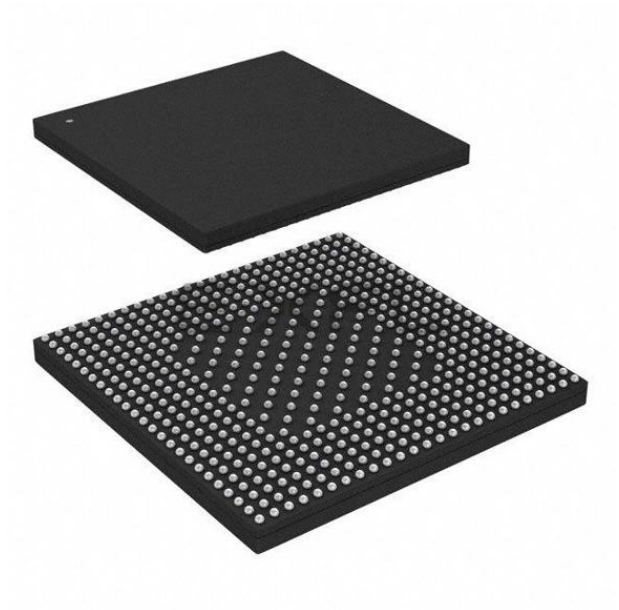

Welcome to E-XFL.COM

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	Active
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	LPDDR2, LPDDR3, DDR3, DDR3L
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-TFBGA (12x12)
Purchase URL	https://www.e-xfl.com/product-detail/nxp-semiconductors/mcimx7s5evk08sc

4.12.5.4 HIGHZ Instruction

All output drivers, including the two-state drivers, are turned off (that is, high impedance). The instruction selects the bypass register.

In this mode, all internal pullup resistors on all the pins (except for the TMS, TDI, TCK, TRSTB pins) are disabled. This disabling functionality is not built into SJC, but should be implemented by some logic in the SOC/IO Pads.

For more details on the function and use of HIGHZ, refer to the IEEE 1149.1 document.

The HIGHZ instruction also asserts internal reset for the cores (through CCM, refer to Figure 4-36) to force a predictable internal state while performing external boundary scan operations.

4.12.5.5 BYPASS Instruction

Selects the single Bit bypass register and the system logic controls the I/O pins.

This creates a shift-register path from TDI to the bypass register and, finally, to TDO, circumventing the boundary scan register. This instruction is used to enhance test efficiency when a component other than the SoC Core based device becomes the device under test.

When the bypass register is selected by the current instruction, the shift-register stage is set to a logic zero on the rising edge of TCK in the capture-DR controller state. Therefore, the first bit to be shifted out after selecting the bypass register is always a logic zero.

For more details on the function and use of BYPASS, refer to the appropriate IEEE 1149.1 document.

4.12.5.6 ENABLE_ExtraDebug Instruction

The TDI and TDO pins are connected directly to the ExtraDebug registers, the SJC TAP controller remaining connected to TDI and TMS.

The ExtraDebug shift register consists of 38 bits (maximum) comprising a 32-bits data field (maximum length, see Accessing ExtraDebug Registers,), a 5 bits address field and read/$\overline{\text{write}}$ bit. On a register read, the data field does not need to be filled in. The particular ExtraDebug register connected between TDI and TDO at a given time is

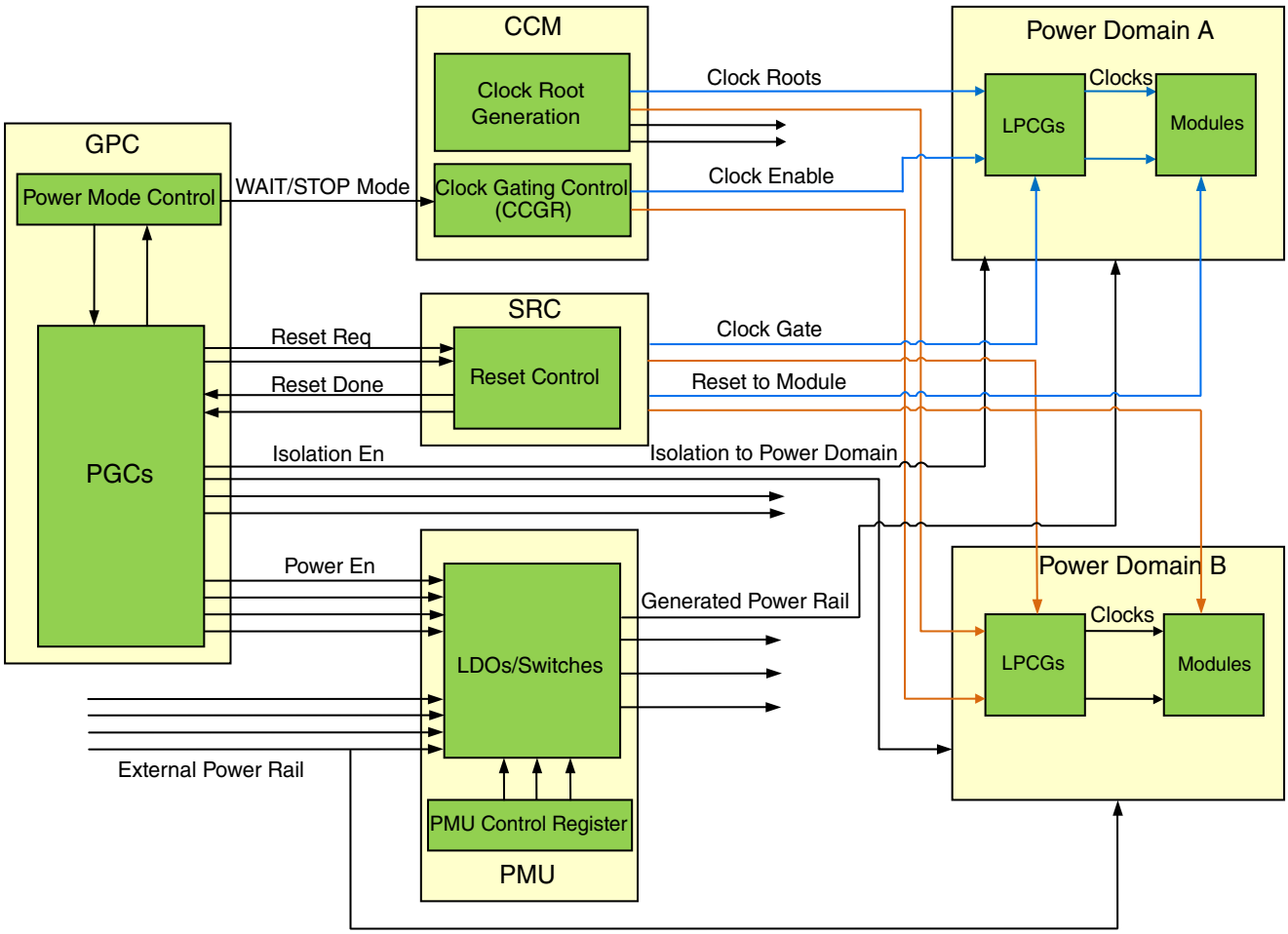

Figure 5-8. Power System

The power generation and management system consists of the GPC, PMU, CCM and SRC.

- The PMU generates the power rails with LDOs or power switches from external power input.
- The GPC is the central controller for all the power system. The key roles of the GPC are chip power mode control and power domains management.
- The CCM generates the clock for modules in each power domain, and also enable/disable the clocks based on the power mode from GPC.
- The SRC generates the reset signal for each power domain during the power up of a domain.

5.2.9.6 Anadig 480MHz PLL Control Register (CCM_ANALOG_PLL_480*n*)

The control register provides control for the 480MHz PLL.

Address: 3036_0000h base + B0h offset + (4d × i), where i=0d to 3d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	LOCK	RSVD1		PFD2_DIV2_CLKGATE	PFD1_DIV2_CLKGATE	PFD0_DIV2_CLKGATE	PFD7_OVERRIDE	PFD6_OVERRIDE	PFD5_OVERRIDE	PFD4_OVERRIDE	PFD3_OVERRIDE	PFD2_OVERRIDE	PFD1_OVERRIDE	PFD0_OVERRIDE	PLL_480_OVERRIDE	BYPASS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	BYPASS_CLK_SRC		ENABLE_CLK	POWERDOWN	HOLD_RING_OFF	DOUBLE_CP	HALF_CP	DOUBLE_LF	HALF_LF	MAIN_DIV4_CLKGATE	MAIN_DIV2_CLKGATE	MAIN_DIV1_CLKGATE	RSVD0			DIV_SELECT
W																
Reset	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0

SRC_A7RCR0 field descriptions (continued)

Field	Description
	0 This register is not assigned to domain3. The master from domain3 cannot write to this register. 1 This register is assigned to domain3. The master from domain3 can write to this register
26 DOMAIN2	Domain2 assignment control. Effective when dom_en is set to 1. 0 This register is not assigned to domain2. The master from domain3 cannot write to this register. 1 This register is assigned to domain2. The master from domain3 can write to this register
25 DOMAIN1	Domain1 assignment control. Effective when dom_en is set to 1. 0 This register is not assigned to domain1. The master from domain3 cannot write to this register. 1 This register is assigned to domain1. The master from domain3 can write to this register
24 DOMAIN0	Domain0 assignment control. Effective when dom_en is set to 1. 0 This register is not assigned to domain0. The master from domain3 cannot write to this register. 1 This register is assigned to domain0. The master from domain3 can write to this register
23–22 -	This field is reserved. Reserved
21 A7_L2RESET	Software reset for A7 Snoop Control Unit (SCU). **NOTE:** This is a self clearing bit. Once it is set to 1, the reset process will begin, and once it finishes, this bit will be self cleared. 0 do not assert SCU reset 1 assert SCU reset
20 A7_SOC_DBG_ RESET	Software reset for system level debug reset. It initializes the shared Debug APB, the CTI, and the CTM. It also causes: • A7_dbgreset[1:0] and A7_etmreset[1:0] to be asserted • debug logic in the processor power domain and in the debug power domain to be reset **NOTE:** This is a self clearing bit. Once it is set to 1, the reset process will begin, and once it finishes, this bit will be self cleared. 0 do not assert system level debug reset 1 assert system level debug reset
19–16 MASK_WDOG1_ RST	Mask wdog1_rst_b source. If these 4 bits are coded from A to 5 then, the wdog1_rst_b input to SRC will be masked and the wdog1_rst_b will not create a reset to the chip. **NOTE:** During the time the WDOG event is masked using SRC logic, it is likely that the WDOG Reset Status Register (WRSR) bit 1 (which indicates a WDOG timeout event) will get asserted. software / OS developer must prepare for this case. Re-enabling the WDOG is possible, by un-masking it in SRC, though it must be preceded by servicing the WDOG. However, for the case that the event has been asserted, the status bit (WRSR bit-1) will remain asserted, regardless of servicing the WDOG module. (Hardware reset is the only way to cause the de-assertion of that bit). Any other code will be coded to 1010 i.e. wdog1_rst_b is not masked 0101 wdog1_rst_b is masked 1010 wdog1_rst_b is not masked
15–14 -	This field is reserved. Reserved

Table continues on the next page...

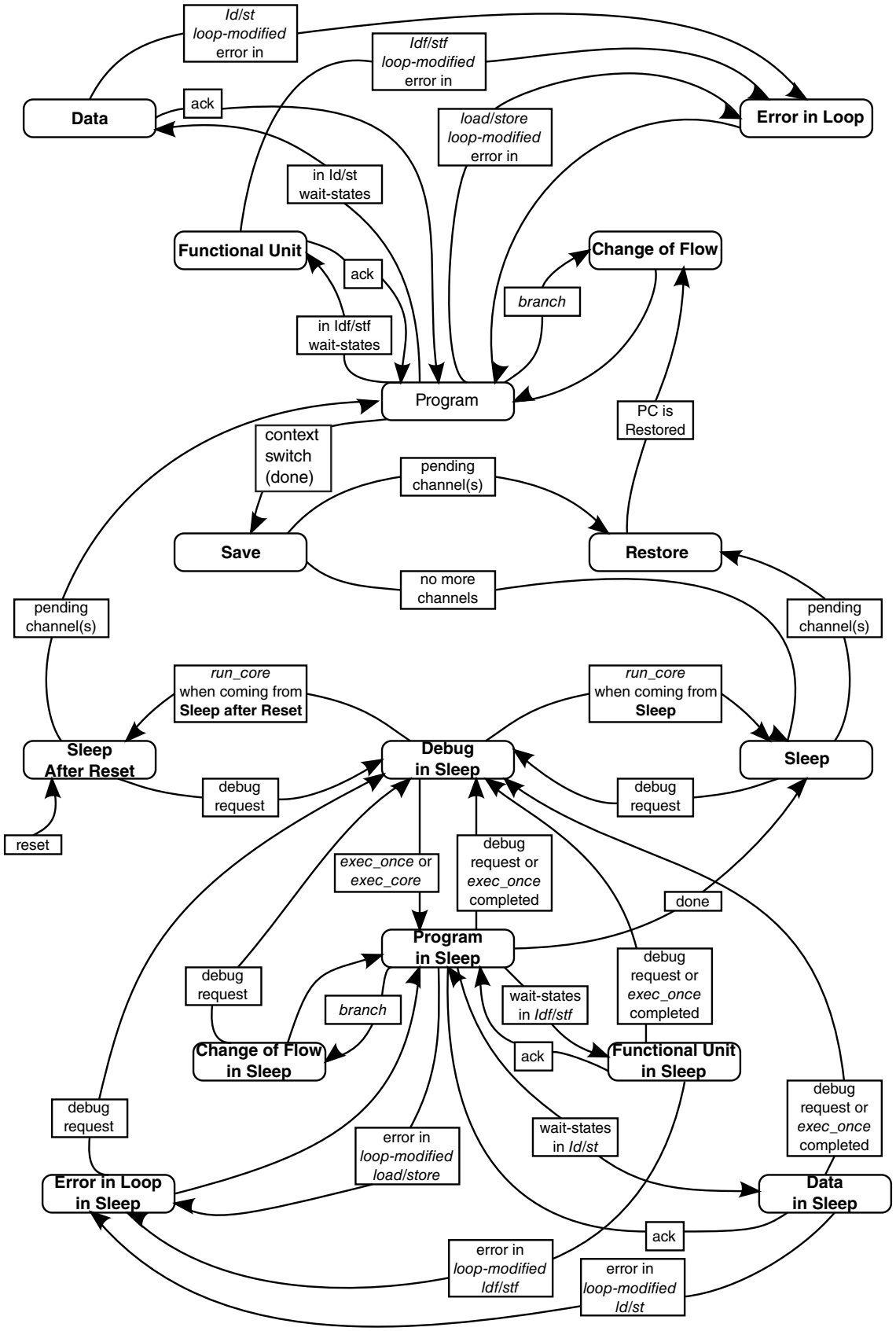

Figure 7-2. PCU State Diagram

- ORIr,immediate-This performs an or between GReg[r] and a zero-extended immediate value and, stores the result in GReg[r].
- ANDNr,s-This performs an and between GReg[r] and the negated GReg[s], and stores the result in GReg[r].
- ANDNIr,immediate-This performs an and between GReg[r] and the negated zero-extended immediate value, and stores the result in GReg[r].
- ANDr,s-This performs an and between GReg[r] and GReg[s], and stores the result in GReg[r].
- ANDIr,immediate-This performs an and between GReg[r] and a zero-extended immediate value, and stores the result in GReg[r].

7.2.4.11.8 Arithmetic Instructions

Arithmetic instructions modify the T bit in the processor status according to the result of the operation. The T bit is set if the result is zero, otherwise it is cleared.

- ADD r,s-This performs the addition of GReg[r] and GReg[s], and stores the result in GReg[r].
- ADDI r,immediate-This performs the addition of GReg[r] and a zero-extended immediate value, and stores the result in GReg[r].
- SUB r,s-This performs the subtraction of GReg[s] from GReg[r], and stores the result in GReg[r].
- SUBIr,immediate-This performs the subtraction of a zero-extended immediate value from GReg[r], and stores the result in GReg[r].

7.2.4.11.9 Compare Instructions

Compare instructions modify the T bit in the processor status according to the result of the operation. The T bit is set if the comparison is true, otherwise it is cleared.

NOTE

Only one version of the immediate form is implemented. Non-equality comparisons to immediate values will require two instructions.

- CMPEQ r,s-This sets T when registers GReg[r] and GReg[s] are equal.
- CMPEQIr,immediate-This sets T when register GReg[r] and the zero-extended immediate value are equal.
- CMPLTr,s-This sets T when register GReg[r] is less than and not equal to GReg[s]. The comparison is signed.
- CMPHS r,s-This sets T when register GReg[r] is greater than or equal to GReg[s]. The comparison is signed.

IOMUXC_SW_MUX_CTL_PAD_SD2_CLK field descriptions (continued)

Field	Description
	011 **ALT3_GPT4_CLK** — Select mux mode: ALT3 mux port: CLK of instance: GPT4 101 **ALT5_GPIO5_IO12** — Select mux mode: ALT5 mux port: IO12 of instance: GPIO5

8.2.7.107 SW_MUX_CTL_PAD_SD2_CMD SW MUX Control Register (IOMUXC_SW_MUX_CTL_PAD_SD2_CMD)

SW_MUX_CTL Register

Address: 3033_0000h base + 1BCh offset = 3033_01BCh

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	1	0	1

IOMUXC_SW_MUX_CTL_PAD_SD2_CMD field descriptions

Field	Description
31–5 -	This field is reserved. Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1 **ENABLED** — Force input path of pad SD2_CMD 0 **DISABLED** — Input Path is determined by functionality
3 -	This field is reserved. Reserved

Table continues on the next page...

The DDRMC SoC application bus interface supports an AMBA 3 AXI interface. The DDRMC has flexible address mapper logic to allow application-specific mapping of row, column, bank, bank group, and rank bits. The DDRMC includes a DFI interface that supports DDR3 (JEDEC JESD79-3*), LPDDR2 (JEDECJESD209-2*), and LPDDR3 (JEDEC 209-3*) standards.

9.2.2.1 Block diagram

The DDRMC converts system bus transactions into memory commands on the DFI interface that are compliant with the DDR protocols. The following figure is the block diagram of the DDRMC .

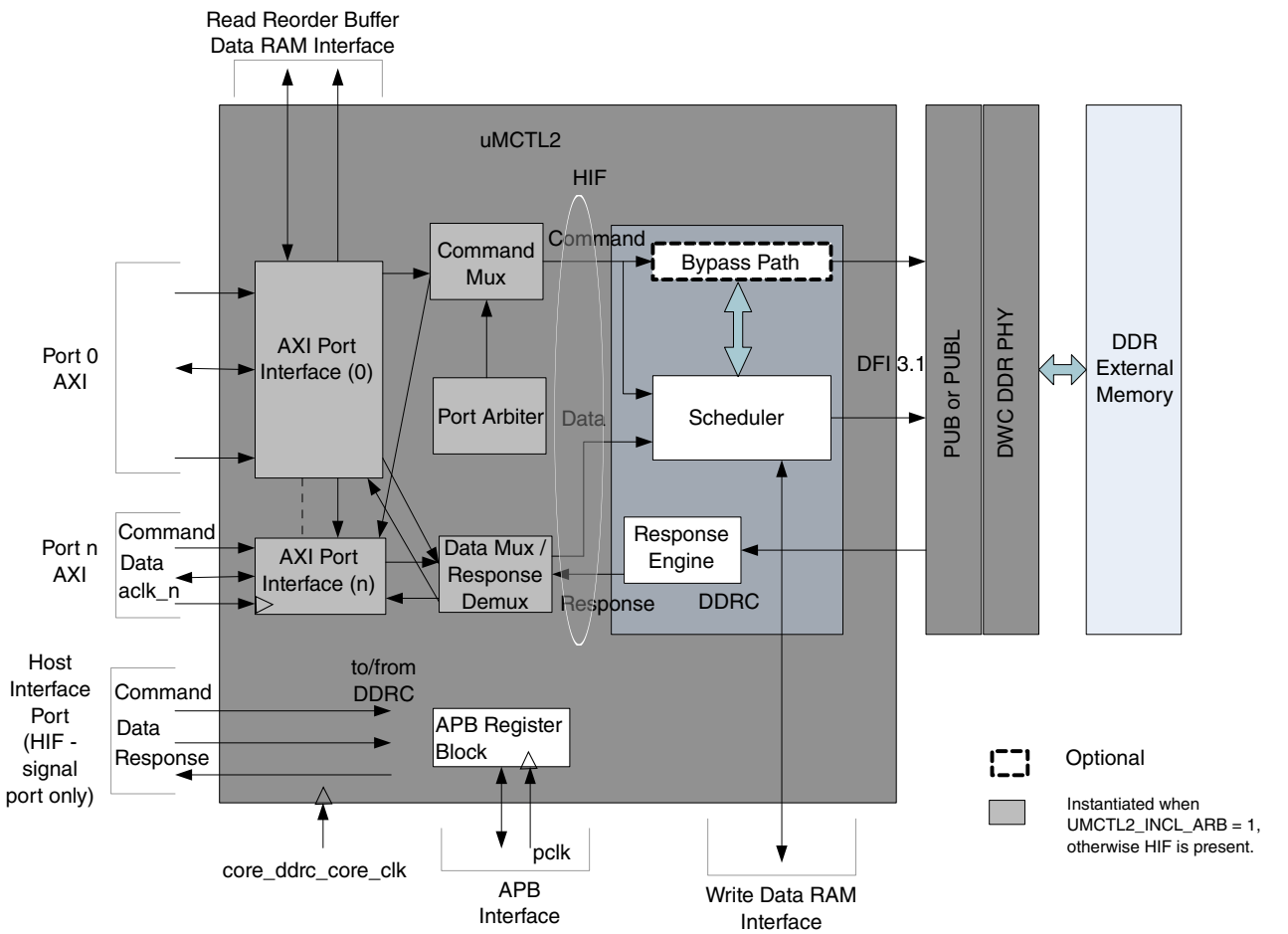

Figure 9-1. DDRMC Block Diagram

The DDRMC contains the following main architectural components:

- The AXI Port Interface (XPI) block: This block provides the interface to the application ports. It provides bus protocol handling, data buffering and reordering for read data, data bus size conversion (upsizing or downsizing), and memory burst address alignment.

APBH_CH*n*_CMD field descriptions

Field	Description
31–16 XFER_COUNT	This field indicates the number of bytes to transfer to or from the appropriate PIO register in the GPMI0 device. A value of 0 indicates a 64 KBytes transfer.
15–12 CMDWORDS	This field indicates the number of command words to send to the GPMI0, starting with the base PIO address of the GPMI0 control register and incrementing from there. Zero means transfer NO command words
11–9 -	This field is reserved. Reserved, always set to zero.
8 HALTONTERMINATE	A value of one indicates that the channel will immeditately terminate the current descriptor and halt the DMA channel if a terminate signal is set. A value of 0 will still cause an immediate terminate of the channel if the terminate signal is set, but the channel will continue as if the count had been exhausted, meaning it will honor IRQONCMPLT, CHAIN, SEMAPHORE, and WAIT4ENDCMD.
7 WAIT4ENDCMD	A value of one indicates that the channel will wait for the end of command signal to be sent from the APBH device to the DMA before starting the next DMA command.
6 SEMAPHORE	A value of one indicates that the channel will decrement its semaphore at the completion of the current command structure. If the semaphore decrements to zero, then this channel stalls until software increments it again.
5 NANDWAIT4READY	A value of one indicates that the NAND DMA channel will will wait until the NAND device reports "ready" before executing the command. It is ignored for non-NAND DMA channels.
4 NANDLOCK	A value of one indicates that the NAND DMA channel will remain "locked" in the arbiter at the expense of other NAND DMA channels. It is ignored for non-NAND DMA channels.
3 IRQONCMPLT	A value of one indicates that the channel will cause the interrupt status bit to be set upon completion of the current command, i.e. after the DMA transfer is complete.
2 CHAIN	A value of one indicates that another command is chained onto the end of the current command structure. At the completion of the current command, this channel will follow the pointer in APBH_CHn_CMDAR to find the next command.
COMMAND	This bitfield indicates the type of current command: 0x0 **NO_DMA_XFER** — Perform any requested PIO word transfers but terminate command before any DMA transfer. 0x1 **DMA_WRITE** — Perform any requested PIO word transfers and then perform a DMA transfer from the peripheral for the specified number of bytes. 0x2 **DMA_READ** — Perform any requested PIO word transfers and then perform a DMA transfer to the peripheral for the specified number of bytes. 0x3 **DMA_SENSE** — Perform any requested PIO word transfers and then perform a conditional branch to the next chained device. Follow the NEXCMD_ADDR pointer if the perpheral sense is true. Follow the BUFFER_ADDRESS as a chain pointer if the peripheral sense line is false.

9.4.5.11 APBH DMA Channel n Buffer Address Register (APBH_CH*n*_BAR)

The APBH DMA Channel n buffer address register contains a pointer to the data buffer for the transfer. For immediate forms, the data is taken from this register. This is a byte address which means transfers can start on any byte boundary.

QuadSPI memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
30BB_03E0	Look-up Table register (QuadSPI1_LUT52)	32	R/W	0000_0000h	10.2.14.32/ 2707
30BB_03E4	Look-up Table register (QuadSPI1_LUT53)	32	R/W	0000_0000h	10.2.14.32/ 2707
30BB_03E8	Look-up Table register (QuadSPI1_LUT54)	32	R/W	0000_0000h	10.2.14.32/ 2707
30BB_03EC	Look-up Table register (QuadSPI1_LUT55)	32	R/W	0000_0000h	10.2.14.32/ 2707
30BB_03F0	Look-up Table register (QuadSPI1_LUT56)	32	R/W	0000_0000h	10.2.14.32/ 2707
30BB_03F4	Look-up Table register (QuadSPI1_LUT57)	32	R/W	0000_0000h	10.2.14.32/ 2707
30BB_03F8	Look-up Table register (QuadSPI1_LUT58)	32	R/W	0000_0000h	10.2.14.32/ 2707
30BB_03FC	Look-up Table register (QuadSPI1_LUT59)	32	R/W	0000_0000h	10.2.14.32/ 2707
30BB_0400	Look-up Table register (QuadSPI1_LUT60)	32	R/W	0000_0000h	10.2.14.32/ 2707
30BB_0404	Look-up Table register (QuadSPI1_LUT61)	32	R/W	0000_0000h	10.2.14.32/ 2707
30BB_0408	Look-up Table register (QuadSPI1_LUT62)	32	R/W	0000_0000h	10.2.14.32/ 2707
30BB_040C	Look-up Table register (QuadSPI1_LUT63)	32	R/W	0000_0000h	10.2.14.32/ 2707
30BB_4000	Module Configuration Register (QuadSPI2_MCR)	32	R/W	000F_4000h	10.2.14.1/ 2673
30BB_4008	IP Configuration Register (QuadSPI2_IPCR)	32	R/W	0000_0000h	10.2.14.2/ 2675
30BB_400C	Flash Configuration Register (QuadSPI2_FLSHCR)	32	R/W	0000_0303h	10.2.14.3/ 2676
30BB_4010	Buffer0 Configuration Register (QuadSPI2_BUF0CR)	32	R/W	0000_0000h	10.2.14.4/ 2677
30BB_4014	Buffer1 Configuration Register (QuadSPI2_BUF1CR)	32	R/W	0000_0000h	10.2.14.5/ 2678
30BB_4018	Buffer2 Configuration Register (QuadSPI2_BUF2CR)	32	R/W	0000_0000h	10.2.14.6/ 2679
30BB_401C	Buffer3 Configuration Register (QuadSPI2_BUF3CR)	32	R/W	See section	10.2.14.7/ 2680
30BB_4020	Buffer Generic Configuration Register (QuadSPI2_BFGENCR)	32	R/W	0000_0000h	10.2.14.8/ 2681
30BB_4030	Buffer0 Top Index Register (QuadSPI2_BUF0IND)	32	R/W	0000_0000h	10.2.14.9/ 2682
30BB_4034	Buffer1 Top Index Register (QuadSPI2_BUF1IND)	32	R/W	0000_0000h	10.2.14.10/ 2682

Table continues on the next page...

- *QSPI_SR[IP_ACC] = 0*
- *QSPI_SR[AHB_ACC] = 0*

Address: Base address + 180h 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 W	TPADA1																Reserved																
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

QuadSPIx_SFA1AD field descriptions

Field	Description
31–10 TPADA1	Top address for Serial Flash A1. In effect, TPADxx is the first location of the next memory.
Reserved	This field is reserved.

10.2.14.24 Serial Flash A2 Top Address (QuadSPIx_SFA2AD)

The QSPI_SFA2AD register provides the address mapping for the serial flash A2.The difference between QSPI_SFA2AD[TPADA2] and QSPI_SFA1AD[TPADA1] defines the size of the memory map for serial flash A2.

Write:

- *QSPI_SR[IP_ACC] = 0*
- *QSPI_SR[AHB_ACC] = 0*

Address: Base address + 184h 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 W	TPADA2																Reserved																
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

QuadSPIx_SFA2AD field descriptions

Field	Description
31–10 TPADA2	Top address for Serial Flash A2. In effect, TPxxAD is the first location of the next memory.
Reserved	This field is reserved.

10.2.14.25 Serial Flash B1Top Address (QuadSPIx_SFB1AD)

The QSPI_SFB1AD register provides the address mapping for the serial flash B1.The difference between QSPI_SFB1AD[TPADB1] and QSPI_SFA2AD[TPADA2] defines the size of the memory map for serial flash B1.

Write:

first time, so that that the hostwill stop at the block gap when the uSDHC controller gets VAULE1 blocks from the device. Also configure bits[3-0] to select the ack timeout value according to the SD clock frequency.

3. Software then needs to configure the Block Attributes Register to set block size and count. If in DDR fast boot mode, the block size only can be configured to 512 bytes. In DMA mode, it is recommended to set the block count (BLK_CNT) to the max value (16'hffff).
4. Software needs to configure Protocol Control Register to set DTW (data transfer width). If in DDR fast boot mode, the DTW only can be configured to 4-bit/8-bit dataline mode.
5. Software enbable ADMA2 by configuring Protocol Control Register bits [9-8].
6. Software need to set at least three pairs ADMA2 descriptor in boot memory (ie, in IRAM, at least 6 word). The first pair descriptor define the start address (ie, IRAM)and data length(ie,512byte*VALUE1) of fisrt part boot code. Software also need to set the second pair descriptor, the second start address (any value that is writeable), data length is suggest to set 1~2word(record as VALUE2). Note: the second couple desc also transfer useful data even at lease 1 word. Because our ADMA2 can't support 0 data_length data transfer descriptor.
7. Software needs to configure Command Argument Register to set argument to 0xFFFFFFFA in alternative fast boot, and don't need set in normal fast boot.
8. Software needs to configure Transfer Type Register to start the boot process . CMDINX, CMDTYP, RSPTYP, CICEN, CCCEN, AC12EN, BCEN and DMAEN are kept default value. DPSEL bit is set to 1, DTDSEL is set to 1, MSBSEL is set to 1. DMAEN is configured as 1 in DMA mode. And if BCEN is configured as 1, better to configure blk no in Bock Attributes Register to the max value. And if in ddr fast boot mode, DDR_EN need to be set to 1.
9. When the step 8 is configured, boot process will begin, the first VALUE1 block number data has transfer. Software need to polling TC bit (bit1 in Interrupt Status Register) to determine first transfer is end. Also software need to polling BGE bit (bit2 in Interrupt Status Register) to determine if first transfer stop at block gap.
10. When TC, BGE bit is 1, . SW can analyzes the first code of VALUE1 block, initializes the new memory device, if required, and sets the third pair of descriptors to define the start address and length of the remaining part of boot code(VALUE3 the remain boot code block). Remeber set the last descriptor with END.
11. Software needs to configure MMC Boot Register (offset 0xc4) again. Set bit 6 to 1(enable boot); and bit 5 to 0(normal fast boot), to 1(alternative boot); and bit 4 to select the ack mode or not. In DMA mode, configure bit 7 to 1 for enable automatically stop at block gap feature. Also configure bit31-bit16 to set the *(BLK_CNT - (VALUE1+1+VALUE3))*, that host will stop at block gap when *the uSDHC controller gets (VALUE1+1+VALUE3)) blocks from device totally include the blocks received in step 9*. And need to configure bit 3-bit0 to select the ack

ENET_RXIC*n* field descriptions (continued)

Field	Description
1 ICCS	Interrupt Coalescing Timer Clock Source Select 0 Use MII/GMII TX clocks. 1 Use ENET system clock.
2–3 Reserved	This field must be set to 0. This field is reserved.
4–11 ICFT	Interrupt coalescing frame count threshold This value determines the number of frames needed to be received for raising an interrupt. Frame counter restarts after reaching this threshold value or after the expiring of the coalescing timer. Must be greater than zero to avoid unpredictable behavior.
12–15 Reserved	This field must be set to 0. This field is reserved.
16–31 ICTT	Interrupt coalescing timer threshold Interrupt coalescing timer threshold in units of 64 clock periods. This value determines the maximum amount of time after receiving a frame before raising an interrupt. The threshold timer is disabled after expiring or number of frame reception defined by ICFT and starts again upon reception of the next first frame. Must be greater than zero to avoid unpredictable behavior.

11.1.5.16 Descriptor Individual Upper Address Register (ENET_IAUR)

IAUR contains the upper 32 bits of the 64-bit individual address hash table. The address recognition process uses this table to check for a possible match with the destination address (DA) field of receive frames with an individual DA. This register is not reset and you must initialize it.

Address: 30BE_0000h base + 118h offset = 30BE_0118h

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R W	IADDR1																															
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

ENET_IAUR field descriptions

Field	Description
0–31 IADDR1	Contains the upper 32 bits of the 64-bit hash table used in the address recognition process for receive frames with a unicast address. Bit 31 of IADDR1 contains hash index bit 63. Bit 0 of IADDR1 contains hash index bit 32.

11.2.4.2.1 Receive Parity Errors and Parity NACK Generation

The SIM receiver checks every byte received for proper parity. The IC control bit in the DATA_FORMAT register controls whether it checks for odd parity or even parity. When checking for odd parity, the number of logic ones contained in the 9 received bits (8 data bits and 1 parity bit) should be odd. Likewise, when checking for even parity, the number of logic ones contained in the 9 received bits (8 data bits and 1 parity bit) should be even.

If NACK generation on errors is disabled (ANACK = 0 in the CNTL register), the PORTx_PE bit will be set when the SIM receiver detects a parity error, the received data byte and its PORTx_PE bit are placed into FIFO. There is no need to clear the parity error flag in the FIFO. It is simply overwritten the next time a byte is received into that position of the FIFO. A parity error cannot cause an interrupt, and it is up to the software to discard data bytes with parity errors.

If NACK generation on errors is enabled (ANACK =1), the SIM can automatically request the SIM card to re-send a byte that has a parity error by generating a NACK pulse on the SIM XMT pin. In this case, Bytes with parity errors are discarded and will not placed into the FIFO.

To control NACK generation by the SIM receiver use the RTH[3:0], Receive NACK threshold, in the RCV_THRESHOLD register. This set of bits specify the number of consecutive NACK's generated by the SIM module on a received byte, before generating an RTE (receive threshold interrupt flag) in the RCV_STATUS register. The RTE flag would also force the SIM port to power down the card if the SAPD0,1 bit is set in the PORT0,1_CNTL register.

NOTE

The ANACK bit must be set in the CNTL register to enable this feature. The control bit ANACK is also used in initial character mode to enable the retransmission of initial characters in the event that an invalid initial character is received.

When a valid character has been received by the SIM, the internal counter keeping track of the number of NACK's transmitted on the current byte resets to zero. Clearing the receive threshold error (RTE) bit would also clear that counter.

When generating a NACK pulse, the SIM will generate the low pulse starting at 10.5 ETUs and lasting for 1 ETU.

11.2.4.2.2 Receive Frame Errors

The SIM receiver checks every byte received for a proper stop bit. A stop bit should exist during at least the first half of the 11th ETU time after the start of the character. If this is not true, a frame error is flagged. When a frame error is detected on a given byte, the

USBNC memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
30B2_043C	USBNC_OTG2_PHY_STATUS	32	R	0000_0000h	11.3.5.5/ 3282
30B3_0400	USBNC_HSIC_CTRL1	32	R/W	3000_1000h	11.3.5.1/ 3270
30B3_0404	USBNC_HSIC_CTRL2	32	R/W	0F00_0000h	11.3.5.2/ 3272
30B3_0440	USBNC_UH_HSICPHY_CFG1	32	R/W	C224_3A1Dh	11.3.5.9/ 3290

Limitation: If BYTE_PACKING_FORMAT [3:0] is 0xF, it indicates that the pixels are packed, that is, there are 4 pixels in 3 words or 12 bytes and H_COUNT must be a multiple of 4 pixels.

- YCBCR422_INPUT=1 implies that the input frame is in YCbCr 4:2:2 format. BYTE_PACKING_FORMAT must be 0xF.

  Limitation: LCD_DATABUS_WIDTH must be 8-bit and H_COUNT must be a multiple of 2 pixels.

  ODD_LINE_PATTERN and EVEN_LINE_PATTERN must be 0 when any of RGB_TO_YCBCR422_CSC or INTERLACE_FIELDS or YCBCR422_INPUT bits is 1.

After the RGB to RGB or RGB to YCbCr 4:2:2 color space conversions, there is one more opportunity to swizzle the data before sending it out to the display or the encoder. This can be done with the CSC_DATA_SWIZZLE field in the LCDIF_CTRL register, and it provides the same options as the INPUT_DATA_SWIZZLE register.

Finally, there is an option to shift the output data before sending it out to the display. This is done based on the SHIFT_DIR and SHIFT_NUM_BITS fields in LCDIF_CTRL register.

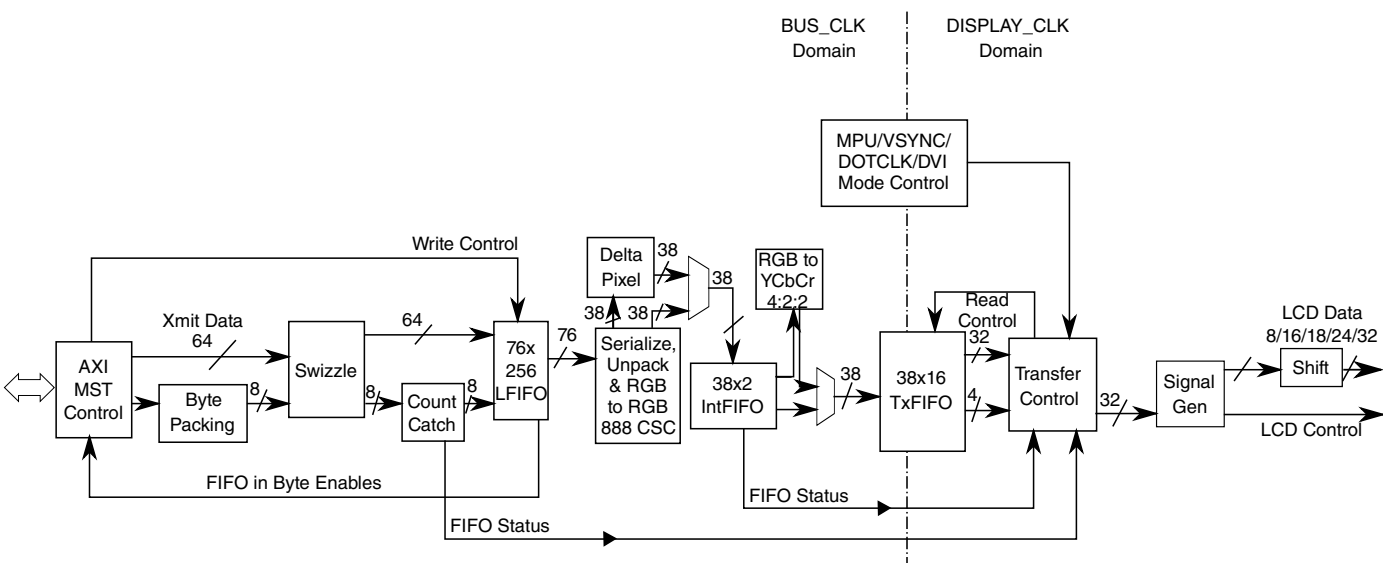

Figure 13-4. General Operations in Write Data Path

The examples in the following figures illustrate some different combinations of register programming for write mode. Assume that the data transferred over the system bus within a 32 bit word is organized as {A7-A0, B7-B0, C7-C0, D7-D0} in 8-bit mode and {A15-A0, B15-B0} in 16-bit mode.

MIPI_CSI2_CSIS_INT_MSK field descriptions (continued)

Field	Description
23–20 MSK_ FRAMEEND	FE packet is received, [CH3,CH2,CH1,CH0] 0 Disable (masked) 1 Enable (unmasked)
19–16 MSK_ERR_ SOT_HS	Start of transmission error [reserved, reserved, Lane1, Lane0] 0 Disable (masked) 1 Enable (unmasked)
15–12 MSK_ERR_ LOST_FS	Lost of Frame Start packet, [CH3,CH2,CH1,CH0] 0 Disable (masked) 1 Enable (unmasked)
11–8 MSK_ERR_ LOST_FE	Lost of Frame End packet, [CH3,CH2,CH1,CH0] 0 Disable (masked) 1 Enable (unmasked)
7–4 MSK_ERR_ OVER	Image FIFO overflow interrupt, [CH3,CH2,CH1,CH0] 0 Disable (masked) 1 Enable (unmasked)
3 MSK_ERR_ WRONG_CFG	Wrong configuration 0 Disable (masked) 1 Enable (unmasked)
2 MSK_ERR_ECC	ECC Error 0 Disable (masked) 1 Enable (unmasked)
1 MSK_ERR_CRC	CRC Error 0 Disable (masked) 1 Enable (unmasked)
0 MSK_ERR_id	Unknown ID Error 0 Disable (masked) 1 Enable (unmasked)

13.6.3.19 DIRECT_RGB454

DIRECT_RGB454 is used for a 16KB (8K pixel) RGB454 to RGB565 lookup.

Pixel formats that are in the YUV color space at the position of the LUT in the PXP data path can also be converted. In DIRECT_RGB454, the src_pixel is RGB/YUV[23;0] data is used to generate the lookup address. The address is generated as: {R/Y[23:20],G/U[15:11],B/V[7:4]}. In DIRECT_RGB454, the memory is pixel (2 byte) addressable.

13.6.3.20 CACHE_RGB565

The CACHE_RGB565 lookup is used for a 128KB (65K pixel) RGB/YUV565 to RGB/YUV565 lookup.

The 128KB memory requirement is too costly from an area perspective to implement a complete lookup table in the LUT's on chip memory. For this reason, a LRU (least recently used) 16KB 2-way set associative cache has been implemented to reference the full 128KB lookup table stored in external memory.

The 2-way set associative cache is organized in the following way:

- 16KB total data storage
- 512 entries split between 256 ways
- 6 pixels/entry cache line

Cache efficiency is very critical. For a cache miss, the PXP will be stalled until the cache line can be filled. For a 32 bit DDR memory interface, the latency can be calculated as follows:

cycles latency for read command through DDR controller + CAS Latency + # burst cycles for read data to be returned + # cycles latency for data through DDR controller to LUT + 1 cycle for cache access of new data

The src_pixel is RGB[23:0] data used to generate the lookup address. To improve the cache efficiency the RGB/YUV565 address and the lookup table must be formatted in the following way:

Address: A[15:0] = $R_7G_7G_6B_7$ $R_6G_5B_6$ $R_5G_4B_5$ $R_4G_3B_4$ $R_3G_2B_3$

The address is organized as follows:

- A[15:12] tag address, used to compare a hit between cache ways
- A[11:4] index address, used to select cache set (i.e. row of memory)
- A[3:0] block address, used to select Pixel in the cache line

Address: 3070_0000h base + 80h offset = 3070_0080h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	RSVD1								X							
W																
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	RSVD0								Y							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

PXP_HW_PXP_OUT_PS_LRC field descriptions

Field	Description
31–30 RSVD1	Reserved, always set to zero.
29–16 X	This field indicates the lower right X-coordinate (in pixels) of the processed surface (PS) in the output frame buffer.
15–14 RSVD0	Reserved, always set to zero.
Y	This field indicates the lower right Y-coordinate (in pixels) of the processed surface in the output frame buffer.

13.6.12.10 Alpha Surface Upper Left Coordinate (PXP_HW_PXP_OUT_AS_ULC)

This register contains the upper left location for the Alpha Surface in the output buffer.

This register contains the upper left coordinate of AS in the output frame buffer (in pixels). Values that are within the REG_OUT_LRC[X,Y] extents are valid. The lowest valid value for these fields is 0,0. Pixel locations that are greater than or equal to the upper left coordinates will use the AS to render pixels in the output buffer.

EXAMPLE

```
REG_OUT_AS_ULC_WR(0,0x0001_0001); // Alpha Surface upper left coordinate at (X,Y) = 1,1.
The AS surface will not effect pixels in the first row or first column of the output buffer.
```

Address: 3070_0000h base + 90h offset = 3070_0090h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	RSVD1								X							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15.4.4.6 Multiple Key Closures

Using the key press and Key release interrupts, the software can detect multiple keys or achieve n key rollover. The key scanning routine can be programmed accordingly.

See the following figures for illustrations of the interfacing of a 2-contact keypad matrix with the KPP controller. With proper enabling of row lines and the performing scan-routine, multiple key presses can be detected. When keys present on the same row are pressed, corresponding row lines (multiple lines) become low when the column is driven low during a scan-routine. By reading the data-register, pressed keys can be detected. Similarly, when keys present on same row line are pressed, the corresponding row line (only one line) becomes low when logic "0" is driven on the column line during a scan-routine.