfer types and flow control, refer to ”Setup/Operation of the HDMA Transfers”. Dependencies: If the configuration parameter DMAH_CH6_FC is set to DMA_FC_ONLY, then TT_FC[2] does not exist and TT_FC[2] always reads back 0. If DMAH_CH6_FC is set to SRC_FC_ONLY, then TT_FC[2:1] does not exist and TT_FC[2:1] always reads back 2’b10. If DMAH_CH6_FC is set to DST_FC_ONLY, then TT_FC[2:1] does not exist and TT_FC[2:1] always reads back 2’b11. For multi-block transfers using linked list operation, TT_FC must be constant for all blocks of this multi-block transfer. **Values:** - 0x0 (TT_FC_0): Transfer type is Memory to Memory and Flow Controller is HDMA - 0x1 (TT_FC_1): Transfer type is Memory to Peripheral and Flow Controller is HDMA - 0x2 (TT_FC_2): Transfer type is Peripheral to Memory and Flow Controller is HDMA - 0x3 (TT_FC_3): Transfer type is Peripheral to Peripheral and Flow Controller is HDMA - 0x4 (TT_FC_4): Transfer type is Peripheral to Memory and Flow Controller is Peripheral - 0x5 (TT_FC_5): Transfer type is Peripheral to Peripheral and Flow Controller is Source Peripheral - 0x6 (TT_FC_6): Transfer type is Memory to Peripheral and Flow Controller is Peripheral - 0x7 (TT_FC_7): Transfer type is Peripheral to Peripheral and Flow Controller is Destination Peripheral **Value after reset:** 0x3 **Volatile:** true
19	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0 **Volatile:** true
18	RW	**DST_SCATTER_EN** Destination scatter enable. Scatter on the destination side is applicable only when the CTL6.DINC bit indicates an incrementing or decrementing address control. **Values:** - 0x0 (DST_SCATTER_DISABLE): Destination Scatter is disabled - 0x1 (DST_SCATTER_ENABLE): Destination Scatter is enabled **Value after reset:** 0x0 **Volatile:** true
17	RW	**SRC_GATHER_EN** Destination gather enable. Gather on the destination side is applicable only when the CTL6.SINC bit indicates an incrementing or decrementing address control. **Values:** - 0x0 (SRC_SCATTER_DISABLE): Source gather is disabled - 0x1 (SRC_SCATTER_ENABLE): Source gather is enabled **Value after reset:** 0x0 **Volatile:** true

16:14	RW	**SRC_MSIZE** Source Burst Transaction Length. Number of data items, each of width CTL6.SRC_TR_WIDTH, to be read from the source every time a burst transferred request is made from either the corresponding hardware or software handshaking interface. **NOTE:** This value is not related to the AHB bus master HBURST bus. For information on the decoding for this field, see the ”Setting Up Transfers” section and for more inforamtion about this field, see the ”Choosing the Receive Watermark level” section in the HDMA Databook. **Values:** - 0x0 (SRC_MSIZE_0): Number of data items to be transferred is 1 - 0x1 (SRC_MSIZE_1): Number of data items to be transferred is 4 - 0x2 (SRC_MSIZE_2): Number of data items to be transferred is 8 - 0x3 (SRC_MSIZE_3): Number of data items to be transferred is 16 - 0x4 (SRC_MSIZE_4): Number of data items to be transferred is 32 - 0x5 (SRC_MSIZE_5): Number of data items to be transferred is 64 - 0x6 (SRC_MSIZE_6): Number of data items to be transferred is 128 - 0x7 (SRC_MSIZE_7): Number of data items to be transferred is 256 **Value after reset:** 0x1 **Volatile:** true
13:11	RW	**DEST_MSIZE** Destination Burst Transaction Length. Number of data items, each of width CTL6.DST_TR_WIDTH, to be written to the destination every time a destination burst transaction request is made from either the corresponding hardware or software handshaking interface. **NOTE:** This value is not related to the AHB bus master HBURST bus. **Values:** - 0x0 (DEST_MSIZE_0): Number of data items to be transferred is 1 - 0x1 (DEST_MSIZE_1): Number of data items to be transferred is 4 - 0x2 (DEST_MSIZE_2): Number of data items to be transferred is 8 - 0x3 (DEST_MSIZE_3): Number of data items to be transferred is 16 - 0x4 (DEST_MSIZE_4): Number of data items to be transferred is 32 - 0x5 (DEST_MSIZE_5): Number of data items to be transferred is 64 - 0x6 (DEST_MSIZE_6): Number of data items to be transferred is 128 - 0x7 (DEST_MSIZE_7): Number of data items to be transferred is 256 **Value after reset:** 0x1 **Volatile:** true
10:9	RW	**SINC** Source Address Increment. Indicates whether to increment or decrement the source address on every source transfer. If the device is fetching data from a source peripheral FIFO with a fixed address, then set this field to ”No change”. **Values:** - 0x0 (SINC_0): Increments the source address - 0x1 (SINC_1): Decrements the source address - 0x2 (SINC_2): No change in the source address - 0x3 (SINC_3): No change in the source address **Value after reset:** 0x0 **Volatile:** true

8 :7	RW	**DINC** Distination Address Increment. Indicates whether to increment or decrement the destination address on every destination transfer. If your device is writing data to a destination peripheral FIFO with a fixed address, then set this field to ”No Change”. **Values:** - 0x0 (DINC_0): Increments the distination address - 0x1 (DINC_1): Decrements the distination address - 0x2 (DINC_2): No change in the distination address - 0x3 (DINC_3): No change in the distination address **Value after reset:** 0x0 **Volatile:** true
6 :4	RW	**SRC_TR_WIDTH** Source Transfer Width. Mapped to AHB bus hsize. For a non-memory peripheral, typically the peripheral (source) FIFO width. This value must be less than or equl to DMAH_Mk_HDATA_WIDTH, where k is the AHB layer 1 to 4 where the source resides. **Values:** - 0x0 (SRC_TR_WIDTH_0): Source transfer width is 8 bits - 0x1 (SRC_TR_WIDTH_1): Source transfer width is 16 bits - 0x2 (SRC_TR_WIDTH_2): Source transfer width is 32 bits - 0x3 (SRC_TR_WIDTH_3): Source transfer width is 64 bits - 0x4 (SRC_TR_WIDTH_4): Source transfer width is 128 bits - 0x5 (SRC_TR_WIDTH_5): Source transfer width is 256 bits - 0x6 (SRC_TR_WIDTH_6): Source transfer width is 256 bits - 0x7 (SRC_TR_WIDTH_7): Source transfer width is 256 bits **Value after reset:** 0x2 **Volatile:** true
3 :1	RW	**DST_TR_WIDTH** Destination Transfer Width. Mapped to AHB bus hsize. For a non-memory peripheral, typically the peripheral (destination) FIFO width. This value must be less than or equal to DMAH_Mk_HDATA_WIDTH where k is the AHB layer 1 to 4 where the destination resides. For the decoding of this field, see the ”Setting Up Transfers” section in the HDMA Databook. **Values:** - 0x0 (DST_TR_WIDTH_0): Destination transfer width is 8 bits - 0x1 (DST_TR_WIDTH_1): Destination transfer width is 16 bits - 0x2 (DST_TR_WIDTH_2): Destination transfer width is 32 bits - 0x3 (DST_TR_WIDTH_3): Destination transfer width is 64 bits - 0x4 (DST_TR_WIDTH_4): Destination transfer width is 128 bits - 0x5 (DST_TR_WIDTH_5): Destination transfer width is 256 bits - 0x6 (DST_TR_WIDTH_6): Destination transfer width is 256 bits - 0x7 (DST_TR_WIDTH_7): Destination transfer width is 256 bits **Value after reset:** 0x2 **Volatile:** true

0	RW	**INT_EN** Interrupt Enable Bit. If set, then all interrupt-generating sources are enabled. Functions as a global mask bit for all interrupts for the channel; raw* interrupt registers still assert if CTL2.INT_EN=0. **Values:** - 0x0 (INTERRUPT_DISABLE): Interrupt is disabled - 0x1 (INTERRUPT_ENABLE): Interrupt is enabled **Value after reset:** 0x1 **Volatile:** true

Offset: 230h	**SSTAT6: Source Status Register for Channel 6**	**Init = 0**
63:32	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0 **Volatile:** true
31:0	RW	**SSTAT** Source status information retrieved by hardware from the address pointed to by the contents of the STATAR6 register **Value after reset:** 0x0 **Volatile:** true

Offset: 238h	**DSTAT6: Destination Status Register for Channel 6**	**Init = 0**
63:32	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0 **Volatile:** true
31:0	RW	**DSTAT** Destination status information retrieved by hardware from the address pointed to by the contents of DSTATAR6 register. **Value after reset:** 0x0 **Volatile:** true

Offset: 240h	**SSTATAR6: Source Status Address Register for Channel 6**	**Init = 0**
63:32	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0 **Volatile:** true
31:0	RW	**SSTATAR** Pointer from where hardware can fetch the source status information, which is registered in the SSTAT6 register and written out to the SSTAT6 register location of the LLI before the start of the next block. **Value after reset:** 0x0 **Volatile:** true

Offset: 248h		DSTATAR6: Source Status Address Register for Channel 6	Init = 0
63:32	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0	
31:0	RW	**DSTATAR** Pointer from where hardware can fetch the destination status information, which is registered in the DSTAT6 register and written out to the DSTAT4 register location of the LLI before the start of the next block. **Value after reset:** 0x0 **Volatile:** true	

Offset: 250h		CFG6: Configuration Register for Channel 6	Init = 0
63:47	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0 **Volatile:** true	
46:43	RW	**DEST_PER** Destination hardware interface. Assigns a hardware handshaking interface (0 : DMAH_NUM_HS_INT-1) to the destination of channel 6 if the CFG6.HS_SEL_DST field is 0; otherwise, this field is ignored. The channel can then communicate with the destination peripheral connected to that interface through the assigned hardware handshaking interface. **Note1:** For correct HDMA operation, only one peripheral (source or destination) should be assigned to the same handshaking interface. **Note2:** This field does not exist if the configuration parameter DMAH_NUM_HS_INT is set to 0. **Value after reset:** 0x0 **Volatile:** true	
42:39	RW	**SRC_PER** Source Hardware Interface. Assigns a hardware handshaking interface (0 : DMAH_NUM_HS_INT-1) to the source of channel 6 if the CFG6.HS_SEL_SRC field is 0; otherwise, this field is ignored. The channel can then communicate with the source peripheral connected to that interface through the assigned hardware handshaking interface. **Note1:** For correct HDMA operation, only one peripheral (source or destination) should be assigned to the same handshaking interface. **Note2:** This field does not exist if the configuration parameter DMAH_NUM_HS_INT is set to 0. **Value after reset:** 0x0 **Volatile:** true	

38	RW	**SS_UPD_EN** Source Status Update Enable. Source status information is fetched only from the location pointed to by the SSTATAR6 register, stored in the SSTAT6 register and written out to the SSTAT6 location of the LLI, if SS_UPD_EN is high. **Note:** This enalbe is applicable only if DMAH_CH6_STAT_SRC is set to True. This field does not exist if the configuration parameter DMAH_CH6_STAT_SRC is set to False; in this case, the read-back value is always 0. **Values:** - 0x0 (DISABLED): Source Status Update is disabled. - 0x1 (ENABLED): Source Status Update is enabled. **Value after reset:** 0x0 **Volatile:** true
37	RW	**DS_UPD_EN** Destination Status Update Enable. Destination status information is fetched only from the location pointed to by the DSTATAR0 register, stored in the DSTAT6 register and written out to the DSTAT6 location of the LLI, if DS_UPD_EN is high. This field does not exist if the configuration parameter DMAH_CH1_SATAT_DST is set to False; in this case, the read-back value is always 0. **Values:** - 0x0 (DISABLED): Destination Status Update is disabled. - 0x1 (ENABLED): Destination Status Update is enabled. **Value after reset:** 0x0 **Volatile:** true
36:34	RW	**PROTCTL** Protection Control bits used to drive the AHB HPROT[3:1] bus. The AMBA Specification recommends that the default of HPROT indicates a non-cached, non-buffered, privileged data access. The reset value is used to indicate such an access. HPROT[0] is tied high because all transfers are data accesses, as there are no opcode fetches. There is a one-to-one mapping of these register bits to the HPROT[3:1] master interface signals. Mapping of HPROT bus is as follows: - 1'b1 to HPROT[0] - CFG1.PROTCTL[1] to HPROT[1] - CFG1.PROTCTL[2] to HPROT[2] - CFG1.PROTCTL[3] to HPROT[3] **Value after reset:** 0x1 **Volatile:** true
33	RW	**FIFO_MODE** FIFO Mode Select. Determines how much space or data needs to be available in the FIFO before a burst transaction request is serviced. **Values:** - 0x0 (FIFO_MODE_0): Space/data available for single AHB transfer of the specified transfer width - 0x1 (FIFO_MODE_1): Data available is greater than or equal to half the FIFO depth for destination transfers and space available is greater than half the fifo depth for source transfers. The exceptions are at the end of a burst transaction request or at the end of a block transfer. **Value after reset:** 0x0 **Volatile:** true

32	RW	**FCMODE** Flow Control Mode. Determines when source transaction requests are serviced when the Destination Peripheral is the flow controller. **Values:** - 0x0 (FCMODE_0): Source transaction requests are serviced when they occur. Data pre-fetching is enabled - 0x1 (FCMODE_1): Source transaction requests are not serviced until a destination transaction request occurs. In this mode, the amount of data transferred from the source is limited so that it is guaranteed to be transferred to the destination prior to block termination by the destination. Data pre-fetching is disabled. **Value after reset:** 0x0 **Volatile:** true
31	RW	**RELOAD_DST** Automatic Destination Reload. The DAR2 register can be automatically reloaded from its initial value at the end of every block for multi-block transfers. A new block transfer is then initiated. This register does not exist if the configuration parameter DMAH_CH6_MULTI_BLK_EN is not selected; in this case, the read-back value is always 0. **Values:** - 0x0 (DISABLE): Destination Reload Disabled. - 0x1 (ENABLE): Destination Reload Enabled. **Value after reset:** 0x0 **Volatile:** true
30	RW	**RELOAD_SRC** Automatic Source Reload. The SAR6 register can be automatically reloaded from its initial value at the end of every block for multi-block transfers. A new block transfer is then initiated. This field does not exist if the configuration parameter DMAH_CH6_MULTI_BLK_EN is not selected; in this case, the read-back value is always 0. **Values:** - 0x0 (DISABLE): Source Reload Disabled. - 0x1 (ENABLE): Source Reload Enabled. **Value after reset:** 0x0 **Volatile:** true
29:20	R	**MAX_ABRST** Maximum AMBA Burst Length. Maximum AMBA burst length that is used for DMA transfers on this channel. A value of 0 indicates that software is not limiting the maximum AMBA burst length for DMA transfers on this channel. This field does not exist if the configuration parameter DMAH_MABRST is not selected; in this case, the read-back value is always 0, and the maximum AMBA burst length cannot be limited by software. **Value after reset:** 0x0 **Volatile:** true

19	RW	**SRC_HS_POL** Source Handshaking Interface Polarity. **Values:** - 0x0 (ACTIVE_HIGH): Source Handshaking Interface Polarity is Active high - 0x1 (ACTIVE_LOW): Source Handshaking Interface Polarity is Active low **Value after reset:** 0x0 **Volatile:** true
18	RW	**DST_HS_POL** Destination Handshaking Interface Polarity. **Values:** - 0x0 (ACTIVE_HIGH): Destination Handshaking Interface Polarity is Active high - 0x1 (ACTIVE_LOW): Destination Handshaking Interface Polarity is Active low **Value after reset:** 0x0 **Volatile:** true
17	RW	**LOCK_B** Bus Lock Bit. When active, the AHB bus master signal hlock is asserted for the duration specified in CFG0.LOCK_B_L. For more information, refer to ”Locked DMA Transfers”. This field does not exist if the configuration parameter DMAH_CH6_LOCK_EN is set to False; in this case, the read-back value is always 0. **Values:** - 0x0 (DISABLED): Bus lock bit is not enabled - 0x1 (ENABLED): Bus lock bit is enabled **Value after reset:** 0x0 **Volatile:** true
16	RW	**LOCK_CH** Channel Lock Bit. When the channel is granted control of the master bus interface and if the CFG6.LOCK_CH bit is asserted, then no other channels are granted control of the master bus interface for the duration specified in CFG6.LOCK_CH_L. Indicates to the master bus interface arbiter that this channel wants exclusive access to the master bus interface for the duration specified in CFG3.LOCK_CH_L. This field does not exist if the configuration parameter DMAH_CH6_LOCK_EN is set to False; in this case, the read-back value is always 0. **Values:** - 0x0 (DISABLED): Channel lock bit is not enabled - 0x1 (ENABLED): Channel lock bit is enabled **Value after reset:** 0x0 **Volatile:** true
14:15	RW	**LOCK_B_L** Bus lock level. Indicates the duration over which CFG6.LOCK_B bit applies. This field does not exist if the parameter DMAH_CH6_LOCK_EN is set to False; in this case, the read-back value is always 0. **Values:** - 0x0 (LOCK_B_L_0): Over complete DMA transfer - 0x1 (LOCK_B_L_1): Over complete DMA block transfer - 0x2 (LOCK_B_L_2): Over complete DMA transaction - 0x3 (LOCK_B_L_3): Over complete DMA transaction **Value after reset:** 0x0 **Volatile:** true

13:12	RW	**LOCK_CH_L** Channel Local Level. Indicates the duration over which CFG6.LOCK_CH applies. This field does not exist if the configuration parameter DMAH_CH6_LOCK_EN is set to False; in this case, the read-back value is always 0. **Values:** - 0x0 (LOCK_CH_L_0): Over complete DMA transfer - 0x1 (LOCK_CH_L_1): Over complete DMA block transfer - 0x2 (LOCK_CH_L_2): Over complete DMA transaction - 0x3 (LOCK_CH_L_3): Over complete DMA transaction **Value after reset:** 0x0 **Volatile:** true
11	RW	**HS_SEL_SRC** Source Software or Hardware Handshaking Select. This register selects which of the handshaking interfaces - hardware or software - is active for source requests on this channel. If the source peripheral is memory, then this bit is ignored. **Values:** - 0x0 (HARDWARE_HS): Hardware handshaking interface. Software initiated transaction requests are ignored. - 0x1 (SOFTWARE_HS): Software handshaking interface. Hardware initiated transaction requests are ignored. **Value after reset:** 0x1 **Volatile:** true
10	RW	**HS_SEL_DST** Destination Software or Hardware Handshaking Select. This register selects which of the handshaking interfaces - hardware or software - is active for destination requests on this channel. If the destination peripheral is memory, then this bit is ignored. **Values:** - 0x0 (HARDWARE_HS): Hardware handshaking interface. Software initiated transaction requests are ignored. - 0x1 (SOFTWARE_HS): Software handshaking interface. Hardware initiated transaction requests are ignored. **Value after reset:** 0x1 **Volatile:** true
9	R	**FIFO_EMPTY** Channel FIFO status. Indicates if there is data left in the channel FIFO. Can be used in conjunction with CFG6.CH_SUSP to cleanly disable a channel. For more information, refer to ”Disabling a Channel Prior to Transfer Completion”. **Values:** - 0x0 (NOT_EMPTY): Channel FIFO is not empty - 0x1 (EMPTY): Channel FIFO is empty **Value after reset:** 0x1 **Volatile:** true

8	R	**CH_SUSP** Channel Suspend. Suspends all DMA data transfers from the source until this bit is cleared. There is no guarantee that the current transaction will complete. Can also be used in conjunction with CFG6.FIFO_EMPTY to cleanly disable a channel without losing any data. For more information, refer to ”Disabling a Channel Prior to Transfer Completion”. **Values:** - 0x0 (NOT_SUSPENDED): DMA transfer from the source is not suspended - 0x1 (SUSPENDED): Suspend DMA transfer from the source **Value after reset:** 0x0 **Volatile:** true
7 :5	RW	**SRC_TR_WIDTH** Source Transfer Width. Mapped to AHB bus hsize. For a non-memory peripheral, typically the peripheral (source) FIFO width. This value must be less than or equl to DMAH_Mk_HDATA_WIDTH, where k is the AHB layer 1 to 4 where the source resides. **Values:** - 0x0 (SRC_TR_WIDTH_0): Source transfer width is 8 bits - 0x1 (SRC_TR_WIDTH_1): Source transfer width is 16 bits - 0x2 (SRC_TR_WIDTH_2): Source transfer width is 32 bits - 0x3 (SRC_TR_WIDTH_3): Source transfer width is 64 bits - 0x4 (SRC_TR_WIDTH_4): Source transfer width is 128 bits - 0x5 (SRC_TR_WIDTH_5): Source transfer width is 256 bits - 0x6 (SRC_TR_WIDTH_6): Source transfer width is 256 bits - 0x7 (SRC_TR_WIDTH_7): Source transfer width is 256 bits **Value after reset:** x **Volatile:** true
4 :0	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0

Offset: 258h	**SGR6: Source Gather Register for Channel 6**	**Init = 0**
63:32	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0
31:25	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0
24:19	RW	**SGC** Source Gather Count. Source contiguous tranfer count between successive gather boundaries. **Value after reset:** 0x0 **Volatile:** true

19:0	RW	**SGI** Source Gather Interval. **Value after reset:** 0x0 **Volatile:** true

Offset: 260h	**DSR6: Source Gather Register for Channel 6**	**Init = 0**
63:32	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0
31:25	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0
24:19	RW	**DSC** Destination Scatter Count. Destination contiguous transfer count between successive scatter boundaries. **Value after reset:** 0x0 **Volatile:** true
19:0	RW	**DSI** Destination Scatter Interval. **Value after reset:** 0x0 **Volatile:** true

Offset: 268h	**SAR7: Source Address for Channel 7**	**Init = 0**
63:32	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0 **Volatile:** true
31:0	RW	**SAR** Current Source Address of DMA transfer. Updated after each source transfer. The SINC field in the CTL7 register determines whether the address increments, decrements, or is left unchanged on every source transfer through the block transfer. **Value after reset:** 0x0 **Volatile:** true

Offset: 270h	**DAR7: Destination Address for Channel 7**	**Init = 0**
63:32	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0 **Volatile:** true

31:0	RW	**DAR** Current Destination address of DMA transfer. Updated after each destination transfer. The DINC field in the CTL7 register determines whether the address increments, decrements, or is left unchanged on every destination transfer throughout the block transfer. **Value after reset:** 0x0 **Volatile:** true

Offset: 278h	**LLP7: Linked List Pointer Register for Channel 7**	**Init = 0**
63:32	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0 **Volatile:** true
31:2	RW	**LOC** Starting Address In Memory of next LLI if block chaining is enabled. Note that the two LSBs of the starting address are not stored because the address is assumed to be aligned to a 32-bit boundary. LLI accesses are always 32-bit accesses (Hsize=2) aligned to 32-bit boundaries and cannot be changed or programmed to anything other than 32-bit. **Value after reset:** 0x0 **Volatile:** true
1 :0	RW	**LMS** List Master Select. Identifies the AHB layer/interface where the memory device that stores the next linked list item resides. This field does not exist if the configuration parameter is not set to NO_HARDCODE. In this case, the read-back value is always the hardcoded value. The maximum value of this field that can be read back is DMAH_NUM_MASTER_INT-1. **Values:** - 0x0 (LIST_MASTER_SELECT_1): The memory device stores the next linked list item on AHB master 1 - 0x1 (LIST_MASTER_SELECT_2): The memory device stores the next linked list item on AHB master 2 - 0x2 (LIST_MASTER_SELECT_3): The memory device stores the next linked list item on AHB master 3 - 0x3 (LIST_MASTER_SELECT_4): The memory device stores the next linked list item on AHB master 4 **Value after reset:** 0x0 **Volatile:** true

Offset: 280h	**CTL7: Control Register for Channel 7**	**Init = 0**
63:45	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0 **Volatile:** true

44	RW	**DONE** Done bit. If status write-back is enabled, the upper word of the control register, CTL7[63:32], is written to the control register location of the Linked List Item (LLI) in system memory at the end of the block transfer with the done bit set. Software can poll the LLI CTL7.DONE bit to see when a block transfer is complete. The LLI CTL7.DONE bit should be cleared when the linked lists are set up in memory prior to enabling the channel. LLI accesses are always 32-bit accesses (Hsize=2) aligned to 32-bit boundaries and cannot be changed or programmed to anything other than 32-bit. For more information, refer to ”Multi-Block Transfers”. **Values:** - 0x0 (DISABLED): DONE bit is deasserted the enb of block transfer - 0x1 (ENABLED): SET the DONE bit at the end of block transfer **Value after reset:** 0x0 **Volatile:** true
43:37	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0 **Volatile:** true
36:32	RW	**BLOCK_TS** Block Transfer Size. When the HDMA is the flow controller, the user writes this field before the channel is enabled in order to indicate the block size. The number programmed into BLOCK_TS indicates the total number of single transactions to perform for every block transfer; a single transaction is mapped to a single AMBA beat **Width:** The width of single transaction is determined by CTL7.SRC_TR_WIDTH. For further information on setting this field, refer to ”Transfer Operation”. Once the transfer starts, the read-back value is the total number of data items already read from the source peripheral, regardless of what is the flow controller. When the source or destination peripheral is assigned as the flow controller, then the maximum block size that can be read back saturates at DMAH_CH7_MAX_BLK_SIZE, but the actual block size can be greater. **Value after reset:** 0x2 **Volatile:** true
31:29	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0 **Volatile:** true
28	RW	**LLP_SRC_EN** Block chaining is enabled on the source side only if the LLP_SRC_EN field is high and LLP7.LOC is non-zero. For more information, see ”Block Chaining using Linked Lists”. **Values:** - 0x0 (LLP_SRC_DISABLE): Block chaining using Linked List is disabled on the Source side - 0x1 (LLP_SRC_ENABLE): Block chaining using Linked List is enabled on the Source side **Value after reset:** 0x0 **Volatile:** true

27	RW	**LLP_DST_EN** Block chaining is enabled on the destination side only if LLP_DST_EN field is high and LLP7.LOC is non-zero. For more information, see ”Block Chaining using Linked Lists” **Values:** - 0x0 (LLP_DST_DISABLE): Block chaining using Linked List is disabled on the Destination side - 0x1 (LLP_DST_ENABLE): Block chaining using Linked List is enabled on the Destination side **Value after reset:** 0x0 **Volatile:** true
26:25	RW	**SMS** Source Master Select. Identifies the Master Interface layer where the source device (peripheral or memory) resides. The maximum value of this field that can be read back is DMAH_NUM_MASTER_INT-1. **Values:** - 0x0 (SMS_0): Source device (peripheral or memory) is accessed from AHB master 1 - 0x1 (SMS_1): Source device (peripheral or memory) is accessed from AHB master 2 - 0x2 (SMS_2): Source device (peripheral or memory) is accessed from AHB master 3 - 0x3 (SMS_3): Source device (peripheral or memory) is accessed from AHB master 4 **Value after reset:** 0x0 **Volatile:** true
24:23	RW	**DMS** Destination Master Select. Identifies the Master Interface layer where the destination device (peripheral or memory) resides. The maximum value of this field that can be read back is DMAH_NUM_MASTER_INT-1. **Values:** - 0x0 (DMS_0): Destination device (peripheral or memory) is accessed from AHB master 1 - 0x1 (DMS_1): Destination device (peripheral or memory) is accessed from AHB master 2 - 0x2 (DMS_2): Destination device (peripheral or memory) is accessed from AHB master 3 - 0x3 (DMS_3): Destination device (peripheral or memory) is accessed from AHB master 4 **Value after reset:** 0x0 **Volatile:** true

22:20	RW	**TT_FC** Transfer Type and Flow Control. Flow control can be assigned to the HDMA, the source peripheral, or the destination peripheral. For more information on transfer types and flow control, refer to ”Setup/Operation of the HDMA Transfers”. Dependencies: If the configuration parameter DMAH_CH7_FC is set to DMA_FC_ONLY, then TT_FC[2] does not exist and TT_FC[2] always reads back 0. If DMAH_CH7_FC is set to SRC_FC_ONLY, then TT_FC[2:1] does not exist and TT_FC[2:1] always reads back 2’b10. If DMAH_CH7_FC is set to DST_FC_ONLY, then TT_FC[2:1] does not exist and TT_FC[2:1] always reads back 2’b11. For multi-block transfers using linked list operation, TT_FC must be constant for all blocks of this multi-block transfer. **Values:** - 0x0 (TT_FC_0): Transfer type is Memory to Memory and Flow Controller is HDMA - 0x1 (TT_FC_1): Transfer type is Memory to Peripheral and Flow Controller is HDMA - 0x2 (TT_FC_2): Transfer type is Peripheral to Memory and Flow Controller is HDMA - 0x3 (TT_FC_3): Transfer type is Peripheral to Peripheral and Flow Controller is HDMA - 0x4 (TT_FC_4): Transfer type is Peripheral to Memory and Flow Controller is Peripheral - 0x5 (TT_FC_5): Transfer type is Peripheral to Peripheral and Flow Controller is Source Peripheral - 0x6 (TT_FC_6): Transfer type is Memory to Peripheral and Flow Controller is Peripheral - 0x7 (TT_FC_7): Transfer type is Peripheral to Peripheral and Flow Controller is Destination Peripheral **Value after reset:** 0x3 **Volatile:** true
19	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0 **Volatile:** true
18	RW	**DST_SCATTER_EN** Destination scatter enable. Scatter on the destination side is applicable only when the CTL7.DINC bit indicates an incrementing or decrementing address control. **Values:** - 0x0 (DST_SCATTER_DISABLE): Destination Scatter is disabled - 0x1 (DST_SCATTER_ENABLE): Destination Scatter is enabled **Value after reset:** 0x0 **Volatile:** true
17	RW	**SRC_GATHER_EN** Destination gather enable. Gather on the destination side is applicable only when the CTL7.SINC bit indicates an incrementing or decrementing address control. **Values:** - 0x0 (SRC_SCATTER_DISABLE): Source gather is disabled - 0x1 (SRC_SCATTER_ENABLE): Source gather is enabled **Value after reset:** 0x0 **Volatile:** true

16:14	RW	**SRC_MSIZE** Source Burst Transaction Length. Number of data items, each of width CTL7.SRC_TR_WIDTH, to be read from the source every time a burst transferred request is made from either the corresponding hardware or software handshaking interface. **NOTE:** This value is not related to the AHB bus master HBURST bus. For information on the decoding for this field, see the ”Setting Up Transfers” section and for more inforamtion about this field, see the ”Choosing the Receive Watermark level” section in the HDMA Databook. **Values:** - 0x0 (SRC_MSIZE_0): Number of data items to be transferred is 1 - 0x1 (SRC_MSIZE_1): Number of data items to be transferred is 4 - 0x2 (SRC_MSIZE_2): Number of data items to be transferred is 8 - 0x3 (SRC_MSIZE_3): Number of data items to be transferred is 16 - 0x4 (SRC_MSIZE_4): Number of data items to be transferred is 32 - 0x5 (SRC_MSIZE_5): Number of data items to be transferred is 64 - 0x6 (SRC_MSIZE_6): Number of data items to be transferred is 128 - 0x7 (SRC_MSIZE_7): Number of data items to be transferred is 256 **Value after reset:** 0x1 **Volatile:** true
13:11	RW	**DEST_MSIZE** Destination Burst Transaction Length. Number of data items, each of width CTL7.DST_TR_WIDTH, to be written to the destination every time a destination burst transaction request is made from either the corresponding hardware or software handshaking interface. **NOTE:** This value is not related to the AHB bus master HBURST bus. **Values:** - 0x0 (DEST_MSIZE_0): Number of data items to be transferred is 1 - 0x1 (DEST_MSIZE_1): Number of data items to be transferred is 4 - 0x2 (DEST_MSIZE_2): Number of data items to be transferred is 8 - 0x3 (DEST_MSIZE_3): Number of data items to be transferred is 16 - 0x4 (DEST_MSIZE_4): Number of data items to be transferred is 32 - 0x5 (DEST_MSIZE_5): Number of data items to be transferred is 64 - 0x6 (DEST_MSIZE_6): Number of data items to be transferred is 128 - 0x7 (DEST_MSIZE_7): Number of data items to be transferred is 256 **Value after reset:** 0x1 **Volatile:** true
10:9	RW	**SINC** Source Address Increment. Indicates whether to increment or decrement the source address on every source transfer. If the device is fetching data from a source peripheral FIFO with a fixed address, then set this field to ”No change”. **Values:** - 0x0 (SINC_0): Increments the source address - 0x1 (SINC_1): Decrements the source address - 0x2 (SINC_2): No change in the source address - 0x3 (SINC_3): No change in the source address **Value after reset:** 0x0 **Volatile:** true

8 :7	RW	**DINC** Distination Address Increment. Indicates whether to increment or decrement the destination address on every destination transfer. If your device is writing data to a destination peripheral FIFO with a fixed address, then set this field to ”No Change”. **Values:** - 0x0 (DINC_0): Increments the distination address - 0x1 (DINC_1): Decrements the distination address - 0x2 (DINC_2): No change in the distination address - 0x3 (DINC_3): No change in the distination address **Value after reset:** 0x0 **Volatile:** true
6 :4	RW	**SRC_TR_WIDTH** Source Transfer Width. Mapped to AHB bus hsize. For a non-memory peripheral, typically the peripheral (source) FIFO width. This value must be less than or equl to DMAH_Mk_HDATA_WIDTH, where k is the AHB layer 1 to 4 where the source resides. **Values:** - 0x0 (SRC_TR_WIDTH_0): Source transfer width is 8 bits - 0x1 (SRC_TR_WIDTH_1): Source transfer width is 16 bits - 0x2 (SRC_TR_WIDTH_2): Source transfer width is 32 bits - 0x3 (SRC_TR_WIDTH_3): Source transfer width is 64 bits - 0x4 (SRC_TR_WIDTH_4): Source transfer width is 128 bits - 0x5 (SRC_TR_WIDTH_5): Source transfer width is 256 bits - 0x6 (SRC_TR_WIDTH_6): Source transfer width is 256 bits - 0x7 (SRC_TR_WIDTH_7): Source transfer width is 256 bits **Value after reset:** 0x2 **Volatile:** true
3 :1	RW	**DST_TR_WIDTH** Destination Transfer Width. Mapped to AHB bus hsize. For a non-memory peripheral, typically the peripheral (destination) FIFO width. This value must be less than or equal to DMAH_Mk_HDATA_WIDTH where k is the AHB layer 1 to 4 where the destination resides. For the decoding of this field, see the ”Setting Up Transfers” section in the HDMA Databook. **Values:** - 0x0 (DST_TR_WIDTH_0): Destination transfer width is 8 bits - 0x1 (DST_TR_WIDTH_1): Destination transfer width is 16 bits - 0x2 (DST_TR_WIDTH_2): Destination transfer width is 32 bits - 0x3 (DST_TR_WIDTH_3): Destination transfer width is 64 bits - 0x4 (DST_TR_WIDTH_4): Destination transfer width is 128 bits - 0x5 (DST_TR_WIDTH_5): Destination transfer width is 256 bits - 0x6 (DST_TR_WIDTH_6): Destination transfer width is 256 bits - 0x7 (DST_TR_WIDTH_7): Destination transfer width is 256 bits **Value after reset:** 0x2 **Volatile:** true

0	RW	**INT_EN** Interrupt Enable Bit. If set, then all interrupt-generating sources are enabled. Functions as a global mask bit for all interrupts for the channel; raw* interrupt registers still assert if CTL2.INT_EN=0. **Values:** - 0x0 (INTERRUPT_DISABLE): Interrupt is disabled - 0x1 (INTERRUPT_ENABLE): Interrupt is enabled **Value after reset:** 0x1 **Volatile:** true

Offset: 288h	**SSTAT7: Source Status Register for Channel 7**	**Init = 0**
63:32	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0 **Volatile:** true
31:0	RW	**SSTAT** Source status information retrieved by hardware from the address pointed to by the contents of the STATAR7 register **Value after reset:** 0x0 **Volatile:** true

Offset: 290h	**DSTAT7: Destination Status Register for Channel 7**	**Init = 0**
63:32	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0 **Volatile:** true
31:0	RW	**DSTAT** Destination status information retrieved by hardware from the address pointed to by the contents of DSTATAR7 register. **Value after reset:** 0x0 **Volatile:** true

Offset: 298h	**SSTATAR7: Source Status Address Register for Channel 7**	**Init = 0**
63:32	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0 **Volatile:** true
31:0	RW	**SSTATAR** Pointer from where hardware can fetch the source status information, which is registered in the SSTAT7 register and written out to the SSTAT7 register location of the LLI before the start of the next block. **Value after reset:** 0x0 **Volatile:** true

Offset: 2A0h		DSTATAR7: Source Status Address Register for Channel 7	Init = 0
63:32	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0	
31:0	RW	**DSTATAR** Pointer from where hardware can fetch the destination status information, which is registered in the DSTAT7 register and written out to the DSTAT4 register location of the LLI before the start of the next block. **Value after reset:** 0x0 **Volatile:** true	

Offset: 2A8h		CFG7: Configuration Register for Channel 7	Init = 0
63:47	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0 **Volatile:** true	
46:43	RW	**DEST_PER** Destination hardware interface. Assigns a hardware handshaking interface (0 : DMAH_NUM_HS_INT-1) to the destination of channel 7 if the CFG7.HS_SEL_DST field is 0; otherwise, this field is ignored. The channel can then communicate with the destination peripheral connected to that interface through the assigned hardware handshaking interface. **Note1:** For correct HDMA operation, only one peripheral (source or destination) should be assigned to the same handshaking interface. **Note2:** This field does not exist if the configuration parameter DMAH_NUM_HS_INT is set to 0. **Value after reset:** 0x0 **Volatile:** true	
42:39	RW	**SRC_PER** Source Hardware Interface. Assigns a hardware handshaking interface (0 : DMAH_NUM_HS_INT-1) to the source of channel 7 if the CFG7.HS_SEL_SRC field is 0; otherwise, this field is ignored. The channel can then communicate with the source peripheral connected to that interface through the assigned hardware handshaking interface. **Note1:** For correct HDMA operation, only one peripheral (source or destination) should be assigned to the same handshaking interface. **Note2:** This field does not exist if the configuration parameter DMAH_NUM_HS_INT is set to 0. **Value after reset:** 0x0 **Volatile:** true	

38	RW	**SS_UPD_EN** Source Status Update Enable. Source status information is fetched only from the location pointed to by the SSTATAR7 register, stored in the SSTAT7 register and written out to the SSTAT7 location of the LLI, if SS_UPD_EN is high. **Note:** This enalbe is applicable only if DMAH_CH7_STAT_SRC is set to True. This field does not exist if the configuration parameter DMAH_CH7_STAT_SRC is set to False; in this case, the read-back value is always 0. **Values:** - 0x0 (DISABLED): Source Status Update is disabled. - 0x1 (ENABLED): Source Status Update is enabled. **Value after reset:** 0x0 **Volatile:** true
37	RW	**DS_UPD_EN** Destination Status Update Enable. Destination status information is fetched only from the location pointed to by the DSTATAR0 register, stored in the DSTAT7 register and written out to the DSTAT6 location of the LLI, if DS_UPD_EN is high. This field does not exist if the configuration parameter DMAH_CH1_SATAT_DST is set to False; in this case, the read-back value is always 0. **Values:** - 0x0 (DISABLED): Destination Status Update is disabled. - 0x1 (ENABLED): Destination Status Update is enabled. **Value after reset:** 0x0 **Volatile:** true
36:34	RW	**PROTCTL** Protection Control bits used to drive the AHB HPROT[3:1] bus. The AMBA Specification recommends that the default of HPROT indicates a non-cached, non-buffered, privileged data access. The reset value is used to indicate such an access. HPROT[0] is tied high because all transfers are data accesses, as there are no opcode fetches. There is a one-to-one mapping of these register bits to the HPROT[3:1] master interface signals. Mapping of HPROT bus is as follows: - 1'b1 to HPROT[0] - CFG1.PROTCTL[1] to HPROT[1] - CFG1.PROTCTL[2] to HPROT[2] - CFG1.PROTCTL[3] to HPROT[3] **Value after reset:** 0x1 **Volatile:** true
33	RW	**FIFO_MODE** FIFO Mode Select. Determines how much space or data needs to be available in the FIFO before a burst transaction request is serviced. **Values:** - 0x0 (FIFO_MODE_0): Space/data available for single AHB transfer of the specified transfer width - 0x1 (FIFO_MODE_1): Data available is greater than or equal to half the FIFO depth for destination transfers and space available is greater than half the fifo depth for source transfers. The exceptions are at the end of a burst transaction request or at the end of a block transfer. **Value after reset:** 0x0 **Volatile:** true

32	RW	**FCMODE** Flow Control Mode. Determines when source transaction requests are serviced when the Destination Peripheral is the flow controller. **Values:** - 0x0 (FCMODE_0): Source transaction requests are serviced when they occur. Data pre-fetching is enabled - 0x1 (FCMODE_1): Source transaction requests are not serviced until a destination transaction request occurs. In this mode, the amount of data transferred from the source is limited so that it is guaranteed to be transferred to the destination prior to block termination by the destination. Data pre-fetching is disabled. **Value after reset:** 0x0 **Volatile:** true
31	RW	**RELOAD_DST** Automatic Destination Reload. The DAR2 register can be automatically reloaded from its initial value at the end of every block for multi-block transfers. A new block transfer is then initiated. This register does not exist if the configuration parameter DMAH_CH7_MULTI_BLK_EN is not selected; in this case, the read-back value is always 0. **Values:** - 0x0 (DISABLE): Destination Reload Disabled. - 0x1 (ENABLE): Destination Reload Enabled. **Value after reset:** 0x0 **Volatile:** true
30	RW	**RELOAD_SRC** Automatic Source Reload. The SAR7 register can be automatically reloaded from its initial value at the end of every block for multi-block transfers. A new block transfer is then initiated. This field does not exist if the configuration parameter DMAH_CH7_MULTI_BLK_EN is not selected; in this case, the read-back value is always 0. **Values:** - 0x0 (DISABLE): Source Reload Disabled. - 0x1 (ENABLE): Source Reload Enabled. **Value after reset:** 0x0 **Volatile:** true
29:20	R	**MAX_ABRST** Maximum AMBA Burst Length. Maximum AMBA burst length that is used for DMA transfers on this channel. A value of 0 indicates that software is not limiting the maximum AMBA burst length for DMA transfers on this channel. This field does not exist if the configuration parameter DMAH_MABRST is not selected; in this case, the read-back value is always 0, and the maximum AMBA burst length cannot be limited by software. **Value after reset:** 0x0 **Volatile:** true

19	RW	**SRC_HS_POL** Source Handshaking Interface Polarity. **Values:** - 0x0 (ACTIVE_HIGH): Source Handshaking Interface Polarity is Active high - 0x1 (ACTIVE_LOW): Source Handshaking Interface Polarity is Active low **Value after reset:** 0x0 **Volatile:** true
18	RW	**DST_HS_POL** Destination Handshaking Interface Polarity. **Values:** - 0x0 (ACTIVE_HIGH): Destination Handshaking Interface Polarity is Active high - 0x1 (ACTIVE_LOW): Destination Handshaking Interface Polarity is Active low **Value after reset:** 0x0 **Volatile:** true
17	RW	**LOCK_B** Bus Lock Bit. When active, the AHB bus master signal hlock is asserted for the duration specified in CFG0.LOCK_B_L. For more information, refer to ”Locked DMA Transfers”. This field does not exist if the configuration parameter DMAH_CH7_LOCK_EN is set to False; in this case, the read-back value is always 0. **Values:** - 0x0 (DISABLED): Bus lock bit is not enabled - 0x1 (ENABLED): Bus lock bit is enabled **Value after reset:** 0x0 **Volatile:** true
16	RW	**LOCK_CH** Channel Lock Bit. When the channel is granted control of the master bus interface and if the CFG7.LOCK_CH bit is asserted, then no other channels are granted control of the master bus interface for the duration specified in CFG7.LOCK_CH_L. Indicates to the master bus interface arbiter that this channel wants exclusive access to the master bus interface for the duration specified in CFG3.LOCK_CH_L. This field does not exist if the configuration parameter DMAH_CH7_LOCK_EN is set to False; in this case, the read-back value is always 0. **Values:** - 0x0 (DISABLED): Channel lock bit is not enabled - 0x1 (ENABLED): Channel lock bit is enabled **Value after reset:** 0x0 **Volatile:** true
14:15	RW	**LOCK_B_L** Bus lock level. Indicates the duration over which CFG7.LOCK_B bit applies. This field does not exist if the parameter DMAH_CH7_LOCK_EN is set to False; in this case, the read-back value is always 0. **Values:** - 0x0 (LOCK_B_L_0): Over complete DMA transfer - 0x1 (LOCK_B_L_1): Over complete DMA block transfer - 0x2 (LOCK_B_L_2): Over complete DMA transaction - 0x3 (LOCK_B_L_3): Over complete DMA transaction **Value after reset:** 0x0 **Volatile:** true

13:12	RW	**LOCK_CH_L** Channel Local Level. Indicates the duration over which CFG7.LOCK_CH applies. This field does not exist if the configuration parameter DMAH_CH7_LOCK_EN is set to False; in this case, the read-back value is always 0. **Values:** - 0x0 (LOCK_CH_L_0): Over complete DMA transfer - 0x1 (LOCK_CH_L_1): Over complete DMA block transfer - 0x2 (LOCK_CH_L_2): Over complete DMA transaction - 0x3 (LOCK_CH_L_3): Over complete DMA transaction **Value after reset:** 0x0 **Volatile:** true
11	RW	**HS_SEL_SRC** Source Software or Hardware Handshaking Select. This register selects which of the handshaking interfaces - hardware or software - is active for source requests on this channel. If the source peripheral is memory, then this bit is ignored. **Values:** - 0x0 (HARDWARE_HS): Hardware handshaking interface. Software initiated transaction requests are ignored. - 0x1 (SOFTWARE_HS): Software handshaking interface. Hardware initiated transaction requests are ignored. **Value after reset:** 0x1 **Volatile:** true
10	RW	**HS_SEL_DST** Destination Software or Hardware Handshaking Select. This register selects which of the handshaking interfaces - hardware or software - is active for destination requests on this channel. If the destination peripheral is memory, then this bit is ignored. **Values:** - 0x0 (HARDWARE_HS): Hardware handshaking interface. Software initiated transaction requests are ignored. - 0x1 (SOFTWARE_HS): Software handshaking interface. Hardware initiated transaction requests are ignored. **Value after reset:** 0x1 **Volatile:** true
9	R	**FIFO_EMPTY** Channel FIFO status. Indicates if there is data left in the channel FIFO. Can be used in conjunction with CFG7.CH_SUSP to cleanly disable a channel. For more information, refer to ”Disabling a Channel Prior to Transfer Completion”. **Values:** - 0x0 (NOT_EMPTY): Channel FIFO is not empty - 0x1 (EMPTY): Channel FIFO is empty **Value after reset:** 0x1 **Volatile:** true

8	R	**CH_SUSP** Channel Suspend. Suspends all DMA data transfers from the source until this bit is cleared. There is no guarantee that the current transaction will complete. Can also be used in conjunction with CFG7.FIFO_EMPTY to cleanly disable a channel without losing any data. For more information, refer to ”Disabling a Channel Prior to Transfer Completion”. **Values:** - 0x0 (NOT_SUSPENDED): DMA transfer from the source is not suspended - 0x1 (SUSPENDED): Suspend DMA transfer from the source **Value after reset:** 0x0 **Volatile:** true
7 :5	RW	**SRC_TR_WIDTH** Source Transfer Width. Mapped to AHB bus hsize. For a non-memory peripheral, typically the peripheral (source) FIFO width. This value must be less than or equl to DMAH_Mk_HDATA_WIDTH, where k is the AHB layer 1 to 4 where the source resides. **Values:** - 0x0 (SRC_TR_WIDTH_0): Source transfer width is 8 bits - 0x1 (SRC_TR_WIDTH_1): Source transfer width is 16 bits - 0x2 (SRC_TR_WIDTH_2): Source transfer width is 32 bits - 0x3 (SRC_TR_WIDTH_3): Source transfer width is 64 bits - 0x4 (SRC_TR_WIDTH_4): Source transfer width is 128 bits - 0x5 (SRC_TR_WIDTH_5): Source transfer width is 256 bits - 0x6 (SRC_TR_WIDTH_6): Source transfer width is 256 bits - 0x7 (SRC_TR_WIDTH_7): Source transfer width is 256 bits **Value after reset:** x **Volatile:** true
4 :0	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0

Offset: 2B0h	**SGR7: Source Gather Register for Channel 7**	**Init = 0**
63:32	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0
31:25	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0
24:19	RW	**SGC** Source Gather Count. Source contiguous tranfer count between successive gather boundaries. **Value after reset:** 0x0 **Volatile:** true

19:0	RW	**SGI** Source Gather Interval. **Value after reset:** 0x0 **Volatile:** true

Offset: 2B8h	**DSR7: Source Gather Register for Channel 7**	**Init = 0**
63:32	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0
31:25	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0
24:19	RW	**DSC** Destination Scatter Count. Destination contiguous transfer count between successive scatter boundaries. **Value after reset:** 0x0 **Volatile:** true
19:0	RW	**DSI** Destination Scatter Interval. **Value after reset:** 0x0 **Volatile:** true

Offset: 2C0h	**RawTfr: Raw Status for IntTfr Interrupt**	**Init = 0**
63:8	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0
7 :0	RW	**RAW** Raw Status for IntTfr Interrupt. **Values:** - 0x0 (INACTIVE): Inactive Raw Interrupt Status - 0x1 (ACTIVE): Active Raw Interrupt Status **Value after reset:** 0x0 **Volatile:** true

Offset: 2C8h	**RawBlock: Raw Status for IntBlock Interrupt**	**Init = 0**
63:8	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0

7 :0	RW	**RAW** Raw Status for IntBlock Interrupt. **Values:** - 0x0 (INACTIVE): Inactive Raw Interrupt Status - 0x1 (ACTIVE): Active Raw Interrupt Status **Value after reset:** 0x0 **Volatile:** true

Offset: 2D0h	**RawSrcTran: Raw Status for IntSrcTran Interrupt**	**Init = 0**
63:8	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0
7 :0	RW	**RAW** Raw Status for IntSrcTran Interrupt. **Values:** - 0x0 (INACTIVE): Inactive Raw Interrupt Status - 0x1 (ACTIVE): Active Raw Interrupt Status **Value after reset:** 0x0 **Volatile:** true

Offset: 2D8h	**RawDstTran: Raw Status for IntDstTran Interrupt**	**Init = 0**
63:8	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0
7 :0	RW	**RAW** Raw Status for IntDstTran Interrupt. **Values:** - 0x0 (INACTIVE): Inactive Raw Interrupt Status - 0x1 (ACTIVE): Active Raw Interrupt Status **Value after reset:** 0x0 **Volatile:** true

Offset: 2E0h	**RawErr: Raw Status for IntErr Interrupt**	**Init = 0**
63:8	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0
7 :0	RW	**RAW** Raw Status for IntErr Interrupt. **Values:** - 0x0 (INACTIVE): Inactive Raw Interrupt Status - 0x1 (ACTIVE): Active Raw Interrupt Status **Value after reset:** 0x0 **Volatile:** true

Offset: 2E8h		StatusTfr: Raw Status for IntErr Interrupt	Init = 0
63:8	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0	
7 :0	RW	**STATUS** Status for IntTfr Interrupt. **Values:** - 0x0 (INACTIVE): Inactive Interrupt Status - 0x1 (ACTIVE): Active Interrupt Status **Value after reset:** 0x0 **Volatile:** true	

Offset: 2F0h		StatusBlock: Status for IntBlock Interrupt	Init = 0
63:8	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0	
7 :0	RW	**STATUS** Status for IntBlock Interrupt. **Values:** - 0x0 (INACTIVE): Inactive Interrupt Status - 0x1 (ACTIVE): Active Interrupt Status **Value after reset:** 0x0 **Volatile:** true	

Offset: 2F8h		StatusSrcTran: Status for IntSrcTran Interrupt	Init = 0
63:8	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0	
7 :0	RW	**STATUS** Status for IntSrcTran Interrupt. **Values:** - 0x0 (INACTIVE): Inactive Interrupt Status - 0x1 (ACTIVE): Active Interrupt Status **Value after reset:** 0x0 **Volatile:** true	

Offset: 300h		StatusDstTran: Status for IntDstTran Interrupt	Init = 0
63:8	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0	

7 :0	RW	**STATUS** Status for IntDstTran Interrupt. **Values:** - 0x0 (INACTIVE): Inactive Interrupt Status - 0x1 (ACTIVE): Active Interrupt Status **Value after reset:** 0x0 **Volatile:** true

Offset: 308h	**StatusErr: Status for IntErr Interrupt**	**Init = 0**
63:8	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0
7 :0	RW	**STATUS** Status for IntErr Interrupt. **Values:** - 0x0 (INACTIVE): Inactive Interrupt Status - 0x1 (ACTIVE): Active Interrupt Status **Value after reset:** 0x0 **Volatile:** true

Offset: 310h	**MaskTfr: Status for IntTfr Interrupt**	**Init = 0**
63:16	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0
15:8	W	**INT_MASK_WE** Interrupt Mask Write Enable. **Values:** - 0x0 (DISABLED): Interrupt mask write disable - 0x1 (ENABLED): Interrupt mask write enable **Value after reset:** 0x0 **Volatile:** true
7 :0	RW	**INT_MASK** Mask for IntTfr Interrupt. **Values:** - 0x0 (MASK): Mask the interrupts - 0x1 (UNMASK): Unmask the interrupts **Value after reset:** 0x0 **Volatile:** true

Offset: 318h	**MaskBlock: Mask for IntBlock Interrupt**	**Init = 0**
63:16	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0

15:8	W	**INT_MASK_WE** Interrupt Mask Write Enable. **Values:** - 0x0 (DISABLED): Interrupt mask write disable - 0x1 (ENABLED): Interrupt mask write enable **Value after reset:** 0x0 **Volatile:** true
7 :0	RW	**INT_MASK** Mask for IntBlock Interrupt. **Values:** - 0x0 (MASK): Mask the interrupts - 0x1 (UNMASK): Unmask the interrupts **Value after reset:** 0x0 **Volatile:** true

Offset: 320h	**MaskSrcTran: Status for IntSrcTran Interrupt**	**Init = 0**
63:16	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0
15:8	W	**INT_MASK_WE** Interrupt Mask Write Enable. **Values:** - 0x0 (DISABLED): Interrupt mask write disable - 0x1 (ENABLED): Interrupt mask write enable **Value after reset:** 0x0 **Volatile:** true
7 :0	RW	**INT_MASK** Mask for IntSrcTran Interrupt. **Values:** - 0x0 (MASK): Mask the interrupts - 0x1 (UNMASK): Unmask the interrupts **Value after reset:** 0x0 **Volatile:** true

Offset: 328h	**MaskDstTran: Mask for IntDstTran Interrupt**	**Init = 0**
63:16	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0
15:8	W	**INT_MASK_WE** Interrupt Mask Write Enable. **Values:** - 0x0 (DISABLED): Interrupt mask write disable - 0x1 (ENABLED): Interrupt mask write enable **Value after reset:** 0x0 **Volatile:** true

7 :0	RW	**INT_MASK** Mask for IntDstTran Interrupt. **Values:** - 0x0 (MASK): Mask the interrupts - 0x1 (UNMASK): Unmask the interrupts **Value after reset:** 0x0 **Volatile:** true

Offset: 330h	**MaskErr: Mask for IntErr Interrupt**	**Init = 0**
63:16	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0
15:8	W	**INT_MASK_WE** Interrupt Mask Write Enable. **Values:** - 0x0 (DISABLED): Interrupt mask write disable - 0x1 (ENABLED): Interrupt mask write enable **Value after reset:** 0x0 **Volatile:** true
7 :0	RW	**INT_MASK** Mask for IntErr Interrupt. **Values:** - 0x0 (MASK): Mask the interrupts - 0x1 (UNMASK): Unmask the interrupts **Value after reset:** 0x0 **Volatile:** true

Offset: 338h	**ClearTfr: Clear for IntTfr Interrupt**	**Init = 0**
63:8	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0
7 :0	RW	**CLEAR** Clear for IntTfr Interrupt. **Values:** - 0x0 (NOT_CLEAR): No effect - 0x1 (CLEAR): Clears interrupts **Value after reset:** 0x0 **Volatile:** true

Offset: 340h	**ClearBlock: Clear for IntBlock Interrupt**	**Init = 0**
63:8	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0

7 :0	RW	**CLEAR** Clear for IntBlock Interrupt. **Values:** - 0x0 (NOT_CLEAR): No effect - 0x1 (CLEAR): Clears interrupts **Value after reset:** 0x0 **Volatile:** true

Offset: 348h	**ClearSrcTran: Clear for IntSrcTran Interrupt**	**Init = 0**
63:8	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0
7 :0	RW	**CLEAR** Clear for IntSrcTran Interrupt. **Values:** - 0x0 (NOT_CLEAR): No effect - 0x1 (CLEAR): Clears interrupts **Value after reset:** 0x0 **Volatile:** true

Offset: 350h	**ClearDstTran: Clear for IntDstTran Interrupt**	**Init = 0**
63:8	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0
7 :0	RW	**CLEAR** Clear for IntDstTran Interrupt. **Values:** - 0x0 (NOT_CLEAR): No effect - 0x1 (CLEAR): Clears interrupts **Value after reset:** 0x0 **Volatile:** true

Offset: 358h	**ClearErr: Clear for IntErr Interrupt**	**Init = 0**
63:8	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0
7 :0	RW	**CLEAR** Clear for IntErr Interrupt. **Values:** - 0x0 (NOT_CLEAR): No effect - 0x1 (CLEAR): Clears interrupts **Value after reset:** 0x0 **Volatile:** true

Offset: 360h		StatusInt: Status for each Interrupt type	Init = 0
63:5	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0	
4	R	**ERR** OR of the contents of StatusErr. **Values:** - 0x0 (INACTIVE): OR of the contents of StatusErr register is 0 - 0x1 (ACTIVE): OR of the contents of StatusErr register is 1 **Value after reset:** 0x0 **Volatile:** true	
3	R	**DSTT** OR of the contents of StatusDstTran. **Values:** - 0x0 (INACTIVE): OR of the contents of StatusDstTran register is 0 - 0x1 (ACTIVE): OR of the contents of StatusDstTran register is 1 **Value after reset:** 0x0 **Volatile:** true	
2	R	**SRCT** OR of the contents of StatusSrcTran. **Values:** - 0x0 (INACTIVE): OR of the contents of StatusSrcTran register is 0 - 0x1 (ACTIVE): OR of the contents of StatusSrcTran register is 1 **Value after reset:** 0x0 **Volatile:** true	
1	R	**BLOCK** OR of the contents of StatusBlock register. **Values:** - 0x0 (INACTIVE): OR of the contents of StatusBlock register is 0 - 0x1 (ACTIVE): OR of the contents of StatusBlock register is 1 **Value after reset:** 0x0 **Volatile:** true	
0	R	**TFR** OR of the contents of StatusTfr register. **Values:** - 0x0 (INACTIVE): OR of the contents of StatusTfr register is 0 - 0x1 (ACTIVE): OR of the contents of StatusTfr register is 1 **Value after reset:** 0x0 **Volatile:** true	

Offset: 368h		ReqSrcReg: Source Software Transaction Request register	Init = 0
63:16	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0	

15:8	W	**SRC_REQ_WE** Source Software Transaction Request write enable. **Values:** - 0x0 (DISABLED): Source request write Disable - 0x1 (ENABLED): Source request write Enable **Value after reset:** 0x0 **Volatile:** true
7 :0	W	**SRC_REQ** Source Software Transaction Request. **Values:** - 0x0 (DISABLED): Source request write is not active - 0x1 (ENABLED): Source request write is active **Value after reset:** 0x0 **Volatile:** true

Offset: 370h	**ReqDstReg: Destination Software Transaction Request register**	**Init = 0**
63:16	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0
15:8	W	**DST_REQ_WE** Destination Software Transaction Request write enable. **Values:** - 0x0 (DISABLED): Destination request write Disable - 0x1 (ENABLED): Destination request write Enable **Value after reset:** 0x0 **Volatile:** true
7 :0	W	**DST_REQ** Destination Software Transaction Request. **Values:** - 0x0 (DISABLED): Destination request write is not active - 0x1 (ENABLED): Destination request write is active **Value after reset:** 0x0 **Volatile:** true

Offset: 378h	**SglRqSrcReg: Source Single Transaction Request register**	**Init = 0**
63:16	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0
15:8	W	**SRC_SGLREQ_WE** Source Single Transaction Request write enable. **Values:** - 0x0 (DISABLED): Single write Disable - 0x1 (ENABLED): Single write Enable **Value after reset:** 0x0 **Volatile:** true

7 :0	W	**SRC_SGLREQ** Source Single Transaction Request. **Values:** - 0x0 (DISABLED): Source request is not active - 0x1 (ENABLED): Source request is active **Value after reset:** 0x0 **Volatile:** true

Offset: 380h	**SglRqDstReg: Destination Single Transaction Request register**	**Init = 0**
63:16	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0
15:8	W	**DST_SGLREQ_WE** Destination Single Transaction Request write enable. **Values:** - 0x0 (DISABLED): Single write Disable - 0x1 (ENABLED): Single write Enable **Value after reset:** 0x0 **Volatile:** true
7 :0	W	**DST_SGLREQ** Destination Single Transaction Request. **Values:** - 0x0 (DISABLED): Destination request is not active - 0x1 (ENABLED): Destination request is active **Value after reset:** 0x0 **Volatile:** true

Offset: 388h	**LstSrcReg: Source Last Transaction Request register**	**Init = 0**
63:16	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0
15:8	W	**LSTSRC_WE** Source Last Transaction Request write enable. **Values:** - 0x0 (DISABLED): Source last transaction request write Disable - 0x1 (ENABLED): Source last transaction request write Enable **Value after reset:** 0x0 **Volatile:** true
7 :0	W	**LSTSRC** Source Last Transaction Request register. **Values:** - 0x0 (NOT_LAST): Not last transaction in current block - 0x1 (LAST): Last transaction in current block **Value after reset:** 0x0 **Volatile:** true

Offset: 390h		LstDstReg: Destination Last Transaction Request register	Init = 0
63:16	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0	
15:8	W	**LSTDST_WE** Destination Last Transaction Request write enable. **Values:** - 0x0 (DISABLED): Destination last transaction request write Disable - 0x1 (ENABLED): Destination last transaction request write Enable **Value after reset:** 0x0 **Volatile:** true	
7 :0	W	**LSTDST** Destination Last Transaction Request register. **Values:** - 0x0 (NOT_LAST): Not last transaction in current block - 0x1 (LAST): Last transaction in current block **Value after reset:** 0x0 **Volatile:** true	

Offset: 398h		DmaCfgReg: HDMA Configuration Register	Init = 0
63:1	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0	
0	RW	**DMA_EN** HDMA Enable bit. **Values:** - 0x0 (DISABLED): HDMA Disabled - 0x1 (ENABLED): HDMA Enabled **Value after reset:** 0x0 **Volatile:** true	

Offset: 3A0h		ChEnReg: HDMA Channel Enable Register	Init = 0
63:16	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0	
15:8	W	**CH_EN_WE** Channel enable register. **Value after reset:** 0x0 **Volatile:** true	

7 :0	RW	**CH_EN** Channel Enable. The ChEnReg.CH_EN bit is automatically cleared by hardware to disable the channel after the last AMBA transfer of the DMA transfer to the destination has completed. Software can therefore poll this bit to determine when this channel is free for a new DMA transfer. **Values:** - 0x0 (DISABLED): Disable the channel - 0x1 (ENABLED): Enable the channel **Value after reset:** 0x0 **Volatile:** true

Offset: 3A8h	**DmaIdReg: HDMA ID register**	**Init = 0**
63:32	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0
31:0	R	**DMA_ID** Hardcoded HDMA peripheral ID. **Value after reset:** 0x0 **Volatile:** true

Offset: 3B0h	**DmaIdReg: HDMA Test registers**	**Init = 0**
63:1	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0
0	RW	**TEST_SLV_IF** DMA Test register. **Values:** - 0x0 (NORMAL_MODE): Puts the AHB slave interface into Normal mode - 0x1 (TEST_MODE): Puts the AHB slave interface into Test mode. In this mode, the readback value of the writable registers always matches the values written **Value after reset:** 0x0 **Volatile:** true

Offset: 3B8h	**DmaLpTimeoutReg: HDMA Low Power Timeout Register**	**Init = 0**
63:4	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0
3 :0	RW	**TEST_SLV_IF** This field holds timeout value of low power counter register. **Value after reset:** 0x8

Offset: 3C8h		DMA_COMP_PARAMS_6: HDMA Component Parameters Register 6	Init = 0
63	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0	
62:60	R	**CH7_FIFO_DEPTH** **Values:** - 0x0 (FIFO_DEPTH_8): Channel 7 FIFO depth is 8 bytes - 0x1 (FIFO_DEPTH_16): Channel 7 FIFO depth is 16 bytes - 0x2 (FIFO_DEPTH_32): Channel 7 FIFO depth is 32 bytes - 0x3 (FIFO_DEPTH_64): Channel 7 FIFO depth is 64 bytes - 0x4 (FIFO_DEPTH_128): Channel 7 FIFO depth is 128 bytes - 0x5 (FIFO_DEPTH_256): Channel 7 FIFO depth is 256 bytes **Value after reset:** 0x1	
59:57	R	**CH7_SMS** **Values:** - 0x0 (MASTER_1): Hardcode the AHB master 1 interface attached to the source of channel 7 - 0x1 (MASTER_2): Hardcode the AHB master 2 interface attached to the source of channel 7 - 0x2 (MASTER_3): Hardcode the AHB master 3 interface attached to the source of channel 7 - 0x3 (MASTER_4): Hardcode the AHB master 4 interface attached to the source of channel 7 - 0x4 (PROGRAMMABLE): Programmable **Value after reset:** 0x0	
56:54	R	**CH7_LMS** **Values:** - 0x0 (MASTER_1): Hardcode the AHB master 1 interface attached to the LLP peripherals of channel 7 - 0x1 (MASTER_2): Hardcode the AHB master 2 interface attached to the LLP peripherals of channel 7 - 0x2 (MASTER_3): Hardcode the AHB master 3 interface attached to the LLP peripherals of channel 7 - 0x3 (MASTER_4): Hardcode the AHB master 4 interface attached to the LLP peripherals of channel 7 - 0x4 (PROGRAMMABLE): Programmable **Value after reset:** 0x0	
53:51	R	**CH7_DMS** **Values:** - 0x0 (MASTER_1): Hardcode the AHB master 1 interface attached to the channel 7 destination - 0x1 (MASTER_2): Hardcode the AHB master 2 interface attached to the channel 7 destination - 0x2 (MASTER_3): Hardcode the AHB master 3 interface attached to the channel 7 destination - 0x3 (MASTER_4): Hardcode the AHB master 4 interface attached to the channel 7 destination - 0x4 (PROGRAMMABLE): Programmable **Value after reset:** 0x0	

50:48	R	**CH7_MAX_MULT_SIZE** **Values:** - 0x0 (MAX_MULT_SIZE_4): Maximum value of burst transaction size that can be programmed for channel 7 is 4 - 0x1 (MAX_MULT_SIZE_8): Maximum value of burst transaction size that can be programmed for channel 7 is 8 - 0x2 (MAX_MULT_SIZE_16): Maximum value of burst transaction size that can be programmed for channel 7 is 16 - 0x3 (MAX_MULT_SIZE_32): Maximum value of burst transaction size that can be programmed for channel 7 is 32 - 0x4 (MAX_MULT_SIZE_64): Maximum value of burst transaction size that can be programmed for channel 7 is 64 - 0x5 (MAX_MULT_SIZE_128): Maximum value of burst transaction size that can be programmed for channel 7 is 128 - 0x6 (MAX_MULT_SIZE_256): Maximum value of burst transaction size that can be programmed for channel 7 is 256 - 0x7 (RESERVED): Reserved **Value after reset:** 0x1
47:46	R	**CH7_FC** **Values:** - 0x0 (F_DMA): Flow controller is DMA for channel 7 - 0x1 (F_SRC): Flow controller is Source for channel 7 - 0x2 (F_DST): Flow controller is Destination for channel 7 - 0x3 (F_ANY): Flow controller is ANY for channel 7 **Value after reset:** 0x0
45	R	**CH7_HC_LLP** **Values:** - 0x0 (PROGRAMMABLE): Exclude logic to hardcode Channel 7 LLP register to 0 - 0x0 (HARDCODED): Hardcode Channel 7 LLP register to 0 **Value after reset:** 0x0
44	R	**CH7_CTL_WB_EN** **Values:** - 0x0 (FALSE): Exclude logic to enable control register writeback after each block transfer on channel 7 - 0x1 (TRUE): Include logic to enable control register writeback after each block transfer on channel 7 **Value after reset:** 0x0
43	R	**CH7_MULTI_BLK_EN** **Values:** - 0x0 (FALSE): Exclude logic to enable channel multi-block DMA transfers on channel 7 - 0x1 (TRUE): Include logic to enable channel multi-block DMA transfers on channel 7 **Value after reset:** 0x0
42	R	**CH7_LOCK_EN** **Values:** - 0x0 (FALSE): Exclude logic to enable channel or bus locking on channel 7 - 0x1 (TRUE): Include logic to enable channel or bus locking on channel 7 **Value after reset:** 0x0

41	R	**CH7_SRC_GAT_EN** **Values:** - 0x0 (FALSE): Exclude logic to enable the gather feature on channel 7 - 0x1 (TRUE): Include logic to enable the gather feature on channel 7 **Value after reset:** 0x0
40	R	**CH7_DST_GAT_EN** **Values:** - 0x0 (FALSE): Exclude logic to enable the gather feature on channel 7 - 0x1 (TRUE): Include logic to enable the gather feature on channel 7 **Value after reset:** 0x0
39	R	**CH7_STAT_SRC** **Values:** - 0x0 (FALSE): Exclude logic to fetch a status register from source peripheral of channel 7 and write this status information to memory at end of each block transfer - 0x1 (TRUE): Include logic to fetch a status register from source peripheral of channel 7 and write this status information to memory at end of each block transfer **Value after reset:** 0x0
38	R	**CH7_STAT_DST** **Values:** - 0x0 (FALSE): Exclude logic to fetch a status register from destination peripheral of channel 7 and write this status information to memory at end of each block transfer - 0x1 (TRUE): Include logic to fetch a status register from destination peripheral of channel 7 and write this status information to memory at end of each block transfer **Value after reset:** 0x0
37:35	R	**CH7_STW** **Values:** - 0x0 (NO_HARDCODE): No hardcode - 0x1 (STW_8): Hardcode the channel 7's source transfer width to 8 bits - 0x2 (STW_16): Hardcode the channel 7's source transfer width to 16 bits - 0x3 (STW_32): Hardcode the channel 7's source transfer width to 32 bits - 0x4 (STW_64): Hardcode the channel 7's source transfer width to 64 bits - 0x5 (STW_128): Hardcode the channel 7's source transfer width to 128 bits - 0x6 (STW_256): Hardcode the channel 7's source transfer width to 256 bits - 0x7 (RESERVED): Reserved **Value after reset:** 0x3
34:32	R	**CH7_DTW** **Values:** - 0x0 (NO_HARDCODE): No hardcode - 0x1 (STW_8): Hardcode the channel 7's destination transfer width to 8 bits - 0x2 (STW_16): Hardcode the channel 7's destination transfer width to 16 bits - 0x3 (STW_32): Hardcode the channel 7's destination transfer width to 32 bits - 0x4 (STW_64): Hardcode the channel 7's destination transfer width to 64 bits - 0x5 (STW_128): Hardcode the channel 7's destination transfer width to 128 bits - 0x6 (STW_256): Hardcode the channel 7's destination transfer width to 256 bits - 0x7 (RESERVED): Reserved **Value after reset:** 0x3

31:0	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0

Offset: 3D0h	**DMA_COMP_PARAMS_5: HDMA Component Parameters Register 5**	**Init = 0**
63	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0
62:60	R	**CH5_FIFO_DEPTH** **Values:** - 0x0 (FIFO_DEPTH_8): Channel 5 FIFO depth is 8 bytes - 0x1 (FIFO_DEPTH_16): Channel 5 FIFO depth is 16 bytes - 0x2 (FIFO_DEPTH_32): Channel 5 FIFO depth is 32 bytes - 0x3 (FIFO_DEPTH_64): Channel 5 FIFO depth is 64 bytes - 0x4 (FIFO_DEPTH_128): Channel 5 FIFO depth is 128 bytes - 0x5 (FIFO_DEPTH_256): Channel 5 FIFO depth is 256 bytes **Value after reset:** 0x1
59:57	R	**CH5_SMS** **Values:** - 0x0 (MASTER_1): Hardcode the AHB master 1 interface attached to the source of channel 5 - 0x1 (MASTER_2): Hardcode the AHB master 2 interface attached to the source of channel 5 - 0x2 (MASTER_3): Hardcode the AHB master 3 interface attached to the source of channel 5 - 0x3 (MASTER_4): Hardcode the AHB master 4 interface attached to the source of channel 5 - 0x4 (PROGRAMMABLE): Programmable **Value after reset:** 0x0
56:54	R	**CH5_LMS** **Values:** - 0x0 (MASTER_1): Hardcode the AHB master 1 interface attached to the LLP peripherals of channel 5 - 0x1 (MASTER_2): Hardcode the AHB master 2 interface attached to the LLP peripherals of channel 5 - 0x2 (MASTER_3): Hardcode the AHB master 3 interface attached to the LLP peripherals of channel 5 - 0x3 (MASTER_4): Hardcode the AHB master 4 interface attached to the LLP peripherals of channel 5 - 0x4 (PROGRAMMABLE): Programmable **Value after reset:** 0x0

53:51	R	**CH5_DMS** **Values:** - 0x0 (MASTER_1): Hardcode the AHB master 1 interface attached to the channel 5 destination - 0x1 (MASTER_2): Hardcode the AHB master 2 interface attached to the channel 5 destination - 0x2 (MASTER_3): Hardcode the AHB master 3 interface attached to the channel 5 destination - 0x3 (MASTER_4): Hardcode the AHB master 4 interface attached to the channel 5 destination - 0x4 (PROGRAMMABLE): Programmable **Value after reset:** 0x0
50:48	R	**CH5_MAX_MULT_SIZE** **Values:** - 0x0 (MAX_MULT_SIZE_4): Maximum value of burst transaction size that can be programmed for channel 5 is 4 - 0x1 (MAX_MULT_SIZE_8): Maximum value of burst transaction size that can be programmed for channel 5 is 8 - 0x2 (MAX_MULT_SIZE_16): Maximum value of burst transaction size that can be programmed for channel 5 is 16 - 0x3 (MAX_MULT_SIZE_32): Maximum value of burst transaction size that can be programmed for channel 5 is 32 - 0x4 (MAX_MULT_SIZE_64): Maximum value of burst transaction size that can be programmed for channel 5 is 64 - 0x5 (MAX_MULT_SIZE_128): Maximum value of burst transaction size that can be programmed for channel 5 is 128 - 0x6 (MAX_MULT_SIZE_256): Maximum value of burst transaction size that can be programmed for channel 5 is 256 - 0x7 (RESERVED): Reserved **Value after reset:** 0x1
47:46	R	**CH5_FC** **Values:** - 0x0 (F_DMA): Flow controller is DMA for channel 5 - 0x1 (F_SRC): Flow controller is Source for channel 5 - 0x2 (F_DST): Flow controller is Destination for channel 5 - 0x3 (F_ANY): Flow controller is ANY for channel 5 **Value after reset:** 0x0
45	R	**CH5_HC_LLP** **Values:** - 0x0 (PROGRAMMABLE): Exclude logic to hardcode Channel 5 LLP register to 0 - 0x0 (HARDCODED): Hardcode Channel 5 LLP register to 0 **Value after reset:** 0x0
44	R	**CH5_CTL_WB_EN** **Values:** - 0x0 (FALSE): Exclude logic to enable control register writeback after each block transfer on channel 5 - 0x1 (TRUE): Include logic to enable control register writeback after each block transfer on channel 5 **Value after reset:** 0x0

43	R	**CH5_MULTI_BLK_EN** **Values:** - 0x0 (FALSE): Exclude logic to enable channel multi-block DMA transfers on channel 5 - 0x1 (TRUE): Include logic to enable channel multi-block DMA transfers on channel 5 **Value after reset:** 0x0
42	R	**CH5_LOCK_EN** **Values:** - 0x0 (FALSE): Exclude logic to enable channel or bus locking on channel 5 - 0x1 (TRUE): Include logic to enable channel or bus locking on channel 5 **Value after reset:** 0x0
41	R	**CH5_SRC_GAT_EN** **Values:** - 0x0 (FALSE): Exclude logic to enable the gather feature on channel 5 - 0x1 (TRUE): Include logic to enable the gather feature on channel 5 **Value after reset:** 0x0
40	R	**CH5_DST_GAT_EN** **Values:** - 0x0 (FALSE): Exclude logic to enable the gather feature on channel 5 - 0x1 (TRUE): Include logic to enable the gather feature on channel 5 **Value after reset:** 0x0
39	R	**CH5_STAT_SRC** **Values:** - 0x0 (FALSE): Exclude logic to fetch a status register from source peripheral of channel 5 and write this status information to memory at end of each block transfer - 0x1 (TRUE): Include logic to fetch a status register from source peripheral of channel 5 and write this status information to memory at end of each block transfer **Value after reset:** 0x0
38	R	**CH5_STAT_DST** **Values:** - 0x0 (FALSE): Exclude logic to fetch a status register from destination peripheral of channel 5 and write this status information to memory at end of each block transfer - 0x1 (TRUE): Include logic to fetch a status register from destination peripheral of channel 5 and write this status information to memory at end of each block transfer **Value after reset:** 0x0
37:35	R	**CH5_STW** **Values:** - 0x0 (NO_HARDCODE): No hardcode - 0x1 (STW_8): Hardcode the channel 5's source transfer width to 8 bits - 0x2 (STW_16): Hardcode the channel 5's source transfer width to 16 bits - 0x3 (STW_32): Hardcode the channel 5's source transfer width to 32 bits - 0x4 (STW_64): Hardcode the channel 5's source transfer width to 64 bits - 0x5 (STW_128): Hardcode the channel 5's source transfer width to 128 bits - 0x6 (STW_256): Hardcode the channel 5's source transfer width to 256 bits - 0x7 (RESERVED): Reserved **Value after reset:** 0x3

34:32	R	**CH5_DTW** **Values:** - 0x0 (NO_HARDCODE): No hardcode - 0x1 (STW_8): Hardcode the channel 5's destination transfer width to 8 bits - 0x2 (STW_16): Hardcode the channel 5's destination transfer width to 16 bits - 0x3 (STW_32): Hardcode the channel 5's destination transfer width to 32 bits - 0x4 (STW_64): Hardcode the channel 5's destination transfer width to 64 bits - 0x5 (STW_128): Hardcode the channel 5's destination transfer width to 128 bits - 0x6 (STW_256): Hardcode the channel 5's destination transfer width to 256 bits - 0x7 (RESERVED): Reserved **Value after reset:** 0x3
31	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0
30:28	R	**CH6_FIFO_DEPTH** **Values:** - 0x0 (FIFO_DEPTH_8): Channel 6 FIFO depth is 8 bytes - 0x1 (FIFO_DEPTH_16): Channel 6 FIFO depth is 16 bytes - 0x2 (FIFO_DEPTH_32): Channel 6 FIFO depth is 32 bytes - 0x3 (FIFO_DEPTH_64): Channel 6 FIFO depth is 64 bytes - 0x4 (FIFO_DEPTH_128): Channel 6 FIFO depth is 128 bytes - 0x5 (FIFO_DEPTH_256): Channel 6 FIFO depth is 256 bytes **Value after reset:** 0x1
27:25	R	**CH6_SMS** **Values:** - 0x0 (MASTER_1): Hardcode the AHB master 1 interface attached to the source of channel 6 - 0x1 (MASTER_2): Hardcode the AHB master 2 interface attached to the source of channel 6 - 0x2 (MASTER_3): Hardcode the AHB master 3 interface attached to the source of channel 6 - 0x3 (MASTER_4): Hardcode the AHB master 4 interface attached to the source of channel 6 - 0x4 (PROGRAMMABLE): Programmable **Value after reset:** 0x0
24:22	R	**CH6_LMS** **Values:** - 0x0 (MASTER_1): Hardcode the AHB master 1 interface attached to the LLP peripherals of channel 6 - 0x1 (MASTER_2): Hardcode the AHB master 2 interface attached to the LLP peripherals of channel 6 - 0x2 (MASTER_3): Hardcode the AHB master 3 interface attached to the LLP peripherals of channel 6 - 0x3 (MASTER_4): Hardcode the AHB master 4 interface attached to the LLP peripherals of channel 6 - 0x4 (PROGRAMMABLE): Programmable **Value after reset:** 0x0

21:19	R	**CH6_DMS** **Values:** - 0x0 (MASTER_1): Hardcode the AHB master 1 interface attached to the channel 6 destination - 0x1 (MASTER_2): Hardcode the AHB master 2 interface attached to the channel 6 destination - 0x2 (MASTER_3): Hardcode the AHB master 3 interface attached to the channel 6 destination - 0x3 (MASTER_4): Hardcode the AHB master 4 interface attached to the channel 6 destination - 0x4 (PROGRAMMABLE): Programmable **Value after reset:** 0x0
18:16	R	**CH6_MAX_MULT_SIZE** **Values:** - 0x0 (MAX_MULT_SIZE_4): Maximum value of burst transaction size that can be programmed for channel 6 is 4 - 0x1 (MAX_MULT_SIZE_8): Maximum value of burst transaction size that can be programmed for channel 6 is 8 - 0x2 (MAX_MULT_SIZE_16): Maximum value of burst transaction size that can be programmed for channel 6 is 16 - 0x3 (MAX_MULT_SIZE_32): Maximum value of burst transaction size that can be programmed for channel 6 is 32 - 0x4 (MAX_MULT_SIZE_64): Maximum value of burst transaction size that can be programmed for channel 6 is 64 - 0x5 (MAX_MULT_SIZE_128): Maximum value of burst transaction size that can be programmed for channel 6 is 128 - 0x6 (MAX_MULT_SIZE_256): Maximum value of burst transaction size that can be programmed for channel 6 is 256 - 0x7 (RESERVED): Reserved **Value after reset:** 0x1
15:14	R	**CH6_FC** **Values:** - 0x0 (F_DMA): Flow controller is DMA for channel 6 - 0x1 (F_SRC): Flow controller is Source for channel 6 - 0x2 (F_DST): Flow controller is Destination for channel 6 - 0x3 (F_ANY): Flow controller is ANY for channel 6 **Value after reset:** 0x0
13	R	**CH6_HC_LLP** **Values:** - 0x0 (PROGRAMMABLE): Exclude logic to hardcode Channel 6 LLP register to 0 - 0x0 (HARDCODED): Hardcode Channel 6 LLP register to 0 **Value after reset:** 0x0
12	R	**CH6_CTL_WB_EN** **Values:** - 0x0 (FALSE): Exclude logic to enable control register writeback after each block transfer on channel 6 - 0x1 (TRUE): Include logic to enable control register writeback after each block transfer on channel 6 **Value after reset:** 0x0

11	R	**CH6_MULTI_BLK_EN** **Values:** - 0x0 (FALSE): Exclude logic to enable channel multi-block DMA transfers on channel 6 - 0x1 (TRUE): Include logic to enable channel multi-block DMA transfers on channel 6 **Value after reset:** 0x0
10	R	**CH6_LOCK_EN** **Values:** - 0x0 (FALSE): Exclude logic to enable channel or bus locking on channel 6 - 0x1 (TRUE): Include logic to enable channel or bus locking on channel 6 **Value after reset:** 0x0
9	R	**CH6_SRC_GAT_EN** **Values:** - 0x0 (FALSE): Exclude logic to enable the gather feature on channel 6 - 0x1 (TRUE): Include logic to enable the gather feature on channel 6 **Value after reset:** 0x0
8	R	**CH6_DST_GAT_EN** **Values:** - 0x0 (FALSE): Exclude logic to enable the gather feature on channel 6 - 0x1 (TRUE): Include logic to enable the gather feature on channel 6 **Value after reset:** 0x0
7	R	**CH6_STAT_SRC** **Values:** - 0x0 (FALSE): Exclude logic to fetch a status register from source peripheral of channel 6 and write this status information to memory at end of each block transfer - 0x1 (TRUE): Include logic to fetch a status register from source peripheral of channel 6 and write this status information to memory at end of each block transfer **Value after reset:** 0x0
6	R	**CH6_STAT_DST** **Values:** - 0x0 (FALSE): Exclude logic to fetch a status register from destination peripheral of channel 6 and write this status information to memory at end of each block transfer - 0x1 (TRUE): Include logic to fetch a status register from destination peripheral of channel 6 and write this status information to memory at end of each block transfer **Value after reset:** 0x0
5 :3	R	**CH6_STW** **Values:** - 0x0 (NO_HARDCODE): No hardcode - 0x1 (STW_8): Hardcode the channel 6's source transfer width to 8 bits - 0x2 (STW_16): Hardcode the channel 6's source transfer width to 16 bits - 0x3 (STW_32): Hardcode the channel 6's source transfer width to 32 bits - 0x4 (STW_64): Hardcode the channel 6's source transfer width to 64 bits - 0x5 (STW_128): Hardcode the channel 6's source transfer width to 128 bits - 0x6 (STW_256): Hardcode the channel 6's source transfer width to 256 bits - 0x7 (RESERVED): Reserved **Value after reset:** 0x3

2 :0	R	**CH6_DTW** **Values:** - 0x0 (NO_HARDCODE): No hardcode - 0x1 (STW_8): Hardcode the channel 6's destination transfer width to 8 bits - 0x2 (STW_16): Hardcode the channel 6's destination transfer width to 16 bits - 0x3 (STW_32): Hardcode the channel 6's destination transfer width to 32 bits - 0x4 (STW_64): Hardcode the channel 6's destination transfer width to 64 bits - 0x5 (STW_128): Hardcode the channel 6's destination transfer width to 128 bits - 0x6 (STW_256): Hardcode the channel 6's destination transfer width to 256 bits - 0x7 (RESERVED): Reserved **Value after reset:** 0x3

Offset: 3D8h	**DMA_COMP_PARAMS_4: HDMA Component Parameters Register 4**	**Init = 0**
63	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0
62:60	R	**CH3_FIFO_DEPTH** **Values:** - 0x0 (FIFO_DEPTH_8): Channel 3 FIFO depth is 8 bytes - 0x1 (FIFO_DEPTH_16): Channel 3 FIFO depth is 16 bytes - 0x2 (FIFO_DEPTH_32): Channel 3 FIFO depth is 32 bytes - 0x3 (FIFO_DEPTH_64): Channel 3 FIFO depth is 64 bytes - 0x4 (FIFO_DEPTH_128): Channel 3 FIFO depth is 128 bytes - 0x5 (FIFO_DEPTH_256): Channel 3 FIFO depth is 256 bytes **Value after reset:** 0x1
59:57	R	**CH3_SMS** **Values:** - 0x0 (MASTER_1): Hardcode the AHB master 1 interface attached to the source of channel 3 - 0x1 (MASTER_2): Hardcode the AHB master 2 interface attached to the source of channel 3 - 0x2 (MASTER_3): Hardcode the AHB master 3 interface attached to the source of channel 3 - 0x3 (MASTER_4): Hardcode the AHB master 4 interface attached to the source of channel 3 - 0x4 (PROGRAMMABLE): Programmable **Value after reset:** 0x0
56:54	R	**CH3_LMS** **Values:** - 0x0 (MASTER_1): Hardcode the AHB master 1 interface attached to the LLP peripherals of channel 3 - 0x1 (MASTER_2): Hardcode the AHB master 2 interface attached to the LLP peripherals of channel 3 - 0x2 (MASTER_3): Hardcode the AHB master 3 interface attached to the LLP peripherals of channel 3 - 0x3 (MASTER_4): Hardcode the AHB master 4 interface attached to the LLP peripherals of channel 3 - 0x4 (PROGRAMMABLE): Programmable **Value after reset:** 0x0

53:51	R	**CH3_DMS** **Values:** - 0x0 (MASTER_1): Hardcode the AHB master 1 interface attached to the channel 3 destination - 0x1 (MASTER_2): Hardcode the AHB master 2 interface attached to the channel 3 destination - 0x2 (MASTER_3): Hardcode the AHB master 3 interface attached to the channel 3 destination - 0x3 (MASTER_4): Hardcode the AHB master 4 interface attached to the channel 3 destination - 0x4 (PROGRAMMABLE): Programmable **Value after reset:** 0x0
50:48	R	**CH3_MAX_MULT_SIZE** **Values:** - 0x0 (MAX_MULT_SIZE_4): Maximum value of burst transaction size that can be programmed for channel 3 is 4 - 0x1 (MAX_MULT_SIZE_8): Maximum value of burst transaction size that can be programmed for channel 3 is 8 - 0x2 (MAX_MULT_SIZE_16): Maximum value of burst transaction size that can be programmed for channel 3 is 16 - 0x3 (MAX_MULT_SIZE_32): Maximum value of burst transaction size that can be programmed for channel 3 is 32 - 0x4 (MAX_MULT_SIZE_64): Maximum value of burst transaction size that can be programmed for channel 3 is 64 - 0x5 (MAX_MULT_SIZE_128): Maximum value of burst transaction size that can be programmed for channel 3 is 128 - 0x6 (MAX_MULT_SIZE_256): Maximum value of burst transaction size that can be programmed for channel 3 is 256 - 0x7 (RESERVED): Reserved **Value after reset:** 0x1
47:46	R	**CH3_FC** **Values:** - 0x0 (F_DMA): Flow controller is DMA for channel 3 - 0x1 (F_SRC): Flow controller is Source for channel 3 - 0x2 (F_DST): Flow controller is Destination for channel 3 - 0x3 (F_ANY): Flow controller is ANY for channel 3 **Value after reset:** 0x0
45	R	**CH3_HC_LLP** **Values:** - 0x0 (PROGRAMMABLE): Exclude logic to hardcode Channel 3 LLP register to 0 - 0x0 (HARDCODED): Hardcode Channel 3 LLP register to 0 **Value after reset:** 0x0
44	R	**CH3_CTL_WB_EN** **Values:** - 0x0 (FALSE): Exclude logic to enable control register writeback after each block transfer on channel 3 - 0x1 (TRUE): Include logic to enable control register writeback after each block transfer on channel 3 **Value after reset:** 0x0

43	R	**CH3_MULTI_BLK_EN** **Values:** - 0x0 (FALSE): Exclude logic to enable channel multi-block DMA transfers on channel 3 - 0x1 (TRUE): Include logic to enable channel multi-block DMA transfers on channel 3 **Value after reset:** 0x0
42	R	**CH3_LOCK_EN** **Values:** - 0x0 (FALSE): Exclude logic to enable channel or bus locking on channel 3 - 0x1 (TRUE): Include logic to enable channel or bus locking on channel 3 **Value after reset:** 0x0
41	R	**CH3_SRC_GAT_EN** **Values:** - 0x0 (FALSE): Exclude logic to enable the gather feature on channel 3 - 0x1 (TRUE): Include logic to enable the gather feature on channel 3 **Value after reset:** 0x0
40	R	**CH3_DST_GAT_EN** **Values:** - 0x0 (FALSE): Exclude logic to enable the gather feature on channel 3 - 0x1 (TRUE): Include logic to enable the gather feature on channel 3 **Value after reset:** 0x0
39	R	**CH3_STAT_SRC** **Values:** - 0x0 (FALSE): Exclude logic to fetch a status register from source peripheral of channel 3 and write this status information to memory at end of each block transfer - 0x1 (TRUE): Include logic to fetch a status register from source peripheral of channel 3 and write this status information to memory at end of each block transfer **Value after reset:** 0x0
38	R	**CH3_STAT_DST** **Values:** - 0x0 (FALSE): Exclude logic to fetch a status register from destination peripheral of channel 3 and write this status information to memory at end of each block transfer - 0x1 (TRUE): Include logic to fetch a status register from destination peripheral of channel 3 and write this status information to memory at end of each block transfer **Value after reset:** 0x0
37:35	R	**CH3_STW** **Values:** - 0x0 (NO_HARDCODE): No hardcode - 0x1 (STW_8): Hardcode the channel 3's source transfer width to 8 bits - 0x2 (STW_16): Hardcode the channel 3's source transfer width to 16 bits - 0x3 (STW_32): Hardcode the channel 3's source transfer width to 32 bits - 0x4 (STW_64): Hardcode the channel 3's source transfer width to 64 bits - 0x5 (STW_128): Hardcode the channel 3's source transfer width to 128 bits - 0x6 (STW_256): Hardcode the channel 3's source transfer width to 256 bits - 0x7 (RESERVED): Reserved **Value after reset:** 0x3

34:32	R	**CH3_DTW** **Values:** - 0x0 (NO_HARDCODE): No hardcode - 0x1 (STW_8): Hardcode the channel 3's destination transfer width to 8 bits - 0x2 (STW_16): Hardcode the channel 3's destination transfer width to 16 bits - 0x3 (STW_32): Hardcode the channel 3's destination transfer width to 32 bits - 0x4 (STW_64): Hardcode the channel 3's destination transfer width to 64 bits - 0x5 (STW_128): Hardcode the channel 3's destination transfer width to 128 bits - 0x6 (STW_256): Hardcode the channel 3's destination transfer width to 256 bits - 0x7 (RESERVED): Reserved **Value after reset:** 0x3
31	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0
30:28	R	**CH4_FIFO_DEPTH** **Values:** - 0x0 (FIFO_DEPTH_8): Channel 4 FIFO depth is 8 bytes - 0x1 (FIFO_DEPTH_16): Channel 4 FIFO depth is 16 bytes - 0x2 (FIFO_DEPTH_32): Channel 4 FIFO depth is 32 bytes - 0x3 (FIFO_DEPTH_64): Channel 4 FIFO depth is 64 bytes - 0x4 (FIFO_DEPTH_128): Channel 4 FIFO depth is 128 bytes - 0x5 (FIFO_DEPTH_256): Channel 4 FIFO depth is 256 bytes **Value after reset:** 0x1
27:25	R	**CH4_SMS** **Values:** - 0x0 (MASTER_1): Hardcode the AHB master 1 interface attached to the source of channel 4 - 0x1 (MASTER_2): Hardcode the AHB master 2 interface attached to the source of channel 4 - 0x2 (MASTER_3): Hardcode the AHB master 3 interface attached to the source of channel 4 - 0x3 (MASTER_4): Hardcode the AHB master 4 interface attached to the source of channel 4 - 0x4 (PROGRAMMABLE): Programmable **Value after reset:** 0x0
24:22	R	**CH4_LMS** **Values:** - 0x0 (MASTER_1): Hardcode the AHB master 1 interface attached to the LLP peripherals of channel 4 - 0x1 (MASTER_2): Hardcode the AHB master 2 interface attached to the LLP peripherals of channel 4 - 0x2 (MASTER_3): Hardcode the AHB master 3 interface attached to the LLP peripherals of channel 4 - 0x3 (MASTER_4): Hardcode the AHB master 4 interface attached to the LLP peripherals of channel 4 - 0x4 (PROGRAMMABLE): Programmable **Value after reset:** 0x0

21:19	R	**CH4_DMS** **Values:** - 0x0 (MASTER_1): Hardcode the AHB master 1 interface attached to the channel 4 destination - 0x1 (MASTER_2): Hardcode the AHB master 2 interface attached to the channel 4 destination - 0x2 (MASTER_3): Hardcode the AHB master 3 interface attached to the channel 4 destination - 0x3 (MASTER_4): Hardcode the AHB master 4 interface attached to the channel 4 destination - 0x4 (PROGRAMMABLE): Programmable **Value after reset:** 0x0
18:16	R	**CH4_MAX_MULT_SIZE** **Values:** - 0x0 (MAX_MULT_SIZE_4): Maximum value of burst transaction size that can be programmed for channel 4 is 4 - 0x1 (MAX_MULT_SIZE_8): Maximum value of burst transaction size that can be programmed for channel 4 is 8 - 0x2 (MAX_MULT_SIZE_16): Maximum value of burst transaction size that can be programmed for channel 4 is 16 - 0x3 (MAX_MULT_SIZE_32): Maximum value of burst transaction size that can be programmed for channel 4 is 32 - 0x4 (MAX_MULT_SIZE_64): Maximum value of burst transaction size that can be programmed for channel 4 is 64 - 0x5 (MAX_MULT_SIZE_128): Maximum value of burst transaction size that can be programmed for channel 4 is 128 - 0x6 (MAX_MULT_SIZE_256): Maximum value of burst transaction size that can be programmed for channel 4 is 256 - 0x7 (RESERVED): Reserved **Value after reset:** 0x1
15:14	R	**CH4_FC** **Values:** - 0x0 (F_DMA): Flow controller is DMA for channel 4 - 0x1 (F_SRC): Flow controller is Source for channel 4 - 0x2 (F_DST): Flow controller is Destination for channel 4 - 0x3 (F_ANY): Flow controller is ANY for channel 4 **Value after reset:** 0x0
13	R	**CH4_HC_LLP** **Values:** - 0x0 (PROGRAMMABLE): Exclude logic to hardcode Channel 4 LLP register to 0 - 0x0 (HARDCODED): Hardcode Channel 4 LLP register to 0 **Value after reset:** 0x0
12	R	**CH4_CTL_WB_EN** **Values:** - 0x0 (FALSE): Exclude logic to enable control register writeback after each block transfer on channel 4 - 0x1 (TRUE): Include logic to enable control register writeback after each block transfer on channel 4 **Value after reset:** 0x0

11	R	**CH4_MULTI_BLK_EN** **Values:** - 0x0 (FALSE): Exclude logic to enable channel multi-block DMA transfers on channel 4 - 0x1 (TRUE): Include logic to enable channel multi-block DMA transfers on channel 4 **Value after reset:** 0x0
10	R	**CH4_LOCK_EN** **Values:** - 0x0 (FALSE): Exclude logic to enable channel or bus locking on channel 4 - 0x1 (TRUE): Include logic to enable channel or bus locking on channel 4 **Value after reset:** 0x0
9	R	**CH4_SRC_GAT_EN** **Values:** - 0x0 (FALSE): Exclude logic to enable the gather feature on channel 4 - 0x1 (TRUE): Include logic to enable the gather feature on channel 4 **Value after reset:** 0x0
8	R	**CH4_DST_GAT_EN** **Values:** - 0x0 (FALSE): Exclude logic to enable the gather feature on channel 4 - 0x1 (TRUE): Include logic to enable the gather feature on channel 4 **Value after reset:** 0x0
7	R	**CH4_STAT_SRC** **Values:** - 0x0 (FALSE): Exclude logic to fetch a status register from source peripheral of channel 4 and write this status information to memory at end of each block transfer - 0x1 (TRUE): Include logic to fetch a status register from source peripheral of channel 4 and write this status information to memory at end of each block transfer **Value after reset:** 0x0
6	R	**CH4_STAT_DST** **Values:** - 0x0 (FALSE): Exclude logic to fetch a status register from destination peripheral of channel 4 and write this status information to memory at end of each block transfer - 0x1 (TRUE): Include logic to fetch a status register from destination peripheral of channel 4 and write this status information to memory at end of each block transfer **Value after reset:** 0x0
5 :3	R	**CH4_STW** **Values:** - 0x0 (NO_HARDCODE): No hardcode - 0x1 (STW_8): Hardcode the channel 4's source transfer width to 8 bits - 0x2 (STW_16): Hardcode the channel 4's source transfer width to 16 bits - 0x3 (STW_32): Hardcode the channel 4's source transfer width to 32 bits - 0x4 (STW_64): Hardcode the channel 4's source transfer width to 64 bits - 0x5 (STW_128): Hardcode the channel 4's source transfer width to 128 bits - 0x6 (STW_256): Hardcode the channel 4's source transfer width to 256 bits - 0x7 (RESERVED): Reserved **Value after reset:** 0x3

2 :0	R	**CH4_DTW** **Values:** - 0x0 (NO_HARDCODE): No hardcode - 0x1 (STW_8): Hardcode the channel 4's destination transfer width to 8 bits - 0x2 (STW_16): Hardcode the channel 4's destination transfer width to 16 bits - 0x3 (STW_32): Hardcode the channel 4's destination transfer width to 32 bits - 0x4 (STW_64): Hardcode the channel 4's destination transfer width to 64 bits - 0x5 (STW_128): Hardcode the channel 4's destination transfer width to 128 bits - 0x6 (STW_256): Hardcode the channel 4's destination transfer width to 256 bits - 0x7 (RESERVED): Reserved **Value after reset:** 0x3

Offset: 3E0h	**DMA_COMP_PARAMS_3: HDMA Component Parameters Register 3**	**Init = 0**
63	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0
62:60	R	**CH1_FIFO_DEPTH** **Values:** - 0x0 (FIFO_DEPTH_8): Channel 1 FIFO depth is 8 bytes - 0x1 (FIFO_DEPTH_16): Channel 1 FIFO depth is 16 bytes - 0x2 (FIFO_DEPTH_32): Channel 1 FIFO depth is 32 bytes - 0x3 (FIFO_DEPTH_64): Channel 1 FIFO depth is 64 bytes - 0x4 (FIFO_DEPTH_128): Channel 1 FIFO depth is 128 bytes - 0x5 (FIFO_DEPTH_256): Channel 1 FIFO depth is 256 bytes **Value after reset:** 0x1
59:57	R	**CH1_SMS** **Values:** - 0x0 (MASTER_1): Hardcode the AHB master 1 interface attached to the source of channel 1 - 0x1 (MASTER_2): Hardcode the AHB master 2 interface attached to the source of channel 1 - 0x2 (MASTER_3): Hardcode the AHB master 3 interface attached to the source of channel 1 - 0x3 (MASTER_4): Hardcode the AHB master 4 interface attached to the source of channel 1 - 0x4 (PROGRAMMABLE): Programmable **Value after reset:** 0x0
56:54	R	**CH1_LMS** **Values:** - 0x0 (MASTER_1): Hardcode the AHB master 1 interface attached to the LLP peripherals of channel 1 - 0x1 (MASTER_2): Hardcode the AHB master 2 interface attached to the LLP peripherals of channel 1 - 0x2 (MASTER_3): Hardcode the AHB master 3 interface attached to the LLP peripherals of channel 1 - 0x3 (MASTER_4): Hardcode the AHB master 4 interface attached to the LLP peripherals of channel 1 - 0x4 (PROGRAMMABLE): Programmable **Value after reset:** 0x0

53:51	R	**CH1_DMS** **Values:** - 0x0 (MASTER_1): Hardcode the AHB master 1 interface attached to the channel 1 destination - 0x1 (MASTER_2): Hardcode the AHB master 2 interface attached to the channel 1 destination - 0x2 (MASTER_3): Hardcode the AHB master 3 interface attached to the channel 1 destination - 0x3 (MASTER_4): Hardcode the AHB master 4 interface attached to the channel 1 destination - 0x4 (PROGRAMMABLE): Programmable **Value after reset:** 0x0
50:48	R	**CH1_MAX_MULT_SIZE** **Values:** - 0x0 (MAX_MULT_SIZE_4): Maximum value of burst transaction size that can be programmed for channel 1 is 4 - 0x1 (MAX_MULT_SIZE_8): Maximum value of burst transaction size that can be programmed for channel 1 is 8 - 0x2 (MAX_MULT_SIZE_16): Maximum value of burst transaction size that can be programmed for channel 1 is 16 - 0x3 (MAX_MULT_SIZE_32): Maximum value of burst transaction size that can be programmed for channel 1 is 32 - 0x4 (MAX_MULT_SIZE_64): Maximum value of burst transaction size that can be programmed for channel 1 is 64 - 0x5 (MAX_MULT_SIZE_128): Maximum value of burst transaction size that can be programmed for channel 1 is 128 - 0x6 (MAX_MULT_SIZE_256): Maximum value of burst transaction size that can be programmed for channel 1 is 256 - 0x7 (RESERVED): Reserved **Value after reset:** 0x1
47:46	R	**CH1_FC** **Values:** - 0x0 (F_DMA): Flow controller is DMA for channel 1 - 0x1 (F_SRC): Flow controller is Source for channel 1 - 0x2 (F_DST): Flow controller is Destination for channel 1 - 0x3 (F_ANY): Flow controller is ANY for channel 1 **Value after reset:** 0x0
45	R	**CH1_HC_LLP** **Values:** - 0x0 (PROGRAMMABLE): Exclude logic to hardcode Channel 1 LLP register to 0 - 0x0 (HARDCODED): Hardcode Channel 1 LLP register to 0 **Value after reset:** 0x0
44	R	**CH1_CTL_WB_EN** **Values:** - 0x0 (FALSE): Exclude logic to enable control register writeback after each block transfer on channel 1 - 0x1 (TRUE): Include logic to enable control register writeback after each block transfer on channel 1 **Value after reset:** 0x0

43	R	**CH1_MULTI_BLK_EN** **Values:** - 0x0 (FALSE): Exclude logic to enable channel multi-block DMA transfers on channel 1 - 0x1 (TRUE): Include logic to enable channel multi-block DMA transfers on channel 1 **Value after reset:** 0x0
42	R	**CH1_LOCK_EN** **Values:** - 0x0 (FALSE): Exclude logic to enable channel or bus locking on channel 1 - 0x1 (TRUE): Include logic to enable channel or bus locking on channel 1 **Value after reset:** 0x0
41	R	**CH1_SRC_GAT_EN** **Values:** - 0x0 (FALSE): Exclude logic to enable the gather feature on channel 1 - 0x1 (TRUE): Include logic to enable the gather feature on channel 1 **Value after reset:** 0x0
40	R	**CH1_DST_GAT_EN** **Values:** - 0x0 (FALSE): Exclude logic to enable the gather feature on channel 1 - 0x1 (TRUE): Include logic to enable the gather feature on channel 1 **Value after reset:** 0x0
39	R	**CH1_STAT_SRC** **Values:** - 0x0 (FALSE): Exclude logic to fetch a status register from source peripheral of channel 1 and write this status information to memory at end of each block transfer - 0x1 (TRUE): Include logic to fetch a status register from source peripheral of channel 1 and write this status information to memory at end of each block transfer **Value after reset:** 0x0
38	R	**CH1_STAT_DST** **Values:** - 0x0 (FALSE): Exclude logic to fetch a status register from destination peripheral of channel 1 and write this status information to memory at end of each block transfer - 0x1 (TRUE): Include logic to fetch a status register from destination peripheral of channel 1 and write this status information to memory at end of each block transfer **Value after reset:** 0x0
37:35	R	**CH1_STW** **Values:** - 0x0 (NO_HARDCODE): No hardcode - 0x1 (STW_8): Hardcode the channel 1's source transfer width to 8 bits - 0x2 (STW_16): Hardcode the channel 1's source transfer width to 16 bits - 0x3 (STW_32): Hardcode the channel 1's source transfer width to 32 bits - 0x4 (STW_64): Hardcode the channel 1's source transfer width to 64 bits - 0x5 (STW_128): Hardcode the channel 1's source transfer width to 128 bits - 0x6 (STW_256): Hardcode the channel 1's source transfer width to 256 bits - 0x7 (RESERVED): Reserved **Value after reset:** 0x3

34:32	R	**CH1_DTW** **Values:** - 0x0 (NO_HARDCODE): No hardcode - 0x1 (STW_8): Hardcode the channel 1's destination transfer width to 8 bits - 0x2 (STW_16): Hardcode the channel 1's destination transfer width to 16 bits - 0x3 (STW_32): Hardcode the channel 1's destination transfer width to 32 bits - 0x4 (STW_64): Hardcode the channel 1's destination transfer width to 64 bits - 0x5 (STW_128): Hardcode the channel 1's destination transfer width to 128 bits - 0x6 (STW_256): Hardcode the channel 1's destination transfer width to 256 bits - 0x7 (RESERVED): Reserved **Value after reset:** 0x3
31	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0
30:28	R	**CH2_FIFO_DEPTH** **Values:** - 0x0 (FIFO_DEPTH_8): Channel 2 FIFO depth is 8 bytes - 0x1 (FIFO_DEPTH_16): Channel 2 FIFO depth is 16 bytes - 0x2 (FIFO_DEPTH_32): Channel 2 FIFO depth is 32 bytes - 0x3 (FIFO_DEPTH_64): Channel 2 FIFO depth is 64 bytes - 0x4 (FIFO_DEPTH_128): Channel 2 FIFO depth is 128 bytes - 0x5 (FIFO_DEPTH_256): Channel 2 FIFO depth is 256 bytes **Value after reset:** 0x1
27:25	R	**CH2_SMS** **Values:** - 0x0 (MASTER_1): Hardcode the AHB master 1 interface attached to the source of channel 2 - 0x1 (MASTER_2): Hardcode the AHB master 2 interface attached to the source of channel 2 - 0x2 (MASTER_3): Hardcode the AHB master 3 interface attached to the source of channel 2 - 0x3 (MASTER_4): Hardcode the AHB master 4 interface attached to the source of channel 2 - 0x4 (PROGRAMMABLE): Programmable **Value after reset:** 0x0
24:22	R	**CH2_LMS** **Values:** - 0x0 (MASTER_1): Hardcode the AHB master 1 interface attached to the LLP peripherals of channel 2 - 0x1 (MASTER_2): Hardcode the AHB master 2 interface attached to the LLP peripherals of channel 2 - 0x2 (MASTER_3): Hardcode the AHB master 3 interface attached to the LLP peripherals of channel 2 - 0x3 (MASTER_4): Hardcode the AHB master 4 interface attached to the LLP peripherals of channel 2 - 0x4 (PROGRAMMABLE): Programmable **Value after reset:** 0x0

21:19	R	**CH2_DMS** **Values:** - 0x0 (MASTER_1): Hardcode the AHB master 1 interface attached to the channel 2 destination - 0x1 (MASTER_2): Hardcode the AHB master 2 interface attached to the channel 2 destination - 0x2 (MASTER_3): Hardcode the AHB master 3 interface attached to the channel 2 destination - 0x3 (MASTER_4): Hardcode the AHB master 4 interface attached to the channel 2 destination - 0x4 (PROGRAMMABLE): Programmable **Value after reset:** 0x0
18:16	R	**CH2_MAX_MULT_SIZE** **Values:** - 0x0 (MAX_MULT_SIZE_4): Maximum value of burst transaction size that can be programmed for channel 2 is 4 - 0x1 (MAX_MULT_SIZE_8): Maximum value of burst transaction size that can be programmed for channel 2 is 8 - 0x2 (MAX_MULT_SIZE_16): Maximum value of burst transaction size that can be programmed for channel 2 is 16 - 0x3 (MAX_MULT_SIZE_32): Maximum value of burst transaction size that can be programmed for channel 2 is 32 - 0x4 (MAX_MULT_SIZE_64): Maximum value of burst transaction size that can be programmed for channel 2 is 64 - 0x5 (MAX_MULT_SIZE_128): Maximum value of burst transaction size that can be programmed for channel 2 is 128 - 0x6 (MAX_MULT_SIZE_256): Maximum value of burst transaction size that can be programmed for channel 2 is 256 - 0x7 (RESERVED): Reserved **Value after reset:** 0x1
15:14	R	**CH2_FC** **Values:** - 0x0 (F_DMA): Flow controller is DMA for channel 2 - 0x1 (F_SRC): Flow controller is Source for channel 2 - 0x2 (F_DST): Flow controller is Destination for channel 2 - 0x3 (F_ANY): Flow controller is ANY for channel 2 **Value after reset:** 0x0
13	R	**CH2_HC_LLP** **Values:** - 0x0 (PROGRAMMABLE): Exclude logic to hardcode Channel 2 LLP register to 0 - 0x0 (HARDCODED): Hardcode Channel 2 LLP register to 0 **Value after reset:** 0x0
12	R	**CH2_CTL_WB_EN** **Values:** - 0x0 (FALSE): Exclude logic to enable control register writeback after each block transfer on channel 2 - 0x1 (TRUE): Include logic to enable control register writeback after each block transfer on channel 2 **Value after reset:** 0x0

11	R	**CH2_MULTI_BLK_EN** **Values:** - 0x0 (FALSE): Exclude logic to enable channel multi-block DMA transfers on channel 2 - 0x1 (TRUE): Include logic to enable channel multi-block DMA transfers on channel 2 **Value after reset:** 0x0
10	R	**CH2_LOCK_EN** **Values:** - 0x0 (FALSE): Exclude logic to enable channel or bus locking on channel 2 - 0x1 (TRUE): Include logic to enable channel or bus locking on channel 2 **Value after reset:** 0x0
9	R	**CH2_SRC_GAT_EN** **Values:** - 0x0 (FALSE): Exclude logic to enable the gather feature on channel 2 - 0x1 (TRUE): Include logic to enable the gather feature on channel 2 **Value after reset:** 0x0
8	R	**CH2_DST_GAT_EN** **Values:** - 0x0 (FALSE): Exclude logic to enable the gather feature on channel 2 - 0x1 (TRUE): Include logic to enable the gather feature on channel 2 **Value after reset:** 0x0
7	R	**CH2_STAT_SRC** **Values:** - 0x0 (FALSE): Exclude logic to fetch a status register from source peripheral of channel 2 and write this status information to memory at end of each block transfer - 0x1 (TRUE): Include logic to fetch a status register from source peripheral of channel 2 and write this status information to memory at end of each block transfer **Value after reset:** 0x0
6	R	**CH2_STAT_DST** **Values:** - 0x0 (FALSE): Exclude logic to fetch a status register from destination peripheral of channel 2 and write this status information to memory at end of each block transfer - 0x1 (TRUE): Include logic to fetch a status register from destination peripheral of channel 2 and write this status information to memory at end of each block transfer **Value after reset:** 0x0
5 :3	R	**CH2_STW** **Values:** - 0x0 (NO_HARDCODE): No hardcode - 0x1 (STW_8): Hardcode the channel 2's source transfer width to 8 bits - 0x2 (STW_16): Hardcode the channel 2's source transfer width to 16 bits - 0x3 (STW_32): Hardcode the channel 2's source transfer width to 32 bits - 0x4 (STW_64): Hardcode the channel 2's source transfer width to 64 bits - 0x5 (STW_128): Hardcode the channel 2's source transfer width to 128 bits - 0x6 (STW_256): Hardcode the channel 2's source transfer width to 256 bits - 0x7 (RESERVED): Reserved **Value after reset:** 0x3

2 :0	R	**CH2_DTW** **Values:** - 0x0 (NO_HARDCODE): No hardcode - 0x1 (STW_8): Hardcode the channel 2's destination transfer width to 8 bits - 0x2 (STW_16): Hardcode the channel 2's destination transfer width to 16 bits - 0x3 (STW_32): Hardcode the channel 2's destination transfer width to 32 bits - 0x4 (STW_64): Hardcode the channel 2's destination transfer width to 64 bits - 0x5 (STW_128): Hardcode the channel 2's destination transfer width to 128 bits - 0x6 (STW_256): Hardcode the channel 2's destination transfer width to 256 bits - 0x7 (RESERVED): Reserved **Value after reset:** 0x3

Offset: 3E8h	**DMA_COMP_PARAMS_2: HDMA Component Parameters Register 2**	**Init = 0**
63:60	R	**CH7_MULTI_BLK_TYPE** Source status information retrieved by hardware from the address pointed to by the contents of the STATAR2 register **Values:** 0x0 - 0x0 (PROGRAMMABLE): Allow all types of multi-support - 0x1 (CONT_RELOAD): Allow only multi-block transfers where SAR7 is contiguous; DAR and CTL are reloaded from their initial values - 0x2 (RELOAD_CONT): Allow only multi-block transfers where SAR7 and CTL7 are reloaded from their initial values; DAR7 is contiguous - 0x3 (RELOAD_RELOAD): Allow only multi-block transfers where SAR7, DAR7,and CTL7 are reloaded from their initial values - 0x4 (CONT_LLP): Allow only multi-block transfers where SAR7 is contiguous;DAR7, CTL7, and LLP7 are loaded from the next linked list item - 0x5 (RELOAD_LLP): Allow only multi-block transfers where SAR7 is reloaded from its initial value; DAR7, CTL7, and LLP7,are loaded from the next linked list item - 0x6 (CNT_LLP): Allow only multi-block transfers where SAR7, CTL7, and LLP7 are loaded from the next linked list item; DAR7 is contiguous - 0x7 (LLP_RELOAD): Allow only multi-block transfers where SAR7, CTL7, and LLP7,are loaded from the next linked list item; DARx is reloaded from its initial values - 0x8 (LLP_LLP): Allow only multi-block transfers where SAR7, DAR7, CTL7, and LLP7 are loaded from the next linked list item **Value after reset:** 0x0 **Volatile:** true

59:56	R	**CH6_MULTI_BLK_TYPE** Source status information retrieved by hardware from the address pointed to by the contents of the STATAR2 register **Values:** 0x0 - 0x0 (PROGRAMMABLE): Allow all types of multi-support - 0x1 (CONT_RELOAD): Allow only multi-block transfers where SAR6 is contiguous; DAR and CTL are reloaded from their initial values - 0x2 (RELOAD_CONT): Allow only multi-block transfers where SAR6 and CTL6 are reloaded from their initial values; DAR6 is contiguous - 0x3 (RELOAD_RELOAD): Allow only multi-block transfers where SAR6, DAR6,and CTL6 are reloaded from their initial values - 0x4 (CONT_LLP): Allow only multi-block transfers where SAR6 is contiguous;DAR6, CTL6, and LLP6 are loaded from the next linked list item - 0x5 (RELOAD_LLP): Allow only multi-block transfers where SAR6 is reloaded from its initial value; DAR6, CTL6, and LLP6,are loaded from the next linked list item - 0x6 (CNT_LLP): Allow only multi-block transfers where SAR6, CTL6, and LLP6 are loaded from the next linked list item; DAR6 is contiguous - 0x7 (LLP_RELOAD): Allow only multi-block transfers where SAR6, CTL6, and LLP6,are loaded from the next linked list item; DARx is reloaded from its initial values. - 0x8 (LLP_LLP): Allow only multi-block transfers where SAR6, DAR6, CTL6, and LLP6 are loaded from the next linked list item **Value after reset:** 0x0 **Volatile:** true
55:52	R	**CH5_MULTI_BLK_TYPE** Source status information retrieved by hardware from the address pointed to by the contents of the STATAR2 register **Values:** 0x0 - 0x0 (PROGRAMMABLE): Allow all types of multi-support - 0x1 (CONT_RELOAD): Allow only multi-block transfers where SAR5 is contiguous; DAR and CTL are reloaded from their initial values - 0x2 (RELOAD_CONT): Allow only multi-block transfers where SAR5 and CTL5 are reloaded from their initial values; DAR5 is contiguous - 0x3 (RELOAD_RELOAD): Allow only multi-block transfers where SAR5, DAR5,and CTL5 are reloaded from their initial values - 0x4 (CONT_LLP): Allow only multi-block transfers where SAR5 is contiguous;DAR5, CTL5, and LLP5 are loaded from the next linked list item - 0x5 (RELOAD_LLP): Allow only multi-block transfers where SAR5 is reloaded from its initial value; DAR5, CTL5, and LLP5,are loaded from the next linked list item - 0x6 (CNT_LLP): Allow only multi-block transfers where SAR5, CTL5, and LLP5 are loaded from the next linked list item; DAR5 is contiguous - 0x7 (LLP_RELOAD): Allow only multi-block transfers where SAR5, CTL5, and LLP5,are loaded from the next linked list item; DARx is reloaded from its initial values. - 0x8 (LLP_LLP): Allow only multi-block transfers where SAR5, DAR5, CTL5, and LLP5 are loaded from the next linked list item **Value after reset:** 0x0 **Volatile:** true

51:48	R	**CH4_MULTI_BLK_TYPE** Source status information retrieved by hardware from the address pointed to by the contents of the STATAR2 register **Values:** 0x0 - 0x0 (PROGRAMMABLE): Allow all types of multi-support - 0x1 (CONT_RELOAD): Allow only multi-block transfers where SAR4 is contiguous; DAR and CTL are reloaded from their initial values - 0x2 (RELOAD_CONT): Allow only multi-block transfers where SAR4 and CTL4 are reloaded from their initial values; DAR4 is contiguous - 0x3 (RELOAD_RELOAD): Allow only multi-block transfers where SAR4, DAR4,and CTL4 are reloaded from their initial values - 0x4 (CONT_LLP): Allow only multi-block transfers where SAR4 is contiguous;DAR4, CTL4, and LLP4 are loaded from the next linked list item - 0x5 (RELOAD_LLP): Allow only multi-block transfers where SAR4 is reloaded from its initial value; DAR4, CTL4, and LLP4,are loaded from the next linked list item - 0x6 (CNT_LLP): Allow only multi-block transfers where SAR4, CTL4, and LLP4 are loaded from the next linked list item; DAR4 is contiguous - 0x7 (LLP_RELOAD): Allow only multi-block transfers where SAR4, CTL4, and LLP4,are loaded from the next linked list item; DARx is reloaded from its initial values. - 0x8 (LLP_LLP): Allow only multi-block transfers where SAR4, DAR4, CTL4, and LLP4 are loaded from the next linked list item **Value after reset:** 0x0 **Volatile:** true
47:44	R	**CH3_MULTI_BLK_TYPE** Source status information retrieved by hardware from the address pointed to by the contents of the STATAR2 register **Values:** 0x0 - 0x0 (PROGRAMMABLE): Allow all types of multi-support - 0x1 (CONT_RELOAD): Allow only multi-block transfers where SAR3 is contiguous; DAR and CTL are reloaded from their initial values - 0x2 (RELOAD_CONT): Allow only multi-block transfers where SAR3 and CTL3 are reloaded from their initial values; DAR3 is contiguous - 0x3 (RELOAD_RELOAD): Allow only multi-block transfers where SAR3, DAR3,and CTL3 are reloaded from their initial values - 0x4 (CONT_LLP): Allow only multi-block transfers where SAR3 is contiguous;DAR3, CTL3, and LLP3 are loaded from the next linked list item - 0x5 (RELOAD_LLP): Allow only multi-block transfers where SAR3 is reloaded from its initial value; DAR3, CTL3, and LLP3,are loaded from the next linked list item - 0x6 (CNT_LLP): Allow only multi-block transfers where SAR3, CTL3, and LLP3 are loaded from the next linked list item; DAR3 is contiguous - 0x7 (LLP_RELOAD): Allow only multi-block transfers where SAR3, CTL3, and LLP3,are loaded from the next linked list item; DARx is reloaded from its initial values. - 0x8 (LLP_LLP): Allow only multi-block transfers where SAR3, DAR3, CTL3, and LLP3 are loaded from the next linked list item **Value after reset:** 0x0 **Volatile:** true

43:40	R	**CH2_MULTI_BLK_TYPE** Source status information retrieved by hardware from the address pointed to by the contents of the STATAR2 register **Values:** 0x0 - 0x0 (PROGRAMMABLE): Allow all types of multi-support - 0x1 (CONT_RELOAD): Allow only multi-block transfers where SAR2 is contiguous; DAR and CTL are reloaded from their initial values - 0x2 (RELOAD_CONT): Allow only multi-block transfers where SAR2 and CTL2 are reloaded from their initial values; DAR3 is contiguous - 0x3 (RELOAD_RELOAD): Allow only multi-block transfers where SAR2, DAR2,and CTL2 are reloaded from their initial values - 0x4 (CONT_LLP): Allow only multi-block transfers where SAR2 is contiguous;DAR2, CTL2, and LLP2 are loaded from the next linked list item - 0x5 (RELOAD_LLP): Allow only multi-block transfers where SAR2 is reloaded from its initial value; DAR2, CTL2, and LLP2,are loaded from the next linked list item - 0x6 (CNT_LLP): Allow only multi-block transfers where SAR2, CTL2, and LLP2 are loaded from the next linked list item; DAR2 is contiguous - 0x7 (LLP_RELOAD): Allow only multi-block transfers where SAR2, CTL2, and LLP2,are loaded from the next linked list item; DARx is reloaded from its initial values. - 0x8 (LLP_LLP): Allow only multi-block transfers where SAR2, DAR2, CTL2, and LLP2 are loaded from the next linked list item **Value after reset:** 0x0 **Volatile:** true
39:36	R	**CH1_MULTI_BLK_TYPE** Source status information retrieved by hardware from the address pointed to by the contents of the STATAR1 register **Values:** 0x0 - 0x0 (PROGRAMMABLE): Allow all types of multi-support - 0x1 (CONT_RELOAD): Allow only multi-block transfers where SAR1 is contiguous; DAR and CTL are reloaded from their initial values - 0x2 (RELOAD_CONT): Allow only multi-block transfers where SAR1 and CTL1 are reloaded from their initial values; DAR3 is contiguous - 0x3 (RELOAD_RELOAD): Allow only multi-block transfers where SAR1, DAR1,and CTL1 are reloaded from their initial values - 0x4 (CONT_LLP): Allow only multi-block transfers where SAR1 is contiguous;DAR1, CTL1, and LLP1 are loaded from the next linked list item - 0x5 (RELOAD_LLP): Allow only multi-block transfers where SAR1 is reloaded from its initial value; DAR1, CTL1, and LLP1,are loaded from the next linked list item - 0x6 (CNT_LLP): Allow only multi-block transfers where SAR1, CTL1, and LLP1 are loaded from the next linked list item; DAR1 is contiguous - 0x7 (LLP_RELOAD): Allow only multi-block transfers where SAR1, CTL1, and LLP1,are loaded from the next linked list item; DARx is reloaded from its initial values. - 0x8 (LLP_LLP): Allow only multi-block transfers where SAR1, DAR1, CTL1, and LLP1 are loaded from the next linked list item **Value after reset:** 0x0 **Volatile:** true

35:32	R	**CH0_MULTI_BLK_TYPE** Source status information retrieved by hardware from the address pointed to by the contents of the STATAR0 register **Values:** 0x0 - 0x0 (PROGRAMMABLE): Allow all types of multi-support - 0x1 (CONT_RELOAD): Allow only multi-block transfers where SAR0 is contiguous; DAR and CTL are reloaded from their initial values - 0x2 (RELOAD_CONT): Allow only multi-block transfers where SAR0 and CTL0 are reloaded from their initial values; DAR3 is contiguous - 0x3 (RELOAD_RELOAD): Allow only multi-block transfers where SAR0, DAR0,and CTL0 are reloaded from their initial values - 0x4 (CONT_LLP): Allow only multi-block transfers where SAR0 is contiguous;DAR0, CTL0, and LLP0 are loaded from the next linked list item - 0x5 (RELOAD_LLP): Allow only multi-block transfers where SAR0 is reloaded from its initial value; DAR0, CTL0, and LLP0,are loaded from the next linked list item - 0x6 (CNT_LLP): Allow only multi-block transfers where SAR0, CTL0, and LLP0 are loaded from the next linked list item; DAR0 is contiguous - 0x7 (LLP_RELOAD): Allow only multi-block transfers where SAR0, CTL0, and LLP0,are loaded from the next linked list item; DARx is reloaded from its initial values. - 0x8 (LLP_LLP): Allow only multi-block transfers where SAR0, DAR0, CTL0, and LLP0 are loaded from the next linked list item **Value after reset:** 0x0 **Volatile:** true
31	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0
30:28	R	**CH0_FIFO_DEPTH** **Values:** - 0x0 (FIFO_DEPTH_8): Channel 0 FIFO depth is 8 bytes - 0x1 (FIFO_DEPTH_16): Channel 0 FIFO depth is 16 bytes - 0x2 (FIFO_DEPTH_32): Channel 0 FIFO depth is 32 bytes - 0x3 (FIFO_DEPTH_64): Channel 0 FIFO depth is 64 bytes - 0x4 (FIFO_DEPTH_128): Channel 0 FIFO depth is 128 bytes - 0x5 (FIFO_DEPTH_256): Channel 0 FIFO depth is 256 bytes **Value after reset:** 0x1
27:25	R	**CH0_SMS** **Values:** - 0x0 (MASTER_1): Hardcode the AHB master 1 interface attached to the source of channel 0 - 0x1 (MASTER_2): Hardcode the AHB master 2 interface attached to the source of channel 0 - 0x2 (MASTER_3): Hardcode the AHB master 3 interface attached to the source of channel 0 - 0x3 (MASTER_4): Hardcode the AHB master 4 interface attached to the source of channel 0 - 0x4 (PROGRAMMABLE): Programmable **Value after reset:** 0x0

24:22	R	**CH0_LMS** **Values:** - 0x0 (MASTER_1): Hardcode the AHB master 1 interface attached to the LLP peripherals of channel 0 - 0x1 (MASTER_2): Hardcode the AHB master 2 interface attached to the LLP peripherals of channel 0 - 0x2 (MASTER_3): Hardcode the AHB master 3 interface attached to the LLP peripherals of channel 0 - 0x3 (MASTER_4): Hardcode the AHB master 4 interface attached to the LLP peripherals of channel 0 - 0x4 (PROGRAMMABLE): Programmable **Value after reset:** 0x0
21:19	R	**CH0_DMS** **Values:** - 0x0 (MASTER_1): Hardcode the AHB master 1 interface attached to the channel 0 destination - 0x1 (MASTER_2): Hardcode the AHB master 2 interface attached to the channel 0 destination - 0x2 (MASTER_3): Hardcode the AHB master 3 interface attached to the channel 0 destination - 0x3 (MASTER_4): Hardcode the AHB master 4 interface attached to the channel 0 destination - 0x4 (PROGRAMMABLE): Programmable **Value after reset:** 0x0
18:16	R	**CH0_MAX_MULT_SIZE** **Values:** - 0x0 (MAX_MULT_SIZE_4): Maximum value of burst transaction size that can be programmed for channel 0 is 4 - 0x1 (MAX_MULT_SIZE_8): Maximum value of burst transaction size that can be programmed for channel 0 is 8 - 0x2 (MAX_MULT_SIZE_16): Maximum value of burst transaction size that can be programmed for channel 0 is 16 - 0x3 (MAX_MULT_SIZE_32): Maximum value of burst transaction size that can be programmed for channel 0 is 32 - 0x4 (MAX_MULT_SIZE_64): Maximum value of burst transaction size that can be programmed for channel 0 is 64 - 0x5 (MAX_MULT_SIZE_128): Maximum value of burst transaction size that can be programmed for channel 0 is 128 - 0x6 (MAX_MULT_SIZE_256): Maximum value of burst transaction size that can be programmed for channel 0 is 256 - 0x7 (RESERVED): Reserved **Value after reset:** 0x1
15:14	R	**CH0_FC** **Values:** - 0x0 (F_DMA): Flow controller is DMA for channel 0 - 0x1 (F_SRC): Flow controller is Source for channel 0 - 0x2 (F_DST): Flow controller is Destination for channel 0 - 0x3 (F_ANY): Flow controller is ANY for channel 0 **Value after reset:** 0x0

13	R	**CH0_HC_LLP** **Values:** - 0x0 (PROGRAMMABLE): Exclude logic to hardcode Channel 0 LLP register to 0 - 0x0 (HARDCODED): Hardcode Channel 0 LLP register to 0 **Value after reset:** 0x0
12	R	**CH0_CTL_WB_EN** **Values:** - 0x0 (FALSE): Exclude logic to enable control register writeback after each block transfer on channel 0 - 0x1 (TRUE): Include logic to enable control register writeback after each block transfer on channel 0 **Value after reset:** 0x0
11	R	**CH0_MULTI_BLK_EN** **Values:** - 0x0 (FALSE): Exclude logic to enable channel multi-block DMA transfers on channel 0 - 0x1 (TRUE): Include logic to enable channel multi-block DMA transfers on channel 0 **Value after reset:** 0x0
10	R	**CH0_LOCK_EN** **Values:** - 0x0 (FALSE): Exclude logic to enable channel or bus locking on channel 0 - 0x1 (TRUE): Include logic to enable channel or bus locking on channel 0 **Value after reset:** 0x0
9	R	**CH0_SRC_GAT_EN** **Values:** - 0x0 (FALSE): Exclude logic to enable the gather feature on channel 0 - 0x1 (TRUE): Include logic to enable the gather feature on channel 0 **Value after reset:** 0x0
8	R	**CH0_DST_GAT_EN** **Values:** - 0x0 (FALSE): Exclude logic to enable the gather feature on channel 0 - 0x1 (TRUE): Include logic to enable the gather feature on channel 0 **Value after reset:** 0x0
7	R	**CH0_STAT_SRC** **Values:** - 0x0 (FALSE): Exclude logic to fetch a status register from source peripheral of channel 0 and write this status information to memory at end of each block transfer - 0x1 (TRUE): Include logic to fetch a status register from source peripheral of channel 0 and write this status information to memory at end of each block transfer **Value after reset:** 0x0

6	R	**CH0_STAT_DST** **Values:** - 0x0 (FALSE): Exclude logic to fetch a status register from destination peripheral of channel 0 and write this status information to memory at end of each block transfer - 0x1 (TRUE): Include logic to fetch a status register from destination peripheral of channel 0 and write this status information to memory at end of each block transfer **Value after reset:** 0x0
5 :3	R	**CH0_STW** **Values:** - 0x0 (NO_HARDCODE): No hardcode - 0x1 (STW_8): Hardcode the channel 0's source transfer width to 8 bits - 0x2 (STW_16): Hardcode the channel 0's source transfer width to 16 bits - 0x3 (STW_32): Hardcode the channel 0's source transfer width to 32 bits - 0x4 (STW_64): Hardcode the channel 0's source transfer width to 64 bits - 0x5 (STW_128): Hardcode the channel 0's source transfer width to 128 bits - 0x6 (STW_256): Hardcode the channel 0's source transfer width to 256 bits - 0x7 (RESERVED): Reserved **Value after reset:** 0x3
2 :0	R	**CH0_DTW** **Values:** - 0x0 (NO_HARDCODE): No hardcode - 0x1 (STW_8): Hardcode the channel 0's destination transfer width to 8 bits - 0x2 (STW_16): Hardcode the channel 0's destination transfer width to 16 bits - 0x3 (STW_32): Hardcode the channel 0's destination transfer width to 32 bits - 0x4 (STW_64): Hardcode the channel 0's destination transfer width to 64 bits - 0x5 (STW_128): Hardcode the channel 0's destination transfer width to 128 bits - 0x6 (STW_256): Hardcode the channel 0's destination transfer width to 256 bits - 0x7 (RESERVED): Reserved **Value after reset:** 0x3

Offset: 3F0h	**DMA_COMP_PARAMS_1: HDMA Component Parameters Register 1**	**Init = 0**
63:62	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0 **Volatile:** true
61	R	**STATIC_ENDIAN_SELECT** **Value after reset:** 0x1 **Volatile:** true
60	R	**ADD_ENCODED_PARAMS** **Values:** - 0x0 (FALSE): Add encoded parameter is not enabled - 0x1 (TRUE): Add encoded parameter is enabled **Value after reset:** 0x1 **Volatile:** true

59:55	R	**NUM_HS_INT** **Values:** - 0x0 (HS_INTERFACE_0): Number of handshaking interfaces is 0 - 0x1 (HS_INTERFACE_1): Number of handshaking interfaces is 1 - 0x2 (HS_INTERFACE_2): Number of handshaking interfaces is 2 - 0x3 (HS_INTERFACE_3): Number of handshaking interfaces is 3 - 0x4 (HS_INTERFACE_4): Number of handshaking interfaces is 4 - 0x5 (HS_INTERFACE_5): Number of handshaking interfaces is 5 - 0x6 (HS_INTERFACE_6): Number of handshaking interfaces is 6 - 0x7 (HS_INTERFACE_7): Number of handshaking interfaces is 7 - 0x8 (HS_INTERFACE_8): Number of handshaking interfaces is 8 - 0x9 (HS_INTERFACE_9): Number of handshaking interfaces is 9 - 0xa (HS_INTERFACE_a): Number of handshaking interfaces is 10 - 0xb (HS_INTERFACE_b): Number of handshaking interfaces is 11 - 0xc (HS_INTERFACE_c): Number of handshaking interfaces is 12 - 0xd (HS_INTERFACE_d): Number of handshaking interfaces is 13 - 0xe (HS_INTERFACE_e): Number of handshaking interfaces is 14 - 0xf (HS_INTERFACE_f): Number of handshaking interfaces is 15 - 0x10 (HS_INTERFACE_10): Number of handshaking interfaces is 16 **Value after reset:** 0x10 **Volatile:** true
54:53	R	**M1_HDATA_WIDTH** **Values:** - 0x0 (DATA_BUS_WIDTH_32): Master1 interface data bus width is 32 bits - 0x1 (DATA_BUS_WIDTH_64): Master1 interface data bus width is 64 bits - 0x2 (DATA_BUS_WIDTH_128): Master1 interface data bus width is 128 bits - 0x3 (DATA_BUS_WIDTH_256): Master1 interface data bus width is 256 bits **Value after reset:** 0x0 **Volatile:** true
52:51	R	**M2_HDATA_WIDTH** **Values:** - 0x0 (DATA_BUS_WIDTH_32): Master2 interface data bus width is 32 bits - 0x1 (DATA_BUS_WIDTH_64): Master2 interface data bus width is 64 bits - 0x2 (DATA_BUS_WIDTH_128): Master2 interface data bus width is 128 bits - 0x3 (DATA_BUS_WIDTH_256): Master2 interface data bus width is 256 bits **Value after reset:** 0x0 **Volatile:** true
50:49	R	**M3_HDATA_WIDTH** **Values:** - 0x0 (DATA_BUS_WIDTH_32): Master3 interface data bus width is 32 bits - 0x1 (DATA_BUS_WIDTH_64): Master3 interface data bus width is 64 bits - 0x2 (DATA_BUS_WIDTH_128): Master3 interface data bus width is 128 bits - 0x3 (DATA_BUS_WIDTH_256): Master3 interface data bus width is 256 bits **Value after reset:** 0x0 **Volatile:** true

48:47	R	**M4_HDATA_WIDTH** **Values:** - 0x0 (DATA_BUS_WIDTH_32): Master4 interface data bus width is 32 bits - 0x1 (DATA_BUS_WIDTH_64): Master4 interface data bus width is 64 bits - 0x2 (DATA_BUS_WIDTH_128): Master4 interface data bus width is 128 bits - 0x3 (DATA_BUS_WIDTH_256): Master4 interface data bus width is 256 bits **Value after reset:** 0x0 **Volatile:** true
46:45	R	**S_HDATA_WIDTH** **Values:** - 0x0 (DATA_BUS_WIDTH_32): Slave interface data bus width is 32 bits - 0x1 (DATA_BUS_WIDTH_64): Slave interface data bus width is 64 bits - 0x2 (DATA_BUS_WIDTH_128): Slave interface data bus width is 128 bits - 0x3 (DATA_BUS_WIDTH_256): Slave interface data bus width is 256 bits **Value after reset:** 0x0 **Volatile:** true
44:43	R	**NUM_MASTER_INT** **Values:** - 0x0 (NUM_MAST_INTERFACE_1): Number of MASTER iterface is 1 - 0x1 (NUM_MAST_INTERFACE_2): Number of MASTER iterface is 2 - 0x2 (NUM_MAST_INTERFACE_3): Number of MASTER iterface is 3 - 0x3 (NUM_MAST_INTERFACE_4): Number of MASTER iterface is 4 **Value after reset:** 0x0 **Volatile:** true
42:40	R	**NUM_CHANNELS** **Values:** - 0x0 (NUM_CHANNEL_1): Number of DMA Channels is 1 - 0x1 (NUM_CHANNEL_2): Number of DMA Channels is 2 - 0x2 (NUM_CHANNEL_3): Number of DMA Channels is 3 - 0x3 (NUM_CHANNEL_4): Number of DMA Channels is 4 - 0x4 (NUM_CHANNEL_5): Number of DMA Channels is 5 - 0x5 (NUM_CHANNEL_6): Number of DMA Channels is 6 - 0x6 (NUM_CHANNEL_7): Number of DMA Channels is 7 - 0x7 (NUM_CHANNEL_8): Number of DMA Channels is 8 **Value after reset:** 0x7 **Volatile:** true
39:36	R	**RSVD** Reserved field - read-only **Value after reset:** 0x0 **Volatile:** true
35	R	**MAX_ABRST** Reserved field - read-only **Values:** - 0x0 (FALSE): Maximum AMBA burst length is not under the control of software - 0x1 (TRUE): Limit the maximum AMBA burst length to a value under software control by writing to the channel configuration register **Value after reset:** 0x0 **Volatile:** true

34:33	R	**INTR_IO** Reserved field - read-only **Values:** - 0x0 (ALL_INT): ALL interrupt-related signals appear as outputs on the design - 0x1 (TYPE_INT): Only TYPE interrupt-related signals appear as outputs on the design - 0x2 (COMBINED_INT): Only COMBINED interruptrelated signals appear as outputs on the design - 0x3 (RESERVED): Reserved **Value after reset:** 0x0 **Volatile:** true
32	R	**BIG_ENDIAN** **Values:** - 0x0 (FALSE): Big Endian - 0x1 (TRUE): Little Endian **Value after reset:** 0x0 **Volatile:** true
31:28	R	**CH7_MAX_BLK_SIZE** Reserved field - read-only **Values:** - 0x0 (MAX_BLOCK_SIZE_3): Maximum block size in source transfer widths is 3 for channel 7 - 0x1 (MAX_BLOCK_SIZE_7): Maximum block size in source transfer widths is 7 for channel 7 - 0x2 (MAX_BLOCK_SIZE_15): Maximum block size in source transfer widths is 15 for channel 7 - 0x3 (MAX_BLOCK_SIZE_31): Maximum block size in source transfer widths is 31 for channel 7 - 0x4 (MAX_BLOCK_SIZE_63): Maximum block size in source transfer widths is 63 for channel 7 - 0x5 (MAX_BLOCK_SIZE_127): Maximum block size in source transfer widths is 127 for channel 7 - 0x6 (MAX_BLOCK_SIZE_255): Maximum block size in source transfer widths is 255 for channel 7 - 0x7 (MAX_BLOCK_SIZE_511): Maximum block size in source transfer widths is 511 for channel 7 - 0x8 (MAX_BLOCK_SIZE_1023): Maximum block size in source transfer widths is 1023 for channel 7 - 0x9 (MAX_BLOCK_SIZE_2047): Maximum block size in source transfer widths is 2047 for channel 7 - 0xa (MAX_BLOCK_SIZE_4095): Maximum block size in source transfer widths is 4095 for channel 7 **Value after reset:** 0x3 **Volatile:** true

27:24	R	**CH6_MAX_BLK_SIZE** Reserved field - read-only **Values:** - 0x0 (MAX_BLOCK_SIZE_3): Maximum block size in source transfer widths is 3 for channel 6 - 0x1 (MAX_BLOCK_SIZE_7): Maximum block size in source transfer widths is 7 for channel 6 - 0x2 (MAX_BLOCK_SIZE_15): Maximum block size in source transfer widths is 15 for channel 6 - 0x3 (MAX_BLOCK_SIZE_31): Maximum block size in source transfer widths is 31 for channel 6 - 0x4 (MAX_BLOCK_SIZE_63): Maximum block size in source transfer widths is 63 for channel 6 - 0x5 (MAX_BLOCK_SIZE_127): Maximum block size in source transfer widths is 127 for channel 6 - 0x6 (MAX_BLOCK_SIZE_255): Maximum block size in source transfer widths is 255 for channel 6 - 0x7 (MAX_BLOCK_SIZE_511): Maximum block size in source transfer widths is 511 for channel 6 - 0x8 (MAX_BLOCK_SIZE_1023): Maximum block size in source transfer widths is 1023 for channel 6 - 0x9 (MAX_BLOCK_SIZE_2047): Maximum block size in source transfer widths is 2047 for channel 6 - 0xa (MAX_BLOCK_SIZE_4095): Maximum block size in source transfer widths is 4095 for channel 6 **Value after reset:** 0x3 **Volatile:** true

23:20	R	**CH5_MAX_BLK_SIZE** Reserved field - read-only **Values:** - 0x0 (MAX_BLOCK_SIZE_3): Maximum block size in source transfer widths is 3 for channel 5 - 0x1 (MAX_BLOCK_SIZE_7): Maximum block size in source transfer widths is 7 for channel 5 - 0x2 (MAX_BLOCK_SIZE_15): Maximum block size in source transfer widths is 15 for channel 5 - 0x3 (MAX_BLOCK_SIZE_31): Maximum block size in source transfer widths is 31 for channel 5 - 0x4 (MAX_BLOCK_SIZE_63): Maximum block size in source transfer widths is 63 for channel 5 - 0x5 (MAX_BLOCK_SIZE_127): Maximum block size in source transfer widths is 127 for channel 5 - 0x6 (MAX_BLOCK_SIZE_255): Maximum block size in source transfer widths is 255 for channel 5 - 0x7 (MAX_BLOCK_SIZE_511): Maximum block size in source transfer widths is 511 for channel 5 - 0x8 (MAX_BLOCK_SIZE_1023): Maximum block size in source transfer widths is 1023 for channel 5 - 0x9 (MAX_BLOCK_SIZE_2047): Maximum block size in source transfer widths is 2047 for channel 5 - 0xa (MAX_BLOCK_SIZE_4095): Maximum block size in source transfer widths is 4095 for channel 5 **Value after reset:** 0x3 **Volatile:** true

19:16	R	**CH4_MAX_BLK_SIZE** Reserved field - read-only **Values:** - 0x0 (MAX_BLOCK_SIZE_3): Maximum block size in source transfer widths is 3 for channel 4 - 0x1 (MAX_BLOCK_SIZE_7): Maximum block size in source transfer widths is 7 for channel 4 - 0x2 (MAX_BLOCK_SIZE_15): Maximum block size in source transfer widths is 15 for channel 4 - 0x3 (MAX_BLOCK_SIZE_31): Maximum block size in source transfer widths is 31 for channel 4 - 0x4 (MAX_BLOCK_SIZE_63): Maximum block size in source transfer widths is 63 for channel 4 - 0x5 (MAX_BLOCK_SIZE_127): Maximum block size in source transfer widths is 127 for channel 4 - 0x6 (MAX_BLOCK_SIZE_255): Maximum block size in source transfer widths is 255 for channel 4 - 0x7 (MAX_BLOCK_SIZE_511): Maximum block size in source transfer widths is 511 for channel 4 - 0x8 (MAX_BLOCK_SIZE_1023): Maximum block size in source transfer widths is 1023 for channel 4 - 0x9 (MAX_BLOCK_SIZE_2047): Maximum block size in source transfer widths is 2047 for channel 4 - 0xa (MAX_BLOCK_SIZE_4095): Maximum block size in source transfer widths is 4095 for channel 4 **Value after reset:** 0x3 **Volatile:** true

15:12	R	**CH3_MAX_BLK_SIZE** Reserved field - read-only **Values:** - 0x0 (MAX_BLOCK_SIZE_3): Maximum block size in source transfer widths is 3 for channel 3 - 0x1 (MAX_BLOCK_SIZE_7): Maximum block size in source transfer widths is 7 for channel 3 - 0x2 (MAX_BLOCK_SIZE_15): Maximum block size in source transfer widths is 15 for channel 3 - 0x3 (MAX_BLOCK_SIZE_31): Maximum block size in source transfer widths is 31 for channel 3 - 0x4 (MAX_BLOCK_SIZE_63): Maximum block size in source transfer widths is 63 for channel 3 - 0x5 (MAX_BLOCK_SIZE_127): Maximum block size in source transfer widths is 127 for channel 3 - 0x6 (MAX_BLOCK_SIZE_255): Maximum block size in source transfer widths is 255 for channel 3 - 0x7 (MAX_BLOCK_SIZE_511): Maximum block size in source transfer widths is 511 for channel 3 - 0x8 (MAX_BLOCK_SIZE_1023): Maximum block size in source transfer widths is 1023 for channel 3 - 0x9 (MAX_BLOCK_SIZE_2047): Maximum block size in source transfer widths is 2047 for channel 3 - 0xa (MAX_BLOCK_SIZE_4095): Maximum block size in source transfer widths is 4095 for channel 3 **Value after reset:** 0x3 **Volatile:** true

11:8	R	**CH2_MAX_BLK_SIZE** Reserved field - read-only **Values:** - 0x0 (MAX_BLOCK_SIZE_3): Maximum block size in source transfer widths is 3 for channel 2 - 0x1 (MAX_BLOCK_SIZE_7): Maximum block size in source transfer widths is 7 for channel 2 - 0x2 (MAX_BLOCK_SIZE_15): Maximum block size in source transfer widths is 15 for channel 2 - 0x3 (MAX_BLOCK_SIZE_31): Maximum block size in source transfer widths is 31 for channel 2 - 0x4 (MAX_BLOCK_SIZE_63): Maximum block size in source transfer widths is 63 for channel 2 - 0x5 (MAX_BLOCK_SIZE_127): Maximum block size in source transfer widths is 127 for channel 2 - 0x6 (MAX_BLOCK_SIZE_255): Maximum block size in source transfer widths is 255 for channel 2 - 0x7 (MAX_BLOCK_SIZE_511): Maximum block size in source transfer widths is 511 for channel 2 - 0x8 (MAX_BLOCK_SIZE_1023): Maximum block size in source transfer widths is 1023 for channel 2 - 0x9 (MAX_BLOCK_SIZE_2047): Maximum block size in source transfer widths is 2047 for channel 2 - 0xa (MAX_BLOCK_SIZE_4095): Maximum block size in source transfer widths is 4095 for channel 2 **Value after reset:** 0x3 **Volatile:** true

7 :4	R	**CH1_MAX_BLK_SIZE** Reserved field - read-only **Values:** - 0x0 (MAX_BLOCK_SIZE_3): Maximum block size in source transfer widths is 3 for channel 1 - 0x1 (MAX_BLOCK_SIZE_7): Maximum block size in source transfer widths is 7 for channel 1 - 0x2 (MAX_BLOCK_SIZE_15): Maximum block size in source transfer widths is 15 for channel 1 - 0x3 (MAX_BLOCK_SIZE_31): Maximum block size in source transfer widths is 31 for channel 1 - 0x4 (MAX_BLOCK_SIZE_63): Maximum block size in source transfer widths is 63 for channel 1 - 0x5 (MAX_BLOCK_SIZE_127): Maximum block size in source transfer widths is 127 for channel 1 - 0x6 (MAX_BLOCK_SIZE_255): Maximum block size in source transfer widths is 255 for channel 1 - 0x7 (MAX_BLOCK_SIZE_511): Maximum block size in source transfer widths is 511 for channel 1 - 0x8 (MAX_BLOCK_SIZE_1023): Maximum block size in source transfer widths is 1023 for channel 1 - 0x9 (MAX_BLOCK_SIZE_2047): Maximum block size in source transfer widths is 2047 for channel 1 - 0xa (MAX_BLOCK_SIZE_4095): Maximum block size in source transfer widths is 4095 for channel 1 **Value after reset:** 0x3 **Volatile:** true

3 :0	R	**CH0_MAX_BLK_SIZE** Reserved field - read-only **Values:** - 0x0 (MAX_BLOCK_SIZE_3): Maximum block size in source transfer widths is 3 for channel 0 - 0x1 (MAX_BLOCK_SIZE_7): Maximum block size in source transfer widths is 7 for channel 0 - 0x2 (MAX_BLOCK_SIZE_15): Maximum block size in source transfer widths is 15 for channel 0 - 0x3 (MAX_BLOCK_SIZE_31): Maximum block size in source transfer widths is 31 for channel 0 - 0x4 (MAX_BLOCK_SIZE_63): Maximum block size in source transfer widths is 63 for channel 0 - 0x5 (MAX_BLOCK_SIZE_127): Maximum block size in source transfer widths is 127 for channel 0 - 0x6 (MAX_BLOCK_SIZE_255): Maximum block size in source transfer widths is 255 for channel 0 - 0x7 (MAX_BLOCK_SIZE_511): Maximum block size in source transfer widths is 511 for channel 0 - 0x8 (MAX_BLOCK_SIZE_1023): Maximum block size in source transfer widths is 1023 for channel 0 - 0x9 (MAX_BLOCK_SIZE_2047): Maximum block size in source transfer widths is 2047 for channel 0 - 0xa (MAX_BLOCK_SIZE_4095): Maximum block size in source transfer widths is 4095 for channel 0 **Value after reset:** 0x3 **Volatile:** true

Offset: 3F8h		**DmaCompsID: DMA Component ID register**	**Init = 0**
63:32	R	**DMA_COMP_VERSION** DMA Component Version **Value after reset:** 0x3232312A **Volatile:** true	
31:0	RW	**DMA_COMP_TYPE** DMA Component Type Number = 0x44571110. This assigned unique hex value is constant and is derived from the two ASCII letters ”DW” followed by a 32-bit unsigned number **Value after reset:** 0x44571110 **Volatile:** true	

56 PECI Controller (PECI)

56.1 Overview

PECI Controller (PECI) supports PECI 1.1, 2.0, 3.0 and 4.0 protocols.

PECI totally implements 56 sets of 32-bit registers, which are listed below, to program the various supported functions. Each register has its own specific offset value, ranging from 0x00 to 0x5C and from 0x80 to 0xFC, to derive its physical address location.

Base Address of PECI = 0x1E78_B000
Physical address of register = (Base address of PECI) + Offset

PECI00: Control Register
PECI04: Timing Negotiation Register
PECI08: Command Register
PECI0C: Read/Write Length Register
PECI10: Expected FCS Data Register
PECI14: Captured FCS Data Register
PECI18: Interrupt Register
PECI1C: Interrupt Status Register
PECI20: Write Data Register #0
PECI24: Write Data Register #1
PECI28: Write Data Register #2
PECI2C: Write Data Register #3
PECI30: Read Data Register #0
PECI34: Read Data Register #1
PECI38: Read Data Register #2
PECI3C: Read Data Register #3
PECI40: Write Data Register #4
PECI44: Write Data Register #5
PECI48: Write Data Register #6
PECI4C: Write Data Register #7
PECI50: Read Data Register #4
PECI54: Read Data Register #5
PECI58: Read Data Register #6
PECI5C: Read Data Register #7
PECI80: Write Data Register e0
PECI84: Write Data Register e1
PECI88: Write Data Register e2
PECI8C: Write Data Register e3
PECI90: Write Data Register e4
PECI94: Write Data Register e5
PECI98: Write Data Register e6
PECI9C: Write Data Register e7
PECIA0: Write Data Register e8
PECIA4: Write Data Register e9
PECIA8: Write Data Register eA
PECIAC: Write Data Register eB
PECIB0: Write Data Register eC
PECIB4: Write Data Register eD
PECIB8: Write Data Register eE
PECIBC: Write Data Register eF
PECIC0: Read Data Register e0
PECIC4: Read Data Register e1

PECIC8: Read Data Register e2
PECICC: Read Data Register e3
PECID0: Read Data Register e4
PECID4: Read Data Register e5
PECID8: Read Data Register e6
PECIDC: Read Data Register e7
PECIE0: Read Data Register e8
PECIE4: Read Data Register e9
PECIE8: Read Data Register eA
PECIEC: Read Data Register eB
PECIF0: Read Data Register eC
PECIF4: Read Data Register eD
PECIF8: Read Data Register eE
PECIFC: Read Data Register eF

56.2 Features

- Directly connected to APB bus
- Intel PECI 4.0/3.0/2.0/1.1 compliant
- Support up to 8 CPU and 2 domains per CPU

56.3 Registers : Base Address = 0x1E78:B000

Offset: 00h	PECI00: Control Register	Init = 0x00000000
Bit	**R/W**	**Description**
31:25	RO	**Reserved (0)**
24	RW	**Disable the provention of possible side effects when PECI power turns off.**
23:20	RW	**I/O driving strength**
19:16	RW	**Read sampling point selection** 0000: 0/16 0001: 1/16 0010: 2/16 0011: 3/16 ... 1111: 15/16 This register is only applied to Point Sampling mode. The whole period of a bit time will be divided into 16 time frames. This register will determine which time frame this controller will sample PECI signal for data read back. Usually in the middle of a bit time is the best sample point.
15:14	RW	**Reserved**
13:12	RW	**Read mode selection** 00: Point Sampling mode 01: Pulse Width Counting mode 10: Debugging mode 11: Invalid PECI supports two kinds of read mode selections. They are Point Sample mode and Pulse Width Counting Mode. Debugging mode is for debugging purpose. It can only be applied to ping command.
11	RW	**Clock source selection** 0: from 25MHz oscillator 1: from HCLK
10:8	RW	**PECI clock divider** 000: Divided by 1 001: Divided by 2 010: Divided by 4 011: Divided by 8 ... 111: Divided by 128 This register will determine the operation frequency of PECI Controller. The input clock source is from 25MHz oscillator or HCLK. Message frequency of PECI controller is roughly about PECI clock / 4 / (PECI04[15:8] * 4 + 1).
7	RW	**Inverse PECI output polarity** 0: Normal polarity 1: Inverse polarity
6	RW	**Inverse PECI input polarity** 0: Normal polarity 1: Inverse polarity
5	RW	**Enable bus contention** 0: Disable 1: Enable

to next page

from previous page

4	RW	**Enable PECI** 0: Disable 1: Enable
3:2	RW	**Reserved**
1	RW	**Enable 64-byte mode**
0	RW	**Enable PECI clock** 0: Disable PECI clock 1: Enable PECI clock This register will stop or enable 25MHz clock source for power saving.

Offset: 04h **PECI04: Timing Negotiation Register** **Init = X**

Bit	R/W	Description
31:16	RO	**Reserved (0)**
15:8	RW	**address timing negotiation tBIT-2 and message timing negotiation tBIT-M** This register will determine the period of address timing negotiation tBIT-2 and message timing negotiation tBIT-M to be issued by PECI Controller. The unit of the programmed value is four times of PECI clock period.
7:0	RW	**address timing negotiation tBIT-1** This register will determine the period of address timing negotiation tBIT-1 to be issued by PECI Controller. The unit of the programmed value is four times of PECI clock period.

Note :
PECI04 is timeout interrupt. When peci enter timing negotiation state, the peci controller will trigger the timer to count and set PECI04 when timer timeout.

Offset: 08h **PECI08: Command Register** **Init = 0x80000000**

Bit	R/W	Description
31	RO	**PECI pin monitoring** This bit can read back the signal status of PECI pin.
30	RO	**raw PECI pin monitoring** This bit can read back the signal status of raw PECI pin.
29:28	RO	**Reserved (0)**
27:24	RO	**PECI Controller state** 0000: PECI Controller is in idle state 0001: Fire state 0010: Initial address timing negotiation state 0011: Address timing negotiation state 0100: Address state 0101: Message timing negotiation state 0110: Write/read length state 0111: Write data state 1000: Reserved 1001: Write FCS state 1010: Read data state 1011: Read FCS state 1100: Stop state others: Reserved
23:1	RO	**Reserved (0)**

to next page

from previous page

0	RW	**Fire a PECI command** 0: No operation 1: Fire a PECI command

Offset: 0Ch **PECI0C: Read/Write Length Register** **Init = X**

Bit	R/W	Description
31	RW	**Enable AW FCS cycle** 0: Disable AW FCS cycle 1: Enable AW FCS cycle This register is only applied to PECI write commands. When enabled, PECI command will be with AW FCS cycle.
30:24	RO	**Reserved (0)**
23:16	RW	**Read data length (bytes)** This register determines the number of bytes to be read. The read back data will be stored in the following registers: PECI30-PECI3C and PECI50-PECI5C or PECIC0-PECIFC. Although this register is 8 bits, the maximum read data length is only 64 bytes.
15:8	RW	**Write data length (bytes)** This register determines the number of bytes to be written. The data to be written has to be pre-stored in the following registers: PECI20-PECI2C and PECI40-PECI4C or PECI80-PECIBC.
7 :0	RW	**Target address** This register determines the 8-bit address of the PECI command to be fired.

Offset: 10h **PECI10: Expected FCS Data Register** **Init = X**

Bit	R/W	Description
31:24	RW	**Write data following PECIBC**
23:16	RO	**Expected read FCS** This register contains the FCS data generated by the internal hardware logic. This is for debugging purpose only.
15:8	RO	**Expected auto AW FCS** This register contains the AW FCS data generated by the internal hardware logic. This is for debugging purpose only.
7 :0	RO	**Expected write FCS** This register contains the FCS data generated by the internal hardware logic. This is for debugging purpose only.

Offset: 14h **PECI14: Captured FCS Data Register** **Init = X**

Bit	R/W	Description
31:24	RO	**Reserved (0)**
23:16	RO	**Captured FCS data from PECI data read command** This register contains the 8-bit FCS data captured by PECI Controller from a PECI data read command.
15:8	RO	**Reserved (0)**
7 :0	RO	**Captured FCS data from PECI write command** This register contains the 8-bit FCS data captured by PECI Controller from a PECI data write command.

Offset: 18h	PECI18: Interrupt Register	Init = 0xX0000000

Bit	R/W	Description
31:30	RW	**Selection of timing negotiation result bit [1:0]** 00: 1st bit of address negotiation 01: 2nd bit of address negotiation 10: message negotiation 11: reserved
29:5	RO	**Reserved (0)**
4	RW	**Enable PECI bus time-out interrupt** 0: Disable 1: Enable
3	RW	**Enable PECI bus contention interrupt** 0: Disable 1: Enable
2	RW	**Enable PECI write FCS bad interrupt** 0: Disable 1: Enable
1	RW	**Enable PECI write FCS abort interrupt** 0: Disable 1: Enable
0	RW	**Enable PECI command done interrupt** 0: Disable 1: Enable

Offset: 1Ch	PECI1C: Interrupt Status Register	Init = 0xXXXXX000

Bit	R/W	Description
31:30	RO	**Reserved (0)**
29:16	RO	**Timing negotiation result** Negotiated PECI Speed is roughly about PECI clock / 4 / (PECI1C[29:16] + 1)
15	RO	**Good1FCS status**
14:5	RO	**Reserved (0)**
4	RW	**PECI bus time-out interrupt status** 0: No interrupt 1: Interrupt is pending Writing ”1” will clear this status. When PECI controller enters a timing negotiation state, it will trigger a timer to count and set this bit when a timeout occurs (over PECI spec. Max limit).
3	RW	**PECI bus contention interrupt status** 0: No interrupt 1: Interrupt is pending Writing ”1” will clear this status.
2	RW	**PECI write FCS bad interrupt status** 0: No interrupt 1: Interrupt is pending Writing ”1” will clear this status.
1	RW	**PECI write FCS abort interrupt status** 0: No interrupt 1: Interrupt is pending Writing ”1” will clear this status.

to next page

from previous page

0	RW	**PECI done interrupt status** 0: No interrupt 1: Interrupt is pending Writing ”1” will clear this status.

Offset: 20h **PECI20: Write Data Register #0** **Init = X**

Bit	R/W	Description
31:0	RW	**Write data byte 4 to byte 1 if PECI00[1]=0**

Offset: 24h **PECI24: Write Data Register #1** **Init = X**

Bit	R/W	Description
31:0	RW	**Write data byte 8 to byte 5 if PECI00[1]=0**

Offset: 28h **PECI28: Write Data Register #2** **Init = X**

Bit	R/W	Description
31:0	RW	**Write data byte 12 to byte 9 if PECI00[1]=0**

Offset: 2Ch **PECI2C: Write Data Register #3** **Init = X**

Bit	R/W	Description
31:0	RW	**Write data byte 16 to byte 13 if PECI00[1]=0**

Offset: 30h **PECI30: Read Data Register #0** **Init = X**

Bit	R/W	Description
31:0	RO	**Read data byte 4 to byte 1**

Offset: 34h **PECI34: Read Data Register #1** **Init = X**

Bit	R/W	Description
31:0	RO	**Read data byte 8 to byte 5**

Offset: 38h **PECI38: Read Data Register #2** **Init = X**

Bit	R/W	Description
31:0	RO	**Read data byte 12 to byte 9**

Offset: 3Ch **PECI3C: Read Data Register #3** **Init = X**

Bit	R/W	Description
31:0	RO	**Read data byte 16 to byte 13**

Offset: 40h	PECI40: Write Data Register #4	Init = X
Bit	**R/W**	**Description**
31:0	RW	**Write data byte 20 to byte 17 if PECI00[1]=0**

Offset: 44h	PECI44: Write Data Register #5	Init = X
Bit	**R/W**	**Description**
31:0	RW	**Write data byte 24 to byte 21 if PECI00[1]=0**

Offset: 48h	PECI48: Write Data Register #6	Init = X
Bit	**R/W**	**Description**
31:0	RW	**Write data byte 28 to byte 25 if PECI00[1]=0**

Offset: 4Ch	PECI4C: Write Data Register #7	Init = X
Bit	**R/W**	**Description**
31:0	RW	**Write data byte 32 to byte 29 if PECI00[1]=0**

Offset: 50h	PECI50: Read Data Register #4	Init = X
Bit	**R/W**	**Description**
31:0	RO	**Read data byte 20 to byte 17**

Offset: 54h	PECI54: Read Data Register #5	Init = X
Bit	**R/W**	**Description**
31:0	RO	**Read data byte 24 to byte 21**

Offset: 58h	PECI58: Read Data Register #6	Init = X
Bit	**R/W**	**Description**
31:0	RO	**Read data byte 28 to byte 25**

Offset: 5Ch	PECI5C: Read Data Register #7	Init = X
Bit	**R/W**	**Description**
31:0	RO	**Read data byte 32 to byte 29**

Offset: 80h	PECI80: Write Data Register e0	Init = X
Bit	**R/W**	**Description**
31:0	RW	**Write data byte 4 to byte 1 if PECI00[1]=1**

Offset: 84h	PECI84: Write Data Register e1	Init = X
Bit	**R/W**	**Description**
31:0	RW	**Write data byte 8 to byte 5 if PECI00[1]=1**

Offset: 88h	PECI88: Write Data Register e2	Init = X
Bit	**R/W**	**Description**
31:0	RW	**Write data byte 12 to byte 9 if PECI00[1]=1**

Offset: 8Ch	PECI8C: Write Data Register e3	Init = X
Bit	**R/W**	**Description**
31:0	RW	**Write data byte 16 to byte 13 if PECI00[1]=1**

Offset: 90h	PECI90: Write Data Register e4	Init = X
Bit	**R/W**	**Description**
31:0	RW	**Write data byte 20 to byte 17 if PECI00[1]=1**

Offset: 94h	PECI94: Write Data Register e5	Init = X
Bit	**R/W**	**Description**
31:0	RW	**Write data byte 24 to byte 21 if PECI00[1]=1**

Offset: 98h	PECI98: Write Data Register e6	Init = X
Bit	**R/W**	**Description**
31:0	RW	**Write data byte 28 to byte 25 if PECI00[1]=1**

Offset: 9Ch	PECI9C: Write Data Register e7	Init = X
Bit	**R/W**	**Description**
31:0	RW	**Write data byte 32 to byte 29 if PECI00[1]=1**

Offset: A0h	PECIA0: Write Data Register e8	Init = X
Bit	**R/W**	**Description**
31:0	RW	**Write data byte 36 to byte 33 if PECI00[1]=1**

Offset: A4h	PECIA4: Write Data Register e9	Init = X
Bit	**R/W**	**Description**
31:0	RW	**Write data byte 40 to byte 37 if PECI00[1]=1**

Offset: A8h	PECIA8: Write Data Register eA	Init = X
Bit	**R/W**	**Description**
31:0	RW	**Write data byte 44 to byte 41 if PECI00[1]=1**

Offset: ACh	PECIAC: Write Data Register eB	Init = X
Bit	**R/W**	**Description**
31:0	RW	**Write data byte 48 to byte 45 if PECI00[1]=1**

Offset: B0h	PECIB0: Write Data Register eC	Init = X
Bit	**R/W**	**Description**
31:0	RW	**Write data byte 52 to byte 49 if PECI00[1]=1**

Offset: B4h	PECIB4: Write Data Register eD	Init = X
Bit	**R/W**	**Description**
31:0	RW	**Write data byte 56 to byte 53 if PECI00[1]=1**

Offset: B8h	PECIB8: Write Data Register eE	Init = X
Bit	**R/W**	**Description**
31:0	RW	**Write data byte 60 to byte 57 if PECI00[1]=1**

Offset: BCh	PECIBC: Write Data Register eF	Init = X
Bit	**R/W**	**Description**
31:0	RW	**Write data byte 64 to byte 61 if PECI00[1]=1**

Offset: C0h	PECIC0: Read Data Register e0	Init = X
Bit	**R/W**	**Description**
31:0	RW	**Read data byte 4 to byte 1**

Offset: C4h	PECIC4: Read Data Register e1	Init = X
Bit	**R/W**	**Description**
31:0	RW	**Read data byte 8 to byte 5**

Offset: C8h	PECIC8: Read Data Register e2	Init = X
Bit	**R/W**	**Description**
31:0	RW	**Read data byte 12 to byte 9**

Offset: CCh	PECICC: Read Data Register e3	Init = X
Bit	**R/W**	**Description**
31:0	RW	**Read data byte 16 to byte 13**

Offset: D0h	PECID0: Read Data Register e4	Init = X
Bit	**R/W**	**Description**
31:0	RW	**Read data byte 20 to byte 17**

Offset: D4h	PECID4: Read Data Register e5	Init = X
Bit	**R/W**	**Description**
31:0	RW	**Read data byte 24 to byte 21**

Offset: D8h	PECID8: Read Data Register e6	Init = X
Bit	**R/W**	**Description**
31:0	RW	**Read data byte 28 to byte 25**

Offset: DCh	PECIDC: Read Data Register e7	Init = X
Bit	**R/W**	**Description**
31:0	RW	**Read data byte 32 to byte 29**

Offset: E0h	PECIE0: Read Data Register e8	Init = X
Bit	**R/W**	**Description**
31:0	RW	**Read data byte 36 to byte 33**

Offset: E4h	PECIE4: Read Data Register e9	Init = X
Bit	**R/W**	**Description**
31:0	RW	**Read data byte 40 to byte 37**

Offset: E8h	PECIE8: Read Data Register eA	Init = X
Bit	**R/W**	**Description**
31:0	RW	**Read data byte 44 to byte 41**

Offset: ECh	PECIEC: Read Data Register eB	Init = X
Bit	**R/W**	**Description**
31:0	RW	**Read data byte 48 to byte 45**

Offset: F0h	PECIF0: Read Data Register eC	Init = X
Bit	**R/W**	**Description**
31:0	RW	**Read data byte 52 to byte 49**

Offset: F4h	PECIF4: Read Data Register eD	Init = X
Bit	**R/W**	**Description**
31:0	RW	**Read data byte 56 to byte 53**

Offset: F8h	PECIF8: Read Data Register eE	Init = X
Bit	**R/W**	**Description**
31:0	RW	**Read data byte 60 to byte 57**

Offset: FCh	PECIFC: Read Data Register eF	Init = X
Bit	**R/W**	**Description**
31:0	RW	**Read data byte 64 to byte 61**

57 PCIe Host Controller (PCIEH)

57.1 Overview

Base Address of PCIe RC/Bridge General Controller = 0x1E6E_D000
Base Address of PCIe RC General Controller = 0x1E6E_D200
Physical address of register = (Base address of PCIe General Controller) + Offset

PEHR00: Device and Vendor ID Register
PEHR04: Class Code Revision Register
PEHR10: Miscellaneous Control 10h Register
PEHR14: Miscellaneous Control 14h Register
PEHR30: Miscellaneous Control 30h Register
PEHR50: Miscellaneous Control 50h Register
PEHR58: Miscellaneous Control 58h Register
PEHR2C: Miscellaneous Control 2Ch Register
PEHR7C: Protection Key Register
PEHRC0: Miscellaneous Status C0h Register
PEHRC4: Miscellaneous Status C4h Register
PEHRD0: Miscellaneous Status D0h Link Register

57.2 Features

PCIe General Controllers

- One PCIe Gen2 x1 Root Complex or Bridge Controller (mutually exclusive)
- One PCIe Gen2 x1 Root Complex Controller
- Support configuration, I/O, 32-bit memory, and message transactions
- Support INTX or MSI (number 32)

57.3 Registers : Base Address = 0x1E6E:D000

Offset: 00h	PEHR00: Device and Vendor ID Register	Init = 0x11501A03
Bit	**R/W**	**Description**
31:16	RW	**Device ID**
15: 0	RW	**Vendor ID**

Offset: 04h	PEHR04: Class Code Revision Register	Init = 0x06040006
Bit	**R/W**	**Description**
31: 8	RW	**Class Code**
7: 0	RW	**Revision** 04: A0 05: A1 and A2 06: A3

Offset: 10h	PEHR10: Miscellaneous Control 10h Register	Init = 0xD7040022
Bit	**R/W**	**Description**
4	RW	**Enable data link layer link active reporting capable**

Offset: 14h	PEHR14: Miscellaneous Control 14h Register	Init = 0x0
Bit	**R/W**	**Description**
0	RW	**Enable attention button present register accessible**
5	RW	**Enable hot-plug surprise register accessible**
6	RW	**Enable hot-plug capable register accessible**

Offset: 2Ch	PEHR2C: Miscellaneous Control 2Ch Register	Init = 0x500464FF
Bit	**R/W**	**Description**
8	RW	**Enable selecting de-emphasis of 3.5 dB (for RC only)**

Offset: 30h	PEHR30: Miscellaneous Control 30h Register	Init = 0x0
Bit	**R/W**	**Description**
20	RW	**Disable RC synchronous reset when link up to link down**
5: 4	RW	**Port type bit [1:0]** 00: bridge 11: rootport others: reserved
1	RW	**Enable RC slot implemented**

Offset: 34h	PEHR34: Miscellaneous Control 34h Register	Init = 0x0
Bit	**R/W**	**Description**
26	RW	**Gen1 only**

Offset: 7Ch	PEHR7C: Protection Key Register	Init = 0x0
Bit	**R/W**	**Description**
7: 0	WT	**unlock by writing 0xA8 and lock by writing others**
0	RO	**unlock status**

Offset: C0h	PEHRC0: Miscellaneous Status C0h Register	Init = 0xX
Bit	**R/W**	**Description**
5	RO	**DLCMSM: DL_Up**

Offset: C4h	PEHRC4: Miscellaneous Status C4h Register	Init = 0xX
Bit	**R/W**	**Description**
19	RO	**PERST#**
12: 5	RO	**Bridge bus number**
4: 0	RO	**Bridge device number**

Offset: D0h	PEHRD0: Miscellaneous Status D0h Link Register	Init = 0xX

Bit	R/W	Description
20	RO	**link width x1 when link up**
17	RO	**link speed 5.0G when link up**
16	RO	**link speed 2.5G when link up**

58 Internal Bridge Controller (SLI)

58.1 Overview

Base address of SLI_Host = 0x1E6F_3000
Base address of SLI_Client = 0x1E6F_3800
Physical address = (Base address of SLI) + Offset

SLI00: Interrupt Status Register
SLI04: Interrupt Enable Register
SLI08: Decoder AHB Short Header Base Address Register
SLI0C: Encoder AHB Short Header Base Address Register
SLI10: SLI Engine Control
SLI14: SLI Clock Delay Stage
SLI20: SLI Encoder Pattern Data
SLI24: SLI Pattern Delay Stage
SLI28: SLI Decoder Pattern Data
SLI2C: SLI Pattern Control
SLI30: SLI SRR Control
SLI34: SLI SRR Weight #0~#3
SLI38: SLI Read Data Size Control
SLI3Ch: SLI Host Statistic Control
SLI3Cc: SLI Client M-Bus Aribtration Grant Length
SLI40: SLI Host Decoder Timer Counter
SLI44: SLI Host Decoder M-Bus Statistic Counter
SLI48: SLI Host Decoder Header Statistic Counter
SLI4C: SLI Host Decoder AHB Statistic Counter
SLI50: SLI Host Encoder Timer Counter
SLI54: SLI Host Encoder M-Bus Statistic Counter
SLI58: SLI Host Encoder Header Statistic Counter
SLI5C: SLI Host Encoder AHB Statistic Counter

58.2 Registers : Base Address = 0x1E6E:3000/0x1E6E:3080

Offset: 00h	SLI00: Interrupt Status Register	Init = 0
Bit	**R/W**	**Description**
		SLI Host
31:22	RO	**Reserved (0)**
21	RO	**(MBus : Mbus_Dat_Wr) Mbus data buffer write full**
20	RO	**(SLIDec: Mbus_WrRqH_OddDat) Mbus write request header with odd data**
19	RO	**(SLIDec: Mbus_Len_Wr) Mbus length buffer write full**
18	RO	**(SLIDec: Mbus_Crd_Wr) Mbus credit buffer write full**
17	RO	**Reserved (0)**
		SLI Client
31:22	RO	**Reserved (0)**
21	RO	**(MBus : Mbus_Inf_Rd) Mbus information buffer read empty and the Mbus complete buffer is ready for read.**
20	RO	**Reserved (0)**
19	RO	**(SLIDec: Mbus_ComH_OddDat) Mbus complete header with odd data**
18	RO	**(SLIDec: Mbus_Com_Wr) Mbus complete buffer write full**
17	RO	**(SLIDec: AMas_CrD_Wr) AHB master credit data buffer write full**
16	RO	**(SLIDec: AMas_CrH_Wr) AHB master credit header buffer write full**
15	RO	**(SLIDec: ASlv_CoD_Wr) AHB slave complete data buffer write full**
14	RO	**(SLIEnc: SRR_Arb_Err) SRR arbiter error**
13	RO	**(SLIEnc: MRq_Len_Err) Mbus read request length overflow**
12	RO	**(SLIEnc: AMas_Dat_Rd) AHB master data buffer read empty**
11	RO	**(SLIEnc: ASlv_WrD_Rd) AHB slave request data buffer read empty**
10	RO	**(AHBMas: AMas_Dat_Wr) AHB master data buffer write full**
9	RO	**(AHBMas: AMas_Len_Wr) AHB master length buffer write full**
8	RO	**(AHBMas: AMas_CrD_Rd) AHB master credit data buffer read empty**
7	RO	**(AHBSlv: ASlv_Mg_Ful) AHB slave write request merge with buffer full**
6	RO	**(AHBSlv: ASlv_Spl_Ctl) AHB slave split control error**
5	RO	**(AHBSlv: AArb_SplitBack) AHB arbiter split back error**
4	RO	**(AHBSlv: ASlv_CoD_Rd) AHB slave complete data buffer read empty**
3	RO	**(AHBSlv: ASlv_WrD_Wr) AHB slave request data buffer write full**
2	RO	**(AHBSlv: ASlv_WrH_Wr) AHB slave request header buffer write full**
1	RO	**(AHBSlv: ASlv_CoH_Wr) AHB slave complete header buffer write full**
0	RO	**(AHBSlv: ASlv_Spl_Wr) AHB slave split buffer write full**

Offset: 04h	SLI04: Interrupt Enable Register	Init = 000f_ffffh
Bit	**R/W**	**Description**
		SLI Host
31:21	RO	**Reserved (0)**
20:14	RW	**Interrupt enable of SLI00 Bit20~14**
13	RO	**Reserved (0)**

to next page

from previous page

12: 0	RW	**Interrupt enable of SLI00 Bit12~0**
SLI Client		
31:21	RO	**Reserved (0)**
20	RW	**Interrupt enable of SLI00 Bit20**
19	RO	**Reserved (0)**
18: 0	RW	**Interrupt enable of SLI00 Bit18~0**

Offset: 08h **SLI08: Decoder AHB Short Header Base Address Register** **Init = X**

Bit	R/W	Description
31:24	RW	**Decoder AHB short header base address[31:24]**
23:0	RO	**Reserved (0)**

Offset: 0Ch **SLI0C: Encoder AHB Short Header Base Address Register** **Init = X**

Bit	R/W	Description
31:24	RW	**Encoder AHB short header base address[31:24]**
23:0	RO	**Reserved (0)**

Offset: 10h **SLI10: SLI Engine Control** **Init = 0**

Bit	R/W	Description
SLI Client		
31:6	RO	**Reserved (0)**
5	RW	**M-Bus encoder aribtration grant length control** 0: Disable grant length control 1: Enable grant length control
4	RW	**M-Bus encoder aribtration priority selection** 0: The aribtration is **fixed** priority (default) 1: The aribtration is round-robin priority
SLI Host		
31: 4	RO	**Reserved (0)**
3	RW	**SLI encoder aribtration priority selection** 0: The aribtration is **round-robin** priority (default) 1: The aribtration is fixed priority
2	RW	**Disable DBI**
1	RW	**Enable SRR**
0	RW	**Enable Encoder AHB Short Header Base Address Register (SLI0C)**

Offset: 14h **SLI14: SLI Clock Delay Stage** **Init = 0**

Bit	R/W	Description
31:14	RO	**Reserved (0)**
13	RW	**SLI Output Clock Delay Invert**
12:8	RW	**SLI Output Clock Delay Stage**
7	RW	**Reserved (0)**
6	RW	**SLI Input Engine Clock Invert**

to next page

from previous page

5	RW	**SLI Input Clock Delay Invert**
4 :0	RW	**SLI Input Clock Delay Stage**

Offset: 20h	**SLI20: SLI Encoder Pattern Data**	**Init = X**
Bit	**R/W**	**Description**
31:0	RW	**SLI Encoder Pattern Data**

Offset: 24h	**SLI24: SLI Pattern Delay Stage**	**Init = X**
Bit	**R/W**	**Description**
31:8	RO	**Reserved (0)**
7	RW	**SLI Delay Stage Setting Selection** 0: Setting SLI delay stage(SLI24[6:0]) to input clock tree 1: Setting SLI delay stage(SLI24[5:0]) to output clock tree
6 :0	RW	**SLI Delay Stage**

Offset: 28h	**SLI28: SLI Decoder Pattern Data**	**Init = X**
Bit	**R/W**	**Description**
31:0	RO	**SLI Decoder Pattern Data**

Offset: 2Ch	**SLI2C: SLI Pattern Control**	**Init = 0**
Bit	**R/W**	**Description**
31	RO	**Patern Busy Status** 0: IDLE 1: The pattern is running
30	RW	**Patern Time-out Status** 0: Normal status 1: The pattern is time-out
29	RW	**Patern Mismatch Status** 0: Normal status 1: The pattern are mismatched with the expected value
28	RO	**Link-Up Status(HCLK)** 0: The SLI is link-down 1: The SLI is link-up
27	RO	**Link-Up Status(SICLK)** 0: The SLI is link-down 1: The SLI is link-up
26	RO	**Link-Up Status(SOCLK)** 0: The SLI is link-down 1: The SLI is link-up
25	RO	**Link-Up Status(MCLK)** 0: The SLI is link-down 1: The SLI is link-up
24:2	RO	**Reserved (0)**

to next page

from previous page

1 :0	RW	**SLI Pattern Type Selection** 0: The link-up pattern 1: The delay stage pattern 2: The automatic pattern 3: The single testing pattern The pattern is 64 clock cycles.

Offset: 30h **SLI30: SLI SRR Control** **Init = 0**

Bit	R/W	Description
31:8	RO	**Reserved (0)**
7 :3	RO	**SLI SRR Counter Initial Value**
2 :0	RW	**SLI SRR(Shared or Shaped Round Robin) Type Selection**

Offset: 34h **SLI34: SLI SRR Weight #0~#3** **Init = ffff_ffffh**

Bit	R/W	Description
31:24	RW	**SLI SRR Weight #3**
23:16	RW	**SLI SRR Weight #2**
15:8	RW	**SLI SRR Weight #1**
7 :0	RW	**SLI SRR Weight #0**

Offset: 38h **SLI38: SLI Read Data Size Control** **Init = 0**

Bit	R/W	Description
31:30	RW	**SLI read data size for the incrementing burst (INCR) of the AHB master #15**
29:28	RW	**SLI read data size for the incrementing burst (INCR) of the AHB master #14**
27:26	RW	**SLI read data size for the incrementing burst (INCR) of the AHB master #13**
25:24	RW	**SLI read data size for the incrementing burst (INCR) of the AHB master #12**
23:22	RW	**SLI read data size for the incrementing burst (INCR) of the AHB master #11**
21:20	RW	**SLI read data size for the incrementing burst (INCR) of the AHB master #10**
19:18	RW	**SLI read data size for the incrementing burst (INCR) of the AHB master #9**
17:16	RW	**SLI read data size for the incrementing burst (INCR) of the AHB master #8**
15:14	RW	**SLI read data size for the incrementing burst (INCR) of the AHB master #7**
13:12	RW	**SLI read data size for the incrementing burst (INCR) of the AHB master #6**
11:10	RW	**SLI read data size for the incrementing burst (INCR) of the AHB master #5**
9 :8	RW	**SLI read data size for the incrementing burst (INCR) of the AHB master #4**
7 :6	RW	**SLI read data size for the incrementing burst (INCR) of the AHB master #3**
5 :4	RW	**SLI read data size for the incrementing burst (INCR) of the AHB master #2**
3 :2	RW	**SLI read data size for the incrementing burst (INCR) of the AHB master #1**
1 :0	RW	**SLI read data size for the incrementing burst (INCR) of the AHB master #0** 0: 1 data 1: 4 data 2: 8 data 3: 16 data

Offset: 3Ch	SLI3Ch: SLI Host Statistic Control	Init = 0
Bit	**R/W**	**Description**
		SLI Host
31:0	RW	**SLI statistic timer** (for debugging purpose only) The SLI will run (SLI3Ch[31:0] + 1) clock cycle to do the statistic. When the statistic is finished, this timer will be reset to value 0.

Offset: 3Ch	SLI3Cc: SLI Client M-Bus Aribtration Grant Length	Init = 0
Bit	**R/W**	**Description**
		SLI Client
31:24	RO	**Reserved (0)**
23:20	RW	**M-Bus encoder aribtration grant length selection for MRrq #5**
19:16	RW	**M-Bus encoder aribtration grant length selection for MRrq #4**
15:12	RW	**M-Bus encoder aribtration grant length selection for MRrq #3**
11:8	RW	**M-Bus encoder aribtration grant length selection for MRrq #2**
7 :4	RW	**M-Bus encoder aribtration grant length selection for MRrq #1**
3 :0	RW	**M-Bus encoder aribtration grant length selection for MRrq #0** 0: 4 data 1: 8 data 2: 12 data 3: 16 data ... 15: 64 data

Offset: 40h	SLI40: SLI Host Decoder Timer Counter	Init = 0
Offset: 44h	SLI44: SLI Host Decoder M-Bus Statistic Counter	Init = 0
Offset: 48h	SLI48: SLI Host Decoder Header Statistic Counter	Init = 0
Offset: 4Ch	SLI4C: SLI Host Decoder AHB Statistic Counter	Init = 0
Offset: 50h	SLI50: SLI Host Encoder Timer Counter	Init = 0
Offset: 54h	SLI54: SLI Host Encoder M-Bus Statistic Counter	Init = 0
Offset: 58h	SLI58: SLI Host Encoder Header Statistic Counter	Init = 0
Offset: 5Ch	SLI5C: SLI Host Encoder AHB Statistic Counter	Init = 0
Bit	**Attr.**	**Description**
		SLI Host
31:0	RO	**Value** (for debugging purpose only)

59 DisplayPort Interface (DP)

59.1 Overview

DisplayPort Source comprises main link transmitter, aux channel tranceiver and hot plug detection

Base address of DPTX = 0x1E6E_B000
Physical address = (Base address of DPTX) + Offset

59.2 Registers : Base Address = 0x1E6E:B000

Offset: 000h	DPTX000: DPTX Interrupt to CPU Configuration Register	Init = 0x19000000
Bit	**R/W**	**Description**
31	RW	**aux_by_gpio_en** DP TX AUX channel signal path selection 0: through DP TX PHY 1: through thre GPIO pins
30	RW	**aux_clk_sel** DP TX AUX channel sampling clock source selection 0: from external 25MHz XTAL 1: from DP TX PHY
29:24	RW	**aux_clk_freq** DP TX AUX channel sampling clock frequency in MHz
23	RW	**aux_reply_timeout_int_2_cpu_en** DP TX AUX channel timeout interrupt to cpu enable
22	RW	**aux_reply_done_int_2_cpu_en** DP TX AUX channel reply done interrupt to cpu enable
21	RW	**hpd_event_int_2_cpu_en** DP TX HPD event interrupt to cpu enable
20	RW	**hpd_low_int_2_cpu_en** DP TX HPD low interrupt to cpu enable
19	RW	**hpd_irq_int_2_cpu_en** DP TX HPD irq interrupt to cpu enable
18	RW	**timer_2_int_2_cpu_en** DP TX cpu timer 2 interrupt to cpu enable
17	RW	**timer_1_int_2_cpu_en** DP TX cpu timer 1 interrupt to cpu enable
16	RW	**timer_0_int_2_cpu_en** DP TX cpu timer 0 interrupt to cpu enable
15	RW	**aux_reply_timeout_int_2_cpu_mask** DP TX AUX channel timeout interrupt to cpu mask 0: ignore interrupt raw status 1: reveal interrupt raw status
14	RW	**aux_reply_done_int_2_cpu_mask** DP TX AUX channel reply done interrupt to cpu mask 0: ignore interrupt raw status 1: reveal interrupt raw status

to next page

from previous page

13	RW	**hpd_event_int_2_cpu_mask** DP TX HPD event interrupt to cpu mask 0: ignore interrupt raw status 1: reveal interrupt raw status
12	RW	**hpd_low_int_2_cpu_mask** DP TX HPD low interrupt to cpu mask 0: ignore interrupt raw status 1: reveal interrupt raw status
11	RW	**hpd_irq_int_2_cpu_mask** DP TX HPD irq interrupt to cpu mask 0: ignore interrupt raw status 1: reveal interrupt raw status
10	RW	**timer_2_int_2_cpu_mask** DP TX cpu timer 2 interrupt to cpu mask 0: ignore interrupt raw status 1: reveal interrupt raw status
9	RW	**timer_1_int_2_cpu_mask** DP TX cpu timer 1 interrupt to cpu mask 0: ignore interrupt raw status 1: reveal interrupt raw status
8	RW	**timer_0_int_2_cpu_mask** DP TX cpu timer 0 interrupt to cpu mask 0: ignore interrupt raw status 1: reveal interrupt raw status
7	WO	**aux_reply_timeout_int_2_cpu_clr** DP TX AUX channel timeout interrupt to cpu clear this bit will go back to 0 after one clock cycle
6	WO	**aux_reply_done_int_2_cpu_clr** DP TX AUX channel reply done interrupt to cpu clear this bit will go back to 0 after one clock cycle
5	WO	**hpd_event_int_2_cpu_clr** DP TX HPD event interrupt to cpu clear this bit will go back to 0 after one clock cycle
4	WO	**hpd_low_int_2_cpu_clr** DP TX HPD low interrupt to cpu clear this bit will go back to 0 after one clock cycle
3	WO	**hpd_irq_int_2_cpu_clr** DP TX HPD irq interrupt to cpu clear this bit will go back to 0 after one clock cycle
2	WO	**timer_2_int_2_cpu_clr** DP TX cpu timer 2 interrupt to cpu clear this bit will go back to 0 after one clock cycle
1	WO	**timer_1_int_2_cpu_clr** DP TX cpu timer 1 interrupt to cpu clear this bit will go back to 0 after one clock cycle
0	WO	**timer_0_int_2_cpu_clr** DP TX cpu timer 0 interrupt to cpu clear this bit will go back to 0 after one clock cycle

Offset: 004h	DPTX004: DPTX Interrupt to CPU Status Register	Init = 0x00000000

Bit	R/W	Description
31:16	RO	**Reserved**
15	RO	**aux_reply_timeout_int_2_cpu_raw** DP TX AUX channel timeout interrupt to cpu raw status
14	RO	**aux_reply_done_int_2_cpu_raw** DP TX AUX channel reply done interrupt to cpu raw status
13	RO	**hpd_event_int_2_cpu_raw** DP TX HPD event interrupt to cpu raw status
12	RO	**hpd_low_int_2_cpu_raw** DP TX HPD low interrupt to cpu raw status
11	RO	**hpd_irq_int_2_cpu_raw** DP TX HPD irq interrupt to cpu raw status
10	RO	**timer_2_int_2_cpu_raw** DP TX cpu timer 2 interrupt to cpu raw status
9	RO	**timer_1_int_2_cpu_raw** DP TX cpu timer 1 interrupt to cpu raw status
8	RO	**timer_0_int_2_cpu_raw** DP TX cpu timer 0 interrupt to cpu raw status
7	RO	**aux_reply_timeout_int_2_cpu** DP TX AUX channel timeout interrupt to cpu
6	RO	**aux_reply_done_int_2_cpu** DP TX AUX channel reply done interrupt to cpu
5	RO	**hpd_event_int_2_cpu** DP TX HPD event interrupt to cpu
4	RO	**hpd_low_int_2_cpu** DP TX HPD low interrupt to cpu
3	RO	**hpd_irq_int_2_cpu** DP TX HPD irq interrupt to cpu
2	RO	**timer_2_int_2_cpu** DP TX cpu timer 2 interrupt to cpu
1	RO	**timer_1_int_2_cpu** DP TX cpu timer 1 interrupt to cpu
0	RO	**timer_0_int_2_cpu** DP TX cpu timer 0 interrupt to cpu

Offset: 008h	DPTX008: DPTX Software Interrupt to CPU Configuration Register	Init = 0x00FFFF00

Bit	R/W	Description
31:24	RO	**Reserved**
23	RW	**reg_bank_int_2_cpu_en[7]** DP TX software interrupt 7 to cpu enable
22	RW	**reg_bank_int_2_cpu_en[6]** DP TX software interrupt 6 to cpu enable
21	RW	**reg_bank_int_2_cpu_en[5]** DP TX software interrupt 5 to cpu enable
20	RW	**reg_bank_int_2_cpu_en[4]** DP TX software interrupt 4 to cpu enable

to next page

from previous page

19	RW	**reg_bank_int_2_cpu_en[3]** DP TX software interrupt 3 to cpu enable
18	RW	**reg_bank_int_2_cpu_en[2]** DP TX software interrupt 2 to cpu enable
17	RW	**reg_bank_int_2_cpu_en[1]** DP TX software interrupt 1 to cpu enable
16	RW	**reg_bank_int_2_cpu_en[0]** DP TX software interrupt 0 to cpu enable
15	RW	**reg_bank_int_2_cpu_mask[7]** DP TX software interrupt 7 to cpu mask 0: ignore interrupt raw status 1: reveal interrupt raw status
14	RW	**reg_bank_int_2_cpu_mask[6]** DP TX software interrupt 6 to cpu mask 0: ignore interrupt raw status 1: reveal interrupt raw status
13	RW	**reg_bank_int_2_cpu_mask[5]** DP TX software interrupt 5 to cpu mask 0: ignore interrupt raw status 1: reveal interrupt raw status
12	RW	**reg_bank_int_2_cpu_mask[4]** DP TX software interrupt 4 to cpu mask 0: ignore interrupt raw status 1: reveal interrupt raw status
11	RW	**reg_bank_int_2_cpu_mask[3]** DP TX software interrupt 3 to cpu mask 0: ignore interrupt raw status 1: reveal interrupt raw status
10	RW	**reg_bank_int_2_cpu_mask[2]** DP TX software interrupt 2 to cpu mask 0: ignore interrupt raw status 1: reveal interrupt raw status
9	RW	**reg_bank_int_2_cpu_mask[1]** DP TX software interrupt 1 to cpu mask 0: ignore interrupt raw status 1: reveal interrupt raw status
8	RW	**reg_bank_int_2_cpu_mask[0]** DP TX software interrupt 0 to cpu mask 0: ignore interrupt raw status 1: reveal interrupt raw status
7	WO	**reg_bank_int_2_cpu_clr[7]** DP TX software interrupt 7 to cpu clear this bit will go back to 0 after one clock cycle
6	WO	**reg_bank_int_2_cpu_clr[6]** DP TX software interrupt 6 to cpu clear this bit will go back to 0 after one clock cycle
5	WO	**reg_bank_int_2_cpu_clr[5]** DP TX software interrupt 5 to cpu clear this bit will go back to 0 after one clock cycle

to next page

from previous page

4	WO	**reg_bank_int_2_cpu_clr[4]** DP TX software interrupt 4 to cpu clear this bit will go back to 0 after one clock cycle
3	WO	**reg_bank_int_2_cpu_clr[3]** DP TX software interrupt 3 to cpu clear this bit will go back to 0 after one clock cycle
2	WO	**reg_bank_int_2_cpu_clr[2]** DP TX software interrupt 2 to cpu clear this bit will go back to 0 after one clock cycle
1	WO	**reg_bank_int_2_cpu_clr[1]** DP TX software interrupt 1 to cpu clear this bit will go back to 0 after one clock cycle
0	WO	**reg_bank_int_2_cpu_clr[0]** DP TX software interrupt 0 to cpu clear this bit will go back to 0 after one clock cycle

Offset: 00Ch DPTX00C: DPTX Software Interrupt to CPU Status Register Init = 0x00000000

Bit	R/W	Description
31:16	RO	**Reserved**
15	RO	**reg_bank_int_2_cpu_raw[7]** DP TX software interrupt 7 to cpu raw status
14	RO	**reg_bank_int_2_cpu_raw[6]** DP TX software interrupt 6 to cpu raw status
13	RO	**reg_bank_int_2_cpu_raw[5]** DP TX software interrupt 5 to cpu raw status
12	RO	**reg_bank_int_2_cpu_raw[4]** DP TX software interrupt 4 to cpu raw status
11	RO	**reg_bank_int_2_cpu_raw[3]** DP TX software interrupt 3 to cpu raw status
10	RO	**reg_bank_int_2_cpu_raw[2]** DP TX software interrupt 2 to cpu raw status
9	RO	**reg_bank_int_2_cpu_raw[1]** DP TX software interrupt 1 to cpu raw status
8	RO	**reg_bank_int_2_cpu_raw[0]** DP TX software interrupt 0 to cpu raw status
7	RO	**reg_bank_int_2_cpu[7]** DP TX software interrupt 7 to cpu
6	RO	**reg_bank_int_2_cpu[6]** DP TX software interrupt 6 to cpu
5	RO	**reg_bank_int_2_cpu[5]** DP TX software interrupt 5 to cpu
4	RO	**reg_bank_int_2_cpu[4]** DP TX software interrupt 4 to cpu
3	RO	**reg_bank_int_2_cpu[3]** DP TX software interrupt 3 to cpu
2	RO	**reg_bank_int_2_cpu[2]** DP TX software interrupt 2 to cpu

to next page

from previous page

1	RO	**reg_bank_int_2_cpu[1]** DP TX software interrupt 1 to cpu
0	RO	**reg_bank_int_2_cpu[0]** DP TX software interrupt 0 to cpu

Offset: 010h **DPTX010: DPTX CPU Timer 0 Configuration Register** **Init = 0x00000000**

Bit	R/W	Description
31	RW	**cpu_timer_0_en** DP TX cpu timer 0 enable
30	RW	**cpu_timer_0_mode_sel** DP TX cpu timer 0 mode selection 0: one time mode 1: continuous mode (auto restart after DP TX cpu timer 0 down counts to zero)
29: 0	RW	**cpu_timer_0_cnt_warn** generate interrupt when DP TX cpu timer 0 down counts to this value

Offset: 014h **DPTX014: DPTX CPU Timer 0 Configuration Register** **Init = 0x00000000**

Bit	R/W	Description
31:30	RW	**cpu_timer_0_tick_sel** DP TX cpu timer 0 tick period selection 0: 1 us 1: 10 us 2: 100 us 3: 1 ms
29: 0	RW	**cpu_timer_0_cnt_init** generate interrupt when DP TX cpu timer 0 down counts from this value to zero

Offset: 018h **DPTX018: DPTX CPU Timer 0 Status Register** **Init = 0x00000000**

Bit	R/W	Description
31	RW	**cpu_timer_0_zero_flag** read as 0 before DP TX cpu timer 0 down counts to zero read as 1 after DP TX cpu timer 0 down counts to zero write with 1 to clear this bit
30	RW	**cpu_timer_0_warn_flag** read as 0 before DP TX cpu timer 0 down counts to cpu_timer_0_cnt_warn read as 1 after DP TX cpu timer 0 down counts to cpu_timer_0_cnt_warn write with 1 to clear this bit
29: 0	RW	**cpu_timer_0_cnt_curr** DP TX cpu timer 0 current value

Offset: 01Ch **DPTX01C: DPTX Version Register** **Init = 0x20190815**

Bit	R/W	Description
31: 0	RO	**DP TX version**

Offset: 020h	DPTX020: DPTX CPU Timer 1 Configuration Register	Init = 0x00000000

Bit	R/W	Description
31	RW	**cpu_timer_1_en** DP TX cpu timer 1 enable
30	RW	**cpu_timer_1_mode_sel** DP TX cpu timer 1 mode selection 0: one time mode 1: continuous mode (auto restart after DP TX cpu timer 1 down counts to zero)
29: 0	RW	**cpu_timer_1_cnt_warn** generate interrupt when DP TX cpu timer 1 down counts to this value

Offset: 024h	DPTX024: DPTX CPU Timer 1 Configuration Register	Init = 0x00000000

Bit	R/W	Description
31:30	RW	**cpu_timer_1_tick_sel** DP TX cpu timer 1 tick period selection 0: 1 us 1: 10 us 2: 100 us 3: 1 ms
29: 0	RW	**cpu_timer_1_cnt_init** generate interrupt when DP TX cpu timer 1 down counts from this value to zero

Offset: 028h	DPTX028: DPTX CPU Timer 1 Status Register	Init = 0x00000000

Bit	R/W	Description
31	RW	**cpu_timer_1_zero_flag** read as 0 before DP TX cpu timer 1 down counts to zero read as 1 after DP TX cpu timer 1 down counts to zero write with 1 to clear this bit
30	RW	**cpu_timer_1_warn_flag** read as 0 before DP TX cpu timer 1 down counts to cpu_timer_0_cnt_warn read as 1 after DP TX cpu timer 1 down counts to cpu_timer_0_cnt_warn write with 1 to clear this bit
29: 0	RW	**cpu_timer_1_cnt_curr** DP TX cpu timer 1 current value

Offset: 030h	DPTX030: DPTX CPU Timer 2 Configuration Register	Init = 0x00000000

Bit	R/W	Description
31	RW	**cpu_timer_2_en** DP TX cpu timer 2 enable
30	RW	**cpu_timer_2_mode_sel** DP TX cpu timer 2 mode selection 0: one time mode 1: continuous mode (auto restart after DP TX cpu timer 2 down counts to zero)
29: 0	RW	**cpu_timer_2_cnt_warn** generate interrupt when DP TX cpu timer 2 down counts to this value

Offset: 034h	DPTX034: DPTX CPU Timer 2 Configuration Register	Init = 0x00000000

Bit	R/W	Description
31:30	RW	**cpu_timer_2_tick_sel** DP TX cpu timer 2 tick period selection 0: 1 us 1: 10 us 2: 100 us 3: 1 ms
29: 0	RW	**cpu_timer_2_cnt_init** generate interrupt when DP TX cpu timer 2 down counts from this value to zero

Offset: 038h	DPTX038: DPTX CPU Timer 2 Status Register	Init = 0x00000000

Bit	R/W	Description
31	RW	**cpu_timer_1_zero_flag** read as 0 before DP TX cpu timer 2 down counts to zero read as 1 after DP TX cpu timer 2 down counts to zero write with 1 to clear this bit
30	RW	**cpu_timer_2_warn_flag** read as 0 before DP TX cpu timer 2 down counts to cpu_timer_0_cnt_warn read as 1 after DP TX cpu timer 2 down counts to cpu_timer_0_cnt_warn write with 1 to clear this bit
29: 0	RW	**cpu_timer_2_cnt_curr** DP TX cpu timer 2 current value

Offset: 040h	DPTX040: DPTX Interrupt to MCU Configuration Register	Init = 0x00C00000

Bit	R/W	Description
31:24	RO	**Reserved**
23	RW	**aux_reply_timeout_int_2_mcu_en** DP TX AUX channel timeout interrupt to mcu enable
22	RW	**aux_reply_done_int_2_mcu_en** DP TX AUX channel reply done interrupt to mcu enable
21	RW	**hpd_event_int_2_mcu_en** DP TX HPD event interrupt to mcu enable
20	RW	**hpd_low_int_2_mcu_en** DP TX HPD low interrupt to mcu enable
19	RW	**hpd_irq_int_2_mcu_en** DP TX HPD irq interrupt to mcu enable
18	RW	**timer_2_int_2_mcu_en** DP TX mcu timer 2 interrupt to mcu enable
17	RW	**timer_1_int_2_mcu_en** DP TX mcu timer 1 interrupt to mcu enable
16	RW	**timer_0_int_2_mcu_en** DP TX mcu timer 0 interrupt to mcu enable
15	RW	**aux_reply_timeout_int_2_mcu_mask** DP TX AUX channel timeout interrupt to mcu mask 0: ignore interrupt raw status 1: reveal interrupt raw status

to next page

from previous page

14	RW	**aux_reply_done_int_2_mcu_mask** DP TX AUX channel reply done interrupt to mcu mask 0: ignore interrupt raw status 1: reveal interrupt raw status
13	RW	**hpd_event_int_2_mcu_mask** DP TX HPD event interrupt to mcu mask 0: ignore interrupt raw status 1: reveal interrupt raw status
12	RW	**hpd_low_int_2_mcu_mask** DP TX HPD low interrupt to mcu mask 0: ignore interrupt raw status 1: reveal interrupt raw status
11	RW	**hpd_irq_int_2_mcu_mask** DP TX HPD irq interrupt to mcu mask 0: ignore interrupt raw status 1: reveal interrupt raw status
10	RW	**timer_2_int_2_mcu_mask** DP TX mcu timer 2 interrupt to mcu mask 0: ignore interrupt raw status 1: reveal interrupt raw status
9	RW	**timer_1_int_2_mcu_mask** DP TX mcu timer 1 interrupt to mcu mask 0: ignore interrupt raw status 1: reveal interrupt raw status
8	RW	**timer_0_int_2_mcu_mask** DP TX mcu timer 0 interrupt to mcu mask 0: ignore interrupt raw status 1: reveal interrupt raw status
7	WO	**aux_reply_timeout_int_2_mcu_clr** DP TX AUX channel timeout interrupt to mcu clear this bit will go back to 0 after one clock cycle
6	WO	**aux_reply_done_int_2_mcu_clr** DP TX AUX channel reply done interrupt to mcu clear this bit will go back to 0 after one clock cycle
5	WO	**hpd_event_int_2_mcu_clr** DP TX HPD event interrupt to mcu clear this bit will go back to 0 after one clock cycle
4	WO	**hpd_low_int_2_mcu_clr** DP TX HPD low interrupt to mcu clear this bit will go back to 0 after one clock cycle
3	WO	**hpd_irq_int_2_mcu_clr** DP TX HPD irq interrupt to mcu clear this bit will go back to 0 after one clock cycle
2	WO	**timer_2_int_2_mcu_clr** DP TX mcu timer 2 interrupt to mcu clear this bit will go back to 0 after one clock cycle
1	WO	**timer_1_int_2_mcu_clr** DP TX mcu timer 1 interrupt to mcu clear this bit will go back to 0 after one clock cycle

to next page

from previous page

0	WO	**timer_0_int_2_mcu_clr** DP TX mcu timer 0 interrupt to mcu clear this bit will go back to 0 after one clock cycle

Offset: 044h DPTX044: DPTX Interrupt to MCU Status Register Init = 0x00000000

Bit	R/W	Description
31:16	RO	**Reserved**
15	RO	**aux_reply_timeout_int_2_mcu_raw** DP TX AUX channel timeout interrupt to mcu raw status
14	RO	**aux_reply_done_int_2_mcu_raw** DP TX AUX channel reply done interrupt to mcu raw status
13	RO	**hpd_event_int_2_mcu_raw** DP TX HPD event interrupt to mcu raw status
12	RO	**hpd_low_int_2_mcu_raw** DP TX HPD low interrupt to mcu raw status
11	RO	**hpd_irq_int_2_mcu_raw** DP TX HPD irq interrupt to mcu raw status
10	RO	**timer_2_int_2_mcu_raw** DP TX mcu timer 2 interrupt to mcu raw status
9	RO	**timer_1_int_2_mcu_raw** DP TX mcu timer 1 interrupt to mcu raw status
8	RO	**timer_0_int_2_mcu_raw** DP TX mcu timer 0 interrupt to mcu raw status
7	RO	**aux_reply_timeout_int_2_mcu** DP TX AUX channel timeout interrupt to mcu
6	RO	**aux_reply_done_int_2_mcu** DP TX AUX channel reply done interrupt to mcu
5	RO	**hpd_event_int_2_mcu** DP TX HPD event interrupt to mcu
4	RO	**hpd_low_int_2_mcu** DP TX HPD low interrupt to mcu
3	RO	**hpd_irq_int_2_mcu** DP TX HPD irq interrupt to mcu
2	RO	**timer_2_int_2_mcu** DP TX mcu timer 2 interrupt to mcu
1	RO	**timer_1_int_2_mcu** DP TX mcu timer 1 interrupt to mcu
0	RO	**timer_0_int_2_mcu** DP TX mcu timer 0 interrupt to mcu

Offset: 048h DPTX048: DPTX Software Interrupt to MCU Configuration Register Init = 0x00000000

Bit	R/W	Description
31:24	RO	**Reserved**
23	RW	**reg_bank_int_2_mcu_en[7]** DP TX software interrupt 7 to mcu enable
22	RW	**reg_bank_int_2_mcu_en[6]** DP TX software interrupt 6 to mcu enable

to next page

from previous page

21	RW	**reg_bank_int_2_mcu_en[5]** DP TX software interrupt 5 to mcu enable
20	RW	**reg_bank_int_2_mcu_en[4]** DP TX software interrupt 4 to mcu enable
19	RW	**reg_bank_int_2_mcu_en[3]** DP TX software interrupt 3 to mcu enable
18	RW	**reg_bank_int_2_mcu_en[2]** DP TX software interrupt 2 to mcu enable
17	RW	**reg_bank_int_2_mcu_en[1]** DP TX software interrupt 1 to mcu enable
16	RW	**reg_bank_int_2_mcu_en[0]** DP TX software interrupt 0 to mcu enable
15	RW	**reg_bank_int_2_mcu_mask[7]** DP TX software interrupt 7 to mcu mask 0: ignore interrupt raw status 1: reveal interrupt raw status
14	RW	**reg_bank_int_2_mcu_mask[6]** DP TX software interrupt 6 to mcu mask 0: ignore interrupt raw status 1: reveal interrupt raw status
13	RW	**reg_bank_int_2_mcu_mask[5]** DP TX software interrupt 5 to mcu mask 0: ignore interrupt raw status 1: reveal interrupt raw status
12	RW	**reg_bank_int_2_mcu_mask[4]** DP TX software interrupt 4 to mcu mask 0: ignore interrupt raw status 1: reveal interrupt raw status
11	RW	**reg_bank_int_2_mcu_mask[3]** DP TX software interrupt 3 to mcu mask 0: ignore interrupt raw status 1: reveal interrupt raw status
10	RW	**reg_bank_int_2_mcu_mask[2]** DP TX software interrupt 2 to mcu mask 0: ignore interrupt raw status 1: reveal interrupt raw status
9	RW	**reg_bank_int_2_mcu_mask[1]** DP TX software interrupt 1 to mcu mask 0: ignore interrupt raw status 1: reveal interrupt raw status
8	RW	**reg_bank_int_2_mcu_mask[0]** DP TX software interrupt 0 to mcu mask 0: ignore interrupt raw status 1: reveal interrupt raw status
7	WO	**reg_bank_int_2_mcu_clr[7]** DP TX software interrupt 7 to mcu clear this bit will go back to 0 after one clock cycle
6	WO	**reg_bank_int_2_mcu_clr[6]** DP TX software interrupt 6 to mcu clear this bit will go back to 0 after one clock cycle

to next page

from previous page

5	WO	**reg_bank_int_2_mcu_clr[5]** DP TX software interrupt 5 to mcu clear this bit will go back to 0 after one clock cycle
4	WO	**reg_bank_int_2_mcu_clr[4]** DP TX software interrupt 4 to mcu clear this bit will go back to 0 after one clock cycle
3	WO	**reg_bank_int_2_mcu_clr[3]** DP TX software interrupt 3 to mcu clear this bit will go back to 0 after one clock cycle
2	WO	**reg_bank_int_2_mcu_clr[2]** DP TX software interrupt 2 to mcu clear this bit will go back to 0 after one clock cycle
1	WO	**reg_bank_int_2_mcu_clr[1]** DP TX software interrupt 1 to mcu clear this bit will go back to 0 after one clock cycle
0	WO	**reg_bank_int_2_mcu_clr[0]** DP TX software interrupt 0 to mcu clear this bit will go back to 0 after one clock cycle

Offset: 04Ch DPTX04C: DPTX Software Interrupt to MCU Status Register Init = 0x00000000

Bit	R/W	Description
31:16	RO	**Reserved**
15	RO	**reg_bank_int_2_mcu_raw[7]** DP TX software interrupt 7 to mcu raw status
14	RO	**reg_bank_int_2_mcu_raw[6]** DP TX software interrupt 6 to mcu raw status
13	RO	**reg_bank_int_2_mcu_raw[5]** DP TX software interrupt 5 to mcu raw status
12	RO	**reg_bank_int_2_mcu_raw[4]** DP TX software interrupt 4 to mcu raw status
11	RO	**reg_bank_int_2_mcu_raw[3]** DP TX software interrupt 3 to mcu raw status
10	RO	**reg_bank_int_2_mcu_raw[2]** DP TX software interrupt 2 to mcu raw status
9	RO	**reg_bank_int_2_mcu_raw[1]** DP TX software interrupt 1 to mcu raw status
8	RO	**reg_bank_int_2_mcu_raw[0]** DP TX software interrupt 0 to mcu raw status
7	RO	**reg_bank_int_2_mcu[7]** DP TX software interrupt 7 to mcu
6	RO	**reg_bank_int_2_mcu[6]** DP TX software interrupt 6 to mcu
5	RO	**reg_bank_int_2_mcu[5]** DP TX software interrupt 5 to mcu
4	RO	**reg_bank_int_2_mcu[4]** DP TX software interrupt 4 to mcu
3	RO	**reg_bank_int_2_mcu[3]** DP TX software interrupt 3 to mcu

to next page

from previous page

2	RO	**reg_bank_int_2_mcu[2]** DP TX software interrupt 2 to mcu
1	RO	**reg_bank_int_2_mcu[1]** DP TX software interrupt 1 to mcu
0	RO	**reg_bank_int_2_mcu[0]** DP TX software interrupt 0 to mcu

Offset: 050h **DPTX050: DPTX MCU Timer 0 Configuration Register** **Init = 0x00000000**

Bit	R/W	Description
31	RW	**mcu_timer_0_en** DP TX mcu timer 0 enable
30	RW	**mcu_timer_0_mode_sel** DP TX mcu timer 0 mode selection 0: one time mode 1: continuous mode (auto restart after DP TX mcu timer 0 down counts to zero)
29: 0	RW	**mcu_timer_0_cnt_warn** generate interrupt when DP TX mcu timer 0 down counts to this value

Offset: 054h **DPTX054: DPTX MCU Timer 0 Configuration Register** **Init = 0x00000000**

Bit	R/W	Description
31:30	RW	**mcu_timer_0_tick_sel** DP TX mcu timer 0 tick period selection 0: 1 us 1: 10 us 2: 100 us 3: 1 ms
29: 0	RW	**mcu_timer_0_cnt_init** generate interrupt when DP TX mcu timer 0 down counts from this value to zero

Offset: 058h **DPTX058: DPTX MCU Timer 0 Status Register** **Init = 0x00000000**

Bit	R/W	Description
31	RW	**mcu_timer_0_zero_flag** read as 0 before DP TX mcu timer 0 down counts to zero read as 1 after DP TX mcu timer 0 down counts to zero write with 1 to clear this bit
30	RW	**mcu_timer_0_warn_flag** read as 0 before DP TX mcu timer 0 down counts to mcu_timer_0_cnt_warn read as 1 after DP TX mcu timer 0 down counts to mcu_timer_0_cnt_warn write with 1 to clear this bit
29: 0	RW	**mcu_timer_0_cnt_curr** DP TX mcu timer 0 current value

Offset: 060h	DPTX060: DPTX MCU Timer 1 Configuration Register	Init = 0x00000000

Bit	R/W	Description
31	RW	**mcu_timer_1_en** DP TX mcu timer 1 enable
30	RW	**mcu_timer_1_mode_sel** DP TX mcu timer 1 mode selection 0: one time mode 1: continuous mode (auto restart after DP TX mcu timer 1 down counts to zero)
29: 0	RW	**mcu_timer_1_cnt_warn** generate interrupt when DP TX mcu timer 1 down counts to this value

Offset: 064h	DPTX064: DPTX MCU Timer 1 Configuration Register	Init = 0x00000000

Bit	R/W	Description
31:30	RW	**mcu_timer_1_tick_sel** DP TX mcu timer 1 tick period selection 0: 1 us 1: 10 us 2: 100 us 3: 1 ms
29: 0	RW	**mcu_timer_1_cnt_init** generate interrupt when DP TX mcu timer 1 down counts from this value to zero

Offset: 068h	DPTX068: DPTX MCU Timer 1 Status Register	Init = 0x00000000

Bit	R/W	Description
31	RW	**mcu_timer_1_zero_flag** read as 0 before DP TX mcu timer 1 down counts to zero read as 1 after DP TX mcu timer 1 down counts to zero write with 1 to clear this bit
30	RW	**mcu_timer_1_warn_flag** read as 0 before DP TX mcu timer 1 down counts to mcu_timer_0_cnt_warn read as 1 after DP TX mcu timer 1 down counts to mcu_timer_0_cnt_warn write with 1 to clear this bit
29: 0	RW	**mcu_timer_1_cnt_curr** DP TX mcu timer 1 current value

Offset: 070h	DPTX070: DPTX MCU Timer 2 Configuration Register	Init = 0x00000000

Bit	R/W	Description
31	RW	**mcu_timer_2_en** DP TX mcu timer 2 enable
30	RW	**mcu_timer_2_mode_sel** DP TX mcu timer 2 mode selection 0: one time mode 1: continuous mode (auto restart after DP TX mcu timer 2 down counts to zero)
29: 0	RW	**mcu_timer_2_cnt_warn** generate interrupt when DP TX mcu timer 2 down counts to this value

Offset: 074h	DPTX074: DPTX MCU Timer 2 Configuration Register	Init = 0x00000000

Bit	R/W	Description
31:30	RW	**mcu_timer_2_tick_sel** DP TX mcu timer 2 tick period selection 0: 1 us 1: 10 us 2: 100 us 3: 1 ms
29: 0	RW	**mcu_timer_2_cnt_init** generate interrupt when DP TX mcu timer 2 down counts from this value to zero

Offset: 078h	DPTX078: DPTX MCU Timer 2 Status Register	Init = 0x00000000

Bit	R/W	Description
31	RW	**mcu_timer_1_zero_flag** read as 0 before DP TX mcu timer 2 down counts to zero read as 1 after DP TX mcu timer 2 down counts to zero write with 1 to clear this bit
30	RW	**mcu_timer_2_warn_flag** read as 0 before DP TX mcu timer 2 down counts to mcu_timer_0_cnt_warn read as 1 after DP TX mcu timer 2 down counts to mcu_timer_0_cnt_warn write with 1 to clear this bit
29: 0	RW	**mcu_timer_2_cnt_curr** DP TX mcu timer 2 current value

Offset: 07Ch	DPTX07C: DPTX HPD Status Register	Init = 0x00000000

Bit	R/W	Description
31:29	RO	**Reserved**
28	RO	**hpd_in** DP TX HPD input level
27:25	RO	**Reserved**
24	RW	**hpd_low_cnt_en** DP TX HPD low period counter enable
23:12	RO	**Reserved**
11: 0	RO	**hpd_low_cnt** DP TX HPD low period count

Offset: 080h	DPTX07C: DPTX HPD IRQ Configuration Register	Init = 0x00000000

Bit	R/W	Description
31:28	RO	**Reserved**
27:16	RW	**hpd_irq_max_us_num** DP TX HPD low period for irq trigger upper bound HPD irq interrupt is triggered if hpd_low_cnt ¿= hpd_irq_min_us_num and hpd_low_cnt ¡= hpd_irq_max_us_num
25:10	RO	**Reserved**
9: 0	RW	**hpd_irq_min_us_num** DP TX HPD low period for irq trigger lower bound HPD irq interrupt is triggered if hpd_low_cnt ¿= hpd_irq_min_us_num and hpd_low_cnt ¡= hpd_irq_max_us_num

Offset: 084h	DPTX084: DPTX HPD Event Configuration Register	Init = 0x00000000

Bit	R/W	Description
31:12	RO	**Reserved**
11: 0	RW	**hpd_event_min_us_num** DP TX HPD low period for event trigger lower bound HPD event interrupt is triggered if hpd_low_cnt ¿= hpd_event_min_us_num

Offset: 088h	DPTX088: DPTX AUX Request Configuration Register	Init = 0x00000000

Bit	R/W	Description
31:28	RO	**aux_req_cmd** DP TX AUX channel request command
27:21	RO	**Reserved**
20	RW	**aux_req_length_end_en** DP TX AUX channel request packet is ended with aux_req_length
19:17	RO	**Reserved**
16	RW	**aux_req_addr_end_en** DP TX AUX channel request packet is ended with aux_req_addr
15:13	RO	**Reserved**
12	RO	**aux_req_awaiting_reply_flag** indicates if DP TX AUX channel is waiting for reply packet
11: 9	RO	**Reserved**
8	RO	**aux_req_done_flag** indicates if DP TX AUX channel request command is done with receiving reply
7: 5	RO	**Reserved**
4	WO	**aux_req_trig_cmd** write with 1 to trigger DP TX AUX channel request packet this bit will go back to 0 after one clock cycle
3: 1	RO	**Reserved**
0	WO	**aux_reset_cmd** write with 1 to reset DP TX AUX channel this bit will go back to 0 after one clock cycle

Offset: 08Ch	DPTX08C: DPTX AUX Request Length And Address Register	Init = 0x00000000

Bit	R/W	Description
31:28	RO	**Reserved**
27:24	RW	**aux_req_length** DP TX AUX channel request length
23:20	RO	**Reserved**
19: 0	WO	**aux_req_addr** DP TX AUX channel request address

Offset: 090h	DPTX090: DPTX AUX Request Data Register	Init = 0x00000000

Bit	R/W	Description
31:24	RW	**aux_req_data_byte_03** DP TX AUX channel request data byte 3

to next page

from previous page

23:16	RW	**aux_req_data_byte_02** DP TX AUX channel request data byte 2
15: 8	RW	**aux_req_data_byte_01** DP TX AUX channel request data byte 1
7: 0	RW	**aux_req_data_byte_00** DP TX AUX channel request data byte 0

Offset: 094h **DPTX094: DPTX AUX Request Data Register** **Init = 0x00000000**

Bit	R/W	Description
31:24	RW	**aux_req_data_byte_07** DP TX AUX channel request data byte 7
23:16	RW	**aux_req_data_byte_06** DP TX AUX channel request data byte 6
15: 8	RW	**aux_req_data_byte_05** DP TX AUX channel request data byte 5
7: 0	RW	**aux_req_data_byte_04** DP TX AUX channel request data byte 4

Offset: 098h **DPTX098: DPTX AUX Request Data Register** **Init = 0x00000000**

Bit	R/W	Description
31:24	RW	**aux_req_data_byte_11** DP TX AUX channel request data byte 11
23:16	RW	**aux_req_data_byte_10** DP TX AUX channel request data byte 10
15: 8	RW	**aux_req_data_byte_09** DP TX AUX channel request data byte 9
7: 0	RW	**aux_req_data_byte_08** DP TX AUX channel request data byte 8

Offset: 09Ch **DPTX09C: DPTX AUX Request Data Register** **Init = 0x00000000**

Bit	R/W	Description
31:24	RW	**aux_req_data_byte_15** DP TX AUX channel request data byte 15
23:16	RW	**aux_req_data_byte_14** DP TX AUX channel request data byte 14
15: 8	RW	**aux_req_data_byte_13** DP TX AUX channel request data byte 13
7: 0	RW	**aux_req_data_byte_12** DP TX AUX channel request data byte 12

Offset: 0A0h **DPTX0A0: DPTX AUX Reply Data Register** **Init = 0x00000000**

Bit	R/W	Description
31:24	RO	**aux_req_data_byte_03** DP TX AUX channel reply data byte 3

to next page

from previous page

23:16	RO	**aux_req_data_byte_02** DP TX AUX channel reply data byte 2
15: 8	RO	**aux_req_data_byte_01** DP TX AUX channel reply data byte 1
7: 0	RO	**aux_req_data_byte_00** DP TX AUX channel reply data byte 0

Offset: 0A4h **DPTX0A4: DPTX AUX Reply Data Register** **Init = 0x00000000**

Bit	R/W	Description
31:24	RO	**aux_req_data_byte_07** DP TX AUX channel reply data byte 7
23:16	RO	**aux_req_data_byte_06** DP TX AUX channel reply data byte 6
15: 8	RO	**aux_req_data_byte_05** DP TX AUX channel reply data byte 5
7: 0	RO	**aux_req_data_byte_04** DP TX AUX channel reply data byte 4

Offset: 0A8h **DPTX0A8: DPTX AUX Reply Data Register** **Init = 0x00000000**

Bit	R/W	Description
31:24	RO	**aux_req_data_byte_11** DP TX AUX channel reply data byte 11
23:16	RO	**aux_req_data_byte_10** DP TX AUX channel reply data byte 10
15: 8	RO	**aux_req_data_byte_09** DP TX AUX channel reply data byte 9
7: 0	RO	**aux_req_data_byte_08** DP TX AUX channel reply data byte 8

Offset: 0ACh **DPTX0AC: DPTX AUX Reply Data Register** **Init = 0x00000000**

Bit	R/W	Description
31:24	RO	**aux_req_data_byte_15** DP TX AUX channel reply data byte 15
23:16	RO	**aux_req_data_byte_14** DP TX AUX channel reply data byte 14
15: 8	RO	**aux_req_data_byte_13** DP TX AUX channel reply data byte 13
7: 0	RO	**aux_req_data_byte_12** DP TX AUX channel reply data byte 12

Offset: 0B0h **DPTX0B0: DPTX AUX Reply Command And Status Register** **Init = 0x00000000**

Bit	R/W	Description
31:24	RO	**aux_reply_rise_num** DP TX AUX channel reply physical layer rise count

to next page

from previous page

23:16	RO	**aux_reply_fall_num** DP TX AUX channel reply physical layer fall count
15:13	RO	**Reserved**
12: 8	RO	**aux_reply_data_num** DP TX AUX channel reply date byte number received
7: 4	RO	**aux_reply_cmd** DP TX AUX channel reply command received
3: 0	RO	**Reserved**

Offset: 0B4h DPTX0B4: DPTX AUX Reply Configuration Register Init = 0x00000000

Bit	R/W	Description
31:17	RO	**Reserved**
16	RW	**aux_reply_test_en** 0: DP TX AUX channel runs in normal mode 1: DP TX AUX channel runs in loop back mode
15:13	RO	**Reserved**
12	RW	**aux_reply_timeout_retry_en** 0: DP TX AUX channel request transaction is NOT restarted after reply time out 1: DP TX AUX channel request transaction is restarted after reply time out
15: 9	RO	**Reserved**
8: 0	RO	**aux_reply_timeout_us_num** DP TX AUX channerl reply time out period in micro second

Offset: 0B8h DPTX0B8: DPTX AUX Read Configuration Register Init = 0x00000000

Bit	R/W	Description
31:29	RO	**Reserved**
28	RW	**host_read_edid_off** this register determines if host will read EDID through PCIe scratch register 0: enable 1: disable
27:25	RO	**Reserved**
24	RW	**video_format_src_sel** this register determines wither host or BMC will send video format to DisplayPort 0: host to send information through PCIe scratch register 0xe0, 0xe1 and 0xe4 1: BMC to send information at address 0x18000de0 and 0x18000de4
23:21	RO	**Reserved**
20	RW	**aux_rd_00205h_en** this register determines whether or not to read DPCD 00205h every 5 ms 0: disable 1: enable
19:17	RO	**Reserved**
16	RW	**rbr_en** this register determines if reduced bit rate (1.62Gbps) is allowed 0: 1.62Gbps is NOT allowed 1: 1.62Gbps is allowed

to next page

from previous page

15:12	RW	**aux_wr_repeat_num** this register determines how many times determins DP TX AUX will write DPCD at the same address with the same value 0: disable 1: enable
11: 9	RO	**Reserved**
8	RW	**aux_read_edid_wr_30h_en** this bit determines whether or not to let DP TX write I2C address 30h over AUX while reading EDID 0: disable 1: enable
7: 5	RO	**Reserved**
4	RW	**aux_rd_00203h_en** this bit determines whether or not to let DP TX read AUX address 00203h while checking link status 0: disable 1: enable
3: 1	RO	**Reserved**
0	RW	**aux_rd_00000h_000ffh_en** this bit determines whether or not to let DP TX read AUX address 00000h 000ffh before starting link training 0: disable 1: enable

Offset: 0C0h **DPTX0C0: DPTX Main Link Configuration Register** **Init = 0x00000000**

Bit	R/W	Description
31:13	RO	**Reserved**
12	RW	**framing_mode_sel** framing mode selection 0: default framing mode 1: enhanced framing mode
11:10	RO	**Reserved**
9: 8	RW	**link_rate_sel** link rate selection 0: 1.62 Gbps 1: 2.70 Gbps 2: 5.40 Gbps
7	RO	**Reserved**
6: 4	RW	**lane_num** lane count 0: all lanes inactive 1: 1 lane 2: 2 lanes others: reserved
3: 1	RO	**Reserved**
0	RW	**pkt_reset_cmd** write with 1 to reset DP TX main link layer and digital PHY this bit will go back to 0 after one clock cycle

Offset: 0C4h	DPTX0C4: DPTX Main Link Configuration Register	Init = 0x00000000
Bit	**R/W**	**Description**
31:30	RO	**Reserved**
29	RW	**vb_id_5** VB-ID bit 5 (HDCP SYNC DETECT)
28:25	RO	**Reserved**
24	RW	**msa_mvid_hw_sel** MSA packet mvid source selection 0: from msa_mvid by firmware 1: from msa_mvid_hw
23: 0	RW	**msa_mvid_hw** MSA packet mvid calculated by hardware period of video clock * msa_mvid_hw = period of DP TX symbol clock * msa_nvid

Offset: 0C8h	DPTX0C8: DPTX Main Link MSA Configuration Register	Init = 0x000028A9
Bit	**R/W**	**Description**
31:24	RO	**Reserved**
23: 0	RW	**msa_mvid** MSA packet mvid set by firmware according to video pixel clock rate and DP TX main link rate

Offset: 0CCh	DPTX0CC: DPTX Main Link MSA Configuration Register	Init = 0x00008000
Bit	**R/W**	**Description**
31:24	RO	**Reserved**
23: 0	RW	**msa_nvid** MSA packet nvid

Offset: 0D0h	DPTX0D0: DPTX Main Link MSA Configuration Register	Init = 0x031B0706
Bit	**R/W**	**Description**
31:16	RW	**msa_vtotal** MSA packet vtotal = VER_SYNC + VER_BACK + VER_ADDR + VER_FRONT
15: 0	RW	**msa_htotal** MSA packet htotal = HOR_SYNC + HOR_BACK + HOR_ADDR + HOR_FRONT

Offset: 0D4h	DPTX0D4: DPTX Main Link MSA Configuration Register	Init = 0x00180170
Bit	**R/W**	**Description**
31:16	RW	**msa_vstart** If 0x1e6eb180[16] video_line_buf_timing_sel = 0, MSA packet vstart = VER_SYNC + VER_BACK If 0x1e6eb180[16] video_line_buf_timing_sel = 1, MSA packet vstart = VER_SYNC + VER_BACK + VER_FRONT
15: 0	RW	**msa_hstart** If 0x1e6eb180[16] video_line_buf_timing_sel = 0, MSA packet hstart = HOR_SYNC + HOR_BACK If 0x1e6eb180[16] video_line_buf_timing_sel = 1, MSA packet hstart = HOR_SYNC + HOR_BACK + HOR_FRONT

Offset: 0D8h	DPTX0D8: DPTX Main Link MSA Configuration Register	Init = 0x03000556
Bit	**R/W**	**Description**
31:16	RW	**msa_vheight** MSA packet vheight = VER_ADDR
15: 0	RW	**msa_hwidth** MSA packet hwidth = HOR_ADDR

Offset: 0DCh	DPTX0DC: DPTX Main Link MSA Configuration Register	Init = 0x00700006
Bit	**R/W**	**Description**
31:16	RW	**msa_vspvsw** MSA packet vspvsw = VER_SYNC
15: 0	RW	**msa_hsphsw** MSA packet hsphsw = HOR_SYNC

Offset: 0E0h	DPTX0E0: DPTX Main Link MSA Configuration Register	Init = 0x00000000
Bit	**R/W**	**Description**
31:16	RO	**Reserved**
15: 8	RW	**msa_misc1** MSA packet misc1
7: 0	RW	**msa_misc0** MSA packet misc0

Offset: 0E4h	DPTX0E4: DPTX Main Link Training Pattern Configuration Register	Init = 0x10000100
Bit	**R/W**	**Description**
31:30	RO	**Reserved**
29:28	RW	**link_pat_sel** link training pattern selection 0: link training is NOT enabled 1: link training pattern 1 (TPS1) 2: link training pattern 2 (TPS2) 3: link training pattern 3 (TPS3)
27:25	RO	**Reserved**
24	RW	**idle_pat_sel** VB-ID idle pattern selection 0: VB-ID idle pattern is NOT enabled 1: VB-ID idle pattern is enabled
23:19	RO	**Reserved**
18:16	RW	**qual_pat_sel** link quality test pattern selection 2: Symbol Error Rate Measurement pattern 5: HBR2 Compliance EYE pattern others: reserved
15: 0	RW	**qual_pat_5_symbol_num** corresponds to HBR2_COMPLIANCE_SCRAMBLER_RESET

Offset: 0E8h DPTX0E8: DPTX Main Link Digital PHY Configuration Register Init = 0x00001642		
Bit	**R/W**	**Description**
31:29	RO	**Reserved**
28	RW	**enc_out_custom_en** DP TX 8b/10b encoder 80-bit custom test pattern enable
27:25	RO	**Reserved**
24	RW	**enc_out_prbs7_en** DP TX 8b/10b encoder PRBS7 test pattern enable
23:21	RO	**Reserved**
20	RW	**enc_out_zero_en** DP TX 8b/10b encoder output 10-bit zero enable
19:17	RO	**Reserved**
16	RW	**enc_bit_order_sel** DP TX 8b/10b encoder outout bit order inverse selection 0: normal 1: reversed
15:14	RO	**Reserved**
13	RW	**scramble_alt_reset_sel** DP TX symbol scrambler LFSR reset alternative selection 0: reset to 0xFFFF 1: reset to 0xFFFE
12	RW	**scramble_en** DP TX symbol scrambler enable
10: 8	RW	**skew_0_3_sel** Reserved
6: 4	RW	**skew_0_2_sel** Reserved
1: 0	RW	**skew_0_1_sel** inter-lane skew from lane 0 to lane 1

Offset: 0ECh DPTX0EC: DPTX Main Link Digital PHY Configuration Register Init = 0xAAAAAAAA		
Bit	**R/W**	**Description**
31: 0	RW	**enc_out_custom_pat_apb[31: 0]** 80-bit custom test pattern

Offset: 0F0h DPTX0F0: DPTX Main Link Digital PHY Configuration Register Init = 0xAAAAAAAA		
Bit	**R/W**	**Description**
31: 0	RW	**enc_out_custom_pat_apb[63:32]** 80-bit custom test pattern

Offset: 0F4h DPTX0F4: DPTX Main Link Digital PHY Configuration Register Init = 0x0000AAAA		
Bit	**R/W**	**Description**
31:16	RO	**Reserved**
15: 0	RW	**enc_out_custom_pat_apb[79:64]** 80-bit custom test pattern

Offset: 0F8h	DPTX0F8: DPTX Main Link Configuration Register	Init = 0x16371120
Bit	**R/W**	**Description**
31	RO	**Reserved**
30:24	RW	**tu_valid_symbol_num_integer** integer part of number of valid symbols in a transfer unit (TU), where TU size is fixed at 64 For example, if the number valid symbols is 51.63264, this register should be written with 51
23:22	RO	**Reserved**
21:16	RW	**tu_valid_symbol_num_frc_256x** 256 times of fractional part of number of valid symbols in a transfer unit (TU), where TU size is fixed at 64 For example, if the number valid symbols is 51.63, this register should be written with 161, which is 0.63 * 256 = 161.28 rounded to nearest lower integer
15:13	RO	**Reserved**
12	RW	**framing_mode_sel_fw** framing mode selection defined by firmware 0: default framing mode 1: enhanced framing mode
11:10	RO	**Reserved**
9: 8	RW	**link_rate_sel_fw** link rate selection defined by firmware 0: 1.62 Gbps 1: 2.70 Gbps 2: 5.40 Gbps
7	RO	**Reserved**
6: 4	RW	**lane_num_fw** lane count defined by firmware 0: all lanes inactive 1: 1 lane 2: 2 lanes others: reserved
3: 2	RO	**Reserved**
1	RW	**sscg_en_fw** this register is used to determine whether or not to enable down spread spectrum feature if Rx DPCD 00003h bit 0 is read as 1 0: disable 1: enable
0	RW	**lane_1_duplicate_en** this register is for a special use case to output exactly the same symbol sequence in physical lane 0 and lane 1, which are routed to the front and the back of the chasis respectively 0: disable 1: enable

Offset: 0FCh	DPTX0FC: DPTX Main Link Configuration Register	Init = 0x00000000
Bit	**R/W**	**Description**
31	RW	**training_fail_2nd_eq** this bit indicates if link training second phase is failed at equalization
30	RW	**training_fail_2nd_cr** this bit indicates if link training second phase is failed at clock recovery

to next page

from previous page

29	RW	**training_fail_2nd** this bit indicates if link training second phase is failed
28	RW	**training_fail_1st** this bit indicates if link training first phase is failed
27:26	RO	**Reserved**
25	RW	**training_done_2nd** this bit indicates if link training second phase is done
24	RW	**training_done_1st** this bit indicates if link training first phase is done
23:22	RO	**Reserved**
21	RW	**training_doing_2nd** this bit indicates if link training second phase is doing
20	RW	**training_doing_1st** this bit indicates if link training first phase is doing
19:16	RW	**debug_sel** DP TX debug signals selection
15:12	RO	**Reserved**
11: 0	RO	**pkt_fsm** DP TX FSM 12'h001: IDLE 12'h002: VER_BLANK_VB_BS_VB_ID_MSA 12'h004: VER_BLANK_HB_BS_VB_ID 12'h008: VER_BLANK_OTHER 12'h010: VIDEO_ACT 12'h020: VIDEO_ACT_HB_BS_VB_ID 12'h040: VIDEO_ACT_HB_OTHER 12'h080: PAT_IDLE 12'h100: PAT_SERM 12'h200: PAT_2520_1 12'h400: PAT_2520_2 12'h800: PAT_2520_3

Offset: 100h **DPTX100: DPTX Configuration Register** **Init = 0x11000000**

Bit	R/W	Description
31:29	RO	**Reserved**
28	RW	**tx_resetn** software reset to DP TX 0: reset 1: NOT reset
27:25	RO	**Reserved**
24	RW	**aux_resetn** software reset to DP TX AUX channel 0: reset 1: NOT reset
23	RO	**Reserved**
22	RW	**tx_symbol_bit_inv_sel** parallel to serial option 0: lsb first 1: msb first

to next page

from previous page

21	RW	**tx_symbol_order_sel** DP TX digital PHY to analog PHY 20-bit symbol bit order selection 0: normal 1: reversed
20	RW	**tx_phy_test_mode_sel** DP TX PHY test chip mode 0: use this chip as SOC 1: use this chip as 1-lane DP TX PHY test chip
19:17	RO	**Reserved**
16	RW	**tx_clk_inv_sel** DP TX PHY 1/10 link rate clock phase selection
15:12	RO	**Reserved**
11	RW	**tx_lane_3_rx_clk_inv_sel** Reserved
10	RW	**tx_lane_2_rx_clk_inv_sel** Reserved
9	RW	**tx_lane_1_rx_clk_inv_sel** DP TX PHY lane 1 serial to parallel debug interface clock phase selection
8	RW	**tx_lane_0_rx_clk_inv_sel** DP TX PHY lane 0 serial to parallel debug interface clock phase selection
7: 4	RO	**Reserved**
3	RW	**tx_lane_3_clk_inv_sel** Reserved
2	RW	**tx_lane_2_clk_inv_sel** Reserved
1	RW	**tx_lane_1_clk_inv_sel** DP TX PHY lane 1 parallel to serial interface clock phase selection
0	RW	**tx_lane_0_clk_inv_sel** DP TX PHY lane 0 parallel to serial interface clock phase selection

Offset: 104h **DPTX104: DPTX PHY Configuration Register** **Init = 0x00000000**

Bit	R/W	Description
31:29	RO	**Reserved**
28	RW	**DP_TX_I_PONRST** Power On Reset Asserting this signal will set all PHYs to be at the default states
27:25	RO	**Reserved**
24	RW	**DP_TX_I_DP_RESETn** DisplayPort Reset Asserting this signal will set all PHYs to be at the default states
23	RO	**Reserved**

to next page

from previous page

20	RW	**DP_TX_I_PLL_EN** PLL-alive mode Enable signal for special usage The following clock signals will be alive under this mode: CLOCK_DATARATE_DIV10 CLOCK_DATARATE_DIV20 1'b0: PLL-alive mode is disabled 1'b1: PLL-alive mode is enabled Before entering this mode, MAIN_ON shall be turned on first until MAIN_RDY is asserted
19:17	RO	**Reserved**
16	RW	**DP_TX_I_INTERNAL_LOOPBACK_EN** Internal loopback mode enable signal
15:14	RO	**Reserved**
13	RO	**DP_TX_O_AUX_RDY** Indicates that Aux Channel is ready 1'b0: aux channel is not ready 1'b1: aux channel is ready
12	RW	**DP_TX_I_AUX_ON** Aux Channel ON/OFF signal 1'b0: aux channel is turned off 1'b1: aux channel is turned on The signal shall not be de-asserted when AUX_RDY is low
11:10	RO	**Reserved**
9	RW	**DP_TX_O_MAIN_RDY** Indicates that Main Link is ready and starts to transmit differential data 1'b0: main link is not ready 1'b1: main link channel is ready and starts to transmit differential data
8	RW	**DP_TX_I_MAIN_ON** Main Link ON/OFF signal 1'b0: main link is turned off 1'b1: main link is turned on The signal shall not be de-asserted when MAIN_RDY is low
7: 5	RO	**Reserved**
4	RW	**DP_TX_I_MAIN_SSCG_ON** 1'b0: Disable Main Link Lane SSCG function 1'b1: Enable Main Link Lane SSCG function
3: 1	RO	**Reserved**
0	RW	**DP_TX_I_MAIN_DATAWIDTH** This input signal reports the width of the data bus that the PHY is configured for 20-bit mode or 40-bit mode 1'b0: 20-bit mode 1'b1: 40-bit mode

Offset: 108h **DPTX108: DPTX PHY Configuration Register** **Init = 0x22113121**

Bit	R/W	Description
31:30	RO	**Reserved**

to next page

from previous page

29:28	RW	**DP_TX_I_MAIN1_TXDEEMP_EN** Select transmitter deemphasis for Main Link Lane1 2'b00: 3.5dB 2'b01: 6dB 2'b10: 0dB 2'b11: 0dB
27:26	RO	**Reserved**
25:24	RW	**DP_TX_I_MAIN0_TXDEEMP_EN** Select transmitter deemphasis for Main Link Lane0 2'b00: 3.5dB 2'b01: 6dB 2'b10: 0dB 2'b11: 0dB
23:21	RO	**Reserved**
20	RW	**DP_TX_I_MAIN1_TX_SEL** It is used to select Main Link Lane1 pad 1'b0: Select MLTX1_P2/MLTX1_N2 1'b1: Select MLTX1_P1/MLTX1_N1
19:17	RO	**Reserved**
16	RW	**DP_TX_I_MAIN0_TX_SEL** It is used to select Main Link Lane0 pad 1'b0: Select MLTX0_P2/MLTX0_N2 1'b1: Select MLTX0_P1/MLTX0_N1
15:14	RO	**Reserved**
13:12	RW	**DP_TX_I_DP_PORT_SEL** Main Line Lane selection 2'b00: Reserved 2'b01: Only select main link lane0 2'b10: Only select main link lane1 2'b11: Both main link lane0 and lane1 are selected
11:10	RO	**Reserved**
9: 8	RW	**DP_TX_I_RATE** Control the link signaling rate 2'b00: 1.62Gbps signaling rate 2'b01: 2.7Gbps signaling rate 2'b10: 5.4Gbps signaling rate 2'b11: Reserved This signal can be changed only when Main Link is off
7: 6	RO	**Reserved**
5: 4	RW	**DP_TX_I_XTLSEL** The frequency selection of input clock source 2'b00: 10MHz input clock source 2'b01: 12MHz input clock source 2'b10: 25MHz input clock source 2'b11: 27MHz input clock source
3: 1	RO	**Reserved**
0	RW	**DP_TX_I_AUX_IE** Data input enable for AUX Channel

Offset: 10Ch	DPTX10C: DPTX PHY Status Register	Init = 0x22001221
Bit	**R/W**	**Description**
31:13	RO	**Reserved**
12	RO	**DP_TX_I_AUX_OE** Data output enable for AUX Channel
11: 9	RO	**Reserved**
8	RO	**DP_TX_I_AUX_VI** Asynchronous transmit data output for AUX Channel
7: 5	RO	**Reserved**
4	RO	**DP_TX_O_AUX_RCV** Asynchronous transmit data input for AUX Channel
3: 1	RO	**Reserved**
0	RO	**DP_TX_O_AUX_HSSQ** Indicates whether differential signaling is detected 1'b0: Differential data is signaling 1'b1: Differential data is not signaling

Offset: 110h	DPTX110: DPTX PHY Configuration And Status Register	Init = 0x00000C00
Bit	**R/W**	**Description**
31:28	RO	**Reserved**
27:24	RW	**DP_TX_I_VCONTROL** DP TX PHY Vendor control Vendordefined 4bit parallel input bus Default value is 4b0000
23:20	RO	**Reserved**
19:16	RW	**DP_TX_I_debug_sel** DP TX PHY Vendor reserved signal Please reserve this signal for SW control
15: 0	RO	**DP_TX_O_debug_out** DP TX PHY Vendor reserved signal Please reserve this signal for SW monitor

Offset: 114h	DPTX114: DPTX PHY Status Register	Init = 0x00002878
Bit	**R/W**	**Description**
31:15	RO	**Reserved**
14: 0	RO	**DP_TX_O_XCFGO** DP TX PHY Vendor reserved signal Please reserve this signal for SW monitor

Offset: 118h	DPTX118: DPTX PHY Configuration Register	Init = 0x00000000
Bit	**R/W**	**Description**
31: 0	RW	**DP_TX_I_XCFGI[31: 0]** DP TX PHY Vendor reserved signal

Offset: 11Ch	DPTX11C: DPTX PHY Configuration Register	Init = 0x00000000
Bit	**R/W**	**Description**
31: 0	RW	**DP_TX_I_XCFGI[63: 32]** DP TX PHY Vendor reserved signal

Offset: 120h	DPTX120: DPTX PHY Configuration Register	Init = 0x00000000
Bit	**R/W**	**Description**
31: 0	RW	**DP_TX_I_XCFGI[95: 64]** DP TX PHY Vendor reserved signal

Offset: 124h	DPTX124: DPTX PHY Configuration Register	Init = 0x00000000
Bit	**R/W**	**Description**
31: 0	RW	**DP_TX_I_XCFGI[127: 96]** DP TX PHY Vendor reserved signal

Offset: 128h	DPTX128: DPTX PHY Configuration Register	Init = 0x00000000
Bit	**R/W**	**Description**
31: 0	RW	**DP_TX_I_XCFGI[159:128]** DP TX PHY Vendor reserved signal

Offset: 12Ch	DPTX12C: DPTX PHY Configuration Register	Init = 0x00000000
Bit	**R/W**	**Description**
31: 2	RW	**Reserved**
1: 0	RW	**DP_TX_I_CORECLKIN_sel** DP TX PHY reference clock selection 0: 25MHz from SCU 1: 25MHz from PAD 2: 25MHz from PCIE port 1 3: 25MHz from PCIE port 2

Offset: 138h	DPTX138: DPTX PHY Configuration Register	Init = 0x00000000
Bit	**R/W**	**Description**
31	RO	**Reserved**
30:24	RW	**DP_TX_O_MAIN1_RXDATA_latch_sel** right shift amount of latched feedback DP TX lane 1 symbols for debugging
23	RO	**Reserved**
22:16	RW	**DP_TX_O_MAIN0_RXDATA_latch_sel** right shift amount of latched feedback DP TX lane 0 symbols for debugging
15: 5	RO	**Reserved**
4	RO	**DP_TX_O_MAIN_RXDATA_latch_ready** this bit indicates if DP_TX_O_MAIN0_RXDATA_127b and DP_TX_O_MAIN1_RXDATA_127b are valid
3: 1	RO	**Reserved**
0	RW	**DP_TX_O_MAIN_RXDATA_latch_cmd** write with 1 to latch feedback DP TX symbols for debugging and verification this bit will go back to 0 after one clock cycle

Offset: 13Ch	DPTX13C: DPTX PHY Status Register	Init = 0x80800000
Bit	**R/W**	**Description**
31:24	RW	**DP_TX_O_MAIN1_RXDATA_latch_sel_auto** this register return the right shift amount of latched feedback DP TX lane 1 symbols by automatic matching for debugging and verification
23:16	RW	**DP_TX_O_MAIN0_RXDATA_latch_sel_auto** this register return the right shift amount of latched feedback DP TX lane 0 symbols by automatic matching for debugging and verification
15: 0	RO	**Reserved**

Offset: 140h	DPTX140: DPTX PHY Status Register	Init = 0x00000000
Bit	**R/W**	**Description**
31: 0	RO	**DP_TX_O_MAIN0_RXDATA_127b[31: 0]** latched feedback DP TX lane 0 symbols

Offset: 144h	DPTX144: DPTX PHY Status Register	Init = 0x00000000
Bit	**R/W**	**Description**
31: 0	RO	**DP_TX_O_MAIN0_RXDATA_127b[63: 32]** latched feedback DP TX lane 0 symbols

Offset: 148h	DPTX148: DPTX PHY Status Register	Init = 0x00000000
Bit	**R/W**	**Description**
31: 0	RO	**DP_TX_O_MAIN0_RXDATA_127b[95: 64]** latched feedback DP TX lane 0 symbols

Offset: 14Ch	DPTX14C: DPTX PHY Status Register	Init = 0x00000000
Bit	**R/W**	**Description**
31	RO	**Reserved**
30: 0	RO	**DP_TX_O_MAIN0_RXDATA_127b[126: 96]** latched feedback DP TX lane 0 symbols

Offset: 150h	DPTX150: DPTX PHY Status Register	Init = 0x00000000
Bit	**R/W**	**Description**
31: 0	RO	**DP_TX_O_MAIN1_RXDATA_127b[31: 0]** latched feedback DP TX lane 1 symbols

Offset: 154h	DPTX154: DPTX PHY Status Register	Init = 0x00000000
Bit	**R/W**	**Description**
31: 0	RO	**DP_TX_O_MAIN1_RXDATA_127b[63: 32]** latched feedback DP TX lane 1 symbols

Offset: 158h	DPTX158: DPTX PHY Status Register	Init = 0x00000000
Bit	**R/W**	**Description**
31: 0	RO	**DP_TX_O_MAIN1_RXDATA_127b[95: 64]** latched feedback DP TX lane 1 symbols

Offset: 15Ch	DPTX15C: DPTX PHY Status Register	Init = 0x00000000
Bit	**R/W**	**Description**
31	RO	**Reserved**
30: 0	RO	**DP_TX_O_MAIN1_RXDATA_127b[126: 96]** latched feedback DP TX lane 1 symbols

Offset: 160h	DPTX160: DPTX Interrupt to VGA Configuration Register	Init = 0x00FFFF00
Bit	**R/W**	**Description**
31:24	RO	**Reserved**
23	RW	**aux_reply_timeout_int_2_vga_en** DP TX AUX channel timeout interrupt to vga enable
22	RW	**aux_reply_done_int_2_vga_en** DP TX AUX channel reply done interrupt to vga enable
21	RW	**hpd_event_int_2_vga_en** DP TX HPD event interrupt to vga enable
20	RW	**hpd_low_int_2_vga_en** DP TX HPD low interrupt to vga enable
19	RW	**hpd_irq_int_2_vga_en** DP TX HPD irq interrupt to vga enable
18:16	RO	**Reserved**
15	RW	**aux_reply_timeout_int_2_vga_mask** DP TX AUX channel timeout interrupt to vga mask 0: ignore interrupt raw status 1: reveal interrupt raw status
14	RW	**aux_reply_done_int_2_vga_mask** DP TX AUX channel reply done interrupt to vga mask 0: ignore interrupt raw status 1: reveal interrupt raw status
13	RW	**hpd_event_int_2_vga_mask** DP TX HPD event interrupt to vga mask 0: ignore interrupt raw status 1: reveal interrupt raw status
12	RW	**hpd_low_int_2_vga_mask** DP TX HPD low interrupt to vga mask 0: ignore interrupt raw status 1: reveal interrupt raw status
11	RW	**hpd_irq_int_2_vga_mask** DP TX HPD irq interrupt to vga mask 0: ignore interrupt raw status 1: reveal interrupt raw status
10: 8	RO	**Reserved**
7	WO	**aux_reply_timeout_int_2_vga_clr** DP TX AUX channel timeout interrupt to vga clear this bit will go back to 0 after one clock cycle

to next page

from previous page

6	WO	**aux_reply_done_int_2_vga_clr** DP TX AUX channel reply done interrupt to vga clear this bit will go back to 0 after one clock cycle
5	WO	**hpd_event_int_2_vga_clr** DP TX HPD event interrupt to vga clear this bit will go back to 0 after one clock cycle
4	WO	**hpd_low_int_2_vga_clr** DP TX HPD low interrupt to vga clear this bit will go back to 0 after one clock cycle
3	WO	**hpd_irq_int_2_vga_clr** DP TX HPD irq interrupt to vga clear this bit will go back to 0 after one clock cycle
2: 0	RO	**Reserved**

Offset: 164h DPTX164: DPTX Interrupt to VGA Status Register Init = 0x00000000

Bit	R/W	Description
31:16	RO	**Reserved**
15	RO	**aux_reply_timeout_int_2_vga_raw** DP TX AUX channel timeout interrupt to vga raw status
14	RO	**aux_reply_done_int_2_vga_raw** DP TX AUX channel reply done interrupt to vga raw status
13	RO	**hpd_event_int_2_vga_raw** DP TX HPD event interrupt to vga raw status
12	RO	**hpd_low_int_2_vga_raw** DP TX HPD low interrupt to vga raw status
11	RO	**hpd_irq_int_2_vga_raw** DP TX HPD irq interrupt to vga raw status
10: 8	RO	**Reserved**
7	RO	**aux_reply_timeout_int_2_vga** DP TX AUX channel timeout interrupt to vga
6	RO	**aux_reply_done_int_2_vga** DP TX AUX channel reply done interrupt to vga
5	RO	**hpd_event_int_2_vga** DP TX HPD event interrupt to vga
4	RO	**hpd_low_int_2_vga** DP TX HPD low interrupt to vga
3	RO	**hpd_irq_int_2_vga** DP TX HPD irq interrupt to vga
2: 0	RO	**Reserved**

Offset: 168h DPTX168: DPTX Software Interrupt to VGA Configuration Register Init = 0x00ffff00

Bit	R/W	Description
31:24	RO	**Reserved**
23	RW	**reg_bank_int_2_vga_en[7]** DP TX software interrupt 7 to vga enable
22	RW	**reg_bank_int_2_vga_en[6]** DP TX software interrupt 6 to vga enable

to next page

from previous page

21	RW	**reg_bank_int_2_vga_en[5]** DP TX software interrupt 5 to vga enable
20	RW	**reg_bank_int_2_vga_en[4]** DP TX software interrupt 4 to vga enable
19	RW	**reg_bank_int_2_vga_en[3]** DP TX software interrupt 3 to vga enable
18	RW	**reg_bank_int_2_vga_en[2]** DP TX software interrupt 2 to vga enable
17	RW	**reg_bank_int_2_vga_en[1]** DP TX software interrupt 1 to vga enable
16	RW	**reg_bank_int_2_vga_en[0]** DP TX software interrupt 0 to vga enable
15	RW	**reg_bank_int_2_vga_mask[7]** DP TX software interrupt 7 to vga mask 0: ignore interrupt raw status 1: reveal interrupt raw status
14	RW	**reg_bank_int_2_vga_mask[6]** DP TX software interrupt 6 to vga mask 0: ignore interrupt raw status 1: reveal interrupt raw status
13	RW	**reg_bank_int_2_vga_mask[5]** DP TX software interrupt 5 to vga mask 0: ignore interrupt raw status 1: reveal interrupt raw status
12	RW	**reg_bank_int_2_vga_mask[4]** DP TX software interrupt 4 to vga mask 0: ignore interrupt raw status 1: reveal interrupt raw status
11	RW	**reg_bank_int_2_vga_mask[3]** DP TX software interrupt 3 to vga mask 0: ignore interrupt raw status 1: reveal interrupt raw status
10	RW	**reg_bank_int_2_vga_mask[2]** DP TX software interrupt 2 to vga mask 0: ignore interrupt raw status 1: reveal interrupt raw status
9	RW	**reg_bank_int_2_vga_mask[1]** DP TX software interrupt 1 to vga mask 0: ignore interrupt raw status 1: reveal interrupt raw status
8	RW	**reg_bank_int_2_vga_mask[0]** DP TX software interrupt 0 to vga mask 0: ignore interrupt raw status 1: reveal interrupt raw status
7	WO	**reg_bank_int_2_vga_clr[7]** DP TX software interrupt 7 to vga clear this bit will go back to 0 after one clock cycle
6	WO	**reg_bank_int_2_vga_clr[6]** DP TX software interrupt 6 to vga clear this bit will go back to 0 after one clock cycle

to next page

from previous page

5	WO	**reg_bank_int_2_vga_clr[5]** DP TX software interrupt 5 to vga clear this bit will go back to 0 after one clock cycle
4	WO	**reg_bank_int_2_vga_clr[4]** DP TX software interrupt 4 to vga clear this bit will go back to 0 after one clock cycle
3	WO	**reg_bank_int_2_vga_clr[3]** DP TX software interrupt 3 to vga clear this bit will go back to 0 after one clock cycle
2	WO	**reg_bank_int_2_vga_clr[2]** DP TX software interrupt 2 to vga clear this bit will go back to 0 after one clock cycle
1	WO	**reg_bank_int_2_vga_clr[1]** DP TX software interrupt 1 to vga clear this bit will go back to 0 after one clock cycle
0	WO	**reg_bank_int_2_vga_clr[0]** DP TX software interrupt 0 to vga clear this bit will go back to 0 after one clock cycle

Offset: 16Ch DPTX16C: DPTX Software Interrupt to VGA Status Register Init = 0x00000000

Bit	R/W	Description
31:16	RO	**Reserved**
15	RO	**reg_bank_int_2_vga_raw[7]** DP TX software interrupt 7 to vga raw status
14	RO	**reg_bank_int_2_vga_raw[6]** DP TX software interrupt 6 to vga raw status
13	RO	**reg_bank_int_2_vga_raw[5]** DP TX software interrupt 5 to vga raw status
12	RO	**reg_bank_int_2_vga_raw[4]** DP TX software interrupt 4 to vga raw status
11	RO	**reg_bank_int_2_vga_raw[3]** DP TX software interrupt 3 to vga raw status
10	RO	**reg_bank_int_2_vga_raw[2]** DP TX software interrupt 2 to vga raw status
9	RO	**reg_bank_int_2_vga_raw[1]** DP TX software interrupt 1 to vga raw status
8	RO	**reg_bank_int_2_vga_raw[0]** DP TX software interrupt 0 to vga raw status
7	RO	**reg_bank_int_2_vga[7]** DP TX software interrupt 7 to vga
6	RO	**reg_bank_int_2_vga[6]** DP TX software interrupt 6 to vga
5	RO	**reg_bank_int_2_vga[5]** DP TX software interrupt 5 to vga
4	RO	**reg_bank_int_2_vga[4]** DP TX software interrupt 4 to vga
3	RO	**reg_bank_int_2_vga[3]** DP TX software interrupt 3 to vga

to next page

from previous page

2	RO	**reg_bank_int_2_vga[2]** DP TX software interrupt 2 to vga
1	RO	**reg_bank_int_2_vga[1]** DP TX software interrupt 1 to vga
0	RO	**reg_bank_int_2_vga[0]** DP TX software interrupt 0 to vga

Offset: 180h DPTX180: DPTX Video Line Buffer Configuration Register Init = 0x00003F3F

Bit	R/W	Description
31:29	RO	**Reserved**
28	RW	**video_line_buf_pat_en** this bit is set to enable video line buffer pattern generation
27:24	RW	**video_line_buf_pat_sel** video line buffer pattern selection 0: R component is fixed at 127 1: G component is fixed at 127 2: B component is fixed at 127 3: R/G/B component are fixed at 127 4: R component is increased by pixel counter 5: G component is increased by pixel counter 6: B component is increased by pixel counter 7: R/G/B component are increased by pixel counter 8: R component is increased by line counter 9: G component is increased by line counter 10: B component is increased by line counter 11: R/G/B component are increased by line counter 12: R component is increased by frame counter 13: G component is increased by frame counter 14: B component is increased by frame counter 15: R/G/B component are increased by frame counter
23:21	RO	**Reserved**
20	RW	**video_line_buf_en** this bit is set to enable video line buffer
19	RO	**Reserved**
18	RW	**video_line_buf_hsync_pol_sel** this register decides whether or not to invert polarity of HSYNC from VGA output 0: NOT to invert polarity of HSYNC from VGA output 1: to invert polarity of HSYNC from VGA output
17	RW	**video_line_buf_vsync_pol_sel** this register decides whether or not to invert polarity of VSYNC from VGA output 0: NOT to invert polarity of VSYNC from VGA output 1: to invert polarity of VSYNC from VGA output
16	RW	**video_line_buf_timing_sel** this register decides timing mode selection for HSYNC, VSYNC, HBLANK, VBLANK and DE signals 0: video input format; there can be both front and back porch 1: retiming format; front porch is eliminated and HSYNC/VSYNC are always located at the very beginning of HBLANK/VBLANK respectively
15	RO	**Reserved**

to next page

from previous page

14: 8	RW	**video_line_buf_empty_latency** this register decides how many pixel clock cycles from the last active pixel to DP BS (blanking start) control symbol
7	RO	**Reserved**
6: 0	RW	**video_line_buf_ready_latency** this register decides how many pixel clock cycles from the first active pixel to DP BE (blanking end) control symbol

60 DisplayPort Micro Controller Unit (DPMCU)

60.1 Overview

DisplayPort Micro Control Unit

Base address of DPMCU = 0x1800_0000
Physical address = (Base address of DPMCU) + Offset

60.2 Registers : Base Address = 0x1800:0000

Offset: 10000h	DPMCU10000: DPMCU Register Bank	Init = 0x00000000
Bit	**R/W**	**Description**
31: 0	RO	**reg_00** register 00

Offset: 10004h	DPMCU10004: DPMCU Register Bank	Init = 0x00000001
Bit	**R/W**	**Description**
31: 0	RO	**reg_01** register 01

Offset: 10008h	DPMCU10008: DPMCU Register Bank	Init = 0x00000002
Bit	**R/W**	**Description**
31: 0	RO	**reg_02** register 02

Offset: 1000Ch	DPMCU1000C: DPMCU Register Bank	Init = 0x00000003
Bit	**R/W**	**Description**
31: 0	RO	**reg_03** register 03

Offset: 10010h	DPMCU10010: DPMCU Register Bank	Init = 0x00000004
Bit	**R/W**	**Description**
31: 0	RO	**reg_04** register 04

Offset: 10014h	DPMCU10014: DPMCU Register Bank	Init = 0x00000005
Bit	**R/W**	**Description**
31: 0	RO	**reg_05** register 05

Offset: 10018h	DPMCU10018: DPMCU Register Bank	Init = 0x00000006

Bit	R/W	Description
31: 0	RO	**reg_06** register 06

Offset: 1001Ch	DPMCU1001C: DPMCU Register Bank	Init = 0x00000007

Bit	R/W	Description
31: 0	RO	**reg_07** register 07

Offset: 10020h	DPMCU10020: DPMCU Register Bank	Init = 0x00000008

Bit	R/W	Description
31: 0	RO	**reg_08** register 08

Offset: 10024h	DPMCU10024: DPMCU Register Bank	Init = 0x00000009

Bit	R/W	Description
31: 0	RO	**reg_09** register 09

Offset: 10028h	DPMCU10028: DPMCU Register Bank	Init = 0x0000000A

Bit	R/W	Description
31: 0	RO	**reg_10** register 10

Offset: 1002Ch	DPMCU1002C: DPMCU Register Bank	Init = 0x0000000B

Bit	R/W	Description
31: 0	RO	**reg_11** register 11

Offset: 10030h	DPMCU10030: DPMCU Register Bank	Init = 0x0000000C

Bit	R/W	Description
31: 0	RO	**reg_12** register 12

Offset: 10034h	DPMCU10034: DPMCU Register Bank	Init = 0x0000000D

Bit	R/W	Description
31: 0	RO	**reg_13** register 13

Offset: 10038h	DPMCU10038: DPMCU Register Bank	Init = 0x0000000E

Bit	R/W	Description
31: 0	RO	**reg_14** register 14

Offset: 1003Ch	DPMCU1003C: DPMCU Register Bank	Init = 0x0000000F

Bit	R/W	Description
31: 0	RO	**reg_15** register 15

Offset: 10040h	DPMCU10040: DPMCU Register Bank	Init = 0x00000010

Bit	R/W	Description
31: 0	RO	**reg_16** register 16

Offset: 10044h	DPMCU10044: DPMCU Register Bank	Init = 0x00000011

Bit	R/W	Description
31: 0	RO	**reg_17** register 17

Offset: 10048h	DPMCU10048: DPMCU Register Bank	Init = 0x00000012

Bit	R/W	Description
31: 0	RO	**reg_18** register 18

Offset: 1004Ch	DPMCU1004C: DPMCU Register Bank	Init = 0x00000013

Bit	R/W	Description
31: 0	RO	**reg_19** register 19

Offset: 10050h	DPMCU10050: DPMCU Register Bank	Init = 0x00000014

Bit	R/W	Description
31: 0	RO	**reg_20** register 20

Offset: 10054h	DPMCU10054: DPMCU Register Bank	Init = 0x00000015

Bit	R/W	Description
31: 0	RO	**reg_21** register 21

Offset: 10058h	DPMCU10058: DPMCU Register Bank	Init = 0x00000016

Bit	R/W	Description
31: 0	RO	**reg_22** register 22

Offset: 1005Ch	DPMCU1005C: DPMCU Register Bank	Init = 0x00000017

Bit	R/W	Description
31: 0	RO	**reg_23** register 23

Offset: 10060h	DPMCU10060: DPMCU Register Bank	Init = 0x00000018

Bit	R/W	Description
31: 0	RO	**reg_24** register 24

Offset: 10064h	DPMCU10064: DPMCU Register Bank	Init = 0x00000019

Bit	R/W	Description
31: 0	RO	**reg_25** register 25

Offset: 10068h	DPMCU10068: DPMCU Register Bank	Init = 0x0000001A

Bit	R/W	Description
31: 0	RO	**reg_26** register 26

Offset: 1006Ch	DPMCU1006C: DPMCU Register Bank	Init = 0x0000001B

Bit	R/W	Description
31: 0	RO	**reg_27** register 27

Offset: 10070h	DPMCU10070: DPMCU Register Bank	Init = 0x00000000

Bit	R/W	Description
31: 0	RO	**reg_28** register 28

Offset: 10074h	DPMCU10074: DPMCU Register Bank	Init = 0x00000000

Bit	R/W	Description
31: 0	RO	**reg_29** register 29

Offset: 10078h	DPMCU10078: DPMCU Register Bank	Init = 0x00000000

Bit	R/W	Description
31: 0	RO	**reg_30** register 30

Offset: 1007Ch	DPMCU1007C: DPMCU Register Bank	Init = 0x00000000

Bit	R/W	Description
31: 0	RO	**reg_31** register 31

Offset: 10080h	DPMCU10080: DPMCU Pause Configuration Register	Init = 0x00000000

Bit	R/W	Description
31:17	RO	**Reserved**
16	WO	**pc_pause_step_pulse** write 1 to step by one instruction when pc_pause_en_force = 1 or pc_pause_en_bp = 1
15:13	RO	**Reserved**
12	RW	**pc_pause_en_force** 0: DP MCU runs in normal state 1: force DP MCU to pause
11: 9	RO	**Reserved**
8	RO	**pc_pause_en_wfi** this bit indicates if DP MCU is paused in WFI state (wait for interrupt)
7: 5	RO	**Reserved**
4	RO	**pc_pause_en_bp** this bit indicates if DP MCU is paused by the program counter reaches the break point (pc_curr == pc_bp) when pc_bp_en = 1
3: 1	RO	**Reserved**
0	RO	**pc_pause_en** this bit indicates if DP MCU is paused (ORed by pc_pause_en_force, pc_pause_en_wfi and pc_pause_en_bp)

Offset: 10084h	DPMCU10084: DPMCU Break Point Configuration Register	Init = 0x00000000

Bit	R/W	Description
31:21	RO	**Reserved**
20	RW	**pc_bp_en** 0: PC is NOT paused when it reaches pc_bp 1: PC is paused when it reaches pc_bp
19:17	RO	**Reserved**
16: 2	RW	**pc_bp_16_02** pc_bp[16:2] point for PC
1: 0	RO	**pc_bp_01_00** pc_bp[1:0] always 0

Offset: 10088h	DPMCU10088: DPMCU Next Program Counter Register	Init = 0x00000000
Bit	**R/W**	**Description**
31:17	RO	**pc_next_imem_base** 15-bit MSBs of instruction memory base byte address
16: 0	RO	**pc_next** next program counter

Offset: 1008Ch	DPMCU1008C: DPMCU Current Program Counter Register	Init = 0x00000000
Bit	**R/W**	**Description**
31:17	RO	**pc_curr_imem_base** 15-bit MSBs of instruction memory base byte address
16: 0	RO	**pc_curr** current program counter

Offset: 10090h	DPMCU10090: DPMCU Next Instruction Register	Init = 0x00000000
Bit	**R/W**	**Description**
31: 0	RO	**inst_next** instruction at next program counter

Offset: 10094h	DPMCU10094: DPMCU Current Instruction Register	Init = 0x2C000000
Bit	**R/W**	**Description**
31: 0	RO	**inst_curr** instruction at current program counter

Offset: 10098h	DPMCU10098: DPMCU Interrupt Enable Status Register	Init = 0xFFFFFFFF
Bit	**R/W**	**Description**
31: 0	RO	**int_en** interrupt enable bits controlled by instruction DP_MCU_INTEN

Offset: 1009Ch	DPMCU1009C: DPMCU Interrupt Enable Status Register	Init = 0x00000000
Bit	**R/W**	**Description**
31: 0	RO	**int_src_d2t** interrupt sources viewed by DP MCU

Offset: 100D0h	DPMCU100D0: DPMCU Version Register	Init = 0x20190828
Bit	**R/W**	**Description**
31: 0	RO	**version** DP MCU version

Offset: 100E0h	DPMCU100E0: DPMCU Configuration Register	Init = 0x10550010
Bit	**R/W**	**Description**
31:29	RO	**Reserved**

to next page

from previous page

28	RW	**config_en** write protect of imem_sel, imem_clk_off, imem_sleep, imem_shut_down, dmem_clk_off, dmem_sleep, dmem_shut_down, core_soft_resetn, ahbm_soft_resetn, ahbs_soft_resetn and ahbs_imem_en 0: NOT programmable 1: programmable
27:25	RO	**Reserved**
24	RW	**imem_sel** Reserved
23	RO	**Reserved**
22	RW	**imem_clk_off** 0: enable internal instruction memory clock 1: disable internal instruction memory clock
21	RW	**imem_sleep** 0: bring internal instruction memory out of sleep state 1: place internal instruction memory in sleep state
20	RW	**imem_shut_down** 0: bring internal instruction memory out of shut down state 1: place internal instruction memory in shut down state
19	RO	**Reserved**
18	RW	**dmem_clk_off** 0: enable internal data memory clock 1: disable internal data memory clock
17	RW	**dmem_sleep** 0: bring internal data memory out of sleep state 1: place internal data memory in sleep state
16	RW	**dmem_shut_down** 0: bring internal data memory out of shut down state 1: place internal data memory in shut down state
15:13	RO	**Reserved**
12	RW	**core_soft_resetn** 0: place DP MCU core in reset state 1: bring DP MCU core out of reset state
11: 9	RO	**Reserved**
8	RW	**ahbm_soft_resetn** 0: place DP MCU AHB master port in reset state 1: bring DP MCU AHB master port out of reset state
7: 5	RO	**Reserved**
4	RW	**ahbs_soft_resetn** 0: place DP MCU AHB slave port in reset state 1: bring DP MCU AHB slave port and the corresponding registers out of reset state
3: 1	RO	**Reserved**
0	RW	**ahbs_imem_en** 0: internal instruction memory is accessed by DP MCU only 1: internal instruction memory is accessed by external AHB masters

Offset: 100E8h DPMCU100E8: DPMCU Software Interrupt to VGA Configuration Register Init = 0x00ffff00

Bit	R/W	Description
31:24	RO	**Reserved**
23	RW	**reg_bank_int_2_vga_en[7]** DP TX software interrupt 7 to vga enable
22	RW	**reg_bank_int_2_vga_en[6]** DP TX software interrupt 6 to vga enable
21	RW	**reg_bank_int_2_vga_en[5]** DP TX software interrupt 5 to vga enable
20	RW	**reg_bank_int_2_vga_en[4]** DP TX software interrupt 4 to vga enable
19	RW	**reg_bank_int_2_vga_en[3]** DP TX software interrupt 3 to vga enable
18	RW	**reg_bank_int_2_vga_en[2]** DP TX software interrupt 2 to vga enable
17	RW	**reg_bank_int_2_vga_en[1]** DP TX software interrupt 1 to vga enable
16	RW	**reg_bank_int_2_vga_en[0]** DP TX software interrupt 0 to vga enable
15	RW	**reg_bank_int_2_vga_mask[7]** DP TX software interrupt 7 to vga mask 0: ignore interrupt raw status 1: reveal interrupt raw status
14	RW	**reg_bank_int_2_vga_mask[6]** DP TX software interrupt 6 to vga mask 0: ignore interrupt raw status 1: reveal interrupt raw status
13	RW	**reg_bank_int_2_vga_mask[5]** DP TX software interrupt 5 to vga mask 0: ignore interrupt raw status 1: reveal interrupt raw status
12	RW	**reg_bank_int_2_vga_mask[4]** DP TX software interrupt 4 to vga mask 0: ignore interrupt raw status 1: reveal interrupt raw status
11	RW	**reg_bank_int_2_vga_mask[3]** DP TX software interrupt 3 to vga mask 0: ignore interrupt raw status 1: reveal interrupt raw status
10	RW	**reg_bank_int_2_vga_mask[2]** DP TX software interrupt 2 to vga mask 0: ignore interrupt raw status 1: reveal interrupt raw status
9	RW	**reg_bank_int_2_vga_mask[1]** DP TX software interrupt 1 to vga mask 0: ignore interrupt raw status 1: reveal interrupt raw status

to next page

from previous page

8	RW	**reg_bank_int_2_vga_mask[0]** DP TX software interrupt 0 to vga mask 0: ignore interrupt raw status 1: reveal interrupt raw status
7	WO	**reg_bank_int_2_vga_clr[7]** DP TX software interrupt 7 to vga clear this bit will go back to 0 after one clock cycle
6	WO	**reg_bank_int_2_vga_clr[6]** DP TX software interrupt 6 to vga clear this bit will go back to 0 after one clock cycle
5	WO	**reg_bank_int_2_vga_clr[5]** DP TX software interrupt 5 to vga clear this bit will go back to 0 after one clock cycle
4	WO	**reg_bank_int_2_vga_clr[4]** DP TX software interrupt 4 to vga clear this bit will go back to 0 after one clock cycle
3	WO	**reg_bank_int_2_vga_clr[3]** DP TX software interrupt 3 to vga clear this bit will go back to 0 after one clock cycle
2	WO	**reg_bank_int_2_vga_clr[2]** DP TX software interrupt 2 to vga clear this bit will go back to 0 after one clock cycle
1	WO	**reg_bank_int_2_vga_clr[1]** DP TX software interrupt 1 to vga clear this bit will go back to 0 after one clock cycle
0	WO	**reg_bank_int_2_vga_clr[0]** DP TX software interrupt 0 to vga clear this bit will go back to 0 after one clock cycle

Offset: 100ECh DPMCU100EC: DPMCU Software Interrupt to VGA Status Register Init = 0x00000000

Bit	R/W	Description
31:16	RO	**Reserved**
15	RO	**reg_bank_int_2_vga_raw[7]** DP TX software interrupt 7 to vga raw status
14	RO	**reg_bank_int_2_vga_raw[6]** DP TX software interrupt 6 to vga raw status
13	RO	**reg_bank_int_2_vga_raw[5]** DP TX software interrupt 5 to vga raw status
12	RO	**reg_bank_int_2_vga_raw[4]** DP TX software interrupt 4 to vga raw status
11	RO	**reg_bank_int_2_vga_raw[3]** DP TX software interrupt 3 to vga raw status
10	RO	**reg_bank_int_2_vga_raw[2]** DP TX software interrupt 2 to vga raw status
9	RO	**reg_bank_int_2_vga_raw[1]** DP TX software interrupt 1 to vga raw status
8	RO	**reg_bank_int_2_vga_raw[0]** DP TX software interrupt 0 to vga raw status

to next page

from previous page

7	RO	**reg_bank_int_2_vga[7]** DP TX software interrupt 7 to vga
6	RO	**reg_bank_int_2_vga[6]** DP TX software interrupt 6 to vga
5	RO	**reg_bank_int_2_vga[5]** DP TX software interrupt 5 to vga
4	RO	**reg_bank_int_2_vga[4]** DP TX software interrupt 4 to vga
3	RO	**reg_bank_int_2_vga[3]** DP TX software interrupt 3 to vga
2	RO	**reg_bank_int_2_vga[2]** DP TX software interrupt 2 to vga
1	RO	**reg_bank_int_2_vga[1]** DP TX software interrupt 1 to vga
0	RO	**reg_bank_int_2_vga[0]** DP TX software interrupt 0 to vga

61 Secure Boot Controller (SB)

61.1 Overview

Secure Boot Controller (SB) includes OTP memory, OTP controller, DMA engine, crypto engine, boot controller, micro processor (SBMCU) and ROM code. When the secure boot function is enabled by hardware strap or OTP config, the SBMCU will start to execute codes inside the ROM. The secure engines in Secure Boot SBMCU is dedicated for the secure boot. The secure data are stored inside OTP. User can program the OTP with software utility.

Base address of Timer = 0x1E6F_2000
Physical address = (Base address of Timer) + Offset

SEC00: Protection Key Register
SEC04: OTP Command Trigger Register
SEC08: OTP Timing Register
SEC10: OTP Address Register
SEC14: Secure Engine Status Register
SEC18: OTP Programming Status Register
SEC20: OTP Data Compare Register 1
SEC24: OTP Data Compare Register 2
SEC28: OTP Data Compare Register 3
SEC2C: OTP Data Compare Register 4
SEC30: OTPTRAP data read back 1
SEC34: OTPTRAP data read back 2
SEC38: OTP QRR data read back
SEC40: OTP QSR data read back
SEC44: OTP QMR data read back
SEC48: OTP QMRA and QMRB data read back
SEC50: Extra Programming Protection Range Register
SEC54: Extra Read Protection Range Register
SEC58: Secure Boot Engine Internal Controller Register
SEC5C: Secure Boot Engine Internal Controller Register
SEC60: Secure Boot Hardware Revision Register 1
SEC64: Secure Boot Hardware Revision Register 2
SEC68: Secure Boot Software Revision Register 1
SEC6C: Secure Boot Software Revision Register 2
SEC70: Secure Boot from SPI Status Register
SEC74: Secure Boot from eMMC Status Register
SEC78: Secure Boot Key Number Registers
SEC80: Secure Boot Engine Internal Controller Register
SEC84: Secure Boot Engine Internal Controller Register
SEC88: Software ECC Control Register
SEC90: Secure Boot Counter Register
SEC94: Secure Boot Counter 2 Register
SEC98: Secure Boot Engine Internal Controller Register
SECA0: eMMC Boot Watchdog Control Register
SECA4: eMMC Boot Watchdog Control Register
SECA8: eMMC Boot Watchdog Control Register
SECB0: Secure Boot Engine Internal Controller Register
SECB4: Secure Boot Engine Internal Controller Register
SEC800: Secure Boot DMA Enable Register
SEC808: Secure Boot DMA Mode Register
SEC80C: Secure Vault Key Control Register
SEC810: Secure Boot Digest Status
SEC814: Secure Boot Digest Check Status
SEC820: Secure Crypto Engine Enable Register

SEC840: Secure Boot DMA Source Address Register
SEC844: Secure Boot DMA Destination Address Register
SEC848: Secure Boot DMA Size Register
SEC854: Secure Boot Crypto AES-GCM AAD Size Register
SEC858: Secure Boot Hash Mode Register
SEC85C: Secure Boot Hash Engine Fire Register
SEC860: Secure Boot Crypto Mode Register
SEC864: Secure Boot Crypto Data Size Register
SEC868: Secure Boot Crypto Data Total Size Register
SEC870: Secure Boot Crypto Low Key Write Trigger Register
SEC874: Secure Boot Crypto High Key Write Trigger Register
SEC878: Secure Boot Crypto Vector Write Trigger Register
SEC87C: Secure Boot Crypto Engine Fire Register
SEC880: Secure Boot Crypto Data Buffer 0 Register
SEC884: Secure Boot Crypto Data Buffer 1 Register
SEC888: Secure Boot Crypto Data Buffer 2 Register
SEC88C: Secure Boot Crypto Data Buffer 3 Register
SEC890: Secure Boot Crypto AES-GCM GHash Key Write Trigger Register
SEC8A0: Secure Boot Crypto Key Buffer 0 Register
SEC8A4: Secure Boot Crypto Key Buffer 1 Register
SEC8A8: Secure Boot Crypto Key Buffer 2 Register
SEC8AC: Secure Boot Crypto Key Buffer 3 Register
SEC8B0: Secure Boot Crypto Key Buffer 4 Register
SEC8B4: Secure Boot Crypto Key Buffer 5 Register
SEC8B8: Secure Boot Crypto Key Buffer 6 Register
SEC8BC: Secure Boot Crypto Key Buffer 7 Register
SEC8C0 ~ SEC8FC: Secure Boot Image Digest Read Back #0 ~ #16
SEC900: Secure Boot First Vault Key 0 Register
SEC904: Secure Boot First Vault Key 1 Register
SEC908: Secure Boot First Vault Key 2 Register
SEC90C: Secure Boot First Vault Key 3 Register
SEC910: Secure Boot First Vault Key 4 Register
SEC914: Secure Boot First Vault Key 5 Register
SEC918: Secure Boot First Vault Key 6 Register
SEC91C: Secure Boot First Vault Key 7 Register
SEC920: Secure Boot Second Vault Key 0 Register
SEC924: Secure Boot Second Vault Key 1 Register
SEC928: Secure Boot Second Vault Key 2 Register
SEC92C: Secure Boot Second Vault Key 3 Register
SEC930: Secure Boot Second Vault Key 4 Register
SEC934: Secure Boot Second Vault Key 5 Register
SEC938: Secure Boot Second Vault Key 6 Register
SEC93C: Secure Boot Second Vault Key 7 Register

Offset: 000h	SEC00: Protection Key Register	Init = 0

Bit	R/W	Reset	Description
31:0	RW	RstARM	**Protection Key** This register is designed to protect SCU registers from unpredictable updates, especially when ARM CPU is out of control. The password of the protection key is **0x349fe38a**. **Unlock SB registers**: Write 0x349fe38a to this register **Lock SB registers**: Write others value to this register **Only firmware can lock the secure boot registers, other softwares (ex. system BIOS/driver) can not do this to prevent disturbing the operation of firmware.** When this register is unlocked, the read back value of this register is 0x00000001. When this register is locked, the read back value of this register is 0x00000000.

Offset: 004h	SEC04: OTP Command Trigger Register	Init = 0

Bit	R/W	Reset	Description
31:0	WT	-	**OTP Command Trigger Register** This register is designed to trigger OTP controller to issue control signals to OTP memory. Write a specified command code can trigger corresponding command. OTP data read command for non-secure region is **0x23b1e361**. (This is a dual DW (64-bit) read command. The data will be stored in SEC20 and SEC24. If ECC is enabled, the data will be repaired by ECC code.) OTP mode register write command is **0x23b1e362**. OTP compare command is **0x23b1e363**. (This is a quad DW (128-bit) command command. The compare data should be stored in SEC20, SEC24, SEC28 and SEC2C before compare. The compare flag will be in SEC14[0]) OTP program command is **0x23b1e364**. OTP precharge command is **0x23b1e365**.

Offset: 008h	SEC08: OTP Timing Register	Init = 0x04190760

Bit	R/W	Reset	Description
27:24	RW	RstARM	**OTP read cycle time** The write cycle time in unit of 40ns
23:16	RW	RstARM	**OTP write wait time** The write wait time in unit of 40ns
15:0	RW	RstARM	**OTP write cycle time** The write cycle time in unit of 40ns

Offset: 010h	SEC10: OTP Address Register	Init = x

Bit	R/W	Reset	Description
15:0	RW	RstARM	**OTP Address Registe** The OTP address register

Offset: 014h	SEC14: Secure Engine Status Register	Init = x

Bit	R/W	Reset	Description
31:15	RO	-	**Reserved(0)**
14	RO	-	**Enable boot SPI or eMMC ABR (second boot)(trap_en_bspiabr** Same as SCU510[11]
13	RO	RstFull	**ABR Image Source** 0: First ABR Image 1: Second ABR Image
12	RO	RstFull	**ABR Image Source when Boot from SPI** 0: First ABR Image Source when Boot from SPI 1: Second ABR Image Source when Boot from SPI
11	RO	RstPwr	**Secure Boot Crypto Engine Key Expansion Done** 0: Crypto Engine Key Expansion is busy 1: Crypto Engine Key Expansion is done
10	RO	-	**Secure Boot Crypto Engine Busy** 0: Crypto Engine is idle 1: Crypto Engine is busy
9	RO	-	**OTP Program Protected** 0: Last OTP program command is completed 1: Last OTP program command is write protected
8	RO	-	**OTP Program Protect** 0: SEC10 OTP address is not protected 1: SEC10 OTP address is write protected
7	RO	-	**Low Security Key Mode** 0: Normal Mode 1: Low Security Mode is enabled
6	RO	-	**Secure Boot Mode** 0: Normal Mode 1: Secure Boot Mode is enabled
5	RO	-	**Boot from Uart Mode** 0: Boot from Uart mode 1: Boot from SPI or eMMC mode
4	RO	-	**Reserved(0)**
3	RO	-	**OTP internal charge pump status** 0: OTP internal charge pump not ready 1: OTP internal charge pump ready
2	RO	RstARM	**OTP controller status** 0: OTP controller busy 1: OTP controller idle
1	RO	RstARM	**OTP memory status** 0: OTP memory busy 1: OTP memory idle
0	RO	RstPwr	**OTP compare status** 0: Quad DW read compare fail 1: Quad DW read compare pass

Offset: 018h	SEC18: OTP Programming Status Register	Init = x

Bit	R/W	Reset	Description
31:2	RO	-	**Reserved(0)**
1	RO	-	**OTP Program Protected (same as SEC10[9])** 0: Last OTP program command is completed 1: Last OTP program command is write protected
0	RO	-	**OTP Program Protect (same as SEC10[8]** 0: SEC10 OTP address is not protected 1: SEC10 OTP address is write protected

Offset: 020h	SEC20: OTP Data Compare Register 1	Init = x

Bit	R/W	Reset	Description
31:0	RW	-	**OTP Data Compare Register 1** This is the first DW for read compare.

Offset: 024h	SEC24: OTP Data Compare Register 2	Init = x

Bit	R/W	Reset	Description
31:0	RW	-	**OTP Data Compare Register 2** This is the second DW for read compare.

Offset: 028h	SEC28: OTP Data Compare Register 3	Init = x

Bit	R/W	Reset	Description
31:0	RW	-	**OTP Data Compare Register 3** This is the third DW for read compare.

Offset: 02Ch	SEC2C: OTP Data Compare Register 4	Init = x

Bit	R/W	Reset	Description
31:0	RO	-	**OTP Data Compare Register 4** This is the forth DW for read compare.

Offset: 030h	SEC30: OTPTRAP data read back 1	Init = 0

Bit	R/W	Reset	Description
31:0	RO	-	**OTPTRAP data read back 1** This is the read back of OTPTRAP[31:0].

Offset: 034h	SEC34: OTPTRAP data read back 2	Init = 0

Bit	R/W	Reset	Description
31:0	RO	-	**OTPTRAP data read back 2** This is the read back of OTPTRAP[63:32].

Offset: 038h	SEC38: OTP QRR data read back	Init = 0

Bit	R/W	Reset	Description
31:0	RO	-	**OTP QRR data read back** This is the read back of OTP QRR[31:0]. It is same as OTPCFG8 and OTPCFG9. Please reference OTPCFG8

Offset: 040h	SEC40: OTP QSR data read back	Init = x

Bit	R/W	Reset	Description
31:0	RO	-	**OTP QSR data read back** This is the read back of OTP QSR[31:0]. It is same as OTPCFG0 and OTPCFG1. Please reference OTPCFG0

Offset: 044h	SEC44: OTP QMR data read back	Init = x

Bit	R/W	Reset	Description
31:0	RO	-	**OTP QMR data read back** This is the read back of OTP QMR[31:0]. Please reference the document of chapter ”12 OTP Programming Guide” of ”Secure Boot User Guide”.

Offset: 048h	SEC48: OTP QMRA and QMRB data read back	Init = 0

Bit	R/W	Reset	Description
15:0	RO	-	**OTP QMRA data read back** This is the read back of OTP QMRA[15:0]. Please reference the document of chapter ”12 OTP Programming Guide” of ”Secure Boot User Guide”.
31:16	RO	-	**OTP QMRB data read back** This is the read back of OTP QMRB[15:0]. Please reference the document of chapter ”12 OTP Programming Guide” of ”Secure Boot User Guide”.

Offset: 050h	SEC50: Extra Programming Protection Range Register	Init = 0

Bit	R/W	Reset	Description
31: 6	RO	-	**Reserved(0)**
6	RW1S	RstARM	**Write Protection of this register SEC50** This register can be set by software.
5: 0	RW	RstARM	**Extra Programming Protection Range Register** Software can set extra OTP programming protection range to replace the secure range in OTP Configuration register. The Extra Programming Protection Range will be effective only when its range is larger than the range of OTP Configuration registers. The update for this register is allowed only when the new value is larger than old value.

Offset: 054h	SEC54: Extra Read Protection Range Register	Init = 0

Bit	R/W	Reset	Description
31:28	RO	-	**Reserved(0)**
6	RW1S	RstARM	**Write Protection of this register SEC54** This register can be set by software.

to next page

from previous page

5: 0	RW	RstARM	**Extra Read Protection Range Register** Software can set extra OTP read protection range to replace the secure range in OTP Configuration register. The Extra Read Protection Range will be effective only when its range is larger than the range of OTP Configuration registers. The update for this register is allowed only when the new value is larger than old value.

Offset: 058h **SEC58: Secure Boot Engine Internal Controller Register** **Init = 3**

Bit	R/W	Reset	Description
31:2	RO	-	**Reserved(0)**
1	RW1S	RstARM	**Write Protection of this register SEC58** This register is set by Secure Boot engine.
0	RW	RstARM	**Secure Boot Engine Internal Controller Register**

Offset: 05Ch **SEC5C: Secure Boot Engine Internal Controller Register** **Init = 0**

Bit	R/W	Reset	Description
31: 1	RO	-	**Reserved(0)**
6	RW1S	RstARM	**Write Protection of this register SEC5C** This register is set by Secure Boot engine.
5: 0	RW	RstARM	**Secure Boot Engine Internal Controller Registers**

Offset: 060h **SEC60: Secure Boot Hardware Revision Register 1** **Init = 0**

Bit	R/W	Reset	Description
31: 0	RW	RstARM	**Secure Boot Hardware Revision Register 1** This is secure boot hardware revision register lower 32 bits. It will be same as OTPCFG10. This register is initialized by secure boot engine and its write protection will be enabled.

Offset: 064h **SEC64: Secure Boot Hardware Revision Register 2** **Init = 0**

Bit	R/W	Reset	Description
31: 0	RW	RstARM	**Secure Boot Hardware Revision Register 2** This is secure boot hardware revision register higher 32 bits. It will be same as OTPCFG11. This register is initialized by secure boot engine and its write protection will be enabled.

Offset: 068h **SEC68: Secure Boot Software Revision Register 1** **Init = 0**

Bit	R/W	Reset	Description
31: 0	RW	RstARM	**Secure Boot Software Revision Register 1** This is secure boot software revision register lower 32 bits. It will be same as secure boot header offset 0x10 (SBH10). This register is initialized by secure boot engine and its write protection will be enabled.

Offset: 06Ch	SEC6C: Secure Boot Software Revision Register 2	Init = 0

Bit	R/W	Reset	Description
31: 0	RW	RstARM	**Secure Boot Software Revision Register 2** This is secure boot software revision register higher 32 bits. It will be same as secure boot header offset 0x14 (SBH14). This register is initialized by secure boot engine and its write protection will be enabled.

Offset: 070h	SEC70: Secure Boot from SPI Status Register	Init = 0

Bit	R/W	Reset	Description
31: 3	RO	-	**Reserved(0)**
2	RW1S	RstARM	**Write Protection of this register SEC70** This register will be set by Secure Boot Engine to prevent update.
1: 0	RW	RstARM	**Secure Boot from SPI Status Registers** 00: Reset initial value and during secure boot period. The value will keep 00 when secure boot is not enabled or Boot from eMMC is enabled. 01: Secure Boot check failed. 10: Reserved 11: Secure Boot check passed.

Offset: 074h	SEC74: Secure Boot from eMMC Status Register	Init = 0

Bit	R/W	Reset	Description
31: 3	RO	-	**Reserved(0)**
2	RW1S	RstARM	**Write Protection of this register SEC74** This register will be set by Secure Boot Engine to prevent update.
1: 0	RW	RstARM	**Secure Boot from SPI Status Registers** 00: Reset initial value and during secure boot period. The value will keep 00 when secure boot is not enabled or Boot from eMMC is not enabled. 01: Secure Boot check failed. 10: Reserved 11: Secure Boot check passed.

Offset: 078h	SEC78: Secure Boot Key Number Register	Init = 0

Bit	R/W	Reset	Description
31: 9	RO	-	**Reserved(0)**
8	RW1S	RstARM	**Write Protection of this register SEC78** This register will be set by Secure Boot Engine to prevent update.
7: 3	RO	-	**Reserved(0)**
2: 0	RW	RstARM	**Secure Boot Key Number Registers** This register indicated the successful boot used Key # in OTP memory

Offset: 080h	SEC80: Secure Boot Engine Internal Controller Register	Init = 0

Bit	R/W	Reset	Description
31: 4	RO	-	**Reserved(0)**
3: 0	RW	RstARM	**Secure Boot Engine Internal Controller Registers**

Offset: 084h	SEC84: Secure Boot Engine Internal Controller Register	Init = 0

Bit	R/W	Reset	Description
31:17	RO	-	**Reserved(0)**
16	RW1S	RstARM	**Write Protection of this register SEC84** This register is set by Secure Boot engine.
15: 0	RW	RstARM	**Secure Boot Engine Internal Controller Registers**

Offset: 088h	SEC88: Software ECC Control Register	Init = 0

Bit	R/W	Reset	Description
31: 3	RO	-	**Reserved(0)**
2	RW1S	RstARM	**Write Protection of this register SEC88** This register is set by software.
1	RW	RstARM	**Software ECC Control for Secure Region Register** 0: normal mode 1: disable ECC for secure region This register is used for OTP memory failure analysis only.
0	RW	RstARM	**Software ECC Control for User Region Register** 0: normal mode 1: disable ECC for user region This register is used for OTP memory failure analysis only.

Offset: 090h	SEC90: Secure Boot Counter Register	Init = 0

Bit	R/W	Reset	Description
31:16	RO	-	**Reserved(0)**
15: 0	RO	RstFull	**Secure Boot Counter Register** This counter will increase when boot pass and fail. Its vaule is reset to 0 when SRSTN# asserted.

Offset: 094h	SEC94: Secure Boot Counter 2 Register	Init = 0

Bit	R/W	Reset	Description
31:16	RO	-	**Reserved(0)**
15: 0	RO	RstFull	**Secure Boot Counter 2 Register** This counter will increase when boot pass, fail. fail including boot from VUART. Its vaule is reset to 0 when SRSTN# asserted.

Offset: 098h	SEC98: Secure Boot Engine Internal Controller Register	Init = 0

Bit	R/W	Reset	Description
31: 8	RO	-	**Reserved(0)**
7: 0	RW	RstARM	**Secure Boot Engine Internal Controller Registers**

Offset: 0A0h	SECA0: eMMC Boot Watchdog Control Register	Init = 0

Bit	R/W	Reset	Description
31:24	WT	RstARM	**P_WDT_CLR** Write 0xEA: trigger P_WDT_CLR Write Others: No effect
23:16	WT	RstARM	**P_WDT_BOOT_CLR** Write 0xEA: trigger P_WDT_BOOT_CLR Write Others: No effect
15: 8	RO	RstARM	**wdt2_acc**
7: 5	RO	RstARM	**Reserved(0)**
4	RO	RstARM	**wdt2_boot2sel**
3: 1	RO	RstARM	**Reserved(0)**
0	RW	RstARM	**P_WDT_EN**

Offset: 0A4h	SECA4: eMMC Boot Watchdog Control Register	Init = 0

Bit	R/W	Reset	Description
31:16	RO	RstARM	**wdt2_cnt**
15:13	RO	RstARM	**Reserved(0)**
12: 0	RW	RstARM	**P_WDT_RELOAD_VAL**

Offset: 0A8h	SECA8: eMMC Boot Watchdog Control Register	Init = 0

Bit	R/W	Reset	Description
31:16	WT	RstARM	**P_WDT_TESTMODE** Write 0xFAFA: set P_WDT_TESTMODE Write Others: clear P_WDT_TESTMODE
15: 0	WT	RstARM	**P_WDT_RELOAD** Write 0x4755: trigger P_WDT_RELOAD Write Others: No effect

Offset: 0B0h	SECB0: Secure Boot Engine Internal Controller Register	Init = 0

Bit	R/W	Reset	Description
31:30	RO	-	**Reserved(0)**
29: 0	RW	RstARM	**Secure Boot Engine Internal Controller Registers**

Offset: 0B4h	SECB4: Secure Boot Engine Internal Controller Register	Init = 0

Bit	R/W	Reset	Description
31: 1	RO	-	**Reserved(0)**
0	RW	RstARM	**Secure Boot Engine Internal Controller Registers**

Offset: 800h	SEC800: Secure Boot DMA Enable Register	Init = 0

Bit	R/W	Reset	Description
31: 0	RO	-	**Reserved(0)**
0	W1T	RstARM	**Secure Boot DMA Enable Register** Write 1: trigger Secure Boot DMA The Secure Boot DMA is for SBMCU use only.

Offset: 808h	SEC808: Secure Boot DMA Mode Register		Init = 0
Bit	**R/W**	**Reset**	**Description**
31: 6	RO	-	**Reserved(0)**
5: 0	RW	RstARM	**Secure Boot DMA Mode Register** The Secure Boot DMA is for SBMCU use only.

Offset: 80Ch	SEC80C: Secure Boot Vault Key Control Register		Init = 0
Bit	**R/W**	**Reset**	**Description**
31: 3	RO	-	**Reserved(0)**
2	RW1S	RstARM	**Vault Key Selection Protection** 0: normal mode 1: Disable write capability of register SEC80C[0].
1	RW1S	RstARM	**Vault Key Write Protection** 0: normal mode 1: Disable write capability of registers SEC900 - SEC93C .
0	RW	RstARM	**Vault Key Selection** 0: Select first vault key for HACE 1: Select second valut key for HACE

Offset: 810h	SEC810: Secure Boot Digest Status		Init = 0
Bit	**R/W**	**Reset**	**Description**
31: 1	RO	-	**Reserved(0)**
0	RW	RstARM	**Secure Boot Digest Status** 0: Secure boot digest check fail when SEC814[0]=1 1: Secure boot digest check pass when SEC814[0]=1

Offset: 814h	SEC814: Secure Boot Digest Check Status		Init = 0
Bit	**R/W**	**Reset**	**Description**
31: 1	RO	-	**Reserved(0)**
0	RW	RstARM	**Secure Boot Digest Status** 0: Secure boot digest check idle 1: Secure boot digest check busy

Offset: 820h	SEC820: Secure Crypto Engine Enable Register		Init = 0
Bit	**R/W**	**Reset**	**Description**
31: 1	RO	-	**Reserved(0)**
0	RW	RstARM	**Secure Crypto Engine Enable Register** 0: Secure crypto engine disable 1: Secure crypto engine enable. HACE won't work when this mode.

Offset: 840h	SEC840: Secure Boot DMA Source Address Register		Init = x
Bit	**R/W**	**Reset**	**Description**
31: 2	RW	-	**Secure Boot DMA Source Address Register** Source starting address of secure boot DMA
1: 0	RO	-	**Reserved(0)**

Offset: 844h	SEC844: Secure Boot DMA Destination Address Register		Init = x
Bit	**R/W**	**Reset**	**Description**
31: 2	RW	-	**Secure Boot DMA Destination Address Register** Destination starting address of secure boot DMA
1: 0	RO	-	**Reserved(0)**

Offset: 848h	SEC848: Secure Boot DMA Size Register		Init = x
Bit	**R/W**	**Reset**	**Description**
31:17	RO	-	**Reserved(0)**
16: 0	RW	-	**Secure Boot DMA Size Register** Size of secure boot DMA

Offset: 854h	SEC854: Secure Boot Crypto AES-GCM AAD Size Register		Init = x
Bit	**R/W**	**Reset**	**Description**
31:17	RO	-	**Reserved(0)**
16: 4	RW	-	**Secure Boot Crypto AES-GCM AAD Size Register** Secure boot AES_GCM AAD size.
3: 0	RO	-	**Reserved(0)**

Offset: 858h	SEC858: Secure Boot Hash Mode Register		Init = x
Bit	**R/W**	**Reset**	**Description**
31: 8	RO	-	**Reserved(0)**
7: 0	RW	-	**Secure Boot Hash Mode Register** This register is controlled by SBMCU.

Offset: 85Ch	SEC85C: Secure Boot Hash Engine Fire Register		Init = 0
Bit	**R/W**	**Reset**	**Description**
31: 0	WT	-	**Secure Boot Hash Engine Fire Register** Write any value to this is register will trigger secure boot hash engine fire. This register is controlled by SBMCU.

Offset: 860h	SEC860: Secure Boot Crypto Mode Register		Init = 0
Bit	**R/W**	**Reset**	**Description**
31: 8	RO	-	**Reserved(0)**
7: 0	RW	RstARM	**Secure Boot Crypto Mode Register** This register is controlled by SBMCU.

Offset: 864h	SEC864: Secure Boot Crypto Data Size Register		Init = x
Bit	**R/W**	**Reset**	**Description**
31:17	RO	-	**Reserved(0)**
16: 4	RW	-	**Secure Boot Crypto Data Size Register** This register is controlled by SBMCU.
3: 0	RO	-	**Reserved(0)**

Offset: 868h	SEC864: Secure Boot Crypto Data Total Size Register		Init = x
Bit	**R/W**	**Reset**	**Description**
31:13	RO	-	**Reserved(0)**
12: 0	RW	-	**Secure Boot Crypto Data Total Size Register** This register is controlled by SBMCU.

Offset: 870h	SEC870: Secure Boot Crypto Low Key Write Trigger Register		Init = 0
Bit	**R/W**	**Reset**	**Description**
31: 0	WT	-	**Secure Boot Crypto Low Key Write Trigger Register** Write any value to this is register will trigger key[127:0] write. This register is controlled by SBMCU.

Offset: 874h	SEC874: Secure Boot Crypto High Key Write Trigger Register		Init = 0
Bit	**R/W**	**Reset**	**Description**
31: 0	WT	-	**Secure Boot Crypto High Key Write Trigger Register** Write any value to this is register will trigger key[255:128] write. This register is controlled by SBMCU.

Offset: 878h	SEC878: Secure Boot Crypto Vector Write Trigger Register		Init = 0
Bit	**R/W**	**Reset**	**Description**
31: 0	WT	-	**Secure Boot Crypto Vector Write Trigger Register** Write any value to this is register will trigger vector write. This register is controlled by SBMCU.

Offset: 87Ch	SEC87C: Secure Boot Crypto Engine Fire Register		Init = 0
Bit	**R/W**	**Reset**	**Description**
31: 0	WT	-	**Secure Boot Crypto Engine Fire Register** Write any value to this is register will trigger vector write. This register is controlled by SBMCU.

Offset: 880h	SEC880: Secure Boot Crypto Data Buffer 0 Register		Init = x
Bit	**R/W**	**Reset**	**Description**
31: 0	WO	-	**Secure Boot Crypto Data Buffer 0 Register** Data bit [31:0] of crypto data. This register is controlled by SBMCU.

Offset: 884h	SEC884: Secure Boot Crypto Data Buffer 1 Register		Init = x
Bit	**R/W**	**Reset**	**Description**
31: 0	WO	-	**Secure Boot Crypto Data Buffer 1 Register** Data bit [63:32] of crypto data. This register is controlled by SBMCU.

Offset: 888h	SEC888: Secure Boot Crypto Data Buffer 2 Register		Init = x
Bit	**R/W**	**Reset**	**Description**
31: 0	WO	-	**Secure Boot Crypto Data Buffer 2 Register** Data bit [95:64] of crypto data. This register is controlled by SBMCU.

Offset: 88Ch	SEC88C: Secure Boot Crypto Data Buffer 3 Register		Init = x
Bit	**R/W**	**Reset**	**Description**
31: 0	WO	-	**Secure Boot Crypto Data Buffer 3 Register** Data bit [127:96] of crypto data. This register is controlled by SBMCU.

Offset: 890h	SEC890: Secure Boot Crypto AES-GCM GHash Key Write Trigger Register		Init = 0
Bit	**R/W**	**Reset**	**Description**
31: 0	WT	-	**Secure Boot Crypto AES-GCM GHash Key Write Trigger Register** Write any value to this is register will trigger GHash write. This register is controlled by SBMCU.

Offset: 8A0h	SEC8A0: Secure Boot Crypto Key Buffer 0 Register		Init = x
Bit	**R/W**	**Reset**	**Description**
31: 0	WO	-	**Secure Boot Crypto Key Buffer 0 Register** Data bit [31:0] of crypto key or vector. This register is controlled by SBMCU.

Offset: 8A4h	SEC8A4: Secure Boot Crypto Key Buffer 1 Register		Init = x
Bit	**R/W**	**Reset**	**Description**
31: 0	WO	-	**Secure Boot Crypto Key Buffer 1 Register** Data bit [63:32] of crypto key or vector. This register is controlled by SBMCU.

Offset: 8A8h	SEC8A8: Secure Boot Crypto Key Buffer 2 Register		Init = x
Bit	**R/W**	**Reset**	**Description**
31: 0	WO	-	**Secure Boot Crypto Key Buffer 2 Register** Data bit [95:64] of crypto key or vector. This register is controlled by SBMCU.

Offset: 8ACh	SEC8AC: Secure Boot Crypto Key Buffer 3 Register		Init = x
Bit	**R/W**	**Reset**	**Description**
31: 0	WO	-	**Secure Boot Crypto Key Buffer 3 Register** Data bit [127:96] of crypto key or vector. This register is controlled by SBMCU.

Offset: 8B0h	SEC8B0: Secure Boot Crypto Key Buffer 4 Register		Init = x
Bit	**R/W**	**Reset**	**Description**
31: 0	WO	-	**Secure Boot Crypto Key Buffer 4 Register** Data bit [159:128] of crypto key . This register is controlled by SBMCU.

Offset: 8B4h	SEC8B4: Secure Boot Crypto Key Buffer 5 Register	Init = x

Bit	R/W	Reset	Description
31: 0	WO	-	**Secure Boot Crypto Key Buffer 5 Register** Data bit [191:160] of crypto key. This register is controlled by SBMCU.

Offset: 8B8h	SEC8B8: Secure Boot Crypto Key Buffer 6 Register	Init = x

Bit	R/W	Reset	Description
31: 0	WO	-	**Secure Boot Crypto Key Buffer 6 Register** Data bit [223:192] of crypto key. This register is controlled by SBMCU.

Offset: 8BCh	SEC8BC: Secure Boot Crypto Key Buffer 7 Register	Init = x

Bit	R/W	Reset	Description
31: 0	WO	-	**Secure Boot Crypto Key Buffer 7 Register** Data bit [255:224] of crypto key. This register is controlled by SBMCU.

SEC8C0 ~ SEC8FC: Secure Boot Image Digest Read Back #0 ~ #16

Offset: 8C0~8FFh	Init = x

Bit	Attr.	Reset	Description
31:0	RW	-	**Secure Boot Image Digest Read Back**

Offset: 900h	SEC900: Secure Boot First Vault Key 0 Register	Init = x

Bit	R/W	Reset	Description
31: 0	WO	-	**Secure Boot First Vault Key 0 Register** Data bit [31:0] of first vault key.

Offset: 904h	SEC904: Secure Boot First Vault Key 1 Register	Init = x

Bit	R/W	Reset	Description
31: 0	WO	-	**Secure Boot First Vault Key 1 Register** Data bit [63:32] of first vault key.

Offset: 908h	SEC908: Secure Boot First Vault Key 2 Register	Init = x

Bit	R/W	Reset	Description
31: 0	WO	-	**Secure Boot First Vault Key 2 Register** Data bit [95:64] of first vault key.

Offset: 90Ch	SEC90C: Secure Boot First Vault Key 3 Register	Init = x

Bit	R/W	Reset	Description
31: 0	WO	-	**Secure Boot First Vault Key 3 Register** Data bit [127:96] of first vault key.

Offset: 910h	SEC910: Secure Boot First Vault Key 4 Register		Init = x
Bit	**R/W**	**Reset**	**Description**
31: 0	WO	-	**Secure Boot First Vault Key 4 Register** Data bit [159:128] of first vault key .

Offset: 914h	SEC914: Secure Boot First Vault Key 5 Register		Init = x
Bit	**R/W**	**Reset**	**Description**
31: 0	WO	-	**Secure Boot First Vault Key 5 Register** Data bit [191:160] of first vault key.

Offset: 918h	SEC918: Secure Boot First Vault Key 6 Register		Init = x
Bit	**R/W**	**Reset**	**Description**
31: 0	WO	-	**Secure Boot First Vault Key 6 Register** Data bit [223:192] of first vault key.

Offset: 91Ch	SEC91C: Secure Boot First Vault Key 7 Register		Init = x
Bit	**R/W**	**Reset**	**Description**
31: 0	WO	-	**Secure Boot First Vault Key 7 Register** Data bit [255:224] of first vault key.

Offset: 920h	SEC920: Secure Boot Second Vault Key 0 Register		Init = x
Bit	**R/W**	**Reset**	**Description**
31: 0	WO	-	**Secure Boot Second Vault Key 0 Register** Data bit [31:0] of second vault key.

Offset: 924h	SEC924: Secure Boot Second Vault Key 1 Register		Init = x
Bit	**R/W**	**Reset**	**Description**
31: 0	WO	-	**Secure Boot Second Vault Key 1 Register** Data bit [63:32] of second vault key.

Offset: 928h	SEC928: Secure Boot Second Vault Key 2 Register		Init = x
Bit	**R/W**	**Reset**	**Description**
31: 0	WO	-	**Secure Boot Second Vault Key 2 Register** Data bit [95:64] of second vault key.

Offset: 92Ch	SEC92C: Secure Boot Second Vault Key 3 Register		Init = x
Bit	**R/W**	**Reset**	**Description**
31: 0	WO	-	**Secure Boot Second Vault Key 3 Register** Data bit [127:96] of second vault key.

Offset: 930h	SEC930: Secure Boot Second Vault Key 4 Register		Init = x
Bit	**R/W**	**Reset**	**Description**
31: 0	WO	-	**Secure Boot Second Vault Key 4 Register** Data bit [159:128] of second vault key .

Offset: 934h	SEC934: Secure Boot Second Vault Key 5 Register		Init = x
Bit	**R/W**	**Reset**	**Description**
31: 0	WO	-	**Secure Boot Second Vault Key 5 Register** Data bit [191:160] of second vault key.

Offset: 938h	SEC938: Secure Boot Second Vault Key 6 Register		Init = x
Bit	**R/W**	**Reset**	**Description**
31: 0	WO	-	**Secure Boot Second Vault Key 6 Register** Data bit [223:192] of second vault key.

Offset: 93Ch	SEC93C: Secure Boot Second Vault Key 7 Register		Init = x
Bit	**R/W**	**Reset**	**Description**
31: 0	WO	-	**Secure Boot Second Vault Key 7 Register** Data bit [255:224] of second vault key.

61.2 OTP Memory

Each memory bit cell inside the OTP memory is capable to be programmed once. Typically the data stored the OTP memory are non-volatile and can preserve permanently, but to improve the FIT (failure in time) of the OTP memory, ECC is recommended to enable.

61.2.1 OTP Memory Regions

The OTP memory comprises three physical regions. Each region has its own addressing space. The OTP memory regions are:

- Configuration region: Total 1k bits
- Data region: Total 64k bits
- Redundancy region: Total 2k bits to repair at most 4 defects

For the data format and usage of data region and redundancy region, please reference to AST2600 Secure Boot Specification for more detail.

61.2.2 OTP Configuration Register

Here are the descriptions of OTP configuration registers in configuration region.

OTP_ADDR: 800h	OTPCFG0: OTP Configuration Register #0	Init = 0

Bit	R/W	Description
31	OTP	**OTP memory lock enable** This bit will lock whole OTP memory and is un-recoverable.
30	OTP	**Disable Auto Boot from UART or VUART** 0: Boot from UART/VUART when normal boot is fail 1: Disable auto UART/VUART boot option When the integrity check failed for the images (both image when ABR is enabled), the secure boot engine will fetch code from VUART port. This is useful for updating firmware from HOST when boot failed.
29	OTP	**Write Protect of OTP key retire bits** Setting this bit enables write protection for OTPCFG4.
28	OTP	**Enable Flash Patch Code** 0: When the secure boot header SBH18 is not 0, the flash path codes will be interpreted. The flash patch codes reside in flash boot image and will be measured before execution. 1: Flash path code will not interpret.
27	OTP	**Enable image encryption** This along with OTPCFG0[26] being set enables image encryption to improve security
26	OTP	**Enable Copy Boot Image** Enables copying of image to internal SRAM instead of direct boot from Flash Device (Image size is limited to 63 KB)
25	OTP	**Write Protect of OTP strap region** Setting this bit enables Write protection for OTPCFG16-OTPCFG31.
24	OTP	**Write Protect of Configure region** Setting this bit enables Write protection for all configuration region and OTPCFG4, OTPCFG10, OTPCFG11 and OTPCFG16-OTPCFG31 are excluded.
23	OTP	**Write Protect of User region** Setting this bit enables Write protection for address greater than or equal to OTPCFG0[21:16]*32 in data region.
22	OTP	**Write Protect of Secure Region** Setting this bit enables Write protection for address little than OTPCFG0[21:16]*32 in data region.
21:16	OTP	**Secure Region size** Set secure region size. It is in unit of 32DW. The range of secure region is from address 0 to address OTPCFG0[21:16] * 32 - 1. If OTPCFG0[21:16]=0, the secure region will be disabled.
15	OTP	**Disable Boot from Uart** 0: When hardware strap pin FWSPICK is pulled high, the secure boot MCU will fetch image codes from UART5 and boot. The image code should be signed when secure boot is enabled. When hardware strap pin FWSPICK is pulled low, the secure boot MCU will not boot from UART. 1: The secure boot MCU will not boot from UART in spite of hardware strap pin.
14	OTP	**Disable Patch code** 0: If the OTPCFG14[16:0] is not 0, the secure boot MCU will execute the patch codes which has been programmed into OTP data region. 1: The secure boot MCU will not execute the patch codes in data region.
13:12	OTP	**Hash selection** 00: SHA224 01: SHA256 10: SHA384 11: SHA512

to next page

from previous page

11:10	OTP	**Secure crypto selection** 00: RSA1024 01: RSA2048 10: RSA3072 11: RSA4096
9	OTP	**Disable Uart Message of Secure Boot Engine** 0: Secure Boot Engine will dump boot messages. 1: Secure Boot Engine message is disabled. The list of messages: 'B' Boot from SPI 'M' Boot from eMMC 'U' Boot from UART 'A' Boot from Secret Vault Keys are found 'S' Secure Boot is enabled '2' Secure boot Mode 2 '3' Secure Boot Mode GCM 'E' Secure Boot image decrypted 'P' Secure Boot Pass 'F' Secure Boot Fail '0' Boot from first image when ABR mode enabled '1' Boot from second image when ABR mode enabled 'c'+'nn' Message codes for different message number 'nn' 'L' Flash Patch code was executed 'V' Boot from VUART
8	OTP	**Configuration region redundancy repair enable** The OTP configuration region memory defect can be repaired. It is only for OTPCFG2-OTPCFG14, OTPCFG16-OTPCFG31.
7	OTP	 0: Mode GCM 1: Mode2
6	OTP	**Disable Secure Boot Enable Strap Source** 0: Secure Boot can be enabled by hardware pin strap FWSPIMOSI or OTPSTRAP[0]. 1: Disable. Secure boot cannot be enabled by hardware pin strap FWSPIMOSI or OTPSTRAP[0]. This config bit it is to enable/disable the strap sources (FWSPIMOSI, OTPSTRAP[0]) for controlling the enable/disable of secure boot.
5	OTP	**Disable low security key #0** 0: OEM DSS key #0 can be enabled by hardware strap pin FWSPIMISO. 1: OEM DSS key #0 is disabled.
4	OTP	**Secure Region ECC enable bit** 0: ECC is disabled for secure region 1: ECC is enabled for secure region and the ECC codes must be programmed into ECC region.
3	OTP	**User region ECC enable** 0: ECC is disabled for user region 1: ECC is enabled for user region and the ECC codes must be programmed into ECC region.
2	OTP	**Initialization programming done bit** This is for software use only to indicate the OTP initialization programming is done. It is the last bit to program during OTP initialization programming.

to next page

from previous page

1	OTP	**Secure Boot Enable** 0: Secure Boot Disabled. When this bit is set to 0 and OTPCFG0[6]=0 secure boot can controlled by FWSPIMOSI/OTPSTRAP[0] 1: Secure Boot Enabled. When this bit is set to 1 the secure boot engine is permanently enabled. Other secure boot enable/disable bits do not have any effect when this bit is set to 1.
0	OTP	**Force to Disable OTP Factory Test Mode** 0: OTP Factory Test Mode is Allowed. 1: Force to Disable OTP Factory Test Mode. Suggest to set this bit to disable OTP memory facory test mode to prevent security issue.

OTP_ADDR: 802h **OTPCFG1: OTP Configuration Register #1** **Init = 0**

Bit	R/W	Description
31:0	OTP	**Backup Register of OTPCFG0** The backup registers will do bitwise OR operation with primary registers per bit. It works as redundancy bits to recover primary bit defect. When there is any bit programming failure in OTPCFG0, the corresponding bit in OTPCFG1 can be programmed to recover. For the purpose of reliability, program same value to back up register is proposed.

OTP_ADDR: 804h **OTPCFG2: OTP Configuration Register #2** **Init = 0**

Bit	R/W	Description
31:16	OTP	**Key Revision** This is for software use only.
15:0	OTP	**Vender ID** This is for software use only.

OTP_ADDR: 806h **OTPCFG3: OTP Configuration Register #3** **Init = 0**

Bit	R/W	Description
31	OTP	**Enable CDI Generation** 0: No CDI generated. 1: Enable to generate CDI (Compound Device Identifier) as DICE (Device Indentifier Composition Engie) KEY = Unique Key 1 is 256-bit key HASH = Image hash is 256-bit IV = 64-bit unique ID + 64-bit OTP Data (OTPCFG6, OTPCFG5) DICE will encrypt HASH by using KEY and IV with AES256 engine. After secure boot, 256-bit CDI(M1) can be read from 0x1e711220.
30	OTP	**Erase signature data after secure boot check** 0: Boot image signature will be kept after secure boot. 1: Boot image signature will be erased after secure boot
29:24	OTP	**Extra Programming Protection Region Size** These OTP bits can set extra programming protection region after secure region. Using this option can allow extra programming protection region after secure region with read protection capability. The unit is same as secure region size. The programming protection region will be secure region (OTPCFG0[21:16]) plus this extra region.
23:21	OTP	**Rollback Prevention Shift Bit Number** Set rollback prevention shift bit number and allow manifest ID to be shifted right with predefined bits before comparison.

to next page

from previous page

20	OTP	**Disable ROM Code Based Programming Control** The OTP configuration region protection is complex and changed to ROM code based protection in A2. This bit will revert the protection back to A1 compatible protection. Practically, the protection of A1 and A2 are almost the same. There is no difference between them except new features, ex. rollback prevention shift bit number.
19	OTP	**Disable Auto Boot from VUART2 over LPC** 0: enable 1: disable auto boot from VUART2 over LPC
18	OTP	**Disable Auto Boot from VUART2 over PCIe** 0: enable 1: disable auto boot from VUART2 over PCIe
17	OTP	**Disable Auto Boot from UART** 0: enable 1: disable auto boot from UART1 or UART5
16	OTP	**Boot From UART Port Selection** 0: UART5 1: UART1
15:0	OTP	**Secure Boot Header Offset** This OTP can re-assigned the Secure Boot Header location in image. The default location of Secure Boot Header is at 0x20 of image when OTPCFG3=0.

OTP_ADDR: 808h **OTPCFG4: OTP Configuration Register #4** **Init = 0**

Bit	R/W	Description
31:24	OTP	**Reserved (0)**
23:16	OTP	**Reserved (0)**
15:8	OTP	**Reserved (0)**
7:0	OTP	**Key Retire bits** Program the selected bit in this register will de-activate the corresponding number of key. This operation is not recoverable. The programming of these register bits are protected by secure boot status in default. When security boot successfully by using key number N, the retire bit number less then N will be programmable and the others are protected. When non-security boot up, all of the retire bits are protected. Programming OTPCFG0[29] to 1 can protect this whole register.

OTP_ADDR: 80Ah **OTPCFG5: OTP Configuration Register #5** **Init = 0**

Bit	R/W	Description
31:0	OTP	**User defined data** Example: random number[31:0]

OTP_ADDR: 80Ch **OTPCFG6: OTP Configuration Register #6** **Init = 0**

Bit	R/W	Description
31:0	OTP	**User defined data** Example: random number[63:32]

OTP_ADDR: 80Eh	OTPCFG7: OTP Configuration Register #7	Init = 0

Bit	R/W	Description
31	OTP	**Enable chip security setting** 0: disable 1: enable to auto load registers SCU0C8 and SCU0D8.
30:16	OTP	**SCU0D8[14:0] auto setting value** After SCU0D8 is loaded, the write protection SCUF10[26] will be set to 1 and SCU0D8 is not possible to modify.
15	OTP	**Reserved (0)**
14:0	OTP	**SCU0C8[14:0] auto setting value** After SCU0C8 is loaded, the write protection SCUF00[26] will be set to 1 and SCU0C8 is not possible to modify.

OTP_ADDR: A00h	OTPCFG8: OTP Configuration Register #8	Init = 0

Bit	R/W	Description
31:24	OTP	**Redundancy Repair of Sector 3 (RR3)**
23:16	OTP	**Redundancy Repair of Sector 3 (RR2)**
15:8	OTP	**Redundancy Repair of Sector 3 (RR1)**
7:0	OTP	**Redundancy Repair of Sector 3 (RR0)**

Note :
Note: reference to AST2600 Secure Boot Specification for more detail.

bit	Redundancy Repair (RR)
7	Repaired row address bit 8
6	Repaired row address bit 7
5	Repaired row address bit 6
4	Repaired row address bit 5
3	Repair Enable
2	Repaired row address bit 4
1	Repaired row address bit 3
0	Repaired row address bit 2

OTP_ADDR: A02h	OTPCFG9: OTP Configuration Register #9	Init = 0

Bit	R/W	Description
31:0	OTP	**Backup Register of OTPCFG8** The backup registers will do bitwise OR operation with OTPCFG8 per bit. It works as redundancy bits to recover primary bit defect. When there is any bit programming failure in OTPCFG8, the corresponding bit in OTPCFG9 can be programmed to recover.

OTP_ADDR: A04h	OTPCFG10: OTP Configuration Register #10	Init = 0

Bit	R/W	Description
31:0	OTP	**Manifest ID[31:0] (prevent rollback)** Header revision ID[63:0] in secure boot header must be equal to or greater than Manifest ID[63:0] (OTPCFG10 and OTPCFG11) for rollback prevention.

OTP_ADDR: A06h	OTPCFG11: OTP Configuration Register #11	Init = 0

Bit	R/W	Description
31:0	OTP	**Manifest ID[63:32] (prevent rollback)** Please reference OTPCFG10

OTP_ADDR: A08h	OTPCFG12: OTP Configuration Register #12	Init = 0

Bit	R/W	Description
31:0	OTP	**Internal bus timing setting** Keep default 0 otherwise may cause system unstable.

OTP_ADDR: A0Ah	OTPCFG13: OTP Configuration Register #13	Init = 0

Bit	R/W	Description
31:0	OTP	**Internal bus timing setting 2** Keep default 0 otherwise may cause system unstable.

OTP_ADDR: A0Ch	OTPCFG14: OTP Configuration Register #14	Init = 0

Bit	R/W	Description
31:27	OTP	**Configuration Repair Address Backup** The backup registers will do bitwise OR operation with OTPCFG14[26:22] per bit. It works as redundancy bits to recover primary bit defect. When there is any bit programming failure in OTPCFG14[26:22], the corresponding bit in OTPCFG14[31:27] can be programmed to recover.
26:22	OTP	**Configuration Repair Address** When there are defect bits in single DW of configuration region, it can be repaired by using Redundancy Repair Data (OTPCFG15). The number n of OTPCFGn memory should be programmed in to OTPCFG14[26:22].
21:17	OTP	**Reserved (0)**
16:11	OTP	**Patch code size** The secure boot MCU can execute patch codes in OTP data region. The patch code size is limited to 64 DWs.
10:0	OTP	**Patch code location** The patch code location in OTP data region.

OTP_ADDR: A0Eh	OTPCFG15: OTP Configuration Register #15	Init = 0

Bit	R/W	Description
31:0	OTP	**Redundancy Repair Data** Data for Configuration Region repair. The hardware can repair only one DW error in configuration region.

OTP_ADDR: C00h	OTPCFG16: OTP Configuration Register #16	Init = 0

Bit	R/W	Description
31:0	OTP	**OTP Strap Option 1 of OTPSTRAP[31:0]** OTPATRAP[31: 0] = OTPCFG16 ^ OTPCFG18 ^ OTPCFG20 ^ OTPCFG22 ^ OTPCFG24 ^ OTPCFG26 ^ is bitwise exclusive-OR (XOR) operation Multiple XORed OTP strap option bits allows user to update the strap values up to maximum 5 times. If further update for the OTP strap is not allowed, set the selected bits OTP protect register in OTPCFG30 and OTPCFG31 can disable the OTP programming capability for selected OTP straps. Please reference 2.3 for detail strap definition.

OTP_ADDR: C02h	OTPCFG17: OTP Configuration Register #17	Init = 0

Bit	R/W	Description
31:0	OTP	**OTP Strap Option 1 of OTPSTRAP[63:32]** OTPATRAP[63:32] = OTPCFG17 ^ OTPCFG19 ^ OTPCFG21 ^ OTPCFG23 ^ OTPCFG25 ^ OTPCFG27 ^ is bitwise exclusive-OR (XOR) operation Multiple XORed OTP strap option bits allows user to update the strap values up to maximum 5 times. If further update for the OTP strap is not allowed, set the selected bits OTP protect register in OTPCFG30 and OTPCFG31 can disable the OTP programming capability for selected OTP straps. Please reference 2.3 for detail strap definition.

OTP_ADDR: C04h	OTPCFG18: OTP Configuration Register #18	Init = 0

Bit	R/W	Description
31:0	OTP	**OTP Strap Option 2 of OTPSTRAP[31:0]**

OTP_ADDR: C06h	OTPCFG19: OTP Configuration Register #19	Init = 0

Bit	R/W	Description
31:0	OTP	**OTP Strap Option 2 of OTPSTRAP[63:32]**

OTP_ADDR: C08h	OTPCFG20: OTP Configuration Register #20	Init = 0

Bit	R/W	Description
31:0	OTP	**OTP Strap Option 3 of OTPSTRAP[31:0]**

OTP_ADDR: C0Ah	OTPCFG21: OTP Configuration Register #21	Init = 0

Bit	R/W	Description
31:0	OTP	**OTP Strap Option 3 of OTPSTRAP[63:32]**

OTP_ADDR: C0Ch	OTPCFG22: OTP Configuration Register #22	Init = 0

Bit	R/W	Description
31:0	OTP	**OTP Strap Option 4 of OTPSTRAP[31:0]**

OTP_ADDR: C0Eh	OTPCFG23: OTP Configuration Register #23	Init = 0
Bit	**R/W**	**Description**
31:0	OTP	**OTP Strap Option 4 of OTPSTRAP[63:32]**

OTP_ADDR: E00h	OTPCFG24: OTP Configuration Register #24	Init = 0
Bit	**R/W**	**Description**
31:0	OTP	**OTP Strap Option 5 of OTPSTRAP[31:0]**

OTP_ADDR: E02h	OTPCFG25: OTP Configuration Register #25	Init = 0
Bit	**R/W**	**Description**
31:0	OTP	**OTP Strap Option 5 of OTPSTRAP[63:32]**

OTP_ADDR: E04h	OTPCFG26: OTP Configuration Register #26	Init = 0
Bit	**R/W**	**Description**
31:0	OTP	**OTP Strap Option 6 of OTPSTRAP[31:0]**

OTP_ADDR: E06h	OTPCFG27: OTP Configuration Register #27	Init = 0
Bit	**R/W**	**Description**
31:0	OTP	**OTP Strap Option 6 of OTPSTRAP[63:32]**

OTP_ADDR: E08h	OTPCFG28: OTP Configuration Register #28	Init = 0
Bit	**R/W**	**Description**
31:0	OTP	**OTP Strap Register Write Protect 1** The value of OTPCFG28 will auto load to SCU508.

OTP_ADDR: E0Ah	OTPCFG29: OTP Configuration Register #29	Init = 0
Bit	**R/W**	**Description**
31:0	OTP	**OTP Strap Register Write Protect 2** The value of OTPCFG28 will auto load to SCU518.

OTP_ADDR: E0Ch	OTPCFG30: OTP Configuration Register #30	Init = 0
Bit	**R/W**	**Description**
31:0	OTP	**OTP Strap Protect 1** Value 1 in OTPCFG30 will protect the programming for corresponding bits in OTPCFG16, OTPCFG18, OTPCFG20, OTPCFG22, OTPCFG24 and OTPCFG26.

OTP_ADDR: E0Eh	OTPCFG31: OTP Configuration Register #31	Init = 0
Bit	**R/W**	**Description**
31:0	OTP	**OTP Strap Protect 2** Value 1 in OTPCFG31 will protect the programming for corresponding bits in OTPCFG17, OTPCFG19, OTPCFG21, OTPCFG23, OTPCFG25 and OTPCFG27.

62 FSI Controller (FSI)

62.1 Overview

FSI Controller implements 2 set of OPB/FSI registers with 2 FSI master physical ports and one DMA engine. Each FSI Master function also include one (FSIC00)FSI Control registers located in OPB0 bus with base address defined in FSIM08 with 4K space. The physical address of these registers can be derived as the following:

Base address of OPB/FSI0= 0x1E79_B000
Base address of OPB/FSI1= 0x1E79_B100
Physical address = (Base address) + Offset

FSIM00: Version Register
FSIM04: Engine Trigger Register
FSIM08: FSI Control Offset Address
FSIM0C: FSI DMA Offset Address
FSIM10: OPB0 Bus Select Register
FSIM14: OPB0 Bus Read/Write Register
FSIM18: OPB0 Bus Transfer Size Register
FSIM1C: OPB0 Bus Address Register
FSIM20: OPB0 Bus Write Data Register
FSIM24: OPB1 DMA Enable Register
FSIM28: OPB1 Bus Select Register
FSIM2C: OPB1 Bus Read/Write Register
FSIM30: OPB1 Bus Transfer Size Register
FSIM34: OPB1 Bus Address Register
FSIM38: OPB1 Bus Write Data Register
FSIM3C: Clock Source Gate Register
FSIM40: FSIM All Interrupts Clear Register
FSIM44: FSIM Interrupts Mask Register
FSIM48: FSIM Interrupts Status/Clear Register
FSIM4C: OPB0 Write Data Byte Order1 Select
FSIM50: OPB0 Write Data Byte Order2 Select
FSIM54: OPB1 Write Data Byte Order1 Select
FSIM58: OPB1 Write Data Byte Order2 Select
FSIM5C: OPB0 Read Data Byte Order Select
FSIM60: OPB1 Read Data Byte Order Select
FSIM64: OPB Bus Retry Counter Number
FSIM80: OPB0 Bus Status Register
FSIM84: OPB0 Read Data Register
FSIM88: OPB1 DMA Status Register
FSIM8C: OPB1 Bus Status Register
FSIM90: OPB1 Read Data Register
FSIMC0: OPB1 DMA Function Debug Register
FSIMC4: DMA Channel0 Data Base Address Register
FSIMC8: DMA Channel0 Control Register
FSIMCC: DMA Channel1 Data Base Address Register
FSIMD0: DMA Channel1 Control Register
FSIMD4: DMA Channel2 Data Base Address Register
FSIMD8: DMA Channel2 Control Register
FSIMDC: DMA Channel3 Data Base Address Register
FSIME0: DMA Channel3 Control Register
FSIME4: DMA Function Enable Register
FSIME8: FSI Control Register

Base address of FSIC00= FSIM08[31:13]
Physical address = (Base address) + Offset
FSIC00: Mode Register

62.2 Features

- Complies with On-Chip Peripheral Bus specification Rev. 2.1
- Complies with OpenFSI specification Rev. 1.0.0
- Supports 2 FSI master ports.
- Supports one DMA engine.
- Implements 2 sets of OPB/FSI registers.

62.3 Procedure to enable FSI ports

1. Set SCU210 bit[10:0] = 4C2, change FSI clock to 166MHz
2. Set SCU050 bit[27] = 1, enable FSI master module reset
3. Set SCU094 bit[30] = 1, enable FSI master module clock
4. wait 1ms for PLL locking
5. Set SCU054 bit[27] = 1, disable FSI master module reset
6. Set SCU4D8 bit[23:20] = ”1111”, enable FSI port0 and port1
7. Set SCU638 bit[19:16] = ”1010”, enable FSI port0 and port1 DATA pull down resistor
8. Set SCU438 bit[23:20] = ”0000”, route FSI port0 and port1 to IO ports

62.4 FSI Registers : Base Address = 0x1E79:B000 or 0x1E79:B100

62.4.1 FSI OPB Interface Register

Offset: 00	FSIM00: Version Register	Init = 0x000000A1
Bit	**R/W**	**Description**
7:0	RO	**FSI Master version**

Offset: 04	FSIM04: Engine Trigger Register	Init = 0x00000000
Bit	**R/W**	**Description**
31:1	RO	**Reserved**
0	RW	**FSIM Engine Trigger** 0 : diable 1 : enable

Offset: 08	FSIM08: FSI Control Offset Address	Init = 0x00000000
Bit	**R/W**	**Description**
31:13	RW	**FSI Control Offset Address, base address of FSIC00 registers**
12:0	RO	**Reserved**

Offset: 0C	FSIM0C: FSI DMA Offset Address	Init = 0x00000000
Bit	**R/W**	**Description**
31:22	RW	**FSI DMA Offset Address**
21:0	RO	**Reserved**

Offset: 10	FSIM10: OPB0 Bus Select Register	Init = 0x00000000
Bit	**R/W**	**Description**
31:1	RO	**Reserved**
0	RW	**OPB0 Bus select to enable bus activity** 0 : disable 1 : enable

Offset: 14	FSIM14: OPB0 Read/Write Register	Init = 0x00000000
Bit	**R/W**	**Description**
31:1	RO	**Reserved**
0	RW	**APB/OPB0 master access direction** 0 : write 1 : read

Offset: 18	FSIM18: OPB0 Bus Transfer Size Register	Init = 0x00000000
Bit	**R/W**	**Description**
31:2	RO	**Reserved**
1:0	RW	**OPB0 bus Transfer Size** 00 : byte 01 : half-word 11 : word

Offset: 1C	FSIM1C: OPB0 Bus Address Register	Init = 0x00000000
Bit	**R/W**	**Description**
31:0	RW	**OPB0 bus access address**

Offset: 20	FSIM20: OPB0 Bus Write Data Register	Init = 0x00000000
Bit	**R/W**	**Description**
31:0	RW	**OPB0 bus write data to OPB0 slave**

Offset: 24	FSIM24: OPB1 DMA Enable Register	Init = 0x00000000
Bit	**R/W**	**Description**
31:4	RO	**Reserved**
3	RW	**APB2OPB1 DMA channel 3**
2	RW	**APB2OPB1 DMA channel 2**
1	RW	**APB2OPB1 DMA channel 1**
0	RW	**APB2OPB1 DMA channel 0**

to next page

from previous page

Note :
Control DMA channel(n) dma function,
0 : disable
1 : enable

Offset: 28	FSIM28: OPB1 Bus Select Register	Init = 0x00000000
Bit	**R/W**	**Description**
31:1	RO	**Reserved**
0	RW	**OPB1 Bus select to enable bus activity** 0 : disable 1 : enable

Offset: 2C	FSIM2C: OPB1 Read/Write Register	Init = 0x00000000
Bit	**R/W**	**Description**
31:1	RO	**Reserved**
0	RW	**APB/OPB1 master access direction** 0 : write 1 : read

Offset: 30	FSIM30: OPB1 Bus Transfer Size Register	Init = 0x00000000
Bit	**R/W**	**Description**
31:2	RO	**Reserved**
1:0	RW	**OPB1 bus Transfer Size** 00 : byte 01 : half-word 11 : word

Offset: 34	FSIM34: OPB1 Bus Address Register	Init = 0x00000000
Bit	**R/W**	**Description**
31:0	RW	**OPB1 bus access address**

Offset: 38	FSIM38: OPB1 Bus Write Data Register	Init = 0x00000000
Bit	**R/W**	**Description**
31:0	RW	**OPB1 bus write data to OPB1 slave**

Offset: 3C	FSIM3C: Clock Source Gate Register	Init = 0x00000000
Bit	**R/W**	**Description**
31:1	RO	**Reserved**
0	RW	**Clock source gated** 0 : disable 1 : enable

Offset: 40	FSIM40: FSIM All Interrupt Clear	Init = 0x00000000

Bit	R/W	Description
31:1	RO	**Reserved**
0	RW	**Wrote 1 to clear all FSIM interrupt event, the bit go to 0 automatically**

Offset: 44	FSIM44: FSIM Interrupts Mask Register	Init = 0x00000000

Bit	R/W	Description
31:29		**Reserved**
28	RW	**Any FSI mater error interrupt mask**
27	RW	**Any FSI port error interrupt mask**
26	RW	**Any FSI hot plug interrupt mask**
25:18	RW	**Any remote sly interrupt mask[7:0]**
17	RW	**OPB1 transfer ACK interrupt mask**
16	RW	**OPB0 transfer ACK interrupt mask**
15:12	RW	**DMA channel TCONT done interrupt mask[3:0]**
11:8	RW	**DMA channel FIFO empty interrupt mask[3:0]**
7:4	RW	**DMA channel FIFO full interrupt mask[3:0]**
3:0	RW	**DMA channel EOT interrupt mask[3:0]**

Note :
Mask the corresponding interrupt event,
0 : disable
1 : enable

Offset: 48	FSIM48: FSIM Interrupts Status/Clear Register	Init = 0x00000000

Bit	R/W	Description
31:29	RO	**Reserved**
28	RW	**Any FSI mater error interrupt**
27	RW	**Any FSI port error interrupt**
26	RW	**Any FSI hot plug interrupt**
25:18	RW	**Any remote sly interrupt[7:0]**
17	RW	**OPB1 transfer ACK interrupt**
16	RW	**OPB0 transfer ACK interrupt**
15:12	RW	**DMA channel TCONT done interrupt[3:0]**
11:8	RW	**DMA channel FIFO empty interrupt[3:0]**
7:4	RW	**DMA channel FIFO full interrupt[3:0]**
3:0	RW	**DMA channel EOT interrupt[3:0]**

Note :
Read the register to get individual interrupt status, write the register to clear the corresponding interrput event.
0 : disable clear function
1 : enable clear function

Offset: 4C	FSIM4C: OPB0 Write Data Byte Order1 Select	Init = 0x0044EEE4

Bit	R/W	Description
31:24	RO	**Reserved**
23:22	RW	**opb0_hw10_selwr_b3**
21:20	RW	**opb0_hw10_selwr_b2**
19:18	RW	**opb0_hw10_selwr_b1**
17:16	RW	**opb0_hw10_selwr_b0**
15:14	RW	**opb0_hw00_selwr_b3**
13:12	RW	**opb0_hw00_selwr_b2**
11:10	RW	**opb0_hw00_selwr_b1**
9:8	RW	**opb0_hw00_selwr_b0**
7:6	RW	**opb0_fw_selwr_b3**
5:4	RW	**opb0_fw_selwr_b2**
3:2	RW	**opb0_fw_selwr_b1**
1:0	RW	**opb0_fw_selwr_b0**

Offset: 50	FSIM50: OPB0 Write Data Byte Order2 Select	Init = 0x0055AAFF

Bit	R/W	Description
31:30	RW	**opb0_by11_selwr_b3**
29:28	RW	**opb0_by11_selwr_b2**
27:26	RW	**opb0_by11_selwr_b1**
25:24	RW	**opb0_by11_selwr_b0**
23:22	RW	**opb0_by10_selwr_b3**
21:20	RW	**opb0_by10_selwr_b2**
19:18	RW	**opb0_by10_selwr_b1**
17:16	RW	**opb0_by10_selwr_b0**
15:14	RW	**opb0_by01_selwr_b3**
13:12	RW	**opb0_by01_selwr_b2**
11:10	RW	**opb0_by01_selwr_b1**
9:8	RW	**opb0_by01_selwr_b0**
7:6	RW	**opb0_by00_selwr_b3**
5:4	RW	**opb0_by00_selwr_b2**
3:2	RW	**opb0_by00_selwr_b1**
1:0	RW	**opb0_by00_selwr_b0**

Offset: 54	FSIM54: OPB1 Write Data Byte Order1 Select	Init = 0x00117717

Bit	R/W	Description
31:24	RO	**Reserved**
23:22	RW	**opb1_hw10_selwr_b3**
21:20	RW	**opb1_hw10_selwr_b2**
19:18	RW	**opb1_hw10_selwr_b1**
17:16	RW	**opb1_hw10_selwr_b0**
15:14	RW	**opb1_hw00_selwr_b3**

to next page

from previous page

13:12	RW	**opb1_hw00_selwr_b2**
11:10	RW	**opb1_hw00_selwr_b1**
9:8	RW	**opb1_hw00_selwr_b0**
7:6	RW	**opb1_fw_selwr_b3**
5:4	RW	**opb1_fw_selwr_b2**
3:2	RW	**opb1_fw_selwr_b1**
1:0	RW	**opb1_fw_selwr_b0**

Offset: 58 **FSIM58: OPB1 Write Data Byte Order2 Select** **Init = 0xFFAA5500**

Bit	R/W	Description
31:30	RW	**opb1_by11_selwr_b3**
29:28	RW	**opb1_by11_selwr_b2**
27:26	RW	**opb1_by11_selwr_b1**
25:24	RW	**opb1_by11_selwr_b0**
23:22	RW	**opb1_by10_selwr_b3**
21:20	RW	**opb1_by10_selwr_b2**
19:18	RW	**opb1_by10_selwr_b1**
17:16	RW	**opb1_by10_selwr_b0**
15:14	RW	**opb1_by01_selwr_b3**
13:12	RW	**opb1_by01_selwr_b2**
11:10	RW	**opb1_by01_selwr_b1**
9:8	RW	**opb1_by01_selwr_b0**
7:6	RW	**opb1_by00_selwr_b3**
5:4	RW	**opb1_by00_selwr_b2**
3:2	RW	**opb1_by00_selwr_b1**
1:0	RW	**opb1_by00_selwr_b0**

Offset: 5C **FSIM5C: OPB0 Read Data Byte Order Select** **Init = 0x0044EEE4**

Bit	R/W	Description
31:24	RO	**Reserved**
23:22	RW	**opb0_by_selrd_b3**
21:20	RW	**opb0_by_selrd_b2**
19:18	RW	**opb0_by_selrd_b1**
17:16	RW	**opb0_by_selrd_b0**
15:14	RW	**opb0_hw_selrd_b3**
13:12	RW	**opb0_hw_selrd_b2**
11:10	RW	**opb0_hw_selrd_b1**
9:8	RW	**opb0_hw_selrd_b0**
7:6	RW	**opb0_fw_selrd_b3**
5:4	RW	**opb0_fw_selrd_b2**
3:2	RW	**opb0_fw_selrd_b1**
1:0	RW	**opb0_fw_selrd_b0**

Offset: 60	FSIM60: OPB1 Read Data Byte Order Select	Init = 0x00117717
Bit	**R/W**	**Description**
31:24	RO	**Reserved**
23:22	RW	**opb1_by_selrd_b3**
21:20	RW	**opb1_by_selrd_b2**
19:18	RW	**opb1_by_selrd_b1**
17:16	RW	**opb1_by_selrd_b0**
15:14	RW	**opb1_hw_selrd_b3**
13:12	RW	**opb1_hw_selrd_b2**
11:10	RW	**opb1_hw_selrd_b1**
9:8	RW	**opb1_hw_selrd_b0**
7:6	RW	**opb1_fw_selrd_b3**
5:4	RW	**opb1_fw_selrd_b2**
3:2	RW	**opb1_fw_selrd_b1**
1:0	RW	**opb1_fw_selrd_b0**

Offset: 64	FSIM64: OPB Bus Retry Counter Number	Init = 0x00000000
Bit	**R/W**	**Description**
31:16	RO	**Reserved**
15:0	RW	**Retry counter number**

Offset: 80	FSIM80: OPB0 Bus Status Register	Init = 0x00000000
Bit	**R/W**	**Description**
31:5	RO	**Reserved**
4	RO	**apb_opb0_timeout**
3	RO	**apb_opb0_retry**
2	RO	**apb_opb0_err_ack**
1	RO	**apb_opb0_fw_ack**
0	RO	**apb_opb0_hw_ack**

Offset: 84	FSIM84: OPB0 Read Data Register	Init = 0x00000000
Bit	**R/W**	**Description**
31:0	RO	**APB2OPB0 read data from OPB0 slave device**

Offset: 88	FSIM88: OPB1 DMA Status Register	Init = 0x00000000
Bit	**R/W**	**Description**
31:16	RO	**Reserved**
15:12	RO	**apb_opb1_dma_eot[0:3]**
11:8	RO	**apb_opb1_dma_req[0:3]**
7:4	RO	**opb1_dma_eot[0:3]**
3:0	RO	**opb1_dma_req[0:3]**

Offset: 8C	FSIM8C: OPB1 Bus Status Register	Init = 0x00000000
Bit	**R/W**	**Description**
31:5	RO	**Reserved**
4	RO	**apb_opb1_timeout**
3	RO	**apb_opb1_retry**
2	RO	**apb_opb1_err_ack**
1	RO	**apb_opb1_fw_ack**
0	RO	**apb_opb1_hw_ack**

Offset: 90	FSIM90: OPB1 Read Data Register	Init = 0x00000000
Bit	**R/W**	**Description**
31:0	RO	**APB2OPB1 read data from OPB1 slave device**

Offset: C0	FSIMC0: OPB1 DMA Function Debug Register	Init = 0x00000000
Bit	**R/W**	**Description**
31:8	RO	**Reserved**
7	RW	**DMA3 opb read request** (for debugging purpose only)
6	RW	**DMA2 opb read request** (for debugging purpose only)
5	RW	**DMA1 opb read request** (for debugging purpose only)
4	RW	**DMA0 opb read request** (for debugging purpose only)
3	RW	**opb_err will disable or not DMA3 read to FIFO** (for debugging purpose only)
2	RW	**opb_err will disable or not DMA2 read to FIFO** (for debugging purpose only)
1	RW	**opb_err will disable or not DMA1 read to FIFO** (for debugging purpose only)
0	RW	**opb_err will disable or not DMA0 read to FIFO** (for debugging purpose only)
Note : bit[7:4]: It is similiar with FSIM88[3:0], 1: enable, 0: no effect. (for debugging purpose only) bit[3:0]: 1: disable DMAx read function, 0: no effect. (for debugging purpose only)		

Offset: C4	FSIMC4: DMA Channel0 Data Base Address Register	Init = 0x00000000
Bit	**R/W**	**Description**
31	RO	**Reserved**
30:2	RW	**DBASE address, 4-bytes aligned.**
1:0	RO	**Reserved**

Offset: C8	FSIMC8: DMA Channel0 Control Register	Init = 0x00000000
Bit	**R/W**	**Description**
31:17	RO	**Reserved**
16	RW	**FSI access direction.** **0: FSI write, DMA to FSI.** **1: FSI read, FSI to DMA.**
15:0	RW	**TCOUNT, transfer counter in double words.**

Offset: CC	FSIMCC: DMA Channel1 Data Base Address Register	Init = 0x00000000
Bit	**R/W**	**Description**
31	RO	**Reserved**
30:2	RW	**DBASE address, 4-bytes aligned.**
1:0	RO	**Reserved**

Offset: D0	FSIMD0: DMA Channel1 Control Register	Init = 0x00000000
Bit	**R/W**	**Description**
31:17	RO	**Reserved**
16	RW	**FSI access direction.** **0: FSI write, DMA to FSI.** **1: FSI read, FSI to DMA.**
15:0	RW	**TCOUNT, transfer counter in double words.**

Offset: D4	FSIMD4: DMA Channel2 Data Base Address Register	Init = 0x00000000
Bit	**R/W**	**Description**
31	RO	**Reserved**
30:2	RW	**DBASE address, 4-bytes aligned.**
1:0	RO	**Reserved**

Offset: D8	FSIMD8: DMA Channel2 Control Register	Init = 0x00000000
Bit	**R/W**	**Description**
31:17	RO	**Reserved**
16	RW	**FSI access direction.** **0: FSI write, DMA to FSI.** **1: FSI read, FSI to DMA.**
15:0	RW	**TCOUNT, transfer counter in double words.**

Offset: DC	FSIMDC: DMA Channel3 Data Base Address Register	Init = 0x00000000
Bit	**R/W**	**Description**
31	RO	**Reserved**
30:2	RW	**DBASE address, 4-bytes aligned.**
1:0	RO	**Reserved**

Offset: E0	FSIME0: DMA Channel3 Control Register	Init = 0x00000000
Bit	**R/W**	**Description**
31:17	RO	**Reserved**
16	RW	**FSI access direction.** **0: FSI write, DMA to FSI.** **1: FSI read, FSI to DMA.**
15:0	RW	**TCOUNT, transfer counter in double words.**

Offset: E4	FSIME4: DMA Function Enable Register	Init = 0x00000000

Bit	R/W	Description
31:4	RO	**Reserved**
3	RW	**Start DMA Channel 3.**
2	RW	**Start DMA Channel 2.**
1	RW	**Start DMA Channel 1.**
0	RW	**Start DMA Channel 0.**

Note :
Start a event in DMAx channel with FSI.
1: enable,
0: disable.

Offset: E8	FSIME8: FSI Control Register	Init = 0x00000000

Bit	R/W	Description
31:11	RO	**Reserved**
10	RW	**OAUTOINC**
9	RW	**OAUTOTRI**
8	RW	**OPBx select trigger enable.** **1: FSIM10[0] or FSIM28[0] transition from 0 to 1 is a similar action what FSIM04[0] does.** **(for debugging purpose only)**
7	RO	**Reserved**
6	RW	**OPB1 read data bit order inverse.**
5	RW	**OPB1 write data bit order inverse.**
4	RW	**OPB1 address bus bit order inverse.**
3	RO	**Reserved**
2	RW	**OPB0 read data bit order inverse.**
1	RW	**OPB0 write data bit order inverse.**
0	RW	**OPB0 address bus bit order inverse.**

62.4.2 FSI Control Register : Base Address = FSIM08[31:12]

Offset: 00	FSIC00: Mode Register	Init = 0x00000000

Bit	R/W	Description
0	RW	**Enable port dma and interrupt polling.** **0 : disable** **1 : enable**
1	RW	**Enable bridge crc error retry.** **0 : disable** **1 : enable**
2	RW	**Enable short address command.** **0 : disable** **1 : enable**
3	RW	**Enable parity check.** **0 : disable** **1 : enable**
4:13	RW	**Clock rate ratio 0**

to next page

from previous page

14:23	RW	**Clock rate ratio 1**
24:31		**Reserved**

63 PCIe to LPC Controller (PCIE2LPC)

63.1 Overview

AST2600 integrates PCIe to LPC Controller, which includes IPMI 2.0/1.1 compliant BMC controller. There are totally 11 registers, which is listed below, to control the various functions supported by AST2600 . Each register has its own specific offset value to derive its physical address location.

Base Address of XLPC Controller = 0x1E78_9800
Physical address of register = (Base address of XLPC Controller) + Offset

XHICRB : XHost Interface Control Register B
XHICRC : XHost Interface Control Register C
XLADR4 : XLPC Channel #4 Address register
XIDR4 : XInput Data Register 4
XODR4 : XOutput Data Register 4
XSTR4 : XStatus Register 4
XMBXDAT_0 : XMailBox Data Register 0
XMBXDAT_1 : XMailBox Data Register 1
XMBXDAT_2 : XMailBox Data Register 2
XMBXDAT_3 : XMailBox Data Register 3
XMBXDAT_4 : XMailBox Data Register 4
XMBXDAT_5 : XMailBox Data Register 5
XMBXDAT_6 : XMailBox Data Register 6
XMBXDAT_7 : XMailBox Data Register 7
XMBXDAT_8 : XMailBox Data Register 8
XMBXDAT_9 : XMailBox Data Register 9
XMBXDAT_A : XMailBox Data Register A
XMBXDAT_B : XMailBox Data Register B
XMBXDAT_C : XMailBox Data Register C
XMBXDAT_D : XMailBox Data Register D
XMBXDAT_E : XMailBox Data Register E
XMBXDAT_F : XMailBox Data Register F
XMBXDAT_10 : XMailBox Data Register 10
XMBXDAT_11 : XMailBox Data Register 11
XMBXDAT_12 : XMailBox Data Register 12
XMBXDAT_13 : XMailBox Data Register 13
XMBXDAT_14 : XMailBox Data Register 14
XMBXDAT_15 : XMailBox Data Register 15
XMBXDAT_16 : XMailBox Data Register 16
XMBXDAT_17 : XMailBox Data Register 17
XMBXDAT_18 : XMailBox Data Register 18
XMBXDAT_19 : XMailBox Data Register 19
XMBXDAT_1A : XMailBox Data Register 1A
XMBXDAT_1B : XMailBox Data Register 1B
XMBXDAT_1C : XMailBox Data Register 1C
XMBXDAT_1D : XMailBox Data Register 1D
XMBXDAT_1E : XMailBox Data Register 1E
XMBXDAT_1F : XMailBox Data Register 1F
XMBXSTS_0 : XMailBox Status Register 0
XMBXSTS_1 : XMailBox Status Register 1
XMBXSTS_2 : XMailBox Status Register 2
XMBXSTS_3 : XMailBox Status Register 3
XMBXBCR : XMailBox BMC Control Register
XMBXHCR : XMailBox Host Control Register

XMBXBIE_0 : XMailBox BMC Interrupt Enable Register 0
XMBXBIE_1 : XMailBox BMC Interrupt Enable Register 1
XMBXBIE_2 : XMailBox BMC Interrupt Enable Register 2
XMBXBIE_3 : XMailBox BMC Interrupt Enable Register 3
XMBXHIE_0 : XMailBox Host Interrupt Enable Register 0
XMBXHIE_1 : XMailBox Host Interrupt Enable Register 1
XMBXHIE_2 : XMailBox Host Interrupt Enable Register 2
XMBXHIE_3 : XMailBox Host Interrupt Enable Register 3

63.2 Features

- Directly connected to APB bus interface
- Operation mode
  - Salve mode: designed for BMC functions (MMIO read write cycles)
- Support MSI (reduce polling time)
- Support two set of XVirtual UART (16550) (MSI)
- Compliant with IPMI version 2.0 KCS mode
  - Channel #4 supports KCS interface
- Three register sets to support four programmable I/O channels. Each register set includes:
  - Input data register (IDR4)
  - Output data register (ODR4)
  - Status register (STR4)

63.3 Registers : Base Address = 0x1E78:9800

Attribute Definition:

Attribute	: Description
R	: Readable
W	: Writable
RO	: Read Only
WO	: Write Only
W0C	: Write '0' to clear value to 0
W1C	: Write '1' to clear value to 0
W1T	: Write '1' to toggle value
W0S	: Write '0' to set value to 1
W1S	: Write '1' to set value to 1
U	: Unknown value
P	: Initialized by PWRST_N
H	: Initialized by LPC_RST_N
L	: Initialized by LPC_LRST_N

XHICRB: XHost Interface Control Register B					Offset: 100h
Bit	**Name**	**Initial**	**Slave**	**Host**	**Description**
1		0P	RW	-	**Enable XKCS channel #4 receive completion interrupt**
0		0P	RW	-	**Enable XKCS channel #4**

XHICRC: XHost Interface Control Register C					Offset: 104h
Bit	**Name**	**Initial**	**Slave**	**Host**	**Description**
31		0	RO	-	**Reserved**
30		0P	RW	-	**DisXSIOLDN14: Disables the XSIO LDN #14, XSIO Mailbox. Refer section 77.1 for list of XSIO LDNs. Default is Enabled.**
29:8		0	RO	-	**Reserved**
7: 4		U	RW	-	**Select ID bit[3:0] of MSI for XKCS channel #4**
1		0P	RW	-	**Disable Host MSI of XKCS channel #4 auto-clear by OBF4**
0		0P	RW	-	**Host MSI enable for XKCS channel #4**

XLADR4: XLPC Channel #4 Address register					Offset: 110h
Bit	**Name**	**Initial**	**Slave**	**Host**	**Description**
29:16		U	RW	-	**XKCS channel #4 second (command/status) base offset register address bit[15:2] behind PCIe BMC Device BAR1**
13: 0		U	RW	-	**XKCS channel #4 first (data) base offset register address bit[15:2] behind PCIe BMC Device BAR1**

XIDR4: XInput Data Register 4					Offset: 114h
Bit	**Name**	**Initial**	**Slave**	**Host**	**Description**
7: 0		U	RO	WO	**XKCS channel #4 input data bit[7:0]**

XODR4: XOutput Data Register 4					Offset: 118h
Bit	**Name**	**Initial**	**Slave**	**Host**	**Description**
7: 0		U	RW	RO	**XKCS channel #4 output data bit[7:0]**

XSTR4: XStatus Register 4					Offset: 11Ch
Bit	**Name**	**Initial**	**Slave**	**Host**	**Description**
7	**DBU47**	1H	RW	RO	**Defined by user**
6	**DBU46**	1H	RW	RO	**Defined by user**
5	**DBU45**	0H	RW	RO	**Defined by user**
4	**DBU44**	0H	RW	RO	**Defined by user**
3	**C/D4**	0P	RO	RO	**Command/Data**
2	**DBU42**	0H	RW	RO	**Defined by user**
1	**IBF4**	0P	RO	RO	**Input data register full**
0	**OBF4**	0P	RW0C	RO	**Output data register full**

XMBXDAT_0: XMailBox Data Register 0 **Offset: 200h**

Bit	Name	Initial	Slave	Host	Description
31:24		0P	RO	-	**XMailBox Data Register 3 bit[7:0]**
23:16		0P	RO	-	**XMailBox Data Register 2 bit[7:0]**
15: 8		0P	RO	-	**XMailBox Data Register 1 bit[7:0]**
7: 0		0P	RW	RW	**XMailBox Data Register 0 bit[7:0]**

XMBXDAT_1: XMailBox Data Register 1 **Offset: 204h**

Bit	Name	Initial	Slave	Host	Description
31: 8		0	RO	-	**Reserved**
7: 0		0P	RW	RW	**XMailBox Data Register 1 bit[7:0]**

XMBXDAT_2: XMailBox Data Register 2 **Offset: 208h**

Bit	Name	Initial	Slave	Host	Description
31: 8		0	RO	-	**Reserved**
7: 0		0P	RW	RW	**XMailBox Data Register 2 bit[7:0]**

XMBXDAT_3: XMailBox Data Register 3 **Offset: 20Ch**

Bit	Name	Initial	Slave	Host	Description
31: 8		0	RO	-	**Reserved**
7: 0		0P	RW	RW	**XMailBox Data Register 3 bit[7:0]**

XMBXDAT_4: XMailBox Data Register 4 **Offset: 210h**

Bit	Name	Initial	Slave	Host	Description
31:24		0P	RO	-	**XMailBox Data Register 7 bit[7:0]**
23:16		0P	RO	-	**XMailBox Data Register 6 bit[7:0]**
15: 8		0P	RO	-	**XMailBox Data Register 5 bit[7:0]**
7: 0		0P	RW	RW	**XMailBox Data Register 4 bit[7:0]**

XMBXDAT_5: XMailBox Data Register 5 **Offset: 214h**

Bit	Name	Initial	Slave	Host	Description
31: 8		0	RO	-	**Reserved**
7: 0		0P	RW	RW	**XMailBox Data Register 5 bit[7:0]**

XMBXDAT_6: XMailBox Data Register 6 **Offset: 218h**

Bit	Name	Initial	Slave	Host	Description
31: 8		0	RO	-	**Reserved**
7: 0		0P	RW	RW	**XMailBox Data Register 6 bit[7:0]**

XMBXDAT_7: XMailBox Data Register 7 **Offset: 21Ch**

Bit	Name	Initial	Slave	Host	Description
31: 8		0	RO	-	**Reserved**
7: 0		0P	RW	RW	**XMailBox Data Register 7 bit[7:0]**

XMBXDAT_8: XMailBox Data Register 8 **Offset: 220h**

Bit	Name	Initial	Slave	Host	Description
31:24		0P	RO	-	**XMailBox Data Register B bit[7:0]**
23:16		0P	RO	-	**XMailBox Data Register A bit[7:0]**
15: 8		0P	RO	-	**XMailBox Data Register 9 bit[7:0]**
7: 0		0P	RW	RW	**XMailBox Data Register 8 bit[7:0]**

XMBXDAT_9: XMailBox Data Register 9 **Offset: 224h**

Bit	Name	Initial	Slave	Host	Description
31: 8		0	RO	-	**Reserved**
7: 0		0P	RW	RW	**XMailBox Data Register 9 bit[7:0]**

XMBXDAT_A: XMailBox Data Register A **Offset: 228h**

Bit	Name	Initial	Slave	Host	Description
31: 8		0	RO	-	**Reserved**
7: 0		0P	RW	RW	**XMailBox Data Register A bit[7:0]**

XMBXDAT_B: XMailBox Data Register B **Offset: 22Ch**

Bit	Name	Initial	Slave	Host	Description
31: 8		0	RO	-	**Reserved**
7: 0		0P	RW	RW	**XMailBox Data Register B bit[7:0]**

XMBXDAT_C: XMailBox Data Register C **Offset: 230h**

Bit	Name	Initial	Slave	Host	Description
31:24		0P	RO	-	**XMailBox Data Register F bit[7:0]**
23:16		0P	RO	-	**XMailBox Data Register E bit[7:0]**
15: 8		0P	RO	-	**XMailBox Data Register D bit[7:0]**
7: 0		0P	RW	RW	**XMailBox Data Register C bit[7:0]**

XMBXDAT_D: XMailBox Data Register D **Offset: 234h**

Bit	Name	Initial	Slave	Host	Description
31: 8		0	RO	-	**Reserved**
7: 0		0P	RW	RW	**XMailBox Data Register D bit[7:0]**

XMBXDAT_E: XMailBox Data Register E **Offset: 238h**

Bit	Name	Initial	Slave	Host	Description
31: 8		0	RO	-	**Reserved**
7: 0		0P	RW	RW	**XMailBox Data Register E bit[7:0]**

XMBXDAT_F: XMailBox Data Register F **Offset: 23Ch**

Bit	Name	Initial	Slave	Host	Description
31: 8		0	RO	-	**Reserved**
7: 0		0P	RW	RW	**XMailBox Data Register F bit[7:0]**

XMBXDAT_10: XMailBox Data Register 10 **Offset: 240h**

Bit	Name	Initial	Slave	Host	Description
31:24		0P	RO	-	**XMailBox Data Register 13 bit[7:0]**
23:16		0P	RO	-	**XMailBox Data Register 12 bit[7:0]**
15: 8		0P	RO	-	**XMailBox Data Register 11 bit[7:0]**
7: 0		0P	RW	RW	**XMailBox Data Register 10 bit[7:0]**

XMBXDAT_11: XMailBox Data Register 11 **Offset: 244h**

Bit	Name	Initial	Slave	Host	Description
31: 8		0	RO	-	**Reserved**
7: 0		0P	RW	RW	**XMailBox Data Register 11 bit[7:0]**

XMBXDAT_12: XMailBox Data Register 12 **Offset: 248h**

Bit	Name	Initial	Slave	Host	Description
31: 8		0	RO	-	**Reserved**
7: 0		0P	RW	RW	**XMailBox Data Register 12 bit[7:0]**

XMBXDAT_13: XMailBox Data Register 13 **Offset: 24Ch**

Bit	Name	Initial	Slave	Host	Description
31: 8		0	RO	-	**Reserved**
7: 0		0P	RW	RW	**XMailBox Data Register 13 bit[7:0]**

XMBXDAT_14: XMailBox Data Register 14 **Offset: 250h**

Bit	Name	Initial	Slave	Host	Description
31:24		0P	RO	-	**XMailBox Data Register 17 bit[7:0]**
23:16		0P	RO	-	**XMailBox Data Register 16 bit[7:0]**
15: 8		0P	RO	-	**XMailBox Data Register 15 bit[7:0]**
7: 0		0P	RW	RW	**XMailBox Data Register 14 bit[7:0]**

XMBXDAT_15: XMailBox Data Register 15 **Offset: 254h**

Bit	Name	Initial	Slave	Host	Description
31: 8		0	RO	-	**Reserved**
7: 0		0P	RW	RW	**XMailBox Data Register 15 bit[7:0]**

XMBXDAT_16: XMailBox Data Register 16 **Offset: 258h**

Bit	Name	Initial	Slave	Host	Description
31: 8		0	RO	-	**Reserved**
7: 0		0P	RW	RW	**XMailBox Data Register 16 bit[7:0]**

XMBXDAT_17: XMailBox Data Register 17 **Offset: 25Ch**

Bit	Name	Initial	Slave	Host	Description
31: 8		0	RO	-	**Reserved**
7: 0		0P	RW	RW	**XMailBox Data Register 17 bit[7:0]**

XMBXDAT_18: XMailBox Data Register 18 **Offset: 260h**

Bit	Name	Initial	Slave	Host	Description
31:24		0P	RO	-	**XMailBox Data Register 1B bit[7:0]**
23:16		0P	RO	-	**XMailBox Data Register 1A bit[7:0]**
15: 8		0P	RO	-	**XMailBox Data Register 19 bit[7:0]**
7: 0		0P	RW	RW	**XMailBox Data Register 18 bit[7:0]**

XMBXDAT_19: XMailBox Data Register 19 **Offset: 264h**

Bit	Name	Initial	Slave	Host	Description
31: 8		0	RO	-	**Reserved**
7: 0		0P	RW	RW	**XMailBox Data Register 19 bit[7:0]**

XMBXDAT_1A: XMailBox Data Register 1A **Offset: 268h**

Bit	Name	Initial	Slave	Host	Description
31: 8		0	RO	-	**Reserved**
7: 0		0P	RW	RW	**XMailBox Data Register 1A bit[7:0]**

XMBXDAT_1B: XMailBox Data Register 1B **Offset: 26Ch**

Bit	Name	Initial	Slave	Host	Description
31: 8		0	RO	-	**Reserved**
7: 0		0P	RW	RW	**XMailBox Data Register 1B bit[7:0]**

XMBXDAT_1C: XMailBox Data Register 1C **Offset: 270h**

Bit	Name	Initial	Slave	Host	Description
31:24		0P	RO	-	**XMailBox Data Register 1F bit[7:0]**
23:16		0P	RO	-	**XMailBox Data Register 1E bit[7:0]**

to next page

from previous page

15: 8		0P	RO	-	**XMailBox Data Register 1D bit[7:0]**
7: 0		0P	RW	RW	**XMailBox Data Register 1C bit[7:0]**

XMBXDAT_1D: XMailBox Data Register 1D **Offset: 274h**

Bit	Name	Initial	Slave	Host	Description
31: 8		0	RO	-	**Reserved**
7: 0		0P	RW	RW	**XMailBox Data Register 1D bit[7:0]**

XMBXDAT_1E: XMailBox Data Register 1E **Offset: 278h**

Bit	Name	Initial	Slave	Host	Description
31: 8		0	RO	-	**Reserved**
7: 0		0P	RW	RW	**XMailBox Data Register 1E bit[7:0]**

XMBXDAT_1F: XMailBox Data Register 1F **Offset: 27Ch**

Bit	Name	Initial	Slave	Host	Description
31: 8		0	RO	-	**Reserved**
7: 0		0P	RW	RW	**XMailBox Data Register 1F bit[7:0]**

MBXSTS_0: XMailBox Status Register 0 **Offset: 280h**

Bit	Name	Initial	Slave	Host	Description
31:24		0	RO	-	**XMailBox Status Register bit[31:24]**
23:16		0	RO	-	**XMailBox Status Register bit[23:16]**
15: 8		0	RO	-	**XMailBox Status Register bit[15: 8]**
7: 0		0P	RW1C	RW1C	**XMailBox Status Register bit[7: 0]**

MBXSTS_1: XMailBox Status Register 1 **Offset: 284h**

Bit	Name	Initial	Slave	Host	Description
31: 8		0	RO	-	**Reserved**
7: 0		0P	RW1C	RW1C	**XMailBox Status Register bit[15: 8]**

MBXSTS_2: XMailBox Status Register 2 **Offset: 288h**

Bit	Name	Initial	Slave	Host	Description
31: 8		0	RO	-	**Reserved**
7: 0		0P	RW1C	RW1C	**XMailBox Status Register bit[23:16]**

MBXSTS_3: XMailBox Status Register 3 **Offset: 28Ch**

Bit	Name	Initial	Slave	Host	Description
31: 8		0	RO	-	**Reserved**
7: 0		0P	RW1C	RW1C	**XMailBox Status Register bit[31:24]**

MBXBCR: XMailBox BMC Control Register **Offset: 290h**

Bit	Name	Initial	Slave	Host	Description
15		0P	RO	-	**Interrupt status from the BMC to the Host**
14:10		0	RO	-	**Reserved**
9		0P	RO	-	**Mask interrupt to the Host from the status bit, XMBXHCR[7]**
8		0P	RO	-	**Generate interrupt to the BMC; set the status bit, XMBXBCR[7]**
7		0P	RW1C	RO	**Interrupt status from the Host to the BMC**
6: 2		0	RO	RO	**Reserved**
1		0P	RW	RO	**Mask interrupt to the BMC from the status bit, XMBXBCR[7]**
0		0P	W1T	RO	**Generate interrupt to the Host; set the status bit, XMBXHCR[7]**

MBXHCR: XMailBox Host Control Register **Offset: 294h**

Bit	Name	Initial	Slave	Host	Description
31: 8		0	RO	-	**Reserved**
7		0P	RO	RW1C	**Interrupt status from the BMC to the Host**
6: 2		0	RO		**Reserved**
1		0P	RO	RW	**Mask interrupt to the Host from the status bit, XMBXHCR[7]**
0		0P	RO	RW	**Generate interrupt to the BMC; set the status bit, XMBXBCR[7]**

MBXBIE_0: XMailBox BMC Interrupt Enable Register 0 **Offset: 2A0h**

Bit	Name	Initial	Slave	Host	Description
31:24		0P	RO	-	**XMailBox BMC Interrupt Enable bit[31:24]**
23:16		0P	RO	-	**XMailBox BMC Interrupt Enable bit[23:16]**
15: 8		0P	RO	-	**XMailBox BMC Interrupt Enable bit[15:8]**
7: 0		0P	RW	RO	**XMailBox BMC Interrupt Enable bit[7:0]**

MBXBIE_1: XMailBox BMC Interrupt Enable Register 1 **Offset: 2A4h**

Bit	Name	Initial	Slave	Host	Description
31: 8		0	RO	-	**Reserved**
7: 0		0P	RW	RO	**XMailBox BMC Interrupt Enable bit[15:8]**

MBXBIE_2: XMailBox BMC Interrupt Enable Register 2 **Offset: 2A8h**

Bit	Name	Initial	Slave	Host	Description
31: 8		0	RO	-	**Reserved**
7: 0		0P	RW	RO	**XMailBox BMC Interrupt Enable bit[23:16]**

MBXBIE_3: XMailBox BMC Interrupt Enable Register 3					Offset: 2ACh
Bit	**Name**	**Initial**	**Slave**	**Host**	**Description**
31: 8		0	RO	-	**Reserved**
7: 0		0P	RW	RO	**XMailBox BMC Interrupt Enable bit[31:24]**

MBXHIE_0: XMailBox Host Interrupt Enable Register 0					Offset: 2B0h
Bit	**Name**	**Initial**	**Slave**	**Host**	**Description**
31:24		0P	RO	-	**XMailBox Host Interrupt Enable bit[31:24]**
23:16		0P	RO	-	**XMailBox Host Interrupt Enable bit[23:16]**
15: 8		0P	RO	-	**XMailBox Host Interrupt Enable bit[15:8]**
7: 0		0P	RO	RW	**XMailBox Host Interrupt Enable bit[7:0]**

MBXHIE_1: XMailBox Host Interrupt Enable Register 1					Offset: 2B4h
Bit	**Name**	**Initial**	**Slave**	**Host**	**Description**
31: 8		0	RO	-	**Reserved**
7: 0		0P	RO	RW	**XMailBox Host Interrupt Enable bit[15:8]**

MBXHIE_2: XMailBox Host Interrupt Enable Register 2					Offset: 2B8h
Bit	**Name**	**Initial**	**Slave**	**Host**	**Description**
31: 8		0	RO	-	**Reserved**
7: 0		0P	RO	RW	**XMailBox Host Interrupt Enable bit[23:16]**

MBXHIE_3: XMailBox Host Interrupt Enable Register 3					Offset: 2BCh
Bit	**Name**	**Initial**	**Slave**	**Host**	**Description**
31: 8		0		-	**Reserved**
7: 0		0P	R	RW	**XMailBox Host Interrupt Enable bit[31:24]**

64 PCIe to VUART (PCIE2VUART)

64.1 Overview

AST2600 integrates two XVirtual UART modules providing PCIe virtual serial communication capabilities between host CPU and ARM CPU. XVirtual UART is equipped with two sets of registers compatible with the industry defector standard - 16550 UART.

One set is for host CPU; the other set is for ARM CPU. Host CPU and ARM CPU can communicate with each other like there is a physical UART link between them, but the related data transfer actually is just through pure register read/write transfers in the chip. The base offset address for host CPU to access UART registers through PCIe bus can be programmed by ARM CPU by the extended related registers (XVxUART28 and XVxUART2C)

In other words, XVirtual UART is a single module with dual heads. One side is exposed to Host over the PCIe BMC Device (see the Host side registers below), the other side is exposed to the BMC (see the Slave side register below). When the host writes to XV1UART00 (Host), the data goes into an internal 16 byte FIFO, the same FIFO can be read from the XV1UART00 (Slave) from the BMC side. Similarly, BMC writes goes to another FIFO that can be read from the host side by reading the XV1UART00 (Host). The rest of the control registers are only required when you take the Byte data and shift it out serially to a connector but in this implementation we never do serial shift internally. The data stays as byte, so whatever programming in host/BMC side to other UART control registers are just ignored.

Base Address of XVirtual 1 UART = 0x1E78_7800
Register Address of XVirtual 1 UART = (Base Address of XV1UART) + Offset

The following registers can be access by host CPU through PCIe bus.

XV1UART00 (Host): Receiving Buffer Register (Read, DLAB = 0)
XV1UART00 (Host): Transmit Holding Register (Write, DLAB = 0)
XV1UART00 (Host): Divisor Latch Low Register (Read/Write, DLAB = 1)
XV1UART04 (Host): Interrupt Enable Register (Read/Write, DLAB = 0)
XV1UART04 (Host): Divisor Latch High Register (Read/Write, DLAB = 1)
XV1UART08 (Host): FIFO Control Register
XV1UART0C (Host): Line Control Register
XV1UART10 (Host): Modem Control Register
XV1UART14 (Host): Line Status Register
XV1UART18 (Host): Modem Status Register
XV1UART1C (Host): Scratch Register

The following registers can be access by ARM CPU through APB bus.

XV1UART00 (Slave): Receiving Buffer Register (Read, DLAB = 0)
XV1UART00 (Slave): Transmit Holding Register (Write, DLAB = 0)
XV1UART00 (Slave): Divisor Latch Low Register (Read/Write, DLAB = 1)
XV1UART04 (Slave): Interrupt Enable Register (Read/Write, DLAB = 0)
XV1UART04 (Slave): Divisor Latch High Register (Read/Write, DLAB = 1)
XV1UART08 (Slave): FIFO Control Register
XV1UART0C (Slave): Line Control Register
XV1UART10 (Slave): Modem Control Register
XV1UART14 (Slave): Line Status Register
XV1UART18 (Slave): Modem Status Register
XV1UART1C (Slave): Scratch Register
XV1UART20 (Slave): General Control Register A

XV1UART24 (Slave): General Control Register B
XV1UART28 (Slave): XVUART Address Register L
XV1UART2C (Slave): XVUART Address Register H
XV1UART30 (Slave): General Control Register E
XV1UART34 (Slave): General Control Register F
XV1UART38 (Slave): General Control Register G
XV1UART3C (Slave): General Control Register H

Base Address of XVirtual 2 UART = 0x1E78_8800
Register Address of XVirtual 2 UART = (Base Address of XV2UART) + Offset

The following registers can be access by host CPU through PCIe bus.

XV2UART00 (Host): Receiving Buffer Register (Read, DLAB = 0)
XV2UART00 (Host): Transmit Holding Register (Write, DLAB = 0)
XV2UART00 (Host): Divisor Latch Low Register (Read/Write, DLAB = 1)
XV2UART04 (Host): Interrupt Enable Register (Read/Write, DLAB = 0)
XV2UART04 (Host): Divisor Latch High Register (Read/Write, DLAB = 1)
XV2UART08 (Host): FIFO Control Register
XV2UART0C (Host): Line Control Register
XV2UART10 (Host): Modem Control Register
XV2UART14 (Host): Line Status Register
XV2UART18 (Host): Modem Status Register
XV2UART1C (Host): Scratch Register

The following registers can be access by ARM CPU through APB bus.

XV2UART00 (Slave): Receiving Buffer Register (Read, DLAB = 0)
XV2UART00 (Slave): Transmit Holding Register (Write, DLAB = 0)
XV2UART00 (Slave): Divisor Latch Low Register (Read/Write, DLAB = 1)
XV2UART04 (Slave): Interrupt Enable Register (Read/Write, DLAB = 0)
XV2UART04 (Slave): Divisor Latch High Register (Read/Write, DLAB = 1)
XV2UART08 (Slave): FIFO Control Register
XV2UART0C (Slave): Line Control Register
XV2UART10 (Slave): Modem Control Register
XV2UART14 (Slave): Line Status Register
XV2UART18 (Slave): Modem Status Register
XV2UART1C (Slave): Scratch Register
XV2UART20 (Slave): General Control Register A
XV2UART24 (Slave): General Control Register B
XV2UART28 (Slave): XVUART Address Register L
XV2UART2C (Slave): XVUART Address Register H
XV2UART30 (Slave): General Control Register E
XV2UART34 (Slave): General Control Register F
XV2UART38 (Slave): General Control Register G
XV2UART3C (Slave): General Control Register H

64.2 Features

- Directly connected to both APB bus and PCIe Bus
- Support two XVirtual UART interfaces
- Separate transmit & receive FIFO buffer (16x8) to reduce CPU overhead
- Programmable base offset address for host CPU to access UART registers through PCIe bus

64.3 XV1UART Registers : Base Address = 0x1E78:7800

Offset: 00h	XV1UART00 (Host)	Init = X
Bit	**R/W**	**Description**
31:8	RO	**Reserved (0)**
RBR: Receiving Buffer Register (DLAB = 0)		
7 :0	RO	**Receiving Buffer Register** The RBR contains the data byte received on the serial input port (sin). The data in this register is valid only if the Data Ready (DR) bit in the Line status Register (UARTLSR) is set. When the FIFOs are programmed ON, this register accesses the head of the receive FIFO. If the receive FIFO is full and this register is not read before the next data character arrives, then the data already in the FIFO will be preserved but any incoming data will be lost. An overrun error will also occur.
THR: Transmit Holding Register (DLAB = 0)		
7:0	WO	**Transmit Holding Register** The THR contains data to be transmitted on the serial output port (sout). Data can be written to the THR any time that the THR Empty (THRE) bit of the Line Status Register (UARTLSR) is set. If FIFOs are enabled and THRE is set, 16 number of characters of data may be written to the THR before the FIFO is full. Any attempt to write data when the FIFO is full results in the write data being lost.
DLL: Divisor Latch Low Register (DLAB = 1)		
7 :0	RW	**Divisor Latch Low Register** This DLL register is designed for software compatible. The actual output baud rate is equal to PCLK clock frequency, 31.25MHz.

Offset: 00h	XV1UART00 (Slave)	Init = X
Bit	**R/W**	**Description**
31:8	RO	**Reserved (0)**
RBR: Receiving Buffer Register (DLAB = 0)		
7 :0	RO	**Receiving Buffer Register** The RBR contains the data byte received on the serial input port (sin). The data in this register is valid only if the Data Ready (DR) bit in the Line status Register (UARTLSR) is set. When the FIFOs are programmed ON, this register accesses the head of the receive FIFO. If the receive FIFO is full and this register is not read before the next data character arrives, then the data already in the FIFO will be preserved but any incoming data will be lost. An overrun error will also occur.
THR: Transmit Holding Register (DLAB = 0)		
7:0	WO	**Transmit Holding Register** The THR contains data to be transmitted on the serial output port (sout). Data can be written to the THR any time that the THR Empty (THRE) bit of the Line Status Register (UARTLSR) is set. If FIFOs are enabled and THRE is set, 16 number of characters of data may be written to the THR before the FIFO is full. Any attempt to write data when the FIFO is full results in the write data being lost.
DLL: Divisor Latch Low Register (DLAB = 1 and XV1UART34[2] = 0)		

to next page

from previous page

7 :0	RW	**Divisor Latch Low Register** This DLL register is designed for software compatible. The actual output baud rate is equal to PCLK clock frequency, 31.25MHz.
DLL: Divisor Latch Low Register (DLAB = 1 and XV1UART34[2] = 1)		
7 :0	RO	**Divisor Latch Low Register (Host)** The slave (BMC) can read the DLL of the Host by this register.

Offset: 04h	**XV1UART04 (Host)**	**Init = 0**
Bit	**R/W**	**Description**
31:8	RO	**Reserved (0)**
IER: Interrupt Enable Register (DLAB = 0)		
7	RW	**Enable FIFO 1/2 full THRE Interrupt Mode if XV1UART34[6]=1** 0: Disable FIFO 1/2 full THRE interrupt mode 1: Enable FIFO 1/2 full THRE interrupt mode if XV1UART34[6]=1
6 :4	RO	**Reserved (0)**
3	RW	**Enable Modem Status Interrupt** 0: Disable interrupt 1: Enable interrupt
2	RW	**Enable Receiver Line Status Interrupt** 0: Disable interrupt 1: Enable interrupt
1	RW	**Enable Transmitter Holding Register Empty Interrupt** 0: Disable interrupt 1: Enable interrupt
0	RW	**Enable Received Data Available Interrupt** 0: Disable interrupt 1: Enable interrupt
DLH: Divisor Latch High Register (DLAB = 1)		
7 :0	RW	**Divisor latch (High)** This DLH register is designed for software compatible. The actual output baud rate is equal to PCLK clock frequency, 31.25MHz.

Offset: 04h	**XV1UART04 (Slave)**	**Init = 0**
Bit	**R/W**	**Description**
31:8	RO	**Reserved (0)**
IER: Interrupt Enable Register (DLAB = 0)		
7 :4	RO	**Reserved (0)**
3	RW	**Enable Modem Status Interrupt** 0: Disable interrupt 1: Enable interrupt
2	RW	**Enable Receiver Line Status Interrupt** 0: Disable interrupt 1: Enable interrupt
1	RW	**Enable Transmitter Holding Register Empty Interrupt** 0: Disable interrupt 1: Enable interrupt

to next page

from previous page

0	RW	**Enable Received Data Available Interrupt** 0: Disable interrupt 1: Enable interrupt
DLH: Divisor Latch High Register (DLAB = 1 and XV1UART34[2] = 0)		
7 :0	RW	**Divisor latch (High)** This DLH register is designed for software compatible. The actual output baud rate is equal to PCLK clock frequency, 31.25MHz.
DLH: Divisor Latch High Register (DLAB = 1 and XV1UART34[2] = 1)		
7 :0	RO	**Divisor latch (High) (Host)** The slave (BMC) can read the DLH of the Host by this register.

Offset: 08h **XV1UART08 (Host): (IIR) Interrupt Identity Register** **Init = 0xC1**

Bit	R/W	Description
31:4	RO	**Reserved (0)**
3:1	RO	**Interrupt Decoding Table** The content of this register can be used to identify the source of the current interrupt based on the following: 000: Modem Status Changed 001: THR empty 010: Received Data Available 011: Receiver Status 110: Character Time Out For more information about Interrupt Identity, see the following Table for detailed description.
0	RO	Indicates that an interrupt is pending when its logic 0. When its 1, no interrupt is pending.
Note : The IIR enables the programmer to retrieve what is the current highest priority pending interrupt.		

XV1UART Interrupt Type Decoding

Bit [3:1]	Priority	Interrupt Type	Interrupt Source	Interrupt Clear Control
011	Highest	Receiver line status	Overrun or Break Interrupt.	Reading the Line Status Register. IER[2](Host)=0
010	2nd	Received Data available	FIFO ON: RX FIFO trigger level reached	FIFO ON: FIFO drops below the trigger level IER[0](Host)=0 FCR[1](Host)=1
110	3rd	Character Timeout indication	Theres at least 1 character in the FIFO but no character has been input to the FIFO or read from it for 1/512, 1/256, 1/128 or 1/64 second.	Reading the RBR. IER[0](Host)=0 FCR[1](Host)=1 VUART34[1](Slave)=1
001	4th	Transmitter Holding register empty	Transmitter Holding register empty	Reading the IIR or writing into THR
000	5th	Modem status	nCTS (Clear to send), nDSR (data set ready)	Reading the MSR.

Offset: 08h	XV1UART08 (Host): (FCR) FIFO Control Register	Init = 0x01

Bit	R/W	Description
31:8	RO	**Reserved (0)**
7:6	W	**Define the Receiver FIFO Interrupt trigger level.** 00: 1 byte received 01: 4 bytes received 10: 8 bytes received 11: 14 bytes received
5:3	RO	**Reserved (0)**
2	W	**Transmit FIFO Reset** Writing 1 to this bit clears the Transmitter FIFO and resets its logic.
1	W	**Receive FIFO Reset** Writing 1 to this bit clears the Receiver FIFO and resets its logic.
0	W	**Enable UART FIFO** 0: Disable FIFO 1: Enable FIFO The value of this register is always logic '1'.

Note :
The FCR allows selection of the FIFO trigger level (the number of bytes in FIFO required to enable the Received Data Available interrupt). In addition, the FIFOs can be cleared using this register.

Offset: 08h	XV1UART08 (Slave): (IIR) Interrupt Identity Register	Init = 0xC1

Bit	R/W	Description
31:4	RO	**Reserved (0)**
3:1	RO	**Interrupt Decoding Table** The content of this register can be used to identify the source of the current interrupt based on the following: 000: Modem Status Changed 001: THR empty 010: Received Data Available 011: Receiver Status 110: Character Time Out For more information about Interrupt Identity, see the following Table for detailed description.
0	RO	Indicates that an interrupt is pending when its logic 0. When its 1, no interrupt is pending.

Note :
The IIR enables the programmer to retrieve what is the current highest priority pending interrupt.

XV1UART Interrupt Type Decoding				
Bit [3:1]	Priority	Interrupt Type	Interrupt Source	Interrupt Clear Control
011	Highest	Receiver line status	Overrun or Break Interrupt.	Reading the Line Status Register. IER[2](Slave)=0 FCR[1](Slave)=1 if Overrun
010	2nd	Received Data available	FIFO ON: RX FIFO trigger level reached	FIFO ON: FIFO drops below the trigger level IER[0](Slave)=0 FCR[1](Slave)=1
110	3rd	Character Timeout indication	Theres at least 1 character in the FIFO but no character has been input to the FIFO or read from it for 1/512, 1/256, 1/128 or 1/64 second.	Reading the RBR. IER[0](Slave)=0 FCR[1](Slave)=1 V1UART34[0](Slave)=1
001	4th	Transmitter Holding register empty	Transmitter Holding register empty	Reading the IIR or writing into THR FCR[2](Slave)=1
000	5th	Modem status	nCTS (Clear to send), nDSR (data set ready)	Reading the MSR.

Offset: 08h	XV1UART08 (Slave): (FCR) FIFO Control Register	Init = 0x01
Bit	**R/W**	**Description**
31:8	RO	**Reserved (0)**
7:6	W	**Define the Receiver FIFO Interrupt trigger level.** 00: 1 byte received 01: 4 bytes received 10: 8 bytes received 11: 14 bytes received
5:3	RO	**Reserved (0)**
2	W	**Transmit FIFO Reset** Writing 1 to this bit clears the Transmitter FIFO and resets its logic.
1	W	**Receive FIFO Reset** Writing 1 to this bit clears the Receiver FIFO and resets its logic.
0	W	**Enable UART FIFO** 0: Disable FIFO 1: Enable FIFO The value of this register is always logic '1'.
Note : The FCR allows selection of the FIFO trigger level (the number of bytes in FIFO required to enable the Received Data Available interrupt). In addition, the FIFOs can be cleared using this register.		

Offset: 0Ch	XV1UART0C (Host): (LCR) Line Control Register	Init = 0x03
Bit	**R/W**	**Description**
31:8	RO	**Reserved (0)**

to next page

from previous page

7	RW	**DLAB: Divisor latch access bit** 0: The normal registers are accessed. 1: The divisor latches can be accessed. Setting this bit will enable the reading and writing of the Divisor Latch register (DLL and DLH) to set the baud rate of the UART. This bit must be cleared after initial baud rate setup in order to access other registers.
6	RW	**Break Control bit.** 0: break is disabled. 1: break event is transmitted to the Slave side.
5:2	RW	**Reserved**
1:0	RW	**Select number of bits per character** 00: 5 bits. 01: 6 bits. 10: 7 bits. 11: 8 bits.

Offset: 0Ch **XV1UART0C (Slave): (LCR) Line Control Register** **Init = 0x03**

Bit	R/W	Description
31:8	RO	**Reserved (0)**
7	RW	**DLAB: Divisor latch access bit** 0: The normal registers are accessed. 1: The divisor latches can be accessed. Setting this bit will enable the reading and writing of the Divisor Latch register (DLL and DLH) to set the baud rate of the UART. This bit must be cleared after initial baud rate setup in order to access other registers.
6	RW	**Break Control bit.** 0: break is disabled. 1: break event is transmitted to the Host side.
5:2	RW	**Reserved**
1:0	RW	**Select number of bits per character** 00: 5 bits. 01: 6 bits. 10: 7 bits. 11: 8 bits.

Offset: 10h **XV1UART10 (Host): (MCR) Modem Control Register** **Init = 0**

Bit	R/W	Description
31:8	RO	**Reserved (0)**
7	RO	**Transmit FIFO full.** 0: transmit FIFO not full. 1: transmit FIFO full.
6:5	RO	**Reserved (0)**

to next page

from previous page

4	RW	**Loopback mode.** 0: normal operation. 1: loopback mode. The signal of the transmitter shift register is internally connected to the input of the receiver shift register. The following connections are made: nDTR → nDSR nRTS → nCTS Out1 → nRI Out2 → nDCD
3	RW	**Out2.** In loopback mode, connected to Data Carrier Detect(nDCD) input.
2	RW	**Out1.** In loopback mode, connected to Ring Indicator (nRI)signal input.
1	RW	**Request To Send (nRTS) signal control.** 0: nRTS is '1' 1: nRTS is '0'
0	RW	**Data Terminal Ready (nDTR) signal control.** 0: nDTR is '1' 1: nDTR is '0'

Offset: 10h **XV1UART10 (Slave): (MCR) Modem Control Register** **Init = 0**

Bit	R/W	Description
31:5	RO	**Reserved (0)**
4	RW	**Loopback mode.** 0: normal operation. 1: loopback mode. The signal of the transmitter shift register is internally connected to the input of the receiver shift register. The following connections are made: nDTR → nDSR nRTS → nCTS Out1 → nRI Out2 → nDCD
3	RW	**Out2.** In loopback mode, connected to Data Carrier Detect(nDCD) input.
2	RW	**Out1.** In loopback mode, connected to Ring Indicator (nRI)signal input.
1	RW	**Request To Send (nRTS) signal control.** 0: nRTS is '1' 1: nRTS is '0'
0	RW	**Data Terminal Ready (nDTR) signal control.** 0: nDTR is '1' 1: nDTR is '0'

Offset: 14h **XV1UART14 (Host): (LSR) Line Status Register** **Init = 0x60**

Bit	R/W	Description
31:7	RO	**Reserved (0)**
6	RO	**Transmitter empty** 1: The THR or FIFO and the Transmitter Shift Register are both empty. 0: Otherwise.

to next page

from previous page

5	RO	**Transmitter holding register empty** 1: The THR or FIFO and the Transmitter Shift Register are both empty. This also causes a THRE Interrupt to occur, if the THRE Interrupt is enabled. 0: Otherwise.
4	RO	**Break interrupt** 1: The Break Control bit of Line Control Register on the opposite side is set. 0: No break condition in the current character.
3:2	RO	**Reserved (0)**
1	RO	**Overrun error** 1: In the FIFO mode, an overrun error occurs when the FIFO is full and a new character arrives at the receiver. 0: No overrun state.
0	RO	**Data ready** 1: The receiver contains at least one character in the RBR or the receiver FIFO. 0: The RBR is read or the receiver FIFO is empty.

Offset: 14h **XV1UART14 (Slave): (LSR) Line Status Register** **Init = 0x60**

Bit	R/W	Description
31:7	RO	**Reserved (0)**
6	RO	**Transmitter empty** 1: The THR or FIFO and the Transmitter Shift Register are both empty. 0: Otherwise.
5	RO	**Transmitter holding register empty** 1: The THR or FIFO and the Transmitter Shift Register are both empty. This also causes a THRE Interrupt to occur, if the THRE Interrupt is enabled. 0: Otherwise.
4	RO	**Break interrupt** 1: The Break Control bit of Line Control Register on the opposite side is set. 0: No break condition in the current character.
3:2	RO	**Reserved (0)**
1	RO	**Overrun error** 1: In the FIFO mode, an overrun error occurs when the FIFO is full and a new character arrives at the receiver. 0: No overrun state.
0	RO	**Data ready** 1: The receiver contains at least one character in the RBR or the receiver FIFO. 0: The RBR is read or the receiver FIFO is empty.

Offset: 18h **XV1UART18 (Host): (MSR) Modem Status Register** **Init = X**

Bit	R/W	Description
31:8	RO	**Reserved (0)**
7	RO	**Complement of the nDSR input or equals to Out2(MCR[3]) in loopback mode.**
6	RO	**Out1(MCR[2]) in loopback mode.**
5	RO	**Complement of the nDSR input or equals to nDTR(MCR[0]) in loopback mode.**
4	RO	**Complement of the nCTS input or equals to nRTS(MCR[1]) in loopback mode.**
3	RO	**Delta Data Carrier Detect (DDCD) indicator** 1: The nDSR line has changed its state since the last time the CPU read the MSR. In loopback mode, DDCD reflects changes on MCR bit 3 (Out2)

to next page

from previous page

2	RO	**Trailing Edge of Ring Indicator (TERI) detector.** In loopback mode, TERI reflects when MCR bit 2 (Out1) has changed state from a high to a low.
1	RO	**Delta Data Set Ready (DDSR) indicator** 1: The nDSR line has changed its state since the last time the CPU read the MSR. In loopback mode, DDSR reflects changes on MCR bit 0 (DTR)
0	RO	**Delta Clear To Send (DCTS) indicator** 1: The nCTS line has changed its state since the last time the CPU read the MSR. In loopback mode, DCTS reflects changes on MCR bit 1 (RTS)

Offset: 18h **XV1UART18 (Slave): (MSR) Modem Status Register** **Init = X**

Bit	R/W	Description
31:8	RO	**Reserved (0)**
7	RO	**Complement of the nDSR input or equals to Out2(MCR[3]) in loopback mode.**
6	RO	**Out1(MCR[2]) in loopback mode.**
5	RO	**Complement of the nDSR input or equals to nDTR(MCR[0]) in loopback mode.**
4	RO	**Complement of the nCTS input or equals to nRTS(MCR[1]) in loopback mode.**
3	RO	**Delta Data Carrier Detect (DDCD) indicator** 1: The nDSR line has changed its state since the last time the CPU read the MSR.
2	RO	**Reserved (0)**
1	RO	**Delta Data Set Ready (DDSR) indicator** 1: The nDSR line has changed its state since the last time the CPU read the MSR.
0	RO	**Delta Clear To Send (DCTS) indicator** 1: The nCTS line has changed its state since the last time the CPU read the MSR.

Offset: 1Ch **XV1UART1C (Host): (SCR) Scratch Register** **Init = X**

Bit	R/W	Description
31:8	RO	**Reserved (0)**
7 :0	RW	**Scratch bits** This register can be used as a temporary storage, no specific definition.

Offset: 1Ch **XV1UART1C (Slave): (SCR) Scratch Register** **Init = X**

Bit	R/W	Description
31:8	RO	**Reserved (0)**
7 :0	RW	**Scratch bits (XV1UART34[2] = 0)** This register can be used as a temporary storage, no specific definition.
7 :0	RO	**Scratch bits (XV1UART34[2] = 1)** The slave (BMC) can read the SCR of the Host by this register.

Offset: 20h **XV1UART20 (Slave): General Control Register A** **Init = 0b00x0_xx00**

Bit	R/W	Description
31:8	RO	**Reserved (0)**
7 :6	RO	**Status of host-side Receiver FIFO Trigger**

to next page

from previous page

5	RW	**Disable Host-Tx-discard mode** 0: Enable Host-Tx-discard mode Slave HW (BMC) will throw data away automatically when Host Tx FIFO is not empty. Background: If Host Tx FIFO is full, and BMC FW or DMA engine cant read data from BMC Rx FIFO (e.g., BMC FW is busy and can't handle BMC Rx FIFO interrupts, XV1UART08 (Slave) [3:1] = 010 or 110), Host SW will not see XV1UART14 (Host) [6:5] = 11, and that may cause Host SW stops the next Host Tx data transit. 1: Disable Host-Tx-discard mode
4	RO	**Status of host-side loopback mode**
3 :2	RW	**Slave-side timeout time width selection bit [1:0]** 00: 1/3600 second if XV1UART38[1]=0 01: 1/7200 second if XV1UART38[1]=0 10: 1/14400 second if XV1UART38[1]=0 11: 1/28800 second if XV1UART38[1]=0 00: 512*PCLK if XV1UART38[1]=1 01: 256*PCLK if XV1UART38[1]=1 10: 128*PCLK if XV1UART38[1]=1 11: 64*PCLK if XV1UART38[1]=1
1	RW	**Reserved (0)**
0	RW	**Enable PCIe virtual UART** 0: Disable 1: Enable
Note : This register is defined for ARM CPU only.		

Offset: 24h **XV1UART24 (Slave): General Control Register B** **Init = 0bxxxx_xx11**

Bit	R/W	Description
31:8	RO	**Reserved (0)**
7 :4	RW	**MSI number selection bit [3:0]**
3 :2	RW	**Host-side timeout period selection** 00: 1/3600 second if XV1UART38[1]=0 01: 1/7200 second if XV1UART38[1]=0 10: 1/14400 second if XV1UART38[1]=0 11: 1/28800 second if XV1UART38[1]=0 00: 512*PCLK if XV1UART38[1]=1 01: 256*PCLK if XV1UART38[1]=1 10: 128*PCLK if XV1UART38[1]=1 11: 64*PCLK if XV1UART38[1]=1
1 :0	RO	**Number of bits per character (host-side)**
Note : This register is defined for ARM CPU only.		

Offset: 28h **XV1UART28 (Slave): XVirtual 1 UART Address Register L** **Init = 0bxxxx_x001**

Bit	R/W	Description
31:8	RO	**Reserved (0)**
7 :3	RW	**XVirtual 1 UART MMIO address bit [9:5]** This register defines the base offset address for host CPU to access PCIe virtual UART registers behind PCIe BMC Device BAR1.

to next page

from previous page

2 :0	RW	**Reserved (0)**
Note : This register is defined for ARM CPU only.		

Offset: 2Ch **XV1UART2C (Slave): XVirtual 1 UART Address Register H** **Init = X**

Bit	R/W	Description
31:6	RO	**Reserved (0)**
5 :0	RW	**XVirtual 1 UART MMIO address bit [15:10]** This register defines the base offset address for host CPU to access PCIe virtual UART registers behind PCIe BMC Device BAR1.
Note : This register is defined for ARM CPU only.		

Offset: 30h **XV1UART30 (Slave): General Control Register E** **Init = 0b0000_1110**

Bit	R/W	Description
31:8	RO	**Reserved (0)**
7	RO	**Transmit FIFO full. (slave-side)** 0: transmit FIFO not full. 1: transmit FIFO full.
6 :4	RO	**THR read pointer bit [2:0] (slave-side)**
3 :0	RO	**Complement of IIR status bit [3:0] (host-side)**
Note : This register is defined for ARM CPU only.		

Offset: 34h **XV1UART34 (Slave): General Control Register F** **Init = X**

Bit	R/W	Description
31:8	RO	**Reserved (0)**
7	RW	**Enable FIFO 1/2 full THRE Interrupt Mode (slave-side)** 0: Disable FIFO 1/2 full THRE interrupt mode 1: Enable FIFO 1/2 full THRE interrupt mode
6	RW	**Enable FIFO 1/2 full THRE Interrupt Mode Control (host-side)** 0: Disable FIFO 1/2 full THRE interrupt mode control 1: Enable FIFO 1/2 full THRE interrupt mode control
5	RW	**Trig a THRE interrupt on host side even through it has been empty already**
4 :3	RW	**Reserved**
2	RW	**Enable the Slave (BMC) to monitor DL and SCR on the Host side.**
1	RW	**Disable character time out interrupt (slave-side)**
0	RW	**Disable character time out interrupt (host-side)**
Note : This register is defined for ARM CPU only.		

Offset: 38h	XV1UART38 (Slave): General Control Register G	Init = X

Bit	R/W	Description
31:8	RO	**Reserved (0)**
7 :0	RO	**THR read back data bit [7:0] (slave-side) if XV1UART34[2]=0**
7 :2	RO	**Reserved if XV1UART34[2]=1**
1	RO	**XV1UART timeout time width control bit (slave-side) if XV1UART34[2]=1**
0	RO	**XV1UART timeout time width control bit (host-side) if XV1UART34[2]=1**
7 :2	W	**Reserved**
1	W	**XV1UART timeout time width control bit (slave-side)**
0	W	**XV1UART timeout time width control bit (host-side)**

Note :
This register is defined for ARM CPU only.

Offset: 3Ch	XV1UART3C (Slave): General Control Register H	Init = X

Bit	R/W	Description
31:8	RO	**Reserved (0)**
7 :6	RO	**Read back data bit [1:0] of receiver FIFO interrupt trigger level (slave-side)**
5	RO	**Interrupt Enable Register bit [7] on the Host side**
4	RO	**Enable UART FIFO on the Host side**
3 :0	RO	**Interrupt Enable Register bit [3:0] on the Host side**

Note :
This register is defined for ARM CPU only.

65 PCI Express Authentication Micro Controller Unit (GPMCU)

65.1 Overview

General Purpose Micro Control Unit

Base address of GPMCU = 0x1880_0000
Physical address = (Base address of GPMCU) + Offset

65.2 Registers : Base Address = 0x1880:0000

Offset: 10000h	GPMCU10000: GPMCU Register Bank	Init = 0x00000000
Bit	**R/W**	**Description**
31: 0	RO	**reg_00** register 00

Offset: 10004h	GPMCU10004: GPMCU Register Bank	Init = 0x00000001
Bit	**R/W**	**Description**
31: 0	RO	**reg_01** register 01

Offset: 10008h	GPMCU10008: GPMCU Register Bank	Init = 0x00000002
Bit	**R/W**	**Description**
31: 0	RO	**reg_02** register 02

Offset: 1000Ch	GPMCU1000C: GPMCU Register Bank	Init = 0x00000003
Bit	**R/W**	**Description**
31: 0	RO	**reg_03** register 03

Offset: 10010h	GPMCU10010: GPMCU Register Bank	Init = 0x00000004
Bit	**R/W**	**Description**
31: 0	RO	**reg_04** register 04

Offset: 10014h	GPMCU10014: GPMCU Register Bank	Init = 0x00000005
Bit	**R/W**	**Description**
31: 0	RO	**reg_05** register 05

Offset: 10018h	GPMCU10018: GPMCU Register Bank	Init = 0x00000006

Bit	R/W	Description
31: 0	RO	**reg_06** register 06

Offset: 1001Ch	GPMCU1001C: GPMCU Register Bank	Init = 0x00000007

Bit	R/W	Description
31: 0	RO	**reg_07** register 07

Offset: 10020h	GPMCU10020: GPMCU Register Bank	Init = 0x00000008

Bit	R/W	Description
31: 0	RO	**reg_08** register 08

Offset: 10024h	GPMCU10024: GPMCU Register Bank	Init = 0x00000009

Bit	R/W	Description
31: 0	RO	**reg_09** register 09

Offset: 10028h	GPMCU10028: GPMCU Register Bank	Init = 0x0000000A

Bit	R/W	Description
31: 0	RO	**reg_10** register 10

Offset: 1002Ch	GPMCU1002C: GPMCU Register Bank	Init = 0x0000000B

Bit	R/W	Description
31: 0	RO	**reg_11** register 11

Offset: 10030h	GPMCU10030: GPMCU Register Bank	Init = 0x0000000C

Bit	R/W	Description
31: 0	RO	**reg_12** register 12

Offset: 10034h	GPMCU10034: GPMCU Register Bank	Init = 0x0000000D

Bit	R/W	Description
31: 0	RO	**reg_13** register 13

Offset: 10038h	GPMCU10038: GPMCU Register Bank	Init = 0x0000000E

Bit	R/W	Description
31: 0	RO	**reg_14** register 14

Offset: 1003Ch	GPMCU1003C: GPMCU Register Bank	Init = 0x0000000F

Bit	R/W	Description
31: 0	RO	**reg_15** register 15

Offset: 10040h	GPMCU10040: GPMCU Register Bank	Init = 0x00000010

Bit	R/W	Description
31: 0	RO	**reg_16** register 16

Offset: 10044h	GPMCU10044: GPMCU Register Bank	Init = 0x00000011

Bit	R/W	Description
31: 0	RO	**reg_17** register 17

Offset: 10048h	GPMCU10048: GPMCU Register Bank	Init = 0x00000012

Bit	R/W	Description
31: 0	RO	**reg_18** register 18

Offset: 1004Ch	GPMCU1004C: GPMCU Register Bank	Init = 0x00000013

Bit	R/W	Description
31: 0	RO	**reg_19** register 19

Offset: 10050h	GPMCU10050: GPMCU Register Bank	Init = 0x00000014

Bit	R/W	Description
31: 0	RO	**reg_20** register 20

Offset: 10054h	GPMCU10054: GPMCU Register Bank	Init = 0x00000015

Bit	R/W	Description
31: 0	RO	**reg_21** register 21

Offset: 10058h	GPMCU10058: GPMCU Register Bank	Init = 0x00000016

Bit	R/W	Description
31: 0	RO	**reg_22** register 22

Offset: 1005Ch	GPMCU1005C: GPMCU Register Bank	Init = 0x00000017

Bit	R/W	Description
31: 0	RO	**reg_23** register 23

Offset: 10060h	GPMCU10060: GPMCU Register Bank	Init = 0x00000018

Bit	R/W	Description
31: 0	RO	**reg_24** register 24

Offset: 10064h	GPMCU10064: GPMCU Register Bank	Init = 0x00000019

Bit	R/W	Description
31: 0	RO	**reg_25** register 25

Offset: 10068h	GPMCU10068: GPMCU Register Bank	Init = 0x0000001A

Bit	R/W	Description
31: 0	RO	**reg_26** register 26

Offset: 1006Ch	GPMCU1006C: GPMCU Register Bank	Init = 0x0000001B

Bit	R/W	Description
31: 0	RO	**reg_27** register 27

Offset: 10070h	GPMCU10070: GPMCU Register Bank	Init = 0x00000000

Bit	R/W	Description
31: 0	RO	**reg_28** register 28

Offset: 10074h	GPMCU10074: GPMCU Register Bank	Init = 0x00000000

Bit	R/W	Description
31: 0	RO	**reg_29** register 29

Offset: 10078h	GPMCU10078: GPMCU Register Bank	Init = 0x00000000

Bit	R/W	Description
31: 0	RO	**reg_30** register 30

Offset: 1007Ch	GPMCU1007C: GPMCU Register Bank	Init = 0x00000000

Bit	R/W	Description
31: 0	RO	**reg_31** register 31

Offset: 10080h	GPMCU10080: GPMCU Pause Configuration Register	Init = 0x00000000

Bit	R/W	Description
31:17	RO	**Reserved**
16	WO	**pc_pause_step_pulse** write 1 to step by one instruction when pc_pause_en_force = 1 or pc_pause_en_bp = 1
15:13	RO	**Reserved**
12	RW	**pc_pause_en_force** 0: DP MCU runs in normal state 1: force DP MCU to pause
11: 9	RO	**Reserved**
8	RO	**pc_pause_en_wfi** this bit indicates if DP MCU is paused in WFI state (wait for interrupt)
7: 5	RO	**Reserved**
4	RO	**pc_pause_en_bp** this bit indicates if DP MCU is paused by the program counter reaches the break point (pc_curr == pc_bp) when pc_bp_en = 1
3: 1	RO	**Reserved**
0	RO	**pc_pause_en** this bit indicates if DP MCU is paused (ORed by pc_pause_en_force, pc_pause_en_wfi and pc_pause_en_bp)

Offset: 10084h	GPMCU10084: GPMCU Break Point Configuration Register	Init = 0x00000000

Bit	R/W	Description
31:21	RO	**Reserved**
20	RW	**pc_bp_en** 0: PC is NOT paused when it reaches pc_bp 1: PC is paused when it reaches pc_bp
19:17	RO	**Reserved**
16: 2	RW	**pc_bp_16_02** pc_bp[16:2] point for PC
1: 0	RO	**pc_bp_01_00** pc_bp[1:0] always 0

Offset: 10088h	GPMCU10088: GPMCU Next Program Counter Register	Init = 0x00000000
Bit	**R/W**	**Description**
31:17	RO	**pc_next_imem_base** 15-bit MSBs of instruction memory base byte address
16: 0	RO	**pc_next** next program counter

Offset: 1008Ch	GPMCU1008C: GPMCU Current Program Counter Register	Init = 0x00000000
Bit	**R/W**	**Description**
31:17	RO	**pc_curr_imem_base** 15-bit MSBs of instruction memory base byte address
16: 0	RO	**pc_curr** current program counter

Offset: 10090h	GPMCU10090: GPMCU Next Instruction Register	Init = 0x00000000
Bit	**R/W**	**Description**
31: 0	RO	**inst_next** instruction at next program counter

Offset: 10094h	GPMCU10094: GPMCU Current Instruction Register	Init = 0x2C000000
Bit	**R/W**	**Description**
31: 0	RO	**inst_curr** instruction at current program counter

Offset: 10098h	GPMCU10098: GPMCU Interrupt Enable Status Register	Init = 0xFFFFFFFF
Bit	**R/W**	**Description**
31: 0	RO	**int_en** interrupt enable bits controlled by instruction DP_MCU_INTEN

Offset: 1009Ch	GPMCU1009C: GPMCU Interrupt Enable Status Register	Init = 0x00000000
Bit	**R/W**	**Description**
31: 0	RO	**int_src_d2t** interrupt sources viewed by DP MCU

Offset: 100D0h	GPMCU100D0: GPMCU Version Register	Init = 0x20190710
Bit	**R/W**	**Description**
31: 0	RO	**version** DP MCU version

Offset: 100E0h	GPMCU100E0: GPMCU Configuration Register	Init = 0x10550010
Bit	**R/W**	**Description**
31:29	RO	**Reserved**

to next page

from previous page

28	RW	**config_en** write protect of imem_sel, imem_clk_off, imem_sleep, imem_shut_down, dmem_clk_off, dmem_sleep, dmem_shut_down, core_soft_resetn, ahbm_soft_resetn, ahbs_soft_resetn and ahbs_imem_en 0: NOT programmable 1: programmable
27:25	RO	**Reserved**
24	RW	**imem_sel** Reserved
23	RO	**Reserved**
22	RW	**imem_clk_off** 0: enable internal instruction memory clock 1: disable internal instruction memory clock
21	RW	**imem_sleep** 0: bring internal instruction memory out of sleep state 1: place internal instruction memory in sleep state
20	RW	**imem_shut_down** 0: bring internal instruction memory out of shut down state 1: place internal instruction memory in shut down state
19	RO	**Reserved**
18	RW	**dmem_clk_off** 0: enable internal data memory clock 1: disable internal data memory clock
17	RW	**dmem_sleep** 0: bring internal data memory out of sleep state 1: place internal data memory in sleep state
16	RW	**dmem_shut_down** 0: bring internal data memory out of shut down state 1: place internal data memory in shut down state
15:13	RO	**Reserved**
12	RW	**core_soft_resetn** 0: place DP MCU core in reset state 1: bring DP MCU core out of reset state
11: 9	RO	**Reserved**
8	RW	**ahbm_soft_resetn** 0: place DP MCU AHB master port in reset state 1: bring DP MCU AHB master port out of reset state
7: 5	RO	**Reserved**
4	RW	**ahbs_soft_resetn** 0: place DP MCU AHB slave port in reset state 1: bring DP MCU AHB slave port and the corresponding registers out of reset state
3: 1	RO	**Reserved**
0	RW	**ahbs_imem_en** 0: internal instruction memory is accessed by DP MCU only 1: internal instruction memory is accessed by external AHB masters

Part IV

PCI Express Interface

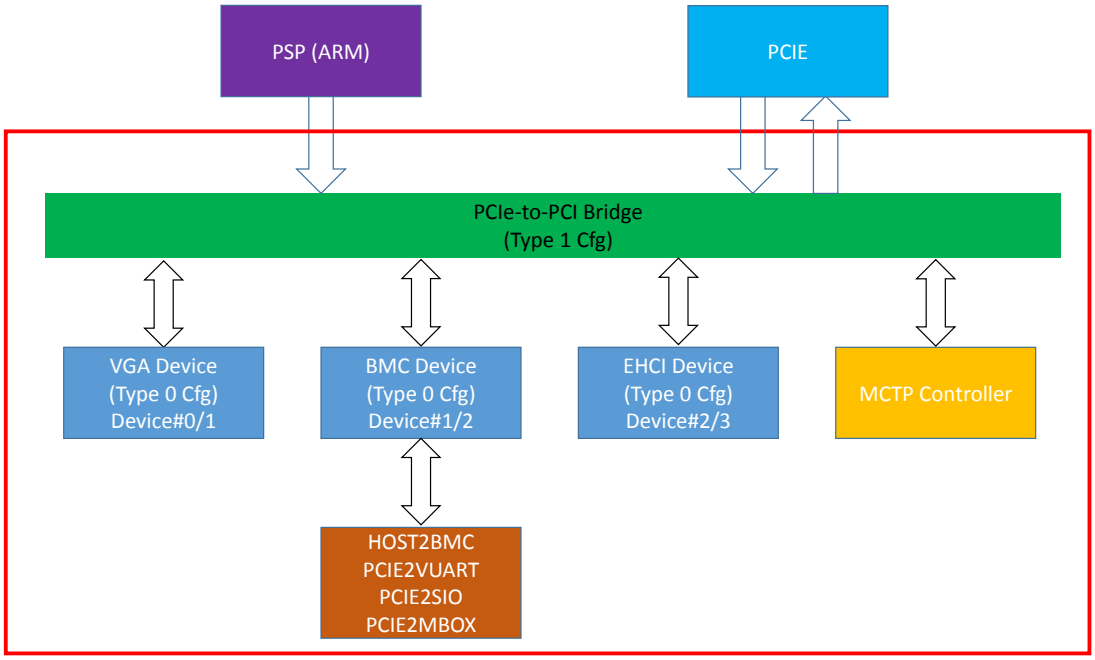


Figure 61: AST26x0 PCIe Architecture

66 PCI Express Controller (PCIE)

66.1 PCI Express Configuration Registers

Byte Offset	31:24	23:16	15:8	7:0
000h..03Fh	TYPE 1 CONFIGURATION REGISTERS			
040h..043h	SSID/SSVID PROTECTION			
044h..04Fh	Reserved			
050h..05Fh	MESSAGE SIGNALED INTERRUPTS			
060h..077h	Reserved			
078h..07Fh	POWER MANAGEMENT CAPABILITY STRUCTURE			
080h..0BBh	PCI EXPRESS			
0BCh..0BFh	Reserved			
0C0h..0C7h	SSID/SSVID CAPABILITY STRUCTURE			
0C8h..0FFh	Reserved			
100h..11Bh	VIRTUAL CHANNEL CAPABILITY STRUCTURE			
200h..22Fh	PCIe DVSEC (ID:03Eh)			
2C0h..2E3h	PCIe DVSEC (ID:02Eh)			
800h..84Bh	PCIe AER Capability			

66.2 Type 1 Configuration Registers (default values)

Byte Offset	31:24	23:16	15:8	7:0
000h	Device ID (1150h)		Vender ID (1A03h)	
004h	Status (0010h)		Command (0000h)	
008h	Class Code (060400h)			Revision ID (06h)
00Ch	Header Type (000100h)			Cache Line (00h)
010h	Base Address 0 (00000000h)			
014h	Base Address 1 (00000000h)			
018h	(00h)	Sub.Bus (00h)	Sec.Bus (00h)	Pri.Bus (00h)
01Ch	Secondary Status (0220h)		I/O Limit (01h)	I/O Base (01h)
020h	Memory Limit (0000h)		Memory Base (0000h)	
024h	Prefetchable Memory Limit (0001h)		Prefetchable Memory Base (0001h)	
028h	Prefetchable Base Upper 32-bit (00000000h)			
02Ch	Prefetchable Limit Upper 32-bit (00000000h)			
030h	I/O Limit Upper 16 Bits (0000h)		I/O Limit Lower 16 Bits (0000h)	
034h	(000000h)			Cap.PTR (50h)
038h	Expansion ROM Base Address (00000000h)			
03Ch	Bridge control (0000h)		Int.Pin (01h)	Int.Line (00h)

66.3 Message signaled interrupts (default values)

Byte Offset	31:24	23:16	15:8	7:0
050h	Message Control (008Ah)		Next Ptr (78h)	Cap.ID (05h)
054h	Message Address (00000000h)			
058h	Message Upper Address (00000000h)			
05Ch	Message Data (00000000h)			

66.4 Power Management Capability Structure (default values)

Byte Offset	31:24	23:16	15:8	7:0
078h	Capbility Register (FFC3h)		Next Ptr (80h)	Cap.ID (01h)
07Ch	Data (00h)	PMCSR_BSE (00h)	PMCSR (0008h)	

66.5 PCI Express Capability Structure (default values)

Byte Offset	31:24	23:16	15:8	7:0
080h	Capability Register (0072h)		Next Ptr (C0h)	Cap.ID (10h)
084h	Device Capability (00008022h)			
088h	Device Status (0010h)		Device Control (2810h)	
08Ch	Link Capability (00434C12h)			
090h	Link Status (1011h)		Link Control (0000h)	
094h	Slot Capability (00000000h)			
098h	Slot Status (0000h)		Slot Control (0000h)	
09Ch	Root Capability (0000h)		Root Control (0000h)	
0A0h	Root Status (00000000h)			
0A4h	Device Capability 2 (0000001Fh)			
0A8h	Device Status 2 (0000h)		Device Control 2 (0000h)	
0ACh	Link Capability 2 (00000000h)			
0B0h	Link Status 2 (0001h)		Link Control 2 (0002h)	
0B4h	Slot Capability 2 (00000000h)			
0B8h	Slot Status 2 (0000h)		Slot Control 2 (0000h)	

66.6 SSID/SSVID Capability Structure (default values)

Byte Offset	31:24	23:16	15:8	7:0
0C0h	Rsvd (0000h)		Next Ptr (00h)	Cap.ID (0Dh)
0C4h	SSID (1150h)		SSVID (1A03h)	

66.7 Virtual Channel Capability Structure (default values)

Byte Offset	31:24	23:16	15:8	7:0
100h	Next PTR (200h)	Vers. (1h)	Extended Cap.ID (0002h)	
104h	Rsvd (0000h)		Port VC Cap 1 (0000h)	
108h	VAT Offset (00h)	Rsvd (0000h)		VC Arb.Cap (00h)
10Ch	Port VC Status (0000h)		Port VC Control (0000h)	
110h	PAT Offset 0 (00h)	VC Resource Capability Register 0 (000000h)		
114h	VC Resource Control Register 0 (800000FFh)			
118h	VC Resource Status Register 0 (0000h)		Rsvd (0000h)	

66.8 PCIe DVSEC ID:03Eh (default values)

Byte Offset	31:24	23:16	15:8	7:0
200h	Next PTR (2C0h)	Vers. (1h)	Extended Cap.ID (0023h)	
204h	DVSEC Length (050h)	DVSEC Rev. (1h)	DVSEC Vendor ID (8086h)	
208h	VALID (00h)	MODIFIED (03h)	DVSEC ID (03Eh)	
20Ch	DIGEST_SEL (00h)	NUM_DIGEST (00h)	TCG_ALG_ID (000Dh)	
210h	DIGEST 0 (00000000h)			
214h	DIGEST 1 (00000000h)			
218h	DIGEST 2 (00000000h)			
21Ch	DIGEST 3 (00000000h)			
220h	DIGEST 4 (00000000h)			
224h	DIGEST 5 (00000000h)			
228h	DIGEST 6 (00000000h)			
22Ch	DIGEST 7 (00000000h)			
230h	DIGEST 8 (00000000h)			
234h	DIGEST 9 (00000000h)			
238h	DIGEST A (00000000h)			
23Ch	DIGEST B (00000000h)			
240h	DIGEST C (00000000h)			
244h	DIGEST D (00000000h)			
248h	DIGEST E (00000000h)			
24Ch	DIGEST F (00000000h)			

66.9 PCIe DVSEC ID:02Eh (default values)

Byte Offset	31:24	23:16	15:8	7:0
2C0h	DVSEC Header 1 (80010023h)			
2C4h	DVSEC Header 2 (02418086h)			
2C8h	DVSEC Header 3 (0000002Eh)			
2CCh	Authentication Header (00000100h)			
2D0h	Authentication Capabilities (01000900h)			
2D4h	Authentication Status (00000021h)			
2D8h	Authentication Control (00000000h)			
2DCh	Write Data Mailbox (00000000h)			
2E0h	Read Data Mailbox (00000000h)			

66.10 PCIe AER Capability (default values)

Byte Offset	31:24	23:16	15:8	7:0
800h	PCIe Enhanced Capability Header (00010001h)			
804h	Uncorrectable Error Status (00000000h)			
808h	Uncorrectable Error Mask (00000000h)			
80Ch	Uncorrectable Error Severity (00062031h)			
810h	Correctable Error Status (00000001h)			
814h	Correctable Error Mask (00002000h)			
818h	Advanced Error Capabilities and Control (00000000h)			
81Ch	Header Log 0 (00000000h)			
820h	Header Log 1 (00000000h)			
824h	Header Log 2 (00000000h)			
828h	Header Log 3 (00000000h)			
82Ch	Sec. Uncorrectable Error Status (00000000h)			
830h	Sec. Uncorrectable Error Mask (000017A8h)			
834h	Sec. Uncorrectable Error Severity (00001340h)			
838h	Sec. Error Capabilities and Control (00000000h)			
83Ch	Sec. Header Log 0 (00000000h)			
840h	Sec. Header Log 1 (00000000h)			
844h	Sec. Header Log 2 (00000000h)			
848h	Sec. Header Log 3 (00000000h)			

67 PCI Bus VGA Controller (P-Bus)

67.1 Overview

AST2600 integrated a bus controller designed to bridge the PCI-E bus to VGA Controller, which can directly communicate with VGA Controller, 2D Graphics Engine, SPI Host Controller, and P2A Bridge. It implements total 13 PCI Configuration registers, they are compatiable to PCI type 0 configuration registers settings, to control the various functions supported by AST2600 VGA.

PCIS00: Device and Vendor ID Register
PCIS04: Command and Status Register
PCIS08: Class and Revision ID Register
PCIS0C: Miscellaneous Register
PCIS10: Base Address 0 Register (for linear frame buffer)
PCIS14: Base Address 1 Register (for MMIO)
PCIS18: Base Address 2 Register (for relatable I/O)
PCIS2C: Subsystem ID Register
PCIS30: Expansion ROM Base Address Register
PCIS34: Capability Register
PCIS3C: Interrupt Register
PCIS40: PCI Power Management Capability Register
PCIS44: PCI Power Management Control and Status Register
PCIS50: Message Capability Register
PCIS54: Message Address Register
PCIS58: Message Upper Address Register
PCIS5C: Message Data Register

67.2 Features

– Support 32-bit 33 MHz PCI bus interface with PCI 2.3 specification compliant

Offset: 00h		**PCIS00: Device and Vendor ID Register**	**Init = 0x2000_1A03**
Bit	**R/W**	**Description**	
31:16	R	**Device ID** The default setting of this register is **0x2000**, which is the device ID code being assigned for AST2600 VGA. The device ID code of AST2600 VGA is the same as AST2000 VGA. This arrangement is to make sure that AST2600 VGA can directly run all the graphics display drivers developed for AST2000 VGA. The content of this register value can be changed by updating the corresponding register in SCU Controller, but its not recommended in normal cases.	
15:0	R	**Vendor ID** The default setting of this register is **0x1A03** which is the vendor ID code being assigned for **ASPEED Technology Inc**. by **PCISIG**. The content of this register value can be changed by updating the corresponding register in SCU Controller, but its not recommended in normal cases.	

Offset: 04h	PCIS04: Command and Status Register	Init = 0x0210_0000

Bit	R/W	Description
31	R	**Detected parity error** AST2600 VGA will not detect any parity errors; therefore, this bit will always return ”0”.
30	R	**Signaled system error** AST2600 VGA will not signal any system errors; therefore, this bit will always return ”0”.
29	R	**Received master abort** AST2600 VGA doesn’t play as a bus master; this register will always return ”0”.
28	R	**Received target abort** AST2600 VGA doesn’t play as a bus master; this register will always return ”0”.
27	R	**Signaled target abort** AST2600 VGA will not issue target abort; this register will always return ”0”.
26:25	R	**DEVSEL timing** AST2600 VGA supports medium timing for DEVSEL signal; this register will always return ”01”.
24	R	**Master data parity error** AST2600 VGA doesn’t play as a bus master; this register will always return ”0”.
23	R	**Fast back-to-back capable** AST2600 VGA doesn’t support fast back-to-back; this register will always return ”0”.
22		**Reserved (0)**
21	R	**66 MHz capable** AST2600 VGA supports 33MHz PCI bus running frequency; this register will always return ”0”.
20	R	**Capabilities list** AST2600 VGA supports a linked list to implement PCI bus power management; this register will always return ”1”.
19	R	**Interrupt status** This read-only bit reflects the state of the only interrupt source generated by CRT Controller for detecting the end of vertical display enable. This is a legacy interrupt from VGA Controller. In most of the cases, this interrupt source will not be enabled.
18:11		**Reserved (0)**
10	RW	**Interrupt disable** 0: Enable interrupt 1: Disable interrupt
9	R	**Fast back-to-back enable** AST2600 VGA doesn’t support fast back-to-back; this register will always return ”0”.
8	R	**SERR# enable** AST2600 VGA wont signal any system errors; this bit will always return ”0”.
7		**Reserved (0)**
6	RW	**Parity error response enable** AST2600 VGA wont detect any parity errors.
5	RW	**VGA palette snoop** AST2600 VGA only provides a read/write bit for this register. But it wont impact any hardware behavior.
4	R	**Memory write and invalidate enable** Since AST2600 VGA wont support this feature; this register will always return ”0”.
3	R	**Special cycles enable** Since AST2600 VGA doesn’t support PCI special cycles, this register will always return ”0”.

to next page

from previous page

2	RW	**Bus master enable** Since AST2600 VGA doesn't need PCI bus master cycles, this register should be programmed as "0".
1	RW	**Memory space access enable** 0: Disable memory space accesses 1: Enable memory space accesses This register will determine whether AST2600 VGA will response to memory space accesses or not.
0	RW	**IO space access enable** 0: Disable I/O space accesses 1: Enable I/O space accesses This register will determine whether AST2600 VGA will response to I/O space accesses or not.

Offset: 08h **PCIS08: Class and Revision ID Register** **Init = 0x0X00_0050**

Bit	R/W	Description
31:8	R	**Class code** When VGA Controller is enabled, AST2600 VGA will always return "0x030000" as the class code for this register to claim that AST2600 VGA is a tandard VGA device. When VGA is disabled by an external strapping resistor, AST2600 VGA will return "0x040000" as the class code of this register to claim that AST2600 VGA is a video device. As a video device, AST2600 VGA will not decode any VGA command cycles.
7 :0	R	**Revision ID** This register defines the revision ID of the current working silicon. It will change whenever a new revision is developed. The revision ID of AST2600 VGA is "50" for A0. The revision ID of AST2600 VGA is "51" for A1 and A2. The revision ID of AST2600 VGA is "52" for A3.

Offset: 0Ch **PCIS0C: Miscellaneous Register** **Init = 0x0000_0000**

Bit	R/W	Description
31	R	**BIST Capable** AST2600 VGA doesn't support BIST; this register will always return "0".
30	R	**Start BIST** AST2600 VGA doesn't support BIST; this register will always return "0".
29:28	R	**Start BIST** AST2600 VGA doesn't support BIST; this register will always return "0".
27:24	R	**Completion code** AST2600 VGA doesn't support BIST; this register will always return "0".
23:16	R	**Header type** This register will always return "0".
15:8	R	**Latency timer** AST2600 VGA doesn't play as a bus master; this register will always return "0".
7 :0	RW	**Cache line size** This register will always return "0".

Offset: 10h	PCIS10: Base Address 0 Register	Init = 0x0000_0000

Bit	R/W	Description
31:0	RW	**Base address 0 register** AST2600 VGA will claim a re-locatable memory space (8MB /16MB /32MB /64MB) for linear frame buffer allocation by this base address register. The size of linear frame buffer will depend on the two corresponding strapping resistors.

Offset: 14h	PCIS14: Base Address 1 Register	Init = 0x0000_0000

Bit	R/W	Description
31:0	RW	**Base address 1 register** AST2600 VGA will claim a 128KB re-locatable I/O memory space allocation by this base address register. The first 64KB is for VGA I/O addressing space, the second 64KB is for P2A Bridge addressing space.

Offset: 18h	PCIS18: Base Address 2 Register	Init = 0x0000_0001

Bit	R/W	Description
31:0	RW	**Base address 2 register** AST2600 VGA will claim a 128B re-locatable I/O space allocation by this base address register. This addressing space is used for VGA legacy and extended I/O cycles.

Offset: 2Ch	PCIS2C: Subsystem ID Register	Init = 0x2000_1A03

Bit	R/W	Description
31:16	RW	**Subsystem ID** This is a per-byte write once register. Once begin updated, this register cannot be modified again until next power-on. The default setting of this register is **0x2000**. Customer can modify this register if necessary.
15:0	RW	**Subsystem vendor ID** This is a per-byte write once register. Once begin updated, this register cannot be modified again until next power-on. The default setting of this register is **0x1A03**, which is following the vendor ID code of **ASPEED Technology Inc**. Customer can modify this register if necessary.

Offset: 30h	PCIS30: Expansion ROM Base Address Register	Init = 0x0000_0000

Bit	R/W	Description
31:0	RW	**Expansion ROM base address** AST2600 VGA will claim 64KB of memory space allocation for VGA BIOS. When VGA BIOS is merged with system BIOS, there will be no need to claim any ROM base address. Under such a condition, this base address claiming can be disabled by an external strapping resistor.

Offset: 34h	PCIS34: Capability Register	Init = 0x0000_0040

Bit	R/W	Description
31:8		**Reserved (0)**
7 :0	R	**Capabilities pointer** This optional register is used to point to a linked list of new capabilities. AST2600 VGA uses this register to point to 0x40 to implement PCI power management capability.

Offset: 3Ch	PCIS3C: Interrupt Register	Init = 0x0000_0100
Bit	**R/W**	**Description**
31:24	RW	**Maximum latency** This register is used for specifying how often the device needs to gain access to the PCI bus.
23:16	RW	**Minimum grant** This register is used for specifying how long a burst period the device needs assuming a clock rate of 33 MHz.
15:8	RW	**Interrupt pin** AST2600 VGA always returns 0x01 for this register to claim that the interrupt pin INTA# will be used by this device.
7 :0	RW	**Interrupt line** AST2600 VGA provides an 8-bit read/write register to implement this function. The content of this register will not impact any hardware behavior.

Offset: 40h	PCIS40: PCI Power Management Capability Register	Init = 0xffc3_5001
Bit	**R/W**	**Description**
31:27	R	**PME support** AST2600 VGA supports D0, D1, D2, D3hot and D3cold states. Therefore, this register will always return 0xF.
26	R	**D2 support** AST2600 supports D2 state; this register will always return ”1”.
25	R	**D1 support** AST2600 VGA supports D1 state; this register will always return ”1”.
24:22	R	**Auxiliary current requirement** This register will always return ”111b”. It means that AST2600 VGA requires 375mA from auxiliary current.
21	R	**Device specific initialization** AST2600 VGA doesn’t need any special initializations. This register will always return ”0”.
20		**Reserved (0)**
19	R	**PME Clock** AST2600 VGA doesn’t need to rely on PCI clock to generate PME#. This register will always return ”0”.
18:16	R	**Version** AST2600 VGA complies with PCI Power Management Revision 1.2. Therefore, This register will always return ”011b”.
15:8	R	**Next item pointer** This optional register is used to point to a linked list of new capabilities. AST2600 VGA uses this register to point to 0x50 to implement Message Signaled Interrupts.
7 :0	R	**ID** This register will always return ”0x01” to identify that the linked list item as being the PCI Power Management registers.

Offset: 44h	PCIS44: PCI Power Management Control and Status Register	Init = 0x0000_0000
Bit	**R/W**	**Description**
31:24	R	**Data register** This function is not implemented; this register always returns ”0x00”.
23	R	**Bus power and clock control enable** There is no secondary PCI bus; this register always returns ”0”.

to next page

from previous page

22	R	**B2/B3 support for D3hot** There is no secondary PCI bus; this register always returns ”0”.
21:16		**Reserved (0)**
15	RW	**PME Status** This bit is set when AST2600 VGA would normally assert the PME signal independent of the state of the PME enable bit. Writing ”1” to this register will cause AST2600 VGA to stop asserting the PME signal. Writing ”0” to this register has no effect.
14:13	R	**Data scale** AST2600 VGA doesn’t implement Data register; this register always returns ”00b”.
12:9	RW	**Data select** AST2600 VGA doesn’t implement Data register; this register always returns ”0000b”.
8	RW	**PME enable** 0: Disable PME assertion 1: Enable PME assertion
7 :4		**Reserved (0)**
3	R	**No soft reset** This register always returns ”0”.
2		**Reserved (0)**
1 :0	RW	**Power state** These two bits are used both to determine the current power state of AST2600 VGA and to set AST2600 VGA into a new power state. It will also impact CRT synchronization signals (HSYNC & VSYNC) and video DAC output. The definition of theses two bits is given below:

Bit[1:0]	State	VESA Mode	HSYNC	VSYNC	DAC
00	D0	Active Mode	On	On	On
01	D1	Standby Mode	Off	On	Off
10	D2	Suspend Mode	On	Off	Off
11	D3	OFF Mode	Off	Off	Off

Offset: 50h **PCIS50: Message Capability Register** **Init = 0x0082_0005**

Bit	R/W	Description
31:24		**Reserved (0)**
23	R	**64 bit address capable** This register will always return ”1b”. It means that AST2600 VGA is capable of generating a 64-bit message address.
22:20	RW	**Multiple Message Enable** System software writes to this field to indicate the number of allocated messages.
19:17	R	**Multiple Message Capable** This register will always return ”001b”. It means that AST2600 VGA requests 2 messages.
16	RW	**MSI Enable** 0: Disable MSI 1: Enable MSI
15:8	R	**Next item pointer** No next item pointer required. This register will always return ”0x00”.
7 :0	R	**ID** This register will always return ”0x05” to identify that the linked list item as being the Message Signaled Interrupts registers.

Offset: 54h	PCIS54: Message Address Register	Init = 0x0000_0000

Bit	R/W	Description
31:0	RW	**Message Address bit [31:0]** System-specified message address.

Offset: 58h	PCIS58: Message Upper Address Register	Init = 0x0000_0000

Bit	R/W	Description
31:0	RW	**Message Address bit [63:32]** System-specified message upper address.

Offset: 5Ch	PCIS5C: Message Data Register	Init = 0x0000_0000

Bit	R/W	Description
31:16		**Reserved (0)**
15:0	RW	**Message Data bit [15:0]** System-specified message.

68 VGA Display Controller (VGA)

68.1 Overview

VGA Display Controller (VGA) is one of the key modules integrated by AST2600 . The system bus interface adopted by VGA is 1-lane PCI Express Gen 2 bus interface, which can operate at 5GHz. VGA class code can be changed by an OTP strap register, OTPSTRAP[17] or SCU500[18].

When VGA class code is enabled, the class code of "VGA Device" will be claimed by PCI configuration registers.
When VGA class code is disabled, the class code of "Video Device" will be claimed instead.

VGA device can be disabled by setting register SCUC20[0] to 0.
When SCUC20[0] is 0, it disables VGA Device, i.e. PCI scan will not find the device 0 behind the bridge.

VGA is an in-band device which should be independent of ARM SOC system. Therefore, it can be reset only when either PCI bus reset or system power-on reset is asserted. VGA shares a portion of SDRAM memory for video frame buffer. The size of the shared frame buffer is determined by external strapping resistors. It will always occupy the highest portion of SDRAM memory. The initialization of SDRAM Controller is done by ARM SOC system. It should be finished well before host platform starting access video frame buffer.

VGA implements several groups of registers, which are listed below, to program the various supported functions. Each register has its own specific legacy address, and an offset value if available. AST2600 also provides memory-mapped I/O addressing mode for the need of advanced operating systems.

68.2 Features

- Fully IBM VGA compliant
- Maximum Display resolution: 1920x1200@60Hz with 165MHz video clock
- Integrate one deducted PLL for video clock generation which can be directly turned off by ARM CPU for power saving
- Support VESA DDC
- Support 64x64 hardware overlay cursor with mono and color formats
- RGB analog output
  - Integrate 200MHz triple DACs compliant with VESA monitor specification
  - Integrate 1.2V reference voltage generator
  - Need an external analog comparator for monitor sense
  - Support DAC power down function directly controlled by ARM CPU or host CPU

68.3 Registers

VGAER: VGA Enable Register

R/W:3C3 **Init = 00h**

Bit	Attr.	Description
7:1	RW	**Reserved (0)**
0	RW	**VGA enable** 0: Disable VGA 1: Enable VGA

VGAMR: Miscellaneous Output Register		
W:3C2 R:3CC		**Init = 00h**
Bit	**Attr.**	**Description**
7	RW	**Vertical sync polarity selection** 0: Select positive polarity 1: Select negative polarity
6	RW	**Horizontal sync polarity selection** 0: Select positive polarity 1: Select negative polarity
5	RW	**Page bit for odd/even modes** 0: Select lower page address 1: Select higher page address
4	RW	**Reserved**
3:2	RW	**Clock selection bit[1:0]** 00: Video clock frequency is 25.175MHz 01: Video clock frequency is 28.322MHz 1x: Video clock frequency is determined by the register programming for D-PLL
1	RW	**Enable video memory at VGA aperture** 0: Disable video memory address decoding 1: Enable video memory address decoding
0	RW	**I/O address selection** 0: Select 3Bx address decoding 1: Select 3Dx address decoding

VGAFCR: Feature Control Register		
W:3BA/3DA R:3CA		**Init = 00h**
Bit	**Attr.**	**Description**
7:4	RW	**Reserved (0)**
3	RW	**Feature control bit[2]**
2	RW	**Reserved**
1:0	RW	**Feature control bit[1:0]**

VGAIR0: Input Status Register #0		
R:3C2		**Init = 00h**
Bit	**Attr.**	**Description**
7	RW	**Vertical retrace interrupt flag**
6:5	RW	**Reserved (0)**
4	RW	**Video DAC comparator read back**
3:0	RW	**Reserved (0)**

VGAIR1: Input Status Register #1		
R:3BA/3DA		**Init = X1h**
Bit	**Attr.**	**Description**
7:6	RW	**Reserved (0)**

to next page

from previous page

5:4	RW	**Diagnostic bit[1:0]** 00: P2, P0 01: P5, P4 10: P3, P1 11: P7, P6 P7 ~ P0 are digital video output signals before RAMDAC controller.
3	RW	**Vertical retrace signal**
2:1	RW	**Reserved (0)**
0	RW	**Inversion of display enable signal** 0: During display enable period 1: Out of display enable period

VGAFBR0: Frame Buffer Segment Address Register #0

R/W: 3CD **Init = 00h**

Bit	Attr.	Description
7:4	RW	**Segment read address bit [3:0]**
3:0	RW	**Segment write address bit [3:0]**

VGAFBR1: Frame Buffer Segment Address Register #1

R/W: 3CB **Init = 00h**

Bit	Attr.	Description
7:4	RW	**Segment read address bit [7:4]**
3:0	RW	**Segment write address bit [7:4]**

68.4 Sequential Controller Registers

VGASRI: Sequential Controller Index Register

R/W:3C4 **Init = 00h**

Bit	Attr.	Description
7:6	RW	**Reserved (0)**
5:0	RW	**Index register bit[5:0]**

VGASR0: Reset Register

R/W:3C5 Index 00 **Init = 00h**

Bit	Attr.	Description
7:2	RW	**Reserved (0)**
1	RW	**Asynchronous reset (active low)** 0: Reset 1: No operation
0	RW	**Synchronous reset (active low)** 0: Reset 1: No operation

VGASR1: Clocking Mode Register		
R/W:3C5 Index:01		**Init = 00h**
Bit	**Attr.**	**Description**
7:6	RW	**Reserved**
5	RW	**Screen off** 0: Screen on 1: Screen off
4	RW	**Shift load by 4**
3	RW	**Divide video clock by 2**
2	RW	**Shift load by 2**
1	RW	**Reserved**
0	RW	**Select 8-dot period of character clock** 0: Select 9-dot character 1: Select 8-dot character

VGASR2: Map Mask Register		
R/W:3C5 Index:02		**Init = 00h**
Bit	**Attr.**	**Description**
7 :4	RW	**Reserved (0)**
3 :0	RW	**Enable memory write map [3:0]**

VGASR3: Character Map Selection Register		
R/W:3C5 Index:03		**Init = 00h**
Bit	**Attr.**	**Description**
7:6	RW	**Reserved (0)**
5	RW	**CG map A selection bit [2]**
4	RW	**CG map B selection bit [2]**
3:2	RW	**CG map A selection bit [1:0]**
1:0	RW	**CG map B selection bit [1:0]**

VGASR4: Character Map Selection Register		
R/W:3C5 Index:04		**Init = 00h**
Bit	**Attr.**	**Description**
7:4	RW	**Reserved (0)**
3	RW	**Enable Chain-4 mode**
2	RW	**Odd/even mode** 0: Odd/even mode 1: Sequential mode
1	RW	**Extended memory** 0: 64KB memory addressing mode 1: 256KB memory addressing mode
0	RW	**Reserved (0)**

68.5 CRT Controller Registers

VGACRI: CRT Controller Index Register		
R/W:3B4/3D4		**Init = 00h**
Bit	**Attr.**	**Description**
7:0	RW	**Index register bit [7:0]**

VGACR0: Horizontal Total Register		
R/W:3B5/3D5 Index:00		**Init = XXh**
Bit	**Attr.**	**Description**
7:0	RW	**Horizontal total bit[7:0] (-5)**

VGACR1: Horizontal Display Enable End Register		
R/W:3B5/3D5 Index:01		**Init = XXh**
Bit	**Attr.**	**Description**
7 :0	RW	**Horizontal display enable bit[7:0] (-1)**

VGACR2: Horizontal Blank Start Register		
R/W:3B5/3D5 Index:02		**Init = XXh**
Bit	**Attr.**	**Description**
7 :0	RW	**Horizontal blank start bit[7:0]**

VGACR3: Horizontal Blank End Register		
R/W:3B5/3D5 Index:03		**Init = XXh**
Bit	**Attr.**	**Description**
7	RW	**Enable register read back for registers indexed from 10-11**
6 :5	RW	**Horizontal display enable skew bit[1:0]**
4 :0	RW	**Horizontal blank end bit[4:0]**

VGACR4: Horizontal Retrace Start Register		
R/W:3B5/3D5 Index:04		**Init = XXh**
Bit	**Attr.**	**Description**
7 :0	RW	**Horizontal retrace start bit [7:0]**

VGACR5: Horizontal Retrace End Register		
R/W:3B5/3D5 Index:05		**Init = XXh**
Bit	**Attr.**	**Description**
7	RW	**Horizontal blank end bit [5]**
6 :5	RW	**Horizontal retrace delay bit [1:0]**
4 :0	RW	**Horizontal retrace end bit [4:0]**

VGACR6: Vertical Total Register		
R/W:3B5/3D5 Index:06		**Init = XXh**
Bit	**Attr.**	**Description**
7:0	RW	**Vertical total bit [7:0]**

VGACR7: Overflow Register		
R/W:3B5/3D5 Index:07		**Init = XXh**
Bit	**Attr.**	**Description**
7	RW	**Vertical retrace start bit [9]**
6	RW	**Vertical display enable end bit [9]**
5	RW	**Vertical total bit [9]**
4	RW	**Line compare bit [8]** This bit is out of the control by CRT register protection bit (Index 11, bit[7])
3	RW	**Vertical blank start bit [8]**
2	RW	**Vertical retrace start bit [8]**
1	RW	**Vertical display enable end bit [8]**
0	RW	**Vertical total bit [8]**

VGACR8: Preset Row Scan Register		
R/W:3B5/3D5 Index:08		**Init = XXh**
Bit	**Attr.**	**Description**
7	RW	**Reserved**
6:5	RW	**Byte panning bit[1:0]**
4:0	RW	**Preset row scan bit[4:0]**

VGACR9: Maximum Scan Line Register		
R/W:3B5/3D5 Index:09		**Init = XXh**
Bit	**Attr.**	**Description**
7	RW	**Enable double scan** Convert 200 scan lines to 400 scan lines
6	RW	**Line compare bit [9]**
5	RW	**Vertical blank bit [9]**
4:0	RW	**Maximum row scan bit [4:0]**

VGACRA: Cursor Start Register		
R/W:3B5/3D5 Index:0A		**Init = XXh**
Bit	**Attr.**	**Description**
7:6	RW	**Reserved (0)**
5	RW	**Cursor off**
4:0	RW	**Cursor start bit [4:0]**

VGACRB: Cursor End Register		
R/W:3B5/3D5 Index:0B		**Init = XXh**
Bit	**Attr.**	**Description**
7	RW	**Reserved (0)**
6:5	RW	**Cursor skew bit [1:0]**
4:0	RW	**Cursor end bit [4:0]**

VGACRC: Starting Address High Register		
R/W:3B5/3D5 Index:0C		**Init = XXh**
Bit	**Attr.**	**Description**
7:0	RW	**Starting address bit[15:8]**

VGACRD: Starting Address Low Register		
R/W:3B5/3D5 Index:0D		**Init = XXh**
Bit	**Attr.**	**Description**
7:0	RW	**Starting address bit[7:0]**

VGACRE: Cursor Location High Register		
R/W:3B5/3D5 Index:0E		**Init = XXh**
Bit	**Attr.**	**Description**
7:0	RW	**Cursor location bit[15:8]**

VGACRF: Cursor Location Low Register		
R/W:3B5/3D5 Index:0F		**Init = XXh**
Bit	**Attr.**	**Description**
7:0	RW	**Cursor location bit[7:0]**

VGACR10: Vertical Retrace Start Register		
R/W:3B5/3D5 Index:10		**Init = XXh**
Bit	**Attr.**	**Description**
7:0	RW	**Vertical retrace start bit[7:0]**

VGACR11: Vertical Retrace End Register		
R/W:3B5/3D5 Index:11		**Init = 00h**
Bit	**Attr.**	**Description**
7	RW	**Protect CRT registers from index 00 to index 07** Index 07[4] is the only exception
6	RW	**Reserved** This bit is for register read/write only
5	RW	**Disable vertical interrupt**
4	RW	**Clear vertical interrupt flag** 0: Clear vertical interrupt flag 1: No operation

to next page

from previous page

3:0	RW	**Vertical retrace end bit[3:0]**

VGACR12: Vertical Display Enable End Register

R/W:3B5/3D5 Index:12 **Init = XXh**

Bit	Attr.	Description
7:0	RW	**Vertical display enable end bit [7:0]**

VGACR13: Offset Register

R/W:3B5/3D5 Index:13 **Init = 00h**

Bit	Attr.	Description
7:0	RW	**Offset bit [7:0]**

VGACR14: Underline Location Register

R/W:3B5/3D5 Index:14 **Init = XXh**

Bit	Attr.	Description
7	RW	**Reserved (0)**
6	RW	**Select double word mode**
5	RW	**Select count-by-4 mode** This function is not implemented
4:0	RW	**Underline location bit [4:0]**

VGACR15: Vertical Blank Start Register

R/W:3B5/3D5 Index:15 **Init = XXh**

Bit	Attr.	Description
7:0	RW	**Vertical blank start [7:0]**

VGACR16: Vertical Blank End Register

R/W:3B5/3D5 Index:16 **Init = XXh**

Bit	Attr.	Description
7:0	RW	**Vertical blank end [7:0]**

VGACR17: Mode Control Register

R/W:3B5/3D5 Index:17 **Init = 00h**

Bit	Attr.	Description
7	RW	**Hardware reset (active low)**
6	RW	**Select byte mode**
5	RW	**Address wrap enable** This function is not implemented
4	RW	**Reserved (0)**
3	RW	**Select count-by-2 mode** This function is not implemented

to next page

from previous page

2	RW	**Horizontal retrace selection** This function is not implemented
1	RW	**Replace MA14 by RA1 (active low)**
0	RW	**Replace MA13 by RA0 (active low)**

VGACR18: Line Compare Register

R/W:3B5/3D5 Index:18 **Init = XXh**

Bit	Attr.	Description
7:0	RW	**Line compare bit [7:0]**

VGACR1E: Graphics Latched Data 0 Register

R:3B5/3D5 Index:1E **Init = XXh**

Bit	Attr.	Description
7:2	R	**Reserved**
1	R	**Attribute controller register index toggle bit**
0	R	**Reserved**

VGACR1F: Graphics Latched Data 1 Register

R:3B5/3D5 Index:1F **Init = 00h**

Bit	Attr.	Description
7:6	R	**Reserved**
5:0	R	**Attribute controller register index bit [5:0]**

VGACR22: Graphics Latched Data 2 Register

R:3B5/3D5 Index:22 **Init = XXh**

Bit	Attr.	Description
7:0	R	**Graphics latched data bit[7:0]**

68.6 Graphics Controller Registers

VGAGRI: Graphics Controller Index Register

R/W:3CE **Init = 00h**

Bit	Attr.	Description
7:4	RW	**Reserved (0)**
3:0	RW	**Index register bit [3:0]**

VGAGR0: Set/Reset Map Register

R/W:3CF Index:00 **Init = 00h**

Bit	Attr.	Description
7:4	RW	**Reserved (0)**
3:0	RW	**Set/reset map bit [3:0]**

VGAGR1: Enable Set/Reset Map Register		
R/W:3CF Index:01		**Init = 00h**
Bit	**Attr.**	**Description**
7:4	RW	**Reserved (0)**
3:0	RW	**Enable set/reset map bit [3:0]**

VGAGR2: Color Compare Register		
R/W:3CF Index:02		**Init = 00h**
Bit	**Attr.**	**Description**
7:4	RW	**Reserved (0)**
3:0	RW	**Color compare map bit [3:0]**

VGAGR3: Data Rotate Register		
R/W:3CF Index:03		**Init = 00h**
Bit	**Attr.**	**Description**
7:5	RW	**Reserved (0)**
4:3	RW	**Function selection bit [1:0]**
2:0	RW	**Data rotate bit [3:0]**

VGAGR4: Read Map Selection Register		
R/W:3CF Index:04		**Init = 00h**
Bit	**Attr.**	**Description**
7:2	RW	**Reserved (0)**
1:0	RW	**Read map selection bit [1:0]**

VGAGR5: Mode Register		
R/W:3CF Index:05		**Init = 00h**
Bit	**Attr.**	**Description**
7	RW	**Reserved (0)**
6	RW	**Enable shift mode for graphics display mode 13**
5	RW	**Enable shift mode for graphics display mode 4 and mode 5**
4	RW	**Enable odd/even mode**
3	RW	**Read mode selection** 0: Select normal read mode 1: Select color compare read mode
2	RW	**Reserved (0)**
1:0	RW	**Write mode selection bit [1:0]**

VGAGR6: Miscellaneous Register		
R/W:3CF Index:06		**Init = 00h**
Bit	**Attr.**	**Description**
7:4	RW	**Reserved (0)**

to next page

from previous page

3:2	RW	**Memory addressing space selection bit [1:0]** 00: A000H/128KB 01: A000H/64KB 10: B000H/32KB 11: B800H/32KB
1	RW	**Chain odd/even plan enable**
0	RW	**Select graphics mode** 0: text mode 1: graphics mode

VGAGR7: Color Don't Care Register

R/W:3CF Index:07 **Init = 00h**

Bit	Attr.	Description
7:4	RW	**Reserved (0)**
3:0	RW	**Color don't care bit[3:0]**

VGAGR8: Bit Mask Register

R/W:3CF Index:08 **Init = 00h**

Bit	Attr.	Description
7:0	RW	**Bit mask bit [7:0]**

68.7 Attribute Controller Registers

VGAARI: Attribute Controller Index Register

R:3C1 W:3C0 **Init = 00h**

Bit	Attr.	Description
7:6	RW	**Reserved (0)**
5	RW	**Pallet address source selection** 0: Address source is from register read/write address issued by CPU 1: Address source is from graphics streaming data
4:0	RW	**Index register bit [4:0]**

VGAAR0–VGAARF: Pallet Register 00 ~ 0F

R:3C1 W:3C0 Index:00–0F **Init = XXh**

Bit	Attr.	Description
7:6	RW	**Reserved (0)**
5:0	RW	**Pallet data bit [5:0]** There are total 16 sets of 5-bit palette registers. Their index number is from 00h to 0Fh. The address source of the pallet is determined by Attribute Controller Index Register bit [5].

VGAAR10: Mode Control Register		
R:3C1 W:3C0 Index:10		**Init = 00h**
Bit	**Attr.**	**Description**
7	RW	**Internal palette size selection** 0: Select 6 bits per pixel 1: Select 4 bits per pixel (cascaded with Index 14[1:0])
6	RW	**Pixel width selection** For Mode 13 only
5	RW	**Pixel panning compatibility** 0: Pixel panning will be applied to both screens (before and after line compare) 1: Pixel panning will only be applied to screen before line compare
4	RW	**Reserved (0)**
3	RW	**Enable blink mode**
2	RW	**Enable line graphics extension for ASCII codes from 0xC0 to 0xDF**
1	RW	**Select monochrome display mode**
0	RW	**Select graphics mode** 0: Select text mode 1: Select graphics mode

VGAAR11: Boarder Color Register		
R:3C1 W:3C0 Index:11		**Init = XXh**
Bit	**Attr.**	**Description**
7:0	RW	**Boarder color bit [7:0]**

VGAAR12: Color Plan Enable Register		
R:3C1 W:3C0 Index:12		**Init = XXh**
Bit	**Attr.**	**Description**
7:6	RW	**Reserved (0)**
5:4	RW	**Video status multiplexing bit [1:0]**
3:0	RW	**Color plan enable bit [3:0]**

VGAAR13: Horizontal Pixel Panning Register		
R:3C1 W:3C0 Index:13		**Init = 0Xh**
Bit	**Attr.**	**Description**
7:4	RW	**Reserved (0)**
3:0	RW	**Horizontal pixel panning bit [3:0]**

VGAAR14: Color Select Register		
R:3C1 W:3C0 Index:14		**Init = 0Xh**
Bit	**Attr.**	**Description**
7:4	RW	**Reserved (0)**
3:0	RW	**Color selection bit [3:0]**

68.8 RAMDAC Registers

VGAPMR: RAMDAC Pixel Mask Register		
R/W:3C6		**Init = FFh**
Bit	**Attr.**	**Description**
7:0	W	**Pixel mask bit [7:0]**

VGADSR: RAMDAC Status Register		
R:3C7		**Init = 00h**
Bit	**Attr.**	**Description**
7:2	R	**Reserved (0)**
1:0	R	**Status bit [1:0]**

VGADRR: RAMDAC Read Mode Address Register		
W:3C7		**Init = 00h**
Bit	**Attr.**	**Description**
7:0	W	**Read mode address bit [7:0]**

VGADWR: RAMDAC Write Mode Address Register		
R/W:3C8		**Init = 00h**
Bit	**Attr.**	**Description**
7:0	RW	**Write mode address bit [7:0]**

VGAPDR: RAMDAC Pallet Data Register		
R/W:3C9		**Init = 00h**
Bit	**Attr.**	**Description**
7:0	RW	**Pallet data bit [7:0]**

68.9 Extended CRT Registers

Index Range	3	2	1	0
Index 83 - 80	VGA Scratch Register			Password
Index 87 - 84	VGA Scratch Register			
Index 8B - 88	VGA Scratch Register			
Index 8F - 8C	VGA Scratch Register			
Index 93 - 90	VGA Scratch Register			
Index 97 - 94	VGA Scratch Register			
Index 9B - 98	VGA Scratch Register			
Index 9F - 9C	VGA Scratch Register			
Index A3 - A0	Color Mode	PCI Bus Control		
Index A7 - A4	CRT Threshold		Segment Adr	Misc Control
Index AB - A8	Power-On Strapping		RAMDAC Control	
Index AF - AC	Starting Overflow	Vertical Overflow	Horizontal Overflow	
Index B3 - B0	CRT Counter Read Back			Offset Overflow
Index B7 - B4	DDC Control	Power Control	Reserved (0)	
Index BB - B8	PLL Overflow	RGB CRC Signature Read Back		
Index BF - BC	28MHz PLL		25MHz PLL	
Index C3 - C0	Hardware Cursor Offset		Video PLL	
Index C7 - C4	Hardware Cursor Y Position		Hardware Cursor X Position	
Index CB - C8	Cursor Mode	Hardware Cursor Pattern Address		
Index CF - CC	Reserved			
Index D3 - D0	SOC Scratch Register Read Back			
Index D7 - D4	SOC Scratch Register Read Back			
Index DC - D8	SOC Scratch Register Read Back			
Index DF - DD	SOC Scratch Register Read Back			
Index E3 - E0	VGA Scratch Register			
Index E7 - E4	VGA Scratch Register			
Index EB - E8	VGA Scratch Register			
Index EF - EC	VGA Scratch Register			
Index F3 - F0	VGA Scratch Register			
Index F7 - F4	VGA Scratch Register			
Index FB - F8	VGA Scratch Register			
Index FF - FC	PLL Control Register			

VGACR80: Password Register

R/W:3B5/3D5 Index:80 MMIO:Base+80 **Init = 00h**

Bit	Attr.	Description
7:0	RW	**Password bit [7:0]** Password: A8h

VGACR81–9E: Scratch Register #1 ~ #30		
R/W:3B5/3D5 Index:81–9E MMIO:Base+81–9E		**Init = XXh**
Bit	**Attr.**	**Description**
7:0	RW	**Scratch register bit [7:0]** Only for the usage of VGA BIOS and Display Drivers

VGACR9F: Scratch Register #31		
R/W:3B5/3D5 Index:9F MMIO:Base+9F		**Init = XXh**
Bit	**Attr.**	**Description**
7:6	R	**Indicate the PCI power state D0 ~ D3** Map to PCIS44 bit[1:0]
5	R	**Indicate the Pallet address source selection** Map to VGAARI bit[5]
4	R	**Indicate the Pixel Mask Status** Map to the Logical-OR result of VGAPMR bit[7:0]
3	R	**Indicate the Reset Status of VGA** Map to VGACR17 bit[7]
2	R	**Indicate the Status of Screen Display** Map to VGASR1 bit[5]
1	R	**Indicate the Reset Status of VGA Controller** Map to the Logical-AND result of VGASR0 bit[0] and bit[1]
0	R	**VGA Enable Status Register** Map to VGAER:3C3 bit[0]

VGACRA0: PCI Control Register #1		
R/W:3B5/3D5 Index:A0 MMIO:Base+A0		**Init = 00h**
Bit	**Attr.**	**Description**
7	RW	**Reserved**
6	RW	**Enable video memory access by 32-bit china-4 mode** This bit is for graphics mode only
5	RW	**Enable linear extended memory access (> 256KB)**
4	RW	**Enable extended segmented memory address (> 256KB)**
3	RW	**Enable burst memory read**
2	RW	**Enable burst memory write**
1	RW	**Enable read-ahead cache**
0	RW	**Enable post-write buffer**

VGACRA1: PCI Control Register #2		
R/W:3B5/3D5 Index:A1 MMIO:Base+A1		**Init = 04h**
Bit	**Attr.**	**Description**
7:4	RW	**Reserved**
3	RW	**Disable re-locatable I/O-mapped VGA I/O address decoding**
2	RW	**Enable re-locatable memory-mapped VGA I/O address decoding**
1	RW	**Disable standard VGA I/O address decoding**
0	RW	**Disable standard VGA memory address (0xA000~0xBFFF) decoding**

VGACRA2: PCI Control Register #3		
R/W:3B5/3D5 Index:A2 MMIO:Base+A2		**Init = 00h**
Bit	**Attr.**	**Description**
7	RW	**Enable big-endian mode**
6	RW	**Enable 16-bit big-endian mode** 0: 32-bit 1: 16-bit
5	RW	**Reserved**
4	RW	**Enable PCI retry for I/O cycles while memory post-write buffer is not empty**
3	RW	**Enable PCI retry for memory write cycles**
2	RW	**Enable PCI retry for memory read cycles**
1	RW	**Enable E2M linear read mode**
0	RW	**Enable E2M linear write mode**

VGACRA3: Enhanced Color Mode Register		
R/W:3B5/3D5 Index:A3 MMIO:Base+A3		**Init = 00h**
Bit	**Attr.**	**Description**
7:4	RW	**Reserved**
3	RW	**Enable 32-BPP true color display mode (ARGB:8888)**
2	RW	**Enable 16-BPP high color display mode (RGB:565)**
1	RW	**Enable 15-BPP high color display mode (RGB:555)**
0	RW	**Enable enhanced 256 color display mode**

VGACRA4: Misc. Control Register		
R/W:3B5/3D5 Index:A4 MMIO:Base+A4		**Init = 00h**
Bit	**Attr.**	**Description**
7	RW	**Software reset 2D engine**
6	RW	**Trigger bit of VGA interrupt to BMC**
5	RW	**Enable Sub-System ID and Sub-Vendor ID write cycles**
4	RW	**Enable VGA BIOS flash write**
3:2	RW	**2D Engine clock source selection** 00: MCLK 01: ~MCLK (inverted clock phase) 10: Reserved 11: Reserved
1	RW	**Enable clock throttling for 2D Engine** When 2D Engine is in idle state, its clock will be automatically slowed down to 1/16 for power saving. When receiving a new command, 2D Engine will speed up at full speed automatically.
0	RW	**Enable 2D Engine** 0: Reset 1: Enable

VGACRA5: Segmented Memory Address Overflow Register		
R/W:3B5/3D5 Index:A5 MMIO:Base+A5		**Init = 00h**
Bit	**Attr.**	**Description**
7:6	RW	**Reserved**
5:4	RW	**Segmented memory read address bit [9:8]**
3:2	RW	**Reserved**
1:0	RW	**Segmented memory write address bit [9:8]**

VGACRA6: CRT Request Threshold Low Register		
R/W:3B5/3D5 Index:A6 MMIO:Base+A6		**Init = 00h**
Bit	**Attr.**	**Description**
7:6	RW	**Reserved**
5:0	RW	**CRT request threshold low bit [5:0]**

VGACRA7: CRT Request Threshold High Register		
R/W:3B5/3D5 Index:A7 MMIO:Base+A7		**Init = 00h**
Bit	**Attr.**	**Description**
7:6	RW	**Reserved**
5:0	RW	**CRT memory request threshold high bit [5:0]**

VGACRA8: RAMDAC Control Register		
R/W:3B5/3D5 Index:A8 MMIO:Base+A8		**Init = 00h**
Bit	**Attr.**	**Description**
7	RW	**en_1v_sync**
6	RW	**Enable RAMDAC test mode for monitor sense application**
5	RW	**Reserved**
4	RW	**Disable RAMDAC mask function**
3	RW	**Reserved**
2	RW	**Protect palette/gamma RAM from write cycles**
1	RW	**Enable 24-bit gamma correction RAM**
0	RW	**Reserved**

VGACRA9: RAMDAC Test Pattern Register		
R/W:3B5/3D5 Index:A9 MMIO:Base+A9		**Init = 00h**
Bit	**Attr.**	**Description**
7:0	RW	**RAMDAC test pattern bit[7:0]**

VGACRAA: Power-On Strapping Status Register #1		
R/W:3B5/3D5 Index:AA MMIO:Base+AA		**Init = X**
Bit	**Attr.**	**Description**
7	R	**CPU clock frequency selection bit[0]**
6	R	**Reserved, always '0'**

to next page

from previous page

5:4	R	**ARM CPU boot code selection** 00: N/A 01: N/A 10: Boot from SPI flash memory 11: Disable ARM CPU operation
3	R	**Enable VGA BIOS ROM** 0: Disable VGA BIOS ROM 1: Enable VGA BIOS ROM
2	R	**DAC display output source** 0: VGA 1: Non VGA
1:0	R	**Total VGA memory size setting** 00: 8MB 01: 16MB 10: 32MB 11: 64MB

VGACRAB: Power-On Strapping Status Register #2

R/W:3B5/3D5 Index:AB MMIO:Base+AB **Init = 00h**

Bit	Attr.	Description
7:6	R	**Chip ID** Reserved
5	R	**Reserved, always '0'**
4	R	**PCI class code selection** 0: A video device is claimed in PCI class code register 1: A VGA device is claimed in PCI class code register
3	R	**PCI VGA Config Prefetch status** 0: Prefetch bit = 0 1: prefetch bit = 1
2:1	R	**Reserved, always '0'**
0	R	**CPU clock frequency selection bit[1]** 00: Select 384 MHz 01: Select 360 MHz 10: Select 336 MHz 11: Select 408 MHz

VGACRAC: Extended Horizontal Overflow Register #1

R/W:3B5/3D5 Index:AC MMIO:Base+AC **Init = 00h**

Bit	Attr.	Description
7:6	RW	**Horizontal retrace start bit [9:8]**
5:4	RW	**Horizontal blank start bit [9:8]**
3:2	RW	**Horizontal display enable end bit [9:8]**
1:0	RW	**Horizontal total bit [9:8]**

VGACRAD: Extended Horizontal Overflow Register #2		
R/W:3B5/3D5 Index:AD MMIO:Base+AD		**Init = 00h**
Bit	**Attr.**	**Description**
7	RW	**Reserved**
6:4	RW	**Horizontal retrace skew bit [2:0]**
3:2	RW	**Horizontal retrace end bit [6:5]**
1:0	RW	**Horizontal blank end bit [7:6]**

VGACRAE: Extended Vertical Overflow Register		
R/W:3B5/3D5 Index:AE MMIO:Base+AE		**Init = 00h**
Bit	**Attr.**	**Description**
7	RW	**Disable line compare**
6:5	RW	**Vertical retrace end bit [5:4]**
4	RW	**Vertical blank end bit [8]**
3	RW	**Vertical retrace start bit [10]**
2	RW	**Vertical blank start bit [10]**
1	RW	**Vertical display enable end bit [10]**
0	RW	**Vertical total bit [10]**

VGACRAF: Extended CRT Starting Address Register		
R/W:3B5/3D5 Index:AF MMIO:Base+AF		**Init = 00h**
Bit	**Attr.**	**Description**
7:0	RW	**CRT starting address bit [23:16]**

VGACRB0: Extended CRT Offset Register		
R/W:3B5/3D5 Index:B0 MMIO:Base+B0		**Init = 00h**
Bit	**Attr.**	**Description**
7:6	RW	**Reserved**
5:0	RW	**Offset bit[13:8]**

VGACRB1: Horizontal Counter read Back Register		
R:3B5/3D5 Index:B1 MMIO:Base+B1		**Init = 00h**
Bit	**Attr.**	**Description**
7:0	RW	**Horizontal counter read back bit[7:0]**

VGACRB2: Vertical Counter read Back Register		
R:3B5/3D5 Index:B2 MMIO:Base+B2		**Init = 00h**
Bit	**Attr.**	**Description**
7:0	RW	**Vertical counter read back bit[7:0]**

VGACRB3: CRT Counter read Back Overflow Register		
R/W:3B5/3D5 Index:B3 MMIO:Base+B3		**Init = 00h**
Bit	**Attr.**	**Description**
7	RW	**Reserved**
6	RW	**Reserved (0)**
5:4	RW	**Horizontal counter read back bit[9:8]**
3:0	RW	**Vertical counter read back bit[11:8]**

VGACRB4: Extended Vertical Overflow 2 Register		
R/W:3B5/3D5 Index:B4 MMIO:Base+B4		**Init = 00h**
Bit	**Attr.**	**Description**
7:6	RW	**Reserved (0)**
5	RW	**Vertical retrace end bit [6]**
4	RW	**Vertical blank end bit [9]**
3	RW	**Vertical retrace start bit [11]**
2	RW	**Vertical blank start bit [11]**
1	RW	**Vertical display enable end bit [11]**
0	RW	**Vertical total bit [11]**

VGACRB5: VGA B5 Register		
R:3B5/3D5 Index:B5 MMIO:Base+B5		**Init = 00h**
Bit	**Attr.**	**Description**
7:0	RW	**Reserved**

VGACRB6: Power Management Register		
R/W:3B5/3D5 Index:B6 MMIO:Base+B6		**Init = 00h**
Bit	**Attr.**	**Description**
7:6	RW	**Reserved**
5	RW	**Reserved**
4	RW	**Enable bypass mode for video PLL**
3	RW	**Power down video PLL**
2	RW	**Power on RAMDAC** 0: RAMDAC is power down 1: RAMDAC is power on
1	RW	**Enable VSYNC off**
0	RW	**Enable HSYNC off**

VGACRB7: DDC Control Register		
R/W:3B5/3D5 Index:B7 MMIO:Base+B7		**Init = 00h**
Bit	**Attr.**	**Description**
7	RW	**Status of CRC signature generation** 0: Invalid (still in progress or never triggered) 1: Valid (finished)
6	RW	**Trig CRC signature generation** 0: No operation 1: Trig CRC signature generation CRC signature generation will take at least one frame of cycle time to finish the task. S/W needs to poll the status of CRC signature generation before reading back RGB signature data.
5	RW	**DDC data input**
4	RW	**DDC clock input**
3	RW	**DDC data output**
2	RW	**Enable DDC data output buffer**
1	RW	**DDC clock output**
0	RW	**Enable DDC clock output buffer**

VGACRB8: Blue CRC Signature Read Back Register		
R/W:3B5/3D5 Index:B8 MMIO:Base+B8		**Init = FCh**
Bit	**Attr.**	**Description**
7	RW	**Blue CRC signature read back bit [7:0]**

VGACRB9: Green CRC Signature Read Back Register		
R/W:3B5/3D5 Index:B9 MMIO:Base+B9		**Init = FCh**
Bit	**Attr.**	**Description**
7	RW	**Green CRC signature read back bit [7:0]**

VGACRBA: Red CRC Signature Read Back Register		
R/W:3B5/3D5 Index:BA MMIO:Base+BA		**Init = FCh**
Bit	**Attr.**	**Description**
7	RW	**Red CRC signature read back bit [7:0]**

VGACRBB: PLL Overflow Register		
R/W:3B5/3D5 Index:BB MMIO:Base+BB		**Init = 1Fh**
Bit	**Attr.**	**Description**
7	RW	**Reserved**
6	RW	**Reserved**
5:4	RW	**Video PLL extended post divider** 00: 1/1 01: 1/2 10: 1/2 11: 1/4

to next page

from previous page

3:2	RW	**28.322MHz PLL extended post divider bit [1:0]** 00: 1/1 01: 1/2 10: 1/2 11: 1/4 The content of this register will only determine the clock frequency of D-PLL when 28.322MHz is selected by legacy register.
1:0	RW	**25.175MHz PLL extended post divider bit [1:0]** 00: 1/1 01: 1/2 10: 1/2 11: 1/4 The content of this register will only determine the clock frequency of D-PLL when 25.175MHz is selected by legacy register.

VGACRBC: 25.175MHz PLL Setting Register

R/W:3B5/3D5 Index:BC MMIO:Base+BC **Init = E9h**

Bit	Attr.	Description
7:0	RW	**Video PLL numerator bit[7:0]**

VGACRBD: 25.175MHz PLL Setting Register

R/W:3B5/3D5 Index:BD MMIO:Base+BD **Init = 65h**

Bit	Attr.	Description
7	RW	**Reserved**
6:5	RW	**Video PLL post divider bit [1:0]** 00: 1/1 01: 1/2 10: 1/2 11: 1/4 The content of this register will only determine the clock frequency of D-PLL when 25.175MHz is selected by legacy register.
4:0	RW	**Video PLL de-numerator bit [4:0]**

VGACRBE: 28.322MHz PLL Setting Register

R/W:3B5/3D5 Index:BE MMIO:Base+BE **Init = 95h**

Bit	Attr.	Description
7:0	RW	**Video PLL numerator bit[7:0]**

VGACRBF: 28.322MHz PLL Setting Register

R/W:3B5/3D5 Index:BF MMIO:Base+BF **Init = 62h**

Bit	Attr.	Description
7	RW	**Reserved**

to next page

from previous page

6:5	RW	**Video PLL post divider bit [1:0]** 00: 1/1 01: 1/2 10: 1/2 11: 1/4 The content of this register will only determine the clock frequency of D-PLL when 28.322MHz is selected by legacy register.
4:0	RW	**Video PLL de-numerator bit[4:0]**

VGACRC0: Video PLL Setting Register

R/W:3B5/3D5 Index:C0 MMIO:Base+C0 **Init = 4Eh**

Bit	Attr.	Description
7:0	RW	**Video PLL numerator bit [7:0]**

VGACRC1: Video PLL Setting Register

R/W:3B5/3D5 Index:C1 MMIO:Base+C1 **Init = 61h**

Bit	Attr.	Description
7	RW	**Reserved**
6:5	RW	**Video PLL post divider bit [1:0]** 00: 1/1 01: 1/2 10: 1/2 11: 1/4
4:0	RW	**Video PLL de-numerator bit [4:0]**

VGACRC2: H/W Cursor X Position Offset Register

R/W:3B5/3D5 Index:C2 MMIO:Base+C2 **Init = 00h**

Bit	Attr.	Description
7:6	RW	**Reserved**
5:0	RW	**H/W cursor X position offset bit [5:0]**

VGACRC3: H/W Cursor Y Position Offset Register

R/W:3B5/3D5 Index:C3 MMIO:Base+C3 **Init = 00h**

Bit	Attr.	Description
7:6	RW	**Reserved**
5:0	RW	**H/W cursor Y position offset bit [5:0]**

VGACRC4: H/W Cursor X Position Register #1

R/W:3B5/3D5 Index:C4 MMIO:Base+C4 **Init = 00h**

Bit	Attr.	Description
7:0	RW	**H/W cursor X position bit [7:0]**

VGACRC5: H/W Cursor X Position Register #2		
R/W:3B5/3D5 Index:C5 MMIO:Base+C5		**Init = 00h**
Bit	**Attr.**	**Description**
7:5	RW	**Reserved**
4:0	RW	**H/W cursor X position bit [12:8]**

VGACRC6: H/W Cursor Y Position Register #1		
R/W:3B5/3D5 Index:C6 MMIO:Base+C6		**Init = 00h**
Bit	**Attr.**	**Description**
7:0	RW	**H/W cursor Y position bit [7:0]**

VGACRC7: H/W Cursor Y Position Register #2		
R/W:3B5/3D5 Index:C7 MMIO:Base+C7		**Init = 00h**
Bit	**Attr.**	**Description**
7:4	RW	**Reserved**
3:0	RW	**H/W cursor Y position bit [11:8]**

VGACRC8: H/W Cursor Pattern Address Register #1		
R/W:3B5/3D5 Index:C8 MMIO:Base+C8		**Init = 00h**
Bit	**Attr.**	**Description**
7:1	RW	**Cursor pattern memory address bit [10:4]** The address must be 16-byte aligned. Therefore memory address bit [3:0] is always ”0”.
0	R	**Reserved**

VGACRC9: H/W Cursor Pattern Address Register #2		
R/W:3B5/3D5 Index:C9 MMIO:Base+C9		**Init = 00h**
Bit	**Attr.**	**Description**
7:0	RW	**Cursor pattern memory address bit [18:11]**

VGACRCA: H/W Cursor Pattern Address Register #3		
R/W:3B5/3D5 Index:CA MMIO:Base+CA		**Init = 00h**
Bit	**Attr.**	**Description**
7	R	**Reserved**
6:0	RW	**Cursor pattern memory address bit [25:19]**

VGACRCB: H/W Cursor Control Register		
R/W:3B5/3D5 Index:CB MMIO:Base+CB		**Init = 00h**
Bit	**Attr.**	**Description**
7:2	RW	**Reserved**
1	RW	**Enable H/W cursor** 0: Disable H/W cursor display 1: Enable H/W cursor display

to next page

from previous page

0	RW	**H/W cursor type selection** 0: Select 2-BPP 1: Select 16-BPP (ARGB:4444)

VGACRCF: Video PLL Divder Register

R/W:3B5/3D5 Index:CF MMIO:Base+CF **Init = 00h**

Bit	Attr.	Description
7:3	RW	**Reserved**
2:0	RW	**Video PLL post divider bit [6:4]** 000: 1/1 001: 1/2 010: 1/3 011: 1/4 100: 1/5 101: 1/6 110: 1/7 111: 1/8

VGACRD0–DF: SOC Scratch Register #0 ~ #15

R/W:3B5/3D5 Index:D0–DF MMIO:Base+D0–DF **Init = 00h**

Bit	Attr.	Description
7:0	R	**SOC Scratch register bit [7:0]**

VGACRE0–FA: Scratch Register #32 ~ #58

R/W:3B5/3D5 Index:E0–FA MMIO:Base+E0–FA **Init = 00h**

Bit	Attr.	Description
7:0	R	**SOC Scratch register bit [7:0]**

VGACRFB: Misc. Control FB Register

R/W:3B5/3D5 Index:FB MMIO:Base+FB **Init = 00h**

Bit	Attr.	Description
7	RW	**Software reset XDMA engine**
6:1	RW	**Reserved**
0	RW	**Enable XDMA Engine** 0: Reset 1: Enable

VGACRFC: PLL Control FC Register

R/W:3B5/3D5 Index:FC MMIO:Base+FC **Init = 00h**

Bit	Attr.	Description
7:2	RW	**Reserved**
1	RW	**Enable extra 4 pixels of Hor Sync Time**
0	RW	**Reserved**

VGACRFD: PLL Control FD Register		
R/W:3B5/3D5 Index:FD MMIO:Base+FD		**Init = 00h**
Bit	**Attr.**	**Description**
7:0	RW	**Reserved**

VGACRFE: PLL Control FE Register		
R/W:3B5/3D5 Index:FE MMIO:Base+FE		**Init = 00h**
Bit	**Attr.**	**Description**
7:0	RW	**Reserved**

VGACRFF: PLL Control FF Register		
R/W:3B5/3D5 Index:FF MMIO:Base+FF		**Init = 00h**
Bit	**Attr.**	**Description**
7:0	RW	**Reserved**

69 2D Graphics Engine (G2D)

69.1 Overview

2D Graphics Engine supports a variety of 2D graphics commands to accelerate rendering performance. The maximum running frequency is 266MHz. The highest throughput this engine can achieve is 64 bits of data output per clock. This throughput number can be converted to 8 pixels per clock for 256 color modes, 4 pixels per clock for high color modes, and 2 pixels per clock for true color modes. AST2600 supports the following commands:

- **BitBLT operations**: logic operations among source, destination, pattern, and mask
- **Font expansion**: expanding monochrome bitmaps to color bitmaps
- **Line drawing**: rendering lines with style option
- **Transparent BitBlt**: logic transparent operations among source, destination
- **Horizontal and vertical Scale**: up-scale and down-scale among source, destination
- **Alpha Blanding**: constrant and source alpha blanding among source, destination

2D Graphics Engine implements a 32-bit registers set, which are listed below, to program the various supported functions. Some of the registers have different definitions for different 2D graphics commands, especially for BitBLT command , Transparent BitBlt command , Scale command , Alpha Blanding command and line drawing command. All these register can be access through PCI memory-mapped I/O cycles regarding to the following formula.

When function as SOC 2D Engine:
Base address of 2D Graphics Engine = 0x1E76_0000

When function as PCI VGA device:
Base address of 2D Graphics Engine = (PCIS14: Base Address 1 Register) + 0x8000

Register address of 2D Graphics Engine = (Base address of 2D Graphics Engine) + Offset

69.2 Features

- Directly access data through M-Bus
- High performance pipelined one-cycle 64-bit 2D graphics engine
- 2D engine commands
  - BitBlt Rectangle Fill
  - BitBlt Pattern Fill
  - BitBlt Rectangle Copy from Source to Destination
  - Support 256 Raster Operations
  - Integrate 8x8 Pattern Registers
  - Integrate 8x8 Mask Registers
  - Support Rectangle Clip
  - Support Color Expansion
  - Support Enhanced Color Expansion
  - Support Line Drawing with Style Pattern
  - Support Line Setup
  - Transparent BitBlt
  - Horizontal and vertical Scale

  - YUV to RGB Transform
  - Constrant & Source Alpha Blanding
- Programmable 256K/512K/1M/2M off-screen command buffer
- Integrate 32 stages of hardware command queue for 2D command pre-fetch from off-screen memory space of frame buffer
- Integrate 64x32 source buffer and 64x32 destination buffer to improve 2D engine performance
- Integrate 98x32 post-write buffer to improve 2D engine performance
- Optional interrupt generation when engine idle

69.3 2D Engine Registers

Offset: 00h	GER00: Base Address of Source Buffer Register	Init = X
Bit	**R/W**	**Description**
31:30		**Reserved (0)**
29:3	RW	**Base address of source buffer bit [29:3]** The address should be always 8-bytes aligned. Therefore, bit [2:0] is always ”0”.
2 :0		**Reserved (0)**
GER00: Base Address of Font Buffer Register (Enhanced Font Expansion)		
31:30		**Reserved (0)**
29:3	RW	**Base address of font buffer bit [29:3]** The address should be always 8-bytes aligned. Therefore, bit [2:0] is always ”0”.
2 :0		**Reserved (0)**

Offset: 04h	GER04: Row Pitch of Source Buffer Register	Init = X
Bit	**R/W**	**Description**
31:30		**Reserved (0)**
29:19	RW	**Row pitch of source buffer bit[13:3]** Row pitch of source buffer is equal to the width of source bitmap multiplied by bytes per pixel. The range of row pitch of source buffer has to meet the following limitations (number of bytes): MODE / Value 256 color / 0008h~07F8h High color / 0008h~0FF8h True color / 0008h~1FF8h
18:0		**Reserved (0)**
GER04: Row Pitch of Font Buffer Register (Enhanced Font Expansion)		
31:27		**Reserved (0)**
26:16	RW	**Row pitch of font buffer bit[10:0]** GER04[26:16] = (GER18 [27:16] + 7) >> 3 1 <= GER04[26:16] * GER18 [11:0] <= fffh
15:0		**Reserved (0)**

Offset: 08h	GER08: Base Address of Destination Buffer Register	Init = X
Bit	**R/W**	**Description**
31:30		**Reserved (0)**
29:3	RW	**Base address of destination buffer (bit [29:3])** The address should be always 8-bytes aligned. Therefore, bit [2:0] is always ”0”.
2 :0		**Reserved (0)**

Offset: 0Ch	GER0C: Row Pitch and Height of Destination Buffer Register	Init = X

Bit	R/W	Description
31:30		**Reserved (0)**
29:19	RW	**Row pitch of destination buffer bit [13:3]** Row pitch of destination buffer is equal to the width of destination buffer multiplied by bytes per pixel. The range of row pitch of destination buffer has to meet the following limitations (number of bytes): MODE / Value 256 color / 0008h~07F8h High color / 0008h~0FF8h True color / 0008h~1FF8h
18:12		**Reserved (0)**
11:0	RW	**Height of destination buffer bit [11:0]** Height of destination buffer has to be in the range of 0001h~07FFh.

Offset: 10h	GER10: Coordinate of Destination Bitmap Register	Init = X

Bit	R/W	Description
31:28		**Reserved (0)**
27:16	RW	**X coordinate of top-left corner of destination bitmap bit [11:0]** The data format of this register is S11.0
15:12		**Reserved (0)**
11:0	RW	**Y coordinate of top-left corner of destination bitmap bit [11:0]** The data format of this register is S11.0
GER10: Coordinate of First Point of Line Drawing Register (Line)		
Enable line setup engine		
31:16	RW	**X coordinate of first point of line drawing bit [15:0]** When GER3C [22] = 0, the data format of this register will be S11.4 When GER3C [22] = 1, the data format of this register will be S11.0
15:0	RW	**Y coordinate of first point of line drawing bit [15:0]** When GER3C [22] = 0, the data format of this register will be S11.4 When GER3C [22] = 1, the data format of this register will be S11.0
Disable line setup engine		
27:16	RW	**X coordinate of start point of line drawing bit [11:0]** The data format of this register is S11.0
15:12		**Reserved (0)**
11:0	RW	**Y coordinate of start point of line drawing bit [11:0]** The data format of this register is S11.0

Offset: 14h	GER14: Coordinate of Source Bitmap Register	Init = X

Bit	R/W	Description
31:28		**Reserved (0)**
27:16	RW	**X coordinate of top-left corner of source bitmap bit [11:0]** The data format of this register is S11.0 When GER3C [2:0] = 2 or 3, 0 MUST be the value.

to next page

from previous page

15:12		**Reserved (0)**
11:0	RW	**Y coordinate of top-left corner of source bitmap bit [11:0]** The data format of this register is S11.0 When GER3C [2:0] = 2 or 3, 0 MUST be the value.
GER14: Coordinate of Secondary Point of Line Drawing Register (Line)		
Enable line setup engine		
31:16	RW	**X coordinate of secondary point of line drawing bit [15:0]** When GER3C [22] = 0, the data format of this register will be S11.4 When GER3C [22] = 1, the data format of this register will be S11.0
15:0	RW	**Y coordinate of secondary point of line drawing bit [15:0]** When GER3C [22] = 0, the data format of this register will be S11.4 When GER3C [22] = 1, the data format of this register will be S11.0
Disable line setup engine		
31:25		**Reserved (0)**
24	RW	**Major axis selection** 0: Select Y-Axis as the major axis for line drawing 1: Select X-Axis as the major axis for line drawing
23:22		**Reserved (0)**
21:0	RW	**Error term of line drawing algorithm bit [21:0]** This register defines the Error Term of a line drawing algorithm.

Offset: 18h **GER18: Drawing Width and Drawing Height Register** **Init = X**

Bit	R/W	Description
31:28		**Reserved (0)**
27:16	RW	**Width of destination bitmap bit [11:0]** Width of destination bitmap should be in the range below. MODE / Value: 256 color 1∼2040; High color 1∼2044; True color 1∼2046
15:12		**Reserved (0)**
11:0	RW	**Height of destination bitmap bit [11:0]**
GER18: Number of Point in Line Point Register # 1 ∼ #64 Register (Line with setup)		
31:7		**Reserved (0)**
6 :0		**Number of point in line point register # 1 ∼ #64 (Value: 0 ∼ 64)**
GER18: Width of Major Axis of Line Drawing Register (Line without setup)		
31:28		**Reserved (0)**
27:16	RW	**Width of major axis of line drawing bit [11:0]**
15:0		**Reserved (0)**

Offset: 1Ch	GER1C: Foreground Color of Pattern Register	Init = X
Bit	**R/W**	**Description**
31:0	RW	**Foreground color of pattern bit[31:0]**
GER1C: High Color Key of Source Bitmap Register (Transparent BitBlt)		
31:0	RW	**High color key of source bitmap[31:0]**
GER1C: Horizontal Initial Scaling Factor (Scale)		
31:19		**Reserved (0)**
18:0	RW	**Horizontal initial scaling factor[18:0]**

Offset: 20h	GER20: Background Color of Pattern Register	Init = X
Bit	**R/W**	**Description**
31:0	RW	**Background color of pattern bit [31:0]**
GER20: Low Color Key of Source Bitmap Register (Transparent BitBlt)		
31:0	RW	**Low color key of source bitmap[31:0]**
GER20: Vertical Initial Scaling Factor (Scale)		
31:19		**Reserved (0)**
18:0	RW	**Vertical initial scaling factor[18:0]**

Offset: 24h	GER24: Foreground Color of Source Register	Init = X
Bit	**R/W**	**Description**
31:0	RW	**Foreground color of source bit [31:0]**
GER24: K1 Term of Line Drawing Register (Line without setup)		
31:22		**Reserved (0)**
21:0	RW	**K1 term of line drawing algorithm bit [21:0]**
GER24: High Color Key of Destination Bitmap Register (Transparent BitBlt)		
31:0	RW	**High color key of destination bitmap[31:0]**
GER24: Horizontal Scaling Factor (Scale)		
31:19		**Reserved (0)**
18:0	RW	**Horizontal scaling factor[18:0]**

Offset: 28h	GER28: Background Color of Source Register	Init = X
Bit	**R/W**	**Description**
31:0	RW	**Background color of source bit [31:0]**
GER28: K2 Term of Line Drawing Register (Line without setup)		
31:22		**Reserved (0)**
21:0	RW	**K2 term of line drawing algorithm bit [21:0]**
GER28: Low Color Key of Destination Bitmap Register (Transparent BitBlt)		
31:0	RW	**Low color key of destination bitmap[31:0]**
GER28: Vertical Scaling Factor (Scale)		
31:19		**Reserved (0)**
18:0	RW	**Vertical scaling factor[18:0]**

Offset: 2Ch	GER2C: Monochrome Mask of Pattern Register # 0	Init = X
Bit	**R/W**	**Description**
31:0	RW	**Monochrome mask of pattern bit [31:0]**
		GER2C: Pattern of Line Style Register # 0 (Line)
31:0	RW	**Pattern of line style bit [31:0]**

Offset: 30h	GER30: Monochrome Mask of Pattern Register # 1	Init = X
Bit	**R/W**	**Description**
31:0	RW	**Monochrome mask of pattern bit [63:32]**
		GER30: Pattern of Line Style Register # 1 (Line)
31:0	RW	**Pattern of line style bit [63:32]**

Offset: 34h	GER34: Top-Left Clipping Corner of Rectangular Register	Init = X
Bit	**R/W**	**Description**
31:29		**Reserved (0)**
28:16	RW	**X coordinate of top-left corner of clipping rectangular bit [12:0]**
15:12		**Reserved (0)**
11:0	RW	**Y coordinate of top-left corner of clipping rectangular bit [11:0]**

Offset: 38h	GER38: Bottom-Right Corner of Clipping Rectangular Register	Init = X
Bit	**R/W**	**Description**
31:29		**Reserved (0)**
28:16	RW	**X coordinate of bottom-right corner of clipping rectangular bit [12:0]**
15:12		**Reserved (0)**
11:0	RW	**Y coordinate of bottom-right corner of clipping rectangular bit [11:0]**

Offset: 3Ch	GER3C: 2D Engine Command Register	Init = 0
Bit	**R/W**	**Description**
31	RW	**Reset line style counter** 0: No operation 1: Reset line style counter This register will determine the line style counter to be reset or not when executing a new line drawing command.
30	RW	**Enable line drawing command with style pattern** 0: Disable (line drawing command without a style pattern) 1: Enable (line drawing command with a style pattern)
29:24	RW	**Line style period[5:0]** Line style period can be up to 64 points at most.
23	RW	**End-point rendering control for line drawing commands** 0: Disable end-point rendering 1: Enable end-point rendering
22		**Line drawing X/Y coordinate format** 0: X/Y coordinate is S11.4 format 1: X/Y coordinate is S11.0 format

to next page

from previous page

21	RW	**X-axis rendering direction control** 0: Rendering in positive-X direction 1: Rendering in negative-X direction
20	RW	**Y-axis rendering direction control** 0: Rendering in positive-Y direction 1: Rendering in negative-Y direction
19	RW	**M-Bus request synchronization for 2D engine idle** 0: Disable M-Bus request synchronization 1: Enable M-Bus request synchronization
18	RW	**Enable transparent font expansion** 0: Enable opaque font expansion 1: Enable transparent font expansion
17:16	RW	**Pattern selection** 00: Pattern is from foreground color of pattern register 01: Pattern is from monochrome mask register 10: Pattern is from pattern register 11: **Invalid**
15:8	RW	**Command code of 256 raster operations bit[7:0]**
7	RW	**Enable transparent of monochrome mask** 0: Enable opaque mode 1: Enable transparent mode
6	RW	**Source bitmap selection** 0: Source bitmap is from video frame buffer 1: Source bitmap is from command queue (Line drawing command does NOT support)
5 :4	RW	**Color mode selection** 00: 256 color mode (8-bpp) 01: High color mode (16-bpp) 10: True color mode (32-bpp) 11: **Invalid**
3	RW	**Enable rectangular clipping** 0: Disable rectangular clipping 1: Enable rectangular clipping
2 :0	RW	**Command type selection** 000: BitBLT command 001: Line drawing command 010: Font expansion command (patterns are from registers) 011: Enhanced font expansion command (patterns are from frame buffer) 100: Transparent BitBlt command 101: Scale command 110: Alpha blanding command 111: **Invalid**
GER3C: Transparent Raster Operations (Transparent BitBlt)		
11:8	RW	**Transparent raster operations bit[3:0]**
GER3C: Scale Command Register (Scale)		
31	RW	**Scale mode** 0: Line Mode 1: Block Mode
30	RW	**HDTV YUV format** 0: SDTV YUV format 1: HDTV YUV format

to next page

from previous page

29:24	RW	**Scale segment number[5:0]**
23	RW	**Scale with equal width** 0: Scale without equal width 1: Scale with equal width
15:14	RW	**Scale format[1:0]** 00: RGB to RGB 01: YUV to RGB 10: RGB 32bpp to 16bpp 11: RGB 16bpp to 32bpp
13:12	RW	**YUV format[1:0]** 00: YUYV 01: YVYU 10: UYVY 11: VYUY
GER3C: Alpha Blanding Command Register (AlphaBland)		
22	RW	**Enable source alpha** 0: Enable source alpha 1: Disable source alpha
15: 8	RW	**Constant alpha[8:0]**

Offset: 40h **GER40: Flipping Command Register** **Init = 0**

Bit	R/W	Description
31	RW	**Flipping control bit** 0: Disable flipping control 1: Enable flipping control
30	R	**Finish Flipping ID**
29:4	RW	**Base address of flipping control [29:4]**
3 :1	RW	**Reserved (0)**
0	RW	**Current Flipping ID**

Offset: 44h **GER44: Command Queue Setting Register** **Init = 0**

Bit	R/W	Description
DRAM 1G Mode		
31:27	RW	**Reserved (0)**
26:0	RW	**Base address of command queue buffer bit[29:3]** The address should be always 8-bytes aligned. Therefore, bit [2:0] is always ”0”.
DRAM 256 Mode		
31:28	RW	**Available size of hardware command queue bit[3:0]** 0000: 8 Bytes 0001: 24 Bytes 0010: 40 Bytes 0011: 56 Bytes ... 1111: 248 Bytes

to next page

from previous page

27:26	RW	**Command queue buffer size** 00: 256KB 01: 512KB 10: 1MB 11: 2MB
25	RW	**Mode of command queue operation** 0: Command data is from video frame buffer 1: Command data is from memory-mapped I/O command
24:0	RW	**Base address of command queue buffer bit[27:3]**

Offset: 48h **GER48: Write-Pointer of Command Queue Register** **Init = 0**

Bit	R/W	Description
31:18		**Reserved (0)**
17:0	RW	**Write-pointer of command queue bit [20:3]**

Offset: 4Ch **GER4C: 2D Engine Status Register** **Init = 0**

Bit	R/W	Description
31	R	**Status of 2D Graphic Engine** 0: Engine is idle 1: Engine is busy
30:25		**Reserved (0)**
24	RW	**Status of 2D IDLE** 0: Not yet IDLE 1: 2D is IDLE Writing ”1” to this bit will clear this register.
23:21		**Reserved (0)**
20	RW	**Status of command queue available space** 0: Not yet exceeded 1: Available space of command queue exceed Writing ”1” to this bit will clear this register.
19:18		**Reserved (0)**
17:0	R	**Read-pointer of command queue bit [20:3]**

Offset: 50h **GER50: AHB Interrupt Control Register** **Init = 0**

Bit	R/W	Description
31:25		**Reserved (0)**
24	RW	**Enable interrupt when 2D engine is IDLE** 0: Disable interrupt 1: Enable interrupt when 2D engine is IDLE
23:21		**Reserved (0)**
20	RW	**Enable interrupt when the available space of command queue exceed GER50[17:0]** 0: Disable interrupt 1: Enable interrupt when the available space of command queue exceed GER50[17:0]
19:18		**Reserved (0)**
17:0	RW	**Interrupt of command queue available space bit [20:3]**

Offset: 58h	GER58: Tag Register #0	Init = 0
Bit	**R/W**	**Description**
31:0	RW	**Tag Register #0 [31:0]**

Offset: 5Ch	GER5C: Tag Register #1	Init = 0
Bit	**R/W**	**Description**
31:0	RW	**Tag Register #1 [31:0]**

Offset: 60h	GER60: 2D Feature Register	Init = 0
Bit	**R/W**	**Description**
31		**Enable DRAM 1G Mode**
30		**Enable Line setup engine**
29:8		**Reserved (0)**
DRAM 1G Mode		
7 :4	RW	**Available size of hardware command queue bit[3:0]** 0000: 8 Bytes 0001: 24 Bytes 0010: 40 Bytes 0011: 56 Bytes ... 1111: 248 Bytes
3 :2	RW	**Command queue buffer size** 00: 256KB 01: 512KB 10: 1MB 11: 2MB
1	RW	**Mode of command queue operation** 0: Command data is from video frame buffer 1: Command data is from memory-mapped I/O command
0	RW	**Reserved (0)**
DRAM 256 Mode		
7 :0	RW	**Reserved (0)**

Offset: 100∼ 1FCh	PTR00 ∼ PTRFC: Pattern Register # 1 ∼ #64	Init = X
Bit	**R/W**	**Description**
31:0	RW	**Pattern register for ROP bit [31:0]**
PTR00 ∼ PTRFC: Monochrome Bitmap Register # 1 ∼ #64 (Font Expansion)		
31:0	RW	**Monochrome bitmap register bit [31:0]**
PTR00 ∼ PTRFC: Line Point Register # 1 ∼ #64 (Line with setup)		
31:16	RW	**X coordinate of point of line drawing bit [15:0]** When GER3C [22] = 0, the data format of this register will be S11.4 When GER3C [22] = 1, the data format of this register will be S11.0
15:0	RW	**Y coordinate of point of line drawing bit [15:0]** When GER3C [22] = 0, the data format of this register will be S11.4 When GER3C [22] = 1, the data format of this register will be S11.0

70 Graphics Hardware Cursor

70.1 Features

- Supports 64x64 monochrome cursor with AND-XOR-RGB444 pixel format
- Supports 64x64 color cursor with ARGB4444 pixel format
- Supports X-Offset & Y-Offset options
- Cursor information can be read from VGA Scratch Registers
- Cursor bit-map can be read from the designated area within VGA frame buffer
- Automatically generates Cursor Interrupt when cursor information or cursor bit-map address is changed

70.2 Register Definition

Offset: 1E70:0008h	VR008: Video Engine Control Register	Init = 0
Bit	**R/W**	**Description**
8	RW	**Disable hardware cursor overlay for internal VGA** 0: With VGA hardware cursor overlay image 1: Without VGA hardware cursor overlay image This register can be set by ARM CPU to inform internal VGA controller to generate video data without hardware cursor overlay image. When this register is enabled, the hardware cursor overlay has to be done in clients by Quick Cursor algorithm. The DAC output of internal VGA controller is, if necessary, with hardware cursor overlay image even this register is set to 1.

Offset: 1E6E:2560h	SCU560: Interrupt Control and Status Register	Init = 0
Bit	**R/W**	**Description**
31:0		**Please refer SCU560 in System Control Unit.**

Offset: 1E6E:2E00	VGA Scratch Register #1	Init = 0
Bit	**R/W**	**Description**
31:30	R	**Reserved**
29:24	R	**Hardware cursor X position offset bit [5:0]**
23:22	R	**Reserved**
21:16	R	**Hardware cursor Y position offset bit[5:0]**
15:10	R	**Reserved**
9	R	**Hardware cursor type selection** 0 : Select 2-BPP 1 : Select 16-BPP (ARGB:4444)
8	R	**Hardware cursor is enabled** 0 : Disabled hardware cursor 1 : Enable hardware cursor
7:0	R	**Reserved**

Offset: 1E6E:2E04h		VGA Scratch Register #2	Init = 0
Bit	**R/W**	**Description**	
31:27	R	**Reserved**	
26:16	R	**Hardware cursor Y position bit[10:0]**	
15:12	R	**Reserved**	
11:0	R	**Hardware cursor X position offset bit[11:0]**	

Offset: 1E6E:2E08h		VGA Scratch Register #3	Init = 0
Bit	**R/W**	**Description**	
31:26	R	**Reserved**	
25:0	R	**Hardware cursor pattern memory address bit [25:0]**	

70.3 Cursor Shape Structure Definition

70.3.1 Monochrome Cursor Format (AND-XOR-RGB444 pixel format)

Bit[15] : AND Mask bit
Bit[14] : XOR Mask bit
Bit[13:12] : Reserved
Bit[11:8] : Cursor R bit[3:0]
Bit[7:4] : Cursor G bit[3:0]
Bit[3:0] : Cursor B bit[3:0]

Description	AND Mask bit	XOR Mask bit	Output Color
Background Color	0	0	Cursor R/G/B
Foreground Color	0	1	Cursor R/G/B
Transparent	1	0	Graphics R/G/B
Inversed	1	1	NOT Graphics R/G/B

70.3.2 Color Cursor Format (ARGB4444 pixel format)

Bit[15:12] : Alpha bit[3:0]
Bit[11:8] : Cursor R bit[3:0]
Bit[7:4] : Cursor G bit[3:0]
Bit[3:0] : Cursor B bit[3:0]

- Output Color = (1-Alpha) * Graphics R/G/B + Alpha * Cursor R/G/B
- Note:
  1. Graphcis R/G/B is the color bit-map decompressed from video stream
  2. The output color should be normalized to the target display format
  3. When X-Offset or Y-Offset is enabled, only a partial bit-map is displayed

71 Message Signaled Interrupts (MSI)

71.1 Overview

MSI engine includes 2 function. One is Message Signaled Interrupts specified in PCIe. The other is virtual INTx also specified in PCIe.

71.2 Features

- 2 exclusive interrupt types: INTx and MSI.
- Total 22 interrupt sources from 5 devices.
- 1 of 22 sources is controlled by APB

71.3 Operation

71.3.1 Interrupt Table

Interrupt sources from VGA Device

INT#	Description
0	VGA Interrrupt
1	2D Interrupt
2	X-DMA Interrupt (If BMC device is not enabled)
3	APB Controlled Interrupt (SCU18[6])(If BMC device is not enabled)

Interrupt sources from BMC Device

INT#	Description
0	X-DMA Interrupt
1	APB Controlled Interrupt (SCU18[6])

71.3.2 Message Signaled Interrupts

Capability Structure for 32-bit Message Address

31:24	23:16	15:8	7:0	
Message Control		Next Pointer	Capability ID	Capability Pointer
Message Address				Capability Pointer + 04h
Reserved		Message Data		Capability Pointer + 08h

Capability Structure for 64-bit Message Address

31:24	23:16	15:8	7:0	
Message Control		Next Pointer	Capability ID	Capability Pointer
Message Address Bit[31:0]				Capability Pointer + 04h
Message Address Bit[63:32]				Capability Pointer + 08h
Reserved		Message Data		Capability Pointer + 0Ch

Offset: 00h	Capability ID	Init = 0x05
Bit	**R/W**	**Description**
7: 0	R	**Constant: 0x05**

Offset: 01h	Next Pointer	Init = 0
Bit	**R/W**	**Description**
7: 0	R	**Next item in the capabilities list**

Offset: 02h	Message Control	Init = 0
Bit	**R/W**	**Description**
15: 8	RW	**Reserved**
7	R	**64 bit capable.** 1: capable of 64-bit message address. 0: not capable of 64-bit messege address.
6: 4	RW	**Number of allocated messages.** It equal to or less than requested. Encoding | number of allocated 000 | 1 001 | 2 010 | 4 011 | 8 100 | 16 101 | 32 110 | Reserved 111 | Reserved
3: 1	R	**Number of requested messages.** Encoding | number of requested 000 | 1 001 | 2 010 | 4 011 | 8 100 | 16 101 | 32 110 | Reserved 111 | Reserved
0	RW	**MSI enable**

Offset: 04h	Message Address	Init = 0
Bit	**R/W**	**Description**
31: 2	RW	**System-specified message address**
1: 0	RW	**Reserved**

Offset: 08h	Message Upper Address (Optional)	Init = 0
Bit	**R/W**	**Description**
31: 0	RW	**System-specified upper message address**

Offset: 08h/0Ch	Message Data (MSG_D)	Init = 0
Bit	**R/W**	**Description**
31:16	RW	**Reserved**
15: 0	RW	**System-specified message**

Upsteam Data for VGA device

	Message Control[6:4]		
	000b	001b	010b or others
INT0	MSG_D[15:0]	MSG_D[15:0]	MSG_D[15:0]
INT1	MSG_D[15:0]	MSG_D[15:0] + 1	MSG_D[15:0] +1
INT2	MSG_D[15:0]	MSG_D[15:0]	MSG_D[15:0] +2
INT3	MSG_D[15:0]	MSG_D[15:0] + 1	MSG_D[15:0] +3

Upsteam Data for BMC device

	Message Control[6:4]	
	000b	001b or others
INT0	MSG_D[15:0]	MSG_D[15:0]
INT1	MSG_D[15:0]	MSG_D[15:0] + 1

Upsteam Data for NIC0 device

	Message Control[6:4] all
INT0	MSG_D[15:0]

Upsteam Data for NIC1 device

	Message Control[6:4] all
INT0	MSG_D[15:0]

Upsteam Data for LPC device

	Message Control[6:4]				
	000b	001b	010b	011b	100b or others
INT0	MSG_D[15:0]	MSG_D[15:0]	MSG_D[15:0]	MSG_D[15:0]	MSG_D[15:0]
INT1	MSG_D[15:0]	MSG_D[15:0] + 1	MSG_D[15:0] + 1	MSG_D[15:0] + 1	MSG_D[15:0] + 1
INT2	MSG_D[15:0]	MSG_D[15:0]	MSG_D[15:0] + 2	MSG_D[15:0] + 2	MSG_D[15:0] + 2
INT3	MSG_D[15:0]	MSG_D[15:0] + 1	MSG_D[15:0] + 3	MSG_D[15:0] + 3	MSG_D[15:0] + 3
INT4	MSG_D[15:0]	MSG_D[15:0]	MSG_D[15:0]	MSG_D[15:0] + 4	MSG_D[15:0] + 4
INT5	MSG_D[15:0]	MSG_D[15:0] + 1	MSG_D[15:0] + 1	MSG_D[15:0] + 5	MSG_D[15:0] + 5
INT6	MSG_D[15:0]	MSG_D[15:0]	MSG_D[15:0] + 2	MSG_D[15:0] + 6	MSG_D[15:0] + 6
INT7	MSG_D[15:0]	MSG_D[15:0] + 1	MSG_D[15:0] + 3	MSG_D[15:0] + 7	MSG_D[15:0] + 7
INT8	MSG_D[15:0]	MSG_D[15:0]	MSG_D[15:0]	MSG_D[15:0]	MSG_D[15:0] + 8
INT9	MSG_D[15:0]	MSG_D[15:0] + 1	MSG_D[15:0] + 1	MSG_D[15:0] + 1	MSG_D[15:0] + 9
INT10	MSG_D[15:0]	MSG_D[15:0]	MSG_D[15:0] + 2	MSG_D[15:0] + 2	MSG_D[15:0] + 10
INT11	MSG_D[15:0]	MSG_D[15:0] + 1	MSG_D[15:0] + 3	MSG_D[15:0] + 3	MSG_D[15:0] + 11
INT12	MSG_D[15:0]	MSG_D[15:0]	MSG_D[15:0]	MSG_D[15:0] + 4	MSG_D[15:0] + 12
INT13	MSG_D[15:0]	MSG_D[15:0] + 1	MSG_D[15:0] + 1	MSG_D[15:0] + 5	MSG_D[15:0] + 13
INT14	MSG_D[15:0]	MSG_D[15:0]	MSG_D[15:0] + 2	MSG_D[15:0] + 6	MSG_D[15:0] + 14
INT15	MSG_D[15:0]	MSG_D[15:0] + 1	MSG_D[15:0] + 3	MSG_D[15:0] + 7	MSG_D[15:0] + 15

71.3.3 Virtual INTx

If one MSI Enable of 5 devices is 0, engine is in Virtual INTx mode. When an interrupt occurs, engine will send a msg ”ASSERT”. When interrupts is cleared, engine will send a msg ”DE-ASSERT”.

72 PCIe BMC Device (PBMC)

72.1 Overview

AST2600 integrated an other bus controller designed to bridge the PCI-E bus to BMC, which can directly communicate with BMC controllers or memories. It implements total 13 PCI Configuration registers, they are compatiable to PCI type 0 configuration registers settings, to control the various functions supported by AST2600 BMC device.

PCIB00: Device and Vendor ID Register
PCIB04: Command and Status Register
PCIB08: Class and Revision ID Register
PCIB0C: Miscellaneous Register
PCIB10: Base Address 0 Register (for linear frame buffer)
PCIB14: Base Address 1 Register (for MMIO)
PCIB18: Base Address 2 Register (for relatable I/O)
PCIB2C: Subsystem ID Register
PCIB30: Expansion ROM Base Address Register
PCIB34: Capability Register
PCIB3C: Interrupt Register
PCIB40: PCI Power Management Capability Register
PCIB44: PCI Power Management Control and Status Register
PCIB50: Message Capability Register
PCIB54: Message Address Register
PCIB58: Message Upper Address Register
PCIB5C: Message Data Register

72.2 Features

- Support 32-bit 33 MHz PCI bus interface with PCI 2.3 specification compliant
- Support PME# control pin

Offset: 00h **PCIB00: Device and Vendor ID Register** **Init = 0x2402_1A03**

Bit	R/W	Description
31:16	R	**Device ID** The default setting of this register is **0x2402**, which is the device ID code being assigned for AST2600 BMC device. The content of this register value can be changed by updating the corresponding register in SCU Controller, but its not recommended in normal cases.
15:0	R	**Vendor ID** The default setting of this register is **0x1A03** which is the vendor ID code being assigned for **ASPEED Technology Inc**. by **PCISIG**. The content of this register value can be changed by updating the corresponding register in SCU Controller, but its not recommended in normal cases.

Offset: 04h **PCIB04: Command and Status Register** **Init = 0x0210_0000**

Bit	R/W	Description
31	R	**Detected parity error** AST2600 BMC device will not detect any parity errors; therefore, this bit will always return ”0”.

to next page

from previous page

30	R	**Signaled system error** AST2600 BMC device will not signal any system errors; therefore, this bit will always return ”0”.
29	R	**Received master abort** AST2600 BMC device doesn’t play as a bus master; this register will always return ”0”.
28	R	**Received target abort** AST2600 BMC device doesn’t play as a bus master; this register will always return ”0”.
27	R	**Signaled target abort** AST2600 BMC device will not issue target abort; this register will always return ”0”.
26:25	R	**DEVSEL timing** AST2600 BMC device supports medium timing for DEVSEL signal; this register will always return ”01”.
24	R	**Master data parity error** AST2600 BMC device doesn’t play as a bus master; this register will always return ”0”.
23	R	**Fast back-to-back capable** AST2600 BMC device doesn’t support fast back-to-back; this register will always return ”0”.
22		**Reserved (0)**
21	R	**66 MHz capable** AST2600 BMC device supports 33MHz PCI bus running frequency; this register will always return ”0”.
20	R	**Capabilities list** AST2600 BMC device supports a linked list to implement PCI bus power management; this register will always return ”1”.
19	R	**Interrupt status** This read-only bit reflects the state of the only interrupt source generated by CRT Controller for detecting the end of vertical display enable. This is a legacy interrupt from BMC device. In most of the cases, this interrupt source will not be enabled.
18:11		**Reserved (0)**
10	RW	**Interrupt disable** 0: Enable interrupt 1: Disable interrupt
9	R	**Fast back-to-back enable** AST2600 BMC device doesn’t support fast back-to-back; this register will always return ”0”.
8	RW	**SERR# enable** AST2600 BMC device wont signal any system errors.
7	R	**Reserved (0)**
6	RW	**Parity error response enable** AST2600 BMC device wont detect any parity errors.
5	R	**Reserved (0)**
4	R	**Memory write and invalidate enable** Since AST2600 BMC device wont support this feature; this register will always return ”0”.
3	R	**Special cycles enable** Since AST2600 BMC device doesn’t support PCI special cycles, this register will always return ”0”.
2	R	**Bus master enable** AST2600 BMC device doesn’t need PCI bus master cycles.

to next page

from previous page

1	RW	**Memory space access enable** 0: Disable memory space accesses 1: Enable memory space accesses This register will determine whether AST2600 BMC device will response to memory space accesses or not.
0	RW	**IO space access enable** 0: Disable I/O space accesses 1: Enable I/O space accesses This register will determine whether AST2600 BMC device will response to I/O space accesses or not.

Offset: 08h **PCIB08: Class and Revision ID Register** **Init = 0x0000_0000**

Bit	R/W	Description
31:8	R	**Class code** The defalut value is 0. Because PCI-SIG didn't define the BMC device class code, the BMC firmware should program the desired class code for BMC device before host boot up. The content of this register value can be changed by updating the corresponding register in SCU Controller.
7 :0	R	**Revision ID** This register defines the revision ID of the current working silicon. It will change whenever a new revision is developed. The default revision ID of AST2600 BMC device is "00".

Offset: 0Ch **PCIB0C: Miscellaneous Register** **Init = 0x0000_0000**

Bit	R/W	Description
31	R	**BIST Capable** AST2600 BMC device doesn't support BIST; this register will always return "0".
30	R	**Start BIST** AST2600 BMC device doesn't support BIST; this register will always return "0".
29:28	R	**Start BIST** AST2600 BMC device doesn't support BIST; this register will always return "0".
27:24	R	**Completion code** AST2600 BMC device doesn't support BIST; this register will always return "0".
23:16	R	**Header type** This register will always return "0".
15:8	R	**Latency timer** AST2600 BMC device doesn't play as a bus master; this register will always return "0".
7 :0	RW	**Cache line size** This register will always return "0".

Offset: 10h **PCIB10: Base Address 0 Register** **Init = 0x0000_0000**

Bit	R/W	Description
31:0	RW	**Base address 0 register** AST2600 BMC device will claim a re-locatable memory space 256KB for linear frame buffer allocation by this base address register.

Offset: 14h	PCIB14: Base Address 1 Register	Init = 0x0000_0000

Bit	R/W	Description
31:0	RW	**Base address 1 register** AST2600 BMC device will claim a 1KB re-locatable I/O space allocation by this base address register.

Offset: 2Ch	PCIB2C: Subsystem ID Register	Init = 0x2000_1A03

Bit	R/W	Description
31:16	RW	**Subsystem ID** This is a per-byte write once register. Once begin updated, this register cannot be modified again until next power-on. The default setting of this register is **0x2000**. This register is identical to PCIS2C.
15:0	RW	**Subsystem vendor ID** This is a per-byte write once register. Once begin updated, this register cannot be modified again until next power-on. The default setting of this register is **0x1A03**, which is following the vendor ID code of **ASPEED Technology Inc**. This register is identical to PCIS2C.

Offset: 34h	PCIB34: Capability Register	Init = 0x0000_0040

Bit	R/W	Description
31:8		**Reserved (0)**
7 :0	R	**Capabilities pointer** This optional register is used to point to a linked list of new capabilities. AST2600 BMC device uses this register to point to 0x40 to implement PCI power management capability.

Offset: 3Ch	PCIB3C: Interrupt Register	Init = 0x0000_0100

Bit	R/W	Description
31:24	R	**Maximum latency** This register is used for specifying how often the device needs to gain access to the PCI bus.
23:16	R	**Minimum grant** This register is used for specifying how long a burst period the device needs assuming a clock rate of 33 MHz.
15:8	R	**Interrupt pin** AST2600 BMC device always returns 0x01 for this register to claim that the interrupt pin INTA# will be used by this device.
7 :0	RW	**Interrupt line** AST2600 BMC device provides an 8-bit read/write register to implement this function. The content of this register will not impact any hardware behavior.

Offset: 40h	PCIB40: PCI Power Management Capability Register	Init = 0xffc3_5001

Bit	R/W	Description
31:27	R	**PME support** AST2600 BMC device supports D0, D1, D2, D3hot and D3cold states. Therefore, this register will always return 0xF.
26	R	**D2 support** AST2600 supports D2 state; this register will always return ”1”.

to next page

from previous page

25	R	**D1 support** AST2600 BMC device supports D1 state; this register will always return ”1”.
24:22	R	**Auxiliary current requirement** This register will always return ”111b”. It means that AST2600 BMC device requires 375mA from auxiliary current.
21	R	**Device specific initialization** AST2600 BMC device doesn’t need any special initializations. This register will always return ”0”.
20		**Reserved (0)**
19	R	**PME Clock** AST2600 BMC device doesn’t need to rely on PCI clock to generate PME#. This register will always return ”0”.
18:16	R	**Version** AST2600 BMC device complies with PCI Power Management Revision 1.2. Therefore, This register will always return ”011b”.
15:8	R	**Next item pointer** This optional register is used to point to a linked list of new capabilities. AST2600 BMC device uses this register to point to 0x50 to implement Message Signaled Interrupts.
7 :0	R	**ID** This register will always return ”0x01” to identify that the linked list item as being the PCI Power Management registers.

Offset: 44h PCIB44: PCI Power Management Control and Status Register Init = 0x0000_0000

Bit	R/W	Description
31:24	R	**Data register** This function is not implemented; this register always returns ”0x00”.
23	R	**Bus power and clock control enable** There is no secondary PCI bus; this register always returns ”0”.
22	R	**B2/B3 support for D3hot** There is no secondary PCI bus; this register always returns ”0”.
21:16		**Reserved (0)**
15	RW	**PME Status** This bit is set when AST2600 BMC device would normally assert the PME signal independent of the state of the PME enable bit. Writing ”1” to this register will cause AST2600 BMC device to stop asserting the PME signal. Writing ”0” to this register has no effect.
14:13	R	**Data scale** AST2600 BMC device doesn’t implement Data register; this register always returns ”00b”.
12:9	RW	**Data select** AST2600 BMC device doesn’t implement Data register; this register always returns ”0000b”.
8	RW	**PME enable** 0: Disable PME assertion 1: Enable PME assertion
7 :4		**Reserved (0)**
3	R	**No soft reset** This register always returns ”0”.
2		**Reserved (0)**

to next page

from previous page

1 :0	RW	**Power state** These two bits are used both to determine the current power state of AST2600 BMC device and to set AST2600 BMC device into a new power state. AST2600 BMC processor will not be controlled by this register.

Offset: 50h **PCIB50: Message Capability Register** **Init = 0x0084_0005**

Bit	R/W	Description
31:24		**Reserved (0)**
23	R	**64 bit address capable** This register will always return ”1b”. It means that AST2600 BMC device is capable of generating a 64-bit message address.
22:20	RW	**Multiple Message Enable** System software writes to this field to indicate the number of allocated messages.
19:17	R	**Multiple Message Capable** This register will always return ”010b”. It means that AST2600 BMC device requests 4 messages.
16	RW	**MSI Enable** 0: Disable MSI 1: Enable MSI
15:8	R	**Next item pointer** No next item pointer required. This register will always return ”0x00”.
7 :0	R	**ID** This register will always return ”0x05” to identify that the linked list item as being the Message Signaled Interrupts registers.

Offset: 54h **PCIB54: Message Address Register** **Init = 0x0000_0000**

Bit	R/W	Description
31:0	RW	**Message Address bit [31:0]** System-specified message address.

Offset: 58h **PCIB58: Message Upper Address Register** **Init = 0x0000_0000**

Bit	R/W	Description
31:0	RW	**Message Address bit [63:32]** System-specified message upper address.

Offset: 5Ch **PCIB5C: Message Data Register** **Init = 0x0000_0000**

Bit	R/W	Description
31:16		**Reserved (0)**
15:0	RW	**Message Data bit [15:0]** System-specified message.

73 PCIe USB2.0 EHCI Host Controller (BEHCI)

73.1 Overview

USB2.0 Host Controller (EHCI) is adapted from EHCI Rev1.0 and only located in physical portA. A RootHub is embedded in the core, by default, only one USB2.0 downstream port that connected to Virtual Hub controller is implemented. It implements several PCI Configuration registers, they are compatiable to PCI type 0 configuration registers settings, to control the AST2600 EHCI device.

EHCI PCIe Configuration Register Set

PCIEHCI00: Device and Vendor ID Register
PCIEHCI08: Class and Revision ID Register
PCIEHCI10: Base Address Register
PCIEHCI2C: Subsystem ID Register
PCIEHCI34: Capability Register
PCIEHCI3C: Interrupt Register
PCIEHCI40: PCI Power Management Capability Register
PCIEHCI44: PCI Power Management Control and Status Register
PCIEHCI50: Message Capability Register
PCIEHCI54: Message Address Register
PCIEHCI58: Message Upper Address Register
PCIEHCI5C: Message Data Register
PCIEHCI60: EHCI Miscellaneous Register
PCIEHCI68: USB Legacy Support EHCI Extended Capability Register
PCIEHCI6C: USB Legacy Support Control and Status Register

Base address of EHCI Host Controller Registers = (PCI Configuration Space offset 0x10-0x13)
Base address for Read from AHB Bus = 0x1E6A_1800

EHCI Register Set

BEHCI00: Capability Registers Length (CAPLENGTH)
BEHCI04: Structural Parameters (HCSPARAMS)
BEHCI08: Capability Parameters (HCCPARAMS)
BEHCI0C: Companion Port Route Description (HCSP-PORTROUTE)
BEHCI20: USB Command Register (USBCMD)
BEHCI24: USB Status Register (USBSTS)
BEHCI28: USB Interrupt Enable Register (USBINTR)
BEHCI2C: Frame Index Register (FRINDEX)
BEHCI34: Periodic Frame List Base Address Register (PERIODICLISTBASE)
BEHCI38: Current Asynchronous List Address Register (ASYNCLISTADDR)
BEHCI60: Configure Flag Register (CONFIGFLAG)
BEHCI64: Port1 Status/Control Register (PORTSC1)
BEHCI80: (AHB Only)Frame Length Adjustment Register (FLADJ)
BEHCI84: (AHB Only)Controler Fine-tune Register
BEHCI88: (AHB Only)Frame Timing Adjustment
BEHCI8C: (AHB Only)Hardware Revision Number Register

73.2 EHCI Features

- Support 32-bit 33 MHz PCI bus interface with PCI 2.3 specification compliant

- Support PME# control pin
- Complies with USB Specification Rev 2.0
- Adapted from Enhanced Host Controller Interface (EHCI) Specification Rev1.0
- Supports 1 port data transfer at high-speed (480 Mbit/s)(EHCI)
- Supports all four types of USB transfers: control, bulk, interrupt and isochronous
- Includes a RootHub with multi-port architecture
- Directly addressable memory architecture; memory can be updated on-the-fly
- Supports USB Host Controller I/O registers for software communication channel
- Re-uses Linux EHCI Host Controller driver with minor change on register offset
- Register set are only capable of double-word read/write
- Supported capabilities:
  * 1 down-stream port directly connected to Virtual Hub controller
  * no port power control (PPC = 0)
  * no companion controller (N_CC = 0)
  * no port indicator
  * no debug port
  * support 64-bit addressing
  * programmable frame list size: 1024, 512, 256
  * programmable asynchronous schedule park (BEHCI84[1])
  * EHCI extended capabilities

73.3 Procedure to enable USB Host port

73.3.1 EHCI to Hub path

- Set SCU440 bit[25:24] = ”00”, PCIe EHCI Host directly connected to USB2.0 Virtual Hub controller
- Set SCU040 bit[14] = 1, enable controller reset
- Set SCU080 bit[14] = 1, enable PHY clock
- wait 10 ms for PLL locking
- Set SCU044 bit[14] = 1, disable controller reset
- Set SCUC20[16] to enable PCIe EHCI device on PCIe bus

73.3.2 EHCI to USB Host port 1

- Set SCU440 bit[25:24] = ”11”
- Set SCU040 bit[14] = 1, enable controller reset
- Set SCU080 bit[14] = 1, enable PHY clock
- wait 10 ms for PLL locking
- Set SCU044 bit[14] = 1, disable controller reset
- Set SCUC20[16] to enable PCIe EHCI device on PCIe bus

73.4 EHCI PCIe Configuration Registers

Offset: 00h	PCIEHCI00: Device and Vendor ID Register	Init = 0x2603_1A03

Bit	R/W	Description
31:16	RO	**Device ID** The default setting of this register is **0x2603**, which is the device ID code being assigned for AST2600 PCIe EHCI device. The content of this register value can be changed by updating the corresponding register [23:16] in SCUC64[7:0], but its not recommended in normal cases.
15:0	RO	**Vendor ID** The default setting of this register is **0x1A03** which is the vendor ID code being assigned for **ASPEED Technology Inc**. by **PCISIG**. The content of this register value can be changed by updating the corresponding register in SCUC00[15:0], but its not recommended in normal cases.

Offset: 08h	PCIEHCI08: Class and Revision ID Register	Init = 0x0C03_2000

Bit	R/W	Description
31:8	RO	**Class code** This register contains the device programming interface information related to the Sub-Class Code and Base Class Code definition. Base Class Code [31:24]. 0Ch = Serial Bus Controller. Sub-Class Code [23:16]. 03h = Universal Serial Bus Host Controller. Programming Interface [15:8]. 20h = USB 2.0 Host Controller.
7 :0	RO	**Revision ID** This register defines the revision ID of the current working silicon. It will change whenever a new revision is developed. The default revision ID of AST2600 PCIe EHCI device is ”00”.

Offset: 10h	PCIEHCI10: Base Address Register	Init = 0x0000_00000

Bit	R/W	Description
31:0	RW	**Base address register** This register contains the base address of the DWord-aligned memory-mapped host controller Registers. Base Address – RW. Corresponds to memory address signals[31:8]. Type [2:1] – RO. May only be mapped into 32-bit addressing space.

Offset: 2Ch	PCIEHCI2C: Subsystem ID Register	Init = 0x2000_1A03

Bit	R/W	Description
31:16	RW	**Subsystem ID** This is a per-byte write once register. Once begin updated, this register cannot be modified again until next power-on. The default setting of this register is **0x2000**. This register is identical to PCIS2C.
15:0	RW	**Subsystem vendor ID** This is a per-byte write once register. Once begin updated, this register cannot be modified again until next power-on. The default setting of this register is **0x1A03**, which is following the vendor ID code of **ASPEED Technology Inc**. This register is identical to PCIS2C.

Offset: 34h	PCIEHCI34: Capability Register	Init = 0x0000_0040

Bit	R/W	Description
31:8	RO	**Reserved (0)**

to next page

from previous page

7 :0	RO	**Capabilities pointer** This optional register is used to point to a linked list of new capabilities. AST2600 PCIe EHCI device uses this register to point to 0x40 to implement PCI power management capability.

Offset: 3Ch **PCIEHCI3C: Interrupt Register** **Init = 0x0000_0100**

Bit	R/W	Description
31:24	RO	**Maximum latency** This register is used for specifying how often the device needs to gain access to the PCI bus.
23:16	RO	**Minimum grant** This register is used for specifying how long a burst period the device needs assuming a clock rate of 33 MHz.
15:8	RO	**Interrupt pin** AST2600 PCIe EHCI device always returns 0x01 for this register to claim that the interrupt pin INTA# will be used by this device.
7 :0	RW	**Interrupt line** AST2600 PCIe EHCI device provides an 8-bit read/write register to implement this function. The content of this register will not impact any hardware behavior.

Offset: 40h **PCIEHCI40: PCI Power Management Capability Register** **Init = 0xffc3_5001**

Bit	R/W	Description
31:27	RO	**PME support** AST2600 PCIe EHCI device supports D0, D1, D2, D3hot and D3cold states. Therefore, this register will always return 0xF.
26	RO	**D2 support** AST2600 supports D2 state; this register will always return ”1”.
25	RO	**D1 support** AST2600 PCIe EHCI device supports D1 state; this register will always return ”1”.
24:22	RO	**Auxiliary current requirement** This register will always return ”111b”. It means that AST2600 PCIe EHCI device requires 375mA from auxiliary current.
21	RO	**Device specific initialization** AST2600 PCIe EHCI device doesn’t need any special initializations. This register will always return ”0”.
20	RO	**Reserved (0)**
19	RO	**PME Clock** AST2600 PCIe EHCI device doesn’t need to rely on PCI clock to generate PME#. This register will always return ”0”.
18:16	RO	**Version** AST2600 PCIe EHCI device complies with PCI Power Management Revision 1.2. Therefore, This register will always return ”011b”.
15:8	RO	**Next item pointer** This optional register is used to point to a linked list of new capabilities. AST2600 PCIe EHCI device uses this register to point to 0x50 to implement Message Signaled Interrupts.
7 :0	RO	**ID** This register will always return ”0x01” to identify that the linked list item as being the PCI Power Management registers.

Offset: 44h PCIEHCI44: PCI Power Management Control and Status Register Init = 0x0000_0000		
Bit	**R/W**	**Description**
31:24	RO	**Data register** This function is not implemented; this register always returns ”0x00”.
23	RO	**Bus power and clock control enable** There is no secondary PCI bus; this register always returns ”0”.
22	RO	**B2/B3 support for D3hot** There is no secondary PCI bus; this register always returns ”0”.
21:16	RO	**Reserved (0)**
15	RW	**PME Status** This bit is set when AST2600 PCIe EHCI device would normally assert the PME signal independent of the state of the PME enable bit. Writing ”1” to this register will cause AST2600 PCIe EHCI device to stop asserting the PME signal. Writing ”0” to this register has no effect.
14:13	RO	**Data scale** AST2600 PCIe EHCI device doesn’t implement Data register; this register always returns ”00b”.
12:9	RO	**Data select** AST2600 PCIe EHCI device doesn’t implement Data register; this register always returns ”0000b”.
8	RW	**PME enable** 0: Disable PME assertion 1: Enable PME assertion
7 :4	RO	**Reserved (0)**
3	RO	**No soft reset** This register always returns ”0”.
2	RO	**Reserved (0)**
1 :0	RW	**Power state** These two bits are used both to determine the current power state of AST2600 PCIe EHCI device and to set AST2600 PCIe EHCI device into a new power state. AST2600 PCIe EHCI processor will not be controlled by this register.

Offset: 50h PCIEHCI50: Message Capability Register Init = 0x0084_0005		
Bit	**R/W**	**Description**
31:24	RO	**Reserved (0)**
23	RO	**64 bit address capable** This register will always return ”1b”. It means that AST2600 PCIe EHCI device is capable of generating a 64-bit message address.
22:20	RW	**Multiple Message Enable** System software writes to this field to indicate the number of allocated messages.
19:17	RO	**Multiple Message Capable** This register will always return ”000b”. It means that AST2600 PCIe EHCI device requests 1 message.
16	RW	**MSI Enable** 0: Disable MSI 1: Enable MSI
15:8	RO	**Next item pointer** No next item pointer required. This register will always return ”0x00”.

to next page

from previous page

7 :0	RO	**ID** This register will always return ”0x05” to identify that the linked list item as being the Message Signaled Interrupts registers.

Offset: 54h **PCIEHCI54: Message Address Register** **Init = 0x0000_0000**

Bit	R/W	Description
31:0	RW	**Message Address bit [31:0]** System-specified message address.

Offset: 58h **PCIEHCI58: Message Upper Address Register** **Init = 0x0000_0000**

Bit	R/W	Description
31:0	RW	**Message Address bit [63:32]** System-specified message upper address.

Offset: 5Ch **PCIEHCI5C: Message Data Register** **Init = 0x0000_0000**

Bit	R/W	Description
31:16	RO	**Reserved (0)**
15:0	RW	**Message Data bit [15:0]** System-specified message.

Offset: 60h **PCIEHCI60: EHCI Miscellaneous Register** **Init = 0x0003_2020**

Bit	R/W	Description
31:18	RO	**Reserved (0)**
17:16	RW	**Port Wake Capability** This is an information only mask register. The bits in this register DO NOT afect the actual operation of the EHCI host controller. The system-specific policy can be established by BIOS initializing this register to a system-specific value. System software uses the information in this register when enableing devices and ports for remote wake-up.
15:14	RO	**Reserved (0)**

to next page

from previous page

13:8	RW	**Frame Length Timing Value** Each decimal value change to this register corresponds to 16 high-speed bit times. The SOF cycle time (number of SOF counter clock periods to generate a SOF micro-frame length) is equal to 59488 + value in this field. The default value is decimal 32 (20h), which gives a SOF cycle time of 60000. Frame Length (# High Speed bit times) (decimal) / FLADJ Value (decimal) 59488 — 0 (00h) 59504 — 1 (01h) 59520 — 2 (02h) ... 59984 — 31 (1Fh) 60000 — 32 (20h) ... 60480 — 62 (3Eh) 60496 — 63 (3Fh)
7 :0	RO	**Serial Bus Release Number** Release 2.0.

Offset: 68h PCIEHCI68: USB Legacy Support Extended Capability Register Init = 0x0000_0001

Bit	R/W	Description
31:25	RO	**Reserved (0)**
24	RW	**HC OS Owned Semaphore** 0 = Default. System software sets this bit to request ownership of the EHCI controller. Ownership is obtained when this bit reads as one and the HC BIOS Owned Semaphore bit reads as zero.
23:17	RO	**Reserved (0)**
16	RW	**HC BIOS Owned Semaphore** 0 = Default. The BIOS sets this bit to establish ownership of the EHCI controller. System BIOS will set this bit to a zero in response to a request for ownership of the EHCI controller by system software.
15:8	RO	**Next EHCI Extended Capability Pointer** This field points to the PCI configuration space offset of the next extended capability pointer. A value of 00h indicates the end of the extended capability list.
7:0	RO	**Capability ID** A value of 01h identifies the capability as Legacy Support.

Offset: 6Ch PCIEHCI6C: USB Legacy Support Control/Status Init = 0x0000_0000

Bit	R/W	Description
31	RW1C	**SMI on BAR** 0 = Default. This bit is set to one whenever the Base Address Register PCIEHCI10 is written.
30	RW1C	**SMI on PCI Command** 0 = Default. This bit is set to one whenever the PCI Command Register is written.

to next page

from previous page

29	RW1C	**SMI on OS Ownership Change** 0 = Default. This bit is set to one whenever the PCIEHCI68[24] bit transitions from 1 to a 0 or 1 to a 1.
28:22	RO	**Reserved (0)**
21	RO	**SMI on Async Advance** 0 = Default. Shadow bit of BEHCI24[5]. To set this bit to a zero, system software must write a one to BEHCI24[5].
20	RO	**SMI on Host System Error** 0 = Default. Shadow bit of BEHCI24[4]. To set this bit to a zero, system software must write a one to BEHCI24[4].
19	RO	**SMI on Frame List Rollover** 0 = Default. Shadow bit of BEHCI24[3]. To set this bit to a zero, system software must write a one to BEHCI24[3].
18	RO	**SMI on Port Change Detect** 0 = Default. Shadow bit of BEHCI24[2]. To set this bit to a zero, system software must write a one to BEHCI24[2].
17	RO	**SMI on USB Error** 0 = Default. Shadow bit of BEHCI24[1]. To set this bit to a zero, system software must write a one to BEHCI24[1].
16	RO	**SMI on USB Complete** 0 = Default. Shadow bit of BEHCI24[0]. To set this bit to a zero, system software must write a one to BEHCI24[0].
15	RW	**SMI on BAR Enable** 0 = Default. When this bit is one and SMI on BAR is one, then the host controller will issue an SMI.
14	RW	**SMI on PCI Command Enable** 0 = Default. When this bit is one and SMI on PCI Command is one, then the host controller will issue an SMI.
13	RW	**SMI on OS Ownership Enable** 0 = Default. When this bit is one and the OS Ownership Change bit is one, then the host controller will issue an SMI.
12:6	RO	**Reserved (0)**
5	RW	**SMI on Async Advance Enable** 0 = Default. When this bit is one and the SMI on Async Advance bit (above) in this register is a one, then the host controller will issue an SMI immediately.
4	RW	**SMI on Host System Error Enable** 0 = Default. When this bit is one and the SMI on Host System Error bit (above) in this register is a one, then the host controller will issue an SMI immediately.
3	RW	**SMI on Frame List Rollover Enable** 0 = Default. When this bit is one and the SMI on Frame List Rollover bit (above) in this register is a one, then the host controller will issue an SMI immediately.
2	RW	**SMI on Port Change Enable** 0 = Default. When this bit is one and the SMI on Port Change Detect bit (above) in this register is a one, then the host controller will issue an SMI immediately.
1	RW	**SMI on USB Error Enable** 0 = Default. When this bit is one and the SMI on USB Error bit (above) in this register is a one, then the host controller will issue an SMI immediately.

to next page

from previous page

0	RW	**USB SMI Enable** 0 = Default. When this bit is one and the SMI on USB Complete bit (above) in this register is a one, then the host controller will issue an SMI immediately.

73.5 EHCI Registers

Offset: 00h **BEHCI00: Capability Registers Length (CAPLENGTH)** **Init = 0x01000020**

Bit	R/W	Description
31:16	RO	**Host Controller Interface Version Number (HCIVERSION)**
15:8	RO	**Reserved (0)**
7:0	RO	**Offset to Operational Registers** This register is used as an offset to add to register base to find the beginning of the Operational Register Space.

Offset: 04h **BEHCI04: Structural Parameters (HCSPARAMS)** **Init = 0x00000001**

Bit	R/W	Description
31:16	RO	**Reserved (0)**
23:20	RO	**Debug Port Number**
19:17	RO	**Reserved (0)**
16	RO	**Port Indicators (P_INDICATOR)** This bit indicates whether the ports support port indicator control. When this bit is a one, the port status and control registers include a read/writeable field for controlling the state of the port indicator.
15:12	RO	**Number of Companion Controller (N_CC)** This field indicates the number of companion controllers associated with this USB 2.0 host controller. A zero in this field indicates there are no companion host controllers. Port-ownership hand-off is not supported. Only high-speed devices are supported on the host controller root ports. A value larger than zero in this field indicates there are companion USB 1.1 host controller(s). Port-ownership hand-offs are supported. High, Full- and Low-speed devices are supported on the host controller root ports.
11:8	RO	**Number of Ports per Companion Controller (N_PCC)** This field indicates the number of ports supported per companion host controller. It is used to indicate the port routing configuration to system software.
7	RO	**Port Routing Rules** This field indicates the method used by this implementation for how all ports are mapped to companion controllers. The value of this field has the following interpretation: **0:** The first N_PCC ports are routed to the lowest numbered function companion host controller, the next N_PCC port are routed to the next lowest function companion controller, and so on. **1:** The port routing is explicitly enumerated by the first N_PORTS elements of the HCSP-PORTROUTE array.
6:5	RO	**Reserved (0)**

to next page

from previous page

4	RO	**Port Power Control (PPC)** This field indicates whether the host controller implementation includes port power control. A one in this bit indicates the ports have port power switches. A zero in this bit indicates the port do not have port power switches. The value of this field affects the functionality of the Port Power field in each port status and control register.
3:0	RO	**N_PORTS** This field specifies the number of physical downstream ports implemented on this host controller. The value of this field determines how many port registers are addressable in the Operational Register Space. Valid values are in the range of 1H to FH.

Offset: 08h BEHCI08: Capability Parameters (HCCPARAMS) Init = 0x00006817

Bit	R/W	Description
31:16	RO	**Reserved (0)**
15:8	RO	**EHCI Extended Capabilities Pointer (EECP)** This optional field indicates the existence of a capabilities list. A value of 00h indicates no extended capabilities are implemented. A non-zero value in this register indicates the offset in PCI configuration space of the first EHCI extended capability. The pointer value must be 40h or greater if implemented to maintain the consistency of the PCI header defined for this class of device.
7:4	RO	**Isochronous Scheduling Threshold** This field indicates, relative to the current position of the executing host controller, where software can reliably update the isochronous schedule. When bit [7] is zero, the value of the least significant 3 bits indicates the number of micro-frames a host controller can hold a set of isochronous data structures (one or more) before flushing the state. When bit [7] is a one, then host software assumes the host controller may cache an isochronous data structure for an entire frame.
3	RO	**Reserved (0)**
2	RO	**Asynchronous Schedule Park Capability** If this bit is set to a one, then the host controller supports the park feature for high-speed queue heads in the Asynchronous Schedule. The feature can be disabled or enabled and set to a specific level by using the Asynchronous Schedule Park Mode Enable and Asynchronous Schedule Park Mode Count fields in the USBCMD register.
1	RO	**Programmable Frame List Flag** If this bit is set to a zero, then system software must use a frame list length of 1024 elements with this host controller. The USBCMD register Frame List Size field is a read-only register and should be set to zero. If set to a one, then system software can specify and use a smaller frame list and configure the host controller via the USBCMD register Frame List Size field. The frame list must always be aligned on a 4K page boundary. This requirement ensures that the frame list is always physically contiguous.
0	RO	**64-bit Addressing Capability** 0: data structures using 32-bit address memory pointers 1: data structures using 64-bit address memory pointers

Offset: 0Ch-1Ch BEHCI0C: Companion Port Route Description (HCSP-PORTROUTE) Init = 0x0

Bit	R/W	Description
159:0	RO	**Reserved (0)**

Offset: 20h	BEHCI20: USB Command Register (USBCMD)	Init = 0x00080002
Bit	**R/W**	**Description**
31:24	RO	**Reserved (0)**
23:16	RW	**Interrupt Threshold Control** This field is used by system software to select the maximum rate at which the host controller will issue interrupts. The only valid values are defined below. If software writes an invalid value to this register, the results are undefined. Value : Maximum Interrupt Interval 00h : Reserved 01h : 1 micro-frame 02h : 2 micro-frames 04h : 4 micro-frames 08h : 8 micro-frames (default, equates to 1 ms) 10h : 16 micro-frames (2 ms) 20h : 32 micro-frames (4 ms) 40h : 64 micro-frames (8 ms) Any other value in this register yields undefined results. Software modifications to this bit while HCHalted bit is equal to zero results in undefined behavior.
15:12	RO	**Reserved (0)**
11	RW	**Asynchronous Schedule Park Mode Enable** If the Asynchronous Park Capability bit in the HCCPARAMS register is a one, then this bit defaults to a 1h and is R/W. Otherwise the bit must be a zero and is RO. Software uses this bit to enable or disable Park mode. When this bit is one, Park mode is enabled. When this bit is a zero, Park mode is disabled.
10	RO	**Reserved (0)**
9:8	RW	**Asynchronous Schedule Park Mode Count** If the Asynchronous Park Capability bit in the HCCPARAMS register is a one, then this field defaults to 3h and is R/W. Otherwise it defaults to zero and is RO. It contains a count of the number of successive transactions the host controller is allowed to execute from a high-speed queue head on the Asynchronous schedule before continuing traversal of the Asynchronous schedule. Valid values are 1h to 3h. Software must not write a zero to this bit when Park Mode Enable is a one as this will result in undefined behavior.
7	RO	**Reserved (0)**
6	RW	**Interrupt on Async Advance Doorbell** This bit is used as a doorbell by software to tell the host controller to issue an interrupt the next time it advances asynchronous schedule. Software must write a 1 to this bit to ring the doorbell. When the host controller has evicted all appropriate cached schedule state, it sets the Interrupt on Async Advance status bit in the USBSTS register. If the Interrupt on Async Advance Enable bit in the USBINTR register is a one then the host controller will assert an interrupt at the next interrupt threshold. The host controller sets this bit to a zero after it has set the Interrupt on Async Advance status bit in the USBSTS register to a one. Software should not write a one to this bit when the asynchronous schedule is disabled. Doing so will yield undefined results.

to next page

from previous page

5	RW	**Asynchronous Schedule Enable** This bit controls whether the host controller skips processing the Asynchronous Schedule. Values mean: 0: Do not process the Asynchronous Schedule 1: Use the ASYNCLISTADDR register to access the Asynchronous Schedule.
4	RW	**Periodic Schedule Enable** This bit controls whether the host controller skips processing the Periodic Schedule. Values mean: 0: Do not process the Periodic Schedule 1: Use the PERIODICLISTBASE register to access the Periodic Schedule.
3:2	RW	**List Size** This field specifies the size of the frame list. The size the frame list controls which bits in the Frame Index Register should be used for the Frame List Current index. Values mean: 00: 1024 elements (4096 bytes) Default value 01: 512 elements (2048 bytes) 10: 256 elements (1024 bytes) – for resource-constrained environments 11: Reserved
1	RW	**Host Controller Reset (HCRESET)** This control bit is used by software to reset the host controller. The effects of this on Root Hub registers are similar to a Chip Hardware Reset. When software writes a one to this bit, the Host Controller resets its internal pipelines, timers, counters, state machines, etc. to their initial value. Any transaction currently in progress on USB is immediately terminated. A USB reset is not driven on downstream ports. PCI Configuration registers are not affected by this reset. All operational registers, including port registers and port state machines are set to their initial values. Port ownership reverts to the companion host controller(s). Software must reinitialize the host controller in order to return the host controller to an operational state. This bit is set to zero by the Host Controller when the reset process is complete. Software cannot terminate the reset process early by writing a zero to this register. Software should not set this bit to a one when the HCHalted bit in the USBSTS register is a zero. Attempting to reset an actively running host controller will result in undefined behavior.
0	RW	**Run/Stop (RS)** 0=Stop. 1=Run. When set to a 1, the Host Controller proceeds with execution of the schedule. The Host Controller continues execution as long as this bit is set to a 1. When this bit is set to 0, the Host Controller completes the current and any actively pipelined transactions on the USB and then halts. The Host Controller must halt within 16 micro-frames after software clears the Run bit. The HC Halted bit in the status register indicates when the Host Controller has finished its pending pipelined transactions and has entered the stopped state. Software must not write a one to this field unless the host controller is in the Halted state (i.e. HCHalted in the USBSTS register is a one). Doing so will yield undefined results.

Offset: 24h	**BEHCI24: USB Status Register (USBSTS)**	**Init = 0x00001000**
Bit	**R/W**	**Description**
31:16	RO	**Reserved (0)**

to next page

from previous page

15	RO	**Asynchronous Schedule Status** The bit reports the current real status of the Asynchronous Schedule. If this bit is a zero then the status of the Asynchronous Schedule is disabled. If this bit is a one then the status of the Asynchronous Schedule is enabled. The Host Controller is not required to immediately disable or enable the Asynchronous Schedule when software transitions the Asynchronous Schedule Enable bit in the USBCMD register. When this bit and the Asynchronous Schedule Enable bit are the same value, the Asynchronous Schedule is either enabled (1) or disabled (0).
14	RO	**Periodic Schedule Status** The bit reports the current real status of the Periodic Schedule. If this bit is a zero then the status of the Periodic Schedule is disabled. If this bit is a one then the status of the Periodic Schedule is enabled. The Host Controller is not required to immediately disable or enable the Periodic Schedule when software transitions the Periodic Schedule Enable bit in the USBCMD register. When this bit and the Periodic Schedule Enable bit are the same value, the Periodic Schedule is either enabled (1) or disabled (0).
13	RO	**Reclamation** This is a read-only status bit, which is used to detect an empty asynchronous schedule.
12	RO	**HCHalted bit** This bit is a zero whenever the Run/Stop bit is a one. The Host Controller sets this bit to one after it has stopped executing as a result of the Run/Stop bit being set to 0, either by software or by the Host Controller hardware (e.g. internal error).
11:6	RO	**Reserved (0)**
5	RW	**Interrupt on Async Advance (WC)** System software can force the host controller to issue an interrupt the next time the host controller advances the asynchronous schedule by writing a one to the Interrupt on Async Advance Doorbell bit in the USBCMD register. This status bit indicates the assertion of that interrupt source.
4	RW	**Host System Error (WC)** The Host Controller sets this bit to 1 when a serious error occurs during a host system access involving the Host Controller module. In a PCI system, conditions that set this bit to 1 include PCI Parity error, PCI Master Abort, and PCI Target Abort. When this error occurs, the Host Controller clears the Run/Stop bit in the Command register to prevent further execution of the scheduled TDs.
3	RW	**Frame List Rollover (WC)** The Host Controller sets this bit to a one when the Frame List Index rolls over from its maximum value to zero. The exact value at which the rollover occurs depends on the frame list size. For example, if the frame list size (as programmed in the Frame List Size field of the USBCMD register) is 1024, the Frame Index Register rolls over every time FRINDEX[13] toggles. Similarly, if the size is 512, the Host Controller sets this bit to a one every time FRINDEX[12] toggles.
2	RW	**Port Change Detect (WC)** The Host Controller sets this bit to a one when any port for which the Port Owner bit is set to zero has a change bit transition from a zero to a one or a Force Port Resume bit transition from a zero to a one as a result of a J-K transition detected on a suspended port. This bit will also be set as a result of the Connect Status Change being set to a one after system software has relinquished ownership of a connected port by writing a one to a port's Port Owner bit. This bit is allowed to be maintained in the Auxiliary power well. Alternatively, it is also acceptable that on a D3 to D0 transition of the EHCI HC device, this bit is loaded with the OR of all of the PORTSC change bits (including: Force port resume, over-current change, enable/disable change and connect status change).

to next page

from previous page

1	RW	**USB Error Interrupt (WC)** The Host Controller sets this bit to 1 when completion of a USB transaction results in an error condition (e.g., error counter underflow). If the TD on which the error interrupt occurred also had its IOC bit set, both this bit and USBINT bit are set.
0	RW	**USB Interrupt (USBINT) (WC)** The Host Controller sets this bit to 1 on the completion of a USB transaction, which results in the retirement of a Transfer Descriptor that had its IOC bit set. The Host Controller also sets this bit to 1 when a short packet is detected (actual number of bytes received was less than the expected number of bytes).
Note : WC : means this status is write '1' clear, write '0' has no any effect. Each status bit of this register is set by H/W automatically when the corresponding event occurred or the transaction finished. The corresponding interrupt enable bit wont impact the setting of the bit. The enable bit only determines the interrupt generation. Whenever S/W has finished the handling of the status bit, it must clear the status bit by writing '1', else it will not be able to recognize new events.		

Offset: 28h **BEHCI28: USB Interrupt Enable Register (USBINTR)** **Init = 0**

Bit	R/W	Description
31:6	RO	**Reserved (0)**
5	RW	**Interrupt on Async Advance Enable** When this bit is a one, and the Interrupt on Async Advance bit in the USBSTS register is a one, the host controller will issue an interrupt at the next interrupt threshold. The interrupt is acknowledged by software clearing the Interrupt on Async Advance bit.
4	RW	**Host System Error Enable** When this bit is a one, and the Host System Error Status bit in the USBSTS register is a one, the host controller will issue an interrupt. The interrupt is acknowledged by software clearing the Host System Error bit.
3	RW	**Frame List Rollover Enable** When this bit is a one, and the Frame List Rollover bit in the USBSTS register is a one, the host controller will issue an interrupt. The interrupt is acknowledged by software clearing the Frame List Rollover bit.
2	RW	**Port Change Interrupt Enable** When this bit is a one, and the Port Change Detect bit in the USBSTS register is a one, the host controller will issue an interrupt. The interrupt is acknowledged by software clearing the Port Change Detect bit.
1	RW	**USB Error Interrupt Enable** When this bit is a one, and the USBERRINT bit in the USBSTS register is a one, the host controller will issue an interrupt at the next interrupt threshold. The interrupt is acknowledged by software clearing the USBERRINT bit.
0	RW	**USB Interrupt Enable** When this bit is a one, and the USBINT bit in the USBSTS register is a one, the host controller will issue an interrupt at the next interrupt threshold. The interrupt is acknowledged by software clearing the USBINT bit.

Offset: 2Ch **BEHCI2C: Frame Index Register (FRINDEX)** **Init = 0**

Bit	R/W	Description
31:14	RO	**Reserved (0)**

to next page

from previous page

13:0	RW	**Frame Index** The value in this register increments at the end of each time frame (e.g. micro-frame). Bits [N:3] are used for the Frame List current index. This means that each location of the frame list is accessed 8 times (frames or micro-frames) before moving to the next index. The following illustrates values of N based on the value of the Frame List Size field in the USBCMD register. USBCMD[Frame List Size] Number Elements N 00 (1024) 12 01 (512) 11 10 (256) 10 11 Reserved

Offset: 34h BEHCI34: Periodic Frame List Base Address Register (PERIODICLISTBASE) Init = X

Bit	R/W	Description
31:12	RW	**Base Address** These bits correspond to memory address signals [31:12], respectively.
11:0	RO	**Reserved (0)**

Offset: 38h BEHCI38: Current Asynchronous List Address Register (ASYNCLISTADDR) Init = X

Bit	R/W	Description
31:5	RW	**Link Pointer Low (LPL)** These bits correspond to memory address signals [31:5], respectively. This field may only reference a Queue Head (QH).
4:0	RO	**Reserved (0)**

Offset: 60h BEHCI60: Configure Flag Register (CONFIGFLAG) Init = 0

Bit	R/W	Description
31:1	RO	**Reserved (0)**
0	RW	**Configure Flag (CF)** Host software sets this bit as the last action in its process of configuring the Host Controller (see Section 4.1). This bit controls the default port-routing control logic. Bit values and side-effects are listed below. **0:** Port routing control logic default-routes each port to an implementation dependent classic host controller. **1:** Port routing control logic default-routes all ports to this host controller.

Offset: 64h BEHCI64: Port1 Status/Control Register (PORTSC1) Init = 0x0000_3000

31:22	RO	**Reserved (0)**
21	RW	**Wake on Disconnect Enable (WKDSCNNT_E)** Writing this bit to a one enables the port to be sensitive to device disconnects as wake-up events.
20	RW	**Wake on Connect Enable (WKCNNT_E)** Writing this bit to a one enables the port to be sensitive to device connects as wake-up events. See Section 4.3 for effects of this bit on resume event behavior. Refer to Section 4.3.1 for operational model.

19:14	RO	**Reserved (0)**
13	RW	**Port Owner** This bit unconditionally goes to a 0b when the Configured bit in the CONFIGFLAG register makes a 0b to 1b transition. This bit unconditionally goes to 1b whenever the Configured bit is zero. System software uses this field to release ownership of the port to a selected host controller (in the event that the attached device is not a high-speed device). Software writes a one to this bit when the attached device is not a high-speed device. A one in this bit means that a companion host controller owns and controls the port.
12	RO	**Reserved (1)**
11:10	RO	**Line Status** These bits reflect the current logical levels of the D+ (bit 11) and D- (bit 10) signal lines. These bits are used for detection of low-speed USB devices prior to the port reset and enable sequence. This field is valid only when the port enable bit is zero and the current connect status bit is set to a one. The encoding of the bits are: Bits[11:10] USB State Interpretation 00b SE0 Not Low-speed device, perform EHCI reset 10b J-state Not Low-speed device, perform EHCI reset 01b K-state Low-speed device, release ownership of port 11b Undefined Not Low-speed device, perform EHCI reset.
9	RO	**Reserved (0)**
8	RW	**Port Reset** 0: Port is not in Reset. 1: Port is in Reset. When software writes a one to this bit (from a zero), the bus reset sequence as defined in the USB Specification Revision 2.0 is started. Software writes a zero to this bit to terminate the bus reset sequence. Software must keep this bit at a one long enough to ensure the reset sequence, as specified in the USB Specification Revision 2.0, completes. Note: when software writes this bit to a one, it must also write a zero to the Port Enable bit. Note that when software writes a zero to this bit there may be a delay before the bit status changes to a zero. The bit status will not read as a zero until after the reset has completed. If the port is in high-speed mode after reset is complete, the host controller will automatically enable this port (e.g. set the Port Enable bit to a one). A host controller must terminate the reset and stabilize the state of the port within 2 milliseconds of software transitioning this bit from a one to a zero. For example: if the port detects that the attached device is high-speed during reset, then the host controller must have the port in the enabled state within 2ms of software writing this bit to a zero. The HCHalted bit in the USBSTS register should be a zero before software attempts to use this bit. The host controller may hold Port Reset asserted to a one when the HCHalted bit is a one.

7	RW	**Suspend** 0: Port not in suspend state. 1: Port in suspend state. Port Enabled Bit and Suspend bit of this register define the port states as follows: Bits [Port Enabled, Suspend] Port State 0X Disable 10 Enable 11 Suspend When in suspend state, downstream propagation of data is blocked on this port, except for port reset. The blocking occurs at the end of the current transaction, if a transaction was in progress when this bit was written to 1. In the suspend state, the port is sensitive to resume detection. Note that the bit status does not change until the port is suspended and that there may be a delay in suspending a port if there is a transaction currently in progress on the USB. A write of zero to this bit is ignored by the host controller. The host controller will unconditionally set this bit to a zero when: – Software sets the Force Port Resume bit to a zero (from a one). – Software sets the Port Reset bit to a one (from a zero). If host software sets this bit to a one when the port is not enabled (i.e. Port enabled bit is a zero) the results are undefined.
6	RW	**Force Port Resume** 0: No resume (K-state) detected/driven on port. 1: Resume detected/driven on port. This functionality defined for manipulating this bit depends on the value of the Suspend bit. For example, if the port is not suspended (Suspend and Enabled bits are a one) and software transitions this bit to a one, then the effects on the bus are undefined. Software sets this bit to a 1 to drive resume signaling. The Host Controller sets this bit to a 1 if a J-to-K transition is detected while the port is in the Suspend state. When this bit transitions to a one because a J-to-K transition is detected, the Port Change Detect bit in the USBSTS register is also set to a one. If software sets this bit to a one, the host controller must not set the Port Change Detect bit. Note that when the EHCI controller owns the port, the resume sequence follows the defined sequence documented in the USB Specification Revision 2.0. The resume signaling (Full-speed 'K') is driven on the port as long as this bit remains a one. Software must appropriately time the Resume and set this bit to a zero when the appropriate amount of time has elapsed. Writing a zero (from one) causes the port to return to high- speed mode (forcing the bus below the port into a high-speed idle). This bit will remain a one until the port has switched to the high-speed idle. The host controller must complete this transition within 2 milliseconds of software setting this bit to a zero.
5:4	RO	**Reserved (0)**
3	RW	**Port Enable/Disable Change (WC)** 0: No change. 1: Port enabled/disabled status has changed. For the root hub, this bit gets set to a one only when a port is disabled due to the appropriate conditions existing at the EOF2 point (See Chapter 11 of the USB Specification for the definition of a Port Error). Software clears this bit by writing a 1 to it.

2	RW	**Port Enabled/Disabled** 0: Disable. 1: Enable. Ports can only be enabled by the host controller as a part of the reset and enable. Software cannot enable a port by writing a one to this field. The host controller will only set this bit to a one when the reset sequence determines that the attached device is a high-speed device. Ports can be disabled by either a fault condition (disconnect event or other fault condition) or by host software. Note that the bit status does not change until the port state actually changes. There may be a delay in disabling or enabling a port due to other host controller and bus events. When the port is disabled (0b) downstream propagation of data is blocked on this port, except for reset.
1	RW	**Connect Status Change (WC)** 0: No change. 1: Change in Current Connect Status. Indicates a change has occurred in the ports Current Connect Status. The host controller sets this bit for all changes to the port device connect status, even if system software has not cleared an existing connect status change. For example, the insertion status changes twice before system software has cleared the changed condition, hub hardware will be ”setting” an already-set bit (i.e., the bit will remain set). Software sets this bit to 0 by writing a 1 to it.
0	RO	**Current Connect Status** 0: No device is present. 1: Device is present on port. This value reflects the current state of the port, and may not correspond directly to the event that caused the Connect Status Change bit (Bit 1) to be set.

Offset: 80h BEHCI80: (AHB Only)Frame Length Adjustment Register (FLADJ) Init = 0x0000_0020

Bit	R/W	Description
31:30	RO	**PCIe bus DMA Error Status** This is a debug status.
29:6	RO	**Reserved (0)**
5:0	RO	**Frame Length Timing Value** Each decimal value change to this register corresponds to 16 high-speed bit times. The SOF cycle time (number of SOF counter clock periods to generate a SOF micro-frame length) is equal to 59488 + value in this field. The default value is decimal 32 (20h), which gives a SOF cycle time of 60000. Frame Length (# High Speed bit times) (decimal) / FLADJ Value (decimal) 59488 — 0 (00h) 59504 — 1 (01h) 59520 — 2 (02h) ... 59984 — 31 (1Fh) 60000 — 32 (20h) ... 60480 — 62 (3Eh) 60496 — 63 (3Fh)

Offset: 84h	BEHCI84: (AHB Only)Controller Fine-tune Register	Init = 0x0000_0A47
Bit	**R/W**	**Description**
31:21	RO	**Reserved (0)**
20	RW	**Clear DMA Error Status**
19:12	RW	**Insert wait delay between IN transactions with Mult > 1** Inserted wait time = UsbClock * delay * 2.
11	RW	**Enable support 64 bit address mode** This bit value reflect to HCCPARAMS bit[0].
10	RW	**Enable FIFO auto power down** 0: Disable (default) 1: Enable
9	RW	**Reserved**
8	RW	**High speed Isochronous IN MaxPacketSize selection** 0 : MaxPacketSize is determined by Transaction_X_Length or Maximum Packet Size (whichever is less). 1 : MaxPacketSize is determined by Maximum Packet Size field.
7:6	RW	**Transmit FIFO Threshold** 00 : 128 bytes 01 : 256 bytes 10 : 512 bytes 11 : 768 bytes
5:2	RW	**Isochronous Scheduling Threshold** This field value reflect to HCCPARAMS bit[7:4].
1	RW	**Asynchronous Schedule Park Capability** This bit value reflect to HCCPARAMS bit[2].
0	RW	**Programmable Frame List Flag** This bit value reflect to HCCPARAMS bit[1].

Offset: 88h	BEHCI88: (AHB Only)Frame Timing Adjustment	Init = 0x40100000
Bit	**R/W**	**Description**
31:22	RW	**preEOF1 timing** EOF1 timing point before SOF is the last executable transaction time, the actual EOF1 time calculation equals MaxPacketSize + preEOF1. The unit is byte.
21:12	RW	**preEOF2 timing** EOF2 timing point before SOF is the time for host controller to stop processing list structure. The unit is byte.
11:0	RW	**SOF transmit delay timing** Setting this field to delay the SOF packet transmission after actual internal SOF timing point. This is used to extend EOF2 timing. The unit is byte.

Offset: 8Ch	BEHCI8C: (AHB Only)Hardware Revision Number Register	Init = 0x0000_0002
Bit	**R/W**	**Description**
31:8	RO	**Reserved (0)**
7:0	RO	**Hardware revision number**

74 PCIe Host to BMC Controller (HOST2BMC)

The primary purpose of this function is to exchange messages and data between the BMC and the Host processor over PCIe link. Along with the messages a large memory area is shared to exchange data between the processors using the messaging interface. Only memory writes/memory reads from the Host processor to the BMC is supported. On enabling this feature the AST2600 will show up as a second device with first device being the graphics controller. The Host to BMC Device interface is described below in detail.

74.1 Overview

The Host processor accesses the BMC memory using the memory allocated by the Host System BIOS for the two memory BARs exposed by this function. The BAR0 is used by the HOST to access the BMC memory. The BAR1 is used to access the messaging queues between the HOST and BMC. Please note that the host to bmc device supports only 4byte aligned memory accesses.

74.2 PCIe Host to BMC Device Registers

Please refer to ”Host to BMC Device” document for more detail.

75 PCIe to MailBox (PCIE2MBOX)

75.1 Overview

There are 32 PCIe general purpose mailbox registers implemented in each node. They can be used to communicate or send messages between the Host and the BMC. Also, the interrupts of signaling the opposite processor are introduced to ease the loading.

The following registers can be accessed by the Host and the BMC. Here shows the descriptions for the Host.

The host side XMailBox control and base address registers are defined in below XSuperIO registers:

XSIORE_30
XSIORE_60
XSIORE_70
XSIORE_71
XSIORE_F0

XSMBXDAT_0: XMailBox Data Register 0
XSMBXDAT_1: XMailBox Data Register 1
XSMBXDAT_2: XMailBox Data Register 2
XSMBXDAT_3: XMailBox Data Register 3
XSMBXDAT_4: XMailBox Data Register 4
XSMBXDAT_5: XMailBox Data Register 5
XSMBXDAT_6: XMailBox Data Register 6
XSMBXDAT_7: XMailBox Data Register 7
XSMBXDAT_8: XMailBox Data Register 8
XSMBXDAT_9: XMailBox Data Register 9
XSMBXDAT_A: XMailBox Data Register A
XSMBXDAT_B: XMailBox Data Register B
XSMBXDAT_C: XMailBox Data Register C
XSMBXDAT_D: XMailBox Data Register D
XSMBXDAT_E: XMailBox Data Register E
XSMBXDAT_F: XMailBox Data Register F

XSMBXDAT_10: XMailBox Data Register 10
XSMBXDAT_11: XMailBox Data Register 11
XSMBXDAT_12: XMailBox Data Register 12
XSMBXDAT_13: XMailBox Data Register 13
XSMBXDAT_14: XMailBox Data Register 14
XSMBXDAT_15: XMailBox Data Register 15
XSMBXDAT_16: XMailBox Data Register 16
XSMBXDAT_17: XMailBox Data Register 17
XSMBXDAT_18: XMailBox Data Register 18
XSMBXDAT_19: XMailBox Data Register 19
XSMBXDAT_1A: XMailBox Data Register 1A
XSMBXDAT_1B: XMailBox Data Register 1B
XSMBXDAT_1C: XMailBox Data Register 1C
XSMBXDAT_1D: XMailBox Data Register 1D
XSMBXDAT_1E: XMailBox Data Register 1E
XSMBXDAT_1F: XMailBox Data Register 1F

XSMBXSTS_0: XMailBox Status Register 0
XSMBXSTS_1: XMailBox Status Register 1

XSMBXSTS_2: XMailBox Status Register 2
XSMBXSTS_3: XMailBox Status Register 3
XSMBXBCR : XMailBox BMC Control Register
XSMBXHCR : XMailBox Host Control Register
XSMBXBIE_0: XMailBox BMC Interrupt Enable Register 0
XSMBXBIE_1: XMailBox BMC Interrupt Enable Register 1
XSMBXBIE_2: XMailBox BMC Interrupt Enable Register 2
XSMBXBIE_3: XMailBox BMC Interrupt Enable Register 3
XSMBXHIE_0: XMailBox Host Interrupt Enable Register 0
XSMBXHIE_1: XMailBox Host Interrupt Enable Register 1
XSMBXHIE_2: XMailBox Host Interrupt Enable Register 2
XSMBXHIE_3: XMailBox Host Interrupt Enable Register 3

Attribute Definition:

Attribute	: Description
R	: Readable
W	: Writable
W1C	: Write '1' to clear value to 0
X	: Unknown value
P	: Initialized by PWRST_N

XSMBXDAT_0: XMailBox Data Register 0 **Offset: 00**

Bit	Name	Initial	Slave	Host	Description
7:0		0P	RW	RW	**XMailBox Data Register 0 bit[7:0]**

XSMBXDAT_1: XMailBox Data Register 1 **Offset: 01**

Bit	Name	Initial	Slave	Host	Description
7:0		0P	RW	RW	**XMailBox Data Register 1 bit[7:0]**

XSMBXDAT_2: XMailBox Data Register 2 **Offset: 02**

Bit	Name	Initial	Slave	Host	Description
7:0		0P	RW	RW	**XMailBox Data Register 2 bit[7:0]**

XSMBXDAT_3: XMailBox Data Register 3 **Offset: 03**

Bit	Name	Initial	Slave	Host	Description
7:0		0P	RW	RW	**XMailBox Data Register 3 bit[7:0]**

XSMBXDAT_4: XMailBox Data Register 4 **Offset: 04**

Bit	Name	Initial	Slave	Host	Description
7:0		0P	RW	RW	**XMailBox Data Register 4 bit[7:0]**

XSMBXDAT_5: XMailBox Data Register 5 **Offset: 05**

Bit	Name	Initial	Slave	Host	Description
7:0		0P	RW	RW	**XMailBox Data Register 5 bit[7:0]**

XSMBXDAT_6: XMailBox Data Register 6 **Offset: 06**

Bit	Name	Initial	Slave	Host	Description
7:0		0P	RW	RW	**XMailBox Data Register 6 bit[7:0]**

XSMBXDAT_7: XMailBox Data Register 7 **Offset: 07**

Bit	Name	Initial	Slave	Host	Description
7:0		0P	RW	RW	**XMailBox Data Register 7 bit[7:0]**

XSMBXDAT_8: XMailBox Data Register 8 **Offset: 08**

Bit	Name	Initial	Slave	Host	Description
7:0		0P	RW	RW	**XMailBox Data Register 8 bit[7:0]**

XSMBXDAT_9: XMailBox Data Register 9 **Offset: 09**

Bit	Name	Initial	Slave	Host	Description
7:0		0P	RW	RW	**XMailBox Data Register 9 bit[7:0]**

XSMBXDAT_A: XMailBox Data Register A **Offset: 0A**

Bit	Name	Initial	Slave	Host	Description
7:0		0P	RW	RW	**XMailBox Data Register A bit[7:0]**

XSMBXDAT_B: XMailBox Data Register B **Offset: 0B**

Bit	Name	Initial	Slave	Host	Description
7:0		0P	RW	RW	**XMailBox Data Register B bit[7:0]**

XSMBXDAT_C: XMailBox Data Register C **Offset: 0C**

Bit	Name	Initial	Slave	Host	Description
7:0		0P	RW	RW	**XMailBox Data Register C bit[7:0]**

XSMBXDAT_D: XMailBox Data Register D **Offset: 0D**

Bit	Name	Initial	Slave	Host	Description
7:0		0P	RW	RW	**XMailBox Data Register D bit[7:0]**

XSMBXDAT_E: XMailBox Data Register E **Offset: 0E**

Bit	Name	Initial	Slave	Host	Description
7:0		0P	RW	RW	**XMailBox Data Register E bit[7:0]**

XSMBXDAT_F: XMailBox Data Register F **Offset: 0F**

Bit	Name	Initial	Slave	Host	Description
7:0		0P	RW	RW	**XMailBox Data Register F bit[7:0]**

XSMBXDAT_10: XMailBox Data Register 10 **Offset: 10**

Bit	Name	Initial	Slave	Host	Description
7:0		0P	RW	RW	**XMailBox Data Register 10 bit[7:0]**

XSMBXDAT_11: XMailBox Data Register 11 **Offset: 11**

Bit	Name	Initial	Slave	Host	Description
7:0		0P	RW	RW	**XMailBox Data Register 11 bit[7:0]**

XSMBXDAT_12: XMailBox Data Register 12 **Offset: 12**

Bit	Name	Initial	Slave	Host	Description
7:0		0P	RW	RW	**XMailBox Data Register 12 bit[7:0]**

XSMBXDAT_13: XMailBox Data Register 13 **Offset: 13**

Bit	Name	Initial	Slave	Host	Description
7:0		0P	RW	RW	**XMailBox Data Register 13 bit[7:0]**

XSMBXDAT_14: XMailBox Data Register 14 **Offset: 14**

Bit	Name	Initial	Slave	Host	Description
7:0		0P	RW	RW	**XMailBox Data Register 14 bit[7:0]**

XSMBXDAT_15: XMailBox Data Register 15 **Offset: 15**

Bit	Name	Initial	Slave	Host	Description
7:0		0P	RW	RW	**XMailBox Data Register 15 bit[7:0]**

XSMBXDAT_16: XMailBox Data Register 16 **Offset: 16**

Bit	Name	Initial	Slave	Host	Description
7:0		0P	RW	RW	**XMailBox Data Register 16 bit[7:0]**

XSMBXDAT_17: XMailBox Data Register 17 **Offset: 17**

Bit	Name	Initial	Slave	Host	Description
7:0		0P	RW	RW	**XMailBox Data Register 17 bit[7:0]**

XSMBXDAT_18: XMailBox Data Register 18 **Offset: 18**

Bit	Name	Initial	Slave	Host	Description
7:0		0P	RW	RW	**XMailBox Data Register 18 bit[7:0]**

XSMBXDAT_19: XMailBox Data Register 19					Offset: 19
Bit	**Name**	**Initial**	**Slave**	**Host**	**Description**
7:0		0P	RW	RW	**XMailBox Data Register 19 bit[7:0]**

XSMBXDAT_1A: XMailBox Data Register 1A					Offset: 1A
Bit	**Name**	**Initial**	**Slave**	**Host**	**Description**
7:0		0P	RW	RW	**XMailBox Data Register 1A bit[7:0]**

XSMBXDAT_1B: XMailBox Data Register 1B					Offset: 1B
Bit	**Name**	**Initial**	**Slave**	**Host**	**Description**
7:0		0P	RW	RW	**XMailBox Data Register 1B bit[7:0]**

XSMBXDAT_1C: XMailBox Data Register 1C					Offset: 1C
Bit	**Name**	**Initial**	**Slave**	**Host**	**Description**
7:0		0P	RW	RW	**XMailBox Data Register 1C bit[7:0]**

XSMBXDAT_1D: XMailBox Data Register 1D					Offset: 1D
Bit	**Name**	**Initial**	**Slave**	**Host**	**Description**
7:0		0P	RW	RW	**XMailBox Data Register 1D bit[7:0]**

XSMBXDAT_1E: XMailBox Data Register 1E					Offset: 1E
Bit	**Name**	**Initial**	**Slave**	**Host**	**Description**
7:0		0P	RW	RW	**XMailBox Data Register 1E bit[7:0]**

XSMBXDAT_1F: XMailBox Data Register 1F					Offset: 1F
Bit	**Name**	**Initial**	**Slave**	**Host**	**Description**
7:0		0P	RW	RW	**XMailBox Data Register 1F bit[7:0]**

XSMBXSTS_0: XMailBox Status Register 0					Offset: 20
Bit	**Name**	**Initial**	**Slave**	**Host**	**Description**
7:0		0P	RW1C	RW1C	**XMailBox Status Register 0 bit[7:0]**

XSMBXSTS_1: XMailBox Status Register 1					Offset: 21
Bit	**Name**	**Initial**	**Slave**	**Host**	**Description**
7:0		0P	RW1C	RW1C	**XMailBox Status Register 1 bit[15:8]**

XSMBXSTS_2: XMailBox Status Register 2					Offset: 22
Bit	**Name**	**Initial**	**Slave**	**Host**	**Description**
7:0		0P	RW1C	RW1C	**XMailBox Status Register 2 bit[23:16]**

XSMBXSTS_3: XMailBox Status Register 3 **Offset: 23**

Bit	Name	Initial	Slave	Host	Description
7:0		0P	RW1C	RW1C	**XMailBox Status Register 3 bit[31:24]**

XSMBXBCR: XMailBox BMC Control Register **Offset: 24**

Bit	Name	Initial	Slave	Host	Description
7		0P	RW1C	RO	**Interrupt status from the Host to the BMC**
6:2		0P	RO	RO	**Reserved**
1		0P	RW	RO	**Mask interrupt to the BMC from the status bit, XSMBXBCR[7]**
0		0P	RW	RO	**Generate interrupt to the Host; set the status bit, XSMBXHCR[7]**

XSMBXHCR: XMailBox Host Control Register **Offset: 25**

Bit	Name	Initial	Slave	Host	Description
7		0P	RO	RW1C	**Interrupt status from the BMC to the Host**
6:2		0P	RO	RO	**Reserved**
1		0P	RO	RW	**Mask interrupt to the Host from the status bit, XSMBXHCR[7]**
0		0P	RO	W1T	**Generate interrupt to the BMC; set the status bit, XSMBXBCR[7]**

XSMBXFCR_1: XMailBox Flow Control Register 1 **Offset: 26**

Bit	Name	Initial	Slave	Host	Description
7:0		0P	RW	RO	**Enable flow control register bit[7:0]. Host can't update data until BMC acknowledges.** bit[0]: XSMBXDAT0 bit[1]: XSMBXDAT1 ... bit[7]: XSMBXDAT7

XSMBXFCR_2: XMailBox Flow Control Register 2 **Offset: 27**

Bit	Name	Initial	Slave	Host	Description
7:0		0P	W1T	RO	**BMC acknowleges flow control register bit[7:0].** bit[0]: XSMBXDAT0 bit[1]: XSMBXDAT1 ... bit[7]: XSMBXDAT7

XSMBXBIE_0: XMailBox BMC Interrupt Enable Register 0 **Offset: 28**

Bit	Name	Initial	Slave	Host	Description
7:0		0P	RW	RO	**XMailBox BMC Interrupt Enable bit[7:0]**

XSMBXBIE_1: XMailBox BMC Interrupt Enable Register 1 **Offset: 29**

Bit	Name	Initial	Slave	Host	Description
7:0		0P	RW	RO	**XMailBox BMC Interrupt Enable bit[15:8]**

XSMBXBIE_2: XMailBox BMC Interrupt Enable Register 2 **Offset: 2A**

Bit	Name	Initial	Slave	Host	Description
7:0		0P	RW	RO	**XMailBox BMC Interrupt Enable bit[23:16]**

XSMBXBIE_3: XMailBox BMC Interrupt Enable Register 3 **Offset: 2B**

Bit	Name	Initial	Slave	Host	Description
7:0		0P	RW	RO	**XMailBox BMC Interrupt Enable bit[31:24]**

XSMBXHIE_0: XMailBox Host Interrupt Enable Register 0 **Offset: 2C**

Bit	Name	Initial	Slave	Host	Description
7:0		0P	RO	RW	**XMailBox Host Interrupt Enable bit[7:0]**

XSMBXHIE_1: XMailBox Host Interrupt Enable Register 1 **Offset: 2D**

Bit	Name	Initial	Slave	Host	Description
7:0		0P	RO	RW	**XMailBox Host Interrupt Enable bit[15:8]**

XSMBXHIE_2: XMailBox Host Interrupt Enable Register 2 **Offset: 2E**

Bit	Name	Initial	Slave	Host	Description
7:0		0P	RO	RW	**XMailBox Host Interrupt Enable bit[23:16]**

XSMBXHIE_3: XMailBox Host Interrupt Enable Register 3 **Offset: 2F**

Bit	Name	Initial	Slave	Host	Description
7:0		0P	RO	RW	**XMailBox Host Interrupt Enable bit[31:24]**

76 PCIe to SuperIO (PCIE2SIO)

76.1 Overview

AST2600 integrates a XSuper I/O module behind PCIe BMC Device (MMIO cycle 0xB8/0xBC). There is a logical device with Mailbox functino:

1. Mailbox (logical device E)

The logical device has its own configuration register above index 0x30.

The following registers can be access by host CPU through PCIe bus.

XSIORP : XSuper IO Password Register
XSIORx_07: XSuper IO Register x_07
XSIORx_20: XSuper IO Register x_20
XSIORx_21: XSuper IO Register x_21
XSIORx_22: XSuper IO Register x_22
XSIORx_23: XSuper IO Register x_23
XSIORx_24: XSuper IO Register x_24
XSIORx_25: XSuper IO Register x_25
XSIORx_26: XSuper IO Register x_26
XSIORx_27: XSuper IO Register x_27
XSIORx_28: XSuper IO Register x_28
XSIORx_29: XSuper IO Register x_29
XSIORx_2A: XSuper IO Register x_2A
XSIORx_2B: XSuper IO Register x_2B
XSIORx_2C: XSuper IO Register x_2C
XSIORx_2D: XSuper IO Register x_2D
XSIORx_2E: XSuper IO Register x_2E
XSIORx_2F: XSuper IO Register x_2F

XSIORE_30: XSuper IO Register E_30
XSIORE_60: XSuper IO Register E_60
XSIORE_70: XSuper IO Register E_70
XSIORE_71: XSuper IO Register E_71

Attribute Definition:

Attribute : Description	
R	: Readable
W	: Writable
W1C	: Write '1' to clear value to 0
X	: Unknown value
P	: Initialized by PWRST_N

Offset: B8/138h		**XSIORP: XSuper IO Password Register**	**Init = 0**
Bit	**R/W**	**Description**	
7:0	WT	**Enable by writing 0xA5 twice; Disable by writing 0xAA once**	

Offset: 07h	XSIORx_07: XSuper IO Register x_07	Init = 0
Bit	**R/W**	**Description**
7:4	RO	**Reserved**
3:0	RW	**Logical device number**

Offset: 20h	XSIORx_20: XSuper IO Register x_20	Init = 26P
Bit	**R/W**	**Description**
7:0	RO	**SIO identification bit[7:0]**

Offset: 21h	XSIORx_21: XSuper IO Register x_21	Init = 0P
Bit	**R/W**	**Description**
3:0	RW	**XSIO to BMC software interrupt bit[3:0] by rising trigger**
7:4	RW1C	**BMC to XSIO software interrupt status bit[3:0] by writing one clear**

Offset: 22h	XSIORx_22: XSuper IO Register x_22	Init = 0P
Bit	**R/W**	**Description**
7:0	RW	**XSIO to BMC scratch register 2 bit[7:0]**

Offset: 23h	XSIORx_23: XSuper IO Register x_23	Init = 0P
Bit	**R/W**	**Description**
7:0	RW	**XSIO to BMC scratch register 3 bit[7:0]**

Offset: 24h	XSIORx_24: XSuper IO Register x_24	Init = 0P
Bit	**R/W**	**Description**
7:0	RW	**XSIO to BMC scratch register 4 bit[7:0]**

Offset: 25h	XSIORx_25: XSuper IO Register x_25	Init = 0P
Bit	**R/W**	**Description**
7:0	RW	**XSIO to BMC scratch register 5 bit[7:0]**

Offset: 26h	XSIORx_26: XSuper IO Register x_26	Init = 0P
Bit	**R/W**	**Description**
7:0	RW	**XSIO to BMC scratch register 6 bit[7:0]**

Offset: 27h	XSIORx_27: XSuper IO Register x_27	Init = 0P
Bit	**R/W**	**Description**
7:0	RW	**XSIO to BMC scratch register 7 bit[7:0]**

Offset: 28h	XSIORx_28: XSuper IO Register x_28	Init = A8P
Bit	**R/W**	**Description**
7:0	RO	**BMC to XSIO scratch register 0 bit[7:0]**

Offset: 29h	XSIORx_29: XSuper IO Register x_29	Init = 0P
Bit	**R/W**	**Description**
7:0	RO	**BMC to XSIO scratch register 1 bit[7:0]**

Offset: 2Ah	XSIORx_2A: XSuper IO Register x_2A	Init = 0P
Bit	**R/W**	**Description**
7:0	RO	**BMC to XSIO scratch register 2 bit[7:0]**

Offset: 2Bh	XSIORx_2B: XSuper IO Register x_2B	Init = 0P
Bit	**R/W**	**Description**
7:0	RO	**BMC to XSIO scratch register 3 bit[7:0]**

Offset: 2Ch	XSIORx_2C: XSuper IO Register x_2C	Init = 0P
Bit	**R/W**	**Description**
7:0	RO	**BMC to XSIO scratch register 4 bit[7:0]**

Offset: 2Dh	XSIORx_2D: XSuper IO Register x_2D	Init = 0P
Bit	**R/W**	**Description**
7:0	RO	**BMC to XSIO scratch register 5 bit[7:0]**

Offset: 2Eh	XSIORx_2E: XSuper IO Register x_2E	Init = 0P
Bit	**R/W**	**Description**
7:0	RO	**BMC to XSIO scratch register 6 bit[7:0]**

Offset: 2Fh	XSIORx_2F: XSuper IO Register x_2F	Init = 0P
Bit	**R/W**	**Description**
7:0	RO	**BMC to XSIO scratch register 7 bit[7:0]**

Offset: 30h	XSIORE_30: XSuper IO Register E_30	Init = 0
Bit	**R/W**	**Description**
7:1	RW	**Reserved**
0	RW	**Enable Mailbox**

Offset: 60h	XSIORE_60: XSuper IO Register E_60	Init = 0CP
Bit	**R/W**	**Description**
7:6	RW	**Reserved**
5:0	RW	**Mailbox MMIO base offset address bit[15:10] behind PCIe BMC Device BAR1**

Offset: 70h	XSIORE_70: XSuper IO Register E_70	Init = 09P
Bit	**R/W**	**Description**
7:4	RO	**Reserved**
3:0	RW	**PCIe: Select ID bit[3:0] of MSI for Mailbox**

Offset: 71h	XSIORE_71: XSuper IO Register E_71	Init = 01P
Bit	**R/W**	**Description**
7:2	RO	**Reserved**
1:0	RW	**Host MSI interrupt type for Mailbox** 00: BIOS setting in PCIe mode

77 PCI Express Authentication (PCIEATH)

77.1 Overview

AST2600 PCIE device security enhancements based on the
https://www.intel.com/content/dam/www/public/us/en/documents/reference-guides/pcie-dev
Please refer to this document for background and details.

Base Address of PCIe Device Security Enhancement 8KB SRAM = 0x1E7F_0000
Address Space of PCIe Device Security Enhancement 8KB SRAM = from 0x1E7F_0000 to 0x1E7F_1FFF

Base Address of PCIe Device Security Enhancement Controller = 0x1E7F_2000
Physical address of register = (Base address of PCIe Device Security Enhancement Controller) + Offset

PDSE00: PCIe Device Security Enhancement Control Register 00
PDSE04: PCIe Device Security Enhancement Control Register 04
PDSE08: PCIe Device Security Enhancement Control Register 08
PDSE0C: PCIe Device Security Enhancement Control Register 0C
PDSE10: PCIe Device Security Enhancement Control Register 10
PDSE14: PCIe Device Security Enhancement Control Register 14
PDSE18: PCIe Device Security Enhancement Control Register 18
PDSE1C: PCIe Device Security Enhancement Control Register 1C
PDSE20: PCIe Device Security Enhancement Control Register 20
PDSE24: PCIe Device Security Enhancement Control Register 24
PDSE28: PCIe Device Security Enhancement Control Register 28
PDSE2C: PCIe Device Security Enhancement Control Register 2C
PDSE30: PCIe Device Security Enhancement Control Register 30
PDSE34: PCIe Device Security Enhancement Control Register 34
PDSE38: PCIe Device Security Enhancement Control Register 38
PDSE3C: PCIe Device Security Enhancement Control Register 3C
PDSE40: PCIe Device Security Enhancement Control Register 40
PDSE44: PCIe Device Security Enhancement Control Register 44
PDSE48: PCIe Device Security Enhancement Control Register 48
PDSE4C: PCIe Device Security Enhancement Control Register 4C
PDSE50: PCIe Device Security Enhancement Control Register 50
PDSE54: PCIe Device Security Enhancement Control Register 54
PDSE58: PCIe Device Security Enhancement Control Register 58
PDSE5C: PCIe Device Security Enhancement Control Register 5C
PDSE60: PCIe Device Security Enhancement Control Register 60
PDSE64: PCIe Device Security Enhancement Control Register 64
PDSE68: PCIe Device Security Enhancement Control Register 68
PDSE6C: PCIe Device Security Enhancement Control Register 6C
PDSE70: PCIe Device Security Enhancement Control Register 70
PDSE74: PCIe Device Security Enhancement Control Register 74
PDSE78: PCIe Device Security Enhancement Control Register 78
PDSE7C: PCIe Device Security Enhancement Control Register 7C
PDSE80: PCIe Device Security Enhancement Control Register 80
PDSE84: PCIe Device Security Enhancement Control Register 84
PDSE88: PCIe Device Security Enhancement Control Register 88
PDSE8C: PCIe Device Security Enhancement Control Register 8C
PDSE90: PCIe Device Security Enhancement Control Register 90
PDSE94: PCIe Device Security Enhancement Control Register 94
PDSE98: PCIe Device Security Enhancement Control Register 98
PDSE9C: PCIe Device Security Enhancement Control Register 9C

PDSEA0: PCIe Device Security Enhancement Control Register A0
PDSEA4: PCIe Device Security Enhancement Control Register A4
PDSEA8: PCIe Device Security Enhancement Control Register A8
PDSEAC: PCIe Device Security Enhancement Control Register AC
PDSEB0: PCIe Device Security Enhancement Control Register B0
PDSEB4: PCIe Device Security Enhancement Control Register B4
PDSEB8: PCIe Device Security Enhancement Control Register B8
PDSEBC: PCIe Device Security Enhancement Control Register BC
PDSEC0: PCIe Device Security Enhancement Control Register C0
PDSEC4: PCIe Device Security Enhancement Control Register C4
PDSEC8: PCIe Device Security Enhancement Control Register C8
PDSECC: PCIe Device Security Enhancement Control Register CC
PDSED0: PCIe Device Security Enhancement Control Register D0
PDSED4: PCIe Device Security Enhancement Control Register D4
PDSED8: PCIe Device Security Enhancement Control Register D8
PDSEDC: PCIe Device Security Enhancement Control Register DC
PDSEE0: PCIe Device Security Enhancement Control Register E0
PDSEE4: PCIe Device Security Enhancement Control Register E4
PDSEE8: PCIe Device Security Enhancement Control Register E8
PDSEEC: PCIe Device Security Enhancement Control Register EC
PDSEF0: PCIe Device Security Enhancement Control Register F0
PDSEF4: PCIe Device Security Enhancement Control Register F4
PDSEF8: PCIe Device Security Enhancement Control Register F8
PDSEFC: PCIe Device Security Enhancement Control Register FC

77.2 Registers : Base Address = 0x1E7F:2000

Offset: 00h PDSE00: PCIe Device Security Enhancement Control Register 00 Init = 0x2C010023

Bit	R/W	Description
31:0	RW	**PCIe DVSEC (03Eh) Header 1**

Offset: 04h PDSE04: PCIe Device Security Enhancement Control Register 04 Init = 0x05018086

Bit	R/W	Description
31:0	RW	**PCIe DVSEC (03Eh) Header 2**

Offset: 08h PDSE08: PCIe Device Security Enhancement Control Register 08 Init = 0x0003003E

Bit	R/W	Description
31:0	RW	**PCIe DVSEC (03Eh) Header 3**

Offset: 0Ch PDSE0C: PCIe Device Security Enhancement Control Register 0C Init = 0x0000000D

Bit	R/W	Description
31:24	RW	**DIGEST_SEL**
23:16	RW	**NUM_DIGEST**
15:0	RW	**TCG_ALG_ID**

Offset: 10h PDSE10: PCIe Device Security Enhancement Control Register 10 Init = 0x00000000

Bit	R/W	Description
31:0	RW	**DIGEST 0**

Offset: 14h PDSE14: PCIe Device Security Enhancement Control Register 14 Init = 0x00000000

Bit	R/W	Description
31:0	RW	**DIGEST 1**

Offset: 18h PDSE18: PCIe Device Security Enhancement Control Register 18 Init = 0x00000000

Bit	R/W	Description
31:0	RW	**DIGEST 2**

Offset: 1Ch PDSE1C: PCIe Device Security Enhancement Control Register 1C Init = 0x00000000

Bit	R/W	Description
31:0	RW	**DIGEST 3**

Offset: 20h PDSE20: PCIe Device Security Enhancement Control Register 20 Init = 0x00000000

Bit	R/W	Description
31:0	RW	**DIGEST 4**

Offset: 24h PDSE24: PCIe Device Security Enhancement Control Register 24 Init = 0x00000000

Bit	R/W	Description
31:0	RW	**DIGEST 5**

Offset: 28h PDSE28: PCIe Device Security Enhancement Control Register 28 Init = 0x00000000

Bit	R/W	Description
31:0	RW	**DIGEST 6**

Offset: 2Ch PDSE2C: PCIe Device Security Enhancement Control Register 2C Init = 0x00000000

Bit	R/W	Description
31:0	RW	**DIGEST 7**

Offset: 30h PDSE30: PCIe Device Security Enhancement Control Register 30 Init = 0x00000000

Bit	R/W	Description
31:0	RW	**DIGEST 8**

Offset: 34h PDSE34: PCIe Device Security Enhancement Control Register 34 Init = 0x00000000

Bit	R/W	Description
31:0	RW	**DIGEST 9**

Offset: 38h PDSE38: PCIe Device Security Enhancement Control Register 38 Init = 0x00000000		
Bit	**R/W**	**Description**
31:0	RW	**DIGEST A**

Offset: 3Ch PDSE3C: PCIe Device Security Enhancement Control Register 3C Init = 0x00000000		
Bit	**R/W**	**Description**
31:0	RW	**DIGEST B**

Offset: 40h PDSE40: PCIe Device Security Enhancement Control Register 40 Init = 0x00000000		
Bit	**R/W**	**Description**
31:0	RW	**DIGEST C**

Offset: 44h PDSE44: PCIe Device Security Enhancement Control Register 44 Init = 0x00000000		
Bit	**R/W**	**Description**
31:0	RW	**DIGEST D**

Offset: 48h PDSE48: PCIe Device Security Enhancement Control Register 48 Init = 0x00000000		
Bit	**R/W**	**Description**
31:0	RW	**DIGEST E**

Offset: 4Ch PDSE4C: PCIe Device Security Enhancement Control Register 4C Init = 0x00000000		
Bit	**R/W**	**Description**
31:0	RW	**DIGEST F**

Offset: C0h PDSEC0: PCIe Device Security Enhancement Control Register C0 Init = 0x80010023		
Bit	**R/W**	**Description**
31:0	RW	**PCIe DVSEC (02Eh) Header 1**

Offset: C4h PDSEC4: PCIe Device Security Enhancement Control Register C4 Init = 0x02418086		
Bit	**R/W**	**Description**
31:0	RW	**PCIe DVSEC (02Eh) Header 2**

Offset: C8h PDSEC8: PCIe Device Security Enhancement Control Register C8 Init = 0x0000002E		
Bit	**R/W**	**Description**
31:0	RW	**PCIe DVSEC (02Eh) Header 3**

Offset: CCh PDSECC: PCIe Device Security Enhancement Control Register CC Init = 0x00000100		
Bit	**R/W**	**Description**
31:0	RW	**Authentication Header**

Offset: D0h PDSED0: PCIe Device Security Enhancement Control Register D0 Init = 0x01000900		
Bit	**R/W**	**Description**
31:0	RW	**Authentication Capabilities**

Offset: D4h PDSED4: PCIe Device Security Enhancement Control Register D4 Init = 0x00000021		
Bit	**R/W**	**Description**
31:0	RW	**Authentication Status**

Offset: D8h PDSED8: PCIe Device Security Enhancement Control Register D8 Init = 0x00000000		
Bit	**R/W**	**Description**
31:0	R	**The Last Data of Write Mailbox**

Offset: DCh PDSEDC: PCIe Device Security Enhancement Control Register DC Init = 0x00000000		
Bit	**R/W**	**Description**
31:27	RW	**Reserved**
26:16	R	**SRAM Current Write Pointer**
15:11	RW	**Reserved**
10:0	RW	**SRAM Initial Write Pointer**

Offset: E0h PDSEE0: PCIe Device Security Enhancement Control Register E0 Init = 0x00000000		
Bit	**R/W**	**Description**
31:27	RW	**Reserved**
26:16	R	**SRAM Current Read Pointer**
15:11	RW	**Reserved**
10:0	RW	**SRAM Initial Read Pointer**

Offset: E4h PDSEE4: PCIe Device Security Enhancement Control Register E4 Init = 0x00000000		
Bit	**R/W**	**Description**
31:11	RW	**Reserved**
10:0	RW	**Length of Read Data Mailbox**

Offset: E8h PDSEE8: PCIe Device Security Enhancement Control Register E8 Init = 0x00000000		
Bit	**R/W**	**Description**
31:0	R	**Scratch of Host Offset 2CCh Write**

Offset: ECh PDSEEC: PCIe Device Security Enhancement Control Register EC Init = 0x00000000		
Bit	**R/W**	**Description**
31:0	RW	**Scratch of Host Offset 2CCh Read**

Offset: F0h PDSEF0: PCIe Device Security Enhancement Control Register F0 Init = 0x00000000

Bit	R/W	Description
31:21	RW	**Reserved**
20	RW	**8K FIFO Depth Mode**
19	RW	**Enable Mailbox Read**
18	RW	**Disable Mailbox Write**
17	RW	**Enable SRAM**
16	RW	**Disable PCIe Device Security Enhancement**
15:0	RW	**BMC Interrupt Enable**

Offset: F4h PDSEF4: PCIe Device Security Enhancement Control Register F4 Init = 0x00000000

Bit	R/W	Description
31	R	**Host Offset 2D8h[1]**
30:24	R	**Reserved**
23:16	R	**DIGEST_SEL Write Status**
15:0	RW	**BMC Interrupt Status**

Offset: F8h PDSEF8: PCIe Device Security Enhancement Control Register F8 Init = 0x000FFFFF

Bit	R/W	Description
31:0	RW	**PDSE Valid bit[31:0]**

Offset: FCh PDSEFC: PCIe Device Security Enhancement Control Register FC Init = 0x017F0000

Bit	R/W	Description
31:0	RW	**PDSE Valid bit[63:32]**