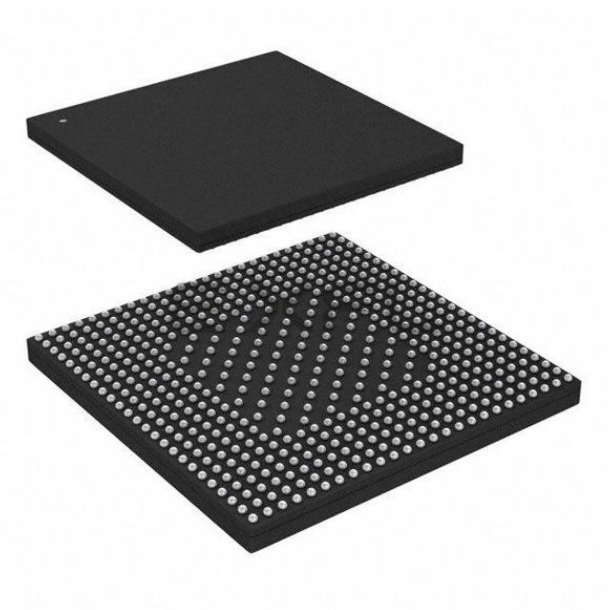

Welcome to E-XFL.COM

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	2 Core, 32-Bit
Speed	1.0GHz
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 (2)
SATA	-
USB	USB 2.0 + PHY (1), USB 2.0 OTG + PHY (2)
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	541-LFBGA
Supplier Device Package	541-MAPBGA (19x19)
Purchase URL	https://www.e-xfl.com/product-detail/nxp-semiconductors/mcimx7d5evm10sc

Section number	Title	Page

9.5.2.1 BCH Limitations and Assumptions........ 2403
9.5.2.2 Flash Page Layout........ 2404
9.5.2.3 Determining the ECC layout for a device........ 2406
9.5.2.3.1 4K+218 flash, 10 bytes metadata, 512 byte data blocks, separate metadata, Assuming GF(213)........ 2406
9.5.2.3.2 4K+128 flash, 10 bytes metadata, 1024 byte data blocks, separate metadata, assuming GF(213) for data and GF(214) for metadata........ 2407
9.5.2.4 Data Buffers in System Memory........ 2407
9.5.3 Memory to Memory (Loopback) Operation........ 2410
9.5.4 Programming the BCH/GPMI Interfaces........ 2411
9.5.4.1 BCH Encoding for NAND Writes........ 2411
9.5.4.1.1 DMA Structure Code Example........ 2414
9.5.4.1.2 Using the BCH Encoder........ 2419
9.5.4.2 BCH Decoding for NAND Reads........ 2420
9.5.4.2.1 DMA Structure Code Example........ 2424
9.5.4.2.2 Using the Decoder........ 2427
9.5.4.3 Interrupts........ 2429
9.5.4.4 Randomizer........ 2430
9.5.5 Behavior During Reset........ 2431
9.5.6 BCH Memory Map/Register Definition........ 2431
9.5.6.1 Hardware BCH ECC Accelerator Control Register (BCH_CTRL*n*)........ 2436
9.5.6.2 Hardware ECC Accelerator Status Register 0 (BCH_STATUS0*n*)........ 2438
9.5.6.3 Hardware ECC Accelerator Mode Register (BCH_MODE*n*)........ 2440
9.5.6.4 Hardware BCH ECC Loopback Encode Buffer Register (BCH_ENCODEPTR*n*)........ 2440
9.5.6.5 Hardware BCH ECC Loopback Data Buffer Register (BCH_DATAPTR*n*)........ 2441
9.5.6.6 Hardware BCH ECC Loopback Metadata Buffer Register (BCH_METAPTR*n*)........ 2441
9.5.6.7 Hardware ECC Accelerator Layout Select Register (BCH_LAYOUTSELECT*n*)........ 2442
9.5.6.8 Hardware BCH ECC Flash 0 Layout 0 Register (BCH_FLASH0LAYOUT0*n*)........ 2443
9.5.6.9 Hardware BCH ECC Flash 0 Layout 1 Register (BCH_FLASH0LAYOUT1*n*)........ 2445

4.2.1.1 Cortex-M4 Block Diagram

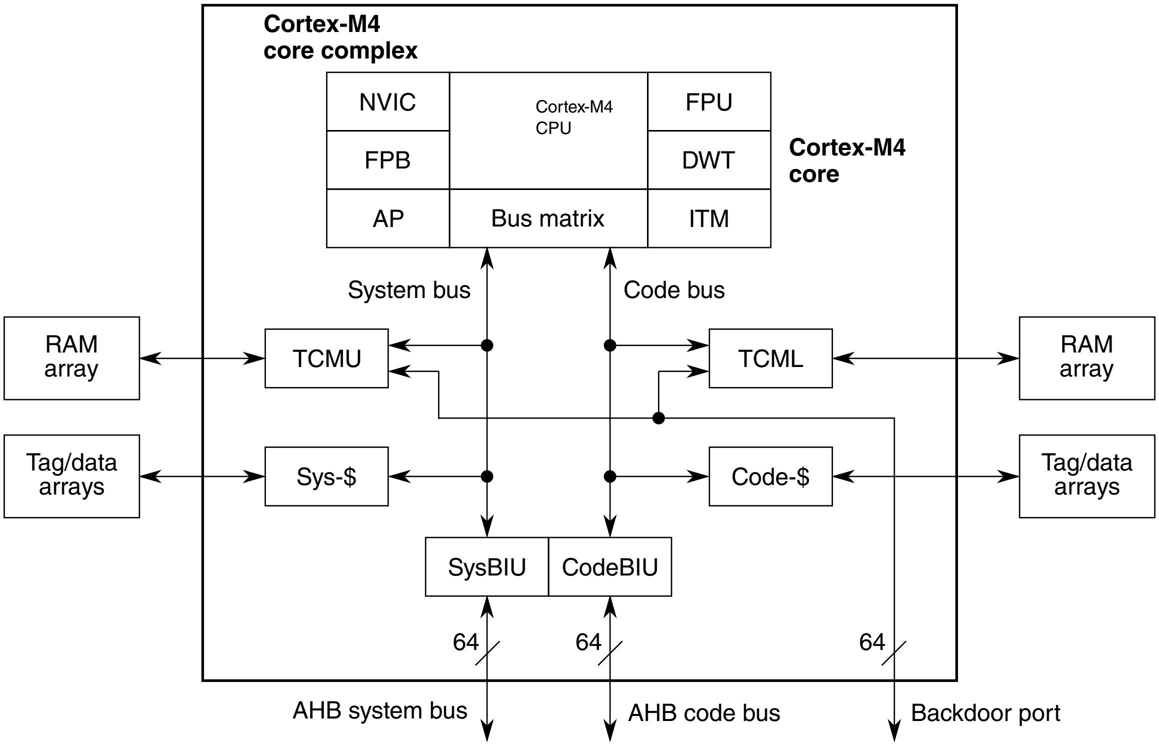

Figure 4-1. Cortex-M4 Block Diagram

Cortex-M4 core features a single issue, three stage pipeline microarchitecture. A high-level spatial pipeline block diagram of the CPU is shown below. The stages of the pipeline include:

- Fe - Instruction fetch stage where data is returned from instruction memory
- De - Instruction decode stage, generation of Load/Store Unit (LSU) address using forwarded register ports and immediate offset of LR register branch forwarding
- Ex - Instruction execute stage, single pipeline with multi-cycle stalls, LSU address/data pipelining to AHB interface, multiply/divide and ALU with branch result

A cache set command is initiated by setting the CCR[GO] bit. This bit also acts as a busy bit for set commands. It stays set while the command is active and is cleared by the hardware when the set command completes.

Supported cache set commands are given in the table below. Set commands work as follows:

- Invalidate – Unconditionally clear valid and modify bits of a cache entry.
- Push – Push a cache entry if it is valid and modified, then clear the modify bit. If entry not valid or not modified, leave as is.
- Clear – Push a cache entry if it is valid and modified, then clear the valid and modify bits. If entry not valid or not modified, clear the valid bit.

Table 4-8. Cache Set Commands

CCR[27:24]				Command
PUSH W1	INVW1	PUSH W0	INVW0	
0	0	0	0	NOP
0	0	0	1	Invalidate all way 0
0	0	1	0	Push all way 0
0	0	1	1	Clear all way 0
0	1	0	0	Invalidate all way 1
0	1	0	1	Invalidate all way 1; invalidate all way 0 (invalidate cache)
0	1	1	0	Invalidate all way 1; push all way 0
0	1	1	1	Invalidate all way 1; clear all way 0
1	0	0	0	Push all way 1
1	0	0	1	Push all way 1; invalidate all way 0
1	0	1	0	Push all way 1; push all way 0 (push cache)
1	0	1	1	Push all way 1; clear all way 0
1	1	0	0	Clear all way 1
1	1	0	1	Clear all way 1; invalidate all way 0
1	1	1	0	Clear all way 1; push all way 0
1	1	1	1	Clear all way 1; clear all way 0 (clear cache)

After a reset, complete an invalidate cache command before using the cache. It is possible to combine the cache invalidate command with the cache enable. That is, setting CCR to 0x8500_0003 will invalidate the cache and enable the cache and write buffer.

4.2.9.3.6.2 Cache line commands

Cache line commands operate on a single line in the cache at a time. Cache line commands can be performed using a physical or cache address.

Table 4-36. MDM-AP Register Description and Connectivity

Bit	Field	Connect to	Description
Status (dap_status)			
22	CTI trigger out	HUGO.event_dap_trigout	CTI interface trigger out
30	M4 DBGRESTARTED	CM4 integration	Indicate CM4 leaves debug mode
Control (dap_ctrl)			
24	CTI Trigger Input	HUGO.event_dap_trigin	Connects to CTI trigger input
25	CTI Trigger Out ACK	HUGO.eventack_dap_trigout	Connects to CTI trigger ACK input
Core halt request (dap_ctrl_halt_req)			
0	CM4 EDBGRQ	CM4 integration	The external debug request debug event which causing CM4 enter Debug state.
2	CA7 EDBGRQ[0]	CA7 integration	The external debug request debug event which causing CA7 CPU#0 enter Debug state.
3	CA7 EDBGRQ[1]	CA7 integration	The external debug request debug event which causing CA7 CPU#1 enter Debug state.
Core restart request (dap_ctrl_restart_req)			
0	CM4 DBGRESTART	CM4	CM4 debug restart input to CM4 core which causing CM4 leaving debug state.
2	CA7 DBGRESTART[0]	CA7 integration	CA7 debug restart input to CM4 core which causing CA7 CPU#0 leaving debug state.
3	CA7 DBGRESTART[1]	CA7 integration	CA7 debug restart input to CM4 core which causing CA7 CPU#1 leaving debug state.
Core halt ack (dap_status_halt_ack)			
0	CM4 HALTED	CM4 integration	Indicate CM4 has entered debug halted mode
2	CA7 DBGACK[0]	CA7 integration	Indicate CA7 CPU#0 has entered debug halted mode
3	CA7 DBGACK[1]	CA7 integration	Indicate CA7 CPU#1 has entered debug halted mode
Identification Register			
0	Type	N/A	4’h0 (MDM-AP)
1			
2			
3			
4	Varient		4’h3 (MX7)
5			
6			
7			
8	Reserved		Read as 8’h00
9			
10			
11			
12			
13			
14			

Table continues on the next page...

5.2.8.4 CCM PLL Control Register (CCM_PLL_CTRL*n*_CLR)

See Input Clocks for PLL control mapping.

NOTE

For the SoC to correctly power up after entering DSM, CCM_PLL_CTRLx must not be set to 0x0 or 0x3 for any domain in use.

Address: 3038_0000h base + 808h offset + (16d × i), where i=0d to 32d

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	Reserved	SETTING3		Reserved	Reserved	SETTING2		Reserved	Reserved	SETTING1		Reserved	Reserved	SETTING0	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

CCM_PLL_CTRL*n*_CLR field descriptions

Field	Description
31–16 -	This field is reserved. Reserved
15 -	This field is reserved. Reserved
14 -	This field is reserved. Reserved
13–12 SETTING3	Clock gate control setting for domain 3. This field can only be written by domain 3 00 Domain clocks not needed

Table continues on the next page...

OCOTP_TIMING field descriptions (continued)

Field	Description
PROG	This count value specifies the strobe period in one time write OTP. tPRW = (PROG - 1) / IPG_CLK_FREQ. It is given in number of IPG_CLK periods.

6.4.5.3 OTP Controller Write Data Register (OCOTP_DATA0)

This register associates with HW_OCOTP_CTRL to perform one-time writes to the OTP. Please see the "Software Write Sequence" section for operating details.

EXAMPLE

```
Empty Example.
```

Address: 3035_0000h base + 20h offset = 3035_0020h

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	DATA0																															
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

OCOTP_DATA0 field descriptions

Field	Description
DATA0	The program data for first fuse word in one 128 bits OTP. It is used to initiate a write to OTP. Please see the "Software Write Sequence" section for operating details.

6.4.5.4 OTP Controller Write Data Register (OCOTP_DATA1)

This register associates with HW_OCOTP_CTRL to perform one-time writes to the OTP. Please see the "Software Write Sequence" section for operating details.

EXAMPLE

```
Empty Example.
```

Address: 3035_0000h base + 30h offset = 3035_0030h

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	DATA1																															
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

OCOTP_DATA1 field descriptions

Field	Description
DATA1	The program data for second fuse word in one 128 bits OTP.

- *PDA (Peripheral Destination Address)* holds the destination byte address in the ARM platform memory map for writing data to this location. This register is automatically modified every time the core writes a new data into PD.
- *PS (Peripheral Setup)* contains the state of the peripheral DMA control, two configuration fields that define the way address registers are modified after every data access, two additional configuration fields that define the data size to access the source and destination devices, and another field that contains the latest transfer error status.

7.2.4.3.2.3 Peripheral DMA Data Transfers

There are three typical usages that involve the peripheral DMA, whether it is the data transfer start-point, endpoint, or both.

Every case requires a different procedure, as described in Data Retrieval from the ARM platform Memory or Peripheral, Storing Data into the ARM platform Memory or Peripheral, and Transferring Data Between Two ARM platform Memory Locations-Peripheral DMA Unit.

7.2.4.3.2.3.1 Data Retrieval from the ARM platform Memory or Peripheral

The following steps retrieve data from ARM platform memory using the peripheral DMA unit:

- Set up the PS fields to reflect the mode and data size for the source (incremented, decremented, or frozen address register; 8-bit, 16-bit, or 32-bit data transfers), then initialize the source address register itself (PSA) with an address that is aligned to the programmed data size.
- Read data from PD using the ldf PD instruction as many times as needed. If an error occurs during the fetch from the ARM platform memory or peripheral, the DMA control tags the error status on the data and the SDMA core SF flag is set when reading this data from PD.

7.2.4.3.2.3.2 Storing Data into the ARM platform Memory or Peripheral

The following steps store data to ARM platform memory using the peripheral DMA unit:

- Set up the PS fields to reflect the mode and data size for the destination (incremented, decremented, or frozen address register; 8-bit, 16-bit, or 32-bit data transfers), then initialize the destination address register itself (PDA) with an address that is aligned to the programmed data size.

IOMUXC_GPR_GPR0 field descriptions (continued)

Field	Description
4 DMAREQ_MUX_ SEL4	Selects between two possible sources for SDMA_EVENT47 0 ENET1 1588 Event1 out 1 ENET2 1588 Event1 out
3 DMAREQ_MUX_ SEL3	Selects between two possible sources for SDMA_EVENT21 0 I2C4 DMA event 1 SIM2 transmit DMA request
2 DMAREQ_MUX_ SEL2	Selects between two possible sources for SDMA_EVENT20 0 I2C3 DMA event 1 SIM1 receive DMA request
1 DMAREQ_MUX_ SEL1	Selects between two possible sources for SDMA_EVENT19 0 I2C2 DMA event 1 SIM1 transmit DMA request
0 DMAREQ_MUX_ SEL0	Selects between two possible sources for SDMA_EVENT18 0 I2C1 DMA event 1 SIM1 receive DMA request

8.2.4.23 GPR22 General Purpose Register (IOMUXC_GPR_GPR22)

GPR Register

Address: 3034_0000h base + 58h offset = 3034_0058h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PCIE_PHY_PLL_LOCKED	1	CHD1_ISO_ENA_1	CHD1_DVDD_STABLE	CHD2_ISO_ENA_1	CHD2_DVDD_STABLE	DFI_INIT_COMPLETE	DDRC_MRR_VALID	DDRC_MRR_DATA							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

IOMUXC_SW_PAD_CTL_PAD_GPIO1_IO15 field descriptions (continued)

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

8.2.7.159 SW_PAD_CTL_PAD_JTAG_MOD SW PAD Control Register (IOMUXC_SW_PAD_CTL_PAD_JTAG_MOD)

SW_PAD_CTL Register

Address: 3033_0000h base + 28Ch offset = 3033_028Ch

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	Reserved									PS		PE	HYS	SRE	DSE	
W	Reserved									PS		PE	HYS			
Reset	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	0

IOMUXC_SW_PAD_CTL_PAD_JTAG_MOD 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: JTAG_MOD 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: JTAG_MOD 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: JTAG_MOD 0 **HYS_0_Hysteresis_Disabled** — Hysteresis Disabled 1 **HYS_1_Hysteresis_Enabled** — Hysteresis Enabled

Table continues on the next page...

IOMUXC_SW_PAD_CTL_PAD_EPDC_DATA03 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_DATA03 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_DATA03 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_DATA03 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_DATA03 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: EPDC_DATA03 00 **DSE_0_X1** — X1 01 **DSE_1_X4** — X4 10 **DSE_2_X2** — X2 11 **DSE_3_X6** — X6

9.2.5.2.53 Scheduler Control Register 1 (DDRC_SCHED1)

Address: 307A_0000h base + 254h offset = 307A_0254h

Bit	31–8	7–0
R	Reserved	PAGECLOSE_TIMER
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

DDRC_SCHED1 field descriptions

Field	Description
31–8 Reserved	This field is reserved. Reserved for future use
PAGECLOSE_ TIMER	This field works in conjunction with SCHED.pageclose. It only has meaning if SCHED.pageclose == 1. If SCHED.pageclose == 1 and pageclose_timer == 0, then an auto-precharge may be scheduled for last read or write command in the CAM with a bank and page hit. Note, sometimes an explicit precharge is scheduled instead of the auto-precharge. See SCHED.pageclose for details of when this may happen. If SCHED.pageclose == 1 and pageclose_timer > 0, then an auto-precharge is not scheduled for last read or write command in the CAM with a bank and page hit. Instead, a timer is started, with pageclose_timer as the initial value. There is a timer on a per bank basis. The timer decrements unless the next read or write in the CAM to a bank is a page hit. It gets reset to pageclose_timer value if the next read or write in the CAM to a bank is a page hit. Once the timer has reached zero, an explcit precharge will be attempted to be scheduled. **Value After Reset**: 0x0 **Exists**: Always

9.2.5.2.54 High Priority Read CAM Register 1 (DDRC_PERFHPR1)

Address: 307A_0000h base + 25Ch offset = 307A_025Ch

Bit	31–24	23–16	15–0
R	HPR_XACT_RUN_LENGTH	Reserved	HPR_MAX_STARVE
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

DDRC_PERFHPR1 field descriptions

Field	Description
31–24 HPR_XACT_ RUN_LENGTH	Number of transactions that are serviced once the HPR queue goes critical is the smaller of: • This number • Number of transactions available. Unit: Transaction FOR PERFORMANCE ONLY. **Value After Reset**: 0xf

Table continues on the next page...

DDR_PHY_PHY_CON0 field descriptions (continued)

Field	Description
	2'b01 To update depending on "ctrl_flock" 2'b10 To update depending on "ctrl_flock" 2'b11 Do not update DLL
21–20 CTRL_UPD_ RANGE	It decides how many differences between the new lock value and the current lock value which is used in Slave-DLL is needed for updating lock value. Initial Value = 2'b00 2'b00 Update when difference is greater than 0 2'b01 Update when difference is greater than 1 2'b10 Update when difference is greater than 7 2'b11 Update when difference is greater than 15
19–17 Reserved	This field is reserved. Initial Value = 3'b001
16 WRLVL_MODE	Write Leveling Mode Enable Initial Value = 1'b0
15–13 Reserved	This field is reserved. Initial Value = 1'b0
12–11 CTRL_DDR_ MODE	Initial Value = 2'b11 2'b00 Reserved 2'b01 DDR3 2'b10 LPDDR2 2'b11 LPDDR3
10 Reserved	This field is reserved. Initial Value = 1'b1
9 CTRL_DFDQS	Initial Value = 1'b1 1'b0 Single-ended DQS — 1'b1 Differential DQS
8 CTRL_SHGATE	This field controls the gate control signal Initial Value = 1'b0 1'b0 Gate signal length = (burst length / 2) + N (DQS Pull-Down mode, ctrl_pulld_dqs[3:0] == 4'b1111, N = 0,1,2…) 1'b1 Gate signal length = (burst length / 2) – 1
7 Reserved	This field is reserved. Initial Value = 1'b0
6 CTRL_ATGATE	This bit should be set to 1'b1. Initial Value = 1'b1
5 Reserved	This field is reserved. Initial Value = 1'b0
4 CTRL_ CMOSRCV	This field controls the input mode of I/O. Initial Value = 1'b0

Table continues on the next page...

10.1.1.2 Modes and Operations

The ECSPI supports the modes described in the indicated sections:

- Master Mode
- Slave Mode
- Low Power Modes

As described in Operations, the ECSPI supports the operations described in the indicated sections:

- Typical Master Mode
    - Master Mode with SPI_RDY
    - Master Mode with Wait States
    - Master Mode with SS_CTL[3:0] Control
    - Master Mode with Phase Control
- Typical Slave Mode

10.1.2 External Signals

Table 10-1. eCSPI External Signals

Signal	Description	Pad	Mode	Direction
ECSPI1_MISO	Master data in; slave data out	ECSPI1_MISO	ALT0	IO
		UART3_RXD	ALT3	
ECSPI1_MOSI	Master data out; slave data in	ECSPI1_MOSI	ALT0	IO
		UART3_TXD	ALT3	
ECSPI1_RDY	SPI data ready signal	UART2_TXD	ALT3	I
ECSPI1_SCLK	SPI clock signal	ECSPI1_SCLK	ALT0	IO
		UART3_RTS	ALT3	
ECSPI1_SS0	Chip select signal	ECSPI1_SS0	ALT0	IO
		UART3_CTS	ALT3	
ECSPI1_SS1	Chip select signal	UART1_RXD	ALT3	IO
ECSPI1_SS2	Chip select signal	UART1_TXD	ALT3	IO
ECSPI1_SS3	Chip select signal	UART2_RXD	ALT3	IO
ECSPI2_MISO	Master data in; slave data out	ECSPI2_MISO	ALT0	IO
		ENET1_TDATA2	ALT2	
ECSPI2_MOSI	Master data out; slave data in	ECSPI2_MOSI	ALT0	IO
		ENET1_RDATA3	ALT2	
ECSPI2_RDY	SPI data ready signal	ENET1_TDATA1	ALT2	I
ECSPI2_SCLK	SPI clock signal	ECSPI2_SCLK	ALT0	IO
		ENET1_RDATA2	ALT2	
ECSPI2_SS0	Chip select signal	ECSPI2_SS0	ALT0	IO

Table continues on the next page...

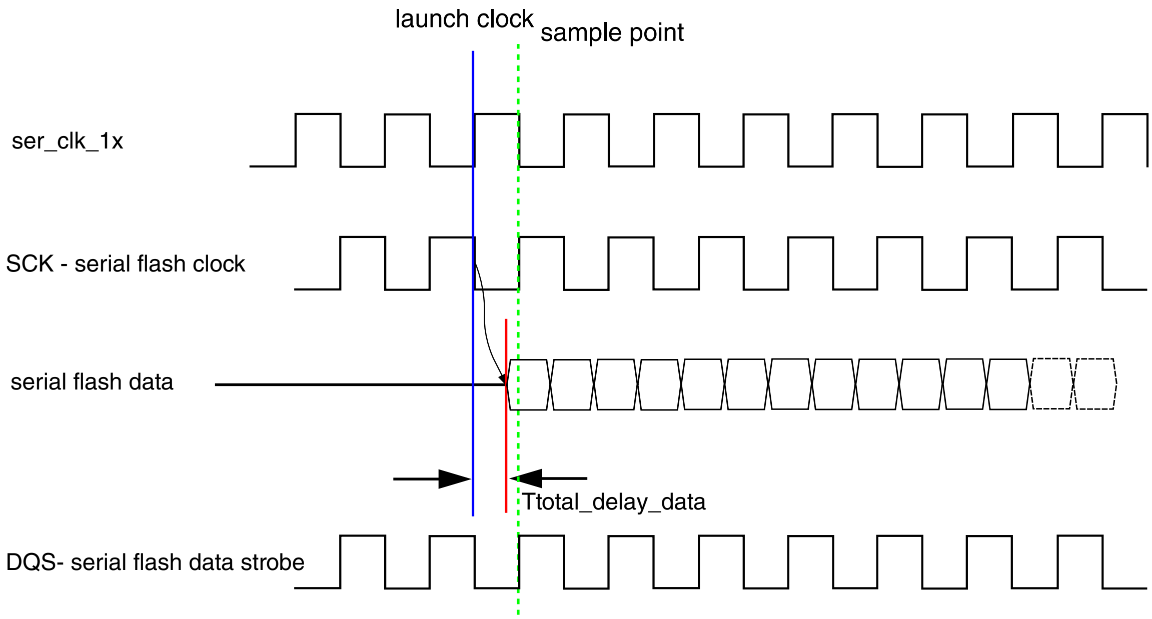

Figure 10-29. Input timing in DDR mode with loopback DQS sampling

In DDR mode and loopback sampling mode, DQS_PHASE_EN should be set 0.

For this sample point, the Setup requirement is: Tcycle >max(Ttotal_delay_data-Ttotal_delay_loopback_dqs). The Hold requirement is: min(Ttotal_delay_data-Ttotal_delay_loopback_dqs)> 0.

10.2.10.5 Input timing in SDR mode with flash DQS sampling

Input Timing diagram in SDR mode with internal sampling is show in the following figure.

priority. Based on this, the memory bandwidth required by the EPDC is a function of the refresh pixel rate which in turn is a function of resolution, frame-rate and blanking/active time.

The refresh bandwidth can be calculated as follows:

Active Time = Active time of the frame-scan time as a fraction (for example, 0.8)

Frame Rate = TFT frame refresh rate (Hz) (for example, 106 Hz)

Bandwidth (MB/s) = roundup4(EPDC_RES[HORIZONTAL]) x EPDC_RES[VERTICAL] x Frame Rate x (1/Active Time) x 2 x(1/1024^2)

For example, a panel with resolution 2048 x 1536 with 106 Hz refresh and estimated 80% active time, 20% blanking time, the refresh bandwidth is:

BW (MB/S) = 2048 x1536 x 106 x (1/0.8) x 2 x (1/1024^2) = 795 MB/s

(Note that 2048 mod 4 = 0.)

The WB requires 2 bytes per pixel.

Remaining bandwidth is used for other operations. Because update processing operations are not real-time (they do not necessarily have to occur at the refresh frame-rate), the time required to perform the update can be calculated as a function of the available bandwidth (actual bandwidth minus refresh bandwidth) and the update size.

13.6.3.11.2 Pixel Latency FIFO

The EPDC contains one working buffer latency FIFO which is used to load working buffer pixel data which is used by the TCE to perform the panel refresh operations.

The FIFO is sized at 1024 pixels each in order to provide significant system memory latency tolerance. The pixel FIFO is dedicated for the main screen and the second is in dual-scan cases for the second half of the screen.

Under no circumstance should the EPDC pixel FIFO reach an under-run condition. The EPDC provides an interrupt status bit to flag such a condition (TCE_UNDERRUN_IRQ). During development, this interrupt must be enabled. The pixel values stored in the FIFO are using by the TCE to perform look-up operations (from the LUTs) to generate TFT voltage control pixels. If an under-run occurs, the result will be unknown data being used as the source for the look-ups, which can damage the panel.

EPDC memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
306F_0328	EPDC Timing Control Engine Timing Register 3 (EPDC_TCE_TIMING3_CLR)	32	R/W	0000_0001h	13.6.4.32/ 3946
306F_032C	EPDC Timing Control Engine Timing Register 3 (EPDC_TCE_TIMING3_TOG)	32	R/W	0000_0001h	13.6.4.32/ 3946
306F_0380	EPDC Pigeon Mode Control Register 0 (EPDC_PIGEON_CTRL0)	32	R/W	0000_0000h	13.6.4.33/ 3947
306F_0384	EPDC Pigeon Mode Control Register 0 (EPDC_PIGEON_CTRL0_SET)	32	R/W	0000_0000h	13.6.4.33/ 3947
306F_0388	EPDC Pigeon Mode Control Register 0 (EPDC_PIGEON_CTRL0_CLR)	32	R/W	0000_0000h	13.6.4.33/ 3947
306F_038C	EPDC Pigeon Mode Control Register 0 (EPDC_PIGEON_CTRL0_TOG)	32	R/W	0000_0000h	13.6.4.33/ 3947
306F_0390	EPDC Pigeon Mode Control Register 1 (EPDC_PIGEON_CTRL1)	32	R/W	0000_0000h	13.6.4.34/ 3947
306F_0394	EPDC Pigeon Mode Control Register 1 (EPDC_PIGEON_CTRL1_SET)	32	R/W	0000_0000h	13.6.4.34/ 3947
306F_0398	EPDC Pigeon Mode Control Register 1 (EPDC_PIGEON_CTRL1_CLR)	32	R/W	0000_0000h	13.6.4.34/ 3947
306F_039C	EPDC Pigeon Mode Control Register 1 (EPDC_PIGEON_CTRL1_TOG)	32	R/W	0000_0000h	13.6.4.34/ 3947
306F_03C0	EPDC IRQ Mask Register for LUT 0~31 (EPDC_IRQ_MASK1)	32	R/W	0000_0000h	13.6.4.35/ 3948
306F_03C4	EPDC IRQ Mask Register for LUT 0~31 (EPDC_IRQ_MASK1_SET)	32	R/W	0000_0000h	13.6.4.35/ 3948
306F_03C8	EPDC IRQ Mask Register for LUT 0~31 (EPDC_IRQ_MASK1_CLR)	32	R/W	0000_0000h	13.6.4.35/ 3948
306F_03CC	EPDC IRQ Mask Register for LUT 0~31 (EPDC_IRQ_MASK1_TOG)	32	R/W	0000_0000h	13.6.4.35/ 3948
306F_03D0	EPDC IRQ Mask Register for LUT 32~63 (EPDC_IRQ_MASK2)	32	R/W	0000_0000h	13.6.4.36/ 3948
306F_03D4	EPDC IRQ Mask Register for LUT 32~63 (EPDC_IRQ_MASK2_SET)	32	R/W	0000_0000h	13.6.4.36/ 3948
306F_03D8	EPDC IRQ Mask Register for LUT 32~63 (EPDC_IRQ_MASK2_CLR)	32	R/W	0000_0000h	13.6.4.36/ 3948
306F_03DC	EPDC IRQ Mask Register for LUT 32~63 (EPDC_IRQ_MASK2_TOG)	32	R/W	0000_0000h	13.6.4.36/ 3948
306F_03E0	EPDC Interrupt Register for LUT 0~31 (EPDC_IRQ1)	32	R/W	0000_0000h	13.6.4.37/ 3949
306F_03E4	EPDC Interrupt Register for LUT 0~31 (EPDC_IRQ1_SET)	32	R/W	0000_0000h	13.6.4.37/ 3949
306F_03E8	EPDC Interrupt Register for LUT 0~31 (EPDC_IRQ1_CLR)	32	R/W	0000_0000h	13.6.4.37/ 3949
306F_03EC	EPDC Interrupt Register for LUT 0~31 (EPDC_IRQ1_TOG)	32	R/W	0000_0000h	13.6.4.37/ 3949

Table continues on the next page...

- Bypass mode which directly outputs the data to the downstream connected Fetch Engine after shift operation. This is done through a valid/ready data interface.
- Support for 8x8 and 16x16 block mode and scanline mode.
- Support for 8, 16 and 32 bpp input data format.

13.7.9.2 Top-level architecture

The PXP consist of several pipelined blocks that perform the video source frame scaling, color space conversion, alpha-blending/color key algorithm, secondary CSC, pixel correction , input and/or output rotation, dithering and waveform processing. It is also capable of fetching data from and storing data to memory.

The legacy blocks operate within the requirements of the legacy PXP architecture, and perform operations on either 8x8 or 16x16 pixel blocks in the representative source buffers. The legacy pipeline operate within the context of two iteration counters that iterate through the appropriate grid of input blocks to produce the rotated output grid blocks in scan-line order. The dither blocks and the waveform processing engines (WFE) will operate in a scan line format based on the active size. The input fetch and store engines can work on either on a block by block basis or in the scan line format.

Figure 13-87 shows the high-level architecture of the scaling, color space conversion, blending, pixel correction , rotation engines, dithering, waveform processing, histogram along with four sets of fetch and store engines. The Alpha Surface Engine fetches one RGB graphics plane alpha surface (AS). The scaling engine fetches a single processed surface (PS), which can be blended with the AS surface. The scaling engine also supports an alpha channel for the PS image. Although the legacy PXP processes NxN pixel macro blocks, each of the AS or PS surfaces can have any pixel alignment within the output buffer. There are no restrictions and any pixel coordinates within the output buffer are valid. The upper left origin of the output buffer is defined as pixel 0,0. The upper left and lower right coordinates for each of the AS and PS are inclusive within the output buffer.

Figure 13-88 represents a sample output buffer configuration with both an AS and PS included. The alignment of each AS and PS within the output buffer can be at any arbitrary pixel locations. For example, the PS has an upper left coordinate (ULC) of 2,2 and a lower right coordinate (LRC) at pixel 13,13. The maximum value for the ULC and LRC for each of the AS and PS is bounded by the LRC of the output buffer, 15,15 for this example.

13.7.12.194 Total Number of Pixels Used by Histogram Engine. (PXP_HW_PXP_HIST_B_TOTAL_PIXEL)

This register shows the total number of pixels used by histogram engine

Address: 3070_0000h base + 2AB0h offset = 3070_2AB0h

Bit	31–24	23–0
R	RSVD0	TOTAL_PIXEL
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

PXP_HW_PXP_HIST_B_TOTAL_PIXEL field descriptions

Field	Description
31–24 RSVD0	Reserved. This field always reads 0.
TOTAL_PIXEL	Total number of pixels used by histogram engine, the pixels got masked will be skipped

13.7.12.195 The X Coordinate Offset for Active Area. (PXP_HW_PXP_HIST_B_ACTIVE_AREA_X)

This register shows the minimal and maximum X coordinate offset for the active area in histogram processing

Address: 3070_0000h base + 2AC0h offset = 3070_2AC0h

Bit	31–28	27–16	15–12	11–0
R	RSVD1	MAX_X_OFFSET	RSVD0	MIN_X_OFFSET
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

PXP_HW_PXP_HIST_B_ACTIVE_AREA_X field descriptions

Field	Description
31–28 RSVD1	Reserved. This field always reads 0.
27–16 MAX_X_OFFSET	Maximum X coordinate offset for the active area in histogram processing
15–12 RSVD0	Reserved. This field always reads 0.
MIN_X_OFFSET	Minimul X coordinate offset for the active area in histogram processing

15.4.4.5 Glitch Suppression on Keypad Inputs

A glitch suppression circuit qualifies the keypad inputs to prevent noise from inadvertently interrupting the ARM platform. The circuit is a 4-state synchronizer clocked from a low frequency reference clock source.

This clock must continue to run in any low power mode where the keypad is a wake-up source, as the ARM platform interrupt is generated from the synchronized input. An interrupt is not generated until all four synchronizer stages have latched a valid key assertion. This guarantees the filtering out of any noise less than three clock periods in duration of a low frequency reference clock. Noise filtering of the duration between three to four clock periods cannot be guaranteed. The interrupt output is latched in an S-R latch and remains asserted until cleared by the software. The Set input of the latch is rising-edge clocked. See the figure below.

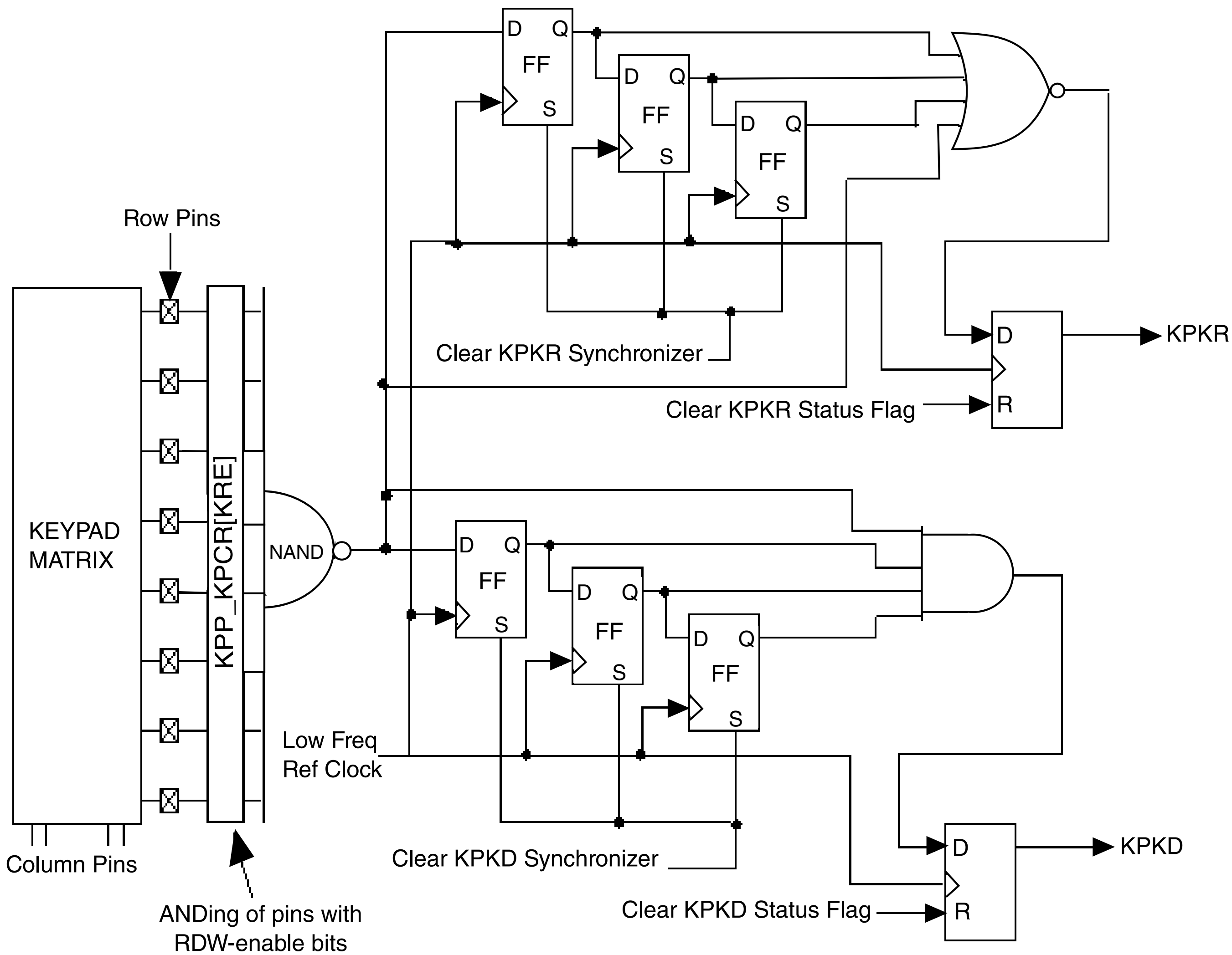

Figure 15-28. Keypad Synchronizer Functional Diagram