Welcome to E-XFL.COM

Understanding Embedded - Microprocessors

Embedded microprocessors are specialized computing chips designed to perform specific tasks within an embedded system. Unlike general-purpose microprocessors found in personal computers, embedded microprocessors are tailored for dedicated functions within larger systems, offering optimized performance, efficiency, and reliability. These microprocessors are integral to the operation of countless electronic devices, providing the computational power necessary for controlling processes, handling data, and managing communications.

Applications of Embedded - Microprocessors

Embedded microprocessors are utilized across a broad spectrum of applications, making them indispensable in

Details	
Product Status	Obsolete
Core Processor	ARM® Cortex®-A7, ARM® Cortex®-M4
Number of Cores/Bus Width	1 Core, 32-Bit
Speed	800MHz
Co-Processors/DSP	Multimedia; NEON™ MPE
RAM Controllers	DDR3, DDR3L, LPDDR2, LPDDR3
Graphics Acceleration	No
Display & Interface Controllers	Keypad, LCD, MIPI
Ethernet	10/100/1000Mbps (1)
SATA	-
USB	USB 2.0 + PHY (1), USB 2.0 OTG + PHY (1)
Voltage - I/O	1.8V, 3.3V
Operating Temperature	-20°C ~ 105°C (TJ)
Security Features	A-HAB, ARM TZ, CAAM, CSU, SJC, SNVS
Package / Case	488-TFBGA
Supplier Device Package	488-MAPBGA (12x12)
Purchase URL	https://www.e-xfl.com/product-detail/nxp-semiconductors/mcimx7s3evk08sc

PLL in ANATOP can be set be controlled by CCM via setting some CCM override bits. Also there is control on 32K IPG_CLK_ROOT synchronized version inside clock source control module.

For PLLs, we have channel on every PLL, every PFD and every divider. PLL is the source of PFDs and dividers. For any clock, its source must be left on when it is kept on. Behavior is undefined if this rule is violated.

For a shutdown clock source, if it is declared depended by writing clock source control registers, the controlling logic will turn on the source immediately, while the setting goes into a shadow register. After clock source get ready, the setting will be accepted by source control logic, and copied from shadow register to setting register.

5.2.6.7 Access control

CCM can implement its own access control based on domain, to provide more precise control on shared resource. Access controls are implemented on clock root generation, clock gate control, and clock source control. Access control logic will never impact on read access, but will block unauthentic write access.

Access controls on clock root generation are independent between every clock root. A sticky authentic fail flag will be set if authentic check failed when a domain is trying to write to some register. There are a whitelist and a semaphore inside access control logic. For each clock root, access control logics are default disabled, after power on reset. Software can enable access control anytime after that. But once access logic is enabled, it cannot be disabled until next power on reset.

Table 5-13. Whitelist

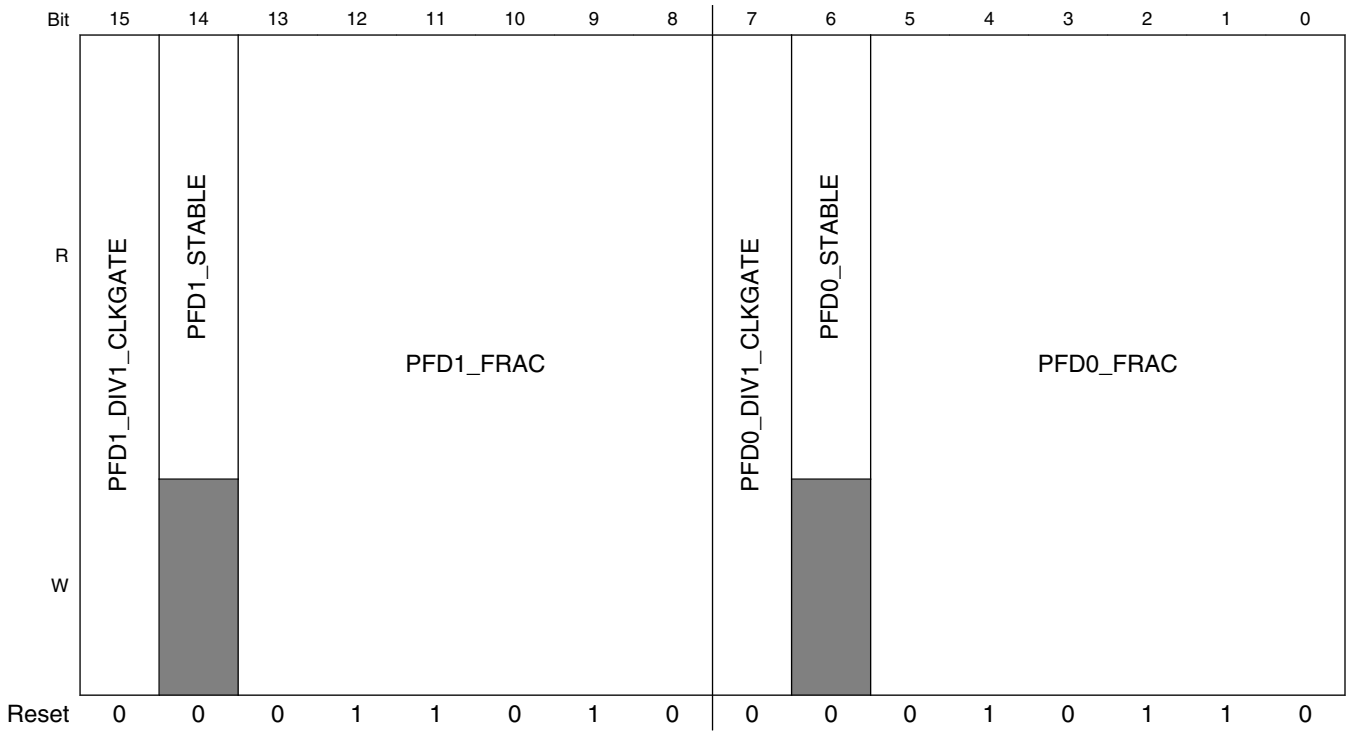
Enabled	Write access will be authenticated before being performed.
Disabled	every access on protected item will be performed.

NOTE

Only domains that are on the whitelist can perform write access to this clock root when access control is enabled.

Table 5-14. Semaphore

Enabled	A domain must obtain the semaphore's ownership before its write access can be authenticated. Only a domain on the whitelist can obtain the ownership and the ownership will last until the domain explicitly releases the ownership. Semaphore obtain will fail if it is already fetched by some other domain.
Disabled	Authentic check will check only on whitelist.



CCM_ANALOG_PFD_480A*n* field descriptions

Field	Description
31 PFD3_DIV1_CLKGATE	IO Clock Gate. If set to 1, the 3rd fractional divider clock (reference ref_pfd3) is off (power savings). 0: ref_pfd3 fractional divider clock is enabled. If the corresponding override bit is set in the 480MHz PLL control register, this field has no effect.
30 PFD3_STABLE	This read-only bitfield is for DIAGNOSTIC PURPOSES ONLY since the fractional divider should become stable quickly enough that this field will never need to be used by either device driver or application code. The value inverts when the new programmed fractional divide value has taken effect. Read this bit, program the new value, and when this bit inverts, the phase divider clock output is stable. Note that the value will not invert when the fractional divider is taken out of or placed into clock-gated state.
29–24 PFD3_FRAC	This field controls the fractional divide value. The resulting frequency shall be 480*18/PFD3_FRAC where PFD3_FRAC is in the range 12-35.
23 PFD2_DIV1_CLKGATE	IO Clock Gate. If set to 1, the IO fractional divider clock (reference ref_pfd2) is off (power savings). 0: ref_pfd2 fractional divider clock is enabled. If the corresponding override bit is set in the 480MHz PLL control register, this field has no effect.
22 PFD2_STABLE	This read-only bitfield is for DIAGNOSTIC PURPOSES ONLY since the fractional divider should become stable quickly enough that this field will never need to be used by either device driver or application code. The value inverts when the new programmed fractional divide value has taken effect. Read this bit, program the new value, and when this bit inverts, the phase divider clock output is stable. Note that the value will not invert when the fractional divider is taken out of or placed into clock-gated state.
21–16 PFD2_FRAC	This field controls the fractional divide value. The resulting frequency shall be 480*18/PFD2_FRAC where PFD2_FRAC is in the range 12-35.
15 PFD1_DIV1_CLKGATE	IO Clock Gate. If set to 1, the IO fractional divider clock (reference ref_pfd1) is off (power savings). 0: ref_pfd1 fractional divider clock is enabled. If the corresponding override bit is set in the 480MHz PLL control register, this field has no effect.
14 PFD1_STABLE	This read-only bitfield is for DIAGNOSTIC PURPOSES ONLY since the fractional divider should become stable quickly enough that this field will never need to be used by either device driver or application code. The value inverts when the new programmed fractional divide value has taken effect. Read this bit,

Table continues on the next page...

6.1.7.9.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	HAC_COUNTER_IV															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	HAC_COUNTER_IV															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

6.1.7.9.4 Fields

Field	Function
31-0 HAC_COUNTER_IV	High Assurance Counter Initial Value This register is used to set the starting count value to the high assurance counter. This register cannot be programmed when HAC_L bit is set.

6.1.7.10 SNVS_HP High Assurance Counter (HPHACR)

6.1.7.10.1 Address

Register	Offset
HPHACR	20h

6.1.7.10.2 Function

The SNVS_HP High Assurance Counter Register contains the value of the high assurance counter. The high assurance counter is a delay introduced before the system security monitor transitions from soft fail to hard fail state if this transition is enabled.

Table 6-35. EIM IOMUX Pin Configuration (continued)

Signal	A/D16 (Muxed, 16-bit data interface)	A+D (Not muxed, 16-bit data interface)
ADDR22	EPDC_BDR0.alt4	
ADDR23	LCD_DAT20.alt4	
ADDR24	LCD_DAT21.alt4	
ADDR25	LCD_DAT22.alt4	
ADDR26	LCD_DAT23.alt4	

6.6.5.2 NAND Flash

The boot ROM supports a number of MLC/SLC NAND Flash devices from different vendors and LBA NAND Flash devices. The Error Correction and Control (ECC) subblock (BCH) is used to detect the errors.

6.6.5.2.1 NAND eFUSE Configuration

The boot ROM determines the configuration of external NAND flash by parameters, either provided by eFUSE, or sampled on GPIO pins, during boot. See Table 6-36 for parameters details.

NOTE

BOOT_CFGx sampled on GPIO pins depends on BT_FUSE_SEL setting. See Boot Fusemap for details.

NOTE

ROM always boots from CS0_B, while for multiple chip selects (such as some NAND chips), all other chip select pins of such NAND chip except CS0_B, should be pull-up in boot progress

Table 6-36. NAND Boot eFUSE Descriptions

Fuse	Config	Definition	GPIO[1]	Shipped Value	Settings
BOOT_CFG[15:12]	OEM	Boot Device Selection	Yes	0	0011 - Boot from NAND Interface
BOOT_CFG[7]	OEM	BT_TOGGLEMODE	Yes	0	0 - raw NAND 1 - toggle mode NAND
BOOT_CFG[11:10]	OEM	Pages In Block	Yes	0	00 - 128 01- 64 10- 32 11- 256
BOOT_CFG[9:8]	OEM	Row Address Cycles	Yes	00	00 - 3

Table continues on the next page...

IOMUXC memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
3033_0728	USB_OTG2_OC_SELECT_INPUT DAISY Register (IOMUXC_USB_OTG2_OC_SELECT_INPUT)	32	R/W	0000_0000h	8.2.7.454/ 2088
3033_072C	USB_OTG1_OC_SELECT_INPUT DAISY Register (IOMUXC_USB_OTG1_OC_SELECT_INPUT)	32	R/W	0000_0000h	8.2.7.455/ 2089
3033_0730	USB_OTG2_ID_SELECT_INPUT DAISY Register (IOMUXC_USB_OTG2_ID_SELECT_INPUT)	32	R/W	0000_0000h	8.2.7.456/ 2089
3033_0734	USB_OTG1_ID_SELECT_INPUT DAISY Register (IOMUXC_USB_OTG1_ID_SELECT_INPUT)	32	R/W	0000_0000h	8.2.7.457/ 2090
3033_0738	SD3_CD_B_SELECT_INPUT DAISY Register (IOMUXC_SD3_CD_B_SELECT_INPUT)	32	R/W	0000_0000h	8.2.7.458/ 2091
3033_073C	SD3_WP_SELECT_INPUT DAISY Register (IOMUXC_SD3_WP_SELECT_INPUT)	32	R/W	0000_0000h	8.2.7.459/ 2091

8.2.7.1 SW_MUX_CTL_PAD_GPIO1_IO08 SW MUX Control Register (IOMUXC_SW_MUX_CTL_PAD_GPIO1_IO08)

SW_MUX_CTL Register

Address: 3033_0000h base + 14h offset = 3033_0014h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved											SION	Reserved	MUX_MODE		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

IOMUXC_SW_PAD_CTL_PAD_EPDC_SDSHR field descriptions (continued)

Field	Description
	01 **DSE_1_X4** — X4 10 **DSE_2_X2** — X2 11 **DSE_3_X6** — X6

8.2.7.185 SW_PAD_CTL_PAD_EPDC_SDCE0 SW PAD Control Register (IOMUXC_SW_PAD_CTL_PAD_EPDC_SDCE0)

SW_PAD_CTL Register

Address: 3033_0000h base + 2F4h offset = 3033_02F4h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved									PS		PE	HYS	SRE	DSE	
Reset	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	0

IOMUXC_SW_PAD_CTL_PAD_EPDC_SDCE0 field descriptions

Field	Description
31–7 -	This field is reserved. Reserved
6–5 PS	Pull Select Field Select one out of next values for pad: EPDC_SDCE0 00 **PS_0_100K_PD** — 100K PD 01 **PS_1_5K_PU** — 5K PU 10 **PS_2_47K_PU** — 47K PU 11 **PS_3_100K_PU** — 100K PU
4 PE	Pull Enable Field Select one out of next values for pad: EPDC_SDCE0 0 **PE_0_Pull_Disabled** — Pull Disabled 1 **PE_1_Pull_Enabled** — Pull Enabled
3 HYS	Hyst. Enable Field Select one out of next values for pad: EPDC_SDCE0 0 **HYS_0_Hysteresis_Disabled** — Hysteresis Disabled 1 **HYS_1_Hysteresis_Enabled** — Hysteresis Enabled
2 SRE	Slew Rate Field Select one out of next values for pad: EPDC_SDCE0

Table continues on the next page...

IOMUXC_SW_PAD_CTL_PAD_LCD_DATA15 field descriptions (continued)

Field	Description
	10 **PS_2_47K_PU** — 47K PU 11 **PS_3_100K_PU** — 100K PU
4 PE	Pull Enable Field Select one out of next values for pad: LCD_DATA15 0 **PE_0_Pull_Disabled** — Pull Disabled 1 **PE_1_Pull_Enabled** — Pull Enabled
3 HYS	Hyst. Enable Field Select one out of next values for pad: LCD_DATA15 0 **HYS_0_Hysteresis_Disabled** — Hysteresis Disabled 1 **HYS_1_Hysteresis_Enabled** — Hysteresis Enabled
2 SRE	Slew Rate Field Select one out of next values for pad: LCD_DATA15 0 **SRE_0_Fast_Slew_Rate** — Fast Slew Rate 1 **SRE_1_Slow_Slew_Rate** — Slow Slew Rate
DSE	Drive Strength Field Select one out of next values for pad: LCD_DATA15 00 **DSE_0_X1** — X1 01 **DSE_1_X4** — X4 10 **DSE_2_X2** — X2 11 **DSE_3_X6** — X6

8.2.7.218 SW_PAD_CTL_PAD_LCD_DATA16 SW PAD Control Register (IOMUXC_SW_PAD_CTL_PAD_LCD_DATA16)

SW_PAD_CTL Register

Address: 3033_0000h base + 378h offset = 3033_0378h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved								PS			PE	HYS	SRE	DSE	
Reset	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	0

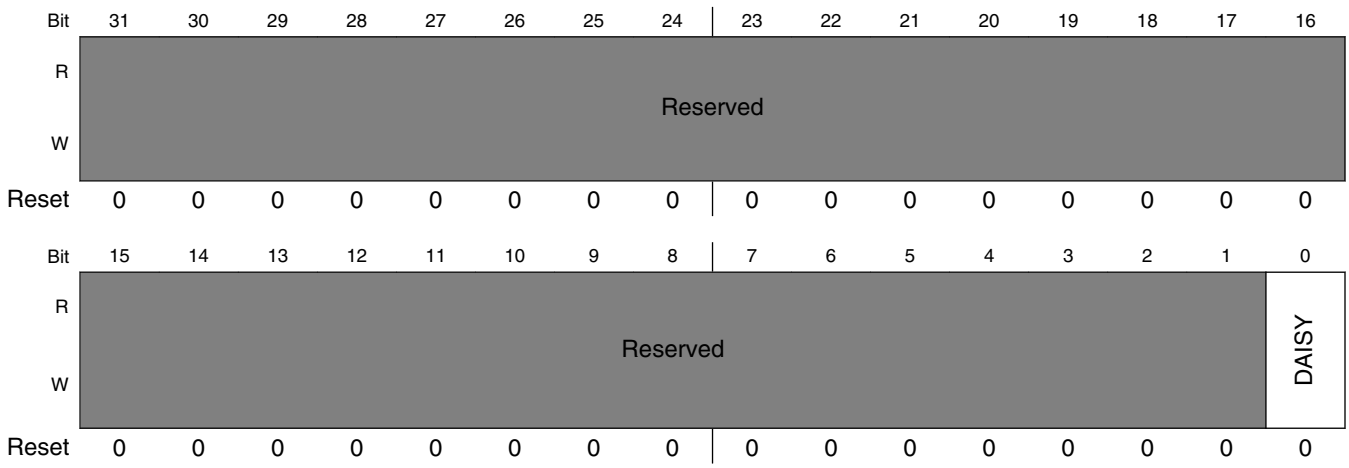
IOMUXC_SW_PAD_CTL_PAD_SD3_STROBE field descriptions

Field	Description
31–7 -	This field is reserved. Reserved
6–5 PS	Pull Select Field Select one out of next values for pad: SD3_STROBE 00 **PS_0_100K_PD** — 100K PD 01 **PS_1_5K_PU** — 5K PU 10 **PS_2_47K_PU** — 47K PU 11 **PS_3_100K_PU** — 100K PU
4 PE	Pull Enable Field Select one out of next values for pad: SD3_STROBE 0 **PE_0_Pull_Disabled** — Pull Disabled 1 **PE_1_Pull_Enabled** — Pull Enabled
3 HYS	Hyst. Enable Field Select one out of next values for pad: SD3_STROBE 0 **HYS_0_Hysteresis_Disabled** — Hysteresis Disabled 1 **HYS_1_Hysteresis_Enabled** — Hysteresis Enabled
2 SRE	Slew Rate Field Select one out of next values for pad: SD3_STROBE 0 **SRE_0_Fast_Slew_Rate** — Fast Slew Rate 1 **SRE_1_Slow_Slew_Rate** — Slow Slew Rate
DSE	Drive Strength Field Select one out of next values for pad: SD3_STROBE 00 **DSE_0_X1** — X1 01 **DSE_1_X4** — X4 10 **DSE_2_X2** — X2 11 **DSE_3_X6** — X6

8.2.7.363 FLEXTIMER2_CH4_SELECT_INPUT DAISY Register (IOMUXC_FLEXTIMER2_CH4_SELECT_INPUT)

DAISY Register

Address: 3033_0000h base + 5BCh offset = 3033_05BCh



IOMUXC_FLEXTIMER2_CH4_SELECT_INPUT field descriptions

Field	Description
31–1 -	This field is reserved. Reserved
0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain. 0 **LCD_DATA20_ALT1** — Selecting Pad: LCD_DATA20 Mode: ALT1 for FLEXTIMER2_CH4 1 **SAI2_TX_SYNC_ALT4** — Selecting Pad: SAI2_TX_SYNC Mode: ALT4 for FLEXTIMER2_CH4

The BCH hardware allows full programmability over the flash page layout to enable users flexibility in balancing ECC correction levels and ever-changing flash page sizes.

To initiate a NAND Flash write, software will program a GPMI DMA operation. The DMA need only program the GPMI control registers (and handle the requisite flash addressing handshakes) since the BCH will handle all data operations using its AXI bus interface. The BCH will then send the data to the GPMI controller to be written to flash as it computes the parity symbols. At the end of each data block the BCH will insert the parity symbols into the data stream so that the GPMI sees only a continuous stream of data to be written.

NAND Flash read operations operate in a similar manner. As the GPMI controller reads the device, all data is sent to the BCH hardware for error detection/correction. The BCH controller writes all incoming read data to system memory and in parallel computes the syndromes used to detect bit errors. If errors are detected within a block, the BCH hardware activates the error correction logic to determine where bit errors have occurred and ultimately correct them in the data buffer in system memory. After an entire flash page has been read and corrected, the BCH will signal an interrupt to the CPU.

The figure below indicates how data read from the GPMI is operated on within the BCH hardware. As the BCH receives data from the GPMI (top row), it is written to memory by the BCH's Bus Interface Unit (BIU) (second row). For blocks requiring correction, the KES logic will be activated after the entire block has been received. Once the error locator polynomial has been computed, the corrections are determined by the Chien Search and fed back to the BIU, which performs a read, modify, write operation on the buffer in memory to correct the data.

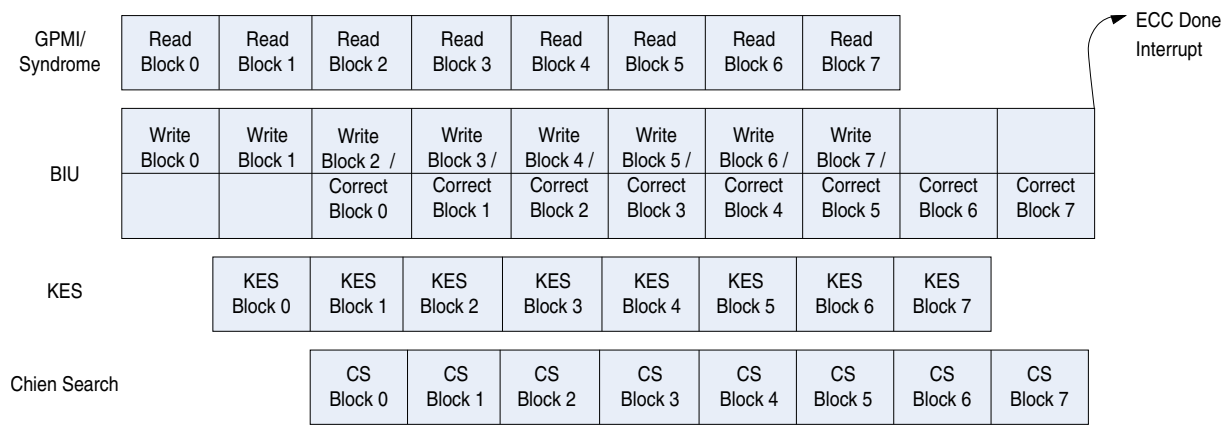
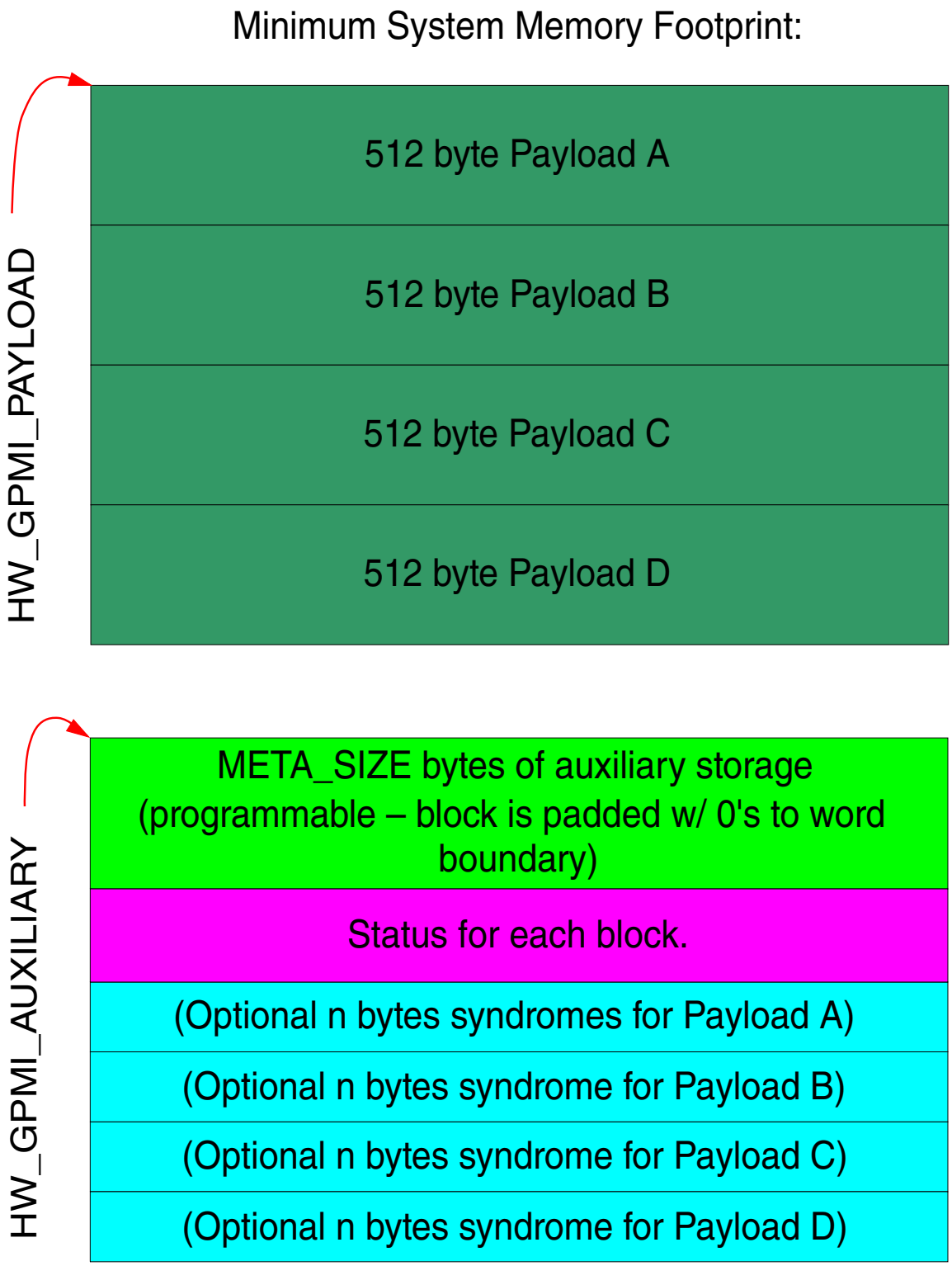


Figure 9-29. Block Pipeline while Reading Flash



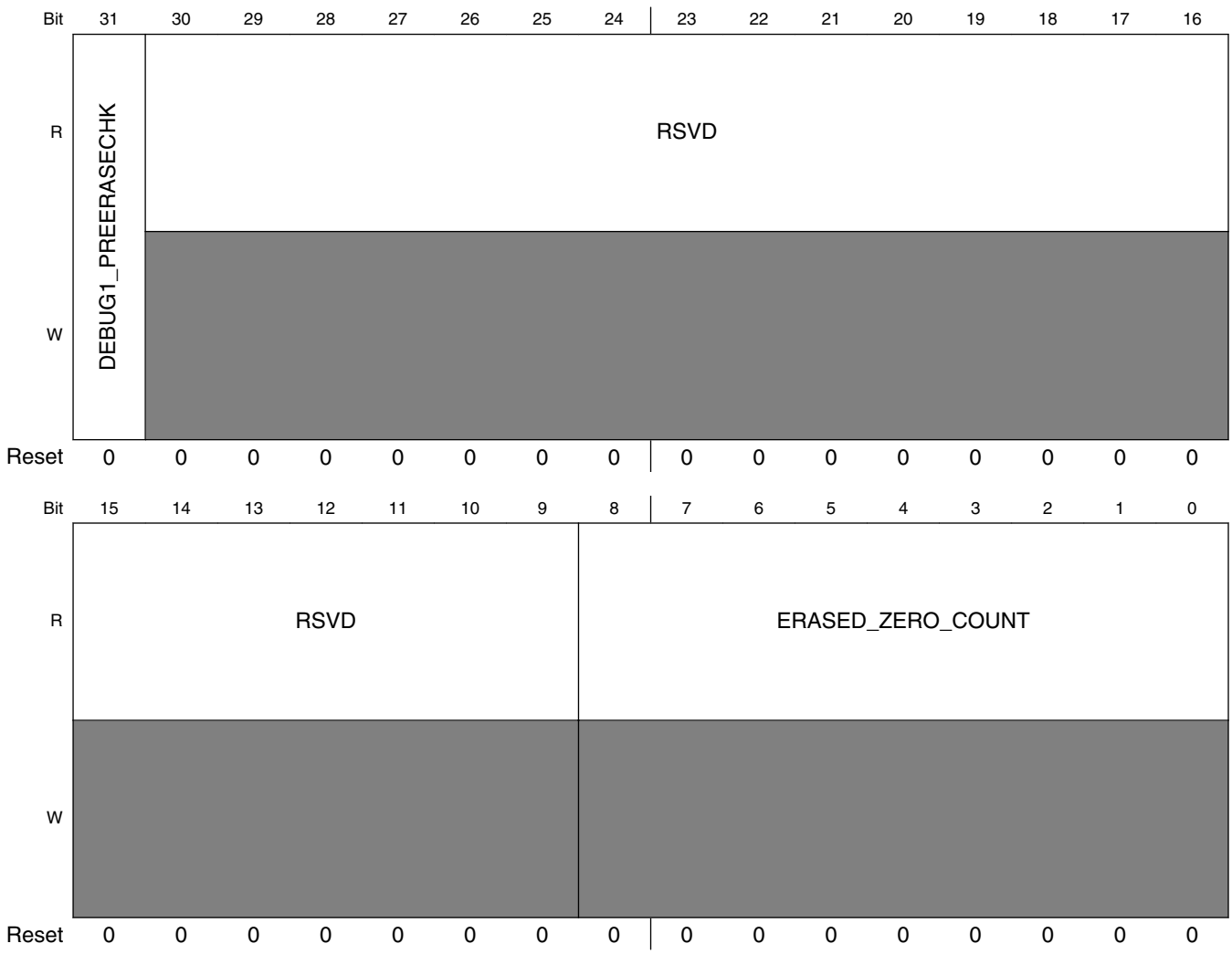
*Computed syndrome area consists of 2*t 13 (or 14)-bit symbols written as 16-bit half word.*

Figure 9-31. BCH Data Buffers in Memory

9.5.6.23 Hardware BCH ECC Debug Register 1 (BCH_DEBUG1*n*)

The BCH_DEBUG1 register provides erased zero count information and pre-erase check.

Address: 3300_4000h base + 170h offset + (4d × i), where i=0d to 3d



BCH_DEBUG1*n* field descriptions

Field	Description
31 DEBUG1_ PREERASECHK	Blank page enables pre-erase check. 0x0 Turn off pre-erase check 0x1 Turn on pre-erase check
30–9 RSVD	This field is reserved. This read-only field is reserved and always has the value 0.
ERASED_ ZERO_COUNT	The zero counts on one page.

QuadSPI memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
30BB_000C	Flash Configuration Register (QuadSPI1_FLSHCR)	32	R/W	0000_0303h	10.2.14.3/ 2676
30BB_0010	Buffer0 Configuration Register (QuadSPI1_BUF0CR)	32	R/W	0000_0000h	10.2.14.4/ 2677
30BB_0014	Buffer1 Configuration Register (QuadSPI1_BUF1CR)	32	R/W	0000_0000h	10.2.14.5/ 2678
30BB_0018	Buffer2 Configuration Register (QuadSPI1_BUF2CR)	32	R/W	0000_0000h	10.2.14.6/ 2679
30BB_001C	Buffer3 Configuration Register (QuadSPI1_BUF3CR)	32	R/W	See section	10.2.14.7/ 2680
30BB_0020	Buffer Generic Configuration Register (QuadSPI1_BFGENCR)	32	R/W	0000_0000h	10.2.14.8/ 2681
30BB_0030	Buffer0 Top Index Register (QuadSPI1_BUF0IND)	32	R/W	0000_0000h	10.2.14.9/ 2682
30BB_0034	Buffer1 Top Index Register (QuadSPI1_BUF1IND)	32	R/W	0000_0000h	10.2.14.10/ 2682
30BB_0038	Buffer2 Top Index Register (QuadSPI1_BUF2IND)	32	R/W	0000_0000h	10.2.14.11/ 2683
30BB_0100	Serial Flash Address Register (QuadSPI1_SFAR)	32	R/W	0000_0000h	10.2.14.12/ 2684
30BB_0108	Sampling Register (QuadSPI1_SMPR)	32	R/W	0000_0000h	10.2.14.13/ 2684
30BB_010C	RX Buffer Status Register (QuadSPI1_RBSR)	32	R	0000_0000h	10.2.14.14/ 2685
30BB_0110	RX Buffer Control Register (QuadSPI1_RBCT)	32	R/W	0000_0000h	10.2.14.15/ 2686
30BB_0150	TX Buffer Status Register (QuadSPI1_TBSR)	32	R	0000_0000h	10.2.14.16/ 2687
30BB_0154	TX Buffer Data Register (QuadSPI1_TBDR)	32	R/W	0000_0000h	10.2.14.17/ 2687
30BB_015C	Status Register (QuadSPI1_SR)	32	R	0000_3800h	10.2.14.18/ 2689
30BB_0160	Flag Register (QuadSPI1_FR)	32	w1c	0800_0000h	10.2.14.19/ 2692
30BB_0164	Interrupt and DMA Request Select and Enable Register (QuadSPI1_RSER)	32	R/W	0000_0000h	10.2.14.20/ 2695
30BB_0168	Sequence Suspend Status Register (QuadSPI1_SPNDST)	32	R	0000_0000h	10.2.14.21/ 2698
30BB_016C	Sequence Pointer Clear Register (QuadSPI1_SPTRCLR)	32	R/W	0000_0000h	10.2.14.22/ 2700
30BB_0180	Serial Flash A1 Top Address (QuadSPI1_SFA1AD)	32	R/W	0000_0000h	10.2.14.23/ 2700
30BB_0184	Serial Flash A2 Top Address (QuadSPI1_SFA2AD)	32	R/W	0000_0000h	10.2.14.24/ 2701

Table continues on the next page...

uSDHCx_INT_STATUS field descriptions (continued)

Field	Description
	register should be confirmed. Because the card state may possibly be changed when the Host Driver clears this bit and the interrupt event may not be generated. When this bit is cleared, it will be set again if a card is inserted. In order to leave it cleared, clear the Card Inserted Status Enable bit in Interrupt Status Enable register. 1 Card inserted 0 Card state unstable or removed
5 BRR	Buffer Read Ready This status bit is set if the Buffer Read Enable bit, in the Present State register, changes from 0 to 1. Refer to the Buffer Read Enable bit in the Present State register for additional information. This bit indicates that cmd19 is finished in tuning process. 1 Ready to read buffer 0 Not ready to read buffer
4 BWR	Buffer Write Ready This status bit is set if the Buffer Write Enable bit, in the Present State register, changes from 0 to 1. Refer to the Buffer Write Enable bit in the Present State register for additional information. 1 Ready to write buffer: 0 Not ready to write buffer
3 DINT	DMA Interrupt Occurs only when the internal DMA finishes the data transfer successfully. Whenever errors occur during data transfer, this bit will not be set. Instead, the DMAE bit will be set. Either Simple DMA or ADMA finishes data transferring, this bit will be set. 1 DMA Interrupt is generated 0 No DMA Interrupt
2 BGE	Block Gap Event If the Stop At Block Gap Request bit in the Protocol Control register is set, this bit is set when a read or write transaction is stopped at a block gap. If Stop At Block Gap Request is not set to 1, this bit is not set to 1. In the case of a Read Transaction: This bit is set at the falling edge of the DATA Line Active Status (When the transaction is stopped at SD Bus timing). The Read Wait must be supported in order to use this function. In the case of Write Transaction: This bit is set at the falling edge of Write Transfer Active Status (After getting CRC status at SD Bus timing). 1 Transaction stopped at block gap 0 No block gap event
1 TC	Transfer Complete This bit is set when a read or write transfer is completed. In the case of a Read Transaction: This bit is set at the falling edge of the Read Transfer Active Status. There are two cases in which this interrupt is generated. The first is when a data transfer is completed as specified by the data length (after the last data has been read to the Host System). The second is when data has stopped at the block gap and completed the data transfer by setting the Stop At Block Gap Request bit in the Protocol Control register (after valid data has been read to the Host System).

Table continues on the next page...

ENET_EIMR field descriptions (continued)

Field	Description
18 RXFLUSH_1	Corresponds to interrupt source EIR[RXFLUSH_1] and determines whether an interrupt condition can generate an interrupt. At every module clock, the EIR samples the signal generated by the interrupting source. EIR[RXFLUSH_1] reflects the state of the interrupt signal even if the corresponding EIMR field is cleared.
19 RXFLUSH_0	Corresponds to interrupt source EIR[RXFLUSH_0] and determines whether an interrupt condition can generate an interrupt. At every module clock, the EIR samples the signal generated by the interrupting source. EIR[RXFLUSH_0] reflects the state of the interrupt signal even if the corresponding EIMR field is cleared.
20–22 Reserved	This field is reserved. This write-only field is reserved. It must always be written with the value 0.
23 Reserved	This field is reserved. This write-only field is reserved. It must always be written with the value 0.
24 TXF2	Transmit frame interrupt, class 2 Corresponds to interrupt source EIR[TXF2] and determines whether an interrupt condition can generate an interrupt. At every module clock, the EIR samples the signal generated by the interrupting source. EIR[TXF2] reflects the state of the interrupt signal even if the corresponding EIMR field is cleared.
25 TXB2	Transmit buffer interrupt, class 2 Corresponds to interrupt source EIR[TXB2] and determines whether an interrupt condition can generate an interrupt. At every module clock, the EIR samples the signal generated by the interrupting source. EIR[TXB2] reflects the state of the interrupt signal even if the corresponding EIMR field is cleared.
26 RXF2	Receive frame interrupt, class 2 Corresponds to interrupt source EIR[RXF2] and determines whether an interrupt condition can generate an interrupt. At every module clock, the EIR samples the signal generated by the interrupting source. EIR[RXF2] reflects the state of the interrupt signal even if the corresponding EIMR field is cleared.
27 RXB2	Receive buffer interrupt, class 2 Corresponds to interrupt source EIR[RXB2] and determines whether an interrupt condition can generate an interrupt. At every module clock, the EIR samples the signal generated by the interrupting source. EIR[RXB2] reflects the state of the interrupt signal even if the corresponding EIMR field is cleared.
28 TXF1	Transmit frame interrupt, class 1 Corresponds to interrupt source EIR[TXF1] and determines whether an interrupt condition can generate an interrupt. At every module clock, the EIR samples the signal generated by the interrupting source. EIR[TXF1] reflects the state of the interrupt signal even if the corresponding EIMR field is cleared.
29 TXB1	Transmit buffer interrupt, class 1 Corresponds to interrupt source EIR[TXB1] and determines whether an interrupt condition can generate an interrupt. At every module clock, the EIR samples the signal generated by the interrupting source. EIR[TXB1] reflects the state of the interrupt signal even if the corresponding EIMR field is cleared.
30 RXF1	Receive frame interrupt, class 1 Corresponds to interrupt source EIR[RXF1] and determines whether an interrupt condition can generate an interrupt. At every module clock, the EIR samples the signal generated by the interrupting source. EIR[RXF1] reflects the state of the interrupt signal even if the corresponding EIMR field is cleared.
31 RXB1	Receive buffer interrupt, class 1

Table continues on the next page...

Table 11-65. iTD Buffer Pointer Page 3-6 (continued)

Bit	Description
11-0 Reserved	These bits reserved for future use and should be set to zero.

11.3.4.2.4 Split Transaction Isochronous Transfer Descriptor (siTD)

All Full-speed isochronous transfers through the internal transaction translator are managed using the siTD data structure. This data structure satisfies the operational requirements for managing the split transaction protocol.

The following table shows the Split Transaction Isochronous Transfer Descriptor (siTD).

Table 11-66. Split Transaction Isochronous Transfer Descriptor

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Addr
Next Link Pointer																											0		Typ		T	03-00
I/O	Port Number							-	Hub Addr							Reserved				EndPt				-	Device Address							07-04[1]
Reserved																µFrame C-mask				µFrame S-mask												0B-08[1]
ioc	P	Reserved				Total Bytes to Transfer										µFrame C-prog-mask				Status												0F-0C[2]
Buffer Pointer (Page 0)																				Current Offset												13-10[2]
Buffer Pointer (Page 1)																				Reserved							TP		T-count			17-14[2]
Back Pointer																											0				T	1B-18

1. 04-0B: Static Endpoint State
2. 0C-13: Transfer results

Host Controller Read/Write

Host Controller Read Only

11.3.4.2.4.1 Next Link Pointer

DWord0 of a siTD is a pointer to the next schedule data structure.

The following table describes the Next Link Pointer fields.

- If the Advance Queue state fails to advance the queue because the target qTD is not active, or
- If the *Halted* bit is one on exit from the Fetch QH state.

There is no functional requirement that the host controller wait until the current transaction is complete before using the horizontal pointer to read the next linked data structure. However, it must wait until the current transaction is complete before executing the next data structure.

11.3.4.3.10.6 Buffer Pointer List Use for Data Streaming with qTDs

A qTD has an array of buffer pointers, which is used to reference the data buffer for a transfer. This specification requires that the buffer associated with the transfer be *virtually contiguous*.

This means: if the buffer spans more than one physical page, it must obey the following rules (the figure below illustrates an example):

- The first portion of the buffer must begin at some offset in a page and extend through the end of the page.
- The remaining buffer cannot be allocated in small chunks scattered around memory. For each 4 K chunk beyond the first page, each buffer portion matches to a full 4 K page. The final portion, which may only be large enough to occupy a portion of a page, must start at the top of the page and be contiguous within that page.

The buffer pointer list in the qTD is long enough to support a maximum transfer size of 20 K bytes. This case occurs when all five buffer pointers are used and the first offset is zero. A qTD handles a 16 Kbyte buffer with any starting buffer alignment.

The host controller uses the field *C_Page* field as an index value to determine which buffer pointer in the list should be used to start the current transaction. The host controller uses a different buffer pointer for each physical page of the buffer. This is always true, even if the buffer is physically contiguous.

The host controller must detect when the current transaction spans a page boundary and automatically move to the next available buffer pointer in the page pointer list. The next available pointer is reached by incrementing *C_Page* and pulling the next page pointer from the list. Software must ensure there are sufficient buffer pointers to move the amount of data specified in the *Bytes to Transfer* field.

The following figure illustrates a nominal example of how System software would initialize the buffer pointers list and the *C_Page* field for a transfer size of 16383 bytes. *C_Page* is set to zero. The upper 20-bits of Page 0 references the start of the physical

FTMx_INVCTRL field descriptions

Field	Description
0–27 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
28 INV3EN	Pair Channels 3 Inverting Enable 0 Inverting is disabled. 1 Inverting is enabled.
29 INV2EN	Pair Channels 2 Inverting Enable 0 Inverting is disabled. 1 Inverting is enabled.
30 INV1EN	Pair Channels 1 Inverting Enable 0 Inverting is disabled. 1 Inverting is enabled.
31 INV0EN	Pair Channels 0 Inverting Enable 0 Inverting is disabled. 1 Inverting is enabled.

12.2.3.24 FTM Software Output Control (FTMx_SWOCTRL)

This register enables software control of channel (n) output and defines the value forced to the channel (n) output:

- The CHnOC bits enable the control of the corresponding channel (n) output by software.
- The CHnOCV bits select the value that is forced at the corresponding channel (n) output.

This register has a write buffer. The fields are updated by the SWOCTRL register synchronization.

Address: Base address + 94h offset

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

PXP_HW_PXP_DITHER_STORE_PITCH field descriptions

Field	Description
31–16 CH1_OUT_ PITCH	This field indicates the channel 1 input pitch
CH0_OUT_ PITCH	This field indicates the channel 0 input pitch

13.6.12.149 PXP_HW_PXP_DITHER_STORE_SHIFT_CTRL_CH0

This register defines the control bits for the pxp store_engine sub-block.

HW_PXP_DITHER_STORE_SHIFT_CTRL_CH0: 0xA30

HW_PXP_DITHER_STORE_SHIFT_CTRL_CH0_SET: 0xA34

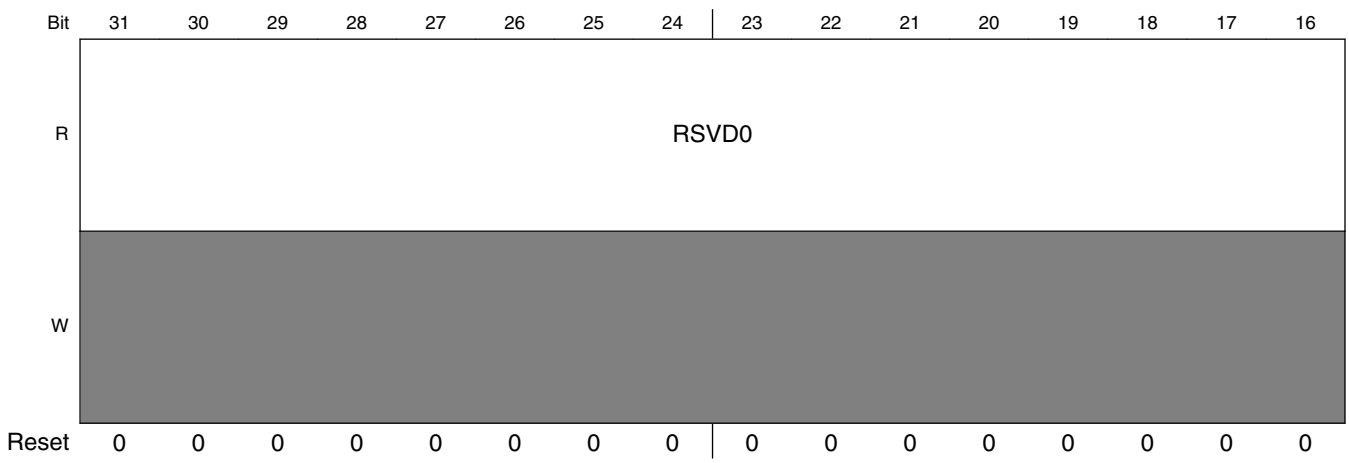
HW_PXP_DITHER_STORE_SHIFT_CTRL_CH0_CLR: 0xA38

HW_PXP_DITHER_STORE_SHIFT_CTRL_CH0_TOG: 0xA3C

The Control register contains the control bits for the pxp store_engine sub-block.

EXAMPLE

Address: 3070_0000h base + A30h offset = 3070_0A30h



ADCx_CHA_B_CNV_RSLT field descriptions (continued)

Field	Description
15–12 Reserved2	This field is reserved. This field is reserved. This read-only field is reserved and always has the value 0.
CHA_CNV_RSLT	Channel A Conversion Result Channel A conversion result stores in this field.

14.1.6.17 Channel C and D Conversion Result (ADCx_CHC_D_CNV_RSLT)

Address: Base address + 100h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved1				CHD_CNV_RSLT												Reserved2				CHC_CNV_RSLT												
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ADCx_CHC_D_CNV_RSLT field descriptions

Field	Description
31–28 Reserved1	This field is reserved. This field is reserved. This read-only field is reserved and always has the value 0.
27–16 CHD_CNV_ RSLT	Channel D Conversion Result Channel D conversion result stores in this field.
15–12 Reserved2	This field is reserved. This field is reserved. This read-only field is reserved and always has the value 0.
CHC_CNV_ RSLT	Channel C Conversion Result Channel C conversion result stores in this field.

14.1.6.18 Channel Software Conversion Result (ADCx_CH_SW_CNV_RSLT)

Address: Base address + 110h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved																				CH_SW_CNV_RSLT												
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0