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	ARM926EJ-S
Number of Cores/Bus Width	1 Core, 32-Bit
Speed	454MHz
Co-Processors/DSP	Data; DCP
RAM Controllers	DRAM
Graphics Acceleration	No
Display & Interface Controllers	LCD, Touchscreen
Ethernet	-
SATA	-
USB	USB 2.0 + PHY (1)
Voltage - I/O	2.0V, 2.5V, 2.7V, 3.0V, 3.3V
Operating Temperature	-10°C ~ 70°C (TA)
Security Features	Cryptography, Hardware ID
Package / Case	128-LQFP
Supplier Device Package	128-LQFP (14x14)
Purchase URL	https://www.e-xfl.com/pro/item?MUrl=&PartUrl=mcimx233dag4b

Table 5-67. HW_ICOLL_INTERRUPT23 Bit Field Descriptions

BITS	LABEL	RW	RESET	DEFINITION
31:5	**RSRVD1**	RO	0x0	Always write zeroes to this bitfield.
4	**ENFIQ**	RW	0x0	Set this to 1 to steer this interrupt to the non-vectored FIQ line. When set to 0 the interrupt will pass through the main IRQ FSM and priority logic. DISABLE = 0x0 Disable ENABLE = 0x1 Enable
3	**SOFTIRQ**	RW	0x0	Set this bit to one to force a software interrupt. NO_INTERRUPT = 0x0 turn off the software interrupt request. FORCE_INTERRUPT = 0x1 force a software interrupt
2	**ENABLE**	RW	0x0	Enable the interrupt bit through the collector. DISABLE = 0x0 Disable ENABLE = 0x1 Enable
1:0	**PRIORITY**	RW	0x0	Set the priority level for this interrupt, 0x3 is highest, 0x0 is lowest (weakest). LEVEL0 = 0x0 level 0, lowest or weakest priority LEVEL1 = 0x1 level 1 LEVEL2 = 0x2 level 2 LEVEL3 = 0x3 level 3, highest or strongest priority

DESCRIPTION:

This register provides a mechanism to specify the priority associated with an interrupt bit. In addition, this register controls the enable and software generated interrupt. WARNING: Modifying the priority of an enabled interrupt may result in undefined behavior. You should always disable an interrupt prior to changing its priority.

EXAMPLE:

```
HW_ICOLL_INTERRUPT23_SET(0,0x00000001);
```

5.4.34 Interrupt Collector Interrupt Register 24 Description

This register provides a mechanism to specify the priority level for an interrupt source. It also provides an enable and software interrupt for each one, as well as security designation.

HW_ICOLL_INTERRUPT24	0x2A0
HW_ICOLL_INTERRUPT24_SET	0x2A4
HW_ICOLL_INTERRUPT24_CLR	0x2A8
HW_ICOLL_INTERRUPT24_TOG	0x2AC

Table 5-68. HW_ICOLL_INTERRUPT24

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
RSRVD1																											ENFIQ	SOFTIRQ	ENABLE	PRIORITY	

Table 5-97. HW_ICOLL_INTERRUPT38 Bit Field Descriptions

BITS	LABEL	RW	RESET	DEFINITION
31:5	**RSRVD1**	RO	0x0	Always write zeroes to this bitfield.
4	**ENFIQ**	RW	0x0	Set this to 1 to steer this interrupt to the non-vectored FIQ line. When set to 0 the interrupt will pass through the main IRQ FSM and priority logic. DISABLE = 0x0 Disable ENABLE = 0x1 Enable
3	**SOFTIRQ**	RW	0x0	Set this bit to one to force a software interrupt. NO_INTERRUPT = 0x0 turn off the software interrupt request. FORCE_INTERRUPT = 0x1 force a software interrupt
2	**ENABLE**	RW	0x0	Enable the interrupt bit through the collector. DISABLE = 0x0 Disable ENABLE = 0x1 Enable
1:0	**PRIORITY**	RW	0x0	Set the priority level for this interrupt, 0x3 is highest, 0x0 is lowest (weakest). LEVEL0 = 0x0 level 0, lowest or weakest priority LEVEL1 = 0x1 level 1 LEVEL2 = 0x2 level 2 LEVEL3 = 0x3 level 3, highest or strongest priority

DESCRIPTION:

This register provides a mechanism to specify the priority associated with an interrupt bit. In addition, this register controls the enable and software generated interrupt. WARNING: Modifying the priority of an enabled interrupt may result in undefined behavior. You should always disable an interrupt prior to changing its priority.

EXAMPLE:

```
HW_ICOLL_INTERRUPT38_SET(0,0x00000001);
```

5.4.49 Interrupt Collector Interrupt Register 39 Description

This register provides a mechanism to specify the priority level for an interrupt source. It also provides an enable and software interrupt for each one, as well as security designation.

HW_ICOLL_INTERRUPT39	0x390
HW_ICOLL_INTERRUPT39_SET	0x394
HW_ICOLL_INTERRUPT39_CLR	0x398
HW_ICOLL_INTERRUPT39_TOG	0x39C

Table 5-98. HW_ICOLL_INTERRUPT39

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
RSRVD1																											ENFIQ	SOFTIRQ	ENABLE	PRIORITY	

Table 5-185. HW_ICOLL_INTERRUPT82 Bit Field Descriptions

BITS	LABEL	RW	RESET	DEFINITION
31:5	**RSRVD1**	RO	0x0	Always write zeroes to this bitfield.
4	**ENFIQ**	RW	0x0	Set this to 1 to steer this interrupt to the non-vectored FIQ line. When set to 0 the interrupt will pass through the main IRQ FSM and priority logic. DISABLE = 0x0 Disable ENABLE = 0x1 Enable
3	**SOFTIRQ**	RW	0x0	Set this bit to one to force a software interrupt. NO_INTERRUPT = 0x0 turn off the software interrupt request. FORCE_INTERRUPT = 0x1 force a software interrupt
2	**ENABLE**	RW	0x0	Enable the interrupt bit through the collector. DISABLE = 0x0 Disable ENABLE = 0x1 Enable
1:0	**PRIORITY**	RW	0x0	Set the priority level for this interrupt, 0x3 is highest, 0x0 is lowest (weakest). LEVEL0 = 0x0 level 0, lowest or weakest priority LEVEL1 = 0x1 level 1 LEVEL2 = 0x2 level 2 LEVEL3 = 0x3 level 3, highest or strongest priority

DESCRIPTION:

This register provides a mechanism to specify the priority associated with an interrupt bit. In addition, this register controls the enable and software generated interrupt. WARNING: Modifying the priority of an enabled interrupt may result in undefined behavior. You should always disable an interrupt prior to changing its priority.

EXAMPLE:

```
HW_ICOLL_INTERRUPT82_SET(0,0x00000001);
```

5.4.93 Interrupt Collector Interrupt Register 83 Description

This register provides a mechanism to specify the priority level for an interrupt source. It also provides an enable and software interrupt for each one, as well as security designation.

HW_ICOLL_INTERRUPT83	0x650
HW_ICOLL_INTERRUPT83_SET	0x654
HW_ICOLL_INTERRUPT83_CLR	0x658
HW_ICOLL_INTERRUPT83_TOG	0x65C

Table 5-186. HW_ICOLL_INTERRUPT83

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
RSRVD1																											ENFIQ	SOFTIRQ	ENABLE	PRIORITY	

8.5.1 References

- *USB 2.0 Transceiver Macrocell Interface (UTMI) Specification*, Version 1.05, March 2001, Jon Lueker, Steve McGowan (Editor) Ken Oliver, Dean Warren. http://www.intel.com
- *VSI Alliance Virtual Component Interface Standard*, Version 2 (OCB 2 2.0), April 2001, On-Chip Bus Development Working Group. http://www.vsi.org
- *Universal Serial Bus Specification*, Revision 2.0, April 2000, Compaq, Hewlett-Packard, Intel, Lucent, Microsoft, NEC, Philips. http://www.usb.org
- *On-The-Go Supplement to the USB 2.0 Specification*, Revision 1.0, Dec 2001, On-The-Go Working Group of the USB-IF. http://www.usb.org
- *Enhanced Host Controller Interface Specification for Universal Serial Bus*, Revision 0.95, November 2000, Intel Corporation. http://www.intel.com
- *Universal Serial Bus Specification*, Revision 1.1, September 1998, Compaq, Intel, Microsoft, NEC. http://www.usb.org
- *AMBA Specification*, Revision 2.0, May 1999, ARM Limited. http://www.arm.com
- *UTMI+ Low Pin Interface (ULPI) Specification*, Revision 1.0 February 2004, ULPI Specification Organization. http://www.ulpi.org

8.6 Programmable Registers

This section includes the programmable registers supported in the USB high-speed Host controller core.

8.6.1 Identification Register Description

The Identification Register provides a simple way to determine if the USB-HS USB 2.0 core is provided in the system. The HW_USBCTRL_ID register identifies the USB-HS USB 2.0 core and its revision. The default value of this register is 0xE241FA05.

HW_USBCTRL_ID 0x000

Table 8-1. HW_USBCTRL_ID

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
CIVERSION			VERSION				REVISION				TAG					RSVD1		NID						RSVD0		ID						

Table 8-25. HW_USBCTRL_HCSPARAMS

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
RSVD2				N_TT				N_PTT				RSVD1			PI	N_CC				N_PCC				RSVD0			PPC	N_PORTS			

Table 8-26. HW_USBCTRL_HCSPARAMS Bit Field Descriptions

BITS	LABEL	RW	RESET	DEFINITION
31:28	**RSVD2**	RO	0x0	Reserved.
27:24	**N_TT**	RO	0x0	Number of Transaction Translators (N_TT). Indicates the number of embedded transaction translators associated with the USB2.0 host controller. This in a non-EHCI field to support embedded TT.
23:20	**N_PTT**	RO	0x0	Number of Ports per Transaction Translator (N_PTT). Indicates the number of ports assigned to each transaction translator within the USB2.0 host controller. This in a non-EHCI field to support embedded TT.
19:17	**RSVD1**	RO	0x0	Reserved.
16	**PI**	RO	0x1	Port Indicators (P INDICATOR). Indicates whether the ports support port indicator control. When set to 1, the port status and control registers include a read/writable field for controlling the state of the port indicator.
15:12	**N_CC**	RO	0x0	Number of Companion Controller (N_CC). Indicates the number of companion controllers associated with this USB2.0 host controller. A 0 in this field indicates there are no internal Companion Controllers. Port-ownership hand-off is not supported. A value larger than 0 in this field indicates there are companion USB host controller(s). Port-ownership hand-offs are supported. High- and Full-speed devices are supported on the host controller root ports.
11:8	**N_PCC**	RO	0x0	Number of Ports per Companion Controller. Indicates the number of ports supported per internal Companion Controller. It is used to indicate the port routing configuration to the system software. For example, if N_PORTS has a value of 6 and N_CC has a value of 2 then N_PCC could have a value of 3. The convention is that the first N_PCC ports are assumed to be routed to companion controller 1, the next N_PCC ports to companion controller 2, etc. In the previous example, the N_PCC could have been 4, where the first 4 are routed to companion controller 1 and the last two are routed to companion controller 2. The number in this field must be consistent with N_PORTS and N_CC.
7:5	**RSVD0**	RO	0x0	Reserved.

9.2.5.1 Switchable High-Speed 45Ω Termination Resistors

High-speed current mode differential signaling requires good 90Ω differential termination at each end of the USB cable. This results from switching in 45Ω terminating resistors from each signal line to ground at each end of the cable. Because each signal is parallel terminated with 45Ω at each end, each driver sees a 22.5Ω load. This is much too low of a load impedance for full-speed signaling levels—hence the need for switchable high-speed terminating resistors. Switchable trimming resistors are provided to tune the actual termination resistance of each device, as shown in Figure 9-3. The HW_USBPHY_TX_TXCAL45DP bit field, for example, allows one of 16 trimming resistor values to be placed in parallel with the 45Ω terminator on the USB_DP signal.

9.2.5.2 Full-Speed Differential Driver

The full-speed differential drivers are essentially “open drain” low-impedance pulldown devices that are switched in a differential mode for full-speed signaling, i.e., either one or the other device is turned on to signal the “J” state or the “K” state. These drivers are both turned on, simultaneously, for high-speed signaling. This has the effect of switching in both 45Ω terminating resistors. The tx_fs_hiz signal originates in the digital transmitter section. The hs_term signal that also controls these drivers comes from the UTMI.

9.2.5.3 High-Speed Differential Driver

The high-speed differential driver receives a 17.78-mA current from the constant current source and essentially steers it down either the USB_DP signal or the USB_DN signal or alternatively to ground. This current will produce approximately a 400-mV drop across the 22.5Ω termination seen by the driver when it is steered onto one of the signal lines. The approximately 17.78-mA current source is referenced back to the integrated voltage-band-gap circuit. The Iref, IBias, and V to I circuits are shared with the integrated battery charger.

9.2.5.4 Switchable 1.5KΩ USB_DP Pullup Resistor

The i.MX23 contains a switchable 1.5KΩ pullup resistor on the USB_DP signal. This resistor is switched on to tell the host/hub controller that a full-speed-capable device is on the USB cable, powered on, and ready. This resistor is switched off at power-on reset so the host does not recognize a USB device until processor software enables the announcement of a full-speed device.

9.2.5.5 Switchable 15KΩ USB_DP Pulldown Resistor

The i.MX23 contains a switchable 15KΩ pulldown resistor on both USB_DP and USB_DN signals. This is used in host mode to tell the device controller that a host is present.

Table 10-80. HW_APBH_CH4_DEBUG2

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
APB_BYTES																AHB_BYTES															

Table 10-81. HW_APBH_CH4_DEBUG2 Bit Field Descriptions

BITS	LABEL	RW	RESET	DEFINITION
31:16	**APB_BYTES**	RO	0x0	This value reflects the current number of APB bytes remaining to be transfered in the current transfer.
15:0	**AHB_BYTES**	RO	0x0	This value reflects the current number of AHB bytes remaining to be transfered in the current transfer.

DESCRIPTION:

This register allows debug visibility of the APBH DMA Channel 4.

EXAMPLE:

```
Empty example.
```

10.5.40 APBH DMA Channel 5 Current Command Address Register Description

The APBH DMA Channel 5 current command address register points to the multiword command that is currently being executed. Commands are threaded on the command address.

HW_APBH_CH5_CURCMDAR 0x270

Table 10-82. HW_APBH_CH5_CURCMDAR

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
CMD_ADDR																															

Table 10-83. HW_APBH_CH5_CURCMDAR Bit Field Descriptions

BITS	LABEL	RW	RESET	DEFINITION
31:0	**CMD_ADDR**	RO	0x00000000	Pointer to command structure currently being processed for channel 5.

DESCRIPTION:

APBH DMA Channel 5 is controlled by a variable sized command structure. This register points to the command structure currently being executed.

11.5.86 APBX DMA Channel 11 Buffer Address Register Description

The APBX DMA Channel 11 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.

HW_APBX_CH11_BAR 0x600

Table 11-174. HW_APBX_CH11_BAR

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
ADDRESS																															

Table 11-175. HW_APBX_CH11_BAR Bit Field Descriptions

BITS	LABEL	RW	RESET	DEFINITION
31:0	**ADDRESS**	RO	0x00000000	Address of system memory buffer to be read or written over the AHB bus.

DESCRIPTION:

This register holds a pointer to the data buffer in system memory. After the command values have been read into the DMA controller and the device controlled by this channel, then the DMA transfer will begin, to or from the buffer pointed to by this register.

EXAMPLE:

```
Empty example.
```

11.5.87 APBX DMA Channel 11 Semaphore Register Description

The APBX DMA Channel 11 semaphore register is used to synchronize between the CPU instruction stream and the DMA chain processing state.

HW_APBX_CH11_SEMA 0x610

Table 11-176. HW_APBX_CH11_SEMA

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
RSVD2								PHORE								RSVD1								INCREMENT_SEMA							

work packet pointer to the value in the NEXT_COMMAND_ADDRESS field at the end of the current work packet.

All channels have the same capabilities, but channel 0 is special in that it has a private interrupt line (dcp_vmi_irq). This allows software to use it for VMI (virtual memory) page-copy operations and have a dedicated interrupt vector to reduce latency. All other channels (and the color-space converter) share the other interrupt (dcp_irq).

16.2.5.2 Context Switching

The control logic maintains four virtual channels that allow the DCP block to time-multiplex encryption, hashing, and memcopy operations, it must also retain state information when changing channels so that when a channel is resumed, it can resume the operation from where it left off. This process is called context-switching.

To minimize the number of registers used in the design, the controller saves context information from each channel into a private context area in system memory. When initializing the DCP module, software must allocate memory for the context buffer and write the address into the Context Buffer Pointer register. As the DCP module processes packets, it saves the context information for each channel to the buffer after completing each control packet. When the channel is subsequently activated, the DCP module's internal registers are then reloaded with the proper context before resuming the operation.

Each channel reserves one-fourth the context buffer area for its context storage. The context buffer consumes 160 bytes of system memory and is formatted as shown in Table 16-1.

Table 16-1. DCP Context Buffer Layout

Range	Channel	Data	Size
0x00–0x0C	3	Cipher Context	16 bytes
0x10–0x24		Hash Context	24 bytes
0x28–0x34	2	Cipher Context	16 bytes
0x38–0x4C		Hash Context	24 bytes
0x50–0x5C	1	Cipher Context	16 bytes
0x60–0x74		Hash Context	24 bytes
0x78–0x84	0	Cipher Context	16 bytes
0x88–0x9C		Hash Context	24 bytes

The control logic writes to the context buffer only if the function is being used. This effectively means that the cipher context is stored for CBC encryption/decryption operations only, and the hash context is written only if SHA-1 is utilized. If neither of these modes are used for a given channel, the memory for the context buffer need not be allocated by software.

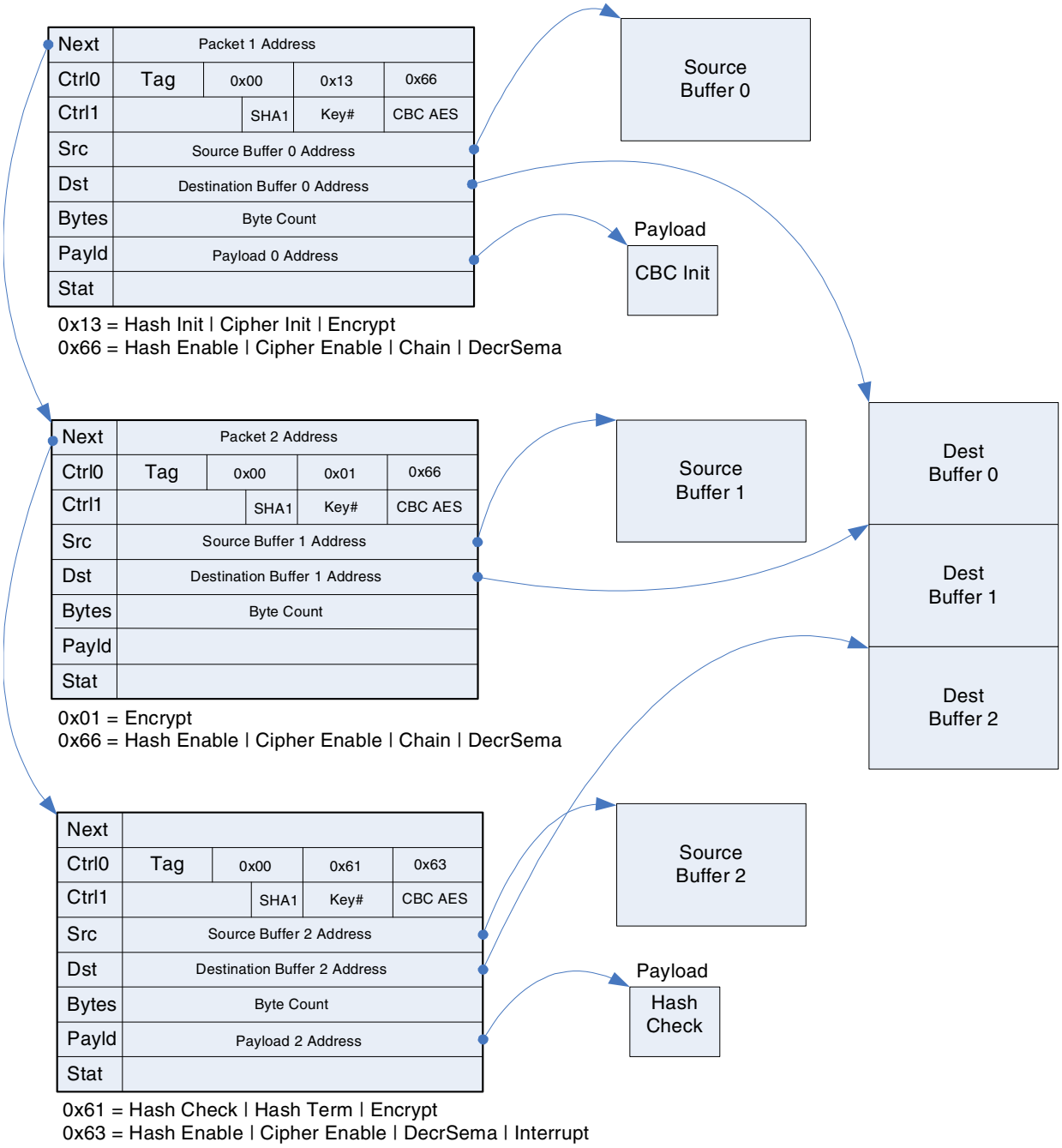


Figure 16-9. Multi-Buffer Scatter/Gather Cipher and Hash Operation

```
<code>

typedef struct _dcp_descriptor
{
    u32              *next;
    hw_dcp_packet1_t  ctrl0;
    hw_dcp_packet2_t  ctrl1;
    u32              *src,
                     *dst,
                      buf_size,
                     *payload,
                      stat;
}  DCP_DESCRIPTOR;
```

Table 21-30. HW_SSP_SDRESP3

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
RESP3																															

Table 21-31. HW_SSP_SDRESP3 Bit Field Descriptions

BITS	LABEL	RW	RESET	DEFINITION
31:0	**RESP3**	RO	0x0	SD/MMC Long Response [127:96]

DESCRIPTION:

SD/SDIO/MMC/CE_ATA Card Response Register 3.

EXAMPLE:

```
No Example.
```

21.10.13 SSP Status Register Description

SSP Read Only Status Registers.

HW_SSP_STATUS 0x0C0

Table 21-32. HW_SSP_STATUS

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
PRESENT	RSVD5	SD_PRESENT	CARD_DETECT	RSVD4						DMASENSE	DMATERM	DMAREQ	DMAEND	SDIO_IRQ	RESP_CRC_ERR	RESP_ERR	RESP_TIMEOUT	DATA_CRC_ERR	TIMEOUT	RECV_TIMEOUT_STAT	RSVD3	FIFO_OVRFLW	FIFO_FULL	RSVD2		FIFO_EMPTY	FIFO_UNDRFLW	CMD_BUSY	DATA_BUSY	RSVD1	BUSY

Table 21-33. HW_SSP_STATUS Bit Field Descriptions

BITS	LABEL	RW	RESET	DEFINITION
31	**PRESENT**	RO	0x1	SSP Present Bit. 0: SSP is not present in this product. 1: SSP is present.
30	**RSVD5**	RO	0x1	Program this field to 0x1.
29	**SD_PRESENT**	RO	0x1	SD/MMC Controller Present bit. 0: SD/MMC controller is not present in this product. 1: SD/MMC controller is present.
28	**CARD_DETECT**	RO	0x0	Reflects the state of the SSP_DETECT input pin.
27:22	**RSVD4**	RO	0x0	Reserved
21	**DMASENSE**	RO	0x0	Reflects the state of the ssp_dmasense output port. It indicates a DMA error (Timeout or CRC) when asserted high at the end of a DMA command.

Chapter 23
Real-Time Clock, Alarm, Watchdog, Persistent Bits

This chapter describes the real-time clock, alarm clock, watchdog reset, persistent bits, and millisecond counter included on the i.MX23. Programmable registers are described in Section 23.8, “Programmable Registers.”

23.1 Overview

The real-time clock (RTC) and alarm share a one-second pulse time domain. The watchdog reset and millisecond counter run on a one-millisecond time domain. The RTC, alarm, and persistent bits use persistent storage and reside in a special power domain (crystal domain) that remains powered up even when the rest of the chip is in its powered-down state. Figure 23-1 illustrates this block.

NOTE: The term *power-down*, as used here, refers to a state in which the DC-DC converter and various parts of the crystal power domain are still powered up, but the rest of the chip is powered down. If the battery is removed, then the persistent bits, the alarm value, and the second counter value will be lost. The *crystal power domain* powers both the 32-kHz and 24-MHz crystals. Note that the 32 kHz crystal is not available in the 128-pin LQFP package.

Upon battery insertion, the crystals (32-kHz and 24-MHz) are in a quiescent state. The activation of these crystals is under software control through the *RTC persistent bits*, as described later in this chapter.

- The XTAL32KHZ_PWRUP bit in the Persistent Register 0 controls the activity of the 32-kHz crystal at all times (chip power on or off).
- The XTAL24MHZ_PWRUP bit in the Persistent Register 0 controls the behavior of the 24-MHz crystal during the power-off state. (The 24-MHz crystal is always on when the chip is powered up.)

The one-second time base is derived either from the 24.0-MHz crystal oscillator or a 32-kHz crystal oscillator (which can be either exactly 32.0 kHz or 32.768 kHz), as controlled by bits in Persistent Register 0. The time base thus generated is used to increment the value of the persistent seconds count register. Like the values of the other persistent registers, the value of the persistent seconds count register is not lost across a power-down state and will continue to count seconds during that time.

Contrary to the one-second time base, no record or count is made of the one-millisecond time base in the crystal power domain. The one-millisecond time base is always derived from the 24.0-MHz crystal oscillator, but is not available when the chip is powered down.

The real-time clock seconds counter, alarm functions, and persistent bit storage are kept in the (always on) crystal power domain. Shadow versions of these values are maintained in the CPU’s power and

Figure 28-1 shows a high-level block diagram of the AUDIOIN module. See Figure 1-4 for a diagram of the audio path and control options.

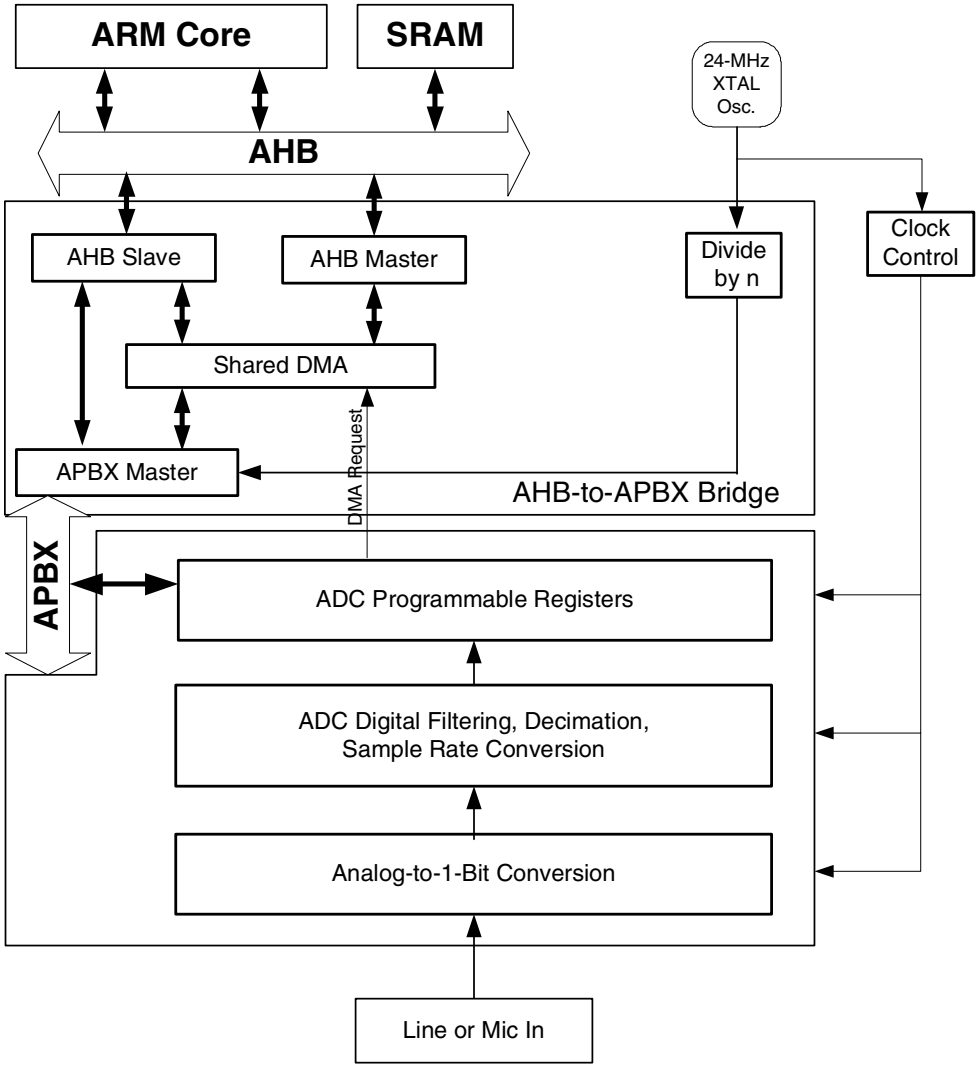


Figure 28-1. AUDIOIN/ADC Block Diagram

28.2 Operation

The first step in receiving audio to the AUDIOIN module requires the analog-to-digital converter (ADC). The i.MX23 includes a high-performance analog stereo sigma-delta ADC. It converts analog audio to two (left and right channel) single-bit digital streams that are input to the AUDIOIN module, along with a clock that runs at the sigma-delta oversampling clock rate. The AUDIOIN module includes hardware for oversampling, decimation, and arbitrary sample rate conversion. The 1-bit stream is input to a cascaded-integrator comb filter where serial-to-parallel data conversion, as well as sample rate conversion, takes place, along with a high-pass filter to eliminate DC offset. Serial audio is first input to an averager that initially converts samples to 8-bit values. The CIC then interpolates/decimates as well as sign-extends the parallel data, converting the samples from the programmed standard external sample

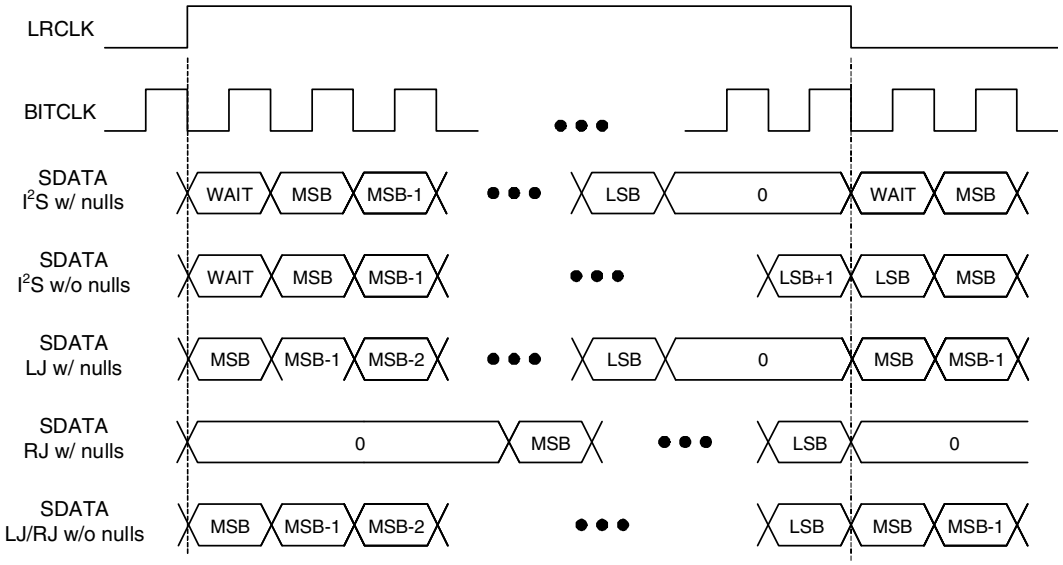
```
Error();
```

SPDIF Block v1.1, Revision 1.26

The bits to program frame format reside within the HW_SAIF_CTRL register.

31.2.7 Pin Timing

Figure 31-2 shows the six basic frame formats supported by the SAIF. Keep in mind that for 16-bit operation, BITCLK runs at either 32x or 48x the sample rate, and for 17-bit through 24-bit operation, it runs at either 48x or 64x the sample rate (i.e., clock frequency relationship of differing data sizes is not shown here).



Note: LRCLK_POLARITY=1 and BITCLK_EDGE=0 for this example.

Figure 31-2. Frame Formats Supported by SAIF

31.3 Programmable Registers

The following registers are available for CPU programmer access and control of the serial audio interface.

31.3.1 SAIF Control Register Description

The SAIF Control Register controls the frame format and operation of the three-wire serial audio interface.

HW_SAIF_CTRL	0x000
HW_SAIF_CTRL_SET	0x004
HW_SAIF_CTRL_CLR	0x008
HW_SAIF_CTRL_TOG	0x00C

On boot up, the hardware defaults to the VDD5V_GT_VDDIO 5V detection method. Even though it is the default, VDD5V_GT_VDDIO is only intended as the chip startup 5V detection method. Firmware should immediately switch to the VBUSVALID detection method to avoid some potential issues.

First, the VDD5V_GT_VDDIO method depends on a fixed offset between VDDIO and the VDD5V voltage. So the VDD5V_GT_VDDIO status will change as the VDDIO level changes.

Second, as with any linear regulator, the VDDIO linear regulator supply strength is based upon its supply voltage which is VDD5V. As the VDD5V level decreases, the VDDIO supply strength decreases. So if the VDDIO rail has too much load and the VDD5V level decreases, the VDDIO output voltage can decrease as well. If the VDDIO voltage drops with VDD5V, then a scenario arises where VDD5V_GT_VDDIO never becomes false. One method for countering this scenario is to enable and properly handle VDDIO brownouts. But this brings other complications such as causing the device to shutdown on a 5V disconnect when it could have performed a 5V-to-battery transfer, or having to add functionality to the VDDIO interrupt handler to determine whether a brownout was caused by a 5V disconnect.

To avoid these issues with the VDD5V_GT_VDDIO functionality, the hardware should be programmed to use the VBUSVALID functionality for detecting 5V by setting VBUSVALID_5VDETECT. But, great care must be taken to coordinate this change properly. Remember the PWD_BATTBRNOUT functionality always uses VDD5V_GT_VDDIO as an internal enable/disable mechanism. A problem can occur when the 5V detection is changed from VDD5V_GT_VDDIO to VBUSVALID if the VDD5V_GT_VDDIO status is true but VBUSVALID status is false due to the VBUSVALID_TRSH level selected. In this case, VDD5V level is higher than the set threshold. A false VBUSVALID will register as a 5V disconnection. A hardware 5V disconnection event causes the DCDC to automatically turn on regardless of the battery voltage level. This could result in the DCDC operating from a voltage lower than the minimum operating voltage specified in the Characteristics and Specifications section and physical damage to the DCDC converter itself. To avoid this situation, follow these rules.

- Before switching to VBUSVALID detection method, ensure the VDD5V voltage level is higher than the VBUSVALID_TRSH level.
- When raising the VBUSVALID_TRSH from one level to another, be sure that the VDD5V is higher than the target threshold level being programmed.

In addition to the power supply 5V detection methods, VDD5V voltage measurements can be taken by the LRADC peripheral.

32.9.5 DCDC Input Protection

The DCDC converter must only be allowed to operate within the valid operational range provided in the "Characteristics and Specifications" section of the datasheet. Failure to do so could result in damage to the DCDC converter.

EXAMPLE:

```
BW_LRADC_CTRL3_HIGH_TIME(BV_LRADC_CTRL3_HIGH_TIME__83NS);
BW_LRADC_CTRL3_INVERT_CLOCK(BV_LRADC_CTRL3_INVERT_CLOCK__NORMAL);
```

33.4.5 LRADC Status Register Description

The LRADC status register returns various read-only status bit field values.

HW_LRADC_STATUS	0x040
HW_LRADC_STATUS_SET	0x044
HW_LRADC_STATUS_CLR	0x048
HW_LRADC_STATUS_TOG	0x04C

Table 33-9. HW_LRADC_STATUS

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
RSRVD3					TEMP1_PRESENT	TEMP0_PRESENT	TOUCH_PANEL_PRESENT	CHANNEL7_PRESENT	CHANNEL6_PRESENT	CHANNEL5_PRESENT	CHANNEL4_PRESENT	CHANNEL3_PRESENT	CHANNEL2_PRESENT	CHANNEL1_PRESENT	CHANNEL0_PRESENT	RSRVD2																TOUCH_DETECT_RAW

Table 33-10. HW_LRADC_STATUS Bit Field Descriptions

BITS	LABEL	RW	RESET	DEFINITION
31:27	**RSRVD3**	RO	0x0	Reserved
26	**TEMP1_PRESENT**	RO	0x1	This read-only bit returns a one when the temperature sensor 1 current source is present on the chip.
25	**TEMP0_PRESENT**	RO	0x1	This read-only bit returns a one when the temperature sensor 0 current source is present on the chip.
24	**TOUCH_PANEL_PRESENT**	RO	0x1	This read-only bit returns a one when the touch panel controller function is present on the chip.
23	**CHANNEL7_PRESENT**	RO	0x1	This read-only bit returns a one when the LRADC channel 7 converter function is present on the chip.
22	**CHANNEL6_PRESENT**	RO	0x1	This read-only bit returns a one when the LRADC channel 6 converter function is present on the chip.
21	**CHANNEL5_PRESENT**	RO	0x1	This read-only bit returns a one when the LRADC channel 5 converter function is present on the chip.
20	**CHANNEL4_PRESENT**	RO	0x1	This read-only bit returns a one when the LRADC channel 4 converter function is present on the chip.
19	**CHANNEL3_PRESENT**	RO	0x1	This read-only bit returns a one when the LRADC channel 3 converter function is present on the chip.
18	**CHANNEL2_PRESENT**	RO	0x1	This read-only bit returns a one when the LRADC channel 2 converter function is present on the chip.
17	**CHANNEL1_PRESENT**	RO	0x1	This read-only bit returns a one when the LRADC channel 1 converter function is present on the chip.
16	**CHANNEL0_PRESENT**	RO	0x1	This read-only bit returns a one when the LRADC channel 0 converter function is present on the chip.

Table 35-11. Media Config Block Parameters

Field	Value
Signature	0x4D454D53
Version	0x00000001
Tag	0x00000050

35.7.2 Master Boot Record (MBR)

The first block of the device contains the master boot record (MBR). The MBR is identified by its signature located at offset 0x1FE of the first sector. The partition table is stored at address 0x1BE. The Freescale partition is identified by MBR_SIGMATEL_ID at an offset 0x04 from partition table. The boot image address is stored at offset 0x08 of partition table and size of the image at offset 0x10 of partition table.

Table 35-12. MBR Signature Bits

Field	Value
MBR Signature	0x55AA
MBR_SIGMATEL_ID	‘S’

35.7.3 Device Identification

SD/MMC boot mode uses the identification processes specified in SD Specifications, Part 1, Physical Layer Specifications, Version 2.0 Draft and MultiMediaCard System Specification, Version 4.2.

35.8 NAND Boot Mode

35.8.1 NAND Control Block (NCB)

As part of the NAND media initialization, the ROM driver uses safe NAND timings to search for a NAND Control Block (NCB) that contains the optimum NAND timings and the Factory Marked Bad Block Table. A flowchart is shown in Figure 35-3.

In the i.MX23, there are no separate boot modes for each type of ECC level. The hardware ECC level to use is embedded inside NCB block. The NCB data structure is itself protected using software ECC (SEC-DED Hamming Codes). Driver reads raw 2112 bytes of first sector and runs through software ECC engine that determines whether NCB data is valid or not.

If the NCB is found, the optimum NAND timings are loaded for further reads. If the software ECC fails, or the fingerprints do not match, the Block Search state machine increments 64 pages and reads that page

EXAMPLE:

```
Empty Example.
```

37.4.2 PINCTRL Pin Mux Select Register 0 Description

The PINCTRL Pin Mux Select Register provides pin function selection for 16 pins in bank0.

HW_PINCTRL_MUXSEL0	0x100
HW_PINCTRL_MUXSEL0_SET	0x104
HW_PINCTRL_MUXSEL0_CLR	0x108
HW_PINCTRL_MUXSEL0_TOG	0x10C

Table 37-7. HW_PINCTRL_MUXSEL0

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
BANK0_PIN15		BANK0_PIN14		BANK0_PIN13		BANK0_PIN12		BANK0_PIN11		BANK0_PIN10		BANK0_PIN09		BANK0_PIN08		BANK0_PIN07		BANK0_PIN06		BANK0_PIN05		BANK0_PIN04		BANK0_PIN03		BANK0_PIN02		BANK0_PIN01		BANK0_PIN00	

Table 37-8. HW_PINCTRL_MUXSEL0 Bit Field Descriptions

BITS	LABEL	RW	RESET	DEFINITION
31:30	**BANK0_PIN15**	RW	0x3	Pin 59, GPMI_D15 pin function selection: 00= gpmi_data15; 01= auart2_tx; 10= gpmi_ce3n; 11= GPIO.
29:28	**BANK0_PIN14**	RW	0x3	Pin 58, GPMI_D14 pin function selection: 00= gpmi_data14; 01= auart2_rx; 10= reserved; 11= GPIO.
27:26	**BANK0_PIN13**	RW	0x3	Pin 57, GPMI_D13 pin function selection: 00= gpmi_data13; 01= lcd_d23; 10= reserved; 11= GPIO.
25:24	**BANK0_PIN12**	RW	0x3	Pin 56, GPMI_D12 pin function selection: 00= gpmi_data12; 01= lcd_d22; 10= reserved; 11= GPIO.
23:22	**BANK0_PIN11**	RW	0x3	Pin 55, GPMI_D11 pin function selection: 00= gpmi_data11; 01= lcd_d21; 10= ssp1_d7; 11= GPIO.

Table 37-15. HW_PINCTRL_MUXSEL4

31 30	29 28	27 26	25 24	23 22	21 20	19 18	17 16	15 14	13 12	11 10	09 08	07 06	05 04	03 02	01 00
BANK2_PIN15	BANK2_PIN14	BANK2_PIN13	BANK2_PIN12	BANK2_PIN11	BANK2_PIN10	BANK2_PIN09	BANK2_PIN08	BANK2_PIN07	BANK2_PIN06	BANK2_PIN05	BANK2_PIN04	BANK2_PIN03	BANK2_PIN02	BANK2_PIN01	BANK2_PIN00

Table 37-16. HW_PINCTRL_MUXSEL4 Bit Field Descriptions

BITS	LABEL	RW	RESET	DEFINITION
31:30	**BANK2_PIN15**	RW	0x3	Pin 108, EMI_A06 pin function selection: 00= emi_addr06; 01= reserved; 10= reserved; 11= GPIO.
29:28	**BANK2_PIN14**	RW	0x3	Pin 107, EMI_A05 pin function selection: 00= emi_addr05; 01= reserved; 10= reserved; 11= GPIO.
27:26	**BANK2_PIN13**	RW	0x3	Pin 109, EMI_A04 pin function selection: 00= emi_addr04; 01= reserved; 10= reserved; 11= GPIO.
25:24	**BANK2_PIN12**	RW	0x3	Pin 110, EMI_A03 pin function selection: 00= emi_addr03; 01= reserved; 10= reserved; 11= GPIO.
23:22	**BANK2_PIN11**	RW	0x3	Pin 111, EMI_A02 pin function selection: 00= emi_addr02; 01= reserved; 10= reserved; 11= GPIO.
21:20	**BANK2_PIN10**	RW	0x3	Pin 112, EMI_A01 pin function selection: 00= emi_addr01; 01= reserved; 10= reserved; 11= GPIO.
19:18	**BANK2_PIN09**	RW	0x3	Pin 113, EMI_A00 pin function selection: 00= emi_addr00; 01= reserved; 10= reserved; 11= GPIO.
17:16	**BANK2_PIN08**	RW	0x3	Pin 38, ROTARYB pin function selection: 00= timrot2; 01= auart2_cts; 10= gpmi_ce3n; 11= GPIO.