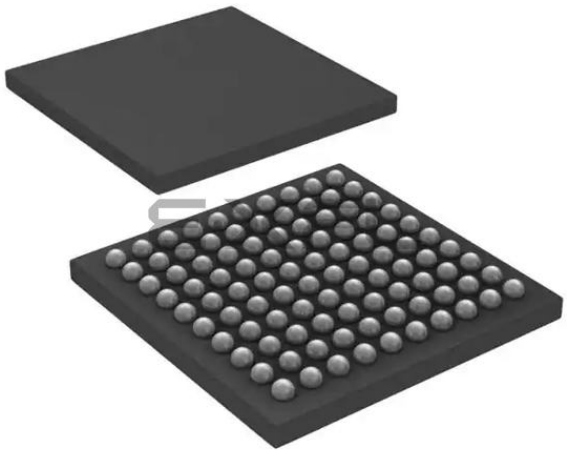


Welcome to E-XFL.COM

What is "Embedded - Microcontrollers"?

"Embedded - Microcontrollers" refer to small, integrated circuits designed to perform specific tasks within larger systems. These microcontrollers are essentially compact computers on a single chip, containing a processor core, memory, and programmable input/output peripherals. They are called "embedded" because they are embedded within electronic devices to control various functions, rather than serving as standalone computers. Microcontrollers are crucial in modern electronics, providing the intelligence and control needed for a wide range of applications.

Applications of "Embedded - Microcontrollers"

Details	
Product Status	Active
Core Processor	ARM® Cortex®-M7
Core Size	32-Bit Single-Core
Speed	300MHz
Connectivity	I²C, IrDA, LINbus, MMC/SD/SDIO, QSPI, SPI, UART/USART, USB
Peripherals	Brown-out Detect/Reset, DMA, I²S, POR, PWM, WDT
Number of I/O	75
Program Memory Size	2MB (2M x 8)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	384K x 8
Voltage - Supply (Vcc/Vdd)	1.08V ~ 3.6V
Data Converters	A/D 10x12b; D/A 2x12b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 105°C (TA)
Mounting Type	Surface Mount
Package / Case	100-VFBGA
Supplier Device Package	100-VFBGA (7x7)
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/atsams70n21a-cfnt

9. Interconnect

The system architecture is based on the ARM Cortex-M7 processor connected to the main AHB Bus Matrix, the embedded Flash, the multi-port SRAM and the ROM.

The 32-bit AHBP interface is a single 32-bit wide interface that accesses the peripherals connected on the main Bus Matrix. It is used only for data access. Instruction fetches are never performed on the AHBP interface. The bus, AHBP or AXIM, accessing the peripheral memory area [0x40000000 to 0x60000000] is selected in the AHBP control register.

The 32-bit AHBS interface provides system access to the ITCM, D1TCM, and D0TCM. It is connected on the main Bus Matrix and allows the XDMA to transfer from memory or peripherals to the instruction or data TCMs.

The 64-bit AXIM interface is a single 64-bit wide interface connected through two ports of the AXI Bridge to the main AHB Bus Matrix and to two ports of the multi-port SRAM. The AXIM interface allows:

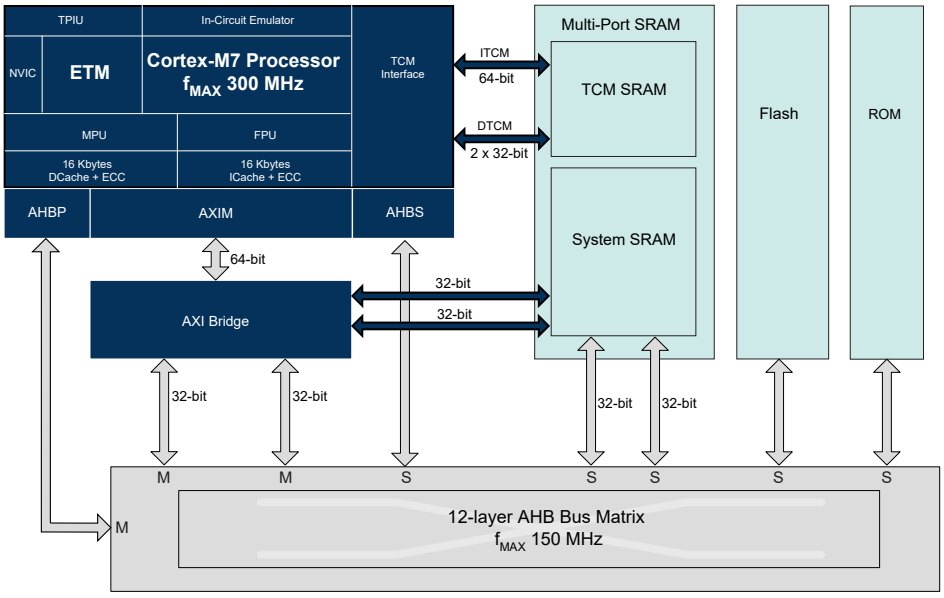
- Instruction fetches
- Data cache linefills and evictions
- Non-cacheable normal-type memory data accesses
- Device and strongly-ordered type data accesses, generally to peripherals

The interleaved multi-port SRAM optimizes the Cortex-M7 accesses to the internal SRAM.

The interconnect of the other masters and slaves is described in 19. Bus Matrix (MATRIX).

The figure below shows the connections of the different Cortex-M7 ports.

Figure 9-1. Interconnect Block Diagram



Clock Name	Peripheral
PCK5	MCANx
PCK6	TCx
PCK7	TC0

Note: USB, GMAC and MLB do not require PCKx to operate independently of core and bus peripherals.

31.9 Peripheral and Generic Clock Controller

The PMC controls the clocks of the embedded peripherals by means of the Peripheral Control register (PMC_PCR). With this register, the user can enable and disable the different clocks used by the peripherals:

- Peripheral clocks (periph_clk[PID]), routed to every peripheral and derived from the master clock (MCK), and
- Generic clocks (GCLK[PID]), routed to I2SC0 and I2SC1. These clocks are independent of the core and bus clocks (HCLK, MCK and periph_clk[PID]). They are generated by selection and division of the following sources: SLCK, MAINCK, UPLLCKDIV, PLLACK and MCK. Refer to the description of each peripheral for the limitation to be applied to GCLK[PID] compared to periph_clk[PID].

To configure a peripheral's clocks, PMC_PCR.CMD must be written to '1' and PMC_PCR.PID must be written with the index of the corresponding peripheral. All other configuration fields must be correctly set.

To read the current clock configuration of a peripheral, PMC_PCR.CMD must be written to '0' and PMC_PCR.PID must be written with the index of the corresponding peripheral regardless of the values of other fields. This write does not modify the configuration of the peripheral. The PMC_PCR can then be read to know the configuration status of the corresponding PID.

The user can also enable and disable these clocks by configuring the Peripheral Clock Enable (PMC_PCERx) and Peripheral Clock Disable (PMC_PCDRx) registers. The status of the peripheral clock activity can be read in the Peripheral Clock Status registers (PMC_PCSRx).

When a peripheral or a generic clock is disabled, it is immediately stopped. These clocks are disabled after a reset.

To stop a peripheral clock, it is recommended that the system software wait until the peripheral has executed its last programmed operation before disabling the clock. This is to avoid data corruption or erroneous behavior of the system.

The bit number in PMC_PCERx, PMC_PCDRx, and PMC_PCSRx is the Peripheral Identifier defined at the product level. The bit number corresponds to the interrupt source number assigned to the peripheral.

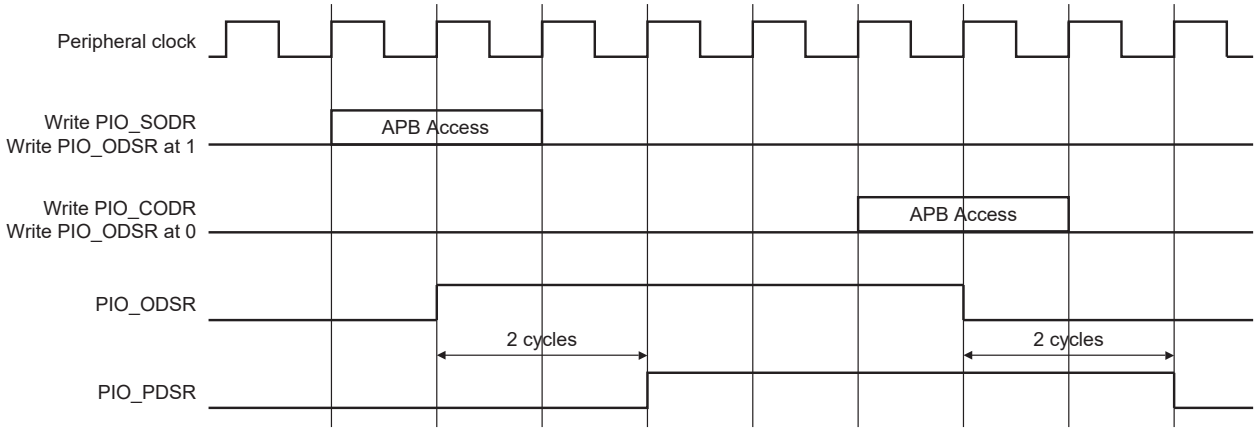
31.10 Asynchronous Partial Wakeup

31.10.1 Description

The asynchronous partial wakeup wakes up a peripheral in a fully asynchronous way when activity is detected on the communication line. The asynchronous partial wakeup function automatically manages the peripheral clock. It reduces overall power consumption of the system by clocking peripherals only when needed.

Asynchronous partial wakeup can be enabled in Wait mode (SleepWalking), or in Active mode.

Figure 32-3. Output Line Timings



32.5.8 Inputs

The level on each I/O line can be read through PIO_PDSR. This register indicates the level of the I/O lines regardless of their configuration, whether uniquely as an input, or driven by the PIO Controller, or driven by a peripheral.

Reading the I/O line levels requires the clock of the PIO Controller to be enabled, otherwise PIO_PDSR reads the levels present on the I/O line at the time the clock was disabled.

32.5.9 Input Glitch and Debouncing Filters

Optional input glitch and debouncing filters are independently programmable on each I/O line.

The glitch filter can filter a glitch with a duration of less than 1/2 peripheral clock and the debouncing filter can filter a pulse of less than 1/2 period of a programmable divided slow clock.

The selection between glitch filtering or debounce filtering is done by writing in the PIO Input Filter Slow Clock Disable Register (PIO_IFSCDR) and the PIO Input Filter Slow Clock Enable Register (PIO_IFSCER). Writing PIO_IFSCDR and PIO_IFSCER, respectively, sets and clears bits in the Input Filter Slow Clock Status Register (PIO_IFSCSR).

The current selection status can be checked by reading the PIO_IFSCSR.

- If PIO_IFSCSR[i] = 0: The glitch filter can filter a glitch with a duration of less than 1/2 master clock period.
- If PIO_IFSCSR[i] = 1: The debouncing filter can filter a pulse with a duration of less than 1/2 programmable divided slow clock period.

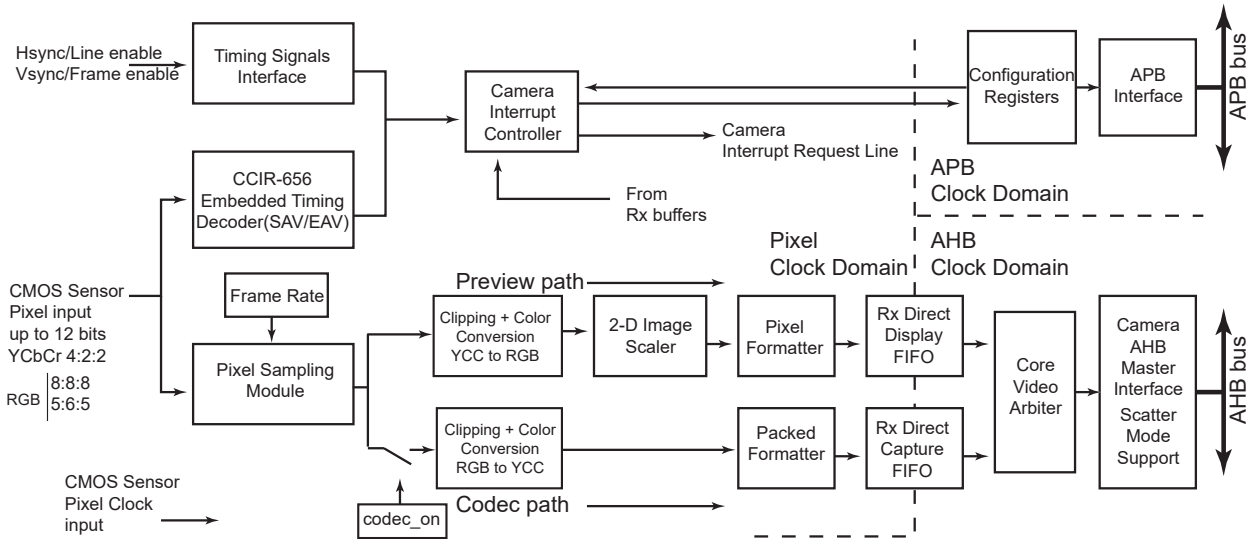
For the debouncing filter, the period of the divided slow clock is defined by writing in the DIV field of the Slow Clock Divider Debouncing Register (PIO_SCDR):

$t_{div\_slck} = ((DIV + 1) \times 2) \times t_{slck}$

When the glitch or debouncing filter is enabled, a glitch or pulse with a duration of less than 1/2 selected clock cycle (selected clock represents peripheral clock or divided slow clock depending on PIO_IFSCDR and PIO_IFSCER programming) is automatically rejected, while a pulse with a duration of one selected clock (peripheral clock or divided slow clock) cycle or more is accepted. For pulse durations between 1/2 selected clock cycle and one selected clock cycle, the pulse may or may not be taken into account, depending on the precise timing of its occurrence. Thus for a pulse to be visible, it must exceed one selected clock cycle, whereas for a glitch to be reliably filtered out, its duration must not exceed 1/2 selected clock cycle.

37.3 Block Diagram

Figure 37-2. ISI Block Diagram



37.4 Product Dependencies

37.4.1 I/O Lines

The pins used for interfacing the compliant external devices can be multiplexed with PIO lines. The programmer must first program the PIO controllers to assign the ISI pins to their peripheral functions.

37.4.2 Power Management

The ISI can be clocked through the Power Management Controller (PMC), thus the programmer must first configure the PMC to enable the ISI clock.

37.4.3 Interrupt Sources

The ISI interface has an interrupt line connected to the interrupt controller. Handling the ISI interrupt requires programming the interrupt controller before configuring the ISI.

37.5 Functional Description

The Image Sensor Interface (ISI) supports direct connection to the ITU-R BT. 601/656 8-bit mode compliant sensors and up to 12-bit grayscale sensors. It receives the image data stream from the image sensor on the 12-bit data bus.

This module receives up to 12 bits for data, the horizontal and vertical synchronizations and the pixel clock. The reduced pin count alternative for synchronization is supported for sensors that embed SAV (start of active video) and EAV (end of active video) delimiters in the data stream.

The Image Sensor Interface interrupt line is connected to the Advanced Interrupt Controller and can trigger an interrupt at the beginning of each frame and at the end of a DMA frame transfer. If the SAV/EAV synchronization is used, an interrupt can be triggered on each delimiter event.

For 8-bit color sensors, the data stream received can be in several possible formats: YCbCr 4:2:2, RGB 8:8:8, RGB 5:6:5 and may be processed before the storage in memory. When the preview DMA channel

Signal Name	Function	MII	RMII
GMDC	Management Data Clock	MDC	MDC
GMDIO	Management Data Input/Output	MDIO	MDIO

Note:

1. Input only. GTXCK must be provided with a 25 MHz / 50 MHz external crystal oscillator for MII / RMII interfaces, respectively.

38.5 Product Dependencies

38.5.1 I/O Lines

The pins used for interfacing the GMAC may be multiplexed with PIO lines. The programmer must first program the PIO Controller to assign the pins to their peripheral function. If I/O lines of the GMAC are not used by the application, they can be used for other purposes by the PIO Controller.

38.5.2 Power Management

The GMAC is not continuously clocked. The user must first enable the GMAC clock in the Power Management Controller before using it.

38.5.3 Interrupt Sources

The GMAC interrupt line is connected to one of the internal sources of the interrupt controller. Using the GMAC interrupt requires prior programming of the interrupt controller.

The GMAC features 6 interrupt sources. Refer to the table "Peripheral Identifiers" in the section "Peripherals" for the interrupt numbers for GMAC priority queues.

Related Links

14.1 Peripheral Identifiers

38.6 Functional Description

38.6.1 Media Access Controller

The Transmit Block of the Media Access Controller (MAC) takes data from FIFO, adds preamble, checks and adds padding and frame check sequence (FCS). Both half duplex and full duplex Ethernet modes of operation are supported.

When operating in half duplex mode, the MAC Transmit Block generates data according to the Carrier Sense Multiple Access with Collision Detect (CSMA/CD) protocol. The start of transmission is deferred if Carrier Sense (CRS) is active. If Collision (COL) is detected during transmission, a jam sequence is asserted and the transmission is retried after a random back off. The CRS and COL signals have no effect in full duplex mode.

The Receive Block of the MAC checks for valid preamble, FCS, alignment and length, and presents received frames to the MAC address checking block and FIFO. Software can configure the GMAC to receive jumbo frames of up to 10240 Bytes. It can optionally strip CRC (Cyclic Redundancy Check) from the received frame before transferring it to FIFO.

The Address Checker recognizes four specific 48-bit addresses, can recognize four different types of ID values, and contains a 64-bit Hash register for matching multicast and unicast addresses as required. It

38.6.12 VLAN Support

The following table describes an Ethernet encoded 802.1Q VLAN tag.

Table 38-5. 802.1Q VLAN Tag

TPID (Tag Protocol Identifier) 16 bits	TCI (Tag Control Information) 16 bits
0x8100	First 3 bits priority, then CFI bit, last 12 bits VID

The VLAN tag is inserted at the 13th byte of the frame adding an extra four bytes to the frame. To support these extra four bytes, the GMAC can accept frame lengths up to 1536 bytes by setting bit 8 in the Network Configuration register.

If the VID (VLAN identifier) is null (0x000) this indicates a priority-tagged frame.

The following bits in the receive buffer descriptor status word give information about VLAN tagged frames:-

- Bit 21 set if receive frame is VLAN tagged (i.e., type ID of 0x8100).
- Bit 20 set if receive frame is priority tagged (i.e., type ID of 0x8100 and null VID). (If bit 20 is set, bit 21 will be set also.)
- Bit 19, 18 and 17 set to priority if bit 21 is set.
- Bit 16 set to CFI if bit 21 is set.

The GMAC can be configured to reject all frames except VLAN tagged frames by setting the discard non-VLAN frames bit in the Network Configuration register.

38.6.13 Wake on LAN Support

The receive block supports Wake on LAN by detecting the following events on incoming receive frames:

- Magic packet
- Address Resolution Protocol (ARP) request to the device IP address
- Specific address 1 filter match
- Multicast hash filter match

These events can be individually enabled through bits [19:16] of the Wake on LAN register. Also, for Wake on LAN detection to occur, receive enable must be set in the Network Control register, however a receive buffer does not have to be available.

In case of an ARP request, specific address 1 or multicast filter events will occur even if the frame is errored. For magic packet events, the frame must be correctly formed and error free.

A magic packet event is detected if all of the following are true:

- Magic packet events are enabled through bit 16 of the Wake on LAN register
- The frame's destination address matches specific address 1
- The frame is correctly formed with no errors
- The frame contains at least 6 bytes of 0xFF for synchronization
- There are 16 repetitions of the contents of Specific Address 1 register immediately following the synchronization

An ARP request event is detected if all of the following are true:

- ARP request events are enabled through bit 17 of the Wake on LAN register
- Broadcasts are allowed by bit 5 in the Network Configuration register

38.8.11 GMAC Interrupt Enable Register

Name: GMAC_IER
Offset: 0x028
Reset: –
Property: Write-only

This register is write-only and will always return zero.

The following values are valid for all listed bit names of this register:

0: No effect.

1: Enables the corresponding interrupt.

Bit	31	30	29	28	27	26	25	24
			TSUTIMCMP	WOL	RXLPISBC	SRI	PDRSFT	PDRQFT
Access			W	W	R	W	W	W
Reset			–	–	–	–	–	–

Bit	23	22	21	20	19	18	17	16
	PDRSFR	PDRQFR	SFT	DRQFT	SFR	DRQFR		
Access	W	W	W	W	W	W		
Reset	–	–	–	–	–	–		

Bit	15	14	13	12	11	10	9	8
	EXINT	PFTR	PTZ	PFNZ	HRESP	ROVR		
Access	W	W	W	W	W	W		
Reset	–	–	–	–	–	–		

Bit	7	6	5	4	3	2	1	0
	TCOMP	TFC	RLEX	TUR	TXUBR	RXUBR	RCOMP	MFS
Access	W	W	W	W	W	W	W	W
Reset	–	–	–	–	–	–	–	–

Bit 29 – TSUTIMCMP TSU Timer Comparison

Bit 28 – WOL Wake On LAN

Bit 27 – RXLPISBC Receive LPI indication Status Bit Change
Receive LPI indication status bit change.

Cleared on read.

Bit 26 – SRI TSU Seconds Register Increment

Bit 25 – PDRSFT PDelay Response Frame Transmitted

Bit 24 – PDRQFT PDelay Request Frame Transmitted

Bit 23 – PDRSFR PDelay Response Frame Received

38.8.14 GMAC PHY Maintenance Register

Name: GMAC_MAN
Offset: 0x034
Reset: 0x00000000
Property: Read/Write

This register is a shift register. Writing to it starts a shift operation which is signaled completed when bit 2 is set in the Network Status Register (GMAC_NSR). It takes about 2000 MCK cycles to complete, when MDC is set for MCK divide by 32 in the Network Configuration Register. An interrupt is generated upon completion.

During this time, the MSB of the register is output on the MDIO pin and the LSB updated from the MDIO pin with each MDC cycle. This causes transmission of a PHY management frame on MDIO. Refer also to section 22.2.4.5 of the IEEE 802.3 standard.

Reading during the shift operation returns the current contents of the shift register. At the end of management operation, the bits will have shifted back to their original locations. For a read operation, the data bits are updated with data read from the PHY. It is important to write the correct values to the register to ensure a valid PHY management frame is produced.

The MDIO interface can read IEEE 802.3 clause 45 PHYs, as well as clause 22 PHYs. To read clause 45 PHYs, bit 30 should be written with a '0' rather than a '1'. To write clause 45 PHYs, bits 31:28 should be written as 0x1:

PHY	Access	Bit Value			
		WZO	CLTTO	OP[1]	OP[0]
Clause 22	Read	0	1	1	0
	Write	0	1	0	1
Clause 45	Read	0	0	1	1
	Write	0	0	0	1
	Read + Address	0	0	1	0

For a description of MDC generation, see also the 'GMAC Network Configuration Register' (GMAC_NCR) description.

39.6.20 Device Endpoint Interrupt Set Register (Isochronous Endpoints)

Name: USBHS_DEVEPTIFRx (ISOENPT)
Offset: 0x0190 + x*0x04 [x=0..9]
Reset: 0
Property: Read/Write

This register view is relevant only if EPTYPE = 0x1 in "Device Endpoint x Configuration Register".

For additional information, see "Device Endpoint x Status Register (Isochronous Endpoints)".

The following configuration values are valid for all listed bit names of this register:

0: No effect.

1: Sets the corresponding bit in USBHS_DEVEPTISRx, which may be useful for test or debug purposes.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
				NBUSYBKS				
Access								
Reset				0				

Bit	7	6	5	4	3	2	1	0
	SHORTPACKETS	CRCERRIS	OVERFIS	HBISOFLUSHIS	HBISOINERRIS	UNDERFIS	RXOUTIS	TXINIS
Access								
Reset	0	0	0	0	0	0	0	0

Bit 12 – NBUSYBKS Number of Busy Banks Interrupt Set

Bit 7 – SHORTPACKETS Short Packet Interrupt Set

Bit 6 – CRCERRIS CRC Error Interrupt Set

Bit 5 – OVERFIS Overflow Interrupt Set

Bit 4 – HBISOFLUSHIS High Bandwidth Isochronous IN Flush Interrupt Set

Bit 3 – HBISOINERRIS High Bandwidth Isochronous IN Underflow Error Interrupt Set

Bit 2 – UNDERFIS Underflow Interrupt Set

Value	Description
0	Cleared when USBHS_HSTPIPIDR.OVERFIEC = 1. This disables the Transmitted IN Data interrupt (USBHS_HSTPIPIMR.OVERFIE).
1	Set when USBHS_HSTPIPIER.OVERFIES = 1. This enables the Transmitted IN Data interrupt (USBHS_HSTPIPIMR.OVERFIE).

Bit 4 – NAKEDE NAKed Interrupt Enable

Value	Description
0	Cleared when USBHS_HSTPIPIDR.NAKEDEC = 1. This disables the Transmitted IN Data interrupt (USBHS_HSTPIPIMR.NAKEDE).
1	Set when USBHS_HSTPIPIER.NAKEDES = 1. This enables the Transmitted IN Data interrupt (USBHS_HSTPIPIMR.NAKEDE).

Bit 3 – PERRE Pipe Error Interrupt Enable

Value	Description
0	Cleared when USBHS_HSTPIPIDR.PERREC = 1. This disables the Transmitted IN Data interrupt (USBHS_HSTPIPIMR.PERRE).
1	Set when USBHS_HSTPIPIER.PERRES = 1. This enables the Transmitted IN Data interrupt (USBHS_HSTPIPIMR.PERRE).

Bit 2 – UNDERFIE Underflow Interrupt Enable

Value	Description
0	Cleared when USBHS_HSTPIPIDR.UNDERFIEC= 1. This disables the Transmitted IN Data interrupt (USBHS_HSTPIPIMR.UNDERFIE).
1	Set when USBHS_HSTPIPIER.UNDERFIES = 1. This enables the Transmitted IN Data interrupt (USBHS_HSTPIPIMR.UNDERFIE).

Bit 1 – TXOUTE Transmitted OUT Data Interrupt Enable

Value	Description
0	Cleared when USBHS_HSTPIPIDR.TXOUTEC = 1. This disables the Transmitted IN Data interrupt (USBHS_HSTPIPIMR.TXOUTE).
1	Set when USBHS_HSTPIPIER.TXOUTES = 1. This enables the Transmitted IN Data interrupt (USBHS_HSTPIPIMR.TXOUTE).

Bit 0 – RXINE Received IN Data Interrupt Enable

Value	Description
0	Cleared when USBHS_HSTPIPIDR.RXINEC = 1. This disables the Transmitted IN Data interrupt (USBHS_HSTPIPIMR.RXINE).
1	Set when USBHS_HSTPIPIER.RXINES= 1. This enables the Transmitted IN Data interrupt (USBHS_HSTPIPIMR.RXINE).

last character transfer. Then, another DMA transfer can be started if SPI_CR.SPIEN has previously been written.

41.7.3.6 SPI Direct Access Memory Controller (DMAC)

In both Fixed and Variable modes, the Direct Memory Access Controller (DMAC) can be used to reduce processor overhead.

The fixed peripheral selection allows buffer transfers with a single peripheral. Using the DMAC is an optimal means, as the size of the data transfer between the memory and the SPI is either 8 bits or 16 bits. However, if the peripheral selection is modified, SPI_MR must be reprogrammed.

The variable peripheral selection allows buffer transfers with multiple peripherals without reprogramming SPI_MR. Data written in SPI_TDR is 32 bits wide and defines the real data to be transmitted and the destination peripheral. Using the DMAC in this mode requires 32-bit wide buffers, with the data in the LSBs and the PCS and LASTXFER fields in the MSBs. However, the SPI still controls the number of bits (8 to 16) to be transferred through MISO and MOSI lines with the chip select configuration registers. This is not the optimal means in terms of memory size for the buffers, but it provides a very effective means to exchange data with several peripherals without any intervention of the processor.

41.7.3.7 Peripheral Chip Select Decoding

The user can program the SPI to operate with up to 15 slave peripherals by decoding the four chip select lines, NPCS0 to NPCS3 with an external decoder/demultiplexer (see figure below). This can be enabled by setting SPI_MR.PCSDEC.

When operating without decoding, the SPI makes sure that in any case only one chip select line is activated, i.e., one NPCS line driven low at a time. If two bits are defined low in a PCS field, only the lowest numbered chip select is driven low.

When operating with decoding, the SPI directly outputs the value defined by the PCS field on the NPCS lines of either SPI_MR or SPI_TDR (depending on PS).

As the SPI sets a default value of 0xF on the chip select lines (i.e., all chip select lines at 1) when not processing any transfer, only 15 peripherals can be decoded.

The SPI has four chip select registers (SPI_CSR0...SPI_CSR3). As a result, when external decoding is activated, each NPCS chip select defines the characteristics of up to four peripherals. As an example, SPI_CRS0 defines the characteristics of the externally decoded peripherals 0 to 3, corresponding to the PCS values 0x0 to 0x3. Consequently, the user has to make sure to connect compatible peripherals on the decoded chip select lines 0 to 3, 4 to 7, 8 to 11 and 12 to 14. The following figure shows this type of implementation.

If SPI_CSRx.CSAAT bit is used, with or without the DMAC, the Mode Fault detection for NPCS0 line must be disabled. This is not needed for all other chip select lines since Mode Fault detection is only on NPCS0.

42.5.2 Power Management

The QSPI may be clocked through the Power Management Controller (PMC), thus the programmer must first configure the PMC to enable the QSPI clock.

42.5.3 Interrupt Sources

The QSPI has an interrupt line connected to the Interrupt Controller. Handling the QSPI interrupt requires programming the interrupt controller before configuring the QSPI.

42.5.4 Direct Memory Access Controller (DMA)

The QSPI can be used in conjunction with the Direct Memory Access Controller (DMA) in order to reduce processor overhead. For a full description of the DMA, refer to the section “DMA Controller (XDMAC)”.

Note: DMA write accesses must be 32-bit aligned. If a single byte is to be written in a 32-bit word, the rest of the word must be filled with ones.

Related Links

36. DMA Controller (XDMAC)

42.6 Functional Description

42.6.1 Serial Clock Baud Rate

The QSPI baud rate clock is generated by dividing the peripheral clock by a value between 1 and 256.

42.6.2 Serial Clock Phase and Polarity

Four combinations of polarity and phase are available for data transfers. The clock polarity is programmed with the CPOL bit in the QSPI Serial Clock register (QSPI_SCR). The CPHA bit in the QSPI_SCR programs the clock phase. These two parameters determine the edges of the clock signal on which data is driven and sampled. Each of the two parameters has two possible states, resulting in four possible combinations that are incompatible with one another. Thus, the interfaced slave must use the same parameter values to communicate.

The table below shows the four modes and corresponding parameter settings.

Table 42-2. QSPI Bus Clock Modes

QSPI Clock Mode	QSPI_SCR.CPOL	QSPI_SCR.CPHA	Shift QSCK Edge	Capture QSCK Edge	QSCK Inactive Level
0	0	0	Falling	Rising	Low
1	0	1	Rising	Falling	Low
2	1	0	Rising	Falling	High
3	1	1	Falling	Rising	High

The following figures show examples of data transfers.

45.8.3 I2SC Status Register

Name: I2SC_SR
Offset: 0x08
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
			TXURCH[1:0]					
Access			R	R				
Reset			0	0				

Bit	15	14	13	12	11	10	9	8
							RXORCH[1:0]	
Access							R	R
Reset							0	0

Bit	7	6	5	4	3	2	1	0
		TXUR	TXRDY	TXEN		RXOR	RXRDY	RXEN
Access		R	R	R		R	R	R
Reset		0	0	0		0	0	0

Bits 21:20 – TXURCH[1:0] Transmit Underrun Channel

Value	Description
0	This field is cleared when I2SC_SCR.TXUR is written to ’1’.
1	Bit i of this field is set when a transmit underrun error occurred in channel i (i = 0 for first channel of the frame).

Bits 9:8 – RXORCH[1:0] Receive Overrun Channel
This field is cleared when I2SC_SCR.RXOR is written to ’1’.

Bit i of this field is set when a receive overrun error occurred in channel i (i = 0 for first channel of the frame).

Bit 6 – TXUR Transmit Underrun

Value	Description
0	This bit is cleared when the corresponding bit in I2SC_SCR is written to ’1’.
1	This bit is set when an underrun error occurs on I2SC_THR or when the corresponding bit in I2SC_SSR is written to ’1’.

Bit 5 – TXRDY Transmit Ready

46.7.43 USART LON IDT Tx Register

Name: US_IDTTX
Offset: 0x0080
Reset: 0x0
Property: Read/Write

This register is relevant only if USART_MODE = 0x9 in the USART Mode Register.

This register can only be written if the WPEN bit is cleared in the USART Write Protection Mode Register.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
	IDTTX[23:16]							
Access								
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	IDTTX[15:8]							
Access								
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	IDTTX[7:0]							
Access								
Reset	0	0	0	0	0	0	0	0

Bits 23:0 – IDTTX[23:0] LON Indeterminate Time after Transmission (comm_type = 1 mode only)

Value	Description
0–16777215	LON indeterminate time after transmission in t_{bit}.

Table 48-8. MediaLB Channel Address to Logical Channel Mapping

Channel Address	Logical Channel
0x0002	1
0x0004	2
0x0006	3
....	....
0x007C	62
0x007E	63
0x01FE	0(1)

Note: 1. Logical Channel 0 is the System Channel and is reserved.

48.6.3.2 Host Bus Interface Block

The Host Bus Interface (HBI) block provides a 16-bit parallel slave port that provides an external Host Controller (HC) with access to all MOST channels and data types.

Up to 64 independent HBI channels are available to the HC, each configurable for either transmitting or receiving a particular application data type (synchronous, isochronous, asynchronous, or control). The HBI block provides source and sink access to the full network data bandwidth.

HBI Physical Addresses

To access a particular HBI DMA channel, hardware must first translate the HBI channel address to a channel allocation table (CAT) physical address. This physical address is then used to retrieve the channel label (CL), which in turn retrieves the channel descriptor.

See the following table for more information on the mapping between the HBI channel address and physical address.

Table 48-9. HBI Channel Address to Physical Address Mapping

HBI Channel	CAT Address	CAT Offset
0x0	0x88	000
0x1	0x88	001
0x2	0x88	010
0x3	0x88	011
0x4	0x88	100
0x5	0x88	101
0x6	0x88	110
0x7	0x88	111
0x8	0x89	000
...	...	...

Value	Description
0	Hardware clears interrupt after a MLB_ACSRn register read
1	Software writes a ‘1’ to clear

52.7.18 AFEC Channel Selection Register

Name: AFEC_CSELR
Offset: 0x64
Reset: 0x00000000
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
					CSEL[3:0]			
Access					R/W	R/W	R/W	R/W
Reset					0	0	0	0

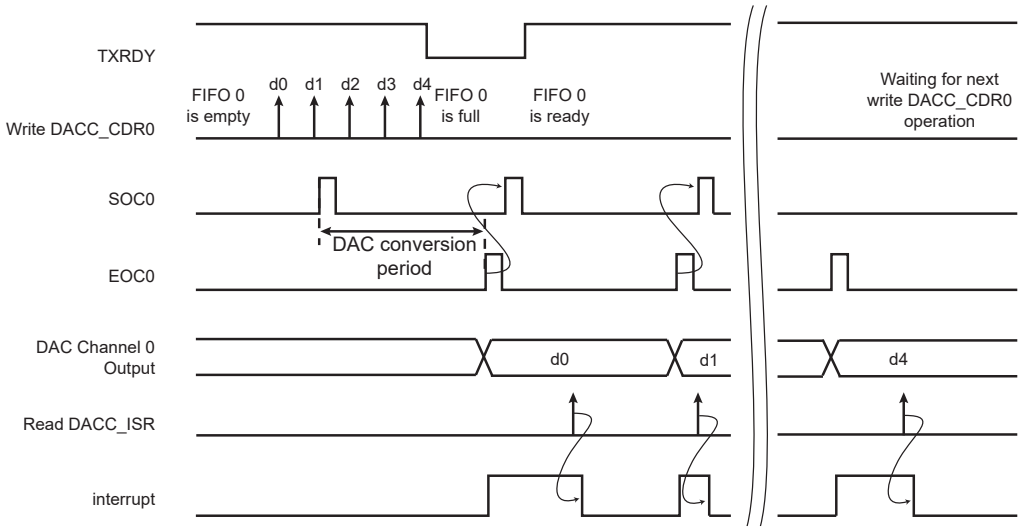
Bits 3:0 – CSEL[3:0] Channel Selection

Value	Description
0-11	Selects the channel to be displayed in AFEC_CDR and AFEC_COCR. To be configured with the appropriate channel number.

the corresponding analog output. The next data is converted only when the EOC of the previous data is set.

If the FIFO is emptied, no conversion occurs and the data is maintained at the output of the DAC.

Figure 53-3. Conversion Sequence in Free-running Mode



53.6.4.3 Max Speed Mode

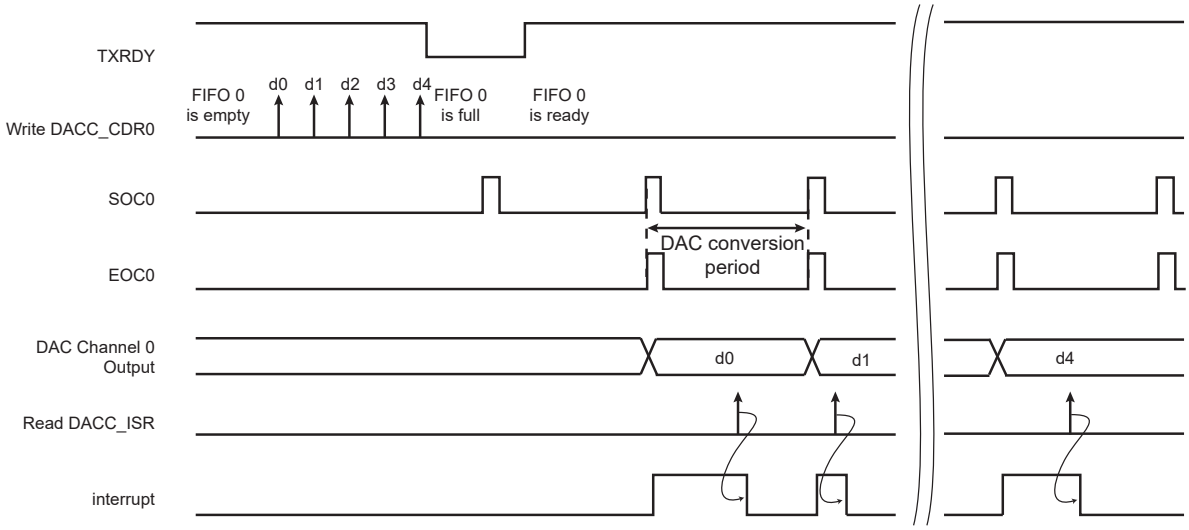
Max speed mode is enabled by setting DACC_TRIGR.TRGENx and DACC_MR.MAXSx.

The conversion rate is forced by the controller, which starts one conversion every 12 DAC clock periods. The controller does not wait for the EOC of the previous data to send a new data to the DAC and the DAC is always clocked.

If the FIFO is emptied, the controller send the last converted data to the DAC at a rate of 12 DAC clock periods.

The DACC_ACR.IBCTLCHx field must be configured for 1 MSps (see the section “Electrical Characteristics”).

Figure 53-4. Conversion Sequence in Max Speed Mode



54.7.9 ACC Write Protection Status Register

Name: ACC_WPSR
Offset: 0xE8
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
								WPVS
Access								R
Reset								0

Bit 0 – WPVS Write Protection Violation Status

Value	Description
0	No write protection violation has occurred since the last read of ACC_WPSR.
1	A write protection violation (WPEN = 1) has occurred since the last read of ACC_WPSR.

55.5 Functional Description

55.5.1 Overview

The Integrity Check Monitor (ICM) is a DMA controller that performs SHA-based memory hashing over memory regions. As shown in figure Integrity Check Monitor Block Diagram, it integrates a DMA interface, a Monitoring Finite State Machine (FSM), an integrity scheduler, a set of context registers, a SHA engine, an interface for configuration and status registers.

The ICM integrates a Secure Hash Algorithm engine (SHA). This engine requires a message padded according to FIPS180-2 specification when used as a SHA calculation unit only. Otherwise, if the ICM is used as integrated check for memory content, the padding is not mandatory. The SHA module produces an N-bit message digest each time a block is read and a processing period ends. N is 160 for SHA1, 224 for SHA224, 256 for SHA256.

When the ICM module is enabled, it sequentially retrieves a circular list of region descriptors from the memory (Main List described in figure ICM Region Descriptor and Hash Areas). Up to four regions may be monitored. Each region descriptor is composed of four words indicating the layout of the memory region (see figure Region Descriptor). It also contains the hashing engine configuration on a per-region basis. As soon as the descriptor is loaded from the memory and context registers are updated with the data structure, the hashing operation starts. A programmable number of blocks (see TRSIZE field of the ICM_RCTRL structure member) is transferred from the memory to the SHA engine. When the desired number of blocks have been transferred, the digest is either moved to memory (Write Back function) or compared with a digest reference located in the system memory (Compare function). If a digest mismatch occurs, an interrupt is triggered if unmasked. The ICM module passes through the region descriptor list until the end of the list marked by an end of list marker (WRAP or EOM bit in ICM_RCFG structure member set to one). To continuously monitor the list of regions, the WRAP bit must be set to one in the last data structure and EOM must be cleared.