AST2500/AST2520

Integrated Remote Management Processor

A2 Datasheet

Version 1.6

May 12, 2017

Headquarters
2F,No.15,Industry East Road 4.,Hsinchu Science Park,
Hsinchu City 30077, Taiwan, R. O. C.
TEL: 886-3-5789568
FAX: 886-3-5789586
http://www.ASPEEDtech.com

Ordering Information

Part number	:AST2500A2 - GP
Solder ball type	:Lead-free
Substrate type	:RoHS Green package
Package size	:19mm x 19mm
Ball pitch	: 0.8mm

Topside mark

ASPEED
AST2500
XXXXXX.XXXXXX
YYWW TAN A2 GP

Part number	:AST2520A2 - GP
Solder ball type	:Lead-free
Substrate type	:RoHS Green package
Package size	:19mm x 19mm
Ball pitch	: 0.8mm

Topside mark

ASPEED
AST2520
XXXXXX.XXXXXX
YYWW TAN A2 GP

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
Jul. 04, 2014	0.1	Initial draft.
Jan. 22, 2015	0.9	1. Release for chip revision A1. 2. Add chip revision A1 migration list at next page. 3. Add per-pin function configuration table at Section 5.3. 4. Add Reset Source Table at Section 6. 5. Add USB ports configuration mode description at Section 1.8.5. 6. Add Firmware update methods description at Section 1.9. 7. Add Firmware boot up sequence diagram at Section 10. 8. Revise IV11D from 1.1V to 1.15V. 9. Add ESD specification. 10. Revise DACRSET external resistor value to 18KΩ.
Jun. 22, 2015	1.0	1. Add CMOS IO 2.5V/1.8V DC spec. 2. Add feature of PCI-Express Master. 3. Rename power ball T11 from PV33D to MPLLAV33, this power ball requires isolated power rail with low noise. 4. Add specification for overshoot, undershoot, and ringback at section 3.3. 5. Add power consumption information at section 3.6. 6. Add DC electrical specification for DDR, USB, BSRAM at section 3.9. 7. Add AC electrical specification at section 3.10. 8. Add thermal specification at section 3.11. 9. Update firmware programming note at section 9. 10. Update the timing waveform of SGPIO master at section 2.6.1. 11. Add verilog example of SGPIO shifter at section 2.6.2. 12. Revise the pull up power of MAC TX strap pins. 13. Revise some values in the power up sequence at section 3.7.
Oct. 15, 2015	1.1	1. Revise boot up sequence diagram at section 10. 2. Add registers specification of APB to PCIe Bus Bridge at section 30. 3. Add registers specification of PCIe Host Controller at section 51. 4. Add firmware related APnote of Application Note 1 and Application Note 2. 5. Add hardware related APnote or Application Note 3. 6. Rename ”Graphics Display Controller” to ”SOC Display Controller”. 7. Remove ”Graphics Display Controller” feature on AST2520. 8. Add architecture block diagram of AST2520 at Figure-2.
Mar. 30, 2016	1.2	1. Release for chip revision A2. 2. Fix eSPI SUART1/SUART2 IRQ number issue. 3. Update chip hardware revision ID at SCU7C.
May. 18, 2016	1.3	1. Revise ADC DC specification. 2. Add power down sequence for firmware at section 3.8.
Jun. 07, 2016	1.31	1. Revise power down sequence for firmware at section 3.8. AST2510 do not require to care the power down sequence.
Nov. 08, 2016	1.4	1. Revise reset function of SPI1/SPI2 controller at section 6.2. 2. Remove MACD0 and MACD4 at section 16. Please refer document E2500-01 or Errata List. 3. Relax AC timing rise/fall time specification of RGMII, SPI, and eSPI. 4. Add recommended operating temperature of T_C.
Jan. 09, 2017	1.5	1. Add Strap input pins AC timing specification at section 3.10.15. 2. Remove USB portB USB2.0 Device Controller feature. Please refer document E2500-02 or Errata List.
May. 12, 2017	1.6	1. Revise SPI master AC timing specification at section 3.10.7. 2. Revise eSPI AC timing specification at section 3.10.8. 3. Revise I2C AC timing specification at section 3.10.11.

to next page

from previous page

4. Revise Overshoot timing specification at section 3.3.
5. Add MDC/MDIO AC timing specification at section 3.10.3.

Text color definition

1. All new added features compare to AST2400/AST1250/AST1400 are highlighted by this color.
2. **All different contents compare to previous release are highlighted by this color.**
3. All important descriptions are highlighted by this color.
4. ~~Removed features are highlighted by this style.~~

AST2500 Migration List from A1 to A2

Functional Bugs in A1 that have fixed in A2 list

1. Fix eSPI SUART1/SUART2 IRQ number issue, change from 7/8 to 4/3.

AST2500 - A2 Errata List

1. In Ethernet MAC controller IP, there are totally 8 MAC address filters, the MAC address filter #4 can not work properly, the affected registers are MACD0 and MACD4.
    - Solution: There is no hardware or software solution for this issue. Please do not use this MAC address filter #4. Instead, please use the other 7 MAC address filters if you need to filter certain MAC address.
2. In USB port B function – USB2.0 device controller, there is a logic failure on device address identification, thus it can not identify the correct device address and respond wrong packets to the host.
    - Solution: There is no hardware or software solution for this issue. Please do not use USB port B device controller function.

Contents

I Functional Specification **8**

1 General Information **8**
1.1 Introduction . . . 8
1.2 Chip Architecture . . . 9
1.3 Summary of Feature Set . . . 11
1.3.1 Package . . . 11
1.3.2 Design for Testability . . . 11
1.3.3 PCI-Express 2.0 Bus Device Controller *(AST2520 Excluded)* . . . 11
1.3.4 PCI-Express 2.0 Bus Master Controller *(AST2520 Excluded)* . . . 12
1.3.5 VGA Display Controller *(AST2520 Excluded)* . . . 12
1.3.6 64-bit 2D Graphics Accelerator *(AST2520 Excluded)* . . . 13
1.3.7 SOC Display Controller *(AST2520 Excluded)* . . . 13
1.3.8 DDR3L/DDR4 SDRAM Controller . . . 13
1.3.9 Embedded ARM1176JZS CPU . . . 14
1.3.10 Embedded Coprocessor CPU *(AST2520 Excluded)* . . . 14
1.3.11 Video Compression Engine *(AST2520 Excluded)* . . . 15
1.3.12 Internal SRAM . . . 16
1.3.13 Battery Backed SRAM . . . 16
1.3.14 System Control Unit (SCU) . . . 16
1.3.15 AHB Controller (AHBC) . . . 17
1.3.16 Interrupt Controller (VIC) . . . 17
1.3.17 Firmware SPI Memory Controller (FMC) . . . 17
1.3.18 SPI Master Controller (SPI) . . . 18
1.3.19 SD/SDIO/eMMC Host Controller . . . 18
1.3.20 USB2.0 Virtual Hub Controller . . . 18
1.3.21 USB1.1 HID Device Controller . . . 19
1.3.22 USB2.0 Host Controller . . . 19
1.3.23 USB1.1 Host Controller . . . 19
1.3.24 10/100/1000 Mbps Fast Ethernet MAC . . . 19
1.3.25 I2C/SMBus Serial Interface Controller . . . 20
1.3.26 GPIO Controller . . . 20
1.3.27 Master Serial GPIO Controller . . . 21
1.3.28 Slave Serial GPIO Monitor . . . 21
1.3.29 UART (16550) . . . 21
1.3.30 Timer . . . 22
1.3.31 Watchdog Timer (WDT) . . . 22
1.3.32 Real Time Clock (RTC) . . . 23
1.3.33 LPC Bus Interface . . . 23
1.3.34 eSPI Interface . . . 24
1.3.35 System SPI Flash Controller . . . 24
1.3.36 Super I/O Controller . . . 25
1.3.37 Hash & Crypto Engine . . . 25
1.3.38 Memory Integrity Check (MIC) Engine . . . 26
1.3.39 ADC Controller . . . 26
1.3.40 PWM Controller . . . 26
1.3.41 Fan Tachometer Controller . . . 26
1.3.42 PECI Controller . . . 27
1.3.43 JTAG Master Controller . . . 27
1.3.44 MCTP Controller *(AST2520 Excluded)* . . . 27
1.3.45 MSI Controller *(AST2520 Excluded)* . . . 27
1.3.46 X-DMA Controller *(AST2520 Excluded)* . . . 27
1.3.47 Software Specifications . . . 28
1.4 Features Comparison (AST2500/AST2400/AST2300) . . . 29

1.5 Features Comparison (AST2520/AST1250/AST1050) . . . 31
1.6 Features Comparison (AST2500/AST2510/AST2520) . . . 33
1.7 Features Comparison (AST2500/AST2520/AST2530) . . . 34
1.8 Applications . . . 34
1.8.1 Flash Interface Architecture . . . 34
1.8.2 Remote BIOS Update . . . 34
1.8.3 UART Interface Application . . . 36
1.8.4 Display Output Interface . . . 36
1.8.5 USB Port Configuration . . . 39
1.9 Firmware Update Methods . . . 40
1.9.1 Update through PCIe (AST2500/AST2510/AST2530) . . . 40
1.9.2 Update through LPC (AST2500/AST2510/AST2520/AST2530) . . . 40
1.9.3 Update through UART (AST2500/AST2510/AST2520/AST2530) . . . 40
1.9.4 Update through LAN by Firmware (AST2500/AST2520/AST2530) . . . 40
1.9.5 Update through SPI programmer (AST2500/AST2510/AST2520/AST2530) . . . 40

2 Pin & I/O Related Specification 41
2.1 Pin Description . . . 41
2.1.1 Not Available Pins for AST2520 . . . 65
2.2 Ball Map . . . 67
2.3 Hardware/Software Strap Definition . . . 70
2.3.1 Step 1: Load from dedicated strap pins . . . 70
2.3.2 Step 2: Hardware Strap – Load from the optional strap pins . . . 71
2.3.3 Step 3: Firmware strap – Load from the Firmware SPI flash . . . 71
2.3.4 Hardware strap configuration method . . . 72
2.3.5 Application Note 3: Hardware Strap Configuration . . . 75
2.4 Digital Video Interface . . . 77
2.4.1 Video Mode . . . 77
2.4.2 Digital Video Single Edge Data Mode . . . 77
2.4.3 Digital Video Dual Edge Display Output Data Mode : 12 Bits Interface . . . 78
2.5 GPIO Summary . . . 79
2.5.1 GPIO Table . . . 79
2.5.2 GPIO Pass Through . . . 84
2.6 Serial GPIO Master . . . 85
2.6.1 Timing Waveform . . . 86
2.6.2 Example Verilog Code for SGPIO Shifter . . . 87
2.7 Serial GPIO Slave Monitor . . . 87

3 Electrical Specifications 89
3.1 Absolute Maximum Ratings . . . 89
3.2 Recommended Operating Conditions . . . 89
3.3 Input Overshoot/Undershoot Tolerance . . . 90
3.4 Input Ringback Tolerance . . . 91
3.5 ESD Capability . . . 91
3.6 Power Consumption . . . 91
3.7 Power Up Sequence . . . 94
3.8 Power Down Sequence for Firmware . . . 96
3.9 I/O DC Electrical Specification . . . 97
3.9.1 3.3V CMOS I/O Parameters . . . 97
3.9.2 2.5V CMOS I/O Parameters . . . 97
3.9.3 1.8V CMOS I/O Parameters . . . 98
3.9.4 DDR3L I/O Parameters . . . 99
3.9.5 DDR4 I/O Parameters . . . 99
3.9.6 DAC I/O Parameters . . . 100
3.9.7 ADC I/O Parameters . . . 100
3.9.8 USB I/O Parameters . . . 100
3.9.9 Battery Backed SRAM Parameters . . . 100

3.9.10 PECI I/O Parameters . . . 101
3.9.11 RGMII/RMII/NCSI I/O Parameters . . . 101
3.10 AC Timing Specification . . . 102
3.10.1 Reference Clock Input . . . 102
3.10.2 LPC Interface . . . 103
3.10.3 RGMII/RMII/NCSI Interface . . . 104
3.10.4 DDR3/DDR4 Interface . . . 106
3.10.5 Video Interface Input/Output: Single Edge . . . 106
3.10.6 Video Interface Output: Dual Edge . . . 106
3.10.7 SPI Master Interface . . . 107
3.10.8 **eSPI Slave Interface** . . . 108
3.10.9 SPI Pass-through Propagation Delay . . . 108
3.10.10 JTAG Master Interface . . . 109
3.10.11 I2C/SMBus Interface . . . 109
3.10.12 SD/eMMC Interface . . . 110
3.10.13 SGPIO Master Interface . . . 111
3.10.14 SGPIO Slave Interface . . . 111
3.10.15 Strap Input Interface . . . 112
3.11 Thermal Specification (Simulation Result) . . . 113
3.11.1 Terminology . . . 113
3.11.2 Testing Conditions . . . 114
3.11.3 Thermal Data . . . 114
3.11.4 Substrate Material Properties . . . 114

4 Package Information **115**
4.1 SMT Soldering Reflow Chart . . . 116

II Firmware Programming Guide **117**

5 Multi-function Pins Mapping and Control **117**
5.1 Function 1-2 . . . 117
5.2 Function 3-4 . . . 127
5.3 Pin Function Configuration . . . 128

6 Reset Source Table **143**
6.1 Function Level Reset Source Table . . . 144
6.2 Register Level Reset Source Table . . . 146

7 Memory Space Allocation Table **161**
7.1 ARM Address Space Mapping . . . 161
7.2 In-compatible List to AST2400 . . . 162
7.3 Coprocessor Address Space Mapping . . . 163
7.4 VGA Memory Space map to ARM Memory Space . . . 163

8 Interrupt Source Table **164**

9 Major Firmware Programming Change List **167**
9.1 Application Note 1: Firmware Programming Note for CLKIN=25MHz . . . 168
9.2 Application Note 2: I2C DMA and UHCI limitation . . . 169

10 Firmware Boot Up Sequence **171**

11 UART Debug Interface **172**

III Function Registers **173**

12 AHB Bus Controller **173**
12.1 Overview . . . 173
12.2 Features . . . 173
12.3 Registers : Base Address = 0x1E60:0000 . . . 174
12.4 Programming Guide . . . 178
12.4.1 Enable Polling . . . 178

13 Firmware SPI Memory Controller **179**
13.1 Overview . . . 179
13.2 Features . . . 180
13.2.1 NOR type flash supporting features . . . 180
13.2.2 SPI type flash supporting features . . . 180
13.2.3 Alternate (2nd) Boot Function . . . 180
13.3 Timing Definition . . . 181
13.4 Registers : Base Address = 0x1E62:0000 . . . 182
13.5 Programming Guide . . . 193
13.5.1 DMA Mode . . . 193
13.5.2 DMA CheckSum Calculation Mode . . . 193
13.5.3 SPI Timing Calibration Sequence . . . 194
13.5.4 Address Continuity Rule . . . 194

14 SPI Flash Controller **195**
14.1 Overview . . . 195
14.2 Registers : Base Address = 0x1E63:x000 . . . 195

15 Memory Integrity Check Controller **205**
15.1 Overview . . . 205
15.2 Features . . . 205
15.3 Registers : Base Address = 0x1E64:0000 . . . 205
15.4 Page Control Bits . . . 209
15.5 Control Buffer Format . . . 209
15.6 Checksum Buffer Format . . . 210
15.7 Porgramming Sequence . . . 211
15.7.1 Parameter Definition . . . 211
15.7.2 MIC Engine Initiation (DRAM 256M Byte Address Mode) . . . 211
15.7.3 MIC Engine Initiation (DRAM 1G Byte Address Mode) . . . 212
15.7.4 Start Page CheckSum Process . . . 212
15.7.5 Stop Page CheckSum Process (DRAM 256M Byte Address Mode) . . . 212
15.7.6 Stop Page CheckSum Process (DRAM 1G Byte Address Mode) . . . 212
15.8 Interrupt Behavior . . . 213

16 10/100/1G Ethernet MAC Controller **214**
16.1 Overview . . . 214
16.2 Features . . . 215
16.3 Registers : . . . 217
16.4 Function Description . . . 237
16.4.1 Transmit Descriptor . . . 237
16.4.2 Receive Descriptor . . . 238
16.4.3 Transmitting Packet . . . 241
16.4.4 Receiving Packet . . . 241
16.4.5 Ethernet Address Filtering . . . 241
16.4.6 MII Management Interface . . . 242
16.5 Initialization . . . 243
16.5.1 Frame Transmitting Procedure . . . 243
16.5.2 Frame Receiving Procedure . . . 244

17 USB2.0 Virtual Hub Controller **245**
17.1 Overview . . . 245
17.2 Features . . . 245
17.3 Procedure to enable USB2.0 Hub port . . . 246
17.3.1 Procedure to enable USB2.0 eye diagram measurement . . . 246
17.4 Registers : Base Address = 0x1E6A:0000 . . . 247
17.4.1 Address Definition . . . 247
17.4.2 Root/Global Register Definition . . . 247
17.4.3 Device #1 — #5 Register Definition . . . 256
17.4.4 Programmable Endpoint #0 — #14 Register Definition . . . 258
17.4.5 Programmable Endpoint DMA Descriptor Definition . . . 265
17.4.6 Register Reset Table . . . 267
17.5 Software Programming Guide . . . 268
17.5.1 Reset Control . . . 268
17.5.2 Initialization Sequence . . . 268
17.5.3 Set Device Address . . . 270
17.5.4 Response STALL . . . 271
17.5.5 Programmable Endpoint OUT Transfer Finish Check . . . 271
17.5.6 Prevent a Transient Read Pointer Value . . . 271
17.5.7 Procedure to enable Interrupt . . . 271
17.5.8 OUT Direction Endpoint Maximum Packet Size Setting . . . 271

18 USB2.0 Device Controller (Removed) **272**

19 USB2.0/1.1 Host Controller **273**
19.1 Overview . . . 273
19.2 UHCI Features . . . 273
19.3 EHCI Features . . . 274
19.4 Procedure to enable USB Host port . . . 274
19.4.1 USB2.0 Host port 1 . . . 274
19.4.2 USB2.0 Host port 2 . . . 275
19.4.3 UHCI controller . . . 275
19.5 UHCI Registers : Base Address = 0x1E6B:0000 . . . 276
19.6 EHCI Registers . . . 282

20 Interrupt Controller **293**
20.1 Overview . . . 293
20.1.1 ARM CPU Interrupt Controller . . . 293
20.1.2 System LPC Interrupt Controller (SVIC) . . . 294
20.1.3 Coprocessor Interrupt Controller (CVIC) . . . 295
20.2 Features . . . 295
20.3 VIC Registers : Base Address = 0x1E6C:0000 . . . 295
20.4 SVIC Registers : Base Address = 0x1E6C:1000 . . . 302
20.5 CVIC Registers : Base Address = 0x1E6C:2000 . . . 304

21 SDRAM Memory Controller **307**
21.1 Overview . . . 307
21.1.1 Features . . . 307
21.1.2 ECC Features . . . 307
21.1.3 Register Summary . . . 307
21.2 Fixed Priority DRAM Request . . . 309
21.3 Registers: Base Address = 0x1E6E:0000 . . . 309
21.4 Address Arrangement . . . 331
21.4.1 Address Translation . . . 331
21.5 Self Refresh Command Sequence . . . 331
21.5.1 Enter Self Refresh . . . 331
21.5.2 Exit Self Refresh . . . 331

22 USB1.1 HID Controller **332**
22.1 Overview . . . 332
22.2 Features . . . 332
22.3 Procedure to enable USB1.1 Device port . . . 332
22.4 Registers : Base Address = 0x1E6E:1000 . . . 333
22.5 Software Programming Guide . . . 338

23 System Control Unit (SCU) **339**
23.1 Overview . . . 339
23.2 Registers : Base Address = 0x1E6E:2000 . . . 340

24 Hash & Crypto Engine (HACE) **385**
24.1 Overview . . . 385
24.2 Features . . . 385
24.3 Registers : Base Address = 0x1E6E:3000 . . . 387
24.4 RSA Buffer Format in Extended SRAM : Extended SRAM Base Address = 0x1E72:0000 . . . 394
24.5 Crypto Context Buffer Format . . . 394
24.5.1 RC4 (272 Bytes) . . . 394
24.5.2 AES-128 (192 Bytes) . . . 394
24.5.3 AES-192 (224 Bytes) . . . 394
24.5.4 AES-256 (256 Bytes) . . . 394
24.5.5 DES (40 Bytes) . . . 395
24.6 Hash Function Programming Sequence . . . 396
24.6.1 Parameter Definition . . . 396
24.6.2 MD5/SHA1/SHA224/SHA256 . . . 396
24.6.3 HMAC MD5/SHA1/SHA224/SHA256 . . . 397
24.6.4 Accumulative Mode . . . 397
24.7 Command Queue Data Format . . . 399

25 JTAG Master Controller **400**
25.1 Overview . . . 400
25.2 Features . . . 400
25.3 Registers : Base Address = 0x1E6E_4000 . . . 400
25.4 Operation . . . 403
25.4.1 Reset State Machine . . . 403
25.4.2 Instruction Transmission . . . 403
25.4.3 Data Transmission . . . 404
25.5 Application Note . . . 404
25.5.1 Introduction . . . 404
25.5.2 Hardware Mode . . . 404
25.5.3 Software Mode . . . 405

26 SOC Display Controller (GFX) **406**
26.1 Overview . . . 406
26.2 Features . . . 406
26.3 SOC Display Controller Registers . . . 406

27 X-DMA Controller **415**
27.1 Overview . . . 415
27.2 Features . . . 415
27.3 Registers . . . 415
27.4 Command Format . . . 421
27.5 Clearing and Setting Procedure . . . 422
27.5.1 Host Command Queue in Dirty Frame Clearing Procedure . . . 422
27.5.2 Down-stream in Dirty Frame Clearing Procedure . . . 422
27.5.3 Host Command Queue Read/Write Pointer Setting Procedure . . . 422
27.5.4 BMC Command Queue Read/Write Pointer Setting Procedure . . . 422

27.5.5 VGA Command Queue Read/Write Pointer Setting Procedure . . . 423

28 MCTP Controller 424
28.1 Overview . . . 424
28.2 Features . . . 424
28.3 Registers : Base Address = 0x1E6E_8000 . . . 424
28.4 Command . . . 426
28.4.1 TX command . . . 426
28.4.2 RX command . . . 427
28.5 Operation . . . 427
28.5.1 Send Packet . . . 427
28.5.2 Receive Packet . . . 427

29 ADC Controller 428
29.1 Overview . . . 428
29.2 Features . . . 429
29.3 Registers : Base Address = 0x1E6E_9000 . . . 429
29.4 Operation . . . 440
29.4.1 Initialize Sequence . . . 440
29.4.2 Compensating Sensing Method . . . 440
29.4.3 Voltage Sense Method . . . 440

30 APB to PCIe Bus Bridge 441
30.1 Overview . . . 441
30.2 Features . . . 441
30.3 Registers : Base Address = 0x1E6F_0000 . . . 441
30.4 Operation . . . 444
30.4.1 Send Packet . . . 444
30.4.2 Receive Packet . . . 444

31 eSPI Controller 445
31.1 Overview . . . 445
31.2 Features . . . 446
31.3 Registers : Base Address = 0x1E6E_E000 . . . 446
31.4 Programming Guide . . . 469
31.4.1 Address of Peripheral Channel Rx Packet.(MemWR and MemRD) . . . 469
31.4.2 Process of receiving packet . . . 469
31.4.3 Procedure to sending packet . . . 469

32 Battery Backed SRAM 471
32.1 Overview . . . 471

33 Video Engine (VE) 472
33.1 Overview . . . 472
33.2 Features . . . 473
33.3 Registers : Base Address = 0x1E70:0000 . . . 475

34 SRAM Memory Buffer 509
34.1 Overview . . . 509

35 SD/SDIO Host Controller 510
35.1 Overview . . . 510
35.2 Features . . . 511
35.3 Registers : Base Address = 0x1E74:0000 . . . 511
35.4 Operation . . . 539
35.4.1 Card Detection Method . . . 539

36 GPIO Controller **540**
36.1 Overview . . . 540
36.2 Features . . . 544
36.2.1 Parallel GPIO . . . 544
36.2.2 Serial GPIO . . . 544
36.2.3 SGPIO Slave Monitor . . . 544
36.3 Registers : Base Address = 0x1E78:0000 . . . 545
36.3.1 Parallel GPIO . . . 545
36.3.2 Serial GPIO . . . 584
36.3.3 SGPIO Slave . . . 590
36.4 Programming Guide . . . 603
36.4.1 LPC port80h direct to GPIO . . . 603
36.4.2 LPC port80h direct to SGPIO . . . 604
36.4.3 Parallel GPIO output driving mode . . . 604

37 Real Time Clock (RTC) **605**
37.1 Overview . . . 605
37.2 Features . . . 605
37.3 Registers : Base Address = 0x1E78:1000 . . . 605
37.4 Operation . . . 608
37.4.1 Initialize Sequence . . . 608
37.4.2 Alarm Mode . . . 608

38 Timer Controller **610**
38.1 Overview . . . 610
38.2 Features . . . 610
38.3 Registers : Base Address = 0x1E78:2000 . . . 611
38.4 Operation . . . 619
38.5 Programming Note . . . 620
38.5.1 Interrupt Generation . . . 620
38.5.2 Programmable Duty Cycle Pulse Generation . . . 620
38.5.3 Programmable Delay Sequence Pulse Generation . . . 620

39 UART Controller (16550) **622**
39.1 Overview . . . 622
39.2 Features . . . 622
39.3 Registers . . . 623
39.3.1 UART_DLL/UART_DLH . . . 628

40 UART DMA **630**
40.1 Overview . . . 630
40.2 Features . . . 630
40.3 Registers : Base Address = 0x1E78:2000 . . . 631

41 Watchdog Timer **636**
41.1 Overview . . . 636
41.2 Features . . . 636
41.3 Registers : Base Address = 0x1E78:5000 . . . 637
41.4 Operation . . . 645
41.4.1 Enable watchdog reset . . . 645
41.4.2 Enable watchdog pulse output . . . 646
41.4.3 Enable watchdog interrupt output . . . 646
41.5 Programming Note . . . 646
41.5.1 PCLK frequency requirement . . . 646

42 PWM & Fan Tacho Controller **647**
42.1 Overview . . . 647
42.2 Features . . . 647
42.3 Registers : Base Address = 0x1E78:6000 . . . 647

43 Virtual UART **659**
43.1 Overview . . . 659
43.2 Features . . . 660
43.3 VUART Registers : Base Address = 0x1E78:7000 . . . 660
43.4 PUART Registers : Base Address = 0x1E78:8000 . . . 671

44 LPC Controller **674**
44.1 Overview . . . 674
44.2 Features . . . 676
44.3 Registers : Base Address = 0x1E78:9000 . . . 677

45 LPC+ Controller **712**
45.1 Overview . . . 712
45.2 Feature . . . 712
45.3 Registers : Base Address = 0x1E6E_C000 . . . 712
45.4 Operation . . . 715
45.4.1 Calibration . . . 715

46 SuperIO Controller (SIO) **716**
46.1 Overview . . . 716

47 System Wake-Up Control (SWC) **728**
47.1 Overview . . . 728

48 MailBox Controller (MBOX) **736**
48.1 Overview . . . 736

49 I2C/SMBus Controller **740**
49.1 Overview . . . 740
49.2 Features . . . 740
49.2.1 I2C Master - all 14 buses . . . 740
49.2.2 I2C Slave - all 14 buses . . . 741
49.2.3 SMBus - all 14 buses . . . 741
49.2.4 General . . . 741
49.3 Migration from AST2300 to AST2400 . . . 742
49.4 Migration from AST2400 to AST2500 . . . 742
49.5 Timing Definition . . . 743
49.5.1 Clock Setting Table . . . 744
49.6 Registers : Base Address = 0x1E78:A000 . . . 745
49.6.1 Address Definition . . . 745
49.6.2 Global Register Definition . . . 745
49.6.3 Device Register Definition . . . 747

50 PECI Controller **757**
50.1 Overview . . . 757
50.2 Features . . . 757
50.3 Registers : Base Address = 0x1E78:B000 . . . 758

51 PCIe Host Controller **764**
51.1 Overview . . . 764
51.2 Features . . . 764
51.3 Registers : Base Address = 0x1E6E:D000 . . . 764

IV PCI Express Interface 766

52 PCI Express Controller 766
52.1 PCI Express Configuration Registers . . . 766
52.2 Type 1 Configuration Registers . . . 766
52.3 Message signaled interrupts . . . 766
52.4 Power Management Capability Structure . . . 767
52.5 PCI Express Capability Structure . . . 767
52.6 SSID/SSVID Capability Structure . . . 767

53 PCI Bus Controller (P-Bus) 768
53.1 Overview . . . 768
53.2 Features . . . 768

54 VGA Display Controller 775
54.1 Overview . . . 775
54.2 Features . . . 775
54.3 Registers . . . 775
54.4 Sequential Controller Registers . . . 777
54.5 CRT Controller Registers . . . 778
54.6 Graphics Controller Registers . . . 783
54.7 Attribute Controller Registers . . . 785
54.8 RAMDAC Registers . . . 786
54.9 Extended CRT Registers . . . 788

55 2D Graphics Engine (G2D) 799
55.1 Overview . . . 799
55.2 Features . . . 799
55.3 2D Engine Registers . . . 801

56 Graphics Hardware Cursor 810
56.1 Features . . . 810
56.2 Register Definition . . . 810
56.3 Cursor Shape Structure Definition . . . 812
56.3.1 Monochrome Cursor Format (AND-XOR-RGB444 pixel format) . . . 812
56.3.2 Color Cursor Format (ARGB4444 pixel format) . . . 812

57 P-Bus to AHB Bridge (P2A) 813
57.1 Overview . . . 813
57.2 Registers : Base Address = MMIOBASE . . . 813

58 Message Signaled Interrupts (MSI) 814
58.1 Overview . . . 814
58.2 Features . . . 814
58.3 Operation . . . 814
58.3.1 Interrupt Table . . . 814
58.3.2 Message Signaled Interrupts . . . 815
58.3.3 Virtual INTx . . . 818

59 PCIe BMC Device 819
59.1 Overview . . . 819
59.2 Features . . . 819

List of Figures

1 AST2500 Chip Architecture . . . 9
2 AST2520 Chip Architecture . . . 10
3 AST2500 system applications . . . 35
4 AST2500 flash interface architecture . . . 35
5 AST2500 Remote BIOS update application . . . 36
6 AST2500 UART interface control . . . 37
7 AST2500 UART interface architecture . . . 38
8 AST2500 display interface architecture . . . 38
9 Ball Map – Left Side . . . 68
10 Ball Map – Right Side . . . 69
11 Single Edge Data Interface . . . 78
12 Dual Edge Data Interface . . . 78
13 GPIO Pass Through Control . . . 85
14 8 Bits SGPIO Waveform . . . 86
15 16 Bits SGPIO Waveform . . . 86
16 24 Bits SGPIO Waveform . . . 86
17 64 Bits SGPIO Waveform . . . 86
18 Slave SGPIO Waveform . . . 88
19 Overshoot/Undershoot . . . 90
20 Ringback . . . 91
21 Power-up sequence . . . 94
22 Power-down sequence . . . 96
23 Timing Definition . . . 102
24 LPC Timing Waveform . . . 103
25 RGMII Timing Waveform . . . 104
26 RMII/NCSI Timing Waveform . . . 105
27 MDC/MDIO Timing Waveform . . . 105
28 Video SDR Timing Waveform . . . 106
29 Video DDR Timing Waveform . . . 106
30 SPI Master Timing Waveform . . . 107
31 eSPI Slave Timing Waveform . . . 108
32 JTAG Master Timing Waveform . . . 109
33 I2C Timing Waveform . . . 109
34 SD/eMMC Timing Waveform . . . 110
35 SGPIO Master Timing Waveform . . . 111
36 SGPIO Slave Timing Waveform . . . 111
37 Strap Input Timing Waveform . . . 112
38 Thermal Terminology . . . 113
39 IC Package . . . 115
40 SMT Soldering Reflow Chart . . . 116
41 Firmware Boot-Up sequence . . . 171
42 NOR Flash Read Timing . . . 181
43 NOR Flash Write Timing . . . 181
44 SPI Flash R/W Timing . . . 181
45 SPI Flash 2 Bits R/W Timing . . . 181
46 Interrupt Flag Priority . . . 213
47 RGMII timing control block diagram . . . 358
48 RMII timing control block diagram . . . 359
49 ADC Reference Circuit . . . 440
50 Parallel GPIO Function Block Diagram . . . 545
51 Programmable duty cycle pulse generation . . . 620
52 Programmable delay sequence pulse generation . . . 621
53 UART Packet Frame . . . 622
54 Clock Generation . . . 743
55 AC Timing . . . 743

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) – AST2500.

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

AST2500 is the 6th 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. Instead of supporting PCI bus, AST2500 is designed to dedicatedly support PCIE Gen2 1x bus interface, which can make PCB layout simpler and fit systems that are going without PCI bus support. Figure-1 and Figure-2 clearly illustrates the primary chip architecture of the device. The detailed features of the individual internal blocks will be descried in the following chapters.

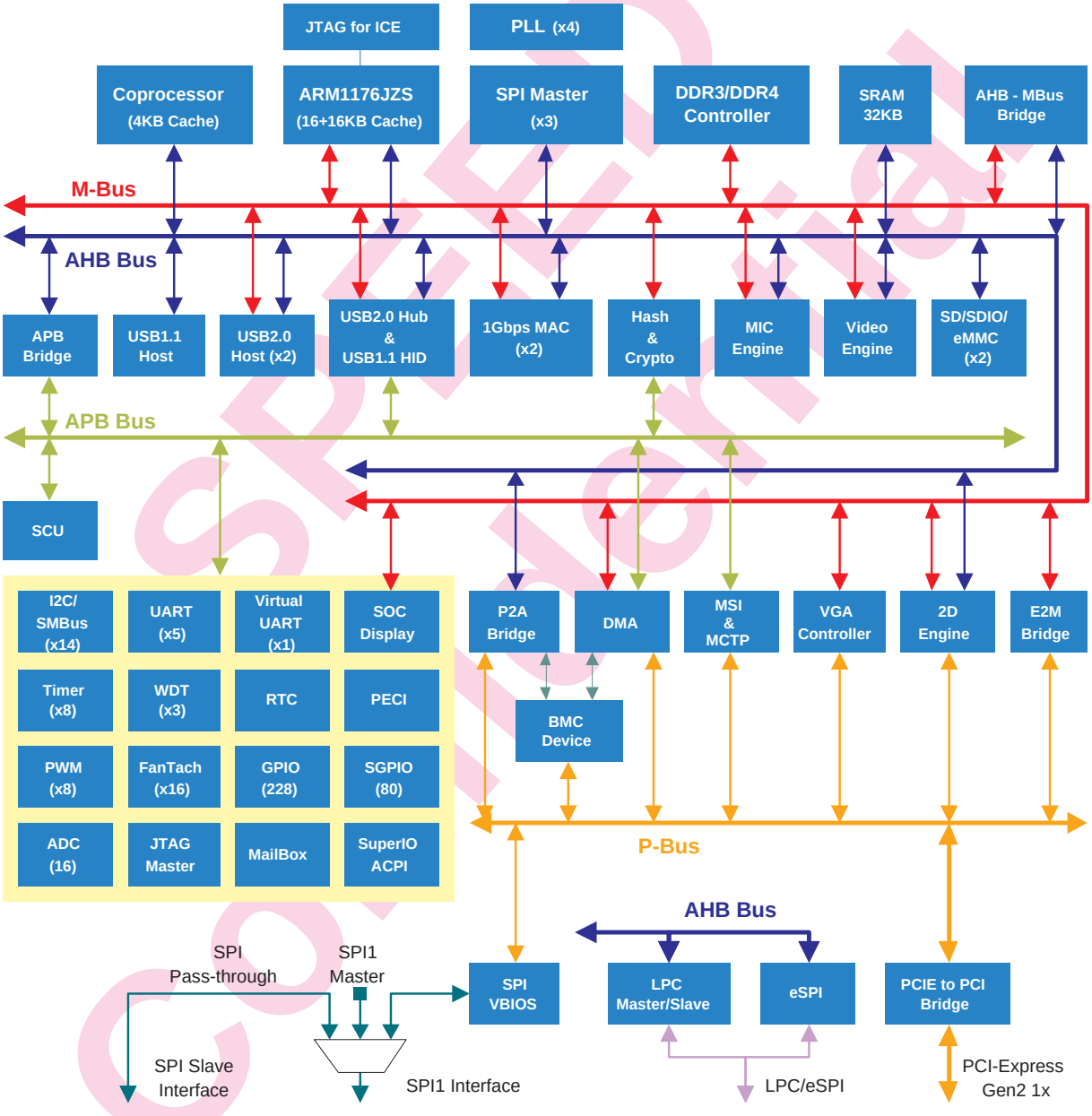


Figure 1: AST2500 Chip Architecture

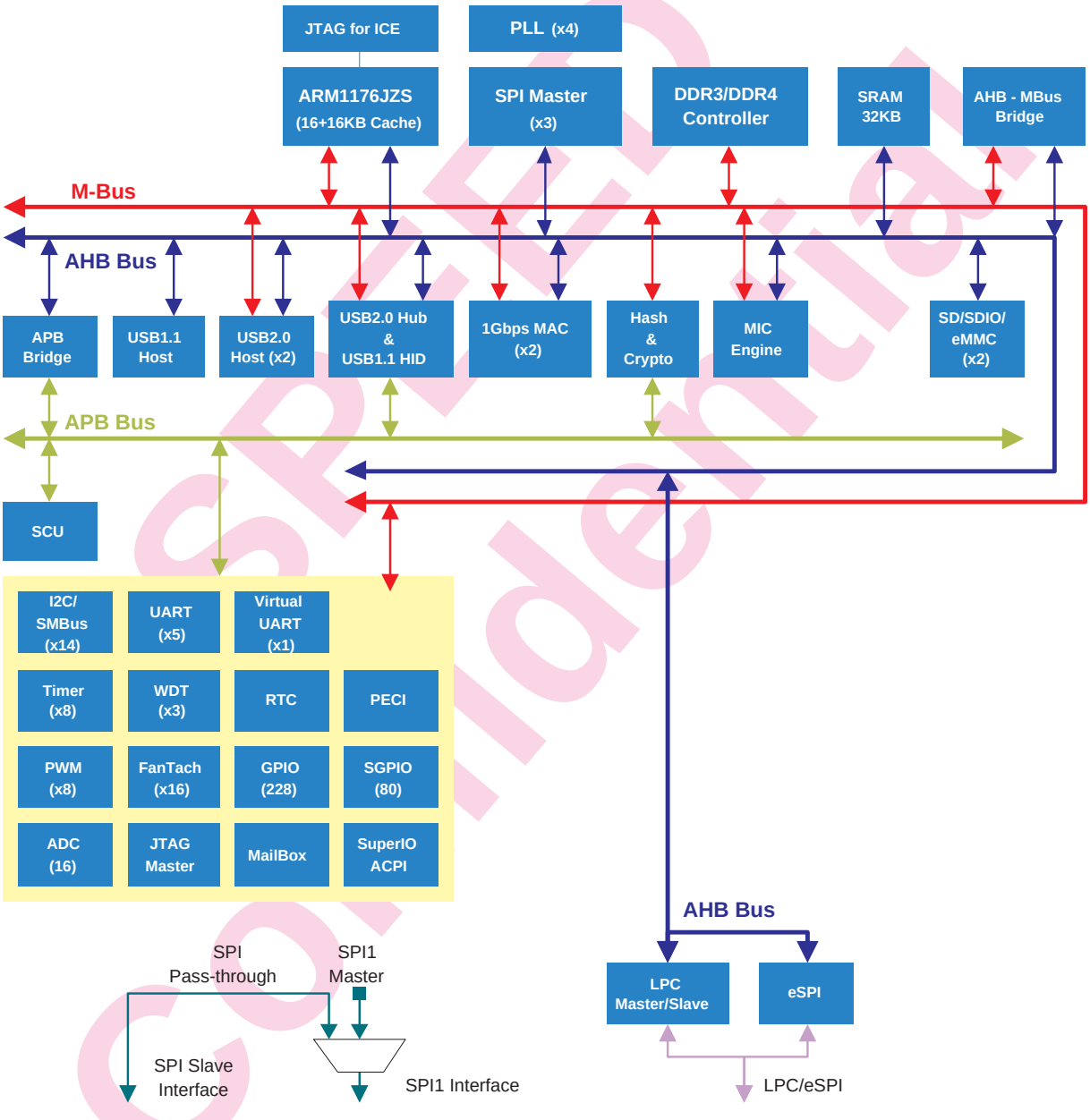


Figure 2: AST2520 Chip Architecture

1.3 Summary of Feature Set

1.3.1 Package

- 19mmx19mm 456-pin TFBGA package
- 0.8 mm ball pitch
- RoHS compliant 4-layer substrate design
- Typical condition chip power consumption (including DRAM power)

  Peak power consumption : < 1.76 W
  Average power consumption : < 1.46 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 *(AST2520 Excluded)*

- 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 decoding and interrupt BMC
- VGA device frame buffer memory allocation can be determined by external strapping resistors (16/64MB)
- VGA device can be disabled by external strapping resistor
- 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 Bus Master Controller *(AST2520 Excluded)*

- Built-in PCI-Express 2.0 Root Complex or End Point 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
- Integrate video 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 INTX, power management, error signaling
- Support native active state power management L0s and L1 states
- Support beacon (wake-up) function

1.3.5 VGA Display Controller *(AST2520 Excluded)*

- 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 1.1V reference voltage generator
  - Support DAC power down function, directly controlled by ARM CPU or Host CPU
- Digital video output options with multiplexed output pins
  - 165MHz 24-bit single-edge DVO (3.3V digital signals)
  - 165MHz 12-bit dual-edge DVO (3.3V digital signals)
- Triple DAC display output supports dynamic switching between VGA and Graphics Controller
- Digital Video output supports dynamic switching between VGA and Graphics Controller
- VGA D-sub connector supports hot-plug detection function for power and DRAM bandwidth saving

1.3.6 64-bit 2D Graphics Accelerator *(AST2520 Excluded)*

- 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 SOC Display Controller *(AST2520 Excluded)*

- Designed to support an extra graphics display function for ARM CPU embedded system
- Share the triple DAC and the digital video output pins with VGA Display Controller
- Support dynamic switching the triple DAC display output between VGA and SOC Display Controller
- Support dynamic switching the Digital Video output between VGA and SOC Display Controller
- Integrate one CRT display controller
  - Integrate one PLL clock generator with max. 165MHz video clock
  - Programmable CRT timing with VESA display timing compliance
  - Support RGB565 and XRGB8888 graphics display mode
  - Support hardware mono/color cursors
  - Support On-Screen Display (OSD)
- Maximum Display resolution: 1920x1080 32bpp@60Hz

1.3.8 DDR3L/DDR4 SDRAM Controller

- Support external 16-bit DDR3L/DDR4 SDRAM data bus width
- Maximum memory clock frequency
  - DDR3L : 800MHz (DDR3-1600)
  - 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 DDR3L DRAM Types: 64MBx16, 128MBx16, 256MBx16, 512MBx16 (stack die)
- Supported DDR4 DRAM Types: 128MBx16, 256MBx16, 512MBx16
- 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
  - 8-bit ECC error counter with reset control, overflow protection and interrupt option

1.3.9 Embedded ARM1176JZS CPU

- Embedded ARM1176JZS 32-bit RISC CPU
- Maximum running frequency: 800MHz
- Instruction cache size: 16KB
- Data cache size: 16KB
- Support Memory Management Unit(MMU)
- Interface: Multiple AXI buses for instruction, data and peripheral access
- Integrated PLL for CPU clock generation
- 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

1.3.10 Embedded Coprocessor CPU *(AST2520 Excluded)*

- Embedded one more 32-bit Coprocessor RISC CPU except the ARM
- Maximum running frequency: 200MHz (AHB Bus clock)
- Instruction cache size: 2KB
- Data cache size: 2KB
- 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.

1.3.11 Video Compression Engine *(AST2520 Excluded)*

- 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

1.3.12 Internal SRAM

- Support 36KB internal SRAM buffer
- Directly accessed by CPU through AHB
- SRAM content is retained even after hardware reset conditions

1.3.13 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

1.3.14 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

1.3.15 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.

1.3.16 Interrupt Controller (VIC)

- Directly connected to AHB bus interface
- Support up to 64 interrupt sources for ARM system
- Support up to 32 interrupt sources for Coprocessor system
- Support up to 18 interrupt sources for Main system through LPC SerialIRQ or SMI
- Support interrupt sources enable and disable control
- Support software triggered interrupt function
- Support hardware automatic Heart beat LED indicator output, which can link to interrupt events to monitor software health.

1.3.17 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 up to 100MHz of clock speed
- Support 2 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#) dose not boot up sucessfully within 22 seconds.
- Support write command and address filter function
- Support register lock until reset function
- Support software strap function, search and fetch software strap code at power up automatically.

1.3.18 SPI Master Controller (SPI)

- Support more 2 sets of SPI master interfaces
- All registers are fully the same as the firmware SPI memory interface.
- Support a flexible command mode for any types and any size of flash access
- Support up to 100MHz of clock speed
- Support 2 bits input/output command modes
- Support 24 or 32 bits address mode.
- Support 2 chip select pins for each set of SPI master
- Support write command and address filter function
- Support register lock until reset function

1.3.19 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 v4.51 Compliant
- Support maximum 32GB flash card size
- Support 2 slots of 4-bits SD/SDIO/eMMC or 1 slot of 8-bits eMMC interfaces
- Integrate DMA controller
- Independent interrupt
- Power and clock of each slot can be controlled independently

1.3.20 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 5 downstream ports with configurable endpoint type
- Support total 15 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 5 of any types of USB devices concurrently
- Support USB over LAN architecture

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 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
- There is a limitation on the using of UHCI, please refer to Application Note 2.

1.3.24 10/100/1000 Mbps Fast Ethernet MAC

- Integrate 2 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 x2)
- Support 2 NCSI interfaces up to 4 loads on 50 ohm impedance traces up to 36 inches in length
- 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

1.3.25 I2C/SMBus Serial Interface Controller

- Directly connected to APB bus
- Integrate 14 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 (<= 16 Byte) of Pool Buffer Mode for improving performance
  - Programmable Size (<= 4095 Byte) of DMA buffer for large amount of data transfer
  - Support 1 dedicated hardware alert pin for each set of SMBus controller
  - Support dual slave address when working at slave mode
- 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
- There is a limitation on the using of DMA mode, please refer to Application Note 2.

1.3.26 GPIO Controller

- Directly connected to APB bus
- Support up to 228 GPIO pins, which are 29 sets
- Each GPIO sets can be programmed to accept command from ARM, LPC(SIO), or Coprocessor.
  - Total 212 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)
  - 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 milisecond)
  - 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 228 GPIO pins have 16mA driving strength, others have 8mA driving strength
- 16 out of the 228 GPIO pins that can support 1.8V mode.
- Support 8 sets of GPIO pass-through (1 GPIO IN $\rightarrow$ 1 GPIO OUT) pin with internal switch control, which is useful for some button function control

1.3.27 Master Serial GPIO Controller

- Directly connected to APB bus
- Co-work with external serial-chained TTL components (74LV165/74LV595)
- Support up to 80 SGPIO input ports and 80 output ports concurrently only at the cost of 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 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)

1.3.28 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 10 drives recording capability for each channel

1.3.29 UART (16550)

- Directly connected to APB bus
- Support 5 UART controllers and 6 UART IO interfaces, each UART controller can be directed to any UART IO interface
- Support 5 sets UART IO interface with full flow control pins, and 1 set with Tx/Rx only for firmware console
- 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 $\sim$ UART4 Separate transmit & receive FIFO buffer (16x8) to reduce CPU interrupts
- Support up to 921.K 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.

1.3.30 Timer

- 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.
- 4 of 8 sets timer can generate LED blink signals or speaker signals.
- 6 of 8 sets timer can generate a programmable sequence signal output, it can be used to control power sequence.

1.3.31 Watchdog Timer (WDT)

- Directly connected to APB bus
- Watchdog function
- Built-in 3 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
  - **Alternate boot signal**: to boot from second flash
- Generate 2 types of reset pulse (programmable) for reseting BMC part or full chip
- Configurable reset selection control for each BMC function module
- Generate failover boot signal for booting from 2nd boot flash (FWSPICS1#)

1.3.32 Real Time Clock (RTC)

- Directly connected to APB bus
- Backward compatible to AST2400
- Reference clock is divided from CLKIN (24MHz) 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

1.3.33 LPC Bus Interface

- Directly connected to APB bus interface
- Maximum running frequency: 33MHz
- Dual operation modes
  - Master mode: designed to update system BIOS, TPM, or LPC controller (I/O, memory, and firmware read/write cycles)
  - Slave mode: designed for BMC functions (I/O and memory or firmware read/write cycles)
- Support Serial IRQ (reduce polling time)
- Support port 80H/81H (programmable address) snooping registers with interrupt options
- Support 1 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 with programmable hot-key detection (for example: [ESC]+'Q' or [ESC]+'(')

1.3.34 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

1.3.35 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 interface by host system
  - Support system boot BIOS flash memory
  - Support up to 100MHz of clock speed
  - Support 2 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
- SPI Pass-through:
  - 1 SPI input interface and 1 SPI output interface with internal switch control
  - Through the switch control, it is easily to update BIOS by the BMC
  - Only support single IO mode of SPI command
  - The maximum SPI clock speed allowed in pass-through mode is 30MHz. For higher frequency of multibit IO requirements, it can use external switch mode for BIOS connection.
- For detail description, see Section 1.8

1.3.36 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 (programmable address) snoop feature with 128 bytes FIFO mode or DMA mode
- Support Port 80h redirect to 8 GPIO Pins for LED indicator
- Support Port 80h redirect to the first 8 bits of Serial GPIO for LED indicator
- Support 7+8 general purpose registers for communication between the host and the BMC with interrupt capability
- Support mailbox with 16 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

1.3.37 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 32 bits to 4064 bits.
- Support multiple message digest standards: MD5/SHA1/SHA224/SHA256, 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

1.3.38 Memory Integrity Check (MIC) Engine

- Directly connected to AHB bus
- Automatic memory integrity check by constant checksum scanning
- Directly access SDRAM memory through M-bus
- 4K bytes of memory per checksum unit
- Each checksum unit can be individually enabled or skipped
- Support Fletcher's Checksum algorithm
- Programmable scanning rate and scanning range control
- Slight extra memory bandwidth and capacity demand
- Generate an interrupt whenever checksum errors are detected

1.3.39 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
- Three programmable conversion rates, divided from 24MHz clock source. The first one is for channels 0-5 and channels 7-13. The second one is for channel 6. The third one is for channels 14-15.
- 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

1.3.40 PWM Controller

- Support 8 PWM outputs
- Support 3 types of frequency mode PWM for fan speed control
- Duty cycle from 0 to 100% with 1/256 resolution
- Support low-frequency PWM pulse stretching for fan speed measurements
- Shared with GPIO pins

1.3.41 Fan Tachometer Controller

- Directly connected to APB 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

1.3.42 PECI Controller

- Directly connected to APB bus
- Intel PECI 3.1 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 32 bytes and receive buffer 32 bytes

1.3.43 JTAG Master Controller

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

1.3.44 MCTP Controller *(AST2520 Excluded)*

- Directly connected to APB bus
- Maximum packet length is 256 bytes (Baseline transmission unit)
- Independent send and receive message engine
- Flexible PCI-Express header
- Interrupt for receiving or sending completion
- Integrated 256-byte input buffer and 256-byte output buffer
- Support DMA option

1.3.45 MSI Controller *(AST2520 Excluded)*

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

1.3.46 X-DMA Controller *(AST2520 Excluded)*

- 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.

1.3.47 Software Specifications

- Manufacturer Test Program (under DOS)
- VGA BIOS and flash utility (under DOS)
- BMC flash utility under DOS, Linux and Windows
- System BIOS flash utility under DOS, Linux and Windows
- Windows Driver
  - Support Windows Server 2008 x86/x64 (with WHQL)
  - Support Windows Server 2008 R2 x64 (with WHQL)
  - Support Windows Server 2012 (with WHQL)
  - Support Windows Server 2012 R2 (with WHQL)
  - Support Windows 8 x64 (with WHQL)
- Linux X Window Driver
  - Support XFree86 4.x.x
  - Support XORG 6.8.x, 6.9.x, 7.x
- FreeBSD X Window Driver
  - Support XORG 6.9, 7.x
- Solaris 10/11 X86
  - Support XORG 6.8, 6.9, 7.x
- Supported Linux Distribution:
  - RHEL
  - SLES
  - Fedora
  - Open SUSE
  - Ubuntu
  - CentOS

1.4 Features Comparison (AST2500/AST2400/AST2300)

The following table shows the major feature comparisons between AST2500, AST2400 and AST2300 product specification.

Feature	AST2500	AST2400	AST2300
VGA/2D Controller	Yes	Yes	Yes
VGA Bus Interafce	PCI-Express Gen2	PCI-Express Gen1	PCI-Express Gen1
PCI-Express Master	PCI-Express Gen2	No	No
SOC Display Controller	Yes	Yes	Yes
Storage Redirection	Yes	Yes	Yes
KVM Redirection	Yes	Yes	Yes
Process Technology	TSMC 40nm LP	TSMC 0.13um	TSMC 0.13um
Package	19mmx19mm TFBGA	19mmx19mm LFBGA	19mmx19mm TFBGA
Pin Count Ball Pitch	456 Pins 0.8 mm	408 Pins 0.8 mm	408 Pins 0.8 mm
Peak Power Consumption (include DRAM)	< 2.1W	< 2.7W	< 3.0W
ARM926 Embedded CPU	No	400MHz (max)	400MHz (max)
ARM1176JZS	800MHz (max)	No	No
32Bit Coprocessor CPU	200MHz (max)	No	No
SDRAM Memory Bus Width	16 Bits	16 Bits	16 Bits
SDRAM Memory Types	DDR3L/DDR4	DDR2/DDR3	DDR2/DDR3
Maximum Memory Capacity	1024MB	512MB	512MB
Maximum Memory Clock Frequency	800MHz/1600Mbps	400MHz/800Mbps	400MHz/800Mbps
ECC Support	Yes (1/8 size)	Yes (1/8 size)	Yes (1/8 size)
Memory Integrity Check Engine	Yes	Yes	Yes
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 1.1 HID Controller	Yes (x1)	Yes (x1)	Yes (x1)
USB 2.0 Host Controller	Yes (x2)	Yes (x1)	No
USB 1.1 Host Ports	Yes (x2)	Yes (x2)	Yes (x1)
Hash Engine	Yes (MD5/SHA/RSA)	Yes (MD5/SHA/RSA)	Yes (MD5/SHA/RSA)
Crypto Engine	Yes (AES/RC4/DES/3DES)	Yes (AES/RC4/DES/3DES)	Yes (AES/RC4/DES/3DES)
I2C/SMBus Controller	Yes (x14)	Yes (x14)	Yes (x9)
PWM Outputs	Yes (x8)	Yes (x8)	Yes (x8)
Fan Tech	Yes (x16)	Yes (x16)	Yes (x16)
PECI 3.1	Yes	Yes	No
PECI IO Buffer	Yes	Yes	Yes
GPIO	228 (max)	216 (max)	152 (max)
SGPIO Master	80 (max)	80 (max)	64 (Max)
SGPIO Monitor	10 devices x2ch x2	10 devices x2ch	10 devices x 2ch
UART	Yes (x5) (Flow control x4)	Yes (x5) (Flow control x4)	Yes (x5) (Flow control x4)
UART Baudrate	921.6K	115.2K	115.2K
Virtual UART	Yes (1)	Yes (1)	Yes (1)
System UART Interface	Yes (4)	Yes (4)	Yes (2)
UART DMA	Yes	No	No
Hardware UART debug	Yes	No	No

to next page

from previous page

LPC Bus Controller	Yes (Slave & Master)	Yes (Slave & Master)	Yes (Slave & Master)
eSPI Bus Controller	Yes (Slave)	No	No
Watchdog Timer	Yes (x3)	Yes (x2)	Yes (x2)
Timer	Yes (x8)	Yes (x8)	Yes (x8)
Real Time Clock (RTC)	Yes	Yes	Yes
Digital Video Output	Yes (24-Bit Single Edge or 12-Bit Dual Edge)	Yes (24-Bit Single Edge or 12-Bit Dual Edge)	Yes (24-Bit Single Edge or 12-Bit Dual Edge)
Digital Video Input	Yes (24-Bit Single Edge)	Yes (24-Bit Single Edge)	Yes (24-Bit Single Edge)
Ethernet MAC Module	Yes (x2)	Yes (x2)	Yes (x2)
Ethernet MAC Throughput	10/100/1000M bps	10/100/1000M bps	10/100M/1000M bps
Ethernet MAC Interface	RGMII (x2) or RMII/NCSI (x2)	RGMII (x2) or RMII/NCSI (x2)	RGMII (x2) or RMII/NCSI (x2)
Flash Memory Controller	SPI Flash	SPI Flash NOR Flash (x8/x16) NAND Flash	SPI Flash NOR Flash (x8/x16) NAND Flash
SD/SDIO/eMMC 4-bits Interface or eMMC 8-bits Interface	Yes (x2) or Yes (x1)	Yes (x2) or No	Yes (x2) or No
Embedded SRAM	Yes (36KB)	Yes (32KB)	Yes (16KB)
Battery Backed SRAM	Yes (64 bytes)	No	No
Chip unique ID	Yes (64 bits)	No	No
System SPI Interface	Yes	Yes	Yes
MCTP function	Yes (with DMA)	Yes (with DMA)	Yes (without 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 (12 voltage)
JTAG-boundary scan	Yes	Yes	Yes
JTAG Master Function	Yes	Yes	Yes

1.5 Features Comparison (AST2520/AST1250/AST1050)

The following table shows the major feature comparisons between AST2520, AST1250 and AST1050 product specification.

Feature	AST2520	AST1250	AST1050
VGA/2D Controller	No	No	No
VGA Bus Interafce	No	No	No
PCI-Express Master	No	No	No
SOC Display Controller	Yes	Yes	Yes
Storage Redirection	Yes	Yes	Yes
KVM Redirection	No	No	No
Process Technology	TSMC 40nm LP	TSMC 0.13um	TSMC 0.13um
Package	19mmx19mm TFBGA	19mmx19mm LFBGA	19mmx19mm TFBGA
Pin Count Ball Pitch	456 Pins 0.8 mm	408 Pins 0.8 mm	408 Pins 0.8 mm
Peak Power Consumption (include DRAM)	< 1.8W	< 2.7W	< 3.0W
ARM926 Embedded CPU	No	400MHz (max)	400MHz (max)
ARM1176JZS	800MHz (max)	No	No
32Bit Coprocessor CPU	No	No	No
SDRAM Memory Bus Width	16 Bits	16 Bits	16 Bits
SDRAM Memory Types	DDR3L/DDR4	DDR2/DDR3	DDR2/DDR3
Maximum Memory Capacity	1024MB	512MB	512MB
Maximum Memory Clock Frequency	800MHz/1600Mbps	400MHz/800Mbps	400MHz/800Mbps
ECC Support	Yes (1/8 size)	Yes (1/8 size)	Yes (1/8 size)
Memory Integrity Check Engine	Yes	Yes	Yes
Maximum Graphics Display Resolutions	No	No	No
Maximum Video Clock Frequency	No	No	No
USB 2.0 Virtual Hub 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 (x1)	No
USB 1.1 Host Ports	Yes (x2)	Yes (x2)	Yes (x1)
Hash Engine	Yes (MD5/SHA/RSA)	Yes (MD5/SHA/RSA)	Yes (MD5/SHA/RSA)
Crypto Engine	Yes (AES/RC4/DES/3DES)	Yes (AES/RC4/DES/3DES)	Yes (AES/RC4/DES/3DES)
I2C/SMBus Controller	Yes (x14)	Yes (x14)	Yes (x9)
PWM Outputs	Yes (x8)	Yes (x8)	Yes (x8)
Fan Tech	Yes (x16)	Yes (x16)	Yes (x16)
PECI 3.1	Yes	Yes	No
PECI IO Buffer	Yes	Yes	Yes
GPIO	228 (max)	216 (max)	152 (max)
SGPIO Master	80 (max)	80 (max)	64 (Max)
SGPIO Monitor	10 devices x2ch x2	10 devices x2ch	10 devices x 2ch
UART	Yes (x5) (Flow control x4)	Yes (x5) (Flow control x4)	Yes (x5) (Flow control x4)
UART Baudrate	921.6K	115.2K	115.2K
Virtual UART	Yes (1)	Yes (1)	Yes (1)
System UART Interface	Yes (4)	Yes (4)	Yes (2)
UART DMA	Yes	No	No
Hardware UART debug	Yes	No	No
LPC Bus Controller	Yes (Slave & Master)	Yes (Slave & Master)	Yes (Slave & Master)

to next page

from previous page

eSPI Bus Controller	Yes (Slave)	No	No
Watchdog Timer	Yes (x3)	Yes (x2)	Yes (x2)
Timer	Yes (x8)	Yes (x8)	Yes (x8)
Real Time Clock (RTC)	Yes	Yes	Yes
Digital Video Output	No	No	No
Digital Video Input	No	No	No
Ethernet MAC Module	Yes (x2)	Yes (x2)	Yes (x2)
Ethernet MAC Throughput	10/100/1000M bps	10/100/1000M bps	10/100M/1000M bps
Ethernet MAC Interface	RGMII (x2) or RMII/NCSI (x2)	RGMII (x2) or RMII/NCSI (x2)	RGMII (x2) or RMII/NCSI (x2)
Flash Memory Controller	SPI Flash	SPI Flash NOR Flash (x8/x16) NAND Flash	SPI Flash NOR Flash (x8/x16) NAND Flash
SD/SDIO/eMMC 4-bits Interface or eMMC 8-bits Interface	Yes (x2) or Yes (x1)	Yes (x2) or No	Yes (x2) or No
Embedded SRAM	Yes (36KB)	Yes (32KB)	Yes (16KB)
Battery Backed SRAM	Yes (64 bytes)	No	No
Chip unique ID	Yes (64 bits)	No	No
System SPI Interface	Yes	Yes	Yes
MCTP function	No	No	No
MSI function	No	No	No
DMA between System and BMC memory	No	No	No
Super I/O Function	Yes	Yes	Yes
ADC Function	Yes (16 voltage)	Yes (16 voltage)	Yes (12 voltage)
JTAG-boundary scan	Yes	Yes	Yes
JTAG Master Function	Yes	Yes	Yes

1.6 Features Comparison (AST2500/AST2510/AST2520)

The following table shows the major feature comparisons between AST2500, AST2510 and AST2520 product specification.

Feature	AST2500	AST2510	AST2520
VGA/2D Controller	Yes	Yes	No
VGA Bus Interafce	PCI-Express Gen2	PCI-Express Gen2	No
PCI-Express Master	PCI-Express Gen2	No	No
SOC Display Controller	Yes	No	No
Storage Redirection	Yes	No	Yes
KVM Redirection	Yes	No	No
Peak Power Consumption (include DRAM)	< 2.1W	< 1.5W	< 1.8W
ARM1176JZS	800MHz (max)	No	800MHz (max)
32Bit Coprocessor CPU	200MHz (max)	200MHz (max)	No
ECC Support	Yes (1/8 size)	No	Yes (1/8 size)
Memory Integrity Check Engine	Yes	No	Yes
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)	No	Yes (x1)
USB 1.1 HID Controller	Yes (x1)	No	Yes (x1)
USB 2.0 Host Controller	Yes (x2)	No	Yes (x2)
USB 1.1 Host Ports	Yes (x2)	No	Yes (x2)
Hash Engine	Yes (MD5/SHA/RSA)	No	Yes (MD5/SHA/RSA)
Crypto Engine	Yes (AES/RC4/DES/3DES)	No	Yes (AES/RC4/DES/3DES)
SGPIO Master	80 (max)	80 (max)	80 (max)
SGPIO Monitor	10 devices x2ch x2	No	10 devices x2ch x2
Virtual UART	Yes (1)	No	Yes (1)
eSPI Bus Controller	Yes (Slave)	Yes (Slave/Partial)	Yes (Slave)
Real Time Clock (RTC)	Yes	No	Yes
Digital Video Output	Yes (24-Bit Single Edge or 12-Bit Dual Edge)	Yes (24-Bit Single Edge or 12-Bit Dual Edge)	No
Digital Video Input	Yes (24-Bit Single Edge)	No	No
Ethernet MAC Module	Yes (x2)	No	Yes (x2)
Ethernet MAC Throughput	10/100/1000M bps	No	10/100/1000M bps
Ethernet MAC Interface	RGMII (x2) or RMII/NCSI (x2)	No	RGMII (x2) or RMII/NCSI (x2)
SD/SDIO/eMMC Interface	Yes (x2)	No	Yes (x2)
MCTP function	Yes	No	No
MSI function	Yes	No	No
DMA between System and BMC memory	Yes	No	No

1.7 Features Comparison (AST2500/AST2520/AST2530)

The following table shows the major feature comparisons between AST2500, AST2520 and AST2530 product specification.

Feature	AST2500	AST2520	AST2530
VGA/2D Controller	Yes	No	Yes
VGA Bus Interafce	PCI-Express Gen2	No	PCI-Express Gen2
PCI-Express Master	PCI-Express Gen2	No	PCI-Express Gen2
KVM Redirection	Yes	No	Yes
Normal Power Consumption (include DRAM)	< 2.1W	< 1.8W	< 1.8W
32Bit Coprocessor CPU	200MHz (max)	No	No
Maximum Graphics Display Resolutions	1920x1200 32bpp@60Hz	No	1920x1200 32bpp@60Hz
Maximum Video Clock Frequency	165MHz	No	165MHz
UART	Yes (x5) (Flow control x4)	Yes (x5) (Flow control x4)	Yes (x13) (Flow control x4)
Digital Video Output	Yes (24-Bit Single Edge or 12-Bit Dual Edge)	No	Yes (24-Bit Single Edge or 12-Bit Dual Edge)
Digital Video Input	Yes (24-Bit Single Edge)	No	No
MCTP function	Yes	No	Yes
MSI function	Yes	No	Yes
DMA between System and BMC memory	Yes	No	Yes

1.8 Applications

Figure-3 illustrates the typical applications of the device in server applications.

1.8.1 Flash Interface Architecture

Figure-4 shows the flash interface architecture. It includes one main SPI flash controller for BMC firmware. And another 2 SPI interfaces, one supports for system BIOS applicatio, and the other for BMC extension. There are 2 types of scheme for BIOS application:

1. System access the SPI BIOS through LPC interface (Blue color path). When BMC want to update system BIOS, it can disable the LPC path.
2. System access the SPI BIOS through its own SPI interface (Red color path), then loopback through AST2500. Here AST2500 functions as a SPI multiplexer. It has embedded the external switch into AST2500. When BMC want to update system BIOS, it can switch the multiplexer output to the internal SPI interface.

The system BIOS SPI port is shared with the VGA BIOS SPI port. Because for the on-board application, VGA BIOS is always embedded in the system BIOS.

1.8.2 Remote BIOS Update

Figure-5 shows 3 methods of remotely update BIOS by BMC.

1. **Option 1: External Bus Switch** — Using an external bus switch to select the SPI BIOS flash, which is directed to Host system or BMC SPI controller interface. The switch internal delay will limit the maximum

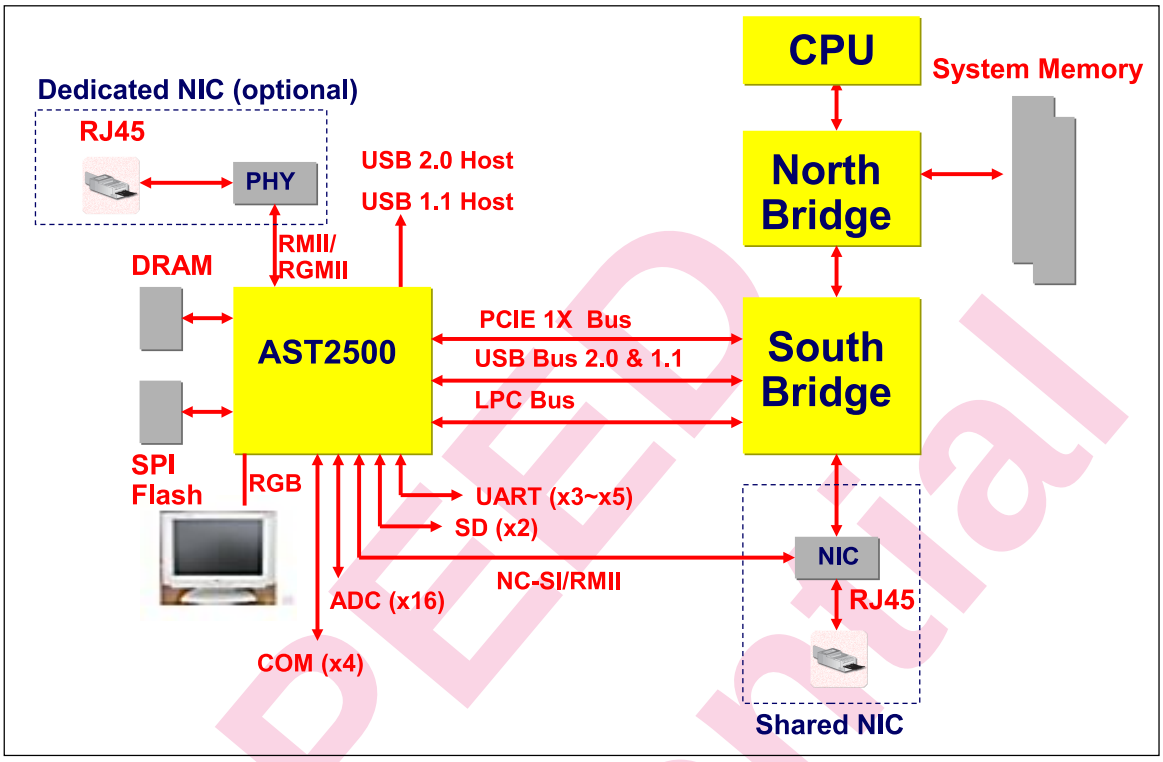


Figure 3: AST2500 system applications

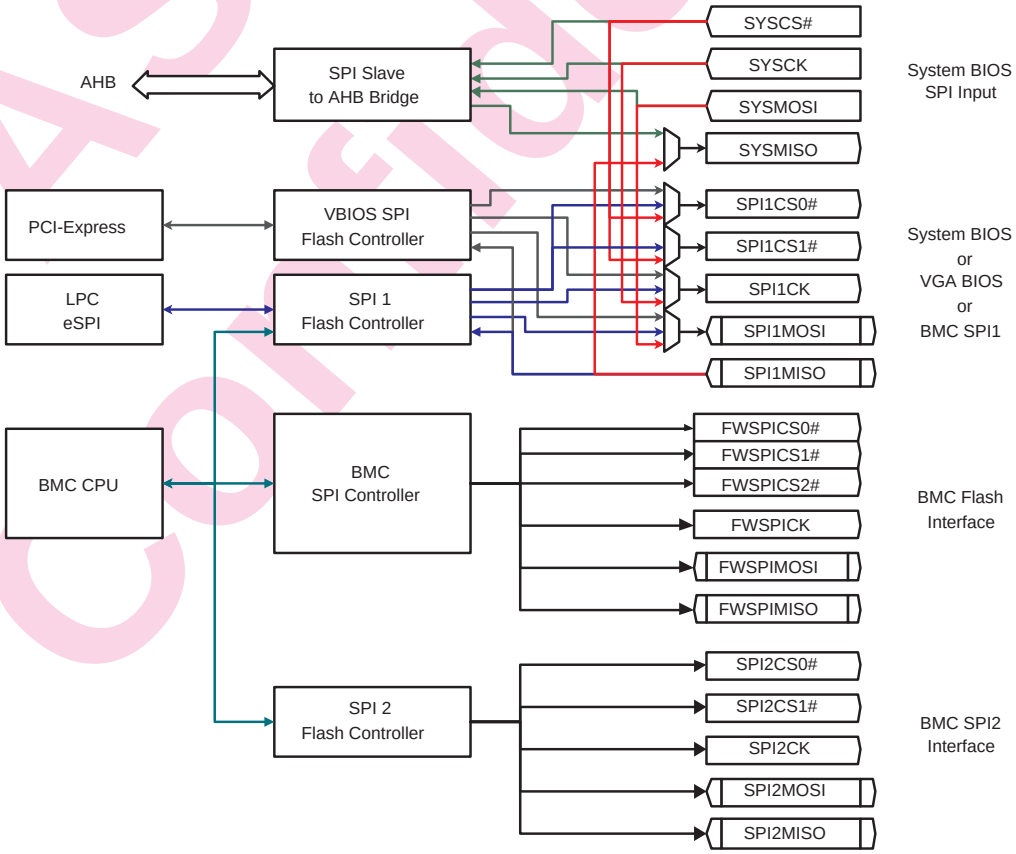


Figure 4: AST2500 flash interface architecture

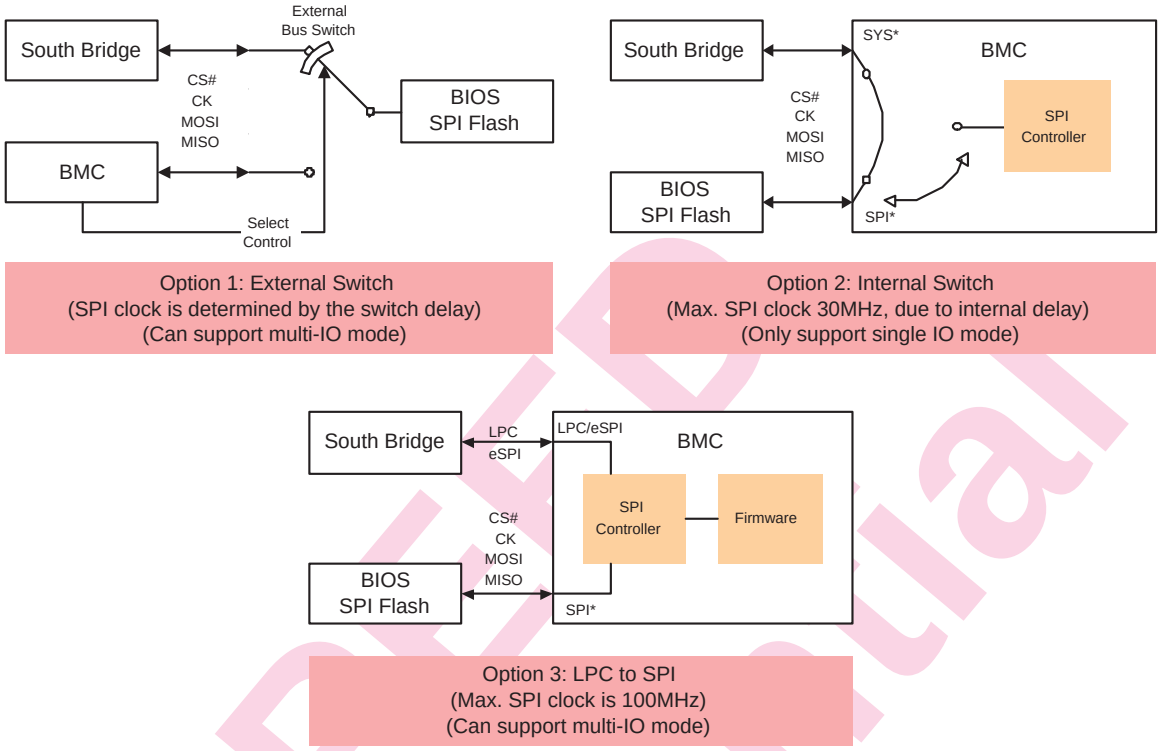


Figure 5: AST2500 Remote BIOS update application

allowable SPI clock speed. For SPI high speed requirement, it usually use this method. This method supports SPI multi-IO modes accessing.

2. Option 2: BMC Internal Switch — Merge the bus switch into BMC. The default SPI path can be selected by hardware strapping, with Host-to-BIOS or BMC-to-BIOS. The maximum SPI clock that can run by Host is about 30MHz, due to the long internal delay of about 7 ns. This method only support SPI single-IO mode accessing. For low cost and no high speed SPI accessing requirement, it can use this method.

3. Option 3: eSPI/LPC to SPI BIOS — Host use eSPI/LPC to access BIOS code through the SPI controller in BMC.

1.8.3 UART Interface Application

Figure-6 shows the UART interface routing architecture.
Figure-7 shows the UART interface architecture. AST2500 includes 5 UARTs(NS-16550) and 1 Virtual-UART. UART1 ~ UART4 are mapped to LPC slave controller to support 4 system UART ports, which UART1 and UART2 are default route to LPC. And Virtual-UART is also mapped to LPC as one system UART port. This port is used for Serial-over-Lan (SOL) function. Besides, SOL function also can be implemented in a legacy mode by physical UART cross connection, this is implemented in AST2500 with internal UART port cross link.

1.8.4 Display Output Interface

Figure-8 shows the display output paths that AST2500 supports. There are a total of 3 paths:

1. Path 1: VGA output, the output target can choose either or both to the DAC or DVO interface.

2. Path 2: Graphics CRT output, the output target can choose either or both to the DAC or DVO interface.

3. Path 3: Video input from DVO, the video input can be used for video engine capture or DAC display output.

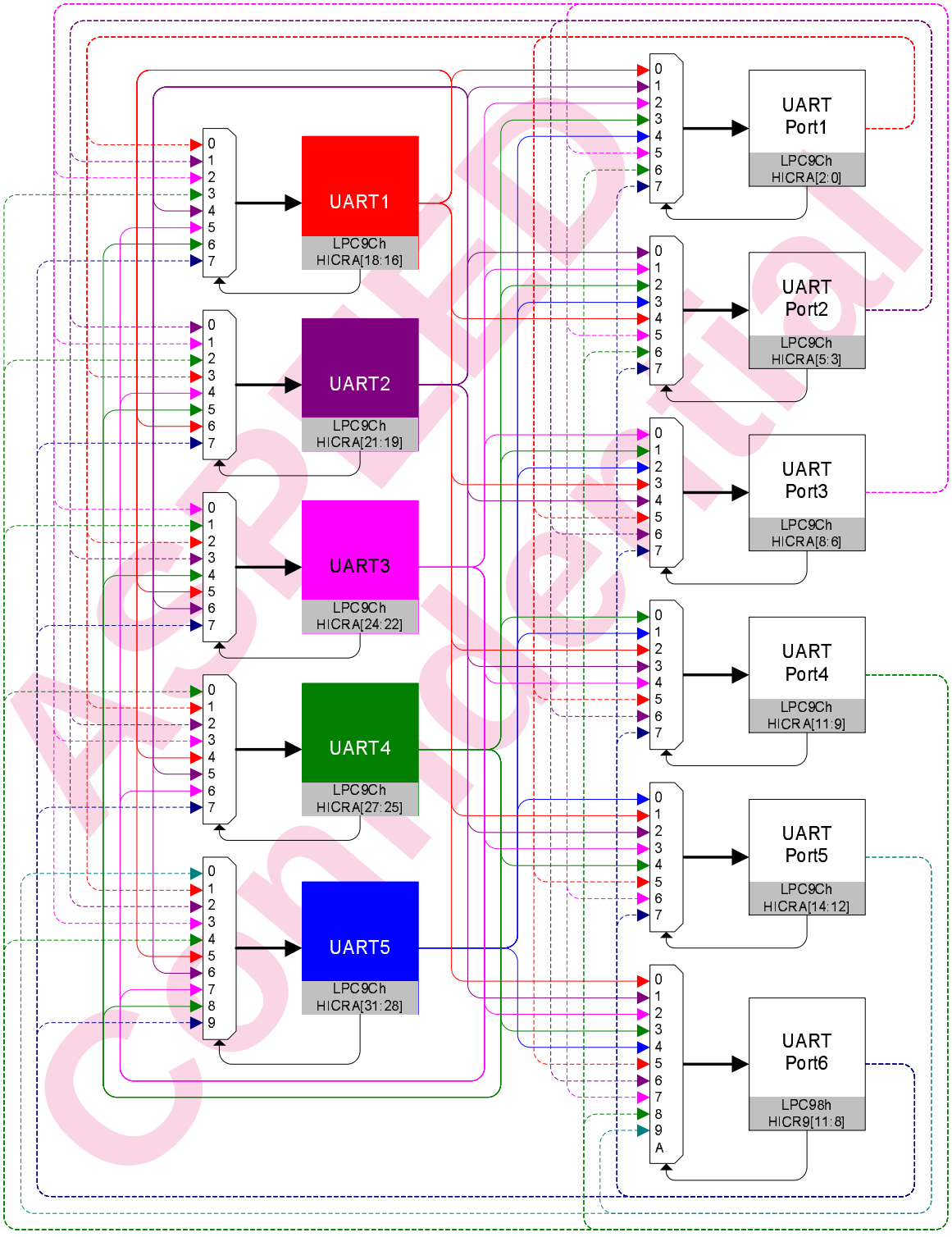


Figure 6: AST2500 UART interface control

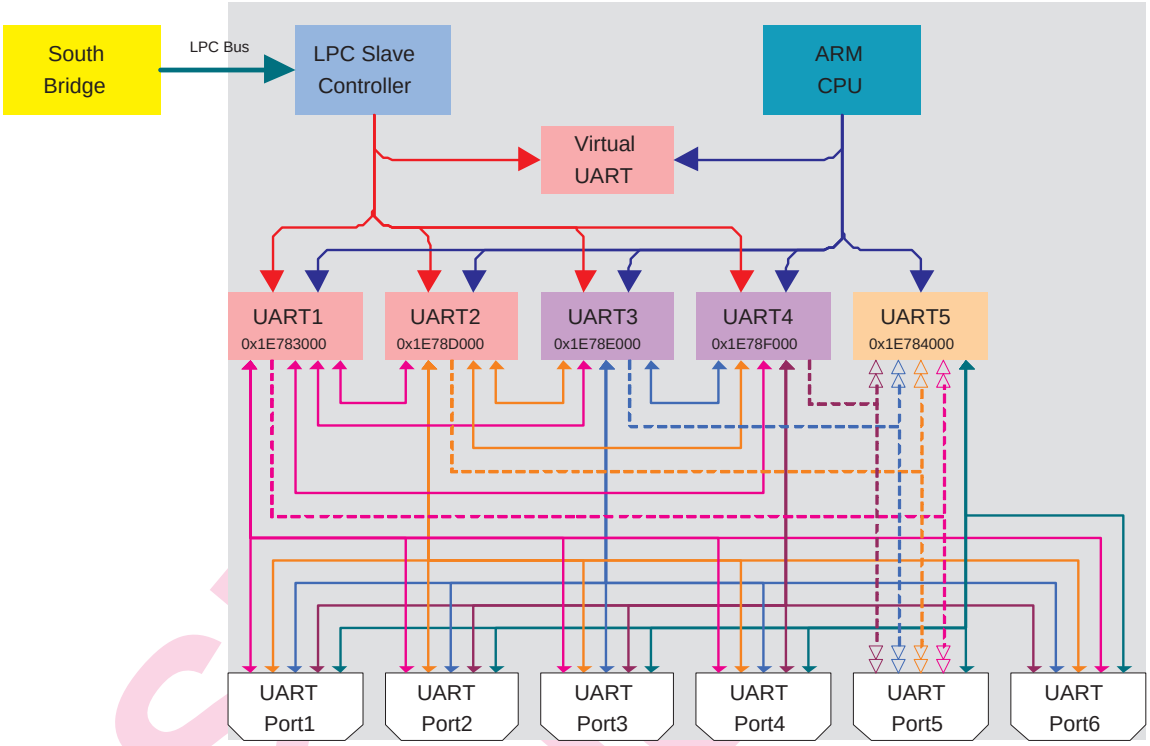


Figure 7: AST2500 UART interface architecture

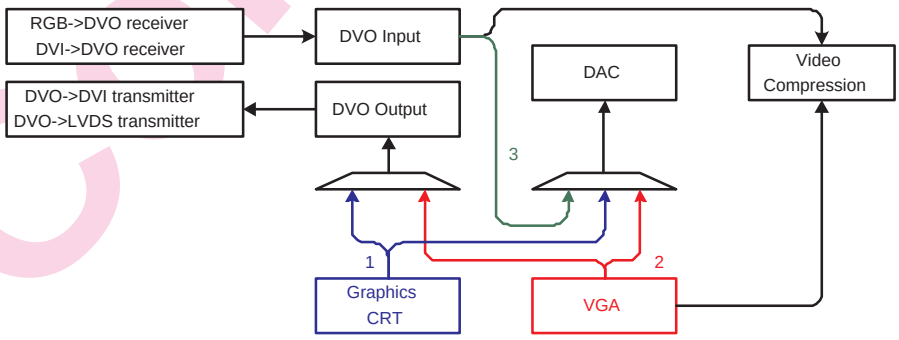


Figure 8: AST2500 display interface architecture

1.8.5 USB Port Configuration

AST2500 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.

1.9 Firmware Update Methods

1.9.1 Update through PCIe (AST2500/AST2510/AST2530)

ASPEED supports flash update utility ”SOCFlash” that can be run under DOS/Windows/Linux. This mode requires the PCIe port connected to the host.

1.9.2 Update through LPC (AST2500/AST2510/AST2520/AST2530)

ASPEED supports flash update utility ”SOCFlash” that can be run under DOS/Windows/Linux. This mode requires the LPC port connected to the host and enables the SuperIO decode feature.

1.9.3 Update through UART (AST2500/AST2510/AST2520/AST2530)

AST25x0 supports a new debug interface through UART (UART1 or UART5), which can work without firmware. The new debug interface also supports firmware update. ASPEED supports Windows version of ”SOCFlash” that can update through UART.

Since the UART speed is slow, to speed up the update process, AST25x0 supports 1Mbps baudrate when updating flash. To use this, hardware design should use the RS232 level-shifter that can work up to 1Mbps.

1.9.4 Update through LAN by Firmware (AST2500/AST2520/AST2530)

The BMC firmware itself supports firmware remote update over LAN.

1.9.5 Update through SPI programmer (AST2500/AST2510/AST2520/AST2530)

The last way to update the flash is to reserve a SPI programmer pin header on the PCB.

1. When wants to update the flash, the BMC should be disabled. Short FWSPICS# pin to GND before AC ON, and then remove the short Jump after AC ON.
2. Connect the USB SPI programmer download cable to the pin header.
3. Then it can update the SPI flash from the PC software operation.

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.
DDR	:	DDR3L/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 36.4.3 for the firmware programming method of GPIO push-pull and open-drain driving mode.

Pin Group List

DDR3L/DDR4 DRAM Interface – 53 pins
PCI Express Interface – 8 pins
LPC/ESPI Interface – 8 pins
RGMII/RMII/NCSI Dual Interface – 29 pins
UART Port – 42 pins
Firmware SPI Memory Interface – 6 pins
Extended SPI Memory Interface – 12 pins
I2C/SMBUS Interface – 20 pins
SD/SDIO/eMMC Interface – 16 pins
Serial GPIO Interface – 12 pins
VGA Interface – 4 pins
PWM/Fan Tachometer – 24 pins
GPIO Interface – 46 pins
PECI Port – 2 pins
JTAG Port – 6 pins
Miscellaneous – 5 pins
USB 2.0 Host/Slave Port – 7 pins
DAC – 6 pins
ADC – 20 pins
PLL Power – 8 pins

Power and Ground – 122 pins

DDR3L/DDR4 DRAM Interface – 53 pins

Ball	Signal	I/O	Type	Description
U10 W10 Y10 AA10 U9 Y9 W9 V9 AA8 Y8 Y7 W8 AA6 W7 V7 U7	MDQ0 MDQ1 MDQ2 MDQ3 MDQ4 MDQ5 MDQ6 MDQ7 MDQ8 MDQ9 MDQ10 MDQ11 MDQ12 MDQ13 MDQ14 MDQ15	I/O	DDR	**DRAM data bus**
U8 AB8	MDM0 MDM1	O	DDR	**DRAM byte mask bus**
AB9 AB7	MDQS0 MDQS1	I/O	DDR	**DRAM data bidirectional strobe pins**
AB10 AB6	MDQS0# MDQS1#	I/O	DDR	**DRAM data bidirectional strobe pins complement phase**
V13 AB14 W14 U14 W15 V15 AB16 AA16 W16 Y16 U13 AA14 Y14 AB15 Y15 U15	MA0 MA1 MA2 MA3 MA4 MA5 MA6 MA7 MA8 MA9 MA10 MA11 MA12 MA13 MA14/MACT# MA15	O	DDR	**DRAM address bus** **MACT# works on DDR4**
U12 Y13 W13	MBA0 MBA1 MBA2/MBG0	O	DDR	**DRAM bank address** MBG0 works on DDR4.

to next page

from previous page

AB11	MCK	O	DDR	**DRAM clock pin**
AB12	MCK#	O	DDR	**DRAM clock pin complement phase**
AA12	MCS#	O	DDR	**DRAM chip select pin**
W12	MRAS#	O	DDR	**DRAM row address select pin**
AB13	MCAS#	O	DDR	**DRAM column address select pin**
Y12	MWE#	O	DDR	**DRAM write enable pin**
Y11	MCKE	O	DDR	**DRAM clock enable control pin**
V11	MODT	O	DDR	**On-die termination enable control**
AB17	MRESET#	O	DDR	**DRAM reset pin**
U16	MALERT#	I/O	DDR	**DDR4 Alert function pin** NC this pin for DDR3 SDRAM.
W11	MIOZ	-	A	**IO calibration reference pin** An external 240 Ω±1% resistor should be connected between MIOZ and ground.
T9	MVREF	-	VREF	**IO reference voltage** DDR IO reference voltage input pin. The voltage is 0.5*MVDD.

PCI Express Interface – 8 pins

Ball	Signal	I/O	Type	Description
L20	PERST#	IDS/O8	CMOS	**PCI Express reset pin** This reset signal reset PCI Express bus controller and VGA/2D device.
K21	PEREFCLKP	I	A	**PCI Express Reference clock input** 100MHz positive input of the differential clock pair
K22	PEREFCLKN	I	A	**PCI Express Reference clock input** 100MHz negative input of the differential clock pair
M21	PERXP	I	A	**PCI Express Serial Data Receiver** It receives positive input of the differential signal pair.
M22	PERXN	I	A	**PCI Express Serial Data Receiver** It receives negative input of the differential signal pair.
L21	PETXP	O	A	**PCI Express Serial Data Transmitter** It transmits positive output of the differential signal pair.
L22	PETXN	O	A	**PCI Express Serial Data Transmitter** It transmits negative output of the differential signal pair.
K20	PEREXT	-	A	**External Reference Resistance** An external 200Ω±1% resistor should be connected between REXT and ground.

LPC/ESPI Interface – 8 pins

Ball	Signal	I/O	Type	Description
IO Power Domain : LPVDD				
G21	GPIOAC0	IUS/O12	CMOS	**GPIO group AC bit 0**
	ESPID0	IUS/O12		**eSPI data bus bit 0**
	LAD0	IUS/O12		**LPC address and data bus bit 0 (default)**
G20	GPIOAC1	IUS/O12	CMOS	**GPIO group AC bit 1**
	ESPID1	IUS/O12		**eSPI data bus bit 1**
	LAD1	IUS/O12		**LPC address and data bus bit 1 (default)**

to next page

from previous page

D22	GPIOAC2	IUS/O12	CMOS	**GPIO group AC bit 2**
	ESPID2	IUS/O12		**eSPI data bus bit 2**
	LAD2	IUS/O12		**LPC address and data bus bit 2 (default)**
E22	GPIOAC3	IUS/O12	CMOS	**GPIO group AC bit 3**
	ESPID3	IUS/O12		**eSPI data bus bit 3**
	LAD3	IUS/O12		**LPC address and data bus bit 3 (default)**
C22	GPIOAC4	IDS/O12	CMOS	**GPIO group AC bit 4**
	ESPICK	IDS		**eSPI clock input**
	LCLK	IDS		**LPC bus clock input (default)**
F21	GPIOAC5	IUS/O12	CMOS	**GPIO group AC bit 5**
	ESPICS#	IUS		**eSPI chip select input**
	LFRAME#	IUS		**LPC FRAME# (default)**
F22	GPIOAC6	IUS/O12	CMOS	**GPIO group AC bit 6**
	ESPIALT#	O12		**eSPI Alert**
	LSIRQ#	IUS/O12		**LPC serial IRQ (default)**
G22	GPIOAC7	IDS/O12	CMOS	**GPIO group AC bit 7**
	ESPIRST#	IDS		**eSPI reset input**
	LPCRST#	IDS		**LPC reset input (default)**

RGMII/RMII/NCSI Dual Interface – 29 pins

Ball	Signal	I/O	Type	Description
IO Power Domain : R1VDD				
B4	RGMII1RXCK	IR	CMOS	**RGMII 1 receive clock**
	RMII1RCLKI	IR		**RMII/NCSI 1 50MHz reference clock input**
	GPIOU4	IR/O8		**GPIO group U bit 4**
A4	RGMII1RXCTL	IR	CMOS	**RGMII 1 receive control**
	GPIOU5	ID/O8		**GPIO group U bit 5**
A3	RGMII1RXD0	IR	CMOS	**RGMII 1 receive data bus from PHY bit 0**
	RMII1RXD0	IR		**RMII/NCSI 1 receive data bus from PHY bit 0**
	GPIOU6	IR/O8		**GPIO group U bit 6**
D6	RGMII1RXD1	IR	CMOS	**RGMII 1 receive data bus from PHY bit 1**
	RMII1RXD1	IR		**RMII/NCSI 1 receive data bus from PHY bit 1**
	GPIOU7	IR/O8		**GPIO group U bit 7**
C5	RGMII1RXD2	IR	CMOS	**RGMII 1 receive data bus from PHY bit 2**
	RMII1CRSDV	IR		**RMII/NCSI 1 receive carrier sense and data valid**
	GPIOV0	IR/O8		**GPIO group V bit 0**
C4	RGMII1RXD3	IR	CMOS	**RGMII 1 receive data bus from PHY bit 3**
	RMII1RXER	IR		**RMII/NCSI 1 receive data error**
	GPIOV1	IR/O8		**GPIO group V bit 1**
B5	RGMII1TXCK	O8	CMOS	**RGMII 1 transmit clock**
	RMII1RCLKO	Op		**RMII/NCSI 1 50MHz reference clock output**
	GPIOT0	ID/Op		**GPIO group T bit 0** When used as RMII/NCSI 50MHz reference clock output, this pin should be connected to RMII1RCLKI and PHY RCLK input.

to next page

from previous page

E9	RGMII1TXCTL RMII1TXEN GPIOT1	O8 Op ID/Op	CMOS	**RGMII 1 transmit control** **RMII/NCSI 1 transmit enable** **GPIO group T bit 1** Hardware strap bit6.
F9	RGMII1TXD0 RMII1TXD0 GPIOT2	O8 Op ID/Op	CMOS	**RGMII 1 transmit data bus to PHY bit 0** **RMII/NCSI 1 transmit data bus to PHY bit 0** **GPIO group T bit 2** Hardware strap bit12.
A5	RGMII1TXD1 RMII1TXD1 GPIOT3	O8 Op ID/Op	CMOS	**RGMII 1 transmit data bus to PHY bit 1** **RMII/NCSI 1 transmit data bus to PHY bit 1** **GPIO group T bit 3** Hardware strap bit13.
E7	RGMII1TXD2 GPIOT4	O8 ID/O8	CMOS	**RGMII 1 transmit data bus to PHY bit 2** **GPIO group T bit 4** Hardware strap bit16.
D7	RGMII1TXD3 GPIOT5	O8 ID/O8	CMOS	**RGMII 1 transmit data bus to PHY bit 3** **GPIO group T bit 5** Hardware strap bit19.
IO Power Domain : R2VDD				
C2	RGMII2RXCK RMII2RCLKI GPIOV2	IR IR IR/O8	CMOS	**RGMII 2 receive clock** **RMII/NCSI 2 50MHz reference clock input** **GPIO group V bit 2**
C1	RGMII2RXCTL GPIOV3	IR ID/O8	CMOS	**RGMII 2 receive control** **GPIO group V bit 3**
C3	RGMII2RXD0 RMII2RXD0 GPIOV4	IR IR IR/O8	CMOS	**RGMII 2 receive data bus from PHY bit 0** **RMII/NCSI 2 receive data bus from PHY bit 0** **GPIO group V bit 4**
D1	RGMII2RXD1 RMII2RXD1 GPIOV5	IR IR IR/O8	CMOS	**RGMII 2 receive data bus from PHY bit 1** **RMII/NCSI 2 receive data bus from PHY bit 1** **GPIO group V bit 5**
D2	RGMII2RXD2 RMII2CRSDV GPIOV6	IR IR IR/O8	CMOS	**RGMII 2 receive data bus from PHY bit 2** **RMII/NCSI 2 receive carrier sense and data valid** **GPIO group V bit 6**
E6	RGMII2RXD3 RMII2RXER GPIOV7	IR IR IR/O8	CMOS	**RGMII 2 receive data bus from PHY bit 3** **RMII/NCSI 2 receive data error** **GPIO group V bit 7**
B2	RGMII2TXCK RMII2RCLKO GPIOT6	O8 Op ID/Op	CMOS	**RGMII 2 transmit clock** **RMII/NCSI 2 50MHz reference clock output** **GPIO group T bit 6** When used as RMII/NCSI 50MHz reference clock output, this pin should be connected to RMII2RCLKI and PHY RCLK input.
B1	RGMII2TXCTL RMII2TXEN	O8 Op	CMOS	**RGMII 2 transmit control** **RMII/NCSI 2 transmit enable**

to next page

from previous page

	GPIOT7	ID/Op		**GPIO group T bit 7** Hardware strap bit7.
A2	RGMII2TXD0 RMII2TXD0 GPIOU0	O8 Op ID/Op	CMOS	**RGMII 2 transmit data bus to PHY bit 0** **RMII/NCSI 2 transmit data bus to PHY bit 0** **GPIO group U bit 0** Hardware strap bit21.
B3	RGMII2TXD1 RMII2TXD1 GPIOU1	O8 Op ID/Op	CMOS	**RGMII 2 transmit data bus to PHY bit 1** **RMII/NCSI 2 transmit data bus to PHY bit 1** **GPIO group U bit 1** Hardware strap bit22.
D5	RGMII2TXD2 GPIOU2	O8 ID/O8	CMOS	**RGMII 2 transmit data bus to PHY bit 2** **GPIO group U bit 2** Hardware strap bit23.
D4	RGMII2TXD3 GPIOU3	O8 ID/O8	CMOS	**RGMII 2 transmit data bus to PHY bit 3** **GPIO group U bit 3** Hardware strap bit24.
D3	RGMIICK	ID	CMOS	**RGMII external 125MHz reference clock input** RGMII 125MHz can be generated internally. When use internal source, this pin can be NC.

Note :
RGMII and RMII/NCSI IO mode was determined by hardware strap bit[7:6].
Bit[6] = 0 : RMII mode, R1VDD = 3.3V
Bit[6] = 1 : RGMII mode, R1VDD = 3.3V/2.5V
Bit[7] = 0 : RMII mode, R2VDD = 3.3V
Bit[7] = 1 : RGMII mode, R2VDD = 3.3V/2.5V

When the MAC function pins not used, they can be used as GPIO, or pull low all the input type pins and open connect (NC) the output type pins.

The driving strength of RGMII TX pins can be selected at SCU90 bit[11:8].

IO Power Domain : PV33D

D8	GPIOR6 MDC1	ID/O8 O8	CMOS	**GPIO group R bit 6 (default)** **MAC1 management interface clock output**
E10	GPIOR7 MDIO1	ID/O8 ID/O8	CMOS	**GPIO group R bit 7 (default)** **MAC1 management interface data input/output** When works as MDIO, requires external 4.7K pull high to PV33D.
C13	GPIOA6 TIMER7 MDC2	IDS/O8 O8 O8	CMOS	**GPIO group A bit 6 (default)** **Timer 7 programmable duty cycle pulse output** **MAC2 management interface clock output**
B13	GPIOA7 TIMER8 MDIO2	IDS/O8 O8 IDS/O8	CMOS	**GPIO group A bit 7 (default)** **Timer 8 programmable duty cycle pulse output** **MAC2 management interface data input/output** When works as MDIO, requires external 4.7K pull high to PV33D.

UART Port – 42 pins

Ball	Signal	I/O	Type	Description
		UART port 1		

to next page

from previous page

T2	GPIOL0	ID/O8	CMOS	**GPIO group L bit 0 (default)**
	NCTS1	ID		**UART 1 clear to send modem status**
T1	GPIOL1	ID/O8	CMOS	**GPIO group L bit 1 (default)** GPIOL1 supports fast interrupt capability.
	NDCD1	ID		**UART 1 data carrier detect modem status**
	VPIDE	ID		**Digital video input port: DE**
U1	GPIOL2	ID/O8	CMOS	**GPIO group L bit 2 (default)**
	NDSR1	ID		**UART 1 data set ready modem status**
U2	GPIOL3	ID/O8	CMOS	**GPIO group L bit 3 (default)** GPIOL3 supports fast interrupt capability.
	NRI1	ID		**UART 1 ring indicator modem status**
	VPIHS	ID		**Digital video input port: H-SYNC**
P4	GPIOL4	ID/O8	CMOS	**GPIO group L bit 4 (default)**
	NDTR1	O8		**UART 1 data terminate ready modem status**
	VPIVS	ID		**Digital video input port: V-SYNC**
P3	GPIOL5	ID/O8	CMOS	**GPIO group L bit 5 (default)**
	NRTS1	O8		**UART 1 request to send modem status**
	VPICLK	ID		**Digital video input port: CLK**
V1	GPIOL6	ID/O8	CMOS	**GPIO group L bit 6 (default)**
	TXD1	O8		**UART 1 transmit serial data output**
W1	GPIOL7	ID/O8	CMOS	**GPIO group L bit 7 (default)**
	RXD1	ID		**UART 1 receive serial data input**
				UART port 2
Y1	GPIOM0	ID/O8	CMOS	**GPIO group M bit 0 (default)**
	NCTS2	ID		**UART 2 clear to send modem status**
	VPIB2	ID		**Digital video input port: B2**
AB2	GPIOM1	ID/O8	CMOS	**GPIO group M bit 1 (default)**
	NDCD2	ID		**UART 2 data carrier detect modem status**
	VPIB3	ID		**Digital video input port: B3**
AA1	GPIOM2	ID/O8	CMOS	**GPIO group M bit 2 (default)**
	NDSR2	ID		**UART 2 data set ready modem status**
	VPIB4	ID		**Digital video input port: B4**
Y2	GPIOM3	ID/O8	CMOS	**GPIO group M bit 3 (default)**
	NRI2	ID		**UART 2 ring indicator modem status**
	VPIB5	ID		**Digital video input port: B5**
AA2	GPIOM4	ID/O8	CMOS	**GPIO group M bit 4 (default)**
	NDTR2	O8		**UART 2 data terminate ready modem status**
	VPIB6	ID		**Digital video input port: B6**
P5	GPIOM5	ID/O8	CMOS	**GPIO group M bit 5 (default)**
	NRTS2	O8		**UART 2 request to send modem status**
	VPIB7	ID		**Digital video input port: B7**
R5	GPIOM6	ID/O8	CMOS	**GPIO group M bit 6 (default)**
	TXD2	O8		**UART 2 transmit serial data output**

to next page

from previous page

	VPIB8	ID		**Digital video input port: B8**
T5	GPIOM7	ID/O8	CMOS	**GPIO group M bit 7 (default)**
	RXD2	ID		**UART 2 receive serial data input**
	VPIB9	ID		**Digital video input port: B9**
				UART port 3
B20	GPIOE0	ID/O8	CMOS	**GPIO group E bit 0 (default)**
	NCTS3	ID		**UART 3 clear to send modem status**
C20	GPIOE1	ID/O8	CMOS	**GPIO group E bit 1 (default)**
	NDCD3	ID		**UART 3 data carrier detect modem status**
F18	GPIOE2	ID/O8	CMOS	**GPIO group E bit 2 (default)**
	NDSR3	ID		**UART 3 data set ready modem status**
F17	GPIOE3	ID/O8	CMOS	**GPIO group E bit 3 (default)**
	NRI3	ID		**UART 3 ring indicator modem status**
E18	GPIOE4	ID/O8	CMOS	**GPIO group E bit 4 (default)**
	NDTR3	O8		**UART 3 data terminate ready modem status**
D19	GPIOE5	ID/O8	CMOS	**GPIO group E bit 5 (default)**
	NRTS3	O8		**UART 3 request to send modem status**
A20	GPIOE6	ID/O8	CMOS	**GPIO group E bit 6 (default)**
	TXD3	O8		**UART 3 transmit serial data output**
B19	GPIOE7	ID/O8	CMOS	**GPIO group E bit 7 (default)**
	RXD3	ID		**UART 3 receive serial data input**
				UART port 4
J19	GPIOF0	ID/O8	CMOS	**GPIO group F bit 0 (default)**
	NCTS4	ID		**UART 4 clear to send modem status**
	LHAD0	ID/O8		**LPC Host/Plus AD bus bit0**
J18	GPIOF1	ID/O8	CMOS	**GPIO group F bit 1 (default)**
	NDCD4	ID		**UART 4 data carrier detect modem status**
	LHAD1	ID/O8		**LPC Host/Plus AD bus bit1**
B22	GPIOF2	ID/O8	CMOS	**GPIO group F bit 2 (default)**
	NDSR4	ID		**UART 4 data set ready modem status**
	LHAD2	ID/O8		**LPC Host/Plus AD bus bit2**
B21	GPIOF3	ID/O8	CMOS	**GPIO group F bit 3 (default)**
	NRI4	ID		**UART 4 ring indicator modem status**
	LHAD3	ID/O8		**LPC Host/Plus AD bus bit3**
A21	GPIOF4	ID/O8	CMOS	**GPIO group F bit 4 (default)**
	NDTR4	O8		**UART 4 data terminate ready modem status**
	LHCLK	ID/O8		**LPC Host/Plus clock input/output**
H19	GPIOF5	ID/O8	CMOS	**GPIO group F bit 5 (default)**
	NRTS4	O8		**UART 4 request to send modem status**
	LHFRAME#	O8		**LPC Host/Plus FRAME#**
G17	GPIOF6	ID/O8	CMOS	**GPIO group F bit 6 (default)**
	TXD4	O8		**UART 4 transmit serial data output**
	LHSIRQ#	ID/O8		**LPC Host SERIRQ#**

to next page

from previous page

H18	GPIOF7 RXD4 LHRST#	ID/O8 ID ID/O8	CMOS	**GPIO group F bit 7 (default)** **UART 4 receive serial data input** **LPC Host/Plus reset input/output**
		UART port 5		
K1	TXD5	IU/O8	CMOS	**UART 5 transmit serial data output** This pin also works as a hardware strap setting to select UART debug port. Strap value '0': select UART1, external 1K ~ 4.7K pull low resistor required. Strap value '1': select UART5, no external pull resistor required.
K2	RXD5	IU	CMOS	**UART 5 receive serial data input**
		UART port 6		
A18	GPIOH0 NCTS6	ID/O8 ID	CMOS	**GPIO group H bit 0 (default)** **UART 6 clear to send modem status**
B18	GPIOH1 NDCD6	ID/O8 ID	CMOS	**GPIO group H bit 1 (default)** **UART 6 data carrier detect modem status**
D17	GPIOH2 NDSR6	ID/O8 ID	CMOS	**GPIO group H bit 2 (default)** **UART 6 data set ready modem status**
C17	GPIOH3 NRI6	ID/O8 ID	CMOS	**GPIO group H bit 3 (default)** **UART 6 ring indicator modem status**
A17	GPIOH4 NDTR6	ID/O8 O8	CMOS	**GPIO group H bit 4 (default)** **UART 6 data terminate ready modem status**
B17	GPIOH5 NRTS6	ID/O8 O8	CMOS	**GPIO group H bit 5 (default)** **UART 6 request to send modem status**
A16	GPIOH6 TXD6	ID/O8 O8	CMOS	**GPIO group H bit 6 (default)** **UART 6 transmit serial data output**
D18	GPIOH7 RXD6	ID/O8 ID	CMOS	**GPIO group H bit 7 (default)** **UART 6 receive serial data input**

Firmware SPI Memory Interface – 6 pins

Ball	Signal	I/O	Type	Description
AB19	FWSPICS0#	IU/O8	CMOS	**Firmware booting SPI memory chip select 0** Default booting chip select. Hardware strap bit0.
AA18	FWSPICK	ID/O8	CMOS	**Firmware booting SPI memory clock output** Hardware strap bit30.
U17	FWSPIMOSI	ID/O8	CMOS	**Firmware booting SPI memory MOSI** SPI flash data MOSI output (single bit IO) SPI flash data D0 input/output (dual bit IO) Hardware strap bit31.
T18	FWSPIMISO	ID/O8	CMOS	**Firmware booting SPI memory MISO** SPI flash data MISO input (single bit IO) SPI flash data D1 input/output (dual bit IO) Hardware strap bit25.
AA19	GPIOR0 FWSPICS1#	ID/O8 O8	CMOS	**GPIO group R bit 0** **Firmware booting SPI memory chip select 1 (default)** Default 2nd boot chip select.

to next page

from previous page

T19	GPIOR1	ID/O8	CMOS	**GPIO group R bit 1**
	FWSPICS2#	O8		**Firmware booting SPI memory chip select 2 (default)**

Extended SPI Memory Interface – 12 pins

Ball	Signal	I/O	Type	Description
C18	GPIOI0	ID/O8	CMOS	**GPIO group I bit 0 (default)**
	SYSCS#	ID		**System SPI pass-through input interface CS input**
E15	GPIOI1	ID/O8	CMOS	**GPIO group I bit 1 (default)**
	SYSCK	ID		**System SPI pass-through input interface CK input**
B16	GPIOI2	ID/O8	CMOS	**GPIO group I bit 2 (default)**
	SYSMOSI	ID		**System SPI pass-through input interface MOSI input**
C16	GPIOI3	ID/O8	CMOS	**GPIO group I bit 3 (default)**
	SYSMISO	O8		**System SPI pass-through input interface MISO output**
B15	GPIOI4	ID/O8	CMOS	**GPIO group I bit 4 (default)**
	SPI1CS0#	O8		**SPI1 master interface CS output channel 0**
	VBCS#	O8		**VGA BIOS SPI flash CS output**
C15	GPIOI5	ID/O8	CMOS	**GPIO group I bit 5 (default)**
	SPI1CK	O8		**SPI1 master interface CK output**
	VBCK	O8		**VGA BIOS SPI flash CK output**
A14	GPIOI6	ID/O8	CMOS	**GPIO group I bit 6 (default)**
	SPI1MOSI	ID/O8		**SPI1 master interface MOSI output/input**
	VBMOSI	O8		**VGA BIOS SPI flash MOSI output**
A15	GPIOI7	ID/O8	CMOS	**GPIO group I bit 7 (default)**
	SPI1MISO	ID/O8		**SPI1 master interface MISO input/output**
	VBMISO	ID		**VGA BIOS SPI flash MISO input**
Note : The default pin function of GPIOI is determined by hardware strap bit[13:12]				
T17	GPIOR2	ID/O8	CMOS	**GPIO group R bit 2 (default)**
	SPI2CS0#	O8		**SPI2 master interface CS output channel 0**
Y19	GPIOR3	ID/O8	CMOS	**GPIO group R bit 3 (default)**
	SPI2CK	O8		**SPI2 master interface CK output**
W19	GPIOR4	ID/O8	CMOS	**GPIO group R bit 4 (default)**
	SPI2MOSI	ID/O8		**SPI2 master interface MOSI output/input**
V19	GPIOR5	ID/O8	CMOS	**GPIO group R bit 5 (default)**
	SPI2MISO	ID/O8		**SPI2 master interface MISO input/output**

I2C/SMBUS Interface – 20 pins

Ball	Signal	I/O	Type	Description
M18	GPIOY4	IDS/O8	CMOS	**GPIO group Y bit 4**
	SCL1	IS/O8		**I2C/SMBUS 1 clock pin (default)**
M19	GPIOY5	IDS/O8	CMOS	**GPIO group Y bit 5**
	SDA1	IS/O8		**I2C/SMBUS 1 data pin (default)**
M20	GPIOY6	IDS/O8	CMOS	**GPIO group Y bit 6**
	SCL2	IS/O8		**I2C/SMBUS 2 clock pin (default)**

to next page

from previous page

P20	GPIOY7 SDA2	IDS/O8 IS/O8	CMOS	**GPIO group Y bit 7** **I2C/SMBUS 2 data pin (default)**
A11	GPIOQ0 SCL3	IDS/O8 IS/O8	CMOS	**GPIO group Q bit 0 (default)** **I2C/SMBUS 3 clock pin**
A10	GPIOQ1 SDA3	IDS/O8 IS/O8	CMOS	**GPIO group Q bit 1 (default)** **I2C/SMBUS 3 data pin**
A9	GPIOQ2 SCL4	IDS/O8 IS/O8	CMOS	**GPIO group Q bit 2 (default)** **I2C/SMBUS 4 clock pin**
B9	GPIOQ3 SDA4	IDS/O8 IS/O8	CMOS	**GPIO group Q bit 3 (default)** **I2C/SMBUS 4 data pin**
L3	GPIOK0 SCL5	IDS/O8 IS/O8	CMOS	**GPIO group K bit 0 (default)** **I2C/SMBUS 5 clock pin**
L4	GPIOK1 SDA5	IDS/O8 IS/O8	CMOS	**GPIO group K bit 1 (default)** **I2C/SMBUS 5 data pin**
L1	GPIOK2 SCL6	IDS/O8 IS/O8	CMOS	**GPIO group K bit 2 (default)** **I2C/SMBUS 6 clock pin**
N2	GPIOK3 SDA6	IDS/O8 IS/O8	CMOS	**GPIO group K bit 3 (default)** **I2C/SMBUS 6 data pin**
N1	GPIOK4 SCL7	IDS/O8 IS/O8	CMOS	**GPIO group K bit 4 (default)** **I2C/SMBUS 7 clock pin**
P1	GPIOK5 SDA7	IDS/O8 IS/O8	CMOS	**GPIO group K bit 5 (default)** **I2C/SMBUS 7 data pin**
P2	GPIOK6 SCL8	IDS/O8 IS/O8	CMOS	**GPIO group K bit 6 (default)** **I2C/SMBUS 8 clock pin**
R1	GPIOK7 SDA8	IDS/O8 IS/O8	CMOS	**GPIO group K bit 7 (default)** **I2C/SMBUS 8 data pin**
C14	GPIOA4 TIMER5 SCL9	IDS/O8 O8 IS/O8	CMOS	**GPIO group A bit 4 (default)** **Timer 5 programmable timer pulse output** **I2C/SMBUS 9 clock pin**
A13	GPIOA5 TIMER6 SDA9	IDS/O8 O8 IS/O8	CMOS	**GPIO group A bit 5 (default)** **Timer 6 programmable timer pulse output** **I2C/SMBUS 9 data pin**
N21	GPIOQ4 SCL14	IDS/O8 IS/O8	CMOS	**GPIO group Q bit 4 (default)** **I2C/SMBUS 14 clock pin**
N22	GPIOQ5 SDA14	IDS/O8 IS/O8	CMOS	**GPIO group Q bit 5 (default)** **I2C/SMBUS 14 data pin**

SD/SDIO/eMMC Interface – 16 pins

Ball	Signal	I/O	Type	Description
C12	GPIOC0 SD1CLK SCL10	IDS/O8 O8 IS/O8	CMOS	**GPIO group C bit 0 (default)** **SD 1 clock output** **I2C/SMBUS 10 clock pin**
A12	GPIOC1 SD1CMD	IDS/O8 IDS/O8	CMOS	**GPIO group C bit 1 (default)** **SD 1 command input/output**

to next page

from previous page

	SDA10	IS/O8		**I2C/SMBUS 10 data pin**
B12	GPIOC2	IDS/O8	CMOS	**GPIO group C bit 2 (default)**
	SD1DAT0	IDS/O8		**SD 1 data bus bit 0**
	SCL11	IS/O8		**I2C/SMBUS 11 clock pin**
D9	GPIOC3	IDS/O8	CMOS	**GPIO group C bit 3 (default)**
	SD1DAT1	IDS/O8		**SD 1 data bus bit 1**
	SDA11	IS/O8		**I2C/SMBUS 11 data pin**
D10	GPIOC4	IDS/O8	CMOS	**GPIO group C bit 4 (default)**
	SD1DAT2	IDS/O8		**SD 1 data bus bit 2**
	SCL12	IS/O8		**I2C/SMBUS 12 clock pin**
E12	GPIOC5	IDS/O8	CMOS	**GPIO group C bit 5 (default)**
	SD1DAT3	IDS/O8		**SD 1 data bus bit 3**
	SDA12	IS/O8		**I2C/SMBUS 12 data pin**
C11	GPIOC6	IDS/O8	CMOS	**GPIO group C bit 6 (default)**
	SD1CD#	IDS		**SD 1 data card detection input**
	SCL13	IS/O8		**I2C/SMBUS 13 clock pin**
B11	GPIOC7	IDS/O8	CMOS	**GPIO group C bit 7 (default)**
	SD1WP#	IDS		**SD 1 write protect input**
	SDA13	IS/O8		**I2C/SMBUS 13 data pin**
F19	GPIOD0	ID/O8	CMOS	**GPIO group D bit 0 (default)**
	SD2CLK	O8		**SD 2 clock output**
E21	GPIOD1	ID/O8	CMOS	**GPIO group D bit 1 (default)**
	SD2CMD	ID/O8		**SD 2 command input/output**
F20	GPIOD2	ID/O8	CMOS	**GPIO group D bit 2 (default)**
	SD2DAT0	ID/O8		**SD 2 data bus bit 0**
D20	GPIOD3	ID/O8	CMOS	**GPIO group D bit 3 (default)**
	SD2DAT1	ID/O8		**SD 2 data bus bit 1**
D21	GPIOD4	ID/O8	CMOS	**GPIO group D bit 4 (default)**
	SD2DAT2	ID/O8		**SD 2 data bus bit 2**
E20	GPIOD5	ID/O8	CMOS	**GPIO group D bit 5 (default)**
	SD2DAT3	ID/O8		**SD 2 data bus bit 3**
G18	GPIOD6	ID/O8	CMOS	**GPIO group D bit 6 (default)**
	SD2CD#	ID		**SD 2 data card detection input**
C21	GPIOD7	ID/O8	CMOS	**GPIO group D bit 7 (default)**
	SD2WP#	ID		**SD 2 write protect input**

Serial GPIO Interface – 12 pins

Ball	Signal	I/O	Type	Description
R2	GPIOJ0	ID/O8	CMOS	**GPIO group J bit 0 (default)**
	SGPMCK	O8		**Master Serial GPIO clock output**
L2	GPIOJ1	ID/O8	CMOS	**GPIO group J bit 1 (default)**
	SGPMLD	O8		**Master Serial GPIO parallel data load output**
N3	GPIOJ2	ID/O8	CMOS	**GPIO group J bit 2 (default)**
	SGPMO	O8		**Master Serial GPIO serial data output**

to next page

from previous page

N4	GPIOJ3 SGPMI	ID/O8 ID	CMOS	**GPIO group J bit 3 (default)** **Master Serial GPIO serial data input**
A19	GPIOG0 SGPS1CK	ID/O8 ID	CMOS	**GPIO group G bit 0 (default)** **Slave serial GPIO bus 1 clock input**
E19	GPIOG1 SGPS1LD	ID/O8 ID	CMOS	**GPIO group G bit 1 (default)** **Slave serial GPIO bus 1 parallel data load input**
C19	GPIOG2 SGPS1I0	ID/O8 ID	CMOS	**GPIO group G bit 2 (default)** **Slave serial GPIO bus 1 serial data input channel 0**
E16	GPIOG3 SGPS1I1	ID/O8 ID	CMOS	**GPIO group G bit 3 (default)** **Slave serial GPIO bus 1 serial data input channel 1**
E17	GPIOG4 SGPS2CK SALT1	ID/O8 ID ID/O8	CMOS	**GPIO group G bit 4 (default)** **Slave serial GPIO bus 2 clock input** **SMBus 1 Alert pin**
D16	GPIOG5 SGPS2LD SALT2	ID/O8 ID ID/O8	CMOS	**GPIO group G bit 5 (default)** **Slave serial GPIO bus 2 parallel data load input** **SMBus 2 Alert pin**
D15	GPIOG6 SGPS2I0 SALT3	ID/O8 ID ID/O8	CMOS	**GPIO group G bit 6 (default)** **Slave serial GPIO bus 2 serial data input channel 0** **SMBus 3 Alert pin**
E14	GPIOG7 SGPS2I1 SALT4	ID/O8 ID ID/O8	CMOS	**GPIO group G bit 7 (default)** **Slave serial GPIO bus 2 serial data input channel 1** **SMBus 4 Alert pin**

VGA Interface – 4 pins

Ball	Signal	I/O	Type	Description
N5	GPIOJ4 VGAHS	ID/O8 O8	CMOS	**GPIO group J bit 4** **VGA horizontal sync output (default)**
R4	GPIOJ5 VGAVS	ID/O8 O8	CMOS	**GPIO group J bit 5** **VGA vertical sync output (default)**
R3	GPIOJ6 DDCCLK	ID/O8 ID/O8	CMOS	**GPIO group J bit 6** **VGA DDC clock pin (default)**
T3	GPIOJ7 DDCDAT	ID/O8 ID/O8	CMOS	**GPIO group J bit 7** **VGA DDC data pin (default)**

PWM/Fan Tachometer – 24 pins

Ball	Signal	I/O	Type	Description
V2	GPION0 PWM0	ID/O8 O8	CMOS	**GPIO group N bit 0 (default)** **PWM output**
W2	GPION1 PWM1	ID/O8 O8	CMOS	**GPIO group N bit 1 (default)** **PWM output**
V3	GPION2 PWM2 VPIG2	ID/O8 O8 ID	CMOS	**GPIO group N bit 2 (default)** **PWM output** **Digital video input port: G2**

to next page

from previous page

U3	GPION3	ID/O8	CMOS	**GPIO group N bit 3 (default)**
	PWM3	O8		**PWM output**
	VPIG3	ID		**Digital video input port: G3**
W3	GPION4	ID/O8	CMOS	**GPIO group N bit 4 (default)**
	PWM4	O8		**PWM output**
	VPIG4	ID		**Digital video input port: G4**
AA3	GPION5	ID/O8	CMOS	**GPIO group N bit 5 (default)**
	PWM5	O8		**PWM output**
	VPIG5	ID		**Digital video input port: G5**
Y3	GPION6	ID/O8	CMOS	**GPIO group N bit 6 (default)**
	PWM6	O8		**PWM output**
	VPIG6	ID		**Digital video input port: G6**
T4	GPION7	ID/O8	CMOS	**GPIO group N bit 7 (default)**
	PWM7	O8		**PWM output**
	VPIG7	ID		**Digital video input port: G7**
U5	GPIOO0	ID/O8	CMOS	**GPIO group O bit 0 (default)**
	TACH0	ID		**Fan Tachometer input**
	VPIG8	ID		**Digital video input port: G8**
U4	GPIOO1	ID/O8	CMOS	**GPIO group O bit 1 (default)**
	TACH1	ID		**Fan Tachometer input**
	VPIG9	ID		**Digital video input port: G9**
V5	GPIOO2	ID/O8	CMOS	**GPIO group O bit 2 (default)**
	TACH2	ID		**Fan Tachometer input**
AB4	GPIOO3	ID/O8	CMOS	**GPIO group O bit 3 (default)**
	TACH3	ID		**Fan Tachometer input**
AB3	GPIOO4	ID/O8	CMOS	**GPIO group O bit 4 (default)**
	TACH4	ID		**Fan Tachometer input**
	VPIR2	ID		**Digital video input port: R2**
Y4	GPIOO5	ID/O8	CMOS	**GPIO group O bit 5 (default)**
	TACH5	ID		**Fan Tachometer input**
	VPIR3	ID		**Digital video input port: R3**
AA4	GPIOO6	ID/O8	CMOS	**GPIO group O bit 6 (default)**
	TACH6	ID		**Fan Tachometer input**
	VPIR4	ID		**Digital video input port: R4**
W4	GPIOO7	ID/O8	CMOS	**GPIO group O bit 7 (default)**
	TACH7	ID		**Fan Tachometer input**
	VPIR5	ID		**Digital video input port: R5**
V4	GPIOP0	ID/O8	CMOS	**GPIO group P bit 0 (default)**
	TACH8	ID		**Fan Tachometer input**
	VPIR6	ID		**Digital video input port: R6**
W5	GPIOP1	ID/O8	CMOS	**GPIO group P bit 1 (default)**
	TACH9	ID		**Fan Tachometer input**
	VPIR7	ID		**Digital video input port: R7**

to next page

from previous page

AA5	GPIOP2 TACH10 VPIR8	ID/O8 ID ID	CMOS	**GPIO group P bit 2 (default)** **Fan Tachometer input** **Digital video input port: R8**
AB5	GPIOP3 TACH11 VPIR9	ID/O8 ID ID	CMOS	**GPIO group P bit 3 (default)** **Fan Tachometer input** **Digital video input port: R9**
Y6	GPIOP4 TACH12	ID/O8 ID	CMOS	**GPIO group P bit 4 (default)** **Fan Tachometer input**
Y5	GPIOP5 TACH13	ID/O8 ID	CMOS	**GPIO group P bit 5 (default)** **Fan Tachometer input**
W6	GPIOP6 TACH14	ID/O8 ID	CMOS	**GPIO group P bit 6 (default)** **Fan Tachometer input**
V6	GPIOP7 TACH15	ID/O8 ID	CMOS	**GPIO group P bit 7 (default)** **Fan Tachometer input**

GPIO Interface – 46 pins

Ball	Signal	I/O	Type	Description
IO Power Domain : PV33D				
B14	GPIOA0 MAC1LINK	ID/O16 ID	CMOS	**GPIO group A bit 0 (default)** **MAC1 PHY link status input** Programmable high/low polarity.
D14	GPIOA1 MAC2LINK	ID/O16 ID	CMOS	**GPIO group A bit 1 (default)** **MAC2 PHY link status input** Programmable high/low polarity.
D13	GPIOA2 TIMER3 SPI1CS1#	ID/O16 O16 O16	CMOS	**GPIO group A bit 2 (default)** **Timer 3 programmable timer pulse output** **SPI1 master interface CS output channel 1**
E13	GPIOA3 TIMER4	ID/O16 O16	CMOS	**GPIO group A bit 3 (default)** **Timer 4 programmable timer pulse output**
IO Power Domain : LPVDD				
K19	GPIOB0	ID/O8	CMOS	**GPIO group B bit 0**
L19	GPIOB1	ID/O8	CMOS	**GPIO group B bit 1**
L18	GPIOB2	ID/O8	CMOS	**GPIO group B bit 2**
K18	GPIOB3	ID/O8	CMOS	**GPIO group B bit 3**
J20	GPIOB4 USBCKI	ID/O8 ID	CMOS	**GPIO group B bit 4 (default)** **Optional USB reference clock input** The USB reference clock input is necessary only when USB function (host/slave) is used and CLKIN is fed by 25MHz. Enabled when hardware strap bit23 is set to 1. **When use USBCKI input, firmware should modify the USB driver loading sequence. Since USB function registers are not accessible if USBCKI is not active. So the USB driver can only be loaded when USBCKI is active.** **It is strongly not recommended to use CLKIN=25MHz mode.** Please refer to Application Note 1 for more information.

to next page

from previous page

H21	GPIOB5 LPCPD# LPCSMI#	ID/O8 ID/O8 ID/O8	CMOS	**GPIO group B bit 5 (default)** **Dedicated LPC power down control (Optional for LPC function)** **Dedicated LPC SMI control (Optional for LPC function)**
H22	GPIOB6 LPCPME#	ID/O8 ID/O8	CMOS	**GPIO group B bit 6 (default)** **Dedicated LPC PME# control (Optional for LPC function)**
H20	GPIOB7	ID/O8	CMOS	**GPIO group B bit 7**
IO Power Domain : PV33D				
B10	GPIOQ6 OSCCLK	IDS/O8 O8	CMOS	**GPIO group Q bit 6 (default)** **Internal oscillator ring clock output**
N20	GPIOQ7 PEWAKE#	IDS/O8 IDS/O8	CMOS	**GPIO group Q bit 7 (default)** **PCI Express Wake up function pin** PEWAKE# signal is an open drain type signal.
V20	GPIOS0 VPOB2 SPI2CS1#	ID/O8 O8 O8	CMOS	**GPIO group S bit 0 (default)** **Digital video output port: B2** **SPI2 master interface CS output channel 1**
U19	GPIOS1 VPOB3 BMCINT	ID/O8 O8 O8	CMOS	**GPIO group S bit 1 (default)** **Digital video output port: B3** **BMC IRQ# interrupt output** Open-drain output with high/low output polarity selection.
R18	GPIOS2 VPOB4 SALT5	ID/O8 O8 ID/O8	CMOS	**GPIO group S bit 2 (default)** **Digital video output port: B4** **SMBus 5 Alert pin**
P18	GPIOS3 VPOB5 SALT6	ID/O8 O8 ID/O8	CMOS	**GPIO group S bit 3 (default)** **Digital video output port: B5** **SMBus 6 Alert pin**
R19	GPIOS4 VPOB6	ID/O8 O8	CMOS	**GPIO group S bit 4 (default)** **Digital video output port: B6**
W20	GPIOS5 VPOB7	ID/O8 O8	CMOS	**GPIO group S bit 5 (default)** **Digital video output port: B7**
U20	GPIOS6 VPOB8	ID/O8 O8	CMOS	**GPIO group S bit 6 (default)** **Digital video output port: B8**
AA20	GPIOS7 VPOB9	ID/O8 O8	CMOS	**GPIO group S bit 7 (default)** **Digital video output port: B9**
R22	GPIOY0 SIOS3#	ID/O8 ID	CMOS	**GPIO group Y bit 0 (default)** **Super IO S3# control (Optional for SIO function)**
R21	GPIOY1 SIOS5#	ID/O8 ID	CMOS	**GPIO group Y bit 1 (default)** **Super IO S5# control (Optional for SIO function)**
P22	GPIOY2 SIOPWREQ#	ID/O8 O8	CMOS	**GPIO group Y bit 2 (default)** **Super IO PWREQ# control (Optional for SIO function)**
P21	GPIOY3 SIOONCTRL#	ID/O8 ID/O8	CMOS	**GPIO group Y bit 3 (default)** **Super IO ONCONTROL# control (Optional for SIO function)**
Y20	GPIOZ0 VPOG2	ID/O8 O8	CMOS	**GPIO group Z bit 0 (default)** **Digital video output port: G2**

to next page

from previous page

	NORA0 SIOPBI#	O8 ID		**Parallel NOR interface : A0** **Super IO PWRBUTTON_I# control (Optional for SIO function)**
AB20	GPIOZ1 VPOG3 NORA1 SIOPWRGD	ID/O8 O8 O8 ID	CMOS	**GPIO group Z bit 1 (default)** **Digital video output port: G3** **Parallel NOR interface : A1** **System power good input (Optional for SIO function)**
AB21	GPIOZ2 VPOG4 NORA2 SIOPBO#	ID/O8 O8 O8 ID/O8	CMOS	**GPIO group Z bit 2 (default)** **Digital video output port: G4** **Parallel NOR interface : A2** **Super IO PWRBUTTON_O# control (Optional for SIO function)**
AA21	GPIOZ3 VPOG5 NORA3 SIOSCI#	ID/O8 O8 O8 O8	CMOS	**GPIO group Z bit 3 (default)** **Digital video output port: G5** **Parallel NOR interface : A3** **System SCI output (Optional for SIO function)**
U21	GPIOZ4 VPOG6 NORA4	ID/O8 O8 O8	CMOS	**GPIO group Z bit 4 (default)** **Digital video output port: G6** **Parallel NOR interface : A4**
W22	GPIOZ5 VPOG7 NORA5	ID/O8 O8 O8	CMOS	**GPIO group Z bit 5 (default)** **Digital video output port: G7** **Parallel NOR interface : A5**
V22	GPIOZ6 VPOG8 NORA6	ID/O8 O8 O8	CMOS	**GPIO group Z bit 6 (default)** **Digital video output port: G8** **Parallel NOR interface : A6**
W21	GPIOZ7 VPOG9 NORA7	ID/O8 O8 O8	CMOS	**GPIO group Z bit 7 (default)** **Digital video output port: G9** **Parallel NOR interface : A7**
Y21	GPIOAA0 VPOR2 NORD0 SALT7	ID/O8 O8 ID/O8 ID/O8	CMOS	**GPIO group AA bit 0 (default)** **Digital video output port: R2** **Parallel NOR interface : D0** **SMBus 7 Alert pin**
V21	GPIOAA1 VPOR3 NORD1 SALT8	ID/O8 O8 ID/O8 ID/O8	CMOS	**GPIO group AA bit 1 (default)** **Digital video output port: R3** **Parallel NOR interface : D1** **SMBus 8 Alert pin**
Y22	GPIOAA2 VPOR4 NORD2 SALT9	ID/O8 O8 ID/O8 ID/O8	CMOS	**GPIO group AA bit 2 (default)** **Digital video output port: R4** **Parallel NOR interface : D2** **SMBus 9 Alert pin**
AA22	GPIOAA3 VPOR5 NORD3	ID/O8 O8 ID/O8	CMOS	**GPIO group AA bit 3 (default)** **Digital video output port: R5** **Parallel NOR interface : D3**

to next page

from previous page

	SALT10	ID/O8		**SMBus 10 Alert pin**
U22	GPIOAA4	ID/O8	CMOS	**GPIO group AA bit 4 (default)**
	VPOR6	O8		**Digital video output port: R6**
	NORD4	ID/O8		**Parallel NOR interface : D4**
	SALT11	ID/O8		**SMBus 11 Alert pin**
T20	GPIOAA5	ID/O8	CMOS	**GPIO group AA bit 5 (default)**
	VPOR7	O8		**Digital video output port: R7**
	NORD5	ID/O8		**Parallel NOR interface : D5**
	SALT12	ID/O8		**SMBus 12 Alert pin**
N18	GPIOAA6	ID/O8	CMOS	**GPIO group AA bit 6 (default)**
	VPOR8	O8		**Digital video output port: R8**
	NORD6	ID/O8		**Parallel NOR interface : D6**
	SALT13	ID/O8		**SMBus 13 Alert pin**
P19	GPIOAA7	ID/O8	CMOS	**GPIO group AA bit 7 (default)**
	VPOR9	O8		**Digital video output port: R9**
	NORD7	ID/O8		**Parallel NOR interface : D7**
	SALT14	ID/O8		**SMBus 14 Alert pin**
N19	GPIOAB0	ID/O8	CMOS	**GPIO group AB bit 0 (default)**
	VPODE	O8		**Digital video output port: DE**
	NOROE#	ID/O8		**Parallel NOR interface : OE#**
T21	GPIOAB1	ID/O8	CMOS	**GPIO group AB bit 1 (default)**
	VPOHS	O8		**Digital video output port: H-SYNC**
	NORWE#	ID/O8		**Parallel NOR interface : WE#**
T22	GPIOAB2	ID/O8	CMOS	**GPIO group AB bit 2 (default)**
	VPOVS	O8		**Digital video output port: V-SYNC**
	WDTRST1	O8		**Watchdog timer 1 timeout pulse output** This is the Watchdog1 pulse mode output. It is programmable of push-pull or open-drain driving types, and active high or low output.
R20	GPIOAB3	ID/O8	CMOS	**GPIO group AB bit 3 (default)**
	VPOCLK	O8		**Digital video output port: CLK**
	WDTRST2	O8		**Watchdog timer 2 timeout pulse output** This is the Watchdog2 pulse mode output. It is programmable of push-pull or open-drain driving types, and active high or low output.

PECI Port – 2 pins

Ball	Signal	I/O	Type	Description
AB18	PECI	I/O	A	**PECI signal input/output**
AA17	PECIVDD	-	P	**PECI power** Input power range 0.85V-1.21V, connect to CPU V_{TT} power. If not used, connect to GND.

JTAG Port – 6 pins

Ball	Signal	I/O	Type	Description
E11	NTRST	ID/O8 O8	CMOS	**ARM JTAG test reset input** **JTAG master reset output** No external pull high when normal operation.
C10	TCK	IU/O8 O8	CMOS	**ARM JTAG test clock input** **JTAG master clock output**
C8	TMS	IU/O8 O8	CMOS	**ARM JTAG test mode select input** **JTAG master mode select output**
D12	TDI	IU/O8 O8	CMOS	**ARM JTAG test data input** **JTAG master data output**
D11	TDO	IU/O8 IU	CMOS	**ARM JTAG test data output** **JTAG master data input**
C9	RTCK	ID/O8	CMOS	**ARM JTAG clock output**

Miscellaneous – 5 pins

Ball	Signal	I/O	Type	Description
K3	ENTEST	IDS	CMOS	**Enable test mode input pin** Pull low the ENTEST to ground (10KΩ) in normal operation mode.
U18	SRST#	IDS	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.
V18	EXTRST#	IDS	CMOS	**SOC level reset input pin** SOC level reset with programmable reset control at register SCU9C. Use this pin to reset firmware at normal working mode. This pin has the same capability as internal watchdog reset.
W18	CLKIN	ID	CMOS	**External reference clock input pin** Connect this pin to a 24MHz or 25MHz(**no recommend**) oscillator source. When connect to 25MHz, hardware strap bit23 must be set to 1; and GPIOB4 must be fed with another 24/48MHz for USB function. If select CLKIN=25MHz, please refer to Application Note 1 for the firmware coding guide.

to next page

from previous page

Y18	HBLED#	O8	CMOS	**Heart Beat LED indicator output.** Open-drain active low output. Support hardware or software blink modes. The hardware mode includes below blinking modes: 1 10Hz : ARM Running on Flash (instruction fetch from flash) 2 2Hz : ARM Running on DRAM without interrupt enabled (instruction fetch from DRAM) 3 0.5Hz : ARM Running on DRAM with interrupt monitor enabled (normal operation mode) 4 0.1Hz : Abnormal mode, some interrupts are not serviced for over 2 seconds 5 0Hz : Always dark indicates firmware is not running or dead The software mode is the same as GPIO control, refer to VIC60. This pin can be used to replace the traditional software heart beat function.

USB 2.0 Host/Slave Port – 7 pins

Ball	Signal	I/O	Type	Description
A7	USB2A_DP	I/O	A	$D+$ **signal of USB 2.0 port A**
A8	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
B6	USB2B_DP	I/O	A	$D+$ **signal of USB 2.0 port B**
A6	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
B8 B7	USB2AVRES USB2BVRES	- -	A	**USB2.0 reference resistor** Connect an external resistor of 8.2KΩ to GND for adjusting the magnitude of output current.
C7	USB2AV33	-	P	**3.3V USB analog power** 4.7uF + 0.1uF decouple capacitor required.

DAC – 6 pins

Ball	Signal	I/O	Type	Description
J4	DACR	O	A	**DAC R channel output**
J3	DACG	O	A	**DAC G channel output**
J2	DACB	O	A	**DAC B channel output**
K4	DACRSET	-	A	**DAC reference resistor** Connect an external resistor of 18KΩ to GND for adjusting the magnitude of DAC full-scale output current. This resistor can be fine-tuned in the range 7.5K $\sim$ 20K to increase/decrease the output current. The formula as below: $I_{out} = 250/DACRSET$

to next page

from previous page

K5	DACDV33	-	P	**3.3V DAC digital power**
K6	DACAV33	-	P	**3.3V DAC analog power**

ADC – 20 pins

Ball	Signal	I/O	Type	Description
F4	ADC0 GPIW0	I IR	A CMOS	**ADC channel 0 analog input** **GPIO group W bit 0**
F5	ADC1 GPIW1	I IR	A CMOS	**ADC channel 1 analog input** **GPIO group W bit 1**
E2	ADC2 GPIW2	I IR	A CMOS	**ADC channel 2 analog input** **GPIO group W bit 2**
E1	ADC3 GPIW3	I IR	A CMOS	**ADC channel 3 analog input** **GPIO group W bit 3**
F3	ADC4 GPIW4	I IR	A CMOS	**ADC channel 4 analog input** **GPIO group W bit 4**
E3	ADC5 GPIW5	I IR	A CMOS	**ADC channel 5 analog input** **GPIO group W bit 5**
G5	ADC6 GPIW6	I IR	A CMOS	**ADC channel 6 analog input** **GPIO group W bit 6**
G4	ADC7 GPIW7	I IR	A CMOS	**ADC channel 7 analog input** **GPIO group W bit 7**
F2	ADC8 GPIX0	I IR	A CMOS	**ADC channel 8 analog input** **GPIO group X bit 0**
G3	ADC9 GPIX1	I IR	A CMOS	**ADC channel 9 analog input** **GPIO group X bit 1**
G2	ADC10 GPIX2	I IR	A CMOS	**ADC channel 10 analog input** **GPIO group X bit 2**
F1	ADC11 GPIX3	I IR	A CMOS	**ADC channel 11 analog input** **GPIO group X bit 3**
H5	ADC12 GPIX4	I IR	A CMOS	**ADC channel 12 analog input** **GPIO group X bit 4**
G1	ADC13 GPIX5	I IR	A CMOS	**ADC channel 13 analog input** **GPIO group X bit 5**
H3	ADC14 GPIX6	I IR	A CMOS	**ADC channel 14 analog input** **GPIO group X bit 6**
H4	ADC15 GPIX7	I IR	A CMOS	**ADC channel 15 analog input** **GPIO group X bit 7**
H2	ADCVREFP	O	A	**ADC positive reference voltage**
H1	ADCVREFN	O	A	**ADC negative reference voltage**
J1	ADCREXT	O	A	**ADC external resistor connection pin. (Optional)** Connect resistor 50KΩ to ground for sensor source current adjustment.
J6	ADCAV33	-	P	**3.3V ADC analog power**

PLL Power – 8 pins				
Ball	**Signal**	**I/O**	**Type**	**Description**
V17	PLLDV11	-	P	**1.15V MPLL/HPLL digital power**
Y17	PLLAV33	-	P	**3.3V MPLL/HPLL analog power**
W17	PLLAV33	-	P	**3.3V MPLL/HPLL analog power**
T11	MPLLAV33	-	P	**3.3V DDRPHY PLL power**
L5	VPLLAV11	-	P	**1.15V VPLL analog power**
L6	VPLLAV11	-	P	**1.15V VPLL analog power**
J7	VPLLAV33	-	P	**3.3V VPLL analog power**
H7	VPLLAV33	-	P	**3.3V VPLL analog power**

Power and Ground – 122 pins				
Ball	**Signal**	**I/O**	**Type**	**Description**
F6	IV11D	-	P	**1.15V Core logic power**
G10	IV11D	-	P	
G11	IV11D	-	P	
G12	IV11D	-	P	
G13	IV11D	-	P	
G14	IV11D	-	P	
G15	IV11D	-	P	
G7	IV11D	-	P	
G8	IV11D	-	P	
G9	IV11D	-	P	
H16	IV11D	-	P	
J16	IV11D	-	P	
L7	IV11D	-	P	
N16	IV11D	-	P	
N7	IV11D	-	P	
P10	IV11D	-	P	
P11	IV11D	-	P	
P12	IV11D	-	P	
P13	IV11D	-	P	
P14	IV11D	-	P	
P16	IV11D	-	P	
P7	IV11D	-	P	
P9	IV11D	-	P	
R16	IV11D	-	P	
R7	IV11D	-	P	
L17	PEAV11	-	P	**1.15V PCIe PHY power**
T10	MVDD	-	P	**1.35V DDR3L SDRAM memory I/O power**
T12	MVDD	-	P	**1.20V DDR4 SDRAM memory I/O power**
T13	MVDD	-	P	
T14	MVDD	-	P	
T8	MVDD	-	P	

to next page

from previous page

F12	PV33D	-	P	**3.3V General purpose I/O power**
F13	PV33D	-	P	
F14	PV33D	-	P	
F15	PV33D	-	P	
G16	PV33D	-	P	
H17	PV33D	-	P	
K7	PV33D	-	P	
N17	PV33D	-	P	
N6	PV33D	-	P	
P17	PV33D	-	P	
P6	PV33D	-	P	
R17	PV33D	-	P	
R6	PV33D	-	P	
T15	PV33D	-	P	
J17	LPVDD	-	P	**1.8V/2.5V/3.3V Adjustable I/O power**
K17	LPVDD	-	P	**Power for GPIOB and GPIOAC**
L16	PEAV33	-	P	**3.3V PCIe PHY power**
F10	R1VDD	-	P	**3.3V RMII/NCSI #1 interface I/O power**
F11	R1VDD	-	P	**3.3V/2.5V RGMII #1 interface I/O power**
F7	R2VDD	-	P	**3.3V RMII/NCSI #2 interface I/O power**
F8	R2VDD	-	P	**3.3V/2.5V RGMII #2 interface I/O power**
E4	BATVDD	-	P	**Battery backed SRAM power, not including RTC**
A1	GND	-	P	**Digital ground**
A22	GND	-	P	
AA11	GND	-	P	
AA13	GND	-	P	
AA15	GND	-	P	
AA7	GND	-	P	
AA9	GND	-	P	
AB1	GND	-	P	
AB22	GND	-	P	
C6	GND	-	P	
E8	GND	-	P	
F16	GND	-	P	
G19	GND	-	P	
G6	GND	-	P	
H6	GND	-	P	
J10	GND	-	P	
J11	GND	-	P	
J12	GND	-	P	
J13	GND	-	P	
J14	GND	-	P	
J21	GND	-	P	

to next page

from previous page

J22	GND	-	P	
J5	GND	-	P	
J9	GND	-	P	
K10	GND	-	P	
K11	GND	-	P	
K12	GND	-	P	
K13	GND	-	P	
K14	GND	-	P	
K16	GND	-	P	
K9	GND	-	P	
L10	GND	-	P	
L11	GND	-	P	
L12	GND	-	P	
L13	GND	-	P	
L14	GND	-	P	
L9	GND	-	P	
M1	GND	-	P	
M10	GND	-	P	
M11	GND	-	P	
M12	GND	-	P	
M13	GND	-	P	
M14	GND	-	P	
M16	GND	-	P	
M17	GND	-	P	
M2	GND	-	P	
M3	GND	-	P	
M4	GND	-	P	
M5	GND	-	P	
M6	GND	-	P	
M7	GND	-	P	
M9	GND	-	P	
N10	GND	-	P	
N11	GND	-	P	
N12	GND	-	P	
N13	GND	-	P	
N14	GND	-	P	
N9	GND	-	P	
T16	GND	-	P	
T6	GND	-	P	
T7	GND	-	P	
U11	GND	-	P	
U6	GND	-	P	
V10	GND	-	P	

to next page

from previous page

V12	GND	-	P	
V14	GND	-	P	
V16	GND	-	P	
V8	GND	-	P	
E5	GND	-	P	

2.1.1 Not Available Pins for AST2520

Pink colored pin function is not available on AST2520.

Ball	Pin Name	Note
K6	DACAV33	Short to PV33D power rail
K5	DACDV33	Short to PV33D power rail
J2	DACB	NC
J3	DACG	NC
J4	DACR	NC
K20	PEREXT	NC
L20	PERST#	Short to GND or Pull low
K22	PEREFCLKN	Short to GND or Pull low
K21	PEREFCLKP	Short to GND or Pull low
M22	PERXN	Short to GND or Pull low
M21	PERXP	Short to GND or Pull low
L22	PETXN	NC
L21	PETXP	NC
N5	GPIOJ4_VGAHS	
R4	GPIOJ5_VGAVS	
R3	GPIOJ6_DDCCLK	
T3	GPIOJ7_DDCDAT	
T1	GPIOL1_NDCD1_VPIDE	
U2	GPIOL3_NRI1_VPIHS	
P4	GPIOL4_NDTR1_VPIVS	
P3	GPIOL5_NRTS1_VPICLK	
Y1	GPIOM0_NCTS2_VPIB2	
AB2	GPIOM1_NDCD2_VPIB3	
AA1	GPIOM2_NDSR2_VPIB4	
Y2	GPIOM3_NRI2_VPIB5	
AA2	GPIOM4_NDTR2_VPIB6	
P5	GPIOM5_NRTS2_VPIB7	
R5	GPIOM6_TXD2_VPIB8	
T5	GPIOM7_RXD2_VPIB9	
V3	GPION2_PWM2_VPIG2	
U3	GPION3_PWM3_VPIG3	
W3	GPION4_PWM4_VPIG4	
AA3	GPION5_PWM5_VPIG5	

Ball	Pin Name	Note
Y3	GPION6_PWM6_VPIG6	
T4	GPION7_PWM7_VPIG7	
U5	GPIOO0_TACH0_VPIG8	
U4	GPIOO1_TACH1_VPIG9	
AB3	GPIOO4_TACH4_VPIR2	
Y4	GPIOO5_TACH5_VPIR3	
AA4	GPIOO6_TACH6_VPIR4	
W4	GPIOO7_TACH7_VPIR5	
V4	GPIOP0_TACH8_VPIR6	
W5	GPIOP1_TACH9_VPIR7	
AA5	GPIOP2_TACH10_VPIR8	
AB5	GPIOP3_TACH11_VPIR9	
V20	GPIOS0_VPOB2_SPI2CS1#	
U19	GPIOS1_VPOB3_BMCINT	
R18	GPIOS2_VPOB4_SALT5	
P18	GPIOS3_VPOB5_SALT6	
R19	GPIOS4_VPOB6	
W20	GPIOS5_VPOB7	
U20	GPIOS6_VPOB8	
AA20	GPIOS7_VPOB9	
Y20	GPIOZ0_VPOG2_NORA0_SIOPBI#	
AB20	GPIOZ1_VPOG3_NORA1_SIOPWRGD	
AB21	GPIOZ2_VPOG4_NORA2_SIOPBO#	
AA21	GPIOZ3_VPOG5_NORA3_SIOSCI#	
U21	GPIOZ4_VPOG6_NORA4	
W22	GPIOZ5_VPOG7_NORA5	
V22	GPIOZ6_VPOG8_NORA6	
W21	GPIOZ7_VPOG9_NORA7	
Y21	GPIOAA0_VPOR2_NORD0_SALT7	
V21	GPIOAA1_VPOR3_NORD1_SALT8	
Y22	GPIOAA2_VPOR4_NORD2_SALT9	
AA22	GPIOAA3_VPOR5_NORD3_SALT10	
U22	GPIOAA4_VPOR6_NORD4_SALT11	
T20	GPIOAA5_VPOR7_NORD5_SALT12	
N18	GPIOAA6_VPOR8_NORD6_SALT13	
P19	GPIOAA7_VPOR9_NORD7_SALT14	
N19	GPIOAB0_VPODE_NOROE#	
T21	GPIOAB1_VPOHS_NORWE#	
T22	GPIOAB2_VPOVS_WDTRST1	
R20	GPIOAB3_VPOCLK_WDTRST2	

2.2 Ball Map

	1	2	3	4	5	6	7	8	9	10	11
A	GND	RGMII2TXD0/ RMII2TXD0/ GPIOU0	RGMII1RXD0/ RMII1RXD0/ GPIOU6	RGMII1RXCTL/ GPIOU5	RGMII1TXD1/ RMII1TXD1/ GPIOT3	USB2B_DN	USB2A_DP	USB2A_DN	GPIOQ2/ SCL4	GPIOQ1/ SDA3	GPIOQ0/ SCL3
B	RGMII2TXCTL/ RMII2TXEN/ GPIOT7	RGMII2TXCK/ RMII2RCLKO/ GPIOT6	RGMII2TXD1/ RMII2TXD1/ GPIOU1	RGMII1RXCK / RMII1RCLKI/ GPIOU4	RGMII1TXCK/ RMII1RCLKO/ GPIOT0	USB2B_DP	USB2BVRES	USB2AVRES	GPIOQ3/ SDA4	GPIOQ6/ OSCCLK	GPIOC7/ SD1WP#/ SDA13
C	RGMII2RXCTL/ GPIOV3	RGMII2RXCK / RMII2RCLKI/ GPIOV2	RGMII2RXD0/ RMII2RXD0/ GPIOV4	RGMII1RXD3/ RMII1RXER/ GPIOV1	RGMII1RXD2/ RMII1CRSDV/ GPIOV0	GND	USB2AV33	TMS	RTCK	TCK	GPIOC6/ SD1CD#/ SCL13
D	RGMII2RXD1/ RMII2RXD1/ GPIOV5	RGMII2RXD2/ RMII2CRSDV/ GPIOV6	RGMIICK	RGMII2TXD3/ GPIOU3	RGMII2TXD2/ GPIOU2	RGMII1RXD1/ RMII1RXD1/ GPIOU7	RGMII1TXD3/ GPIOT5	GPIOR6/ MDC1	GPIOC3/ SD1DAT1/ SDA11	GPIOC4/ SD1DAT2/ SCL12	TDO
E	ADC3/ GPIW3	ADC2/ GPIW2	ADC5/ GPIW5	BATVDD	GND	RGMII2RXD3/ RMII2RXER/ GPIOV7	RGMII1TXD2/ GPIOT4	GND	RGMII1TXCTL/ RMII1TXEN/ GPIOT1	GPIOR7/ MDIO1	NTRST
F	ADC11/ GPIX3	ADC8/ GPIX0	ADC4/ GPIW4	ADC0/ GPIW0	ADC1/ GPIW1	IV11D	R2VDD	R2VDD	RGMII1TXD0/ RMII1TXD0/ GPIOT2	R1VDD	R1VDD
G	ADC13/ GPIX5	ADC10/ GPIX2	ADC9/ GPIX1	ADC7/ GPIW7	ADC6/ GPIW6	GND	IV11D	IV11D	IV11D	IV11D	IV11D
H	ADCVREFN	ADCVREFP	ADC14/ GPIX6	ADC15/ GPIX7	ADC12/ GPIX4	GND	VPLLAV33				
J	ADCREXT	DACB	DACG	DACR	GND	ADCAV33	VPLLAV33		GND	GND	GND
K	TXD5	RXD5	ENTEST	DACRSET	DACDV33	DACAV33	PV33D		GND	GND	GND
L	GPIOK2/ SCL6	GPIOJ1/ SGPMLD	GPIOK0/ SCL5	GPIOK1/ SDA5	VPLLAV11	VPLLAV11	IV11D		GND	GND	GND
M	GND	GND	GND	GND	GND	GND	GND		GND	GND	GND
N	GPIOK4/ SCL7	GPIOK3/ SDA6	GPIOJ2/ SGPMO	GPIOJ3/ SGPMI	GPIOJ4/ VGAHS	PV33D	IV11D		GND	GND	GND
P	GPIOK5/ SDA7	GPIOK6/ SCL8	GPIOL5/ NRTS1/ VPICLK	GPIOL4/ NDTR1/ VPIVS	GPIOM5/ NRTS2/ VPIB7	PV33D	IV11D		IV11D	IV11D	IV11D
R	GPIOK7/ SDA8	GPIOJ0/ SGPMCK	GPIOJ6/ DDCCLK	GPIOJ5/ VGAVS	GPIOM6/ TXD2/ VPIB8	PV33D	IV11D				
T	GPIOL1/ NDCD1/ VPIDE	GPIOL0/ NCTS1	GPIOJ7/ DDCDAT	GPION7/ PWM7/ VPIG7	GPIOM7/ RXD2/ VPIB9	GND	GND	MVDD	MVREF	MVDD	MPLLAV33
U	GPIOL2/ NDSR1	GPIOL3/ NRI1/ VPIHS	GPION3/ PWM3/ VPIG3	GPIOO1/ TACH1/ VPIG9	GPIOO0/ TACH0/ VPIG8	GND	MDQ15	MDM0	MDQ4	MDQ0	GND
V	GPIOL6/ TXD1	GPION0/ PWM0	GPION2/ PWM2/ VPIG2	GPIOP0/ TACH8/ VPIR6	GPIOO2/ TACH2	GPIOP7/ TACH15	MDQ14	GND	MDQ7	GND	MODT
W	GPIOL7/ RXD1	GPION1/ PWM1	GPION4/ PWM4/ VPIG4	GPIOO7/ TACH7/ VPIR5	GPIOP1/ TACH9/ VPIR7	GPIOP6/ TACH14	MDQ13	MDQ11	MDQ6	MDQ1	MIOZ
Y	GPIOM0/ NCTS2/ VPIB2	GPIOM3/ NRI2/ VPIB5	GPION6/ PWM6/ VPIG6	GPIOO5/ TACH5/ VPIR3	GPIOP5/ TACH13	GPIOP4/ TACH12	MDQ10	MDQ9	MDQ5	MDQ2	MCKE
AA	GPIOM2/ NDSR2/ VPIB4	GPIOM4/ NDTR2/ VPIB6	GPION5/ PWM5/ VPIG5	GPIOO6/ TACH6/ VPIR4	GPIOP2/ TACH10/ VPIR8	MDQ12	GND	MDQ8	GND	MDQ3	GND
AB	GND	GPIOM1/ NDCD2/ VPIB3	GPIOO4/ TACH4/ VPIR2	GPIOO3/ TACH3	GPIOP3/ TACH11/ VPIR9	MDQS1#	MDQS1	MDM1	MDQS0	MDQS0#	MCK
	1	2	3	4	5	6	7	8	9	10	11

Figure 9: Ball Map – Left Side

12	13	14	15	16	17	18	19	20	21	22	
GPIOC1/ SD1CMD/ SDA10	GPIOA5/ TIMER6/ SDA9	GPIOI6/ SPI1MOSI/ VBMOSI	GPIOI7/ SPI1MISO/ VBMISO	GPIOH6/ TXD6	GPIOH4/ NDTR6	GPIOH0/ NCTS6	GPIOG0/ SGPS1CK	GPIOE6/ TXD3	GPIOF4/ NDTR4/ LHCLK	GND	A
GPIOC2/ SD1DAT0/ SCL11	GPIOA7/ TIMER8/ MDIO2	GPIOA0/ MAC1LINK	GPIOI4/ SPI1CS0#/ VBCS#	GPIOI2/ SYSMOSI	GPIOH5/ NRTS6	GPIOH1/ NDCD6	GPIOE7/ RXD3	GPIOE0/ NCTS3	GPIOF3/ NRI4/ LHAD3	GPIOF2/ NDSR4/ LHAD2	B
GPIOC0/ SD1CLK/ SCL10	GPIOA6/ TIMER7/ MDC2	GPIOA4/ TIMER5/ SCL9	GPIOI5/ SPI1CK/ VBCK	GPIOI3/ SYSMISO	GPIOH3/ NRI6	GPIOI0/ SYSCS#	GPIOG2/ SGPS1I0	GPIOE1/ NDCD3	GPIOD7/ SD2WP#	GPIOAC4/ ESPICK/ LCLK	C
TDI	GPIOA2/ TIMER3/ SPI1CS1#	GPIOA1/ MAC2LINK	GPIOG6/ SGPS2I0/ SALT3	GPIOG5/ SGPS2LD/ SALT2	GPIOH2/ NDSR6	GPIOH7/ RXD6	GPIOE5/ NRTS3	GPIOD3/ SD2DAT1	GPIOD4/ SD2DAT2	GPIOAC2/ ESPID2/ LAD2	D
GPIOC5/ SD1DAT3/ SDA12	GPIOA3/ TIMER4	GPIOG7/ SGPS2I1/ SALT4	GPIOI1/ SYSCK	GPIOG3/ SGPS1I1	GPIOG4/ SGPS2CK/ SALT1	GPIOE4/ NDTR3	GPIOG1/ SGPS1LD	GPIOD5/ SD2DAT3	GPIOD1/ SD2CMD	GPIOAC3/ ESPID3/ LAD3	E
PV33D	PV33D	PV33D	PV33D	GND	GPIOE3/ NRI3	GPIOE2/ NDSR3	GPIOD0/ SD2CLK	GPIOD2/ SD2DAT0	GPIOAC5/ ESPICS#/ LFRAME#	GPIOAC6/ ESPIALT#/ LSIRQ#	F
IV11D	IV11D	IV11D	IV11D	PV33D	GPIOF6/ TXD4/ LHSIRQ#	GPIOD6/ SD2CD#	GND	GPIOAC1/ ESPID1/ LAD1	GPIOAC0/ ESPID0/ LAD0	GPIOAC7/ ESPIRST#/ LPCRST#	G
				IV11D	PV33D	GPIOF7/ RXD4/ LHRST#	GPIOF5/ NRTS4/ LHFRAME#	GPIOB7	GPIOB5/ LPCPD#/ LPCSMI#	GPIOB6/ LPCPME#	H
GND	GND	GND		IV11D	LPVDD	GPIOF1/ NDCD4/ LHAD1	GPIOF0/ NCTS4/ LHAD0	GPIOB4/ USBCKI	GND	GND	J
GND	GND	GND		GND	LPVDD	GPIOB3	GPIOB0	PEREXT	PEREFCLKP	PEREFCLKN	K
GND	GND	GND		PEAV33	PEAV11	GPIOB2	GPIOB1	PERST#	PETXP	PETXN	L
GND	GND	GND		GND	GND	GPIOY4/ SCL1	GPIOY5/ SDA1	GPIOY6/ SCL2	PERXP	PERXN	M
GND	GND	GND		IV11D	PV33D	GPIOAA6/ VPOR8/ NORD6/ SALT13	GPIOAB0/ VPODE/ NOROE#	GPIOQ7/ PEWAKE#	GPIOQ4/ SCL14	GPIOQ5/ SDA14	N
IV11D	IV11D	IV11D		IV11D	PV33D	GPIOS3/ VPOB5/ SALT6	GPIOAA7/ VPOR9/ NORD7/ SALT14	GPIOY7/ SDA2	GPIOY3/ SIOONCTRL#	GPIOY2/ SIOPWREQ#	P
				IV11D	PV33D	GPIOS2/ VPOB4/ SALT5	GPIOS4/ VPOB6	GPIOAB3/ VPOCLK/ WDTRST2	GPIOY1/ SIOS5#	GPIOY0/ SIOS3#	R
MVDD	MVDD	MVDD	PV33D	GND	GPIOR2/ SPI2CS0#	FWSPIMISO	GPIOR1/ FWSPICS2#	GPIOAA5/ VPOR7/ NORD5/ SALT12	GPIOAB1/ VPOHS/ NORWE#	GPIOAB2/ VPOVS/ WDTRST1	T
MBA0	MA10	MA3	MA15	MALERT#	FWSPIMOSI	SRST#	GPIOS1/ VPOB3/ BMCINT	GPIOS6/ VPOB8	GPIOZ4/ VPOG6/ NORA4	GPIOAA4/ VPOR6/ NORD4/ SALT11	U
GND	MA0	GND	MA5	GND	PLLDV11	EXTRST#	GPIOR5/ SPI2MISO	GPIOS0/ VPOB2/ SPI2CS1#	GPIOAA1/ VPOR3/ NORD1/ SALT8	GPIOZ6/ VPOG8/ NORA6	V
MRAS#	MBA2/MBG0	MA2	MA4	MA8	PLLAV33	CLKIN	GPIOR4/ SPI2MOSI	GPIOS5/ VPOB7	GPIOZ7/ VPOG9/ NORA7	GPIOZ5/ VPOG7/ NORA5	W
MWE#	MBA1	MA12	MA14/MACT#	MA9	PLLAV33	HBLED#	GPIOR3/ SPI2CK	GPIOZ0/ VPOG2/ NORA0/ SIOPBI#	GPIOAA0/ VPOR2/ NORD0/ SALT7	GPIOAA2/ VPOR4/ NORD2/ SALT9	Y
MCS#	GND	MA11	GND	MA7	PECIVDD	FWSPICK	GPIOR0/ FWSPICS1#	GPIOS7/ VPOB9	GPIOZ3/ VPOG5/ NORA3/ SIOSCI#	GPIOAA3/ VPOR5/ NORD3/ SALT10	AA
MCK#	MCAS#	MA1	MA13	MA6	MRESET#	PECI	FWSPICS0#	GPIOZ1/ VPOG3/ NORA1/ SIOPWRGD	GPIOZ2/ VPOG4/ NORA2/ SIOPBO#	GND	AB
12	13	14	15	16	17	18	19	20	21	22	

Figure 10: Ball Map – Right Side

2.3 Hardware/Software Strap Definition

AST2500 defines a 3 steps procedure for functional strap setting input.

1. Step 1: Load from dedicated strap pins.
   - Based on the setting at step 1 to load step 2 or step 3 strap inputs.
2. Step 2: Hardware Strap: Load from the optional strap pins when SRST# active. (Must to be Enabled)
3. Step 3: Software Strap: Load from the Firmware SPI flash when SRST# inactive.
   - The step 2 and step 3 strap input modes can be enabled simutaneously. But software strap executes later and has higher priority to overwrite hardware strap.

The hardware strap registers (denotes HWTrap later) load the external pin voltage state when SRST# reset input is low. And keep the value unchanged after SRST# goes high. The strap register is defined at address 0x1E6E2070 in the System Control Unit (SCU) module.

When GPIO is used as hardware strap, it will constrain 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 Step 1: Load from dedicated strap pins

There are 16 dedicated strap function pins. Listed as below:

FWSPICS0#	0x1E6E2070 bit[0]	Enable CPU running
FWSPICK	0x1E6E2070 bit[30]	Enable GPIO Strap Mode (Step 2)
FWSPIMOSI	0x1E6E2070 bit[31]	Enable SPI Flash Strap Auto Fetch Mode (Step 3)
FWSPIMISO	0x1E6E2070 bit[25]	Enable eSPI
RTCK	0x1E6E2070 bit[26]	Enable eSPI Flash Mode
TXD5	0x1E6E2070 bit[29]	Select UART Debug port
RGMII1TXCTL	0x1E6E2070 bit[6]	Define MAC#1 interface type
RGMII2TXCTL	0x1E6E2070 bit[7]	Define MAC#2 interface type
RGMII1TXD0	0x1E6E2070 bit[12]	SPI1 interface mode selection bit[0]
RGMII1TXD1	0x1E6E2070 bit[13]	SPI1 interface mode selection bit[1]
RGMII1TXD2	0x1E6E2070 bit[16]	SuperIO configuration address selection
RGMII1TXD3	0x1E6E2070 bit[19]	Enable ACPI function
RGMII2TXD0	0x1E6E2070 bit[21]	Enable GPIOD pass-through mode
RGMII2TXD1	0x1E6E2070 bit[22]	Enable GPIOE pass-through mode
RGMII2TXD2	0x1E6E2070 bit[23]	Select 25 MHz reference clock input mode
RGMII2TXD3	0x1E6E2070 bit[24]	Select DDR4 SDRAM

2.3.2 Step 2: Hardware Strap – Load from the optional strap pins

GPIOS4	0x1E6E2070 bit[3]	VGA memory size selection (16MB)
GPIOS5	0x1E6E2070 bit[4]	Reserved (0)
GPIOS6	0x1E6E2070 bit[5]	Enable VGA BIOS ROM (0)
GPIOS7	0x1E6E2070 bit[15]	VGA Class Code selection (1)
GPIOZ4	0x1E6E2070 bit[17]	Enable BMC 2nd boot watchdog timer (1)
GPIOZ5	0x1E6E2070 bit[18]	Select USBCKI input frequency (0)
GPIOZ6	0x1E6E2070 bit[20]	Disable LPC to decode SuperIO address (0)
GPIOZ7	0x1E6E2070 bit[27]	Enable fast reset mode for ARM ICE debugger (0)

2.3.3 Step 3: Firmware strap – Load from the Firmware SPI flash

AST2500 defines an auto-load command format for hardware to recognize the strap setting code area. Customers can edit the desired power-on preload setting commands and store in the specified SPI flash area. Hardware will search for the code before CPU can start to fetch the code.

The flash area valid for strap code storage is at the lowest and highest 8KB of the full SPI flash area at FWSPICS0#. The strap code can be stored anywhere within the 8KB area, but must follow 4-bytes address alignment. And all strap commands must be stored continuously and finalized with an ending code.

The maximum SPI flash size supported by software strap function is 128MB (1Gbits).

When use the SPI flash of size larger than 16MB (32MB or larger) and power up boot with 24bits address mode, the firmware strap code can only be placed at the beginning 8KB area.
This is because for such a flash type, it could be configured to be accessed by 24bits or 32bits address mode, depending on the firmware selection. If place the strap code at high 8KB area, then it should be placed at 2 positions, except the actual highest 8KB area, it still must duplicate another set and place at top 8KB of 24bits addressing (top of 16MB). Otherwise it would lose the strap code fetching if at wrong address mode.

Strap command format:

Start Code: 12 bytes

Byte[3:0] = 0x5453412A
Byte[7:4] = 0x74537748
Byte[11:8] = 0x23706172

Command Code: 8 bytes

Bit[7:0]	address bit[31:24] = 0x1E
Bit[15:8]	address bit[23:16]
Bit[23:16]	address bit[15:8]
Bit[31:26]	address bit[7:2]
Bit[25]	even parity of address bit[23:2]
Bit[24]	even parity of data bit[31:0]
Bit[63:32]	data bit[31:0]

Ending Code: 4 bytes

Byte[3:0] = 0x2A415354

The strap code search sequence as below:

1. Scan address 0xFFE000 ∼ 0xFFFFFF with 24 bits address mode
2. Scan address 0x0 ∼ 0x1FFF with 24 bits address mode
3. Scan address 0x7FFE000 ∼ 0x7FFFFFF with 32 bits address mode
4. Scan address 0x0 ∼ 0x1FFF with 32 bits address mode

2.3.4 Hardware strap configuration method

These hardware strap pins had embedded an internal pull resistor of value between 52K to 195K Ω.

- FWSPICS0#: embedded internal pull high, strap value = inverse of input
  - Strap value = 0, pull high the pin to 3.3V standby power by a resistor of value 2.2K Ω.
  - Strap value = 1, pull-down the pin to ground by a resistor of value 1K ∼ 4.7K Ω.
- TXD5: embedded internal pull high
  - Strap value = 0, pull-down the pin to ground by a resistor of value 1K ∼ 4.7K Ω.
  - Strap value = 1, just leave the pin floating or pull up.
- FWSPICK, FWSPIMOSI, FWSPIMISO, RTCK: embedded internal pull low
  - 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 ∼ 4.7K Ω.
- RGMII1TXCTL, RGMII2TXCTL, RGMII1TXD[3:0], RGMII2TXD[3:0]: embedded internal pull low
  - Strap value = 0, pull-down the pin to ground by a resistor of value 1K ∼ 4.7K Ω. No leave the pin floating, since PHY may have internal pull high on these pin.
  - Strap value = 1, pull-high the pin to **R1VDD/R2VDD standby power by a resistor of value 1K ∼ 4.7K Ω.**
- GPIOS[7:4], GPIOZ[7:4]: embedded internal pull low
  - 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 ∼ 4.7K Ω.

Offset: 0x1E6E2070 **Hardware Strap Table** **Init = X**

Bit	R/W	Description
31	RW	**Enable SPI Flash Strap Auto Fetch Mode** 0: Disable 1: Enable
30	RW	**Enable GPIO Strap Mode** 0: Disable 1: Enable
29	RW	**Select UART Debug Port** 0: Select UART1 as BMC console port 1: Select UART5 as BMC console port This bit is used to select the UART port for user debugging. It is not the BMC console function. The function is similar to the PCIe-to-AHB bridge, which is working as a hardware u-boot interface, and no firmware required to operate. This interface can support firmware update capability without firmware working required.

to next page

from previous page

28	RW	**Reserved (1)**
27	RW	**Enable fast reset mode for ARM ICE debugger** 0: Long reset mode, normal operation (default) 1: Fast reset mode, for ICE debugger connection Enable fast reset mode to enable ICE debugger can stop ARM at starting address 0.
26	RW	**Enable eSPI flash mode** 0: eSPI respond with no flash attached 1: eSPI respond with flash attached This bit is valid only when eSPI is enabled.
25	RW	**Enable eSPI mode** 0: LPC mode 1: eSPI mode
24	RW	**Select DDR4 SDRAM** 0: DDR3 SDRAM 1: DDR4 SDRAM
23	RW	**Select 25 MHz reference clock input mode** 0: CLKIN is 24 MHz and USBCKI not used 1: CLKIN is 25 MHz and USBCKI = 24/48 MHz (determined by bit[18])
22	RW	**Enable GPIOE pass-through mode** 0: Disable, pass through can be enabled by SCU8C[15:12]. 1: Enable pass-through at power on. Pass-through pins set: GPIOE0 → GPIOE1 GPIOE2 → GPIOE3 GPIOE4 → GPIOE5 GPIOE6 → GPIOE7
21	RW	**Enable GPIOD pass-through mode** 0: Disable, pass through can be enabled by SCU8C[11:8]. 1: Enable pass-through at power on. Pass-through pins set: GPIOD0 → GPIOD1 GPIOD2 → GPIOD3 GPIOD4 → GPIOD5 GPIOD6 → GPIOD7
20	RW	**Disable LPC to decode SuperIO 0x2E/0x4E address** 0: Enable address decoding (default) 1: Disable address decoding
19	RW	**Enable ACPI function** 0: Disable ACPI 1: Enable ACPI
18	RW	**Select USBCKI input frequency** 0: 24 MHz (default) 1: 48 MHz
17	RW	**Enable BMC 2nd boot watchdog timer** 0: Disable 1: Enable BMC 2nd boot watchdog timer start counting at power up. (default) The watchdog timer was located at WDT2. After watchdog timeout, it will reset BMC and restart booting from the 2nd boot flash at CS1#. CS1# must be the same flash type as CS0# and could be the same firmware as CS0#.

to next page

from previous page

16	RW	**SuperIO configuration address selection** 0: Decode 0x2E 1: Decode 0x4E
15	RW	**VGA Class Code selection** 0: Select the Class Code for video device 1: Select the Class Code for VGA device (default)
14	RW	**Select dedicated LPC reset input** 0: LPC reset is shared with PCIe reset pin 1: LPC reset is located at pin number G22, shared with GPIOAC7. (default)
13:12	RW	**SPI mode selection** 00: Disable SPI interface 01: Enable SPI Master 10: Reserved, enable SPI Master and SPI Slave to AHB Bridge (debug mode) 11: Enable SPI Pass-through
11:9	RW	**AXI/AHB clock frequency ratio selection** 000: undefined 001: Select AXI:AHB = 2:1 (default) 010: Select AXI:AHB = 3:1 011: Select AXI:AHB = 4:1 100: Select AXI:AHB = 5:1 101: Select AXI:AHB = 6:1 110: Select AXI:AHB = 7:1 111: Select AXI:AHB = 8:1 HCLK freq = H-PLL / 2 / (ratio_of_AXI-AHB)
8	RW	**Reserved (0)**
7	RW	**Define MAC#2 interface** 0: RMII/NCSI 1: RGMII
6	RW	**Define MAC#1 interface** 0: RMII/NCSI 1: RGMII
5	RW	**Enable dedicated VGA BIOS ROM** 0: No VGA BISO ROM, VGA BIOS is merged in the system BIOS (default) 1: Enable dedicated VGA BIOS ROM
4	RW	**Reserved (0)**

to next page

from previous page

3:2	RW	**VGA memory size selection** 00: Select 8 MB VGA memory 01: Select 16 MB VGA memory (default) 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
1	R	**Reserved (1)**
0	RW	**Disable CPU boot** 0: Enable boot 1: Disable CPU operation, when no firmware exist
Note : The write operation to 0x1E6E2070 only can set to '1', to clear to '0', it must write '1' to 0x1E6E207C (write 1 clear).		

2.3.5 Application Note 3: Hardware Strap Configuration

Please enable the hardware strap setting for resolving power up initial state issue.

Issue description:

1. This issue only occurs randomly at the first power on, or after SRST# was triggered.
2. When this issue occur, the hardware HBLED# output will blink fast (10Hz) for about 3 seconds and then stop. No any message output to the UART debug console. And measuring the firmware SPI bus, only one Read SFDP (0x5A) command was outputted.
3. This issue can be resolved by another AC OFF/ON cycle or SRST# reset event.

Solution:

- Set the GPIO strap GPIOZ7 = 1.
- And please also update the platform.S (DRAM initial sequence in firmware u-boot) to revision V7 (or after V7).

For your schematics design, please follow below schematic modification.

1. Pull up FWSPICK pin with resistor of 1K ~ 4.7K Ω to enable GPIO strap mode.
2. Pull up GPIOZ7 pin with resistor of 1K ~ 4.7K Ω.
3. Since this solution requires to enable GPIO strap pins, it will limit the usage of the 8 GPIO strap pins. It is better to use the 8 pins as GPIO output, and with power up default state matches the strap value setting.
   (a) When set strap value = 0, the power up pin state should be Low. It can be assigned to be GPIO output of power up state = Low.

(b) When set strap value = 1, the power up pin state should be High. It can be assigned to be GPIO output of power up state = High.

(c) If digital video output (DVO) interface is used, then no any limitation on the strap value setting.

4. GPIO strap pins definition.

Pin	Definition	Value = 0	Value = 1	Valid for PN
GPIOS4	VGA memory size selection	16 MB	64 MB	2500/2530
GPIOS5	Reserved	Default	Disallowed	2500/2530
GPIOS6	Enable VGA BIOS ROM	No	Yes	2500/2530
GPIOS7	VGA Class Code selection	Non-VGA	VGA	2500/2530
GPIOZ4	Enable BMC 2nd boot watchdog timer	No	Yes	2500/2520/2530
GPIOZ5	Select USBCKI input frequency	24MHz	48MHz	2500/2520/2530
GPIOZ6	Disable LPC to decode SuperIO address	No	Yes	2500/2520/2530
GPIOZ7	Enable fast reset mode	No	Yes	2500/2520/2530

(a) **GPIOS4**:

i. When VGA function is enabled, set value = 0.
ii. If no VGA, then initial state is not cared and no limitation on GPIO assignment.
iii. For AST2520, initial state is not cared and no limitation on GPIO assignment.

(b) **GPIOS5**:

i. Set value = 0.
ii. For AST2520, initial state is not cared and no limitation on GPIO assignment.

(c) **GPIOS6**:

i. When VGA function is enabled and VGA BIOS is merged in the system BIOS, set value = 0.
ii. When VGA function is enabled and VGA BIOS use dedicated SPI flash, set value = 1.
iii. If no VGA, then initial state is not cared and no limitation on GPIO assignment.
iv. For AST2520, initial state is not cared and no limitation on GPIO assignment.

(d) **GPIOS7**:

i. When VGA function is enabled, set value = 1.
ii. If no VGA, set value = 0.
iii. For AST2520, initial state is not cared and no limitation on GPIO assignment.

(e) **GPIOZ4**:

i. If 2nd boot function enabled, set value = 1.
ii. If no 2nd boot function, then initial state is not cared and no limitation on GPIO assignment.

(f) **GPIOZ5**:

i. If CLKIN = 25MHz, then strap value should match the USBCKI input frequency.
ii. If CLKIN = 24MHz, then this strap is not used. So initial state is not cared and no limitation on GPIO assignment.

(g) **GPIOZ6**:

i. If LPC is connected and SuperIO decode enabled, set value = 0.
ii. If LPC is connected and SuperIO decode disabled, set value = 1. (SuperIO decode functions include: COM port, ACPI pins, Mailbox, L2H bridge for socflash utility.)
iii. If LPC is not connected, then this strap is not used. So initial state is not cared and no limitation on GPIO assignment.

(h) **GPIOZ7**: Set value = 1.

2.4 Digital Video Interface

2.4.1 Video Mode

Video Output	Video Input
24 Bits (R8/G8/B8)	24 Bits (R8/G8/B8)
18 Bits (R6/G6/B6)	-
15 Bits (R5/G5/B5)	-
12 Bits (Dual Edge)	-

2.4.2 Digital Video Single Edge Data Mode

Inout Pin	Output Pin	24 Bits	18 Bits	15 Bits
VPIB2	VPOB2	B0	-	-
VPIB3	VPOB3	B1	-	-
VPIB4	VPOB4	B2	B0	-
VPIB5	VPOB5	B3	B1	B0
VPIB6	VPOB6	B4	B2	B1
VPIB7	VPOB7	B5	B3	B2
VPIB8	VPOB8	B6	B4	B3
VPIB9	VPOB9	B7	B5	B4
VPIG2	VPOG2	G0	-	-
VPIG3	VPOG3	G1	-	-
VPIG4	VPOG4	G2	G0	-
VPIG5	VPOG5	G3	G1	G0
VPIG6	VPOG6	G4	G2	G1
VPIG7	VPOG7	G5	G3	G2
VPIG8	VPOG8	G6	G4	G3
VPIG9	VPOG9	G7	G5	G4
VPIR2	VPOR2	R0	-	-
VPIR3	VPOR3	R1	-	-
VPIR4	VPOR4	R2	R0	-
VPIR5	VPOR5	R3	R1	R0
VPIR6	VPOR6	R4	R2	R1
VPIR7	VPOR7	R5	R3	R2
VPIR8	VPOR8	R6	R4	R3
VPIR9	VPOR9	R7	R5	R4

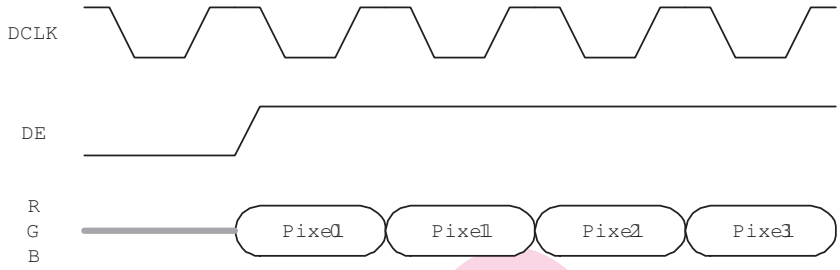


Figure 11: Single Edge Data Interface

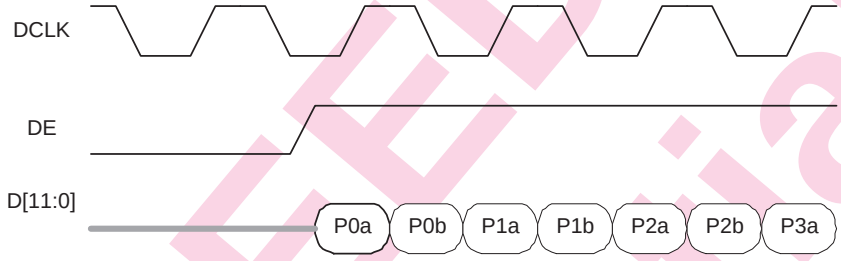


Figure 12: Dual Edge Data Interface

2.4.3 Digital Video Dual Edge Display Output Data Mode : 12 Bits Interface

Pixel #	P0a (rise)	P0b (fall)	P1a (rise)	P1b (fall)	P2a (rise)	P2b (fall)
VPOG5	G0[3]	R0[7]	G1[3]	R1[7]	G2[3]	R2[7]
VPOG4	G0[2]	R0[6]	G1[2]	R1[6]	G2[2]	R2[6]
VPOG3	G0[1]	R0[5]	G1[1]	R1[5]	G2[1]	R2[5]
VPOG2	G0[0]	R0[4]	G1[0]	R1[4]	G2[0]	R2[4]
VPOB9	B0[7]	R0[3]	B1[7]	R1[3]	B2[7]	R2[3]
VPOB8	B0[6]	R0[2]	B1[6]	R1[2]	B2[6]	R2[2]
VPOB7	B0[5]	R0[1]	B1[5]	R1[1]	B2[5]	R2[1]
VPOB6	B0[4]	R0[0]	B1[4]	R1[0]	B2[4]	R2[0]
VPOB5	B0[3]	G0[7]	B1[3]	G1[7]	B2[3]	G2[7]
VPOB4	B0[2]	G0[6]	B1[2]	G1[6]	B2[2]	G2[6]
VPOB3	B0[1]	G0[5]	B1[1]	G1[5]	B2[1]	G2[5]
VPOB2	B0[0]	G0[4]	B1[0]	G1[4]	B2[0]	G2[4]

2.5 GPIO Summary

- Each GPIO set can be programmed to accept command from ARM, Corprocessor or LPC.
- All 228 GPIOs are Full featured
  - Input interrupt with sensitive high/low level trigger, rising/falling edge trigger mode
  - Interrupt direction option (ARM, Corprocessor or LPC)
  - Input debouncing filter (3 sets of programmable debounce timer, 0 $\sim$ 1 second)
  - Watchdog reset tolerance (for non-interrupted related registers only)
  - The minimum input pulse width for edge trigger must longer than 2 PCLK cycle
- All GPIO have internal pull up/down resistor of value 52 $\sim$ 195 KΩ(typical 80KΩ). The pull up/down resistor can be turned-off per group at register SCU8C[31:16]. If enabled, the recommended external pull up resistor should not larger than 4.7K ohm.
- Please refer section 36.4.3 for the firmware programming method of push-pull and open-drain driving mode.

2.5.1 GPIO Table

GPIO	Ball	Pin Name	Driving	Input Buffer	Internal Resistor
GPIOA0	B14	GPIOA0/MAC1LINK	16mA	CMOS	PD
GPIOA1	D14	GPIOA1/MAC2LINK	16mA	CMOS	PD
GPIOA2	D13	GPIOA2/TIMER3/SPI1CS1#	16mA	CMOS	PD
GPIOA3	E13	GPIOA3/TIMER4	16mA	CMOS	PD
GPIOA4	C14	GPIOA4/TIMER5/SCL9	8mA	Schmitt	PD(I)
GPIOA5	A13	GPIOA5/TIMER6/SDA9	8mA	Schmitt	PD(I)
GPIOA6	C13	GPIOA6/TIMER7/MDC2	8mA	Schmitt	PD
GPIOA7	B13	GPIOA7/TIMER8/MDIO2	8mA	Schmitt	PD
GPIOB0	K19	GPIOB0	8mA	CMOS	PD
GPIOB1	L19	GPIOB1	8mA	CMOS	PD
GPIOB2	L18	GPIOB2	8mA	CMOS	PD
GPIOB3	K18	GPIOB3	8mA	CMOS	PD
GPIOB4	J20	GPIOB4/USBCKI	8mA	CMOS	PD
GPIOB5	H21	GPIOB5/LPCPD#/LPCSMI#	8mA	CMOS	PD
GPIOB6	H22	GPIOB6/LPCPME#	8mA	CMOS	PD
GPIOB7	H20	GPIOB7	8mA	CMOS	PD
GPIOC0	C12	GPIOC0/SD1CLK/SCL10	8mA	Schmitt	PD(I)
GPIOC1	A12	GPIOC1/SD1CMD/SDA10	8mA	Schmitt	PD(I)
GPIOC2	B12	GPIOC2/SD1DAT0/SCL11	8mA	Schmitt	PD(I)
GPIOC3	D9	GPIOC3/SD1DAT1/SDA11	8mA	Schmitt	PD(I)
GPIOC4	D10	GPIOC4/SD1DAT2/SCL12	8mA	Schmitt	PD(I)
GPIOC5	E12	GPIOC5/SD1DAT3/SDA12	8mA	Schmitt	PD(I)
GPIOC6	C11	GPIOC6/SD1CD#/SCL13	8mA	Schmitt	PD(I)
GPIOC7	B11	GPIOC7/SD1WP#/SDA13	8mA	Schmitt	PD(I)
GPIOD0	F19	GPIOD0/SD2CLK	8mA	CMOS	PD

to next page

from previous page

GPIOD1	E21	GPIOD1/SD2CMD	8mA	CMOS	PD
GPIOD2	F20	GPIOD2/SD2DAT0	8mA	CMOS	PD
GPIOD3	D20	GPIOD3/SD2DAT1	8mA	CMOS	PD
GPIOD4	D21	GPIOD4/SD2DAT2	8mA	CMOS	PD
GPIOD5	E20	GPIOD5/SD2DAT3	8mA	CMOS	PD
GPIOD6	G18	GPIOD6/SD2CD#	8mA	CMOS	PD
GPIOD7	C21	GPIOD7/SD2WP#	8mA	CMOS	PD
GPIOE0	B20	GPIOE0/NCTS3	8mA	CMOS	PD
GPIOE1	C20	GPIOE1/NDCD3	8mA	CMOS	PD
GPIOE2	F18	GPIOE2/NDSR3	8mA	CMOS	PD
GPIOE3	F17	GPIOE3/NRI3	8mA	CMOS	PD
GPIOE4	E18	GPIOE4/NDTR3	8mA	CMOS	PD
GPIOE5	D19	GPIOE5/NRTS3	8mA	CMOS	PD
GPIOE6	A20	GPIOE6/TXD3	8mA	CMOS	PD
GPIOE7	B19	GPIOE7/RXD3	8mA	CMOS	PD
GPIOF0	J19	GPIOF0/NCTS4/LHAD0	8mA	CMOS	PD
GPIOF1	J18	GPIOF1/NDCD4/LHAD1	8mA	CMOS	PD
GPIOF2	B22	GPIOF2/NDSR4/LHAD2	8mA	CMOS	PD
GPIOF3	B21	GPIOF3/NRI4/LHAD3	8mA	CMOS	PD
GPIOF4	A21	GPIOF4/NDTR4/LHCLK	8mA	CMOS	PD
GPIOF5	H19	GPIOF5/NRTS4/LHFRAME#	8mA	CMOS	PD
GPIOF6	G17	GPIOF6/TXD4/LHSIRQ#	8mA	CMOS	PD
GPIOF7	H18	GPIOF7/RXD4/LHRST#	8mA	CMOS	PD
GPIOG0	A19	GPIOG0/SGPS1CK	8mA	CMOS	PD
GPIOG1	E19	GPIOG1/SGPS1LD	8mA	CMOS	PD
GPIOG2	C19	GPIOG2/SGPS1I0	8mA	CMOS	PD
GPIOG3	E16	GPIOG3/SGPS1I1	8mA	CMOS	PD
GPIOG4	E17	GPIOG4/SGPS2CK/SALT1	8mA	CMOS	PD
GPIOG5	D16	GPIOG5/SGPS2LD/SALT2	8mA	CMOS	PD
GPIOG6	D15	GPIOG6/SGPS2I0/SALT3	8mA	CMOS	PD
GPIOG7	E14	GPIOG7/SGPS2I1/SALT4	8mA	CMOS	PD
GPIOH0	A18	GPIOH0/NCTS6	8mA	CMOS	PD
GPIOH1	B18	GPIOH1/NDCD6	8mA	CMOS	PD
GPIOH2	D17	GPIOH2/NDSR6	8mA	CMOS	PD
GPIOH3	C17	GPIOH3/NRI6	8mA	CMOS	PD
GPIOH4	A17	GPIOH4/NDTR6	8mA	CMOS	PD
GPIOH5	B17	GPIOH5/NRTS6	8mA	CMOS	PD
GPIOH6	A16	GPIOH6/TXD6	8mA	CMOS	PD
GPIOH7	D18	GPIOH7/RXD6	8mA	CMOS	PD
GPIOI0	C18	GPIOI0/SYSCS#	8mA	CMOS	PD
GPIOI1	E15	GPIOI1/SYSCK	8mA	CMOS	PD
GPIOI2	B16	GPIOI2/SYSMOSI	8mA	CMOS	PD

to next page

from previous page

GPIOI3	C16	GPIOI3/SYSMISO	8mA	CMOS	PD
GPIOI4	B15	GPIOI4/SPI1CS0#/VBCS#	8mA	CMOS	PD
GPIOI5	C15	GPIOI5/SPI1CK/VBCK	8mA	CMOS	PD
GPIOI6	A14	GPIOI6/SPI1MOSI/VBMOSI	8mA	CMOS	PD
GPIOI7	A15	GPIOI7/SPI1MISO/VBMISO	8mA	CMOS	PD
GPIOJ0	R2	GPIOJ0/SGPMCK	8mA	CMOS	PD
GPIOJ1	L2	GPIOJ1/SGPMLD	8mA	CMOS	PD
GPIOJ2	N3	GPIOJ2/SGPMO	8mA	CMOS	PD
GPIOJ3	N4	GPIOJ3/SGPMI	8mA	CMOS	PD
GPIOJ4	N5	GPIOJ4/VGAHS	8mA	CMOS	PD
GPIOJ5	R4	GPIOJ5/VGAVS	8mA	CMOS	PD
GPIOJ6	R3	GPIOJ6/DDCCLK	8mA	CMOS	PD
GPIOJ7	T3	GPIOJ7/DDCDAT	8mA	CMOS	PD
GPIOK0	L3	GPIOK0/SCL5	8mA	Schmitt	PD(I)
GPIOK1	L4	GPIOK1/SDA5	8mA	Schmitt	PD(I)
GPIOK2	L1	GPIOK2/SCL6	8mA	Schmitt	PD(I)
GPIOK3	N2	GPIOK3/SDA6	8mA	Schmitt	PD(I)
GPIOK4	N1	GPIOK4/SCL7	8mA	Schmitt	PD(I)
GPIOK5	P1	GPIOK5/SDA7	8mA	Schmitt	PD(I)
GPIOK6	P2	GPIOK6/SCL8	8mA	Schmitt	PD(I)
GPIOK7	R1	GPIOK7/SDA8	8mA	Schmitt	PD(I)
GPIOL0	T2	GPIOL0/NCTS1	8mA	CMOS	PD
GPIOL1	T1	GPIOL1/NDCD1/VPIDE	8mA	CMOS	PD
GPIOL2	U1	GPIOL2/NDSR1	8mA	CMOS	PD
GPIOL3	U2	GPIOL3/NRI1/VPIHS	8mA	CMOS	PD
GPIOL4	P4	GPIOL4/NDTR1/VPIVS	8mA	CMOS	PD
GPIOL5	P3	GPIOL5/NRTS1/VPICLK	8mA	CMOS	PD
GPIOL6	V1	GPIOL6/TXD1	8mA	CMOS	PD
GPIOL7	W1	GPIOL7/RXD1	8mA	CMOS	PD
GPIOM0	Y1	GPIOM0/NCTS2/VPIB2	8mA	CMOS	PD
GPIOM1	AB2	GPIOM1/NDCD2/VPIB3	8mA	CMOS	PD
GPIOM2	AA1	GPIOM2/NDSR2/VPIB4	8mA	CMOS	PD
GPIOM3	Y2	GPIOM3/NRI2/VPIB5	8mA	CMOS	PD
GPIOM4	AA2	GPIOM4/NDTR2/VPIB6	8mA	CMOS	PD
GPIOM5	P5	GPIOM5/NRTS2/VPIB7	8mA	CMOS	PD
GPIOM6	R5	GPIOM6/TXD2/VPIB8	8mA	CMOS	PD
GPIOM7	T5	GPIOM7/RXD2/VPIB9	8mA	CMOS	PD
GPION0	V2	GPION0/PWM0	8mA	CMOS	PD
GPION1	W2	GPION1/PWM1	8mA	CMOS	PD
GPION2	V3	GPION2/PWM2/VPIG2	8mA	CMOS	PD
GPION3	U3	GPION3/PWM3/VPIG3	8mA	CMOS	PD
GPION4	W3	GPION4/PWM4/VPIG4	8mA	CMOS	PD

to next page

from previous page

GPION5	AA3	GPION5/PWM5/VPIG5	8mA	CMOS	PD
GPION6	Y3	GPION6/PWM6/VPIG6	8mA	CMOS	PD
GPION7	T4	GPION7/PWM7/VPIG7	8mA	CMOS	PD
GPIOO0	U5	GPIOO0/TACH0/VPIG8	8mA	CMOS	PD
GPIOO1	U4	GPIOO1/TACH1/VPIG9	8mA	CMOS	PD
GPIOO2	V5	GPIOO2/TACH2	8mA	CMOS	PD
GPIOO3	AB4	GPIOO3/TACH3	8mA	CMOS	PD
GPIOO4	AB3	GPIOO4/TACH4/VPIR2	8mA	CMOS	PD
GPIOO5	Y4	GPIOO5/TACH5/VPIR3	8mA	CMOS	PD
GPIOO6	AA4	GPIOO6/TACH6/VPIR4	8mA	CMOS	PD
GPIOO7	W4	GPIOO7/TACH7/VPIR5	8mA	CMOS	PD
GPIOP0	V4	GPIOP0/TACH8/VPIR6	8mA	CMOS	PD
GPIOP1	W5	GPIOP1/TACH9/VPIR7	8mA	CMOS	PD
GPIOP2	AA5	GPIOP2/TACH10/VPIR8	8mA	CMOS	PD
GPIOP3	AB5	GPIOP3/TACH11/VPIR9	8mA	CMOS	PD
GPIOP4	Y6	GPIOP4/TACH12	8mA	CMOS	PD
GPIOP5	Y5	GPIOP5/TACH13	8mA	CMOS	PD
GPIOP6	W6	GPIOP6/TACH14	8mA	CMOS	PD
GPIOP7	V6	GPIOP7/TACH15	8mA	CMOS	PD
GPIOQ0	A11	GPIOQ0/SCL3	8mA	Schmitt	PD(I)
GPIOQ1	A10	GPIOQ1/SDA3	8mA	Schmitt	PD(I)
GPIOQ2	A9	GPIOQ2/SCL4	8mA	Schmitt	PD(I)
GPIOQ3	B9	GPIOQ3/SDA4	8mA	Schmitt	PD(I)
GPIOQ4	N21	GPIOQ4/SCL14	8mA	Schmitt	PD(I)
GPIOQ5	N22	GPIOQ5/SDA14	8mA	Schmitt	PD(I)
GPIOQ6	B10	GPIOQ6/OSCCLK	8mA	Schmitt	PD
GPIOQ7	N20	GPIOQ7/PEWAKE#	8mA	Schmitt	PD
GPIOR0	AA19	GPIOR0/FWSPICS1#	8mA	CMOS	PD
GPIOR1	T19	GPIOR1/FWSPICS2#	8mA	CMOS	PD
GPIOR2	T17	GPIOR2/SPI2CS0#	8mA	CMOS	PD
GPIOR3	Y19	GPIOR3/SPI2CK	8mA	CMOS	PD
GPIOR4	W19	GPIOR4/SPI2MOSI	8mA	CMOS	PD
GPIOR5	V19	GPIOR5/SPI2MISO	8mA	CMOS	PD
GPIOR6	D8	GPIOR6/MDC1	8mA	CMOS	PD
GPIOR7	E10	GPIOR7/MDIO1	8mA	CMOS	PD
GPIOS0	V20	GPIOS0/VPOB2/SPI2CS1#	8mA	CMOS	PD
GPIOS1	U19	GPIOS1/VPOB3/BMCINT	8mA	CMOS	PD
GPIOS2	R18	GPIOS2/VPOB4/SALT5	8mA	CMOS	PD
GPIOS3	P18	GPIOS3/VPOB5/SALT6	8mA	CMOS	PD
GPIOS4	R19	GPIOS4/VPOB6	8mA	CMOS	PD
GPIOS5	W20	GPIOS5/VPOB7	8mA	CMOS	PD
GPIOS6	U20	GPIOS6/VPOB8	8mA	CMOS	PD

to next page

from previous page

GPIOS7	AA20	GPIOS7/VPOB9	8mA	CMOS	PD
GPIOT0	B5	RGMII1TXCK/RMII1RCLKO/GPIOT0	8/16mA	CMOS	PD
GPIOT1	E9	RGMII1TXCTL/RMII1TXEN/GPIOT1	8/16mA	CMOS	PD
GPIOT2	F9	RGMII1TXD0/RMII1TXD0/GPIOT2	8/16mA	CMOS	PD
GPIOT3	A5	RGMII1TXD1/RMII1TXD1/GPIOT3	8/16mA	CMOS	PD
GPIOT4	E7	RGMII1TXD2/GPIOT4	8mA	CMOS	PD
GPIOT5	D7	RGMII1TXD3/GPIOT5	8mA	CMOS	PD
GPIOT6	B2	RGMII2TXCK/RMII2RCLKO/GPIOT6	8/16mA	CMOS	PD
GPIOT7	B1	RGMII2TXCTL/RMII2TXEN/GPIOT7	8/16mA	CMOS	PD
GPIOU0	A2	RGMII2TXD0/RMII2TXD0/GPIOU0	8/16mA	CMOS	PD
GPIOU1	B3	RGMII2TXD1/RMII2TXD1/GPIOU1	8/16mA	CMOS	PD
GPIOU2	D5	RGMII2TXD2/GPIOU2	8mA	CMOS	PD
GPIOU3	D4	RGMII2TXD3/GPIOU3	8mA	CMOS	PD
GPIOU4	B4	RGMII1RXCK/RMII1RCLKI/GPIOU4	8mA	CMOS	PD(P)
GPIOU5	A4	RGMII1RXCTL/GPIOU5	8mA	CMOS	PD(P)
GPIOU6	A3	RGMII1RXD0/RMII1RXD0/GPIOU6	8mA	CMOS	PD(P)
GPIOU7	D6	RGMII1RXD1/RMII1RXD1/GPIOU7	8mA	CMOS	PD(P)
GPIOV0	C5	RGMII1RXD2/RMII1CRSDV/GPIOV0	8mA	CMOS	PD(P)
GPIOV1	C4	RGMII1RXD3/RMII1RXER/GPIOV1	8mA	CMOS	PD(P)
GPIOV2	C2	RGMII2RXCK/RMII2RCLKI/GPIOV2	8mA	CMOS	PD(P)
GPIOV3	C1	RGMII2RXCTL/GPIOV3	8mA	CMOS	PD(P)
GPIOV4	C3	RGMII2RXD0/RMII2RXD0/GPIOV4	8mA	CMOS	PD(P)
GPIOV5	D1	RGMII2RXD1/RMII2RXD1/GPIOV5	8mA	CMOS	PD(P)
GPIOV6	D2	RGMII2RXD2/RMII2CRSDV/GPIOV6	8mA	CMOS	PD(P)
GPIOV7	E6	RGMII2RXD3/RMII2RXER/GPIOV7	8mA	CMOS	PD(P)
GPIW0	F4	ADC0/GPIW0	-	CMOS	PD(P)
GPIW1	F5	ADC1/GPIW1	-	CMOS	PD(P)
GPIW2	E2	ADC2/GPIW2	-	CMOS	PD(P)
GPIW3	E1	ADC3/GPIW3	-	CMOS	PD(P)
GPIW4	F3	ADC4/GPIW4	-	CMOS	PD(P)
GPIW5	E3	ADC5/GPIW5	-	CMOS	PD(P)
GPIW6	G5	ADC6/GPIW6	-	CMOS	PD(P)
GPIW7	G4	ADC7/GPIW7	-	CMOS	PD(P)
GPIX0	F2	ADC8/GPIX0	-	CMOS	PD(P)
GPIX1	G3	ADC9/GPIX1	-	CMOS	PD(P)
GPIX2	G2	ADC10/GPIX2	-	CMOS	PD(P)
GPIX3	F1	ADC11/GPIX3	-	CMOS	PD(P)
GPIX4	H5	ADC12/GPIX4	-	CMOS	PD(P)
GPIX5	G1	ADC13/GPIX5	-	CMOS	PD(P)
GPIX6	H3	ADC14/GPIX6	-	CMOS	PD(P)
GPIX7	H4	ADC15/GPIX7	-	CMOS	PD(P)
GPIOY0	R22	GPIOY0/SIOS3#	8mA	CMOS	PD

to next page

from previous page

GPIOY1	R21	GPIOY1/SIOS5#	8mA	CMOS	PD
GPIOY2	P22	GPIOY2/SIOPWREQ#	8mA	CMOS	PD
GPIOY3	P21	GPIOY3/SIOONCTRL#	8mA	CMOS	PD
GPIOY4	M18	GPIOY4/SCL1	8mA	Schmitt	PD(I)
GPIOY5	M19	GPIOY5/SDA1	8mA	Schmitt	PD(I)
GPIOY6	M20	GPIOY6/SCL2	8mA	Schmitt	PD(I)
GPIOY7	P20	GPIOY7/SDA2	8mA	Schmitt	PD(I)
GPIOZ0	Y20	GPIOZ0/VPOG2/NORA0/SIOPBI#	8mA	CMOS	PD
GPIOZ1	AB20	GPIOZ1/VPOG3/NORA1/SIOPWRGD	8mA	CMOS	PD
GPIOZ2	AB21	GPIOZ2/VPOG4/NORA2/SIOPBO#	8mA	CMOS	PD
GPIOZ3	AA21	GPIOZ3/VPOG5/NORA3/SIOSCI#	8mA	CMOS	PD
GPIOZ4	U21	GPIOZ4/VPOG6/NORA4	8mA	CMOS	PD
GPIOZ5	W22	GPIOZ5/VPOG7/NORA5	8mA	CMOS	PD
GPIOZ6	V22	GPIOZ6/VPOG8/NORA6	8mA	CMOS	PD
GPIOZ7	W21	GPIOZ7/VPOG9/NORA7	8mA	CMOS	PD
GPIOAA0	Y21	GPIOAA0/VPOR2/NORD0/SALT7	8mA	CMOS	PD
GPIOAA1	V21	GPIOAA1/VPOR3/NORD1/SALT8	8mA	CMOS	PD
GPIOAA2	Y22	GPIOAA2/VPOR4/NORD2/SALT9	8mA	CMOS	PD
GPIOAA3	AA22	GPIOAA3/VPOR5/NORD3/SALT10	8mA	CMOS	PD
GPIOAA4	U22	GPIOAA4/VPOR6/NORD4/SALT11	8mA	CMOS	PD
GPIOAA5	T20	GPIOAA5/VPOR7/NORD5/SALT12	8mA	CMOS	PD
GPIOAA6	N18	GPIOAA6/VPOR8/NORD6/SALT13	8mA	CMOS	PD
GPIOAA7	P19	GPIOAA7/VPOR9/NORD7/SALT14	8mA	CMOS	PD
GPIOAB0	N19	GPIOAB0/VPODE/NOROE#	8mA	CMOS	PD
GPIOAB1	T21	GPIOAB1/VPOHS/NORWE#	8mA	CMOS	PD
GPIOAB2	T22	GPIOAB2/VPOVS/WDTRST1	8mA	CMOS	PD
GPIOAB3	R20	GPIOAB3/VPOCLK/WDTRST2	16mA	CMOS	PD
GPIOAC0	G21	GPIOAC0/ESPID0/LAD0	12mA	Schmitt	PU
GPIOAC1	G20	GPIOAC1/ESPID1/LAD1	12mA	Schmitt	PU
GPIOAC2	D22	GPIOAC2/ESPID2/LAD2	12mA	Schmitt	PU
GPIOAC3	E22	GPIOAC3/ESPID3/LAD3	12mA	Schmitt	PU
GPIOAC4	C22	GPIOAC4/ESPICK/LCLK	12mA	Schmitt	PD
GPIOAC5	F21	GPIOAC5/ESPICS#/LFRAME#	12mA	Schmitt	PU
GPIOAC6	F22	GPIOAC6/ESPIALT#/LSIRQ#	12mA	Schmitt	PU
GPIOAC7	G22	GPIOAC7/ESPIRST#/LPCRST#	12mA	Schmitt	PD

Note:
1. **PD(P)** – indicates programmable internal pull low resistor and default OFF.
2. **PD(I)** – indicates internal pull low resistor is OFF when configured as I2C function.

2.5.2 GPIO Pass Through

GPIOD and GPIOE groups support maximum 8 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:

GPIOD0 → GPIOD1
GPIOD2 → GPIOD3
GPIOD4 → GPIOD5
GPIOD6 → GPIOD7
GPIOE0 → GPIOE1
GPIOE2 → GPIOE3
GPIOE4 → GPIOE5
GPIOE6 → GPIOE7

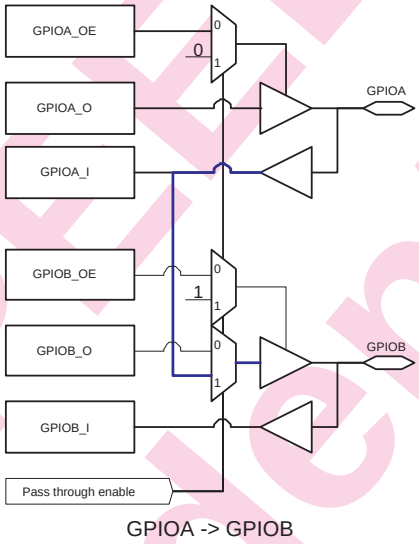


Figure 13: GPIO Pass Through Control

2.6 Serial GPIO Master

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

2.6.1 Timing Waveform

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

8-bits Mode

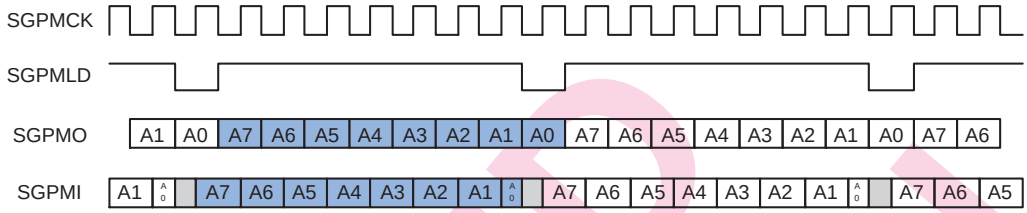


Figure 14: 8 Bits SGPIO Waveform

16-bits Mode

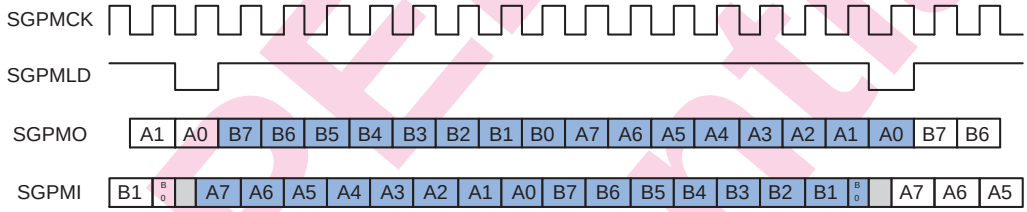


Figure 15: 16 Bits SGPIO Waveform

24-bits Mode

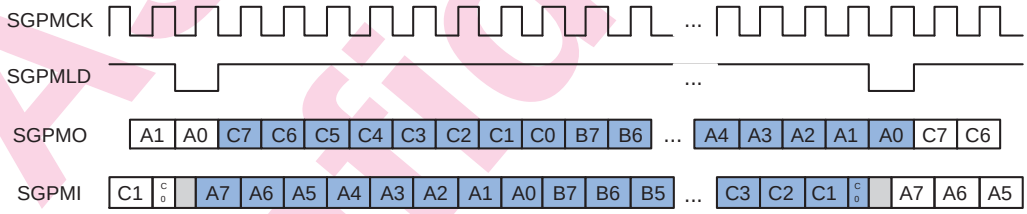


Figure 16: 24 Bits SGPIO Waveform

64-bits Mode

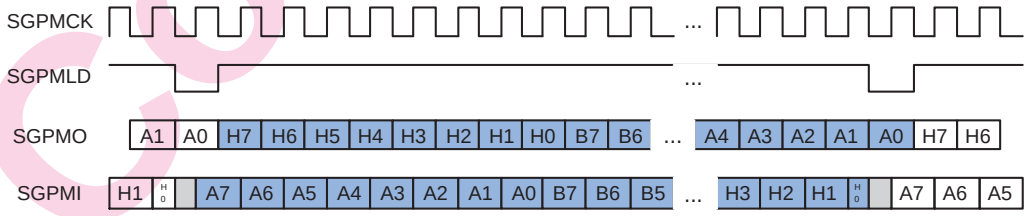


Figure 17: 64 Bits SGPIO Waveform

2.6.2 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
```

2.7 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 10 drives recording capability for each channel

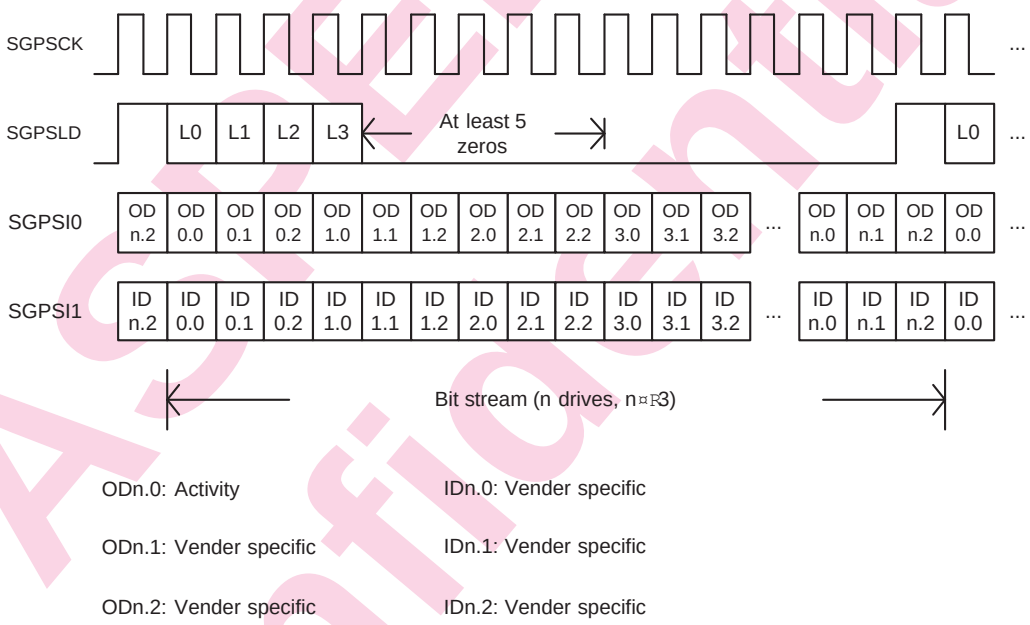


Figure 18: Slave SGPIO Waveform

3 Electrical Specifications

3.1 Absolute Maximum Ratings

PARAMETER	SYMBOL	MIN	TYP	MAX	UNIT
Core and PLL power	IV11D PEAV11 PLLDV11 VPLLAV11 PECIVDD	GND-0.3		1.26	V
DDR power	MVDD	GND-0.3		1.55	V
RGMII and RMII/NCSI power	R1VDD R2VDD	GND-0.3		3.96	V
I/O, AD/DA, PLL and USB power	ADCAV33 DACAV33 DACDV33 BATVDD LPVDD PLLAV33 MPLLAV33 VPLLAV33 PV33D PEAV33 USB2AV33	GND-0.3		3.96	V
CMOS IO input range		GND-0.3		3.6	V
RGMII and RMII 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
Core and PLL power	IV11D PEAV11 PLLDV11 VPLLAV11	1.09	1.15	1.21	V
PECI supply power	PECIVDD	0.85		1.21	V
DDR3L power	MVDD	1.283	1.35	1.45	V
DDR4 power	MVDD	1.14	1.2	1.26	V
RGMII/RMII/NCSI power	R1VDD R2VDD	2.375		3.465	V

to next page

from previous page

I/O, AD/DA, PLL and USB power	ADCAV33 DACAV33 DACDV33 PLLAV33 MPLLAV33 VPLLAV33 PV33D PEAV33 USB2AV33	3.135	3.3	3.465	V
Adjustable I/O power - 1.8V	LPVDD	1.71	1.8	1.89	V
Adjustable I/O power - 2.5V	LPVDD	2.375	2.5	2.625	V
Adjustable I/O power - 3.3V	LPVDD	3.135	3.3	3.465	V
Battery Power	BATVDD	2.0		3.465	V
AC Power Noise (under BGA power balls) (PLLAV33, VPLLAV33, MPLLAV33) (others)	1.15V 1.35V 1.2V 1.8V 2.5V 3.3V 3.3V			80 100 100 100 150 150 200	mV_{pp}
Ambient operation temperature	T_A	0		70	^oC
Chip top surface temperature	T_C	0		90	^oC

3.3 Input Overshoot/Undershoot Tolerance

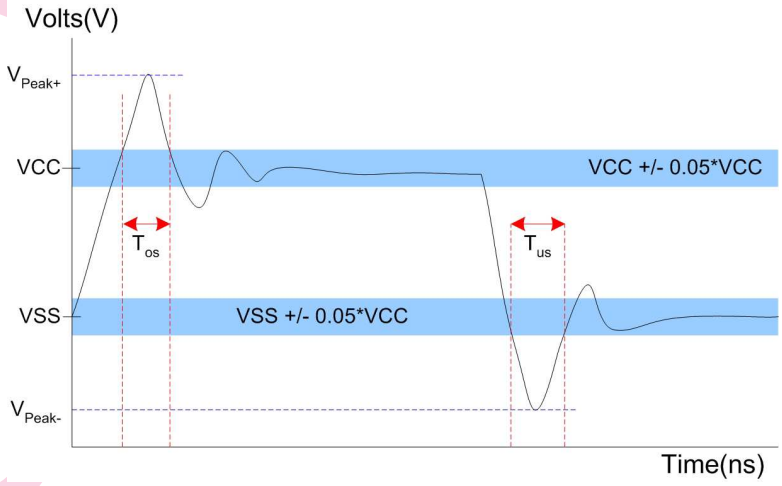


Figure 19: Overshoot/Undershoot

PARAMETER	SYMBOL	MIN	TYP	MAX	UNIT
Overshoot peak voltage	V_{Peak+}			min(1.3*VCC, 4)	V
Overshoot period	T_{OS}			5	ns

to next page

from previous page

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

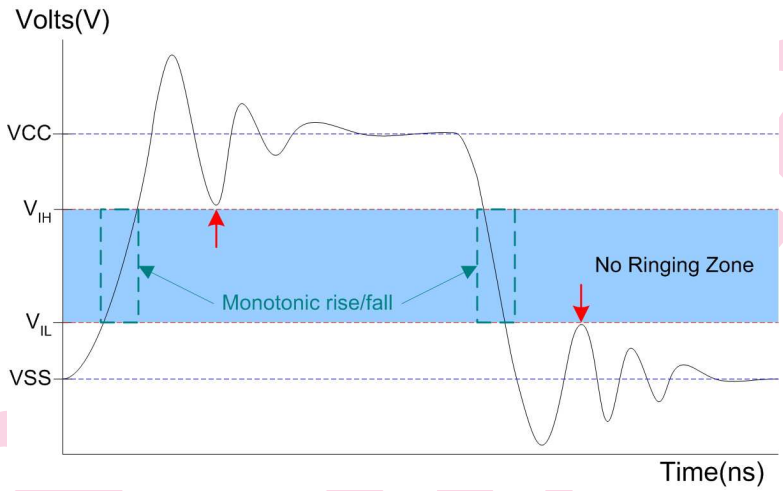


Figure 20: Ringback

- 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:

- (DDR3) 512MB DDR3L x1 SDRAM: Hynix H5TC4G63AFR-PBA
- (DDR4) 512MB DDR4 x1 SDRAM: Micron EDY4016AABG-DR-F
- 32MB SPI Flash x1: Macronix MX25L25(6/7)35FMI-10G or Winbond W25Q25(6/7)FVFQ

- LAN x 1 RGMII 1G: RTL8211E
- CPU frequency: 792 MHz
- AHB frequency: 198 MHz
- APB frequency: 24.75 MHz
- DDR frequency: DDR3L-1600/DDR4-1600 (792 MHz)

The power was measured based on the following conditions:

- **COND1**: Maximum Condition Power (AST2500/AST2530)
  1. Windows 1680x1050x32bpp@60Hz + 2D Bltblt stress
  2. Firmware run R-KVM + Iperf LAN stress
  3. No other digital IO swing, except LAN1(RGMII 125MHz)
- **COND2**: Normal Condition Power (AST2500/AST2530)
  1. Windows 1680x1050x32bpp@60Hz + 2D Bltblt stress
  2. Firmware run Iperf LAN stress
  3. No other digital IO swing, except LAN1(RGMII 125MHz)
- **COND3**: Minimum Condition Power (AST2500/AST2520/AST2530)
  1. VGA OFF
  2. Firmware Idle
  3. No any digital IO swing
- **COND4**: Initial Condition Power (AST2500/AST2520/AST2530)
  1. VGA OFF
  2. Firmware booting
  3. No other digital IO swing, except firmware SPI
- **COND5**: Standby Condition Power (AST2500/AST2520/AST2530)
  1. VGA OFF
  2. Firmware run Iperf LAN stress
  3. No other digital IO swing, except LAN1(RGMII 125MHz)

Power rail	COND1		COND2		COND3		COND4		COND5	
(units: mA)	mean	max	mean	max	mean	max	mean	max	mean	max
Core 1.15V power	495	619	475	586	323	330	442	663	408	510
DDR3 1.35V power *(Include DRAM)*	394	501	360	457	294	374	300	382	324	412
DDR4 1.20V power *(Include DRAM)*	395	495	348	487	232	397	240	305	276	385
3.3V power *PV33D/LPVDD*	20	21	20	21	3	4	5	7	5	7
3.3V power *R1VDD/R2VDD*	16	17	16	17	8	9	16	17	16	17
3.3V power *PLLAV33/MPLLAV33*	6	7	6	7	6	7	6	7	6	7
3.3V power *VPLLAV33*	2	3	2	3	2	3	2	3	2	3
3.3V power *ADCAV33*	2	3	2	3	2	3	2	3	2	3
3.3V power *DACAV33/DACDV33*	60	61	60	61	2	2	2	2	2	2
Chip total power *(DDR3, Watt)*	1.46	1.76	1.39	1.66	0.85	0.98	1.03	1.41	1.02	1.28
Chip total power *(DDR4, Watt)*	1.40	1.68	1.32	1.63	0.73	0.95	0.91	1.26	0.91	1.18

3.7 Power Up Sequence

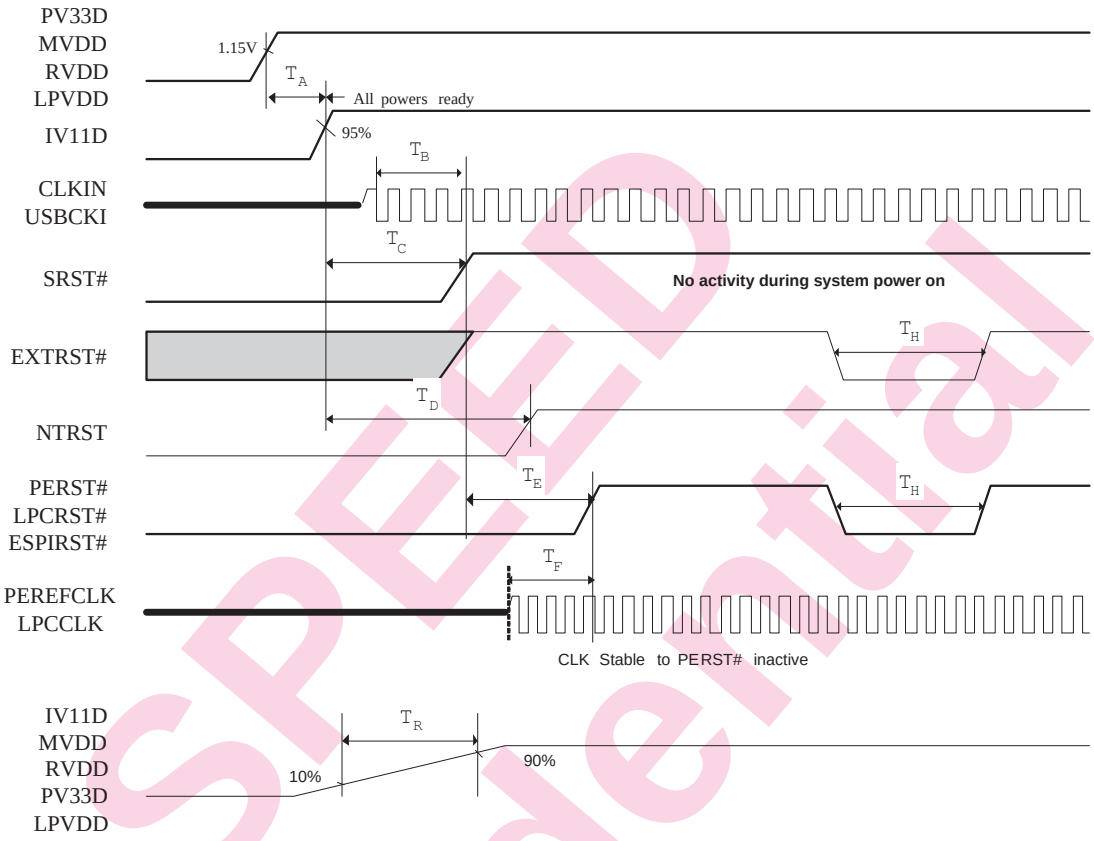


Figure 21: Power-up sequence

Power-up Sequence Parameters

PARAMETER	SYMBOL	MIN	TYP	MAX	UNIT
PV33D, MVDD, RVDD or LPVDD to IV11D delay, PV33D, MVDD, RVDD and LPVDD do not require sequence with each other	T_A	0			ms
CLKIN/USBCKI 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/LPC/eSPI reset	T_E	10			ms
PCI-E/LPC 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				300	ns

Notes:

1. There is no power-up sequence requirement for PECIVDD, it is connected to the CPU V_{TT} power.
2. When CLKIN=25MHz, then USBCKI is required. And USBCKI should be running during firmware booting. The USB driver requires USB clock is running so can initialize successfully.

3. To reduce firmware effort, we don't recommend to use CLKIN=25MHz configuration in the design.

3.8 Power Down Sequence for Firmware

§ **The power down sequence is only defined for firmware to be stopped safely.**
§ **Chip hardware has no power down sequence requirement.**

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.

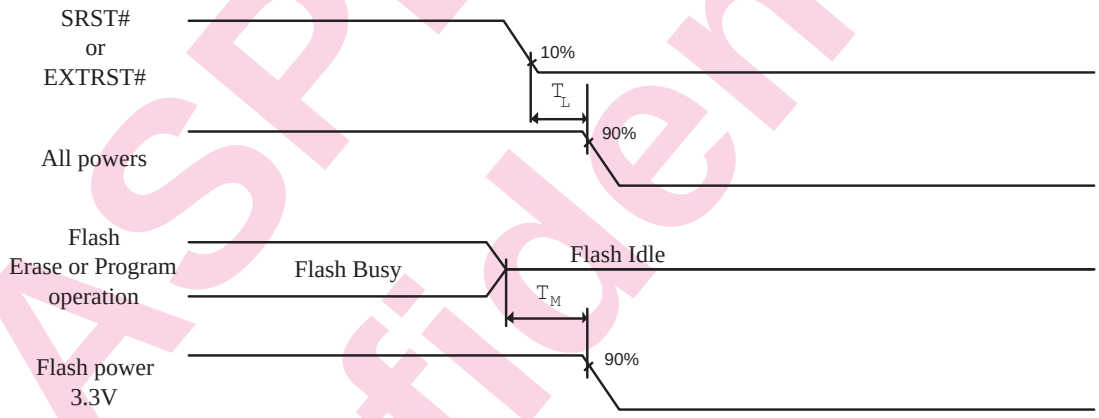


Figure 22: Power-down sequence

Power-down Sequence Parameters

PARAMETER	SYMBOL	MIN	TYP	MAX	UNIT
SRST# or EXTRST# (either one or both) active to power drop delay. There is no power drop sequence requirement for different power rails. 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

3.9 I/O DC Electrical Specification

3.9.1 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	1.34	1.45	1.59	V
Threshold Point with PU Resistor Enabled	V_Tpu	1.31	1.42	1.55	V
Threshold Point with PD Resistor Enabled	V_Tpd	1.37	1.49	1.63	V
Schmitt Trig. Low to High Threshold	V_T+	1.57	1.68	1.81	V
Schmitt Trig. High to Low Threshold	V_T-	1.21	1.32	1.45	V
Schmitt Trig. Low to High Threshold with PD Resistor Enabled	$V_T + pd$	1.61	1.72	1.86	V
Schmitt Trig. High to Low Threshold with PD Resistor Enabled	$V_T - pd$	1.23	1.35	1.49	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[1]	R_{PU}	58	86	133	KΩ
Pull-down Resistor	R_{PD}	52	78	128	KΩ
Low Level Output Current - O8 @ V_{OL}=0.4V	I_{OL}	12.7	19.3	25.9	mA
Low Level Output Current - O12 @ V_{OL}=0.4V	I_{OL}	16.9	25.6	34.1	mA
Low Level Output Current - O16 @ V_{OL}=0.4V	I_{OL}	21.0	31.9	42.5	mA
High Level Output Current - O8 @ V_{OH}=2.4V	I_{OH}	18.7	37.0	62.8	mA
High Level Output Current - O12 @ V_{OH}=2.4V	I_{OH}	24.0	47.5	80.5	mA
High Level Output Current - O16 @ V_{OH}=2.4V	I_{OH}	32.0	63.2	107.1	mA

3.9.2 2.5V CMOS I/O Parameters

PARAMETER	SYMBOL	MIN	TYP	MAX	UNIT
Input Low Voltage	V_{IL}	-0.3		0.7	V
Input High Voltage	V_{IH}	1.7		3.6	V
Output Low Voltage @ I_{OL}(min)	V_{OL}			0.7	V
Output High Voltage @ I_{OH}(min)	V_{OH}	1.7			V
Threshold Point	V_T	1.03	1.13	1.23	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 with PU Resistor Enabled	V_Tpu	1.01	1.11	1.20	V
Threshold Point with PD Resistor Enabled	V_Tpd	1.06	1.15	1.25	V
Schmitt Trig. Low to High Threshold	V_T+	1.23	1.34	1.43	V
Schmitt Trig. High to Low Threshold	V_T-	0.92	1.01	1.12	V
Schmitt Trig. Low to High Threshold with PD Resistor Enabled	$V_T + pd$	1.25	1.37	1.46	V
Schmitt Trig. High to Low Threshold with PD Resistor Enabled	$V_T - pd$	0.93	1.03	1.15	V
Input Leakage Current @ V_I=2.5V or 0V	I_L			±10	μA
Tri-state Output Leakage Current @ V_O=2.5V or 0V	I_{OZ}			±10	μA
Pull-up Resistor	R_{PU}	76	119	195	KΩ
Pull-down Resistor	R_{PD}	63	101	178	KΩ
Low Level Output Current - O8 @ V_{OL}=0.7V	I_{OL}	14.8	24.6	36.0	mA
Low Level Output Current - O12 @ V_{OL}=0.7V	I_{OL}	19.7	32.7	47.7	mA
Low Level Output Current - O16 @ V_{OL}=0.7V	I_{OL}	24.5	40.8	59.4	mA
High Level Output Current - O8 @ V_{OH}=1.7V	I_{OH}	13.0	26.0	45.4	mA
High Level Output Current - O12 @ V_{OH}=1.7V	I_{OH}	16.6	33.4	58.3	mA
High Level Output Current - O16 @ V_{OH}=1.7V	I_{OH}	22.2	44.4	77.5	mA

3.9.3 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		3.6	V
Output Low Voltage @ I_{OL}(min)	V_{OL}			0.45	V
Output High Voltage @ I_{OH}(min)	V_{OH}	1.35			V
Threshold Point	V_T	0.77	0.84	0.91	V
Threshold Point with PU Resistor Enabled	V_Tpu	0.75	0.83	0.90	V
Threshold Point with PD Resistor Enabled	V_Tpd	0.78	0.86	0.93	V
Schmitt Trig. Low to High Threshold	V_T+	0.93	1.02	1.11	V
Schmitt Trig. High to Low Threshold	V_T-	0.62	0.73	0.82	V
Schmitt Trig. Low to High Threshold with PU Resistor Enabled	$V_T + pu$	0.92	1.01	1.10	V
Schmitt Trig. High to Low Threshold with PU Resistor Enabled	$V_T - pu$	0.61	0.72	0.80	V
Schmitt Trig. Low to High Threshold with PD Resistor Enabled	$V_T + pd$	0.95	1.05	1.13	V

to next page

from previous page

Schmitt Trig. High to Low Threshold with PD Resistor Enabled	$V_T - pd$	0.63	0.74	0.83	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}	117	194	331	KΩ
Pull-down Resistor	R_{PD}	91	159	291	KΩ
Low Level Output Current - O8 @ V_{OL}=0.45V	I_{OL}	6.8	12.2	19.5	mA
Low Level Output Current - O12 @ V_{OL}=0.45V	I_{OL}	9.1	16.2	25.8	mA
Low Level Output Current - O16 @ V_{OL}=0.45V	I_{OL}	11.3	20.2	32.2	mA
High Level Output Current - O8 @ V_{OH}=1.35V	I_{OH}	4.8	11.4	22.2	mA
High Level Output Current - O12 @ V_{OH}=1.35V	I_{OH}	6.2	14.7	28.5	mA
High Level Output Current - O16 @ V_{OH}=1.35V	I_{OH}	8.2	19.5	38.0	mA

3.9.4 DDR3L I/O Parameters

PARAMETER	SYMBOL	MIN	TYP	MAX	UNIT
Supply power voltage	$MVDD$	1.283	1.35	1.45	V
Reference Voltage	V_{REF}	0.49*MVDD	0.5*MVDD	0.51*MVDD	V
Termination Voltage	V_{TT}	V_{REF}-0.04	V_{REF}	V_{REF}+0.04	V
Input High Voltage	V_{IH}	V_{REF}+0.1		MVDD	V
Input Low Voltage	V_{IL}	0		V_{REF}-0.1	V
Note : Follow JEDEC JESD79-3 standard specification.					

3.9.5 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.					

3.9.6 DAC I/O Parameters

PARAMETER	SYMBOL	MIN	TYP	MAX	UNIT
DACAV33/DAVDV33 supply DC current	$\|I\|$			78	mA
Power down current	$\|I\|$		30	50	uA
DACVREF output voltage		1.179	1.205	1.226	V

3.9.7 ADC I/O Parameters

PARAMETER	SYMBOL	MIN	TYP	MAX	UNIT
ADCAV33 supply DC current	$\|I\|$		1.6	2.6	mA
Power down current	$\|I\|$		10		uA
Resolution			10		Bit
VREFP voltage (ENVR=1)			1.8		V
VREFN voltage			0		V
Analog input range ADC[11:0]		0		1.8	V
Voltage accuracy (before compensation)			$\pm$45		mV
Voltage accuracy (after compensation)			$\pm$18		mV
Differential Non-Linearity (DNL)	DNL	-1		1	LSB
Integral Non-Linearity (INL)	INL	-1.5		1.5	LSB
Sampling rate	F_S			1	MHz

Note:

1. **DNL - Differential Non-Linearity**: For an ideal ADC the output is divided into 2 power n uniform steps each with the width. Any deviation from the ideal step width is the Differential Non-Linearity (DNL). It is expressed as counts. DNL is a function of each ADC's particular architecture. It is not possible to remove its effects with calibration.
2. **INL - Integral Non-Linearity**: DNL errors accumulate to produce a total Integral Non-Linearity (INL). It is defined as the maximum deviation from the ideal slope of the ADC and is measured from the center of the step. It is expressed as counts. INL is a function of each ADC's particular architecture. It is not possible to remove its effects with calibration.

3.9.8 USB I/O Parameters

PARAMETER	SYMBOL	MIN	TYP	MAX	UNIT
USB2AV33 supply power voltage		3.135	3.3	3.465	V
Maximum current (2 ports)	I_{max}			50	mA
Idle state current (1 port)	I_{idle}			7	mA
Suspend state current (1 port)	I_{spd}			52	uA

3.9.9 Battery Backed SRAM Parameters

PARAMETER	SYMBOL	MIN	TYP	MAX	UNIT
BATVDD supply power voltage	$BATVDD$	2.0		3.6	V

to next page

from previous page

Operation current @ IV11D=0V	I_{leak_pd}			1.04	uA
Operation current @ IV11D=1.15V	I_{leak_po}			3.8	uA

3.9.10 PECI I/O Parameters

PARAMETER	SYMBOL	MIN	TYP	MAX	UNIT
PECIVDD supply power voltage	V_{TT}	0.85		1.21	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}	0.47		1.1	mA
Output High Current @ 0.75V_{TT}	I_{OH}	6		20	mA

3.9.11 RGMII/RMII/NCSI I/O Parameters

R1VDD/R2VDD = 3.3V

PARAMETER	SYMBOL	MIN	TYP	MAX	UNIT
Supply power voltage	$RVDD$	3.135	3.3	3.465	V
Input High Voltage	V_{IH}	2		3.6	V
Input Low Voltage	V_{IL}	-0.3		0.8	V
Output Low Voltage	V_{OL}			0.4	V
Output High Voltage	V_{OH}	2.4		RVDD	V

R1VDD/R2VDD = 2.5V

PARAMETER	SYMBOL	MIN	TYP	MAX	UNIT
Supply power voltage	$RVDD$	2.375	2.5	2.625	V
Input High Voltage	V_{IH}	1.7		3.6	V
Input Low Voltage	V_{IL}	-0.3		0.7	V
Output Low Voltage	V_{OL}			0.7	V
Output High Voltage	V_{OH}	1.7		RVDD	V

3.10 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-}

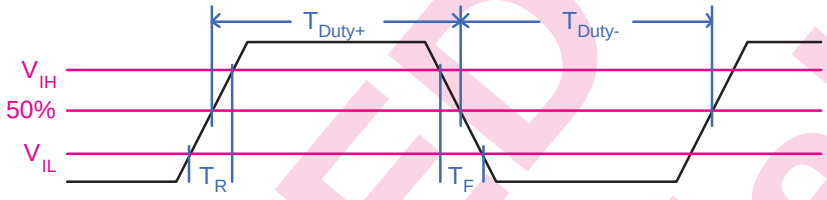


Figure 23: Timing Definition

3.10.1 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			24/25		MHz
Frequency Stability/Tolerance		-50		+50	ppm
Duty Cycle Ratio		45		55	%
Input Rise time	T_R			5	ns
Input Fall time	T_F			5	ns

USBCKI Input Requirements

PARAMETER	SYMBOL	MIN	TYP	MAX	UNIT
Input Low Voltage (LPVDD = 3.3V)	V_{IL}	-0.3		0.8	V
Input High Voltage (LPVDD = 3.3V)	V_{IH}	2.0		3.6	V
Input Low Voltage (LPVDD = 1.8V)	V_{IL}	-0.3		0.63	V
Input High Voltage (LPVDD = 1.8V)	V_{IH}	1.17		3.6	V
Nominal Frequency			24/48		MHz
Frequency Stability/Tolerance		-50		+50	ppm
Duty Cycle Ratio		45		55	%
Input Rise time	T_R			5/2.5	ns
Input Fall time	T_F			5/2.5	ns

Note :
When use USBCKI input, firmware should modify the USB driver loading sequence. Since USB function registers are not accessible if USBCKI is not active. So the USB driver can only be loaded when USBCKI is active.
No recommend to use CLKIN=25MHz in design.

Power on default clock frequency (no recommend to use 25MHz mode)

Clock name	CLKIN=24MHz	CLKIN=25MHz	Units
CPUCLK	792	825	MHz
HCLK	198	206.25	MHz
PCLK	24.75	25.78125	MHz
DDRCK	792	800	MHz

3.10.2 LPC Interface

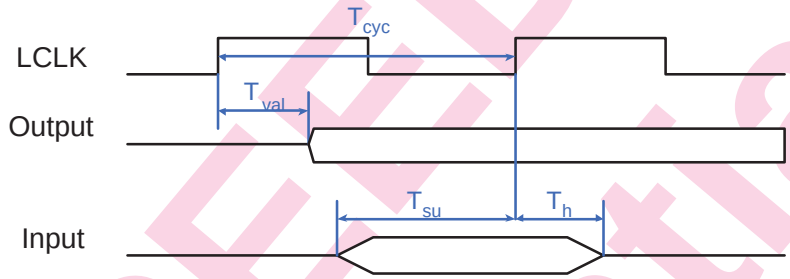


Figure 24: LPC Timing Waveform

LPC 33MHz

PARAMETER	SYMBOL	MIN	TYP	MAX	UNIT
Clock cycle time	T_{cyc}	30			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

3.10.3 RGMII/RMII/NCSI Interface

AST2500 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. And adjust the timing setting in firmware (in the code of DRAM initial sequence) 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 for RGMII interface was set as below:

- Use internal generated 125MHz reference clock.
- TX path : center-aligned output
- RX path : edge-aligned input

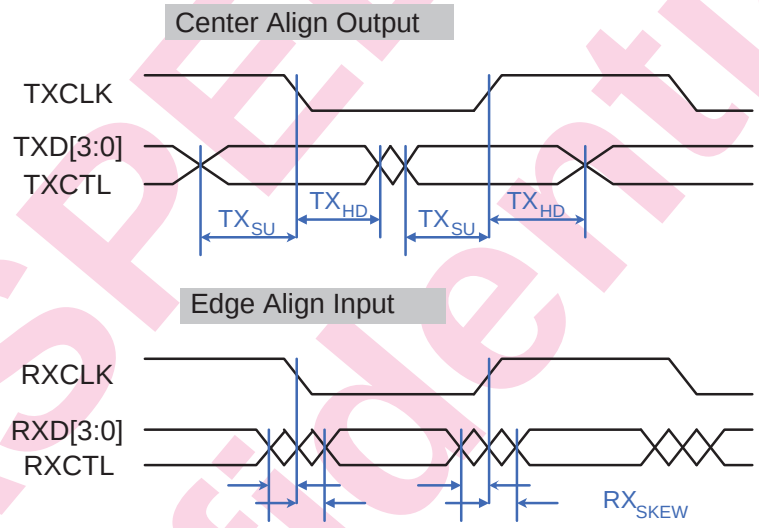


Figure 25: RGMII Timing Waveform

RGMII Timing

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 (RnVDD=2.5V, n=1,2)	T_R			1.8	ns
Input Fall time (RnVDD=2.5V, n=1,2)	T_F			1.8	ns
Input Rise time (RnVDD=3.3V, n=1,2)	T_R			2.0	ns
Input Fall time (RnVDD=3.3V, n=1,2)	T_F			2.0	ns

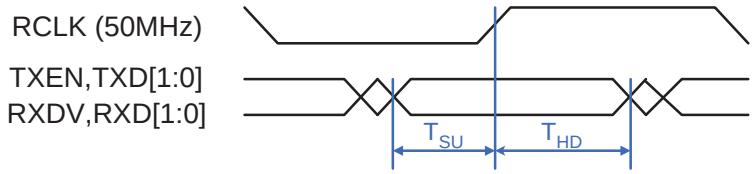


Figure 26: RMII/NCSI Timing Waveform

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	%
Data output setup time	TX_{SU}	10			ns
Data output hold time	TX_{HD}	5			ns
Data input setup time	RX_{SU}	1			ns
Data input hold time	RX_{HD}	5			ns
Input Rise time	T_R			3.5	ns
Input Fall time	T_F			3.5	ns

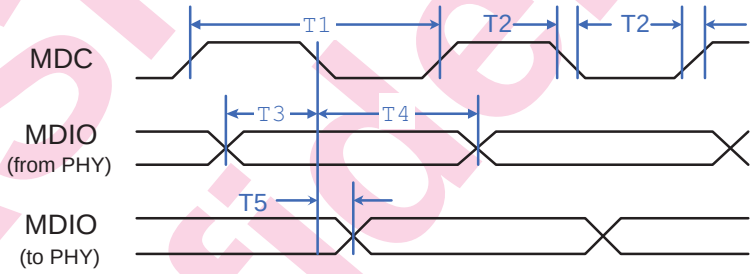


Figure 27: MDC/MDIO Timing Waveform

MDC/MDIO Timing

PARAMETER	SYMBOL	MIN	TYP	MAX	UNIT
MDC cycle time	$T1$			800	ns
MDC rise/fall time	$T2$			10	ns
MDIO input setup time	$T3$	20			ns
MDIO input hold time	$T4$	20			ns
MDIO output delay	$T5$			20	ns

3.10.4 DDR3/DDR4 Interface

Embedded a physical layer PHY for DDR3L/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}		1.25		ns
Note : Follow JEDEC JESD79-3 and JESD79-4 specification.					

3.10.5 Video Interface Input/Output: Single Edge

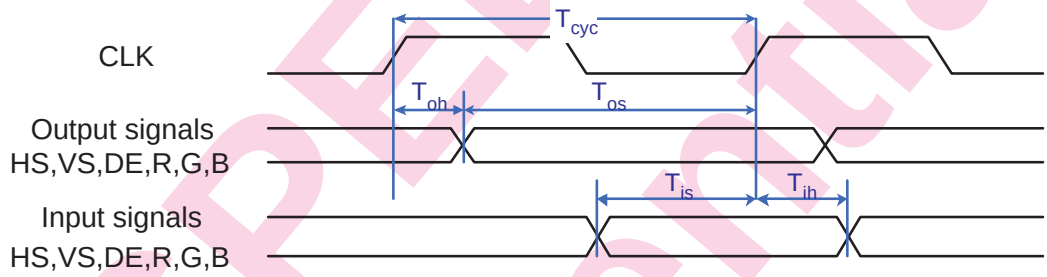


Figure 28: Video SDR Timing Waveform

Video SDR Timing

PARAMETER	SYMBOL	MIN	TYP	MAX	UNIT
Clock cycle time	T_{cyc}	6		40	ns
Data output setup time	T_{os}	1			ns
Data output hold time	T_{oh}	1			ns
Data input setup time	T_{is}	1			ns
Data input hold time	T_{ih}	1			ns
Input Rise time	T_R			1	ns
Input Fall time	T_F			1	ns

3.10.6 Video Interface Output: Dual Edge

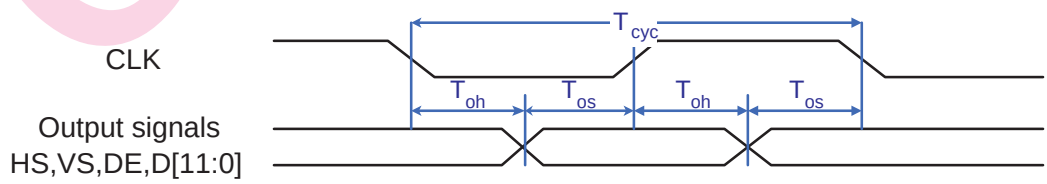


Figure 29: Video DDR Timing Waveform

Video DDR Timing

PARAMETER	SYMBOL	MIN	TYP	MAX	UNIT
Clock cycle time	T_{cyc}	6		40	ns
Data output setup time	T_{os}	1			ns
Data output hold time	T_{oh}	1			ns

3.10.7 SPI Master Interface

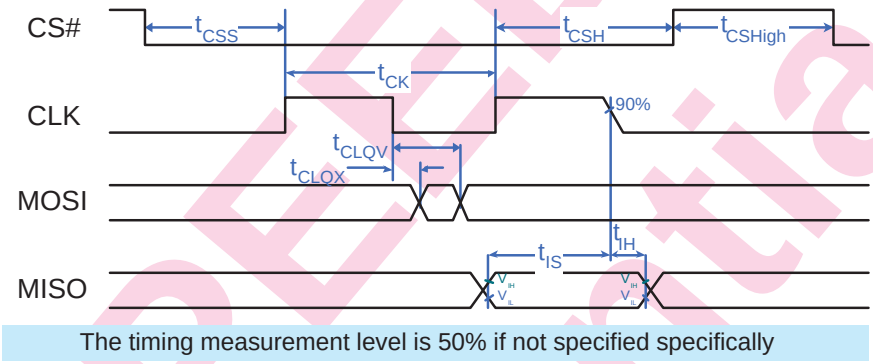


Figure 30: SPI Master Timing Waveform

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/16		MHz
Programmable SPI clock frequency	f_{SPI}	HCLK/64		100	MHz
Programmable SPI Clock period	t_{CK}	10		$64t_{AHB}$	ns
CS# Setup time	t_{CSS}	0.5			t_{CK}
CS# Hold time	t_{CSH}	0.5			t_{CK}
CS# Inactive time (programmable)	t_{CSHigh}	2		32	t_{AHB}
Data output Hold time	t_{CLQX}	-1			ns
Data output Valid time	t_{CLQV}			2	ns
Data input from flash Setup time (no path delay compensation)	t_{IS}	10			ns
Data input from flash 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.

3.10.8 eSPI Slave Interface

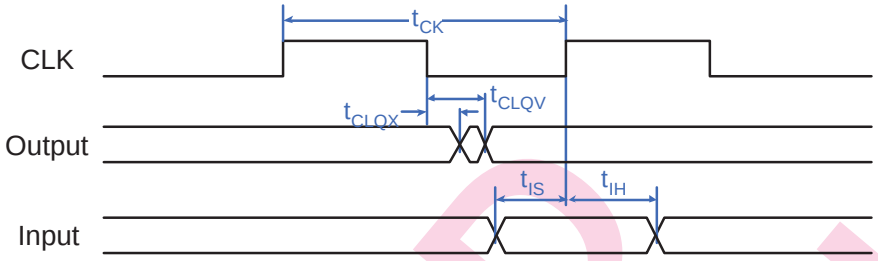


Figure 31: eSPI Slave Timing Waveform

PARAMETER	SYMBOL	MIN	TYP	MAX	UNIT
eSPI clock frequency (load $\leq$ 10pF)	f_{eSPI}			48	MHz
eSPI clock frequency (load $\leq$ 30pF)	f_{eSPI}			33	MHz
eSPI clock frequency (load $>$ 30pF)	f_{eSPI}			25	MHz
Data output Hold time	t_{CLQX}	3			ns
Data output Valid time (load $\leq$ 10pF)	t_{CLQV}			8	ns
Data output Valid time (load $\leq$ 30pF)	t_{CLQV}			10	ns
Data output Valid time (load $>$ 30pF)	t_{CLQV}			15	ns
Data input from master Setup time	t_{IS}	3			ns
Data input from master Hold time	t_{IH}	3			t_{CK}
Input Rise time	T_R			3.5	ns
Input Fall time	T_F			3.5	ns

3.10.9 SPI Pass-through Propagation Delay

PARAMETER	SYMBOL	MIN	TYP	MAX	UNIT
SYSCS# to SPI1CS0#	tpd_{CS}		4.5	5	ns
SYSCK to SPI1CK	tpd_{CK}		4.5	5	ns
SYSDO to SPI1MOSI	tpd_{DO}		4.5	5	ns
SPI1MISO to SYSDI	tpd_{DI}		4.5	5	ns
Note : Due to the progagation delay, the SPI clock rate from the host should not over 30MHz.					

3.10.10 JTAG Master Interface

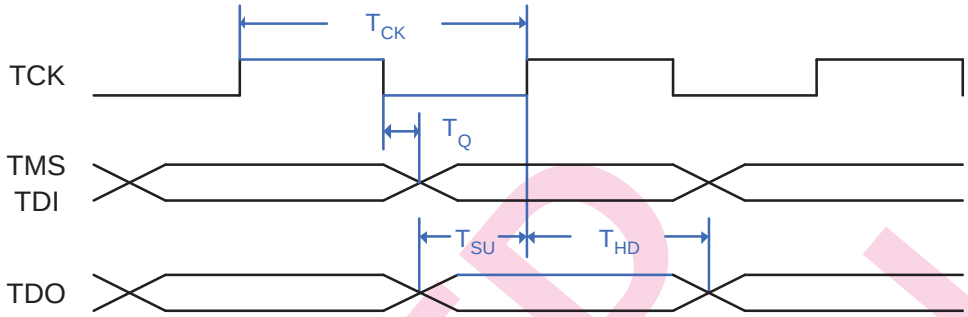


Figure 32: JTAG Master Timing Waveform

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

3.10.11 I2C/SMBus Interface

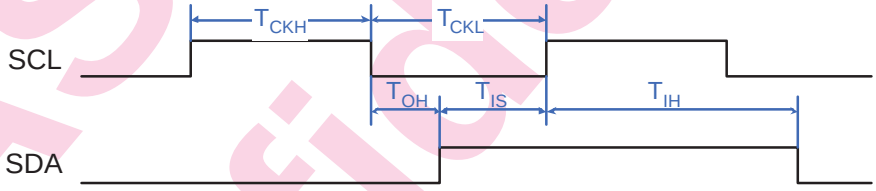


Figure 33: I2C Timing Waveform

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

3.10.12 SD/eMMC Interface

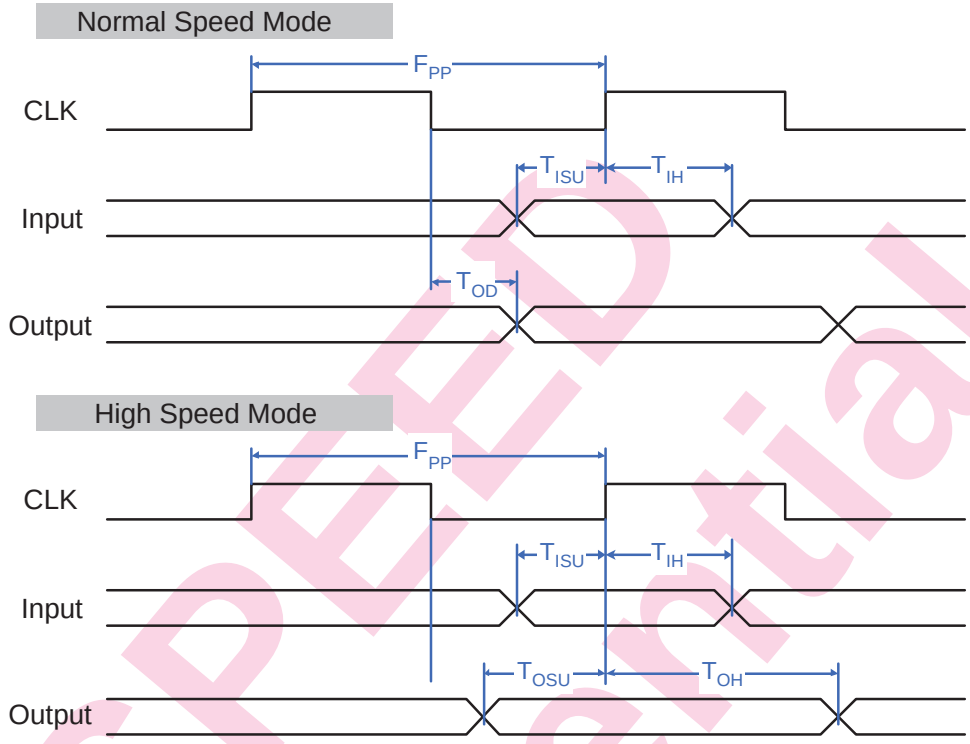


Figure 34: SD/eMMC Timing Waveform

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}	5			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}	6			ns
Input Hold time	T_{IH}	2			ns
Input Rise time	T_R			3	ns
Input Fall time	T_F			3	ns

3.10.13 SGPIO Master Interface

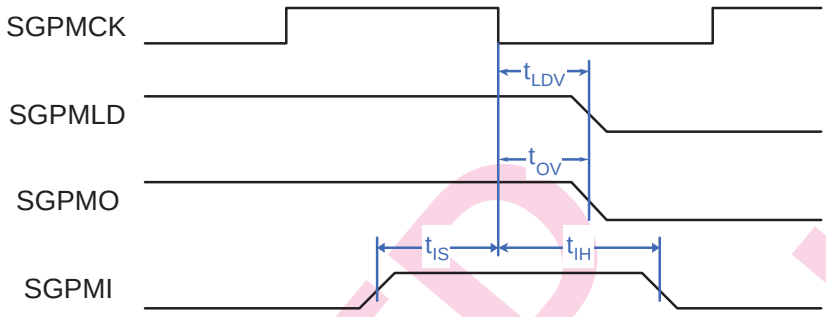


Figure 35: SGPIO Master Timing Waveform

PARAMETER	SYMBOL	MIN	TYP	MAX	UNIT
Clock frequency (programmable)	f_{CK}			20	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

3.10.14 SGPIO Slave Interface

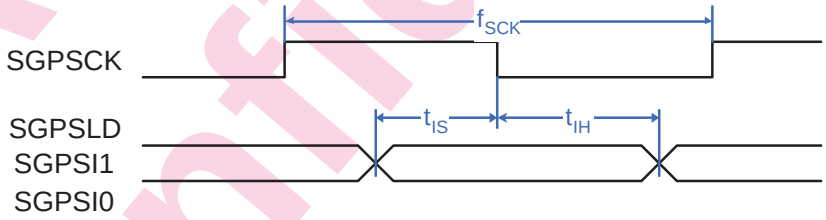


Figure 36: SGPIO Slave Timing Waveform

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

3.10.15 Strap Input Interface

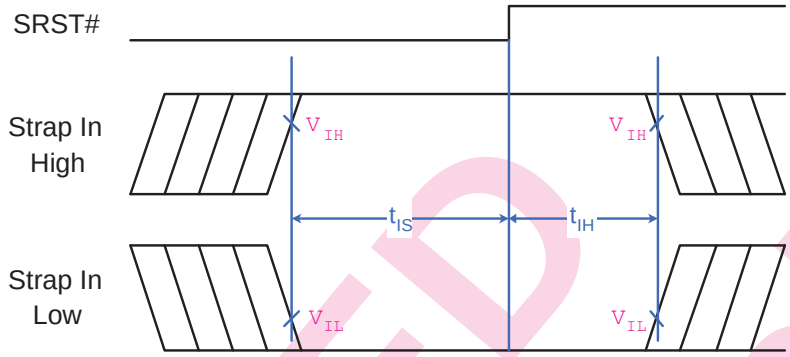


Figure 37: Strap Input Timing Waveform

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

Note :
The input voltage V_{IH} and V_{IL} of RGMII TX pins should follow the DC specification of RGMII IO at section 3.9.11 with respective voltage of R1VDD or R2VDD.
For other strap pins, it should follow 3.3V CMOS IO specification.

3.11 Thermal Specification (Simulation Result)

3.11.1 Terminology

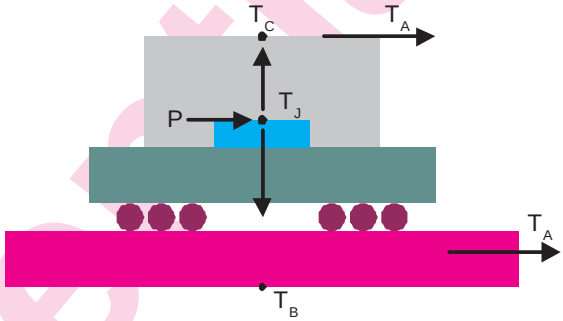
The major thermal dissipation paths can be illustrated as following:

- 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 $_{JC}$

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

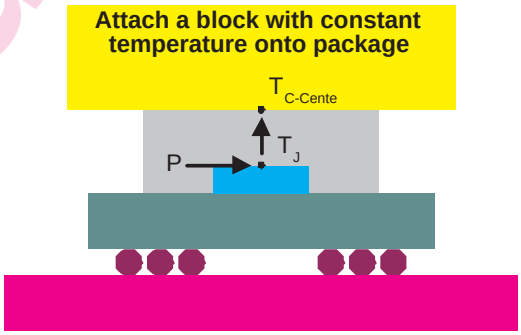


Figure 38: Thermal Terminology

3.11.2 Testing Conditions

Package Conditions	
Package Type	LFBGA Package
Ball Count	456
Package Dimension (L x W)	19 x 19 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

3.11.3 Thermal Data

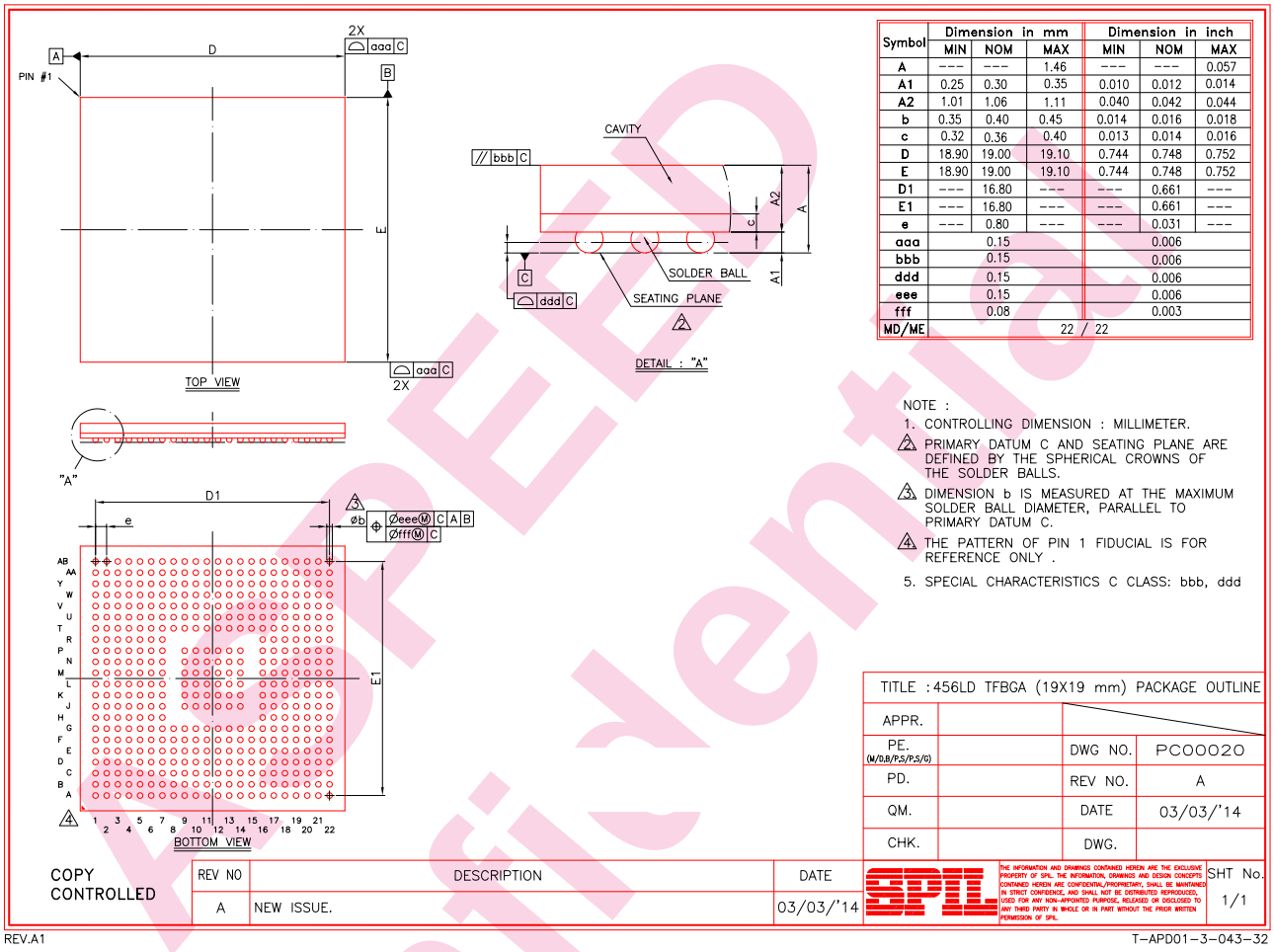
TDP		$\theta_{JA}(^oC/W)$			$\theta_{JB}(^oC/W)$	$\theta_{JC}(^oC/W)$
DDR3L	DDR4	0 m/s	1 m/s	2 m/s		
1.46W	1.40W	27.0	24.8	23.7	19.7	12.9

TDP power includes the DRAM and termination power.

3.11.4 Substrate Material Properties

Item	Material	Thermal conductivity K(W/mK)
Molding Compiund	G1250LKDS	0.9
Die	Si	147
Epoxy (Die Attached)	ATB125	0.2
Solder ball	SAC105	55
Substrate Core	CCL-HL832NX	0.8
Substrate PP	GHPL830NX	0.8
Substrate Metal	Cu	400
Solder mask	ASU308	0.2

4 Package Information



Symbol	Dimension in mm			Dimension in inch		
	MIN	NOM	MAX	MIN	NOM	MAX
A	---	---	1.46	---	---	0.057
A1	0.25	0.30	0.35	0.010	0.012	0.014
A2	1.01	1.06	1.11	0.040	0.042	0.044
b	0.35	0.40	0.45	0.014	0.016	0.018
c	0.32	0.36	0.40	0.013	0.014	0.016
D	18.90	19.00	19.10	0.744	0.748	0.752
E	18.90	19.00	19.10	0.744	0.748	0.752
D1	---	16.80	---	---	0.661	---
E1	---	16.80	---	---	0.661	---
e	---	0.80	---	---	0.031	---
aaa		0.15			0.006	
bbb		0.15			0.006	
ddd		0.15			0.006	
eee		0.15			0.006	
fff		0.08			0.003	
MD/ME		22 / 22				

Figure 39: IC Package

4.1 SMT Soldering Reflow Chart

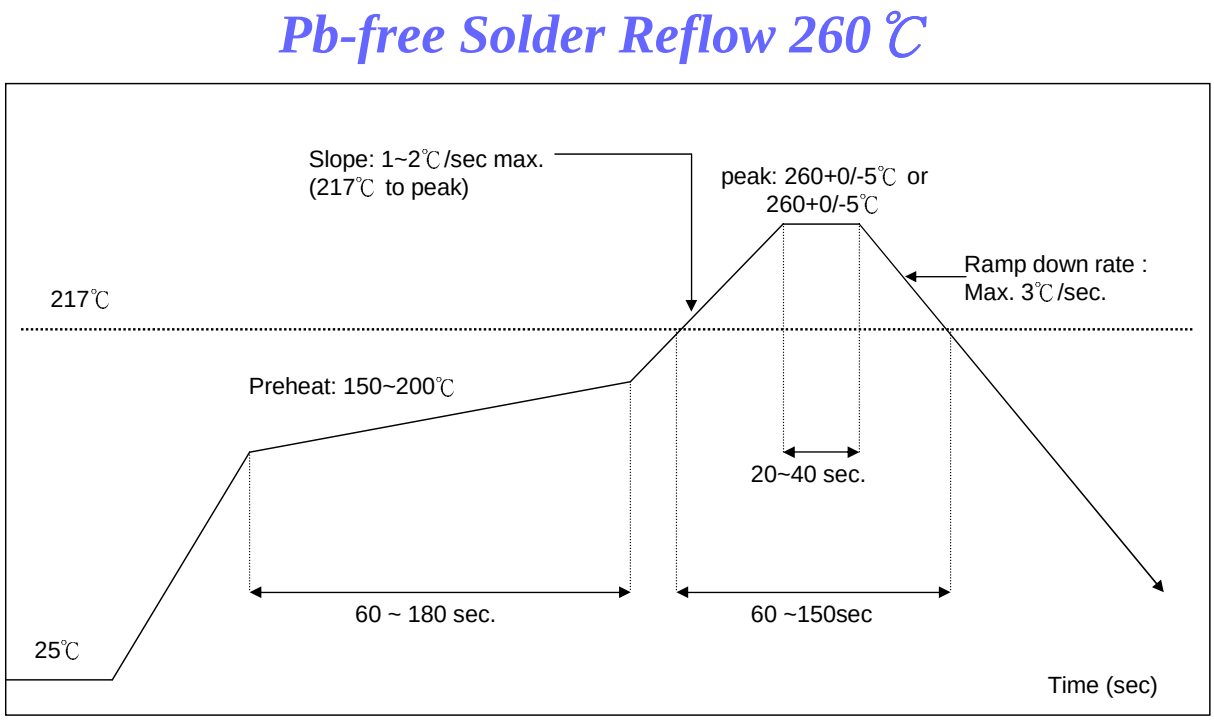


Figure 40: SMT Soldering Reflow Chart

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 4” (Lowest).

- Fan Tachometer function can only work when GPIO is in ”input mode”.
- COND1 means {SCU90[6]=0}
- COND2 means {SCU94[1:0]=0}
- COND3 means {SCU94[1:0]!=0 & GFX064[7]=1}
- COND4 means {SCU94[1:0]!=0 & GFX064[7:6]=2}
- COND5 means {SCU94[1]=1 & GFX064[7:6]=2}

5.1 Function 1-2

Ball	Default	Function 1	Control 1	Function 2	Control 2	Others
B14	GPIOA0	MAC1LINK	SCU80[0]=1			GPIOA0
D14	GPIOA1	MAC2LINK	SCU80[1]=1			GPIOA1
D13	GPIOA2	SPI1CS1#	SCU80[15]=1	TIMER3	SCU80[2]=1	GPIOA2
E13	GPIOA3			TIMER4	SCU80[3]=1	GPIOA3
C14	GPIOA4	SCL9	SCU90[22]=1 & COND1	TIMER5	SCU80[4]=1 & COND1	GPIOA4
A13	GPIOA5	SDA9	SCU90[22]=1 & COND1	TIMER6	SCU80[5]=1 & COND1	GPIOA5
C13	GPIOA6	MDC2	SCU90[2]=1 & COND1	TIMER7	SCU80[6]=1 & COND1	GPIOA6
B13	GPIOA7	MDIO2	SCU90[2]=1 & COND1	TIMER8	SCU80[7]=1 & COND1	GPIOA7
K19	GPIOB0					GPIOB0
L19	GPIOB1					GPIOB1
L18	GPIOB2					GPIOB2
K18	GPIOB3					GPIOB3
J20	GPIOB4	USBCKI	Strap[23]=1			GPIOB4
H21	GPIOB5	LPCPD#	SCU80[13]=1 & SIORD_30[1]=0	LPCSMI#	SCU80[13]=1 & SIORD_30[1]=1	GPIOB5
H22	GPIOB6	LPCPME#	SCU80[14]=1			GPIOB6
H20	GPIOB7					GPIOB7
C12	GPIOC0	SD1CLK	SCU90[0]=1	SCL10	SCU90[23]=1	GPIOC0
A12	GPIOC1	SD1CMD	SCU90[0]=1	SDA10	SCU90[23]=1	GPIOC1
B12	GPIOC2	SD1DAT0	SCU90[0]=1	SCL11	SCU90[24]=1	GPIOC2
D9	GPIOC3	SD1DAT1	SCU90[0]=1	SDA11	SCU90[24]=1	GPIOC3
D10	GPIOC4	SD1DAT2	SCU90[0]=1	SCL12	SCU90[25]=1	GPIOC4
E12	GPIOC5	SD1DAT3	SCU90[0]=1	SDA12	SCU90[25]=1	GPIOC5
C11	GPIOC6	SD1CD#	SCU90[0]=1	SCL13	SCU90[26]=1	GPIOC6
B11	GPIOC7	SD1WP#	SCU90[0]=1	SDA13	SCU90[26]=1	GPIOC7
F19	GPIOD0	SD2CLK	SCU90[1]=1	GPID0(In)	SCU8C[8]=1 ‖ Strap[21]=1	GPIOD0
E21	GPIOD1	SD2CMD	SCU90[1]=1	GPID0(Out)	SCU8C[8]=1 ‖ Strap[21]=1	GPIOD1
F20	GPIOD2	SD2DAT0	SCU90[1]=1	GPID2(In)	SCU8C[9]=1 ‖ Strap[21]=1	GPIOD2
D20	GPIOD3	SD2DAT1	SCU90[1]=1	GPID2(Out)	SCU8C[9]=1 ‖ Strap[21]=1	GPIOD3
D21	GPIOD4	SD2DAT2	SCU90[1]=1	GPID4(In)	SCU8C[10]=1 ‖ Strap[21]=1	GPIOD4

to next page

from previous page

Ball	Default	Function 1	Control 1	Function 2	Control 2	Others
E20	GPIOD5	SD2DAT3	SCU90[1]=1	GPID4(Out)	SCU8C[10]=1 || Strap[21]=1	GPIOD5
G18	GPIOD6	SD2CD#	SCU90[1]=1	GPID6(In)	SCU8C[11]=1 || Strap[21]=1	GPIOD6
C21	GPIOD7	SD2WP#	SCU90[1]=1	GPID6(Out)	SCU8C[11]=1 || Strap[21]=1	GPIOD7
B20	GPIOE0	NCTS3	SCU80[16]=1	GPIE0(In)	SCU8C[12]=1 || Strap[22]=1	GPIOE0
C20	GPIOE1	NDCD3	SCU80[17]=1	GPIE0(Out)	SCU8C[12]=1 || Strap[22]=1	GPIOE1
F18	GPIOE2	NDSR3	SCU80[18]=1	GPIE2(In)	SCU8C[13]=1 || Strap[22]=1	GPIOE2
F17	GPIOE3	NRI3	SCU80[19]=1	GPIE2(Out)	SCU8C[13]=1 || Strap[22]=1	GPIOE3
E18	GPIOE4	NDTR3	SCU80[20]=1	GPIE4(In)	SCU8C[14]=1 || Strap[22]=1	GPIOE4
D19	GPIOE5	NRTS3	SCU80[21]=1	GPIE4(Out)	SCU8C[14]=1 || Strap[22]=1	GPIOE5
A20	GPIOE6	TXD3	SCU80[22]=1	GPIE6(In)	SCU8C[15]=1 || Strap[22]=1	GPIOE6
B19	GPIOE7	RXD3	SCU80[23]=1	GPIE6(Out)	SCU8C[15]=1 || Strap[22]=1	GPIOE7
J19	GPIOF0	LHAD0	LHCR0[0]=1 || SCU90[30]=1	NCTS4	SCU80[24]=1	GPIOF0
J18	GPIOF1	LHAD1	LHCR0[0]=1 || SCU90[30]=1	NDCD4	SCU80[25]=1	GPIOF1
B22	GPIOF2	LHAD2	LHCR0[0]=1 || SCU90[30]=1	NDSR4	SCU80[26]=1	GPIOF2
B21	GPIOF3	LHAD3	LHCR0[0]=1 || SCU90[30]=1	NRI4	SCU80[27]=1	GPIOF3
A21	GPIOF4	LHCLK	LHCR0[0]=1 || SCU90[30]=1	NDTR4	SCU80[28]=1	GPIOF4
H19	GPIOF5	LHFRAME#	LHCR0[0]=1 || SCU90[30]=1	NRTS4	SCU80[29]=1	GPIOF5
G17	GPIOF6	LHSIRQ#	LHCR0[0]=1	TXD4	SCU80[30]=1	GPIOF6
H18	GPIOF7	LHRST#	LHCR0[0]=1 || SCU90[30]=1	RXD4	SCU80[31]=1	GPIOF7
A19	GPIOG0	SGPS1CK	COND1 & SCU84[0]=1			GPIOG0
E19	GPIOG1	SGPS1LD	COND1 & SCU84[1]=1			GPIOG1
C19	GPIOG2	SGPS1I0	COND1 & SCU84[2]=1			GPIOG2
E16	GPIOG3	SGPS1I1	COND1 & SCU84[3]=1			GPIOG3
E17	GPIOG4	SGPS2CK	COND1 & SCU94[12]=1	SALT1	COND1 & SCU84[4]=1	GPIOG4
D16	GPIOG5	SGPS2LD	COND1 & SCU94[12]=1	SALT2	COND1 & SCU84[5]=1	GPIOG5
D15	GPIOG6	SGPS2I0	COND1 & SCU94[12]=1	SALT3	COND1 & SCU84[6]=1	GPIOG6
E14	GPIOG7	SGPS2I1	COND1 & SCU94[12]=1	SALT4	COND1 & SCU84[7]=1	GPIOG7
A18	GPIOH0	–	COND1 & SCU94[5]=1	NCTS6	COND1 & SCU90[7]=1	GPIOH0

to next page

from previous page

Ball	Default	Function 1	Control 1	Function 2	Control 2	Others
B18	GPIOH1	–	COND1 & SCU94[5]=1	NDCD6	COND1 & SCU90[7]=1	GPIOH1
D17	GPIOH2	–	COND1 & SCU94[6]=1	NDSR6	COND1 & SCU90[7]=1	GPIOH2
C17	GPIOH3	–	COND1 & SCU94[6]=1	NRI6	COND1 & SCU90[7]=1	GPIOH3
A17	GPIOH4	–	COND1 & SCU94[7]=1	NDTR6	COND1 & SCU90[7]=1	GPIOH4
B17	GPIOH5	–	COND1 & SCU94[7]=1	NRTS6	COND1 & SCU90[7]=1	GPIOH5
A16	GPIOH6	TXD6	COND1 & SCU90[7]=1			GPIOH6
D18	GPIOH7	RXD6	COND1 & SCU90[7]=1			GPIOH7
C18	GPIOI0	SYSCS#	COND1 & Strap[13]=1			GPIOI0
E15	GPIOI1	SYSCK	COND1 & Strap[13]=1			GPIOI1
B16	GPIOI2	SYSMOSI	COND1 & Strap[13]=1			GPIOI2
C16	GPIOI3	SYSMISO	COND1 & Strap[13]=1			GPIOI3
B15	GPIOI4	SPI1CS0#	COND1 & Strap[13:12]!=0	VBCS#	COND1 & Strap[5]=1	GPIOI4
C15	GPIOI5	SPI1CK	COND1 & Strap[13:12]!=0	VBCK	COND1 & Strap[5]=1	GPIOI5
A14	GPIOI6	SPI1MOSI	COND1 & Strap[13:12]!=0	VBMOSI	COND1 & Strap[5]=1	GPIOI6
A15	GPIOI7	SPI1MISO	COND1 & Strap[13:12]!=0	VBMISO	COND1 & Strap[5]=1	GPIOI7
R2	GPIOJ0	SGPMCK	SCU84[8]=1			GPIOJ0
L2	GPIOJ1	SGPMLD	SCU84[9]=1			GPIOJ1
N3	GPIOJ2	SGPMO	SCU84[10]=1			GPIOJ2
N4	GPIOJ3	SGPMI	SCU84[11]=1			GPIOJ3
N5	VGAHS	VGAHS	SCU84[12]=1	–	SCU94[8]=1	GPIOJ4
R4	VGAVS	VGAVS	SCU84[13]=1	–	SCU94[8]=1	GPIOJ5
R3	DDCCLK	DDCCLK	SCU84[14]=1	–	SCU94[9]=1	GPIOJ6
T3	DDCDAT	DDCDAT	SCU84[15]=1	–	SCU94[9]=1	GPIOJ7
L3	GPIOK0	SCL5	SCU90[18]=1			GPIOK0
L4	GPIOK1	SDA5	SCU90[18]=1			GPIOK1
L1	GPIOK2	SCL6	SCU90[19]=1			GPIOK2
N2	GPIOK3	SDA6	SCU90[19]=1			GPIOK3
N1	GPIOK4	SCL7	SCU90[20]=1			GPIOK4

to next page

from previous page

Ball	Default	Function 1	Control 1	Function 2	Control 2	Others
P1	GPIOK5	SDA7	SCU90[20]=1			GPIOK5
P2	GPIOK6	SCL8	SCU90[21]=1			GPIOK6
R1	GPIOK7	SDA8	SCU90[21]=1			GPIOK7
T2	GPIOL0			NCTS1	SCU84[16]=1	GPIOL0
T1	GPIOL1	VPIDE	SCU90[5]=1 & SCU84[17]=1 & COND2	NDCD1	SCU84[17]=1 & COND2	GPIOL1
U1	GPIOL2	–	SCU90[5]=1 & SCU84[18]=1	NDSR1	SCU84[18]=1	GPIOL2
U2	GPIOL3	VPIHS	SCU90[5]=1 & SCU84[19]=1 & COND2	NRI1	SCU84[19]=1 & COND2	GPIOL3
P4	GPIOL4	VPIVS	SCU90[5]=1 & SCU84[20]=1 & COND2	NDTR1	SCU84[20]=1 & COND2	GPIOL4
P3	GPIOL5	VPICLK	SCU90[5]=1 & SCU84[21]=1 & COND2	NRTS1	SCU84[21]=1 & COND2	GPIOL5
V1	GPIOL6	–	SCU90[5:4]=3 & SCU84[22]=1	TXD1	SCU84[22]=1 & COND2	GPIOL6
W1	GPIOL7	–	SCU90[5:4]=3 & SCU84[23]=1	RXD1	SCU84[23]=1 & COND2	GPIOL7
Y1	GPIOM0	VPIB2	SCU90[5]=1 & SCU84[24]=1 & COND2	NCTS2	SCU84[24]=1 & COND2	GPIOM0
AB2	GPIOM1	VPIB3	SCU90[5]=1 & SCU84[25]=1 & COND2	NDCD2	SCU84[25]=1 & COND2	GPIOM1
AA1	GPIOM2	VPIB4	SCU90[5]=1 & SCU84[26]=1 & COND2	NDSR2	SCU84[26]=1 & COND2	GPIOM2
Y2	GPIOM3	VPIB5	SCU90[5]=1 & SCU84[27]=1 & COND2	NRI2	SCU84[27]=1 & COND2	GPIOM3
AA2	GPIOM4	VPIB6	SCU90[5]=1 & SCU84[28]=1 & COND2	NDTR2	SCU84[28]=1 & COND2	GPIOM4
P5	GPIOM5	VPIB7	SCU90[5]=1 & SCU84[29]=1 & COND2	NRTS2	SCU84[29]=1 & COND2	GPIOM5
R5	GPIOM6	VPIB8	SCU90[5]=1 & SCU84[30]=1 & COND2	TXD2	SCU84[30]=1 & COND2	GPIOM6
T5	GPIOM7	VPIB9	SCU90[5]=1 & SCU84[31]=1 & COND2	RXD2	SCU84[31]=1 & COND2	GPIOM7

to next page

from previous page

Ball	Default	Function 1	Control 1	Function 2	Control 2	Others
V2	GPION0	–	SCU90[5:4]=3 & SCU88[0]=1	PWM0	SCU88[0]=1 & COND2	GPION0
W2	GPION1	–	SCU90[5:4]=3 & SCU88[1]=1	PWM1	SCU88[1]=1 & COND2	GPION1
V3	GPION2	VPIG2	SCU90[5]=1 & SCU88[2]=1 & COND2	PWM2	SCU88[2]=1 & COND2	GPION2
U3	GPION3	VPIG3	SCU90[5]=1 & SCU88[3]=1 & COND2	PWM3	SCU88[3]=1 & COND2	GPION3
W3	GPION4	VPIG4	SCU90[5]=1 & SCU88[4]=1 & COND2	PWM4	SCU88[4]=1 & COND2	GPION4
AA3	GPION5	VPIG5	SCU90[5]=1 & SCU88[5]=1 & COND2	PWM5	SCU88[5]=1 & COND2	GPION5
Y3	GPION6	VPIG6	SCU90[5:4]=2 & SCU88[6]=1 & COND2	PWM6	SCU88[6]=1 & COND2	GPION6
T4	GPION7	VPIG7	SCU90[5:4]=2 & SCU88[7]=1 & COND2	PWM7	SCU88[7]=1 & COND2	GPION7
U5	GPIOO0/TACH0	VPIG8	SCU90[5:4]=2 & SCU88[8]=1 & COND2			GPIOO0/TACH0
U4	GPIOO1/TACH1	VPIG9	SCU90[5:4]=2 & SCU88[9]=1 & COND2			GPIOO1/TACH1
V5	GPIOO2/TACH2	–	SCU90[5:4]=3 & SCU88[10]=1			GPIOO2/TACH2
AB4	GPIOO3/TACH3	–	SCU90[5:4]=3 & SCU88[11]=1			GPIOO3/TACH3
AB3	GPIOO4/TACH4	VPIR2	SCU90[5:4]=2 & SCU88[12]=1 & COND2			GPIOO4/TACH4
Y4	GPIOO5/TACH5	VPIR3	SCU90[5:4]=2 & SCU88[13]=1 & COND2			GPIOO5/TACH5
AA4	GPIOO6/TACH6	VPIR4	SCU90[5:4]=2 & SCU88[14]=1 & COND2			GPIOO6/TACH6
W4	GPIOO7/TACH7	VPIR5	SCU90[5:4]=2 & SCU88[15]=1 & COND2			GPIOO7/TACH7
V4	GPIOP0/TACH8	VPIR6	SCU90[5:4]=2 & SCU88[16]=1 & COND2			GPIOP0/TACH8

to next page

from previous page

Ball	Default	Function 1	Control 1	Function 2	Control 2	Others
W5	GPIOP1/TACH9	VPIR7	SCU90[5:4]=2 & SCU88[17]=1 & COND2			GPIOP1/TACH9
AA5	GPIOP2/TACH10	VPIR8	SCU90[5:4]=2 & SCU88[18]=1 & COND2			GPIOP2/TACH10
AB5	GPIOP3/TACH11	VPIR9	SCU90[5:4]=2 & SCU88[19]=1 & COND2			GPIOP3/TACH11
Y6	GPIOP4/TACH12	–	SCU90[28]=1 & SCU88[20]=1			GPIOP6/TACH14
Y5	GPIOP5/TACH13	–	SCU90[28]=1 & SCU88[21]=1			GPIOP6/TACH14
W6	GPIOP6/TACH14	–	SCU90[28]=1 & SCU88[22]=1			GPIOP6/TACH14
V6	GPIOP7/TACH15	–	SCU90[28]=1 & SCU88[23]=1			GPIOP7/TACH15
A11	GPIOQ0	SCL3	SCU90[16]=1			GPIOQ0
A10	GPIOQ1	SDA3	SCU90[16]=1			GPIOQ1
A9	GPIOQ2	SCL4	SCU90[17]=1			GPIOQ2
B9	GPIOQ3	SDA4	SCU90[17]=1			GPIOQ3
N21	GPIOQ4	SCL14	SCU90[27]=1			GPIOQ4
N22	GPIOQ5	SDA14	SCU90[27]=1			GPIOQ5
B10	GPIOQ6	OSCCLK	SCU2C[1]=1			GPIOQ6
N20	GPIOQ7	PEWAKE#	SCU2C[29]=1			GPIOQ7
AA19	FWSPICS1#	FWSPICS1#	SCU88[24]=1 & COND2			GPIOR0
T19	FWSPICS2#	FWSPICS2#	SCU88[25]=1 & COND2			GPIOR1
T17	GPIOR2	SPI2CS0#	SCU88[26]=1 & COND2			GPIOR2
Y19	GPIOR3	SPI2CK	SCU88[27]=1 & COND2			GPIOR3
W19	GPIOR4	SPI2MOSI	SCU88[28]=1 & COND2			GPIOR4
V19	GPIOR5	SPI2MISO	SCU88[29]=1 & COND2			GPIOR5
D8	GPIOR6	MDC1	SCU88[30]=1			GPIOR6
E10	GPIOR7	MDIO1	SCU88[31]=1			GPIOR7
V20	GPIOS0	VPOB2	SCU8C[0]=1 & COND3	SPI2CS1#	SCU8C[0]=1	GPIOS0
U19	GPIOS1	VPOB3	SCU8C[1]=1 & COND3	BMCINT	SCU8C[1]=1	GPIOS1
R18	GPIOS2	VPOB4	SCU8C[2]=1 & COND3	SALT5	SCU8C[2]=1	GPIOS2

to next page

from previous page

Ball	Default	Function 1	Control 1	Function 2	Control 2	Others
P18	GPIOS3	VPOB5	SCU8C[3]=1 & COND3	SALT6	SCU8C[3]=1	GPIOS3
R19	GPIOS4	VPOB6	SCU8C[4]=1 & COND3			GPIOS4
W20	GPIOS5	VPOB7	SCU8C[5]=1 & COND3			GPIOS5
U20	GPIOS6	VPOB8	SCU8C[6]=1 & COND3			GPIOS6
AA20	GPIOS7	VPOB9	SCU8C[7]=1 & COND3			GPIOS7
B5	RGMII1TXCK	GPIOT0	SCUA0[0]=1	RMII1RCLKO	Strap[6]=0 & SCU48[29]=1	RGMII1TXCK
E9	RGMII1TXCTL	GPIOT1	SCUA0[1]=1	RMII1TXEN	Strap[6]=0	RGMII1TXCTL
F9	RGMII1TXD0	GPIOT2	SCUA0[2]=1	RMII1TXD0	Strap[6]=0	RGMII1TXD0
A5	RGMII1TXD1	GPIOT3	SCUA0[3]=1	RMII1TXD1	Strap[6]=0	RGMII1TXD1
E7	RGMII1TXD2	GPIOT4	SCUA0[4]=1	–	Strap[6]=0	RGMII1TXD2
D7	RGMII1TXD3	GPIOT5	SCUA0[5]=1	–	Strap[6]=0	RGMII1TXD3
B2	RGMII2TXCK	GPIOT6	SCUA0[6]=1	RMII2RCLKO	Strap[7]=0 & SCU48[30]=1	RGMII2TXCK
B1	RGMII2TXCTL	GPIOT7	SCUA0[7]=1	RMII2TXEN	Strap[7]=0	RGMII2TXCTL
A2	RGMII2TXD0	GPIOU0	SCUA0[8]=1	RMII2TXD0	Strap[7]=0	RGMII2TXD0
B3	RGMII2TXD1	GPIOU1	SCUA0[9]=1	RMII2TXD1	Strap[7]=0	RGMII2TXD1
D5	RGMII2TXD2	GPIOU2	SCUA0[10]=1	–	Strap[7]=0	RGMII2TXD2
D4	RGMII2TXD3	GPIOU3	SCUA0[11]=1	–	Strap[7]=0	RGMII2TXD3
B4	RGMII1RXCK	GPIOU4	SCUA0[12]=1	RMII1RCLKI	Strap[6]=0	RGMII1RXCK
A4	RGMII1RXCTL	GPIOU5	SCUA0[13]=1	–	Strap[6]=0	RGMII1RXCTL
A3	RGMII1RXD0	GPIOU6	SCUA0[14]=1	RMII1RXD0	Strap[6]=0	RGMII1RXD0
D6	RGMII1RXD1	GPIOU7	SCUA0[15]=1	RMII1RXD1	Strap[6]=0	RGMII1RXD1
C5	RGMII1RXD2	GPIOV0	SCUA0[16]=1	RMII1CRSDV	Strap[6]=0	RGMII1RXD2
C4	RGMII1RXD3	GPIOV1	SCUA0[17]=1	RMII1RXER	Strap[6]=0	RGMII1RXD3
C2	RGMII2RXCK	GPIOV2	SCUA0[18]=1	RMII2RCLKI	Strap[7]=0	RGMII2RXCK
C1	RGMII2RXCTL	GPIOV3	SCUA0[19]=1	–	Strap[7]=0	RGMII2RXCTL
C3	RGMII2RXD0	GPIOV4	SCUA0[20]=1	RMII2RXD0	Strap[7]=0	RGMII2RXD0
D1	RGMII2RXD1	GPIOV5	SCUA0[21]=1	RMII2RXD1	Strap[7]=0	RGMII2RXD1
D2	RGMII2RXD2	GPIOV6	SCUA0[22]=1	RMII2CRSDV	Strap[7]=0	RGMII2RXD2

to next page

from previous page

Ball	Default	Function 1	Control 1	Function 2	Control 2	Others
E6	RGMII2RXD3	GPIOV7	SCUA0[23]=1	RMII2RXER	Strap[7]=0	RGMII2RXD3
F4	ADC0	GPIW0	SCUA0[24]=1			ADC0
F5	ADC1	GPIW1	SCUA0[25]=1			ADC1
E2	ADC2	GPIW2	SCUA0[26]=1			ADC2
E1	ADC3	GPIW3	SCUA0[27]=1			ADC3
F3	ADC4	GPIW4	SCUA0[28]=1			ADC4
E3	ADC5	GPIW5	SCUA0[29]=1			ADC5
G5	ADC6	GPIW6	SCUA0[30]=1			ADC6
G4	ADC7	GPIW7	SCUA0[31]=1			ADC7
F2	ADC8	GPIX0	SCUA4[0]=1			ADC8
G3	ADC9	GPIX1	SCUA4[1]=1			ADC9
G2	ADC10	GPIX2	SCUA4[2]=1			ADC10
F1	ADC11	GPIX3	SCUA4[3]=1			ADC11
H5	ADC12	GPIX4	SCUA4[4]=1			ADC12
G1	ADC13	GPIX5	SCUA4[5]=1			ADC13
H3	ADC14	GPIX6	SCUA4[6]=1			ADC14
H4	ADC15	GPIX7	SCUA4[7]=1			ADC15
R22	GPIOY0	SIOS3#	SCUA4[8]=1 ‖ Strap[19]=1	–	SCU94[10]=1	GPIOY0
R21	GPIOY1	SIOS5#	SCUA4[9]=1 ‖ Strap[19]=1	–	SCU94[10]=1	GPIOY1
P22	GPIOY2	SIOPWREQ#	SCUA4[10]=1 ‖ Strap[19]=1	–	SCU94[11]=1	GPIOY2
P21	GPIOY3	SIOONCTRL#	SCUA4[11]=1 ‖ Strap[19]=1	–	SCU94[11]=1	GPIOY3
M18	SCL1	SCL1	SCUA4[12]=1			GPIOY4
M19	SDA1	SDA1	SCUA4[13]=1			GPIOY5
M20	SCL2	SCL2	SCUA4[14]=1			GPIOY6
P20	SDA2	SDA2	SCUA4[15]=1			GPIOY7
Y20	GPIOZ0	VPOG2	SCUA4[16]=1 & COND3	SIOPBI#	SCUA4[16]=1 ‖ Strap[19]=1	GPIOZ0
AB20	GPIOZ1	VPOG3	SCUA4[17]=1 & COND3	SIOPWRGD	SCUA4[17]=1 ‖ Strap[19]=1	GPIOZ1
AB21	GPIOZ2	VPOG4	SCUA4[18]=1 & COND3	SIOPBO#	SCUA4[18]=1 ‖ Strap[19]=1	GPIOZ2

to next page

from previous page

Ball	Default	Function 1	Control 1	Function 2	Control 2	Others
AA21	GPIOZ3	VPOG5	SCUA4[19]=1 & COND3	SIOSCI#	SCUA4[19]=1 ‖ Strap[19]=1	GPIOZ3
U21	GPIOZ4	VPOG6	SCUA4[20]=1 & COND4			GPIOZ4
W22	GPIOZ5	VPOG7	SCUA4[21]=1 & COND4			GPIOZ5
V22	GPIOZ6	VPOG8	SCUA4[22]=1 & COND4			GPIOZ6
W21	GPIOZ7	VPOG9	SCUA4[23]=1 & COND4			GPIOZ7
Y21	GPIOAA0	VPOR2	SCUA4[24]=1 & COND4	SALT7	SCUA4[24]=1	GPIOAA0
V21	GPIOAA1	VPOR3	SCUA4[25]=1 & COND4	SALT8	SCUA4[25]=1	GPIOAA1
Y22	GPIOAA2	VPOR4	SCUA4[26]=1 & COND4	SALT9	SCUA4[26]=1	GPIOAA2
AA22	GPIOAA3	VPOR5	SCUA4[27]=1 & COND4	SALT10	SCUA4[27]=1	GPIOAA3
U22	GPIOAA4	VPOR6	SCUA4[28]=1 & COND4	SALT11	SCUA4[28]=1	GPIOAA4
T20	GPIOAA5	VPOR7	SCUA4[29]=1 & COND4	SALT12	SCUA4[29]=1	GPIOAA5
N18	GPIOAA6	VPOR8	SCUA4[30]=1 & COND4	SALT13	SCUA4[30]=1	GPIOAA6
P19	GPIOAA7	VPOR9	SCUA4[31]=1 & COND4	SALT14	SCUA4[31]=1	GPIOAA7
N19	GPIOAB0	VPODE	SCUA8[0]=1 & COND3			GPIOAB0
T21	GPIOAB1	VPOHS	SCUA8[1]=1 & COND3			GPIOAB1
T22	GPIOAB2	VPOVS	SCUA8[2]=1 & COND3	WDTRST1	SCUA8[2]=1	GPIOAB2
R20	GPIOAB3	VPOCLK	SCUA8[3]=1 & COND3	WDTRST2	SCUA8[3]=1	GPIOAB3
G21	LAD0	ESPID0	Strap[25]=1	LAD0	SCUAC[0]=1	GPIOAC0
G20	LAD1	ESPID1	Strap[25]=1	LAD1	SCUAC[1]=1	GPIOAC1
D22	LAD2	ESPID2	Strap[25]=1	LAD2	SCUAC[2]=1	GPIOAC2
E22	LAD3	ESPID3	Strap[25]=1	LAD3	SCUAC[3]=1	GPIOAC3
C22	LCLK	ESPICK	Strap[25]=1	LCLK	SCUAC[4]=1	GPIOAC4
F21	LFRAME#	ESPICS	Strap[25]=1	LFRAME#	SCUAC[5]=1	GPIOAC5
F22	LSIRQ#	ESPIALT	Strap[25]=1	LSIRQ#	SCUAC[6]=1	GPIOAC6
G22	LPCRST#	ESPIRST#	Strap[25]=1	LPCRST#	SCUAC[7]=1	GPIOAC7

5.2 Function 3-4

Ball	Default	Function 3	Control 3	Function 4	Control 4	Others
Y20	GPIOZ0	NORA0	SCU90[31]=1			GPIOZ0
AB20	GPIOZ1	NORA1	SCU90[31]=1			GPIOZ1
AB21	GPIOZ2	NORA2	SCU90[31]=1			GPIOZ2
AA21	GPIOZ3	NORA3	SCU90[31]=1			GPIOZ3
U21	GPIOZ4	NORA4	SCU90[31]=1			GPIOZ4
W22	GPIOZ5	NORA5	SCU90[31]=1			GPIOZ5
V22	GPIOZ6	NORA6	SCU90[31]=1			GPIOZ6
W21	GPIOZ7	NORA7	SCU90[31]=1			GPIOZ7
Y21	GPIOAA0	NORD0	SCU90[31]=1			GPIOAA0
V21	GPIOAA1	NORD1	SCU90[31]=1			GPIOAA1
Y22	GPIOAA2	NORD2	SCU90[31]=1			GPIOAA2
AA22	GPIOAA3	NORD3	SCU90[31]=1			GPIOAA3
U22	GPIOAA4	NORD4	SCU90[31]=1			GPIOAA4
T20	GPIOAA5	NORD5	SCU90[31]=1			GPIOAA5
N18	GPIOAA6	NORD6	SCU90[31]=1			GPIOAA6
P19	GPIOAA7	NORD7	SCU90[31]=1			GPIOAA7
N19	GPIOAB0	NOROE#	SCU90[31]=1			GPIOAB0
T21	GPIOAB1	NORWE#	SCU90[31]=1			GPIOAB1

5.3 Pin Function Configuration

LPC Slave Interface
LPC Master Interface
SuperIO ACPI Interface
eSPI Interface
SPI Interface
NOR Interface
SD/SDIO/eMMC Interface
RGMII1 Interface
RGMII2 Interface
RMII1 Interface
RMII2 Interface
UART Interface
VGA Interface
Digital Video Output Interface
Digital Video Input Interface
I2C Interface
PWM Interface
Tachometer Interface
ADC Interface
SGPIO Master Interface
SGPIO Slave Interface
GPIO Interface
Misc. Interface

Ball	Name	I/O	Type	Power-Up	Multi-function Control
LPC Slave Interface					
G21	LAD0	IUS/O12	Bidir	Hi-Z,Input	Strap[25]=0 & SCUAC[0]=1
G20	LAD1	IUS/O12	Bidir	Hi-Z,Input	Strap[25]=0 & SCUAC[1]=1
D22	LAD2	IUS/O12	Bidir	Hi-Z,Input	Strap[25]=0 & SCUAC[2]=1
E22	LAD3	IUS/O12	Bidir	Hi-Z,Input	Strap[25]=0 & SCUAC[3]=1
C22	LCLK	IDS/O12	In	Input	Strap[25]=0 & SCUAC[4]=1
F21	LFRAME#	IUS/O12	In	Input	Strap[25]=0 & SCUAC[5]=1
F22	LSIRQ	IUS/O12	Bidir	Hi-Z,Input	Strap[25]=0 & SCUAC[6]=1
G22	LPCRST#	IDS/O12	In	Input	Strap[25]=0 & SCUAC[7]=1
H21	LPCPD#	ID/O8	In	Hi-Z,Input	SCU80[13]=1 & SIORD_30[1]=0
H21	LPCSMI#	ID/O8	Bidir	Hi-Z,Input	SCU80[13]=1 & SIORD_30[1]=1
H22	LPCPME#	ID/O8	Out	Hi-Z,Input	SCU80[14]=1
LPC Master Interface					
J19	LHAD0	ID/O8	Bidir	Hi-Z,Input	LHCR0[0]=1 & SCU90[30]=0
J18	LHAD1	ID/O8	Bidir	Hi-Z,Input	LHCR0[0]=1 & SCU90[30]=0
B22	LHAD2	ID/O8	Bidir	Hi-Z,Input	LHCR0[0]=1 & SCU90[30]=0
B21	LHAD3	ID/O8	Bidir	Hi-Z,Input	LHCR0[0]=1 & SCU90[30]=0
A21	LHCLK	ID/O8	Bidir	Hi-Z,Input	LHCR0[0]=1 & SCU90[30]=0
H19	LHFRAME#	ID/O8	Out	Hi-Z,Input	LHCR0[0]=1 & SCU90[30]=0
G17	LHSIRQ#	ID/O8	Bidir	Hi-Z,Input	LHCR0[0]=1
H18	LHRST#	ID/O8	Bidir	Hi-Z,Input	LHCR0[0]=1 & SCU90[30]=0
SuperIO ACPI Interface					
R22	SIOS3#	ID/O8	In	Hi-Z,Input	SCUA4[8]=1 ‖ Strap[19]=1

Ball	Name	I/O	Type	Power-Up	Multi-function Control
R21	SIOS5#	ID/O8	In	Hi-Z,Input	SCUA4[9]=1 ‖ Strap[19]=1
P22	SIOPWREQ#	ID/O8	Out	Hi-Z	SCUA4[10]=1 ‖ Strap[19]=1
P21	SIOONCTRL#	ID/O8	Bidir	Hi-Z,Input	(SCUA4[11]=1 ‖ Strap[19]=1) & SCU94[1:0]=0
Y20	SIOPBI#	ID/O8	In	Hi-Z,Input	(SCUA4[16]=1 ‖ Strap[19]=1) & SCU94[1:0]=0
AB20	SIOPWRGD	ID/O8	In	Hi-Z,Input	(SCUA4[17]=1 ‖ Strap[19]=1) & SCU94[1:0]=0
AB21	SIOPBO#	ID/O8	Bidir	Hi-Z,Input	(SCUA4[18]=1 ‖ Strap[19]=1) & SCU94[1:0]=0
AA21	SIOSCI#	ID/O8	Out	Hi-Z	(SCUA4[19]=1 ‖ Strap[19]=1) & SCU94[1:0]=0
eSPI Interface					
G21	ESPID0	IUS/O12	Bidir	Hi-Z,Input	Strap[25]=1
G20	ESPID1	IUS/O12	Bidir	Hi-Z,Input	Strap[25]=1
D22	ESPID2	IUS/O12	Bidir	Hi-Z,Input	Strap[25]=1
E22	ESPID3	IUS/O12	Bidir	Hi-Z,Input	Strap[25]=1
C22	ESPICK	IDS/O12	In	Input	Strap[25]=1
F21	ESPICS#	IUS/O12	In	Input	Strap[25]=1
F22	ESPIALT#	IUS/O12	Bidir	Hi-Z,Input	Strap[25]=1
G22	ESPIRST#	IDS/O12	In	Input	Strap[25]=1
SPI Interface					
AA19	FWSPICS1#	ID/O8	Out	Hi-Z	SCU88[24]=1 & SCU94[1:0]=0
T19	FWSPICS2#	ID/O8	Out	Hi-Z	SCU88[25]=1 & SCU94[1:0]=0
B15	SPI1CS0#	ID/O8	Out	Hi-Z	SCU90[6]=0 & Strap[13:12]="01"
D13	SPI1CS1#	ID/O16	Out	Hi-Z	SCU80[15]=1
C15	SPI1CK	ID/O8	Out	Hi-Z	SCU90[6]=0 & Strap[13:12]="01"
A14	SPI1MOSI	ID/O8	Bidir	Hi-Z	SCU90[6]=0 & Strap[13:12]="01"
A15	SPI1MISO	ID/O8	Bidir	Hi-Z	SCU90[6]=0 & Strap[13:12]="01"
B15	VBCS#	ID/O8	Out	Hi-Z	SCU90[6]=0 & Strap[13:12,5]="001"
C15	VBCK	ID/O8	Out	Hi-Z	SCU90[6]=0 & Strap[13:12,5]="001"
A14	VBMOSI	ID/O8	Out	Hi-Z	SCU90[6]=0 & Strap[13:12,5]="001"
A15	VBMISO	ID/O8	In	Hi-Z	SCU90[6]=0 & Strap[13:12,5]="001"
C18	SYSCS#	ID/O8	In	Hi-Z	SCU90[6]=0 & Strap[13:12]="11"
E15	SYSCK	ID/O8	In	Hi-Z	SCU90[6]=0 & Strap[13:12]="11"
B16	SYSMOSI	ID/O8	In	Hi-Z	SCU90[6]=0 & Strap[13:12]="11"
C16	SYSMISO	ID/O8	Out	Hi-Z	SCU90[6]=0 & Strap[13:12]="11"
T17	SPI2CS0#	ID/O8	Out	Hi-Z	SCU94[1:0]=0 & SCU88[26]=1
V20	SPI2CS1#	ID/O8	Out	Hi-Z	SCU94[1:0]=0 & SCU8C[0]=1
Y19	SPI2CK	ID/O8	Out	Hi-Z	SCU94[1:0]=0 & SCU88[27]=1
W19	SPI2MOSI	ID/O8	Bidir	Hi-Z	SCU94[1:0]=0 & SCU88[28]=1
V19	SPI2MISO	ID/O8	Bidir	Hi-Z	SCU94[1:0]=0 & SCU88[29]=1
NOR Interface					
Y20	NORA0	ID/O8	Out	Hi-Z	SCU90[31]=1 & SCU94[1:0]=0 & SCUA4[16]=0 & Strap[19]=0
AB20	NORA1	ID/O8	Out	Hi-Z	SCU90[31]=1 & SCU94[1:0]=0 & SCUA4[17]=0 & Strap[19]=0

Ball	Name	I/O	Type	Power-Up	Multi-function Control
AB21	NORA2	ID/O8	Out	Hi-Z	SCU90[31]=1 & SCU94[1:0]=0 & SCUA4[18]=0 & Strap[19]=0
AA21	NORA3	ID/O8	Out	Hi-Z	SCU90[31]=1 & SCU94[1:0]=0 & SCUA4[19]=0 & Strap[19]=0
U21	NORA4	ID/O8	Out	Hi-Z	SCU90[31]=1 & SCU94[1:0]=0
W22	NORA5	ID/O8	Out	Hi-Z	SCU90[31]=1 & SCU94[1:0]=0
V22	NORA6	ID/O8	Out	Hi-Z	SCU90[31]=1 & SCU94[1:0]=0
W21	NORA7	ID/O8	Out	Hi-Z	SCU90[31]=1 & SCU94[1:0]=0
Y21	NORD0	ID/O8	Bidir	Hi-Z	SCU90[31]=1 & SCU94[1:0]=0 & SCUA4[24]=0
V21	NORD1	ID/O8	Bidir	Hi-Z	SCU90[31]=1 & SCU94[1:0]=0 & SCUA4[25]=0
Y22	NORD2	ID/O8	Bidir	Hi-Z	SCU90[31]=1 & SCU94[1:0]=0 & SCUA4[26]=0
AA22	NORD3	ID/O8	Bidir	Hi-Z	SCU90[31]=1 & SCU94[1:0]=0 & SCUA4[27]=0
U22	NORD4	ID/O8	Bidir	Hi-Z	SCU90[31]=1 & SCU94[1:0]=0 & SCUA4[28]=0
T20	NORD5	ID/O8	Bidir	Hi-Z	SCU90[31]=1 & SCU94[1:0]=0 & SCUA4[29]=0
N18	NORD6	ID/O8	Bidir	Hi-Z	SCU90[31]=1 & SCU94[1:0]=0 & SCUA4[30]=0
P19	NORD7	ID/O8	Bidir	Hi-Z	SCU90[31]=1 & SCU94[1:0]=0 & SCUA4[31]=0
N19	NOROE#	ID/O8	Out	Hi-Z	SCU90[31]=1 & SCU94[1:0]=0
T21	NORWE#	ID/O8	Out	Hi-Z	SCU90[31]=1 & SCU94[1:0]=0
SD/SDIO/eMMC Interface					
C12	SD1CLK	IDS/O8	Out	Hi-Z	SCU90[0]=1
A12	SD1CMD	IDS/O8	Bidir	Hi-Z	SCU90[0]=1
B12	SD1DAT0	IDS/O8	Bidir	Hi-Z	SCU90[0]=1
D9	SD1DAT1	IDS/O8	Bidir	Hi-Z	SCU90[0]=1
D10	SD1DAT2	IDS/O8	Bidir	Hi-Z	SCU90[0]=1
E12	SD1DAT3	IDS/O8	Bidir	Hi-Z	SCU90[0]=1
F20	SD1DAT4	ID/O8	Bidir	Hi-Z	SCU90[3]=1
D20	SD1DAT5	ID/O8	Bidir	Hi-Z	SCU90[3]=1
D21	SD1DAT6	ID/O8	Bidir	Hi-Z	SCU90[3]=1
E20	SD1DAT7	ID/O8	Bidir	Hi-Z	SCU90[3]=1
C11	SD1CD#	IDS/O8	In	Input	SCU90[0]=1
B11	SD1WP#	IDS/O8	In	Input	SCU90[0]=1
F19	SD2CLK	ID/O8	Out	Hi-Z	SCU90[1]=1
E21	SD2CMD	ID/O8	Bidir	Hi-Z	SCU90[1]=1
F20	SD2DAT0	ID/O8	Bidir	Hi-Z	SCU90[1]=1
D20	SD2DAT1	ID/O8	Bidir	Hi-Z	SCU90[1]=1
D21	SD2DAT2	ID/O8	Bidir	Hi-Z	SCU90[1]=1
E20	SD2DAT3	ID/O8	Bidir	Hi-Z	SCU90[1]=1
G18	SD2CD#	ID/O8	In	Input	SCU90[1]=1
C21	SD2WP#	ID/O8	In	Input	SCU90[1]=1
RGMII1 Interface					
B4	RGMII1RXCK	IR/O8	In	Hi-Z,Input	Strap[6]=1 & SCUA0[12]=1
A4	RGMII1RXCTL	IR/O8	In	Hi-Z,Input	Strap[6]=1 & SCUA0[13]=1
A3	RGMII1RXD0	IR/O8	In	Hi-Z,Input	Strap[6]=1 & SCUA0[14]=1

Ball	Name	I/O	Type	Power-Up	Multi-function Control
D6	RGMII1RXD1	IR/O8	In	Hi-Z,Input	Strap[6]=1 & SCUA0[15]=1
C5	RGMII1RXD2	IR/O8	In	Hi-Z,Input	Strap[6]=1 & SCUA0[16]=1
C4	RGMII1RXD3	IR/O8	In	Hi-Z,Input	Strap[6]=1 & SCUA0[17]=1
B5	RGMII1TXCK	ID/O8	Out	Hi-Z	Strap[6]=1 & SCUA0[0]=0
E9	RGMII1TXCTL	ID/O8	Out	Hi-Z	Strap[6]=1 & SCUA0[1]=0
F9	RGMII1TXD0	ID/O8	Out	Hi-Z	Strap[6]=1 & SCUA0[2]=0
A5	RGMII1TXD1	ID/O8	Out	Hi-Z	Strap[6]=1 & SCUA0[3]=0
E7	RGMII1TXD2	ID/O8	Out	Hi-Z	Strap[6]=1 & SCUA0[4]=0
D7	RGMII1TXD3	ID/O8	Out	Hi-Z	Strap[6]=1 & SCUA0[5]=0
D8	MDC1	ID/O8	Out	Hi-Z	SCU88[30]=1
E10	MDIO1	ID/O8	Bidir	Hi-Z	SCU88[31]=1
RGMII2 Interface					
C2	RGMII2RXCK	IR/O8	In	Hi-Z,Input	Strap[7]=1 & SCUA0[18]=0
C1	RGMII2RXCTL	IR/O8	In	Hi-Z,Input	Strap[7]=1 & SCUA0[19]=0
C3	RGMII2RXD0	IR/O8	In	Hi-Z,Input	Strap[7]=1 & SCUA0[20]=0
D1	RGMII2RXD1	IR/O8	In	Hi-Z,Input	Strap[7]=1 & SCUA0[21]=0
D2	RGMII2RXD2	IR/O8	In	Hi-Z,Input	Strap[7]=1 & SCUA0[22]=0
E6	RGMII2RXD3	IR/O8	In	Hi-Z,Input	Strap[7]=1 & SCUA0[23]=0
B2	RGMII2TXCK	ID/O8	Out	Hi-Z	Strap[7]=1 & SCUA0[6]=0
B1	RGMII2TXCTL	ID/O8	Out	Hi-Z	Strap[7]=1 & SCUA0[7]=0
A2	RGMII2TXD0	ID/O8	Out	Hi-Z	Strap[7]=1 & SCUA0[8]=0
B3	RGMII2TXD1	ID/O8	Out	Hi-Z	Strap[7]=1 & SCUA0[9]=0
D5	RGMII2TXD2	ID/O8	Out	Hi-Z	Strap[7]=1 & SCUA0[10]=0
D4	RGMII2TXD3	ID/O8	Out	Hi-Z	Strap[7]=1 & SCUA0[11]=0
C13	MDC2	IS/O16	Out	Hi-Z	SCU90[2]=1 & SCU90[6]=0
B13	MDIO2	IS/O16	Bidir	Hi-Z	SCU90[2]=1 & SCU90[6]=0
RMII1 Interface					
B4	RMII1RCLKI	IR/O8	In	Hi-Z,Input	Strap[6]=0 & SCUA0[12]=0
A3	RMII1RXD0	IR/O8	In	Hi-Z,Input	Strap[6]=0 & SCUA0[14]=0
D6	RMII1RXD1	IR/O8	In	Hi-Z,Input	Strap[6]=0 & SCUA0[15]=0
C5	RMII1CRSDV	IR/O8	In	Hi-Z,Input	Strap[6]=0 & SCUA0[16]=0
C4	RMII1RXER	IR/O8	In	Hi-Z,Input	Strap[6]=0 & SCUA0[17]=0
B5	RMII1RCLKO	ID/Op	Out	Hi-Z	Strap[6]=0 & SCUA0[0]=0 & SCU48[29]=1
E9	RMII1TXEN	ID/Op	Out	Hi-Z	Strap[6]=0 & SCUA0[1]=0
F9	RMII1TXD0	ID/Op	Out	Hi-Z	Strap[6]=0 & SCUA0[2]=0
A5	RMII1TXD1	ID/Op	Out	Hi-Z	Strap[6]=0 & SCUA0[3]=0
RMII2 Interface					
C2	RMII2RCLKI	IR/O8	In	Hi-Z,Input	Strap[7]=0 & SCUA0[18]=0
C3	RMII2RXD0	IR/O8	In	Hi-Z,Input	Strap[7]=0 & SCUA0[20]=0
D1	RMII2RXD1	IR/O8	In	Hi-Z,Input	Strap[7]=0 & SCUA0[21]=0
D2	RMII2CRSDV	IR/O8	In	Hi-Z,Input	Strap[7]=0 & SCUA0[22]=0
E6	RMII2RXER	IR/O8	In	Hi-Z,Input	Strap[7]=0 & SCUA0[23]=0
B2	RMII2RCLKO	ID/Op	Out	Hi-Z,	Strap[7]=0 & SCUA0[6]=0 & SCU48[30]=1

Ball	Name	I/O	Type	Power-Up	Multi-function Control
B1	RMII2TXEN	ID/Op	Out	Hi-Z,	Strap[7]=0 & SCUA0[7]=0
A2	RMII2TXD0	ID/Op	Out	Hi-Z,	Strap[7]=0 & SCUA0[8]=0
B3	RMII2TXD1	ID/Op	Out	Hi-Z,	Strap[7]=0 & SCUA0[9]=0
UART Interface					
T2	NCTS1	ID/O8	In	Hi-Z	SCU84[16]=1
T1	NDCD1	ID/O8	In	Hi-Z	SCU84[17]=1 & SCU90[5]=0 & SCU94[1:0]=0
U1	NDSR1	ID/O8	In	Hi-Z	SCU84[18]=1 & SCU90[5]=0
U2	NRI1	ID/O8	In	Hi-Z	SCU84[19]=1 & SCU90[5]=0 & SCU94[1:0]=0
P4	NDTR1	ID/O8	Out	Hi-Z	SCU84[20]=1 & SCU90[5]=0 & SCU94[1:0]=0
P3	NRTS1	ID/O8	Out	Hi-Z	SCU84[21]=1 & SCU90[5]=0 & SCU94[1:0]=0
V1	TXD1	ID/O8	Out	Hi-Z	SCU84[22]=1 & SCU90[5]=0 & SCU94[1:0]=0
W1	RXD1	ID/O8	In	Hi-Z	SCU84[23]=1 & SCU90[5]=0 & SCU94[1:0]=0
Y1	NCTS2	ID/O8	In	Hi-Z	SCU84[24]=1 & SCU90[5]=0 & SCU94[1:0]=0
AB2	NDCD2	ID/O8	In	Hi-Z	SCU84[25]=1 & SCU90[5]=0 & SCU94[1:0]=0
AA1	NDSR2	ID/O8	In	Hi-Z	SCU84[26]=1 & SCU90[5]=0 & SCU94[1:0]=0
Y2	NRI2	ID/O8	In	Hi-Z	SCU84[27]=1 & SCU90[5]=0 & SCU94[1:0]=0
AA2	NDTR2	ID/O8	Out	Hi-Z	SCU84[28]=1 & SCU90[5]=0 & SCU94[1:0]=0
P5	NRTS2	ID/O8	Out	Hi-Z	SCU84[29]=1 & SCU90[5]=0 & SCU94[1:0]=0
R5	TXD2	ID/O8	Out	Hi-Z	SCU84[30]=1 & SCU90[5]=0 & SCU94[1:0]=0
T5	RXD2	ID/O8	In	Hi-Z	SCU84[31]=1 & SCU90[5]=0 & SCU94[1:0]=0
B20	NCTS3	ID/O8	In	Hi-Z	SCU80[16]=1
C20	NDCD3	ID/O8	In	Hi-Z	SCU80[17]=1
F18	NDSR3	ID/O8	In	Hi-Z	SCU80[18]=1
F17	NRI3	ID/O8	In	Hi-Z	SCU80[19]=1
E18	NDTR3	ID/O8	Out	Hi-Z	SCU80[20]=1
D19	NRTS3	ID/O8	Out	Hi-Z	SCU80[21]=1
A20	TXD3	ID/O8	Out	Hi-Z	SCU80[22]=1
B19	RXD3	ID/O8	In	Hi-Z	SCU80[23]=1
J19	NCTS4	ID/O8	In	Hi-Z	SCU80[24]=1 & LHCR0[0]=0 & SCU90[30]=0
J18	NDCD4	ID/O8	In	Hi-Z	SCU80[25]=1 & LHCR0[0]=0 & SCU90[30]=0
B22	NDSR4	ID/O8	In	Hi-Z	SCU80[26]=1 & LHCR0[0]=0 & SCU90[30]=0
B21	NRI4	ID/O8	In	Hi-Z	SCU80[27]=1 & LHCR0[0]=0 & SCU90[30]=0
A21	NDTR4	ID/O8	Out	Hi-Z	SCU80[28]=1 & LHCR0[0]=0 & SCU90[30]=0
H19	NRTS4	ID/O8	Out	Hi-Z	SCU80[29]=1 & LHCR0[0]=0 & SCU90[30]=0
G17	TXD4	ID/O8	Out	Hi-Z	SCU80[30]=1 & LHCR0[0]=0
H18	RXD4	ID/O8	In	Hi-Z	SCU80[31]=1 & LHCR0[0]=0 & SCU90[30]=0
A18	NCTS6	ID/O8	In	Hi-Z	SCU90[7:6]=”10” & SCU94[5]=0
B18	NDCD6	ID/O8	In	Hi-Z	SCU90[7:6]=”10” & SCU94[5]=0
D17	NDSR6	ID/O8	In	Hi-Z	SCU90[7:6]=”10” & SCU94[6]=0
C17	NRI6	ID/O8	In	Hi-Z	SCU90[7:6]=”10” & SCU94[6]=0
A17	NDTR6	ID/O8	Out	Hi-Z	SCU90[7:6]=”10” & SCU94[7]=0
B17	NRTS6	ID/O8	Out	Hi-Z	SCU90[7:6]=”10” & SCU94[7]=0
A16	TXD6	ID/O8	Out	Hi-Z	SCU90[7:6]=”10”

Ball	Name	I/O	Type	Power-Up	Multi-function Control
D18	RXD6	ID/O8	In	Hi-Z	SCU90[7:6]=”10”
VGA Interface					
N5	VGAHS	ID/O8	Out	Hi-Z	SCU84[12]=1
R4	VGAVS	ID/O8	Out	Hi-Z	SCU84[13]=1
R3	DDCCLK	ID/O8	Bidir	Hi-Z	SCU84[14]=1
T3	DDCDAT	ID/O8	Bidir	Hi-Z	SCU84[15]=1
Digital Video Output Interface					
V20	VPOB2	ID/O8	Out	Hi-Z	SCU8C[0]=1 & SCU94[1:0]!=0 & GFX064[7]=1
U19	VPOB3	ID/O8	Out	Hi-Z	SCU8C[1]=1 & SCU94[1:0]!=0 & GFX064[7]=1
R18	VPOB4	ID/O8	Out	Hi-Z	SCU8C[2]=1 & SCU94[1:0]!=0 & GFX064[7]=1
P18	VPOB5	ID/O8	Out	Hi-Z	SCU8C[3]=1 & SCU94[1:0]!=0 & GFX064[7]=1
R19	VPOB6	ID/O8	Out	Hi-Z	SCU8C[4]=1 & SCU94[1:0]!=0 & GFX064[7]=1
W20	VPOB7	ID/O8	Out	Hi-Z	SCU8C[5]=1 & SCU94[1:0]!=0 & GFX064[7]=1
U20	VPOB8	ID/O8	Out	Hi-Z	SCU8C[6]=1 & SCU94[1:0]!=0 & GFX064[7]=1
AA20	VPOB9	ID/O8	Out	Hi-Z	SCU8C[7]=1 & SCU94[1:0]!=0 & GFX064[7]=1
Y20	VPOG2	ID/O8	Out	Hi-Z	SCUA4[16]=1 & SCU94[1:0]!=0 & GFX064[7]=1
AB20	VPOG3	ID/O8	Out	Hi-Z	SCUA4[17]=1 & SCU94[1:0]!=0 & GFX064[7]=1
AB21	VPOG4	ID/O8	Out	Hi-Z	SCUA4[18]=1 & SCU94[1:0]!=0 & GFX064[7]=1
AA21	VPOG5	ID/O8	Out	Hi-Z	SCUA4[19]=1 & SCU94[1:0]!=0 & GFX064[7]=1
U21	VPOG6	ID/O8	Out	Hi-Z	SCUA4[20]=1 & SCU94[1:0]!=0 & GFX064[7:6]=2
W22	VPOG7	ID/O8	Out	Hi-Z	SCUA4[21]=1 & SCU94[1:0]!=0 & GFX064[7:6]=2
V22	VPOG8	ID/O8	Out	Hi-Z	SCUA4[22]=1 & SCU94[1:0]!=0 & GFX064[7:6]=2
W21	VPOG9	ID/O8	Out	Hi-Z	SCUA4[23]=1 & SCU94[1:0]!=0 & GFX064[7:6]=2
Y21	VPOR2	ID/O8	Out	Hi-Z	SCUA4[24]=1 & SCU94[1:0]!=0 & GFX064[7:6]=2
V21	VPOR3	ID/O8	Out	Hi-Z	SCUA4[25]=1 & SCU94[1:0]!=0 & GFX064[7:6]=2
Y22	VPOR4	ID/O8	Out	Hi-Z	SCUA4[26]=1 & SCU94[1:0]!=0 & GFX064[7:6]=2
AA22	VPOR5	ID/O8	Out	Hi-Z	SCUA4[27]=1 & SCU94[1:0]!=0 & GFX064[7:6]=2
U22	VPOR6	ID/O8	Out	Hi-Z	SCUA4[28]=1 & SCU94[1:0]!=0 & GFX064[7:6]=2
T20	VPOR7	ID/O8	Out	Hi-Z	SCUA4[29]=1 & SCU94[1:0]!=0 & GFX064[7:6]=2
N18	VPOR8	ID/O8	Out	Hi-Z	SCUA4[30]=1 & SCU94[1:0]!=0 & GFX064[7:6]=2
P19	VPOR9	ID/O8	Out	Hi-Z	SCUA4[31]=1 & SCU94[1:0]!=0 & GFX064[7:6]=2
N19	VPODE	ID/O8	Out	Hi-Z	SCUA8[0]=1 & SCU94[1:0]!=0 & GFX064[7]=1
T21	VPOHS	ID/O8	Out	Hi-Z	SCUA8[1]=1 & SCU94[1:0]!=0 & GFX064[7]=1
T22	VPOVS	ID/O8	Out	Hi-Z	SCUA8[2]=1 & SCU94[1:0]!=0 & GFX064[7]=1
R20	VPOCLK	ID/O8	Out	Hi-Z	SCUA8[3]=1 & SCU94[1:0]!=0 & GFX064[7]=1
Digital Video Input Interface					
T1	VPIDE	ID/O8	In	Hi-Z	SCU90[5]=1 & SCU84[17]=1 & SCU94[1:0]=0
U2	VPIHS	ID/O8	In	Hi-Z	SCU90[5]=1 & SCU84[19]=1 & SCU94[1:0]=0
P4	VPIVS	ID/O8	In	Hi-Z	SCU90[5]=1 & SCU84[20]=1 & SCU94[1:0]=0
P3	VPICLK	ID/O8	In	Hi-Z	SCU90[5]=1 & SCU84[21]=1 & SCU94[1:0]=0
Y1	VPIB2	ID/O8	In	Hi-Z	SCU90[5]=1 & SCU84[24]=1 & SCU94[1:0]=0
AB2	VPIB3	ID/O8	In	Hi-Z	SCU90[5]=1 & SCU84[25]=1 & SCU94[1:0]=0
AA1	VPIB4	ID/O8	In	Hi-Z	SCU90[5]=1 & SCU84[26]=1 & SCU94[1:0]=0

Ball	Name	I/O	Type	Power-Up	Multi-function Control
Y2	VPIB5	ID/O8	In	Hi-Z	SCU90[5]=1 & SCU84[27]=1 & SCU94[1:0]=0
AA2	VPIB6	ID/O8	In	Hi-Z	SCU90[5]=1 & SCU84[28]=1 & SCU94[1:0]=0
P5	VPIB7	ID/O8	In	Hi-Z	SCU90[5]=1 & SCU84[29]=1 & SCU94[1:0]=0
R5	VPIB8	ID/O8	In	Hi-Z	SCU90[5]=1 & SCU84[30]=1 & SCU94[1:0]=0
T5	VPIB9	ID/O8	In	Hi-Z	SCU90[5]=1 & SCU84[31]=1 & SCU94[1:0]=0
V3	VPIG2	ID/O8	In	Hi-Z	SCU90[5]=1 & SCU88[2]=1 & SCU94[1:0]=0
U3	VPIG3	ID/O8	In	Hi-Z	SCU90[5]=1 & SCU88[3]=1 & SCU94[1:0]=0
W3	VPIG4	ID/O8	In	Hi-Z	SCU90[5]=1 & SCU88[4]=1 & SCU94[1:0]=0
AA3	VPIG5	ID/O8	In	Hi-Z	SCU90[5]=1 & SCU88[5]=1 & SCU94[1:0]=0
Y3	VPIG6	ID/O8	In	Hi-Z	SCU90[5:4]=2 & SCU88[6]=1 & SCU94[1:0]=0
T4	VPIG7	ID/O8	In	Hi-Z	SCU90[5:4]=2 & SCU88[7]=1 & SCU94[1:0]=0
U5	VPIG8	ID/O8	In	Hi-Z	SCU90[5:4]=2 & SCU88[8]=1 & SCU94[1:0]=0
U4	VPIG9	ID/O8	In	Hi-Z	SCU90[5:4]=2 & SCU88[9]=1 & SCU94[1:0]=0
AB3	VPIR2	ID/O8	In	Hi-Z	SCU90[5:4]=2 & SCU88[12]=1 & SCU94[1:0]=0
Y4	VPIR3	ID/O8	In	Hi-Z	SCU90[5:4]=2 & SCU88[13]=1 & SCU94[1:0]=0
AA4	VPIR4	ID/O8	In	Hi-Z	SCU90[5:4]=2 & SCU88[14]=1 & SCU94[1:0]=0
W4	VPIR5	ID/O8	In	Hi-Z	SCU90[5:4]=2 & SCU88[15]=1 & SCU94[1:0]=0
V4	VPIR6	ID/O8	In	Hi-Z	SCU90[5:4]=2 & SCU88[16]=1 & SCU94[1:0]=0
W5	VPIR7	ID/O8	In	Hi-Z	SCU90[5:4]=2 & SCU88[17]=1 & SCU94[1:0]=0
AA5	VPIR8	ID/O8	In	Hi-Z	SCU90[5:4]=2 & SCU88[18]=1 & SCU94[1:0]=0
AB5	VPIR9	ID/O8	In	Hi-Z	SCU90[5:4]=2 & SCU88[19]=1 & SCU94[1:0]=0
I2C Interface					
M18	SCL1	IS/O8	Bidir	Hi-Z	SCUA4[12]=1
M19	SDA1	IS/O8	Bidir	Hi-Z	SCUA4[13]=1
M20	SCL2	IS/O8	Bidir	Hi-Z	SCUA4[14]=1
P20	SDA2	IS/O8	Bidir	Hi-Z	SCUA4[15]=1
A11	SCL3	IS/O8	Bidir	Hi-Z	SCU90[16]=1
A10	SDA3	IS/O8	Bidir	Hi-Z	SCU90[16]=1
A9	SCL4	IS/O8	Bidir	Hi-Z	SCU90[17]=1
B9	SDA4	IS/O8	Bidir	Hi-Z	SCU90[17]=1
L3	SCL5	IS/O8	Bidir	Hi-Z	SCU90[18]=1
L4	SDA5	IS/O8	Bidir	Hi-Z	SCU90[18]=1
L1	SCL6	IS/O8	Bidir	Hi-Z	SCU90[19]=1
N2	SDA6	IS/O8	Bidir	Hi-Z	SCU90[19]=1
N1	SCL7	IS/O8	Bidir	Hi-Z	SCU90[20]=1
P1	SDA7	IS/O8	Bidir	Hi-Z	SCU90[20]=1
P2	SCL8	IS/O8	Bidir	Hi-Z	SCU90[21]=1
R1	SDA8	IS/O8	Bidir	Hi-Z	SCU90[21]=1
C14	SCL9	IS/O8	Bidir	Hi-Z	SCU90[22]=1 & SCU90[6]=0
A13	SDA9	IS/O8	Bidir	Hi-Z	SCU90[22]=1 & SCU90[6]=0
C12	SCL10	IS/O8	Bidir	Hi-Z	SCU90[23]=1 & SCU90[0]=0
A12	SDA10	IS/O8	Bidir	Hi-Z	SCU90[23]=1 & SCU90[0]=0
B12	SCL11	IS/O8	Bidir	Hi-Z	SCU90[24]=1 & SCU90[0]=0

Ball	Name	I/O	Type	Power-Up	Multi-function Control
D9	SDA11	IS/O8	Bidir	Hi-Z	SCU90[24]=1 & SCU90[0]=0
D10	SCL12	IS/O8	Bidir	Hi-Z	SCU90[25]=1 & SCU90[0]=0
E12	SDA12	IS/O8	Bidir	Hi-Z	SCU90[25]=1 & SCU90[0]=0
C11	SCL13	IS/O8	Bidir	Hi-Z	SCU90[26]=1 & SCU90[0]=0
B11	SDA13	IS/O8	Bidir	Hi-Z	SCU90[26]=1 & SCU90[0]=0
N21	SCL14	IS/O8	Bidir	Hi-Z	SCU90[27]=1
N22	SDA14	IS/O8	Bidir	Hi-Z	SCU90[27]=1
E17	SALT1	ID/O8	Bidir	Hi-Z	SCU84[4]=1 & SCU90[6]=0 & SCU94[12]=0
D16	SALT2	ID/O8	Bidir	Hi-Z	SCU84[5]=1 & SCU90[6]=0 & SCU94[12]=0
D15	SALT3	ID/O8	Bidir	Hi-Z	SCU84[6]=1 & SCU90[6]=0 & SCU94[12]=0
E14	SALT4	ID/O8	Bidir	Hi-Z	SCU84[7]=1 & SCU90[6]=0 & SCU94[12]=0
R18	SALT5	ID/O8	Bidir	Hi-Z	SCU8C[2]=1 & SCU94[1:0]=0
P18	SALT6	ID/O8	Bidir	Hi-Z	SCU8C[3]=1 & SCU94[1:0]=0
Y21	SALT7	ID/O8	Bidir	Hi-Z	SCUA4[24]=1 & SCU94[1:0]=0
V21	SALT8	ID/O8	Bidir	Hi-Z	SCUA4[25]=1 & SCU94[1:0]=0
Y22	SALT9	ID/O8	Bidir	Hi-Z	SCUA4[26]=1 & SCU94[1:0]=0
AA22	SALT10	ID/O8	Bidir	Hi-Z	SCUA4[27]=1 & SCU94[1:0]=0
U22	SALT11	ID/O8	Bidir	Hi-Z	SCUA4[28]=1 & SCU94[1:0]=0
T20	SALT12	ID/O8	Bidir	Hi-Z	SCUA4[29]=1 & SCU94[1:0]=0
N18	SALT13	ID/O8	Bidir	Hi-Z	SCUA4[30]=1 & SCU94[1:0]=0
P19	SALT14	ID/O8	Bidir	Hi-Z	SCUA4[31]=1 & SCU94[1:0]=0
PWM Interface					
V2	PWM0	ID/O8	Out	Hi-Z	SCU88[0]=1 & SCU94[1:0]=0 & SCU90[5]=0
W2	PWM1	ID/O8	Out	Hi-Z	SCU88[1]=1 & SCU94[1:0]=0 & SCU90[5]=0
V3	PWM2	ID/O8	Out	Hi-Z	SCU88[2]=1 & SCU94[1:0]=0 & SCU90[5]=0
U3	PWM3	ID/O8	Out	Hi-Z	SCU88[3]=1 & SCU94[1:0]=0 & SCU90[5]=0
W3	PWM4	ID/O8	Out	Hi-Z	SCU88[4]=1 & SCU94[1:0]=0 & SCU90[5]=0
AA3	PWM5	ID/O8	Out	Hi-Z	SCU88[5]=1 & SCU94[1:0]=0 & SCU90[5]=0
Y3	PWM6	ID/O8	Out	Hi-Z	SCU88[6]=1 & SCU94[1:0]=0 & SCU90[5]=0
T4	PWM7	ID/O8	Out	Hi-Z	SCU88[7]=1 & SCU94[1:0]=0 & SCU90[5]=0
Tachometer Interface					
U5	TACH0	ID/O8	In	Hi-Z,Input	SCU88[8]=0 & GPIO07C[16]=0
U4	TACH1	ID/O8	In	Hi-Z,Input	SCU88[9]=0 & GPIO07C[17]=0
V5	TACH2	ID/O8	In	Hi-Z,Input	SCU88[10]=0 & GPIO07C[18]=0
AB4	TACH3	ID/O8	In	Hi-Z,Input	SCU88[11]=0 & GPIO07C[19]=0
AB3	TACH4	ID/O8	In	Hi-Z,Input	SCU88[12]=0 & GPIO07C[20]=0
Y4	TACH5	ID/O8	In	Hi-Z,Input	SCU88[13]=0 & GPIO07C[21]=0
AA4	TACH6	ID/O8	In	Hi-Z,Input	SCU88[14]=0 & GPIO07C[22]=0
W4	TACH7	ID/O8	In	Hi-Z,Input	SCU88[15]=0 & GPIO07C[23]=0
V4	TACH8	ID/O8	In	Hi-Z,Input	SCU88[16]=0 & GPIO07C[24]=0
W5	TACH9	ID/O8	In	Hi-Z,Input	SCU88[17]=0 & GPIO07C[25]=0
AA5	TACH10	ID/O8	In	Hi-Z,Input	SCU88[18]=0 & GPIO07C[26]=0
AB5	TACH11	ID/O8	In	Hi-Z,Input	SCU88[19]=0 & GPIO07C[27]=0

Ball	Name	I/O	Type	Power-Up	Multi-function Control
Y6	TACH12	ID/O8	In	Hi-Z,Input	SCU88[20]=0 & GPIO07C[28]=0
Y5	TACH13	ID/O8	In	Hi-Z,Input	SCU88[21]=0 & GPIO07C[29]=0
W6	TACH14	ID/O8	In	Hi-Z,Input	SCU88[22]=0 & GPIO07C[30]=0
V6	TACH15	ID/O8	In	Hi-Z,Input	SCU88[23]=0 & GPIO07C[31]=0
ADC Interface					
F4	ADC0	I	In	Input	SCUA0[24]=0
F5	ADC1	I	In	Input	SCUA0[25]=0
E2	ADC2	I	In	Input	SCUA0[26]=0
E1	ADC3	I	In	Input	SCUA0[27]=0
F3	ADC4	I	In	Input	SCUA0[28]=0
E3	ADC5	I	In	Input	SCUA0[29]=0
G5	ADC6	I	In	Input	SCUA0[30]=0
G4	ADC7	I	In	Input	SCUA0[31]=0
F2	ADC8	I	In	Input	SCUA4[0]=0
G3	ADC9	I	In	Input	SCUA4[1]=0
G2	ADC10	I	In	Input	SCUA4[2]=0
F1	ADC11	I	In	Input	SCUA4[3]=0
H5	ADC12	I	In	Input	SCUA4[4]=0
G1	ADC13	I	In	Input	SCUA4[5]=0
H3	ADC14	I	In	Input	SCUA4[6]=0
H4	ADC15	I	In	Input	SCUA4[7]=0
SGPIO Master Interface					
R2	SGPMCK	ID/O8	Out	Hi-Z	SCU84[8]=1
L2	SGPMLD	ID/O8	Out	Hi-Z	SCU84[9]=1
N3	SGPMO	ID/O8	Out	Hi-Z	SCU84[10]=1
N4	SGPMI	ID/O8	In	Hi-Z	SCU84[11]=1
SGPIO Slave Interface					
A19	SGPS1CK	ID/O8	In	Hi-Z	SCU90[6]=0 & SCU84[0]=1
E19	SGPS1LD	ID/O8	In	Hi-Z	SCU90[6]=0 & SCU84[1]=1
C19	SGPS1I0	ID/O8	In	Hi-Z	SCU90[6]=0 & SCU84[2]=1
E16	SGPS1I1	ID/O8	In	Hi-Z	SCU90[6]=0 & SCU84[3]=1
E17	SGPS2CK	ID/O8	In	Hi-Z	SCU90[6]=0 & SCU94[12]=1
D16	SGPS2LD	ID/O8	In	Hi-Z	SCU90[6]=0 & SCU94[12]=1
D15	SGPS2I0	ID/O8	In	Hi-Z	SCU90[6]=0 & SCU94[12]=1
E14	SGPS2I1	ID/O8	In	Hi-Z	SCU90[6]=0 & SCU94[12]=1
GPIO Interface					
B14	GPIOA0	ID/O16	Bidir	Hi-Z,Input	SCU80[0]=0
D14	GPIOA1	ID/O16	Bidir	Hi-Z,Input	SCU80[1]=0
D13	GPIOA2	ID/O16	Bidir	Hi-Z,Input	SCU80[2]=0 & SCU80[15]=0
E13	GPIOA3	ID/O16	Bidir	Hi-Z,Input	SCU80[3]=0
C14	GPIOA4	IDS/O8	Bidir	Hi-Z,Input	SCU90[6,22]=0 & SCU80[4]=0
A13	GPIOA5	IDS/O8	Bidir	Hi-Z,Input	SCU90[6,22]=0 & SCU80[5]=0
C13	GPIOA6	IDS/O8	Bidir	Hi-Z,Input	SCU90[6,2]=0 & SCU80[6]=0

Ball	Name	I/O	Type	Power-Up	Multi-function Control
B13	GPIOA7	IDS/O8	Bidir	Hi-Z,Input	SCU90[6,2]=0 & SCU80[7]=0
J20	GPIOB4	ID/O8	Bidir	Hi-Z,Input	Strap[23]=0
H21	GPIOB5	ID/O8	Bidir	Hi-Z,Input	SCU80[13]=0
H22	GPIOB6	ID/O8	Bidir	Hi-Z,Input	SCU80[14]=0
C12	GPIOC0	IDS/O8	Bidir	Hi-Z,Input	SCU90[0]=0 & SCU90[23]=0
A12	GPIOC1	IDS/O8	Bidir	Hi-Z,Input	SCU90[0]=0 & SCU90[23]=0
B12	GPIOC2	IDS/O8	Bidir	Hi-Z,Input	SCU90[0]=0 & SCU90[24]=0
D9	GPIOC3	IDS/O8	Bidir	Hi-Z,Input	SCU90[0]=0 & SCU90[24]=0
D10	GPIOC4	IDS/O8	Bidir	Hi-Z,Input	SCU90[0]=0 & SCU90[25]=0
E12	GPIOC5	IDS/O8	Bidir	Hi-Z,Input	SCU90[0]=0 & SCU90[25]=0
C11	GPIOC6	IDS/O8	Bidir	Hi-Z,Input	SCU90[0]=0 & SCU90[26]=0
B11	GPIOC7	IDS/O8	Bidir	Hi-Z,Input	SCU90[0]=0 & SCU90[26]=0
F19	GPIOD0	ID/O8	Bidir	Hi-Z,Input	SCU90[1]=0 & SCU8C[8]=0 & Strap[21]=0
E21	GPIOD1	ID/O8	Bidir	Hi-Z,Input	SCU90[1]=0 & SCU8C[8]=0 & Strap[21]=0
F20	GPIOD2	ID/O8	Bidir	Hi-Z,Input	SCU90[1]=0 & SCU8C[9]=0 & Strap[21]=0
D20	GPIOD3	ID/O8	Bidir	Hi-Z,Input	SCU90[1]=0 & SCU8C[9]=0 & Strap[21]=0
D21	GPIOD4	ID/O8	Bidir	Hi-Z,Input	SCU90[1]=0 & SCU8C[10]=0 & Strap[21]=0
E20	GPIOD5	ID/O8	Bidir	Hi-Z,Input	SCU90[1]=0 & SCU8C[10]=0 & Strap[21]=0
G18	GPIOD6	ID/O8	Bidir	Hi-Z,Input	SCU90[1]=0 & SCU8C[11]=0 & Strap[21]=0
C21	GPIOD7	ID/O8	Bidir	Hi-Z,Input	SCU90[1]=0 & SCU8C[11]=0 & Strap[21]=0
B20	GPIOE0	ID/O8	Bidir	Hi-Z,Input	SCU80[16]=0 & SCU8C[12]=0 & Strap[22]=0
C20	GPIOE1	ID/O8	Bidir	Hi-Z,Input	SCU80[17]=0 & SCU8C[12]=0 & Strap[22]=0
F18	GPIOE2	ID/O8	Bidir	Hi-Z,Input	SCU80[18]=0 & SCU8C[13]=0 & Strap[22]=0
F17	GPIOE3	ID/O8	Bidir	Hi-Z,Input	SCU80[19]=0 & SCU8C[13]=0 & Strap[22]=0
E18	GPIOE4	ID/O8	Bidir	Hi-Z,Input	SCU80[20]=0 & SCU8C[14]=0 & Strap[22]=0
D19	GPIOE5	ID/O8	Bidir	Hi-Z,Input	SCU80[21]=0 & SCU8C[14]=0 & Strap[22]=0
A20	GPIOE6	ID/O8	Bidir	Hi-Z,Input	SCU80[22]=0 & SCU8C[15]=0 & Strap[22]=0
B19	GPIOE7	ID/O8	Bidir	Hi-Z,Input	SCU80[23]=0 & SCU8C[15]=0 & Strap[22]=0
J19	GPIOF0	ID/O8	Bidir	Hi-Z,Input	LHCR0[0]=0 & SCU80[24]=0 & SCU90[30]=0
J18	GPIOF1	ID/O8	Bidir	Hi-Z,Input	LHCR0[0]=0 & SCU80[25]=0 & SCU90[30]=0
B22	GPIOF2	ID/O8	Bidir	Hi-Z,Input	LHCR0[0]=0 & SCU80[26]=0 & SCU90[30]=0
B21	GPIOF3	ID/O8	Bidir	Hi-Z,Input	LHCR0[0]=0 & SCU80[27]=0 & SCU90[30]=0
A21	GPIOF4	ID/O8	Bidir	Hi-Z,Input	LHCR0[0]=0 & SCU80[28]=0 & SCU90[30]=0
H19	GPIOF5	ID/O8	Bidir	Hi-Z,Input	LHCR0[0]=0 & SCU80[29]=0 & SCU90[30]=0
G17	GPIOF6	ID/O8	Bidir	Hi-Z,Input	LHCR0[0]=0 & SCU80[30]=0
H18	GPIOF7	ID/O8	Bidir	Hi-Z,Input	LHCR0[0]=0 & SCU80[31]=0 & SCU90[30]=0
A19	GPIOG0	ID/O8	Bidir	Hi-Z,Input	SCU90[6]=0 & SCU84[0]=0
E19	GPIOG1	ID/O8	Bidir	Hi-Z,Input	SCU90[6]=0 & SCU84[1]=0
C19	GPIOG2	ID/O8	Bidir	Hi-Z,Input	SCU90[6]=0 & SCU84[2]=0
E16	GPIOG3	ID/O8	Bidir	Hi-Z,Input	SCU90[6]=0 & SCU84[3]=0
E17	GPIOG4	ID/O8	Bidir	Hi-Z,Input	SCU90[6]=0 & SCU84[4]=0 & SCU94[12]=0
D16	GPIOG5	ID/O8	Bidir	Hi-Z,Input	SCU90[6]=0 & SCU84[5]=0 & SCU94[12]=0
D15	GPIOG6	ID/O8	Bidir	Hi-Z,Input	SCU90[6]=0 & SCU84[6]=0 & SCU94[12]=0

Ball	Name	I/O	Type	Power-Up	Multi-function Control
E14	GPIOG7	ID/O8	Bidir	Hi-Z,Input	SCU90[6]=0 & SCU84[7]=0 & SCU94[12]=0
A18	GPIOH0	ID/O8	Bidir	Hi-Z,Input	SCU90[7:6]=0 & SCU10[8]=0 & SCU94[5]=0
B18	GPIOH1	ID/O8	Bidir	Hi-Z,Input	SCU90[7:6]=0 & SCU10[8]=0 & SCU94[5]=0
D17	GPIOH2	ID/O8	Bidir	Hi-Z,Input	SCU90[7:6]=0 & SCU10[8]=0 & SCU94[6]=0
C17	GPIOH3	ID/O8	Bidir	Hi-Z,Input	SCU90[7:6]=0 & SCU10[8]=0 & SCU94[6]=0
A17	GPIOH4	ID/O8	Bidir	Hi-Z,Input	SCU90[7:6]=0 & SCU10[8]=0 & SCU94[7]=0
B17	GPIOH5	ID/O8	Bidir	Hi-Z,Input	SCU90[7:6]=0 & SCU10[8]=0 & SCU94[7]=0
A16	GPIOH6	ID/O8	Bidir	Hi-Z,Input	SCU90[7:6]=0 & SCU10[8]=0
D18	GPIOH7	ID/O8	Bidir	Hi-Z,Input	SCU90[7:6]=0 & SCU10[8]=0
C18	GPIOI0	ID/O8	Bidir	Hi-Z,Input	SCU90[6]=0 & Strap[13]=0
E15	GPIOI1	ID/O8	Bidir	Hi-Z,Input	SCU90[6]=0 & Strap[13]=0
B16	GPIOI2	ID/O8	Bidir	Hi-Z,Input	SCU90[6]=0 & Strap[13]=0
C16	GPIOI3	ID/O8	Bidir	Hi-Z,Input	SCU90[6]=0 & Strap[13]=0
B15	GPIOI4	ID/O8	Bidir	Hi-Z,Input	SCU90[6]=0 & Strap[13:12,5]=0
C15	GPIOI5	ID/O8	Bidir	Hi-Z,Input	SCU90[6]=0 & Strap[13:12,5]=0
A14	GPIOI6	ID/O8	Bidir	Hi-Z,Input	SCU90[6]=0 & Strap[13:12,5]=0
A15	GPIOI7	ID/O8	Bidir	Hi-Z,Input	SCU90[6]=0 & Strap[13:12,5]=0
R2	GPIOJ0	ID/O8	Bidir	Hi-Z,Input	SCU10[8]=0 & SCU84[8]=0
L2	GPIOJ1	ID/O8	Bidir	Hi-Z,Input	SCU10[8]=0 & SCU84[9]=0
N3	GPIOJ2	ID/O8	Bidir	Hi-Z,Input	SCU10[8]=0 & SCU84[10]=0
N4	GPIOJ3	ID/O8	Bidir	Hi-Z,Input	SCU10[8]=0 & SCU84[11]=0
N5	GPIOJ4	ID/O8	Bidir	Hi-Z	SCU10[8]=0 & SCU84[12]=0 & SCU94[8]=0
R4	GPIOJ5	ID/O8	Bidir	Hi-Z	SCU10[8]=0 & SCU84[13]=0 & SCU94[8]=0
R3	GPIOJ6	ID/O8	Bidir	Hi-Z	SCU10[8]=0 & SCU84[14]=0 & SCU94[9]=0
T3	GPIOJ7	ID/O8	Bidir	Hi-Z	SCU10[8]=0 & SCU84[15]=0 & SCU94[9]=0
L3	GPIOK0	IDS/O8	Bidir	Hi-Z,Input	SCU90[18]=0
L4	GPIOK1	IDS/O8	Bidir	Hi-Z,Input	SCU90[18]=0
L1	GPIOK2	IDS/O8	Bidir	Hi-Z,Input	SCU90[19]=0
N2	GPIOK3	IDS/O8	Bidir	Hi-Z,Input	SCU90[19]=0
N1	GPIOK4	IDS/O8	Bidir	Hi-Z,Input	SCU90[20]=0
P1	GPIOK5	IDS/O8	Bidir	Hi-Z,Input	SCU90[20]=0
P2	GPIOK6	IDS/O8	Bidir	Hi-Z,Input	SCU90[21]=0
R1	GPIOK7	IDS/O8	Bidir	Hi-Z,Input	SCU90[21]=0
T2	GPIOL0	ID/O8	Bidir	Hi-Z,Input	SCU84[16]=0
T1	GPIOL1	ID/O8	Bidir	Hi-Z,Input	SCU84[17]=0
U1	GPIOL2	ID/O8	Bidir	Hi-Z,Input	SCU84[18]=0
U2	GPIOL3	ID/O8	Bidir	Hi-Z,Input	SCU84[19]=0
P4	GPIOL4	ID/O8	Bidir	Hi-Z,Input	SCU84[20]=0
P3	GPIOL5	ID/O8	Bidir	Hi-Z,Input	SCU84[21]=0
V1	GPIOL6	ID/O8	Bidir	Hi-Z,Input	SCU84[22]=0
W1	GPIOL7	ID/O8	Bidir	Hi-Z,Input	SCU84[23]=0
Y1	GPIOM0	ID/O8	Bidir	Hi-Z,Input	SCU84[24]=0
AB2	GPIOM1	ID/O8	Bidir	Hi-Z,Input	SCU84[25]=0

Ball	Name	I/O	Type	Power-Up	Multi-function Control
AA1	GPIOM2	ID/O8	Bidir	Hi-Z,Input	SCU84[26]=0
Y2	GPIOM3	ID/O8	Bidir	Hi-Z,Input	SCU84[27]=0
AA2	GPIOM4	ID/O8	Bidir	Hi-Z,Input	SCU84[28]=0
P5	GPIOM5	ID/O8	Bidir	Hi-Z,Input	SCU84[29]=0
R5	GPIOM6	ID/O8	Bidir	Hi-Z,Input	SCU84[30]=0
T5	GPIOM7	ID/O8	Bidir	Hi-Z,Input	SCU84[31]=0
V2	GPION0	ID/O8	Bidir	Hi-Z,Input	SCU88[0]=0
W2	GPION1	ID/O8	Bidir	Hi-Z,Input	SCU88[1]=0
V3	GPION2	ID/O8	Bidir	Hi-Z,Input	SCU88[2]=0
U3	GPION3	ID/O8	Bidir	Hi-Z,Input	SCU88[3]=0
W3	GPION4	ID/O8	Bidir	Hi-Z,Input	SCU88[4]=0
AA3	GPION5	ID/O8	Bidir	Hi-Z,Input	SCU88[5]=0
Y3	GPION6	ID/O8	Bidir	Hi-Z,Input	SCU88[6]=0
T4	GPION7	ID/O8	Bidir	Hi-Z,Input	SCU88[7]=0
U5	GPIOO0	ID/O8	Bidir	Hi-Z,Input	SCU88[8]=0 || (SCU90[5]=0 & SCU90[5:4]=0)
U4	GPIOO1	ID/O8	Bidir	Hi-Z,Input	SCU88[9]=0 || (SCU90[5]=0 & SCU90[5:4]=0)
V5	GPIOO2	ID/O8	Bidir	Hi-Z,Input	SCU88[10]=0 || SCU90[5]=0
AB4	GPIOO3	ID/O8	Bidir	Hi-Z,Input	SCU88[11]=0 || SCU90[5]=0
AB3	GPIOO4	ID/O8	Bidir	Hi-Z,Input	SCU88[12]=0 || (SCU90[5]=0 & SCU90[5:4]=0)
Y4	GPIOO5	ID/O8	Bidir	Hi-Z,Input	SCU88[13]=0 || (SCU90[5]=0 & SCU90[5:4]=0)
AA4	GPIOO6	ID/O8	Bidir	Hi-Z,Input	SCU88[14]=0 || (SCU90[5]=0 & SCU90[5:4]=0)
W4	GPIOO7	ID/O8	Bidir	Hi-Z,Input	SCU88[15]=0 || (SCU90[5]=0 & SCU90[5:4]=0)
V4	GPIOP0	ID/O8	Bidir	Hi-Z,Input	SCU88[16]=0 || (SCU90[5]=0 & SCU90[5:4]=0)
W5	GPIOP1	ID/O8	Bidir	Hi-Z,Input	SCU88[17]=0 || (SCU90[5]=0 & SCU90[5:4]=0)
AA5	GPIOP2	ID/O8	Bidir	Hi-Z,Input	SCU88[18]=0 || (SCU90[5]=0 & SCU90[5:4]=0)
AB5	GPIOP3	ID/O8	Bidir	Hi-Z,Input	SCU88[19]=0 || (SCU90[5]=0 & SCU90[5:4]=0)
Y6	GPIOP4	ID/O8	Bidir	Hi-Z,Input	SCU88[20]=0 || SCU90[28]=0
Y5	GPIOP5	ID/O8	Bidir	Hi-Z,Input	SCU88[21]=0 || SCU90[28]=0
W6	GPIOP6	ID/O8	Bidir	Hi-Z,Input	SCU88[22]=0 || SCU90[28]=0
V6	GPIOP7	ID/O8	Bidir	Hi-Z,Input	SCU88[23]=0 || SCU90[28]=0
A11	GPIOQ0	IDS/O8	Bidir	Hi-Z,Input	SCU90[16]=0
A10	GPIOQ1	IDS/O8	Bidir	Hi-Z,Input	SCU90[16]=0
A9	GPIOQ2	IDS/O8	Bidir	Hi-Z,Input	SCU90[17]=0
B9	GPIOQ3	IDS/O8	Bidir	Hi-Z,Input	SCU90[17]=0
N21	GPIOQ4	IDS/O8	Bidir	Hi-Z,Input	SCU90[27]=0
N22	GPIOQ5	IDS/O8	Bidir	Hi-Z,Input	SCU90[27]=0
B10	GPIOQ6	IDS/O8	Bidir	Hi-Z,Input	SCU2C[1]=0
N20	GPIOQ7	IDS/O8	Bidir	Hi-Z,Input	SCU2C[29]=0
AA19	GPIOR0	ID/O8	Bidir	Hi-Z	SCU88[24]=0
T19	GPIOR1	ID/O8	Bidir	Hi-Z	SCU88[25]=0
T17	GPIOR2	ID/O8	Bidir	Hi-Z,Input	SCU88[26]=0
Y19	GPIOR3	ID/O8	Bidir	Hi-Z,Input	SCU88[27]=0
W19	GPIOR4	ID/O8	Bidir	Hi-Z,Input	SCU88[28]=0

Ball	Name	I/O	Type	Power-Up	Multi-function Control
V19	GPIOR5	ID/O8	Bidir	Hi-Z,Input	SCU88[29]=0
D8	GPIOR6	ID/O8	Bidir	Hi-Z,Input	SCU88[30]=0
E10	GPIOR7	ID/O8	Bidir	Hi-Z,Input	SCU88[31]=0
V20	GPIOS0	ID/O8	Bidir	Hi-Z,Input	SCU8C[0]=0
U19	GPIOS1	ID/O8	Bidir	Hi-Z,Input	SCU8C[1]=0
R18	GPIOS2	ID/O8	Bidir	Hi-Z,Input	SCU8C[2]=0
P18	GPIOS3	ID/O8	Bidir	Hi-Z,Input	SCU8C[3]=0
R19	GPIOS4	ID/O8	Bidir	Hi-Z,Input	SCU8C[4]=0 || SCU94[1:0]=0
W20	GPIOS5	ID/O8	Bidir	Hi-Z,Input	SCU8C[5]=0 || SCU94[1:0]=0
U20	GPIOS6	ID/O8	Bidir	Hi-Z,Input	SCU8C[6]=0 || SCU94[1:0]=0
AA20	GPIOS7	ID/O8	Bidir	Hi-Z,Input	SCU8C[7]=0 || SCU94[1:0]=0
B5	GPIOT0	ID/Op	Bidir	Hi-Z	SCUA0[0]=1
E9	GPIOT1	ID/Op	Bidir	Hi-Z	SCUA0[1]=1
F9	GPIOT2	ID/Op	Bidir	Hi-Z	SCUA0[2]=1
A5	GPIOT3	ID/Op	Bidir	Hi-Z	SCUA0[3]=1
E7	GPIOT4	ID/O8	Bidir	Hi-Z	SCUA0[4]=1
D7	GPIOT5	ID/O8	Bidir	Hi-Z	SCUA0[5]=1
B2	GPIOT6	ID/Op	Bidir	Hi-Z	SCUA0[6]=1
B1	GPIOT7	ID/Op	Bidir	Hi-Z	SCUA0[7]=1
A2	GPIOU0	ID/Op	Bidir	Hi-Z	SCUA0[8]=1
B3	GPIOU1	ID/Op	Bidir	Hi-Z	SCUA0[9]=1
D5	GPIOU2	ID/O8	Bidir	Hi-Z	SCUA0[10]=1
D4	GPIOU3	ID/O8	Bidir	Hi-Z	SCUA0[11]=1
B4	GPIOU4	IR/O8	Bidir	Hi-Z	SCUA0[12]=1
A4	GPIOU5	ID/O8	Bidir	Hi-Z	SCUA0[13]=1
A3	GPIOU6	IR/O8	Bidir	Hi-Z	SCUA0[14]=1
D6	GPIOU7	IR/O8	Bidir	Hi-Z	SCUA0[15]=1
C5	GPIOV0	IR/O8	Bidir	Hi-Z	SCUA0[16]=1
C4	GPIOV1	IR/O8	Bidir	Hi-Z	SCUA0[17]=1
C2	GPIOV2	IR/O8	Bidir	Hi-Z	SCUA0[18]=1
C1	GPIOV3	ID/O8	Bidir	Hi-Z	SCUA0[19]=1
C3	GPIOV4	IR/O8	Bidir	Hi-Z	SCUA0[20]=1
D1	GPIOV5	IR/O8	Bidir	Hi-Z	SCUA0[21]=1
D2	GPIOV6	IR/O8	Bidir	Hi-Z	SCUA0[22]=1
E6	GPIOV7	IR/O8	Bidir	Hi-Z	SCUA0[23]=1
F4	GPIW0	IR	Bidir	Hi-Z	SCUA0[24]=1
F5	GPIW1	IR	Bidir	Hi-Z	SCUA0[25]=1
E2	GPIW2	IR	Bidir	Hi-Z	SCUA0[26]=1
E1	GPIW3	IR	Bidir	Hi-Z	SCUA0[27]=1
F3	GPIW4	IR	Bidir	Hi-Z	SCUA0[28]=1
E3	GPIW5	IR	Bidir	Hi-Z	SCUA0[29]=1
G5	GPIW6	IR	Bidir	Hi-Z	SCUA0[30]=1
G4	GPIW7	IR	Bidir	Hi-Z	SCUA0[31]=1

Ball	Name	I/O	Type	Power-Up	Multi-function Control
F2	GPIX0	IR	Bidir	Hi-Z	SCUA4[0]=1
G3	GPIX1	IR	Bidir	Hi-Z	SCUA4[1]=1
G2	GPIX2	IR	Bidir	Hi-Z	SCUA4[2]=1
F1	GPIX3	IR	Bidir	Hi-Z	SCUA4[3]=1
H5	GPIX4	IR	Bidir	Hi-Z	SCUA4[4]=1
G1	GPIX5	IR	Bidir	Hi-Z	SCUA4[5]=1
H3	GPIX6	IR	Bidir	Hi-Z	SCUA4[6]=1
H4	GPIX7	IR	Bidir	Hi-Z	SCUA4[7]=1
R22	GPIOY0	ID/O8	Bidir	Hi-Z,Input	Strap[19]=0 & SCUA4[8]=0 & SCU94[10]=0
R21	GPIOY1	ID/O8	Bidir	Hi-Z,Input	Strap[19]=0 & SCUA4[9]=0 & SCU94[10]=0
P22	GPIOY2	ID/O8	Bidir	Hi-Z,Input	Strap[19]=0 & SCUA4[10]=0 & SCU94[11]=0
P21	GPIOY3	ID/O8	Bidir	Hi-Z,Input	Strap[19]=0 & SCUA4[11]=0 & SCU94[11]=0
M18	GPIOY4	IDS/O8	Bidir	Hi-Z	SCUA4[12]=0
M19	GPIOY5	IDS/O8	Bidir	Hi-Z	SCUA4[13]=0
M20	GPIOY6	IDS/O8	Bidir	Hi-Z	SCUA4[14]=0
P20	GPIOY7	IDS/O8	Bidir	Hi-Z	SCUA4[15]=0
Y20	GPIOZ0	ID/O8	Bidir	Hi-Z,Input	SCU90[31]=0 & SCUA4[16]=0 & Strap[19]=0
AB20	GPIOZ1	ID/O8	Bidir	Hi-Z,Input	SCU90[31]=0 & SCUA4[17]=0 & Strap[19]=0
AB21	GPIOZ2	ID/O8	Bidir	Hi-Z,Input	SCU90[31]=0 & SCUA4[18]=0 & Strap[19]=0
AA21	GPIOZ3	ID/O8	Bidir	Hi-Z,Input	SCU90[31]=0 & SCUA4[19]=0 & Strap[19]=0
U21	GPIOZ4	ID/O8	Bidir	Hi-Z,Input	SCU90[31]=0 & (SCUA4[20]=0 || SCU94[1:0]=0)
W22	GPIOZ5	ID/O8	Bidir	Hi-Z,Input	SCU90[31]=0 & (SCUA4[21]=0 || SCU94[1:0]=0)
V22	GPIOZ6	ID/O8	Bidir	Hi-Z,Input	SCU90[31]=0 & (SCUA4[22]=0 || SCU94[1:0]=0)
W21	GPIOZ7	ID/O8	Bidir	Hi-Z,Input	SCU90[31]=0 & (SCUA4[23]=0 || SCU94[1:0]=0)
Y21	GPIOAA0	ID/O8	Bidir	Hi-Z,Input	SCU90[31]=0 & SCUA4[24]=0
V21	GPIOAA1	ID/O8	Bidir	Hi-Z,Input	SCU90[31]=0 & SCUA4[25]=0
Y22	GPIOAA2	ID/O8	Bidir	Hi-Z,Input	SCU90[31]=0 & SCUA4[26]=0
AA22	GPIOAA3	ID/O8	Bidir	Hi-Z,Input	SCU90[31]=0 & SCUA4[27]=0
U22	GPIOAA4	ID/O8	Bidir	Hi-Z,Input	SCU90[31]=0 & SCUA4[28]=0
T20	GPIOAA5	ID/O8	Bidir	Hi-Z,Input	SCU90[31]=0 & SCUA4[29]=0
N18	GPIOAA6	ID/O8	Bidir	Hi-Z,Input	SCU90[31]=0 & SCUA4[30]=0
P19	GPIOAA7	ID/O8	Bidir	Hi-Z,Input	SCU90[31]=0 & SCUA4[31]=0
N19	GPIOAB0	ID/O8	Bidir	Hi-Z,Input	SCU90[31]=0 & (SCUA8[0]=0 || SCU94[1:0]=0)
T21	GPIOAB1	ID/O8	Bidir	Hi-Z,Input	SCU90[31]=0 & (SCUA8[1]=0 || SCU94[1:0]=0)
T22	GPIOAB2	ID/O8	Bidir	Hi-Z,Input	SCUA8[2]=0
R20	GPIOAB3	ID/O8	Bidir	Hi-Z,Input	SCUA8[3]=0
G21	GPIOAC0	IUS/O12	Bidir	Hi-Z	Strap[25]=0 & SCUAC[0]=0
G20	GPIOAC1	IUS/O12	Bidir	Hi-Z	Strap[25]=0 & SCUAC[1]=0
D22	GPIOAC2	IUS/O12	Bidir	Hi-Z	Strap[25]=0 & SCUAC[2]=0
E22	GPIOAC3	IUS/O12	Bidir	Hi-Z	Strap[25]=0 & SCUAC[3]=0
C22	GPIOAC4	IDS/O12	Bidir	Hi-Z	Strap[25]=0 & SCUAC[4]=0
F21	GPIOAC5	IUS/O12	Bidir	Hi-Z	Strap[25]=0 & SCUAC[5]=0
F22	GPIOAC6	IUS/O12	Bidir	Hi-Z	Strap[25]=0 & SCUAC[6]=0

Ball	Name	I/O	Type	Power-Up	Multi-function Control
G22	GPIOAC7	IDS/O12	Bidir	Hi-Z	Strap[25]=0 & SCUAC[7]=0
Misc. Interface					
D13	TIMER3	ID/O16	Out	Hi-Z	SCU80[2]=1 & SCU80[15]=0
E13	TIMER4	ID/O16	Out	Hi-Z	SCU80[3]=1
C14	TIMER5	IDS/O8	Out	Hi-Z	SCU80[4]=1 & SCU90[6]=0 & SCU90[22]=0
A13	TIMER6	IDS/O8	Out	Hi-Z	SCU80[5]=1 & SCU90[6]=0 & SCU90[22]=0
C13	TIMER7	IDS/O8	Out	Hi-Z	SCU80[6]=1 & SCU90[6]=0 & SCU90[2]=0
B13	TIMER8	IDS/O8	Out	Hi-Z	SCU80[7]=1 & SCU90[6]=0 & SCU90[2]=0
B14	MAC1LINK	ID/O16	In	Hi-Z	SCU80[0]=1
D14	MAC2LINK	ID/O16	In	Hi-Z	SCU80[1]=1
J20	USBCKI	ID/O8	In	Hi-Z	Strap[23]=1
B10	OSCCLK	IDS/O8	Out	Hi-Z	SCU2C[1]=1
N20	PEWAKE#	IDS/O8	Out	Hi-Z	SCU2C[29]=1
U19	BMCINT	ID/O8	Out	Hi-Z	SCU8C[1]=1 & SCU94[1:0]=0
T22	WDTRST1	ID/O8	Out	Hi-Z	SCUA8[2]=1 & SCU94[1:0]=0
R20	WDTRST2	ID/O8	Out	Hi-Z	SCUA8[3]=1 & SCU94[1:0]=0

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
LPCRST#	:	Reset input pin
LHRST#	:	Reset input pin
ESPICS#	:	eSPI chip select input pin
ESPIRST#	:	Reset input pin
PLTRST#	:	eSPI platform reset command input
WDT_Full	:	Watchdog full reset event
WDT_SOC	:	Watchdog SOC reset event
WDT_ARM	:	Watchdog ARM reset event

Reset Function List	
ResetSOC[n]	EXTRST#[n] || WDT_SOC[n]
ResetFunc1	SRST#
ResetFunc2	SRST# || WDT_Full
ResetFunc3	SRST# || WDT_Full || ResetSOC[0] || WDT_ARM
ResetFunc4	SRST# || WDT_Full || ResetSOC[25]
ResetFunc5	SRST# || ResetSOC[1] || SCU100[1]
ResetFunc6	SRST# || WDT_Full || ResetSOC[20] || SCU04[23]
ResetFunc7	SRST# || WDT_Full || ResetSOC[12] || SCU04[4]
ResetFunc8	SRST# || WDT_Full || ResetSOC[3] || SCU04[1]
ResetFunc9	SRST# || WDT_Full || ResetSOC[7] || SCU04[13]
ResetFunc10	SRST# || WDT_Full || ResetSOC[7] || SCU04[13] || SCUD4[5]
ResetFunc14	((ResetSOC[16] || SCU04[4]) & SCU2C[7]) || (PERST# & !SCU2C[7])
ResetFunc15	SRST# || WDT_Full || ResetSOC[4] || SCU04[2]
ResetFunc16	SRST# || WDT_Full || ResetSOC[19] || SCU04[22]
ResetFunc17	SCU70[14] ? LPCRST# : PERST#
ResetFunc18	SRST# || WDT_Full || ResetSOC[13] || SCU04[5]
ResetFunc19	!LHCR0[0] || LHRST#
ResetFunc20	!SCU90[30] || LHRST#
ResetFunc21	ESPICS#
ResetFunc22	HICR9[4] ? (SCU70[14] ? LPCRST# : PERST#) : (SRST# || WDT_Full)

ResetFunc23	HICR9[5] ? (SCU70[14] ? LPCRST# : PERST#) : (SRST# || WDT_Full)
ResetFunc24	HICR9[6] ? (SCU70[14] ? LPCRST# : PERST#) : (SRST# || WDT_Full)
ResetFunc25	HICR9[7] ? (SCU70[14] ? LPCRST# : PERST#) : (SRST# || WDT_Full)
ResetFunc26	SRST# || WDT_Full || (!SCU48[28] & (SCU04[11] || ResetSOC[5]) & (SCU04[12] || ResetSOC[6]))
ResetFunc27	SRST# || WDT_Full || ResetSOC[5] || SCU04[11]
ResetFunc28	SRST# || WDT_Full || ResetSOC[6] || SCU04[12]
ResetFunc31	SRST# || WDT_Full || ResetSOC[22] || SCU04[24] || PERST#
ResetFunc32	SRST# || WDT_Full || ResetSOC[15] || SCU04[18]
ResetFunc33	SRST# || ResetSOC[2] || (SCU04[0] & WDT_Full)
ResetFunc34	SRST# || WDT_Full || ResetSOC[3] || SCU04[1] || PERST#
ResetFunc35	PERST#
ResetFunc36	PERST# || SCU180[7]
ResetFunc37	SRST# || WDT_Full || ResetSOC[18] || SCU04[10]
ResetFunc38	SRST# || WDT_Full || ResetSOC[17] || SCU04[9]
ResetFunc39	SRST# || WDT_Full || ResetSOC[14] || SCU04[16]
ResetFunc40	SRST# || WDT_Full || ResetSOC[9] || SCU04[15]
ResetFunc41	SRST# || WDT_Full || ResetSOC[10] || SCU04[3] || (SCU94[14:13]!=0)
ResetFunc42	SRST# || WDT_Full || ResetSOC[8] || SCU04[14]
ResetFunc43	SRST# || WDT_Full || ResetSOC[8] || SCU04[14] || SCU90[29]
ResetFunc44	SRST# || WDT_Full || ResetSOC[8] || SCU04[14] || !SCU90[29]
ResetFunc45	SRST# || WDT_Full || ResetSOC[10] || SCU04[3] || (SCU94[14:13]=0)
ResetFunc46	SRST# || WDT_Full || ResetSOC[10] || SCU04[3] || (SCU94[14:13]!=1)
ResetFunc47	SRST# || WDT_Full || ResetSOC[10] || SCU04[3] || !SCU94[14]
ResetFunc48	SRST# || WDT_Full || ResetSOC[11] || SCU04[6]
ResetFunc54	SRST# || WDT_Full || ResetSOC[23] || SCU04[25] || PERST#
ResetFunc55	SRST# || WDT_Full || ResetSOC[21]
ResetFunc56	SRST# || WDT_Full || ResetSOC[24]

6.1 Function Level Reset Source Table

Function Module	Reset Source
AHB Bus Controller	ResetFunc1 || ResetFunc8
ARM CPU	ResetFunc3
Coprocessor CPU	ResetFunc5
AHB to SDRAM Bridge	ResetFunc8
AHB to APB Bridge	ResetFunc8
SDRAM Memory Controller	ResetFunc33
Interrupt Controller	ResetFunc3 || ResetFunc5 || ResetFunc2
Firmware SPI Memory Controller	ResetFunc1 || ResetFunc3
SPI1/SPI2 Flash Controller	ResetFunc1 || ResetFunc56
10/100/1G Ethernet MAC Controller (MAC1)	ResetFunc2 || ResetFunc26 || ResetFunc27

10/100/1G Ethernet MAC Controller (MAC2)	ResetFunc2 || ResetFunc26 || ResetFunc28
USB1.1 UHCI Host Controller	ResetFunc40
USB2.0 EHCI Host Controller (EHCI1)	ResetFunc2 || ResetFunc42 || ResetFunc44
USB2.0 EHCI Host Controller (EHCI2)	ResetFunc2 || ResetFunc45 || ResetFunc47
USB2.0 Virtual Hub Controller	ResetFunc2 || ResetFunc42 || ResetFunc43
USB1.1 HID Controller	ResetFunc41
System Control Unit Registers	ResetFunc2
Hash & Crypto Engine	ResetFunc7
JTAG Master Controller	ResetFunc16
SOC Display Controller	ResetFunc9 || ResetFunc10
Memory Integrity Check Controller	ResetFunc32
X-DMA Controller	ResetFunc35 || ResetFunc54
MCTP Controller	ResetFunc35 || ResetFunc31
eSPI Controller	ResetFunc2 || ResetFunc18 || ResetFunc21
ADC Controller	ResetFunc6
APB to PCIe Bus Bridge	ResetFunc35 || ResetFunc31
Battery Backed SRAM	ResetFunc4
Video Engine	ResetFunc48 || ResetFunc10
SRAM Memory Buffer	ResetFunc4
SD/SDIO Host Controller	ResetFunc39
GPIO Controller	ResetFunc2 || ResetFunc55
Real Time Clock	ResetFunc1 || ResetFunc4
Timer Controller	ResetFunc2 || ResetFunc4
Watchdog Timer	ResetFunc1 || ResetFunc4
UART Controller (UART1)	ResetFunc22
UART Controller (UART2)	ResetFunc23
UART Controller (UART3)	ResetFunc24
UART Controller (UART4)	ResetFunc25
UART Controller (UART5)	ResetFunc4
UART DMA	ResetFunc4 || ResetFunc3
I2C/SMBus Controller	ResetFunc15
PWM & Fan Tacho Controller	ResetFunc38
Virtual UART	ResetFunc2 || ResetFunc17
LPC Device Controller	ResetFunc2 || ResetFunc18 || ResetFunc17
LPC Host Controller	ResetFunc2 || ResetFunc19
LPC+ Controller	ResetFunc20
SuperIO Controller	ResetFunc2 || ResetFunc17
System Wake-Up Control	ResetFunc2 || ResetFunc18 || ResetFunc17
MailBox Controller	ResetFunc2
PECI Controller	ResetFunc37
PCI Express Controller	SRST# || PERST#
PCI Bus Controller (P-Bus)	ResetFunc2 || ResetFunc35 || ResetFunc8
XDMA Controller	ResetFunc54 || ResetFunc35

VGA Display Controller	ResetFunc2 ‖ ResetFunc35
2D Graphics Engine	ResetFunc14
P-Bus to AHB Bridge	ResetFunc34 ‖ ResetFunc35
P-Bus to LPC Bridge	ResetFunc34 ‖ ResetFunc35
Message Signaled Interrupts	ResetFunc36 ‖ ResetFunc31

6.2 Register Level Reset Source Table

Registers	Reset Source
AHB Bus Controller	
AHBC00	ResetFunc4
AHBC40-AHBC4C	ResetFunc1
AHBC5C-AHBC7C	–
AHBC80-AHBC94	ResetFunc4
Firmware SPI Memory Controller	
FMC00	ResetFunc2 ‖ (!FMCA8[0] & ResetFunc4)
FMC04	ResetFunc2 ‖ (!FMCA8[1] & ResetFunc4)
FMC08-FMC0C	ResetFunc4
FMC10	ResetFunc2 ‖ (!FMCA8[2] & ResetFunc4)
FMC14	ResetFunc2 ‖ (!FMCA8[3] & ResetFunc4)
FMC18	ResetFunc2 ‖ (!FMCA8[4] & ResetFunc4)
FMC30	ResetFunc2 ‖ (!FMCA8[5] & ResetFunc4)
FMC34	ResetFunc2 ‖ (!FMCA8[6] & ResetFunc4)
FMC38	ResetFunc2 ‖ (!FMCA8[7] & ResetFunc4)
FMC54	–
FMC80	ResetFunc4
FMC84-FMC90	–
FMC94	ResetFunc4
FMC9C	ResetFunc2
FMCA0	ResetFunc2 ‖ (!FMCA8[9] & ResetFunc4)
FMCA4	ResetFunc2 ‖ (!FMCA8[10] & ResetFunc4)
FMCA8	ResetFunc2
FMCAC	ResetFunc4
FMCB0-FMCC0	–
SPI1/SPI2 Flash Controller	
SPIR00	ResetFunc2 ‖ (!SPIRA8[0] & ResetFunc56)
SPIR04[31]	ResetFunc56
SPIR04[30:0]	ResetFunc2 ‖ (!SPIRA8[1] & ResetFunc56)
SPIR08-SPIR0C	ResetFunc56
SPIR10	ResetFunc2 ‖ (!SPIRA8[2] & ResetFunc56)
SPIR14	ResetFunc2 ‖ (!SPIRA8[3] & ResetFunc56)

Registers	Reset Source
SPIR30	ResetFunc2 || (!SPIRA8[5] & ResetFunc56)
SPIR34	ResetFunc2 || (!SPIRA8[6] & ResetFunc56)
SPIR54	–
SPIR94	ResetFunc56
SPIRA0	ResetFunc2 || (!SPIRA8[9] & ResetFunc56)
SPIRA4	ResetFunc2 || (!SPIRA8[10] & ResetFunc56)
SPIRA8	ResetFunc2
SPIRAC	ResetFunc56
SPIRB0-SPIRC0	–
Memory Integrity Check Controller	
MIC00-MIC04	–
MIC08	ResetFunc32
MIC0C-MIC10	–
MIC14[31:16]	ResetFunc32 (DRAM 256M mode)
MIC14[15:0]	!MIC0C[28] (DRAM 256M mode)
MIC14[31:18]	ResetFunc32 (DRAM 1G mode)
MIC14[17:0]	!MIC24[31] (DRAM 1G mode)
MIC18[31:28]	ResetFunc32
MIC18[27:0]	–
MIC1C[31:28]	ResetFunc32
MIC1C[27:0]	–
MIC20-MIC24	–
10/100/1G Ethernet MAC Controller	
MAC00	ResetFunc27/ResetFunc28 || MAC50[31]
MAC04	ResetFunc27/ResetFunc28
MAC08-MAC0C	ResetFunc2
MAC10-MAC14	ResetFunc27/ResetFunc28
MAC20-MAC24	ResetFunc27/ResetFunc28
MAC2C-MAC38	ResetFunc27/ResetFunc28
MAC40	ResetFunc2
MAC48-MAC4C	ResetFunc27/ResetFunc28
MAC50[31]	ResetFunc2
MAC50[19]	ResetFunc2
MAC50[18:10]	ResetFunc27/ResetFunc28 || MAC50[31]
MAC50[9:8]	ResetFunc2
MAC50[7:0]	ResetFunc27/ResetFunc28 || MAC50[31]
MAC54	ResetFunc27/ResetFunc28 || MAC50[31]
MAC58	ResetFunc2
MAC60-MAC64	ResetFunc27/ResetFunc28 || MAC50[31]
MAC68[0]	ResetFunc27/ResetFunc28
MAC68[1]	ResetFunc27/ResetFunc28 || MAC50[31]
MAC68[2]	ResetFunc27/ResetFunc28
MAC68[4:3]	ResetFunc27/ResetFunc28 || MAC50[31]

Registers	Reset Source
MAC68[31:8]	ResetFunc27/ResetFunc28
MAC6C	ResetFunc27/ResetFunc28
MAC70[3:0]	ResetFunc27/ResetFunc28
MAC70[15]	ResetFunc27/ResetFunc28 || MAC50[31]
MAC74[1:0]	ResetFunc27/ResetFunc28 || MAC50[31]
MAC74[2]	ResetFunc27/ResetFunc28 || MAC50[31] || !MAC70[2]
MAC74[3]	ResetFunc27/ResetFunc28 || MAC50[31] || !MAC70[3]
MAC78-MAC8C	ResetFunc27/ResetFunc28
MACD0-MACF4	ResetFunc27/ResetFunc28
MACF8-MAC14C	–
MAC150	–
MAC154[31:8]	–
MAC154[4:0]	ResetFunc27/ResetFunc28
MAC158	–
USB2.0 Virtual Hub Controller	
HUB00	ResetFunc43
HUB04	ResetFunc43 || USB_BusReset
HUB08-HUB14	ResetFunc43
HUB18-HUB1C	ResetFunc43 || USB_BusReset || HUB20[9]
HUB20	ResetFunc43
HUB24-HUB28	–
HUB2C	ResetFunc43 || HUB20[0] || USB_BusReset
HUB30[2:0]	ResetFunc43 || HUB20[0] || USB_BusReset
HUB30[31:3]	–
HUB34	–
HUB38-HUB3C	ResetFunc43
DEV00[14:8,0]	ResetFunc43 || USB_BusReset
DEV00[6:1]	ResetFunc43
DEV04	ResetFunc43 || USB_BusReset || HUB20[5..1]
DEV08[2:0]	ResetFunc43 || USB_BusReset || HUB20[5..1]
DEV08[31:3]	–
DEV0C	–
EPP00	ResetFunc43
EPP04[3:0]	ResetFunc43
EPP04[31:4]	ResetFunc43 || USB_BusReset || HUB20[9] || HUB20[5..1] || EPP00[0]
EPP08	–
EPP0C	ResetFunc43 || USB_BusReset || HUB20[9] || HUB20[5..1] || EPP00[0]
USB2.0 EHCI Host Controller	
EHCI00	–
EHCI04	ResetFunc2
EHCI08	ResetFunc2
EHCI20	ResetFunc44 || EHCI20[1]
EHCI24	ResetFunc44 || EHCI20[1]

Registers	Reset Source
EHCI28	ResetFunc44 || EHCI20[1]
EHCI2C	ResetFunc44 || EHCI20[1]
EHCI34	–
EHCI38	–
EHCI60	ResetFunc44 || EHCI20[1]
EHCI64[0]	ResetFunc44
EHCI64[31:1]	ResetFunc44 || EHCI20[1]
EHCI80	ResetFunc2
EHCI84	ResetFunc2
EHCI8C	–
USB1.1 UHCI Host Controller	
UHCI00	ResetFunc40
UHCI04-UHCI08	ResetFunc40 || UHCI00[2] || UHCI00[1]
UHCI0C	–
UHCI40	ResetFunc40
UHCI80-UHCI84	ResetFunc40 || UHCI00[2] || UHCI00[1]
UHCI88-UHCI8C	ResetFunc40
USB1.1 HID Controller	
USBL00	ResetFunc41
USBL04	ResetFunc41 || USB_BusReset
USBL0C	–
USBL10	ResetFunc41
USBL14	ResetFunc41
USBL18[2:0]	ResetFunc41 || USB_BusReset
USBL18[31:3]	–
USBL1C[1:0]	ResetFunc41 || USB_BusReset
USBL1C[31:2]	–
USBL20[1:0]	ResetFunc41 || USB_BusReset
USBL20[31:2]	–
USBL24-USBL40	–
Interrupt Controller	
VIC00-VICE4	ResetFunc3
SVIC00-SVIC3C	ResetFunc2
CVIC00-CVIC3C	ResetFunc5
SDRAM Memory Controller	
MCR00-MCR70	ResetFunc33
MCR74-MCR7C	–
MCR80-MCR8C	ResetFunc33
MCR120	–
System Control Unit	
SCU00-SCU0C	ResetFunc2
SCU10[15:8,5:0]	ResetFunc2
SCU10[31:16,7:6]	–

Registers	Reset Source
SCU14	–
SCU18-SCU4C	ResetFunc2
SCU50-SCU6C	–
SCU70	ResetFunc1
SCU74	ResetFunc2
SCU78-SCU7C	–
SCU80-SCU1DC	ResetFunc2
Hash & Crypto Engine	
HACE00-HACE0C	–
HACE10	ResetFunc7
HACE18	–
HACE1C	ResetFunc7
HACE20-HACE2C	–
HACE30	ResetFunc7
HACE40	–
HACE4C	ResetFunc7
HACE50-HACE54	–
HACE58	ResetFunc7
HACE5C	–
HACE60	ResetFunc7
JTAG Master Controller	
JTAG00-JTAG18	ResetFunc16
SOC Display Controller	
GFX060	ResetFunc9
GFX064[7:0]	ResetFunc9
GFX064[31:8]	–
GFX068[4:0]	ResetFunc10
GFX068[7:6]	ResetFunc9
GFX068[31:8]	ResetFunc10
GFX06C[31:24]	ResetFunc9
GFX070-GFX07C	–
GFX080[1]	ResetFunc9
GFX080[29:4]	–
GFX084-GFX094	–
GFX098[1]	ResetFunc9
GFX098[29:4]	–
GFX09C	ResetFunc9
GFX0A0-GFX0A4	–
GFX0A8[1]	ResetFunc9
GFX0A8[29:4]	–
GFX0AC	–
GFX0B0[20:0]	–
GFX0B0[25:24]	ResetFunc9

Registers	Reset Source
GFX0B4	–
GFX0B8	ResetFunc10
GFX0BC	ResetFunc9
GFX0D0-GFX0FC	–
X-DMA Controller	
XDMA00-XDMA08	ResetFunc35
XDMA0C-XDMA20	ResetFunc54
XDMA24[3:0]	ResetFunc35
XDMA24[6:4]	ResetFunc54
XDMA24[31:25]	ResetFunc54
XDMA28	ResetFunc54
XDMA30[10:0]	ResetFunc35
XDMA30[13:11]	ResetFunc54
XDMA30[18:14]	ResetFunc35
XDMA30[22:19]	ResetFunc54
XDMA30[25:24]	ResetFunc35
XDMA30[27:26]	ResetFunc54
XDMA30[29:28]	ResetFunc35
XDMA30[31:30]	ResetFunc54
XDMA34[23:0]	ResetFunc35
XDMA34[29:24]	ResetFunc54
XDMA34[30]	ResetFunc35
XDMA38-XDMA54	–
XDMA60-XDMA6C	ResetFunc35
XDMA70-XDMA74	ResetFunc54
XDMA78	ResetFunc35
MCTP Controller	
MCTP00-MCTP18	ResetFunc31
ADC Controller	
ADC00-ADCC4	ResetFunc6
eSPI Controller	
ESPI000[0]	ResetFunc2 || ESPIRST#
ESPI000[1]	ResetFunc2
ESPI000[2]	ResetFunc2 || ESPIRST#
ESPI000[4:3]	ResetFunc2
ESPI000[6]	ResetFunc2 || ESPIRST#
ESPI000[7]	ResetFunc2
ESPI000[22:8]	ResetFunc18
ESPI000[31:24]	ResetFunc2
ESPI004[0]	–
ESPI004[8:4]	ResetFunc18
ESPI004[14:9]	ResetFunc18 || ESPIRST#
ESPI004[16]	ResetFunc18 || !ESPI000[24]

Registers	Reset Source
ESPI004[17]	ResetFunc18 || !ESPI000[25] || ESPIRST#
ESPI004[18]	ResetFunc18 || !ESPI000[26] || ESPIRST#
ESPI004[19]	ResetFunc18 || !ESPI000[27] || ESPIRST#
ESPI004[20]	ResetFunc18 || !ESPI000[28]
ESPI004[21]	ResetFunc18 || !ESPI000[29] || ESPIRST#
ESPI004[22]	ResetFunc18 || !ESPI000[30] || ESPIRST#
ESPI004[23]	ResetFunc18 || !ESPI000[31] || ESPIRST#
ESPI004[31:28]	ResetFunc18
ESPI008-ESPI010	ResetFunc18
ESPI014	ResetFunc2
ESPI018	–
ESPI020-ESPI024	ResetFunc18
ESPI028	–
ESPI030-ESPI034	ResetFunc18
ESPI038	–
ESPI040	ResetFunc18
ESPI044	ResetFunc2
ESPI048	–
ESPI050-ESPI054	ResetFunc18
ESPI058	–
ESPI060	ResetFunc18
ESPI064	ResetFunc2
ESPI068	–
ESPI070-ESPI078	ResetFunc18
ESPI084-ESPI08C	ResetFunc2
ESPI090-ESPI094	ResetFunc18
ESPI098[7:0]	ResetFunc2 || ESPIRST#
ESPI098[11:8]	ResetFunc2 || PLTRST#
ESPI098[23:16]	ResetFunc2 || ESPIRST#
ESPI098[27:24]	ResetFunc2 || PLTRST#
ESPI09C	ResetFunc2
ESPI0A0	ResetFunc2 || ESPIRST#
ESPI0A4	ResetFunc2 || ESPIRST# || PLTRST#
ESPI0A8-ESPI0B0	ResetFunc2 || ESPIRST#
ESPI0B4-ESPI0BC	ResetFunc2
ESPI100-ESPI12C	ResetFunc18
Battery Backed SRAM	
BSRAM00	ResetFunc4
Video Engine	
VR000-VR008	ResetFunc48
VR00C-VR024	–
VR02C[15:0]	ResetFunc48
VR02C[31:16]	–

Registers	Reset Source
VR030-VR034	–
VR038-VR03C	ResetFunc48
VR040-VR058	–
VR05C-VR064	ResetFunc48
VR068-VR06C	–
VR070-VR07C	ResetFunc48
VR080	–
VR084-VR08C	ResetFunc48
VR090-VR0A0	–
VR0A4-VR0B4	ResetFunc48
VR0B8-VR0D0	–
VR204-VR208	ResetFunc48
VR214-VR234	–
VR238-VR23C	ResetFunc48
VR240-VR254	–
VR25C-VR260	ResetFunc48
VR264-VR268	–
VR270-VR278	ResetFunc48
VR280	–
VR300-VR308	ResetFunc48
VR30C	–
VR310-VR328	ResetFunc48
VR330-VR364	–
VR3F0-VR3FC	–
SD/SDIO Host Controller	
SDIO*	ResetFunc39
GPIO Controller	
GPIO000-GPIO004	ResetFunc2 || (GPIO01C & ResetSOC[21])
GPIO008-GPIO018	ResetFunc55
GPIO01C	ResetFunc2
GPIO020-GPIO024	ResetFunc2 || (GPIO03C & ResetSOC[21])
GPIO028-GPIO038	ResetFunc55
GPIO03C	ResetFunc2
GPIO040-GPIO06C	ResetFunc55
GPIO070-GPIO074	ResetFunc2 || (GPIO0AC & ResetSOC[21])
GPIO078-GPIO07C	ResetFunc2 || (GPIO0FC & ResetSOC[21])
GPIO080-GPIO084	ResetFunc2 || (GPIO12C & ResetSOC[21])
GPIO088-GPIO08C	ResetFunc2 || (GPIO15C & ResetSOC[21])
GPIO090-GPIO0A8	ResetFunc55
GPIO0AC	ResetFunc2
GPIO0B0-GPIO0F8	ResetFunc55
GPIO0FC	ResetFunc2
GPIO100-GPIO128	ResetFunc55

Registers	Reset Source
GPIO12C	ResetFunc2
GPIO130-GPIO158	ResetFunc55
GPIO15C	ResetFunc2
GPIO160-GPIO188	ResetFunc55
GPIO18C	ResetFunc2
GPIO190-GPIO1B8	ResetFunc55
GPIO1BC	ResetFunc2
GPIO1C0-GPIO1D4	ResetFunc55
GPIO1E0-GPIO1E4	ResetFunc2 || (GPIO18C & ResetSOC[21])
GPIO1E8-GPIO1EC	ResetFunc2 || (GPIO1BC & ResetSOC[21])
GPIO200-GPIO3A0	ResetFunc55
Real Time Clock	
RTC00-RTC08	–
RTC10	ResetFunc1
RTC14	ResetFunc4
Timer Controller	
TMC00	–
TMC04-TMC0C	ResetFunc4
TMC10	–
TMC14-TMC1C	ResetFunc4
TMC20	–
TMC24-TMC30	ResetFunc4
TMC34	ResetFunc2
TMC38	ResetFunc4
TMC40	–
TMC44-TMC4C	ResetFunc4
TMC50	–
TMC54-TMC5C	ResetFunc4
TMC60	–
TMC64-TMC6C	ResetFunc4
TMC70	–
TMC74-TMC7C	ResetFunc4
TMC80	–
TMC84-TMC8C	ResetFunc4
UART DMA	
UDMA000-UDMA008	ResetFunc4
UDMA00C	–
UDMA020-UDMA040	ResetFunc4
UDMA044	UDMA020[0]
UDMA048	–
UDMA050	UDMA024[0]
UDMA054	ResetFunc4
UDMA058	–

Registers	Reset Source
UDMA060	ResetFunc4
UDMA064	UDMA020[1]
UDMA068	–
UDMA070	UDMA024[1]
UDMA074	ResetFunc4
UDMA078	–
UDMA080	ResetFunc4
UDMA084	UDMA020[2]
UDMA088	–
UDMA090	UDMA024[2]
UDMA094	ResetFunc4
UDMA098	–
UDMA0A0	ResetFunc4
UDMA0A4	UDMA020[3]
UDMA0A8	–
UDMA0B0	UDMA024[3]
UDMA0B4	ResetFunc4
UDMA0B8	–
Watchdog Timer	
WDT00-WDT04	ResetFunc4
WDT0C[31:2,0]	ResetFunc4
WDT0C[1]	ResetFunc1
WDT10[31:3,1:0]	ResetFunc1
WDT10[2]	ResetFunc4
WDT18-WDT1C	ResetFunc1
WDT20-WDT24	ResetFunc4
WDT2C[31:2,0]	ResetFunc4
WDT2C[1]	ResetFunc1
WDT30[31:3,1:0]	ResetFunc1
WDT30[2]	ResetFunc4
WDT38-WDT3C	ResetFunc1
WDT40-WDT44	ResetFunc4
WDT4C[31:2,0]	ResetFunc4
WDT4C[1]	ResetFunc1
WDT50[31:3,1:0]	ResetFunc1
WDT50[2]	ResetFunc4
WDT5C	ResetFunc1
PWM & Fan Tacho Controller	
PTCR00[11:0]	ResetFunc38
PTCR00[31:12]	–
PTCR04-PTCR0C	–
PTCR10[6:0]	–
PTCR10[7]	ResetFunc38

Registers	Reset Source
PTCR10[31:8]	–
PTCR14	–
PTCR18[6:0]	–
PTCR18[7]	ResetFunc38
PTCR18[31:8]	–
PTCR1C-PTCR2C	–
PTCR30-PTCR34	ResetFunc38
PTCR38-PTCR3C	–
PTCR40[11:0]	ResetFunc38
PTCR40[31:12]	–
PTCR44-PTCR4C	–
PTCR50[0]	ResetFunc38
PTCR50[31:1]	–
PTCR54	–
PTCR60	ResetFunc38
PTCR78	–
Virtual UART	
(Host)VUART00	ResetFunc2
(Host)VUART04-VUART1C	ResetFunc17
(Slave)VUART00-VUART1C	ResetFunc2
(Slave)VUART20[1:0]	ResetFunc2
(Slave)VUART20[3:2]	–
(Slave)VUART20[4]	ResetFunc17
(Slave)VUART20[5]	–
(Slave)VUART20[7:6]	ResetFunc17
(Slave)VUART24[1:0]	ResetFunc17
(Slave)VUART24[7:2]	–
(Slave)VUART28-VUART30	–
(Slave)VUART34	ResetFunc2
(Slave)VUART38	–
(Slave)VUART3C[5:0]	ResetFunc17
(Slave)VUART3C[7:6]	ResetFunc2
LPC Controller	
HICR0	ResetFunc18
HICR1[4:2]	ResetFunc18
HICR2[3:0]	ResetFunc18
HICR2[6:4]	ResetFunc2
HICR3	–
HICR4[4:0]	ResetFunc18
HICR4[6]	ResetFunc2
HICR4[7]	ResetFunc18
LADR3H	ResetFunc18
LADR3L	ResetFunc18

Registers	Reset Source
LADR12H	ResetFunc18
LADR12L	ResetFunc18
IDR1-IDR3	–
ODR1-ODR3	–
STR1[1:0]	ResetFunc2
STR1[2]	ResetFunc18
STR1[3]	ResetFunc2
STR1[7:4]	ResetFunc18
STR2[1:0]	ResetFunc2
STR2[2]	ResetFunc18
STR2[3]	ResetFunc2
STR2[7:4]	ResetFunc18
STR3[1:0]	ResetFunc2
STR3[2]	ResetFunc18
STR3[3]	ResetFunc2
STR3[7:4]	ResetFunc18
BTR0-BTR1	ResetFunc18
BTCSR0-BTCSR1	ResetFunc18
BTCR	ResetFunc18
BTDTR	–
BTIMSR	ResetFunc18
BTFVSR0-BTFVSR1	ResetFunc18
SIRQCR0[7:0]	ResetFunc18
SIRQCR1-SIRQCR3	ResetFunc18
HICR5[3:0]	ResetFunc2
HICR5[5:4]	ResetFunc18
HICR5[8:6]	ResetFunc2
HICR5[9]	ResetFunc18
HICR5[11:10]	ResetFunc2
HICR5[12]	ResetFunc18
HICR5[13]	ResetFunc2
HICR5[14]	ResetFunc18
HICR5[15]	ResetFunc2
HICR5[28:16]	–
HICR5[31:29]	ResetFunc2
HICR6[6:0]	ResetFunc2
HICR6[7]	ResetFunc17
HICR6[16]	ResetFunc2
HICR7-HICR8	ResetFunc2
SNPWADR	–
SNPWDR	ResetFunc2
HICR9	ResetFunc2
HICRA	ResetFunc2

Registers	Reset Source
LHCR0[0]	ResetFunc2
LHCR0[6:1]	ResetFunc18
LHCR0[7]	–
LHCR0[12:8]	ResetFunc2
LHCR0[22:14]	ResetFunc18
LHCR0[23]	–
LHCR1[1:0]	ResetFunc17
LHCR1[15]	ResetFunc2
LHCR1[31:16]	–
LHCR2-LHCR3	ResetFunc18
LHCR4-LHCR7	–
LHCR8	ResetFunc17
PCCR6	ResetFunc2
LHCRA	–
LHCRB	–
PCCR4-PCCR5	–
HICRB	ResetFunc2
HICRC[3:0]	ResetFunc2
HICRC[7:4]	–
HICRC[15:8]	ResetFunc2
HISR0-HISR1	ResetFunc2
LADR4	–
IDR4	–
ODR4	–
STR4[1:0]	ResetFunc2
STR4[2]	ResetFunc18
STR4[3]	ResetFunc2
STR4[7:4]	ResetFunc18
LSADR12	–
IDR5	–
ODR5	–
STR5[1:0]	ResetFunc2
STR5[2]	ResetFunc18
STR5[3]	ResetFunc2
STR5[7:4]	ResetFunc18
PCCR0[6:0]	ResetFunc2
PCCR0[13:8]	–
PCCR0[14]	ResetFunc2
PCCR0[19:16]	–
PCCR0[23:20]	ResetFunc2
PCCR0[30:24]	–
PCCR0[31]	ResetFunc2
PCCR1[21:0]	–

Registers	Reset Source
PCCR1[30:24]	ResetFunc2
PCCR2	ResetFunc2
PCCR3[7:0]	–
PCCR3[31:8]	ResetFunc2
iBTCR0[15:0]	ResetFunc2
iBTCR0[31:18]	–
iBTCR1	ResetFunc18
iBTCR2[14:0]	ResetFunc18
iBTCR2[15]	ResetFunc2
iBTCR2[31:16]	–
iBTCR3	ResetFunc17
iBTCR4	ResetFunc18
iBTCR5	–
iBTCR6[6:0]	ResetFunc17
iBTCR6[7]	ResetFunc2
SRUART1	(HICR9[4] & ResetFunc17) || (!HICR9[4] & ResetFunc2)
SRUART2	(HICR9[5] & ResetFunc17) || (!HICR9[5] & ResetFunc2)
SRUART3	(HICR9[6] & ResetFunc17) || (!HICR9[6] & ResetFunc2)
SRUART4	(HICR9[7] & ResetFunc17) || (!HICR9[7] & ResetFunc2)
SCR0SIO-SCR3SIO	ResetFunc2
SWCR0300-SWCR1F1C	ResetFunc2
ACPIE3E0	ResetFunc2
ACPIC1C0	ResetFunc2
ACPIB3B0	ResetFunc2
ACPIB7B4	ResetFunc2
MBXDAT0-MBXDATF	–
MBXSTS0-MBXSTS1	ResetFunc2
MBXBCR	ResetFunc2
MBXHCR	ResetFunc2
MBXBIE0-MBXBIE1	ResetFunc2
MBXHIE0-MBXHIE1	ResetFunc2
LPC+ Controller	
LPCP00-LPCP34	ResetFunc20
I2C/SMBus Controller	
I2CG08-I2CG0C	ResetFunc15
I2CD00	ResetFunc15
I2CD04-I2CD08	–
I2CD0C	ResetFunc15
I2CD10	ResetFunc15 || I2CD00[1:0]=0
I2CD14[15:12]	ResetFunc15
I2CD14[11:0]	ResetFunc15 || I2CD00[1:0]=0
I2CD18-I2CD28	–
PECI Controller	

Registers	Reset Source
PECI00[4:2,0]	ResetFunc37
PECI00[11:6]	ResetFunc37
PECI00[5,1]	–
PECI00[19:12]	–
PECI04	–
PECI08[0]	ResetFunc37
PECI0C-PECI10	–
PECI18[4:0]	ResetFunc37
PECI18[31:30]	–
PECI1C[4:0]	ResetFunc37
PECI20-PECI5C	–
2D Graphics Engine	
GER00-GER38	–
GER3C	ResetFunc14
GER40-GER60	ResetFunc14
PTR00-PTRFC	–

7 Memory Space Allocation Table

7.1 ARM Address Space Mapping

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)
0000:0000-0FFF:FFFF	256M	1/2/4	1/2/4	~~SDRAM (After Re-map)~~
1000:0000-15FF:FFFF	96M	1/2/4	1/2/4	~~Legacy BMC Static Memory~~
1600:0000-17FF:FFFF	32M	4	1/2/4	~~Legacy BMC Static Memory Controller (SMC)~~
1E60:0000-1E61:FFFF	128K	4	1/2/4	AHB Bus Controller (AHBC)
1E62:0000-1E62:FFFF	64K	4	1/2/4	Firmware SPI Memory Controller (FMC)
1E63:0000-1E63:0FFF	4K	4	1/2/4	SPI1 Memory Controller
1E63:1000-1E63:1FFF	4K	4	1/2/4	SPI2 Memory Controller
1E64:0000-1E65:FFFF	128K	4	1/2/4	Memory Integrity Check Controller (MIC)
1E66:0000-1E67:FFFF	128K	1/2/4	1/2/4	Ethernet MAC Controller #1 (MAC1)
1E68:0000-1E69:FFFF	128K	1/2/4	1/2/4	Ethernet MAC Controller #2 (MAC2)
1E6A:0000-1E6A:0FFF	4K	4	1/2/4	USB2.0 Hub Controller
1E6A:1000-1E6A:1FFF	4K	4	1/2/4	USB2.0 EHCI Host Controller #1
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 #2
1E6B:0000-1E6B:FFFF	64K	4	1/2/4	USB1.1 UHCI Host Controller
1E6C:0000-1E6D:FFFF	128K	4	1/2/4	Interrupt Controller (VIC)
1E6E:0000-1E6E:0FFF	4K	4	1/2/4	SDRAM 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:3000-1E6E:3FFF	4K	4	1/2/4	Hash & Crypto Engine (HACE)
1E6E:4000-1E6E:4FFF	4K	4	1/2/4	JTAG Master
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
1E6E:8000-1E6E:8FFF	4K	4	1/2/4	MCTP Controller
1E6E:9000-1E6E:9FFF	4K	4	1/2/4	ADC Voltage Monitor
1E6E:C000-1E6E:CFFF	4K	4	1/2/4	LPC+ Controller
1E6E:D000-1E6E:DFFF	4K	4	1/2/4	PCIE Host Controller
1E6E:E000-1E6E:EFFF	4K	4	1/2/4	e-SPI Controller
1E6E:F000-1E6E:FFFF	4K	4	1/2/4	Battery Backed SRAM
1E6F:0000-1E6F:0FFF	4K	4	1/2/4	APB to PCIe Bridge
1E70:0000-1E71:FFFF	128K	1/2/4	1/2/4	Video Engine
1E72:0000-1E73:FFFF	128K	1/2/4	1/2/4	36KB SRAM
1E74:0000-1E75:FFFF	128K	4	1/2/4	SD/SDIO/eMMC Controller
1E76:0000-1E77:FFFF	128K	4	1/2/4	2D Engine
1E78:0000-1E78:0FFF	4K	4	1/2/4	GPIO Controller
1E78:1000-1E78:1FFF	4K	4	1/2/4	Real-Time Clock (RTC)

to next page

from previous page

1E78:2000-1E78:2FFF	4K	4	1/2/4	Timer #1 ~ #8 Controller
1E78:3000-1E78:3FFF	4K	4	1/2/4	UART - #1 (LPC UART1)
1E78:4000-1E78:4FFF	4K	4	1/2/4	UART - #5 (BMC Debug)
1E78:5000-1E78:5FFF	4K	4	1/2/4	Watchdog Timer (WDT)
1E78:6000-1E78:6FFF	4K	4	1/2/4	PWM & Fan Tacho Controller
1E78:7000-1E78:7FFF	4K	4	1/2/4	Virtual UART (VUART)
1E78:8000-1E78:8FFF	4K	4	1/2/4	Pass Through UART (PUART)
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 PCI Bridge
1E78:D000-1E78:DFFF	4K	4	1/2/4	UART - #2 (LPC UART2)
1E78:E000-1E78:EFFF	4K	4	1/2/4	UART - #3 (LPC UART3)
1E78:F000-1E78:FFFF	4K	4	1/2/4	UART - #4 (LPC UART4)
1E79:E000-1E79:EFFF	4K	4	1/2/4	UART DMA
2000:0000-2FFF:FFFF	256M	1/2/4	1/2/4	BMC SPI Flash Memory
3000:0000-37FF:FFFF	128M	1/2/4	1/2/4	SPI1 Flash Memory
3800:0000-3FFF:FFFF	128M	1/2/4	1/2/4	SPI2 Flash Memory
6000:0000-6FFF:FFFF	256M	1/2/4	1/2/4	AHB Bus to LPC Bus Bridge
7000:0000-7FFF:FFFF	256M	1/2/4	1/2/4	AHB Bus to LPC+ Bus Bridge
8000:0000-BFFF:FFFF	1024M	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 In-compatible List to AST2400

AST2500	Legacy chips	IP Module
–	0000:0000-0FFF:FFFF	~~SDRAM (After Re-map)~~
–	1000:0000-15FF:FFFF	~~Legacy BMC Static Memory~~
–	1600:0000-17FF:FFFF	~~Legacy BMC Static Memory Controller (SMC)~~
1E63:0000-1E63:0FFF	1E63:0000-1E63:FFFF	SPI1 Memory Controller
1E63:1000-1E63:1FFF	–	SPI2 Memory Controller
1E6A:2000-1E6A:2FFF	–	~~USB2.0 Device Controller~~
1E6A:3000-1E6A:3FFF	–	USB2.0 Host Controller #2
1E6E:D000-1E6E:DFFF	–	PCIE Host Controller
1E6E:E000-1E6E:EFFF	–	e-SPI Controller
1E6E:F000-1E6E:FFFF	–	Battery Backed SRAM
1E6F:0000-1E6F:0FFF	–	APB to PCIe Bridge
1E79:E000-1E79:EFFF	–	UART DMA
3000:0000-37FF:FFFF	3000:0000-3FFF:FFFF	SPI1 Flash Memory
3800:0000-3FFF:FFFF	–	SPI2 Flash Memory
8000:0000-BFFF:FFFF	4000:0000-5FFF:FFFF	SDRAM

7.3 Coprocessor Address Space Mapping

Address Range	Size	Mapping Area
00:0000-0F:FFFF	1M	Default map to register area 0x00000000 - 0x000FFFFF
10:0000-1F:FFFF	1M	Default map to register area 0x00100000 - 0x001FFFFF
20:0000-2F:FFFF	1M	Default map to register area 0x1E600000 - 0x1E6FFFFF
30:0000-3F:FFFF	1M	Default map to register area 0x1E700000 - 0x1E7FFFFF
40:0000-4F:FFFF	1M	Remap area 1, address bit[31:21] is programmable
50:0000-5F:FFFF	1M	Remap area 2, address bit[31:21] is programmable
60:0000-6F:FFFF	1M	Remap area 3, address bit[31:21] is programmable
70:0000-7F:FFFF	1M	Remap area 4, address bit[31:21] is programmable,
80:0000-FF:FFFF	8M	Map to DRAM memory, address bit[31:23] is programmable

7.4 VGA Memory Space map to ARM Memory Space

VGA Size	Strap[3:2]	DRAM Size			
		128MB	256MB	512MB	1024MB
8MB	0	8780:0000 ~ 87FF:FFFF	8F80:0000 ~ 8FFF:FFFF	9F80:0000 ~ 9FFF:FFFF	BF80:0000 ~ BFFF:FFFF
16MB	1	8700:0000 ~ 87FF:FFFF	8F00:0000 ~ 8FFF:FFFF	9F00:0000 ~ 9FFF:FFFF	BF00:0000 ~ BFFF:FFFF
32MB	2	8600:0000 ~ 87FF:FFFF	8E00:0000 ~ 8FFF:FFFF	9E00:0000 ~ 9FFF:FFFF	BE00:0000 ~ BFFF:FFFF
64MB	3	8400:0000 ~ 87FF:FFFF	8C00:0000 ~ 8FFF:FFFF	9C00:0000 ~ 9FFF:FFFF	BC00:0000 ~ BFFF:FFFF

8 Interrupt Source Table

Table 72: ARM Interrupt Source Table

INT#	Description	Attribute
0	SDRAM interrupt	Sensitive high level trigger
1	MIC interrupt	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 Hub/Host interrupt	Sensitive high level trigger
6	X-DMA interrupt	Sensitive high level trigger
7	Video Engine interrupt	Sensitive high level trigger
8	LPC interrupt	Sensitive high level trigger
9	UART1 interrupt	Sensitive high level trigger
10	UART5 interrupt	Sensitive high level trigger
11	Reserved	Reserved
12	I2C/SMBus interrupt	Sensitive high level trigger
13	USB 1.1 HID/ USB2.0 Host2 interrupt	Sensitive high level trigger
14	USB 1.1 Host interrupt	Sensitive high level trigger
15	PECI interrupt	Sensitive high level trigger
16	Timer 1 interrupt	Both-edge trigger
17	Timer 2 interrupt	Both-edge trigger
18	Timer 3 interrupt	Both-edge trigger
19	FMC interrupt	Sensitive high level trigger
20	GPIO interrupt	Sensitive high level trigger
21	SCU interrupt	Sensitive high level trigger
22	RTC alarm interrupt	Sensitive high level trigger
23	eSPI interrupt	Sensitive high level trigger
24	Reserved	Reserved
25	Graphics CRT interrupt	Sensitive high level trigger
26	SD/SDIO interrupt	Sensitive high level trigger
27	WDT alarm interrupt	Sensitive high level trigger
28	PWM/Tachometer interrupt	Sensitive high level trigger
29	Graphics 2D interrupt	Sensitive high level trigger
30	System Wakeup Control	Sensitive high level trigger
31	ADC interrupt	Sensitive high level trigger
32	UART2 interrupt	Sensitive high level trigger
33	UART3 interrupt	Sensitive high level trigger
34	UART4 interrupt	Sensitive high level trigger
35	Timer 4 interrupt	Both-edge trigger
36	Timer 5 interrupt	Both-edge trigger
37	Timer 6 interrupt	Both-edge trigger
38	Timer 7 interrupt	Both-edge trigger

to next page

from previous page

39	Timer 8 interrupt	Both-edge trigger
40	SGPIO Master interrupt	Sensitive high level trigger
41	SGPIO Slave interrupt	Sensitive high level trigger
42	MCTP interrupt	Sensitive high level trigger
43	JTAG Master interrupt	Sensitive high level trigger
44	Host interrupt	Software trigger
45	Coprocessor interrupt	Software trigger
46	MailBox interrupt	Sensitive high level trigger
47	AHBC interrupt	Sensitive high level trigger
48	GPIOL1 direct input	Programmable high/low level trigger
49	GPIOL3 direct input	Programmable high/low level trigger
50	UART DMA interrupt	Sensitive high level trigger
51	Reserved	Reserved
52	Reserved	Reserved
53	Reserved	Reserved
54	Reserved	Reserved
55	Reserved	Reserved
56	Reserved	Reserved
57	Reserved	Reserved
58	Reserved	Reserved
59	SPI interrupt	Sensitive high level trigger
60	Reserved	Reserved
61	Reserved	Reserved
62	Reserved	Reserved
63	Reserved	Reserved

Table 73: Coprocessor Interrupt Source Table

INT#	Description	Attribute
0	SDRAM interrupt	Sensitive high level trigger
1	ARM interrupt	Software trigger
2	LPC interrupt	Sensitive high level trigger
3	UART1 interrupt	Sensitive high level trigger
4	UART2 interrupt	Sensitive high level trigger
5	UART3 interrupt	Sensitive high level trigger
6	UART4 interrupt	Sensitive high level trigger
7	UART5 interrupt	Sensitive high level trigger
8	Timer 1 interrupt	Both-edge trigger
9	Timer 2 interrupt	Both-edge trigger
10	Timer 3 interrupt	Both-edge trigger
11	Timer 4 interrupt	Both-edge trigger
12	Timer 5 interrupt	Both-edge trigger
13	Timer 6 interrupt	Both-edge trigger

to next page

from previous page

14	Timer 7 interrupt	Both-edge trigger
15	AHBC interrupt	Sensitive high level trigger
16	I2C/SMBus interrupt	Sensitive high level trigger
17	PECI interrupt	Sensitive high level trigger
18	FMC interrupt	Sensitive high level trigger
19	GPIO interrupt	Sensitive high level trigger
20	SCU interrupt	Sensitive high level trigger
21	RTC alarm interrupt	Sensitive high level trigger
22	WDT alarm interrupt	Sensitive high level trigger
23	PWM/Tachometer interrupt	Sensitive high level trigger
24	ADC interrupt	Sensitive high level trigger
25	SGPIO Master interrupt	Sensitive high level trigger
26	SGPIO Slave interrupt	Sensitive high level trigger
27	eSPI interrupt	Sensitive high level trigger
28	MCTP interrupt	Sensitive high level trigger
29	JTAG Master interrupt	Sensitive high level trigger
30	Reserved	Reserved
31	LPC Host interrupt	Software trigger

Table 74: System LPC Interrupt Source Table

INT#	Description	Attribute
0	GPIO interrupt	Sensitive high level trigger
1	I2C/SMBus interrupt	Sensitive high level trigger
2	PECI interrupt	Sensitive high level trigger
3	RTC alarm interrupt	Sensitive high level trigger
4	WDT alarm interrupt	Sensitive high level trigger
5	SD/SDIO interrupt	Sensitive high level trigger
6	PWM/Tachometer interrupt	Sensitive high level trigger
7	ADC interrupt	Sensitive high level trigger
8	Timer 5 interrupt	Both-edge trigger
9	Timer 6 interrupt	Both-edge trigger
10	Timer 7 interrupt	Both-edge trigger
11	Timer 8 interrupt	Both-edge trigger
12	SGPIO Master interrupt	Sensitive high level trigger
13	SGPIO Slave interrupt	Sensitive high level trigger
14	SPI interrupt	Sensitive high level trigger
15	Reserved	Reserved
16	ARM interrupt	Software trigger
17	Coprocessor interrupt	Software trigger

9 Major Firmware Programming Change List

1. Hardware strap register (SCU70) programming method.
   - Write '1' to SCU70 can set the specific bit with value '1'. Write '0' has no effect.
   - Write '1' to SCU7C can clear the specific bit of SCU70 to value '0'. Write '0' has no effect.
2. Timer control register (TMC30) programming method.
   - Add an alternative way to programming TMC30, set and clear TMC30 by different address.
   - Write 0xAE to TMC38 to enable the separate programming mode.
   - Under separate programming mode, write '1' to TMC30 can set the specific bit with value '1', write '0' has no effect.
   - Under separate programming mode, write '1' to TMC3C can clear the specific bit of TMC30 to value '1', write '0' has no effect.
3. I2C/SMBus related programming change
   - Add a buffer power control bit at I2CG0C[0]. It must be set to '1' when buffer or DMA mode used.
   - Increase I2C SCL pulse width (I2CD04[19:12]) setting from 3 bits to 4 bits to increase frequency resolution, refer to page 742.
   - Modify I2C buffer mode maximum buffer size to dedicated 16 bytes buffer.
   - Add supporting I2C DMA mode to maximum 4K bytes. *(Refer to Application Note 2 for the limitation of DMA mode.)*
   - Add supporting I2C bus lock auto-release feature at I2CD00[17].
4. Add supporting register read/write logging to memory function. Refer to section 12.
5. Add support hardware heart beat LED output at dedicated pin. Refer to section 20.
6. ADC initial sequence change. Refer to section 29.
   - A global initial control bit has to be set before any other initial process.
   - Add a auto method to get Compensation Value.
7. JTAG Master controller programming change. Refer to section 25.
   - Add a bit to prevent state machine going to Idle/Run-Test state.
8. Add support 80h redirect to SGPIO (group A). Refer to section 36.
9. Modify the CS0# and CS1# mapping method when firmware 2nd boot mode is active.
   - In AST2400/AST2300, CS0# is not accessible when 2nd boot is active.
   - In AST2500 , CS0# and CS1# are swapped when 2nd boot is active. Including control registers and address space.
10. RSA memory allocation was changed. Refer to section 24.
11. Support 3 paths for host can access into BMC internal registers/DRAM/flash.
    - PCIe to AHB bridge (P2A): write can be disabled by registers SCU2C[25:22], and fully disabled by SCU180[1].
    - LPC to AHB bridge (L2A): can be disabled by LPC registers HICR5[10] (set to 0) and HICRB[6] (set to 1).
    - UART to AHB bridge : can be disabled by register SCU2C[10].
12. D2-PLL was default used by MAC as RGMII 125MHz and RMII/NCSI 50MHz clock source. If wants to use Graphics CRT, it has a 40MHz clock source (set at SCU2C[21]) that can support display resolution of 800x600. If it requires to support other resolution modes, then it must use external clock for RGMII and RMII reference clock source. And set SCU2C[21]=0, SCU48[31:29]=0, SCU08[8]=1.

9.1 Application Note 1: Firmware Programming Note for CLKIN=25MHz

1. The I2C clock rate would increase about 4.2% relative to CLKIN=24 MHz case. If the resulting clock rate is over the spec, it can be fine-tuned by adding 1 on the clock high (I2CD04[19:16]) or clock low (I2CD04[15:12]) parameter, but not both.
2. The clock rate for all below interfaces will also increase about 4.2%. But these interfaces are not very sensitive to the little increment on the clock rate. Customers can check if it is acceptable to tolerate the change, and determine if wants to change the register setting.
   - SPI, SD/eMMC, PWM, SGPIO, JTAG Master, PECI.
3. When use CLKIN=25MHz option, USBCKI clock input is required for USB functions. All USB controllers registers cannot be accessed when no USBCKI clock running, and may cause bus hang condition if written to a clock stopped USB controller.
   Firmware should modify the USB driver loading sequence. So the USB driver will only be loaded when USBCKI is active. And firmware should disable the USB clock enable setting at SCU0C immediately once detected USBCKI will be stopped. This can protect the bus to be locked.
4. The D2-PLL parameter at SCU1C/SCU13C/SCU140/SCU144 should be changed if use the "SOC Display Controller" function. The PLL formula C code as below:

```
#define FIN                 25.0
#define FREF_MIN            5
#define FREF_MAX            100
#define FVCO_MIN            1000
#define FVCO_MAX            2000
#define FOUT_MIN            16
#define FOUT_MAX            1000

#define N_MIN               1
#define N_MAX               32
#define M_MIN               20
#define M_MAX               256
#define P_MIN               1
#define P_MAX               64


void pllout(double ftarget)
{
    unsigned N, M, P, SIC, SIP, INIT, SCU1C, SCU13C, SCU140, SCU144;
    double fout, fvco, fref, fdiff, M_target;
    int good_duty;

    for(N = N_MIN;N <= N_MAX;N++){
       for(M = M_MIN;M <= M_MAX;M++){
          fvco = (double)FIN * ((double)M / (double)N);
          fref = (double)FIN/(double)N;
          if(fvco < FVCO_MIN || fvco > FVCO_MAX){
             continue;
          }
          if(fref < FREF_MIN || fref > FREF_MAX){
             continue;
          }
          // continuing to next page at this level
```

```
          for(P = P_MIN;P <= P_MAX;P++){
             fout = fvco / (double)P;
             fdiff = ftarget - fout;
             if(fdiff != 0.0){
                continue;
             }
             if(M <= 29){ SIC = 0x0C; SIP = 0x05; } else
             if(M <= 39){ SIC = 0x0F; SIP = 0x08; } else
             if(M <= 49){ SIC = 0x12; SIP = 0x0B; } else
             if(M <= 99){ SIC = 0x15; SIP = 0x0E; } else
                        { SIC = 0x18; SIP = 0x11; }

             if(P & 0x1){
                good_duty = 0;
             }else{
                good_duty = 1;
             }
             SCU1C = (M-1) | ((N-1) << 8) | ((P-1) << 13) | (SIC << 22) |
                     (SIP << 27);
             SCU13C = 0x2C << 5;
             SCU140 = 0;
             SCU144 = 0;
             if(good_duty){
                printf(", Good_DUTY");
             }
          }
       }
    }
}
```

9.2 Application Note 2: I2C DMA and UHCI limitation

There is a limitation when you enable I2C DMA mode or UHCI host controller.
This limitation is related to the 5 controllers listed below.

1. I2C (DMA mode)
2. UHCI host controller
3. SD/eMMC (DMA mode)
4. Port80 Snoop (DMA mode)
5. MCTP controller

Please be noted:

- I2C controller can be set as either DMA mode or byte mode or buffer mode.
- SD/eMMC can be set as either DMA mode or PIO mode.
- Port80 Snoop can be set as either DMA mode or FIFO mode.
- UHCI host controller and MCTP operate with DMA mode only.

The limitation of enabling I2C DMA mode is:

- If you set I2C as DMA mode, please do not enable UHCI host and MCTP, and please do not set SD/eMMC and Port80 Snoop as DMA mode.

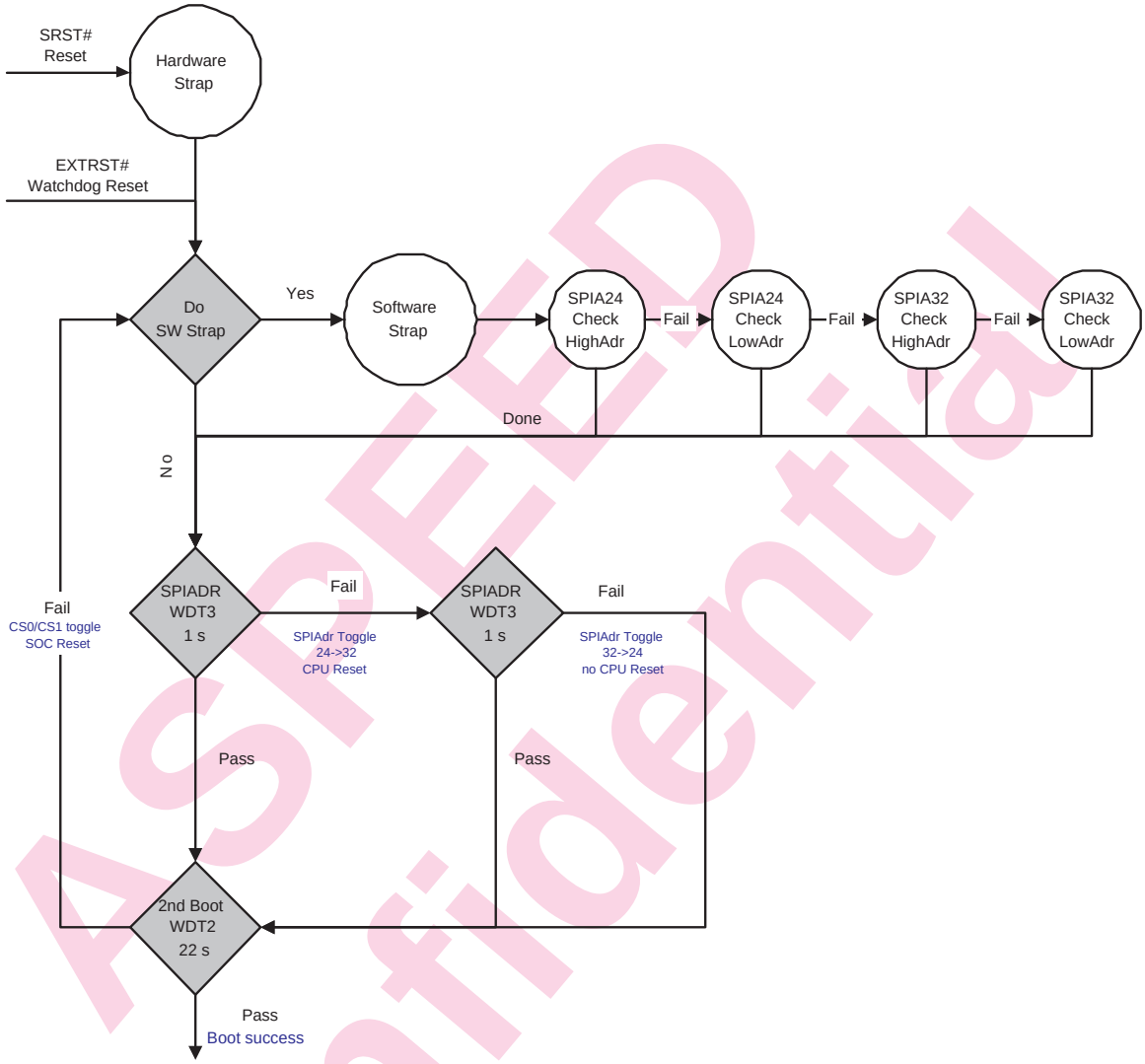
The limitation of enabling UHCI host controller is:

- If you enable UHCI host controller, please do not enable MCTP, and please do not set I2C, SD/eMMC and Port80 Snoop as DMA mode.

Except DMA mode, SD/eMMC supports PIO mode, Port80 Snoop supports FIFO mode, I2C supports byte mode (or buffer mode), the following combination can work properly.

1. You can enable I2C DMA if you set SD/eMMC to PIO mode and set Port80 snoop to FIFO mode. It means the combination of I2C DMA + SD/eMMC (PIO mode) + Port80 snoop (FIFO mode) is o.k. However, UHCI host controller and MCTP still can not be enabled under this combination.
2. You can enable UHCI host controller if you set SD/eMMC to PIO mode and set Port80 snoop to FIFO mode and set I2C to byte mode (or buffer mode). It means the combination of UHCI + SD/eMMC (PIO mode) + Port80 snoop (FIFO mode) + I2C byter mode (or buffer mode) is o.k. However, MCTP still can not be enabled under this combination.

10 Firmware Boot Up Sequence



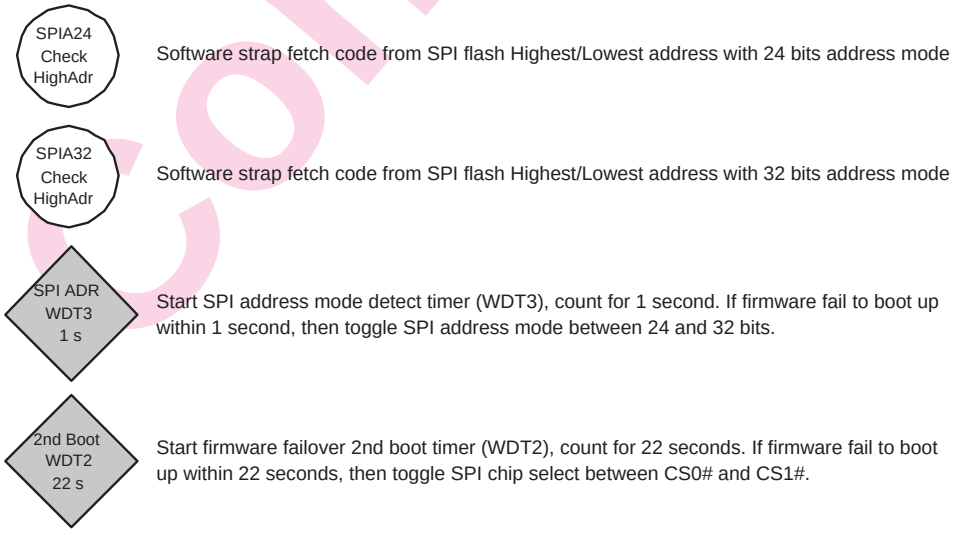


Figure 41: Firmware Boot-Up sequence

11 UART Debug Interface

AST2500 integrates a UART debug interface which can input commands to AST2500 by using simple terminal program without the assistance of CPU.

The default debug port is UART5. It can be changed to UART1 by setting hardware strap bit [29] = 0. For security issue, this feature can be disabled by setting register SCU2C[10] = 1.

The debug UART port is default in normal UART function mode. 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 AHB Bus Controller

12.1 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 ~ 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

12.2 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

12.3 Registers : Base Address = 0x1E60:0000

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

Bit	R/W	Description
31:0	RW	**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	Description
31:11		**Reserved (0)**
10:8	RW	**Buffer Size** 000: 4KB 001: 8KB 010: 16KB 011: 32KB 100: 128KB 101: 256KB 110: 512KB 111: 1024KB
6:4	RW	**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]
2	RW	**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	**Polling Mode** 0: polling read command 1: polling write command
0	RW	**Polling Enable** Disable the polling function will reset the FIFO.

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

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

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

Bit	R/W	Description
31:0	R	**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.	Description
31:0	R	**FIFO Data**

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

Bit	R/W	Description
31:9		**Reserved (0)**
8:0	RW	**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	Description
31:17		**Reserved (0)**
16	RW	**Bus lock interrupt status**
15:1		**Reserved (0)**
0	RW	**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	Description
31:30		**Reserved (0)**
29	R	**APB Lock: APB-to-PCIE**
28	R	**APB Lock: APB-to-LPC**

to next page

from previous page

27	R	**APB Lock: APB-to-PCI**
26	R	**APB Lock: I2C**
25	R	**APB Lock: LPC**
24	R	**APB Lock: UART4**
23	R	**APB Lock: UART3**
22	R	**APB Lock: UART2**
21	R	**APB Lock: UART1**
20	R	**APB Lock: P2X**
19	R	**APB Lock: BRAM**
18	R	**APB Lock: ESPI**
17	R	**APB Lock: XDMA**
16	R	**APB Lock: MMC**
15	RW	**Disable target USB2.0 port2**
14	RW	**Reserved**
13	RW	**Disable target SPI2/SPI3**
12	RW	**Disable target AHB-to-DRAM**
11	RW	**Disable target Firmware Flash**
10	RW	**Disable target SD/SDIO**
9	RW	**Disable target 2D Engine**
8	RW	**Disable target Video Engine**
7	RW	**Disable target APB Bridge**
6	RW	**Disable target VIC**
5	RW	**Disable target USB2.0 port1**
4	RW	**Disable target USB1.1 Host**
3	RW	**Disable target MAC**
2	RW	**Disable target MIC**
1	RW	**Disable target SRAM**
0	RW	**Disable target LPC Plus**

Note :
This specific register bit will be set to '1' when watchdog timeout.

Offset: 8Ch	**AHBC8C: Address Remapping Register**	**Init = 0**
Bit	**R/W**	**Description**
31:8		**Reserved (0)**
7:6	RW	**Reserved**
5	RW	**PCI-E Mapping Enable** 0: Disable mapping 1: Enable mapping This bit enables the mapping of physical address space range 0x7000_0000 ~ 0x7FFF_FFFF to PCI-E Host controller. Remap mechanism provide to speed up access time of program code.

to next page

from previous page

4	RW	**LPC+ Mapping Enable** 0: Disable mapping 1: Enable mapping This bit enables the mapping of physical address space range 0x7000_0000 ~ 0x7FFF_FFFF to LPC+ Host controller. Remap mechanism provide to speed up access time of program code.
3:1	RW	**Reserved**
0	RW	~~**Boot Area Remap**~~

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

Bit	R/W	Description
31:16		**Reserved (0)**
15:0	R	**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	Description
31:16		**Reserved (0)**
15:0	RW	**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.

12.4 Programming Guide

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

12.4.1 Enable Polling

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

13 Firmware SPI Memory Controller

13.1 Overview

AST2500 firmware flash interface only support boot from SPI. Below is the list of removed features from AST2400.

- Remove legacy register set located at 0x1600_0000.
- Remove boot from NAND and NOR flash type.
- Remove NAND flash type supporting.
- Parallel NOR flash interface can only support 8 bits data and 8 bits address.
- Remove support of CE3 and CE4.
- Remove support 4 bits data mode of SPI flash.
- Remove support of SPI Mode 3.
- Freeze the flash type of CE0/CE1 to be only SPI.

Below is the new added features.

- Add address mode (3 bytes or 4 bytes) auto-detection capability.
- Add register lock from write feature, for both SRST# and Watchdog reset events.
- Add write command decoding capability. And can limit the write command that is allowed to be executed.
- Add 5 sets of write address filter for segment write protection.
- Maximum decoded address space for a single SPI chip select is 256MB.
- Modify the address mapping method of 2nd boot condition as swapping CE0 and CE1.

Base address of FMC = 0x1E62_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
FMC54: SPI Dummy Cycle Data 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: SPI Flash Read Timing Compensation
FMC9C: Software Strap Status
FMCA0: Write 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

13.2 Features

AST2500 supports 2 types of flash memory: SPI flash and NOR flash memory. Additionally, AST2500 also provides 3 chip select pins (CE0 ∼ CE2) to control at most 3 flash memory devices. CE2 can be programmed to be SPI or NOR types of flash interface. Moreover, CE0 ∼ CE2 can be assigned to different non-overlapping and continuous address regions with programmable starting address. The allowable flash memory address space is:

- 0x2000_0000 ∼ 0x2FFF_FFFF.

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

CE0 is the default firmware booting source, and only support SPI flash type. 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.

13.2.1 NOR type flash supporting features

- Support maximum 8 address bits.
- Support x8 data interface.
- Programmable AC timing.

13.2.2 SPI type flash supporting features

- Support maximum 128 MBytes flash memory.
- Programmable clock speed, 1/1 ∼ 1/64 AHB bus clock.
- Support 1 and 2 bits input/output SPI flash memory.
- Support 24bits and 32bits address mode.
- Support high clock rate (> 50MHz) timing calibration function.

13.2.3 Alternate (2nd) Boot Function

- Alternate boot function is enabled by hardware strapping bit17.
- When the main flash booting successfully, it should disable the Watchdog timer 2 for boot time counting.
- If the firmware running on the main flash didn't disable the Watchdog timer 2 within 22 seconds. Then BMC would be reset and start to boot from the 2nd flash.
- In the 2nd flash booting mode, the address mapping to CE0 and CE1 would be swapped. To restore the address mapping, firmware should clear the 2nd boot mode register in the WDT2 status register WDT30.bit[1].
- The firmware code in the 2nd flash can be the same as the main flash.

13.3 Timing Definition

- Figure 42: NOR type flash normal read timing
- Figure 43: NOR type flash normal write timing
- Figure 44: SPI type flash read/write timing
- Figure 45: SPI type flash read/write timing with 2 bits IO mode

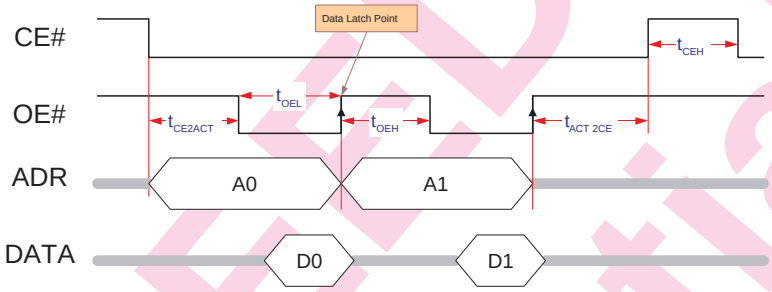


Figure 42: NOR Flash Read Timing

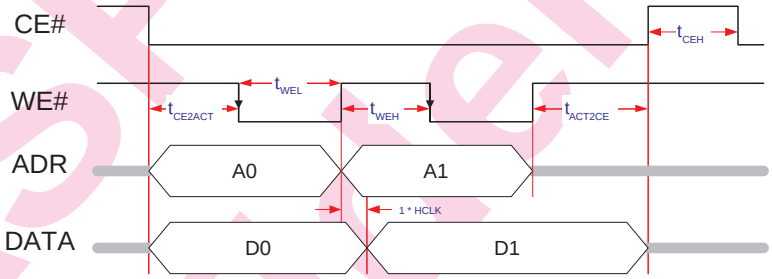


Figure 43: NOR Flash Write Timing

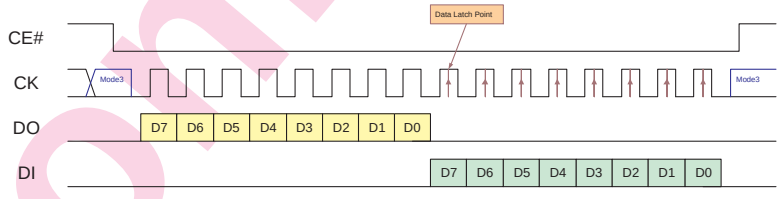


Figure 44: SPI Flash R/W Timing

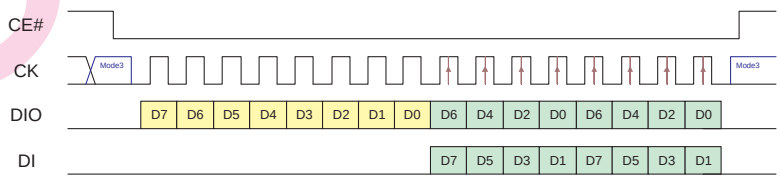


Figure 45: SPI Flash 2 Bits R/W Timing

13.4 Registers : Base Address = 0x1E62:0000

Offset: 00h	FMC00: CE Type Setting Register	Init = 0x8000002A

Bit	R/W	Description
31		**Reserved (1)**
30:19		**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		**Reserved (0)**
5:4	RW	**CE2 flash type selection** 0x: Select NOR flash type 1x: Select SPI flash type
3:2	R	**CE1 flash type selection**
1:0	R	**CE0 flash type selection** 10: Select SPI flash type

Offset: 04h	FMC04: CE Control Register	Init = 0x00000700

Bit	R/W	Description
31:11		**Reserved (0)**
10	RW	**Enable CE2 flash timing Div2 mode**
9	RW	**Enable CE1 flash timing Div2 mode**
8	RW	**Enable CE0 flash timing Div2 mode** 0: normal speed 1: half speed, double (2X) clock cycles When enable the Div2 timing mode, some timing at FMC10~FMC18 would be affect, and it depends on different flash type. • NOR flash: timing defined at bit[27:0] would be divided by 2 – t-CE2ACT, t-OEL, t-OEH, t-WEL, t-WEH, t-ACT2CE, tCEH • SPI flash: timing defined at bit[27:24] would be divided by 2 – CE# inactive pulse width – if FMC10~FMC18 bit[13] also be set to '1', then SPI clock would be divided by 4
7:3		**Reserved (0)**
2	RW	**CE2 SPI address mode selection**
1	RW	**CE1 SPI address mode selection**

to next page

from previous page

0	RW	**CE0 SPI address mode selection** 0: 3 bytes (smaller than or equal to 16MB) 1: 4 bytes (larger than 16MB) CE0 would be set at power up first boot by auto-detect function.

Offset: 08h **FMC08: Interrupt Control and Status Register** **Init = 0x0**

Bit	R/W	Description
31:12	R	**Reserved (0)**
11	R	**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. • Read with un-supported command when command filter is enabled. • Write with un-supported command when command filter is enabled. • 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.
8:4	R	**Reserved (0)**
3	RW	**DMA Interrupt Enable** 0: Disable 1: Enable
2	RW	**SPI Command Abort Interrupt Enable** 0: Disable 1: Enable
1	RW	**SPI Write Address Protected Interrupt Enable** 0: Disable 1: Enable
0	R	**Reserved (0)**

Offset: 0Ch **FMC0C: Command Control Register** **Init = 0x0**

Bit	R/W	Description
31:8	R	**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 value 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 Read command mode and Write command mode. When all address and data byte lane are disabled, only command field will be issued.

to next page

from previous page

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

Bit	Attr.	Description
		SPI Flash Interface
31	RW	**Reserved (keep 0)**
30:28	RW	**IO Mode** 000: single bit or controlled by bit[3]. 010: dual bit read/write, data cycle only. 011: dual bit read/write, including address and dummy byte cycle(not indicate dummy clock cycle). others: reserved The IO mode also apply to User mode.
27:24	RW	**CE# Inactive pulse width** 0000: 16T (1T = 1 HCLK clock) 0001: 15T .... 1111: 1T The setting timing will be doubled if 2X timing was set at FMC04[10:8].
23:16	RW	**Command data** The content of this register is used as the data for Fast 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 fast read command (high bits)**
13	RW	**SPI clock divide 4 mode enable** This bit must be set in companion with FMC04[10:8]. When both bits set to 1, the SPI clock output would be divided by 4 again.
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.

to next page

from previous page

11:8	RW	**SPI clock frequency selection (t-CK)** 0000: HCLK/16 (default) 0001: HCLK/14 0010: HCLK/12 0011: HCLK/10 0100: HCLK/8 0101: HCLK/6 0110: HCLK/4 0111: HCLK/2 1000: HCLK/15 1001: HCLK/13 1010: HCLK/11 1011: HCLK/9 1100: HCLK/7 1101: HCLK/5 1110: HCLK/3 1111: HCLK
7:6	RW	**Dummy cycles before data for fast 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** ~~Clock Mode_0/Mode_3 selection~~
3	RW	**Enable dual data input mode** 0: 1 bit data input each clock 1: 2 bits data input each clock When enabled this bit and the SPI flash memory device supports dual data input mode, the data rate will be doubled. This bit is valid only when bit[30:28] = 0.
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: Normal Read (03h + Address + Read data [1/2/3/4 bytes]) 01: Read Command (CMD + Address + Read data [1/2/3/4 bytes]) 10: Write Command (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 Read/Write command supported. CMD = 1 byte of data from bit[23:16] of this register Address = 3 or 4 bytes of data from address bus of AHB 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.
For Parallel NOR Type Flash Interface		
31:28	RW	**Reserved (0)**
27:24	RW	**CE# high pulse width for each access (t-CEH)** 0000: No CE# high pulse width requirement 0001: > 2T (1T = 1HCLK) 0001: > 3T .... 1111: > 16T This timing defines the CE# high pulse minimum width requirement, if it is set to nonzero value, then CE# will raise high and keep at least the defined cycles for each AHB bus command. It still will have burst possibility if the AHB bus command is not a byte command.
23:20	RW	**OE#/WE# High to CE# High Delay (t-ACT2CE)**
19:16	RW	**WE# High Pulse Width (t-WEH)**
15:12	RW	**WE# Low Pulse Width (t-WEL)**
11:8	RW	**OE# High Pulse Width (t-OEH)**
7:4	RW	**OE# Low Pulse Width (t-OEL)**
3:0	RW	**CE# Low to OE#/WE# Low Delay (t-CE2ACT)** The following table can be applied to all the above timing settings of bit[23:0]. 0000: 16T (default) .... 1110: 2T 1111: 1T (1T = period of HCLK) When the Div2 option at FMC04 is set, the actual timing would be: 0000: 31T (default) .... 1110: 3T 1111: 1T (1T = period of HCLK) These registers define the read/write timings for ”NOR” flash read/write cycles. For read cycles, read data are latched at the rising edge of OE# signal. For write cycles, write data are latched at the rising edge of WE# signal.

Offset: 30h	FMC30: CE0 Address Decoding Range Register	Init = 0x5040_0000
Offset: 34h	FMC34: CE1 Address Decoding Range Register	Init = 0x5450_0000
Offset: 38h	FMC38: CE2 Address Decoding Range Register	Init = 0x5854_0000

Bit	Attr.	Description
31:30		**End address H bit[30:29]**
29:24	RW	**End address L bit[28:23]** This register defines the CEx end address limit in the unit of 8MB. CE0 segment size is default set to 128MB. All others segment size are default set to 32MB.
23:22		**Start address H bit[30:29]**
21:16	RW	**Start address L bit[28:23]** The valid flash decoding address is: 0x2000_0000 - 0x2FFF_FFFF This register defines the CEx starting address in the unit of 8MB, so it equals to the real address bit A28-A23. The valid address segment for each CE is: StartAdr $\leq$ CEx $<$ EndAdr. The default address segment for each CE listed below: CE0 = 0x2000_0000 - 0x27FF_FFFF (128MB) CE1 = 0x2800_0000 - 0x29FF_FFFF (32MB) CE2 = 0x2A00_0000 - 0x2BFF_FFFF (32MB) The address segment setting of 5 chips select must not be overlapped. It will cause error when overlapped. The Start address for CE0 is read only, default at 0x0 or 0x20000000. There is a rule must be followed in defining the Start address for NOR or SPI type flash. The address bits of zero start must larger or equal to the maximum flash size. For example: 0x20000000 can address 0 $\sim$ 64MB flash chip 0x24000000 can address 0 $\sim$ 64MB flash chip 0x22000000 can address 0 $\sim$ 32MB flash chip 0x21000000 can address 0 $\sim$ 16MB flash chip 0x20800000 can address 0 $\sim$ 8MB flash chip
15:0		**Reserved (0)**

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

Bit	R/W	Description
31:8		**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 3 chip selects.

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

Bit	R/W	Description
31:12		**Reserved (0)**
11:8	RW	**SPI read data input delay cycle setting** Calibration mode use only. The value definition is the same as FMC94.
7:4	RW	**SPI clock frequency setting** Calibration mode use only. The value definition is the same as CEx Control Register.

to next page

from previous page

3	RW	**Calibration Mode** 0: Norma mode 1: Calibration mode. At this mode, SPI clock rate and read delay cycle parameters setting will be replaced by bit[11:4]. 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.
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 :
The procedure to enable DMA:
1. Set FMC84
2. Set FMC88
3. Set FMC8C
4. Set FMC80[1:0]

The procedure to reset or disable DMA: Set FMC80[0] = 0.

Offset: 84h **FMC84: DMA Flash Side Address** **Init = 0x2xxx_xxxx**

Bit	R/W	Description
31:28		**Reserved (0x2)**
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 0x2000_0000 - 0x2FFF_FFFF. When read, it showed the current working address.
1:0		**Reserved**

Offset: 88h **FMC88: DMA DRAM Side Address** **Init = 0x8xxx_xxxx**

Bit	R/W	Description
31:30		**Reserved (0x2)**
29:2	RW	**DRAM side start address** 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. And the valid address range is 0x8000_0000 - 0xBFFF_FFFF. When read, it showed the current working address.
1:0		**Reserved**

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

Bit	R/W	Description
31:25		**Reserved (0)**
24:2	RW	**DMA Length** From 4 bytes to 32MB 0: 4 bytes 0x7FFFFF: 32M bytes When DMA on-going, the length register will count down, and reach 0 when DMA finished.
1:0		**Reserved**

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

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

Offset: 94h	FMC94: SPI Flash Read Timing Compensation	Init = 0

Bit	R/W	Description
31:19		**Reserved (0)**
19:16	RW	**SPICLK = HCLK/5, data input delay cycle**
15:12	RW	**SPICLK = HCLK/4, data input delay cycle**
11:8	RW	**SPICLK = HCLK/3, data input delay cycle**
7:4	RW	**SPICLK = HCLK/2, data input delay cycle**
3:0	RW	**SPICLK = HCLK, data input delay cycle** Bit[3]: 0: DI input no delay 1: DI input delay 4ns 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: 9Ch	FMC9C: Software Strap Status	Init = 0

Bit	R/W	Description
31:16	R	**Reserved**
15:8	R	**Strap Command Count**
7:4	R	**Strap Parity Error Command Count**
3:2	R	**Reserved**
1	R	**Strap Operation Result** 0: Success 1: Fail, no valid strap code found If failed, strap operation will do again for each time CPU is reset by watchdog.

to next page

from previous page

0	RW	**Strap Operation Status** 0: NOP 1: Strapping done Write FMC9C by 0xAEEDFC20 to clear full register to 0. Clear bit0 to 0 will force strap operation again when watchdog reset.

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

Bit	R/W	Description
31:15	R	**Reserved**
14	RW	**Enable command 0xDC + ADR4 (4B Address Block Erase)**
13	RW	**Enable command 0xD8 + ADR3/4 (Block Erase)**
12	RW	**Enable command 0xC7 (Chip Erase)**
11	RW	**Enable command 0x60 (Chip Erase)**
10	RW	**Enable command 0x5C + ADR4 (4B Address SubBlock Erase)**
9	RW	**Enable command 0x52 + ADR3/4 (SubBlock Erase)**
8	RW	**Enable command 0x50 (Clear Flag Status Register)**
7	RW	**Enable command 0x21 + ADR4 (4B Address Sector Erase)**
6	RW	**Enable command 0x20 + ADR3/4 (Sector Erase)**
5	RW	**Enable command 0x12 + ADR4 (4B Address Page Program)**
4	RW	**Enable command 0x06 (Write Enable)**
3	RW	**Enable command 0x04 (Write Disable)**
2	RW	**Enable command 0x02 + ADR3/4 (Page Program)**
1	RW	**Enable command 0x01 (Write Status Register)**
0	RW	**Enable Write command filter**

Note :
When write command filter function is enabled (bit0 = 1),

- Write mode can only accept the enabled commands.
- User mode will be disabled.
- Read mode only support commands as listed below:
  - 0x03, 0x05, 0x0B, 0x0C, 0x13, 0x15, 0x3B, 0x3C, 0x5A, 0x90, 0x9F, 0xB7, 0xBB, 0xBC, 0xC5, 0xC8, 0xE9
- It must use FMC04[2:0] and FMC0C[7:0] to generate the desired input/output byte lanes of address and data.

When an invalid command is issued, interrupt flag FMC08[10] will be set.
ADR3/4 means 3 bytes or 4 bytes address defined by FMC04[2:0]
ADR4 means 4 bytes address mode only

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

Bit	R/W	Description
31:10	R	**Reserved**
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**

to next page

from previous page

1:0	RW	**Mode of write address filter #1** 0x: NOP 10: write enabled 11: write disabled

Note :
When address filter function is enabled, User mode will be disabled.
Write filter only applied to write command with address field larger or equal to 3 bytes.
The write filter mode is controlled 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 defined at FMCB0 ~ FMCC0 controls the address range for write allowed or disallowed.

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

Bit	R/W	Description
31:16	R	**Reserved**
15	RW	**Lock FMCC0 from write until reset by SRST#**
14	RW	**Lock FMCBC from write until reset by SRST#**
13	RW	**Lock FMCB8 from write until reset by SRST#**
12	RW	**Lock FMCB4 from write until reset by SRST#**
11	RW	**Lock FMCB0 from write until reset by SRST#**
10	RW	**Lock FMCA4 from write until reset by SRST#**
9	RW	**Lock FMCA0 from write until reset by SRST#**
8	RW	**Lock 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
31:16	R	**Reserved**
15	RW	**Lock FMCC0 from write until reset by watchdog**
14	RW	**Lock FMCBC from write until reset by watchdog**
13	RW	**Lock FMCB8 from write until reset by watchdog**
12	RW	**Lock FMCB4 from write until reset by watchdog**
11	RW	**Lock FMCB0 from write until reset by watchdog**

to next page

from previous page

10	RW	**Lock FMCA4 from write until reset by watchdog**
9	RW	**Lock FMCA0 from write until reset by watchdog**
8	RW	**Lock 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**

Bit	R/W	Description
31:16	RW	**Upper address limit of segment address bit[27:12]**
15:0	RW	**Lower address limit of segment address bit[27:12]**

Note :
The segment is defined as below address range for each CE:
lower_limit $\leq$ segment address $\leq$ upper_limit
The address filter control is defined in FMCA4.

Offset: B4h **FMCB4: Write Address Filter Register #2** **Init = 0**

Bit	R/W	Description
31:16	RW	**Upper address limit of segment address bit[27:12]**
15:0	RW	**Lower address limit of segment address bit[27:12]**

Note :
The segment is defined as below address range for each CE:
lower_limit $\leq$ segment address $\leq$ upper_limit
The address filter control is defined in FMCA4.

Offset: B8h **FMCB8: Write Address Filter Register #3** **Init = 0**

Bit	R/W	Description
31:16	RW	**Upper address limit of segment address bit[27:12]**
15:0	RW	**Lower address limit of segment address bit[27:12]**

Note :
The segment is defined as below address range for each CE:
lower_limit $\leq$ segment address $\leq$ upper_limit
The address filter control is defined in FMCA4.

Offset: BCh **FMCBC: Write Address Filter Register #4** **Init = 0**

Bit	R/W	Description
31:16	RW	**Upper address limit of segment address bit[27:12]**

to next page

from previous page

15:0	RW	**Lower address limit of segment address bit[27:12]**

Note :
The segment is defined as below address range for each CE:
lower_limit $\leq$ segment address $\leq$ upper_limit
The address filter control is defined in FMCA4.

Offset: C0h **FMCC0: Write Address Filter Register #5** **Init = 0**

Bit	R/W	Description
31:16	RW	**Upper address limit of segment address bit[27:12]**
15:0	RW	**Lower address limit of segment address bit[27:12]**

Note :
The segment is defined as below address range for each CE:
lower_limit $\leq$ segment address $\leq$ upper_limit
The address filter control is defined in FMCA4.

13.5 Programming Guide

13.5.1 DMA Mode

- DMA operation can not across the CE segment boundary.
- DMA DRAM 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]
- Reset or terminate DMA procedure:
  1. Set FMC80[0] = 0

13.5.2 DMA CheckSum Calculation Mode

- CheckSum calculation only applied to flash read mode.
- Command procedure:
  1. Set FMC84
  2. Set FMC88
  3. Set FMC8C
  4. Set FMC80[1:0] = ”01”
  5. FMC80[2] is set when only checksum calculation is necessary, no data movement
  6. Wait DMA done

13.5.3 SPI Timing Calibration Sequence

1. Calculate the golden checksum by slower speed, record the checksum value in memory
2. Scan the checksum by different clock rate and delay cycle

```
for(spiclk = HCLK/5 to spiclk = HCLK){
    Set FMC80[7:4] = spiclk
    for(delay_count = 0; delay_count <= 15; delay_count++){
        Set FMC80[10:8] = delay_count
        Set FMC80[3:0] = ”1101”
        Wait FMC08[11] = 1
        Compare FMC94 with the golden checksum, if matched then record pass for this delay_count
        Set FMC80[3:0] = ”0000”
    }
    Select an appropriate delay cycle value and set to FMC94
}
```

13.5.4 Address Continuity Rule

For not ”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.

14 SPI Flash Controller

14.1 Overview

SPI Memory Controller (SPI) registers are revised to fully compliant to flash memory controller of BMC .

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 = 0x3800_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
SPIR30: SPI1 CE0 Address Decoding Range Register
SPIR34: SPI1 CE1 Address Decoding Range Register
SPIR30: SPI2 CE0 Address Decoding Range Register
SPIR34: SPI2 CE1 Address Decoding Range Register
SPIR54: SPI Dummy Cycle Data Register
SPIR94: SPI Flash Read Timing Compensation
SPIRA0: Write 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

14.2 Registers : Base Address = 0x1E63:x000

Offset: 00h	SPIR00: SPI Flash Configuration Register	Init = 0x0000000A
Bit	**R/W**	**Description**
31:18		**Reserved (0)**
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 companion with the Write Address Filter register at SPIRA4 controls 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		**Reserved**

Offset: 04h	SPIR04: CE Control Register	Init = 0x00000300
Bit	**R/W**	**Description**
31	RW	**CE swap control** 0: CE0 and CE1 output use the default CE0/CE1 specific control registers 1: CE0 and CE1 output swap all the CE0/CE1 specific control registers. That is at swap mode, CE1 output controlled by all the registers originally defined for CE0, and vice versa for CE0 output. **For SPI1 interface, the swap bit also controls the pass-through mode CE0/CE1 output.** To set this bit to '1', write SPIR04 bit[31:16] = 0xAEED To set this bit to '0', write SPIR04 bit[31:16] = 0xEADE **This bit is not write protected.**
30:10		**Reserved (0)**
9	RW	**Enable CE1 flash timing Div2 mode**
8	RW	**Enable CE0 flash timing Div2 mode** 0: normal speed 1: half speed, double (2X) clock cycles When enable the Div2 timing mode, some timing at SPIR10~SPIR14 would be affect. • SPI flash: CE inactive pulse width timing defined at bit[27:24] would be divided by 2 • If SPIR10~SPIR14 bit[13] is also set to '1', then SPI clock would be divided by 4
7:2		**Reserved (0)**
1	RW	**CE1 SPI Address mode selection**
0	RW	**CE0 SPI Address mode selection** 0: 3 bytes (smaller than or equal to 16MB) 1: 4 bytes (larger than 16MB)

Offset: 08h	SPIR08: Interrupt Control and Status Register	Init = 0x0
Bit	**R/W**	**Description**
31:11	R	**Reserved (0)**
10	RW	**SPI Command Abort Status** Command abort conditions as below: • User mode read/write command when command or address filter are enabled. • Read with un-supported command when command filter is enabled. • Write with un-supported command when command filter is enabled. • 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.
8:3	R	**Reserved (0)**
2	RW	**SPI Command Abort Interrupt Enable** 0: Disable 1: Enable

to next page

from previous page

1	RW	**SPI Write Address Protected Interrupt Enable** 0: Disable 1: Enable
0	R	**Reserved (0)**

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

Bit	R/W	Description
31:8	R	**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 value 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 Read command mode and Write command mode. 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_0200**
Offset: 14h **SPIR14: CE1 Control Register** **Init = 0x0000_0200**

Bit	Attr.	Description
31:30		**Reserved**
29:28	RW	**IO Mode** 00: single bit or controlled by bit[3]. 10: dual bit read/write, data cycle only. 11: dual bit read/write, including address and dummy byte cycle(not indicate dummy clock cycle). The IO mode also applicable to User mode.
27:24	RW	**CE# Inactive pulse width** 0000: 16T (1T = 1 HCLK clock) 0001: 15T .... 1111: 1T The setting timing will be doubled if 2X timing was set at SPIR04[9:8].
23:16	RW	**Command data** The content of this register is used as the data for Fast 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 fast read command (high bits)**
13	RW	**SPI clock divide 4 mode enable** This bit must be set in companion with SPIR04[9:8]. When both bits set to 1, the SPI clock output would be divided by 4 again.

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: HCLK/16 (default) 0001: HCLK/14 0010: HCLK/12 0011: HCLK/10 0100: HCLK/8 0101: HCLK/6 0110: HCLK/4 0111: HCLK/2 1000: HCLK/15 1001: HCLK/13 1010: HCLK/11 1011: HCLK/9 1100: HCLK/7 1101: HCLK/5 1110: HCLK/3 1111: HCLK
7:6	RW	**Dummy cycles before data for fast 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** ~~Clock Mode_0/Mode_3 selection~~
3	RW	**Enable dual data input mode** 0: 1 bit data input each clock 1: 2 bits data input each clock When enabled this bit and the SPI flash memory device supports dual data input mode, the data rate will be doubled. This bit is valid only when bit[29:28] = 0.
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: Normal Read (03h + Address + Read data [1/2/3/4 bytes]) 01: Read Command (CMD + Address + Read data [1/2/3/4 bytes]) 10: Write Command (CMD + Address + Write data [1/2/3/4 bytes]) 11: User Mode (Read/write data [1/2/3/4 bytes]) At User mode, address has no meaning, all address decoded in the same CE address range are valid, and data will be read/write to/from the LSB byte first of each 32 bits command. This mode provides a flexible programming method for specific command type other than Read/Write command supported. CMD = 1 byte of data from bit[23:16] of this register Address = 3 or 4 bytes of data from address bus of AHB 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 Except User Mode, the address space can support up to 16 MBytes maximum.

Offset: 30h	**SPIR30: SPI1 CE0 Address Decoding Range Register**	**Init = 0x6460_0000**
Offset: 34h	**SPIR34: SPI1 CE1 Address Decoding Range Register**	**Init = 0x7064_0000**
Offset: 30h	**SPIR30: SPI2 CE0 Address Decoding Range Register**	**Init = 0x7470_0000**
Offset: 34h	**SPIR34: SPI2 CE1 Address Decoding Range Register**	**Init = 0x8074_0000**

Bit	Attr.	Description
31:28	R	**End address H bit[30:27]**
27:24	RW	**End address L bit[26:23]** This register defines the CEx end address limit in the unit of 8MB.
23:20	R	**Start address H bit[30:27]**

to next page

from previous page

19:16	RW	**Start address L bit[26:23]** The valid flash decoding address is: SPI1: 0x3000_0000 - 0x37FF_FFFF SPI2: 0x3800_0000 - 0x3FFF_FFFF This register defines the CEx starting address in the unit of 8MB, so it equals to the real address bit A26-A23. The valid address segment for each CE is: StartAdr $\leq$ CEx $<$ EndAdr. The default address segment for each CE of SPI1 listed below: CE0 = 0x3000_0000 - 0x31FF_FFFF (32MB) CE1 = 0x3200_0000 - 0x37FF_FFFF (96MB) The default address segment for each CE of SPI2 listed below: CE0 = 0x3800_0000 - 0x39FF_FFFF (32MB) CE1 = 0x3A00_0000 - 0x3FFF_FFFF (96MB) The address segment setting of 2 chips select must not be overlapped. It will cause error when overlapped. The Start address for CE0 is read only, SPI1 default at 0x30000000. SPI2 default at 0x38000000. The End address for CE1 is read only, SPI1 default at 0x38000000. SPI2 default at 0x40000000. There is a rule must be followed in defining the Start address. The address bits of zero start must larger or equal to the maximum flash size. For example: 0x30000000 can address 0 ~ 64MB flash chip 0x34000000 can address 0 ~ 64MB flash chip 0x32000000 can address 0 ~ 32MB flash chip 0x31000000 can address 0 ~ 16MB flash chip 0x30800000 can address 0 ~ 8MB flash chip
15:0		**Reserved (0)**

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

Bit	R/W	Description
31:8		**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 2 chip selects.

Offset: 94h **SPIR94: SPI Flash Read Timing Compensation** **Init = 0**

Bit	R/W	Description
31:19		**Reserved (0)**
19:16	RW	**SPICLK = HCLK/5, data input delay cycle**
15:12	RW	**SPICLK = HCLK/4, data input delay cycle**
11:8	RW	**SPICLK = HCLK/3, data input delay cycle**

to next page

from previous page

7:4	RW	**SPICLK = HCLK/2, data input delay cycle**
3:0	RW	**SPICLK = HCLK, data input delay cycle** Bit[3]: 0: DI input no delay 1: DI input delay 4ns 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: Write Command Filter Control Register** **Init = 0**

Bit	R/W	Description
31:15	R	**Reserved**
14	RW	**Enable command 0xDC + ADR4 (4B Address Block Erase)**
13	RW	**Enable command 0xD8 + ADR3/4 (Block Erase)**
12	RW	**Enable command 0xC7 (Chip Erase)**
11	RW	**Enable command 0x60 (Chip Erase)**
10	RW	**Enable command 0x5C + ADR4 (4B Address SubBlock Erase)**
9	RW	**Enable command 0x52 + ADR3/4 (SubBlock Erase)**
8	RW	**Enable command 0x50 (Clear Flag Status Register)**
7	RW	**Enable command 0x21 + ADR4 (4B Address Sector Erase)**
6	RW	**Enable command 0x20 + ADR3/4 (Sector Erase)**
5	RW	**Enable command 0x12 + ADR4 (4B Address Page Program)**
4	RW	**Enable command 0x06 (Write Enable)**
3	RW	**Enable command 0x04 (Write Disable)**
2	RW	**Enable command 0x02 + ADR3/4 (Page Program)**
1	RW	**Enable command 0x01 (Write Status Register)**
0	RW	**Enable Write command filter**

Note :
When write command filter function is enabled,

1. Write mode can only accept the enabled commands.
2. User mode will be disabled.
3. Read mode only support commands as listed below:
   - 0x03, 0x05, 0x0B, 0x0C, 0x13, 0x15, 0x3B, 0x3C, 0x5A, 0x90, 0x9F, 0xB7, 0xBB, 0xBC, 0xC5, 0xC8, 0xE9
4. It must use SPIR04[1:0] and SPIR0C[7:0] to generate the desired address and data bytes output.

ADR3/4 means 3 bytes or 4 bytes address defined by SPIR04[1:0]
ADR4 means 4 bytes address mode only

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

Bit	R/W	Description
31:10	R	**Reserved**
9:8	RW	**Write address filter #5**
7:6	RW	**Write address filter #4**
5:4	RW	**Write address filter #3**
3:2	RW	**Write address filter #2**
1:0	RW	**Write address filter #1** 0x: NOP 10: write enabled 11: write disabled

Note :
When address filter function is enabled, User mode will be disabled.
The write operation is controlled 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 defined at SPIRB0 ~ SPIRC0 controls the address range for write allowed or disallowed.

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

Bit	R/W	Description
31:16	R	**Reserved**
15	RW	**Lock SPIRC0 from write until reset by SRST#**
14	RW	**Lock SPIRBC from write until reset by SRST#**
13	RW	**Lock SPIRB8 from write until reset by SRST#**
12	RW	**Lock SPIRB4 from write until reset by SRST#**
11	RW	**Lock SPIRB0 from write until reset by SRST#**
10	RW	**Lock SPIRA4 from write until reset by SRST#**
9	RW	**Lock SPIRA0 from write until reset by SRST#**
8		**Reserved**
7		**Reserved**
6	RW	**Lock SPIR34 from write until reset by SRST#**
5	RW	**Lock SPIR30 from write until reset by SRST#**
4		**Reserved**
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
31:16	R	**Reserved**
15	RW	**Lock SPIRC0 from write until reset by watchdog**

to next page

from previous page

14	RW	**Lock SPIRBC from write until reset by watchdog**
13	RW	**Lock SPIRB8 from write until reset by watchdog**
12	RW	**Lock SPIRB4 from write until reset by watchdog**
11	RW	**Lock SPIRB0 from write until reset by watchdog**
10	RW	**Lock SPIRA4 from write until reset by watchdog**
9	RW	**Lock SPIRA0 from write until reset by watchdog**
8	RW	**Reserved**
7	RW	**Reserved**
6	RW	**Lock SPIR34 from write until reset by watchdog**
5	RW	**Lock SPIR30 from write until reset by watchdog**
4	RW	**Reserved**
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**

Bit	R/W	Description
31:16	RW	**Upper address limit of segment address bit[27:12]**
15:0	RW	**Lower address limit of segment address bit[27:12]**

Note :
The segment is defined as below address range for each CE:
lower_limit $\leq$ segment address $\leq$ upper_limit
The address filter control is defined in SPIRA4.

Offset: B4h **SPIRB4: Write Address Filter Register #2** **Init = 0**

Bit	R/W	Description
31:16	RW	**Upper address limit of segment address bit[27:12]**
15:0	RW	**Lower address limit of segment address bit[27:12]**

Note :
The segment is defined as below address range for each CE:
lower_limit $\leq$ segment address $\leq$ upper_limit
The address filter control is defined in SPIRA4.

Offset: B8h **SPIRB8: Write Address Filter Register #3** **Init = 0**

Bit	R/W	Description
31:16	RW	**Upper address limit of segment address bit[27:12]**
15:0	RW	**Lower address limit of segment address bit[27:12]**

Note :
The segment is defined as below address range for each CE:
lower_limit $\leq$ segment address $\leq$ upper_limit
The address filter control is defined in SPIRA4.

Offset: BCh	SPIRBC: Write Address Filter Register #4	Init = 0

Bit	R/W	Description
31:16	RW	**Upper address limit of segment address bit[27:12]**
15:0	RW	**Lower address limit of segment address bit[27:12]**

Note :
The segment is defined as below address range for each CE:
lower_limit $\leq$ segment address $\leq$ upper_limit
The address filter control is defined in SPIRA4.

Offset: C0h	SPIRC0: Write Address Filter Register #5	Init = 0

Bit	R/W	Description
31:16	RW	**Upper address limit of segment address bit[27:12]**
15:0	RW	**Lower address limit of segment address bit[27:12]**

Note :
The segment is defined as below address range for each CE:
lower_limit $\leq$ segment address $\leq$ upper_limit
The address filter control is defined in SPIRA4.

15 Memory Integrity Check Controller

15.1 Overview

Memory Integrity Check Engine (MICE) implements 10 registers, which is listed below, to program the various functions supported by AST2500 . Each register has its own specific offset value to derive its physical address location.

Base address of MICE = 0x1E64_0000
Physical address = (Base address of MICE) + Offset

MIC00: Base Address of Control Buffer Register
MIC04: Base Address of Checksum Buffer Register
MIC08: Rate Control Register
MIC0C: Control Register
MIC10: Stop-Page Register
MIC14: Error Status and Interrupt Mask Register
MIC18: First Page Error Status Register
MIC1C: Secondary Page Error Status Register
MIC20: Feature Register
MIC24: Control Register

15.2 Features

- Directly connected to AHB bus
- Automatic memory integrity check by constant checksum scanning
- Directly access SDRAM memory through M-bus
- 4K bytes of memory per checksum unit
- Each checksum unit can be individually enabled or skipped
- Support Fletchers Checksum algorithm
- Programmable scanning rate and scanning range control
- Slight extra memory bandwidth and capacity demand
- Generate an interrupt whenever detecting checksum errors

15.3 Registers : Base Address = 0x1E64:0000

Offset: 00h	MIC00: Base Address of Control Buffer Register	Init = X
Bit	**R/W**	**Description**
31:30		**Reserved (0)**
29:3	RW	**Base address of control buffer [29:3]** This register determines the base address of control buffer which storing the control information for all memory pages. The base address of control buffer has to be 8-byte aligned. Therefore, address bit [2:0] are always 0.
2 :0		**Reserved (0)**

Offset: 04h	MIC04: Base Address of Checksum Buffer Register	Init = X
Bit	**R/W**	**Description**
31:30		**Reserved (0)**
29:3	RW	**Base address of checksum buffer [29:3]** This register determines the base address of checksum buffer which storing the checksum information for all memory pages. The base address of checksum buffer has to be 8-byte aligned. Therefore, address bit [2:0] are always 0.
2 :0		**Reserved (0)**

Offset: 08h	MIC08: Rate Control Register	Init = X
Bit	**R/W**	**Description**
31:16		**Reserved (0)**
15:0	RW	**Rate control setting** This register determines the rate of MICE doing memory integrity check. Higher value means slower rate.

Offset: 0Ch	MIC0C: Control Register	Init = 0xxx_xxxxh
Bit	**R/W**	**Description**
DRAM 256M Byte Address Mode		
31:29		**Reserved (0)**
28	RW	**Enable MICE** 0: Reset MICE (default) 1: Enable MICE Before enabling MICE, the contents of control buffer and checksum buffer have to be prepared. The related registers have to be programmed as well.
27:12	RW	**Number of pages to be checked** There are 16 bits of register to program the number of pages to be checked when MICE is enabled (Each page is 4K Byte). Therefore, the maximum number of pages to be checked is 64K pages. It implies that the maximum memory size that MICE can support is 4KB*64K = 256MB. The first page to be checked is always page #0 which is with starting memory address 0000_0000h. The number of pages must be 16 aligned (#15, or #31, or #47, ...). See ”Control Buffer Format” & ”Checksum Buffer Format” for detail information.
11:0		**Reserved (0)**
DRAM 1G Byte Address Mode		
31:30		**Reserved (0)**

to next page

from previous page

29:12	RW	**Number of pages to be checked** This register is used to be programmed the number of pages to be checked when MICE is enabled (Each page is 4K Byte). Therefore, the maximum number of pages to be checked is 256K pages. It implies that the maximum memory size that MICE can support is 4KB*256K = 1GB. The first page to be checked is always page #0 which is with starting memory address 0000_0000h. The number of pages must be 16 aligned (#15, or #31, or #47, ...). See ”Control Buffer Format” & ”Checksum Buffer Format” for detail information.
11:0		**Reserved (0)**

Offset: 10h **MIC10: Stop-Page Register** **Init = X**

Bit	R/W	Description
DRAM 256M Byte Address Mode		
31:16	RW	**Writ-back value for checksum buffer**
15:0	RW	**Page number of stop-page**
Note : MIC10 [15:0] must not bigger than MIC0C [27:12]. If (MIC10 [31:16] != 0), MICE will write a 32-bits value (Bit 31∼16 = MIC10 [31:16], Bit 15∼0 = 0) to the checksum buffer of page number #N which N equal to MIC10 [15:0]. Writing this register will also cause MICE to stop and skip the memory integrity check at current process page when current process page number equal to MIC10 [15:0].		
DRAM 1G Byte Address Mode		
31:18	RW	**Writ-back value for checksum buffer**
17:0	RW	**Page number of stop-page**
Note : MIC10 [17:0] must not bigger than MIC0C [29:12]. If (MIC10 [31:18] != 0), MICE will write a 32-bits value (Bit 31∼18 = MIC10 [31:18], Bit 17∼0 = 0) to the checksum buffer of page number #N which N equal to MIC10 [17:0]. Writing this register will also cause MICE to stop and skip the memory integrity check at current process page when current process page number equal to MIC10 [17:0].		

Offset: 14h **MIC14: Error Status and Interrupt Mask Register** **Init = 0**

Bit	R/W	Description
31		**Reserved (0)**
30	R	**Lost page error flag** 0: Lost page error has NOT been detected 1: Lost page error has been detected
29	R	**Secondary page error flag (come from MIC1C [29])** 0: Secondary page error has NOT been detected 1: Secondary page error has been detected

to next page

from previous page

28	R	**First page error flag (come from MIC18 [28])** 0: Secondary page error has NOT been detected 1: Secondary page error has been detected
DRAM 256M Byte Address Mode		
27:18		**Reserved (0)**
17:16	RW	**Interrupt mask bits** 0x: Disable engine interrupt to CPU when secondary page error flag is set 1x: Enable engine interrupt to CPU when secondary page error flag is set x0: Disable engine interrupt to CPU when first page error flag is set x1: Enable engine interrupt to CPU when first page error flag is set
15:0	R	**Process page number of engine**
DRAM 1G Byte Address Mode		
27:26	RW	**Interrupt mask bits** 0x: Disable engine interrupt to CPU when secondary page error flag is set 1x: Enable engine interrupt to CPU when secondary page error flag is set x0: Disable engine interrupt to CPU when first page error flag is set x1: Enable engine interrupt to CPU when first page error flag is set
25:18		**Reserved (0)**
17:0	R	**Process page number of engine**

Offset: 18h **MIC18: First Page Error Status Register** **Init = 000x_xxxx**

Bit	R/W	Description
31		**Reserved (0)**
30	R	**Lost page error flag (come from MIC14 [30])** 0: Lost page error has NOT been detected 1: Lost page error has been detected
29		**Reserved (0)**
28	RW	**First Page Error Flag (cleared by writing 1)** 0: First page error has NOT been detected 1: First page error has been detected
DRAM 256M Byte Address Mode		
27:16		**Reserved (0)**
15:0	R	**Page Number of First Page Error**
DRAM 1G Byte Address Mode		
27:18		**Reserved (0)**
17:0	R	**Page Number of First Page Error**

Offset: 1Ch **MIC1C: Secondary Page Error Status Register** **Init = 000x_xxxx**

Bit	R/W	Description
31		**Reserved (0)**
30	R	**Lost page error flag (come from MIC14 [30])** 0: Lost page error has NOT been detected 1: Lost page error has been detected
29	RW	**Secondary Page Error Flag (cleared by writing 1)** 0: Secondary page error has NOT been detected 1: Secondary page error has been detected

to next page

from previous page

DRAM 256M Byte Address Mode		
28:16		**Reserved (0)**
15:0	R	**Page Number of Secondary Page Error**
DRAM 1G Byte Address Mode		
28:18		**Reserved (0)**
17:0	R	**Page Number of Secondary Page Error**

Offset: 20h **MIC20: Feature Register** **Init = 0**

Bit	R/W	Description
31	RW	**DRAM 1G Byte Address Mode** 0: DRAM 256M Byte Address Mode 1: DRAM 1G Byte Address Mode
30:0		**Reserved (0)**

Offset: 24h **MIC24: Control Register** **Init = 0**

Bit	R/W	Description
DRAM 256M Byte Address Mode		
31:0		**Reserved (0)**
DRAM 1G Byte Address Mode		
31	RW	**Enable MICE** 0: Reset MICE (default) 1: Enable MICE Before enabling MICE, the contents of control buffer and checksum buffer have to be prepared. The related registers have to be programmed as well.
30:0		**Reserved (0)**

15.4 Page Control Bits

- Page Control Bits: Behavior of MICE on each page is presented with 2 bits.

Page Control Bits	Read DRAM Data	Update Checksum Buffer	Update Error Status
`00 (Skip)`	No	No	No
`01 (ECC Mode)`	Yes	No	No
`10 (Debug Mode)`	Yes	Always	No
`11 (MIC Mode)`	Yes	When checksum buffer is initiative value	Yes

15.5 Control Buffer Format

- Page Control Bits: Behavior of MICE on each page is presented with **2 bits**.
- When DRAM is 256M Byte Address Mode, M = MIC0C [27:12]
- When DRAM is 1G Byte Address Mode, M = MIC0C [29:12]

Bit Range	Description
000 - 001	Page control bits of page number #0 (address 0000_0000h - 0000_0FFFh)
002 - 003	Page control bits of page number #1 (address 0000_1000h - 0000_1FFFh)
004 - 005	Page control bits of page number #2 (address 0000_2000h - 0000_2FFFh)
006 - 007	Page control bits of page number #3 (address 0000_3000h - 0000_3FFFh)
008 - 009	Page control bits of page number #4 (address 0000_4000h - 0000_4FFFh)
00A - 00B	Page control bits of page number #5 (address 0000_5000h - 0000_5FFFh)
00C - 00D	Page control bits of page number #6 (address 0000_6000h - 0000_6FFFh)
00E - 00F	Page control bits of page number #7 (address 0000_7000h - 0000_7FFFh)
.....	.....
2*M - 2*M+1	Page control bits of page number #M (address 1000h*M - 1000h*M+FFFh)

15.6 Checksum Buffer Format

- Checksum Value Bytes: Checksum value on each page is presented with **4 bytes**.
- Software need to initiate the checksum buffer with value 0.
- When DRAM is 256M Byte Address Mode, M = MIC0C [27:12]
- When DRAM is 1G Byte Address Mode, M = MIC0C [29:12]

Byte Range	Description
000 - 003	Checksum value of page number #0 (address 0000_0000h - 0000_0FFFh)
004 - 007	Checksum value of page number #1 (address 0000_1000h - 0000_1FFFh)
008 - 00B	Checksum value of page number #2 (address 0000_2000h - 0000_2FFFh)
00C - 00F	Checksum value of page number #3 (address 0000_3000h - 0000_3FFFh)
010 - 013	Checksum value of page number #4 (address 0000_4000h - 0000_4FFFh)
014 - 017	Checksum value of page number #5 (address 0000_5000h - 0000_5FFFh)
018 - 01B	Checksum value of page number #6 (address 0000_6000h - 0000_6FFFh)
01C - 01F	Checksum value of page number #7 (address 0000_7000h - 0000_7FFFh)
.....	.....
4*M - 4*M+3	Checksum value of page number #M (address 1000h*M - 1000h*M+FFFh)

15.7 Porgramming Sequence

15.7.1 Parameter Definition

- $Max\_CheckSumMem\_Size$ (4K-byte aligned):
  The maximum memory size to be checked when MICE is enable.
- Max_Page_Number:
  The maximum page number to be checked when MICE is enable.

  Max_Page_Number = ($Max\_CheckSumMem\_Size$ - 1) $>>$ 12.
- $Max\_Page\_Number\_16Aligned$ (16 page aligned):
  The 16 aligned maximum page number.

  $Max\_Page\_Number\_16Aligned$ = ((((Max_Page_Number $>>$ 4) + 1) $<<$ 4) - 1.
- $Page\_ControlBuf\_Base\_Adr$ (8-byte aligned):
  Base address of page control buffer which store page control bits.
  Size of page control buffer is 2*$Max\_Page\_Number\_16Aligned$ bits.
- $Page\_CheckSumBuf\_Base\_Adr$ (8-byte aligned):
  Base address of page checksum buffer which store checksum value.
  Size of page checksum buffer is 4*$Max\_Page\_Number\_16Aligned$ bytes.
- $Rate\_Control\_Val$:
  Maximum DRAM read/write request rate of MICE.

  $Request\_Rate$ = 1 / (1 + $Rate\_Control\_Val$).
- $Interrupt\_Mask$:
  Set bit with value 1 to enable interrupt.
- TAG:
  A 16 bits accumulative tag value with value 0 is invalid.

15.7.2 MIC Engine Initiation (DRAM 256M Byte Address Mode)

1. Clear page control buffer with value 0.
2. Clear page checksum buffer with value 0.
3. Clear TAG with value 1.
4. MIC00 = $Page\_ControlBuf\_Base\_Adr$ (8-byte aligned).
5. MIC04 = $Page\_CheckSumBuf\_Base\_Adr$ (8-byte aligned).
6. MIC08 = $Rate\_Control\_Val$.
7. MIC0C = ($Max\_Page\_Number\_16Aligned$ $<<$ 12).
8. MIC14 = $Interrupt\_Mask$ $<<$ 16.
9. Read MIC0C untill MIC0C equal to ($Max\_Page\_Number\_16Aligned$ $<<$ 12).
10. MIC0C = (10000000h | ($Max\_Page\_Number\_16Aligned$ $<<$ 12)).

15.7.3 MIC Engine Initiation (DRAM 1G Byte Address Mode)

1. MIC20 = 80000000h.
2. Clear page control buffer with value 0.
3. Clear page checksum buffer with value 0.
4. Clear TAG with value 1.
5. MIC00 = $Page\_ControlBuf\_Base\_Adr$ (8-byte aligned).
6. MIC04 = $Page\_CheckSumBuf\_Base\_Adr$ (8-byte aligned).
7. MIC08 = $Rate\_Control\_Val$.
8. MIC0C = ($Max\_Page\_Number\_16Aligned << 12$).
9. MIC14 = $Interrupt\_Mask << 26$.
10. Read MIC0C untill MIC0C equal to ($Max\_Page\_Number\_16Aligned << 12$).
11. MIC24 = 80000000h.

15.7.4 Start Page CheckSum Process

This process will start checksum algorithm at page number #n.

1. Set page control bits of page number #n to 3h.

15.7.5 Stop Page CheckSum Process (DRAM 256M Byte Address Mode)

This process will stop checksum algorithm at page number #N.

1. Set page control bits of page number #N to 0h.
2. Write MIC10 [31:0] = (N | ($TAG << 16$)).
3. Read checksum value of page number #N, until this value equal to ($TAG << 16$).
4. If TAG equal to ffffh then TAG = 1, otherwise TAG = TAG + 1.

15.7.6 Stop Page CheckSum Process (DRAM 1G Byte Address Mode)

This process will stop checksum algorithm at page number #N.

1. Set page control bits of page number #N to 0h.
2. Write MIC10 [31:0] = (N | ($TAG << 18$)).
3. Read checksum value of page number #N, until this value equal to ($TAG << 18$).
4. If TAG equal to 3fffh then TAG = 1, otherwise TAG = TAG + 1.

15.8 Interrupt Behavior

- page error: MICE find page checksum error

- Write MIC18 [28] to 1: The action of clear First Page Error Flag
- MIC18 [28]: First Page Error Flag

- Write MIC1C [29] to 1: The action of clear Secondary Page Error Flag
- MIC1C [29]: Secondary Page Error Flag

- MIC14 [30]: Lost Page Error Flag

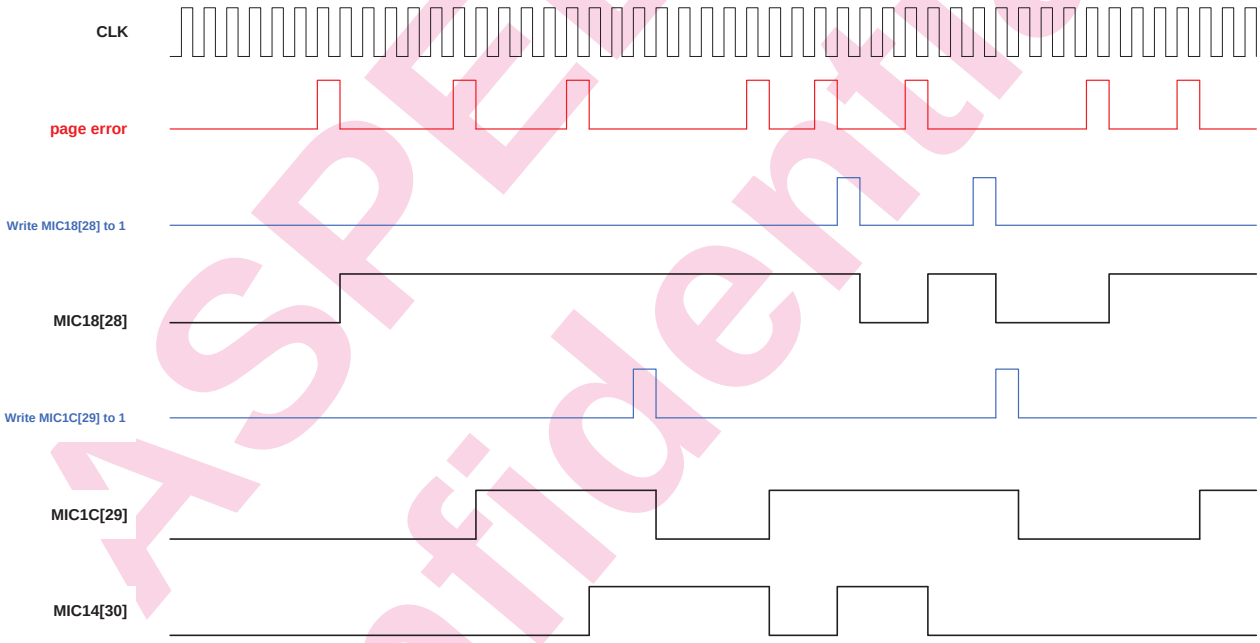


Figure 46: Interrupt Flag Priority

16 10/100/1G Ethernet MAC Controller

16.1 Overview

AST2500 integrates two 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
Physical address = (Base address of Ethernet MAC) + Offset

Important Notice: MACD0 and MACD4 are removed due to functional bug.

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)
MAC60: PHY Control Register (PHYCR)
MAC64: PHY Data Register (PHYDATA)
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 (RMII x2), Reduced Gigabit Media Independent Interface (RGMII x2)
- 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

Offset: 00h	MAC00: Interrupt Status Register (ISR)	Init = 0
Bit	**R/W**	**Description**
31:13		**Reserved (0)**
12	RW	**WOL_STUS: Wake-On-Lan Status** Clearing MAC74 Bit3~0 to clear this status flag.
11		**Reserved (0)**
10	RW	**HPTXBUF_UNAVA: High priority transmit buffer unavailable (Writing "1" to clear)**
9	RW	**PHYSTS_CHG: PHY link status change (Writing "1" to clear)**
8		**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		**Reserved (0)**
12	RW	**Interrupt enable of MAC00 Bit12**
11		**Reserved (0)**
10: 9	RW	**Interrupt enable of MAC00 Bit10~9**
8		**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		**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	W	**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	W	**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**
31:30		**Reserved (0)**
29:4	RW	**NPTXR_BADR: Base address of the normal priority transmit ring [29:4]** **The base address must be 16 bytes aligned**
3 :0		**Reserved (0)**

Offset: 24h	MAC24: Receive Ring Base Address Register (RXR_BADR)	Init = 0
Bit	**R/W**	**Description**
31:30		**Reserved (0)**
29:4	RW	**RXR_BADR: Base address of the receive ring [29:4]** **The base address must be 16 bytes aligned**
3 :0		**Reserved (0)**

Offset: 28h	MAC28: High Priority Transmit Poll Demand Register (HPTXPD)	Init = 0
Bit	**R/W**	**Description**
31:0	W	**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**
31:30		**Reserved (0)**
29:4	W	**HPTXR_BADR : Base address of the high priority transmit ring [29:4]** **The base address must be 16 bytes aligned**
3 :0		**Reserved (0)**

Offset: 30h	MAC30: Interrupt Timer Control Register (ITC)	Init = 0
Bit	**R/W**	**Description**
31:16		**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.
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)**.

to next page

from previous page

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		**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.
7 :5		**Reserved (0)**

to next page

from previous page

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		**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		**Reserved (0)**
Note : Recommended value = 0002_2500h		

Offset: 3Ch		**MAC3C: Debug Status Register (DBG_STS)**	**Init = 0**
Bit	**R/W**	**Description**	
31	R	**TX_DMABUF_EMPTY**	**(for debugging purpose only)**
30	R	**RX_DMABUF_EMPTY**	**(for debugging purpose only)**
29:27		**Reserved (0)**	
26:24	R	**RX_RqFSM[2:0]**	**(for debugging purpose only)**

to next page

from previous page

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

Offset: 40h **MAC40: Feature Register (FEAR)** **Init = 0000_0100h**

Bit	R/W	Description
31	R	**New MDC/MDIO Interface** 0: Old MDC/MDIO Interface 1: New MDC/MDIO Interface
30	R	**Internal loop back** 0: Disable internal loop back 1: Enable internal loop back
29:0		**Reserved (0)**

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

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

Bit	Attr.	Description
31:16		**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		**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		**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, received packets with length more than **2044** (**2048** for packets with VLAN tag) are treated as long frames. When cleared, received packets with length more than **1518** (**1522** for packets with VLAN tag) are treated as long frames.
12	RW	**RX_RUNT** Receive the incoming packet even if its length is less than **64** bytes. The incoming packet length must be longer than or equal to **10** bytes.
11	RW	**PHY link level sensitiver selection** 0: High-level sensitiver 1: Low-level sensitive
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	RW	**PHY link status detection** 1: Rising and falling edge trigger 0: Level sensitive
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		**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)**

to next page

from previous page

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: 60h **MAC60: PHY Control Register (PHYCR)** **Init = 0000_000fh**

Bit	R/W	Description
		Old MDC/MDIO Interface
31:28		**Reserved (0)**
27	RW	**MIIWR** Setting this bit to 1 initializes a write sequence to PHY. This bit would be auto cleared after the write operation is finished.
26	RW	**MIIRD** Setting this bit to 1 initializes a read sequence to PHY. This bit would be auto cleared after the read operation is finished.
25:21	RW	**REGAD: PHY register address**
20:16	RW	**PHYAD: PHY address**
15:6		**Reserved (0)**
5 :0	RW	**MDC_CYCTHR: MDC cycle threshold[5:0]** This field defines the period of **MDC time**. The period of **MDC time** is **(MAC60 [5:0] + 1) * 8 * (MHCLK clock period)**.
		New MDC/MDIO Interface
31:16	RW	**MIIWDATA: Write data to PHY**
15	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.
14:13		**Reserved (0)**
12	RW	**ST Code** 0: clause 45 1: clause 22
11:10	RW	**OP Code** **Clause 22:** 00: Reserved 01: Write 10: Read 11: Reserved **Clause 45:** 00: Address 01: Write 10: Post Read Increment Address 11: Read
9 :5	RW	**PHYAD/PRTAD: PHY address(clause 22) or Port address(clause 45)**
4 :0	RW	**REGAD/DEVAD: PHY register address(clause 22) or Device address(clause 45)**

Offset: 64h **MAC64: PHY Data Register (PHYDATA)** **Init = 0**

Bit	R/W	Description
		Old MDC/MDIO Interface

to next page

from previous page

31:16	R	**MIIRDATA: Read data from PHY**
15:0	RW	**MIIWDATA: Write data to PHY**
New MDC/MDIO Interface		
31:24	RW	**MDC_CYCTHR: MDC cycle threshold [7:0]** This field defines the period of **MDC time**. The period of **MDC time** is **(MAC64 [31:24] + 1) * 2 * (MHCLK clock period)**.
23:16		**Reserved (0)**
15:0	R	**MIIRDATA: Read data from PHY**

Offset: 68h **MAC6MAC68: 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	W	**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		**Reserved (0)**
4	RW	**RX_PAUSE: Receive pause frame (Writing "1" to clear)**
3	R	**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.
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		**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		**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		**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		**Reserved (0)**
2	RW	**MAGICPKT_EN: Magic packet event enable**
1	RW	**LINKCHG1_EN: Link change to 1 event enable**
0	RW	**LINKCHG0_EN: Link change to 0 event enable**

Offset: 74h	MAC74: Wake-On-LAN Status Register (WOL_SR)	Init = 0

Bit	R/W	Description
31:3		**Reserved (0)**
2	RW	**MAGICPKT_STS: Magic packet event status (Writing "1" to clear)**
1	RW	**LINKCHG1_STS: Link change to 1 event status (Writing "1" to clear)**
0	RW	**LINKCHG0_STS: Link change to 0 event status (Writing "1" to clear)**

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		**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	R	**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	R	**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	R	**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	R	**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		**Reserved (0)**
25:16	R	**TXCNT_MCOL: Packets transmitted OK with 2~15 collisions (for debugging purpose only)**
15:10		**Reserved (0)**
9 :0	R	**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		**Reserved (0)**
25:16	R	**TXCNT_LCOL: Packets failed in transmission (due to late collision)** (for debugging purpose only)
15:10		**Reserved (0)**
9 :0	R	**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		**Reserved (0)**
15:0	R	**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	R	**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	R	**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		**Reserved (0)**
25:16	R	**RXCNT_COL: Receive collision counter** (for debugging purpose only)
15:10		**Reserved (0)**
9 :0	R	**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		**Reserved (0)**
25:16	R	**RXCNT_CRCERR_ODD: Discarded CRC error packets with odd nibble.** (for debugging purpose only)
15:10		**Reserved (0)**
9 :0	R	**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		**Reserved (0)**

to next page

from previous page

25:16	R	**RXCNT_MUL: Received multicast packets** **(for debugging purpose only)**
15:10		**Reserved (0)**
9 :0	R	**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		**Reserved (0)**
25:16	R	**RXCNT_FTL: Received FTL packets** **(for debugging purpose only)**
15:10		**Reserved (0)**
9 :0	R	**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		**Reserved (0)**
9 :0	R	**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		**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		**Reserved (0)**

to next page

from previous page

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
23:16	RW	**Enable source port filter set #0 ~ #7 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		**Reserved (0)**

to next page

from previous page

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**

to next page

from previous page

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		**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**

to next page

from previous page

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 **2052** bytes (**2052** bytes if VLAN tag is inserted).
- 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		**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		**Reserved (0)**
13:0	RW	**TXBUF_SIZE: Transmit buffer size in byte** The transmit buffer size can not be zero.
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		**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		**Reserved (0)**

Offset: 0Ch	TXDES#3: Transmit Buffer Base Address	Init = X

Bit	R/W	Description
31:30		**Reserved (0)**
29:0	RW	**TXBUF_BADR: Transmit buffer base address [29:0]**

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 **2044** bytes (**2048** bytes for packets with VLAN tag).
- 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	R	**FRS: First Receive Segment descriptor** When set, it indicates that this is the first descriptor of a received packet.
28	R	**LRS: Last Receive Segment descriptor.** When set, it indicates that this is the last descriptor of a received packet.
27:26		**Reserved (0)**
25	R	**PAUSE_FRMAE: PAUSE_FRAME Pause frame** When set, it indicates that the receive packet is a pause frame.
24	R	**PAUSE_OPCODE: Pause frame OP code** When set, it indicates that there is pause frame OP code in the receive packet.
23	R	**FIFO_FULL: FIFO full** When set, it indicates that RX FIFO is full when the packet is received.
22	R	**RX_ODD_NB: Receive Odd Nibbles** When set, it indicates receiving a packet with odd nibbles.
21	R	**RUNT: Runt packet** When set, it indicates that the received packet length is less than 64 bytes.
20	R	**FTL: Frame Too Long** When set, it indicates that the received packet length exceeds the long frame length. If JUMBO_LF=0, the long frame length is **1518** (**1522** for the receive packet with VLAN tag) bytes. If JUMBO_LF=1, the long frame length is **2044** (**2048** for the receive packet with VLAN tag) bytes.

to next page

from previous page

19	R	**CRC_ERR: CRC error** When set, it indicates that the CRC error occurs on the received packet.
18	R	**RX_ERR: Receive error** When set, it indicates that receive error happens when receiving a packet.
17	R	**BROADCAST: Broadcast frame.** When set, it indicates that the received packet is a broadcast frame.
16	R	**MULTICAST: Multicast frame.** When set, it indicates that the received packet is a multicast frame.
15	R	**UDP_OPT_CHKSUM: UDP Optional Checksum** It indicates that the UDP header with optional checksum.
14		**Reserved (0)**
13:0	R	**VDBC: valid data byte count.** The field indicates the valid data in the receive buffer. **The unit is 1 byte**.
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		**Reserved (0)**
27	R	**IPCS_FAIL: IP checksum failure** When set, MAC engine detects IP checksum failure.
26	R	**UDPCS_FAIL: UDP checksum failure** When set, MAC engine detects UDP checksum failure.
25	R	**TCPCS_FAIL: TCP checksum failure** When set, MAC engine detects TCP checksum failure.
24	R	**VLAN_AVA: VLAN Tag Available** When set, the receive packet is a packet with IEEE 802.1Q VLAN Tag Type.
23	R	**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	R	**LLC_PKT: LLC packet** When set, it indicates that the receive packet is LLC packet.
21:20	R	**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	R	**IPv6: IPv6 packet** When set, it indicates that the frame is IPv6 packet
18	R	**SKIP_FRAME: Skip Frame** When set, it indicates that the frame is skipped
17	R	**CHKSUM_LEN_FAIL: Checksum Length Fail** When set, it indicates that the checksum length fail
16		**Reserved (0)**

to next page

from previous page

15:0	R	**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)
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		**Reserved (0)**
15:0	R	**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**
31:30		**Reserved (0)**
29:0	RW	**RXBUF_BADR: Receive buffer base address [29:0]**

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 USB2.0 Virtual Hub Controller

17.1 Overview

USB2.0 Controller implements 1 set of USB Hub register and 5 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

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
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

17.2 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 5 downstream device ports.
- Supports 15 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 15 programmable endpoints.
- Supports USB remote wake-up function (Suspend/Resume operation).

17.3 Procedure to enable USB2.0 Hub port

1. Set SCU90 bit[29] = 0, select USB2.0 port 1 mode as device
2. Set SCU04 bit[14] = 1, enable controller reset
3. Set SCU0C bit[14] = 1, enable PHY clock
4. wait 10 ms for PLL locking
5. Set SCU04 bit[14] = 0, disable controller reset
6. Set HUB00 bit[11] = 1, disable PHY reset
7. Load driver

17.3.1 Procedure to enable USB2.0 eye diagram measurement

- Measure Host TX waveform, enable device port termination
  1. Measurement point on the AST2500 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 AST2500 into high speed termination mode.
  3. Enter firmware u-boot
  4. mw 1e6e2000 1688a8a8
  5. mw 1e6e200c eff47e8b
  6. mw 1e6e2004 0
  7. mw 1e6a0000 841
  8. If termination not set correctly, then do below 2 steps repeatly until termination was correct.
  9. mw 1e6a0000 0
  10. mw 1e6a0000 841
- 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 1e6e200c eff47e8b
  6. mw 1e6e2004 0
  7. mw 1e6a0000 c41

17.4 Registers : Base Address = 0x1E6A:0000

17.4.1 Address Definition

Offset	Size(Byte)	Description
0x03F-0x000	64	Root/Global Device Register
0x07F-0x040	64	Reserved
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
0x0FF-0x0B0	80	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
0x1FF-0x150	176	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

17.4.2 Root/Global Register Definition

Offset: 00	HUB00: Root Function Control & Status Register	Init = 0
Bit	**R/W**	**Description**
31	R	**USB PHY clock enable status** 0: USB PHY clock is disabled by SCU9C bit[14]. 1: USB PHY clock is enabled. The procedure to enable USB2.0 controller was updated for A1 or newer chip versions: 1. Enable USB2.0 clock running(SCU9C[14] = 1), wait 10 ms for clock stable 2. Disable USB2.0 global reset by setting SCU04[14] = 0 3. Disable USB2.0 PHY reset by setting HUB00[11] = 1 4. Start using USB2.0 controller
30:26	RW	**Reserved (0)**
25	RW	**PHY Self-loopback test enable** 0: Disable 1: Enable PHY internal loopback test mode Turn on this bit and bit[12] together.
24	RW	**DN 15K pull down enable**
23	RW	**DP 15K pull down enable**
22:19	RW	**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	R	**Loop Back test result** 0: Fail 1: Pass This status will be cleared when disabling USB Test Mode
14	R	**Loop Back test status** 0: Not finish 1: Finished This status will be cleared when disabling USB Test Mode

to next page

from previous page

13	R	**USB PHY BIST result** 0: Pass 1: Fail 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: 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	HUB04: Root Configuration Setting Register	Init = 0

Bit	R/W	Description
31:24		**Reserved**
23:16	R	**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		**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		**Reserved (0)**
17	RW	**Enable Programmable Endpoint Pool NAK Interrupt**
16	RW	**Enable Programmable Endpoint Pool ACK/STALL Interrupt**
15:14		**Reserved (0)**
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**

to next page

from previous page

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
31:19		**Reserved (0)**
18	R	**~~USB command bus is dead locked~~**
17	R	**Programmable Endpoint Pool NAK Interrupt Occurs**
16	R	**Programmable Endpoint Pool ACK/STALL Interrupt Occurs**
15:14		**Reserved (0)**
13	R	**Device #5 Controller Interrupt Occurs**
12	R	**Device #4 Controller Interrupt Occurs**
11	R	**Device #3 Controller Interrupt Occurs**
10	R	**Device #2 Controller Interrupt Occurs**
9	R	**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.

to next page

from previous page

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
31:15		**Reserved (0)**
14:0	RW	**Programmable Endpoint ACK Interrupt Enable** bit 0: Programmable Endpoint number #0 bit 1: Programmable Endpoint number #1 .... bit14: Programmable Endpoint number #14 This register is to control the ACK interrupt enable bits of the 15 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
31:15		**Reserved (0)**
14:0	RW	**Programmable Endpoint NAK Interrupt Enable** bit 0: Programmable Endpoint number #0 bit 1: Programmable Endpoint number #1 .... bit14: Programmable Endpoint number #14 This register is to control the ACK interrupt enable bits of the 15 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
31:15		**Reserved (0)**

to next page

from previous page

14:0	RW	**Programmable Endpoint ACK Interrupt Occurs** bit 0: Programmable Endpoint number #0 bit 1: Programmable Endpoint number #1 .... bit14: Programmable Endpoint number #14 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
31:15		**Reserved (0)**
14:0	RW	**Programmable Endpoint NAK Interrupt Occurs** bit 0: Programmable Endpoint number #0 bit 1: Programmable Endpoint number #1 .... bit14: Programmable Endpoint number #14 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 = 0x33F

Bit	R/W	Description
31:10		**Reserved (0)**
9	RW	**Enable Programmable Endpoint Pool software Reset**
8	RW	**Enable DMA Controller software Reset**
7:6		**Reserved (0)**
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**

to next page

from previous page

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	HUB24: USB Status Register (for debugging purpose only)	Init = X
Bit	**R/W**	**Description**
31	R	**USB Suspend State**
30	R	**USB Bus Reset State**
29	R	**USB Bus Line State DN**
28	R	**USB Bus Line State DP**
27	R	**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	R	**USB Last Frame Number record**
15	R	**UTMI State XcvrSelect** 0: High Speed 1: Full Speed
14	R	**UTMI State TermSelect** 0: High Speed 1: Full Speed
13:12	R	**UTMI State OPMode** 0: Normal mode 1: Non Driving 2: Disable Bit-Stuff and NRZI encoding 3: Reserved
11:8	R	**Endpoint Number of the Last USB Transaction**
7		**Reserved (0)**
6:0	R	**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		**Reserved (0)**
8	W	**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		**Reserved (0)**
4:0	W	**Programmable Endpoint Index** 0-14: Endpoint number index 15-31: Invalid This index value determines which Endpoint data toggle bit will be initialized. Reading this register always returns "0".

to next page

from previous page

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		**Reserved (0)**
25:16	RW	**Isochronous OUT Failure Counter**
15:10		**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		**Reserved (0)**
22:16	R	**Endpoint 0 OUT received data byte count**
15		**Reserved (0)**
14:8	RW	**Endpoint 0 IN data byte count for transfer**
7:3		**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
31:30		**Reserved (0)**
29:3	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.

to next page

from previous page

2:0		**Reserved (0)**

Offset: 38	**HUB38: Endpoint 1 Control/Status Register**	**Init = 0**
Bit	**R/W**	**Description**
31:3		**Reserved (0)**
2	W	**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**
31:6		**Reserved**
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.		

17.4.3 Device #1 — #5 Register Definition

Offset: 00	**DEV00: Downstream Device Function Enable Control Register**	**Init = 0**
Bit	**R/W**	**Description**
31:15		**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		**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		**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		**Reserved (0)**
28	R	**CSPLIT IN Wait** **(for debugging purpose only)**

to next page

from previous page

27	R	**Normal IN Wait** **(for debugging purpose only)**
26	R	**Start SPLIT Cycle** **(for debugging purpose only)**
25:24	R	**Status of Transmit DMA State Machine** **(for debugging purpose only)** 00 : Idle 01 : DMA Request 10 : DMA Done and Data Ready 11 : Reserved
23		**Reserved (0)**
22:16	R	**Endpoint 0 OUT received data byte count**
15		**Reserved (0)**
14:8	RW	**Endpoint 0 IN data byte count for transfer**
7:3		**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
31:30		**Reserved (0)**
29:3	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.
2:0		**Reserved (0)**

17.4.4 Programmable Endpoint #0 — #14 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		**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 4095 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 disable the auto data toggle mode. **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		**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
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
31:21		**Reserved (0)**
20	R	**Occupied Transmit IN Buffer Status** **(for debugging purpose only)** 0 : No buffer 1 : 1 buffer occupied
19:16	R	**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.

to next page

from previous page

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	R	**The Current Interrupt Generation Flag** **(for debugging purpose only)** This bit shows the interrupt generation status, fetched from the current stage descriptor.
11	R	**CSPLIT IN Wait** **(for debugging purpose only)**
10:9	R	**Auto Data Toggle Count** **(for debugging purpose only)**
8	R	**Start SPLIT Cycle** **(for debugging purpose only)**
7:4	R	**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.

to next page

from previous page

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
31:30		**Reserved (0)**
29:3	RW	**Base address** Descriptor Enabled : Descriptor list base address. Descriptor Disabled : DMA data buffer base address. 8 bytes(64 bits) boundary
2:0		**Reserved (0)**

EPP0C: DMA Descriptor List Read(DMA)/Write(CPU) Pointer and Status

Offset: 0C **Init = 0**

Bit	Attr.	Description
31	R	**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		**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.		

17.4.5 Programmable Endpoint DMA Descriptor Definition

Offset: 31:0		DES_0: Data Buffer Base Address	Init = X
Bit	**R/W**	**Description**	
31:29		**Reserved**	
28:3	RW	**DMA Data Buffer Base Address** 1. OUT direction endpoint must be 8 bytes (64 bits) address boundary. 2. IN direction endpoint can be byte boundary.	
2:0	RW	**DMA Data Buffer Base Address Byte Start** Define the start byte position of the DMA data buffer.	

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	R	**Device Port Number** 000 : Root Hub 001 : Port1 010 : Port2 011 : Port3 100 : Port4 101 : Port5 This port number status only used for indicating the received transaction status. No any meaning for transmit path.	
59:56	R	**Endpoint Number** This endpoint number status only used for indicating the received transaction status. No any meaning for transmit path.	
55		**Reserved**	
54:48	R	**Device Address** This device address status only used for indicating the received transaction status. No any meaning for transmit path.	
47:46	R	**Data Packet PID** 00 : DATA0 01 : DATA2 10 : DATA1 11 : MDATA This is the Data PID that current packet received.	
45	R	**End of Packet(E)** This is used for Isochronous OUT to a Full Speed device.	

to next page

from previous page

44	R	**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	R	**OUT Packet Valid Flag** This bit will be written 1 by hardware when OUT Transaction DMA done.
42:32	R	**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		**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.
45:44		**Reserved**
43:32	RW	**Packet Length in Bytes** CPU must initialize this field to indicate the packet length for transmit.

17.4.6 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.

17.5 Software Programming Guide

17.5.1 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 SCU04[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 15 programmable endpoints can be reset by the following conditions:

1. Global reset at SCU04[14] will reset all registers to its initial value defined.
2. HUB20 bit[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.

17.5.2 Initialization Sequence

Hub Connection

1. Enable USB2.0 clock running(SCU9C[14] = 1), wait 10 ms for clock stable
2. Disable USB2.0 global reset by setting SCU04[14] = 0
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 SCU04[14] = 1
2. Disable USB2.0 clock running by setting SCU9C[14] = 0

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

17.5.3 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.

17.5.4 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.

17.5.5 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.

17.5.6 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.

17.5.7 Procedure to enable Interrupt

The USB2.0 interrupt function must follow the sequence to enable.

1. Enable USB2.0 clock at SCU9C[14]=1 and wait 10 ms for clock stable.
2. Disable USB2.0 global reset at SCU04[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.

17.5.8 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.

18 USB2.0 Device Controller (Removed)

19 USB2.0/1.1 Host Controller

19.1 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; and only Memory-mapped 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, 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
Base address of EHCI Host Controller #2 = 0x1E6A_3000

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
EHCI8C: Hardware Revision Number Register

19.2 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

19.3 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)
  - no companion controller (N_CC = 0)
  - no port indicator
  - no debug port
  - no 64-bit addressing
  - programmable frame list size: 1024, 512, 256
  - no asynchronous schedule park
  - no EHCI extended capabilities

19.4 Procedure to enable USB Host port

19.4.1 USB2.0 Host port 1

1. Set SCU90 bit[29] = 1, select USB2.0 port 1 mode as host
2. Set SCU04 bit[14] = 1, enable controller reset
3. Set SCU0C bit[14] = 1, enable PHY clock
4. wait 10 ms for PLL locking
5. Set SCU04 bit[14] = 0, disable controller reset

19.4.2 USB2.0 Host port 2

1. Set SCU94 bit[14:13] = ”10”, select USB2.0 port 2 mode as host
2. Set SCU04 bit[3] = 1, enable controller reset
3. Set SCU0C bit[7] = 0, enable PHY clock
4. wait 10 ms for PLL locking
5. Set SCU04 bit[3] = 0, disable controller reset
6. Load EHCI driver

19.4.3 UHCI controller

1. USB2.0 Port 2 must be enabled first
2. Set SCU04 bit[15] = 1, enable controller reset
3. Set SCU0C bit[9] = 0, enable controller clock
4. wait 10 us for clock stable
5. Set SCU04 bit[15] = 0, disable controller reset
6. Load UHCI driver

19.5 UHCI Registers : Base Address = 0x1E6B:0000

Offset: 00h	UHCI00: USB Command Register (USBCMD)	Init = 0
Bit	**R/W**	**Description**
31:8		**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		**Reserved (0)**
5	RW	**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	RW	**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	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 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	RW	**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	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 Bit 0 are set.
0	RW	**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		**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		**Reserved (0)**

Offset: 40h	UHCI40: Test Control Register	Init = 0

Bit	R/W	Description
31:14		**Reserved**
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.
9	R	**Loop Back test result** 0: Fail 1: Pass This status will be cleared when disabling USB Test Mode
8	R	**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		**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		**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		**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		**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	R	**Low Speed Device Attached** 0: Full speed device. 1: Low speed device is attached to this port. Writes have no effect.
7		**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	R	**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	RW	**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	RW	**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	R	**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.

19.6 EHCI Registers

Offset: 00h	**EHCI00: Capability Registers Length (CAPLENGTH)**	**Init = 0x01000020**
Bit	**R/W**	**Description**
31:16	R	**Host Controller Interface Version Number (HCIVERSION)**
15:8		**Reserved (0)**
7:0	R	**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	R	**Reserved (0)**
23:20	R	**Debug Port Number**
19:17	R	**Reserved (0)**
16	R	**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	R	**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	R	**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	R	**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	R	**Reserved (0)**

to next page

from previous page

4	R	**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	R	**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 = 0x00000076

Bit	R/W	Description
31:16	R	**Reserved (0)**
15:8	R	**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	R	**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	R	**Reserved (0)**
2	R	**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	R	**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	R	**64-bit Addressing Capability** 0: data structures using 32-bit address memory pointers 1: data structures using 64-bit address memory pointers

Offset: 0Ch-10h EHCI0C: Companion Port Route Description (HCSP-PORTROUTE) Init = 0x0

Bit	R/W	Description
63:0	R	**Reserved (0)**

Offset: 20h	EHCI20: USB Command Register (USBCMD)	Init = 0x00080B00
Bit	**R/W**	**Description**
31:24	R	**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	R	**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	R	**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	R	**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	R	**Reserved (0)**

to next page

from previous page

15	R	**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	R	**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	R	**Reclamation** This is a read-only status bit, which is used to detect an empty asynchronous schedule.
12	R	**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	R	**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 **EHCI28: USB Interrupt Enable Register (USBINTR)** **Init = 0**

Bit	R/W	Description
31:6		**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		**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		**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		**Reserved (0)**

Offset: 60h EHCI60: Configure Flag Register (CONFIGFLAG) Init = 0

Bit	R/W	Description
31:1		**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		**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		**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		**Reserved (1)**
11:10	R	**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		**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		**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	R	**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
31:7		**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_025F

Bit	R/W	Description
31:11		**Reserved (0)**
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: 8Ch	EHCI8C: Hardware Revision Number Register	Init = 0x0000_0001

Bit	R/W	Description
31:8		**Reserved (0)**
7:0	R	**Hardware revision number**

20 Interrupt Controller

20.1 Overview

Interrupt Controller (VIC) is an AMBA salve device directly connected to AHB bus. It provides a hardware interface to interrupt ARM CPU, Coprocessor or System LPC SERIRQ.

Limitation: Each interrupt source can only be assigned to an interrupt target.

20.1.1 ARM CPU Interrupt Controller

It provides two kinds of priority levels for different interrupt requests.

Fast Interrupt Request (FIQ): For higher priority and low latency interrupt requests
Interrupt Request (IRQ): For general interrupt requests

Its highly recommending that only one of all the interrupt sources is assigned as FIQ interrupt request. VIC supports up to 64 interrupt requests. Each interrupt request is fixed to its designed trigger modes. AST2500 has assigned different interrupt requests for VIC as shown in Section 8.

VIC implements the following 26 registers to support various interrupt functions. Each register has its own specific offset value to derive its physical address location.

Hardware Heart Beat Function

VIC controller integrates a hardware Heart Beat output function. With different LED blinking frequency to indicate different hardware states. Below are the list of hardware blinking types:

1 10Hz : ARM Running on Flash (instruction fetch from flash)
2 2Hz : ARM Running on DRAM without interrupt enabled (instruction fetch from DRAM)
3 0.5Hz : ARM Running on DRAM with interrupt monitor enabled (normal operation mode)
4 0.1Hz : Abnormal mode, some interrupts are not serviced for over 2 seconds
5 0Hz : Always dark indicates FW is not running or dead

Firmware can switch the Heart Beat blink type to software mode, this is the same as the traditional GPIO mode.

Base address of VIC = 0x1E6C_0000
Physical address = (Base address of VIC) + Offset

Legacy Mapping

VIC00: (L)IRQ Status Register
VIC04: (L)FIQ Status Register
VIC08: (L)Raw Interrupt Status Register
VIC0C: (L)Interrupt Selection Register
VIC10: (L)Interrupt Enable Register
VIC14: (L)Interrupt Enable Clear Register
VIC18: (L)Software Interrupt Register
VIC1C: (L)Software Interrupt Clear Register
VIC24: (L)Interrupt Sensitivity Register
VIC28: (L)Interrupt Both Edge Trigger Control Register
VIC2C: (L)Interrupt Event Register
VIC38: (L)Edge-Triggered Interrupt Clear Register

Hardware Heart Beat Control

VIC50: (L)Interrupt Monitor Mask Register
VIC54: (H)Interrupt Monitor Mask Register
VIC60: Hardware Heart Beat Status Register
VIC64: Heart Beat LED Output

New Mapping

VIC80: (L)IRQ Status Register
VIC84: (H)IRQ Status Register
VIC88: (L)FIQ Status Register
VIC8C: (H)FIQ Status Register
VIC90: (L)Raw Interrupt Status Register
VIC94: (H)Raw Interrupt Status Register
VIC98: (L)Interrupt Selection Register
VIC9C: (H)Interrupt Selection Register
VICA0: (L)Interrupt Enable Register
VICA4: (H)Interrupt Enable Register
VICA8: (L)Interrupt Enable Clear Register
VICAC: (H)Interrupt Enable Clear Register
VICB0: (L)Software Interrupt Register
VICB4: (H)Software Interrupt Register
VICB8: (L)Software Interrupt Clear Register
VICBC: (H)Software Interrupt Clear Register
VICC0: (L)Interrupt Sensitivity Register
VICC4: (H)Interrupt Sensitivity Register
VICC8: (L)Interrupt Both Edge Trigger Control Register
VICCC: (H)Interrupt Both Edge Trigger Control Register
VICD0: (L)Interrupt Event Register
VICD4: (H)Interrupt Event Register
VICD8: (L)Edge-Triggered Interrupt Clear Register
VICDC: (H)Edge-Triggered Interrupt Clear Register
VICE0: (L)Edge-Triggered Interrupt Status Register
VICE4: (H)Edge-Triggered Interrupt Status Register

20.1.2 System LPC Interrupt Controller (SVIC)

AST2500 supports another interrupt path up to the system through LPC SERIRQ or SMI pin. It support one way for the system can use or get the ststus of the AST2500 internal IPs.

SVIC supports partial (18) interrupt requests for system. The interrupt requests are shown in Section 8 Table 74.

SVIC implements the following 9 registers to support various interrupt functions. Each register has its own specific offset value to derive its physical address location.

Base address of SVIC = 0x1E6C_1000
Physical address = (Base address of SVIC) + Offset

SVIC00: IRQ Status Register
SVIC08: Raw Interrupt Status Register
SVIC10: Interrupt Enable Register
SVIC14: Interrupt Enable Clear Register

SVIC18: Software Interrupt Register
SVIC1C: Software Interrupt Clear Register
SVIC24: Interrupt Sensitivity Register
SVIC28: Interrupt Both Edge Trigger Control Register
SVIC2C: Interrupt Event Register
SVIC38: Edge-Triggered Interrupt Clear Register
SVIC3C: Edge-Triggered Interrupt Status Register

20.1.3 Coprocessor Interrupt Controller (CVIC)

AST2500 supports an dedicated interrupt path for the Coprocessor. CVIC supports partial (32) interrupt requests for Coprocessor. The interrupt requests are shown in Section 8 Table 73.

CVIC implements the following 9 registers to support various interrupt functions. Each register has its own specific offset value to derive its physical address location.

Base address of VIC = 0x1E6C_2000
Physical address = (Base address of VIC) + Offset

CVIC00: IRQ Status Register
CVIC08: Raw Interrupt Status Register
CVIC10: Interrupt Enable Register
CVIC14: Interrupt Enable Clear Register
CVIC18: Software Interrupt Register
CVIC1C: Software Interrupt Clear Register
CVIC24: Interrupt Sensitivity Register
CVIC28: Interrupt Both Edge Trigger Control Register
CVIC2C: Interrupt Event Register
CVIC38: Edge-Triggered Interrupt Clear Register
CVIC3C: Edge-Triggered Interrupt Status Register

20.2 Features

- Directly connected to AHB bus interface
- Support up to 64 interrupt sources for ARM CPU
- Support up to 18 interrupt sources for System LPC
- Support up to 32 interrupt sources for Corprocessor
- Hardwired pre-defined interrupt trigger type settings

20.3 VIC Registers : Base Address = 0x1E6C:0000

Offset: 00h	VIC00: (L)IRQ Status Register	Init = 0
Bit	**R/W**	**Description**
31:0	R	**IRQ status (IRQ0-IRQ31)** Shows the status of the interrupt after masking by **VIC10** and **VIC0C** registers. Value ”1” indicates that the interrupt is active, and generates an interrupt to the processor.

Offset: 04h	VIC04: (L)FIQ Status Register	Init = 0

Bit	R/W	Description
31:0	R	**FIQ status (IRQ0-IRQ31)** Shows the status of the interrupt after masking by the **VIC10** and **VIC0C** register. Value ”1” indicates that the interrupt is active, and generates an interrupt to the processor.

Offset: 08h	VIC08: (L)Raw Interrupt Status Register	Init = 0

Bit	R/W	Description
31:0	R	**Raw interrupt status (IRQ0-IRQ31)** Shows the status of the interrupt before masking by the **VIC10** register. Value ”1” indicates that an interrupt request is active before masking.

Offset: 0Ch	VIC0C: (L)Interrupt Selection Register	Init = 0

Bit	R/W	Description
31:0	RW	**Interrupt selection (IRQ0-IRQ31)** Select the types of interrupt for ARM interrupt request input: 1: FIQ interrupt 0: IRQ interrupt

Offset: 10h	VIC10: (L)Interrupt Enable Register	Init = 0

Bit	R/W	Description
31:0	RW	**Enable the interrupt source (IRQ0-IRQ31)** When read: 1: interrupt enable. Allow interrupt request to processor 0: interrupt disable When write: 1: enables the interrupt 0: no effect To clear this register bits from ”1” to ”0”, write the corresponding bits in VIC14 with value ’1’.

Offset: 14h	VIC14: (L)Interrupt Enable Clear Register	Init = 0

Bit	R/W	Description
31:0	W	**Clear bits in the VIC10 register (IRQ0-IRQ31)** Write ’1’ clears the corresponding bit in the VIC10 register to value ”0”, and write ’0’ has no effect.

Offset: 18h	VIC18: (L)Software Interrupt Register	Init = 0

Bit	R/W	Description
31:0	RW	**Software Interrupt Generation (IRQ0-IRQ31)** 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.

Offset: 1Ch	VIC1C: (L)Software Interrupt Clear Register	Init = 0

Bit	R/W	Description
31:0	W	**Clear bits in the VIC18 register (IRQ0-IRQ31)** Write '1' clears the corresponding bit in the VIC18 register to value "0", and write '0' has no effect.

Offset: 24h	VIC24: (L)Interrupt Sensitivity Register	Init = 0xFFF8FFFF

Bit	R/W	Description
31:0	R	**Sensitivity type of interrupt for interrupt request (IRQ0-IRQ31)** 1: level sensitive 0: edge trigger

Offset: 28h	VIC28: (L)Interrupt Both Edge Trigger Control Register	Init = 0x00070000

Bit	R/W	Description
31:0	R	**Both or single edge for edge-trigger interrupt request (IRQ0-IRQ31)** 1: both edge 0: single edge When sensitivity type is level sensitive, this register has no effect.

Offset: 2Ch	VIC2C: (L)Interrupt Event Register	Init = 0xFFF8FFFF

Bit	R/W	Description
31:0	R	**Sensitivity type of interrupt for interrupt request** 1: High-level sensitive or rising edge triggered. 0: Low-level sensitive or falling edge triggered.

Offset: 38h	VIC38: (L)Edge-Triggered Interrupt Clear Register	Init = 0

Bit	R/W	Description
31:0	W	**Clear bits in the edge detection register (IRQ0-IRQ31)** Write '1' clears the corresponding bit in the edge detection register, and write '0' has no effect. When select edge triggered interrupt sensitivity type, firstly it must clear the detection register by writing '1' in this register, then it can enable the interrupt in VIC10, otherwise old interrupt status will take effect again.

Offset: 50h	VIC50: (L)Interrupt Monitor Mask Register	Init = 0
Offset: 54h	VIC54: (H)Interrupt Monitor Mask Register	Init = 0

Bit	Attr.	Description
63:0	RW	**IRQ Raw status monitor mask (IRQ0-IRQ63)** 0: disable 1: enable Mask for selecting the interrupt source for heardware heart beat event monitor.

Offset: 60h	VIC60: Hardware Heart Beat Status Register	Init = 0x03

Bit	R/W	Description
31:18		**Reserved**

to next page

from previous page

17:16	R	**Heart Beat States** 00: 10Hz, Running on Flash (instruction fetch from flash) 01: 2Hz, Running on DRAM without interrupt enabled (instruction fetch from DRAM) 10: 0.5Hz, Running on DRAM with interrupt monitor enabled (normal operation mode) 11: 0.1Hz, Abnormal mode, some interrupts are not serviced for over 2 seconds If always dark indicates FW is not running or dead
15:5		**Reserved**
4	RW	**Software blinking mode** 0: Hardware blinking mode 1: Software blinking mode. Software toggle VIC64 to control the blinking. This is the same as traditional heart beat mode that using GPIO.
3:0	RW	**Blinking debounce timer** This value is used to set a debounce time for the blinking. The unit is blinking count under current blinking rate.

Offset: 64h **VIC64: Heart Beat LED Output** **Init = 0x00**

Bit	R/W	Description
31:1		**Reserved**
0	RW	**LED output** 0: LED is dark 1: LED is bright At hardware blink mode, this bit reflects the current LED status. At software blink mode, this bit controls the LED output.

Offset: 80h **VIC80: (L)IRQ Status Register** **Init = 0**

Bit	R/W	Description
31:0	R	**IRQ status (IRQ0-IRQ31)** Shows the status of the interrupt after masking by **VIC98** and **VICA0** registers. Value ”1” indicates that the interrupt is active, and generates an interrupt to the processor.

Offset: 84h **VIC84: (H)IRQ Status Register** **Init = 0**

Bit	R/W	Description
31:0	R	**IRQ status (IRQ32-IRQ63)** Shows the status of the interrupt after masking by **VIC9C** and **VICA4** registers. Value ”1” indicates that the interrupt is active, and generates an interrupt to the processor.

Offset: 88h **VIC88: (L)FIQ Status Register** **Init = 0**

Bit	R/W	Description
31:0	R	**FIQ status (IRQ0-IRQ31)** Shows the status of the interrupt after masking by the **VIC98** and **VICA0** register. Value ”1” indicates that the interrupt is active, and generates an interrupt to the processor.

Offset: 8Ch **VIC8C: (L)FIQ Status Register** **Init = 0**

Bit	R/W	Description
31:0	R	**FIQ status (IRQ32-IRQ63)** Shows the status of the interrupt after masking by the **VIC9C** and **VICA4** register. Value ”1” indicates that the interrupt is active, and generates an interrupt to the processor.

Offset: 90h	VIC90: (L)Raw Interrupt Status Register	Init = 0
Bit	**R/W**	**Description**
31:0	R	**Raw interrupt status (IRQ0-IRQ31)** Shows the status of the interrupt before masking by the **VICA0** register. Value ”1” indicates that an interrupt request is active before masking.

Offset: 94h	VIC94: (H)Raw Interrupt Status Register	Init = 0
Bit	**R/W**	**Description**
31:0	R	**Raw interrupt status (IRQ32-IRQ63)** Shows the status of the interrupt before masking by the **VICA4** register. Value ”1” indicates that an interrupt request is active before masking.

Offset: 98h	VIC98: (L)Interrupt Selection Register	Init = 0
Bit	**R/W**	**Description**
31:0	RW	**Interrupt selection (IRQ0-IRQ31)** Select the types of interrupt for ARM interrupt request input: 1: FIQ interrupt 0: IRQ interrupt

Offset: 9Ch	VIC9C: (H)Interrupt Selection Register	Init = 0
Bit	**R/W**	**Description**
31:0	RW	**Interrupt selection (IRQ32-IRQ63)** Select the types of interrupt for ARM interrupt request input: 1: FIQ interrupt 0: IRQ interrupt

Offset: A0h	VICA0: (L)Interrupt Enable Register	Init = 0
Bit	**R/W**	**Description**
31:0	RW	**Enable the interrupt source (IRQ0-IRQ31)** When read: 1: interrupt enable. Allow interrupt request to processor 0: interrupt disable When write: 1: enables the interrupt 0: no effect To clear this register bits from ”1” to ”0”, write the corresponding bits in VICA8 with value ’1’.

Offset: A4h	VICA4: (H)Interrupt Enable Register	Init = 0
Bit	**R/W**	**Description**
31:0	RW	**Enable the interrupt source (IRQ32-IRQ63)** When read: 1: interrupt enable. Allow interrupt request to processor 0: interrupt disable When write: 1: enables the interrupt 0: no effect To clear this register bits from ”1” to ”0”, write the corresponding bits in VICAC with value ’1’.

Offset: A8h	VICA8: (L)Interrupt Enable Clear Register	Init = 0
Bit	**R/W**	**Description**
31:0	W	**Clear bits in the VICA0 register (IRQ0-IRQ31)** Write ’1’ clears the corresponding bit in the VICA0 register to value ”0”, and write ’0’ has no effect.

Offset: ACh	VICAC: (H)Interrupt Enable Clear Register	Init = 0
Bit	**R/W**	**Description**
31:0	W	**Clear bits in the VICA4 register (IRQ32-IRQ63)** Write ’1’ clears the corresponding bit in the VICA4 register to value ”0”, and write ’0’ has no effect.

Offset: B0h	VICB0: (L)Software Interrupt Register	Init = 0
Bit	**R/W**	**Description**
31:0	RW	**Software Interrupt Generation (IRQ0-IRQ31)** 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.

Offset: B4h	VICB4: (H)Software Interrupt Register	Init = 0
Bit	**R/W**	**Description**
31:0	RW	**Software Interrupt Generation (IRQ32-IRQ63)** 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.

Offset: B8h	VICB8: (L)Software Interrupt Clear Register	Init = 0
Bit	**R/W**	**Description**
31:0	W	**Clear bits in the VICB0 register (IRQ0-IRQ31)** Write ’1’ clears the corresponding bit in the VICB0 register to value ”0”, and write ’0’ has no effect.

Offset: BCh	VICBC: (H)Software Interrupt Clear Register	Init = 0

Bit	R/W	Description
31:0	W	**Clear bits in the VICB4 register (IRQ32-IRQ63)** Write '1' clears the corresponding bit in the VICB4 register to value "0", and write '0' has no effect.

Offset: C0h	VICC0: (L)Interrupt Sensitivity Register	Init = 0xFFF8FFFF

Bit	R/W	Description
31:0	R	**Sensitivity type of interrupt for interrupt request (IRQ0-IRQ31)** 1: level sensitive 0: edge trigger

Offset: C4h	VICC4: (H)Interrupt Sensitivity Register	Init = 0xFFFFCF07

Bit	R/W	Description
31:0	R	**Sensitivity type of interrupt for interrupt request (IRQ32-IRQ63)** 1: level sensitive 0: edge trigger

Offset: C8h	VICC8: (L)Interrupt Both Edge Trigger Control Register	Init = 0x00070000

Bit	R/W	Description
31:0	R	**Both or single edge for edge-trigger interrupt request (IRQ0-IRQ31)** 1: both edge 0: single edge When sensitivity type is level sensitive, this register has no effect.

Offset: CCh	VICCC: (H)Interrupt Both Edge Trigger Control Register	Init = 0x000000F8

Bit	R/W	Description
31:0	R	**Both or single edge for edge-trigger interrupt request (IRQ32-IRQ63)** 1: both edge 0: single edge When sensitivity type is level sensitive, this register has no effect.

Offset: D0h	VICD0: (L)Interrupt Event Register	Init = 0xFFF8FFFF

Bit	R/W	Description
31:0	R	**Sensitivity type of interrupt for interrupt request (IRQ0-IRQ31)** 1: High-level sensitive or rising edge triggered. 0: Low-level sensitive or falling edge triggered.

Offset: D4h	VICD4: (H)Interrupt Event Register	Init = 0xFFFCCF07

Bit	R/W	Description
31:18	R	**Sensitivity type of interrupt for interrupt request (IRQ50-IRQ63)** 1: High-level sensitive or rising edge triggered. 0: Low-level sensitive or falling edge triggered.
17:16	RW	**Sensitivity type of interrupt for interrupt request (IRQ48-IRQ49)** 1: High-level sensitive or rising edge triggered. 0: Low-level sensitive or falling edge triggered.

to next page

from previous page

15:0	R	**Sensitivity type of interrupt for interrupt request (IRQ32-IRQ47)** 1: High-level sensitive or rising edge triggered. 0: Low-level sensitive or falling edge triggered.

Offset: D8h	**VICD8: (L)Edge-Triggered Interrupt Clear Register**	**Init = 0**
Bit	**R/W**	**Description**
31:0	W	**Clear bits in the edge detection register (IRQ0-IRQ31)** Write ’1’ clears the corresponding bit in the edge detection register, and write ’0’ has no effect. When select edge triggered interrupt sensitivity type, firstly it must clear the detection register by writing ’1’ in this register, then it can enable the interrupt in VICA0, otherwise old interrupt status will take effect again.

Offset: DCh	**VICDC: (H)Edge-Triggered Interrupt Clear Register**	**Init = 0**
Bit	**R/W**	**Description**
31:0	W	**Clear bits in the edge detection register (IRQ32-IRQ63)** Write ’1’ clears the corresponding bit in the edge detection register, and write ’0’ has no effect. When select edge triggered interrupt sensitivity type, firstly it must clear the detection register by writing ’1’ in this register, then it can enable the interrupt in VICA4, otherwise old interrupt status will take effect again.

Offset: E0h	**VICE0: (L)Edge-Triggered Interrupt Status Register**	**Init = 0**
Bit	**R/W**	**Description**
31:0	R	**Edge-triggered interrupt status (IRQ0-IRQ31)** Shows the status of the edge-triggered interrupt before masking by the **VICA0** register. Value ”1” indicates that an interrupt request is active before masking. To clear this register bits from ”1” to ”0”, write the corresponding bits in VICD8 with value ’1’.

Offset: E4h	**VICE4: (L)Edge-Triggered Interrupt Status Register**	**Init = 0**
Bit	**R/W**	**Description**
31:0	R	**Edge-triggered interrupt status (IRQ32-IRQ63)** Shows the status of the edge-triggered interrupt before masking by the **VICA4** register. Value ”1” indicates that an interrupt request is active before masking. To clear this register bits from ”1” to ”0”, write the corresponding bits in VICDC with value ’1’.

20.4 SVIC Registers : Base Address = 0x1E6C:1000

Offset: 00h	**SVIC00: IRQ Status Register**	**Init = 0**
Bit	**R/W**	**Description**
31:18		**Reserved**

to next page

from previous page

17:0	R	**IRQ status** Shows the status of the interrupt after masking by **SVIC10** register. Value ”1” indicates that the interrupt is active, and generates an interrupt to the processor.

Offset: 08h **SVIC08: Raw Interrupt Status Register** **Init = 0**

Bit	R/W	Description
31:18		**Reserved**
17:0	R	**Raw interrupt status** Shows the status of the interrupt before masking by the **SVIC10** register. Value ”1” indicates that an interrupt request is active before masking.

Offset: 10h **SVIC10: Interrupt Enable Register** **Init = 0**

Bit	R/W	Description
31:18		**Reserved**
17:0	RW	**Enable the interrupt source** When read: 1: interrupt enable. Allow interrupt request to processor 0: interrupt disable When write: 1: enables the interrupt 0: no effect To clear this register bits from ”1” to ”0”, write the corresponding bits in SVIC14 with value ’1’.

Offset: 14h **SVIC14: Interrupt Enable Clear Register** **Init = 0**

Bit	R/W	Description
31:18		**Reserved**
17:0	W	**Clear bits in the SVIC10 register** Write ’1’ clears the corresponding bit in the SVIC10 register to value ”0”, and write ’0’ has no effect.

Offset: 18h **SVIC18: Software Interrupt Register** **Init = 0**

Bit	R/W	Description
31:18		**Reserved**
17:0	RW	**Software Interrupt Generation** 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.

Offset: 1Ch **SVIC1C: Software Interrupt Clear Register** **Init = 0**

Bit	R/W	Description
31:18		**Reserved**
17:0	W	**Clear bits in the SVIC18 register** Write ’1’ clears the corresponding bit in the SVIC18 register to value ”0”, and write ’0’ has no effect.

Offset: 24h	SVIC24: Interrupt Sensitivity Register	Init = 0x0000F0FF
Bit	**R/W**	**Description**
31:16		**Reserved**
15:0	R	**Sensitivity type of interrupt for interrupt request** 1: level sensitive 0: edge trigger

Offset: 28h	SVIC28: Interrupt Both Edge Trigger Control Register	Init = 0x00000F00
Bit	**R/W**	**Description**
31:16		**Reserved**
15:0	R	**Both or single edge for edge-trigger interrupt request** 1: both edge 0: single edge When sensitivity type is level sensitive, this register has no effect.

Offset: 2Ch	SVIC2C: Interrupt Event Register	Init = 0x0000F0FF
Bit	**R/W**	**Description**
31:16		**Reserved**
15:0	R	**Sensitivity type of interrupt for interrupt request** 1: High-level sensitive or rising edge triggered. 0: Low-level sensitive or falling edge triggered.

Offset: 38h	SVIC38: Edge-Triggered Interrupt Clear Register	Init = 0
Bit	**R/W**	**Description**
31:16		**Reserved**
15:0	W	**Clear bits in the edge detection register** Write '1' clears the corresponding bit in the edge detection register, and write '0' has no effect. When select edge triggered interrupt sensitivity type, firstly it must clear the detection register by writing '1' in this register, then it can enable the interrupt in SVIC10, otherwise old interrupt status will take effect again.

Offset: 3Ch	SVIC3C: Edge-Triggered Interrupt Status Register	Init = 0
Bit	**R/W**	**Description**
31:16		**Reserved**
15:0	R	**Edge-triggered interrupt detection register** Write '1' to SVIC38 clears the corresponding bit in the edge detection register, and write '0' has no effect.

20.5 CVIC Registers : Base Address = 0x1E6C:2000

Offset: 00h	CVIC00: IRQ Status Register	Init = 0
Bit	**R/W**	**Description**
31:0	R	**IRQ status** Shows the status of the interrupt after masking by **CVIC10** register. Value "1" indicates that the interrupt is active, and generates an interrupt to the processor.

Offset: 08h	CVIC08: Raw Interrupt Status Register	Init = 0

Bit	R/W	Description
31:0	R	**Raw interrupt status** Shows the status of the interrupt before masking by the **CVIC10** register. Value ”1” indicates that an interrupt request is active before masking.

Offset: 10h	CVIC10: Interrupt Enable Register	Init = 0

Bit	R/W	Description
31:0	RW	**Enable the interrupt source** When read: 1: interrupt enable. Allow interrupt request to processor 0: interrupt disable When write: 1: enables the interrupt 0: no effect To clear this register bits from ”1” to ”0”, write the corresponding bits in CVIC14 with value ’1’.

Offset: 14h	CVIC14: Interrupt Enable Clear Register	Init = 0

Bit	R/W	Description
31:0	W	**Clear bits in the CVIC10 register** Write ’1’ clears the corresponding bit in the CVIC10 register to value ”0”, and write ’0’ has no effect.

Offset: 18h	CVIC18: Software Interrupt Register	Init = 0

Bit	R/W	Description
31:0	RW	**Software Interrupt Generation** 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.

Offset: 1Ch	CVIC1C: Software Interrupt Clear Register	Init = 0

Bit	R/W	Description
31:0	W	**Clear bits in the CVIC18 register** Write ’1’ clears the corresponding bit in the CVIC18 register to value ”0”, and write ’0’ has no effect.

Offset: 24h	CVIC24: Interrupt Sensitivity Register	Init = 0x7FFF00FF

Bit	R/W	Description
31		**Reserved**
30:0	R	**Sensitivity type of interrupt for interrupt request** 1: level sensitive 0: edge trigger

Offset: 28h	CVIC28: Interrupt Both Edge Trigger Control Register	Init = 0x0000FF00
Bit	**R/W**	**Description**
31		**Reserved**
30:0	R	**Both or single edge for edge-trigger interrupt request** 1: both edge 0: single edge When sensitivity type is level sensitive, this register has no effect.

Offset: 2Ch	CVIC2C: Interrupt Event Register	Init = 0x7FFF00FF
Bit	**R/W**	**Description**
31		**Reserved**
30:0	R	**Sensitivity type of interrupt for interrupt request** 1: High-level sensitive or rising edge triggered. 0: Low-level sensitive or falling edge triggered.

Offset: 38h	CVIC38: Edge-Triggered Interrupt Clear Register	Init = 0
Bit	**R/W**	**Description**
31		**Reserved**
30:0	W	**Clear bits in the edge detection register** Write '1' clears the corresponding bit in the edge detection register, and write '0' has no effect. When select edge triggered interrupt sensitivity type, firstly it must clear the detection register by writing '1' in this register, then it can enable the interrupt in CVIC10, otherwise old interrupt status will take effect again.

Offset: 3Ch	CVIC3C: Edge-Triggered Interrupt Status Register	Init = 0
Bit	**R/W**	**Description**
31		**Reserved**
30:0	R	**Edge-triggered interrupt detection register** Write '1' to CVIC38 clears the corresponding bit in the edge detection register, and write '0' has no effect.

21 SDRAM Memory Controller

21.1 Overview

The DRAM controller of AST2500 is not backward compatible to AST2400.

21.1.1 Features

- Only support the DRAM size that has Column address (CA) = 10 bits (A0∼A9).
- Only support the DRAM BurstLen = 8 mode.
- Support I/O timing calibration function.
- Support programmable size ECC protection function, the overhead of memory size equals to 1/8 ECC protected memory size.

21.1.2 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 256 bytes data cascaded by 32 bytes 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 fully firmware transparency.
- ECC Overhead:
  - Best case: access 128 bytes of data with extra 16 bytes ECC code, 12.5% overhead
  - Worst case: access 16 bytes of data with extra 16 bytes ECC code, 100% overhead

21.1.3 Register Summary

SDRAM Controller implements 34 registers, which is listed below, to program the various SDRAM functions supported by AST2500 . Each register has its own specific offset value to derive its physical address location.

Base address of MCR = 0x1E6E_0000
Physical address = (Base address of SDRAM Controller) + Offset

MCR00: Protection Key Register
MCR04: Configuration Register
MCR08: Graphics Memory Protection Register
MCR0C: Refresh Timing Register
MCR10: AC Timing Register #1
MCR14: AC Timing Register #2
MCR18: AC Timing Register #3
MCR1C: Misc Control Register
MCR20: MR4/MR6 Mode Setting Register
MCR24: MR5 Mode Setting Register
MCR28: Mode Setting Control Register
MCR2C: MR0/MR2 Mode Setting Register
MCR30: MR1/MR3 Mode Setting Register
MCR34: Power Control Register
MCR38: Request Queued Limitation Mask Register
MCR3C: Priority Group Setting Register
MCR40: Maximum Grant Length Register #1
MCR44: Maximum Grant Length Register #2
MCR48: Maximum Grant Length Register #3

MCR4C: Maximum Grant Length Register #4
MCR50: Interrupt Control/Status Register
MCR54: ECC/Cache Address Range Control Register
MCR58: Address of First Un-Recoverable ECC Error Address
MCR5C: Address of Last Recoverable ECC Error Address
MCR60: DDR PHY Control/Status Register #1
MCR64: DDR PHY Control/Status Register #2
MCR68: DDR PHY Control/Status Register #3
MCR6C: DDR PHY Control/Status Register #4
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: DDR PHY Debug Interface Control
MCR84: DDR PHY Debug Interface Data
MCR88: Reserved Register
MCR8C: Reserved Register
MCR100: AST2000 Backward Compatible SCU Password
MCR120: AST2000 Backward Compatible SCU MPLL Parameter
MCR170: AST2000 Backward Compatible SCU Hardware Strapping Value

Chang the setting of Memory Controller registers often results in significant impact on SOC operations. Therefore, all these registers have to be well protected.

21.2 Fixed Priority DRAM Request

Priority	Request ID	Request Source
1	REQ0	VGA hardware cursor read
2	REQ1	VGA text mode CG font read
3	REQ2	VGA text mode ASCII code read
4	REQ3	VGA CRT read
5	REG4	SOC Display Controller cursor read
6	REG5	SOC Display Controller OSD read
7	REG6	SOC Display Controller CRT read
8	REQ7	Video high priority write
9	REQ8	USB2.0 Hub/EHCI1 DMA read/write
10	REQ9	~~USB2.0 Device~~/EHCI2 DMA read/write
11	REQ10	CPU data read/write
12	REQ11	AHB2 bus read/write
13	REQ12	AHB bus read/write
14	REQ13	MAC0 DMA read/write
15	REQ14	MAC1 DMA read/write
16	REQ15	Reserved
17	REQ16	PCI-E bus read/write
18	REQ17	X-DMA Meassage read/write
19	REQ18	Encryption engine read/write
20	REQ19	Reserved
21	REQ20	Reserved
22	REQ21	Video flag read/write
23	REQ22	Reserved
24	REQ23	Reserved
25	REQ24	Reserved
26	REQ25	Reserved
27	REQ26	Reserved
28	REQ27	Reserved
29	REQ28	Video low priority write
30	REQ29	2D engine data read/write
31	REQ30	Memory Integrity Check engine read

21.3 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 the registers (MCR04 ~ MCR7C) of SDRAM controller from unpredictable updates, especially when ARM CPU is out of control; even under such a circumstance, properly protecting SDRAM registers can make sure that Graphic Display Controller is still able to successfully access SDRAM for displaying graphics correctly. The password of the protection key is **0xFC600309**. Reading back SDRAM registers is irrelevant with this register. **Unlock SDRAM registers**: Write 0xFC600309 to this register **Lock SDRAM registers**: Write others value 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 at locked mode. When this register is locked, the read back value of this register is 0x00000000. When this register is unlocked, the read back value of this register is 0x00000001.

Offset: 04h	MCR04: Configuration Register	Init = 0x1000000C

Bit	R/W	Description
31:28	R	**DRAM Controller Hardware Version** 0: earlier chip revision 1: AST2500 others: Reserved
27:20	RW	**DRAM Controller Software Version** 0: earlier chip revision others: Software defined
19	R	**Cache initial done status** 0: Initial sequence not active or in busy state 1: Initial done
18:14		**Reserved (0)**
13	RW	**DDR4 DRAM configuration** 0: Single X16 DRAM 1: Dual X8 DRAM This option only can be applied to 512MbitX8 DRAM part.
12	RW	**Enable Cache initial** 0: Normal cache operation 1: Enable cache initial sequence Cache initial sequence can only be enabled under cache disable mode.
11	RW	**Enable Cache with address range control** 0: Cache operates for all address range 1: Cache operates within address range set at MCR54
10	RW	**Enable Cache function** 0: Disable cache 1: Enable cache When ECC is enabled, cache also must be enabled.
9	RW	**Data Scramble pattern selection** 0: Pattern 1 1: Pattern 2
8	RW	**Enable Data Scramble function** 0: Disable scramble function 1: Enable data scramble function, data will be scrambled to reduce maximum SSO current.

to next page

from previous page

7	RW	**Enable ECC function** 0: Disable ECC function 1: Enable ECC function, need extra 1/8 memory size cost for ECC data.
6	R	**Reserved for VGA driver backward compatible** 1: 16 bits
5	RW	**DRAM Rank configuration** Select DRAM Rank number for stacked DRAM 0: One rank 1: Two rank, MA15 function as CS1 The 2 rank mode only can be applied on stacked 8Gbit DRAM part.
4	RW	**Select DRAM type** 0: DDR3 1: DDR4
3:2	R	**Select graphics memory aperture 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 hardware strap bit[3:2]
1:0	RW	**Select the total memory capacity for oversized write protection** 00: Select 128M bytes or smaller as the total data memory capacity 01: Select 256M bytes as the total data memory capacity 10: Select 512M bytes as the total data memory capacity 11: Select 1024M bytes as the total data memory capacity 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 [5:4]. When being programmed to be 0, the request is not changed.

Offset: 0Ch	MCR0C: Refresh Timing Register	Init = 0

Bit	R/W	Description
31	RW	**ZQCS exit delay** 0: 64 nCK 1: 128 nCK
30:16	RW	**Period of DDR3 ZQCS cycle** This value define the period of executing DDR3 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** AST2500 uses 12MHz clock source as the reference clock to generate high-priority SDRAM refresh cycle requests according to the following equation. SDRAM Refresh Frequency = 12MHz / (period of refresh cycles)
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 fully utilize the available memory bandwidth. Whenever SDRAM refresh counter is counting over 1/4 of the period of SDRAM refresh cycles (MCR0C [15:8]), SDRAM Controller will automatically issue low-priority SDRAM refresh cycle requests. The maximum number of refresh cycles is 8 times per issued request. Low-Priority refresh cycles will stop whenever other requests are 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.
3:0	RW	**Refresh cycles per refresh period** 0000: Disable refresh cycles 0001: 1 refresh cycle per refresh period 0010: 2 refresh cycles per refresh period .... 1xxx: 8 refresh cycles per refresh period DRAM Read data will be valid only if refresh is enabled, else the read back data will be random value.
Note : When working at higher ambient temperature (80^oC), the refresh period must be faster.		

Offset: 10h	MCR10: AC Timing Register #1	Init = 0

Bit	R/W	Description
31:28	RW	**t-RP timing setting bit[3:0]** Timing = 2n + 3 (CK), n is the register value.

to next page

from previous page

27:24	RW	**t-RRD timing setting bit[3:0] (active-to-active)** Timing = 2n + 1 (CK), n is the register value.
23:20	RW	**t-RCD timing setting bit[3:0] (active-to-read/write)** Timing = 2n + 3 (CK), n is the register value.
19:16	RW	**t-APD timing setting bit[3:0]** Timing = 2n + 1 (CK), n is the register value. This timing setting determines the number of delay cycles from ACT/PRE command to read/write command between different banks.
15:12	RW	**t-RTP timing setting bit[3:0] (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.
11:8	RW	**t-WTP Timing Setting bit[3:0] (write to pre-charge)** Timing = 2n + 0 (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.
7:4	RW	**t-RTW Timing Setting bit[3:0] (Read to Write)** Timing = 2n + 3 (CK), n is the register value. This timing setting determines the number of delay cycles from read command to write command.
3:0	RW	**t-WTR Timing Setting bit[3:0] (Write to Read)** 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
31:28	RW	**t-FAW four bank active window bit[3:0]** Timing = 2n + 1 (CK), n is the register value.
27:24	RW	**t-XSNR timing setting** Timing = 32n + 32 (CK), n is the register value.
23:20	RW	**DRAM Write Latency (CWL) timing setting** 0000: not supported Timing = CWL - 2
19:16	RW	**DRAM Read Latency (CL) timing setting** 0000: not supported DDR3: Timing = CL - 2 DDR4: Timing = CL - 1
15:12	RW	**t-RAS timing setting bit[3:0] (active to minimum precharge timing)** Timing = 2n + 1 (CK), n is the register value.
11:8	RW	**t-MRD and t-MOD Mode Set Interval** Select the maximum one. Timing = 4n + 2 (CK), n is the register value.
7:0	RW	**t-RFC 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
16	RW	**Write data delay phase selection** In normal operation, this bit indicates the CWL is even or odd. 0: even 1: odd
15:14	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.
13:12	RW	**t-CCD_L timing setting** Timing = 2n + 4 (CK) This timing is required for DDR4 SDRAM only.
11:10	RW	**Different Rank Read command gap delay** Timing = 2n + 2 (CK) The delay is required for different rank DRAM read, to prevent the overlaping of data and preamble/postamble.
9	RW	**t-FAW timing setting bit4**
8	RW	**t-RAS timing setting bit4**
7	RW	**t-RP timing setting bit4**
6	RW	**t-RRD timing setting bit4**
5	RW	**t-RCD timing setting bit4**
4	RW	**t-APD timing setting bit4**
3	RW	**t-RTP timing setting bit4**
2	RW	**t-WTP Timing Setting bit4**
1	RW	**t-RTW Timing Setting bit4**
0	RW	**t-WTR Timing Setting bit4**

Offset: 1Ch	MCR1C: Misc Control Register	Init = 0

Bit	R/W	Description
31:5		**Reserved (0)**
4:0	RW	**Request queued number control threshold value** This register is designed to limit the request count in Request Queue. Used in combination with MCR38.

Offset: 20h	MCR20: MR4/MR6 Mode Setting Register	Init = X

Bit	R/W	Description
31:29		**Reserved (0)**
28:16	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**:

to next page

from previous page

		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)
15:13		**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

to next page

from previous page

		Bit [0] (all '0')

Offset: 24h **MCR24: MR5 Mode Setting Register** **Init = X**

Bit	R/W	Description
31:16		**Reserved (0)**
15:13		**Reserved (0)**
12: 0	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 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

Offset: 28h **MCR28: Mode Setting Control Register** **Init = 0**

Bit	R/W	Description
31:6		**Reserved (0)**

to next page

from previous page

5	RW	**Rank select** 0: Select rank 0 for mode setting 1: Select rank 1 for mode setting
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: 2Ch **MCR2C: MR0/MR2 Mode Setting Register** **Init = X**

Bit	R/W	Description
31:29		**Reserved (0)**
28:16	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)

to next page

from previous page

		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 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')
15:13		**Reserved (0)**
12: 0	RW	**Mode Register 0 Setting (MR0)** The definition of this register depends on the populated SDRAM type.

to next page

from previous page

		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) 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)

to next page

from previous page

		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) Bit [1:0] (Burst Length) 00: Burst 8 01: BC4 or BL8 (on the fly) 10: BC4 (Fixed) 11: Reserved

Offset: 30h **MCR30: MR1/MR3 Mode Setting Register** **Init = X**

Bit	R/W	Description
31:29		**Reserved (0)**
28:16	RW	**Mode Register 1 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

to next page

from previous page

		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
15:13		**Reserved (0)**
12: 0	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

to next page

from previous page

		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 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

Offset: 34h **MCR34: Power Control Register** **Init = 0**

Bit	R/W	Description
31	R	**Current CKE pin output value** **(for debugging purpose only)**
30:28	R	**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
27:18		**Reserved (0)**

to next page

from previous page

17	RW	**Disable memory request input** 0: Enable, normal operation 1: Disable, when DDR init cycle
16	RW	**Reserved**
15	RW	**Enable SDRAM Read Gap Control** 0: no gap control 1: enable gap control for PHY read output
14:12	RW	**t-CKE, CKE minimum high/low pulse width control** MIN pulse width = 2n+2 (CK), n is the register value.
11	RW	**Enable extend SDRAM ODT cycle** 0: No extend 1: Extended ODT cycle For DDR3, this bit must set to 1 when tWL setting bit[0] is 0
10	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.)
9	RW	**Reserved**
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.
6:4	RW	**CKE exit delay from power down mode to any command** Timing = 2n + 4 (CK), n is the register value.
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
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: 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 request number over the threshold value at MCR1C [4:0]). 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: 3Ch	MCR3C: Priority Group Setting Register	Init = 0

Bit	R/W	Description
n	RW	**Priority setting for REQ(n) and REQ(n+1)** 0: Priority of REQ(n) > Priority of REQ(n+1) 1: Priority of REQ(n) = Priority of REQ(n+1)
...	RW	
1	RW	**Priority setting for REQ(0) and REQ(1)** 0: Priority of REQ(1) > Priority of REQ(2) 1: Priority of REQ(1) = Priority of REQ(2)
0	RW	**Priority setting for REQ(0) and REQ(1)** 0: Priority of REQ(0) > Priority of REQ(1) 1: Priority of REQ(0) = Priority of REQ(1)

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**
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**

to next page

from previous page

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**

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		**Reserved (0)**
29	R	**Read/write of the last oversize access** 0: Read 1: Write

to next page

from previous page

28:24	R	**Request ID of the last oversize access**
23:16	R	**ECC recoverable error counter** This counter with saturation logic can record up to 255 ECC recoverable error events.
15:12	R	**ECC un-recoverable error counter** This counter with saturation logic can record up to 15 ECC un-recoverable error events.
11:7		**Reserved (0)**
6	R	**Interrupt flag of over sized write error events** 0: No interrupt pending 1: Interrupt pending (whenever a over sized write error event happens) This bit will be cleared by MCR50 [6].
5	R	**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 [6].
4	R	**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 [6].
3		**Reserved (0)**
2	RW	**Enable over sized access 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/Cache Address Range Control Register** **Init = 0**

Bit	R/W	Description
31:30		**Reserved (0)**
29:20	RW	**Maximum ECC/Cache work address range** ECC/Cache work area = address $\leq$ max_range. The ECC/Cache work area must be set in the unit of 1M Bytes. When ECC is enabled, firmware must reserve extra 1/8 address space for ECC spare data storage. If Cache range conrtol is enabled, this register must be set to limit the cache operating area.
19:0		**Reserved (0)**

Offset: 58h **MCR58: Address of First Un-Recoverable ECC Error Address** **Init = 0**

Bit	R/W	Description
31:30		**Reserved (0)**
29:4	R	**Address of first un-recoverable ECC error** The address will be recorded again when the error accumulation counter be reset.
3:0		**Reserved (0)**

Offset: 5Ch	MCR5C: Address of Last Recoverable ECC Error Address	Init = 0

Bit	R/W	Description
31:30		**Reserved (0)**
29:4	R	**Address of last recoverable ECC error** The address will be recorded again when the error accumulation counter be reset.
3:0		**Reserved (0)**

Offset: 60h	MCR60: DDR PHY Control/Status Register #1	Init = 0

Bit	R/W	Description
31:15	R	**Training Interface Read Leveling Response** For Gate Train Response: bit[0] = response of DQ[7:0] bit[8] = response of DQ[15:8] For Read Data Eye Train Response: bit[7:0] = response of DQ[7:0] bit[15:8] = response of DQ[15:8]
15:14		**Reserved**
13:12	R	**Training Interface Write Leveling Support** 00: Write leveling is not support by the PHY. Others: Reserved
11:10	R	**Training Interface Data Eye Training Support** 01: MC evaluation mode is support by PHY. Others: Reserved
9:8	R	**Training Interface Gate Training Support** 01: MC evaluation mode is support by PHY. Others: Reserved
2	RW	**DDR PHY Reset Control** 0: Enable reset 1: Disable Reset
1	RW	**DDR PHY Auto-Update Control** 0: Disable 1: Enable auto-upadte
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
31:8		**Reserved**
7:6	RW	**Training Interface Read Leveling Read DQS Edge** 0: Positive edge 1: Negative edge
5:4	RW	**Training Interface Read Leveling Load**
3:2	RW	**Training Interface PHY Data Eye Training Logic Enable**
1:0	RW	**Training Interface Gate Training Logic Enable**

Offset: 68h	MCR68: DDR PHY Control/Staus Register #3	Init = 0

Bit	R/W	Description
31:0	RW	**Training Interface Gate Training Delay Value**

Offset: 6Ch	MCR6C: DDR PHY Control/Staus Register #4	Init = 0

Bit	R/W	Description
31:16	RW	**Training Interface Data Eye Training Delay Value for DQS#**
15:0	RW	**Training Interface Data Eye Training Delay Value for DQS**

Offset: 70h	MCR70: ECC and Testing Control/Status Register	Init = 0

Bit	R/W	Description
31:16	R	**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	R	**Testing result flag** 0: Pass 1: Fail This flag will be cleared whenever disabling SDRAM tests. AST2500 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	R	**Testing finish flag** 0: Busy 1: Finish This flag will be cleared whenever disabling SDRAM tests.
11		**Reserved (0)**
10	RW	**Enable auto scrubbing for ECC recoverable errors** 0: No scrub memory 1: Auto scrub memory after recoverable ECC data error occurred.
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. ~~1xx: Enable bypass I/O buffer loopback mode.~~

to next page

from previous page

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.
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 = 0**

Bit	R/W	Description
31:30		**Reserved (0)**
29:24	RW	**Testing start address base** This value defines the testing base address segment. It is defined at 16MB boundary.
23:4	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 16MB.
3:0		**Reserved (0)**

Offset: 78h **MCR78: Testing Fail DQ Bit Register** **Init = 0**

Bit	R/W	Description
31:0	R	**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 = 0

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: DDR PHY Debug Interface Control	Init = 0

Bit	R/W	Description
31:1		**Reserved (0)**
0	RW	**Command mode** 0: Read. 1: Write.

Offset: 84h	MCR84: DDR PHY Debug Interface Data	Init = 0

Bit	R/W	Description
31:24		**Reserved (0)**
23:16	R	**Read data**
15:8	RW	**Write data**
7:0	RW	**Command address**

Offset: 88h	MCR88: Reserved Register	Init = 0

Bit	R/W	Description
31:0	RW	**Reserved**

Offset: 8Ch	MCR8C: Reserved Register	Init = 0
Bit	**R/W**	**Description**
31:0	RW	**Reserved**

Offset: 100h	MCR100: AST2000 Backward Compatible SCU Password	Init = 0x000000A8
Bit	**R/W**	**Description**
31:0	R	**0x000000A8**

Offset: 120h	MCR120: AST2000 Backward Compatible SCU MPLL Parameter	Init = 0
Bit	**R/W**	**Description**
31:16		**Reserved (0)**
15:14	RW	**Post Divider**
13:5	RW	**Numerator**
4:0	RW	**Denumerator**

Offset: 170h	MCR170: AST2000 Backward Compatible SCU Hardware Strapping Value	Init = 0
Bit	**R/W**	**Description**
31:0	R	**All '0'.**

21.4 Address Arrangement

21.4.1 Address Translation

The maximum supported addressing space of AST2500 is 1024 MB, thus 30 bits address internally. The SDRAM address is coordinated as Row address(Page), Bank address and Column address. AST2500 only support the DRAM type with CA bits = 10 (A0∼A9).

There is write protection for the out-of-range address accessing. But had no read protection or masking about the out-of-range address accessing. The maximum address outputed to the SDRAM would be 1024 MB or constrainted by the size setting in MCR04. When read to an address over the SDRAM size, the MSB will be ignored by the SDRAM, and targeting to a space with the same low bits address. When write to an address over the SDRAM size, the operation will be ignored.

21.5 Self Refresh Command Sequence

21.5.1 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

21.5.2 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

22 USB1.1 HID Controller

22.1 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)

22.2 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 an external 24MHz clock source
- 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

22.3 Procedure to enable USB1.1 Device port

1. Set SCU94 bit[14:13] = ”00”, select USB2.0 port 2 mode as HID device
2. Set SCU04 bit[3] = 1, enable controller reset

3. Set SCU0C bit[7] = 0, enable PHY clock
4. wait 10 ms for PLL locking
5. Set SCU04 bit[3] = 0, disable controller reset
6. Load driver

22.4 Registers : Base Address = 0x1E6E:1000

Offset: 00h	USBL00: Function Control and Status Register	Init = 0
Bit	**R/W**	**Description**
31:11		**Reserved (0)**
10	R	**Loop back test result** 0: Fail in the last loop back test 1: Pass in the last loop back test This bit will be cleared whenever Test Mode is disabled. (USB00[8:6] = ”000”)
9	R	**Loop back test status** 0: Loop back test is on going or has not yet fired 1: Lop back test has been done This bit is valid only when Test Mode is enabled. This bit will be cleared whenever Test Mode is disabled. (USB00[8:6] = ”000”)
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 : Enable Test Loop Back
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 AST2500 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		**Reserved (0)**
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		**Reserved (0)**
1	W	**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	W	**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	R	**USB Suspend State**
30	R	**USB Bus Reset State**
29	R	**USB Bus Line State FullSpeedMode = DN, LowSpeedMode = DP**
28	R	**USB Bus Line State FullSpeedMode = DP, LowSpeedMode = DN**
27		**Reserved (0)**
26:16	R	**USB Last Frame Number**
15:11		**Reserved (0)**
10:4	R	**USB Last Transaction Device Address**
3:0	R	**USB Last Transaction Endpoint Number**

Offset: 10h	USBL10: Interrupt Control Register	Init = 0

Bit	R/W	Description
31:9		**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.bit[9].

to next page

from previous page

7	RW	**Enable EP2 In Data packet ACK Interrupt**
6	RW	**Enable EP1 In Data packet ACK Interrupt**
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		**Reserved (0)**
9	R	**EP0 In/Out Transaction Cycle Status** 0 : EP0 is under IN cycle 1 : EP0 is under OUT cycle
8	RW	**EP0 In/Out NAK Response Occurs (WC)**
7	RW	**EP2 In Data packet ACK returned (WC)**
6	RW	**EP1 In Data packet ACK returned (WC)**
5	RW	**EP0 In Data packet ACK/STALL returned (WC)**
4	RW	**EP0 Out Data packet ACK returned (WC)**
3	RW	**EP0 Setup Data Arrives (WC)**
2	RW	**USB Suspend Resume Occurs (WC)**
1	RW	**USB Suspend Entry Occurs (WC)**
0	RW	**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		**Reserved (0)**
11:8	R	**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.
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		**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		**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		**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		**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	R	**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.

22.5 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.bit3
   (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.

23 System Control Unit (SCU)

23.1 Overview

System Control Unit (SCU) implements chip level control registers, which is listed below, to control the various functions supported by AST2500 . 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

SCU00: Protection Key Register
SCU04: System Reset Control Register
SCU08: Clock Selection Register
SCU0C: Clock Stop Control Register
SCU10: Frequency Counter Control Register (non-key protected)
SCU14: Frequency counter comparison range (non-key protected)
SCU18: Interrupt Control and Status Register (non-key protected)
SCU1C: D2-PLL Parameter Register
SCU20: M-PLL Parameter Register
SCU24: H-PLL Parameter Register
SCU28: D-PLL Parameter Register
SCU2C: Misc. Control Register
SCU30: PCI Configuration Setting Register #1
SCU34: PCI Configuration Setting Register #2
SCU38: PCI Configuration Setting Register #3
SCU3C: System Reset Control/Status Register
SCU40: ASPEED Defined for VGA Function Handshake
SCU44: ASPEED Defined for VGA Function Handshake
SCU48: MAC Interface Clock Delay Setting
SCU4C: Misc. 2 Control Register
SCU50 ~ SCU6C: VGA Scratch Register
SCU70: Hardware Strap Register
SCU74: Random Number Generator Control
SCU78: Random Number Generator Data Output
SCU7C: Silicon Revision ID Register
SCU80: Multi-function Pin Control #1
SCU84: Multi-function Pin Control #2
SCU88: Multi-function Pin Control #3
SCU8C: Multi-function Pin Control #4
SCU90: Multi-function Pin Control #5
SCU94: Multi-function Pin Control #6
SCU9C: EXTRST# Reset Selection
SCUA0: Multi-function Pin Control #7
SCUA4: Multi-function Pin Control #8
SCUA8: Multi-function Pin Control #9
SCUAC: Multi-function Pin Control #10
SCUB8: MAC Interface Clock Delay 100M Setting
SCUBC: MAC Interface Clock Delay 10M Setting
SCUC0: Power Saving Wakeup Enable Register
SCUC4: Power Saving Wakeup Control Register
SCUD4: System Reset Control Register Set 2
SCUD8: Clock Selection Register Set 2
SCUDC: Clock Stop Control Register Set 2
SCUE0: SCU Free Run Counter Read Back
SCUE4: SCU Free Run Counter Extended Read Back
SCUE8: Clock Duty Measurement Control

SCUEC: Clock Duty Measurement Result

Below registers are not key protected

SCU100: Coprocessor (CPU2) Control Register
SCU104: CPU2 Base Address for segment 0x00:0000-0x0F:FFFF
SCU108: CPU2 Base Address for segment 0x10:0000-0x1F:FFFF
SCU10C: CPU2 Base Address for segment 0x20:0000-0x2F:FFFF
SCU110: CPU2 Base Address for segment 0x30:0000-0x3F:FFFF
SCU114: CPU2 Base Address for segment 0x40:0000-0x4F:FFFF
SCU118: CPU2 Base Address for segment 0x50:0000-0x5F:FFFF
SCU11C: CPU2 Base Address for segment 0x60:0000-0x6F:FFFF
SCU120: CPU2 Base Address for segment 0x70:0000-0x7F:FFFF
SCU124: CPU2 Base Address for segment 0x80:0000-0xFF:FFFF
SCU128: CPU2 Cache Function Control
SCU130: D-PLL Extended Parameter Register
SCU134: D-PLL Extended Parameter Register
SCU138: D-PLL Extended Parameter Register
SCU13C: D2-PLL Extended Parameter Register
SCU140: D2-PLL Extended Parameter Register
SCU144: D2-PLL Extended Parameter Register
SCU148: Extended Parameter of M/H-PLL Register
SCU150: Chip Unique ID L
SCU154: Chip Unique ID H
SCU160: Generate UART 24 MHz Reference from H-PLL when CLKIN is 25 MHz
SCU180: PCI-Express Configuration Setting Contol Register
SCU184: BMC MMIO Decode Setting Register
SCU188: First relocated controller decode area location
SCU18C: Second relocated controller decode area location
SCU190: Mailbox decode area location
SCU194: Shared SRAM area decode location
SCU198: Shared SRAM area decode location
SCU19C: BMC device class code and revision ID
SCU1A4: BMC device ID
SCU1DC: Clock Duty Selection

Changing SCU registers usually results in significant impact on SOC operations. Therefore, all these registers have to be well protected.

23.2 Registers : Base Address = 0x1E6E:2000

Offset: 00h	SCU00: Protection Key Register	Init = 0

Bit	R/W	Description
31:0	RW	**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: 04h	SCU04: System Reset Control Register	Init = 0xF7CFFEDC

Bit	R/W	Description
31:26	RW	**Reserved, must keep at value ”111111”**
25	RW	**Reset X-DMA controller** 0: No operation 1: Reset X-DMA controller (asynchronous reset) (default)
24	RW	**Reset MCTP controller** 0: No operation 1: Reset MCTP controller (asynchronous reset) (default)
23	RW	**Reset ADC controller** 0: No operation 1: Reset ADC controller (asynchronous reset) (default)
22	RW	**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.**
21:19	RW	**Reserved, must keep at value ”001”**
18	RW	**Reset MIC controller** 0: No operation 1: Reset MIC controller (asynchronous reset) (default)
17	RW	**Reserved, must keep at value ”1”**
16	RW	**Reset SD/SDIO card controller** 0: No operation 1: Reset SD/SDIO controller (asynchronous reset) (default)
15	RW	**Reset USB1.1 Host controller** 0: No operation 1: Reset USB1.1 Host controller (asynchronous reset) (default)

to next page

from previous page

14	RW	**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 SCU90[29]. This reset bit affects both Host and Hub, and the controller not selected by SCU90[29] is always stayed at reset state.
13	RW	**Reset CRT controller** 0: No operation 1: Reset CRT controller (asynchronous reset) (default)
12	RW	**Reset MAC#2 controller** 0: No operation 1: Reset MAC#2 controller (asynchronous reset) (default)
11	RW	**Reset MAC#1 controller** 0: No operation 1: Reset MAC#1 controller (asynchronous reset) (default)
10	RW	**Reset PECI controller** 0: No operation 1: Reset PECI controller (default)
9	RW	**Reset PWM controller** 0: No operation 1: Reset PWM controller (default)
8	RW	**Disable PCI bus controller and VGA controller** 0: No operation (default) 1: Disable PCI bus controller and VGA controller
7	RW	**Reset 2D engine** 0: No operation 1: Reset 2D engine (default) This register is valid only at CRT Mode (SCU2C[7]).
6	RW	**Reset Video engine** 0: No operation 1: Reset Video engine (asynchronous reset) (default)
5	RW	**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.
4	RW	**Reset HAC engine** 0: No operation 1: Reset HAC engine (asynchronous reset) (default)
3	RW	**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 SCU94[14:13]. This reset bit affects all 3 controllers, and the controllers not selected by SCU94[14:13] are always stayed at reset state.

to next page

from previous page

2	RW	**Reset I2C/SMBus controller** 0: No operation 1: Reset SMBus/I2C controller (asynchronous reset) (default) Writing ”1” to this register will cause all the 14 sets of SMBus/I2C controllers to be **asynchronously** reset. Actually each of the 14 controllers has its own control register to **synchronously** reset itself.
1	RW	**Reset AHB bridges** 0: No operation (default) 1: Reset all the AHB related bridges This register will reset the following three bridges. AHB—to—M-Bus Bridge AHB—to—APB Bridge PCI—to—AHB Bridge Hardware will clear this bit to ’0’ after reset completed.
0	RW	**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 full chip Watchdog reset occur. 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.
SCUD4: System Reset Control Register Set 2		
8:6	RW	**Reserved, must keep value at ”111”**
5	RW	**Reset CRT controller** 0: No operation 1: Reset CRT1 controller (asynchronous reset) (default)
4:0	RW	**Reserved, must keep value at ”11111”**

Offset: 08h **SCU08: Clock Selection Register** **Init = 0xF3F40000**

Bit	R/W	Description
31	RW	**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	**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	RW	**2D Engine GCLK clock source selection** 0: GCLK is from MCLK 1: GCLK is from inverted MCLK This register is valid only at CRT Mode (SCU2C[7]).
26	RW	**2D Engine GCLK clock throttling enable** 0: Disable 1: Enable throttling This register is valid only at CRT Mode (SCU2C[7]).

to next page

from previous page

25:23	RW	**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	**LPC Host LHCLK divider selection** 000: LHCLK = H-PLL/4 001: LHCLK = H-PLL/8 010: LHCLK = H-PLL/12 011: LHCLK = H-PLL/16 100: LHCLK = H-PLL/20 101: LHCLK = H-PLL/24 110: LHCLK = H-PLL/28 111: LHCLK = H-PLL/32
19	RW	**LPC Host LHCLK clock generation/output enable control** 0: Disable, LHCLK come from external source 1: Enable, LHCLK is generated and output internally
18:16	RW	**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	**SD/SDIO clock running enable** 0: Stop clock 1: Enable clock
14:12	RW	**SD/SDIO divider selection** 000: SDCLK = H-PLL/4 001: SDCLK = H-PLL/8 010: SDCLK = H-PLL/12 011: SDCLK = H-PLL/16 100: SDCLK = H-PLL/20 101: SDCLK = H-PLL/24 110: SDCLK = H-PLL/28 111: SDCLK = H-PLL/32
11:8	RW	**Reserved, must keep value at default value**

to next page

from previous page

8	RW	**Graphics CRT port output clock selection, low bit** 00: D-PLL 01: D2-PLL 1x: 40MHz from USB2.0 port1 PHY The high bit is located at SCU2C[21].
7	RW	**ARM CPU/AHB clock static slow down enable** 0: Disable 1: Enable slow down Enable this bit will disable dynamic mode.
6:4	RW	**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	**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 D1-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	**CPU/AHB clock dynamic slow down idle timer** 0: 128 HCLK idle (default) 1: 256 HCLK idle
0	RW	**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].
SCUD8: Clock Selection Register Set 2		
31	RW	**Reserved**
30	RW	**Video input port clock delay control** 0: delay is controlled by VR008[11:9]+bit[26:24] 1: delay is controlled by bit[29:24]
29:24	RW	**Video input port clock delay selection** bit[29]: inverted output bit[28:24]: delay line stage
23:12	RW	**Reserved**
11:6	RW	**Video port output clock delay control** bit[11]: inverted output bit[10:6]: delay line stage
5:3	RW	**Reserved**

to next page

from previous page

2:0	RW	**P-Bus BCLK divider selection** 000: BCLK = H-PLL/4 001: BCLK = H-PLL/8 010: BCLK = H-PLL/12 011: BCLK = H-PLL/16 100: BCLK = H-PLL/20 101: BCLK = H-PLL/24 110: BCLK = H-PLL/28 111: BCLK = H-PLL/32

Offset: 0Ch **SCU0C: Clock Stop Control Register** **Init = 0xEFF43E8B**

Bit	R/W	Description
Note : The initial sequence for the clock and reset must follow below steps: 1. Enable engine reset at SCU04/SCUD4 2. Delay 100 us 3. Enable clock running SCU0C/SCUDC 4. Delay 10 ms 5. Disable engine reset at SCU04/SCUD4 The sequence must do once whenever the engine is started from the clock stopped state.		
31:29	RW	**Reserved, must keep at value ”111”**
28	RW	**Stop LHCLK (For LPC Master and LPC Plus Controller)** 0: Enable clock running 1: Stop clock running (default)
27	RW	**Stop SDCLK (For SD/SDIO Controller)** 0: Enable clock running 1: Stop clock running (default)
26	RW	**Stop UART4CLK (For UART4 controller)** 0: Enable clock running (default) 1: Stop clock running
25	RW	**Stop UART3CLK (For UART3 controller)** 0: Enable clock running (default) 1: Stop clock running
24	RW	**Stop RSACLK (For RSA Engine) Clock** 0: Enable clock running 1: Stop clock running (default)
23:22	RW	**Reserved, must keep at value ”11”**
21	RW	**Stop MAC#2 (For MAC#2 Controller) Clock** 0: Enable clock running 1: Stop clock running (default)
20	RW	**Stop MAC#1 (For MAC#1 Controller) Clock** 0: Enable clock running 1: Stop clock running (default)
19	RW	**Stop eSPI Clock** 0: Enable clock running 1: Stop clock running (default)
18	RW	**Reserved, must keep at value ”1”**

to next page

from previous page

17	RW	**Stop UART5CLK (For UART5 controller)** 0: Enable clock running (default) 1: Stop clock running
16	RW	**Stop UART2CLK (For UART2 controller)** 0: Enable clock running (default) 1: Stop clock running
15	RW	**Stop UART1CLK (For UART1 controller)** 0: Enable clock running (default) 1: Stop clock running
14	RW	**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 SCU90[29] 2. Enable USB2.0 global reset by setting SCU04[14] = 1 3. Enable USB2.0 clock running, wait 10 ms for clock stable 4. Disable USB2.0 global reset by setting SCU04[14] = 0 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 SCU90[29]. This clock bit controls both Host and Hub, decided by SCU90[29].
13	RW	**Stop YCLK (For HAC)** 0: Enable clock running 1: Stop clock running (default)
12:11	RW	**Reserved, must keep at value ”11”**
10	RW	**Stop D1CLK (For GFXCRT)** 0: Enable clock running 1: Stop clock running (default)
9	RW	**Stop USB1.1 Host Controller Clock** 0: Enable clock running 1: Stop clock running (default)
8	RW	**Stop LCLK (for LPC Controller)** 0: Enable clock running (default) 1: Stop clock running

to next page

from previous page

7	RW	**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 SCU94[14:13] 2. Enable USB2.0 global reset by setting SCU04[3] = 1 3. Enable USB2.0 clock running, wait 10 ms for clock stable 4. Disable USB2.0 global reset by setting SCU04[3] = 0 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 SCU94[14:13]. This bit enables the clock of controller selected by SCU94[14:13].
6	RW	**REFCLK Stop Enable** 0: Enable clock running (default) 1: Stop clock running
5	RW	**Stop DCLK (For DAC)** 0: Enable clock running (default) 1: Stop clock running
4	RW	**Stop BCLK (For PCIE/PCI Bus)** 0: Enable clock running (default) 1: Stop clock running
3	RW	**Stop VCLK (For Video Capture)** 0: Enable clock running 1: Stop clock running (default)
2	RW	**Stop MCLK (For SDRAM Controller)** 0: Enable clock running (default) 1: Stop clock running
1	RW	**Stop GCLK (For 2D Engine)** 0: Enable clock running 1: Stop clock running (default) This register is valid only at CRT Mode (SCU2C[7]).
0	RW	**Stop ECLK (For Video Engine)** 0: Enable clock running 1: Stop clock running (default)
SCUDC: Clock Stop Control Register Set 2		
10:9	RW	**Reserved, must keep at value "11"**
8:6	RW	**Reserved, must keep at value "111"**
4:3	RW	**Reserved, must keep at value "11"**
2:0	RW	**Reserved, must keep at value "111"**

Offset: 10h	**SCU10: Frequency Counter Control Register**	**Init = 0**
Bit	**R/W**	**Description**
31:30		**Reserved (0)**

to next page

from previous page

29:16	R	**Value of frequency measurement counter** Reset to 0 automatically when starts counting Algorithm of frequency measurement: Frequency = (24MHz / 512) * (Value + 1)
15:9	RW	**Delay ring stage control**
8	RW	**Enable OSC counter result output to pin** 0: disable 1: enable (for debugging purpose only)
7	R	**Clock frequency measurement compare result** 0 : Fail 1 : Pass This status flag is the result by comparing the output counter value at SCU14 with the upper and lower limit defined at SCU28, if the lower_limit <= counter <= upper_limit, then the compare result is Pass.
6	R	**Clock frequency measurement finished** 0 : Not finished 1 : Finished This status flag can be cleared by setting SCU10[1] to '0'
5:2	RW	**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 0011: Select DLY32 0100: Select D-PLL 0101: Select D2-PLL 0110: Select XPCLK 0111: Select BCLK 1000: Select MCLK 1001: Select HCLK 1010: Select GRCLK 1011: Select LCLK 1100: Select 12MHz 1101: Select VPBCLK 1110: Select VPACLK 1111: Select PCLK This register is designed to select the clock source for clock frequency measurement.
1	RW	**Oscillator Counter Enable** 0 : Reset frequency measurement counter 1 : Enable frequency measurement counter
0	RW	**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 SCU10 = 0x30
2. Wait until SCU10[29:16] = 0
3. Set SCU10[0] = 1 and SCU10[5:2] = clock for measurement
4. Delay 1ms
5. Set SCU10[1] = 1
6. Wait until SCU10[6] = 1
7. Read SCU10[29:16] and calculate the result frequency using following equation

Oscillator Counter Algorithm :
When the reference clock CLK24M count from 0 to 512, measure the OSCCLK counting value, then

OSCCLK frequency = CLK24M / 512 * (SCU10[29:16] + 1)

Offset: 14h **SCU14: Frequency counter comparison range** **Init = X**

Bit	R/W	Description
31:30		**Reserved (0)**
29:16	RW	**Upper Limit**
15:14		**Reserved (0)**
13:0	RW	**Lower Limit**

Offset: 18h **SCU18: Interrupt Control and Status Register** **Init = 0**

Bit	R/W	Description
31:22		**Reserved (0)**
21	RW	**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	**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	RW	**PCI-E 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.
18	RW	**PCI-E 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.
17	RW	**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	RW	**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.

to next page

from previous page

15:7		**Reserved (0)**
6	RW	**Enable issue MSI interrupt** 0 : NOP 1 : Enable issue interrupt to MSI→PCI-E→System
5	RW	**Enable LPC reset high-to-low interrupt** 0 : Disable interrupt 1 : Enable interrupt generation
4	RW	**Enable LPC reset low-to-high interrupt** 0 : Disable interrupt 1 : Enable interrupt generation
3	RW	**Enable PCI-E reset high-to-low interrupt** 0 : Disable interrupt 1 : Enable interrupt generation
2	RW	**Enable PCI-E reset low-to-high interrupt** 0 : Disable interrupt 1 : Enable interrupt generation
1	RW	**Enable VGA scratch register change interrupt** 0 : Disable interrupt 1 : Enable interrupt generation
0	RW	**Enable VGA cursor change interrupt** 0 : Disable interrupt 1 : Enable interrupt generation

Offset: 1Ch **SCU1C: D2-PLL Parameter Register** **Init = 0x0004E029**

Bit	R/W	Description
31:27	RW	**D2-PLL SIP**
26:22	RW	**D2-PLL SIC**
21:19	RW	**D2-PLL Output Divider (OD)**
18:13	RW	**D2-PLL Post Divider (P)**
12:8	RW	**D2-PLL Denumerator (N)**
7:0	RW	**D2-PLL Numerator (M)** The output frequency of D2-PLL PLL is based on the following equation: (Output frequency) = CLKIN(24MHz) * [(M+1) / (N+1)] / (P+1) / (OD+1) The default setting of D2-PLL is OFF.
SCU13C: D2-PLL Extended Parameter Register		
13:5	RW	**D2-PLL parameter bit[36:28]**
4:3	RW	**Select D2-PLL mode**
2	RW	**Enable D2-PLL reset** 0: Normal operation 1: Reset PLL
1	RW	**Enable D2-PLL bypass mode** 0: No operation 1: Enable D2-PLL bypass mode When enabling D2-PLL bypass mode, the output clock of D2-PLL is directly from the external CLKIN input pin.

to next page

from previous page

0	RW	**Turn off D2-PLL** 0: No operation 1: Turn Off D2-PLL When D2-PLL is turned off, it will enter power down mode. The output signal is always ”0”. D2-PLL default is OFF to reduce the power when power up.
SCU140: D2-PLL Extended Parameter Register		
27:0	RW	**D2-PLL parameter bit[27:0]**
SCU144: D2-PLL Extended Parameter Register		
18:0	RW	**D2-PLL Fraction**

Offset: 20h **SCU20: M-PLL Parameter Register** **Init = 0x93002400**

Bit	R/W	Description
31:22	RW	**M-PLL parameter bit[9:0]**
21	RW	**Enable M-PLL reset** 0: Normal operation 1: Reset PLL
20	RW	**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.
19	RW	**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”. M-PLL default is OFF and bypassed to reduce the power when power up.
18:13	RW	**M-PLL Post Divider (P)**
12:5	RW	**M-PLL Numerator (M)**
4:0	RW	**M-PLL Denumerator (N)** M-PLL is preliminarily designed to generate the running frequency of memory controller. The output frequency of M-PLL is based on the following equation: (Output frequency) = CLKIN(24MHz) * [(M+1) / (N+1)] / (P+1) The default setting of M-PLL is OFF and bypassed.
SCU148: Extended Parameter of M/H-PLL Register		
5:0	RW	**M-PLL parameter bit[15:10]**

Offset: 24h **SCU24: H-PLL Parameter Register** **Init = 0x93000400**

Bit	R/W	Description
31:22	RW	**H-PLL parameter bit[9:0]**
21	RW	**Enable H-PLL reset** 0: Normal operation 1: Reset PLL

to next page

from previous page

20	RW	**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.
19	RW	**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”.
18:13	RW	**H-PLL Post Divider (P)**
12:5	RW	**H-PLL Numerator (M)**
4:0	RW	**H-PLL Denumerator (N)** 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(24MHz) * [(M+1) / (N+1)] / (P+1) The default frequency of H-PLL is 792MHz when CLKIN = 24MHz.
SCU148: Extended Parameter of M/H-PLL Register		
21:16	RW	**H-PLL parameter bit[15:10]**

Offset: 28h **SCU28: D-PLL Parameter Register** **Init = 0x0004E029**

Bit	R/W	Description
31:27	RW	**D-PLL SIP**
26:22	RW	**D-PLL SIC**
21:19	RW	**D-PLL Output Divider (OD)**
18:13	RW	**D-PLL Post Divider (P)**
12:8	RW	**D-PLL Denumerator (N)**
7:0	RW	**D-PLL Numerator (M)** The output frequency of D-PLL PLL is based on the following equation: (Output frequency) = CLKIN(24MHz) * [(M+1) / (N+1)] / (P+1) / (OD+1) The default setting of D-PLL is OFF.
SCU130: D-PLL Extended Parameter Register		
13:5	RW	**D-PLL parameter bit[36:28]**
4:3	RW	**Select D-PLL mode**
2	RW	**Enable D-PLL reset** 0: Normal operation 1: Reset PLL
1	RW	**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 directly from the external CLKIN input pin.

to next page

from previous page

0	RW	**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 is always ”0”. D-PLL default is OFF to reduce the power when power up.
SCU134: D-PLL Extended Parameter Register		
27:0	RW	**D-PLL parameter bit[27:0]**
SCU138: D-PLL Extended Parameter Register		
18:0	RW	**D-PLL Fraction**

Offset: 2Ch **SCU2C: Misc. Control Register** **Init = 0x00000010**

Bit	R/W	Description
31	RW	**Reserved (0)**
30	RW	**Enable PCIe PEWAKE# output low** 0: inactive, tri-stated output 1: active, output low
29	RW	**Enable PCIe PEWAKE# function pin**
28	RW	**Select 24MHz BCLK** 0: divided by SCUD8[2:0] 1: 24MHz
27:26	RW	**Reserved (0)**
25	RW	**Disable DRAM address space write from P2A bridge** When disabled, P2A will mask all write command to address space: 1. 0x80000000 - 0xFFFFFFFF
24	RW	**Disable LPC Host/Plus address space write from P2A bridge** When disabled, P2A will mask all write command to address space: 1. 0x60000000 - 0x7FFFFFFF
23	RW	**Disable SOC address space write from P2A bridge** When disabled, P2A will mask all write command to address space: 1. 0x10000000 - 0x1FFFFFFF 2. 0x40000000 - 0x5FFFFFFF
22	RW	**Disable flash address space write from P2A bridge** When disabled, P2A will mask all write command to address space: 1. 0x00000000 - 0x0FFFFFFF 2. 0x20000000 - 0x3FFFFFFF
21	RW	**Graphics CRT port output clock selection, high bit** 00: D-PLL 01: D2-PLL 1x: 40MHz from USB2.0 port1 PHY The low bit is located at SCU08[8].
20	RW	**Select D-PLL parameter source** 0: from VGA. (default) 1: from SCU
19	RW	**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	**Select the DVO source for display output** 0: VGA mode (default) 1: Graphics CRT mode

to next page

from previous page

17:16	RW	**Select the DAC source for display output** 00: VGA mode (default) 01: Graphics CRT mode 10: Pass-through mode from Video input port-A 11: Pass-through mode from Video input port-B
15:14	RW	**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	**Timeout control bit for VUART** For the detail, reference VUART20 and VUART24.
12	RW	**Enable the reference clock divider (div13) for all UARTs (1 - 5)** 0: baud rate = 24MHz / (16*divisor) (default) 1: baud rate = (24MHz/13) / (16*divisor)
11	RW	**Enable inverting YCLK** 0: YCLK from MCLK 1: YCLK from inverted MCLK
10	RW	**Disable UART port debug function**
9:8	RW	**Reserved**
8	RW	~~**Disable PCI bus to AHB bus bridge**~~ Function replaced by SCU180[1].
7	RW	**Enable 2D CRT Mode function** 0: VGA mode (default) 1: CRT mode
6	RW	**Disable VGA CRT display when using Video Direct Fetch mode** 0: Enable VGA CRT display (default) 1: Disable VGA CRT display
5	RW	**Enable VGA registers access when not strap in VGA mode** 0: VGA registers access is controlled by VGA mode strap (default) 1: Force enables VGA registers access
4	RW	**Disable D2-PLL** 0: Enable D2-PLL 1: Disable D2-PLL (default) Highest Priority.
3	RW	**Disable video DAC** 0: Enable video DAC (default) 1: Disable video DAC Highest Priority.
2	RW	**Disable D-PLL** 0 : Enable D-PLL (default) 1 : Disable D-PLL Highest Priority.
1	RW	**OSC clock output pin selection (For test mode only)** 0: No OSC clock output (default) 1: OSC clock will output from GPIOQ6 (B10) pin
0	RW	**Disable LPC to SPI flash interface** 0: Enable (default) 1: Disable This bit is valid when the hardware strap bit[13:12] = ”10” or ”01”.

Offset: 30h	SCU30: PCI Configuration Setting Register #1	Init = 0x20001A03
Bit	**R/W**	**Description**
31:16	RW	**PCI Device ID** The register can support firmware code the flexibility to change PCI Device ID of AST2500 . But changing the ID is usually not recommended.
15:0	RW	**PCI Vendor ID** The register can support firmware code the flexibility to change PCI Vendor ID of AST2500 . But changing the ID is usually not recommended.

Offset: 34h	SCU34: PCI Configuration Setting Register #2	Init = 0x20001A03
Bit	**R/W**	**Description**
31:16	RW	**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	**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: 38h	SCU38: PCI Configuration Setting Register #3	Init = 0x04000041
Bit	**R/W**	**Description**
31:8	RW	**Class Code** The register can support firmware code the flexibility to change PCI Class Code ID of AST2500 . But changing the ID is usually not recommended.
7:0	RW	**PCI Revision ID** The register can support firmware code the flexibility to change PCI Revision ID of AST2500 . But changing the ID is usually not recommended.

Offset: 3Ch	SCU3C: System Reset Control/Status Register	Init = 0x00000001
Bit	**R/W**	**Description**
31:5		**Reserved (0)**
4	RW	**Watch dog #3 event flag** This register bit was cleared by external power reset pin, SRST#, and set by internal watchdog #3 timeout event signal. It can be cleared by software after software checked the bit status.
3	RW	**Watch dog #2 event flag** This register bit was cleared by external power reset pin, SRST#, and set by internal watchdog #2 timeout event signal. It can be cleared by software after software checked the bit status.
2	RW	**Watch dog #1 event flag** This register bit was cleared by external power reset pin, SRST#, and set by internal watchdog #1 timeout event signal. It can be cleared by software after software checked the bit status.
1	RW	**External reset flag** This register bit was cleared by external power reset pin, SRST#, and set by external reset signal EXTRST#. It can be cleared by software after software checked the bit status.
0	RW	**Power on reset flag** This register bit was set by external power reset pin, SRST#. It can be cleared by software after software checked the bit status.

Offset: 40h	SCU40: ASPEED Defined for VGA Function Handshake	Init = 0
Bit	**R/W**	**Description**
31:24	RW	**Reserved for ASPEED SDK firmware and SLT test program handshaking** 0x5A: Embedded Linux boot to Linux Properly others: Not defined
23:16	RW	**Reserved for Customers definition**
15:8	RW	**Reserved for ASPEED definition**
7	RW	**DRAM Initial Selection (see note 1)** 0: VBIOS Initial the DRAM 1: SOC Firmware Initial the DRAM
6	RW	**SOC Firmware Initial DRAM Status (see note 1)** 0: DRAM Initial is not ready 1: DRAM Initial is Ready
5	RW	**Reserved (AST2000 use only)**
4	RW	**KVM Virtual EDID Function Selection (see note 2)** 0: disable 1: enable
3	RW	**Reserved (AST2000 use only)**
2	RW	**Reserved (AST2000 use only)**
1	RW	**BMC Firmware Protection** 1: Forbid SOCFlash to update flash(support from SOCFlash v.1.02.01)
0	RW	**iKVM support Wide Screen resolution** 0: iKVM cant 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: 44h	SCU44: ASPEED Defined for VGA Function Handshake	Init = 0
Bit	**R/W**	**Description**
31:0	RW	**The last serviced IRQ number**

Offset: 48h	SCU48: MAC Interface Clock Delay Setting	Init = 0

Bit	R/W	Description
31	RW	**RGMII 125MHz clock source selection** 0: PAD_RGMIICK 1: Internal PLL
30	RW	**RMII2 50MHz RCLK output enable** 0: Disable 1: Enable
29	RW	**RMII1 50MHz RCLK output enable** 0: Disable 1: Enable
28:26	RW	**Reserved**
25	RW	**MAC#2 RMII transmit data at clock falling edge**
24	RW	**MAC#1 RMII transmit data at clock falling edge**
23:18	RW	**MAC#2 RMII_RCLK/RGMII_RXCLK (1G) clock input delay**
17:12	RW	**MAC#1 RMII_RCLK/RGMII_RXCLK (1G) clock input delay**
11:6	RW	**MAC#2 RGMII_TXCLK (1G) clock output delay**
5:0	RW	**MAC#1 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 (SCU04)

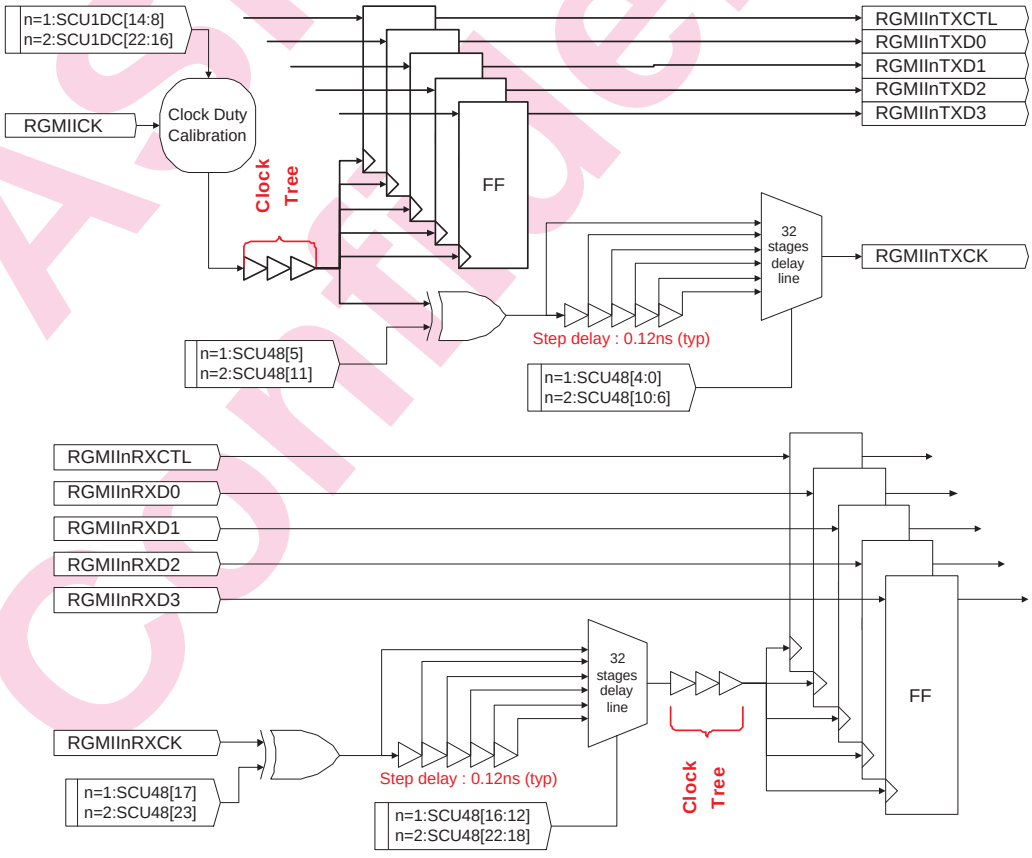


Figure 47: RGMII timing control block diagram

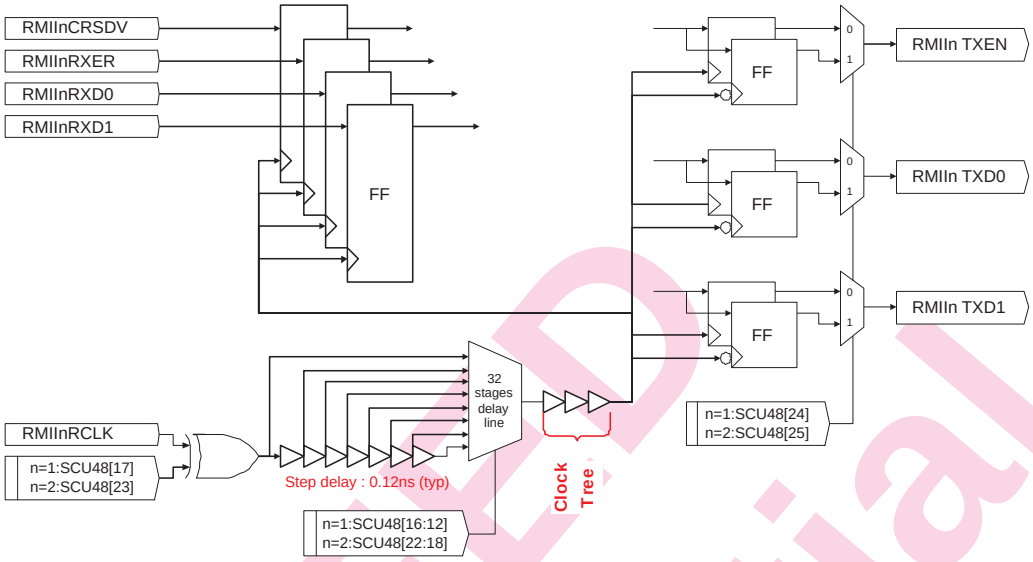


Figure 48: RMII timing control block diagram

Offset: 4Ch	SCU4C: Misc. 2 Control Register	Init = 0x03
Bit	**R/W**	**Description**
31	RW	**Reserved**
30	RW	**Select UART debug port baud rate** 0: 115200 bps 1: 921600 bps
29	RW	**Reserved (0)**
28	RW	**Select UART5 clock source**
27	RW	**Select UART4 clock source**
26	RW	**Select UART3 clock source**
25	RW	**Select UART2 clock source**
24	RW	**Select UART1 clock source** 0: 24MHz 1: 192MHz from USB2.0 port1 PHY
23	RW	**RGMII 125MHz clock divider source selection** 0: D2-PLL 1: H-PLL
22:20	RW	**RGMII 125MHz clock divider ratio** 000: div2 001: div2 010: div3 .... 111: div8
19	RW	**RMII 50MHz clock divider source selection** 0: D2-PLL 1: M-PLL

to next page

from previous page

18:16	RW	**RMII 50MHz clock divider ratio** 000: div2 001: div2 010: div3 .... 111: div8
15	RW	**PCIe mamory mapping address high** Bit2 of SCU70[3:2]
14:9	RW	**Reserved (0)**
8	RW	**MALI DTY mode**
7	RW	**MALI RC mode**
6	RW	**PCI BM enable**
5:3	RW	**DAC Mode Selection** 000: Normal mode others: Test mode
2:0	RW	**DAC Driver full swing (FS) setting** $I_{out} = (1 + FS/10) * 250/DACRSET$

Offset: 50h–6Ch **SCU50 ∼ SCU6C: VGA Scratch Register** **Init = 0**

Offset	Bit	Attr.	Description
50h	31:0	R	**VGA scratch register bit[31:0]**
54h	31:0	R	**VGA scratch register bit[63:32]**
58h	31:0	R	**VGA scratch register bit[95:64]**
5Ch	31:0	R	**VGA scratch register bit[127:96]**
60h	31:0	R	**VGA scratch register bit[159:128]**
64h	31:0	R	**VGA scratch register bit[191:160]**
68h	31:0	R	**VGA scratch register bit[223:192]**
6Ch	31:0	R	**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: 70h **SCU70: Hardware Strap Register** **Init = X**

Bit	R/W	Description
31	RW	**Enable SPI Flash Strap Auto Fetch Mode** 0: Disable 1: Enable
30	RW	**Enable GPIO Strap Mode** 0: Disable 1: Enable
29	RW	**Select UART Debug Port** 0: Select UART1 as BMC console port 1: Select UART5 as BMC console port This bit is used to select the UART port for user debugging. It is not the BMC console function. The function is similar to the PCIe-to-AHB bridge, which is working as a hardware u-boot interface, and no firmware required to operate. This interface can support firmware update capability without firmware working required.

to next page

from previous page

28	RW	**Reserved (1)**
27	RW	**Enable fast reset mode for ARM ICE debugger** 0: Long reset mode, normal operation (default) 1: Fast reset mode, for ICE debugger connection Enable fast reset mode to enable ICE debugger can stop ARM at starting address 0.
26	RW	**Enable eSPI flash mode** 0: eSPI respond with no flash attached 1: eSPI respond with flash attached This bit is valid only when eSPI is enabled.
25	RW	**Enable eSPI mode** 0: LPC mode 1: eSPI mode
24	RW	**Select DDR4 SDRAM** 0: DDR3 SDRAM 1: DDR4 SDRAM
23	RW	**Select 25 MHz reference clock input mode** 0: CLKIN is 24 MHz and USBCKI not used 1: CLKIN is 25 MHz and USBCKI = 24/48 MHz (determined by bit[18])
22	RW	**Enable GPIOE pass-through mode** 0: Disable, pass through can be enabled by SCU8C[15:12]. 1: Enable pass-through at power on. Pass-through pins set: GPIOE0 → GPIOE1 GPIOE2 → GPIOE3 GPIOE4 → GPIOE5 GPIOE6 → GPIOE7
21	RW	**Enable GPIOD pass-through mode** 0: Disable, pass through can be enabled by SCU8C[11:8]. 1: Enable pass-through at power on. Pass-through pins set: GPIOD0 → GPIOD1 GPIOD2 → GPIOD3 GPIOD4 → GPIOD5 GPIOD6 → GPIOD7
20	RW	**Disable LPC to decode SuperIO 0x2E/0x4E address** 0: Enable address decoding (default) 1: Disable address decoding
19	RW	**Enable ACPI function** 0: Disable ACPI 1: Enable ACPI
18	RW	**Select USBCKI input frequency** 0: 24 MHz (default) 1: 48 MHz
17	RW	**Enable BMC 2nd boot watchdog timer** 0: Disable 1: Enable BMC 2nd boot watchdog timer start counting at power up. (default) The watchdog timer was located at WDT2. After watchdog timeout, it will reset BMC and restart booting from the 2nd boot flash at CS1#. CS1# must be the same flash type as CS0# and could be the same firmware as CS0#.

to next page

from previous page

16	RW	**SuperIO configuration address selection** 0: Decode 0x2E 1: Decode 0x4E
15	RW	**VGA Class Code selection** 0: Select the Class Code for video device 1: Select the Class Code for VGA device (default)
14	RW	**Select dedicated LPC reset input** 0: LPC reset is shared with PCIe reset pin 1: LPC reset is located at pin number G22, shared with GPIOAC7. (default)
13:12	RW	**SPI mode selection** 00: Disable SPI interface 01: Enable SPI Master 10: Reserved, enable SPI Master and SPI Slave to AHB Bridge (debug mode) 11: Enable SPI Pass-through
11:9	RW	**AXI/AHB clock frequency ratio selection** 000: undefined 001: Select AXI:AHB = 2:1 (default) 010: Select AXI:AHB = 3:1 011: Select AXI:AHB = 4:1 100: Select AXI:AHB = 5:1 101: Select AXI:AHB = 6:1 110: Select AXI:AHB = 7:1 111: Select AXI:AHB = 8:1 HCLK freq = H-PLL / 2 / (ratio_of_AXI-AHB)
8	RW	**Reserved (0)**
7	RW	**Define MAC#2 interface** 0: RMII/NCSI 1: RGMII
6	RW	**Define MAC#1 interface** 0: RMII/NCSI 1: RGMII
5	RW	**Enable dedicated VGA BIOS ROM** 0: No VGA BISO ROM, VGA BIOS is merged in the system BIOS (default) 1: Enable dedicated VGA BIOS ROM
4	RW	**Reserved (0)**

to next page

from previous page

3:2	RW	**VGA memory size selection** 00: Select 8 MB VGA memory 01: Select 16 MB VGA memory (default) 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
1	R	**Reserved (1)**
0	RW	**Disable CPU boot** 0: Enable boot 1: Disable CPU operation, when no firmware exist
Note : The write operation to 0x1E6E2070 only can set to '1'. To clear to '0', it must write '1' to 0x1E6E207C (write 1 clear).		

Offset: 74h **SCU74: Random Number Generator Control** **Init = 0xE**

Bit	R/W	Description
31:4		**Reserved**
3:1	RW	**Random number generator mode selection** There are 8 types of random number generate methods.
0	RW	**Random number generator disable control** 0: Enable 1: Disable

Offset: 78h **SCU78: Random Number Generator Data Output** **Init = 0**

Bit	R/W	Description
31:0	R	**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: 7Ch **SCU7C: Silicon Revision ID Register** **Init = -**

Bit	R/W	Description
31:24		**Reserved (0x04)**

to next page

from previous page

23:16	R	**Hardware Revision ID** 0: Represent A0 silicon 1: Represent A1 silicon 3: Represent A2 silicon .... And so forth.
15:8	R	**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	R	**Reserved for backward compatible**

The following table shows a list of silicon revision ID.

AST1100-A0	0x00000200
AST1100-A1	0x00000201
AST1100-A2	0x00000202
AST1100-A3	0x00000202
AST2050-A0	0x00000200
AST2050-A1	0x00000201
AST2050-A2	0x00000202
AST2050-A3	0x00000202
AST2100-A0	0x00000300
AST2100-A1	0x00000301
AST2100-A2	0x00000302
AST2100-A3	0x00000302
AST2150-A0	0x00000202
AST2150-A1	0x00000202
AST2200-A0	0x00000102
AST2200-A1	0x00000102
AST2300-A0	0x01000003
AST2300-A1	0x01010303
AST1300-A1	0x01010003
AST1050-A1	0x01010203
AST2400-A0	0x02000303
AST2400-A1	0x02010303
AST1400-A1	0x02010103
AST1250-A1	0x02010303
AST2500-A0	0x04000303
AST2510-A0	0x04000103
AST2520-A0	0x04000203
AST2530-A0	0x04000403
AST2500-A1	0x04010303
AST2510-A1	0x04010103
AST2520-A1	0x04010203
AST2530-A1	0x04010403
AST2500-A2	0x04030303
AST2510-A2	0x04030103
AST2520-A2	0x04030203
AST2530-A2	0x04030403

Offset: 80h **SCU80: Multi-function Pin Control #1** **Init = 0**

Bit	R/W	Description
31	RW	**Enable UART4 RXD4 function pin**
30	RW	**Enable UART4 TXD4 function pin**
29	RW	**Enable UART4 NRTS4 function pin**
28	RW	**Enable UART4 NDTR4 function pin**
27	RW	**Enable UART4 NRI4 function pin**

to next page

from previous page

26	RW	**Enable UART4 NDSR4 function pin**
25	RW	**Enable UART4 NDCD4 function pin**
24	RW	**Enable UART4 NCTS4 function pin**
23	RW	**Enable UART3 RXD3 function pin**
22	RW	**Enable UART3 TXD3 function pin**
21	RW	**Enable UART3 NRTS3 function pin**
20	RW	**Enable UART3 NDTR3 function pin**
19	RW	**Enable UART3 NRI3 function pin**
18	RW	**Enable UART3 NDSR3 function pin**
17	RW	**Enable UART3 NDCD3 function pin**
16	RW	**Enable UART3 NCTS3 function pin**
15	RW	**Reserved**
14	RW	**Enable LPC LPCPME# function pin**
13	RW	**Enable LPC LPCPD#/LPCSMI# function pin**
12:8	RW	**Reserved**
7	RW	**Enable Timer8 Pulse output function pin**
6	RW	**Enable Timer7 Pulse output function pin**
5	RW	**Enable Timer6 Pulse output function pin**
4	RW	**Enable Timer5 Pulse output function pin**
3	RW	**Enable Timer4 Pulse output function pin**
2	RW	**Enable Timer3 Pulse output function pin**
1	RW	**Enable MAC#2 PHY Link interrupt MAC2LINK input function pin**
0	RW	**Enable MAC#1 PHY Link interrupt MAC1LINK input function pin**

Offset: 84h **SCU84: Multi-function Pin Control #2** **Init = 0x0000F000**

Bit	R/W	Description
31	RW	**Enable UART2 RXD1 or Video VPIB9 function pin** SCU90[5]=1 select Video pin
30	RW	**Enable UART2 TXD1 or Video VPIB8 function pin** SCU90[5]=1 select Video pin
29	RW	**Enable UART2 NRTS1 or Video VPIB7 function pin** SCU90[5]=1 select Video pin
28	RW	**Enable UART2 NDTR1 or Video VPIB6 function pin** SCU90[5]=1 select Video pin
27	RW	**Enable UART2 NRI1 or Video VPIB5 function pin** SCU90[5]=1 select Video pin
26	RW	**Enable UART2 NDSR1 or Video VPIB4 function pin** SCU90[5]=1 select Video pin
25	RW	**Enable UART2 NDCD1 or Video VPIB3 function pin** SCU90[5]=1 select Video pin
24	RW	**Enable UART2 NCTS1 or Video VPIB2 function pin** SCU90[5]=1 select Video pin
23	RW	**Enable UART1 RXD1 function pin** SCU90[5:4]=0x3 select Video pin

to next page

from previous page

22	RW	**Enable UART1 TXD1 function pin** SCU90[5:4]=0x3 select Video pin
21	RW	**Enable UART1 NRTS1 or Video VPICLK function pin** SCU90[5]=1 select Video pin
20	RW	**Enable UART1 NDTR1 or Video VPIVS function pin** SCU90[5]=1 select Video pin
19	RW	**Enable UART1 NRI1 or Video VPIHS function pin** SCU90[5]=1 select Video pin
18	RW	**Enable UART1 NDSR1 function pin** SCU90[5]=1 select Video pin
17	RW	**Enable UART1 NDCD1 or Video VPIDE function pin** SCU90[5]=1 select Video pin
16	RW	**Enable UART1 NCTS1 function pin**
15	RW	**Enable VGA/CRT DDCDAT output function pin**
14	RW	**Enable VGA/CRT DDCCLK output function pin**
13	RW	**Enable VGA/CRT VGAVS output function pin**
12	RW	**Enable VGA/CRT VGAHS output function pin**
11	RW	**Enable Master SGPIO SGPMI input function pin**
10	RW	**Enable Master SGPIO SGPMO output function pin**
9	RW	**Enable Master SGPIO SGPMLD output function pin**
8	RW	**Enable Master SGPIO SGPMCK output function pin**
7	RW	**Enable I2C4 dedicated Alert SALT4 function pin**
6	RW	**Enable I2C3 dedicated Alert SALT3 function pin**
5	RW	**Enable I2C2 dedicated Alert SALT2 function pin**
4	RW	**Enable I2C1 dedicated Alert SALT1 function pin**
3	RW	**Enable Slave SGPIO SGPS1I1 input function pin**
2	RW	**Enable Slave SGPIO SGPS1I0 input function pin**
1	RW	**Enable Slave SGPIO SGPS1LD input function pin**
0	RW	**Enable Slave SGPIO SGPS1CK input function pin**

Offset: 88h **SCU88: Multi-function Pin Control #3** **Init = 0x03000000**

Bit	R/W	Description
31	RW	**Enable MAC#1 MDIO1 function pin**
30	RW	**Enable MAC#1 MDC1 function pin**
29	RW	**Enable flash SPI2MISO function pin**
28	RW	**Enable flash SPI2MOSI function pin**
27	RW	**Enable flash SPI2CK function pin**
26	RW	**Enable flash SPI2CS0# function pin**
25	RW	**Enable flash FWSPICS2# function pin**
24	RW	**Enable flash FWSPICS1# function pin**
23:20	RW	**Reserved**
19	RW	**Enable VPIR9 function pin** SCU90[5:4]=0x2 select Video pin
18	RW	**Enable VPIR8 function pin** SCU90[5:4]=0x2 select Video pin

to next page

from previous page

17	RW	**Enable VPIR7 function pin** SCU90[5:4]=0x2 select Video pin
16	RW	**Enable VPIR6 function pin** SCU90[5:4]=0x2 select Video pin
15	RW	**Enable VPIR5 function pin** SCU90[5:4]=0x2 select Video pin
14	RW	**Enable VPIR4 function pin** SCU90[5:4]=0x2 select Video pin
13	RW	**Enable VPIR3 function pin** SCU90[5:4]=0x2 select Video pin
12	RW	**Enable VPIR2 function pin** SCU90[5:4]=0x2 select Video pin
11	RW	**Reserved**
10	RW	**Reserved**
9	RW	**Enable VPIG9 function pin** SCU90[5:4]=0x2 select Video pin
8	RW	**Enable VPIG8 function pin** SCU90[5:4]=0x2 select Video pin
7	RW	**Enable PWM7 or VPIG7 function pin** SCU90[5:4]=0x2 select Video pin
6	RW	**Enable PWM6 or VPIG6 function pin** SCU90[5:4]=0x2 select Video pin
5	RW	**Enable PWM5 or VPIG5 function pin** SCU90[5]=1 select Video pin
4	RW	**Enable PWM4 or VPIG4 function pin** SCU90[5]=1 select Video pin
3	RW	**Enable PWM3 or VPIG3 function pin** SCU90[5]=1 select Video pin
2	RW	**Enable PWM2 or VPIG2 function pin** SCU90[5]=1 select Video pin
1	RW	**Enable PWM1 function pin** SCU90[5:4]=0x3 select Video pin
0	RW	**Enable PWM0 function pin** SCU90[5:4]=0x3 select Video pin

Offset: 8Ch **SCU8C: Multi-function Pin Control #4** **Init = 0x0**

Bit	R/W	Description
31	RW	**Disable GPIOP internal pull down resistor**
30	RW	**Disable GPIOO internal pull down resistor**
29	RW	**Disable GPION internal pull down resistor**
28	RW	**Disable GPIOM internal pull down resistor**
27	RW	**Disable GPIOL internal pull down resistor**
26	RW	**Disable GPIOK internal pull down resistor**
25	RW	**Disable GPIOJ internal pull down resistor**
24	RW	**Disable GPIOI internal pull down resistor**
23	RW	**Disable GPIOH/GPIOAC internal pull down resistor**

to next page

from previous page

22	RW	**Disable GPIOG/GPIOAB internal pull down resistor**
21	RW	**Disable GPIOF/GPIOAA internal pull down resistor**
20	RW	**Disable GPIOE/GPIOZ internal pull down resistor**
19	RW	**Disable GPIOD/GPIOY internal pull down resistor**
18	RW	**Disable GPIOC/GPIOS internal pull down resistor**
17	RW	**Disable GPIOB/GPIOR internal pull down resistor**
16	RW	**Disable GPIOA/GPIOQ internal pull down resistor**
15	RW	**Enable GPIOE6 pass-through to GPIOE7 function**
14	RW	**Enable GPIOE4 pass-through to GPIOE5 function**
13	RW	**Enable GPIOE2 pass-through to GPIOE3 function**
12	RW	**Enable GPIOE0 pass-through to GPIOE1 function** GPIOE[7:0] pass through function also can be enabled by hardware strap bit[22]. The pass through can support debounce mode, controlled by SCUA8 bit[27:24]. The programming sequence to change the pass through from strap bit[22] control to per-pit control as below, 1. Enable per-bit control 2. Disable strap bit[22] = 0
11	RW	**Enable GPIOD6 pass-through to GPIOD7 function**
10	RW	**Enable GPIOD4 pass-through to GPIOD5 function**
9	RW	**Enable GPIOD2 pass-through to GPIOD3 function**
8	RW	**Enable GPIOD0 pass-through to GPIOD1 function** GPIOD[7:0] pass through function also can be enabled by hardware strap bit[21]. The pass through can support debounce mode, controlled by SCUA8 bit[23:20]. The programming sequence to change the pass through from strap bit[21] control to per-pit control as below, 1. Enable per-bit control 2. Disable strap bit[21] = 0
7	RW	**Enable VPOB9 function pin** SCU94[1:0]!=0 select Video pin
6	RW	**Enable VPOB8 function pin** SCU94[1:0]!=0 select Video pin
5	RW	**Enable VPOB7 function pin** SCU94[1:0]!=0 select Video pin
4	RW	**Enable VPOB6 function pin or BMCINT output level** SCU94[1:0]!=0 select Video pin. When not used as video pin, it indicates the BMCINT pin output active level. 0: active low 1: active high
3	RW	**Enable VPOB5 or I2C6 dedicated Alert SALT6 function pin** SCU94[1:0]!=0 select Video pin
2	RW	**Enable VPOB4 or I2C5 dedicated Alert SALT5 function pin** SCU94[1:0]!=0 select Video pin
1	RW	**Enable VPOB3 or BMCINT output function pin** SCU94[1:0]!=0 select Video pin
0	RW	**Enable VPOB2 or flash SPI2CS1# function pin** SCU94[1:0]!=0 select Video pin

Offset: 90h	SCU90: Multi-function Pin Control #5	Init = 0x0000A000

Bit	R/W	Description
31	RW	**Enable Parallel NOR function pins**
30	RW	**Enable LPC Plus interface** LPC Plus interface is used to connect AST1070 companion chip.
29	RW	**Enable USB2.0 Host port #1 function** USB2.0 Host controller port 1 is shared with USB2.0 Hub port, when enabled Host mode, Hub function will be disabled.
28	RW	**Reserved**
27	RW	**Enable I2C14 function pins**
26	RW	**Enable I2C13 function pins**
25	RW	**Enable I2C12 function pins**
24	RW	**Enable I2C11 function pins**
23	RW	**Enable I2C10 function pins**
22	RW	**Enable I2C9 function pins**
21	RW	**Enable I2C8 function pins**
20	RW	**Enable I2C7 function pins**
19	RW	**Enable I2C6 function pins**
18	RW	**Enable I2C5 function pins**
17	RW	**Enable I2C4 function pins**
16	RW	**Enable I2C3 function pins**
15	RW	**RGMII2/RMII2 RX pins internal pull down resistor disable** 0: enable 1: disable (default) When select at RMII mode, RGMII2RXCTL pin is controlled by bit14.
14	RW	**RGMII2/RMII2 TX pins internal pull down resistor disable** 0: enable (default) 1: disable Including RGMIICK pin.
13	RW	**RGMII1/RMII1 RX pins internal pull down resistor disable** 0: enable 1: disable (default) When select at RMII mode, RGMII1RXCTL pin is controlled by bit12.
12	RW	**RGMII1/RMII1 TX pins internal pull down resistor disable** 0: enable (default) 1: disable
11:10	RW	**RGMII2/RMII2 TX pins driving strength** x0: RGMII2TXCK/RMII2RCLKO normal driving x1: RGMII2TXCK/RMII2RCLKO high driving 0x: RMII2TXEN/RMII2TXD[1:0]normal driving 1x: RMII2TXEN/RMII2TXD[1:0]high driving
9:8	RW	**RGMII1/RMII1 TX pins driving strength** x0: RGMII1TXCK/RMII1RCLKO normal driving x1: RGMII1TXCK/RMII1RCLKO high driving 0x: RMII1TXEN/RMII1TXD[1:0]normal driving 1x: RMII1TXEN/RMII1TXD[1:0]high driving
7	RW	**Enable UART6 function pins** UART6 pins are shared with GPIOH[7:0].
6	RW	**Reserved, must keep at value "0"**

to next page

from previous page

5:4	RW	**Enable digital video input port function pins** 0x: disable 10: 24 bits (R8/G8/B8) video mode. 11: reserved
3	RW	**Enable SD1 port 8 bits mode** SD1 data bit[7:4] share with SD2 data pins. So when enable 8 bits mode, SD2 can not be enabled (SCU90[1]=0).
2	RW	**Enable MAC#2 MDC2/MDIO2 function pins**
1	RW	**Enable SD2 function pins**
0	RW	**Enable SD1 function pins**

Offset: 94h **SCU94: Multi-function Pin Control #6** **Init = 0**

Bit	R/W	Description
31:24	R	**Reserved**
23:21	RW	**Reserved**
20:15	RW	**Reserved**
14:13	RW	**Select USB2.0 Port#2 function mode** 00: USB1.1 HID controller 01: ~~USB2.0 Device controller~~ 1x: USB2.0 Host2 controller
12	RW	**Enable Slave SGPIO port 2 function pins**
11:5	RW	**Reserved**
4:3	RW	**Reserved**
2	RW	**Reserved**
1:0	RW	**Enable digital video output function pins** 00: disable 01: (R8/G8/B8), 24 bits single edge or 12 bits dual edge video mode. 1x: disable

Offset: 9Ch **SCU9C: EXTRST# Reset Selection** **Init = 0x23FFFF3**

Bit	R/W	Description
31:26	R	**Reserved**
25	RW	**Enable reset Misc. SOC controller** The Misc. controller includes: RTC, Timer, UART, SRAM.
24	RW	**Enable reset SPI controller**
23	RW	**Enable reset X-DMA controller**
22	RW	**Enable reset MCTP controller**
21	RW	**Enable reset GPIO controller**
20	RW	**Enable reset ADC controller**
19	RW	**Enable reset JTAG master controller**
18	RW	**Enable reset PECI controller**
17	RW	**Enable reset PWM controller**
16	RW	**Enable reset CRT mode 2D engine**
15	RW	**Enable reset MIC controller**
14	RW	**Enable reset SD/SDIO controller**

to next page

from previous page

13	RW	**Enable reset LPC controller**
12	RW	**Enable reset HAC engine**
11	RW	**Enable reset Video engine**
10	RW	**Enable reset USB1.1 HID/USB2.0 Host EHCI2 controller**
9	RW	**Enable reset USB1.1 Host controller**
8	RW	**Enable reset USB2.0 Host/Hub controller**
7	RW	**Enable reset Graphics CRT controller**
6	RW	**Enable reset MAC#2 controller**
5	RW	**Enable reset MAC#1 controller**
4	RW	**Enable reset I2C controller**
3	RW	**Enable reset AHB bridges**
2	RW	**Enable reset SDRAM controller**
1	RW	**Enable reset Coprocessor**
0	RW	**Enable reset ARM**
Note : This register controls the IPs to be reset by the EXTRST# pin input.		

Offset: A0h **SCUA0: Multi-function Pin Control #7** **Init = 0**

Bit	R/W	Description
31:24	RW	**Enable GPIW[7:0] input function pins** 0: disable input 1: enable input
23:16	RW	**Enable GPIOV[7:0] function pins** 0: MAC function mode 1: GPIO function mode
15:8	RW	**Enable GPIOU[7:0] function pins** 0: MAC function mode 1: GPIO function mode
7:0	RW	**Enable GPIOT[7:0] function pins** 0: MAC function mode 1: GPIO function mode

Offset: A4h **SCUA4: Multi-function Pin Control #8** **Init = 0x0000F000**

Bit	R/W	Description
31	RW	**Enable I2C14 dedicated Alert SALT14 or VPOR9 function pin** SCU94[1:0]!=0 select Video pin
30	RW	**Enable I2C13 dedicated Alert SALT13 or VPOR8 function pin** SCU94[1:0]!=0 select Video pin
29	RW	**Enable I2C12 dedicated Alert SALT12 or VPOR7 function pin** SCU94[1:0]!=0 select Video pin
28	RW	**Enable I2C11 dedicated Alert SALT11 or VPOR6 function pin** SCU94[1:0]!=0 select Video pin
27	RW	**Enable I2C10 dedicated Alert SALT10 or VPOR5 function pin** SCU94[1:0]!=0 select Video pin
26	RW	**Enable I2C9 dedicated Alert SALT9 or VPOR4 function pin** SCU94[1:0]!=0 select Video pin

to next page

from previous page

25	RW	**Enable I2C8 dedicated Alert SALT8 or VPOR3 function pin** SCU94[1:0]!=0 select Video pin
24	RW	**Enable I2C7 dedicated Alert SALT7 or VPOR2 function pin** SCU94[1:0]!=0 select Video pin
23	RW	**Enable VPOG9 function pin** SCU94[1:0]!=0 select Video pin
22	RW	**Enable VPOG8 function pin** SCU94[1:0]!=0 select Video pin
21	RW	**Enable VPOG7 function pin** SCU94[1:0]!=0 select Video pin
20	RW	**Enable VPOG6 function pin** SCU94[1:0]!=0 select Video pin
19	RW	**Enable SIOSCI# or VPOG5 function pin** SCU94[1:0]!=0 select Video pin SCI# pin also can be enabled by hardware strap bit[19].
18	RW	**Enable SIOPBO# or VPOG4 function pin** SCU94[1:0]!=0 select Video pin PWRBTO# pin also can be enabled by hardware strap bit[19].
17	RW	**Enable SIOPWRGD or VPOG3 function pin** SCU94[1:0]!=0 select Video pin PWRGOOD pin also can be enabled by hardware strap bit[19].
16	RW	**Enable SIOPBI# or VPOG2 function pin** SCU94[1:0]!=0 select Video pin PWRBTI# pin also can be enabled by hardware strap bit[19].
15	RW	**Enable SDA2 function pin**
14	RW	**Enable SCL2 function pin**
13	RW	**Enable SDA1 function pin**
12	RW	**Enable SCL1 function pin**
11	RW	**Enable SIOONCTRL# function pin** SIOONCTRL# pin also can be enabled by hardware strap bit[19].
10	RW	**Enable SIOPWREQ# function pin** SIOPWREQ# pin also can be enabled by hardware strap bit[19].
9	RW	**Enable SIOS5# function pin** SIOS5# pin also can be enabled by hardware strap bit[19].
8	RW	**Enable SIOS3# function pin** SIOS3# pin also can be enabled by hardware strap bit[19].
7:0	RW	**Enable GPIX[7:0] input function pins** 0: disable input 1: enable input

Offset: A8h **SCUA8: Multi-function Pin Control #9** **Init = 0xFFFF0**

Bit	R/W	Description
31:28	R	**Reserved**
27	RW	**Enable GPIOE[7:6] pass through function debounce mode**
26	RW	**Enable GPIOE[5:4] pass through function debounce mode**
25	RW	**Enable GPIOE[3:2] pass through function debounce mode**
24	RW	**Enable GPIOE[1:0] pass through function debounce mode**

to next page

from previous page

23	RW	**Enable GPIOD[7:6] pass through function debounce mode**
22	RW	**Enable GPIOD[5:4] pass through function debounce mode**
21	RW	**Enable GPIOD[3:2] pass through function debounce mode**
20	RW	**Enable GPIOD[1:0] pass through function debounce mode** The debouncing timer is defined by the GPIO registers.
19	RW	**Disable ADC15 internal pull down resistor**
18	RW	**Disable ADC14 internal pull down resistor**
17	RW	**Disable ADC13 internal pull down resistor**
16	RW	**Disable ADC12 internal pull down resistor**
15	RW	**Disable ADC11 internal pull down resistor**
14	RW	**Disable ADC10 internal pull down resistor**
13	RW	**Disable ADC9 internal pull down resistor**
12	RW	**Disable ADC8 internal pull down resistor**
11	RW	**Disable ADC7 internal pull down resistor**
10	RW	**Disable ADC6 internal pull down resistor**
9	RW	**Disable ADC5 internal pull down resistor**
8	RW	**Disable ADC4 internal pull down resistor**
7	RW	**Disable ADC3 internal pull down resistor**
6	RW	**Disable ADC2 internal pull down resistor**
5	RW	**Disable ADC1 internal pull down resistor**
4	RW	**Disable ADC0 internal pull down resistor**
3	RW	**Enable VPOCLK or Watchdog #2 WDTRST2 output function pin** SCU94[1:0]!=0 select Video pin
2	RW	**Enable VPOVS or Watchdog #1 WDTRST1 output function pin** SCU94[1:0]!=0 select Video pin
1	RW	**Enable VPOHS function pin** SCU94[1:0]!=0 select Video pin
0	RW	**Enable VPODE function pin** SCU94[1:0]!=0 select Video pin

Offset: ACh **SCUAC: Multi-function Pin Control #10** **Init = 0xFF**

Bit	R/W	Description
31:8	R	**Reserved**
7	RW	**Enable LPC LPCRST# function pin**
6	RW	**Enable LPC LSIRQ# function pin**
5	RW	**Enable LPC LFRAME# function pin**
4	RW	**Enable LPC LCLK function pin**
3	RW	**Enable LPC LAD3 function pin**
2	RW	**Enable LPC LAD2 function pin**
1	RW	**Enable LPC LAD1 function pin**
0	RW	**Enable LPC LAD0 function pin**

Offset: B8h	SCUB8: MAC Interface Clock Delay 100M Setting	Init = 0

Bit	R/W	Description
31:25	R	**Reserved**
24	RW	**Enable RGMII 100M delay setting**
23:18	RW	**MAC#2 RGMII_RXCLK 100M clock input delay**
17:12	RW	**MAC#1 RGMII_RXCLK 100M clock input delay**
11:6	RW	**MAC#2 RGMII_TXCLK 100M clock output delay**
5:0	RW	**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 SCU48 when working at 100M speed.

Offset: BCh	SCUBC: MAC Interface Clock Delay 10M Setting	Init = 0

Bit	R/W	Description
31:25	R	**Reserved**
24	RW	**Enable RGMII 10M delay setting**
23:18	RW	**MAC#2 RGMII_RXCLK 10M clock input delay**
17:12	RW	**MAC#1 RGMII_RXCLK 10M clock input delay**
11:6	RW	**MAC#2 RGMII_TXCLK 10M clock output delay**
5:0	RW	**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 SCU48 when working at 10M speed.

Offset: C0h	SCUC0: Power Saving Wakeup Enable Register	Init = 0

Bit	R/W	Description
31:0	W	**Enable Power Saving Wakeup** Write 0xFEEDA5A5 to this register to enable the power saving wakeup mode.

Offset: C4h	SCUC4: Power Saving Wakeup Control Register	Init = 0

Bit	R/W	Description
31:9	R	**Reserved**
8	RW	**Enable Wakeup timer test mode** 0: wakeup delay 1.3 ms 1: wakeup delay 5 us
7	RW	**Set the wakeup event polarity for USB Host** It must set to '1' for this event.
6	RW	**Set the wakeup event polarity for GPIOF5**
5	RW	**Set the wakeup event polarity for GPIOF4**
4	RW	**Set the wakeup event polarity for GPIOF0** 0: active low 1: active high
3	RW	**Enable wakeup from USB Host controller**
2	RW	**Enable wakeup from GPIOF5**
1	RW	**Enable wakeup from GPIOF4**
0	RW	**Enable wakeup from GPIOF0**

Offset: D4h	SCUD4: System Reset Control Register Set 2	Init = 0x1FF

Bit	R/W	Description
31:9	R	**Reserved**
8:6	RW	**Reserved, must keep value at "111"**
5	RW	**Reset CRT controller** 0: No operation 1: Reset CRT1 controller (asynchronous reset) (default)
4:0	RW	**Reserved, must keep value at "11111"**

Offset: D8h	SCUD8: Clock Selection Register Set 2	Init = 0x3D

Bit	R/W	Description
31	RW	**Reserved**
30	RW	**Video input port clock delay control** 0: delay is controlled by VR008[11:9]+bit[26:24] 1: delay is controlled by bit[29:24]
29:24	RW	**Video input port clock delay selection** bit[29]: inverted output bit[28:24]: delay line stage
23:12	RW	**Reserved**
11:6	RW	**Video port output clock delay control** bit[11]: inverted output bit[10:6]: delay line stage
5:3	RW	**Reserved**
2:0	RW	**P-Bus BCLK divider selection** 000: BCLK = H-PLL/4 001: BCLK = H-PLL/8 010: BCLK = H-PLL/12 011: BCLK = H-PLL/16 100: BCLK = H-PLL/20 101: BCLK = H-PLL/24 110: BCLK = H-PLL/28 111: BCLK = H-PLL/32

Offset: DCh	SCUDC: Clock Stop Control Register Set 2	Init = 0x7FF

Bit	R/W	Description
10:9	RW	**Reserved, must keep at value "11"**
8:6	RW	**Reserved, must keep at value "111"**
4:3	RW	**Reserved, must keep at value "11"**
2:0	RW	**Reserved, must keep at value "111"**

Offset: E0h	SCUE0: SCU Free Run Counter Read Back	Init = 0

Bit	R/W	Description
31:0	R	**SCU free run counter bit[31:0] read back** The SCU free run counter is a 48-bit couter. Its value is reset by SRST# signal. The counter tick is 24MHz, but the read back register of the counter is updated every 24 ticks (1MHz).

Offset: E4h	SCUE4: SCU Free Run Counter Extended Read Back	Init = 0

Bit	R/W	Description
31:16	R	**Reserved**
15:0	R	**SCU free run counter bit[47:32] read back**

Offset: E8h	SCUE8: Clock Duty Measurement Control	Init = 0x0

Bit	R/W	Description
31:22	R	**Reserved**
21	R	**VPODCLK Duty count done status**
20	R	**VPOCCLK Duty count done status**
19	R	**VPOBCLK Duty count done status**
18	R	**VPOACLK Duty count done status**
17	R	**RGMII2TXCK Duty count done status**
16	R	**RGMII1TXCK Duty count done status**
15:7	R	**Reserved**
6:4	RW	**Measurement result read select** 000: RGMII1TXCK 001: RGMII2TXCK 010: VPOACLK 011: VPOBCLK 100: VPOCCLK 101: VPODCLK
3	RW	**Start calculate duty**
2:1	RW	**Ring clock select**
0	RW	**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: ECh	SCUEC: Clock Duty Measurement Result	Init = X

Bit	R/W	Description
29:16	R	**N-phase counting value**
13:0	R	**P-phase counting value**

Offset: 100h	SCU100: Coprocessor (CPU2) Control Register	Init = 0

Bit	R/W	Description
31:3	RW	**Reserved**
2	RW	**Enable Coprocessor ICE debug pin** 0: Disable 1: Enable When Coprocessor disabled, it is also be disabled.
1	RW	**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	**Coprocessor Enable** 0: Disable 1: Enable When turned OFF, it must keep at least 1 us. There are 2 ways to enable the coprocessor. 1. Enable this register 2. BOND for AST2510 and HWTrap[1:0] != ”11”

Offset: 104h	SCU104: CPU2 Base Address for segment 0x00:0000-0x0F:FFFF	Init = 0x00000001

Bit	R/W	Description
31:20	RW	**Base address bit[31:20]**
19:1		**Reserved**
0	RW	**Big Endian swap enable** 0: No swap 1: enable swap For flash and DRAM memory space, it needs turn on swap mode.
Note : When coprocessor write this register, it must follow by a read register(any SCU register) command to flush the bus before access to the new address.		

Offset: 108h	SCU108: CPU2 Base Address for segment 0x10:0000-0x1F:FFFF	Init = 0x00100001

Bit	R/W	Description
31:20	RW	**Base address bit[31:20]**
19:1		**Reserved**
0	RW	**Big Endian swap enable** 0: No swap 1: enable swap For flash and DRAM memory space, it needs turn on swap mode.
Note : When coprocessor write this register, it must follow by a read register(any SCU register) command to flush the bus before access to the new address.		

Offset: 10Ch	SCU10C: CPU2 Base Address for segment 0x20:0000-0x2F:FFFF	Init = 0x1E600000

Bit	R/W	Description
31:20	RW	**Base address bit[31:20]**
19:1		**Reserved**

to next page

from previous page

0	RW	**Big Endian swap enable** 0: No swap 1: enable swap For flash and DRAM memory space, it needs turn on swap mode.
Note : When coprocessor write this register, it must follow by a read register(any SCU register) command to flush the bus before access to the new address.		

Offset: 110h SCU110: CPU2 Base Address for segment 0x30:0000-0x3F:FFFF Init = 0x1E700000

Bit	R/W	Description
31:20	RW	**Base address bit[31:20]**
19:1		**Reserved**
0	RW	**Big Endian swap enable** 0: No swap 1: enable swap For flash and DRAM memory space, it needs turn on swap mode.
Note : This register is default mapped to the register space. It is not recommended for the coprocessor to update this register.		

Offset: 114h SCU114: CPU2 Base Address for segment 0x40:0000-0x4F:FFFF Init = 0x0

Bit	R/W	Description
31:20	RW	**Base address bit[31:20]**
19:1		**Reserved**
0	RW	**Big Endian swap enable** 0: No swap 1: enable swap For flash and DRAM memory space, it needs turn on swap mode.
Note : This register is default mapped to the register space. It is not recommended for the coprocessor to update this register.		

Offset: 118h SCU118: CPU2 Base Address for segment 0x50:0000-0x5F:FFFF Init = 0x0

Bit	R/W	Description
31:20	RW	**Base address bit[31:20]**
19:1		**Reserved**
0	RW	**Big Endian swap enable** 0: No swap 1: enable swap For flash and DRAM memory space, it needs turn on swap mode.
Note : This register is default mapped to the register space. It is not recommended for the coprocessor to update this register.		

Offset: 11Ch SCU11C: CPU2 Base Address for segment 0x60:0000-0x6F:FFFF Init = 0x1E600000

Bit	R/W	Description
31:20	RW	**Base address bit[31:20]**

to next page

from previous page

19:1		**Reserved**
0	RW	**Big Endian swap enable** 0: No swap 1: enable swap For flash and DRAM memory space, it needs turn on swap mode.

Note :
This register is default mapped to the register space. It is not recommended for the coprocessor to update this register.

Offset: 120h SCU120: CPU2 Base Address for segment 0x70:0000-0x7F:FFFF Init = 0x1E700000

Bit	R/W	Description
31:20	RW	**Base address bit[31:20]**
19:1		**Reserved**
0	RW	**Big Endian swap enable** 0: No swap 1: enable swap For flash and DRAM memory space, it needs turn on swap mode.

Note :
This register is default mapped to the register space. It is not recommended for the coprocessor to update this register.

Offset: 124h SCU124: CPU2 Base Address for segment 0x80:0000-0xFF:FFFF Init = 0x80000001

Bit	R/W	Description
31:23	RW	**Base address bit[31:23]**
22:1		**Reserved**
0	RW	**Big Endian swap enable** 0: No swap 1: enable swap For flash and DRAM memory space, it needs turn on swap mode.

Note :
When coprocessor write this register, it must follow by a read register(any SCU register) command to flush the bus before access to the new address.

Offset: 128h SCU128: CPU2 Cache Function Control Init = 0

Bit	R/W	Description
31:10		**Reserved**
9	RW	**Enable cache function for address 0x800000 - 0xFFFFFF**
8	RW	**Enable cache function for address 0x700000 - 0x7FFFFF**
7	RW	**Enable cache function for address 0x600000 - 0x6FFFFF**
6	RW	**Enable cache function for address 0x500000 - 0x5FFFFF**
5	RW	**Enable cache function for address 0x400000 - 0x4FFFFF**
4	RW	**Enable cache function for address 0x300000 - 0x3FFFFF**
3	RW	**Enable cache function for address 0x200000 - 0x2FFFFF**
2	RW	**Enable cache function for address 0x100000 - 0x1FFFFF**
1	RW	**Enable cache function for address 0x000000 - 0x0FFFFF** 0: disable 1: enable

to next page

from previous page

0	RW	**Cache Enable** 0: disable cache 1: enable cache

Note :
To dirty specific cache line (due to DMA operation), write the target address for dirty to below address:
I-cache : 0x1E608000
D-cache : 0x1E608004

The code for dirty the cache line in an specific address range:

```
sadr = start_address_for_dirty;
eadr = end_address_for_dirty;
do{
    iowrite(0x1E608000,sadr);          // 0x1E608004 for D-cache line
    sadr = (sadr + 32);
}while(sadr < eadr);
iowrite(0x1E608000,eadr);          // this line is executed depends on the end address wants to be dirty or not
```

Offset: 130h **SCU130: D-PLL Extended Parameter Register** **Init = 0x0**

Bit	R/W	Description
13:5	RW	**D-PLL parameter bit[36:28]**
4:3	RW	**Select D-PLL mode**
2	RW	**Enable D-PLL reset** 0: Normal operation 1: Reset PLL
1	RW	**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 directly from the external CLKIN input pin.
0	RW	**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 is always ”0”. D-PLL default is OFF to reduce the power when power up.

Offset: 134h **SCU134: D-PLL Extended Parameter Register** **Init = 0x0**

Bit	R/W	Description
28	R	**D-PLL lock status**
27:0	RW	**D-PLL parameter bit[27:0]**

Offset: 138h **SCU138: D-PLL Extended Parameter Register** **Init = 0x0**

Bit	R/W	Description
18:0	RW	**D-PLL Fraction**

Offset: 13Ch **SCU13C: D2-PLL Extended Parameter Register** **Init = 0x0**

Bit	R/W	Description
13:5	RW	**D2-PLL parameter bit[36:28]**
4:3	RW	**Select D2-PLL mode**

to next page

from previous page

2	RW	**Enable D2-PLL reset** 0: Normal operation 1: Reset PLL
1	RW	**Enable D2-PLL bypass mode** 0: No operation 1: Enable D2-PLL bypass mode When enabling D2-PLL bypass mode, the output clock of D2-PLL is directly from the external CLKIN input pin.
0	RW	**Turn off D2-PLL** 0: No operation 1: Turn Off D2-PLL When D2-PLL is turned off, it will enter power down mode. The output signal is always ”0”. D2-PLL default is OFF to reduce the power when power up.

Offset: 140h **SCU140: D2-PLL Extended Parameter Register** **Init = 0x0**

Bit	R/W	Description
28	R	**D2-PLL lock status**
27:0	RW	**D2-PLL parameter bit[27:0]**

Offset: 144h **SCU144: D2-PLL Extended Parameter Register** **Init = 0x0**

Bit	R/W	Description
18:0	RW	**D2-PLL Fraction**

Offset: 148h **SCU148: Extended Parameter of M/H-PLL Register** **Init = 0x000190019**

Bit	R/W	Description
22	R	**H-PLL lock status**
21:16	RW	**H-PLL parameter bit[15:10]**
6	R	**M-PLL lock status**
5:0	RW	**M-PLL parameter bit[15:10]**

Offset: 150h **SCU150: Chip Unique ID L** **Init = 0xxxxx**
Offset: 154h **SCU154: Chip Unique ID H** **Init = 0xxxxx**

Bit	Attr.	Description
63:60	R	**ADC compensation**
59:0	R	**Chip unique ID bit [63:0]**

SCU160: Generate UART 24 MHz Reference from H-PLL when CLKIN is 25 MHz

Offset: 160h **Init = 0x00011320**

Bit	Attr.	Description
31:18	R	**Reserved**
17:8	RW	**N-Value**
7:0	RW	**R-Value**

Note :
CLK24M frequency = H-PLL * R / (N * 4)

Offset: 180h	SCU180: PCI-Express Configuration Setting Contol Register	Init = 0x000C007B
Bit	**R/W**	**Description**
31:22	R	**Reserved**
21	RW	**Enable memory mapped LPC decode on BMC device**
20	RW	**Enable memory mapped LPC decode on VGA device**
19:16	RW	**Reserved**
15	RW	**Enable E2L**
14	RW	**Enable PCI Express DMA on BMC device**
13	RW	**Enable interrupt on BMC device**
12	RW	**Enable MCTP on BMC device**
11	RW	**Enable MSI on BMC device**
10	RW	**Enable relocate LPC IO on BMC device**
9	RW	**Enable BMC MMIO on BMC device**
8	RW	**Enable BMC device (device 1)**
7	RW	**Disable MSI function on VGA device or BMC device**
6	RW	**Enable PCI Express DMA on VGA device**
5	RW	**Enable interrupt on VGA device**
4	RW	**Enable MCTP on VGA device**
3	RW	**Enable MSI on VGA device**
2	RW	**Enable relocate LPC IO decode on VGA device**
1	RW	**Enable BMC MMIO decode on VGA device**
0	RW	**Enable VGA device (device 0)**

Offset: 184h	SCU184: BMC MMIO Decode Setting Register	Init = 0
Bit	**R/W**	**Description**
31:20	R	**Reserved**
19:13	RW	**Reserved**
12	RW	**Set MSI 32-bit mode**
11:9	RW	**Reserved**
8	RW	**Enable MMIO offset 0x0e000 to shared DRAM area location decode**
7	RW	**Enable MMIO offset 0x0d000 to shared SRAM area location decode**
6	RW	**Enable MMIO offset 0x0c000 to mailbox location decode**
5	RW	**Enable MMIO offset 0x07000 to LPC or relocated controller decode**
4	RW	**Enable MMIO offset 0x06000 to PECI or relocated controller decode**
3	RW	**Enable MMIO offset 0x05000 to I2C controller decode**
2	RW	**Enable MMIO offset 0x04000 to GPIO controller decode**
1	RW	**Enable MMIO offset 0x03000 to PWM/TACH controller decode**
0	RW	**Enable MMIO offset 0x02000 to ADC controller decode**

Offset: 188h	SCU188: First relocated controller decode area location	Init = 0
Bit	**R/W**	**Description**
31:12	RW	**First relocated controller decode area location**
11:0	R	**Reserved**

Offset: 18Ch	SCU18C: Second relocated controller decode area location	Init = 0
Bit	**R/W**	**Description**
31:12	RW	**Second relocated controller decode area location**
11:0	R	**Reserved**

Offset: 190h	SCU190: Mailbox decode area location	Init = 0
Bit	**R/W**	**Description**
31:12	RW	**Mailbox decode area location**
11:0	R	**Reserved**

Offset: 194h	SCU194: Shared SRAM area decode location	Init = 0
Bit	**R/W**	**Description**
31:15	R	**Reserved**
14:12	RW	**Shared SRAM area decode location** The high address bit[31:16] is 0x1e72, and bit15 is 0.
11:0	R	**Reserved**

Offset: 198h	SCU198: Shared SRAM area decode location	Init = 0
Bit	**R/W**	**Description**
31:29	R	**Reserved**
28:12	RW	**Shared SRAM area decode location** The high address bit[31:29] is 0x2.
11:0	R	**Reserved**

Offset: 19Ch	SCU19C: BMC device class code and revision ID	Init = 0
Bit	**R/W**	**Description**
31:0	RW	**BMC device class code and revision ID**

Offset: 1A4h	SCU1A4: BMC device ID	Init = 0x25032402
Bit	**R/W**	**Description**
31:16	RW	**Reserved**
15:0	RW	**BMC device ID**

Offset: 1DCh	SCU1DC: Clock Duty Selection	Init = 0x0
Bit	**R/W**	**Description**
22:16	RW	**RGMII2TXCK**
14:8	RW	**RGMII1TXCK**
6:0	RW	**DCLK** bit[6]: enable duty calibration bit[5]: invert source clock bit[4:0]: select duty calibration stage

24 Hash & Crypto Engine (HACE)

24.1 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 11 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 = 0x1E6E_3000
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
HACE18: HAC Engine Tag 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
HACE40: RSA Exponent Value Register
HACE4C: RSA Engine Command 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

24.2 Features

- Directly connected to APB bus
- Register programming through APB 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)
  - 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 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/SHA1/SHA224/SHA256, HMAC-MD5/HMAC-SHA1/HMAC-SHA224/HMAC-SHA256
- 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
- Performance Target (200MHz memory clock)
  - RC4 : throughput up to 230 Mbps.
  - DES : throughput up to 490 Mbps.
  - 3DES : throughput up to 205 Mbps.
  - AES-128 : throughput up to 350 Mbps.
  - AES-192 : throughput up to 300 Mbps.
  - AES-256 : throughput up to 265 Mbps.
  - MD5/SHA-1/SHA-224/SHA-256 : throughput up to 295 Mbps.
  - HMAC-MD5/HMAC-SHA-1/HMAC-SHA-224/HMAC-SHA-256 : throughput up to 295 Mbps.

24.3 Registers : Base Address = 0x1E6E:3000

Offset: 00h	HACE00: Crypto Data Source Base Address Register	Init = X
Bit	**R/W**	**Description**
31:30		**Reserved (0)**
29:0	RW	**Base address of crypto data source**
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**
31:30		**Reserved (0)**
29:0	RW	**Base address of crypto data destination**
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**
31:30		**Reserved (0)**
29:3	RW	**Base address of crypto context buffer** The base address of crypto context buffer must be 8-byte aligned.
2 :0		**Reserved (0)**

Offset: 0Ch	HACE0C: Crypto Data Length Register	Init = X
Bit	**R/W**	**Description**
31:28		**Reserved (0)**
27:0	RW	**Crypto data length (bytes)** 0: **Invalid** 1: 1 byte 2: 2 bytes ... The register determines the data length to be encrypted or decrypted. The data length for a **RC4** crypto command is **byte aligned**. The data length for a **AES** crypto command is **16-byte aligned**. The data length for a **DES** crypto command is **8-byte aligned**. The maximum data length is up to **(256MB-1) bytes** for a **RC4/AES/DES** crypto command. The minimum data length can be **1 byte** for a **RC4** crypto command. The minimum data length can be **16 bytes** for a **AES** crypto command. The minimum data length can be **8 bytes** for a **DES** crypto command.
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**
31:18		**Reserved (0)**

to next page

from previous page

17	RW	**Enable triple DES** 0: Single DES 1: Triple DES
16	RW	**AES/DES Engine Selection** 0: AES Engine 1: DES Engine
15:13		**Reserved (0)**
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: ECB mode (Initial Vector is **NOT** required) 001: CBC mode (Initial Vector is required) 010: CFB mode (Initial Vector is required) 011: OFB mode (Initial Vector is required) 100: CTR mode (Initial Vector is required) 101: **Invalid** 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: 18h	**HACE18: HAC Engine Tag Register**	**Init = X**
Bit	**R/W**	**Description**
31	RW	**Enable write tag interrupt** 0: Disable write tag interrupt 1: Enable write tag interrupt when write value to HAC engine tag register
30	RW	**Wait RSA Engine IDLE Control** 0: Update the tag value without checking whether the RSA engine is IDLE 1: Update the tag value with checking the RSA Engine is IDLE
29	RW	**Wait Crypto Engine IDLE Control** 0: Update the tag value without checking whether the crypto engine is IDLE 1: Update the tag value with checking the crypto engine is IDLE
28	RW	**Wait Hash Engine IDLE Control** 0: Update the tag value without checking whether the hash engine is IDLE 1: Update the tag value with checking the hash engine is IDLE
27: 0	RW	**HAC engine tag register[27:0]**

Offset: 1Ch	**HACE1C: HAC Engine Status Register**	**Init = 0**
Bit	**R/W**	**Description**
31:16		**Reserved (0)**
15	RW	**Write Tag interrupt flag** 0: No interrupt 1: Interrupt is pending When write tag interrupt is enabled, this bit will be set to ”1” when write value to HAC engine tag register. Writing ”1” to this bit will clear this register.
14		**Reserved (0)**
13	RW	**RSA interrupt flag** 0: No interrupt 1: Interrupt is pending When RSA interrupt is enabled, this bit will be set to ”1” when RSA command has been finished. Writing ”1” to this bit will clear this register.
12	RW	**Crypto interrupt flag** 0: No interrupt 1: Interrupt is pending When crypto interrupt is enabled, this bit will be set to ”1” when crypto command has been finished. Writing ”1” to this bit will clear this register.

to next page

from previous page

11:10		**Reserved (0)**
9	RW	**Hash interrupt flag** 0: No interrupt 1: Interrupt is pending When hash interrupt is enabled, this bit will be set to ”1” when hash command has been finished. Writing ”1” to this bit will clear this register.
8 :3		**Reserved (0)**
2	R	**RSA engine status flag** 0: RSA engine is idle 1: RSA engine is busy
1	R	**Crypto engine status flag** 0: Crypto engine is idle 1: Crypto engine is busy
0	R	**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
31:30		**Reserved (0)**
29:0	RW	**Base address of hash data source**

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
31:30		**Reserved (0)**
29:3	RW	**Base address of hash digest write buffer** The base address of hash digest write buffer must be 8-byte aligned.
2 :0		**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

Offset: 28h **HACE28: Hash HMAC Key Buffer Base Address Register** **Init = X**

Bit	R/W	Description
31:30		**Reserved (0)**
29:6	RW	**Base address of HMAC key buffer** The base address of HMAC key buffer must be 64-byte aligned.
5 :0		**Reserved (0)**

to next page

from previous page

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		**Reserved (0)**
27:0	RW	**Hash data length** 0: 0 byte 1: 1 byte 2: 2 bytes ... The register determines the data length to be hashed. When HACE30 [8:7] = 2, the data length for a hash command is **64 byte aligned**. When HACE30 [8:7] != 2, the data length for a hash command is **byte aligned**. The maximum data length is up to **(256MB-1) bytes** for a hash command. When HACE30 [8:7] = 2, the minimum data length can be **64 byte** for a hash command. When HACE30 [8:7] != 2, the minimum data length can be **0 byte** for a hash command.

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
31:10		**Reserved (0)**
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)
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: **Invalid** 111: **Invalid**
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 hash commands (big-endian) 11: **Invalid**

to next page

from previous page

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: 40h	**HACE40: RSA Modulator and Exponent Bit Number Register**	**Init = X**
Bit	**R/W**	**Description**
31:28	RW	**Reserved (0)**
27:16	RW	**RSA modulator bit number** The RSA engine support: A = (X ^{e}) mod n This register is used to set the modulator 'n' effective bit number. Its minimum value is 32.
15:12	RW	**Reserved (0)**
11: 0	RW	**RSA exponent bit number** This register is used to set the exponent 'e' effective bit number. Its minimum value is 2.

Offset: 4Ch	**HACE4C: RSA Engine Command Register**	**Init = 0**
Bit	**R/W**	**Description**
31:14		**Reserved (0)**
13	RW	**Enable RSA interrupt** 0: Disable hash interrupt 1: Enable hash interrupt when hash command finished
12	RW	**Enable RSA extended SRAM in engine mode** 0: RSA extended SRAM is accessable by ARM CPU 1: RSA extended SRAM is accessable by RSA engine
11		**Fire RSA engine selection** 0: Disable fire RSA engine 1: Enable fire RSA engine
10: 0		**Reserved (0)**

Offset: 50h	**HACE50: Command Queue Base Address**	**Init = X**
Bit	**R/W**	**Description**
31:30		**Reserved (0)**
29: 3	RW	**Command queue base address[29:3]**
2: 0		**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		**Reserved (0)**
11: 0	RW	**Command queue end pointer[11:0]**
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		**Reserved (0)**
11: 0	RW	**Command queue write pointer[11:0]**

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		**Reserved (0)**
11: 0	R	**Command queue read pointer[11:0]**

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 ~ HACE10, HACE18, HACE20 ~ HACE30 register data from APB bus. 1: Write HACE00 ~ HACE10, HACE18, HACE20 ~ HACE30 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: 0		**Reserved (0)**

24.4 RSA Buffer Format in Extended SRAM : Extended SRAM Base Address = 0x1E72:0000

The RSA accelerator calculate: A = (X^e) mod n Software need to initialize SRAM buffer with following values: e, X, n, n'. The n' can be calculated by extended Euclidean algorithm with following equation: R * R' - n * n' = 1 The R is 2^r. The r is the effective bit number of n. The software only requires to calculate n' once for each crypto key. The result A of RSA calculation is stored at the same location as the input value X.

Offset	Description
8000 - 81FF	e Buffer
8200 - 83FF	X/A Buffer
8400 - 85FF	n' Buffer
8600 - 87FF	Temp 1 Buffer
8800 - 89FF	n Buffer
8A00 - 8BFF	Temp 2 Buffer
8C00 - 8FFF	Temp 3 Buffer

24.5 Crypto Context Buffer Format

24.5.1 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

24.5.2 AES-128 (192 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

24.5.3 AES-192 (224 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

24.5.4 AES-256 (256 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

24.5.5 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

24.6 Hash Function Programming Sequence

24.6.1 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
  - SHA1 : Digest is 20 bytes, digest write buffer is 20 bytes
  - SHA224 : Digest is 28 bytes, digest write buffer is 32 bytes
  - SHA256 : Digest is 32 bytes, digest write buffer is 32 bytes
- $Hash\_Acc\_Digest\_Write\_Buffer$ (64-byte aligned): Accumulative hash digest write buffer.
  - MD5 : Accumulative digest is 16 bytes, accumulative digest write buffer is 16 bytes
  - SHA1 : Accumulative digest is 20 bytes, accumulative digest write buffer is 20 bytes
  - SHA224 : Accumulative digest is 28 bytes, accumulative digest write buffer is 32 bytes
  - SHA256 : Accumulative digest is 32 bytes, accumulative digest write buffer is 32 bytes

- $Hash\_Acc\_Digest\_Base\_Adr$ (64-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): 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 K_0 buffer.
- $HMAC\_Key\_Buffer\_Base\_Adr$ (64-byte aligned): Base address of 64 byte buffer which is used to store the result of calculate HMAC key command (HACE30 [8:7] = 3).

24.6.2 MD5/SHA1/SHA224/SHA256

- $Hash\_Input\_Data\_Base\_Adr$ (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
   - SHA1 : 28h or 228h
   - SHA224 : 48h or 248h
   - SHA256 : 58h or 258h

24.6.3 HMAC MD5/SHA1/SHA224/SHA256

- $K_0\_Buffer\_Base\_Adr$ (8-byte aligned)
- $HMAC\_Key\_Buffer\_Base\_Adr$ **(64-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 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 = $HMAC\_Key\_Buffer\_Base\_Adr$ **(64-byte aligned)**
   (c) HACE2C = 40h
   (d) HACE30:
   - HMAC-MD5 : 184h or 384h
   - HMAC-SHA1 : 1A8h or 3A8h
   - HMAC-SHA224 : 1C8h or 3C8h
   - HMAC-SHA256 : 1D8h or 3D8h

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 = $HMAC\_Key\_Buffer\_Base\_Adr$ **(64-byte aligned)**
   (d) HACE2C = $Hash\_Input\_Size$ (byte aligned)
   (e) HACE30:
   - HMAC-MD5 : 84h or 284h
   - HMAC-SHA1 : A8h or 2A8h
   - HMAC-SHA224 : C8h or 2C8h
   - HMAC-SHA256 : D8h or 2D8h

24.6.4 Accumulative Mode

- $HMAC\_Key\_Buffer\_Base\_Adr$ **(64-byte aligned)**
- $Hash\_Input\_Data\_Base\_Adr$ (byte aligned)
- $Hash\_Acc\_Digest\_Base\_Adr$ **(64-byte aligned)**
- $Hash\_Acc\_Input\_Size$ **(64-byte aligned)**

1. Allocating & Initiating Accumulative Hash Digest Write Buffer ($Hash\_Acc\_Digest\_Write\_Buffer$):
   This sequence is neeed ONLY before processing first accumulative data.

- MD5/SHA1/SHA224/SHA256 ($Hash\_Acc\_Digest\_Write\_Buffer$):
  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

- Phase 1 of HMAC MD5/SHA1/SHA224/SHA256 ($Hash\_Acc\_Digest\_Write\_Buffer$):
  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/SHA1/SHA224/SHA256
00 - 03	HMAC_Key_Buffer[00:03] (MD5/SHA1/SHA224/SHA256)
04 - 07	HMAC_Key_Buffer[04:07] (MD5/SHA1/SHA224/SHA256)
08 - 0B	HMAC_Key_Buffer[08:0B] (MD5/SHA1/SHA224/SHA256)
0C - 0F	HMAC_Key_Buffer[0C:0F] (MD5/SHA1/SHA224/SHA256)
10 - 13	HMAC_Key_Buffer[10:13] (SHA1/SHA224/SHA256)
14 - 17	HMAC_Key_Buffer[14:17] (SHA224/SHA256)
18 - 1B	HMAC_Key_Buffer[18:1B] (SHA224/SHA256)
1C - 1F	HMAC_Key_Buffer[1C:1F] (SHA224/SHA256)

- Phase 2 of HMAC MD5/SHA1/SHA224/SHA256 ($Hash\_Acc\_Digest\_Write\_Buffer$):
  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/SHA1/SHA224/SHA256
00 - 03	HMAC_Key_Buffer[20:23] (MD5/SHA1/SHA224/SHA256)
04 - 07	HMAC_Key_Buffer[24:27] (MD5/SHA1/SHA224/SHA256)
08 - 0B	HMAC_Key_Buffer[28:2B] (MD5/SHA1/SHA224/SHA256)
0C - 0F	HMAC_Key_Buffer[2C:2F] (MD5/SHA1/SHA224/SHA256)
10 - 13	HMAC_Key_Buffer[30:33] (SHA1/SHA224/SHA256)
14 - 17	HMAC_Key_Buffer[34:37] (SHA224/SHA256)
18 - 1B	HMAC_Key_Buffer[38:3B] (SHA224/SHA256)
1C - 1F	HMAC_Key_Buffer[3C:3F] (SHA224/SHA256)

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 and SHA1/SHA224/SHA256.

   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
SHA1	N * 8	(64 + N) * 8	(64 + 20) * 8
SHA224	N * 8	(64 + N) * 8	(64 + 28) * 8
SHA256	N * 8	(64 + N) * 8	(64 + 32) * 8

   (b) HACE20 = $Hash\_Input\_Data\_Base\_Adr$ (byte aligned)
   (c) HACE24 = $Hash\_Acc\_Digest\_Base\_Adr$ **(64-byte aligned)**
   (d) HACE28 = $Hash\_Acc\_Digest\_Base\_Adr$ **(64-byte aligned)**
   (e) HACE2C = $Hash\_Acc\_Input\_Size$ **(64-byte aligned)**
   (f) HACE30:
   - MD5 or HMAC-MD5 : 104h or 304h
   - SHA1 or HMAC-SHA1 : 128h or 328h
   - SHA224 or HMAC-SHA224 : 148h or 348h
   - SHA256 or HMAC-SHA256 : 158h or 358h

24.7 Command Queue Data Format

- Command queue data Size: 8 Bytes
- Maximum capacity of Command queue buffer: 32 KBytes

Offset: 0h	HACCQ#0: Command Queue Data[31:0]	Init = X
Bit	**R/W**	**Description**
31:0		**Register wirte data[31:0]**

Offset: 4h	HACCQ#1: Command Queue Data[64:32]	Init = X
Bit	**R/W**	**Description**
31:4	RW	**Command Queue Data ID[27:0] (0x6655990)**
3 :0	RW	**Register wirte adddress[3:0]**

25 JTAG Master Controller

25.1 Overview

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.

JTAG00: Data register.
JTAG04: Instruction register.
JTAG08: Engine control.
JTAG0C: Interrupt status and enable.
JTAG10: Test mode and status.
JTAG14: TCK control.

25.2 Features

- Following state diagram in section 6.1 of IEEE 1149-1.
- Flexible instruction/data combination.
- Interrupt when data or instruction transmission complete or pause.
- Software mode to direct control TCK, TMS and TDI through APB register.

25.3 Registers : Base Address = 0x1E6E_4000

Offset: 00h	JTAG00: Data register	Init = 0
Bit	**R/W**	**Description**
31: 0	RW	**Data register** Write: value of data to be sent. Read: value of data received.

Offset: 04h	JTAG04: Instruction register	Init = 0
Bit	**R/W**	**Description**
31: 0	RW	**Instruction register** Write: value of instruction to be sent. Read: value of instruction received.

Offset: 08h	JTAG08: Engine control	Init = 0
Bit	**R/W**	**Description**
31	RW	**Engine enable.**
30	RW	**Engine output enable.** 1: output enable. 0: output disable.
29	RW	**Force controller and slave into Reset state by TMS.**
28:27	RW	**Reserved**
26	RW	**Stop State Machine at Update-IR**

to next page

from previous page

25:20	RW	**Length of instuction in one transmission** 0: forbidden 1: 1 bit 32: 32 bits 33 and more is not supported.
19	RW	**MSB first of instruction.** 1: MSB first. 0: LSB first.
18	RW	**Terminating transmission of instruction** 1: terminate. 0: normal operation.
17	RW	**Last transmission of instruction.** 1: last transmission. 0: more data waited.
16	RW	**Enable transmission of instruction.**
15:11	RW	**Reserved**
10	RW	**Stop State Machine at Update-DR**
9: 4	RW	**Length of data in one transmission** 0: forbidden 1: 1 bit 32: 32 bits 33 and more is not supported.
3	RW	**MSB first of data.** 1: MSB first. 0: LSB first.
2	RW	**Terminating transmission of data** 1: terminate. 0: normal operation.
1	RW	**Last transmission of data** 1: last transmission. 0: more data waited.
0	RW	**Enable transmission of data.**

Offset: 0Ch **JTAG0C: Interrupt status and enable** **Init = 0**

Bit	R/W	Description
31:20	RW	**Reserved**
19	RW	**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	**Instruction transmission completed.** 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

17	RW	**Data transmission pause.** Read 0: No interrupt pending Read 1: interrupt pending Write 0: No operation Write 1: Clear interrupt status flag
16	RW	**Data transmission completed.** Read 0: No interrupt pending Read 1: interrupt pending Write 0: No operation Write 1: Clear interrupt status flag
15: 4	RW	**Reserved**
3	RW	**Enable of Instruction transmission pause.** 0: Disable interrupt 1: Enable interrupt
2	RW	**Enable of Instruction transmission completed.** 0: Disable interrupt 1: Enable interrupt
1	RW	**Enable of Data transmission pause.** 0: Disable interrupt 1: Enable interrupt
0	RW	**Enable of Data transmission completed.** 0: Disable interrupt 1: Enable interrupt

Offset: 10h **JTAG10: Software mode and status** **Init = 0**

Bit	R/W	Description
31:20	RW	**Reserved**
19	RW	**Software mode enable.**
18	RW	**Software TCK.** Direct connect to TCK when software mode enabled.
17	RW	**Software TMS.** Direct connect to TMS when software mode enabled.
16	RW	**Software TDI and TDO.** Write: Direct connect to TDI when software mode enabled. Read : Direct connect to TDO when software mode enabled
15: 3	RW	**Reserved**
2	RW	**Instuction transmission pause.**
1	RW	**Data transmission pause.**
0	RW	**Engine idle.**

Offset: 14h **JTAG14: TCK Control** **Init = 0x0000_0007**

Bit	R/W	Description
31	RW	**TCK inverse.** 1: Inverse TCK. 0: Not inverse TCK.
30:11	RW	**Reserved**
10: 0	RW	**TCK divisor.** TCK period = Period of PCLK * (JTAG14[10:0] + 1)

Offset: 18h	JTAG18: Engine Control 1	Init = 0
Bit	**R/W**	**Description**
18	RW	**Control of TRSTn.** 1: TRSTn is high. 0: TRSTn is low.
30: 1		**Reserved**
0	RW	**Return State Machine to Idle/Run-Test state.** Only valid when JTAG08[26] or JTAG08[10] is set. 1: Return to Idel/Run-Test state. 0: No operation.

25.4 Operation

25.4.1 Reset State Machine

- Set JTAG08[31:30] to 0x3.
- Set JTAG08[29]. This will assert TMS for at least 5 cycles.
- Wait JTAG08[29] to be 0.

25.4.2 Instruction Transmission

If instruction length is not larger than 32 bits.

- Write instruction to JTAG04.
- Write instruction length to JTAG08[25:20].
- Set JTAG08[26] if state machine should not go to Idle/Run-Test after this transmission.
- Set JTAG08[17] to indicate that it is last transmission.
- Set JTAG08[16] to enable transmission.
- Wait JTAG0C[18] or JTAG10[0] to be 1.

If insturction length is larger than 32 bits.

- Write lower 32 bits of instruction to JTAG04.
- Write instruction length as 0x20 to JTAG08[25:20].
- Set JTAG08[16] to enable transmission.
- Wait JTAG0C[19] or JTAG10[2] to be 1.
- Write rest of instruction to JTAG04.
- Write rest instruction length to JTAG08[25:20].
- Set JTAG08[17] to indicate that it is last transmission.
- Set JTAG08[26] if state machine should not go to Idle/Run-Test after this transmission.
- Set JTAG08[16] to enable transmission.
- Wait JTAG0C[18] or JTAG10[0] to be 1.

25.4.3 Data Transmission

If data length is not larger than 32 bits.

- Write instruction to JTAG00.
- Write instruction length to JTAG08[9:4].
- Set JTAG08[1] to indicate that it is last transmission.
- Set JTAG08[10] if state machine should not go to Idle/Run-Test after this transmission.
- Set JTAG08[0] to enable transmission.
- Wait JTAG0C[16] or JTAG10[0] to be 1.

If insturction length is larger than 32 bits.

- Write lower 32 bits of instruction to JTAG00.
- Write instruction length as 0x20 to JTAG08[9:4].
- Set JTAG08[0] to enable transmission.
- Wait JTAG0C[17] or JTAG10[1] to be 1.
- Write rest of instruction to JTAG04.
- Write rest instruction length to JTAG08[9:4].
- Set JTAG08[1] to indicate that it is last transmission.
- Set JTAG08[10] if state machine should not go to Idle/Run-Test after this transmission.
- Set JTAG08[0] to enable transmission.
- Wait JTAG0C[16] or JTAG10[0] to be 1.

25.5 Application Note

25.5.1 Introduction

There are 2 operation modes in AST2500 JTAG master controller.
One is hardware mode (or accelerated mode), another 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.

25.5.2 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 detail initiation sequence, please contact ASPEED for the sample code.

25.5.3 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.

26 SOC Display Controller (GFX)

26.1 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
GFX06C: CRT Misc Setting 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
GFX08C: CRT Scaling-Up Factor Register
GFX090: CRT Hardware Cursor X & Y Offset Register
GFX094: CRT Hardware Cursor X & Y Position Register
GFX098: CRT Hardware Cursor Pattern Address Register
GFX09C: CRT 9C Register
GFX0A0: CRT OSD Horizontal Start/End Register
GFX0A4: CRT OSD Vertical Start/End Register
GFX0A8: CRT OSD Pattern Address Register
GFX0AC: CRT OSD Offset Register
GFX0B0: CRT OSD Threshold & Alpha Register
GFX0B4: CRT B4 Register
GFX0B8: CRT Status Register V
GFX0BC: Scratchpad Register
GFX0D0: CRT Display BB0 Starting Address Register
GFX0D4: CRT Display BB1 Starting Address Register
GFX0D8: CRT Display BB Terminal Count Register
GFX0E0: OSD Color Palette Index 1 & 0 Register
GFX0E4: OSD Color Palette Index 3 & 2 Register
GFX0E8: OSD Color Palette Index 5 & 4 Register
GFX0EC: OSD Color Palette Index 7 & 6 Register
GFX0F0: OSD Color Palette Index 9 & 8 Register
GFX0F4: OSD Color Palette Index 11 & 10 Register
GFX0F8: OSD Color Palette Index 13 & 12 Register
GFX0FC: OSD Color Palette Index 15 & 14 Register

26.2 Features

- Support dynamic switching the triple DAC display output between VGA and SOC Display Controller
- Support RGB565 and XRGB8888 graphics display mode
- Support hardware mono/color cursor
- Support On-Screen Display (OSD)
- Maximum display resolution: 1920x1080 32bpp@60Hz

26.3 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	**Enable CRT OSD display** 0: disable 1: enable
3	RW	**Enable CRT interlace timing** 0: disable 1: enable
4	RW	**Enable CRT scaling-up in X-direction** 0: disable 1: enable
5	RW	**Enable CRT scaling-up in Y-direction** 0: disable 1: enable
6	RW	**CRTEnVEFlip**
9:7	RW	**CRT color format selection** 000 : RGB565 (RGB output) 010 : XRGB8888(RGB output) 011 : RGB888 (RGB output) 101 : YUV444 (RGB output) 111 : YUV422 (RGB output) other: Reserved
10	RW	**CRT hardware cursor pattern format** 0: XRGB4444 1: ARGB4444
11	RW	**CRTEnSclD**
13:12	RW	**CRT YUV format selection** 00: Mode0 01: Reserved 10: Reserved 11: Mode3
14	RW	**EnHDTVYUV**
15	RW	**EnTile**
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

to next page

from previous page

19	RW	**Enable CRT vertical sync off** 0: disable 1: enable
20	RW	**Enable CRT screen off** 0: disable 1: enable
21	RW	**CRT field sync polarity selection** 1: negative polarity 0: positive polarity
22	RW	**Enable CRT field sync off** 0: disable 1: enable
23	RW	**Enable CRT desk off** 0: disable 1: enable
24		
25		
26		
27		
28		
29		
30	RW	**Enable CRT vertical interrupt to ARM** 0: disable 1: enable
31	RW	**CRT vertical interrupt status** 0: no interrupt 1: interrupt is pending Writing ”1” will clear this status

Offset: 64h **GFX064: CRT Control Register II** **Init = xxxxxx00h**

Bit	R/W	Description
0	RW	**Power on CRT video DAC** 0: disable 1: enable
1	RW	**Enable CRT video DAC test mode** 0: disable 1: enable
2	RW	**Trig CRT RGB signature generator (triggered by 0 – > 1)**
3	R	**Status of CRT signature generator** 0: busy 1: done
4	RW	**EnTest**
5	R	**FIFOFull**
6	RW	**CRT DVO mode selection** 0: single edge 1: dual edge
7	RW	**Enable CRT DVO** 0: disable 1: enable

to next page

from previous page

19:8	RW	**TestDVO**
31:20	RW	**CRT verical interrupt line number bit[11:0]**

Offset: 68h **GFX068: CRT Status Register** **Init = xxxxxxxxh**

Bit	R/W	Description
0	R	**CRT vertical retrace signal read back**
1	R	**CRT vertical display enable read back**
2	R	**CRT horizontal retrace signal read back**
3	R	**CRT horizontal display enable signal read back**
4	R	**OddField**
5	R	**OddFieldSync**
6	RW	**CRTEnBBFlip**
7	RW	**Enable VGA DAC sense** 0: disable 1: enable
15:8	R	**CRT blue signature read back bit[7:0]**
23:16	R	**CRT green signature read back bit[7:0]**
31:24	R	**CRT red signature read back bit[7:0]**

Offset: 6Ch **GFX06C: CRT Misc Setting Register** **Init = 00000000h**

Bit	R/W	Description
24:0		**Reserved**
25		**EnSwUV**
26		**En709**
27	RW	**AllNoTGen**
29:28	RW	**Analog channel selection bit[1:0] for VGA DAC sense** 0:R 1:G 2:B 3:reg_1v_sync
31:30	RW	**Mode selection bit[1:0] for VGA DAC sense** 0:1st line rising 1:2nd line rising 2:1st line falling 3:2nd line falling

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		**Reserved**
28:16	RW	**CRT horizontal display enable end bit[12:0] (-1)**
31:29		**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		**Reserved**
28:16	RW	**CRT horizontal retrace end bit[12:0] (-1)**
31:29		**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		**Reserved**
27:16	RW	**CRT vertical display enable end bit[11:0] (-1)**
31:28		**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		**Reserved**
27:16	RW	**CRT vertical retrace end bit[11:0] (-1)**
31:28		**Reserved**

Offset: 80h	GFX080: CRT Display Starting Address Register	Init = xxxxxxxxh
Bit	**R/W**	**Description**
0		**Reserved**
1	RW	**EnIntIAg**
2		**Reserved**
3		**Reserved**
29:4	RW	**CRT display starting address bit[29:4]**
31:30		**Reserved**

Offset: 84h	GFX084: CRT Display Offset & Terminal Count Register	Init = 0xxxxxxxh
Bit	**R/W**	**Description**
2:0	RW	**BlkIniV**
3		**Reserved**
15:4	RW	**CRT display offset bit[15:4]**
28:16	RW	**CRT terminal count[12:0] (-0)**
31:29		**Reserved**

Offset: 88h	GFX088: CRT Threshold Register	Init = 00xxxxxxh
Bit	**R/W**	**Description**
5:0	RW	**CRT threshold low bit[5:0]**
7:6		**Reserved**
13:8	RW	**CRT threshold high bit[5:0]**

to next page

from previous page

15:14		**Reserved**
19:16	RW	**CRT X scaling-up factor bit[19:16]**
23:20	RW	**CRT Y scaling-up factor bit[19:16]**
31:24		**Reserved**

Offset: 8Ch **GFX08C: CRT Scaling-Up Factor Register** **Init = xxxxxxxxh**

Bit	R/W	Description
15:0	RW	**CRT X scaling-up factor bit[15:0]**
31:16	RW	**CRT Y scaling-up factor bit[15:0]**

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		**Reserved**
13:8	RW	**CRT hardware cursor Y offset bit[5:0]**
31:14		**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		**Reserved**
27:16	RW	**CRT hardware cursor Y position bit[11:0]** Update cursor after writing this register
31:28		**Reserved**

Offset: 98h **GFX098: CRT Hardware Cursor Pattern Address Register** **Init = xxxxxxxxh**

Bit	R/W	Description
0		**Reserved**
1	RW	**EnIntIAc**
2		**Reserved**
3		**Reserved**
29:4	RW	**CRT hardware cursor pattern address bit[29:4]**
31:30		**Reserved**

Offset: 9Ch **GFX09C: CRT 9C Register** **Init = xxxxxxxxh**

Bit	R/W	Description
7:0	RW	**XPnAddr**
13:8	RW	**XPnSync**
15:14		**Reserved**
23:16		**YPnAddr**
31:24		**Reserved**

Offset: A0h	GFX0A0: CRT OSD Horizontal Start/End Register	Init = xxxxxxxxh
Bit	**R/W**	**Description**
12:0	RW	**CRT OSD horizontal start bit[12:0] (-1)**
15:13		**Reserved**
28:16	RW	**CRT OSD horizontal end bit[12:0] (-0)**
31:29		**Reserved**

Offset: A4h	GFX0A4: CRT OSD Vertical Start/End Register	Init = 0xxx0xxxh
Bit	**R/W**	**Description**
11:0	RW	**CRT OSD vertical start bit[11:0] (-1)**
15:12		**Reserved**
27:16	RW	**CRT OSD vertical end bit[11:0] (-0)**
31:28		**Reserved**

Offset: A8h	GFX0A8: CRT OSD Pattern Address Register	Init = 0xxxxxxxh
Bit	**R/W**	**Description**
0		**Reserved**
1	RW	**EnIntIAo**
2		**Reserved**
3		**Reserved**
29:4	RW	**CRT OSD pattern address bit[29:4]**
31:30		**Reserved**

Offset: ACh	GFX0AC: CRT OSD Offset Register	Init = 0xxx0xxxh
Bit	**R/W**	**Description**
3:0		**Reserved**
14:4	RW	**CRT OSD offset bit[14:4]**
15		**Reserved**
27:16	RW	**CRT OSD terminal count[11:0] (-0)**
31:28		**Reserved**

Offset: B0h	GFX0B0: CRT OSD Threshold & Alpha Register	Init = 000x0x0xh
Bit	**R/W**	**Description**
4:0	RW	**CRT OSD threshold low bit[4:0]**
7:5		**Reserved**
12:8	RW	**CRT OSD threshold high bit[4:0]**
15:13		**Reserved**
20:16	RW	**CRT OSD alpha bit[4:0]**
23:21		**Reserved**
24		**EnOSD16b**
25		**EnOSDTrp**
31:26		**Reserved**

Offset: B4h	GFX0B4: CRT B4 Register	Init = xxxxxxxxh
Bit	**R/W**	**Description**
15:0	RW	**OSDTrpDat**
31:16		**Reserved**

Offset: B8h	GFX0B8: CRT Status Register V	Init = 00000xxxh
Bit	**R/W**	**Description**
11:0	R	**VCnt**
31:12		**Reserved**

Offset: BCh	GFX0BC: Scratchpad Register	Init = 00000000h
Bit	**R/W**	**Description**
31:0	RW	**Scratchpad bit[31:0]**

Offset: D0h	GFX0D0: CRT Display BB0 Starting Address Register	Init = xxxxxxxxh
Bit	**R/W**	**Description**
3:0	R	**Reserved**
29:4	RW	**BB0StAdr bit[29:4]**
31:30	R	**Reserved**

Offset: D4h	GFX0D4: CRT Display BB1 Starting Address Register	Init = xxxxxxxxh
Bit	**R/W**	**Description**
3:0	R	**Reserved**
29:4	RW	**BB1StAdr bit[29:4]**
31:30	R	**Reserved**

Offset: D8h	GFX0D8: CRT Display BB Terminal Count Register	Init = xxxxxxxxh
Bit	**R/W**	**Description**
12:0	RW	**BB0TermCnt bit[12:0]**
15:13	R	**Reserved**
28:16	RW	**BB1TermCnt bit[12:0]**
31:29	R	**Reserved**

Offset: E0h	GFX0E0: OSD Color Palette Index 1 & 0 Register	Init = xxxxxxxxh
Bit	**R/W**	**Description**
15:0	RW	**OSD color palette index 0 bit[15:0] (XRGB1555)**
31:16	RW	**OSD color palette index 1 bit[15:0] (XRGB1555)**

Offset: E4h	GFX0E4: OSD Color Palette Index 3 & 2 Register	Init = xxxxxxxxh
Bit	**R/W**	**Description**
15:0	RW	**OSD color palette index 2 bit[15:0] (XRGB1555)**
31:16	RW	**OSD color palette index 3 bit[15:0] (XRGB1555)**

Offset: E8h	GFX0E8: OSD Color Palette Index 5 & 4 Register	Init = xxxxxxxxh
Bit	**R/W**	**Description**
15:0	RW	**OSD color palette index 4 bit[15:0] (XRGB1555)**
31:16	RW	**OSD color palette index 5 bit[15:0] (XRGB1555)**

Offset: ECh	GFX0EC: OSD Color Palette Index 7 & 6 Register	Init = xxxxxxxxh
Bit	**R/W**	**Description**
15:0	RW	**OSD color palette index 6 bit[15:0] (XRGB1555)**
31:16	RW	**OSD color palette index 7 bit[15:0] (XRGB1555)**

Offset: F0h	GFX0F0: OSD Color Palette Index 9 & 8 Register	Init = xxxxxxxxh
Bit	**R/W**	**Description**
15:0	RW	**OSD color palette index 8 bit[15:0] (XRGB1555)**
31:16	RW	**OSD color palette index 9 bit[15:0] (XRGB1555)**

Offset: F4h	GFX0F4: OSD Color Palette Index 11 & 10 Register	Init = xxxxxxxxh
Bit	**R/W**	**Description**
15:0	RW	**OSD color palette index 10 bit[15:0] (XRGB1555)**
31:16	RW	**OSD color palette index 11 bit[15:0] (XRGB1555)**

Offset: F8h	GFX0F8: OSD Color Palette Index 13 & 12 Register	Init = xxxxxxxxh
Bit	**R/W**	**Description**
15:0	RW	**OSD color palette index 12 bit[15:0] (XRGB1555)**
31:16	RW	**OSD color palette index 13 bit[15:0] (XRGB1555)**

Offset: FCh	GFX0FC: OSD Color Palette Index 15 & 14 Register	Init = xxxxxxxxh
Bit	**R/W**	**Description**
15:0	RW	**OSD color palette index 14 bit[15:0] (XRGB1555)**
31:16	RW	**OSD color palette index 15 bit[15:0] (XRGB1555)**

27 X-DMA Controller

27.1 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 Quad-word End Pointer
XDMA08: Host Command Queue Quad-word Write Pointer
XDMA0C: Host Command Queue Quad-word Read Pointer
XDMA10: BMC Command Queue Base Address
XDMA14: BMC Command Queue Quad-word End Pointer
XDMA18: BMC Command Queue Quad-word Write Pointer
XDMA1C: BMC Command Queue Quad-word Read Pointer
XDMA20: Interrupt Enable and Engine Control
XDMA24: Interrupt Flag and Engine Status
XDMA28: Down-stream Current Transmitting Frame Size
XDMA30: Probe of Down-stream PCIe
XDMA34: Probe of Up-stream PCIe
XDMA38: In Processing Down-stream Command #1[31:0]
XDMA3C: In Processing Down-stream Command #2[31:0]
XDMA40: In Processing Up-stream Command #0[31:0]
XDMA44: In Processing Up-stream Command #0[63:32]
XDMA48: In Processing Up-stream Command #1[31:0]
XDMA4C: In Processing Up-stream Command #1[63:32]
XDMA50: In Processing Up-stream Command #2[31:0]
XDMA54: In Processing Up-stream Command #2[63:32]
XDMA60: Host Command Queue Base Address [63:32]
XDMA64: VGA Command Queue Base Address [31:3]
XDMA68: VGA Command Queue Quad-word End Pointer
XDMA6C: VGA Command Queue Quad-word Write Pointer
XDMA70: VGA Command Queue Quad-word Read Pointer
XDMA74: VGA Command Engine Status
XDMA78: VGA Command Queue Base Address [63:32]

27.2 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.

27.3 Registers

Offset: 00h	XDMA00: Host Command Queue Base Address [31:3]	Init = 0
Bit	**R/W**	**Description**
31:3	RW	**Base address of host command queue [31:3].**
2 :0		**Reserved (0)**

Offset: 04h	XDMA04: Host Command Queue Quad-word End Pointer	Init = 0
Bit	**R/W**	**Description**
31:18		**Reserved (0)**
17:0	RW	**Quad-word end pointer of host command queue [17:0].**

Offset: 08h	XDMA08: Host Command Queue Quad-word Write Pointer	Init = 0
Bit	**R/W**	**Description**
31:18		**Reserved (0)**
17:0	RW	**Quad-word write pointer of host command queue [17:0].**

Offset: 0Ch	XDMA0C: Host Command Queue Quad-word Read Pointer	Init = 0
Bit	**R/W**	**Description**
31	R	**Valid for setting read/write pointer of host command queue**
30:18		**Reserved (0)**
17:0	RW	**Quad-word read pointer of host command queue [17:0].**

Offset: 10h	XDMA10: BMC Command Queue Base Address	Init = 0
Bit	**R/W**	**Description**
31:30		**Reserved (0)**
29:4	RW	**Base address of BMC command queue [29:4].**
2 :0		**Reserved (0)**

Offset: 14h	XDMA14: BMC Command Queue Quad-word End Pointer	Init = 0
Bit	**R/W**	**Description**
31:18		**Reserved (0)**
17:0	RW	**Quad-word end pointer of BMC command queue [17:0].**

Offset: 18h	XDMA18: BMC Command Queue Quad-word Write Pointer	Init = 0
Bit	**R/W**	**Description**
31:18		**Reserved (0)**
17:0	RW	**Quad-word write pointer of BMC command queue [17:0].**

Offset: 1Ch	XDMA1C: BMC Command Queue Quad-word Read Pointer	Init = X
Bit	**R/W**	**Description**
31	R	**Valid for setting read/write pointer of BMC command queue**
30:18		**Reserved (0)**
17:0	RW	**Quad-word read pointer of BMC command queue [17:0].**

Offset: 20h	XDMA20: Interrupt Enable and Engine Control	Init = 0
Bit	**R/W**	**Description**
31	RW	**Enable PCIe 64-bits Address Mode.**
30		**Reserved (0)**
29	RW	**Enable Check Down-stream Command ID.**
28	RW	**Enable Down-stream Data Buffer Time Out.**
27:26	RW	**Down-stream Data Buffer Threshold[1:0].**
25:24	RW	**Down-stream Out Standing Header Selection[1:0].**
23:21	RW	**Down-stream Out Standing Data Selection[2:0].**
20		**Reserved (0)**
19:17	RW	**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.
16:14	RW	**Down-stream Write Request Delay Timer[2:0].**
13	RW	**Interrupt type.** 0: Level-trigger. 1: Edge-trigger.
12	RW	**Clear Host Dirty Frame (XDMA24[26]) or Down-stream Frame End (XDMA24[25]) Control.** 0: Write XDMA24[3] or XDMA24[2] with 1 to clear. 1: Write XDMA24[26] or XDMA24[25] with 1 to clear.
11:7		**Reserved (0)**
6	RW	**Enable down-stream dirty frame INT.** **Only access by BMC**
5	RW	**Enable down-stream complete INT.** **Only access by BMC**
4	RW	**Enable up-stream complete INT.** **Only access by BMC**
3	RW	**Enable host command queue dirty frame INT.** **Only access by HOST**
2	RW	**Enable down-stream dirty frame INT.** **Only access by HOST**
1	RW	**Enable down-stream complete INT.** **Only access by HOST**
0	RW	**Enable up-stream complete INT.** **Only access by HOST**

Offset: 24h	XDMA24: Interrupt Flag and Engine Status	Init = X
Bit	**R/W**	**Description**
31	R	**XDMA all engine IDLE.**
30	R	**Host command queue IDLE.**
29	R	**BMC command queue IDLE.**
28	R	**Down-stream engine IDLE.**

to next page

from previous page

27	R	**Up-steram engine IDLE.**
26	RW	**Host command queue in Dirty Frame.**
25	RW	**Down-stream in Dirty Frame.**
24:7		**Reserved (0)**
6	R	**Down-stream Dirty Frame Flag.**
5	R	**Down-stream Command Finished Flag.**
4	R	**Up-stream Command Finished Flag.**
3	RW	**Host Command Queue Dirty Frame Flag.**
2	RW	**Down-stream Dirty Frame Flag.**
1	R	**Down-stream Command Finished Flag.**
0	R	**Up-stream Command Finished Flag.**

Offset: 24h **XDMA24: Interrupt Flag and Engine Status** **Init = X**

Bit	R/W	Description
31	R	**XDMA all engine IDLE.**
30	R	**Host command queue IDLE.**
29	R	**BMC command queue IDLE.**
28	R	**Down-stream engine IDLE.**
27	R	**Up-steram engine IDLE.**
26	R	**Host command queue in Dirty Frame.**
25	RW	**Down-stream in Dirty Frame.**
24:7		**Reserved (0)**
6	RW	**Down-stream Dirty Frame Flag.**
5	R	**Down-stream Command Finished Flag.**
4	R	**Up-stream Command Finished Flag.**
3	R	**Host Command Queue Dirty Frame Flag.**
2	R	**Down-stream Dirty Frame Flag.**
1	R	**Down-stream Command Finished Flag.**
0	R	**Up-stream Command Finished Flag.**

Offset: 28h **XDMA28: Down-stream Current Transmitting Frame Size** **Init = X**

Bit	R/W	Description
30:15		**Reserved (0)**
14:3	R	**Down-stream current transmitting line byte size[14:3].** Value 0h is 0 byte, and value 1h is 8 bytes.
2 :0		**Reserved (0)**
Note : When (XDMA28[14:3] == 0), Frame is (((XDMA28[27:16]-1) * XDMA3C[14:3]) * 8) bytes. When (XDMA28[14:3] != 0), Frame is ((XDMA28[27:16] * XDMA3C[14:3] + (XDMA28[14:3]-1)) * 8) bytes.		

Offset: 30h **XDMA30: Probe of Down-stream PCIe** **Init = X**

Bit	R/W	Description
31:0	R	**Probe of down-stream pcie[31:0].**

Offset: 34h	XDMA34: Probe of Up-stream PCIe	Init = X
Bit	**R/W**	**Description**
31:0	R	**Probe of up-stream pcie[31:0].**

Offset: 38h	XDMA38: In Processing Down-stream Command #1[31:0]	Init = X
Bit	**R/W**	**Description**
31	R	**Down-stream Frame Start.**
30:28		**Reserved (0)**
27:3	R	**Base address of MBus data[27:3].**
2 :0		**Reserved (0)**

Offset: 3Ch	XDMA3C: In Processing Down-stream Command #2[31:0]	Init = X
Bit	**R/W**	**Description**
31	R	**Enable Interrupt at Command Finished.**
30:28		**Reserved (0)**
27:16	R	**Frame line number[11:0].** Value 0h is 1 line, and value 1h is 2 lines.
15	R	**Interrupt Direction** 0: Interrupt to Host. 1: Interrupt to BMC.
14:3	R	**Frame line byte size[14:3].** Value 0h is 0 byte, and value 1h is 8 bytes.
2 :0		**Reserved (0)**

Offset: 40h	XDMA40: In Processing Up-stream Command #0[31:0]	Init = X
Bit	**R/W**	**Description**
31:3	R	**Base address of PCIe data[31:3].**
2 :0		**Reserved (0)**

Offset: 44h	XDMA44: In Processing Up-stream Command #0[63:32]	Init = X
Bit	**R/W**	**Description**
31:0	R	**Base address of PCIe data[63:32].**

Offset: 48h	XDMA48: In Processing Up-stream Command #1[31:0]	Init = X
Bit	**R/W**	**Description**
31	R	**Down-stream Frame Start.**
30:28		**Reserved (0)**
27:3	R	**Base address of MBus data[27:3].**
2 :0		**Reserved (0)**

Offset: 4Ch	XDMA4C: In Processing Up-stream Command #1[63:32]	Init = X
Bit	**R/W**	**Description**
31		**Reserved (0)**
30:19	R	**Pitch of MBus data[14:3].**

to next page

from previous page

18:15		**Reserved (0)**
14:3	R	**Pitch of PCIe data[14:3].**
2 :0		**Reserved (0)**

Offset: 50h	**XDMA50: In Processing Up-stream Command #2[31:0]**	**Init = X**
Bit	**R/W**	**Description**
31	R	**Enable Interrupt at Command Finished.**
30:28		**Reserved (0)**
27:16	R	**Frame line number[11:0].** Value 0h is 1 line, and value 1h is 2 lines.
15	R	**Interrupt Direction** 0: Interrupt to Host. 1: Interrupt to BMC.
14:3	R	**Frame line byte size[14:3].** Value 0h is 0 byte, and value 1h is 8 bytes.
2 :0		**Reserved (0)**

Offset: 54h	**XDMA54: In Processing Up-stream Command #2[63:32]**	**Init = X**
Bit	**R/W**	**Description**
31:16		**Reserved (0)**
15:0	R	**Up-stream Tag[15:0].**

Offset: 60h	**XDMA60: Host Command Queue Base Address [63:32]**	**Init = X**
Bit	**R/W**	**Description**
31:0	RW	**Base address of host command queue [63:32].**

Offset: 64h	**XDMA64: VGA Command Queue Base Address [31:3]**	**Init = 0**
Bit	**R/W**	**Description**
31:3	RW	**Base address of vga command queue [31:3].**
2 :0		**Reserved (0)**

Offset: 68h	**XDMA68: VGA Command Queue Quad-word End Pointer**	**Init = 0**
Bit	**R/W**	**Description**
31:18		**Reserved (0)**
17:0	RW	**Quad-word end pointer of vga command queue [17:0].**

Offset: 6Ch	**XDMA6C: VGA Command Queue Quad-word Write Pointer**	**Init = 0**
Bit	**R/W**	**Description**
31:18		**Reserved (0)**
17:0	RW	**Quad-word write pointer of vga command queue [17:0].**

Offset: 70h	XDMA70: VGA Command Queue Quad-word Read Pointer	Init = 0
Bit	**R/W**	**Description**
31	R	**Valid for setting read/write pointer of vga command queue**
30:18		**Reserved (0)**
17:0	RW	**Quad-word read pointer of vga command queue [17:0].**

Offset: 74h	XDMA74: VGA Command Engine Status	Init = X
Bit	**R/W**	**Description**
31	R	**XDMA all engine IDLE.**
30	R	**VGA command queue IDLE.**
29:28		**Reserved (1)**
27	R	**Up-steram engine IDLE.**
26	RW	**VGA command queue in Dirty Frame.**
25:0		**Reserved (0)**

Offset: 78h	XDMA78: VGA Command Queue Base Address [63:32]	Init = X
Bit	**R/W**	**Description**
31:0	RW	**Base address of vga command queue [63:32].**

27.4 Command Format

Offset: 00h	Comm#0: PCIe Data Base Address	Init = X
Bit	**R/W**	**Description**
63:3	RW	**Base address of PCIe data[63:3].**
2 :0		**0h (Command ID)**

Offset: 08h	Comm#1: Pitch and Base Address	Init = X
Bit	**R/W**	**Description**
63		**Reserved (0)**
62:51	RW	**Pitch of MBus data[14:3].**
50:47		**Reserved (0)**
46:35	RW	**Pitch of PCIe data[14:3].**
34:32		**Reserved (0)**
31	RW	**Command Direction** 0: Down-stream. 1: Up-stream.
30		**Reserved (0)**
29:4	RW	**Base address of MBus data[29:4].**
3		**Reserved (0)**
2 :0		**1h (Command ID)**

Offset: 10h	Comm#2: Fire Command	Init = X
Bit	**R/W**	**Description**
63:50		**Reserved (0)**

to next page

from previous page

49:48	RW	**Up Command Direction.** 2: VGA 1: Host 0: BMC
47:32	RW	**Up-stream Tag[15:0].**
31	RW	**Enable Interrupt at Command Finished.**
30	RW	**XDMA in video engine interface.** 0: DRAM interface. 1: Video engine interface.
29	RW	**Enable down-stream horizontal synchronization.**
28	RW	**Enable down-stream vertical synchronization.**
27:16	RW	**Frame line number[11:0].** Value 1h is 1 line, and value 2h is 2 lines.
15	RW	**Interrupt Direction** 0: Interrupt to Host. 1: Interrupt to BMC.
14:4	RW	**Frame line byte size[14:4].** Value 0h is 0 byte, and value 1h is 16 bytes.
3		**Reserved (0)**
2 :0		**2h (Command ID)**

Offset: 18h	**Comm#3: Reserved Command**	**Init = X**
Bit	**R/W**	**Description**
63: 0		**Reserved (0)**

27.5 Clearing and Setting Procedure

27.5.1 Host Command Queue in Dirty Frame Clearing Procedure

When XDMA20[12] == 0, Write XDMA24[3] with 1 to clear down-stream in dirty frame.
When XDMA20[12] == 1, Write XDMA24[26] with 1 to clear down-stream in dirty frame.

27.5.2 Down-stream in Dirty Frame Clearing Procedure

For Host,
when XDMA20[12] == 0, Write XDMA24[2] with 1 to clear up-stream in frame end.
when XDMA20[12] == 1, Write XDMA24[25] with 1 to clear up-stream in frame end.
For BMC,
when XDMA20[12] == 0, Write XDMA24[6] with 1 to clear up-stream in frame end.
when XDMA20[12] == 1, Write XDMA24[25] with 1 to clear up-stream in frame end.

27.5.3 Host Command Queue Read/Write Pointer Setting Procedure

1. Write XDMA0C with password.
2. Write XDMA08 with pointer value.

27.5.4 BMC Command Queue Read/Write Pointer Setting Procedure

1. Write XDMA1C with password.
2. Write XDMA18 with pointer value.

27.5.5 VGA Command Queue Read/Write Pointer Setting Procedure

1. Write XDMA70 with password.
2. Write XDMA6C with pointer value.

28 MCTP Controller

28.1 Overview

MCTP Engine receives Msg which carries MCTP packet through PCIe from Host or sends it to Host. Support transmission unit is baseline transmission unit(64 bytes).

MCTP00: Engine Status and Engine Control.
MCTP04: TX Command Address.
MCTP08: RX Command Address.
MCTP0C: Interrupt Status
MCTP10: Interrupt Enable
MCTP14: MCTP EID
MCTP18: OBFF Control

28.2 Features

- Descriptor type DMA engine.
- Maximum packet payload length is 64 bytes (Baseline transmission unit)
- Flexible for receiving all kinds of MsgD or MCTP only.
- Option of matching EID.
- Interrupt for receiving or sending complete

28.3 Registers : Base Address = 0x1E6E_8000

Offset: 00h	MCTP00: Engine Status and Engine Control	Init = 0x00330000
Bit	**R/W**	**Description**
31:24	R	**Current command count**
23:22		**Reserved**
21	R	**RX PCIe idle**
20	R	**RX DMA idle**
19:18		**Reserved**
17	R	**TX PCIe idle**
16	R	**TX DMA idle**
15	RW	**Enable CPL 2 MCTP**
9	RW	**Matching MCTP EID** Only valid with MCTP00[8] = 1 1: Only accept MCTP EID matches MCTP14.
8	RW	**Msg Mask** 1: MCTP type only 0: Accept all MsgD
7: 5		**Reserved**
4	RW	**RX Command is ready.** This bit will be cleared when RX last command is used.
3: 0		**Reserved**
0	RW	**Trigger TX operation.**

Offset: 04h	MCTP04: TX Command Address	Init = 0
Bit	**R/W**	**Description**
31:29		**Reserved**
28: 3	RW	**TX command address**
2: 0		**Reserved**

Offset: 08h	MCTP08: RX Command Address	Init = 0
Bit	**R/W**	**Description**
31:29		**Reserved**
28: 3	RW	**RX command address**
2: 0		**Reserved**

Offset: 0Ch	MCTP0C: Interrupt Status	Init = 0
Bit	**R/W**	**Description**
31:28		**Reserved**
27	R	**MSG obff state changes.**
26	R	**MSG obff in active state.**
25	R	**MSG obff in idle state.**
24	R	**MSG obff in obff state.**
23:20		**Reserved**
19	R	**Wake obff in active state.**
18	R	**Wake obff in post-obff state.**
17	R	**Wake obff in obff state.**
16	R	**Wake obff in idle state.**
15:10		**Reserved**
9	R	**No more RX command.**
8	R	**RX receiving complete**
7: 2		**Reserved**
1	R	**Set when CMD1[15] is set**
0	R	**TX sending complete**

Offset: 10h	MCTP10: Interrupt Enable	Init = 0
Bit	**R/W**	**Description**
31:28		**Reserved**
27	R	**Enable interrupt of MSG obff state changes.**
26	R	**Enable interrupt of MSG obff in active state.**
25	R	**Enable interrupt of MSG obff in idle state.**
24	R	**Enable interrupt of MSG obff in obff state.**
23:20		**Reserved**
19	R	**Enable interrupt of Wake obff in active state.**
18	R	**Enable interrupt of Wake obff in post-obff state.**
17	R	**Enable interrupt of Wake obff in obff state.**
16	R	**Enable interrupt of Wake obff in idle state.**

to next page

from previous page

15:10		**Reserved**
9	R	**Enable interrupt of No more RX command.**
8	R	**Enable interrupt of RX receiving complete**
7: 2		**Reserved**
1	R	**Enalbe interrupt of TX last command is sent**
0	R	**Enalbe interrupt of TX sending complete**

Offset: 14h	**MCTP14: MCTP EID**	**Init = 0**
Bit	**R/W**	**Description**
31:28	RW	**Memory space mapping**
27: 8		**Reserved**
7: 0	RW	**MCTP EID** Valid when MCTP00[9:8] = 2'b11 Compare with EID of incoming packet.

Offset: 18h	**MCTP18: OBFF Control**	**Init = 0**
Bit	**R/W**	**Description**
29:16	R	**WAKE OBFF status**
15		**Reserved**
14	R	**MSG OBFF is in active state**
13	R	**MSG OBFF is in idle state**
12	R	**MSG OBFF is in obff state**
11: 6		**Reserved**
5: 0	RW	**Wake OBFF control**

28.4 Command

28.4.1 TX command

Offset: 00h	**Commnand 0**	**Init = 0**
Bit	**R/W**	**Description**
63		**Last command**
62:40		**Data address[29:7]**
39:32		**Destination Endpoint ID**
31:24		**Destination PCIe bus number**
23:19		**Destination PCIe device number**
18:16		**Destination PCIe function number**
15		**Enable Interrupt**
14		**PCIe rounting type** 0: Route to Root Complex 1: Route by ID
13		**MCTP tag owner**
12: 2		**Packet size in 4bytes**
1: 0		**Padding length**

28.4.2 RX command

Offset: 00h		Commnand 0	Init = 0
Bit	**R/W**	**Description**	
63		**Last command**	
62		**Reserved**	
61:39		**Data address[29:7]**	
38:31		**Reserved**	
30:24		**Packet length in 4bytes**	
23:16		**Source Endpoint ID**	
15:14		**PCIe rounting type**	
13		**MCTP tag owner**	
12:11		**MCTP sequence number**	
10: 8		**MCTP tag**	
7		**MCTP start of message**	
6		**MCTP end of message**	
5: 1		**Reserved**	
0		**Command updated** 1: Command[30:6] are updated. When new command is writing to this address, this bit should be cleared by FW.	

28.5 Operation

28.5.1 Send Packet

1. Set proper address to MCTP04.
2. Prepare command to address set to MCTP04.
3. Set MCTP00[0] to 1 to trigger TX.
4. Waiting transaction complete.
   - MCTP0C[0] indicates one command is completed.
   - MCTP0C[1] indicates the last command is completed.

28.5.2 Receive Packet

1. Set proper address to MCTP08.
2. Prepare command to address set to MCTP08.
3. Set MCTP00[4] to enable receiving.
4. Waiting transaction complete.
   - MCTP0C[8] indicates one command is completed.
   - MCTP0C[9] indicates the last command is completed.
5. Corresponding command is updated with information of MCTP header.
6. Read data in memory.

29 ADC Controller

29.1 Overview

ADC Engine has 16 voltage sensing channels. 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.

ADC00: Engine Control.
ADC04: Interrupt Enable and Interrupt Status.
ADC08: ADC VGA Detect Control.
ADC0C: ADC Clock Control.
ADC10: Data of Channel 1 and 0.
ADC14: Data of Channel 3 and 2.
ADC18: Data of Channel 5 and 4.
ADC1C: Data of Channel 7 and 6.
ADC20: Data of Channel 9 and 8.
ADC24: Data of Channel 11 and 10.
ADC28: Data of Channel 13 and 12.
ADC2C: Data of Channel 15 and 14. .
ADC30: Upper and Lower bound 0 of Channel 0.
ADC34: Upper and Lower bound 0 of Channel 1.
ADC38: Upper and Lower bound 0 of Channel 2.
ADC3C: Upper and Lower bound 0 of Channel 3.
ADC40: Upper and Lower bound 0 of Channel 4.
ADC44: Upper and Lower bound 0 of Channel 5.
ADC48: Upper and Lower bound 0 of Channel 6.
ADC4C: Upper and Lower bound 0 of Channel 7.
ADC50: Upper and Lower bound 0 of Channel 8.
ADC54: Upper and Lower bound 0 of Channel 9.
ADC58: Upper and Lower bound 0 of Channel 10.
ADC5C: Upper and Lower bound 0 of Channel 11.
ADC60: Upper and Lower bound 0 of Channel 12.
ADC64: Upper and Lower bound 0 of Channel 13.
ADC68: Upper and Lower bound 0 of Channel 14.
ADC6C: Upper and Lower bound 0 of Channel 15.
ADC70: Hysteresis Control and bound of Channel 0.
ADC74: Hysteresis Control and bound of Channel 1.
ADC78: Hysteresis Control and bound of Channel 2.
ADC7C: Hysteresis Control and bound of Channel 3.
ADC80: Hysteresis Control and bound of Channel 4.
ADC84: Hysteresis Control and bound of Channel 5.
ADC88: Hysteresis Control and bound of Channel 6.
ADC8C: Hysteresis Control and bound of Channel 7.
ADC90: Hysteresis Control and bound of Channel 8.
ADC94: Hysteresis Control and bound of Channel 9.
ADC98: Hysteresis Control and bound of Channel 10.
ADC9C: Hysteresis Control and bound of Channel 11.
ADCA0: Hysteresis Control and bound of Channel 12.
ADCA4: Hysteresis Control and bound of Channel 13.
ADCA8: Hysteresis Control and bound of Channel 14.
ADCAC: Hysteresis Control and bound of Channel 15.
ADCC0: Interrupt Source Selection.
ADCC4: Compensating and Trimming.

29.2 Features

- 10-bits resolution for 16 voltage channels.
- Channel scanning can be non-continuous.
- 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.

29.3 Registers : Base Address = 0x1E6E_9000

Offset: 00h	ADC00: Engine Control	Init = 0
Bit	**R/W**	**Description**
31:16	RW	**Channel enable.** 1: Enable 0: Skip ADC00[31] for channel 15. ADC00[30] for channel 14. ADC00[29] for channel 13. ADC00[28] for channel 12. ADC00[27] for channel 11. ADC00[26] for channel 10. ADC00[25] for channel 9. ADC00[24] for channel 8. 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: 9		**Reserved**
8	R	**Initial sequence complete**
7: 6		**Reserved**
5	RW	**Auto Compensating sensing mode.** 1: Trigger compensating method. 0: Compensating method is done.
4	RW	**Compensating sensing mode.**
3: 1	RW	**ADC Operation mode.** 000b: Power down mode. 001b: Standby mode. 111b: Normal mode.
0	RW	**Engine enable.**

Offset: 04h	ADC04: Interrupt Enable and Interrupt Status	Init = 0
Bit	**R/W**	**Description**
31:16	RW	**Interrupt enable** 1: Enable 0: Disable ADC04[31] for channel 13. ADC04[30] for channel 12. ADC04[29] for channel 13. ADC04[28] for channel 12. ADC04[27] for channel 11. ADC04[26] for channel 10. ADC04[25] for channel 9. ADC04[24] for channel 8. 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: 0	RW	**Interrupt status** Write 1 to clear ADC04[15] for channel 15. ADC04[14] for channel 14. ADC04[13] for channel 13. ADC04[12] for channel 12. ADC04[11] for channel 11. ADC04[10] for channel 10. ADC04[09] for channel 9. ADC04[08] for channel 8. 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: 08h	ADC08: ADC VGA Detect Control	Init = 0x0000_000F
Bit	**R/W**	**Description**
31:17		**Reserved**
16	RW	**ADC VGA detect enable**
15:10		**Reserved**
9: 0	RW	**Divisor of ADC clock used for VGA detection** Period ADC clock = period of PCLK * 2 * (ADC08[9:0] + 1)

Offset: 0Ch	ADC0C: ADC Clock Control	Init = 0x0000_000F
Bit	**R/W**	**Description**
31:17	RW	**Pre-Divisor of ADC clock**

to next page

from previous page

16		**Reserved**
15:10		**Reserved**
9: 0	RW	**Divisor of ADC clock** Period of ADC clock = period of PCLK * 2 * (ADC0C[31:17] + 1) * (ADC0C[9:0] + 1) Less than 6MHz is recommand. Sample rate is Period of ADC clock * 12

Offset: 10h **ADC10: Data of Channel 1 and 0** **Init = 0**

Bit	R/W	Description
31:26		**Reserved**
25:16	RW	**Data of Channel 1**
15:10		**Reserved**
9: 0	RW	**Data of Channel 0**

Offset: 14h **ADC14: Data of Channel 3 and 2** **Init = 0**

Bit	R/W	Description
31:26		**Reserved**
25:16	RW	**Data of Channel 3**
15:10		**Reserved**
9: 0	RW	**Data of Channel 2**

Offset: 18h **ADC18: Data of Channel 5 and 4** **Init = 0**

Bit	R/W	Description
31:26		**Reserved**
25:16	RW	**Data of Channel 5**
15:10		**Reserved**
9: 0	RW	**Data of Channel 4**

Offset: 1Ch **ADC1C: Data of Channel 7 and 6** **Init = 0**

Bit	R/W	Description
31:26		**Reserved**
25:16	RW	**Data of Channel 7**
15:10		**Reserved**
9: 0	RW	**Data of Channel 6**

Offset: 20h **ADC20: Data of Channel 9 and 8** **Init = 0**

Bit	R/W	Description
31:26		**Reserved**
25:16	RW	**Data of Channel 9**
15:10		**Reserved**
9: 0	RW	**Data of Channel 8**

Offset: 24h	ADC24: Data of Channel 11 and 10	Init = 0
Bit	**R/W**	**Description**
31:26		**Reserved**
25:16	RW	**Data of Channel 11**
15:10		**Reserved**
9: 0	RW	**Data of Channel 10**

Offset: 28h	ADC28: Data of Channel 13 and 12	Init = 0
Bit	**R/W**	**Description**
31:26		**Reserved**
25:16	RW	**Data of Channel 13**
15:10		**Reserved**
9: 0	RW	**Data of Channel 12**

Offset: 2Ch	ADC2C: Data of Channel 15 and 14	Init = 0
Bit	**R/W**	**Description**
31:26		**Reserved**
25:16	RW	**Data of Channel 15**
15:10		**Reserved**
9: 0	RW	**Data of Channel 14**

Offset: 30h	ADC30: Upper and Lower bound of Channel 0	Init = 0
Bit	**R/W**	**Description**
31:26		**Reserved**
25:16	RW	**Upper bound**
15:10		**Reserved**
9: 0	RW	**Lower bound**

Offset: 34h	ADC34: Upper and Lower bound of Channel 1	Init = 0
Bit	**R/W**	**Description**
31:26		**Reserved**
25:16	RW	**Upper bound**
15:10		**Reserved**
9: 0	RW	**Lower bound**

Offset: 38h	ADC38: Upper and Lower bound of Channel 2	Init = 0
Bit	**R/W**	**Description**
31:26		**Reserved**
25:16	RW	**Upper bound**
15:10		**Reserved**
9: 0	RW	**Lower bound**

Offset: 3Ch	ADC3C: Upper and Lower bound of Channel 3	Init = 0
Bit	**R/W**	**Description**
31:26		**Reserved**
25:16	RW	**Upper bound**
15:10		**Reserved**
9: 0	RW	**Lower bound**

Offset: 40h	ADC40: Upper and Lower bound of Channel 4	Init = 0
Bit	**R/W**	**Description**
31:26		**Reserved**
25:16	RW	**Upper bound**
15:10		**Reserved**
9: 0	RW	**Lower bound**

Offset: 44h	ADC44: Upper and Lower bound of Channel 5	Init = 0
Bit	**R/W**	**Description**
31:26		**Reserved**
25:16	RW	**Upper bound**
15:10		**Reserved**
9: 0	RW	**Lower bound**

Offset: 48h	ADC48: Upper and Lower bound of Channel 6	Init = 0
Bit	**R/W**	**Description**
31:26		**Reserved**
25:16	RW	**Upper bound**
15:10		**Reserved**
9: 0	RW	**Lower bound**

Offset: 4Ch	ADC4C: Upper and Lower bound of Channel 7	Init = 0
Bit	**R/W**	**Description**
31:26		**Reserved**
25:16	RW	**Upper bound**
15:10		**Reserved**
9: 0	RW	**Lower bound**

Offset: 50h	ADC50: Upper and Lower bound of Channel 8	Init = 0
Bit	**R/W**	**Description**
31:26		**Reserved**
25:16	RW	**Upper bound**
15:10		**Reserved**
9: 0	RW	**Lower bound**

Offset: 54h	ADC54: Upper and Lower bound of Channel 9	Init = 0
Bit	**R/W**	**Description**
31:26		**Reserved**
25:16	RW	**Upper bound**
15:10		**Reserved**
9: 0	RW	**Lower bound**

Offset: 58h	ADC58: Upper and Lower bound of Channel 10	Init = 0
Bit	**R/W**	**Description**
31:26		**Reserved**
25:16	RW	**Upper bound**
15:10		**Reserved**
9: 0	RW	**Lower bound**

Offset: 5Ch	ADC5C: Upper and Lower bound of Channel 11	Init = 0
Bit	**R/W**	**Description**
31:26		**Reserved**
25:16	RW	**Upper bound**
15:10		**Reserved**
9: 0	RW	**Lower bound**

Offset: 60h	ADC60: Upper and Lower bound of Channel 12	Init = 0
Bit	**R/W**	**Description**
31:26		**Reserved**
25:16	RW	**Upper bound**
15:10		**Reserved**
9: 0	RW	**Lower bound**

Offset: 64h	ADC64: Upper and Lower bound of Channel 13	Init = 0
Bit	**R/W**	**Description**
31:26		**Reserved**
25:16	RW	**Upper bound**
15:10		**Reserved**
9: 0	RW	**Lower bound**

Offset: 68h	ADC68: Upper and Lower bound of Channel 14	Init = 0
Bit	**R/W**	**Description**
31:26		**Reserved**
25:16	RW	**Upper bound**
15:10		**Reserved**
9: 0	RW	**Lower bound**

Offset: 6Ch	ADC6C: Upper and Lower bound of Channel 15	Init = 0
Bit	**R/W**	**Description**
31:26		**Reserved**
25:16	RW	**Upper bound**
15:10		**Reserved**
9: 0	RW	**Lower bound**

Offset: 70h	ADC70: Hysteresis Control and bound of Channel 0	Init = 0
Bit	**R/W**	**Description**
31	RW	**Hysteresis Control** 1: Enable hysteresis. 0: Disable hysteresis.
30:26		**Reserved**
25:16	RW	**Upper bound**
15:10		**Reserved**
9: 0	RW	**Lower bound**

Offset: 74h	ADC74: Hysteresis Control and bound of Channel 1	Init = 0
Bit	**R/W**	**Description**
31	RW	**Hysteresis Control** 1: Enable hysteresis. 0: Disable hysteresis.
30:26		**Reserved**
25:16	RW	**Upper bound**
15:10		**Reserved**
9: 0	RW	**Lower bound**

Offset: 78h	ADC78: Hysteresis Control and bound of Channel 2	Init = 0
Bit	**R/W**	**Description**
31	RW	**Hysteresis Control** 1: Enable hysteresis. 0: Disable hysteresis.
30:26		**Reserved**
25:16	RW	**Upper bound**
15:10		**Reserved**
9: 0	RW	**Lower bound**

Offset: 7Ch	ADC7C: Hysteresis Control and bound of Channel 3	Init = 0
Bit	**R/W**	**Description**
31	RW	**Hysteresis Control** 1: Enable hysteresis. 0: Disable hysteresis.
30:26		**Reserved**
25:16	RW	**Upper bound**
15:10		**Reserved**
9: 0	RW	**Lower bound**

Offset: 80h	ADC80: Hysteresis Control and bound of Channel 4	Init = 0
Bit	**R/W**	**Description**
31	RW	**Hysteresis Control** 1: Enable hysteresis. 0: Disable hysteresis.
30:26		**Reserved**
25:16	RW	**Upper bound**
15:10		**Reserved**
9: 0	RW	**Lower bound**

Offset: 84h	ADC84: Hysteresis Control and bound of Channel 5	Init = 0
Bit	**R/W**	**Description**
31	RW	**Hysteresis Control** 1: Enable hysteresis. 0: Disable hysteresis.
30:26		**Reserved**
25:16	RW	**Upper bound**
15:10		**Reserved**
9: 0	RW	**Lower bound**

Offset: 88h	ADC88: Hysteresis Control and bound of Channel 6	Init = 0
Bit	**R/W**	**Description**
31	RW	**Hysteresis Control** 1: Enable hysteresis. 0: Disable hysteresis.
30:26		**Reserved**
25:16	RW	**Upper bound**
15:10		**Reserved**
9: 0	RW	**Lower bound**

Offset: 8Ch	ADC8C: Hysteresis Control and bound of Channel 7	Init = 0
Bit	**R/W**	**Description**
31	RW	**Hysteresis Control** 1: Enable hysteresis. 0: Disable hysteresis.
30:26		**Reserved**
25:16	RW	**Upper bound**
15:10		**Reserved**
9: 0	RW	**Lower bound**

Offset: 90h	ADC90: Hysteresis Control and bound of Channel 8	Init = 0
Bit	**R/W**	**Description**
31	RW	**Hysteresis Control** 1: Enable hysteresis. 0: Disable hysteresis.

to next page

from previous page

30:26		**Reserved**
25:16	RW	**Upper bound**
15:10		**Reserved**
9: 0	RW	**Lower bound**

Offset: 94h **ADC94: Hysteresis Control and bound of Channel 9** **Init = 0**

Bit	R/W	Description
31	RW	**Hysteresis Control** 1: Enable hysteresis. 0: Disable hysteresis.
30:26		**Reserved**
25:16	RW	**Upper bound**
15:10		**Reserved**
9: 0	RW	**Lower bound**

Offset: 98h **ADC98: Hysteresis Control and bound of Channel 10** **Init = 0**

Bit	R/W	Description
31	RW	**Hysteresis Control** 1: Enable hysteresis. 0: Disable hysteresis.
30:26		**Reserved**
25:16	RW	**Upper bound**
15:10		**Reserved**
9: 0	RW	**Lower bound**

Offset: 9Ch **ADC9C: Hysteresis Control and bound of Channel 11** **Init = 0**

Bit	R/W	Description
31	RW	**Hysteresis Control** 1: Enable hysteresis. 0: Disable hysteresis.
30:26		**Reserved**
25:16	RW	**Upper bound**
15:10		**Reserved**
9: 0	RW	**Lower bound**

Offset: A0h **ADCA0: Hysteresis Control and bound of Channel 12** **Init = 0**

Bit	R/W	Description
31	RW	**Hysteresis Control** 1: Enable hysteresis. 0: Disable hysteresis.
30:26		**Reserved**
25:16	RW	**Upper bound**
15:10		**Reserved**
9: 0	RW	**Lower bound**

Offset: A4h	ADCA4: Hysteresis Control and bound of Channel 13	Init = 0
Bit	**R/W**	**Description**
31	RW	**Hysteresis Control** 1: Enable hysteresis. 0: Disable hysteresis.
30:26		**Reserved**
25:16	RW	**Upper bound**
15:10		**Reserved**
9: 0	RW	**Lower bound**

Offset: A8h	ADCA8: Hysteresis Control and bound of Channel 14	Init = 0
Bit	**R/W**	**Description**
31	RW	**Hysteresis Control** 1: Enable hysteresis. 0: Disable hysteresis.
30:26		**Reserved**
25:16	RW	**Upper bound**
15:10		**Reserved**
9: 0	RW	**Lower bound**

Offset: ACh	ADCAC: Hysteresis Control and bound of Channel 15	Init = 0
Bit	**R/W**	**Description**
31	RW	**Hysteresis Control** 1: Enable hysteresis. 0: Disable hysteresis.
30:26		**Reserved**
25:16	RW	**Upper bound**
15:10		**Reserved**
9: 0	RW	**Lower bound**

Offset: C0h	ADCC0: Interrupt Source	Init = 0
Bit	**R/W**	**Description**
31:16		**Reserved**

to next page

from previous page

15: 0	RW	**Interrupt Source.** 1: ARM 0: Coprocessor ADCC0[15] for channel 15. ADCC0[14] for channel 14. ADCC0[13] for channel 13. ADCC0[12] for channel 12. ADCC0[11] for channel 11. ADCC0[10] for channel 10. ADCC0[9] for channel 9. ADCC0[8] for channel 8. 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: C4h **ADCC4: Compensating and Trimming** **Init = 0**

Bit	R/W	Description
31:26		**Reserved**
25:16	R	**Compensating value.**
15: 4		**Reserved**
3: 0	RW	**Trimming value.**

29.4 Operation

29.4.1 Initialize Sequence

1. Set ADC00 to 0x0000000F.
2. Wait bit 8 of ADC00 to be set.

29.4.2 Compensating Sensing Method

- Auto compensating sensing mode
  1. Set ADC00 to 0x0000002F.
  2. Wait bit 5 of ADC00 to be reset.
  3. Storing Compensating Value(CV) = 0x200 - ADCC4[25:16].
  4. Adding CV to every channel reading(i.e. ADC10[9:0] or ADC10[25:16])
- Compensating sensing mode
  1. Set ADC00 to 0x0000001F
  2. Set ADC00 to 0x0001001F
  3. Wait a sensing cycle.
     - Sensing cycle = 12 * period of PCLK * 2 * (ADC0C[31:17] + 1) * (ADC0C[9:0] + 1)
  4. Storing a sample of Compensating Value(CV) = 0x200 - ADC10[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])

29.4.3 Voltage Sense Method

Value read from register(VR) = $(V2 + \frac{R2}{R1+R2}(V1 - V2)) * \frac{1024}{1.8} - 1$

V1 = $(\frac{1.8}{1024}\frac{R1+R2}{R2}(VR + 1)) - \frac{R1}{R2}V2$

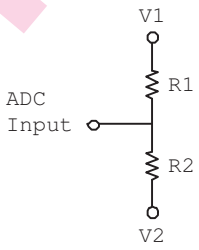


Figure 49: ADC Reference Circuit

30 APB to PCIe Bus Bridge

30.1 Overview

P2X Engine is a bridge between APB bus and PCIe bus.
It also supports a direct mapping space. Base address and size is programmable

P2X00: Engine Status and Engine Control.
P2X04: Interrupt Enable and Interrupt Status.
P2X08: Target ID and Mask.
P2X10: Sending Descriptor[127:96]
P2X14: Sending Descriptor[95:64]
P2X18: Sending Descriptor[63:32]
P2X1C: Sending Descriptor[31: 0]
P2X20: Sending Data Port
P2X40: Received Descriptor[127:96]
P2X44: Received Descriptor[95:64]
P2X48: Received Descriptor[63:32]
P2X4C: Received Descriptor[31: 0]
P2X50: Received Data Port
P2X70: MSI interrupt enable.
P2X74: MSI interrupt status.
P2X80: Direct mapping space base address.
P2X84: Direct mapping space mask.
P2X88: Direct mapping tag and time out.

30.2 Features

- Connect to APB bus and PCIe bus.
- Flexible packet type for sending and receiving.
- Interrupt for receiving or sending complete

30.3 Registers : Base Address = 0x1E6F_0000

Offset: 00h	P2X00: Engine Status and Engine Control	Init = 0x00010000
Bit	**R/W**	**Description**
31:23		**Reserved**
22:18	R	**Length of received data.** Value 0 is 0 byte. Value 1 is 4 bytes. ... Value 16 is 64 bytes. 17 or larger is forbidden.
17	R	**Receive engine idle**
16	R	**Send engine idle**
15: 4		**Reserved**
5	RW	**Enable to receive MSI packet.**
4	RW	**Unlock receive buffer.**

to next page

from previous page

3	RW	**Receive packet control.** 1: Receive packet with Tag matching P2X08. 0: Receive all packets.
2	RW	**Disable dropping incoming packet when fifo is not empty.** 1: Receive packet only when fifo is empty. This will hold PCIe bus. 0: Drop packet.
1	W	**Begin to send packet.**
0	RW	**Enable receive packet.**

Offset: 04h **P2X04: Interrupt Enable and Interrupt Status** **Init = 0**

Bit	R/W	Description
31:22		**Reserved**
21	RW	**Interrupt enable of incoming INTD.**
20	RW	**Interrupt enable of incoming INTC.**
19	RW	**Interrupt enable of incoming INTB.**
18	RW	**Interrupt enable of incoming INTA.**
17	RW	**Interrupt enable of receiving complete**
16	RW	**Interrupt enable of sending complete**
15: 7		**Reserved**
6	R	**Interrupt status of incoming MSI.**
5	R	**Interrupt status of incoming INTD.**
4	R	**Interrupt status of incoming INTC.**
3	R	**Interrupt status of incoming INTB.**
2	R	**Interrupt status of incoming INTA.**
1	RW	**Interrupt status of receiving complete.** Write 1 to clear.
0	RW	**Interrupt status of sending complete.** Write 1 to clear.

Offset: 08h **P2X08: Target ID and Mask** **Init = 0**

Bit	R/W	Description
31: 8		**Reserved**
7: 0	RW	**Target Tag**

Offset: 10h **P2X10: Sending Descriptor [127:96]** **Init = 0**

Bit	R/W	Description
31: 0	RW	**Sending Descriptor[127:96]**

Offset: 14h **P2X14: Sending Descriptor [95:64]** **Init = 0**

Bit	R/W	Description
31: 0	RW	**Sending Descriptor[95:64]**

Offset: 18h	P2X18: Sending Descriptor [63:32]	Init = 0
Bit	**R/W**	**Description**
31: 0	RW	**Sending Descriptor[63:32]**

Offset: 1Ch	P2X1C: Sending Descriptor [31:0]	Init = 0
Bit	**R/W**	**Description**
31: 0	RW	**Sending Descriptor[31:0]**

Offset: 20h	P2X20: Sending Data Port	Init = 0
Bit	**R/W**	**Description**
31: 0	W	**Sending Data**

Offset: 40h	P2X40: Received Descriptor [127:96]	Init = 0
Bit	**R/W**	**Description**
31: 0	R	**Received Descriptor[127:96]**

Offset: 44h	P2X44: Received Descriptor [95:64]	Init = 0
Bit	**R/W**	**Description**
31: 0	R	**Received Descriptor[95:64]**

Offset: 48h	P2X48: Received Descriptor [63:32]	Init = 0
Bit	**R/W**	**Description**
31: 0	R	**Received Descriptor[63:32]**

Offset: 4Ch	P2X4C: Received Descriptor [31:0]	Init = 0
Bit	**R/W**	**Description**
31: 0	R	**Received Descriptor[31:0]**

Offset: 50h	P2X50: Received Data Port	Init = 0
Bit	**R/W**	**Description**
31: 0	R	**Received Data**

Offset: 70h	P2X70: MSI interrupt enable	Init = 0
Bit	**R/W**	**Description**
31: 0	RW	**MSI interrupt enable**

Offset: 74h	P2X74: MSI interrupt status	Init = 0
Bit	**R/W**	**Description**
31: 0	RW	**MSI interrupt status**

Offset: 80h	P2X80: Direct mapping space base address.	Init = 0
Bit	**R/W**	**Description**
31: 0	RW	**Direct mapping space base address**

Offset: 84h	P2X84: Direct mapping space mask.	Init = 0

Bit	R/W	Description
31: 0	RW	**Direct mapping space mask**

Offset: 88h	P2X88: Direct mapping tag and time out.	Init = 0

Bit	R/W	Description
31:13		**Reserved**
12: 8	RW	**Direct mapping time out counter**
7: 0	RW	**Direct mapping tag**

30.4 Operation

30.4.1 Send Packet

1. Check previous sending is done. (P2X04[0] is 1 or P2X00[16] is 1)
2. Writing PCIe header to P2X10 P2X1C.
   - Writing to descriptor[31:0](P2X1C) will make HW to latch descriptor.
3. Writing P2X Payload to P2X20.
4. Set P2X00[1]. This will trigger HW to send.
5. Waiting transaction complete.
   (a) Polling P2X00[16] to be set. Or
   (b) Wait P2X04[0] to be set. This may cause an interrupt if P2X04[16] is also 1.

30.4.2 Receive Packet

1. Enable receiving. (Set P2X00[0])
2. Wait packet fully received.
   (a) Polling P2X00[17] to be set. Or
   (b) Wait P2X04[1] to be set. This may cause an interrupt if P2X04[17] is enabled.
3. Read PCIe header (P2X40 to P2X4C)
4. Length of data received is decoded from PCIe header or from P2X00[22:18].
5. Read P2X50 repeatly to get data received.

31 eSPI Controller

31.1 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.
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: GPIO Direction of Virtual Wire Channel.
ESPI0B8: GPIO Selection of Virtual Wire Channel.
ESPI0BC: GPIO Reset Selection of Virtual Wire Channel.
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.

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.

31.2 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.
  - Support 32 GPIOs. These GPIOs can be connected to physical pins or software mode.
- Out-Of-Band Channel:
  - Maximum payload size is 64 bytes.
- Run-time Flash Sharing Channel:
  - Maximum payload size is 64 bytes.
  - Supports both master- and slave-attached flash sharing.

31.3 Registers : Base Address = 0x1E6E_E000

Offset: 000h	ESPI000: Engine Control	Init = 0xff003000
Bit	**R/W**	**Description**
31	RW	**Flash Tx SW Reset** 0: Reset. This bit is to reset Flash Channel Tx queue.

to next page

from previous page

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**
18	R	**Reserved**
17	RW	**Peripheral Channel Posted Tx DMA Enable**
16	RW	**Peripheral Channel Posted Rx DMA Enable**
15:14		**Reserved**
13	RW	**Direction of eSPI_Reset#** 1b: Input. 0b: Output.
12	RW	**Value of eSPI_Reset#**
11	R	**Reserved**
10	RW	**Software Mode of Flash Read operation**
9	RW	**Software Mode of GPIO through Virturl Wire Channel.**
8		**Reserved**
7	RW	**Flash Channel Software Ready.** This bit must be set to Tx/Rx Flash Channel Write/Erase cycles.
6	R	**Flash Channel Ready.**
5		**Reserved**
4	RW	**OOB Channel Ready.** This bit must be set to Tx/Rx OOB Channel.
3	RW	**Virtual Wire Channel Software Ready.** This bit must be set to Tx/Rx Virtual Wire Channel.

to next page

from previous page

2	R	**Virtual Wire Channel Ready.**
1	RW	**Peripheral Channel Software Ready.** This bit must be set to Tx/Rx Peripheral Channel Massage cycle.
0	R	**Peripheral Channel Ready.**

Offset: 004h **ESPI004: Engine Status** **Init = 0x00001010**

Bit	R/W	Description
31:28	R	**Command Abort Counter**
27:24	R	**Reserved**
23	R	**Flash Channel Tx Busy**
22	R	**Flash Channel Rx Busy**
21	R	**OOB Channel Tx Busy**
20	R	**OOB Channel Rx Busy**
19	R	**Peripheral Channel Non-Posted Tx Busy**
18	R	**Peripheral Channel Non-Posted Rx Busy**
17	R	**Peripheral Channel Posted Tx Busy**
16	R	**Peripheral Channel Posted Rx Busy**
15	R	**Reserved**
14:12	R	**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	R	**Reserved**
9	R	**Virtual Wire Channel Tx Locked** Tx is locked before any Rx received.
8	R	**Peripheral Channel Bus Master Enable**
7	R	**Flash Channel Enable**
6	R	**OOB Channel Enable**
5	R	**Virtual Wire Channel Enable**
4	R	**Peripheral Channel Enable**
3: 1	R	**Reserved**
0	R	**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. Write 1 Clear
30:23	R	**Reserved**

to next page

from previous page

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
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

to next page

from previous page

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	R	**Reserved**
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	RW	**Enable of Hardware Reset Event**
30:22	R	**Reserved**
21	RW	**Enable of Flash Channel Tx Error**
20	RW	**Enable of OOB Channel Tx Error**
19	RW	**Enable of Flash Channel Tx Abort**
18	RW	**Enable of OOB Channel Tx Abort**
17	RW	**Enable of Peripheral Channel Non-Posted Tx Abort**
16	RW	**Enable of Peripheral Channel Posted Tx Abort**
15	RW	**Enable of Flash Channel Rx Abort**
14	RW	**Enable of OOB Channel Rx Abort**
13	RW	**Enable of Peripheral Channel Non-Posted Rx Abort**
12	RW	**Enable of Peripheral Channel Posted Rx Abort**
11	RW	**Enable of Peripheral Channel Non-Posted Tx Error**
10	RW	**Enable of Peripheral Channel Posted Tx Error**
9	RW	**Enable of Virtual Wire GPIO Event**
8	RW	**Enable of Virtual Wire System Event**
7	RW	**Enable of Flash Channel Tx Complete**
6	RW	**Enable of Flash Channel Rx Complete**
5	RW	**Enable of OOB Channel Tx Complete**
4	RW	**Enable of OOB Channel Rx Complete**
3	RW	**Enable of Peripheral Channel Non-Posted Tx Complete**
2	R	**Reserved**
1	RW	**Enable of Peripheral Channel Posted Tx Complete**
0	RW	**Enable of Peripheral Channel Posted Rx Complete**

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		**Reserved**

Offset: 014h	ESPI014: Control of Peripheral Channel Posted Rx Packet	Init = 0
Bit	**R/W**	**Description**
31	W	**Trigger** Set to 1 when current packet is serviced.
30:24		**Reserved**
23:12	R	**Length**
11: 8	R	**Tag**
7: 0	R	**Cycle Type**

Offset: 018h	ESPI018: Data port of Peripheral Channel Posted Rx Packet	Init = 0
Bit	**R/W**	**Description**
31: 8		**Reserved**
7: 0	R	**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		**Reserved**

Offset: 024h	ESPI024: Control of Peripheral Channel Posted Tx Packet	Init = 0
Bit	**R/W**	**Description**
31	W	**Trigger** Set to 1 to trigger transaction.
30:24		**Reserved**
23:12	RW	**Length**
11: 8	RW	**Tag**
7: 0	RW	**Cycle Type**

Offset: 028h	ESPI028: Data port of Peripheral Channel Posted Tx Packet	Init = 0
Bit	**R/W**	**Description**
31: 8		**Reserved**
7: 0	W	**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		**Reserved**

Offset: 034h	ESPI034: Control of Peripheral Channel Non-Posted Tx Packet	Init = 0
Bit	**R/W**	**Description**
31	W	**Trigger** Set to 1 to trigger transaction.
30:24		**Reserved**
23:12	RW	**Length**
11: 8	RW	**Tag**
7: 0	RW	**Cycle Type**

Offset: 038h	ESPI038: Data port of Peripheral Channel Non-Posted Tx Packet	Init = 0
Bit	**R/W**	**Description**
31: 8		**Reserved**
7: 0	W	**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		**Reserved**

Offset: 044h	ESPI044: Control of OOB Channel Rx Packet	Init = 0
Bit	**R/W**	**Description**
31	W	**Serviced** Set to 1 when current packet is serviced.
30:24		**Reserved**
23:12	R	**Length**
11: 8	R	**Tag**
7: 0	R	**Cycle Type**

Offset: 048h	ESPI048: Data port of OOB Channel Rx Packet	Init = 0
Bit	**R/W**	**Description**
31: 8		**Reserved**
7: 0	R	**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		**Reserved**

Offset: 054h	ESPI054: Control of OOB Channel Tx Packet	Init = 0
Bit	**R/W**	**Description**
31	W	**Trigger** Set to 1 to trigger transaction.
30:24		**Reserved**
23:12	RW	**Length**

to next page

from previous page

11: 8	RW	**Tag**
7: 0	RW	**Cycle Type**

Offset: 058h **ESPI058: Data port of OOB Channel Tx Packet** **Init = 0**

Bit	R/W	Description
31: 8		**Reserved**
7: 0	W	**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		**Reserved**

Offset: 064h **ESPI064: Control of Flash Channel Rx Packet** **Init = 0**

Bit	R/W	Description
31	W	**Serviced** Set to 1 when current packet is serviced.
30:24		**Reserved**
23:12	R	**Length**
11: 8	R	**Tag**
7: 0	R	**Cycle Type**

Offset: 068h **ESPI068: Data port of Flash Channel Rx Packet** **Init = 0**

Bit	R/W	Description
31: 8		**Reserved**
7: 0	R	**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		**Reserved**

Offset: 074h **ESPI074: Control of Flash Channel Tx Packet** **Init = 0**

Bit	R/W	Description
31	W	**Trigger** Set to 1 to trigger transaction.
30:24		**Reserved**
23:12	RW	**Length**
11: 8	RW	**Tag**
7: 0	RW	**Cycle Type**

Offset: 078h	ESPI078: Data port of Flash Channel Tx Packet	Init = 0
Bit	**R/W**	**Description**
31: 8		**Reserved**
7: 0	W	**Data**

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 = 0
Bit	**R/W**	**Description**
31: 0	RW	**Mapping Address Mask of Peripheral Channel Rx Packet**

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		**Reserved**
15: 8	RW	**Mapping Mask of Flash Channel**
9: 0		**Reserved**

Offset: 094h	ESPI094: Interrupt enable of System Event from Master	Init = 0
Bit	**R/W**	**Description**
31:11		**Reserved**
10	RW	**Enable Interrupt of NMI Out**
9	RW	**Enable Interrupt of SMI Out**
8	RW	**Enable Interrupt of Host Reset Warn**
7		**Reserved**
6	RW	**Enable Interrupt of OOB Reset Warn**
5	RW	**Enable Interrupt of PLTRSTN**
4	RW	**Enable Interrupt of Suspend Status**
3		**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 = 0x04800600
Bit	**R/W**	**Description**
31:28		**Reserved**
27	RW	**Host Reset Acknowledge**

to next page

from previous page

26	RW	**Reset CPU Init#**
25:24		**Reserved**
23	RW	**Slave Boot Status**
22	RW	**Non-Fatal Error**
21	RW	**Fatal Error**
20	RW	**Slave Boot Done**
19:17		**Reserved**
16	RW	**OOB Reset Acknowledge**
15:11		**Reserved**
10	R	**NMI Out**
9	R	**SMI Out**
8	R	**Host Reset Warn**
7		**Reserved**
6	R	**OOB Reset Warn**
5	R	**PLTRSTN**
4	R	**Suspend Status**
3		**Reserved**
2	R	**S5 Sleep Control**
1	R	**S4 Sleep Control**
0	R	**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 = 0x0304000f**

Bit	R/W	Description
31	R	**CRC Check Enable**
30	R	**Response Modifier Enable**
29		**Reserved**
28	R	**Alert Mode**
27:26	R	**IO Mode Select** 00b Single IO. 01b Dual IO. 10b Quad IO. 11b Reserved
25:24	R	**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	R	**Open Drain Alert# Select** 0b: Alert# pin is a driven output. 1b: Alert# pin is an open-drain output.

to next page

from previous page

22:20	R	**Operating Frequency.** 000b: 20MHz. 001b: 25MHz. 010b: 33MHz. 011b: 50MHz. 100b: 66MHz. Others: Reserved.
19	R	**Alert# pin type status** 1b: Open-drain Alert# pin is supported.
18:16	R	**Maximum Frequency Support** 000b 20 MHz 001b 25 MHz 010b 33 MHz 011b 50 MHz 100b 66 MHz Others Reserved
15:12	R	**Maximum Wait State allowed**
11: 8		**Reserved**
7: 0	R	**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 = 0x00001111**

Bit	R/W	Description
31:15		**Reserved**
14:12	R	**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		**Reserved**
10: 8	R	**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		**Reserved**
6: 4	R	**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

to next page

from previous page

3		**Reserved**
2	R	**Bus Master Enable**
1	R	**Peripheral Channel Ready**
0	R	**Peripheral Channel Enable**

Offset: 0A8h **ESPI0A8: Channel 1 Capabilities and Configurations** **Init = 0x00000700**

Bit	R/W	Description
31:22		**Reserved**
21:16	R	**Operating Maximum Virtual Wire Count**
15:14		**Reserved**
13: 8	R	**Maximum Maximum Virtual Wire Count Supported**
7: 2		**Reserved**
1	R	**Virtual Wire Channel Ready**
0	R	**Virtual Wire Channel Enable**

Offset: 0ACh **ESPI0AC: Channel 2 Capabilities and Configurations** **Init = 0x00000110**

Bit	R/W	Description
31:11		**Reserved**
10: 8	R	**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
7		**Reserved**
6: 4	R	**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		**Reserved**
1	R	**OOB Message Channel Ready**
0	R	**OOB Message Channel Enable**

Offset: 0B0h **ESPI0B0: Channel 3 Capabilities and Configurations** **Init = 0x00001124**

Bit	R/W	Description
31:15		**Reserved**
14:12	R	**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.

to next page

from previous page

11	R	**Flash Sharing Mode** 0b: Master attached flash sharing. 1b: Slave attached flash sharing.
10: 8	R	**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	R	**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	R	**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	R	**Flash Access Channel Ready**
0	R	**Flash Access Channel Enable**

Offset: 0B4h **ESPI0B4: GPIO Direction of Virtual Wire Channel** **Init = 0**

Bit	R/W	Description
31: 0	R	**GPIO Direction of Virtual Wire Channel** 0b: Input 1b: Output

Offset: 0B8h	ESPI0B8: GPIO Selection of Virtual Wire Channel	Init = 0
Bit	**R/W**	**Description**
31:24	R	**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	R	**Select GPIO group for GPIO23 - GPIO16**
15: 8	R	**Select GPIO group for GPIO15 - GPIO08**
7: 0	R	**Select GPIO group for GPIO07 - GPIO00**

Offset: 0BCh	ESPI0BC: GPIO Reset Selection of Virtual Wire Channel	Init = 0
Bit	**R/W**	**Description**
31: 0	R	**GPIO Reset Selection of Virtual Wire Channel** 0b: Reset by eSPI_Reset# 1b: Reset by PLTRSTN

Offset: 100h	ESPI100: Interrupt enable of System Event 1 from Master	Init = 0
Bit	**R/W**	**Description**
31:17		**Reserved**
16	RW	**Enable Interrput of Host C10**
15: 8	RW	**Enable Interrput of PCH Generic**
7: 6		**Reserved**
5	RW	**Enable Interrput of Wireless Lan Sleep Control**
4	RW	**Enable Interrput of Lan Sleep Control**

to next page

from previous page

3	RW	**Enable Interrput of A# Sleep Control**
2		**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	R	**BMC Generic**
23:21		**Reserved**
20	RW	**Suspend Ack**
19:17		**Reserved**
16	R	**Host C10**
15: 8	R	**PCH Generic**
7: 6		**Reserved**
5	R	**Wireless Lan Sleep Control**
4	R	**Lan Sleep Control**
3	R	**A# Sleep Control**
2		**Reserved**
1	R	**Suspend PowerDown Ack**
0	R	**Suspend Warn**

Offset: 110h **ESPI110: Interrupt type 0 of System Event from Master** **Init = 0**

Bit	R/W	Description
31:11		**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		**Reserved**
6	RW	**Interrupt type 0 of OOB Reset Warn**
5	RW	**Interrupt type 0 of PLTRSTN**
4	RW	**Interrupt type 0 of Suspend Status**
3		**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		**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**

to next page

from previous page

7		**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		**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		**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		**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		**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		**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		**Reserved**

to next page

from previous page

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		**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		**Reserved**
16	RW	**Interrupt type 0 of Host C10**
15: 8	RW	**Interrupt type 0 of PCH Generic**
7: 6		**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		**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		**Reserved**
16	RW	**Interrupt type 1 of Host C10**
15: 8	RW	**Interrupt type 1 of PCH Generic**
7: 6		**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		**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		**Reserved**
16	RW	**Interrupt type 2 of Host C10**

to next page

from previous page

15: 8	RW	**Interrupt type 2 of PCH Generic**
7: 6		**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		**Reserved**
1	RW	**Interrupt type 2 of Suspend PowerDown Ack**
0	RW	**Interrupt type 2 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: 12Ch **ESPI12C: Interrupt status of System Event 1 from Master** **Init = 0**

Bit	R/W	Description
31:17		**Reserved**
16	RW	**Interrupt status of Host C10**
15: 8	RW	**Interrupt status of PCH Generic**
7: 6		**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		**Reserved**
1	RW	**Interrupt status of Suspend PowerDown Ack**
0	RW	**Interrupt status of Suspend Warn**

Offset: 004h **ESPICFG004: Device Identification** **Init = 0000_0001**

Bit	R/W	Description
31: 8		**Reserved.**
31: 0	R	**Version ID**

Offset: 008h **ESPICFG008: General Capabilities and Configurations** **Init = 0304_000f**

Bit	R/W	Description
31	RW	**CRC Checking Enable.**
30	RW	**Response Modifier Enable.**
29		**Reserved.**

to next page

from previous page

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	R	**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	R	**Alert# pin type status** 1b: Open-drain Alert# pin is supported.
22:20	R	**Maximum Frequency Supported.** 000b: 20MHz. 001b: 25MHz. 010b: 33MHz. 011b: 50MHz. 100b: 66MHz. Others: Reserved.
15:12	RW	**Maximum WAIT STATE Allowed.**
11: 4		**Reserved.**
3: 0	R	**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_1113**
Bit	**R/W**	**Description**
31:15		**Reserved.**

to next page

from previous page

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		**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		**Reserved.**
6: 4	R	**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		**Reserved.**
2	RW	**Bus Master Enable.**
1	R	**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		**Reserved.**
27:16	RW	**Operating Maximum Virtual Wire Count.**
15:14		**Reserved.**
13: 8	RW	**Maximum Virtual Wire Count Supported.**
7: 2		**Reserved.**
1	R	**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		**Reserved.**

to next page

from previous page

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		**Reserved.**
6: 4	R	**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		**Reserved.**
1	R	**OOB Message Channel Ready.**
0	RW	**OOB Message Channel Enable.**

Offset: 040h ESPICFG040: Channel 3 Capabilities and Configurations Init = 0000_1124

Bit	R/W	Description
31:15		**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.
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.

to next page

from previous page

7: 5	R	**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	R	**Flash Access Channel Ready.**
0	RW	**Flash Access Channel Enable.**

Offset: 044h ESPICFG044: Channel 3 Capabilities and Configurations 2 Init = 0000_ff00

Bit	R/W	Description
31:22		**Reserved.**
21:16	R	**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	R	**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	R	**Reserved.**
2: 0	R	**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.**

31.4 Programming Guide

31.4.1 Address of Peripheral Channel Rx Packet.(MemWR and MemRD)

- When a memory write or memory read received, the address in packet must follow rules below. Response non-fatal error if otherwise.
  - Addr in packet = ESPICFG810 & ∼ESPICFG818.
- The mapped address is like below:
  - Target Addr = (ESPICFG810 & ∼ESPICFG818) || (ESPICFG814 & ESPICFG818).

31.4.2 Process of receiving packet

- Channel 0, 2, and 3
  1. After receiving a packet of channel 0, 2, 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.

31.4.3 Procedure to sending packet

- Channel 0, 2, 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

32 Battery Backed SRAM

32.1 Overview

Battery Backed SRAM is a 64 bytes memory that is direct connected to APB bus.
Address space is from 0x1E6E_0000 to 0x1E6E_FFFF. The DRAM data will be backed by a chip external 3.0 - 3.5V battery when AST2500 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 Timer = 0x1E6E_F000
Physical address = (Base address of Timer) + Offset

Offset: 000h	**BSRAM00: Protection Key Register**	**Init = 0**
Bit	**R/W**	**Description**
31:0	RW	**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 **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.

BSRAM100 ~ BSRAM13C: Battery Backed SRAM #0 ~ #15		
Offset: 100~13Fh		**Init = X**
Bit	**Attr.**	**Description**
31:0	RW	**Battery Backed SRAM** The SRAM data backed by external battery.

33 Video Engine (VE)

33.1 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

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)
VR204: Video Management Sequence Control Register
VR208: Video Management Control Register
VR244: Video Management Based Address of Video Source Buffer Register
VR254: Video Management Base Address of Compressed Video Buffer Register
VR258: Video Management Buffer SizeRegister
VR260: Video Management Compression or Video Profile 2-5 Decompression Control Register
VR278: Video Management Frame-End Offset of Compressed Video Stream Buffer Read Back Register
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

33.2 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

33.3 Registers : Base Address = 0x1E70:0000

VR000: Protection Key Register		
Offset: 000h		**Init = 0**
Bit	**Attr.**	**Description**
31:0	RW	**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.**	**Description**
31:24	R	**Reserved (0)**
23	R	**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	R	**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	R	**Video halt engine read back** 0: Video halt engine mode is not triggered 1: Video halt engine mode is triggered
20	R	**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	R	**Reserved (0)**
18	R	**Video compression engine status** 0: Video compression engine is in busy state 1: Video compression engine is in idle state
17	R	**Reserved (0)**
16	R	**Video capture engine status** 0: Video capture engine is in busy state 1: Video capture engine is in idle state
15	RW	**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.
14	RW	**Enable new watchdog enable** 0: ASPEED proprietary compression mode 1: Enable JPEG compatible mode This bit is used to replace CR004[0] when mode change watchdog enabled. This bit will preserve the VR0B8(VR098) value when input changed.

to next page

from previous page

13	RW	**Enable JPEG JFIF compatible mode compression/decompression** 0: ASPEED proprietary compression mode 1: Enable JPEG compatible mode
12	RW	**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	**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	**Reserved** This register must always be ”0”
8	RW	**Enable YUV to RGB transform dither** 0: Disable dither 1: Enable dither to reduce quantization error effects
7	RW	**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	**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	**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	RW	**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	**Enable capturing multiple frames** 0: Capturing single frame 1: Capture multiple frames AST2500 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	**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	RW	**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 trigering register setting. Software must make sure that the VR004[16] is '1' before trigger this bit.
0	RW	**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	NA	NA
1	1	Multiple frames	Hardware auto trigger	Stream buffer mode

VR008: Video Control Register

Offset: 008h **Init = 0**

Bit	Attr.	Description
31:24	R	**Reserved (0)**
23:16	RW	**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	**HSYNC polarity control for mode detection** 0: Normal HSYNC 1: Inverted HSYNC polarity for mode detection
14	RW	**Video source interlace mode** 0: Capture source is in the progressive mode 1: Capture source is in the interlace mode
13:12	RW	**Reserved** This register must be always ”0”
11:9	RW	**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	**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.

to next page

from previous page

8	RW	**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	**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 AST2500 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	**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	**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.
4	RW	**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	**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	**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	**Video source selection** 0: Video source is from the integrated VGA controller 1: Video source is from an external video source
1	RW	**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.

to next page

from previous page

0	RW	**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.	Description
31:29	R	**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.	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

to next page

from previous page

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.	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.	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 AST2500 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.	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.**	**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.**	**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 = 0**
Bit	**Attr.**	**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	R	**Reserved (0)**
7:5	RW	**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	**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.

to next page

from previous page

2	RW	**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	**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	**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.	Description
31:29	R	**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.	Description
31:29	R	**Reserved (0)**
28:16	RW	**Horizontal total pixels to be compressed**
15:13	R	**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 = X**

Bit	Attr.	Description
31:24	R	**Reserved (0)**
23:7	RW	**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	R	**Reserved (0)**

VR03C: Video Compression Stream Buffer Read Offset Register		
Offset: 03Ch		**Init = X**
Bit	**Attr.**	**Description**
31:24	R	**Reserved (0)**
23:7	RW	**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	R	**Reserved (0)**

VR040: Video Base Address of JPEG Header Buffer Register when VR004[13]=1		
Offset: 040h		**Init = X**
Bit	**Attr.**	**Description**
31:30	R	**Reserved (0)**
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 2KB. 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	R	**Reserved (0)**

VR044: Video Based Address of Video Source Buffer #1 Register		
Offset: 044h		**Init = X**
Bit	**Attr.**	**Description**
31:30	R	**Reserved (0)**
29:8	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	R	**Reserved (0)**

VR048: Video Scan Line Offset of Video Source Buffer Register		
Offset: 048h		**Init = X**
Bit	**Attr.**	**Description**
31:15	R	**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	R	**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.**	**Description**
31:30	R	**Reserved (0)**
29:8	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	R	**Reserved (0)**

VR050: Video Base Address of BCD Flag Buffer Register		
Offset: 050h		**Init = X**
Bit	**Attr.**	**Description**
31:30	R	**Reserved (0)**
29:4	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	R	**Reserved (0)**

VR054: Video Base Address of Compressed Video Stream Buffer Register		
Offset: 054h		**Init = X**
Bit	**Attr.**	**Description**
31:30	R	**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	RW	**Reserved (0)**

VR058: Video Stream Buffer Size Register		
Offset: 058h		**Init = X**
Bit	**Attr.**	**Description**
31:6	R	**Reserved (0)**
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

to next page

from previous page

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 = X**

Bit	Attr.	Description
31:24	R	**Reserved (0)**
23:7	R	**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	R	**Reserved (0)**

VR060: Video Compression Control Register

Offset: 060h **Init = 0**

Bit	Attr.	Description
31:27	RW	**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

to next page

from previous page

26:22	RW	**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	**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.
19	RW	**Reserved** This register must be always ”1”.
18:17	RW	**DCT engine hardware test control** For testing hardware only. Writing 0 for a normal condition.
16	RW	**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	**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.

to next page

from previous page

10:6	RW	**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.
5	RW	**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	**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	**Reserved** This register must be always ”0”.
2	RW	**Compression/decompression mode setting for high video quality** 0: DCT mode 1: Quantization mode
1	RW	**Reserved** This register must be always ”0”.
0	RW	**Reserved** This register can be ”0” or ”1”.

VR064: Video JPEG effecive bit control Register

Offset: 064h **Init = 0**

Bit	Attr.	Description
31:7	R	**Reserved (0)**
6	RW	**JPEG chrominance dither mode** 0 Disable dither 1 Enable dithe
5	RW	**JPEG luminance dither mode** 0 Disable dither 1 Enable dithe
4	RW	**Reserved (0)**

to next page

from previous page

3 :2	RW	**Effective JPEG chrominance bits** 00: 8-bit 01: 7-bit 10: 6-bit 11: 5-bit
1 :0	RW	**Effective JPEG luminance bits** 00: 8-bit 01: 7-bit 10: 6-bit 11: 5-bit
1 :0	RW	**Reserved (0)**

VR068: Video Quantization value

Offset: 068h **Init = X**

Bit	Attr.	Description
31	R	**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	R	**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.	Description
31:30	R	**Reserved (0)**
29:4	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	R	**Reserved (0)**

VR070: Video Total Size of Compressed Video Stream Read Back Register

Offset: 070h **Init = X**

Bit	Attr.	Description
31:30	R	**Reserved (0)**
29:28	R	**Compressed block counter read back (number of blocks) overflow bits**
27:26	R	**Reserved (0)**
25:24	R	**Processed total block counter read back (number of blocks) overflow bits**
23:20	R	**Reserved (0)**
21:0	R	**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.	Description
31:16	R	**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	R	**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 = X**

Bit	Attr.	Description
31:24	R	**Reserved (0)**
23:3	R	**Frame-end offset of compressed video stream buffer Bit [23:3]** 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	R	**Reserved (0)**

VR07C: Video Compressed Frame Counter Read Back Register

Offset: 07Ch **Init = X**

Bit	Attr.	Description
31:0	R	**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.	Description
31:16	R	**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.

VR090: Video Source Left/Right Edge Detection Read Back Register

Offset: 090h **Init = X**

Bit	Attr.	Description
31	R	**Video interlace mode detected** 0: Progressive mode detected 1: Interlace mode detected
30:28	R	**Reserved (0)**
27:16	R	**Video source right edge location from the rising edge of HSYNC Bit [11:0]** The unit of this register is one pixel.

to next page

from previous page

15	R	**No display clock detected** 0: No display clock detected 1: Display clock detected
14	R	**No active display detected** 0: No active display detected 1: Active display detected
13	R	**No HSYNC detected** 0: No HSYNC detected 1: HSYNC detected
12	R	**No VSYNC detected** 0: No VSYNC detected 1: VSYNC detected
11:0	R	**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.	Description
31:29	R	**Reserved (0)**
28:16	R	**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	R	**Reserved (0)**
12:0	R	**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.	Description
31	R	**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	R	**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	R	**Mode detection HSYNC polarity** 0: Source HSYNC polarity is positive 1: Source HSYNC polarity is negative

to next page

from previous page

28	R	**Mode detection VSYNC polarity** 0: Source VSYNC polarity is positive 1: Source VSYNC polarity is negative
27:16	RW	**Mode detection vertical scan lines Bit [11:0]** read: 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”. 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	R	**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	R	**Mode detection vertical signal stable** 0: Vertical signal detection is not stable 1: Vertical signal detection is stable
13	R	**Mode detection horizontal signal stable** 0: Horizontal signal detection is not stable 1: Horizontal signal detection is stable
12	R	**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	RW	**Mode detection horizontal period Bit [11:0]** read: 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 24MHz. write: The write value will initialize the mode detection watch dog HSYNC period which is in unit of 24 MHz reference cycle. 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.

VR09C: Video Sync End Location Detection Read Back Register

Offset: 09Ch **Init = X**

Bit	Attr.	Description
31:28	R	**Reserved (0)**
27:16	R	**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	R	**Reserved (0)**
11:0	R	**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.**	**Description**
31:12	R	**Reserved (0)**
11:0	R	**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.**	**Description**
31	RW	**Video Interlace Signal Generated by Internal Timing Generator** 0: Disable 1: Enable
30	RW	**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	**Long Horizontal Stable Time Detecion Mode** 0: Disable 1: Enable for more stable mode detection
28:24	RW	**Internal Timing Generator Horizontal Sync Width Setting** The unit of this register is in unit of pixel clock cycle.
23:16	RW	**Internal Timing Generator Horizontal Sync Delay Setting** The unit of this register is in unit of display clock cycle.
15:8	RW	**Mode Detection Right Margin Setting** The unit of this register is in unit of display clock cycle.
7 :0	RW	**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.**	**Description**
31	RW	**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	**Reserved (0)**
29	RW	**Disable flipping deflicker during switching between YUV444 and YUV420** 0: Enable 1: Disable
28	RW	**Video input port YUV dual edge mode** 0: Disable 1: Enable
27	RW	**Video input port U/V pin swap** 0: Disable 1: Enable

to next page

from previous page

26	RW	**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	RW	**HDMI 2.0 YUV420 format**
24	RW	**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	**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	**Reserved (0)**
13	RW	**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	**Enable Direct Mode Virtual Timing Generator Programmable** 0: Disable 1: Enable This bit allows frame mode capturing to reach 60 fps.
11	RW	**Enable Fast video capture ready mode** 0: Disable 1: Enable register VR0AC and VR0B0
10:5	RW	**Mode Detection HSYNC De-glitch Setting** The unit of this register is in unit of display clock cycle.
4 :0	RW	**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.	Description
31:24	RW	**Direct Access Mode Virtual Vertical Back Porch Width Timing** The unit of this register is in unit of display clock cycle.
23:16	RW	**Direct Access Mode Virtual Horizontal Back Porch Width Timing** The unit of this register is in unit of display clock cycle.
15:8	RW	**Direct Access Mode Virtual Horizontal Front Porch Width Timing** The unit of this register is in unit of display clock cycle.
7 :0	RW	**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.**	**Description**
31:24	RW	**Direct Access Mode Frame Time Control** The unit of this register is in unit of 64us.
23:16	R	**Reserved (0)**
15:8	RW	**Direct Access Mode Virtual Vertical Front Porch Width Timing** The unit of this register is in unit of line.
7 :0	RW	**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.**	**Description**
31:12	R	**Reserved (0)**
11:7	RW	**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	**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	**Enable big software frame header** 0: Disable 1: Enable It is used for software header and not compatible to old stream.
0	RW	**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.**	**Description**
31:29	R	**Reserved (0)**
28:16	R	**Video source right edge location from the rising edge of HSYNC Bit [12:0]**
15:13	R	**Reserved (0)**
12:0	R	**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.**	**Description**
31:29	R	**Reserved (0)**

to next page

from previous page

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	R	**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.

VR204: Video Management Sequence Control Register

Offset: 204h **Init = 0**

Bit	Attr.	Description
31:19	R	**Reserved (0)**
18	R	**Video management compression status** 0: Video management compression is in busy state 1: Video management compression is in idle state
17	R	**Reserved (0)**
16	R	**Video management capture status** 0: Video management capture is in busy state 1: Video management capture is in idle state
15:12	RW	**Reserved** Thess registers must always be ”0”
11:10	RW	**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	**Reserved** This register must be always ”0”
8	RW	**Enable JPEG compatible mode management compression** 0: ASPEED proprietary compression mode 1: Enable JPEG compatible mode
7:5	R	**Reserved** This register must be always ”0”

to next page

from previous page

4	RW	**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	R	**Reserved** This register must be always ”0”
1	RW	**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	R	**Reserved** This register must be always ”0”

VR208: Video Management Control Register

Offset: 208h **Init = 0**

Bit	Attr.	Description
31:8	R	**Reserved (0)**
7 :6	RW	**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 AST2500 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	R	**Reserved (0)**
0	RW	**Video management auto source register** 0: VR244 is required for management compression 1: VR244 is not required for management compression

VR244: Video Management Based Address of Video Source Buffer Register

Offset: 244h **Init = X**

Bit	Attr.	Description
31:28	R	**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.
7 :0	R	**Reserved (0)**

VR254: Video Management Base Address of Compressed Video Buffer Register		
Offset: 254h		**Init = X**
Bit	**Attr.**	**Description**
31:28	R	**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.
6 :0	R	**Reserved (0)**

VR258: Video Management Buffer SizeRegister		
Offset: 258h		**Init = X**
Bit	**Attr.**	**Description**
31:4	R	**Reserved (0)**
3 :0	RW	**The buffer size of compressed management video stream buffer Bit [3:0]** 0000: 1 M byte 0001: 2 M byte 0010: 3 M byte 0011: 4 M byte .... 1111: 16 M byte

VR260: Video Management Compression or Video Profile 2-5 Decompression Control Register		
Offset: 260h		**Init = 0**
Bit	**Attr.**	**Description**
31:16	R	**Reserved (0)**
15:11	RW	**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.

to next page

from previous page

10:6	RW	**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	**Reserved** This register must be always ”0”.
1	RW	**Reserved** This register must be always ”0”.
0	RW	**Reserved** This register can be ”0” or ”1”.

VR278: Video Management Frame-End Offset of Compressed Video Stream Buffer Read Back Register

Offset: 278h **Init = X**

Bit	Attr.	Description
31:22	R	**Reserved (0)**
21:3	R	**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	R	**Reserved (0)**

VR300: Video Control Register

Offset: 300h **Init = 0**

Bit	Attr.	Description
31:30	RW	**VR400-VR5FC Address mapping selection** 00: Mapping to crypro table 01: Mapping to quantization/inverse quantization table 10: Mapping to Huffman table 11: Reserved
29:28	RW	**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

to next page

from previous page

27	RW	**Decode output write RGB mode** 0: Decode write is tile YUV mode 1: Decode write is linear RGB mode
26:24	RW	**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	**Vertical border suto 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	**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	**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	**Enable profile mode** 0: Disable 1: Enable
19	RW	**Enable programmable Huffman table** 0: Use default Huffman table 1: Use programmable Huffman table. Can't set when encode mode.
18	RW	**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.
17	RW	**Crypto mode selection** 0: RC4 crypto mode 1: AES crypto mode
16	RW	**Fast crypto mode** This register is recommended to be always "0"
15	RW	**Crypto status software reset** 0: Normal operation mode 1: Reset crypto state
14	RW	**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

to next page

from previous page

13:12	RW	**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	**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	**RC4 test mode** This register is recommended to be always ”0”
8	RW	**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	**Reserved** This register must be always ”0”
3 :2	RW	**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	**Reserved (0)**

VR304: Video Interrupt Control Register

Offset: 304h **Init = 0**

Bit	Attr.	Description
31:6	R	**Reserved (0)**
17	RW	**Enable video management compression complete interrupt** 0: Disable 1: Enable
16	RW	**Enable video management frame capture complete interrupt** 0: Disable 1: Enable
15:12	R	**Reserved (0)**
11	RW	**Enable video VSYNC descriptor interrupt** 0: Disable 1: Enable
10	RW	**Enable video stream descriptor interrupt** 0: Disable 1: Enable
9	RW	**Enable video hang watchdog interrupt** 0: Disable 1: Enable
8	RW	**Enable video halt ready interrupt** 0: Disable 1: Enable
7	R	**Reserved (0)**

to next page

from previous page

6	RW	**Enable video decode stream error interrupt** 0: Disable 1: Enable
5	RW	**Enable video frame complete interrupt** 0: Disable 1: Enable
4	RW	**Enable video mode detection ready interrupt** 0: Disable 1: Enable
3	RW	**Enable video compression complete interrupt** 0: Disable 1: Enable
2	RW	**Enable video compression packet ready interrupt** 0: Disable 1: Enable
1	RW	**Enable video frame capture complete interrupt** 0: Disable 1: Enable
0	RW	**Enable video mode detection watchdog out of lock interrupt** 0: Disable 1: Enable

VR308: Video Interrupt Control Register

Offset: 308h **Init = X**

Bit	Attr.	Description
31:10	R	**Reserved (X)**
17	RW	**Video management compression complete interrupt status** 0: No interrupt 1: Interrupt is pending. Clear this register by writing 1.
16	RW	**Video management capture complete interrupt status** 0: No interrupt 1: Interrupt is pending Clear this register by writing 1.
15:12	R	**Reserved (X)**
11	RW	**Video VSYNC descriptor interrupt status** 0: No interrupt 1: Interrupt is pending Clear this register by writing 1.
10	RW	**Video stream descriptor interrupt status** 0: No interrupt 1: Interrupt is pending Clear this register by writing 1.
9	RW	**Video hang watchdog interrupt status** 0: No interrupt 1: Interrupt is pending Clear this register by writing 1.

to next page

from previous page

8	RW	**Video halt ready interrupt status** 0: No interrupt 1: Interrupt is pending Clear this register by writing 1.
7	R	**Reserved (X)**
6	RW	**Video decode stream error interrupt status** 0: No interrupt 1: Interrupt is pending Clear this register by writing 1.
5	RW	**Video frame complete interrupt status** 0: No interrupt 1: Interrupt is pending Clear this register by writing 1.
4	RW	**Video mode detection ready interrupt status** 0: No interrupt 1: Interrupt is pending Clear this register by writing 1.
3	RW	**Video compression complete interrupt status** 0: No interrupt 1: Interrupt is pending. Clear this register by writing 1.
2	RW	**Video compression packet ready interrupt status** 0: No interrupt 1: Interrupt is pending Clear this register by writing 1.
1	RW	**Video capture complete interrupt status** 0: No interrupt 1: Interrupt is pending Clear this register by writing 1.
0	RW	**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.	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 24MHz. The unit of this register is one clock period of 24MHz 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.

to next page

from previous page

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.	Description
31:28	R	**Reserved (0)**
27:16	RW	**Video memory restriction area starting address** The address must be 64 KB alignment.
15:0	R	**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.	Description
31:28	R	**Reserved (0)**
27:16	RW	**Video memory restriction srea end sddress** The address must be 64 KB alignment.
15:0	R	**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.	Description
31:28	R	**Reserved (0)**
27:16	RW	**Video memory restriction area starting address** The address must be 64 KB alignment.
15:0	R	**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.	Description
31:16	R	**Reserved (0)**

to next page

from previous page

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.	Description
31:24	R	**Reserved (0)**
23:0	R	**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.	Description
31:30	R	**Reserved (0)**
29:24	R	**VGA hardware cursor X position offset bit[5:0]**
23:22	R	**Reserved (0)**
21:16	R	**VGA hardware cursor Y position offset bit[5:0]**
15:10	R	**Reserved (0)**
9	R	**VGA hardware cursor type** 0: Monochrome cursor type. 1: Color cursor type.
8	R	**VGA hardware cursor is enabled** 0: VGA hardware cursor is disabled. 1: VGA hardware cursor is enabled.

to next page

from previous page

7 :0	R	**Remap to VGA CR80 register**

VR344: VGA Scratch Remap Read Back Register

Offset: 344h **Init = X**

Bit	Attr.	Description
31:27	R	**Reserved (0)**
26:16	R	**Hardware cursor Y position bit[10:0]**
15:12	R	**Reserved (0)**
11:0	R	**Hardware cursor X position bit[11:0]**

VR348: VGA Scratch Remap Read Back Register

Offset: 348h **Init = X**

Bit	Attr.	Description
31:26	R	**Reserved (0)**
25:3	R	**Hardware cursor pattern memory address bit [25:3]**
2 :0	R	**Reserved (0)**

VR34C: VGA Scratch Remap Read Back Register

Offset: 34Ch **Init = X**

Bit	Attr.	Description
31:24	R	**Remap to VGA CR8F register**
23:16	R	**Remap to VGA CR8E register**
15:8	R	**Remap to VGA CR8D register**
7 :0	R	**Remap to VGA CR8C register**

VR350: VGA Scratch Remap Read Back Register

Offset: 350h **Init = X**

Bit	Attr.	Description
31:24	R	**Remap to VGA CR93 register**
23:16	R	**Remap to VGA CR92 register**
15:8	R	**Remap to VGA CR91 register**
7 :0	R	**Remap to VGA CR90 register**

VR354: VGA Scratch Remap Read Back Register

Offset: 354h **Init = X**

Bit	Attr.	Description
31:24	R	**Remap to VGA CR97 register**
23:16	R	**Remap to VGA CR96 register**
15:8	R	**Remap to VGA CR95 register**
7 :0	R	**Remap to VGA CR94 register**

VR358: VGA Scratch Remap Read Back Register		
Offset: 358h		**Init = X**
Bit	**Attr.**	**Description**
31:24	R	**Remap to VGA CR9B register**
23:16	R	**Remap to VGA CR9A register**
15:8	R	**Remap to VGA CR99 register**
7 :0	R	**Remap to VGA CR98 register**

VR35C: VGA Scratch Remap Read Back Register		
Offset: 35Ch		**Init = X**
Bit	**Attr.**	**Description**
31:30	R	**VGA power state**
29	R	**VGA attribute index register bit 5**
28	R	**VGA mask register not zero**
27	R	**VGA CRT reset**
26	R	**VGA screen off**
25	R	**VGA reset**
24	R	**VGA enable**
23:16	R	**Remap to VGA CR9E register**
15:8	R	**Remap to VGA CR9D register**
7 :0	R	**Remap to VGA CR9C register**

VR360: VGA Scratch Remap Read Back Register		
Offset: 360h		**Init = X**
Bit	**Attr.**	**Description**
31:25	R	**Reserved (0)**
24:16	R	**Remap to VGA horzontal total register**
15:9	R	**Reserved (0)**
8 :0	R	**Remap to VGA horzontal end register**

VR364: VGA Scratch Remap Read Back Register		
Offset: 364h		**Init = X**
Bit	**Attr.**	**Description**
31:27	R	**Reserved (0)**
26:16	R	**Remap to VGA vertical total register**
15:11	R	**Reserved (0)**
10:0	R	**Remap to VGA vertical end register**

VR3F0: Video Crypto Vector Register 1		
Offset: 3F0h		**Init = X**
Bit	**Attr.**	**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.	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.	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.	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.	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.	Description
31:0	RW	**Quantization table SRAM** The address map to quantization table when VR300[30]=1

VR400 ~ VR55C: JPEG Huffman Table #0 ~ #63

Offset: 400~55Ch **Init = X**

Bit	Attr.	Description
31:0	RW	**JPEG Huffman table SRAM** The address map to quantization table when VR300[31]=1

34 SRAM Memory Buffer

34.1 Overview

Embedded SRAM provide 36K Bytes address space, direct connect to AHB bus.
Address space is from 0x1E72_0000 to 0x1E72_8FFF.

35 SD/SDIO Host Controller

35.1 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
SDIO018: HWInit SDIO148 Configuration for slot #0
SDIO020: HWInit SDIO240 Configuration for slot #1
SDIO028: HWInit SDIO248 Configuration for slot #1
SDIO0EC: Write Protection Polarity Control
SDIO0F0: Card Detect 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
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
SDIO248: Maximum Current Capabilities Register
SDIO250: Force Event Register
SDIO254: ADMA Error Status Register
SDIO258: ADMA System Address Register

35.2 Features

- SD Memory Card Version 3.00
- SDIO Card Version 2.00
- MMC Card Version 4.3
- SDIO Host Specification Version 2.00
- SD1/SD4 modes of operation
- 2 slots supported.
- Indepent clock/configuration for each slot.
- Indepent register set for each slot.
- Integrated ADMA2 controllers.
- Master SD card side clock and each slot's clock can be switched off.

35.3 Registers : Base Address = 0x1E74:0000

Offset: 000h		**SDIO000: General Information Register**	**Init = 0x0003_0000**
Bit	**R/W**	**Description**	
31:26	RW	**Reserved**	
25	RW	**S1MMC8** 1: 8-bit mode 0: 1 or 4 bit mode (depends on SDIO228[1])	
24	RW	**S0MMC8** 1: 8-bit mode 0: 1 or 4 bit mode (depends on SDIO128[1])	
23:18	RW	**Reserved**	
17	R	**Slot #1 available** 1: slot is available 0: slot is not available	
16	R	**Slot #0 available** 1: slot is available 0: slot is not available	
15: 1	RW	**Reserved**	
0	RW	**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**	**Description**
31:24	RW	**Reserved**
23: 0	RW	**Debounce Period** Debounce period = SDIO004[23:0] * period of HCLK

Offset: 008h	SDIO008: Bus Settings Register	Init = 0
Bit	**R/W**	**Description**
31: 4	RW	**Reserved**
3: 0	RW	**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**	**Description**
31: 0	RW	**Mirror of SDIO140** Write to update corresponding SRS140 value

Offset: 018h	SDIO018: HWInit SDIO148 Configuration for slot #0	Init = 0
Bit	**R/W**	**Description**
31: 0	RW	**Mirror of SDIO148** Write to update corresponding SRS148 value

Offset: 020h	SDIO020: HWInit SDIO240 Configuration for slot #1	Init = 0
Bit	**R/W**	**Description**
31: 0	RW	**Mirror of SDIO240** Write to update corresponding SRS240 value

Offset: 028h	SDIO028: HWInit SDIO248 Configuration for slot #1	Init = 0
Bit	**R/W**	**Description**
31: 0	RW	**Mirror of SDIO248** Write to update corresponding SRS248 value

Offset: 0ECh	SDIO0EC: Write Protection Polarity Control	Init = 0

Bit	R/W	Description
15:14		**Reserved**
13	RW	**Slot 1 Disable Card Detect** 1: Disable. 0: Enable.
12	RW	**Slot 0 Disable Card Detect** 1: Disable. 0: Enable.
11:10		**Reserved**
9	RW	**Slot 1 Card Detect Mode** 1: Share Card Detect with DAT3. 0: Dedicate Card Detect Pin.
8	RW	**Slot 0 Card Detect Mode** 1: Share Card Detect with DAT3. 0: Dedicate Card Detect Pin.
7: 6		**Reserved**
5	RW	**Slot 1 Clock Inverse bit** 1: Inverse. 0: No operation.
4	RW	**Slot 0 Clock Inverse bit** 1: Inverse. 0: No operation.
3: 2		**Reserved**
1	RW	**Slot 1 Write Protection Inverse bit** 1: Inverse. 0: No operation.
0	RW	**Slot 0 Write Protection Inverse bit** 1: Inverse. 0: No operation.

Note :
SDIO0EC is equaled to SDIO0F4 for backward compatible.
Write to SDIO0EC will affect SDIO0F4, and vice versa.

Offset: 0F0h	SDIO0F0: Card Detect Control	Init = 0

Bit	R/W	Description
31:14		**Reserved**
13	RW	**Slot 1 Disable Card Detect** 1: Disable. 0: Enable.
12	RW	**Slot 0 Disable Card Detect** 1: Disable. 0: Enable.
11:10		**Reserved**
9	RW	**Slot 1 Card Detect Mode** 1: Share Card Detect with DAT3. 0: Dedicate Card Detect Pin.
8	RW	**Slot 0 Card Detect Mode** 1: Share Card Detect with DAT3. 0: Dedicate Card Detect Pin.

to next page

from previous page

7: 6		**Reserved**
5	RW	**Slot 1 Clock Inverse bit** 1: Inverse. 0: No operation.
4	RW	**Slot 0 Clock Inverse bit** 1: Inverse. 0: No operation.
3: 2		**Reserved**
1	RW	**Slot 1 Write Protection Inverse bit** 1: Inverse. 0: No operation.
0	RW	**Slot 0 Write Protection Inverse bit** 1: Inverse. 0: No operation.
Note : SDIO0F0 is equaled to SDIO0EC for backward compatible. Write to SDIO0F0 will affect SDIO0EC, and vice versa.		

Offset: 0FCh	**SDIO0FC: Interrupt Status**	**Init = 0x00010000**
Bit	**R/W**	**Description**
31: 2		**Reserved**
1: 0	RW	**Interrupt Status** Bit 1: Interrput pending in Slot1. Bit 0: Interrput pending in Slot0.

Offset: 100h	**SDIO100: SDMA System Address Register**	**Init = 0**
Bit	**R/W**	**Description**
31: 0	RW	**SDMA System Address**

Offset: 104h	**SDIO104: Block Count and Size Register**	**Init = 0**
Bit	**R/W**	**Description**
31:16	RW	**Block Count for Current Transfer** 0000h: 0 0001h: 1 ... FFFFh: 65535
15	RW	**Reserved**
14:12	RW	**Host DMA Buffer Boundary** 000b 4kB 001b 8kB 010b 16kB 011b 32kB 100b 64kB 101b 128kB 110b 256kB 111b 512kB
11: 0	RW	**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**	**Description**
31: 0	RW	**Command Argument** Contain bit [39:8] of command argument

Offset: 10Ch	SDIO10C: Command and Transfer Mode Register	Init = 0
Bit	**R/W**	**Description**
31:30	RW	**Reserved**
29:24	RW	**Command Index** 00h: CMD0 / ACMD0 01h: CMD1 / ACMD1 ... 3Fh: CMD63 / ACMD63
23:22	RW	**Command Type** 11b Abort Command 10b Suspend Command 01b Resume Command 00b Normal Command
21	RW	**Data Present Select** 1: Command which transfer data 0: all other commands
20	RW	**Command Index Check Enable** 1: Check if command index field of the response equal to SDIO10C[29:24].
19	RW	**Command CRC Check Enable** 1: Check if CRC field of the response is valid.
18	RW	**Reserved**
17:16	RW	**Response Type Select** 00b no response 01b 136-bit response 10b 48-bit response 11b 48-bit response with BUSY
15: 6	RW	**Reserved**
5	RW	**Multi/Single Block Select** 1: Multi block data transfer. 0: Single block data transfer.
4	RW	**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	RW	**Reserved**
2	RW	**Auto CMD12 Enable** 1: Enable. 0: Disable.
1	RW	**Block Count Enable** 1: Enable. 0: Disable.
0	RW	**DMA Enable** 1: Enable. 0: Disable.

Offset: 110h	SDIO110: Response #0	Init = 0
Bit	**R/W**	**Description**
31: 0	R	**Command response[31: 0]**

Offset: 114h	SDIO114: Response #1	Init = 0
Bit	**R/W**	**Description**
31: 0	R	**Command response[63: 32]**

Offset: 118h	SDIO118: Response #2	Init = 0
Bit	**R/W**	**Description**
31: 0	R	**Command response[95: 64]**

Offset: 11Ch	SDIO11C: Response #3	Init = 0
Bit	**R/W**	**Description**
31: 0	R	**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**	**Description**
31: 0	RW	**Buffer Data Port**

Offset: 124h	SDIO124: Present State Register	Init = 0
Bit	**R/W**	**Description**
31:25	RW	**Reserved**
24	R	**CMD Line Signal Level** Equal to the actual signal level on CMD line of the interface.
23:20	R	**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	R	**Write Protect Switch Pin Level** Equal to the actual signal level on Write Protect line of the interface.
18	R	**Card Detect Switch Pin Level** Equal to the actual signal level on Card Detect line of the interface.

to next page

from previous page

17	R	**Card State Stable** Indicate if SDIO124[18] is stable. 1: SDIO124[18] is stable. 0: SDIO124[18] is unstable.
16	R	**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	RW	**Reserved**
11	R	**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	R	**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	R	**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	R	**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: 3	RW	**Reserved**
2	R	**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	R	**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	R	**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	Description
31:27	RW	**Reserved**
26	RW	**Wakeup Event Enable on SD Card Removal** 1: Enable. 0: Disable.
25	RW	**Wakeup Event Enable on SD Card Inserted** 1: Enable. 0: Disable.

to next page

from previous page

24	RW	**Wakeup Event Enable on SD Card Interrupt** 1: Enable. 0: Disable.
23:20	RW	**Reserved**
19	RW	**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	**Read Wait Control** 1: Enable. 0: Disable.
17	RW	**Continue Request** 1: Restart transfer stopped previously using SDIO128[16]. This bit is cleared automatically.
16	RW	**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: 9	RW	**Reserved**
8	RW	**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	**Card Detect Signal Selection** 1: SDIO128[6]. 0: From slot.
6	RW	**Card Detect Test Level** 1: Card inserted. 0: No card.
5	RW	**Reserved**
4: 3	RW	**DMA Select** 00b: SDMA 01b: ADMA1 10b: ADMA2 11b: Reserved
2	RW	**Hish Speed Enable** 1: Enable 0: Disable
1	RW	**Data Transfer Width** 1: SD4 (4-bits) mode. 0: SD1 (1-bit) mode.
0	RW	**Reserved**

Offset: 12Ch **SDIO12C: Host Control Settings #1 Register** **Init = 0**

Bit	R/W	Description
31:27	RW	**Reserved**
26	RW	**Software Reset for DAT Line** 1: Reset data path, including data buffer and DMA logics. This bit is cleared automatically.

to next page

from previous page

25	RW	**Software Reset for CMD Line** 1: Reset command/response generation and checking. This bit is cleared automatically.
24	RW	**Software Reset for All** 1: Entire SDIO-Host is reset. This bit is cleared automatically.
23:20	RW	**Reserved**
19:16	RW	**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	**SDCLK Frequency Select** 80h Period of SDMCLK * 256 40h Period of SDMCLK * 128 20h Period of SDMCLK * 64 10h Period of SDMCLK * 32 08h Period of SDMCLK * 16 04h Period of SDMCLK * 8 02h Period of SDMCLK * 4 01h Period of SDMCLK * 2 00h Forbidden
7: 3	RW	**Reserved**
2	RW	**SD Clock Enable** 1: Enable SD slot clock. 0: Disable SD slot clock.
1	R	**Internal Clock Stable** 1: Indicates clock pin of SDMCLK is stable. 0: Indicates clock pin of SDMCLK is not stable.
0	RW	**Internal Clock Enable** 1: Enable SDMCLK.

Offset: 130h **SDIO130: Interrupt Status Register** **Init = 0**

Bit	R/W	Description
31:26	RW	**Reserved**
25	RW	**ADMA Error** Read 1: Error occurs during ADMA read or write transfer. Write 1 to clear.
24	RW	**Auto CMD12 Error** Read 1: Error occurs during Auto CMD12 command transmition. Write 1 to clear.
23	RW	**Current Limit Error** Read 1: SD Card is not powered due to some failure. Write 1 to clear.
22	RW	**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.

to next page

from previous page

21	RW	**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	**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	**Command Index Error** Read 1: Index error occurs in the command response. Write 1 to clear.
18	RW	**Command End Bit Error** Read 1: Indicate detecting the end bit of a command response is 0. Write 1 to clear.
17	RW	**Command CRC Error** Read 1: Command CRC error occurs. Write 1 to clear.
16	RW	**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	R	**Error Interrupt** 1: One of SDIO130[25:16] is set.
14: 9	RW	**Reserved**
8	R	**Card Interrupt** 1: Indicate the card interrupt.
7	RW	**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	**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	**Buffer Read Ready** Read 1: Indicate data buffer can be read. Write 1 to clear.
4	RW	**Buffer Write Ready** Read 1: Indicate data buffer can be written. Write 1 to clear.
3	RW	**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	RW	**Block Gap Event** Read 1: Indicate read/write transaction is stopped at a block gap. Write 1 to clear.
1	RW	**Transfer Complete** Read 1: Indicate transfer using DAT line is completed. Write 1 to clear.
0	RW	**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	Description
31:26	RW	**Reserved**
25	RW	**ADMA Error Status Enable** 1: Enable. 0: Masked.
24	RW	**Auto CMD12 Error Status Enable** 1: Enable. 0: Masked.
23	RW	**Current Limit Error Status Enable** 1: Enable. 0: Masked.
22	RW	**Data End Bit Error Status Enable** 1: Enable. 0: Masked.
21	RW	**Data CRC Error Status Enable** 1: Enable. 0: Masked.
20	RW	**Data Timeout Error Status Enable** 1: Enable. 0: Masked.
19	RW	**Command Index Error Status Enable** 1: Enable. 0: Masked.
18	RW	**Command End Bit Error Status Enable** 1: Enable. 0: Masked.
17	RW	**Command CRC Error Status Enable** 1: Enable. 0: Masked.
16	RW	**Command Timeout Error Status Enable** 1: Enable. 0: Masked.
15: 9	RW	**Reserved**
8	R	**Card Interrupt Status Enable** 1: Enable. 0: Masked.

to next page

from previous page

7	RW	**Card Removal Status Enable** 1: Enable. 0: Masked.
6	RW	**Card Inserted Status Enable** 1: Enable. 0: Masked.
5	RW	**Buffer Read Ready Status Enable** 1: Enable. 0: Masked.
4	RW	**Buffer Write Ready Status Enable** 1: Enable. 0: Masked.
3	RW	**DMA Interrupt Status Enable** 1: Enable. 0: Masked.
2	RW	**Block Gap Event Status Enable** 1: Enable. 0: Masked.
1	RW	**Transfer Complete Status Enable** 1: Enable. 0: Masked.
0	RW	**Command Complete Status Enable** 1: Enable. 0: Masked.

Offset: 138h **SDIO138: Interrupt Enable Register** **Init = 0**

Bit	R/W	Description
31:26	RW	**Reserved**
25	RW	**ADMA Error Interrupt Enable** 1: Enable. 0: Masked.
24	RW	**Auto CMD12 Error Interrupt Enable** 1: Enable. 0: Masked.
23	RW	**Current Limit Error Interrupt Enable** 1: Enable. 0: Masked.
22	RW	**Data End Bit Error Interrupt Enable** 1: Enable. 0: Masked.
21	RW	**Data CRC Error Interrupt Enable** 1: Enable. 0: Masked.
20	RW	**Data Timeout Error Interrupt Enable** 1: Enable. 0: Masked.
19	RW	**Command Index Error Interrupt Enable** 1: Enable. 0: Masked.

to next page

from previous page

18	RW	**Command End Bit Error Interrupt Enable** 1: Enable. 0: Masked.
17	RW	**Command CRC Error Interrupt Enable** 1: Enable. 0: Masked.
16	RW	**Command Timeout Error Interrupt Enable** 1: Enable. 0: Masked.
15: 9	RW	**Reserved**
8	R	**Card Interrupt Interrupt Enable** 1: Enable. 0: Masked.
7	RW	**Card Removal Interrupt Enable** 1: Enable. 0: Masked.
6	RW	**Card Inserted Interrupt Enable** 1: Enable. 0: Masked.
5	RW	**Buffer Read Ready Interrupt Enable** 1: Enable. 0: Masked.
4	RW	**Buffer Write Ready Interrupt Enable** 1: Enable. 0: Masked.
3	RW	**DMA Interrupt Interrupt Enable** 1: Enable. 0: Masked.
2	RW	**Block Gap Event Interrupt Enable** 1: Enable. 0: Masked.
1	RW	**Transfer Complete Interrupt Enable** 1: Enable. 0: Masked.
0	RW	**Command Complete Interrupt Enable** 1: Enable. 0: Masked.

Offset: 13Ch **SDIO13C: Auto CMD12 Error Status Register** **Init = 0**

Bit	R/W	Description
31: 8	RW	**Reserved**
7	R	**Command Not Issue by Auto CMD12 Error** 1: Command was not executed by SDIO-Host due to previous Auto CMD12 error.
6: 5	RW	**Reserved**
4	R	**Auto CMD12 Index Error** 1: Command Index error occurs in Auto CMD12 response.
3	R	**Auto CMD12 End Bit Error** 1: End bit of Auto CMD12 response is 0.

to next page

from previous page

2	R	**Auto CMD12 CRC Error** 1: CRC error in Auto CMD12 response.
1	R	**Auto CMD12 Timeout Error** 1: No response is returned within 64 SDCLK cycle from the end bit of Auto CMD12.
0	RW	**Auto CMD12 Not Executed** 1: Auto CMD12 can not be issued due to some error.

Offset: 140h **SDIO140: Capability Register** **Init = 0**

Bit	R/W	Description
31:29	RW	**Reserved**
28	R	**64-bits System Bus Support** Always set to 0.
27	RW	**Reserved**
26	R	**Voltage Support 1.8V** 1: 1.8V supported. 0: 1.8V not supported.
25	R	**Voltage Support 3.0V** 1: 3.0V supported. 0: 3.0V not supported.
24	R	**Voltage Support 3.3V** 1: 3.3V supported. 0: 3.3V not supported.
23	R	**Suspend/Resume Support** 1: Suspend/Resume enable. 0: Suspend/Resume disable.
22	R	**SDMA Support** 1: SDMA supported. 0: SDMA not supported.
21	R	**High Speed Support** 1: High Speed supported. 0: High Speed not supported.
20	R	**ADMA1 Support** 1: ADMA1 supported. 0: ADMA1 not supported.
19	R	**ADMA2 Support** 1: ADMA2 supported. 0: ADMA2 not supported.
18	RW	**Reserved**
17:16	R	**Max Block Length** 00b: 512 Bytes 01b: 1024 Bytes 10b: 2048 Bytes 11b: Reserved
15:14	RW	**Reserved**

to next page

from previous page

13: 8	R	**Base Clock Frequency for SD Clock** 00h: Obtain clock information via another method. 01h: 1MHz 02h: 2MHz 03h: 3MHz ... 3Fh: 63MHz
7	R	**Timeout Clock Unit** Unit of SDIO140[5:0] 0: kHz 1: MHz
6	RW	**Reserved**
5: 0	R	**Timeout Clock Frequency** 00h: Obtain clock information via another method. 01h: 1 02h: 2 03h: 3 ... 3Fh: 63

Offset: 148h **SDIO148: Maximum Current Capabilities Register** **Init = 0**

Bit	R/W	Description
31:24	RW	**Reserved**
23:16	R	**Maximum Current for 1.8V** 00h: Obtain current information via another method. 01h: 4mA 02h: 8mA 03h: 12mA ... FFh: 1020mA
15: 8	R	**Maximum Current for 3.0V** 00h: Obtain current information via another method. 01h: 4mA 02h: 8mA 03h: 12mA ... FFh: 1020mA
7: 0	R	**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	Description
31:26	RW	**Reserved**
25	W	**Force ADMA Error Event**
24	W	**Force Auto CMD12 Error Event**

to next page

from previous page

23	W	**Force Current Limit Error Event**
22	W	**Force Data End Bit Error Event**
21	W	**Force Data CRC Error Event**
20	W	**Force Data Timeout Error Event**
19	W	**Force Command Index Error Event**
18	W	**Force Command End Bit Error Event**
17	W	**Force Command CRC Error Event**
16	W	**Force Command Timeout Error Event**
15: 9	W	**Force Reserved**
8	W	**Force Card Interrupt Event**
7	W	**Force Card Removal Event**
6	W	**Force Card Inserted Event**
5	W	**Force Buffer Read Ready Event**
4	W	**Force Buffer Write Ready Event**
3	W	**Force DMA Interrupt Event**
2	W	**Force Block Gap Event Event**
1	W	**Force Transfer Complete Event**
0	W	**Force Command Complete Event**

Offset: 154h **SDIO154: ADMA Error Status Register** **Init = 0**

Bit	R/W	Description
31: 3	RW	**Reserved**
2	RW	**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	**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	Description
31: 0	RW	**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**	**Description**
31: 0	RW	**SDMA System Address**

Offset: 204h	SDIO204: Block Count and Size Register	Init = 0
Bit	**R/W**	**Description**
31:16	RW	**Block Count for Current Transfer** 0000h: 0 0001h: 1 ... FFFFh: 65535
15	RW	**Reserved**
14:12	RW	**Host DMA Buffer Boundary** 000b 4kB 001b 8kB 010b 16kB 011b 32kB 100b 64kB 101b 128kB 110b 256kB 111b 512kB
11: 0	RW	**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**	**Description**
31: 0	RW	**Command Argument** Contain bit [39:8] of command argument

Offset: 20Ch	SDIO20C: Command and Transfer Mode Register	Init = 0
Bit	**R/W**	**Description**
31:30	RW	**Reserved**
29:24	RW	**Command Index** 00h: CMD0 / ACMD0 01h: CMD1 / ACMD1 ... 3Fh: CMD63 / ACMD63
23:22	RW	**Command Type** 11b Abort Command 10b Suspend Command 01b Resume Command 00b Normal Command
21	RW	**Data Present Select** 1: Command which transfer data 0: all other commands
20	RW	**Command Index Check Enable** 1: Check if command index field of the response equal to SDIO20C[29:24].
19	RW	**Command CRC Check Enable** 1: Check if CRC field of the response is valid.

to next page

from previous page

18	RW	**Reserved**
17:16	RW	**Response Type Select** 00b no response 01b 136-bit response 10b 48-bit response 11b 48-bit response with BUSY
15: 6	RW	**Reserved**
5	RW	**Multi/Single Block Select** 1: Multi block data transfer. 0: Single block data transfer.
4	RW	**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	RW	**Reserved**
2	RW	**Auto CMD12 Enable** 1: Enable. 0: Disable.
1	RW	**Block Count Enable** 1: Enable. 0: Disable.
0	RW	**DMA Enable** 1: Enable. 0: Disable.

Offset: 210h **SDIO210: Response #0** **Init = 0**

Bit	R/W	Description
31: 0	R	**Command response[31: 0]**

Offset: 214h **SDIO214: Response #1** **Init = 0**

Bit	R/W	Description
31: 0	R	**Command response[63: 32]**

Offset: 218h **SDIO218: Response #2** **Init = 0**

Bit	R/W	Description
31: 0	R	**Command response[95: 64]**

Offset: 21Ch **SDIO21C: Response #3** **Init = 0**

Bit	R/W	Description
31: 0	R	**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	Description
31: 0	RW	**Buffer Data Port**

Offset: 224h **SDIO224: Present State Register** **Init = 0**

Bit	R/W	Description
31:25	RW	**Reserved**
24	R	**CMD Line Signal Level** Equal to the actual signal level on CMD line of the interface.
23:20	R	**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	R	**Write Protect Switch Pin Level** Equal to the actual signal level on Write Protect line of the interface.
18	R	**Card Detect Switch Pin Level** Equal to the actual signal level on Card Detect line of the interface.
17	R	**Card State Stable** Indicate if SDIO224[18] is stable. 1: SDIO224[18] is stable. 0: SDIO224[18] is unstable.
16	R	**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	RW	**Reserved**
11	R	**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	R	**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.

to next page

from previous page

9	R	**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	R	**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: 3	RW	**Reserved**
2	R	**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	R	**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	R	**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	Description
31:27	RW	**Reserved**
26	RW	**Wakeup Event Enable on SD Card Removal** 1: Enable. 0: Disable.
25	RW	**Wakeup Event Enable on SD Card Inserted** 1: Enable. 0: Disable.
24	RW	**Wakeup Event Enable on SD Card Interrupt** 1: Enable. 0: Disable.
23:20	RW	**Reserved**
19	RW	**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	**Read Wait Control** 1: Enable. 0: Disable.
17	RW	**Continue Request** 1: Restart transfer stopped previously using SDIO228[16]. This bit is cleared automatically.
16	RW	**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.

to next page

from previous page

15: 9	RW	**Reserved**
8	RW	**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	**Card Detect Signal Selection** 1: SDIO228[6]. 0: From slot.
6	RW	**Card Detect Test Level** 1: Card inserted. 0: No card.
5	RW	**Reserved**
4: 3	RW	**DMA Select** 00b: SDMA 01b: ADMA1 10b: ADMA2 11b: Reserved
2	RW	**Hish Speed Enable** 1: Enable 0: Disable
1	RW	**Data Transfer Width** 1: SD4 (4-bits) mode. 0: SD1 (1-bit) mode.
0	RW	**Reserved**

Offset: 22Ch **SDIO22C: Host Control Settings #1 Register** **Init = 0**

Bit	R/W	Description
31:27	RW	**Reserved**
26	RW	**Software Reset for DAT Line** 1: Reset data path, including data buffer and DMA logics. This bit is cleared automatically.
25	RW	**Software Reset for CMD Line** 1: Reset command/response generation and checking. This bit is cleared automatically.
24	RW	**Software Reset for All** 1: Entire SDIO-Host is reset. This bit is cleared automatically.
23:20	RW	**Reserved**
19:16	RW	**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

to next page

from previous page

15: 8	RW	**SDCLK Frequency Select** 80h Period of SDMCLK * 256 40h Period of SDMCLK * 128 20h Period of SDMCLK * 64 10h Period of SDMCLK * 32 08h Period of SDMCLK * 16 04h Period of SDMCLK * 8 02h Period of SDMCLK * 4 01h Period of SDMCLK * 2 00h Forbidden
7: 3	RW	**Reserved**
2	RW	**SD Clock Enable** 1: Enable SD slot clock. 0: Disable SD slot clock.
1	R	**Internal Clock Stable** 1: Indicates clock pin of SDMCLK is stable. 0: Indicates clock pin of SDMCLK is not stable.
0	RW	**Internal Clock Enable** 1: Enable SDMCLK.

Offset: 230h **SDIO230: Interrupt Status Register** **Init = 0**

Bit	R/W	Description
31:26	RW	**Reserved**
25	RW	**ADMA Error** Read 1: Error occurs during ADMA read or write transfer. Write 1 to clear.
24	RW	**Auto CMD12 Error** Read 1: Error occurs during Auto CMD12 command transmition. Write 1 to clear.
23	RW	**Current Limit Error** Read 1: SD Card is not powered due to some failure. Write 1 to clear.
22	RW	**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	**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	**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	**Command Index Error** Read 1: Index error occurs in the command response. Write 1 to clear.

to next page

from previous page

18	RW	**Command End Bit Error** Read 1: Indicate detecting the end bit of a command response is 0. Write 1 to clear.
17	RW	**Command CRC Error** Read 1: Command CRC error occurs. Write 1 to clear.
16	RW	**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	R	**Error Interrupt** 1: One of SDIO230[25:16] is set.
14: 9	RW	**Reserved**
8	R	**Card Interrupt** 1: Indicate the card interrupt.
7	RW	**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	**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	**Buffer Read Ready** Read 1: Indicate data buffer can be read. Write 1 to clear.
4	RW	**Buffer Write Ready** Read 1: Indicate data buffer can be written. Write 1 to clear.
3	RW	**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	**Block Gap Event** Read 1: Indicate read/write transaction is stopped at a block gap. Write 1 to clear.
1	RW	**Transfer Complete** Read 1: Indicate transfer using DAT line is completed. Write 1 to clear.
0	RW	**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	Description
31:26	RW	**Reserved**

to next page

from previous page

25	RW	**ADMA Error Status Enable** 1: Enable. 0: Masked.
24	RW	**Auto CMD12 Error Status Enable** 1: Enable. 0: Masked.
23	RW	**Current Limit Error Status Enable** 1: Enable. 0: Masked.
22	RW	**Data End Bit Error Status Enable** 1: Enable. 0: Masked.
21	RW	**Data CRC Error Status Enable** 1: Enable. 0: Masked.
20	RW	**Data Timeout Error Status Enable** 1: Enable. 0: Masked.
19	RW	**Command Index Error Status Enable** 1: Enable. 0: Masked.
18	RW	**Command End Bit Error Status Enable** 1: Enable. 0: Masked.
17	RW	**Command CRC Error Status Enable** 1: Enable. 0: Masked.
16	RW	**Command Timeout Error Status Enable** 1: Enable. 0: Masked.
15: 9	RW	**Reserved**
8	R	**Card Interrupt Status Enable** 1: Enable. 0: Masked.
7	RW	**Card Removal Status Enable** 1: Enable. 0: Masked.
6	RW	**Card Inserted Status Enable** 1: Enable. 0: Masked.
5	RW	**Buffer Read Ready Status Enable** 1: Enable. 0: Masked.
4	RW	**Buffer Write Ready Status Enable** 1: Enable. 0: Masked.
3	RW	**DMA Interrupt Status Enable** 1: Enable. 0: Masked.

to next page

from previous page

2	RW	**Block Gap Event Status Enable** 1: Enable. 0: Masked.
1	RW	**Transfer Complete Status Enable** 1: Enable. 0: Masked.
0	RW	**Command Complete Status Enable** 1: Enable. 0: Masked.

Offset: 238h **SDIO238: Interrupt Enable Register** **Init = 0**

Bit	R/W	Description
31:26	RW	**Reserved**
25	RW	**ADMA Error Interrupt Enable** 1: Enable. 0: Masked.
24	RW	**Auto CMD12 Error Interrupt Enable** 1: Enable. 0: Masked.
23	RW	**Current Limit Error Interrupt Enable** 1: Enable. 0: Masked.
22	RW	**Data End Bit Error Interrupt Enable** 1: Enable. 0: Masked.
21	RW	**Data CRC Error Interrupt Enable** 1: Enable. 0: Masked.
20	RW	**Data Timeout Error Interrupt Enable** 1: Enable. 0: Masked.
19	RW	**Command Index Error Interrupt Enable** 1: Enable. 0: Masked.
18	RW	**Command End Bit Error Interrupt Enable** 1: Enable. 0: Masked.
17	RW	**Command CRC Error Interrupt Enable** 1: Enable. 0: Masked.
16	RW	**Command Timeout Error Interrupt Enable** 1: Enable. 0: Masked.
15: 9	RW	**Reserved**
8	R	**Card Interrupt Interrupt Enable** 1: Enable. 0: Masked.
7	RW	**Card Removal Interrupt Enable** 1: Enable. 0: Masked.

to next page

from previous page

6	RW	**Card Inserted Interrupt Enable** 1: Enable. 0: Masked.
5	RW	**Buffer Read Ready Interrupt Enable** 1: Enable. 0: Masked.
4	RW	**Buffer Write Ready Interrupt Enable** 1: Enable. 0: Masked.
3	RW	**DMA Interrupt Interrupt Enable** 1: Enable. 0: Masked.
2	RW	**Block Gap Event Interrupt Enable** 1: Enable. 0: Masked.
1	RW	**Transfer Complete Interrupt Enable** 1: Enable. 0: Masked.
0	RW	**Command Complete Interrupt Enable** 1: Enable. 0: Masked.

Offset: 23Ch **SDIO23C: Auto CMD12 Error Status Register** **Init = 0**

Bit	R/W	Description
31: 8	RW	**Reserved**
7	R	**Command Not Issue by Auto CMD12 Error** 1: Command was not executed by SDIO-Host due to previous Auto CMD12 error.
6: 5	RW	**Reserved**
4	R	**Auto CMD12 Index Error** 1: Command Index error occurs in Auto CMD12 response.
3	R	**Auto CMD12 End Bit Error** 1: End bit of Auto CMD12 response is 0.
2	R	**Auto CMD12 CRC Error** 1: CRC error in Auto CMD12 response.
1	R	**Auto CMD12 Timeout Error** 1: No response is returned within 64 SDCLK cycle from the end bit of Auto CMD12.
0	RW	**Auto CMD12 Not Executed** 1: Auto CMD12 can not be issued due to some error.

Offset: 240h **SDIO240: Capability Register** **Init = 0**

Bit	R/W	Description
31:29	RW	**Reserved**
28	R	**64-bits System Bus Support** Always set to 0.
27	RW	**Reserved**
26	R	**Voltage Support 1.8V** 1: 1.8V supported. 0: 1.8V not supported.

to next page

from previous page

25	R	**Voltage Support 3.0V** 1: 3.0V supported. 0: 3.0V not supported.
24	R	**Voltage Support 3.3V** 1: 3.3V supported. 0: 3.3V not supported.
23	R	**Suspend/Resume Support** 1: Suspend/Resume enable. 0: Suspend/Resume disable.
22	R	**SDMA Support** 1: SDMA supported. 0: SDMA not supported.
21	R	**High Speed Support** 1: High Speed supported. 0: High Speed not supported.
20	R	**ADMA1 Support** 1: ADMA1 supported. 0: ADMA1 not supported.
19	R	**ADMA2 Support** 1: ADMA2 supported. 0: ADMA2 not supported.
18	RW	**Reserved**
17:16	R	**Max Block Length** 00b: 512 Bytes 01b: 1024 Bytes 10b: 2048 Bytes 11b: Reserved
15:14	RW	**Reserved**
13: 8	R	**Base Clock Frequency for SD Clock** 00h: Obtain clock information via another method. 01h: 1MHz 02h: 2MHz 03h: 3MHz ... 3Fh: 63MHz
7	R	**Timeout Clock Unit** Unit of SDIO240[5:0] 0: kHz 1: MHz
6	RW	**Reserved**
5: 0	R	**Timeout Clock Frequency** 00h: Obtain clock information via another method. 01h: 1 02h: 2 03h: 3 ... 3Fh: 63

Offset: 248h	SDIO248: Maximum Current Capabilities Register	Init = 0

Bit	R/W	Description
31:24	RW	**Reserved**
23:16	R	**Maximum Current for 1.8V** 00h: Obtain current information via another method. 01h: 4mA 02h: 8mA 03h: 12mA ... FFh: 1020mA
15: 8	R	**Maximum Current for 3.0V** 00h: Obtain current information via another method. 01h: 4mA 02h: 8mA 03h: 12mA ... FFh: 1020mA
7: 0	R	**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	Description
31:26	RW	**Reserved**
25	W	**Force ADMA Error Event**
24	W	**Force Auto CMD12 Error Event**
23	W	**Force Current Limit Error Event**
22	W	**Force Data End Bit Error Event**
21	W	**Force Data CRC Error Event**
20	W	**Force Data Timeout Error Event**
19	W	**Force Command Index Error Event**
18	W	**Force Command End Bit Error Event**
17	W	**Force Command CRC Error Event**
16	W	**Force Command Timeout Error Event**
15: 9	W	**Force Reserved**
8	W	**Force Card Interrupt Event**
7	W	**Force Card Removal Event**
6	W	**Force Card Inserted Event**
5	W	**Force Buffer Read Ready Event**
4	W	**Force Buffer Write Ready Event**
3	W	**Force DMA Interrupt Event**
2	W	**Force Block Gap Event Event**
1	W	**Force Transfer Complete Event**
0	W	**Force Command Complete Event**

Offset: 254h	SDIO254: ADMA Error Status Register	Init = 0
Bit	**R/W**	**Description**
31: 3	RW	**Reserved**
2	RW	**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	**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**	**Description**
31: 0	RW	**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.

35.4 Operation

35.4.1 Card Detection Method

Use slot 0 as example.

- Dedicate Card Detection Pin
  1. Set SDIO0F0[8] to 0.
  2. Interrupt will be generated when card is inserting or removing.
- Share Card Detection with DAT3
  1. Set SDIO0F0[8] to 1.
  2. Set SDIO0F0[9] to 1 just before data transfer.
  3. When data transfer complete, set SDIO0F0[9] to 0 to resume card detection.
  4. Interrupt will be generated only when SDIO0F0[9] is 0. HW will assume card is always inserted otherwise.

36 GPIO Controller

36.1 Overview

AST2500 Integrates one set of Parallel GPIO Controller with maximum 228 control pins, which are 29 sets, to provide general-purpose input/output functions, one set of Serial GPIO Controller with maximum 80 serial input and 80 serial output , and two set of Serial GPIO slave controller which follows SFF-8485. Please reference Section 2.5.

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 up to 80 SGPIO input ports and 80 output ports concurrently 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 maximum 10 drives recording capability for each channel.
- Support 4 interrupt types.

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/AA/AB Data Read Register
GPIO0DC: GPIO_AC 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/AA/AB Command Source 0
GPIO174: GPIO_Y/Z/AA/AB Command Source 1
GPIO178: GPIO_Y/Z/AA/AB Interrupt Enable Register
GPIO17C: GPIO_Y/Z/AA/AB Interrupt Sensitivity Type 0 Register
GPIO180: GPIO_Y/Z/AA/AB Interrupt Sensitivity Type 1 Register
GPIO184: GPIO_Y/Z/AA/AB Interrupt Sensitivity Type 2 Register
GPIO188: GPIO_Y/Z/AA/AB Interrupt Status Register
GPIO18C: GPIO_Y/Z/AA/AB Reset Tolerant Register
GPIO190: GPIO_Y/Z/AA/AB Debounce Setting #1
GPIO194: GPIO_Y/Z/AA/AB Debounce Setting #2
GPIO198: GPIO_Y/Z/AA/AB Input Mask
GPIO1A0: GPIO_AC Command Source 0
GPIO1A4: GPIO_AC Command Source 1
GPIO1A8: GPIO_AC Interrupt Enable Register
GPIO1AC: GPIO_AC Interrupt Sensitivity Type 0 Register
GPIO1B0: GPIO_AC Interrupt Sensitivity Type 1 Register
GPIO1B4: GPIO_AC Interrupt Sensitivity Type 2 Register
GPIO1B8: GPIO_AC Interrupt Status Register
GPIO1BC: GPIO_AC Reset Tolerant Register
GPIO1C0: GPIO_AC Debounce Setting #1
GPIO1C4: GPIO_AC Debounce Setting #2
GPIO1C8: GPIO_AC Input Mask
GPIO1D0: GPIO_A/B/C/D Input Mask
GPIO1D4: GPIO_E/F/G/H Input Mask
GPIO1E0: GPIO_Y/Z/AA/AB Data Value Register
GPIO1E4: GPIO_Y/Z/AA/AB Direction Register
GPIO1E8: GPIO_AC Data Value Register

GPIO1EC: GPIO_AC Direction Register

Serial GPIO implements 25 sets of 32-bit registers, which are listed below, to program the various supported functions including interrupt sensitivity, WDT tolerance, and de-bouncing options. Each register has its own specific offset value, ranging from 0x00 to 0x54h, 0x70h, and 0x74h to derive its physical address location.

Base Address of GPIO = 0x1E78_0000
Register Address of GPIO = (Base Address of GPIO) + Offset

GPIO200: Serial GPIO_A/B/C/D Data Value Register
GPIO204: Serial GPIO_A/B/C/D Interrupt Enable Register
GPIO208: Serial GPIO_A/B/C/D Interrupt Sensitivity Type 0 Register
GPIO20C: Serial GPIO_A/B/C/D Interrupt Sensitivity Type 1 Register
GPIO210: Serial GPIO_A/B/C/D Interrupt Sensitivity Type 2 Register
GPIO214: Serial GPIO_A/B/C/D Interrupt Status Register
GPIO218: Serial GPIO_A/B/C/D Reset Tolerant Register
GPIO21C: Serial GPIO_E/F/G/H Data Value Register
GPIO220: Serial GPIO_E/F/G/H Interrupt Enable Register
GPIO224: Serial GPIO_E/F/G/H Interrupt Sensitivity Type 0 Register
GPIO228: Serial GPIO_E/F/G/H Interrupt Sensitivity Type 1 Register
GPIO22C: Serial GPIO_E/F/G/H Interrupt Sensitivity Type 2 Register
GPIO230: Serial GPIO_E/F/G/H Interrupt Status Register
GPIO234: Serial GPIO_E/F/G/H Reset Tolerant Register
GPIO238: Serial GPIO_I/J Data Value Register
GPIO23C: Serial GPIO_I/J Interrupt Enable Register
GPIO240: Serial GPIO_I/J Interrupt Sensitivity Type 0 Register
GPIO244: Serial GPIO_I/J Interrupt Sensitivity Type 1 Register
GPIO248: Serial GPIO_I/J Interrupt Sensitivity Type 2 Register
GPIO24C: Serial GPIO_I/J Interrupt Status Register
GPIO250: Serial GPIO_I/J Reset Tolerant Register
GPIO254: Serial GPIO Contral Register
GPIO270: Serial GPIO_A/B/C/D Data Read Register
GPIO274: Serial GPIO_E/F/G/H Data Read Register
GPIO278: Serial GPIO_I/J Data Read Register

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. Support 4 interrupt types: End of bit-stream, Target status changing, Matching of target status and pattern, and Matching of SLoad and pattern.

Base Address of GPIO = 0x1E78_0000
Register Address of GPIO = (Base Address of GPIO) + Offset

GPIO300: SGPIO Slave Data from Initiator
GPIO304: SGPIO Slave Data from Target
GPIO308: SGPIO Slave Load from Initiator
GPIO30C: SGPIO Slave Interrupt Enable 0
GPIO310: SGPIO Slave Interrupt Enable 1
GPIO314: SGPIO Slave Interrupt Enable 2
GPIO318: SGPIO Slave Interrupt Status 0
GPIO31C: SGPIO Slave Interrupt Status 1
GPIO320: SGPIO Slave Interrupt Status 2
GPIO380: SGPIO Slave Data from Initiator
GPIO384: SGPIO Slave Data from Target
GPIO388: SGPIO Slave Load from Initiator
GPIO38C: SGPIO Slave Interrupt Enable 0

GPIO390: SGPIO Slave Interrupt Enable 1
GPIO394: SGPIO Slave Interrupt Enable 2
GPIO398: SGPIO Slave Interrupt Status 0
GPIO39C: SGPIO Slave Interrupt Status 1
GPIO3A0: SGPIO Slave Interrupt Status 2

36.2 Features

36.2.1 Parallel GPIO

- Directly connected to APB bus
- Support 8 dedicated and 144 shared GPIO pins
- Programmable reset tolerance option for all GIPO pin.
- Support interrupt triggered by all GPIO pins.
- All input pin is with 0ms/1us/1ms/5ms/10ms de-bouncing logic option.
- Default internal pull-down resistors for each GPIO pins
- Need external pull-up resistors

36.2.2 Serial GPIO

- Directly connected to APB bus
- Programmable reset tolerance option for 80 Serial GIPO pin.
- Support interrupt triggered by 64 Serial GPIO pins.

36.2.3 SGPIO Slave Monitor

- Monitor Bus behavior between Initiator and Target, which follows SPF-8485.
- Maximum 10 drives.
- 4 types of interrupt
  - End of bit-stream
  - Target status changing.
  - Matching of target status and pattern.
  - Matching of SLoad and pattern.

36.3 Registers : Base Address = 0x1E78:0000

36.3.1 Parallel GPIO

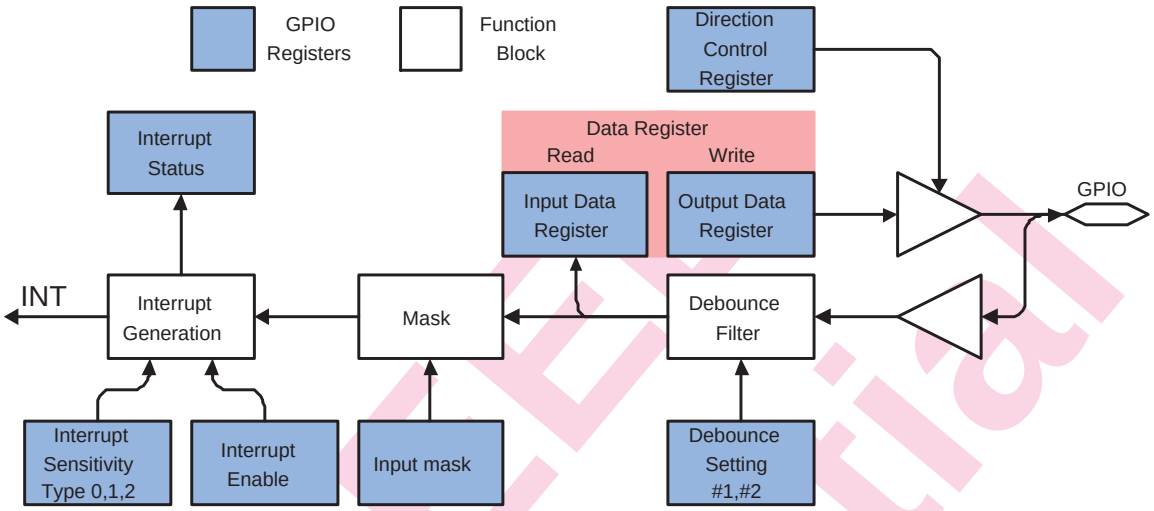


Figure 50: Parallel GPIO Function Block Diagram

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	RW	**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	RW	**Port GPIOT[7:0] direction control** 0: Select input mode 1: Select output mode
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	R	**Port GPIOX[7:0] data register**
23:16	R	**Port GPIOW[7:0] data register**
15:8	RW	**Port GPIOV[7:0] data register**
7 :0	RW	**Port GPIOU[7:0] data register**

Offset: 08Ch	GPIO08C: GPIO_U/V/W/X Direction Register	Init = 0
Bit	**R/W**	**Description**
31:24		**Reserved** GPIOX[7:0] are input only
23:16		**Reserved** GPIOW[7:0] are input only
15:8	RW	**Port GPIOV[7:0] direction control** 0: Select input mode 1: Select output mode
7 :0	RW	**Port GPIOU[7:0] direction control** 0: Select input mode 1: Select output mode

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

to next page

from previous page

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

to next page

from previous page

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/AA/AB Data Read Register	Init = 0
Bit	**R/W**	**Description**
31:28		**Reserved**
27:0	R	**Data written to GPIO1E0.(GPIOAB/GPIOAA/GPIOZ/GPIOY)**

Offset: 0DCh	GPIO0DC: GPIO_AC Data Read Register	Init = 0

Bit	R/W	Description
31:8		**Reserved**
7 :0	R	**Data written to GPIO1E8.(GPIOAC)**

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

to next page

from previous page

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

to next page

from previous page

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.
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 GPIOT[7:0] Command Source 0**
23:17		**Reserved**
16	RW	**Port GPIOS[7:0] Command Source 0**
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 GPIOT[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 GPIOT[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 GPIOT[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 GPIOT[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 GPIOT[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 GPIOT[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

to next page

from previous page

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 GPIOT[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 GPIOT[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 GPIOT[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 GPIOT[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.

to next page

from previous page

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 GPIOU[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**
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 GPIOU[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 GPIOU[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 GPIOU[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 GPIOU[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 GPIOU[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
7 :0	RW	**Port GPIOU[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.

to next page

from previous page

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 GPIOU[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 GPIOU[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 GPIOU[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.
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 GPIOU[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/AA/AB Command Source 0** **Init = 0**

Bit	R/W	Description
31:25		**Reserved**
24	RW	**Port GPIOAB[7:0] Command Source 0**
23:17		**Reserved**
16	RW	**Port GPIOAA[7:0] Command Source 0**
15: 9		**Reserved**

to next page

from previous page

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/AA/AB Command Source 1** **Init = 0**

Bit	R/W	Description
31:25		**Reserved**
24	RW	**Port GPIOAB[7:0] Command Source 1**
23:17		**Reserved**
16	RW	**Port GPIOAA[7:0] Command Source 1**
15: 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/AA/AB Interrupt Enable Register** **Init = 0**

Bit	R/W	Description
31:28		**Reserved**
27:24	RW	**Port GPIOAB[3:0] interrupt enable** 0: Disable interrupt 1: Enable interrupt
23:16	RW	**Port GPIOAA[7:0] interrupt enable** 0: Disable interrupt 1: Enable interrupt
15:8	RW	**Port GPIOZ[7:0] interrupt enable** 0: Disable interrupt 1: Enable interrupt
7 :0	RW	**Port GPIOY[7:0] interrupt enable** 0: Disable interrupt 1: Enable interrupt

Offset: 17Ch	GPIO17C: GPIO_Y/Z/AA/AB Interrupt Sensitivity Type 0 Register	Init = 0
Bit	**R/W**	**Description**
31:28		**Reserved**
27:24	RW	**Port GPIOAB[3:0] 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 GPIOAA[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 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/AA/AB Interrupt Sensitivity Type 1 Register	Init = 0
Bit	**R/W**	**Description**
31:28		**Reserved**
27:24	RW	**Port GPIOAB[3:0] interrupt sensitivity type 1 selection** 0: Select edge trigger mode 1: Select level trigger mode
23:16	RW	**Port GPIOAA[7:0] interrupt sensitivity type 1 selection** 0: Select edge trigger mode 1: Select level trigger mode
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/AA/AB Interrupt Sensitivity Type 2 Register	Init = 0
Bit	**R/W**	**Description**
31:28		**Reserved**
27:24	RW	**Port GPIOAB[3:0] interrupt sensitivity type 2 selection** 0: Select edge or level trigger mode 1: Select dual-edge trigger mode
23:16	RW	**Port GPIOAA[7:0] interrupt sensitivity type 2 selection** 0: Select edge or level trigger mode 1: Select dual-edge trigger mode
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/AA/AB Interrupt Status Register	Init = 0

Bit	R/W	Description
31:28		**Reserved**
27:24	RW	**Port GPIOAB[3:0] 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 GPIOAA[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 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/AA/AB Reset Tolerant Register	Init = 0

Bit	R/W	Description
31:28		**Reserved**
27:24	RW	**Port GPIOAB[3:0] 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.
23:16	RW	**Port GPIOAA[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.
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/AA/AB Debounce Setting Register #1	Init = 0

Bit	R/W	Description
31:28		**Reserved**
27:24	RW	**Port GPIOAB[3:0] debounce setting register #1**
23:16	RW	**Port GPIOAA[7:0] debounce setting register #1**
15:8	RW	**Port GPIOZ[7:0] debounce setting register #1**

to next page

from previous page

7 :0	RW	**Port GPIOY[7:0] debounce setting register #1**

Offset: 194h **GPIO194: GPIO_Y/Z/AA/AB Debounce Setting Register #2** **Init = 0**

Bit	R/W	Description
31:28		**Reserved**
27:24	RW	**Port GPIOAB[3:0] debounce setting register #2**
23:16	RW	**Port GPIOAA[7:0] debounce setting register #2**
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/AA/AB Input Mask Register** **Init = 0**

Bit	R/W	Description
31:28		**Reserved**
27:24	RW	**Port GPIOAB[3:0] input mask** 0: Read from GPIO080 will be updated. 1: Read from GPIO080 will not be updated.
23:16	RW	**Port GPIOAA[7:0] input mask** 0: Read from GPIO080 will be updated. 1: Read from GPIO080 will not be updated.
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: 1A0h **GPIO1A0: GPIO_AC Command Source 0** **Init = 0**

Bit	R/W	Description
31: 1		**Reserved**
0	RW	**Port GPIOAC[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: 1A4h **GPIO1A4: GPIO_AC Command Source 1** **Init = 0**

Bit	R/W	Description
31: 1		**Reserved**
0	RW	**Port GPIOAC[7:0] Command Source 1**

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: 1A8h **GPIO1A8: GPIO_AC Interrupt Enable Register** **Init = 0**

Bit	R/W	Description
31:8		**Reserved**
7 :0	RW	**Port GPIOAC[7:0] interrupt enable** 0: Disable interrupt 1: Enable interrupt

Offset: 1ACh **GPIO1AC: GPIO_AC Interrupt Sensitivity Type 0 Register** **Init = 0**

Bit	R/W	Description
31:8		**Reserved**
7 :0	RW	**Port GPIOAC[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: 1B0h **GPIO1B0: GPIO_AC Interrupt Sensitivity Type 1 Register** **Init = 0**

Bit	R/W	Description
31:8		**Reserved**
7 :0	RW	**Port GPIOAC[7:0] interrupt sensitivity type 1 selection** 0: Select edge trigger mode 1: Select level trigger mode

Offset: 1B4h **GPIO1B4: GPIO_AC Interrupt Sensitivity Type 2 Register** **Init = 0**

Bit	R/W	Description
31:8		**Reserved**
7 :0	RW	**Port GPIOAC[7:0] interrupt sensitivity type 2 selection** 0: Select edge or level trigger mode 1: Select dual-edge trigger mode

Offset: 1B8h **GPIO1B8: GPIO_AC Interrupt Status Register** **Init = 0**

Bit	R/W	Description
31:8		**Reserved**
7 :0	RW	**Port GPIOAC[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: 1BCh	GPIO1BC: GPIO_AC Reset Tolerant Register	Init = 0
Bit	**R/W**	**Description**
31:8		**Reserved**
7 :0	RW	**Port GPIOAC[7:0] WDT reset tolerance enable** 0: GPIO1E8 and GPIO1EC registers will be reset by WDT reset 1: GPIO1E8 and GPIO1EC registers will not be reset by WDT reset Each GPIO pin can be individually programmed to be WDT reset tolerant or not.

Offset: 1C0h	GPIO1C0: GPIO_AC Debounce Setting Register #1	Init = 0
Bit	**R/W**	**Description**
31:8		**Reserved**
7 :0	RW	**Port GPIOAC[7:0] debounce setting register #1**

Offset: 1C4h	GPIO1C4: GPIO_AC Debounce Setting Register #2	Init = 0
Bit	**R/W**	**Description**
31:8		**Reserved**
7 :0	RW	**Port GPIOAC[7:0] debounce setting register #2**

Offset: 1C8h	GPIO1C8: GPIO_AC Input Mask Register	Init = 0
Bit	**R/W**	**Description**
31:8		**Reserved**
7 :0	RW	**Port GPIOAC[7:0] input mask** 0: Read from GPIO1E8 will be updated. 1: Read from GPIO1E8 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.
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.

to next page

from previous page

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/AA/AB Data Value Register 0**	**Init = 0**
Bit	**R/W**	**Description**
31:28		**Reserved**
27:24	RW	**Port GPIOAB[3:0] data register**
23:16	RW	**Port GPIOAA[7:0] data register**
15:8	RW	**Port GPIOZ[7:0] data register**
7 :0	RW	**Port GPIOY[7:0] data register**

Offset: 1E4h	**GPIO1E4: GPIO_Y/Z/AA/AB Direction Register 0**	**Init = 0**
Bit	**R/W**	**Description**
31:28		**Reserved**
27:24	RW	**Port GPIOAB[3:0] direction control** 0: Select input mode 1: Select output mode
23:16	RW	**Port GPIOAA[7:0] direction control** 0: Select input mode 1: Select output mode
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: 1E8h	**GPIO1E8: GPIO_AC Data Value Register 0**	**Init = 0**
Bit	**R/W**	**Description**
31:8		**Reserved**
7 :0	RW	**Port GPIOAC[7:0] data register**

Offset: 1ECh	**GPIO1EC: GPIO_AC Direction Register 0**	**Init = 0**
Bit	**R/W**	**Description**
31:8		**Reserved**
7 :0	RW	**Port GPIOAC[7:0] direction control** 0: Select input mode 1: Select output mode

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
GPIOT	O	O	O
GPIOU	O	O	O
GPIOV	O	O	O
GPIOW	O	O	O
GPIOX	O	O	O
GPIOY	O	O	O
GPIOZ	O	O	O
GPIOAA	O	O	O
GPIOAB	O	O	O
GPIOAC	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

to next page

from previous page

GPIO020	O	O (If GPIO03C is set)
GPIO024	O	O (If GPIO03C is set)
GPIO028	O	O
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)

to next page

from previous page

GPIO0DC	O	O (If GPIO1BC is set)
GPIO0E0	O	O
GPIO0E4	O	O
GPIO0E8	O	O
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

to next page

from previous page

GPIO1A0	O	O
GPIO1A4	O	O
GPIO1A8	O	O
GPIO1AC	O	O
GPIO1B0	O	O
GPIO1B4	O	O
GPIO1B8	O	O
GPIO1BC	O	X
GPIO1C0	O	O
GPIO1C4	O	O
GPIO1C8	O	O
GPIO1D0	O	O
GPIO1D4	O	O
GPIO1E0	O	O (If GPIO18C is set)
GPIO1E4	O	O (If GPIO18C is set)
GPIO1E8	O	O (If GPIO1BC is set)
GPIO1EC	O	O (If GPIO1BC 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
GPIOT	O	O
GPIOU	O	O
GPIOV	O	O
GPIOW	O	X
GPIOX	O	X
GPIOY	O	O
GPIOZ	O	O
GPIOAA	O	O
GPIOAB	O	O
GPIOAC	O	O

36.3.2 Serial GPIO

Offset: 200h	GPIO200: Serial GPIO_A/B/C/D 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: 204h	GPIO204: Serial GPIO_A/B/C/D 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: 208h	GPIO208: Serial GPIO_A/B/C/D 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: 20Ch	GPIO20C: Serial GPIO_A/B/C/D 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: 210h GPIO210: Serial GPIO_A/B/C/D 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: 214h GPIO214: Serial GPIO_A/B/C/D 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: 218h	GPIO218: Serial GPIO_A/B/C/D Reset Tolerant Register	Init = 0
Bit	**R/W**	**Description**
31:24	RW	**Port Serial GPIOD[7:0] WDT reset tolerance enable** 0: GPIO200 registers will be reset by WDT reset 1: GPIO200 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: GPIO200 registers will be reset by WDT reset 1: GPIO200 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: GPIO200 registers will be reset by WDT reset 1: GPIO200 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: GPIO200 registers will be reset by WDT reset 1: GPIO200 registers will not be reset by WDT reset Each Serial GPIO pin can be individually programmed to be WDT reset tolerant or not.

Offset: 21Ch	GPIO21C: Serial GPIO_E/F/G/H 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: 220h	GPIO220: Serial GPIO_E/F/G/H 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: 224h	GPIO224: Serial GPIO_E/F/G/H 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: 228h GPIO228: Serial GPIO_E/F/G/H 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: 22Ch GPIO22C: Serial GPIO_E/F/G/H 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: 230h GPIO230: Serial GPIO_E/F/G/H 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
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

to next page

from previous page

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: 234h **GPIO234: Serial GPIO_E/F/G/H Reset Tolerant Register** **Init = 0**

Bit	R/W	Description
31:24	RW	**Port Serial GPIOH[7:0] WDT reset tolerance enable** 0: GPIO21C registers will be reset by WDT reset 1: GPIO21C 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: GPIO21C registers will be reset by WDT reset 1: GPIO21C 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: GPIO21C registers will be reset by WDT reset 1: GPIO21C 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: GPIO21C registers will be reset by WDT reset 1: GPIO21C registers will not be reset by WDT reset Each Serial GPIO pin can be individually programmed to be WDT reset tolerant or not.

Offset: 238h **GPIO238: Serial GPIO_I/J 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: 23Ch **GPIO23C: Serial GPIO_I/J 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
7 :0	RW	**Port Serial GPIOI[7:0] interrupt enable** 0: Disable interrupt 1: Enable interrupt

Offset: 240h	GPIO240: Serial GPIO_I/J 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: 244h	GPIO244: Serial GPIO_I/J 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: 248h	GPIO248: Serial GPIO_I/J 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: 24Ch	GPIO24C: Serial GPIO_I/J 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: 250h	GPIO250: Serial GPIO_I/J Reset Tolerant Register	Init = 0
Bit	**R/W**	**Description**
31:16	R	**Reserved**

to next page

from previous page

15:8	RW	**Port Serial GPIOJ[7:0] WDT reset tolerance enable** 0: GPIO238 registers will be reset by WDT reset 1: GPIO238 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: GPIO238 registers will be reset by WDT reset 1: GPIO238 registers will not be reset by WDT reset Each Serial GPIO pin can be individually programmed to be WDT reset tolerant or not.

Offset: 254h	**GPIO254: Serial GPIO Configuration Register**	**Init = 0**
Bit	**R/W**	**Description**
31:16	RW	**Serial GPIO clock division** Serial GPIO clock period = period of PCLK * 2 * (GPIO254[31:16] + 1)
15:10		**Reserved**
9 :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: 270h	**GPIO270: Serial GPIO_A/B/C/D Data Read Register**	**Init = 0**
Bit	**R/W**	**Description**
31: 0	R	**Data written to GPIO200**

Offset: 274h	**GPIO274: Serial GPIO_E/F/G/H Data Read Register**	**Init = 0**
Bit	**R/W**	**Description**
31: 0	R	**Data written to GPIO21C**

Offset: 278h	**GPIO278: Serial GPIO_I/J Data Read Register**	**Init = 0**
Bit	**R/W**	**Description**
31: 0	R	**Data written to GPIO238**

36.3.3 SGPIO Slave

Offset: 300h	GPIO300: SGPIO Slave Initiator Data Register	Init = 0
Bit	**R/W**	**Description**
31:30		**Reserved**
29:0	R	**SGPIO Slave Initiator Data Register**

Offset: 304h	GPIO304: SGPIO Slave Target Data Register	Init = 0
Bit	**R/W**	**Description**
31:30		**Reserved**
29:0	R	**SGPIO Slave Target Data Register**

Offset: 308h	GPIO308: SGPIO Slave Initiator Vendor-Specific Pattern Register	Init = 0
Bit	**R/W**	**Description**
31:20		**Reserved**
19:16	RW	**Pattern for SLoad matching. Valid when GPIO314[26] is 1.**
15:4		**Reserved**
3 :0	R	**SGPIO Slave Initiator Vendor-Specific Pattern Register**

Offset: 30Ch	GPIO30C: SGPIO Slave Interrupt Enable 0	Init = 0
Bit	**R/W**	**Description**
31:24	RW	**Interrupt enable for device 3 matching** Bit 7 : matching 0x7. Bit 6 : matching 0x6. ... Bit 1 : matching 0x1. Bit 0 : matching 0x0. 0: Disable interrupt 1: Enable interrupt
23:16	RW	**Interrupt enable for device 2 matching** Bit 7 : matching 0x7. Bit 6 : matching 0x6. ... Bit 1 : matching 0x1. Bit 0 : matching 0x0. 0: Disable interrupt 1: Enable interrupt
15: 8	RW	**Interrupt enable for device 1 matching** Bit 7 : matching 0x7. Bit 6 : matching 0x6. ... Bit 1 : matching 0x1. Bit 0 : matching 0x0. 0: Disable interrupt 1: Enable interrupt

to next page

from previous page

7: 0	RW	**Interrupt enable for device 0 matching** Bit 7 : matching 0x7. Bit 6 : matching 0x6. ... Bit 1 : matching 0x1. Bit 0 : matching 0x0. 0: Disable interrupt 1: Enable interrupt

Offset: 310h **GPIO310: SGPIO Slave Interrupt Enable 1** **Init = 0**

Bit	R/W	Description
31:24	RW	**Interrupt enable for device 7 matching** Bit 7 : matching 0x7. Bit 6 : matching 0x6. ... Bit 1 : matching 0x1. Bit 0 : matching 0x0. 0: Disable interrupt 1: Enable interrupt
23:16	RW	**Interrupt enable for device 6 matching** Bit 7 : matching 0x7. Bit 6 : matching 0x6. ... Bit 1 : matching 0x1. Bit 0 : matching 0x0. 0: Disable interrupt 1: Enable interrupt
15: 8	RW	**Interrupt enable for device 5 matching** Bit 7 : matching 0x7. Bit 6 : matching 0x6. ... Bit 1 : matching 0x1. Bit 0 : matching 0x0. 0: Disable interrupt 1: Enable interrupt
7: 0	RW	**Interrupt enable for device 4 matching** Bit 7 : matching 0x7. Bit 6 : matching 0x6. ... Bit 1 : matching 0x1. Bit 0 : matching 0x0. 0: Disable interrupt 1: Enable interrupt

Offset: 314h **GPIO314: SGPIO Slave Interrupt Enable 2** **Init = 0**

Bit	R/W	Description
31	RW	**Interrupt enable for end of bit-stream** 0: Disable interrupt 1: Enable interrupt
30:27		**Reserved**
26	RW	**Interrupt enable for SLoad matching**

to next page

from previous page

25	RW	**Interrupt enable for device 9 changing** 0: Disable interrupt 1: Enable interrupt
24	RW	**Interrupt enable for device 8 changing** 0: Disable interrupt 1: Enable interrupt
23	RW	**Interrupt enable for device 7 changing** 0: Disable interrupt 1: Enable interrupt
22	RW	**Interrupt enable for device 6 changing** 0: Disable interrupt 1: Enable interrupt
21	RW	**Interrupt enable for device 5 changing** 0: Disable interrupt 1: Enable interrupt
20	RW	**Interrupt enable for device 4 changing** 0: Disable interrupt 1: Enable interrupt
19	RW	**Interrupt enable for device 3 changing** 0: Disable interrupt 1: Enable interrupt
18	RW	**Interrupt enable for device 2 changing** 0: Disable interrupt 1: Enable interrupt
17	RW	**Interrupt enable for device 1 changing** 0: Disable interrupt 1: Enable interrupt
16	RW	**Interrupt enable for device 0 changing** 0: Disable interrupt 1: Enable interrupt
15: 8	RW	**Interrupt enable for device 9 matching** Bit 7 : matching 0x7. Bit 6 : matching 0x6. ... Bit 1 : matching 0x1. Bit 0 : matching 0x0. 0: Disable interrupt 1: Enable interrupt
7: 0	RW	**Interrupt enable for device 8 matching** Bit 7 : matching 0x7. Bit 6 : matching 0x6. ... Bit 1 : matching 0x1. Bit 0 : matching 0x0. 0: Disable interrupt 1: Enable interrupt

Offset: 318h	GPIO318: SGPIO Slave Interrupt Status 0	Init = 0

Bit	R/W	Description
31:24	RW	**Interrupt status for device 3 matching** Bit 7 : matching 0x7. Bit 6 : matching 0x6. ... Bit 1 : matching 0x1. Bit 0 : matching 0x0. Read 0: No interrupt pending Read 1: interrupt pending Write 0: No operation Write 1: Clear interrupt status flag
23:16	RW	**Interrupt status for device 2 matching** Bit 7 : matching 0x7. Bit 6 : matching 0x6. ... Bit 1 : matching 0x1. Bit 0 : matching 0x0. Read 0: No interrupt pending Read 1: interrupt pending Write 0: No operation Write 1: Clear interrupt status flag
15: 8	RW	**Interrupt status for device 1 matching** Bit 7 : matching 0x7. Bit 6 : matching 0x6. ... Bit 1 : matching 0x1. Bit 0 : matching 0x0. Read 0: No interrupt pending Read 1: interrupt pending Write 0: No operation Write 1: Clear interrupt status flag
7: 0	RW	**Interrupt status for device 0 matching** Bit 7 : matching 0x7. Bit 6 : matching 0x6. ... Bit 1 : matching 0x1. Bit 0 : matching 0x0. Read 0: No interrupt pending Read 1: interrupt pending Write 0: No operation Write 1: Clear interrupt status flag

Offset: 31Ch	GPIO31C: SGPIO Slave Interrupt Status 1	Init = 0

Bit	R/W	Description
31:24	RW	**Interrupt status for device 7 matching** Bit 7 : matching 0x7. Bit 6 : matching 0x6. ... Bit 1 : matching 0x1. Bit 0 : matching 0x0. Read 0: No interrupt pending Read 1: interrupt pending Write 0: No operation Write 1: Clear interrupt status flag
23:16	RW	**Interrupt status for device 6 matching** Bit 7 : matching 0x7. Bit 6 : matching 0x6. ... Bit 1 : matching 0x1. Bit 0 : matching 0x0. Read 0: No interrupt pending Read 1: interrupt pending Write 0: No operation Write 1: Clear interrupt status flag
15: 8	RW	**Interrupt status for device 5 matching** Bit 7 : matching 0x7. Bit 6 : matching 0x6. ... Bit 1 : matching 0x1. Bit 0 : matching 0x0. Read 0: No interrupt pending Read 1: interrupt pending Write 0: No operation Write 1: Clear interrupt status flag
7: 0	RW	**Interrupt status for device 4 matching** Bit 7 : matching 0x7. Bit 6 : matching 0x6. ... Bit 1 : matching 0x1. Bit 0 : matching 0x0. Read 0: No interrupt pending Read 1: interrupt pending Write 0: No operation Write 1: Clear interrupt status flag

Offset: 320h	GPIO320: SGPIO Slave Interrupt Status 2	Init = 0

Bit	R/W	Description
31	RW	**Interrupt status for end of bit-stream** Read 0: No interrupt pending Read 1: interrupt pending Write 0: No operation Write 1: Clear interrupt status flag
30:27		**Reserved**
26	RW	**Interrupt status for SLoad matching**

to next page

from previous page

25	RW	**Interrupt status for device 9 changing** Read 0: No interrupt pending Read 1: interrupt pending Write 0: No operation Write 1: Clear interrupt status flag
24	RW	**Interrupt status for device 8 changing** Read 0: No interrupt pending Read 1: interrupt pending Write 0: No operation Write 1: Clear interrupt status flag
23	RW	**Interrupt status for device 7 changing** Read 0: No interrupt pending Read 1: interrupt pending Write 0: No operation Write 1: Clear interrupt status flag
22	RW	**Interrupt status for device 6 changing** Read 0: No interrupt pending Read 1: interrupt pending Write 0: No operation Write 1: Clear interrupt status flag
21	RW	**Interrupt status for device 5 changing** Read 0: No interrupt pending Read 1: interrupt pending Write 0: No operation Write 1: Clear interrupt status flag
20	RW	**Interrupt status for device 4 changing** Read 0: No interrupt pending Read 1: interrupt pending Write 0: No operation Write 1: Clear interrupt status flag
19	RW	**Interrupt status for device 3 changing** Read 0: No interrupt pending Read 1: interrupt pending Write 0: No operation Write 1: Clear interrupt status flag
18	RW	**Interrupt status for device 2 changing** Read 0: No interrupt pending Read 1: interrupt pending Write 0: No operation Write 1: Clear interrupt status flag
17	RW	**Interrupt status for device 1 changing** Read 0: No interrupt pending Read 1: interrupt pending Write 0: No operation Write 1: Clear interrupt status flag
16	RW	**Interrupt status for device 0 changing** 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

15: 8	RW	**Interrupt status for device 9 matching** Bit 7 : matching 0x7. Bit 6 : matching 0x6. ... Bit 1 : matching 0x1. Bit 0 : matching 0x0. Read 0: No interrupt pending Read 1: interrupt pending Write 0: No operation Write 1: Clear interrupt status flag
7: 0	RW	**Interrupt status for device 8 matching** Bit 7 : matching 0x7. Bit 6 : matching 0x6. ... Bit 1 : matching 0x1. Bit 0 : matching 0x0. Read 0: No interrupt pending Read 1: interrupt pending Write 0: No operation Write 1: Clear interrupt status flag

Offset: 380h **GPIO380: SGPIO Slave Initiator Data Register** **Init = 0**

Bit	R/W	Description
31:30		**Reserved**
29:0	R	**SGPIO Slave Initiator Data Register**

Offset: 384h **GPIO384: SGPIO Slave Target Data Register** **Init = 0**

Bit	R/W	Description
31:30		**Reserved**
29:0	R	**SGPIO Slave Target Data Register**

Offset: 388h **GPIO388: SGPIO Slave Initiator Vendor-Specific Pattern Register** **Init = 0**

Bit	R/W	Description
31:20		**Reserved**
19:16	RW	**Pattern for SLoad matching. Valid when GPIO314[26] is 1.**
15:4		**Reserved**
3 :0	R	**SGPIO Slave Initiator Vendor-Specific Pattern Register**

Offset: 38Ch **GPIO38C: SGPIO Slave Interrupt Enable 0** **Init = 0**

Bit	R/W	Description
31:24	RW	**Interrupt enable for device 3 matching** Bit 7 : matching 0x7. Bit 6 : matching 0x6. ... Bit 1 : matching 0x1. Bit 0 : matching 0x0. 0: Disable interrupt 1: Enable interrupt

to next page

from previous page

23:16	RW	**Interrupt enable for device 2 matching** Bit 7 : matching 0x7. Bit 6 : matching 0x6. ... Bit 1 : matching 0x1. Bit 0 : matching 0x0. 0: Disable interrupt 1: Enable interrupt
15: 8	RW	**Interrupt enable for device 1 matching** Bit 7 : matching 0x7. Bit 6 : matching 0x6. ... Bit 1 : matching 0x1. Bit 0 : matching 0x0. 0: Disable interrupt 1: Enable interrupt
7: 0	RW	**Interrupt enable for device 0 matching** Bit 7 : matching 0x7. Bit 6 : matching 0x6. ... Bit 1 : matching 0x1. Bit 0 : matching 0x0. 0: Disable interrupt 1: Enable interrupt

Offset: 390h **GPIO390: SGPIO Slave Interrupt Enable 1** **Init = 0**

Bit	R/W	Description
31:24	RW	**Interrupt enable for device 7 matching** Bit 7 : matching 0x7. Bit 6 : matching 0x6. ... Bit 1 : matching 0x1. Bit 0 : matching 0x0. 0: Disable interrupt 1: Enable interrupt
23:16	RW	**Interrupt enable for device 6 matching** Bit 7 : matching 0x7. Bit 6 : matching 0x6. ... Bit 1 : matching 0x1. Bit 0 : matching 0x0. 0: Disable interrupt 1: Enable interrupt
15: 8	RW	**Interrupt enable for device 5 matching** Bit 7 : matching 0x7. Bit 6 : matching 0x6. ... Bit 1 : matching 0x1. Bit 0 : matching 0x0. 0: Disable interrupt 1: Enable interrupt

to next page

from previous page

7: 0	RW	**Interrupt enable for device 4 matching** Bit 7 : matching 0x7. Bit 6 : matching 0x6. ... Bit 1 : matching 0x1. Bit 0 : matching 0x0. 0: Disable interrupt 1: Enable interrupt

Offset: 394h **GPIO394: SGPIO Slave Interrupt Enable 2** **Init = 0**

Bit	R/W	Description
31	RW	**Interrupt enable for end of bit-stream** 0: Disable interrupt 1: Enable interrupt
30:27		**Reserved**
26	RW	**Interrupt enable for SLoad matching**
25	RW	**Interrupt enable for device 9 changing** 0: Disable interrupt 1: Enable interrupt
24	RW	**Interrupt enable for device 8 changing** 0: Disable interrupt 1: Enable interrupt
23	RW	**Interrupt enable for device 7 changing** 0: Disable interrupt 1: Enable interrupt
22	RW	**Interrupt enable for device 6 changing** 0: Disable interrupt 1: Enable interrupt
21	RW	**Interrupt enable for device 5 changing** 0: Disable interrupt 1: Enable interrupt
20	RW	**Interrupt enable for device 4 changing** 0: Disable interrupt 1: Enable interrupt
19	RW	**Interrupt enable for device 3 changing** 0: Disable interrupt 1: Enable interrupt
18	RW	**Interrupt enable for device 2 changing** 0: Disable interrupt 1: Enable interrupt
17	RW	**Interrupt enable for device 1 changing** 0: Disable interrupt 1: Enable interrupt
16	RW	**Interrupt enable for device 0 changing** 0: Disable interrupt 1: Enable interrupt

to next page

from previous page

15: 8	RW	**Interrupt enable for device 9 matching** Bit 7 : matching 0x7. Bit 6 : matching 0x6. ... Bit 1 : matching 0x1. Bit 0 : matching 0x0. 0: Disable interrupt 1: Enable interrupt
7: 0	RW	**Interrupt enable for device 8 matching** Bit 7 : matching 0x7. Bit 6 : matching 0x6. ... Bit 1 : matching 0x1. Bit 0 : matching 0x0. 0: Disable interrupt 1: Enable interrupt

Offset: 398h **GPIO398: SGPIO Slave Interrupt Status 0** **Init = 0**

Bit	R/W	Description
31:24	RW	**Interrupt status for device 3 matching** Bit 7 : matching 0x7. Bit 6 : matching 0x6. ... Bit 1 : matching 0x1. Bit 0 : matching 0x0. Read 0: No interrupt pending Read 1: interrupt pending Write 0: No operation Write 1: Clear interrupt status flag
23:16	RW	**Interrupt status for device 2 matching** Bit 7 : matching 0x7. Bit 6 : matching 0x6. ... Bit 1 : matching 0x1. Bit 0 : matching 0x0. Read 0: No interrupt pending Read 1: interrupt pending Write 0: No operation Write 1: Clear interrupt status flag
15: 8	RW	**Interrupt status for device 1 matching** Bit 7 : matching 0x7. Bit 6 : matching 0x6. ... Bit 1 : matching 0x1. Bit 0 : matching 0x0. 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	**Interrupt status for device 0 matching** Bit 7 : matching 0x7. Bit 6 : matching 0x6. ... Bit 1 : matching 0x1. Bit 0 : matching 0x0. Read 0: No interrupt pending Read 1: interrupt pending Write 0: No operation Write 1: Clear interrupt status flag

Offset: 39Ch **GPIO39C: SGPIO Slave Interrupt Status 1** **Init = 0**

Bit	R/W	Description
31:24	RW	**Interrupt status for device 7 matching** Bit 7 : matching 0x7. Bit 6 : matching 0x6. ... Bit 1 : matching 0x1. Bit 0 : matching 0x0. Read 0: No interrupt pending Read 1: interrupt pending Write 0: No operation Write 1: Clear interrupt status flag
23:16	RW	**Interrupt status for device 6 matching** Bit 7 : matching 0x7. Bit 6 : matching 0x6. ... Bit 1 : matching 0x1. Bit 0 : matching 0x0. Read 0: No interrupt pending Read 1: interrupt pending Write 0: No operation Write 1: Clear interrupt status flag
15: 8	RW	**Interrupt status for device 5 matching** Bit 7 : matching 0x7. Bit 6 : matching 0x6. ... Bit 1 : matching 0x1. Bit 0 : matching 0x0. Read 0: No interrupt pending Read 1: interrupt pending Write 0: No operation Write 1: Clear interrupt status flag
7: 0	RW	**Interrupt status for device 4 matching** Bit 7 : matching 0x7. Bit 6 : matching 0x6. ... Bit 1 : matching 0x1. Bit 0 : matching 0x0. Read 0: No interrupt pending Read 1: interrupt pending Write 0: No operation Write 1: Clear interrupt status flag

Offset: 3A0h	GPIO3A0: SGPIO Slave Interrupt Status 2	Init = 0
Bit	**R/W**	**Description**
31	RW	**Interrupt status for end of bit-stream** Read 0: No interrupt pending Read 1: interrupt pending Write 0: No operation Write 1: Clear interrupt status flag
30:27		**Reserved**
26	RW	**Interrupt status for SLoad matching**
25	RW	**Interrupt status for device 9 changing** Read 0: No interrupt pending Read 1: interrupt pending Write 0: No operation Write 1: Clear interrupt status flag
24	RW	**Interrupt status for device 8 changing** Read 0: No interrupt pending Read 1: interrupt pending Write 0: No operation Write 1: Clear interrupt status flag
23	RW	**Interrupt status for device 7 changing** Read 0: No interrupt pending Read 1: interrupt pending Write 0: No operation Write 1: Clear interrupt status flag
22	RW	**Interrupt status for device 6 changing** Read 0: No interrupt pending Read 1: interrupt pending Write 0: No operation Write 1: Clear interrupt status flag
21	RW	**Interrupt status for device 5 changing** Read 0: No interrupt pending Read 1: interrupt pending Write 0: No operation Write 1: Clear interrupt status flag
20	RW	**Interrupt status for device 4 changing** Read 0: No interrupt pending Read 1: interrupt pending Write 0: No operation Write 1: Clear interrupt status flag
19	RW	**Interrupt status for device 3 changing** Read 0: No interrupt pending Read 1: interrupt pending Write 0: No operation Write 1: Clear interrupt status flag
18	RW	**Interrupt status for device 2 changing** 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

17	RW	**Interrupt status for device 1 changing** Read 0: No interrupt pending Read 1: interrupt pending Write 0: No operation Write 1: Clear interrupt status flag
16	RW	**Interrupt status for device 0 changing** Read 0: No interrupt pending Read 1: interrupt pending Write 0: No operation Write 1: Clear interrupt status flag
15: 8	RW	**Interrupt status for device 9 matching** Bit 7 : matching 0x7. Bit 6 : matching 0x6. ... Bit 1 : matching 0x1. Bit 0 : matching 0x0. Read 0: No interrupt pending Read 1: interrupt pending Write 0: No operation Write 1: Clear interrupt status flag
7: 0	RW	**Interrupt status for device 8 matching** Bit 7 : matching 0x7. Bit 6 : matching 0x6. ... Bit 1 : matching 0x1. Bit 0 : matching 0x0. Read 0: No interrupt pending Read 1: interrupt pending Write 0: No operation Write 1: Clear interrupt status flag

36.4 Programming Guide

36.4.1 LPC port80h direct to GPIO

In AST2500 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.

36.4.2 LPC port80h direct to SGPIO

In AST2500 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 GPIO254[9:6] to larger than or equal to 0x1.
   (b) Set GPIO254[0] to 1 to enable SGPIO.
3. Set SuperIO
   (a) Set SIOR7_30h to 0x40.

36.4.3 Parallel GPIO output driving mode

For all parallel GPIO pins of AST2500, 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

37 Real Time Clock (RTC)

37.1 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

37.2 Features

- Directly connected to APB bus
- Clock source is divided from 24MHz clock input
- Precision ≈ 50ppm (24MHz 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

37.3 Registers : Base Address = 0x1E78:1000

Offset: 00h	RTC00: Counter Status Register #1	Init = X

Bit	R/W	Description
31:29		**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		**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		**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		**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		**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		**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		**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		**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		**Reserved (0)**
5	RW	**Wakeup alarm status** 1: Alarmed 0: Idle To clear this bit, set RTC10[8] = 0 for at least 3 us.

to next page

from previous page

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.		

37.4 Operation

37.4.1 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

37.4.2 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.

38 Timer Controller

38.1 Overview

Timer Controller (TMC) includes 8 sets of 32-bit decrement counters, based on either APB clock 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.

TMC totally implements 34 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 0x8Ch, to derive its physical address location.

Base address of Timer = 0x1E78_2000
Physical address = (Base address of Timer) + Offset

TMC00: Counter #1 Status Register
TMC04: Counter #1 Reload Value Register
TMC08: Counter #1 First Matching Register
TMC0C: Counter #1 Second Matching Register
TMC10: Counter #2 Status Register
TMC14: Counter #2 Reload Value Register
TMC18: Counter #2 First Matching Register
TMC1C: Counter #2 Second Matching Register
TMC20: Counter #3 Status Register
TMC24: Counter #3 Reload Value Register
TMC28: Counter #3 First Matching Register
TMC2C: Counter #3 Second Matching Register
TMC30: Control Register
TMC34: Control Register 2
TMC38: Control Register 3
TMC3C: TMC30 Clear Register
TMC40: Counter #4 Status Register
TMC44: Counter #4 Reload Value Register
TMC48: Counter #4 First Matching Register
TMC4C: Counter #4 Second Matching Register
TMC50: Counter #5 Status Register
TMC54: Counter #5 Reload Value Register
TMC58: Counter #5 First Matching Register
TMC5C: Counter #5 Second Matching Register
TMC60: Counter #6 Status Register
TMC64: Counter #6 Reload Value Register
TMC68: Counter #6 First Matching Register
TMC6C: Counter #6 Second Matching Register
TMC70: Counter #7 Status Register
TMC74: Counter #7 Reload Value Register
TMC78: Counter #7 First Matching Register
TMC7C: Counter #7 Second Matching Register
TMC80: Counter #8 Status Register
TMC84: Counter #8 Reload Value Register
TMC88: Counter #8 First Matching Register
TMC8C: Counter #8 Second Matching Register

38.2 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.

38.3 Registers : Base Address = 0x1E78:2000

Offset: 00h	TMC00: Counter #1 Status Register	Init = 0
Bit	**R/W**	**Description**
31:0	RW	**Counter #1 Counter status** This register stores the current status of counter #1. 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
Bit	**R/W**	**Description**
31:0	RW	**Counter #1 reload value register** When counter #1 decrease to zero, the reload value will be loaded to counter #1 automatically.

Offset: 08h	TMC08: Counter #1 First Matching Register	Init = 0
Bit	**R/W**	**Description**
31:0	RW	**First set match register** When counter #1 match this register, the timer will generate an edge triggered interrupt to CPU.

Offset: 0Ch	TMC0C: Counter #1 Second Matching Register	Init = 0
Bit	**R/W**	**Description**
31:0	RW	**Secondary match register** When counter #1 match this register, the timer will generate an edge triggered interrupt to CPU.

Offset: 10h	TMC10: Counter #2 Status Register	Init = 0
Bit	**R/W**	**Description**
31:0	RW	**Counter #2 Counter status** This register stores the current status of counter #2. When timer enable bit TMC30 [4] is disabled, the reload register will be loaded into this counter. When timer enable bit TMC30 [4] is set, the counter will start to decrement. CPU can update this register value when enable bit is set.

Offset: 14h	TMC14: Counter #2 Reload Value Register	Init = 0

Bit	R/W	Description
31:0	RW	**Counter #1 reload value register** When counter #2 decrease to zero, the reload value will be reload to counter #2 automatically.

Offset: 18h	TMC18: Counter #2 First Matching Register	Init = 0

Bit	R/W	Description
31:0	RW	**First set match register** When counter #2 match this register, the timer will generate an edge trigger interrupt to CPU.

Offset: 1Ch	TMC1C: Counter #2 Second Matching Register	Init = 0

Bit	R/W	Description
31:0	RW	**Secondary match register** When counter #2 match this register, the timer will generate an edge trigger interrupt to CPU.

Offset: 20h	TMC20: Counter #3 Status Register	Init = 0

Bit	R/W	Description
31:0	RW	**Counter #3 Counter status** This register stores the current status of counter #3. When timer enable bit TMC30 [8] is disabled, the reload register will be loaded into this counter. When timer enable bit TMC30 [8] is set, the counter will start to decrement. CPU can update this register value when enable bit is set.

Offset: 24h	TMC24: Counter #3 Reload Value Register	Init = 0

Bit	R/W	Description
31:0	RW	**Counter #1 reload value register** When counter #3 decrease to zero, the reload value will be reload to counter #3 automatically.

Offset: 28h	TMC28: Counter #3 First Matching Register	Init = 0

Bit	R/W	Description
31:0	RW	**First set match register** When counter #3 match this register, the timer will generate an edge trigger interrupt to CPU.

Offset: 2Ch	TMC2C: Counter #3 Second Matching Register	Init = 0

Bit	R/W	Description
31:0	RW	**Secondary match register** When counter #3 match this register, the timer will generate an edge trigger interrupt to CPU.

Offset: 30h		TMC30: Control Register	Init = 0
Bit	**R/W**	**Description**	
31	RW	**Enable Timer #8 Pulse generation and output enable** 0: disable 1: enable Enable the timer pulse output mode can generate a programmable frequency and duty cycle square wave pulse. The output pulse will switch to 1 when first matched, and switch to 0 when timer count to 0. When disabled, the pulse output also disabled.	
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 Pulse generation and output enable** 0: disable 1: enable Enable the timer pulse output mode can generate a programmable frequency and duty cycle square wave pulse. The output pulse will switch to 1 when first matched, and switch to 0 when timer count to 0. When disabled, the pulse output also disabled.	
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	
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 Pulse generation and output enable** 0: disable 1: enable Enable the timer pulse output mode can generate a programmable frequency and duty cycle square wave pulse. The output pulse will switch to 1 when first matched, and switch to 0 when timer count to 0. When disabled, the pulse output also disabled.	
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	

to next page

from previous page

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 Pulse generation and output enable** 0: disable 1: enable Enable the timer pulse output mode can generate a programmable frequency and duty cycle square wave pulse. The output pulse will switch to 1 when first matched, and switch to 0 when timer count to 0. When disabled, the pulse output also disabled.
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		**Reserved (0)**
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		**Reserved (0)**
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
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		**Reserved (0)**
6	RW	**Enable Overflow Interrupt for Timer/Counter #2** 0: disable. 1: enable. when timer overflow (count to 0) occurred, interrupt will be generated

to next page

from previous page

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		**Reserved (0)**
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.

Offset: 34h **TMC34: Control Register 2** **Init = 0**

Bit	R/W	Description
31:24		**Reserved (0)**
23	RW	**Timer8 Sequence Pulse initial output value**
22	RW	**Timer7 Sequence Pulse initial output value**
21	RW	**Timer6 Sequence Pulse initial output value**
20	RW	**Timer5 Sequence Pulse initial output value**
19	RW	**Timer4 Sequence Pulse initial output value**
18	RW	**Timer3 Sequence Pulse initial output value** When timer count to 0, this value will be toggled.
17:16		**Reserved (0)**
15	RW	**Enable Timer8 Sequence Pulse generation start counting**
14	RW	**Enable Timer7 Sequence Pulse generation start counting**
13	RW	**Enable Timer6 Sequence Pulse generation start counting**
12	RW	**Enable Timer5 Sequence Pulse generation start counting**
11	RW	**Enable Timer4 Sequence Pulse generation start counting**
10	RW	**Enable Timer3 Sequence Pulse generation start counting** 0: disable 1: enable When disabled, the timer will load the re-load register value as the initial value. When enable, timer will start decrement. When count to 0, output value will be toggled and this bit also will be cleared automatically.
9:8		**Reserved (0)**
7	RW	**Enable Timer8 Sequence Pulse generation and output enable**
6	RW	**Enable Timer7 Sequence Pulse generation and output enable**
5	RW	**Enable Timer6 Sequence Pulse generation and output enable**

to next page

from previous page

4	RW	**Enable Timer5 Sequence Pulse generation and output enable**
3	RW	**Enable Timer4 Sequence Pulse generation and output enable**
2	RW	**Enable Timer3 Sequence Pulse generation and output enable** 0: disable 1: enable When enable, the value at bit[23:18] will output. When disabled, the signal output also disabled, the pin stays at tri-state mode.
1:0		**Reserved (0)**

Note :
This register only be reset by power on reset SRST# or full chip watchdog reset.
When use the sequence pulse generation function, timer must be disabled (TMC30). After the sequence pulse generation finished (all counting to 0), the timer can be used again normally.

Offset: 38h — **TMC38: Control Register 3** — **Init = 0**

Bit	R/W	Description
31:8		**Reserved (0)**
7:0	W	**Command to set bit[0]** Write 0xAE can set bit[0] => 1 Write 0xEA can set bit[0] => 0
0	R	**Enable separate clear control for TMC30** 0: Backward compatible mode, write TMC30 can set 1 or 0. 1: Separate mode, write TMC30 can only set to 1, clear TMC30 to 0 must write TMC3C with 1 at the same bit position.

Offset: 3Ch — **TMC3C: TMC30 Clear Register** — **Init = 0**

Bit	R/W	Description
31:0	W	**Clear TMC30** When TMC38[0] = 1, acts for clearing TMC30 from 1 to 0. write '1', clear bit to 0 write '0', no effect When TMC38[0] = 0, no effect.

Offset: 40h — **TMC40: Counter #4 Status Register** — **Init = 0**

Bit	R/W	Description
31:0	RW	**Counter #4 Counter status** This register stores the current status of counter #4. When timer enable bit TMC30 [12] is disabled, the reload register will be loaded into this counter. When timer enable bit TMC30 [12] is set, the counter will start to decrement. CPU can update this register value when enable bit is set.

Offset: 44h — **TMC44: Counter #4 Reload Value Register** — **Init = 0**

Bit	R/W	Description
31:0	RW	**Counter #4 reload value register** When counter #4 decrease to zero, the reload value will be loaded to counter #4 automatically.

Offset: 48h	TMC48: Counter #4 First Matching Register	Init = 0
Bit	**R/W**	**Description**
31:0	RW	**First set match register** When counter #4 match this register, the timer will generate an edge triggered interrupt to CPU.

Offset: 4Ch	TMC4C: Counter #4 Second Matching Register	Init = 0
Bit	**R/W**	**Description**
31:0	RW	**Secondary match register** When counter #4 match this register, the timer will generate an edge triggered interrupt to CPU.

Offset: 50h	TMC50: Counter #5 Status Register	Init = 0
Bit	**R/W**	**Description**
31:0	RW	**Counter #5 Counter status** This register stores the current status of counter #5. When timer enable bit TMC30 [16] is disabled, the reload register will be loaded into this counter. When timer enable bit TMC30 [16] is set, the counter will start to decrement. CPU can update this register value when enable bit is set.

Offset: 54h	TMC54: Counter #5 Reload Value Register	Init = 0
Bit	**R/W**	**Description**
31:0	RW	**Counter #5 reload value register** When counter #5 decrease to zero, the reload value will be loaded to counter #5 automatically.

Offset: 58h	TMC58: Counter #5 First Matching Register	Init = 0
Bit	**R/W**	**Description**
31:0	RW	**First set match register** When counter #5 match this register, the timer will generate an edge triggered interrupt to CPU.

Offset: 5Ch	TMC5C: Counter #5 Second Matching Register	Init = 0
Bit	**R/W**	**Description**
31:0	RW	**Secondary match register** When counter #5 match this register, the timer will generate an edge triggered interrupt to CPU.

Offset: 60h	TMC60: Counter #6 Status Register	Init = 0
Bit	**R/W**	**Description**
31:0	RW	**Counter #6 Counter status** This register stores the current status of counter #6. When timer enable bit TMC30 [20] is disabled, the reload register will be loaded into this counter. When timer enable bit TMC30 [20] is set, the counter will start to decrement. CPU can update this register value when enable bit is set.

Offset: 64h	TMC64: Counter #6 Reload Value Register	Init = 0

Bit	R/W	Description
31:0	RW	**Counter #6 reload value register** When counter #6 decrease to zero, the reload value will be loaded to counter #6 automatically.

Offset: 68h	TMC68: Counter #6 First Matching Register	Init = 0

Bit	R/W	Description
31:0	RW	**First set match register** When counter #6 match this register, the timer will generate an edge triggered interrupt to CPU.

Offset: 6Ch	TMC6C: Counter #6 Second Matching Register	Init = 0

Bit	R/W	Description
31:0	RW	**Secondary match register** When counter #6 match this register, the timer will generate an edge triggered interrupt to CPU.

Offset: 70h	TMC70: Counter #7 Status Register	Init = 0

Bit	R/W	Description
31:0	RW	**Counter #7 Counter status** This register stores the current status of counter #7. When timer enable bit TMC30 [24] is disabled, the reload register will be loaded into this counter. When timer enable bit TMC30 [24] is set, the counter will start to decrement. CPU can update this register value when enable bit is set.

Offset: 74h	TMC74: Counter #7 Reload Value Register	Init = 0

Bit	R/W	Description
31:0	RW	**Counter #7 reload value register** When counter #7 decrease to zero, the reload value will be loaded to counter #7 automatically.

Offset: 78h	TMC78: Counter #7 First Matching Register	Init = 0

Bit	R/W	Description
31:0	RW	**First set match register** When counter #7 match this register, the timer will generate an edge triggered interrupt to CPU.

Offset: 7Ch	TMC7C: Counter #7 Second Matching Register	Init = 0

Bit	R/W	Description
31:0	RW	**Secondary match register** When counter #7 match this register, the timer will generate an edge triggered interrupt to CPU.

Offset: 80h	TMC80: Counter #8 Status Register	Init = 0
Bit	**R/W**	**Description**
31:0	RW	**Counter #8 Counter status** This register stores the current status of counter #8. When timer enable bit TMC30 [28] is disabled, the reload register will be loaded into this counter. When timer enable bit TMC30 [28] is set, the counter will start to decrement. CPU can update this register value when enable bit is set.

Offset: 84h	TMC84: Counter #8 Reload Value Register	Init = 0
Bit	**R/W**	**Description**
31:0	RW	**Counter #8 reload value register** When counter #8 decrease to zero, the reload value will be loaded to counter #8 automatically.

Offset: 88h	TMC88: Counter #8 First Matching Register	Init = 0
Bit	**R/W**	**Description**
31:0	RW	**First set match register** When counter #8 match this register, the timer will generate an edge triggered interrupt to CPU.

Offset: 8Ch	TMC8C: Counter #8 Second Matching Register	Init = 0
Bit	**R/W**	**Description**
31:0	RW	**Secondary match register** When counter #8 match this register, the timer will generate an edge triggered interrupt to CPU.

38.4 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

Initial Sequence Pulse Generation function:

1. Disable the related timer that want to generate pulse
2. Set reload value for each timer
3. Set timer control for each timer, reference clock selection and interrupt
4. Set sequence output default value (TMC34[23:18])
5. Enable signal output (TMC34[7:2])
6. Enable counting (TMC34[15:10])
7. Wait all timer finished, this can be achieved by enabling the interrupt at the longest timer. Beware to set the match register to a higher value, so it will only generate interrupt when counting to 0.
8. All timer can be used normally again

38.5 Programming Note

38.5.1 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.

38.5.2 Programmable Duty Cycle Pulse Generation

Timer #5, #6, #7 and #8 had the function to generate a periodic pulse output with programmable duty cycle. The generated pulse will switch to '1' output every time the counter value match the First matching register, and switch back to '0' output when counter count to 0.

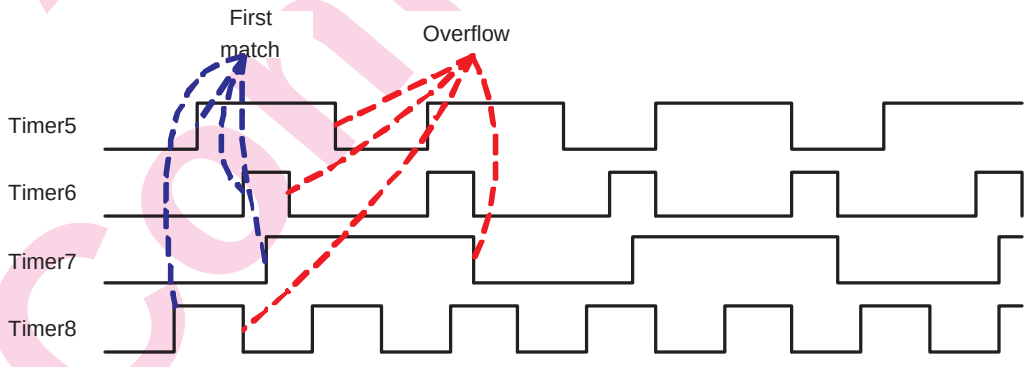


Figure 51: Programmable duty cycle pulse generation

38.5.3 Programmable Delay Sequence Pulse Generation

Timer #3, #4, #5, #6, #7 and #8 had the function to generate a programmable delay output pulse. The generated pulse start by the initial value, and toggle after timer count to 0.

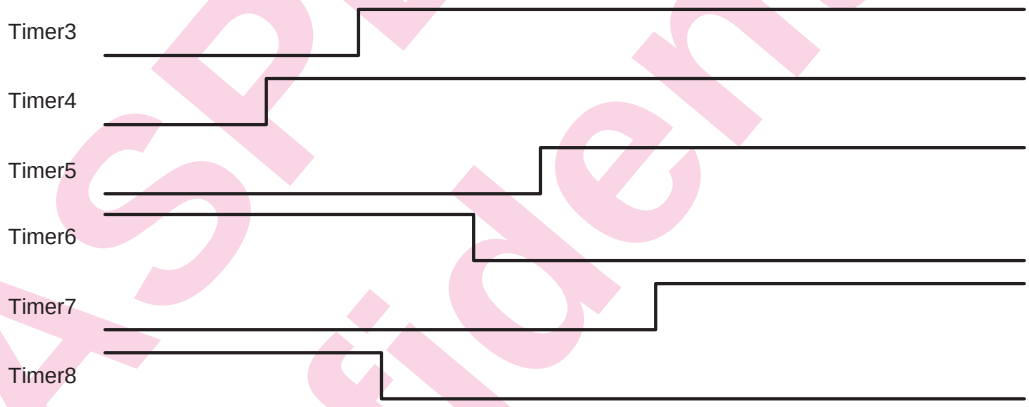


Figure 52: Programmable delay sequence pulse generation

39 UART Controller (16550)

39.1 Overview

AST2500 integrates 5 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 two 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
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 53.

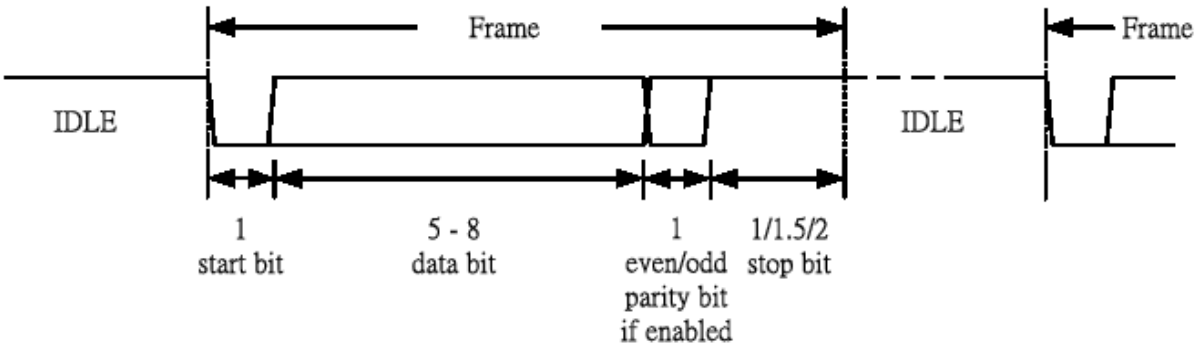


Figure 53: UART Packet Frame

39.2 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 115.2K baud-rate
- 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

39.3 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		**Reserved (0)**
7 :0	R	**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.
7:0	W	**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** and THRE is set, 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** and THRE is set, 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		**Reserved (0)**
7	RW	**PTIME: Enable Programmable THRE Interrupt Mode** 0: Disable THRE interrupt mode 1: Enable THRE interrupt mode
6 :4		**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		**Reserved (0)**
7:6	R	**FIFO-Enabled Bits** 00: FIFOs disabled 11: FIFOs enabled
5:4		**Reserved (0)**
3:1	R	**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.
0	R	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	IER[7] = 0 : THR Empty IER[7] = 1 : TX FIFO at or below threshold	Reading the IIR or writing into THR (FIFO or IER[7] disabled) or TX FIFO above threshold (FIFO or IER[7] enabled)
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		**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
5:4	W	**Define the Transmitter FIFO Interrupt trigger level.** 00: FIFO empty 01: 2 bytes in FIFO 10: FIFO 1/4 full 11: FIFO 1/2 full
3		**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 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		**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		**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		**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 → 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'

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		**Reserved (0)**
7	R	**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	R	**Transmitter empty** 1: The UART_THR or FIFO and the Transmitter Shift Register are both empty. 0: Otherwise.
5	R	**THRE: Transmitter holding register empty** 1: When FIFO or THRE mode is disabled, the UART_THR or FIFO is empty. When FIFO and THRE mode is enabled, the transmitter FIFO is full. This also causes a THRE Interrupt to occur, if the THRE Interrupt is enabled. 0: Otherwise.
4	R	**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	R	**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	R	**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	R	**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	R	**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	R	**Reserved (0)**
7	R	**Complement of the nDCD input or equals to Out2(MCR[3]) in loopback mode.**
6	R	**Complement of the nRI input or equals to Out1(MCR[2]) in loopback mode.**
5	R	**Complement of the nDSR input or equals to nDTR(MCR[0]) in loopback mode.**
4	R	**Complement of the nCTS input or equals to nRTS(MCR[1]) in loopback mode.**

to next page

from previous page

3	R	**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	R	**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	R	**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	R	**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		**Reserved (0)**
7 :0	RW	**Scratch bits** This register can be used as a temporary storage, no specific definition.

39.3.1 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 = 24MHz / (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		**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		**Reserved.**
7:0	RW	**The MSB of the baud-rate divisor latch.**

40 UART DMA

40.1 Overview

Uart DMA Controller (UDMA) includes 12 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 Timer = 0x1E79_E000
Physical address = (Base address of Timer) + 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
UDMA050: UART1 RX read pointer
UDMA054: UART1 RX write pointer
UDMA058: UART1 RX buffer base address
UDMA060: UART2 TX read pointer
UDMA064: UART2 TX write pointer
UDMA068: UART2 TX buffer base address
UDMA070: UART2 RX read pointer
UDMA074: UART2 RX write pointer
UDMA078: UART2 RX buffer base address
UDMA080: UART3 TX read pointer
UDMA084: UART3 TX write pointer
UDMA088: UART3 TX buffer base address
UDMA090: UART3 RX read pointer
UDMA094: UART3 RX write pointer
UDMA098: UART3 RX buffer base address
UDMA0A0: UART4 TX read pointer
UDMA0A4: UART4 TX write pointer
UDMA0A8: UART4 TX buffer base address
UDMA0B0: UART4 RX read pointer
UDMA0B4: UART4 RX write pointer
UDMA0B8: UART4 RX buffer base address

40.2 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

40.3 Registers : Base Address = 0x1E78:2000

Offset: 000h	UDMA000: UART TX DMA enable	Init = 0
Bit	**R/W**	**Description**
3:0	RW	**UART TX DMA enable register** This register the Uart DMA TX controllers.

Offset: 004h	UDMA004: UART RX DMA enable	Init = 0
Bit	**R/W**	**Description**
3:0	RW	**UART RX DMA enable register** This register the Uart DMA RX controllers.

Offset: 008h	UDMA008: Misc, buffer size	Init = x
Bit	**R/W**	**Description**
1:0	RW	**TX buffer size** 00: 1KB 01: 4KB 10: 16KB 11: 64KB
3:2	RW	**RX buffer size** 00: 1KB 01: 4KB 10: 16KB 11: 64KB
4	RW	**disable time out** 0: enable time out timer 1: disable time out timer

Offset: 00Ch	UDMA00C: UART DMA time out timer	Init = x
Bit	**R/W**	**Description**
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**
3:0	RW	**UART TX DMA reset** This register can reset the read pointer for TX controller.

Offset: 024h	UDMA024: UART RX DMA reset	Init = 0
Bit	**R/W**	**Description**
3:0	RW	**UART RX DMA reset** This register can reset the write pointer for RX controller.

Offset: 030h	UDMA030: UART TX DMA interrrupt enable	Init = x

Bit	R/W	Description
3:0	RW	**UART TX DMA interrrupt enable** This register interrupt of TX controller.

Offset: 034h	UDMA034: UART TX DMA interrrupt status	Init = x

Bit	R/W	Description
3:0	RW	**UART TX DMA interrrupt status** This register is interrupt status of TX controller. Write 1 clear.

Offset: 038h	UDMA038: UART RX DMA interrrupt enable	Init = x

Bit	R/W	Description
3:0	RW	**UART RX DMA interrrupt enable** This register interrupt of RX controller.

Offset: 03Ch	UDM0A3C: UART RX DMA interrrupt status	Init = x

Bit	R/W	Description
3:0	RW	**UART RX DMA interrrupt 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
15:0	R	**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
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
15:0	RW	**UART1 TX buffer base address** This base address of UART1 TX buffer.

Offset: 050h	UDMA050: UART1 RX read pointer	Init = x

Bit	R/W	Description
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
15:0	R	**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
15:0	RW	**UART1 TX buffer base address** This base address of UART1 RX buffer.

Offset: 060h	UDMA060: UART2 TX read pointer	Init = x

Bit	R/W	Description
15:0	R	**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
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
15:0	RW	**UART2 TX buffer base address** This base address of UART2 TX buffer.

Offset: 070h	UDMA070: UART2 RX read pointer	Init = x

Bit	R/W	Description
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
15:0	R	**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
15:0	RW	**UART2 TX buffer base address** This base address of UART2 RX buffer.

Offset: 080h	UDMA080: UART3 TX read pointer	Init = x

Bit	R/W	Description
15:0	R	**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
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
15:0	RW	**UART3 TX buffer base address** This base address of UART3 TX buffer.

Offset: 090h	UDMA090: UART3 RX read pointer	Init = x

Bit	R/W	Description
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
15:0	R	**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
15:0	RW	**UART3 TX buffer base address** This base address of UART3 RX buffer.

Offset: 0A0h	UDMA0A0: UART4 TX read pointer	Init = x

Bit	R/W	Description
15:0	R	**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
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
15:0	RW	**UART4 TX buffer base address** This base address of UART4 TX buffer.

Offset: 0B0h	UDMA0B0: UART4 RX read pointer	Init = x

Bit	R/W	Description
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
15:0	R	**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
15:0	RW	**UART4 TX buffer base address** This base address of UART4 RX buffer.

41 Watchdog Timer

41.1 Overview

Watchdog Timer (WDT) includes 3 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 6 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 external reset controller (only WDT1 and WDT2)
- **Alternate boot signal**: to boot from altenate block

WDT totally implements 24 sets of 32-bit registers, which are listed below, to program the various functions supported by WDT. Each register has its own specific offset value to derive its physical address location.

Base address of WDT = 0x1E78_5000
Physical address = (Base address of WDT) + Offset

WDT00: WDT1 Counter Status Register
WDT04: WDT1 Counter Reload Value Register
WDT08: WDT1 Counter Restart Register
WDT0C: WDT1 Control Register
WDT10: WDT1 Timeout Status Register
WDT14: WDT1 Clear Timeout Status Register
WDT18: WDT1 Reset Width Register
WDT1C: WDT1 Reset Mask Register
WDT20: WDT2 Counter Status Register
WDT24: WDT2 Counter Reload Value Register
WDT28: WDT2 Counter Restart Register
WDT2C: WDT2 Control Register
WDT30: WDT2 Timeout Status Register
WDT34: WDT2 Clear Timeout Status Register
WDT38: WDT2 Reset Width Register
WDT3C: WDT2 Reset Mask Register
WDT40: WDT3 Counter Status Register
WDT44: WDT3 Counter Reload Value Register
WDT48: WDT3 Counter Restart Register
WDT4C: WDT3 Control Register
WDT50: WDT3 Timeout Status Register
WDT54: WDT3 Clear Timeout Status Register
WDT5C: WDT3 Reset Mask Register

41.2 Features

- Directly connected to APB bus
- Watchdog function
- Built-in 3 sets of 32-bit WDT modules

- Generate either interrupt or reset after counting down to zero (programmable)
- Generate 2 types of reset pulse (programmable) to reset SOC part or full chip.
- Generate alternate boot signal for booting from alternate boot block.

41.3 Registers : Base Address = 0x1E78:5000

Offset: 00h	WDT00: WDT1 Counter Status Register	Init = 0x014FB180

Bit	R/W	Description
31:0	R	**Counter status** This register stores the current status of counter. After SOC reset, this register is set to 0x14FB180 (22 seconds). When the programmer writes 0x4755 to Restart register, Reload register will be loaded into this register. Counter starts to decrease once WDT0C[0] enable bit is set. If watchdog timer is disabled, it will hold the value.

Offset: 04h	WDT04: WDT1 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.

Offset: 08h	WDT08: WDT1 Counter Restart Register	Init = 0

Bit	R/W	Description
31:16		**Reserved (0)**
15:0	W	**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: WDT1 Control Register	Init = 0x10

Bit	R/W	Description
31:12	RW	**Pre-timeout value for interrupt generation** When watchdog timer match below equation, issue the interrupt. WDT00[31:20] == 0 && WDT00[19:0] == pre-timeout
11:8		**Reserved (0)**
7	RW	**Enable second boot code** 0: Use default boot code whenever WDT reset. 1: Use second boot code whenever WDT reset.
6:5	RW	**Reset system mode** 00: SOC system (gated by reset mask registers) 01: Full chip 1x: ARM CPU only, just reboot firmware, no any other IPs will be reset.
4	R	**Clock select for WDT Counter** The counter will be decreased by selected clock. ~~0: PCLK~~ 1: 1MHz clock source

to next page

from previous page

3	RW	**wdt_ext: External signal enable after timeout** This signal is connected to an external WDTRST1 output pin T22. 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 equal to the pre-timer 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.
0	RW	**WDT enable signal** 0: disable 1: enable

Offset: 10h **WDT10: WDT1 Timeout Status Register** **Init = 0**

Bit	R/W	Description
31:16		**Reserved (0)**
15:8	R	**Watchdog event counter** This field report the count of the WDT timeout event happened.
7:3		**Reserved (0)**
2	R	**Indicate interrupt** Interrupt register is a record when watchdog counter match the pre-timeout setting.
1	R	**Indicate boot code source** 0: default boot code 1: second boot code Firmware must clear this bit value to '0' after jump to the physical flash address.
0	R	**Indicate timeout** Timeout register is a record. If the WDT had ever occurred timeout condition, this bit will be set automatically. 0: timeout never occur 1: timeout occur

Offset: 14h **WDT14: WDT1 Clear Timeout Status Register** **Init = 0**

Bit	R/W	Description
31:8		**Reserved (0)**
7:1	W	**Clear timeout counter status** Write 0x3B value into this field to clear WDT counter register.
0	W	**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: WDT1 Reset Width Register	Init = 0xFF

Bit	R/W	Description
31	RW	**Reset pulse polarity selection** 0: open drain active low output 1: open drain 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.
29:20		**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: WDT1 Reset Mask Register	Init = 0x23FFFF3

Bit	R/W	Description
31:26	R	**Reserved**
25	RW	**Enable reset Misc. SOC controller** The Misc. controller includes: WDT, RTC, Timer, UART, SRAM.
24	RW	**Enable reset SPI controller**
23	RW	**Enable reset X-DMA controller**
22	RW	**Enable reset MCTP controller**
21	RW	**Enable reset GPIO controller**
20	RW	**Enable reset ADC controller**
19	RW	**Enable reset JTAG master controller**
18	RW	**Enable reset PECI controller**
17	RW	**Enable reset PWM controller**
16	RW	**Enable reset CRT mode 2D engine**
15	RW	**Enable reset MIC controller**
14	RW	**Enable reset SD/SDIO controller**
13	RW	**Enable reset LPC controller**
12	RW	**Enable reset HAC engine**
11	RW	**Enable reset Video engine**
10	RW	**Enable reset USB1.1 HID/USB2.0 Host EHCI2 controller**
9	RW	**Enable reset USB1.1 Host controller**
8	RW	**Enable reset USB2.0 Host/Hub controller**
7	RW	**Enable reset Graphics CRT controller**
6	RW	**Enable reset MAC#2 controller**
5	RW	**Enable reset MAC#1 controller**

to next page

from previous page

4	RW	**Enable reset I2C controller**
3	RW	**Enable reset AHB bridges**
2	RW	**Enable reset SDRAM controller**
1	RW	**Enable reset Coprocessor**
0	RW	**Enable reset ARM**

Note :
This register controls the IPs to be reset by the watchdog event. The watchdog event indicates the Watchdog_reset_SOC generated by WDT1 controller.

Offset: 20h **WDT20: WDT2 Counter Status Register** **Init = 0x014FB180**

Bit	R/W	Description
31:0	R	**Counter status** This register stores the current status of counter. After SOC reset, this register is set to 22 seconds, which is 0x014FB180. When the programmer writes 0x4755 to Restart register, Reload register will be loaded into this register. Counter starts to decrease once WDT2C[0] enable bit is set. If watchdog timer is disabled, it will hold the value.

Offset: 24h **WDT24: WDT2 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 WDT20 register. When reset or restart, Reload value will be automatically loaded into WDT20 register.

Offset: 28h **WDT28: WDT2 Counter Restart Register** **Init = 0**

Bit	R/W	Description
31:16		**Reserved (0)**
15:0	W	**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 WDT20 register and WDT20 register restarts to decrease if WDT2C[0] register is set.

Offset: 2Ch **WDT2C: WDT2 Control Register** **Init = 0x52**

Bit	R/W	Description
31:12	RW	**Pre-timeout value for interrupt generation** When watchdog timer match below equation, issue the interrupt. WDT20[31:20] == 0 && WDT20[19:0] == pre-timeout
11:8		**Reserved (0)**
7	RW	**Enable second boot code** 0: Use default boot code whenever WDT reset. 1: Use second boot code whenever WDT reset.
6:5	RW	**Reset system mode** 00: SOC system (gated by reset mask registers) 01: Full chip 1x: ARM CPU only, just reboot firmware, no any other IPs will be reset.

to next page

from previous page

4	R	**Clock select for WDT Counter** The counter will be decreased by selected clock. ~~0: PCLK~~ 1: 1MHz clock source
3	RW	**wdt_ext: External signal enable after timeout** This signal is connected to an external WDTRST2 output pin R20. If timeout occurs, and this bit is enabled, an active high output pulse will be generated. The generated pulse width is controller by WDT38. 0: disable 1: enable
2	RW	**wdt_intr: Generate interrupt when timer count equal to the pre-timer 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.
0	RW	**WDT enable signal** 0: disable 1: enable This bit is default loaded with hardware strapping 2nd boot option value. Firmware must disable this bit after booting successfully.

Offset: 30h **WDT30: WDT2 Timeout Status Register** **Init = 0**

Bit	R/W	Description
31:16		**Reserved (0)**
15:8	R	**Watchdog event counter** This field report the count of the WDT timeout event happened.
7:3		**Reserved (0)**
2	R	**Indicate interrupt** Interrupt register is a record when watchdog counter match the pre-timeout setting.
1	R	**Indicate boot code source** 0: default boot code 1: second boot code Firmware must clear this bit value to '0' after jump to the physical flash address.
0	R	**Indicate timeout** Timeout register is a record. If the WDT had ever occurred timeout condition, this bit will be set automatically. 0: timeout never occur 1: timeout occur

Offset: 34h **WDT34: WDT2 Clear Timeout Status Register** **Init = 0**

Bit	R/W	Description
31:8		**Reserved (0)**
7:1	W	**Clear timeout counter status** Write 0x3B value into this field to clear WDT counter register.
0	W	**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: 38h	WDT38: WDT2 Reset Width Register	Init = 0xFF

Bit	R/W	Description
31	RW	**Reset pulse polarity selection** 0: open drain active low output 1: open drain 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.
29:20		**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: 3Ch	WDT3C: WDT2 Reset Mask Register	Init = 0x23FFFF3

Bit	R/W	Description
31:26	R	**Reserved**
25	RW	**Enable reset Misc. SOC controller** The Misc. controller includes: WDT, RTC, Timer, UART, SRAM.
24	RW	**Enable reset SPI controller**
23	RW	**Enable reset X-DMA controller**
22	RW	**Enable reset MCTP controller**
21	RW	**Enable reset GPIO controller**
20	RW	**Enable reset ADC controller**
19	RW	**Enable reset JTAG master controller**
18	RW	**Enable reset PECI controller**
17	RW	**Enable reset PWM controller**
16	RW	**Enable reset CRT mode 2D engine**
15	RW	**Enable reset MIC controller**
14	RW	**Enable reset SD/SDIO controller**
13	RW	**Enable reset LPC controller**
12	RW	**Enable reset HAC engine**
11	RW	**Enable reset Video engine**
10	RW	**Enable reset USB1.1 HID/USB2.0 Host EHCI2 controller**
9	RW	**Enable reset USB1.1 Host controller**
8	RW	**Enable reset USB2.0 Host/Hub controller**
7	RW	**Enable reset Graphics CRT controller**
6	RW	**Enable reset MAC#2 controller**
5	RW	**Enable reset MAC#1 controller**

to next page

from previous page

4	RW	**Enable reset I2C controller**
3	RW	**Enable reset AHB bridges**
2	RW	**Enable reset SDRAM controller**
1	RW	**Enable reset Coprocessor**
0	RW	**Enable reset ARM**

Note :
This register controls the IPs to be reset by the watchdog event. The watchdog event indicates the Watchdog_reset_SOC generated by WDT2 controller.

Offset: 40h **WDT40: WDT3 Counter Status Register** **Init = 0x000F4240**

Bit	R/W	Description
31:0	R	**Counter status** This register stores the current status of counter. After SOC reset, this register is set to 1 seconds, which is 0x000F4240. When the programmer writes 0x4755 to Restart register, Reload register will be loaded into this register. Counter starts to decrease once WDT4C[0] enable bit is set. If watchdog timer is disabled, it will hold the value.

Offset: 44h **WDT44: WDT3 Counter Reload Value Register** **Init = 0x000F4240**

Bit	R/W	Description
31:0	RW	**Counter reload value register** Reload register contains value which will be loaded into WDT40 register. When reset or restart, Reload value will be automatically loaded into WDT40 register.

Offset: 48h **WDT48: WDT3 Counter Restart Register** **Init = 0**

Bit	R/W	Description
31:16		**Reserved (0)**
15:0	W	**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 WDT40 register and WDT40 register restarts to decrease if WDT4C[0] register is set.

Offset: 4Ch **WDT4C: WDT3 Control Register** **Init = 0x153**

Bit	R/W	Description
31:12	RW	**Pre-timeout value for interrupt generation** When watchdog timer match below equation, issue the interrupt. WDT40[31:20] == 0 && WDT40[19:0] == pre-timeout
11:9		**Reserved (0)**
8	RW	**Enable SPI address mode toggling control** 0: NOP 1: Toggle SPI address mode (3-bytes or 4-bytes) whenever WDT reset.
7	RW	**Enable second boot code** 0: Use default boot code whenever WDT reset. 1: Use second boot code whenever WDT reset.

to next page

from previous page

6:5	RW	**Reset system mode** 00: SOC system (gated by reset mask registers) 01: Full chip 1x: ARM CPU only, just reboot firmware, no any other IPs will be reset.
4	R	**Clock select for WDT Counter** The counter will be decreased by selected clock. ~~0: PCLK~~ 1: 1MHz clock source
3	R	**Reserved**
2	RW	**wdt_intr: Generate interrupt when timer count equal to the pre-timer 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.
0	RW	**WDT enable signal** 0: disable 1: enable This bit is default enabled for SPI 3-bytes or 4-bytes address mode detection function. Firmware should disable this bit at the early satge of booting.

Offset: 50h **WDT50: WDT3 Timeout Status Register** **Init = 0**

Bit	R/W	Description
31:16		**Reserved (0)**
15:8	R	**Watchdog event counter** This field report the count of the WDT timeout event happened.
7:3		**Reserved (0)**
2	R	**Indicate interrupt** Interrupt register is a record when watchdog counter match the pre-timeout setting.
1	R	**Indicate boot code source** 0: default boot code 1: second boot code Firmware must clear this bit value to '0' after jump to the physical flash address.
0	R	**Indicate timeout** Timeout register is a record. If the WDT had ever occurred timeout condition, this bit will be set automatically. 0: timeout never occur 1: timeout occur

Offset: 54h **WDT54: WDT3 Clear Timeout Status Register** **Init = 0**

Bit	R/W	Description
31:8		**Reserved (0)**
7:1	W	**Clear timeout counter status** Write 0x3B value into this field to clear WDT counter register.
0	W	**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: 5Ch	WDT5C: WDT3 Reset Mask Register	Init = 0x23FFFF3

Bit	R/W	Description
31:26	R	**Reserved**
25	RW	**Enable reset Misc. SOC controller** The Misc. controller includes: WDT, RTC, Timer, UART, SRAM.
24	RW	**Enable reset SPI controller**
23	RW	**Enable reset X-DMA controller**
22	RW	**Enable reset MCTP controller**
21	RW	**Enable reset GPIO controller**
20	RW	**Enable reset ADC controller**
19	RW	**Enable reset JTAG master controller**
18	RW	**Enable reset PECI controller**
17	RW	**Enable reset PWM controller**
16	RW	**Enable reset CRT mode 2D engine**
15	RW	**Enable reset MIC controller**
14	RW	**Enable reset SD/SDIO controller**
13	RW	**Enable reset LPC controller**
12	RW	**Enable reset HAC engine**
11	RW	**Enable reset Video engine**
10	RW	**Enable reset USB1.1 HID/USB2.0 Host EHCI2 controller**
9	RW	**Enable reset USB1.1 Host controller**
8	RW	**Enable reset USB2.0 Host/Hub controller**
7	RW	**Enable reset Graphics CRT controller**
6	RW	**Enable reset MAC#2 controller**
5	RW	**Enable reset MAC#1 controller**
4	RW	**Enable reset I2C controller**
3	RW	**Enable reset AHB bridges**
2	RW	**Enable reset SDRAM controller**
1	RW	**Enable reset Coprocessor**
0	RW	**Enable reset ARM**

Note :
This register controls the IPs to be reset by the watchdog event. The watchdog event indicates the Watchdog_reset_SOC generated by WDT3 controller.

41.4 Operation

41.4.1 Enable watchdog reset

To enable watchdog timer, the programmer needs to set Reload register first. Reload decides the period of timeout. The default value of Reload is 0x3EF1480, it means that in a 66MHz system, the period of timeout is one second. The programmer can change this, if you need. For example, setting Reload to 0xEC08CE00, it means the period of timeout is 1 minute and it guarantees the system will be reset after one minute. This can be used to avoid the deadlock of system. The programmer needs to write 0x4755 to Restart register before timeout, when WDT is enable.

After setting the Reload, Restart and WDT0C[4] bit, the programmer can enable WatchDog by write '1' to WDT0C[0] bit. And then watchdog timer starts to count down. The following steps are the summary.

1. disable watchdog timer
2. set Reload register
3. write 0x4755 to Restart register
4. set WDT0C[4] bit
5. enable watchdog timer

41.4.2 Enable watchdog pulse output

1. wdt_ext output can be programmed as open-drain or push-pull type, which is different from previous BMC
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

41.4.3 Enable watchdog interrupt output

1. disable watchdog timer
2. set Reload register
3. write 0x4755 to Restart register
4. set WDT0C[2] = 1 and WDT0C[7:1] to the required value
5. enable watchdog timer
6. when interrupt triggered, clear WDT10 bit[0] by writing WDT14 = 0x1 to clear interrupt

41.5 Programming Note

41.5.1 PCLK frequency requirement

When choose 1MHz clock as the timer base clock. The frequency of PCLK must larger than 10MHz, else the timer may have accuracy problem. For lower PCLK (smaller than 10MHz) requirement, the base clock must use the PCLK.

42 PWM & Fan Tacho Controller

42.1 Overview

Base Address of PWM & Fan Tach Controller = 0x1E78_6000
Physical address of register = (Base address of PWM & Fan Tach Controller) + Offset

PTCR00: General Control Register
PTCR04: Clock Control Register
PTCR08: Duty Control 0 Register
PTCR0C: Duty Control 1 Register
PTCR10: Type M Control 0 Register
PTCR14: Type M Control 1 Register
PTCR18: Type N Control 0 Register
PTCR1C: Type N Control 1 Register
PTCR20: Tach Source Register
PTCR28: Trigger Register
PTCR2C: Result Register
PTCR30: Interrupt Control Register
PTCR34: Interrupt Status Register
PTCR38: Type M Limit Register
PTCR3C: Type N Limit Register
PTCR40: General Control Extension #1 Register
PTCR44: Clock Control Extension #1 Register
PTCR48: Duty Control 2 Register
PTCR4C: Duty Control 3 Register
PTCR50: Type O Control 0 Register
PTCR54: Type O Control 1 Register
PTCR60: Tach Source Extension #1 Register
PTCR78: Type O Limit Register

42.2 Features

PWM Controller

- Support 8 PWM outputs
- Support both low-frequency and high-frequency (e.g. 25KHz and 50KHz) PWM for fan speed control
- Duty cycle from 0 to 100% with 1/256 resolution incremental
- Support low-frequency PWM pulse stretching for fan speed measurements
- Shared with GPIO pins

Fan Tachometer Controller

- Directly connected to APB bus
- Support 16 tachometer inputs
- Measurement schemes: rising edge, falling edge or both edges
- Support interrupt trigger when over fan speed limitation setting
- 4 tachometer input pins are dedicated, 12 tachometer input pins are shared with DVO input pins

42.3 Registers : Base Address = 0x1E78:6000

Offset: 00h	PTCR00: General Control Register	Init = 0xXXXXX000

Bit	R/W	Description
31:16	RW	**Enable Fan Tach #15 ~ Fan Tach #0** Bit[31]: enable fan tach #15 0: disable 1: enable Bit[30]: enable fan tach #14 0: disable 1: enable ... Bit[16]: enable fan tach #0 0: disable 1: enable
15	RW	**Type selection bit [0] of PWM D port** 00: type M 01: type N 1X: type O
14	RW	**Type selection bit [0] of PWM C port** 00: type M 01: type N 1X: type O
13	RW	**Type selection bit [0] of PWM B port** 00: type M 01: type N 1X: type O
12	RW	**Type selection bit [0] of PWM A port** 00: type M 01: type N 1X: type O
11	RW	**Enable PWM D port** 0: disable 1: enable
10	RW	**Enable PWM C port** 0: disable 1: enable
9	RW	**Enable PWM B port** 0: disable 1: enable
8	RW	**Enable PWM A port** 0: disable 1: enable
7	RW	**Type selection bit [1] of PWM D port**
6	RW	**Type selection bit [1] of PWM C port**
5	RW	**Type selection bit [1] of PWM B port**
4	RW	**Type selection bit [1] of PWM A port**
3 :2	RW	**Reserved**
1	RW	**Clock source selection** 0: from 24MHz 1: from MCLK

to next page

from previous page

0	RW	**Enable PWM & Fan Tach clock** 0: disable 1: enable

Offset: 04h **PTCR04: Clock Control Register** **Init = X**

Bit	R/W	Description
31:24	RW	**Type N PWM period bit [7:0] (in units of type N PWM clock)**
23:20	RW	**Type N PWM clock division H bit [3:0]** 0000: divide 1 0001: divide 2 0010: divide 4 0011: divide 8 ... 1111: divide 32768
19:16	RW	**Type N PWM clock division L bit [3:0]** 0000: divide 1 0001: divide 2 0010: divide 4 0011: divide 6 ... 1111: divide 30
15:8	RW	**Type M PWM period bit [7:0] (in units of type M PWM clock)**
7 :4	RW	**Type M PWM clock division H bit [3:0]** 0000: divide 1 0001: divide 2 0010: divide 4 0011: divide 8 ... 1111: divide 32768
3 :0	RW	**Type M PWM clock division L bit [3:0]** 0000: divide 1 0001: divide 2 0010: divide 4 0011: divide 6 ... 1111: divide 30

Offset: 08h **PTCR08: Duty Control 0 Register** **Init = X**

Bit	R/W	Description
31:24	RW	**PWM B falling point bit [7:0] of period**
23:16	RW	**PWM B rising point bit [7:0] of period**
15:8	RW	**PWM A falling point bit [7:0] of period**
7 :0	RW	**PWM A rising point bit [7:0] of period**

Offset: 0Ch **PTCR0C: Duty Control 1 Register** **Init = X**

Bit	R/W	Description
31:24	RW	**PWM D falling point bit [7:0] of period**
23:16	RW	**PWM D rising point bit [7:0] of period**

to next page

from previous page

15:8	RW	**PWM C falling point bit [7:0] of period**
7 :0	RW	**PWM C rising point bit [7:0] of period**

Offset: 10h **PTCR10: Type M Control 0 Register** **Init = X**

Bit	R/W	Description
31:16	RW	**Type M fan tach period bit [15:0] (in unit of type M PWM clock)**
15:8		**Reserved (0)**
7	RW	**EnFlatM**
6	RW	**Reserved**
5 :4	RW	**Type M fan tach mode selection bit [1:0]** 00: falling edge 01: rising edge 10: both edges 11: reserved
3 :1	RW	**Type M fan tach clock division bit [1:0]** 000: divide 4 001: divide 16 010: divide 64 011: divide 256 ... 111: divide 65536
0	RW	**Enable fan tach of type M** 0: disable 1: enable

Offset: 14h **PTCR14: Type M Control 1 Register** **Init = X**

Bit	R/W	Description
31:16	RW	**Type M fan tach falling point bit [15:0] of period**
15:0	RW	**Type M fan tach rising point bit [15:0] of period**

Offset: 18h **PTCR18: Type N Control 0 Register** **Init = X**

Bit	R/W	Description
31:16	RW	**Type N fan tach period bit [15:0] (in unit of type N PWM clock)**
15:8		**Reserved (0)**
7	RW	**EnFlatN**
6	RW	**Reserved**
5 :4	RW	**Type N fan tach mode selection bit [1:0]** 00: falling edge 01: rising edge 10: both edges 11: reserved
3 :1	RW	**Type N fan tach clock division bit [1:0]** 000: div 4 001: div 16 010: div 64 011: div 256 ... 111: div 65536

to next page

from previous page

0	RW	**Enable fan tach of type N** 0: disable 1: enable

Offset: 1Ch	**PTCR1C: Type N Control 1 Register**	**Init = X**
Bit	**R/W**	**Description**
31:16	RW	**Type N fan tach falling point bit [15:0] of period**
15:0	RW	**Type N fan tach rising point bit [15:0] of period**

Offset: 20h	**PTCR20: Tach Source Register**	**Init = X**
Bit	**R/W**	**Description**
31:30	RW	**PWM source of fan tach #15 bit [1:0]** 000: PWM A 001: PWM B 010: PWM C 011: PWM D 100: PWM E 101: PWM F 110: PWM G 111: PWM H
29:28	RW	**PWM source of fan tach #14 bit [1:0]** 000: PWM A 001: PWM B 010: PWM C 011: PWM D 100: PWM E 101: PWM F 110: PWM G 111: PWM H
27:26	RW	**PWM source of fan tach #13 bit [1:0]** 000: PWM A 001: PWM B 010: PWM C 011: PWM D 100: PWM E 101: PWM F 110: PWM G 111: PWM H
25:24	RW	**PWM source of fan tach #12 bit [1:0]** 000: PWM A 001: PWM B 010: PWM C 011: PWM D 100: PWM E 101: PWM F 110: PWM G 111: PWM H

to next page

from previous page

23:22	RW	**PWM source of fan tach #11 bit [1:0]** 000: PWM A 001: PWM B 010: PWM C 011: PWM D 100: PWM E 101: PWM F 110: PWM G 111: PWM H
21:20	RW	**PWM source of fan tach #10 bit [1:0]** 000: PWM A 001: PWM B 010: PWM C 011: PWM D 100: PWM E 101: PWM F 110: PWM G 111: PWM H
19:18	RW	**PWM source of fan tach #9 bit [1:0]** 000: PWM A 001: PWM B 010: PWM C 011: PWM D 100: PWM E 101: PWM F 110: PWM G 111: PWM H
17:16	RW	**PWM source of fan tach #8 bit [1:0]** 000: PWM A 001: PWM B 010: PWM C 011: PWM D 100: PWM E 101: PWM F 110: PWM G 111: PWM H
15:14	RW	**PWM source of fan tach #7 bit [1:0]** 000: PWM A 001: PWM B 010: PWM C 011: PWM D 100: PWM E 101: PWM F 110: PWM G 111: PWM H

to next page

from previous page

13:12	RW	**PWM source of fan tach #6 bit [1:0]** 000: PWM A 001: PWM B 010: PWM C 011: PWM D 100: PWM E 101: PWM F 110: PWM G 111: PWM H
11:10	RW	**PWM source of fan tach #5 bit [1:0]** 000: PWM A 001: PWM B 010: PWM C 011: PWM D 100: PWM E 101: PWM F 110: PWM G 111: PWM H
9: 8	RW	**PWM source of fan tach #4 bit [1:0]** 000: PWM A 001: PWM B 010: PWM C 011: PWM D 100: PWM E 101: PWM F 110: PWM G 111: PWM H
7: 6	RW	**PWM source of fan tach #3 bit [1:0]** 000: PWM A 001: PWM B 010: PWM C 011: PWM D 100: PWM E 101: PWM F 110: PWM G 111: PWM H
5: 4	RW	**PWM source of fan tach #2 bit [1:0]** 000: PWM A 001: PWM B 010: PWM C 011: PWM D 100: PWM E 101: PWM F 110: PWM G 111: PWM H

to next page

from previous page

3: 2	RW	**PWM source of fan tach #1 bit [1:0]** 000: PWM A 001: PWM B 010: PWM C 011: PWM D 100: PWM E 101: PWM F 110: PWM G 111: PWM H
1: 0	RW	**PWM source of fan tach #0 bit [1:0]** 000: PWM A 001: PWM B 010: PWM C 011: PWM D 100: PWM E 101: PWM F 110: PWM G 111: PWM H

Offset: 28h **PTCR28: Trigger Register** **Init = X**

Bit	R/W	Description
31:16		**Reserved (0)**
15:0	RW	**Trigger to read fan tach #15 ~ fan tach #0 (0-to-1 trigger)** Bit[15]: 0-to-1 trigger fan tach #15 Bit[14]: 0-to-1 trigger fan tach #14 ... Bit[0]: 0-to-1 trigger fan tach #0

Offset: 2Ch **PTCR2C: Result Register** **Init = X**

Bit	R/W	Description
31	R	**Fan tach # full measurement status** 0: partial measurement 1: full measurement
30:20		**Reserved (0)**
19:0	R	**Measured fan tach # value bit [19:0]**

Offset: 30h **PTCR30: Interrupt Control Register** **Init = X**

Bit	R/W	Description
31:16		**Reserved (0)**
15:0	RW	**Enable fan tach #15 ~ fan tach #0 interrupt** Bit[15]: enable fan tach #15 interrupt 0: disable 1: enable Bit[14]: enable fan tach #14 interrupt 0: disable 1: enable ... Bit[0]: enable fan tach #0 interrupt 0: disable 1: enable

Offset: 34h	PTCR34: Interrupt Status Register	Init = X
Bit	**R/W**	**Description**
31:16		**Reserved (0)**
15:0	RW	**Fan tach #15 ~ fan tach #0 interrupt status** Bit[15]: fan tach #15 interrupt status 0: no interrupt 1: interrupt pending Bit[14]: fan tach #14 interrupt status 0: no interrupt 1: interrupt pending ... Bit[0]: fan tach #0 interrupt status 0: no interrupt 1: interrupt pending

Offset: 38h	PTCR38: Type M Limit Register	Init = X
Bit	**R/W**	**Description**
31:20		**Reserved (0)**
19:0	RW	**Type M fan tach limit bit [19:0]**

Offset: 3Ch	PTCR3C: Type N Limit Register	Init = X
Bit	**R/W**	**Description**
31:20		**Reserved (0)**
19:0	RW	**Type N fan tach limit bit [19:0]**

Offset: 40h	PTCR40: General Control Extension #1 Register	Init = 0x0000X000
Bit	**R/W**	**Description**
31:16		**Reserved (0)**
15	RW	**Type selection bit [0] of PWM H port** 00: type M 01: type N 1X: type O
14	RW	**Type selection bit [0] of PWM G port** 00: type M 01: type N 1X: type O
13	RW	**Type selection bit [0] of PWM F port** 00: type M 01: type N 1X: type O
12	RW	**Type selection bit [0] of PWM E port** 00: type M 01: type N 1X: type O
11	RW	**Enable PWM H port** 0: disable 1: enable
10	RW	**Enable PWM G port** 0: disable 1: enable

to next page

from previous page

9	RW	**Enable PWM F port** 0: disable 1: enable
8	RW	**Enable PWM E port** 0: disable 1: enable
7	RW	**Type selection bit [1] of PWM H port**
6	RW	**Type selection bit [1] of PWM G port**
5	RW	**Type selection bit [1] of PWM F port**
4	RW	**Type selection bit [1] of PWM E port**
3 :0	RW	**Reserved**

Offset: 44h **PTCR44: Clock Control Extension #1 Register** **Init = X**

Bit	R/W	Description
31:16		**Reserved (0)**
15:8	RW	**Type O PWM period bit [7:0] (in units of type O PWM clock)**
7 :4	RW	**Type O PWM clock division H bit [3:0]** 0000: divide 1 0001: divide 2 0010: divide 4 0011: divide 8 ... 1111: divide 32768
3 :0	RW	**Type O PWM clock division L bit [3:0]** 0000: divide 1 0001: divide 2 0010: divide 4 0011: divide 6 ... 1111: divide 30

Offset: 48h **PTCR48: Duty Control 2 Register** **Init = X**

Bit	R/W	Description
31:24	RW	**PWM F falling point bit [7:0] of period**
23:16	RW	**PWM F rising point bit [7:0] of period**
15:8	RW	**PWM E falling point bit [7:0] of period**
7 :0	RW	**PWM E rising point bit [7:0] of period**

Offset: 4Ch **PTCR4C: Duty Control 3 Register** **Init = X**

Bit	R/W	Description
31:24	RW	**PWM H falling point bit [7:0] of period**
23:16	RW	**PWM H rising point bit [7:0] of period**
15:8	RW	**PWM G falling point bit [7:0] of period**
7 :0	RW	**PWM G rising point bit [7:0] of period**

Offset: 50h	PTCR50: Type O Control 0 Register	Init = X

Bit	R/W	Description
31:16	RW	**Type O fan tach period bit [15:0] (in unit of type O PWM clock)**
15:8		**Reserved (0)**
7 :6	RW	**Reserved**
5 :4	RW	**Type O fan tach mode selection bit [1:0]** 00: falling edge 01: rising edge 10: both edges 11: reserved
3 :1	RW	**Type O fan tach clock division bit [1:0]** 000: divide 4 001: divide 16 010: divide 64 011: divide 256 ... 111: divide 65536
0	RW	**Enable fan tach of type O** 0: disable 1: enable

Offset: 54h	PTCR54: Type O Control 1 Register	Init = X

Bit	R/W	Description
31:16	RW	**Type O fan tach falling point bit [15:0] of period**
15:0	RW	**Type O fan tach rising point bit [15:0] of period**

Offset: 60h	PTCR60: Tach Source Extension #1 Register	Init = 0x00000000

Bit	R/W	Description
31		**Reserved (0)**
30	RW	**PWM source of fan tach #15 bit [2]**
29		**Reserved (0)**
28	RW	**PWM source of fan tach #14 bit [2]**
27		**Reserved (0)**
26	RW	**PWM source of fan tach #13 bit [2]**
25		**Reserved (0)**
24	RW	**PWM source of fan tach #12 bit [2]**
23		**Reserved (0)**
22	RW	**PWM source of fan tach #11 bit [2]**
21		**Reserved (0)**
20	RW	**PWM source of fan tach #10 bit [2]**
19		**Reserved (0)**
18	RW	**PWM source of fan tach #9 bit [2]**
17		**Reserved (0)**
16	RW	**PWM source of fan tach #8 bit [2]**
15		**Reserved (0)**
14	RW	**PWM source of fan tach #7 bit [2]**

to next page

from previous page

13		**Reserved (0)**
12	RW	**PWM source of fan tach #6 bit [2]**
11		**Reserved (0)**
10	RW	**PWM source of fan tach #5 bit [2]**
9		**Reserved (0)**
8	RW	**PWM source of fan tach #4 bit [2]**
7		**Reserved (0)**
6	RW	**PWM source of fan tach #3 bit [2]**
5		**Reserved (0)**
4	RW	**PWM source of fan tach #2 bit [2]**
3		**Reserved (0)**
2	RW	**PWM source of fan tach #1 bit [2]**
1		**Reserved (0)**
0	RW	**PWM source of fan tach #0 bit [2]**

Offset: 78h	**PTCR78: Type O Limit Register**	**Init = X**
Bit	**R/W**	**Description**
31:20		**Reserved (0)**
19:0	RW	**Type O fan tach limit bit [19:0]**

†† **RPM = (24000000*60) / (2 * TachoValue * TachoClkDivision)**

43 Virtual UART

43.1 Overview

AST2500 integrates a Virtual UART module providing virtual serial communication capabilities between host CPU and ARM CPU. The 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 (VUART28 and VUART2C)

Base Address of Virtual UART = 0x1E78_7000
Register Address of Virtual UART = (Base Address of VUART) + Offset

The following registers can be access by host CPU through LPC bus.

VUART00 (Host): Receiving Buffer Register (Read, DLAB = 0)
VUART00 (Host): Transmit Holding Register (Write, DLAB = 0)
VUART00 (Host): Divisor Latch Low Register (Read/Write, DLAB = 1)
VUART04 (Host): Interrupt Enable Register (Read/Write, DLAB = 0)
VUART04 (Host): Divisor Latch High Register (Read/Write, DLAB = 1)
VUART08 (Host): FIFO Control Register
VUART0C (Host): Line Control Register
VUART10 (Host): Modem Control Register
VUART14 (Host): Line Status Register
VUART18 (Host): Modem Status Register
VUART1C (Host): Scratch Register

The following registers can be access by ARM CPU through APB bus.

VUART00 (Slave): Receiving Buffer Register (Read, DLAB = 0)
VUART00 (Slave): Transmit Holding Register (Write, DLAB = 0)
VUART00 (Slave): Divisor Latch Low Register (Read/Write, DLAB = 1)
VUART04 (Slave): Interrupt Enable Register (Read/Write, DLAB = 0)
VUART04 (Slave): Divisor Latch High Register (Read/Write, DLAB = 1)
VUART08 (Slave): FIFO Control Register
VUART0C (Slave): Line Control Register
VUART10 (Slave): Modem Control Register
VUART14 (Slave): Line Status Register
VUART18 (Slave): Modem Status Register
VUART1C (Slave): Scratch Register
VUART20 (Slave): General Control Register A
VUART24 (Slave): General Control Register B
VUART28 (Slave): VUART Address Register L
VUART2C (Slave): VUART Address Register H
VUART30 (Slave): General Control Register E
VUART34 (Slave): General Control Register F
VUART38 (Slave): General Control Register G
VUART3C (Slave): General Control Register H

AST2500 also integrates a pass-through mode of UART1 or UART2. It creats a control path from the LPC bus, through AHB/APB, to UART1 or UART2. Host can directly access UART1 or UART2 by LPC I/O cycles without any firmware help. It could be used to replace one COM port of Super I/O on host side.

The base address for host CPU to access UART1 or UART2 registers through LPC bus can be programmed

by ARM CPU by the extended related registers (PUART28 and PUART2C)

Base Address of Pass-through PUART = 0x1E78_8000
Register Address of Pass-through UART = (Base Address of PUART) + Offset

The following registers can be accessed by ARM CPU through APB bus.

PUART20: General Control Register A
PUART24: General Control Register B
PUART28: PUART Address Register L
PUART2C: PUART Address Register H
PUART30: General Control Register E
PUART34: General Control Register F
PUART38: General Control Register G
PUART3C: General Control Register H

43.2 Features

- Directly connected to both APB bus and LPC Bus
- Support one Virtual UART interfaces and one Pass-through UART interface
- Separate transmit & receive FIFO buffer (16x8) to reduce CPU overhead
- Programmable base address for host CPU to access UART registers through LPC bus

43.3 VUART Registers : Base Address = 0x1E78:7000

Offset: 00h		VUART00 (Host)	Init = X
Bit	**R/W**	**Description**	
31:8		**Reserved (0)**	
RBR: Receiving Buffer Register (DLAB = 0)			
7 :0	R	**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	W	**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		VUART00 (Slave)	Init = X
Bit	**R/W**	**Description**	
31:8		**Reserved (0)**	
RBR: Receiving Buffer Register (DLAB = 0)			
7 :0	R	**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	W	**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 VUART34[2] = 0)			
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 VUART34[2] = 1)			
7 :0	R	**Divisor Latch Low Register (Host)** The slave (BMC) can read the DLL of the Host by this register.	

Offset: 04h		VUART04 (Host)	Init = 0
Bit	**R/W**	**Description**	
31:8		**Reserved (0)**	
IER: Interrupt Enable Register (DLAB = 0)			
7	RW	**Enable FIFO 1/2 full THRE Interrupt Mode if VUART34[6]=1** 0: Disable FIFO 1/2 full THRE interrupt mode 1: Enable FIFO 1/2 full THRE interrupt mode if VUART34[6]=1	
6 :4		**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)			

to next page

from previous page

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		**VUART04 (Slave)** — **Init = 0**
Bit	**R/W**	**Description**
31:8		**Reserved (0)**
		IER: Interrupt Enable Register (DLAB = 0)
7 :4		**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 and VUART34[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 VUART34[2] = 1)
7 :0	R	**Divisor latch (High) (Host)** The slave (BMC) can read the DLH of the Host by this register.

Offset: 08h		**VUART08 (Host): (IIR) Interrupt Identity Register** — **Init = 0xC1**
Bit	**R/W**	**Description**
31:4		**Reserved (0)**
3:1	R	**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	R	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.		

VUART 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	VUART08 (Host): (FCR) FIFO Control Register	Init = 0x01
Bit	**R/W**	**Description**
31:8		**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		**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	VUART08 (Slave): (IIR) Interrupt Identity Register	Init = 0xC1
Bit	**R/W**	**Description**
31:4		**Reserved (0)**

to next page

from previous page

3:1	R	**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	R	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.		

VUART 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 VUART34[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 **VUART08 (Slave): (FCR) FIFO Control Register** **Init = 0x01**

Bit	R/W	Description
31:8		**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		**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.

to next page

from previous page

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 **VUART0C (Host): (LCR) Line Control Register** **Init = 0x03**

Bit	R/W	Description
31:8		**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 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 **VUART0C (Slave): (LCR) Line Control Register** **Init = 0x03**

Bit	R/W	Description
31:8		**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	VUART10 (Host): (MCR) Modem Control Register	Init = 0
Bit	**R/W**	**Description**
31:8		**Reserved (0)**
7	R	**Transmit FIFO full.** 0: transmit FIFO not full. 1: transmit FIFO full.
6:5		**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: 10h	VUART10 (Slave): (MCR) Modem Control Register	Init = 0
Bit	**R/W**	**Description**
31:5		**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	VUART14 (Host): (LSR) Line Status Register	Init = 0x60
Bit	**R/W**	**Description**
31:7		**Reserved (0)**
6	R	**Transmitter empty** 1: The THR or FIFO and the Transmitter Shift Register are both empty. 0: Otherwise.
5	R	**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	R	**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		**Reserved (0)**
1	R	**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	R	**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	VUART14 (Slave): (LSR) Line Status Register	Init = 0x60
Bit	**R/W**	**Description**
31:7		**Reserved (0)**
6	R	**Transmitter empty** 1: The THR or FIFO and the Transmitter Shift Register are both empty. 0: Otherwise.
5	R	**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	R	**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		**Reserved (0)**
1	R	**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	R	**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	VUART18 (Host): (MSR) Modem Status Register	Init = X
Bit	**R/W**	**Description**
31:8		**Reserved (0)**
7	R	**Complement of the nDSR input or equals to Out2(MCR[3]) in loopback mode.**
6	R	**Out1(MCR[2]) in loopback mode.**

to next page

from previous page

5	R	**Complement of the nDSR input or equals to nDTR(MCR[0]) in loopback mode.**
4	R	**Complement of the nCTS input or equals to nRTS(MCR[1]) in loopback mode.**
3	R	**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)
2	R	**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	R	**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	R	**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 **VUART18 (Slave): (MSR) Modem Status Register** **Init = X**

Bit	R/W	Description
31:8		**Reserved (0)**
7	R	**Complement of the nDSR input or equals to Out2(MCR[3]) in loopback mode.**
6	R	**Out1(MCR[2]) in loopback mode.**
5	R	**Complement of the nDSR input or equals to nDTR(MCR[0]) in loopback mode.**
4	R	**Complement of the nCTS input or equals to nRTS(MCR[1]) in loopback mode.**
3	R	**Delta Data Carrier Detect (DDCD) indicator** 1: The nDSR line has changed its state since the last time the CPU read the MSR.
2	R	**Reserved (0)**
1	R	**Delta Data Set Ready (DDSR) indicator** 1: The nDSR line has changed its state since the last time the CPU read the MSR.
0	R	**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 **VUART1C (Host): (SCR) Scratch Register** **Init = X**

Bit	R/W	Description
31:8		**Reserved (0)**
7 :0	RW	**Scratch bits** This register can be used as a temporary storage, no specific definition.

Offset: 1Ch **VUART1C (Slave): (SCR) Scratch Register** **Init = X**

Bit	R/W	Description
31:8		**Reserved (0)**
7 :0	RW	**Scratch bits (VUART34[2] = 0)** This register can be used as a temporary storage, no specific definition.
7 :0	R	**Scratch bits (VUART34[2] = 1)** The slave (BMC) can read the SCR of the Host by this register.

Offset: 20h	VUART20 (Slave): General Control Register A	Init = 0b00x0_xx00

Bit	R/W	Description
31:8		**Reserved (0)**
7 :6	R	**Status of host-side Receiver FIFO Trigger**
5	RW	**Disable Host-Tx-discard mode** 0: Enable Host-Tx-discard mode Slave will throw data away automatically when Host Tx FIFO is not empty. 1: Disable Host-Tx-discard mode
4	R	**Status of host-side loopback mode**
3 :2	RW	**Slave-side timeout time width selection bit [1:0]** 00: 1/64 second if PUART34[3]=0 and SCU2C[13]=0 01: 1/128 second if PUART34[3]=0 and SCU2C[13]=0 10: 1/256 second if PUART34[3]=0 and SCU2C[13]=0 11: 1/512 second if PUART34[3]=0 and SCU2C[13]=0 00: 1/3600 second if PUART34[3]=0 and SCU2C[13]=1 01: 1/7200 second if PUART34[3]=0 and SCU2C[13]=1 10: 1/14400 second if PUART34[3]=0 and SCU2C[13]=1 11: 1/28800 second if PUART34[3]=0 and SCU2C[13]=1 00: 512*LCLK if PUART34[3]=1 01: 256*LCLK if PUART34[3]=1 10: 128*LCLK if PUART34[3]=1 11: 64*LCLK if PUART34[3]=1
1	RW	**SIRQ polarity** 0: The output is high impedance when host interrupt has been cleared. 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.
0	RW	**Enable virtual UART** 0: Disable 1: Enable

Note :
This register is defined for ARM CPU only.

Offset: 24h	VUART24 (Slave): General Control Register B	Init = 0bxxxx_xx11

Bit	R/W	Description
31:8		**Reserved (0)**
7 :4	RW	**SIRQ number selection bit [3:0]** 0000: IRQ0 0001: IRQ1 0010: SMI ... 1111: IRQ15

to next page

from previous page

3 :2	RW	**Host-side timeout period selection** 00: 1/64 second if PUART34[3]=0 and SCU2C[13]=0 01: 1/128 second if PUART34[3]=0 and SCU2C[13]=0 10: 1/256 second if PUART34[3]=0 and SCU2C[13]=0 11: 1/512 second if PUART34[3]=0 and SCU2C[13]=0 00: 1/3600 second if PUART34[3]=0 and SCU2C[13]=1 01: 1/7200 second if PUART34[3]=0 and SCU2C[13]=1 10: 1/14400 second if PUART34[3]=0 and SCU2C[13]=1 11: 1/28800 second if PUART34[3]=0 and SCU2C[13]=1 00: 512*PCLK if PUART34[3]=1 01: 256*PCLK if PUART34[3]=1 10: 128*PCLK if PUART34[3]=1 11: 64*PCLK if PUART34[3]=1
1 :0	R	**Number of bits per character (host-side)**
Note : This register is defined for ARM CPU only.		

Offset: 28h **VUART28 (Slave): Virtual UART Address Register L** **Init = X**

Bit	R/W	Description
31:8		**Reserved (0)**
7 :0	RW	**Virtual UART address bit [7:0]** This register defines the base address (the low bytes) for host CPU to access virtual UART registers through LPC bus.
Note : This register is defined for ARM CPU only.		

Offset: 2Ch **VUART2C (Slave): Virtual UART Address Register H** **Init = X**

Bit	R/W	Description
31:8		**Reserved (0)**
7 :0	RW	**Virtual 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 **VUART30 (Slave): General Control Register E** **Init = 0b0000_1110**

Bit	R/W	Description
31:8		**Reserved (0)**
7	R	**Transmit FIFO full. (slave-side)** 0: transmit FIFO not full. 1: transmit FIFO full.
6 :4	R	**THR read pointer bit [2:0] (slave-side)**
3 :0	R	**Complement of IIR status bit [3:0] (host-side)**
Note : This register is defined for ARM CPU only.		

Offset: 34h	VUART34 (Slave): General Control Register F	Init = X
Bit	**R/W**	**Description**
31:8		**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	VUART38 (Slave): General Control Register G	Init = X
Bit	**R/W**	**Description**
31:8		**Reserved (0)**
7 :0	R	**THR read back data bit [7:0] (slave-side)**
Note : This register is defined for ARM CPU only.		

Offset: 3Ch	VUART3C (Slave): General Control Register H	Init = X
Bit	**R/W**	**Description**
31:8		**Reserved (0)**
7 :6	R	**Read back data bit [1:0] of receiver FIFO interrupt trigger level (slave-side)**
5	R	**Interrupt Enable Register bit [7] on the Host side**
4	R	**Enable UART FIFO on the Host side**
3 :0	R	**Interrupt Enable Register bit [3:0] on the Host side**
Note : This register is defined for ARM CPU only.		

43.4 PUART Registers : Base Address = 0x1E78:8000

Offset: 20h	PUART20: General Control Register A	Init = 0b00x0_xx00
Bit	**R/W**	**Description**
31:8		**Reserved (0)**
7 :6	R	**Status of host-side Receiver FIFO Trigger**
5		**Reserved (0)**
4	R	**Status of host-side loopback mode**
3 :2		**Reserved (0)**

to next page

from previous page

1	RW	**SIRQ polarity** 0: The output is high impedance when host interrupt has been cleared. 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.
0	RW	**Enable pass-through UART** 0: Disable 1: Enable
Note : This register is defined for ARM CPU only.		

Offset: 24h **PUART24: General Control Register B** **Init = 0bxxxx_xx11**

Bit	R/W	Description
31:8		**Reserved (0)**
7 :4	RW	**SIRQ number selection bit [3:0]** 0000: IRQ0 0001: IRQ1 0010: SMI ... 1111: IRQ15
3 :2		**Reserved (0)**
1 :0	R	**Number of bits per character (host-side)**
Note : This register is defined for ARM CPU only.		

Offset: 28h **PUART28: Pass-through UART Address Register L** **Init = X**

Bit	R/W	Description
31:8		**Reserved (0)**
7 :0	RW	**Pass-through UART address bit [7:0]** This register defines the base address (the low bytes) for host CPU to access pass-through UART registers through LPC bus.
Note : This register is defined for ARM CPU only.		

Offset: 2Ch **PUART2C: Pass-through UART Address Register H** **Init = X**

Bit	R/W	Description
31:8		**Reserved (0)**
7 :0	RW	**Pass-through UART address bit [15:8]** This register defines the base address (the high bytes) for host CPU to access pass-through UART registers through LPC bus.
Note : This register is defined for ARM CPU only.		

Offset: 30h **PUART30: General Control Register E** **Init = 0**

Bit	R/W	Description
31:8		**Reserved (0)**

to next page

from previous page

7	R	**Complement of Enable Transmitter Holding Register Empty Interrupt (host-side)**
6	R	**Complement of Enable Received Data Available Interrupt (host-side)**
5 :0	R	**Complement of LCR bit [7:2] (host-side)**
Note : This register is defined for ARM CPU only.		

Offset: 34h **PUART34: General Control Register F** **Init = X**

Bit	R/W	Description
31:8		**Reserved (0)**
7	RW	**Pass-through mode selection** 0: Pass-through mode for UART1 1: Pass-through mode for UART2
6 :5	RW	**Reserved**
4	RW	**Disable SIRQ of PUART**
3	RW	**VUART timeout time width control bit (slave-side)**
2	RW	**VUART timeout time width control bit (host-side)**
1 :0	RW	**Reserved**
Note : This register is defined for ARM CPU only.		

Offset: 38h **PUART38: General Control Register G** **Init = X**

Bit	R/W	Description
31:8		**Reserved (0)**
7 :0		**Reserved**
Note : This register is defined for ARM CPU only.		

Offset: 3Ch **PUART3C: General Control Register H** **Init = 0x00**

Bit	R/W	Description
31:8		**Reserved (0)**
7	R	**Complement of Enable Modem Status Interrupt (host-side)**
6	R	**Complement of Enable Receiver Line Status Interrupt (host-side)**
5 :0		**Reserved (0)**
Note : This register is defined for ARM CPU only.		

44 LPC Controller

44.1 Overview

AST2500 integrates both LPC Host Controller and LPC Slave Controller, but only one of the two controllers can be enabled at one time. LPC Slave Controller also integrates IPMI 2.0/1.1 compliant BMC controller. There are totally 81 registers, which is listed below, to control the various functions supported by AST2500 . 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
IDR5 : Input Data Register 5
ODR5 : Output Data Register 5
STR5 : Status Register 5
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
MBXSTS_0 : MailBox Status Register 0
MBXSTS_1 : MailBox Status Register 1
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
MBXHIE_0 : MailBox Host Interrupt Enable Register 0
MBXHIE_1 : MailBox Host Interrupt Enable Register 1

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 AST2500 .

44.2 Features

- Directly connected to APB bus interface
- Dual operation modes
  - Master mode: designed to update system BIOS, TPM or LPC controller (I/O, memory, firmware read write cycles)
  - 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 one 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 64x8 Embedded SRAM for BT mode support

- Two sets of 64x8 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

44.3 Registers : Base Address = 0x1E78:9000

Attribute Definition:

Attribute	:	Description
R	:	Readable
W	:	Writable
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	R	-	**Reserved**
3	**SDWNE**	0H	RW	-	**Enable LPC software shutdown**
2	**PMEE**	0H	RW	-	**Enable PME output**
1		0	R	-	**Reserved**
0		0	R	-	**Reserved**

HICR1: Host Interface Control Register 1 **Offset: 04h**

Bit	Name	Initial	Slave	Host	Description
7	**LPCBSY**	0L	R	-	**LPC busy flag**
6		0	R	-	**Reserved**
5	**IRQBSY**	0L	R	-	**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	R	-	**Reserved**
0		0	R	-	**Reserved**

HICR2: Host Interface Control Register 2 **Offset: 08h**

Bit	Name	Initial	Slave	Host	Description
7		0	R	-	**Reserved**
6	**LRST**	0P	RW0C	-	**LPC reset interrupt status**
5	**SDWN**	0P	RW0C	-	**LPC shutdown interrupt status**
4	**ABRT**	0P	RW0C	-	**LPC Abort Interrupt status**
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**

HICR3: Host Interface Control Register 3 **Offset: 0Ch**

Bit	Name	Initial	Slave	Host	Description
7	**LFRAME**	U	R	-	**LFRAME pin monitoring**
6		0	R	-	**Reserved**
5	**SERIRQ**	U	R	-	**SERIRQ pin monitoring**
4	**LRESET**	U	R	-	**LRESET pin monitoring**
3	**LPCPD**	U	R	-	**LPCPD pin monitoring**
2	**PME**	U	R	-	**PME pin monitoring**
1		0	R	-	**Reserved**
0		0	R	-	**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	R	-	**Status of LRESET rising interrupt**
4	**EnIntLRstR**	0H	RW	-	**Enable LRESET rising interrupt**
3		0	R	-	**Reserved**
2	**KCSENBL**	0H	RW	-	**Enable KCS interface in Channel #3**
1		0	R	-	**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	**Bit3**	0H	RW	-	**Channel #3 address Bit[3]**
2		0	R	-	**Reserved**
1	**Bit1**	0H	RW	-	**Channel #3 address Bit[1]**
0		0	R	-	**Reserved**

LADR12H: LPC Channel #1/#2 Address Register H **Offset: 1Ch**

Bit	Name	Initial	Slave	Host	Description
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
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	R	W	**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	R	W	**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	R	W	**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	R	**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	R	**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	R	**Channel #3 output data Bit[7:0]**

STR1: Status Register 1 **Offset: 3Ch**

Bit	Name	Initial	Slave	Host	Description
7	**DBU17**	0H	RW	R	**Defined by user**
6	**DBU16**	0H	RW	R	**Defined by user**
5	**DBU15**	0H	RW	R	**Defined by user**
4	**DBU14**	0H	RW	R	**Defined by user**
3	**C/D1**	0P	R	R	**Command/Data**
2	**DBU12**	0H	RW	R	**Defined by user**
1	**IBF1**	0P	R	R	**Input data register full**
0	**OBF1**	0P	RW0C	R	**Output data register full**

STR2: Status Register 2 **Offset: 40h**

Bit	Name	Initial	Slave	Host	Description
7	**DBU27**	0H	RW	R	**Defined by user**
6	**DBU26**	0H	RW	R	**Defined by user**
5	**DBU25**	0H	RW	R	**Defined by user**
4	**DBU24**	0H	RW	R	**Defined by user**
3	**C/D2**	0P	R	R	**Command/Data**
2	**DBU22**	0H	RW	R	**Defined by user**
1	**IBF2**	0P	R	R	**Input data register full**
0	**OBF2**	0P	RW0C	R	**Output data register full**

STR3: Status Register 3 **Offset: 44h**

Bit	Name	Initial	Slave	Host	Description
7	**DBU37**	0H	RW	R	**Defined by user**
6	**DBU36**	0H	RW	R	**Defined by user**
5	**DBU35**	0H	RW	R	**Defined by user**
4	**DBU34**	0H	RW	R	**Defined by user**
3	**C/D3**	0P	R	R	**Command/Data**
2	**DBU32**	0H	RW	R	**Defined by user**
1	**IBF3**	0P	R	R	**Input data register full**
0	**OBF3**	0P	RW0C	R	**Output data register full**

BTR0: BT Status Register 0 **Offset: 48h**

Bit	Name	Initial	Slave	Host	Description
7		0	R	-	**Reserved**
6		0	R	-	**Reserved**
5		0	R	-	**Reserved**
4	**FRDI**	0H	RW0C	-	**FIFO read request interrupt**

to next page

from previous page

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	R	-	**Reserved**
6	**HRSTI**	0H	RW0C	-	**BT reset interrupt**
5		0	R	-	**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	R	-	**Reserved**
6	**FSEL1**	0H	RW	-	**BT Transfer FIFO selection bit 1**
5	**FSEL0**	0H	RW	-	**T 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**
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	R	-	**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	R	**BT write transfer busy flag**
6	**H_BUSY**	0H	R	W1T	**BT read transfer busy flag**

to next page

from previous page

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	R	R	**Reserved**
5		0	R	R	**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	R	R	**Reserved**
0		0	R	R	**Reserved**

BTFVSR0: BT FIFO Valid Size Register 0 **Offset: 64h**

Bit	Name	Initial	Slave	Host	Description
7:0	**N7 to N0**	0H	R	-	**Valid bytes in the FIFO for host write transfer**

BTFVSR1: BT FIFO Valid Size Register 1 **Offset: 68h**

Bit	Name	Initial	Slave	Host	Description
7:0	**N7 to N0**	0H	R	-	**Valid bytes in the FIFO for host read transfer**

SIRQCR0: SERIRQ Control Register 0 **Offset: 70h**

Bit	Name	Initial	Slave	Host	Description
7	**Q_C**	0L	R	-	**Quiet/Continuous Mode Flag**
6		0H	RW	-	**Reserved**
5	**IEDIR**	0H	RW	-	**Interrupt Enable Direct Mode**
4		0	R	-	**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
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**
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
7	**IEDIR3**	0H	RW	-	**Interrupt Enable Direct Mode 3**
6:0		0	R	-	**Reserved**

SIRQCR3: SERIRQ Control Register 3 **Offset: 7Ch**

Bit	Name	Initial	Slave	Host	Description
7		0	R	-	**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	-	**EnSIOGIO**
30		0P	RW	-	**En80hGIO**
29		0P	RW	-	**EnInvGIO**
28:24		U	RW	-	**Sel80hGIO**
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

to next page

from previous page

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).
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
31:17		U	RW	-	**Reserved**
16		0P	R	-	**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.
15:8		0	R	-	**Reserved**
7	**SIRQSTOP**	0L	R	-	**Reserved**
6	**ST_ENPME**	0P	R	-	**Reserved**
5		0	R	-	**Reserved**

to next page

from previous page

4		0	R	-	**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]**	U	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]**	U	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	R	-	**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	R	-	**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	R	-	**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	R	-	**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**
31:12		0	R	-	**Reserved**
11:8	**sel6IO**	AP	RW	-	**0000 : Route UART1 to IO6** **0001 : Route UART2 to IO6** **0010 : Route UART3 to IO6** **0011 : Route UART4 to IO6** **0100 : Route UART5 to IO6** **0101 : Route IO1 to IO6** **0110 : Route IO2 to IO6** **0111 : Route IO3 to IO6** **1000 : Route IO4 to IO6** **1001 : Route IO5 to IO6** **others: Reserved**
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	R	-	**Reserved**
2	**EnU5d13**	0P	RW	-	**Enable UART5 divide 13**
1: 0		0	R	-	**Reserved**

HICRA: Host Interface Control Register A					Offset: 9Ch
Bit	**Name**	**Initial**	**Slave**	**Host**	**Description**
31:28	**sel5DW**	0P	RW	-	**0000 : Route IO5 to UART5** **0001 : Route IO1 to UART5** **0010 : Route IO2 to UART5** **0011 : Route IO3 to UART5** **0100 : Route IO4 to UART5** **0101 : Route UART1 to UART5** **0110 : Route UART2 to UART5** **0111 : Route UART3 to UART5** **1000 : Route UART4 to UART5** **1001 : Route IO6 to UART5** **others: Reserved**
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**
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**
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**
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**
15		0P	RW	-	**Reserved**
14:12	**sel5IO**	0P	RW	-	**000 : Route UART5 to IO5** **001 : Route UART1 to IO5** **010 : Route UART2 to IO5** **011 : Route UART3 to IO5** **100 : Route UART4 to IO5** **101 : Route IO1 to IO5** **110 : Route IO3 to IO5** **111 : Route IO6 to IO5**

to next page

from previous page

11: 9	**sel4IO**	0P	RW	-	**000 : Route UART4 to IO4** **001 : Route UART5 to IO4** **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**
8: 6	**sel3IO**	0P	RW	-	**000 : Route UART3 to IO3** **001 : Route UART4 to IO3** **010 : Route UART5 to IO3** **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**
5: 3	**sel2IO**	0P	RW	-	**000 : Route UART2 to IO2** **001 : Route UART3 to IO2** **010 : Route UART4 to IO2** **011 : Route UART5 to IO2** **100 : Route UART1 to IO2** **101 : Route IO3 to IO2** **110 : Route IO4 to IO2** **111 : Route IO6 to IO2**
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 UART5 to IO1** **101 : Route IO3 to IO1** **110 : Route IO4 to IO1** **111 : Route IO6 to IO1**

LHCR0: LPC Host Control Register 0 **Offset: A0h**

Bit	Name	Initial	Slave	Host	Description
31		U	R	-	**Inputs of LPC SIRQ pin**
30		U	R	-	**Inputs of LPC LPCPDN pin**
29		U	R	-	**Inputs of LPC LCLK pin**
28		U	R	-	**Inputs of LPC FrameN pin**
27:24		U	R	-	**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

to next page

from previous page

14		1H	RW	-	**LPC Host LPCPDN pin output control**
13		1H	RW	-	**Reserved**
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	R	-	**Reserved**
4	**ENP2L**	0H	RW	-	**Enable APB to LPC bridge**
3		0	R	-	**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 AST2500 .

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**
14:2		0	R	-	**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	R	-	**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	R	-	**Reserved**
3	**ENLHTO-INT**	0H	RW	-	**Enable LPC host time-out interrupt** 0: Disable 1: Enable
2	**ENLHES-INT**	0H	RW	-	**Enable LPC host sync error interrupt** 0: Disable 1: Enable

to next page

from previous page

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	R	-	**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	R	-	**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
31:28	**P2LBASE**	U	RW	-	**Remapping address base bit [31:28] of APB to LPC bridge** Note: The default address decoding base bit [31:0] of APB is 0x50000000 with the range 256MB.
27:8		0	R	-	**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	R	-	**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	R	-	**LPC host SIRQ frame number**
26:24		0L	R	-	**LPC host SIRQ sub-frame number**
23:21		0	R	-	**Reserved**
20: 0		0L	R	-	**LPC host SIRQ state bit [20:0]**

PCCR6: Post Code Control Register 6 **Offset: C4h**

Bit	Name	Initial	Slave	Host	Description
31:30		0	R	-	**Reserved**
29:28		0	R	-	**Post Code DMA current write pointer bit[1:0]**
27: 0		0	R	-	**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	R	-	**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	R	-	**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	R	-	**Reserved**

PCCR5: Post Code Control Register 5 **Offset: D4h**

Bit	Name	Initial	Slave	Host	Description
31:11		0	R	-	**Reserved**
10: 0		U	RW	-	**Post Code DMA length bit[10:0]**

HICRB: Host Interface Control Register B **Offset: 100h**

Bit	Name	Initial	Slave	Host	Description
31:16		0	R	-	**Reserved**
15		0P	RW	-	**EnSNP1D**
14		0P	RW	-	**EnSNP0D**
13		0P	RW	-	**En80hSGIO**
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**
6		0P	RW	-	**Disable SIO iLPC2AHB function**
5		0P	RW	-	**En16LADR2**
4		0P	RW	-	**En16LADR1**
3		0P	RW	-	**Enable KCS channel #5 receive completion interrupt**
2		0P	RW	-	**Enable KCS channel #5 OBF and DBU interrupt**
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
31:16		0	R	-	**Reserved**

to next page

from previous page

15:14		3P	RW	-	**Reserved**
13		0P	RW	-	**Disable Host SerIRQ interrupt of KCS channel #5 auto-clear by DBU5**
12		0P	RW	-	**Host SerIRQ interrupt enable for KCS channel #5M**
11:10		3P	RW	-	**Reserved**
9		0P	RW	-	**Disable Host SerIRQ interrupt of KCS channel #5 auto-clear by OBF5**
8		0P	RW	-	**Host SerIRQ interrupt enable for KCS channel #5K**
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
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: 8		0P		-	**En485Sel**
7: 4		0P		-	**Reserved**
3: 0		0P	RW	-	**BMC to SIO interrupt bit[3:0] by SIORx_20[7:4]**

HISR1: Host Interface Scratch Register 1 **Offset: 10Ch**

Bit	Name	Initial	Slave	Host	Description
31		0P	R	-	**Status of SIO SMI enable**
30		0P	R	-	**Status of SIO iLPC2AHB enable**
29		0P	R	-	**Status of SIO GPIO enable**
28		0P	R	-	**Status of SIO KBC enable**
27		0P	R	-	**Status of SIO SUART2 enable**
26		0P	R	-	**Status of SIO SUART1 enable**
25		0P	R	-	**Status of SIO SUART4 enable**
24		0P	R	-	**Status of SIO SUART3 enable**
23:11		0P	RW	-	**Reserved**
10		0P	RW	-	**clr_str5_IBF**
9		0P	RW	-	**clr_str4_IBF**
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_20[3:0]**
3: 0		0P	R	-	**SIORx_20**

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	R	W	**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	R	**KCS channel #4 output data bit[7:0]**

STR4: Status Register 4 **Offset: 11Ch**

Bit	Name	Initial	Slave	Host	Description
7	**DBU47**	1H	RW	R	**Defined by user**
6	**DBU46**	1H	RW	R	**Defined by user**
5	**DBU45**	0H	RW	R	**Defined by user**
4	**DBU44**	0H	RW	R	**Defined by user**
3	**C/D4**	0P	R	R	**Command/Data**
2	**DBU42**	0H	RW	R	**Defined by user**
1	**IBF4**	0P	R	R	**Input data register full**
0	**OBF4**	0P	RW0C	R	**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]**

IDR5: Input Data Register 5 **Offset: 124h**

Bit	Name	Initial	Slave	Host	Description
7: 0		U	R	W	**KCS channel #5 input data bit[7:0]**

ODR5: Output Data Register 5 **Offset: 128h**

Bit	Name	Initial	Slave	Host	Description
7: 0		U	RW	R	**KCS channel #5 output data bit[7:0]**

STR5: Status Register 5 **Offset: 12Ch**

Bit	Name	Initial	Slave	Host	Description
9		0	R	-	**Reserved**
8		0P	R	-	**Reserved**

to next page

from previous page

7	**DBU57**	0H	RW	R	**Defined by user**
6	**DBU56**	0H	RW	R	**Defined by user**
5	**DBU55**	0H	RW	R	**Defined by user**
4	**DBU54**	0H	RW	R	**Defined by user**
3	**C/D5**	0P	R	R	**Command/Data**
2	**DBU52**	0H	RW	R	**Defined by user**
1	**IBF5**	0P	R	R	**Input data register full**
0	**OBF5**	0P	RW0C	R	**Output data register full**

PCCR0: Post Code Control Register 0 **Offset: 130h**

Bit	Name	Initial	Slave	Host	Description
31		0P	RW	-	**Enable post code DMA interrupt**
30:29		U	RW	-	**DMA timer selection**
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**
15		0	R	-	**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
11		0	R	-	**Reserved**
10: 8		U	RW	-	**Trigger level bit[2:0] of Rx 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		0	RW1S	-	**Enable write of FIFO read pointer bit[6:0]**
30:24		0P	RW	-	**FIFO read pointer bit[6:0]**
23:22		0	R	-	**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		0	R	-	**Reserved**
30:24		0P	R	-	**FIFO write pointer bit[6:0]**
23:18		0P	R	-	**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	R	-	**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: 5		0	R	-	**Reserved**
4		0P	RW1C	-	**Data ready indicator (read only) if non-DMA mode DMA done indicator (read and write 1 clear) if DMA mode**
3		0P	RW1C	-	**Interrupt status of RxOvr**
2		0P	R	-	**Interrupt status of RxTO**
1		0P	R	-	**Interrupt status of RxAva**
0		0P	R	-	**Interrupt status of the post code function**

PCCR3: Post Code Control Register 3 **Offset: 13Ch**

Bit	Name	Initial	Slave	Host	Description
31		0P	R	-	**FIFO full indicator**
30:24		0P	R	-	**FIFO write pointer bit[6:0]**
23		0P	R	-	**Data ready indicator**
22:16		0P	R	-	**FIFO read pointer bit[6:0]**
15: 8		0P	R	-	**Rx data count bit[7:0]**
7: 0		U	R	-	**FIFO data bit[7:0]**

to next page

from previous page

Note :
The data structure of the FIFO is mode dependent.

For I/O 1B mode:
dataX[7:0], dataY[7:0], ...

For I/O 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 I/O 4B mode:
dataX[7:0], abort_write_sirqTa_sirqTb_ioCycN_fwhCyc_In4679[1:0],
addrX[7:0], addrX[15:8],
dataY[7:0], abort_write_sirqTa_sirqTb_ioCycN_fwhCyc_In4679[1:0],
addrY[7:0], addrY[15:8], ...

For full mode:
dataX[7:0], abort_write_sirqTa_sirqTb_ioCycN_fwhCyc_In4679[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_In4679[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	R	-	**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
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
31:18		0	R	-	**Reserved**
17:16	**EnSIRQOEM**	0H	RW	-	**Reserved**

to next page

from previous page

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	R	-	**FIFO data bit[7:0] of BMC to Host**
23:16		U	R	-	**FIFO data bit[7:0] of Host to BMC**
15		0P	R	-	**Interrupt status of iBT BMC_HWRST**
14		0H	R	-	**Interrupt status of iBT OEM3**
13		0H	R	-	**Interrupt status of iBT OEM2**
12		0H	R	-	**Interrupt status of iBT OEM1**
11		0H	RW1C	-	**Interrupt status of iBT slave read overrun**
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		0	R	-	**Reserved**
30:24		0L	R	-	**FIFO read pointer bit[6:0] of BMC to Host**
23		0	R	-	**Reserved**
22:16		0H	R	-	**FIFO write pointer bit[6:0] of BMC to Host**

to next page

from previous page

15		0	R	-	**Reserved**
14: 8		0H	R	-	**FIFO read pointer bit[6:0] of Host to BMC**
7		0	R	-	**Reserved**
6: 0		0L	R	-	**FIFO write pointer bit[6:0] of Host to BMC**

iBTCR4: iBT Control Register 4 **Offset: 150h**

Bit	Name	Initial	Slave	Host	Description
31: 8		0	R	-	**Reserved**
7		1H	RW1T	R	**B_BUSY**
6		0L	R	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	R	-	**Reserved**
7: 0		U	RW	RW	**FIFO data bit[7:0]**

iBTCR6: iBT Control Register 6 **Offset: 158h**

Bit	Name	Initial	Slave	Host	Description
31: 8		0	R	-	**Reserved**
7		0P	RW1C	RW1S	**BMC_HWRST**
6: 5		0H	RW	RW	**Reserved**
4		0H	RW	RW	**OEM3**
3		0H	RW	RW	**OEM2**
2		0H	RW	RW	**OEM1**
1		0L	R	RW	**B2H_IRQ**
0		0L	R	RW	**B2H_IRQ_EN**

SRUART1: Status Register of UART1 **Offset: 160h**

Bit	Name	Initial	Slave	Host	Description
31:30		0	R	-	**RxFIFO trigger level bit [1:0]**
29		0	R	-	**Buadrate write**
28		0	R	-	**EnFIFO**
27:24		0	R	-	**MSR bit [3:0]**
23:16		0	R	-	**LSR bit [7:0]**
15:12		1	R	-	**IIR bit [3:0]**
11: 0		0	R	-	**Buadrate bit [11:0]**

SRUART2: Status Register of UART2 **Offset: 164h**

Bit	Name	Initial	Slave	Host	Description
31:30		0	R	-	**RxFIFO trigger level bit [1:0]**
29		0	R	-	**Buadrate write**
28		0	R	-	**EnFIFO**
27:24		0	R	-	**MSR bit [3:0]**
23:16		0	R	-	**LSR bit [7:0]**
15:12		1	R	-	**IIR bit [3:0]**
11: 0		0	R	-	**Buadrate bit [11:0]**

SRUART3: Status Register of UART3 **Offset: 168h**

Bit	Name	Initial	Slave	Host	Description
31:30		0	R	-	**RxFIFO trigger level bit [1:0]**
29		0	R	-	**Buadrate write**
28		0	R	-	**EnFIFO**
27:24		0	R	-	**MSR bit [3:0]**
23:16		0	R	-	**LSR bit [7:0]**
15:12		1	R	-	**IIR bit [3:0]**
11: 0		0	R	-	**Buadrate bit [11:0]**

SRUART4: Status Register of UART4 **Offset: 16Ch**

Bit	Name	Initial	Slave	Host	Description
31:30		0	R	-	**RxFIFO trigger level bit [1:0]**
29		0	R	-	**Buadrate write**
28		0	R	-	**EnFIFO**
27:24		0	R	-	**MSR bit [3:0]**
23:16		0	R	-	**LSR bit [7:0]**
15:12		1	R	-	**IIR bit [3:0]**
11: 0		0	R	-	**Buadrate bit [11:0]**

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		0P	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	R	-	**SIORx_23**
23:16		0P	R	-	**SIORx_22**
15: 8		0P	R	-	**SIORx_21**
7: 0		0P	R	-	**SIORx_20**

SCR3SIO: Scratch Register 3 of SIO **Offset: 17Ch**

Bit	Name	Initial	Slave	Host	Description
31:24		0P	R	-	**SIORx_27**
23:16		0P	R	-	**SIORx_26**
15: 8		0P	R	-	**SIORx_25**
7: 0		0P	R	-	**SIORx_24**

SWCR_03_00: SWC Register 03_00 **Offset: 180h**

Bit	Name	Initial	Slave	Host	Description
31		0P	RW1C	-	**Setting sleep state event status**
30		0P	R	-	**PWRGD status after debounce**
29		0P	RW1C	-	**PWRGD rising event status**
28		0P	RW1C	-	**PWRGD falling event status**
27		1P	R	-	**PWRBTN status after debounce**
26		0P	RW1C	-	**PWRBTN rising event status**
25		0P	RW1C	-	**PWRBTN falling event status**
24		1P	R	-	**RI status after debounce**
23		0P	RW1C	-	**RI rising event status**
22		0P	RW1C	-	**RI falling event status**
21		0P	R	-	**S5n status after debounce**
20		0P	RW1C	-	**S5n rising event status**
19		0P	RW1C	-	**S5n falling event status**
18		0P	R	-	**S3n status after debounce**
17		0P	RW1C	-	**S3n rising event status**
16		0P	RW1C	-	**S3n falling event status**
15		1P	R	-	**PWBTO raw status**
14		0P	R	-	**Last_ONCTL status**
13		1P	RW	-	**Was_pfail status**
12		0P	RW1C	-	**Crowbar status**
11		0P	RW1C	-	**PWRBTN Override status**
10				-	**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				-	**Reserved**
29		0P	RW	-	**Enable BMC interrupt as PWRGD rising event**
28		0P	RW	-	**Enable BMC interrupt as PWRGD falling event**
27				-	**Reserved**
26		0P	RW	-	**Enable BMC interrupt as PWRBTN rising event**
25		0P	RW	-	**Enable BMC interrupt as PWRBTN falling event**
24				-	**Reserved**
23		0P	RW	-	**Enable BMC interrupt as RI rising event**
22		0P	RW	-	**Enable BMC interrupt as RI falling event**
21				-	**Reserved**
20		0P	RW	-	**Enable BMC interrupt as S5n rising event**
19		0P	RW	-	**Enable BMC interrupt as S5n falling event**
18				-	**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				-	**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
31:26				-	**Reserved**
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				-	**Reserved**
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				-	**Reserved**
26:24		5P	RW	-	**Sleep type encoding of S3**
23:19				-	**Reserved**
18:16		2P	RW	-	**Sleep type encoding of S2**
15:11				-	**Reserved**
10: 8		1P	RW	-	**Sleep type encoding of S1**
7: 3				-	**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]**
23				-	**Reserved**

to next page

from previous page

22:21		0P	R	-	**slp_now_st[1:0]**
20		0P	RW	-	**Lock PWRBTN**
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				-	**Reserved**
10: 8		7P	RW	-	**Sleep type encoding of S5**
7: 3				-	**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	R	-	**SMI current minute count bit[7:0] (unit 1 min)**
23:16		0P	R	-	**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				-	**Reserved**
15: 8				-	**Reserved**
7: 6				-	**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
31:25				-	**Reserved**
24		0P	RW	-	**Enable SCI from PWRBTN event**
23:16				-	**Reserved**
15		0P	RW1C	-	**Wake-up event status**
14: 9				-	**Reserved**
8		0P	RW1C	-	**PWRBTN event status**
7: 0				-	**Reserved**

ACPI_C1_C0: ACPI Register C1_C0 **Offset: 1A4h**

Bit	Name	Initial	Slave	Host	Description
31:16				-	**Reserved**
15:14				-	**Reserved**
13		0P	RW	-	**Update Sleep Type value**
12:10		0P	RW	-	**Sleep Type bit[2:0]**
9: 8				-	**Reserved**
7: 0				-	**Reserved**

ACPI_B3_B0: ACPI Register B3_B0 **Offset: 1A8h**

Bit	Name	Initial	Slave	Host	Description
31:16				-	**Reserved**
15:10				-	**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				-	**Reserved**
15:10				-	**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		X	R	-	**MailBox Data Register 3 bit[7:0]**
23:16		X	R	-	**MailBox Data Register 2 bit[7:0]**
15: 8		X	R	-	**MailBox Data Register 1 bit[7:0]**
7: 0		X	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		-	**Reserved**
7: 0		X	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		-	**Reserved**
7: 0		X	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		-	**Reserved**
7: 0		X	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		X	R	-	**MailBox Data Register 7 bit[7:0]**
23:16		X	R	-	**MailBox Data Register 6 bit[7:0]**
15: 8		X	R	-	**MailBox Data Register 5 bit[7:0]**
7: 0		X	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		-	**Reserved**
7: 0		X	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		-	**Reserved**
7: 0		X	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		-	**Reserved**
7: 0		X	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		X	R	-	**MailBox Data Register B bit[7:0]**
23:16		X	R	-	**MailBox Data Register A bit[7:0]**
15: 8		X	R	-	**MailBox Data Register 9 bit[7:0]**
7: 0		X	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		-	**Reserved**
7: 0		X	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		-	**Reserved**
7: 0		X	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		-	**Reserved**
7: 0		X	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		X	R	-	**MailBox Data Register F bit[7:0]**
23:16		X	R	-	**MailBox Data Register E bit[7:0]**
15: 8		X	R	-	**MailBox Data Register D bit[7:0]**
7: 0		X	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		-	**Reserved**
7: 0		X	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		-	**Reserved**
7: 0		X	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		-	**Reserved**
7: 0		X	RW	RW	**MailBox Data Register F bit[7:0]**

MBXSTS_0: MailBox Status Register 0 **Offset: 240h**

Bit	Name	Initial	Slave	Host	Description
31		0	R	-	**Interrupt status from the BMC to the Host**
30:26		0		-	**Reserved**
25		0	R	-	**Mask interrupt to the Host from the status bit, MBXHCR[7]**
24		0	R	-	**Generate interrupt to the BMC; set the status bit, MBXBCR[7]**
23		0		-	**Interrupt status from the Host to the BMC**
22:18		0		-	**Reserved**
17		0		-	**Mask interrupt to the BMC from the status bit, MBXBCR[7]**
16		0		-	**Generate interrupt to the Host; set the status bit, MBXHCR[7]**
15: 8		0	R	-	**MailBox Status Register bit[15: 8]**
7: 0		0	RW1C	RW1C	**MailBox Status Register bit[7: 0]**

MBXSTS_1: MailBox Status Register 1 **Offset: 244h**

Bit	Name	Initial	Slave	Host	Description
31: 8		0		-	**Reserved**
7: 0		0	RW1C	RW1C	**MailBox Status Register bit[15: 8]**

MBXBCR: MailBox BMC Control Register **Offset: 248h**

Bit	Name	Initial	Slave	Host	Description
31: 8		0		-	**Reserved**
7		0		RW1C	**Interrupt status from the Host to the BMC**
6: 2		0			**Reserved**
1		0		RW	**Mask interrupt to the BMC from the status bit, MBXBCR[7]**
0		0		RW	**Generate interrupt to the Host; set the status bit, MBXHCR[7]**

MBXHCR: MailBox Host Control Register **Offset: 24Ch**

Bit	Name	Initial	Slave	Host	Description
31: 8		0		-	**Reserved**
7		0	RW1C		**Interrupt status from the BMC to the Host**
6: 2		0			**Reserved**
1		0	RW		**Mask interrupt to the Host from the status bit, MBXHCR[7]**
0		0	RW		**Generate interrupt to the BMC; set the status bit, MBXBCR[7]**

MBXBIE_0: MailBox BMC Interrupt Enable Register 0 **Offset: 250h**

Bit	Name	Initial	Slave	Host	Description
31:24		0		RW	**MailBox Host Interrupt Enable bit[15:8]**
23:16		0		RW	**MailBox Host Interrupt Enable bit[7:0]**
15: 8		0	RW	-	**MailBox BMC Interrupt Enable bit[15:8]**
7: 0		0	RW	-	**MailBox BMC Interrupt Enable bit[7:0]**

MBXBIE_1: MailBox BMC Interrupt Enable Register 1 **Offset: 254h**

Bit	Name	Initial	Slave	Host	Description
31: 8		0		-	**Reserved**
7: 0		0	RW	-	**MailBox BMC Interrupt Enable bit[15:8]**

MBXHIE_0: MailBox Host Interrupt Enable Register 0 **Offset: 258h**

Bit	Name	Initial	Slave	Host	Description
7: 0		0		RW	**MailBox Host Interrupt Enable bit[7:0]**

MBXHIE_1: MailBox Host Interrupt Enable Register 1 **Offset: 25Ch**

Bit	Name	Initial	Slave	Host	Description
31: 8		0		-	**Reserved**
7: 0		0		RW	**MailBox Host Interrupt Enable bit[15:8]**

45 LPC+ Controller

45.1 Overview

LPCPlus is a modified version of LPC Bus, and used to connect companion chip.
It supports 1/2/4 bytes access, and reduces address cycles if possible.

LPCP00: Engine control register.
LPCP04: Calibration data.
LPCP08: Byte area control for the first companion chip.
LPCP0C: Byte area control for the second companion chip.
LPCP10: Byte area control for the third companion chip.
LPCP14: Byte area control for the fourth companion chip.
LPCP18: Extend wait cycle and extend address for the first companion chip.
LPCP1C: Extend wait cycle and extend address for the second companion chip.
LPCP20: Extend wait cycle and extend address for the third companion chip.
LPCP24: Extend wait cycle and extend address for the fourth companion chip.
LPCP28: Extend address of UART for the first companion chip.
LPCP2C: Extend address of UART for the second companion chip.
LPCP30: Extend address of UART for the third companion chip.
LPCP34: Extend address of UART for the fourth companion chip.

45.2 Feature

- Support bus timing calibration.
- Reduce address cycles if part of address are repeating.
- Reduce unnecessary data cycles.
- Support max 4 companion chips.

45.3 Registers : Base Address = 0x1E6E_C000

Offset: 00h	LPCP00: Extend wait cycle and extend address	Init = 0x80000000
Bit	**R/W**	**Description**
31	R	**Engine Idle**
30:20	R	**Reserved**
19:18	RW	**Bus timing control for the fourth companion chip** 11: Delay one and half cycles. 10: Delay one cycle. 01: Delay half cycle. 00: Normal timing.
17:16	RW	**Bus timing control for the third companion chip** 11: Delay one and half cycles. 10: Delay one cycle. 01: Delay half cycle. 00: Normal timing.
15:14	RW	**Bus timing control for the second companion chip** 11: Delay one and half cycles. 10: Delay one cycle. 01: Delay half cycle. 00: Normal timing.

to next page

from previous page

13:12	RW	**Bus timing control for the first companion chip** 11: Delay one and half cycles. 10: Delay one cycle. 01: Delay half cycle. 00: Normal timing.
11	RW	**Enable calibration for the fourth companion chip**
10	RW	**Enable calibration for the third companion chip**
9	RW	**Enable calibration for the second companion chip**
8	RW	**Enable calibration for the first companion chip**
7: 1	R	**Reserved**
0	RW	**Engine enable**

Offset: 04h **LPCP04: Calibration data** **Init = 0x00000000**

Bit	R/W	Description
31: 0	R	**Data received after enable calibration.** A successful calibration data is 0x5a5a5a5a.

Offset: 08h **LPCP08: Byte area control for the first companion chip** **Init = 0x00002000**

Bit	R/W	Description
31	RW	**Enable byte area.**
30:16	R	**Reserved**
15: 0	RW	**Byte area address** Address space less than this field will be forced into 1 byte access to reduce data cycle.

Offset: 0Ch **LPCP0C: Byte area control for the second companion chip** **Init = 0x00002000**

Bit	R/W	Description
31	RW	**Enable byte area.**
30:16	R	**Reserved**
15: 0	RW	**Byte area address** Address space less than this field will be forced into 1 byte access to reduce data cycle.

Offset: 10h **LPCP10: Byte area control for the third companion chip** **Init = 0x00002000**

Bit	R/W	Description
31	RW	**Enable byte area.**
30:16	R	**Reserved**
15: 0	RW	**Byte area address** Address space less than this field will be forced into 1 byte access to reduce data cycle.

Offset: 14h **LPCP14: Byte area control for the fourth companion chip** **Init = 0x00002000**

Bit	R/W	Description
31	RW	**Enable byte area.**
30:16	R	**Reserved**
15: 0	RW	**Byte area address** Address space less than this field will be forced into 1 byte access to reduce data cycle.

LPCP18: Extend wait cycle and extend address for the first companion chip

Offset: 18h **Init = 0x06005000**

Bit	Attr.	Description
31:24	RW	**Extend wait cycles.**
23:16	R	**Reserved**
15: 0	RW	**Extend wait address** Read address space larger than this field will wait for more cycles.

LPCP1C: Extend wait cycle and extend address for the second companion chip

Offset: 1Ch **Init = 0x06005000**

Bit	Attr.	Description
31:24	RW	**Extend wait cycles.**
23:16	R	**Reserved**
15: 0	RW	**Extend wait address** Read address space larger than this field will wait for more cycles.

LPCP20: Extend wait cycle and extend address for the third companion chip

Offset: 20h **Init = 0x06005000**

Bit	Attr.	Description
31:24	RW	**Extend wait cycles.**
23:16	R	**Reserved**
15: 0	RW	**Extend wait address** Read address space larger than this field will wait for more cycles.

LPCP24: Extend wait cycle and extend address for the fourth companion chip

Offset: 24h **Init = 0x06005000**

Bit	Attr.	Description
31:24	RW	**Extend wait cycles.**
23:16	R	**Reserved**
15: 0	RW	**Extend wait address** Read address space larger than this field will wait for more cycles.

Offset: 28h LPCP28: Extend address of UART for the first companion chip Init = 0x00001000

Bit	R/W	Description
31:16	R	**Reserved**
15: 0	RW	**Extend wait address of UART** Read address space less than this field will wait for 2 more cycles.

Offset: 2Ch LPCP2C: Extend address of UART for the second companion chip Init = 0x00001000

Bit	R/W	Description
31:16	R	**Reserved**
15: 0	RW	**Extend wait address of UART** Read address space less than this field will wait for 2 more cycles.

Offset: 30h	LPCP30: Extend address of UART for the third companion chip	Init = 0x00001000
Bit	**R/W**	**Description**
31:16	R	**Reserved**
15: 0	RW	**Extend wait address of UART** Read address space less than this field will wait for 2 more cycles.

Offset: 34h	LPCP34: Extend address of UART for the fourth companion chip	Init = 0x00001000
Bit	**R/W**	**Description**
31:16	R	**Reserved**
15: 0	RW	**Extend wait address of UART** Read address space less than this field will wait for 2 more cycles.

45.4 Operation

45.4.1 Calibration

Use the first companion chip as example.

1. Set LPCP00[0] to enable engine.
2. Set proper bus timing control to LPCP00[13:12].
3. Set LPCP00[8] to enable calibration.
4. Waiting LPCP00[8] is cleared.
   - If LPCP04 is 0x5a5a5a5a, calibration is successful.
   - If LPCP04 is not 0x5a5a5a5a, change bus timing control to LPCP00[13:12] and try again.

46 SuperIO Controller (SIO)

46.1 Overview

AST2500 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; reserved in eSPI mode)
6. SUART4 (logical device C; reserved in eSPI mode)
7. iLPC2AHB (logical device D)
8. Mailbox (logical device E)

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_38: Super IO Register 7_38
SIOR7_70: Super IO Register 7_70
SIOR7_71: Super IO Register 7_71

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

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	W	**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	R	**Reserved**
3:0	RW	**Logical device number**

Offset: 20h	**SIORx_20: Super IO Register x_20**	**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: 21h	**SIORx_21: Super IO Register x_21**	**Init = 0P**
Bit	**R/W**	**Description**
7:0	RW	**SIO to BMC scratch register 1 bit[7:0]**

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 = 0
Bit	**R/W**	**Description**
7:0	R	**BMC to SIO scratch register 0 bit[7:0]**

Offset: 29h	SIORx_29: Super IO Register x_29	Init = 0
Bit	**R/W**	**Description**
7:0	R	**BMC to SIO scratch register 1 bit[7:0]**

Offset: 2Ah	SIORx_2A: Super IO Register x_2A	Init = 0
Bit	**R/W**	**Description**
7:0	R	**BMC to SIO scratch register 2 bit[7:0]**

Offset: 2Bh	SIORx_2B: Super IO Register x_2B	Init = 0
Bit	**R/W**	**Description**
7:0	R	**BMC to SIO scratch register 3 bit[7:0]**

Offset: 2Ch	SIORx_2C: Super IO Register x_2C	Init = 0
Bit	**R/W**	**Description**
7:0	R	**BMC to SIO scratch register 4 bit[7:0]**

Offset: 2Dh	SIORx_2D: Super IO Register x_2D	Init = 0
Bit	**R/W**	**Description**
7:0	R	**BMC to SIO scratch register 5 bit[7:0]**

Offset: 2Eh	SIORx_2E: Super IO Register x_2E	Init = 0
Bit	**R/W**	**Description**
7:0	R	**BMC to SIO scratch register 6 bit[7:0]**

Offset: 2Fh	SIORx_2F: Super IO Register x_2F	Init = 0
Bit	**R/W**	**Description**
7:0	R	**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	R	**Reserved**
3:0	RW	**LPC: Select ID bit[3:0] of SerIRQ for SUART1** **eSPI: fixed 4h**

Offset: 71h	SIOR2_71: Super IO Register 2_71	Init = 02
Bit	**R/W**	**Description**
7:2	R	**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**
7:2	R	**Reserved**

to next page

from previous page

1	RW	**SUART1 clock source** 0: 24MHz / 13 1: 24MHz / 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	R	**Reserved**
3:0	RW	**LPC: Select ID bit[3:0] of SerIRQ for SUART2** **eSPI: fixed 3h**

Offset: 71h **SIOR3_71: Super IO Register 3_71** **Init = 02**

Bit	R/W	Description
7:2	R	**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
7:2	R	**Reserved**
1	RW	**SUART2 clock source** 0: 24MHz / 13 1: 24MHz / 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 = 8
Bit	**R/W**	**Description**
7:0	RW	**SWC base address bit[15:8]**

Offset: 61h	SIOR4_61: Super IO Register 4_61	Init = E6
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 = 8
Bit	**R/W**	**Description**
7:0	RW	**PM1b_EVT base address bit[15:8]**

Offset: 63h	SIOR4_63: Super IO Register 4_63	Init = E0
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 = 8
Bit	**R/W**	**Description**
7:0	RW	**PM1b_CNT base address bit[15:8]**

Offset: 65h	SIOR4_65: Super IO Register 4_65	Init = E4
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 = 8
Bit	**R/W**	**Description**
7:0	RW	**GPE1_BLK base address bit[15:8]**

Offset: 67h	SIOR4_67: Super IO Register 4_67	Init = E8
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 = 09
Bit	**R/W**	**Description**
7:4	R	**Reserved**
3:0	RW	**LPC: Select ID bit[3:0] of SerIRQ for SWC** **eSPI: fixed Ch**

Offset: 71h	SIOR4_71: Super IO Register 4_71	Init = 01
Bit	**R/W**	**Description**
7:2	R	**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**
6:1	RW	**Reserved**
0	RW	**Enable SIO GPIO**

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: 70h	SIOR7_70: Super IO Register 7_70	Init = 0B
Bit	**R/W**	**Description**
7:4	R	**Reserved**
3:0	RW	**LPC: Select ID bit[3:0] of SerIRQ for SIO GPIO** **eSPI: fixed Ah**

Offset: 71h	SIOR7_71: Super IO Register 7_71	Init = 01
Bit	**R/W**	**Description**
7:2	R	**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: 30h	SIORB_30: Super IO Register B_30	Init = 0
Bit	**R/W**	**Description**
7:1	RW	**Reserved**

to next page

from previous page

0	RW	**Enable SUART3** **Reserved in eSPI mode**

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	R	**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	R	**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**
7:2	R	**Reserved**
1	RW	**SUART3 clock source** 0: 24MHz / 13 1: 24MHz / 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** **Reserved in eSPI mode**

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	R	**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	R	**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**
7:2	R	**Reserved**
1	RW	**SUART4 clock source** 0: 24MHz / 13 1: 24MHz / 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	R	**Reserved**
3:0	RW	**LPC: Select ID bit[3:0] of SerIRQ for BMC to SIO scratch** **eSPI: fixed Bh**

Offset: 71h	SIORD_71: Super IO Register D_71	Init = 01
Bit	**R/W**	**Description**
7:2	R	**Reserved**

to next page

from previous page

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 = 0**

Bit	R/W	Description
7	R	**SIO iLPC2AHB address alignment valid**
6	R	**Disable SIO iLPC2AHB**
5:2	RW	**Reserved**

to next page

from previous page

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 = 08**
Bit	**R/W**	**Description**
7:0	RW	**Mailbox base address bit[15:8]**

Offset: 61h	**SIORE_61: Super IO Register E_61**	**Init = C0**
Bit	**R/W**	**Description**
7:5	RW	**Mailbox base address bit[7:5]**
4:0	RW	**Reserved**

Offset: 70h	**SIORE_70: Super IO Register E_70**	**Init = 09**
Bit	**R/W**	**Description**
7:4	R	**Reserved**
3:0	RW	**LPC: Select ID bit[3:0] of SerIRQ for Mailbox** **eSPI: fixed Dh**

Offset: 71h	**SIORE_71: Super IO Register E_71**	**Init = 01**
Bit	**R/W**	**Description**
7:2	R	**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

47 System Wake-Up Control (SWC)

47.1 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	R	**PWBTO raw status**
6	R	**Last_ONCTL status**
5	RW1C	**Was_pfail status**
4	R	**Crowbar status**
3	R	**PWRBTN Override status**
2	R	**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	R	**RI rising event status**
6	R	**RI falling event status**
5	R	**S5n status after debounce**
4	R	**S5n rising event status**
3	R	**S5n falling event status**
2	R	**S3n status after debounce**
1	R	**S3n rising event status**
0	R	**S3n falling event status**

Offset: 03h	**SWCR_03: SWC Miscellaneous Event Status 2**	**Init = 0000_1001P**
Bit	**R/W**	**Description**
7	R	**Setting sleep state event status**
6	R	**PWRGD status after debounce**
5	R	**PWRGD rising event status**
4	R	**PWRGD falling event status**

to next page

from previous page

3	R	**PWRBTN status after debounce**
2	R	**PWRBTN rising event status**
1	R	**PWRBTN falling event status**
0	R	**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	R	**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	R	**Enable BMC interrupt as RI rising event**
6	R	**Enable BMC interrupt as RI falling event**
5	R	**Reserved**
4	R	**Enable BMC interrupt as S5n rising event**
3	R	**Enable BMC interrupt as S5n falling event**
2	R	**Reserved**
1	R	**Enable BMC interrupt as S3n rising event**
0	R	**Enable BMC interrupt as S3n falling event**

Offset: 07h **SWCR_07: SWC BMC Interrupt Enable 1** **Init = 0P**

Bit	R/W	Description
7	R	**Enable BMC interrupt as setting sleep state event**
6	R	**Reserved**
5	R	**Enable BMC interrupt as PWRGD rising event**
4	R	**Enable BMC interrupt as PWRGD falling event**
3	R	**Reserved**
2	R	**Enable BMC interrupt as PWRBTN rising event**
1	R	**Enable BMC interrupt as PWRBTN falling event**
0	R	**Reserved**

Offset: 08h	SWCR_08: SWC Control 0	Init = 0P

Bit	R/W	Description
7	RW	Enable software mode of S3n and S5n
6	R	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	R	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
7:2	R	Reserved
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	R	**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	R	**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	R	**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	R	**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	R	**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	R	**Reserved**
2:0	RW	**Sleep type encoding of S5**

Offset: 16h	SWCR_16: SWC Sleep Control	Init = 0P
Bit	**R/W**	**Description**
7:6	R	**slp_now_st[1:0]**
5	R	**Reserved**
4	RW	**Lock PWRBTN**
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	R	**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	R	**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	R	**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	R	**Reserved**

Offset: 1Eh	SWCR_1E: SWC Control A	Init = 0P
Bit	**R/W**	**Description**
7:0	R	**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	R	**Reserved**

Offset: 01h	ACPI_E1: PM1b_STS_HIGH	Init = 0P
Bit	**R/W**	**Description**
7	RW1C	**Wake-up event status**
6:1	R	**Reserved**
0	RW1C	**PWRBTN event status**

Offset: 02h	ACPI_E2: PM1b_EN_LOW	Init = 0P
Bit	**R/W**	**Description**
7:0	R	**Reserved**

Offset: 03h	ACPI_E3: PM1b_EN_HIGH	Init = 0P
Bit	**R/W**	**Description**
7:1	R	**Reserved**
0	RW	**Enable SCI from PWRBTN event**

Offset: 00h	ACPI_C0: PM1b_CNT_LOW	Init = 0P
Bit	**R/W**	**Description**
7:0	R	**Reserved**

Offset: 01h	ACPI_C1: PM1b_CNT_HIGH	Init = 0P
Bit	**R/W**	**Description**
7:6	R	**Reserved**
5	RW	**Update Sleep Type value**
4:2	RW	**Sleep Type bit[2:0]**
1:0	R	**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	R	**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	R	**Reserved**

Offset: 03h	ACPI_B3: GPE1_STS_3	Init = 0P
Bit	**R/W**	**Description**
7:0	R	**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	R	**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	R	**Reserved**

Offset: 07h	ACPI_B7: GPE1_EN_3	Init = 0P
Bit	**R/W**	**Description**
7:0	R	**Reserved**

48 MailBox Controller (MBOX)

48.1 Overview

There are 16 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

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
MBXSTS_0: MailBox Status Register 0
MBXSTS_1: MailBox Status Register 1
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
MBXHIE_0: MailBox Host Interrupt Enable Register 0
MBXHIE_1: MailBox Host Interrupt Enable Register 1

Attribute Definition:

Attribute	: Description
R	: Readable
W	: Writable
W1C	: Write ’1’ to clear value to 0
X	: Unknown value

MBXDAT_0: MailBox Data Register 0 **Offset: 00**

Bit	Name	Initial	Slave	Host	Description
7:0		X	RW	RW	**MailBox Data Register 0 bit[7:0]**

MBXDAT_1: MailBox Data Register 1 **Offset: 01**

Bit	Name	Initial	Slave	Host	Description
7:0		X	RW	RW	**MailBox Data Register 1 bit[7:0]**

MBXDAT_2: MailBox Data Register 2 **Offset: 02**

Bit	Name	Initial	Slave	Host	Description
7:0		X	RW	RW	**MailBox Data Register 2 bit[7:0]**

MBXDAT_3: MailBox Data Register 3 **Offset: 03**

Bit	Name	Initial	Slave	Host	Description
7:0		X	RW	RW	**MailBox Data Register 3 bit[7:0]**

MBXDAT_4: MailBox Data Register 4 **Offset: 04**

Bit	Name	Initial	Slave	Host	Description
7:0		X	RW	RW	**MailBox Data Register 4 bit[7:0]**

MBXDAT_5: MailBox Data Register 5 **Offset: 05**

Bit	Name	Initial	Slave	Host	Description
7:0		X	RW	RW	**MailBox Data Register 5 bit[7:0]**

MBXDAT_6: MailBox Data Register 6 **Offset: 06**

Bit	Name	Initial	Slave	Host	Description
7:0		X	RW	RW	**MailBox Data Register 6 bit[7:0]**

MBXDAT_7: MailBox Data Register 7 **Offset: 07**

Bit	Name	Initial	Slave	Host	Description
7:0		X	RW	RW	**MailBox Data Register 7 bit[7:0]**

MBXDAT_8: MailBox Data Register 8 **Offset: 08**

Bit	Name	Initial	Slave	Host	Description
7:0		X	RW	RW	**MailBox Data Register 8 bit[7:0]**

MBXDAT_9: MailBox Data Register 9 **Offset: 09**

Bit	Name	Initial	Slave	Host	Description
7:0		X	RW	RW	**MailBox Data Register 9 bit[7:0]**

MBXDAT_A: MailBox Data Register A **Offset: 0A**

Bit	Name	Initial	Slave	Host	Description
7:0		X	RW	RW	**MailBox Data Register A bit[7:0]**

MBXDAT_B: MailBox Data Register B **Offset: 0B**

Bit	Name	Initial	Slave	Host	Description
7:0		X	RW	RW	**MailBox Data Register B bit[7:0]**

MBXDAT_C: MailBox Data Register C **Offset: 0C**

Bit	Name	Initial	Slave	Host	Description
7:0		X	RW	RW	**MailBox Data Register C bit[7:0]**

MBXDAT_D: MailBox Data Register D **Offset: 0D**

Bit	Name	Initial	Slave	Host	Description
7:0		X	RW	RW	**MailBox Data Register D bit[7:0]**

MBXDAT_E: MailBox Data Register E **Offset: 0E**

Bit	Name	Initial	Slave	Host	Description
7:0		X	RW	RW	**MailBox Data Register E bit[7:0]**

MBXDAT_F: MailBox Data Register F **Offset: 0F**

Bit	Name	Initial	Slave	Host	Description
7:0		X	RW	RW	**MailBox Data Register F bit[7:0]**

MBXSTS_0: MailBox Status Register 0 **Offset: 10**

Bit	Name	Initial	Slave	Host	Description
7:0		00	RW1C	RW1C	**MailBox Status Register 0 bit[7:0]**

MBXSTS_1: MailBox Status Register 1 **Offset: 11**

Bit	Name	Initial	Slave	Host	Description
7:0		00	RW1C	RW1C	**MailBox Status Register 1 bit[15:8]**

MBXBCR: MailBox BMC Control Register **Offset: 12**

Bit	Name	Initial	Slave	Host	Description
7		0		RW1C	**Interrupt status from the Host to the BMC**
6:2		0			**Reserved**
1		0		RW	**Mask interrupt to the BMC from the status bit, MBXBCR[7]**
0		0		RW	**Generate interrupt to the Host; set the status bit, MBXHCR[7]**

MBXHCR: MailBox Host Control Register **Offset: 13**

Bit	Name	Initial	Slave	Host	Description
7		0	RW1C		**Interrupt status from the BMC to the Host**
6:2		0			**Reserved**
1		0	RW		**Mask interrupt to the Host from the status bit, MBXHCR[7]**

to next page

from previous page

0		0	RW		**Generate interrupt to the BMC; set the status bit, MBXBCR[7]**

MBXBIE_0: MailBox BMC Interrupt Enable Register 0 **Offset: 14**

Bit	Name	Initial	Slave	Host	Description
7:0		0		RW	**MailBox BMC Interrupt Enable bit[7:0]**

MBXBIE_1: MailBox BMC Interrupt Enable Register 1 **Offset: 15**

Bit	Name	Initial	Slave	Host	Description
7:0		0		RW	**MailBox BMC Interrupt Enable bit[15:8]**

MBXHIE_0: MailBox Host Interrupt Enable Register 0 **Offset: 16**

Bit	Name	Initial	Slave	Host	Description
7:0		0	RW		**MailBox Host Interrupt Enable bit[7:0]**

MBXHIE_1: MailBox Host Interrupt Enable Register 1 **Offset: 17**

Bit	Name	Initial	Slave	Host	Description
7:0		0	RW		**MailBox Host Interrupt Enable bit[15:8]**

49 I2C/SMBus Controller

49.1 Overview

I2C/SMBus Controller implements one set of global registers and 14 sets of device registers to program the various functions supported by AST2500 . Each register has its own specific offset value to derive its physical address location.

Base address of Global Register = 0x1E78_A000
Physical register address = (Base address) + Offset

I2CG00: Device Interrupt Status Register
I2CG08: Device Owner Assignment Register
I2CG0C: Global Control Register
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

49.2 Features

49.2.1 I2C Master - all 14 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, 0.5Kbps - 4Mbps if APB clock = 50MHz
- 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 (<= 16 Byte) of Pool Buffer Mode for improving performance
- Programmable Size (<= 4095 Byte) of DMA buffer for large amount of data transfer
- Support bus lock recovery function

49.2.2 I2C Slave - all 14 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 dual slave address capability

49.2.3 SMBus - all 14 buses

- Compatible with SBS SMBus Specification Version 2.0
- Involved all features of I2C
- Support transmission speed from 0.5Kbps - 8Mbps if APB clock = 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

49.2.4 General

- Support totally 14 I2C/SMBus devices.
- All devices can be configured to Alertable SMBus device.
- Support 3 transfer modes:
  - **Byte Buffer**: 1 byte dedicated register
  - **Pool Buffer**: 16 bytes of internal SRAM for each device
  - **DMA Buffer**: Maximum 4095 bytes to or from SDRAM memory
- Schmitt type of input data buffer and input clock buffers
- Anti-glitch data input filter
- Need external pull-up resistors

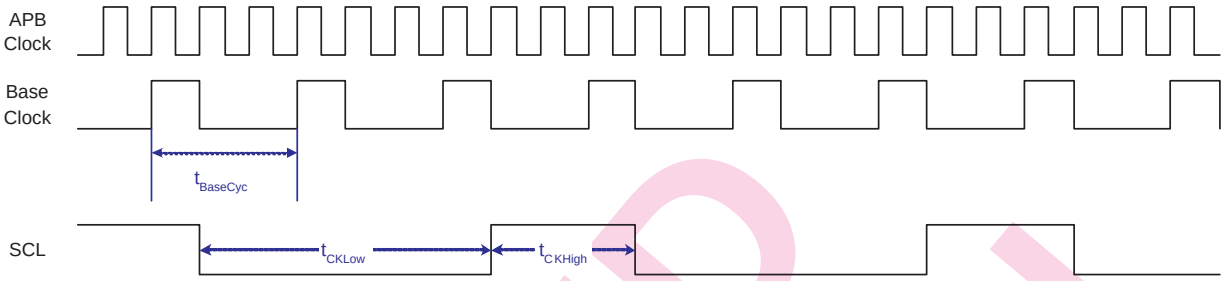
49.3 Migration from AST2300 to AST2400

1. Add supporting from 9 devices to 14 devices
2. Add pool buffer size from 256 bytes to 2048 bytes
3. Add supporting High speed master mode at I2CD00 bit[6]
4. Add buffer page selection at I2CD00 bit[22:20]
5. Modify functionality for direct drive 1T at I2CD00 bit[8:7]
6. Remove the Alert pins mux with I2C7 and I2C8
7. Remove LSB first mode at I2CD00 bit[9]
8. Remove the interrupt mask control I2CD0C bit[11:8]

49.4 Migration from AST2400 to AST2500

1. Add FIFO control at I2CG0C[0], it must be enabled before using Buffer poll or DMA mode.
2. Add support DMA mode (to/from DRAM).
3. Reduce poll buffer size of each device to 16 bytes.
4. Add bus lock auto-release feature at I2CD00[17].
5. Add timeout detection of stuck at slave active mode.
6. Add support dual slave address mode, at I2CD10[31] and I2CD18[15:8].
7. Add support dedicated Alert pin function for all I2C buses.
8. Add 1 more bit for tCKHigh and tCKLow timing to increase resolution.
9. Remove support tBUF, tHDSTA, and tACST timing, fixed at value 0xF.
10. Remove support interrupt control of ”Enable Slave Address Received Match interrupt” at I2CD0C[7].
11. Change the I2C timeout counter (I2CD08) bit from 3 to 5.

49.5 Timing Definition

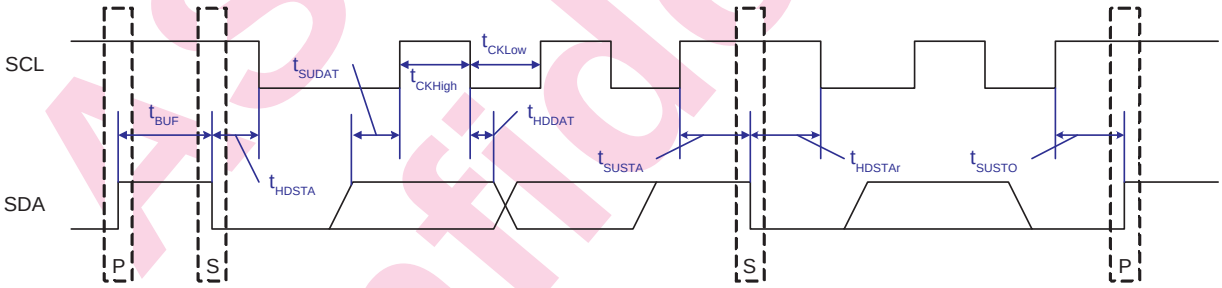


$Freq_{SCL} = Freq_{APB\_Clock} / (t_{BaseCyc} * (t_{CKLow} + t_{CKHigh}))$

$t_{BaseCyc}$ = 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 54: 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} and t_{HDSTAr} and t_{SUSTO} are merged to a common timing setting t_{ACST}, so the setting for t_{ACST} is the max(t_{SUSTA}, t_{HDSTAr}, t_{SUSTO})

Figure 55: AC Timing

49.5.1 Clock Setting Table

- Clock Divisor = APB Clock Frequency / Data Bit Rate

Divisor	Base Clock	tCK High	tCK Low	Divisor	Base Clock	tCK High	tCK Low	Divisor	Base Clock	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

49.6 Registers : Base Address = 0x1E78:A000

49.6.1 Address Definition

Offset	Size(Byte)	Description
03F-000	64	Global Register
07F-040	64	Device 1
0BF-080	64	Device 2
0FF-0C0	64	Device 3
13F-100	64	Device 4
17F-140	64	Device 5
1BF-180	64	Device 6
1FF-1C0	64	Device 7
20F-200	16	Device 1 buffer
21F-210	16	Device 2 buffer
22F-220	16	Device 3 buffer
23F-230	16	Device 4 buffer
24F-240	16	Device 5 buffer
25F-250	16	Device 6 buffer
26F-260	16	Device 7 buffer
27F-270	16	Device 8 buffer
28F-280	16	Device 9 buffer
29F-290	16	Device 10 buffer
2AF-2A0	16	Device 11 buffer
2BF-2B0	16	Device 12 buffer
2CF-2C0	16	Device 13 buffer
2DF-2D0	16	Device 14 buffer
2FF-2E0	32	Reserved
33F-300	64	Device 8
37F-340	64	Device 9
3BF-380	64	Device 10
3FF-3C0	64	Device 11
43F-400	64	Device 12
47F-440	64	Device 13
4BF-480	64	Device 14

49.6.2 Global Register Definition

Offset: 00h	I2CG00: Device Interrupt Status Register	Init = 0
Bit	**R/W**	**Description**
31:14		**Reserved (0)**
13	R	**I2C/SMBus Device #14 Interrupt**
12	R	**I2C/SMBus Device #13 Interrupt**

to next page

from previous page

11	R	**I2C/SMBus Device #12 Interrupt**
10	R	**I2C/SMBus Device #11 Interrupt**
9	R	**I2C/SMBus Device #10 Interrupt**
8	R	**I2C/SMBus Device #9 Interrupt**
7	R	**I2C/SMBus Device #8 Interrupt**
6	R	**I2C/SMBus Device #7 Interrupt**
5	R	**I2C/SMBus Device #6 Interrupt**
4	R	**I2C/SMBus Device #5 Interrupt**
3	R	**I2C/SMBus Device #4 Interrupt**
2	R	**I2C/SMBus Device #3 Interrupt**
1	R	**I2C/SMBus Device #2 Interrupt**
0	R	**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 the 14 devices. There is no need to clear the interrupt status for this register.

Offset: 08h **I2CG08: Device Owner Assignment Register** **Init = 0**

Bit	R/W	Description
31:14		**Reserved (0)**
13	RW	**I2C/SMBus Device #14 Owner**
12	RW	**I2C/SMBus Device #13 Owner**
11	RW	**I2C/SMBus Device #12 Owner**
10	RW	**I2C/SMBus Device #11 Owner**
9	RW	**I2C/SMBus Device #10 Owner**
8	RW	**I2C/SMBus Device #9 Owner**
7	RW	**I2C/SMBus Device #8 Owner**
6	RW	**I2C/SMBus Device #7 Owner**
5	RW	**I2C/SMBus Device #6 Owner**
4	RW	**I2C/SMBus Device #5 Owner**
3	RW	**I2C/SMBus Device #4 Owner**
2	RW	**I2C/SMBus Device #3 Owner**
1	RW	**I2C/SMBus Device #2 Owner**
0	RW	**I2C/SMBus Device #1 Owner** 0 : Assign to ARM 1 : Assign to Coprocessor

Note :
This global register assign the individual bus controlled by which processor. Including interrupt report direction and register write capability.

Offset: 0Ch **I2CG0C: Global Control Register** **Init = 0**

Bit	R/W	Description
31:1		**Reserved (0)**

to next page

from previous page

0	RW	**SRAM Buffer Enable** 0 : Disable 1 : Enable Before accessing or using the SRAM buffer for Buffer Pool or DMA mode, it must be enabled.

49.6.3 Device Register Definition

Offset: 00h		**I2CD00: Function Control Register**	**Init = 0x0**
Bit	**R/W**	**Description**	
31:23		**Reserved (0)**	
22:20		**Reserved (0)** ~~Buffer Page selection~~	
19:18		**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.	
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 under single master function mode only and no clock stretch devices exist. Slave mode can not be enabled when use this mode.	
13:10		**Reserved (0)**	
9	R	~~**Data sequence MSB-First/LSB-First selection**~~ 0 : MSB First ~~1 : LSB First~~	
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 APB clock 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 APB clock 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.	

to next page

from previous page

6	RW	**Enable High Speed master mode** 0 : normal speed mode 1 : high speed mode (3.4Mbps) High speed mode can only use buffer mode for transfer. And only master mode supports speed switching capability.
5	RW	**Enable I2C/SMBus to respond the Default Address (1100_001)** 0 : Disable 1 : Enable
4	RW	**Enable I2C/SMBus to respond the Alert Address (0001_100)** 0 : Disable 1 : Enable
3	RW	**Enable I2C/SMBus to respond the ARP Host Address (0001_000)** 0 : Disable 1 : Enable
2	RW	**Enable I2C/SMBus to respond the General Call Address (0000_0000)** 0 : Disable 1 : Enable
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 : Interrupt Status Register 2. I2CD14 : Command Control Register		

Offset: 04h **I2CD04: Clock and AC Timing Control Register #1** **Init = 0xFFFxxx0x**

Bit	R/W	Description
31:28	R	**Bus free time between Stop and Start timing pattern (tBUF)** ~~x000: 1 Base Clock~~ ~~x001: 2 Base Clock~~ .... 1111: 16 Base Clock
27:24	R	**Hold time of master Start timing pattern (tHDSTA)** ~~x000: 1 Base Clock~~ ~~x001: 2 Base Clock~~ .... 1111: 16 Base Clock
23:20	R	**Setup/Hold time of master Start/Stop timing pattern (tACST)** ~~x000: 1 Base Clock~~ ~~x001: 2 Base Clock~~ .... 1111: 16 Base Clock This register defines the setup time of Start (tSUSTA), the hold time of Repeated Start (tHDSTAr), and the setup time of Stop (tSUSTO); therefore the setting of this register must be the maximum value of all the three timing requirements.

to next page

from previous page

19:16	RW	**Cycles of master SCL clock-high pulse width (tCKHigh)** The Divisor in the table is the Base Clock divisor (I2CD04: Bit [3:0]). The unit for the table is one cycle of Base Clock.

	Disabled Divisor = 0	Disabled Divisor = 1	Disabled Divisor > 1
0000	3	1.5	1
0001	4	2	2
....			
1111	18	16	16

15:12	RW	**Cycles of master SCL clock-low pulse width (tCKLow)** 0000: 1 cycle of Base Clock 0001: 2 cycles of Base Clock .... 1111: 16 cycles of Base Clock
11:10	RW	**Hold time of master/slave data (tHDDAT)** 00 Master 1 Slave 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** This divisor defines the Base Clock for Timeout Counter; the base clock is divided from APB Bus clock. 00: Divided by 16384 (16K) 01: Divided by 65536 (64K) 10: Divided by 262144 (256K) 11: Divided by 1048576 (1024K)
7:4		**Reserved**
3:0	RW	**Base Clock divisor** The divisor defines the frequency of Base Clock which is divided from APB bus clock. 0000: Divided by 1 0001: Divided by 2 0010: Divided by 4 .... 1111: Divided by 32768 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 **I2CD08: Clock and AC Timing Control Register #2** **Init = X**

Bit	R/W	Description
31:5		**Reserved**

to next page

from previous page

4:0	RW	**Cycles of clock or data low timeout (tTimeOut)** 00000: No Timeout Control 00001: 1-2 cycles of Timeout Base Clock .... 11111: 31-32 cycles of Timeout Base Clock Since Timeout Counter is a free run counter, there will be one cycle of uncertainty to generate timeout event.

Offset: 0Ch **I2CD0C: Interrupt Control Register** **Init = 0**

Bit	R/W	Description
31:16		**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		**Reserved** ~~Enable SMBus ARP Host Address Detection interrupt~~ The interrupt mask is removed and please check the interrupt status in the Receive Done interrupt service routine.
10		**Reserved** ~~Enable SMBus Device Alert Response Address Detection interrupt~~ The interrupt mask is removed and please check the interrupt status in the Receive Done interrupt service routine.
9		**Reserved** ~~Enable SMBus Device Default Address Detection interrupt~~ The interrupt mask is removed and please check the interrupt status in the Receive Done interrupt service routine.
8		**Reserved** ~~Enable General Call Address Detection interrupt~~ The interrupt mask is removed and please check the interrupt status in the Receive Done interrupt service routine.
7		**Reserved** ~~Enable Slave Address Received Match interrupt~~ This interrupt will raise in companion with the Receive Done interrupt, and occurred in front of the Received Done interrupt. So it is recommended not to enable this interrupt event. And check the Slave Match interrupt flag in the Receive Done interrupt service routine.
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 Enable**
2	RW	**Enable Receive Done interrupt** Receive Done means : 1. Master : all the expected bytes have been received or the buffer is full. 2. Slave : the buffer is full, or salve controller receives a terminated signaling, and (applied to both master & slave modes) a. Last ACK/NACK returned b. Data has been received The buffer can be Byte Buffer or DMA Buffer.

to next page

from previous page

1	RW	**Enable Transmit with NACK Returned interrupt**
0	RW	**Enable Transmit Done with ACK Returned interrupt** Transmit Done means all the data buffer been transferred.
Note : The definition of this register is : 0 : Disable 1 : Enable		

Offset: 10h **I2CD10: Interrupt Status Register** **Init = 0**

Bit	R/W	Description
31	R	**Slave address match indicator** 0: address 1 1: address 2
30	R	**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.
29:16		**Reserved**
15	RW	**(WC) Slave mode inactive timeout interrupt 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**
11	RW	**(WC) SMBus ARP Host Address Detection interrupt status**
10	RW	**(WC) SMBus Device Alert Response Address Detection interrupt status**
9	RW	**(WC) SMBus Device Default Address Detection interrupt status**
8	RW	**(WC) General Call Address Detection interrupt status**
7	RW	**(WC) Slave Address Received Match interrupt status**
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 allow next data receiving. And at byte buffer mode, this interrupt status may be set concurrently along with bit[11:7].
1	RW	**(WC) Transmit with NACK Returned interrupt status**
0	RW	**(WC) Transmit Done with ACK Returned interrupt status**
Note : 'WC' means this bit is cleared by writing '1'.		

Offset: 14h **I2CD14: Command/Status Register** **Init = 0**

Bit	R/W	Description
31:29		**Reserved (0)**
28	R	**SDA_OE** **(for debugging purpose only)**
27	R	**SDA_O** **(for debugging purpose only)**
26	R	**SCL_OE** **(for debugging purpose only)**

to next page

from previous page

25	R	**SCL_O** **(for debugging purpose only)**
24:23	R	**Transfer Mode Timing Stage** **(for debugging purpose only)** 00: T0 01: T1 10: T2 11: T3
22:19	R	**Transfer Mode State Machine** **(for debugging purpose only)** 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	R	**Sampled SCL Line State**
17	R	**Sampled SDA Line State**
16	R	**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 ∼ 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 (Receiving data and writing into Byte Buffer) 1: Enable (Receiving data to DMA Buffer) 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 (Transmitting data from Byte Buffer) 1: Enable (Transmitting data stored in DMA Buffer) 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 (Receiving data and writing into either byte buffer or DMA buffer) 1: Enable (Receiving data and writing into Pool Buffer) 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 (Transmitting data from either Byte Buffer or Pool Buffer) 1: Enable (Transmitting data from Pool Buffer 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 lastest 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 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 = X**

Bit	R/W	Description
31:16		**Reserved (0)**
15	RW	**Enable Slave Device Address 2** 0: disable 1: enable
14:8	RW	**Slave Device Address 2**
7		**Reserved (0)**
6:0	RW	**Slave Device Address 1**

Offset: 1Ch **I2CD1C: Pool Buffer Control Register** **Init = X**

Bit	R/W	Description
31:29		**Reserved**
28:24	RW	**Actual Received Pool Buffer Size** 0 = 0 bytes 1 = 1 byte 2 = 2 bytes .... 16 = 16 bytes Write command can clear this field to 0.
23:20		**Reserved**

to next page

from previous page

19:16	RW	**Receive Pool Buffer Size** 0 = 1 byte space 1 = 2 bytes space .... 15 = 16 bytes space This value defines the maximum receive buffer size for Slave mode, or receive command data byte count for Master mode.
15:12		**Reserved**
11:8	RW	**Transmit Data Byte Count** 0 = 1 byte 1 = 2 bytes 2 = 3 bytes 3 = 4 bytes .... 15 = 16 bytes
7:0		**Reserved**

Offset: 20h **I2CD20: Transmit/Receive Byte Buffer Register** **Init = X**

Bit	R/W	Description
31:16		**Reserved**
15:8	R	**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 = X**

Bit	R/W	Description
31:30	R	**Reserved (0x0)**
29:2	RW	**DMA Buffer Base Address** DMA Buffer can be allocated at any location of the memory. The starting address must be 4-bytes alignment for I2C read/write operation. And the maximum address of a transfer must not over the 4 KBytes boundary. The address will be updated by hardware to the next memory location when I2C transfer is in progress or done. And the address increment will wrap around at 4 Kbytes boundary. When in Slave transmit mode, if the assigned buffer length longer than the actual transmitted length, the final address read will not equal to the last transmitted adderss. It will be larger due to prefetch operation.
1:0	R	**Reserved (0)**

Offset: 28h **I2CD28: DMA Transfer Length Register** **Init = X**

Bit	R/W	Description
31:13		**Reserved**

to next page

from previous page

11:0	RW	**DMA Transfer Length (Byte)** Define the maximum packet size for transmitting or receiving. The length defined should be within the 4 Kbytes boundary. That is, base_address + length should not over 4 KBytes boundary. 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 smaller than or equal to the actual transmitting/receiving length. Due to the transmit/receive job may be terminated by master at any length.

50 PECI Controller

50.1 Overview

PECI Controller (PECI) supports PECI 1.1, 2.0, and 3.0 protocols.

PECI totally implements 24 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 0x3Ch, 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

50.2 Features

- Directly connected to APB bus
- Intel PECI 3.0/2.0/1.1 compliant
- Support up to 8 CPU and 2 domains per CPU

50.3 Registers : Base Address = 0x1E78:B000

Offset: 00h	PECI00: Control Register	Init = 0x000XX00X
Bit	**R/W**	**Description**
31:20		**Reserved (0)**
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 24MHz oscillator 1: from MCLK
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 24MHz oscillator.
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
4	RW	**Enable PECI** 0: Disable 1: Enable
3:1	RW	**Reserved**

to next page

from previous page

0	RW	**Enable PECI clock** 0: Disable PECI clock 1: Enable PECI clock This register will stop or enable 24MHz clock source for power saving.

Offset: 04h **PECI04: Timing Negotiation Register** **Init = X**

Bit	R/W	Description
31:16		**Reserved (0)**
15:8	RW	**Message timing negotiation** This register will determine the period of message timing negotiation 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** This register will determine the period of address timing negotiation to be issued by PECI Controller. The unit of the programmed value is four times of PECI clock period.

Offset: 08h **PECI08: Command Register** **Init = 0x80000000**

Bit	R/W	Description
31	R	**PECI pin monitoring** This bit can read back the signal status of PECI pin.
30:28		**Reserved (0)**
27:24	R	**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		**Reserved (0)**
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		**Reserved (0)**

to next page

from previous page

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, PECI34, PECI38, and PECI3C. Although this register is 8 bits, the maximum read data length is only 16 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, PECI24, PECI28, and PECI2C.
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		**Reserved (0)**
23:16	R	**Expected read FCS** This register contains the FCS data generated by the internal hardware logic. This is for debugging purpose only.
15:8	R	**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	R	**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		**Reserved (0)**
23:16	R	**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		**Reserved (0)**
7 :0	R	**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		**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

to next page

from previous page

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		**Reserved (0)**
29:16	R	**Timing negotiation result bit [13:0]**
15	R	**Good1FCS**
15:4		**Reserved (0)**
4	RW	**PECI bus time-out interrupt status** 0: No interrupt 1: Interrupt is pending Writing ”1” will clear this status.
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.
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 bit [31:0]**

Offset: 24h **PECI24: Write Data Register #1** **Init = X**

Bit	R/W	Description
31:0	RW	**Write data bit [63:32]**

Offset: 28h	PECI28: Write Data Register #2	Init = X
Bit	**R/W**	**Description**
31:0	RW	**Write data bit [95:64]**

Offset: 2Ch	PECI2C: Write Data Register #3	Init = X
Bit	**R/W**	**Description**
31:0	RW	**Write data bit [127:96]**

Offset: 30h	PECI30: Read Data Register #0	Init = X
Bit	**R/W**	**Description**
31:0	R	**Read data bit [31:0]**

Offset: 34h	PECI34: Read Data Register #1	Init = X
Bit	**R/W**	**Description**
31:0	R	**Read data bit [63:32]**

Offset: 38h	PECI38: Read Data Register #2	Init = X
Bit	**R/W**	**Description**
31:0	R	**Read data bit [95:64]**

Offset: 3Ch	PECI3C: Read Data Register #3	Init = X
Bit	**R/W**	**Description**
31:0	R	**Read data bit [127:96]**

Offset: 40h	PECI40: Write Data Register #4	Init = X
Bit	**R/W**	**Description**
31:0	RW	**Write data bit [159:128]**

Offset: 44h	PECI44: Write Data Register #5	Init = X
Bit	**R/W**	**Description**
31:0	RW	**Write data bit [191:160]**

Offset: 48h	PECI48: Write Data Register #6	Init = X
Bit	**R/W**	**Description**
31:0	RW	**Write data bit [223:192]**

Offset: 4Ch	PECI4C: Write Data Register #7	Init = X
Bit	**R/W**	**Description**
31:0	RW	**Write data bit [225:224]**

Offset: 50h	PECI50: Read Data Register #4	Init = X
Bit	**R/W**	**Description**
31:0	R	**Read data bit [159:128]**

Offset: 54h	PECI54: Read Data Register #5	Init = X
Bit	**R/W**	**Description**
31:0	R	**Read data bit [191:160]**

Offset: 58h	PECI58: Read Data Register #6	Init = X
Bit	**R/W**	**Description**
31:0	R	**Read data bit [223:192]**

Offset: 5Ch	PECI5C: Read Data Register #7	Init = X
Bit	**R/W**	**Description**
31:0	R	**Read data bit [225:224]**

51 PCIe Host Controller

51.1 Overview

Base Address of PCIe Host Controller = 0x1E6E_D000
Physical address of register = (Base address of PCIe Host Controller) + Offset

PEHR04: Class Code Revision Register
PEHR30: Miscellaneous Control 30h Register
PEHR50: Miscellaneous Control 50h Register
PEHR58: Miscellaneous Control 58h Register
PEHR7C: Protection Key Register
PEHRD0: Miscellaneous Status D0h Link Register

51.2 Features

PCIe Host Controller

- PCIe Gen2 x1 Rootport or Bridge Controller (mutually exclusive)
- Support configuration, I/O, 32-bit memory, and message transactions
- Support INTX or MSI (number 32)

51.3 Registers : Base Address = 0x1E6E:D000

Offset: 04h	PEHR04: Class Code Revision Register	Init = 0x06040004
Bit	**R/W**	**Description**
31: 8	RW	**Class Code**
7: 0	RW	**Revision**

Offset: 30h	PEHR30: Miscellaneous Control 30h Register	Init = 0x0
Bit	**R/W**	**Description**
5: 4	RW	**Port type bit [1:0]** 00: bridge 11: rootport others: reserved

Offset: 50h	PEHR50: Miscellaneous Control 50h Register	Init = 0x0
Bit	**R/W**	**Description**
31: 0	RW	**cfg_din**

Offset: 58h	PEHR58: Miscellaneous Control 58h Register	Init = 0x0
Bit	**R/W**	**Description**
13	RW	**cfg_enwr**
12	RW	**cfg_enrd**
11: 0	RW	**cfg_adr**

Offset: 7Ch	PEHR7C: Protection Key Register	Init = 0x0
Bit	**R/W**	**Description**
7: 0	W	**unlock by writing 0xA8 and lock by writing others**
0	R	**unlock status**

Offset: D0h	PEHRD0: Miscellaneous Status D0h Link Register	Init = 0xX
Bit	**R/W**	**Description**
20	R	**link width x1**
17	R	**link speed 5.0G**
16	R	**link speed 2.5G**

Part IV
PCI Express Interface

52 PCI Express Controller

52.1 PCI Express Configuration Registers

Byte Offset	31:24	23:16	15:8	7:0
000h..03Fh	TYPE 1 CONFIGURATION REGISTERS			
040h..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			

52.2 Type 1 Configuration Registers

Byte Offset	31:24	23:16	15:8	7:0
000h	Device ID		Vender ID	
004h	Status		Command	
008h	Class Code			Revision ID
00Ch				Cache Line
010h	Base Address 0			
014h	Base Address 1			
018h		Sub.Bus Number	Sec Bus number	Prim Bus Number
01Ch	Secondary Status		I/O Limit	I/O Base
020h	Memory Limit		Memory Base	
024h	Prefetchable Memory Limit		Prefetchable Memory Base	
028h	Prefetchable Base Upper 32-bit			
030h	I/O Limit Upper 16 Bits		I/O Limit Lower 16 Bits	
034h				Capability PTR
038h	Expansion ROM Base Address			
03Ch	Bridge control		Int.Pin	Int. Line

52.3 Message signaled interrupts

Byte Offset	31:24	23:16	15:8	7:0
050h	Message Control		Next Ptr	Cap ID
054h	Message Address			
058h	Message Upper Address			
05Ch	Message Data			

52.4 Power Management Capability Structure

Byte Offset	31:24	23:16	15:8	7:0
078h	Capability Register		Next ap Ptr	Cap ID
07Ch	Data	PM Control/status Bridge Extensions	Power Management Status & Control	

52.5 PCI Express Capability Structure

Byte Offset	31:24	23:16	15:8	7:0
080h	Capability Register		Next Ptr	Cap ID
084h	Device Capability			
088h	Device Status		Device Control	
08Ch	Link Capability			
090h	Link Status		Link Control	
094h	Slot Capability			
098h	Slot Status		Slot Control	
09Ch	Root Capability		Root Control	
0A0h	Root Status			
0A4h	Device Capability 2			
0A8h	Device Status 2		Device Control 2	
0ACh	Link Capability 2			
0B0h	Link Status 2		Link Control 2	
0B4h	Slot Capability 2			
0B8h	Slot Status 2		Slot Control 2	

52.6 SSID/SSVID Capability Structure

Byte Offset	31:24	23:16	15:8	7:0
0C0h	reserved		Next ap Ptr	Cap ID
0C4h	SSID		SSVID	

53 PCI Bus Controller (P-Bus)

53.1 Overview

AST2500 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 AST2500 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
PCIS80: Enable Subsystem ID Write Register

53.2 Features

- Support 32-bit 33 MHz PCI bus interface with PCI 2.3 specification compliant
- Support PME# control pin

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 AST2500 VGA. The device ID code of AST2500 VGA is the same as AST2000 VGA. This arrangement is to make sure that AST2500 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** AST2500 VGA will not detect any parity errors; therefore, this bit will always return ”0”.
30	R	**Signaled system error** AST2500 VGA will not signal any system errors; therefore, this bit will always return ”0”.
29	R	**Received master abort** AST2500 VGA doesn’t play as a bus master; this register will always return ”0”.
28	R	**Received target abort** AST2500 VGA doesn’t play as a bus master; this register will always return ”0”.
27	R	**Signaled target abort** AST2500 VGA will not issue target abort; this register will always return ”0”.
26:25	R	**DEVSEL timing** AST2500 VGA supports medium timing for DEVSEL signal; this register will always return ”01”.
24	R	**Master data parity error** AST2500 VGA doesn’t play as a bus master; this register will always return ”0”.
23	R	**Fast back-to-back capable** AST2500 VGA doesn’t support fast back-to-back; this register will always return ”0”.
22		**Reserved (0)**
21	R	**66 MHz capable** AST2500 VGA supports 33MHz PCI bus running frequency; this register will always return ”0”.
20	R	**Capabilities list** AST2500 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** AST2500 VGA doesn’t support fast back-to-back; this register will always return ”0”.
8	R	**SERR# enable** AST2500 VGA wont signal any system errors; this bit will always return ”0”.
7		**Reserved (0)**
6	R	**Parity error response enable** AST2500 VGA wont detect any parity errors; this bit will always return ”0”.
5	RW	**VGA palette snoop** AST2500 VGA only provides a read/write bit for this register. But it wont impact any hardware behavior.
4	RW	**Memory write and invalidate enable** Since AST2500 VGA wont support this feature; this register will always return ”0”.
3	R	**Special cycles enable** Since AST2500 VGA doesn’t support PCI special cycles, this register will always return ”0”.

to next page

from previous page

2	R	**Bus master enable** Since AST2500 VGA doesn't support PCI bus master cycles, this register will always return "0".
1	RW	**Memory space access enable** 0: Disable memory space accesses 1: Enable memory space accesses This register will determine whether AST2500 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 AST2500 VGA will response to I/O space accesses or not.

Offset: 08h **PCIS08: Class and Revision ID Register** **Init = 0x0X00_0041**

Bit	R/W	Description
31:8	R	**Class code** When VGA Controller is enabled, AST2500 VGA will always return "0x030000" as the class code for this register to claim that AST2500 VGA is a tandard VGA device. When VGA is disabled by an external strapping resistor, AST2500 VGA will return "0x040000" as the class code of this register to claim that AST2500 VGA is a video device. As a video device, AST2500 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 AST2500 VGA is "41" for A1 and later revision.

Offset: 0Ch **PCIS0C: Miscellaneous Register** **Init = 0x0000_0000**

Bit	R/W	Description
31	R	**BIST Capable** AST2500 VGA doesn't support BIST; this register will always return "0".
30	R	**Start BIST** AST2500 VGA doesn't support BIST; this register will always return "0".
29:28	R	**Start BIST** AST2500 VGA doesn't support BIST; this register will always return "0".
27:24	R	**Completion code** AST2500 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** AST2500 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** AST2500 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** AST2500 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** AST2500 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** AST2500 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. AST2500 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** AST2500 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** AST2500 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** AST2500 VGA supports D0, D1, D2, D3hot and D3cold states. Therefore, this register will always return 0xF.
26	R	**D2 support** AST2500 supports D2 state; this register will always return ”1”.
25	R	**D1 support** AST2500 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 AST2500 VGA requires 375mA from auxiliary current.
21	R	**Device specific initialization** AST2500 VGA doesn’t need any special initializations. This register will always return ”0”.
20		**Reserved (0)**
19	R	**PME Clock** AST2500 VGA doesn’t need to rely on PCI clock to generate PME#. This register will always return ”0”.
18:16	R	**Version** AST2500 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. AST2500 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 AST2500 VGA would normally assert the PME signal independent of the state of the PME enable bit. Writing ”1” to this register will cause AST2500 VGA to stop asserting the PME signal. Writing ”0” to this register has no effect.
14:13	R	**Data scale** AST2500 VGA doesn’t implement Data register; this register always returns ”00b”.
12:9	RW	**Data select** AST2500 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 AST2500 VGA and to set AST2500 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	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 AST2500 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 AST2500 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.

Offset: 80h	PCIS80: Enable Subsystem ID Write Register	Init = 0x0000_0000
Bit	**R/W**	**Description**
31:0	RW	**Password bit [31:0]** Password: AA1A_03AAh

54 VGA Display Controller

54.1 Overview

VGA Display Controller (VGA) is one of the key modules integrated by AST2500 . The system bus interface adopted by VGA is 32-Bit PCI bus interface, which can be operating at 33MHz. VGA can be disabled by an external strapping resistor.

When VGA is enabled, the class code of ”VGA Device” will be claimed by PCI configuration registers.
When VGA is disabled, the class code of ”Video Device” will be claimed for instead.

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. AST2500 also provides memory-mapped I/O addressing mode for the need of advanced operating systems.

54.2 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
- Digital video outputs: 165MHz 24-bit single-edge DVO (3.3V digital signals)

54.3 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]**

54.4 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)**

54.5 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
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
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]**

54.6 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)**
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]**

54.7 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]**

54.8 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]**

54.9 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			

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]

to next page

from previous page

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**

to next page

from previous page

3	RW	**Enable PCI retry for memory write cycles**
2	RW	**Enable PCI retry for memory read cycles**
1	RW	**EnE2MRd**
0	RW	**EnE2MWr**

VGACRA3: Enhanced Color Mode Register

R/W:3B5/3D5 Index:A3 MMIO:Base+A3 **Init = 00h**

Bit	Attr.	Description
7	RW	**Enable DVO interface**
6	RW	**Enable dual-edge DVO interface**
5:4	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'**
5:4	R	**ARM CPU boot code selection** 00: Boot from NOR flash memory 01: Boot from NAND flash memory 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

to next page

from previous page

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**

to next page

from previous page

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]**

to next page

from previous page

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
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**
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**
7:0	RW	**Cursor pattern memory address bit [11:4]** The address must be 16-byte aligned. Therefore address bit [3:0] is always ”0”.

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 [19:12]**

VGACRCA: H/W Cursor Pattern Address Register #3		
R/W:3B5/3D5 Index:CA MMIO:Base+CA		**Init = 00h**
Bit	**Attr.**	**Description**
7:0	RW	**Cursor pattern memory address bit [27:20]**

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
0	RW	**H/W cursor type selection** 0: Select 2-BPP 1: Select 16-BPP (ARGB:1555)

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–D7: Scratch Register #32 ~ #39		
R/W:3B5/3D5 Index:D0–D7 MMIO:Base+D0–D7		**Init = XXh**
Bit	**Attr.**	**Description**
7:0	RW	**Scratch register bit [7:0]**

55 2D Graphics Engine (G2D)

55.1 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. AST2500 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

55.2 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

55.3 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)

to next page

from previous page

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
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		**Reserved (0)**

to next page

from previous page

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
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

to next page

from previous page

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
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	

56 Graphics Hardware Cursor

56.1 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

56.2 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:2018h	SCU18: Interrupt Control and Status Register	Init = 0

Bit	R/W	Description
31:18		**Reserved (0)**
17	RW	**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	RW	**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.
15:2		**Reserved (0)**
1	RW	**Enable VGA scratch register change interrupt** 0 : Disable Interrupt 1 : Enable Interrupt generation
0	RW	**Enable VGA cursor change interrupt** 0 : Disable Interrupt 1 : Enable Interrupt generation

Offset: 1E6E:2050	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]**

to next page

from previous page

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 monochrome cursor type 1 : Select color cursor type
8	R	**Hardware cursor is enabled** 0 : Disabled hardware cursor 1 : Enable hardware cursor
7:0	R	**Reserved**

Offset: 1E6E:2054h **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:2058h **VGA Scratch Register #3** **Init = 0**

Bit	R/W	Description
31:28	R	**Reserved**
27:0	R	**Hardware cursor pattern memory address bit [27:0]**

56.3 Cursor Shape Structure Definition

56.3.1 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

56.3.2 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 = Alpha x Graphics R/G/B + (1-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

57 P-Bus to AHB Bridge (P2A)

57.1 Overview

P-to-AHB Bus Bridge (P2A) is an interface controller bridging two internal buses:

P-Bus: The internal expansion bus supporting bus commands from PCI bus controller
AHB : The internal system bus supporting ARM SOC subsystem

P2A is a one-way bus bridge providing a back door for host CPU to access all the internal IP modules in ARM SOC sub-system. Since P2A is a one-way bridge, ARM CPU cannot issue any PCI bus commands through the help of this bridge. In a normal condition, this back door should be well locked. The two potential usages of this bus bridge are:

1. Updating flash memory through host CPU
2. H/W or S/W debugging through host CPU

P2A only implements two sets of 32-bit registers to provide a protection mechanism and specify the base address of the 64KB address re-mapping window.

57.2 Registers : Base Address = MMIOBASE

Offset: F000h	P2A00: Protection Key Register	Init = 0
Bit	**R/W**	**Description**
31:1		**Reserved (0)**
0	RW	**Protection key** 0: Disable P2A bridge 1: Enable P2A bridge When P2A is disabled, it will ignore all the P-Bus commands. Therefore, there will be no command conversion from P-Bus to AHB. Always keep this protect key in disabled state when there is no need.

Offset: F004h	P2A04: Re-mapping Base Address Register	Init = X
Bit	**R/W**	**Description**
31:16	R	**Re-mapping base address** This register defines the address re-mapping scheme from P-Bus to AHB. Bit [31:16] of AHB address is from the Bit [31:16] of this register, Bit [15:0] is directly from P-Bus command address. AHB Address = (Re-mapping base address[31:16]) + (P-bus address[15:0]) P2A will convert all the commands from P-bus with 64KB address range from **(MMIOBASE + 0x10000)** to **(MMIOBASE + 0x1FFFF)**. Where MMIOBASE is the re-locatable memory-mapped I/O base address defined in PCI configuration space. P2A supports byte, word or double word type of access commands.
15:0		**Reserved (0)**

58 Message Signaled Interrupts (MSI)

58.1 Overview

MSI engine includes 2 function. One is Message Signaled Interrupts specified in PCIe.
The other is virtual INTx also specified in PCIe.

58.2 Features

- 2 exclusive interrupt types: INTx and MSI.
- Total 22 interrupt sources from 5 devices.
- 1 of 22 sources is controlled by APB

58.3 Operation

58.3.1 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])

Interrupt sources from NIC0 Device

INT#	Description
0	NIC0 Interrupt

Interrupt sources from NIC1 Device

INT#	Description
0	NIC1 Interrupt

Interrupt sources from LPC Device

INT#	Description
0	Virtual UART
1	Pass-through UART
2	LPC Channel #2
3	LPC Channel #3
4	LPC Channel #4
5	LPC Channel #5K
6	LPC Channel #5M
7	SUART1
8	SUART2
9	LPC iBT
10	SIO GPIO
11	SIO iLPC2AHB
12	SIO ACPI
13	SIO Mailbox
14	SUART3
15	SUART4

58.3.2 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

58.3.3 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”.

59 PCIe BMC Device

59.1 Overview

AST2500 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 AST2500 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
PCIB80: Enable Subsystem ID Write Register

59.2 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 AST2500 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** AST2500 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** AST2500 BMC device will not signal any system errors; therefore, this bit will always return ”0”.
29	R	**Received master abort** AST2500 BMC device doesn’t play as a bus master; this register will always return ”0”.
28	R	**Received target abort** AST2500 BMC device doesn’t play as a bus master; this register will always return ”0”.
27	R	**Signaled target abort** AST2500 BMC device will not issue target abort; this register will always return ”0”.
26:25	R	**DEVSEL timing** AST2500 BMC device supports medium timing for DEVSEL signal; this register will always return ”01”.
24	R	**Master data parity error** AST2500 BMC device doesn’t play as a bus master; this register will always return ”0”.
23	R	**Fast back-to-back capable** AST2500 BMC device doesn’t support fast back-to-back; this register will always return ”0”.
22		**Reserved (0)**
21	R	**66 MHz capable** AST2500 BMC device supports 33MHz PCI bus running frequency; this register will always return ”0”.
20	R	**Capabilities list** AST2500 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** AST2500 BMC device doesn’t support fast back-to-back; this register will always return ”0”.
8	R	**SERR# enable** AST2500 BMC device wont signal any system errors; this bit will always return ”0”.
7	R	**Reserved (0)**
6	R	**Parity error response enable** AST2500 BMC device wont detect any parity errors; this bit will always return ”0”.
5	R	**Reserved (0)**
4	RW	**Memory write and invalidate enable** Since AST2500 BMC device wont support this feature; this register will always return ”0”.
3	R	**Special cycles enable** Since AST2500 BMC device doesn’t support PCI special cycles, this register will always return ”0”.

to next page

from previous page

2	R	**Bus master enable** Since AST2500 BMC device doesn't support PCI bus master cycles, this register will always return "0".
1	RW	**Memory space access enable** 0: Disable memory space accesses 1: Enable memory space accesses This register will determine whether AST2500 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 AST2500 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 AST2500 BMC device is "00".

Offset: 0Ch **PCIB0C: Miscellaneous Register** **Init = 0x0000_0000**

Bit	R/W	Description
31	R	**BIST Capable** AST2500 BMC device doesn't support BIST; this register will always return "0".
30	R	**Start BIST** AST2500 BMC device doesn't support BIST; this register will always return "0".
29:28	R	**Start BIST** AST2500 BMC device doesn't support BIST; this register will always return "0".
27:24	R	**Completion code** AST2500 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** AST2500 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** AST2500 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** AST2500 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. AST2500 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	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** AST2500 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** AST2500 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** AST2500 BMC device supports D0, D1, D2, D3hot and D3cold states. Therefore, this register will always return 0xF.
26	R	**D2 support** AST2500 supports D2 state; this register will always return ”1”.
25	R	**D1 support** AST2500 BMC device supports D1 state; this register will always return ”1”.

to next page

from previous page

24:22	R	**Auxiliary current requirement** This register will always return ”111b”. It means that AST2500 BMC device requires 375mA from auxiliary current.
21	R	**Device specific initialization** AST2500 BMC device doesn’t need any special initializations. This register will always return ”0”.
20		**Reserved (0)**
19	R	**PME Clock** AST2500 BMC device doesn’t need to rely on PCI clock to generate PME#. This register will always return ”0”.
18:16	R	**Version** AST2500 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. AST2500 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 AST2500 BMC device would normally assert the PME signal independent of the state of the PME enable bit. Writing ”1” to this register will cause AST2500 BMC device to stop asserting the PME signal. Writing ”0” to this register has no effect.
14:13	R	**Data scale** AST2500 BMC device doesn’t implement Data register; this register always returns ”00b”.
12:9	RW	**Data select** AST2500 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)**
1 :0	RW	**Power state** These two bits are used both to determine the current power state of AST2500 BMC device and to set AST2500 BMC device into a new power state. AST2500 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 AST2500 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 AST2500 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.