Welcome to E-XFL.COM

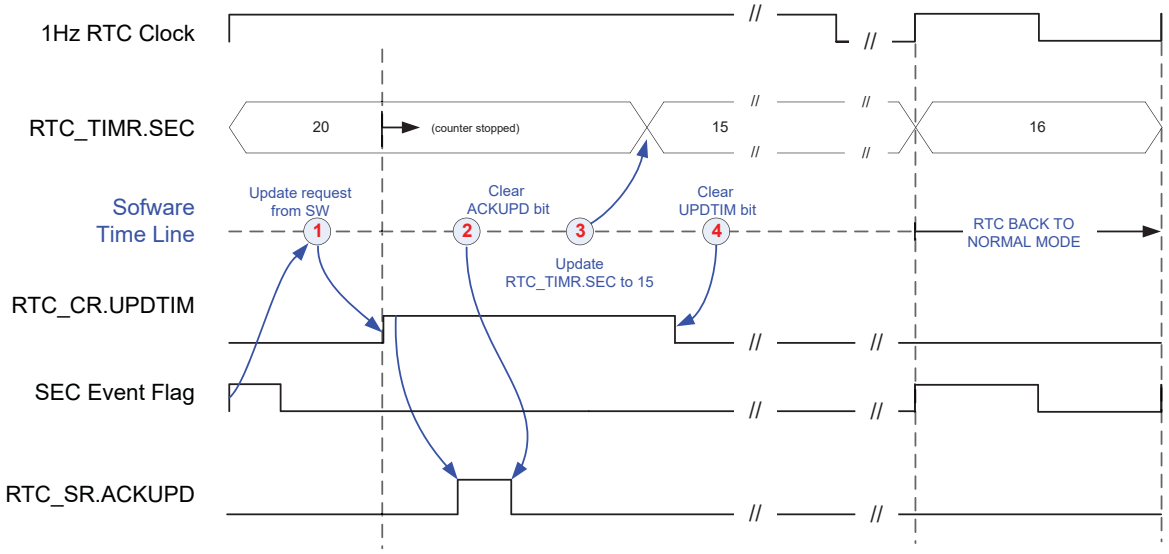
What is "Embedded - Microcontrollers"?

"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	CANbus, EBI/EMI, I²C, IrDA, LINbus, MMC/SD/SDIO, QSPI, SPI, SSC, UART/USART, USB
Peripherals	Brown-out Detect/Reset, DMA, I²S, POR, PWM, WDT
Number of I/O	114
Program Memory Size	1MB (1M x 8)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	384K x 8
Voltage - Supply (Vcc/Vdd)	3V ~ 3.6V
Data Converters	A/D 24x12b; D/A 2x12b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 105°C (TA)
Mounting Type	Surface Mount
Package / Case	144-TFBGA
Supplier Device Package	144-TFBGA (10x10)
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/atsamv70q20b-cb

Figure 27-2. Time/Calendar Update Timing Diagram



31.12 Programmable Clock Output Controller

The PMC controls three signals to be output on the external pins PCKx. Each signal can be independently programmed via the Programmable Clock registers (PMC_PCKx).

PCKx can be independently selected between SLCK, MAINCK, PLLACK, UPLLCKDIV and MCK by configuring PMC_PCKx.CSS. Each output signal can also be divided by 1 to 256 by configuring PMC_PCKx.PRES.

Each output signal can be enabled and disabled by writing a ‘1’ to the corresponding bits PMC_SCER.PCKx and PMC_SCDR.PCKx, respectively. The status of the active programmable output clocks is given in PMC_SCSR.PCKx.

The status flag PMC_SR.PCKRDYx indicates that PCKx is actually what has been programmed in registers PMC_PCKx.

As the Programmable Clock Controller does not manage with glitch prevention when switching clocks, it is strongly recommended to disable PCKx before any configuration change and to re-enable it after the change is performed.

31.13 Fast Startup

At exit from Wait mode, the device allows the processor to restart in several microseconds only if the C-code function that manages the Wait mode entry and exit is linked to and executed from on-chip SRAM.

The fast startup time cannot be achieved if the first instruction after an exit is located in the embedded Flash.

If fast startup is not required, or if the first instruction after exit from Wait mode is located in embedded Flash, see "Startup from Embedded Flash".

To instruct the device to enter Wait mode, refer to section “Power Considerations”.

A fast startup occurs upon the detection of a programmed level on one of the 14 wakeup inputs (WKUP) or upon an active alarm from the RTC, RTT and USB Controller. The polarity of each of the 14 wakeup inputs is programmable in the PMC Fast Startup Polarity Register (PMC_FSPR).

⚠WARNING The duration of the WKUPx pins active level must be greater than four MAINCK cycles.

The fast startup circuitry, as shown in the following figure, is fully asynchronous and provides a fast startup signal to the PMC. As soon as the fast startup signal is asserted, the Main RC oscillator restarts automatically.

When entering Wait mode, the embedded Flash can be placed in one of the low-power modes (Deep-powerdown or Standby mode) with PMC_FSMR.FLPM. FLPM can be configured at any time and its value will be applied to the next Wait mode period.

The power consumption reduction is optimal when PMC_FSMR.FLPM is configured to ‘1’ (Deep-powerdown mode). If the field is configured to ‘0’ (Standby mode), the power consumption is slightly higher than in Deep-powerdown mode.

When PMC_FSMR.FLPM is configured to ‘2’, the Wait mode Flash power consumption is equivalent to that of the Active mode when there is no read access on the Flash.

37. Image Sensor Interface (ISI)

37.1 Description

The Image Sensor Interface (ISI) connects a CMOS-type image sensor to the processor and provides image capture in various formats. The ISI performs data conversion, if necessary, before the storage in memory through DMA.

The ISI supports color CMOS image sensor and grayscale image sensors with a reduced set of functionalities.

In Grayscale mode, the data stream is stored in memory without any processing and so is not compatible with the LCD controller.

Internal FIFOs on the preview and codec paths are used to store the incoming data. The RGB output on the preview path is compatible with the LCD controller. This module outputs the data in RGB format (LCD compatible) and has scaling capabilities to make it compliant to the LCD display resolution (see the table RGB Format in Default Mode, RGB_CFG = 00, No Swap).

Several input formats such as preprocessed RGB or YCbCr are supported through the data bus interface.

The ISI supports two synchronization modes:

- Hardware with ISI_VSYNC and ISI_HSYNC signals
- International Telecommunication Union Recommendation ITU-R BT.656-4 Start-of-Active-Video (SAV) and End-of-Active-Video (EAV) synchronization sequence

Using EAV/SAV for synchronization reduces the pin count (ISI_VSYNC, ISI_HSYNC not used). The polarity of the synchronization pulse is programmable to comply with the sensor signals.

Table 37-1. I/O Description

Signal	Direction	Description
ISI_VSYNC	In	Vertical Synchronization
ISI_HSYNC	In	Horizontal Synchronization
ISI_DATA[11..0]	In	Sensor Pixel Data
ISI_MCK	Out	Master Clock provided to the Image Sensor. Refer to “Clocks”.
ISI_PCK	In	Pixel Clock provided by the Image Sensor

Specific Address 1 Bottom register (GMAC_SAB1) (Address 0x088) 0x87654321

Specific Address 1 Top register (GMAC_SAT1) (Address 0x08C) 0x0000CBA9

For a successful match to the type ID, the following Type ID Match 1 register must be set up:

Type ID Match 1 register (GMAC_TIDM1) (Address 0x0A8) 0x80004321

38.6.8 Broadcast Address

Frames with the broadcast address of 0xFFFFFFFFFFFF are stored to memory only if the 'no broadcast' bit in the Network Configuration register is set to zero.

38.6.9 Hash Addressing

The hash address register is 64 bits long and takes up two locations in the memory map. The least significant bits are stored in Hash Register Bottom and the most significant bits in Hash Register Top.

The unicast hash enable and the multicast hash enable bits in the Network Configuration register enable the reception of hash matched frames. The destination address is reduced to a 6-bit index into the 64-bit Hash register using the following hash function: The hash function is an XOR of every sixth bit of the destination address.

hash_index[05] = da[05] ^ da[11] ^ da[17] ^ da[23] ^ da[29] ^ da[35] ^ da[41] ^ da[47]

hash_index[04] = da[04] ^ da[10] ^ da[16] ^ da[22] ^ da[28] ^ da[34] ^ da[40] ^ da[46]

hash_index[03] = da[03] ^ da[09] ^ da[15] ^ da[21] ^ da[27] ^ da[33] ^ da[39] ^ da[45]

hash_index[02] = da[02] ^ da[08] ^ da[14] ^ da[20] ^ da[26] ^ da[32] ^ da[38] ^ da[44]

hash_index[01] = da[01] ^ da[07] ^ da[13] ^ da[19] ^ da[25] ^ da[31] ^ da[37] ^ da[43]

hash_index[00] = da[00] ^ da[06] ^ da[12] ^ da[18] ^ da[24] ^ da[30] ^ da[36] ^ da[42]

da[0] represents the least significant bit of the first byte received, that is, the multicast/unicast indicator, and da[47] represents the most significant bit of the last byte received.

If the hash index points to a bit that is set in the Hash register then the frame will be matched according to whether the frame is multicast or unicast.

A multicast match will be signaled if the multicast hash enable bit is set, da[0] is logic 1 and the hash index points to a bit set in the Hash register.

A unicast match will be signaled if the unicast hash enable bit is set, da[0] is logic 0 and the hash index points to a bit set in the Hash register.

To receive all multicast frames, the Hash register should be set with all ones and the multicast hash enable bit should be set in the Network Configuration register.

38.6.10 Copy all Frames (Promiscuous Mode)

If the Copy All Frames bit is set in the Network Configuration register then all frames (except those that are too long, too short, have FCS errors or have GRXER asserted during reception) will be copied to memory. Frames with FCS errors will be copied if bit 26 is set in the Network Configuration register.

38.6.11 Disable Copy of Pause Frames

Pause frames can be prevented from being written to memory by setting the disable copying of pause frames control bit 23 in the Network Configuration register. When set, pause frames are not copied to memory regardless of the Copy All Frames bit, whether a hash match is found, a type ID match is identified or if a destination address match is found.

Offset	Name	Bit Pos.								
		23:16	NFTX[23:16]							
		31:24	NFTX[31:24]							
0x0124	GMAC_TBFT511	7:0	NFTX[7:0]							
		15:8	NFTX[15:8]							
		23:16	NFTX[23:16]							
		31:24	NFTX[31:24]							
0x0128	GMAC_TBFT1023	7:0	NFTX[7:0]							
		15:8	NFTX[15:8]							
		23:16	NFTX[23:16]							
		31:24	NFTX[31:24]							
0x012C	GMAC_TBFT1518	7:0	NFTX[7:0]							
		15:8	NFTX[15:8]							
		23:16	NFTX[23:16]							
		31:24	NFTX[31:24]							
0x0130	GMAC_GTBFT1518	7:0	NFTX[7:0]							
		15:8	NFTX[15:8]							
		23:16	NFTX[23:16]							
		31:24	NFTX[31:24]							
0x0134	GMAC_TUR	7:0	TXUNR[7:0]							
		15:8							TXUNR[9:8]	
		23:16								
		31:24								
0x0138	GMAC_SCF	7:0	SCOL[7:0]							
		15:8	SCOL[15:8]							
		23:16							SCOL[17:16]	
		31:24								
0x013C	GMAC_MCF	7:0	MCOL[7:0]							
		15:8	MCOL[15:8]							
		23:16							MCOL[17:16]	
		31:24								
0x0140	GMAC_EC	7:0	XCOL[7:0]							
		15:8							XCOL[9:8]	
		23:16								
		31:24								
0x0144	GMAC_LC	7:0	LCOL[7:0]							
		15:8							LCOL[9:8]	
		23:16								
		31:24								
0x0148	GMAC_DTF	7:0	DEFT[7:0]							
		15:8	DEFT[15:8]							
		23:16							DEFT[17:16]	
		31:24								
0x014C	GMAC_CSE	7:0	CSR[7:0]							
		15:8							CSR[9:8]	
		23:16								
		31:24								

38.8.58 GMAC Octets Received Low Register

Name: GMAC_ORLO
Offset: 0x150
Reset: 0x00000000
Property: -

When reading the Octets Transmitted and Octets Received Registers, bits [31:0] should be read prior to bits [47:32] to ensure reliable operation.

Bit	31	30	29	28	27	26	25	24
	RXO[31:24]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

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

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

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

Bits 31:0 – RXO[31:0] Received Octets
Received octets in frame without errors [31:0]. The number of octets received in valid frames of any type. This counter is 48-bits and is read through two registers. This count does not include octets from pause frames, and is only incremented if the frame is successfully filtered and copied to memory.

38.8.111 GMAC Interrupt Disable Register Priority Queue x

Name: GMAC_IDRPQx
Offset: 0x0620 + x*0x04 [x=0..4]
Reset: –
Property: Write-only

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

0: No effect.

1: Disables the corresponding interrupt.

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
					HRESP	ROVR		
Access					W	W		
Reset					–	–		

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

Bit 11 – HRESP HRESP Not OK

Bit 10 – ROVR Receive Overrun

Bit 7 – TCOMP Transmit Complete

Bit 6 – TFC Transmit Frame Corruption Due to AHB Error

Bit 5 – RLEX Retry Limit Exceeded or Late Collision

Bit 2 – RXUBR RX Used Bit Read

Bit 1 – RCOMP Receive Complete

- Device Connection (USBHS_HSTISR.DCONNI)
- Device Disconnection (USBHS_HSTISR.DDISCI)
- USB Reset Sent (USBHS_HSTISR.RSTI)
- Downstream Resume Sent (USBHS_HSTISR.RSMEDI)
- Upstream Resume Received (USBHS_HSTISR.RXRSMI)
- Host Start of Frame (USBHS_HSTISR.HSOFI)
- Host Wakeup (USBHS_HSTISR.HWUPI)
- Pipe x (USBHS_HSTISR.PEP_x)
- DMA Channel x (USBHS_HSTISR.DMAxINT)

There is no exception host global interrupt.

Pipe Interrupts

The processing host pipe interrupts are:

- Received IN Data (USBHS_HSTPIPISRx.RXINI)
- Transmitted OUT Data (USBHS_HSTPIPISRx.TXOUTI)
- Transmitted SETUP (USBHS_HSTPIPISRx.TXSTPI)
- Short Packet (USBHS_HSTPIPISRx.SHORTPACKETI)
- Number of Busy Banks (USBHS_HSTPIPISRx.NBUSYBK)

The exception host pipe interrupts are:

- Underflow (USBHS_HSTPIPISRx.UNDERFI)
- Pipe Error (USBHS_HSTPIPISRx.PERRI)
- NAKed (USBHS_HSTPIPISRx.NAKEDI)
- Overflow (USBHS_HSTPIPISRx.OVERFI)
- Received STALLed (USBHS_HSTPIPISRx.RXSTALLDI)
- CRC Error (USBHS_HSTPIPISRx.CRCERRI)
  DMA Interrupts

The processing host DMA interrupts are:

- The End of USB Transfer Status (USBHS_HSTDMASTATUSx.END_TR_ST)
- The End of Channel Buffer Status (USBHS_HSTDMASTATUSx.END_BF_ST)
- The Descriptor Loaded Status (USBHS_HSTDMASTATUSx.DESC_LDST)

There is no exception host DMA interrupt.

39.5.4 USB DMA Operation

USB packets of any length may be transferred when required by the USBHS. These transfers always feature sequential addressing. Such characteristics mean that in case of high USBHS throughput, both AHB ports benefit from “incrementing burst of unspecified length” since the average access latency of AHB slaves can then be reduced.

The DMA uses word “incrementing burst of unspecified length” of up to 256 beats for both data transfers and channel descriptor loading. A burst may last on the AHB busses for the duration of a whole USB packet transfer, unless otherwise broken by the AHB arbitration or the AHB 1-Kbyte boundary crossing.

Packet data AHB bursts may be locked on a DMA buffer basis for drastic overall AHB bus bandwidth performance boost with paged memories. This prevents large AHB bursts from being broken in case of conflict with other AHB bus masters, thus avoiding access latencies due to memory row changes. This

Value	Description
0	Cleared when USBHS_DEVEPTIDRx.NAKINEC = 1. This disables the NAKed IN interrupt (USBHS_DEVEPTISRx.NAKINI).
1	Set when USBHS_DEVEPTIERx.NAKINES = 1. This enables the NAKed IN interrupt (USBHS_DEVEPTISRx.NAKINI).

Bit 3 – NAKOUTE NAKed OUT Interrupt

Value	Description
0	Cleared when USBHS_DEVEPTIDRx.NAKOUTEC = 1. This disables the NAKed OUT interrupt (USBHS_DEVEPTISRx.NAKOUTI).
1	Set when USBHS_DEVEPTIERx.NAKOUTES = 1. This enables the NAKed OUT interrupt (USBHS_DEVEPTISRx.NAKOUTI).

Bit 2 – RXSTPE Received SETUP Interrupt

Value	Description
0	Cleared when USBHS_DEVEPTIERx.RXSTPEC = 1. This disables the Received SETUP interrupt (USBHS_DEVEPTISRx.RXSTPI).
1	Set when USBHS_DEVEPTIERx.RXSTPES = 1. This enables the Received SETUP interrupt (USBHS_DEVEPTISRx.RXSTPI).

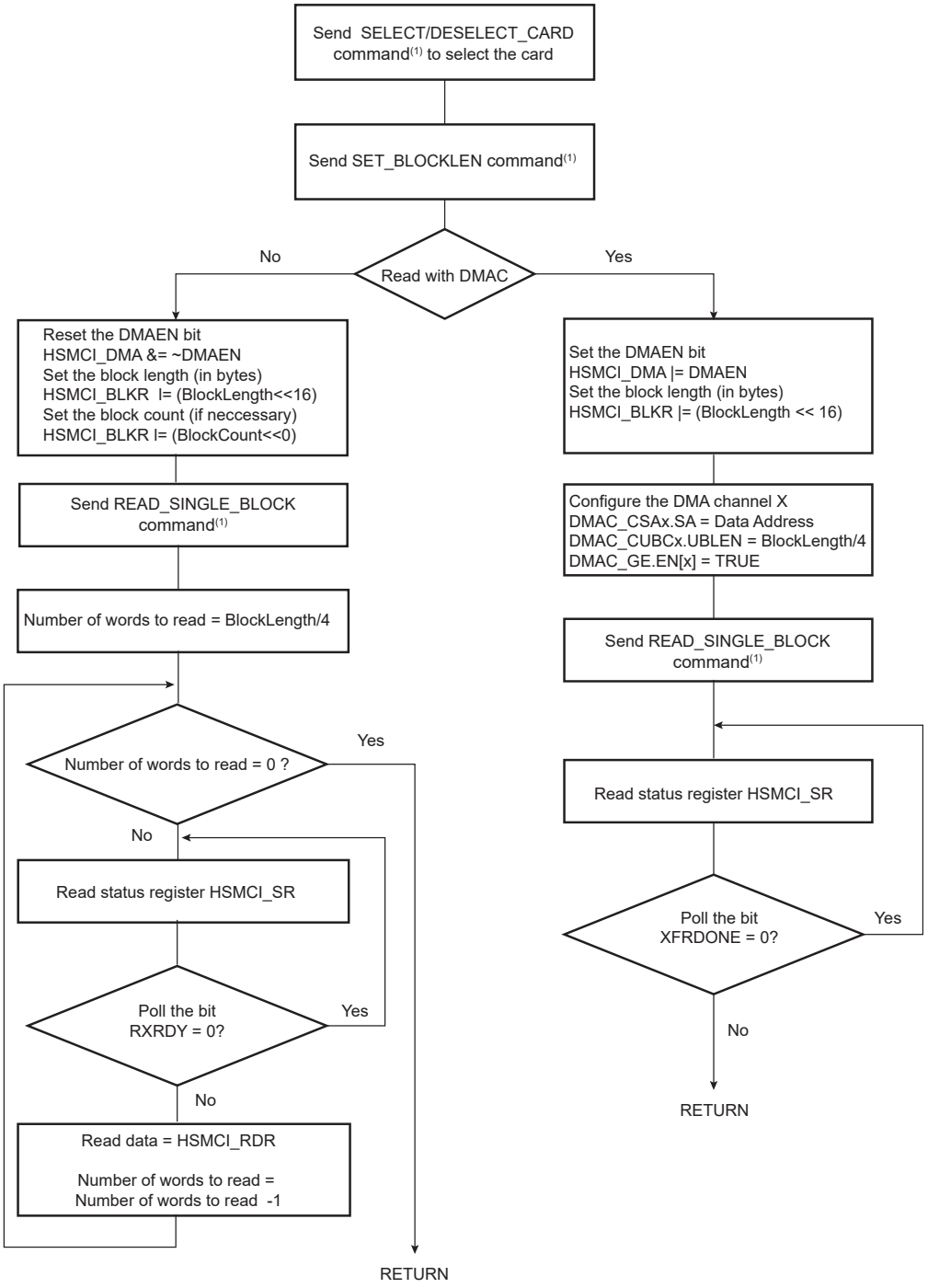
Bit 1 – RXOUTE Received OUT Data Interrupt

Value	Description
0	Cleared when USBHS_DEVEPTIDRx.RXOUTEC = 1. This disables the Received OUT Data interrupt (USBHS_DEVEPTISRx.RXOUTI).
1	Set when USBHS_DEVEPTIERx.RXOUTES = 1. This enables the Received OUT Data interrupt (USBHS_DEVEPTISRx.RXOUTI).

Bit 0 – TXINE Transmitted IN Data Interrupt

Value	Description
0	Cleared when USBHS_DEVEPTIDRx.TXINEC = 1. This disables the Transmitted IN Data interrupt (USBHS_DEVEPTISRx.TXINI).
1	Set when USBHS_DEVEPTIERx.TXINES = 1. This enables the Transmitted IN Data interrupt (USBHS_DEVEPTISRx.TXINI).

Figure 40-8. Read Functional Flow Diagram



Note 1: It is assumed that this command has been correctly sent (see the Command/Response Functional Flow Diagram).

40.8.4 Write Operation

In write operation, the HSMCI Mode Register (HSMCI_MR) is used to define the padding value when writing non-multiple block size. If the bit PADV is 0, then 0x00 value is used when padding data, otherwise 0xFF is used.

Value	Description
0	No boot operation error since the last read of HSMCI_SR
1	Corrupted Boot Acknowledge signal received since the last read of HSMCI_SR.

Bit 28 – ACKRCV Boot Operation Acknowledge Received (cleared on read)

Value	Description
0	No Boot acknowledge received since the last read of the HSMCI_SR.
1	A Boot acknowledge signal has been received since the last read of HSMCI_SR.

Bit 27 – XFRDONE Transfer Done flag

Value	Description
0	A transfer is in progress.
1	Command Register is ready to operate and the data bus is in the idle state.

Bit 26 – FIFOEMPTY FIFO empty flag

Value	Description
0	FIFO contains at least one byte.
1	FIFO is empty.

Bit 24 – BLKOVRE DMA Block Overrun Error (cleared on read)

Value	Description
0	No error.
1	A new block of data is received and the DMA controller has not started to move the current pending block, a block overrun is raised.

Bit 23 – CSTOE Completion Signal Time-out Error (cleared on read)

Value	Description
0	No error.
1	The completion signal time-out set by CSTOCYC and CSTOMUL in HSMCI_CSTOR has been exceeded.

Bit 22 – DTOE Data Time-out Error (cleared on read)

Value	Description
0	No error.
1	The data time-out set by DTOCYC and DTOMUL in HSMCI_DTOR has been exceeded.

Bit 21 – DCRCE Data CRC Error (cleared on read)

Value	Description
0	No error.
1	A CRC16 error has been detected in the last data block.

Bit 20 – RTOE Response Time-out Error (cleared by writing in HSMCI_CMDR)

Bit 3 – OVRES Overrun Error Status (cleared on read)
An overrun occurs when QSPI_RDR is loaded at least twice from the serializer since the last read of the QSPI_RDR.

Value	Description
0	No overrun has been detected since the last read of QSPI_SR.
1	At least one overrun error has occurred since the last read of QSPI_SR.

Bit 2 – TXEMPTY Transmission Registers Empty (cleared by writing QSPI_TDR)

Value	Description
0	As soon as data is written in QSPI_TDR.
1	QSPI_TDR and the internal shifter are empty. If a transfer delay has been defined, TXEMPTY is set after the completion of such delay.

Bit 1 – TDRE Transmit Data Register Empty (cleared by writing QSPI_TDR)
TDRE equals zero when the QSPI is disabled or at reset. The QSPI enable command sets this bit to one.

Value	Description
0	Data has been written to QSPI_TDR and not yet transferred to the serializer.
1	The last data written in the QSPI_TDR has been transferred to the serializer.

Bit 0 – RDRF Receive Data Register Full (cleared by reading QSPI_RDR)

Value	Description
0	No data has been received since the last read of QSPI_RDR.
1	Data has been received and the received data has been transferred from the serializer to QSPI_RDR since the last read of QSPI_RDR.

46.6.10.1 Mode of Operation

To configure the USART to act as a LON node, the value 0x9 must be written to US_MR.USART_MODE.

To avoid unpredictable behavior, any change of the LON node configuration must be preceded by a software reset of the transmitter and the receiver (except the initial node configuration after a hardware reset) and followed by a transmitter/receiver enable. See Section 7.10.2.

46.6.10.2 Receiver and Transmitter Control

See “Receiver and Transmitter Control”.

46.6.10.3 Character Transmission

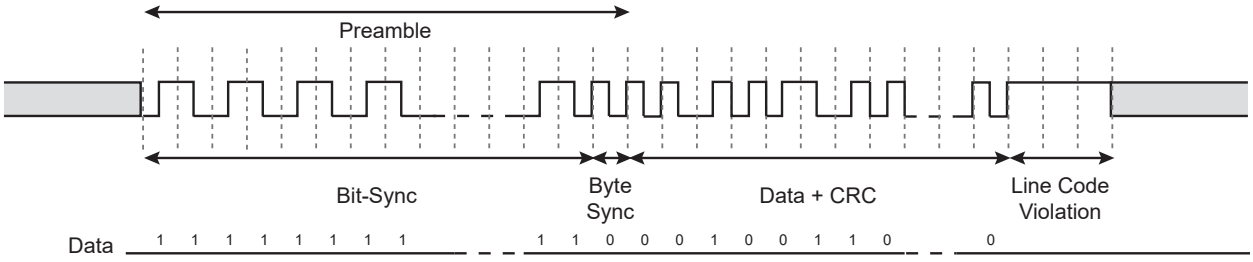
A LON frame is made up of a preamble, a data field (up to 256 bytes) and a 16-bit CRC field. The preamble and CRC fields are automatically generated and the LON node starts the transmission algorithm upon US_LONL2HDR register write. See “Sending A Frame”.

46.6.10.4 Character Reception

When receiving a LON frame, the Receive Holding register (US_RHR) is updated upon completed character reception and the RXRDY bit in the Status register rises. If a character is completed while the RXRDY bit is set, the OVRE (Overrun Error) bit is set. The LON preamble field is only used for synchronization, therefore only the Data and CRC fields are transmitted to the Receive Holding register (US_RHR). See “Receiving A Frame”.

46.6.10.5 LON Frame

Figure 46-55. LON Framing



46.6.10.5.1 Encoding / Decoding

The USART configured in LON mode encodes transmitted data and decodes received data using differential Manchester encoding. In differential Manchester encoding, a ‘1’ bit is indicated by making the first half of the signal equal the last half of the previous bit's signal (no transition at the start of the bit-time). A ‘0’ bit is indicated by making the first half of the signal opposite to the last half of the previous bit's signal (a zero bit is indicated by a transition at the beginning of the bit-time). As is the case with normal Manchester encoding, missing transition at the middle of bit-time represents a Manchester code violation.

US_MAN.RXIDLEV informs the USART of the receiver line idle state value (receiver line inactive) thus ensuring higher reliability of preamble synchronization. By default, RXIDLEV is set to ‘1’ (receiver line is at level 1 when there is no activity).

Differential Manchester encoding is polarity insensitive.

Figure 46-56. LON PPDU

46.6.10.5.2 Preamble Transmission

Each LON frame begins with a preamble of variable length which consists of a bit-sync field and a byte-sync field. The LONPL field of the USART LON Preamble register (US_LONPR) defines the preamble length. Note that preamble length of ‘0’ is not allowed.

48.7.24 MIF Address Register

Name: MLB_MADR
Offset: 0x0E4
Reset: 0x00000000
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
	WNR	TB						
Access								
Reset	0	0						

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

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

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

Bit 31 – WNR Write-Not-Read Selection

Value	Description
0	Read
1	Write

Bit 30 – TB Target Location Bit
0 (CTR): Selects CTR

1 (DBR): Selects DBR

Bits 13:0 – ADDR[13:0] CTR or DBR Address
CTR address of 128-bit entry or

DBR address of 8-bit entry - bits[7:0]

Bits 21:16 – F1PI[5:0] Receive FIFO 1 Put Index
Receive FIFO 1 write index pointer, range 0 to 63.

Bits 13:8 – F1GI[5:0] Receive FIFO 1 Get Index
Receive FIFO 1 read index pointer, range 0 to 63.

Bits 6:0 – F1FL[6:0] Receive FIFO 1 Fill Level
Number of elements stored in Receive FIFO 1, range 0 to 64.

Offset	Name	Bit Pos.								
		31:24								CVM
0x0144	PWM_CMPVUPD1	7:0	CVUPD[7:0]							
		15:8	CVUPD[15:8]							
		23:16	CVUPD[23:16]							
		31:24								CVMUPD
0x0148	PWM_CMPM1	7:0	CTR[3:0]							CEN
		15:8	CPRCNT[3:0]				CPR[3:0]			
		23:16	CUPRCNT[3:0]				CUPR[3:0]			
		31:24								
0x014C	PWM_CMPMUPD1	7:0	CTRUPD[3:0]							CENUPD
		15:8					CPRUPD[3:0]			
		23:16					CUPRUPD[3:0]			
		31:24								
0x0150	PWM_CMPV2	7:0	CV[7:0]							
		15:8	CV[15:8]							
		23:16	CV[23:16]							
		31:24								CVM
0x0154	PWM_CMPVUPD2	7:0	CVUPD[7:0]							
		15:8	CVUPD[15:8]							
		23:16	CVUPD[23:16]							
		31:24								CVMUPD
0x0158	PWM_CMPM2	7:0	CTR[3:0]							CEN
		15:8	CPRCNT[3:0]				CPR[3:0]			
		23:16	CUPRCNT[3:0]				CUPR[3:0]			
		31:24								
0x015C	PWM_CMPMUPD2	7:0	CTRUPD[3:0]							CENUPD
		15:8					CPRUPD[3:0]			
		23:16					CUPRUPD[3:0]			
		31:24								
0x0160	PWM_CMPV3	7:0	CV[7:0]							
		15:8	CV[15:8]							
		23:16	CV[23:16]							
		31:24								CVM
0x0164	PWM_CMPVUPD3	7:0	CVUPD[7:0]							
		15:8	CVUPD[15:8]							
		23:16	CVUPD[23:16]							
		31:24								CVMUPD
0x0168	PWM_CMPM3	7:0	CTR[3:0]							CEN
		15:8	CPRCNT[3:0]				CPR[3:0]			
		23:16	CUPRCNT[3:0]				CUPR[3:0]			
		31:24								
0x016C	PWM_CMPMUPD3	7:0	CTRUPD[3:0]							CENUPD
		15:8					CPRUPD[3:0]			
		23:16					CUPRUPD[3:0]			
		31:24								
0x0170	PWM_CMPV4	7:0	CV[7:0]							

Bit 0 – WRDY Write Ready for Synchronous Channels Update

Value	Description
0	New duty-cycle and dead-time values for the synchronous channels cannot be written.
1	New duty-cycle and dead-time values for the synchronous channels can be written.

6. Fill AES_IDATARx.IDATA with the message to process according to the SMOD configuration used. If Manual Mode or Auto Mode is used, the DATRDY bit indicates when the data have been processed (however, no output data are generated when processing *AAD*).
7. Wait for TAGRDY to be set (use interrupt if needed), then read AES_TAGRx.TAG to obtain the authentication tag of the message.

57.4.4.3.2 Processing a Complete Message without Tag Generation

Processing a message without generating the Tag can be used to customize the Tag generation, or to process a fragmented message. To manually generate the GCM Tag, see Manual GCM Tag Generation.

To process a complete message without Tag generation, the sequence is as follows:

1. Set AES_MR.OPMOD to GCM and AES_MR.GTAGEN to '0'.
2. Set the AES Key Register and wait until AES_ISR.DATRDY is set (GCM hash subkey generation complete); use interrupt if needed. After the GCM hash subkey generation is complete the GCM hash subkey can be read or overwritten with specific value in AES_GCMHRx. See Key Writing and Automatic Hash Subkey Calculation.
3. Calculate the J0 value as described in NIST documentation J_0 = IV || 0^{31} || 1 when len(IV) = 96 and J_0 = $GHASH_H$(IV || 0^{s+64} || [len(IV)]64) if len(IV) ≠ 96. See Processing a Message with only AAD (GHASHH) for J_0 generation example when len(IV) ≠ 96.
4. Set AES_IVRx.IV with inc32(J_0) (J_0 + 1 on 32 bits).
5. Configure AES_AADLENR.AADLEN and AES_CLENR.CLEN.
6. Fill AES_IDATARx.IDATA with the message to process according to the SMOD configuration used. If Manual Mode or Auto Mode is used, the DATRDY bit indicates when the data have been processed (however, no output data are generated when processing *AAD*).
7. Make sure the last output data have been read if AES_CLENR.CLEN ≠ 0 (or wait for DATRDY), then read AES_GHASHRx.GHASH to obtain the hash value after the last processed data.

57.4.4.3.3 Processing a Fragmented Message without Tag Generation

If needed, a message can be processed by fragments, in such case automatic GCM Tag generation is not supported.

To process a message by fragments, the sequence is as follows:

- First fragment:

1. Set AES_MR.OPMOD to GCM and AES_MR.GTAGEN to '0'.
2. Set the AES Key Register and wait for AES_ISR.DATRDY to be set (GCM hash subkey generation complete); use interrupt if needed. After the GCM hash subkey generation is complete the GCM hash subkey can be read or overwritten with specific value in AES_GCMHRx. See Key Writing and Automatic Hash Subkey Calculation.
3. Calculate the J_0 value as described in NIST documentation J_0 = IV || 0^{31} || 1 when len(IV) = 96 and J_0 = $GHASH_H$(IV || 0^{s+64} || [len(IV)]64) if len(IV) ≠ 96. See Processing a Message with only AAD (GHASHH) for J_0 generation example when len(IV) ≠ 96.
4. Set AES_IVRx.IV with inc32(J_0) (J_0 + 1 on 32 bits).
5. Configure AES_AADLENR.AADLEN and AES_CLENR.CLEN according to the length of the first fragment, or set the fields with the full message length (both configurations work).
6. Fill AES_IDATARx.IDATA with the first fragment of the message to process (aligned on 16-byte boundary) according to the SMOD configuration used. If Manual Mode or Auto Mode is used the DATRDY bit indicates when the data have been processed (however, no output data are generated when processing *AAD*).

Signal Name	Recommended Pin Connection	Description
		WARNING Power up and power down sequences given in the "Power Considerations" chapter must be respected.
VDDIO	Decoupling/filtering capacitors (100 nF) (1, 2)	Powers the Peripheral I/O lines (Input/Output Buffers), backup part, 1 Kbytes of Backup SRAM, 32 kHz crystal oscillator, oscillator pads Decoupling/filtering capacitors must be added to improve startup stability and reduce source voltage drop. Supply ripple must not exceed 30 mVrms for 10 kHz to 10 MHz range. **WARNING** VDDIN and VDDIO must have the same level and must always be higher than VDDCORE. **WARNING** Powerup and powerdown sequences given in the "Power Considerations" chapter must be respected.
VDDUTMII	Decoupling capacitor (100 nF) (1) (2)	Powers the USB transceiver interface. Must be connected to VDDIO. For USB operations, VDDUTMII and VDDIO voltage ranges must be from 3.0V to 3.6V. Must always be connected even if the USB is not used. Decoupling/filtering capacitors must be added to improve startup stability and reduce source voltage drop. Supply ripple must not exceed 20 mVrms for 10 kHz to 10 MHz range.
VDDPLLUSB	Decoupling/filtering RLC circuit (1)	Powers the UTMI PLL and the 3 to 20 MHz oscillator. For USB operations, VDDPLLUSB should be between 3.0V and 3.6V. The VDDPLLUSB power supply pin draws small current, but it is noise sensitive. Care must be taken in VDDPLLUSB power supply routing, decoupling and also on bypass capacitors. Supply ripple must not exceed 10 mVrms for 10 kHz to 10 MHz range.
VDDOUT	Left unconnected	Voltage Regulator Output
VDDCORE	Decoupling capacitor (100 nF) (1) (2)	Powers the core, embedded memories and peripherals.

Date	Comments
	Section 23. “Supply Controller (SUPC)” Section 23.4.3 “Core Voltage Regulator Control/Backup Low-power Mode”: removed information on Backup mode entry via WFE. Corrected ONREG polarity. Figure 23-6 “Raising the VDDIO Power Supply”: removed Flash frequency in Note. Section 23.4.10 “Register Write Protection”: deleted Section 23.5.7 “Supply Controller Wakeup Inputs Register” from list of write-protected registers. Added Section 23.5.9 “System Controller Write Protection Mode Register”.
cont’d	
01-June-16	Section 24. “Watchdog Timer (WDT)” Section 24.4 “Functional Description”: Added detail on LOCKMR bit in paragraph starting “WDT_MR can be written...” Modified paragraph starting with “The reload of the WDT must occur...”. Table 24-1 “Register Mapping”: modified Access for Section 24.5.2 “Watchdog Timer Mode Register”. Section 24.5.1 “Watchdog Timer Control Register”: LOCKMR bit now at index 4 (was ‘reserved’). Section 24.5.2 “Watchdog Timer Mode Register”: modified access and updated Note (1).
	Section 25. “Reinforced Safety Watchdog Timer (RSWDT)” Updated Figure 25-1 “Reinforced Safety Watchdog Timer Block Diagram”.
	Section 26. “Reset Controller (RSTC)” ‘Slow crystal’ changed to ‘32.768 kHz’ throughout. Updated figures: - Figure 26-1 “Reset Controller Block Diagram” - Figure 26-3 “General Reset Timing Diagram” - Figure 26-4 “Watchdog Reset Timing Diagram” - Figure 26-5 “Software Reset Timing Diagram”” - Figure 26-6 “User Reset Timing Diagram” Added Note to Section 26.4.1 “Overview”. Updated Section 26.4.3.3 “Watchdog Reset”: Replaced “is set” with “is written to 1” and “is reset” with “is written to 0”. Section 26.5.3 “RSTC Mode Register”: updated URSTIEN description.
	Section 27. “Real-time Clock (RTC)” Updated Section 27.5.6 “Updating Time/Calendar”.
	Section 29. “SDRAM Controller (SDRAMC)” Section 29.5.1 “SDRAM Device Initialization”: updated first step.
cont’d	