Welcome to E-XFL.COM

What is "Embedded - Microcontrollers"?

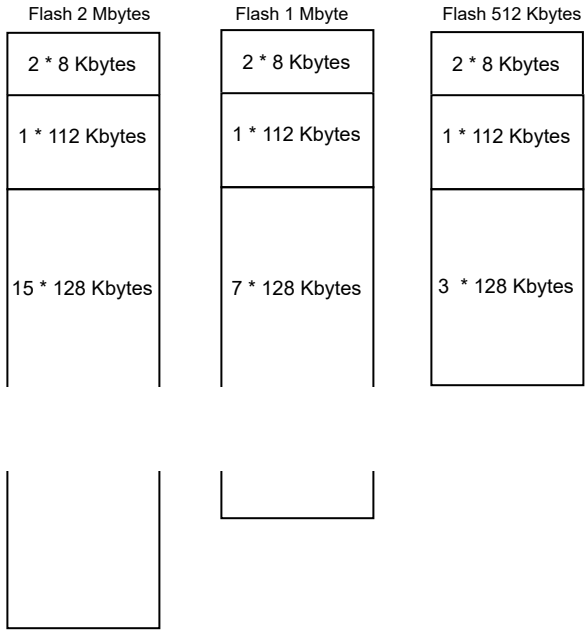
"Embedded - Microcontrollers" refer to small, integrated circuits designed to perform specific tasks within larger systems. These microcontrollers are essentially compact computers on a single chip, containing a processor core, memory, and programmable input/output peripherals. They are called "embedded" 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	EBI/EMI, 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	1MB (1M x 8)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	384K x 8
Voltage - Supply (Vcc/Vdd)	1.62V ~ 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	144-LFBGA
Supplier Device Package	144-LFBGA (10x10)
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/atsams70q20b-cfn

The figure below illustrates the organization of the Flash depending on its size.

Figure 11-3. Flash Size



Erasing the memory can be performed:

- by block of 8 Kbytes
- by sector of 128 Kbytes
- by 512-byte page for up to 8 Kbytes within a specific small sector
- Chip Erase

The memory has one additional reprogrammable page that can be used as page signature by the user. It is accessible through specific modes, for erase, write and read operations. Erase pin assertion will not erase the User Signature page.

Erase memory by page is possible only in a sector of 8 Kbytes.

EWP and EWPL commands can be only used in 8-Kbyte sectors.

11.1.5.2 Enhanced Embedded Flash Controller

Each Enhanced Embedded Flash Controller manages accesses performed by the masters of the system. It enables reading the Flash and writing the write buffer. It also contains a User Interface, mapped on the APB.

The Enhanced Embedded Flash Controller ensures the interface of the Flash block.

It manages the programming, erasing, locking and unlocking sequences of the Flash using a full set of commands.

One of the commands returns the embedded Flash descriptor definition that informs the system about the Flash organization, thus making the software generic.

11.1.5.3 Flash Speed

The user must set the number of wait states depending on the system frequency.

For more details, refer to Embedded Flash Characteristics.

Table 22-2. Flash Descriptor Definition

Symbol	Word Index	Description
FL_ID	0	Flash interface description
FL_SIZE	1	Flash size in bytes
FL_PAGE_SIZE	2	Page size in bytes
FL_NB_PLANE	3	Number of planes
FL_PLANE[0]	4	Number of bytes in the plane
FL_NB_LOCK	4 + FL_NB_PLANE	Number of lock bits. A bit is associated with a lock region. A lock bit is used to prevent write or erase operations in the lock region.
FL_LOCK[0]	4 + FL_NB_PLANE + 1	Number of bytes in the first lock region

22.4.3.2 Write Commands

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 written with ones.

Several commands are used to program the Flash.

Only 0 values can be programmed using Flash technology; 1 is the erased value. In order to program words in a page, the page must first be erased. Commands are available to erase the full memory plane or a given number of pages. With the EWP and EWPL commands, a page erase is done automatically before a page programming.

After programming, the page (the entire lock region) can be locked to prevent miscellaneous write or erase sequences. The lock bit can be automatically set after page programming using WPL or EWPL commands.

Data to be programmed in the Flash must be written in an internal latch buffer before writing the programming command in EEFC_FCR. Data can be written at their final destination address, as the latch buffer is mapped into the Flash memory address space and wraps around within this Flash address space.

Byte and half-word AHB accesses to the latch buffer are not allowed. Only 32-bit word accesses are supported.

32-bit words must be written continuously, in either ascending or descending order. Writing the latch buffer in a random order is not permitted. This prevents mapping a C-code structure to the latch buffer and accessing the data of the structure in any order. It is instead recommended to fill in a C-code structure in SRAM and copy it in the latch buffer in a continuous order.

Write operations in the latch buffer are performed with the number of wait states programmed for reading the Flash.

The latch buffer is automatically re-initialized, i.e., written with logical ‘1’, after execution of each programming command.

The programming sequence is the following:

1. Write the data to be programmed in the latch buffer.
2. Write the programming command in EEFC_FCR. This automatically clears the bit EEFC_FSR.FRDY.

Until bodcore_in is deactivated, the vddcore_nreset signal remains active.

23.4.8 Controlling the SRAM Power Supply

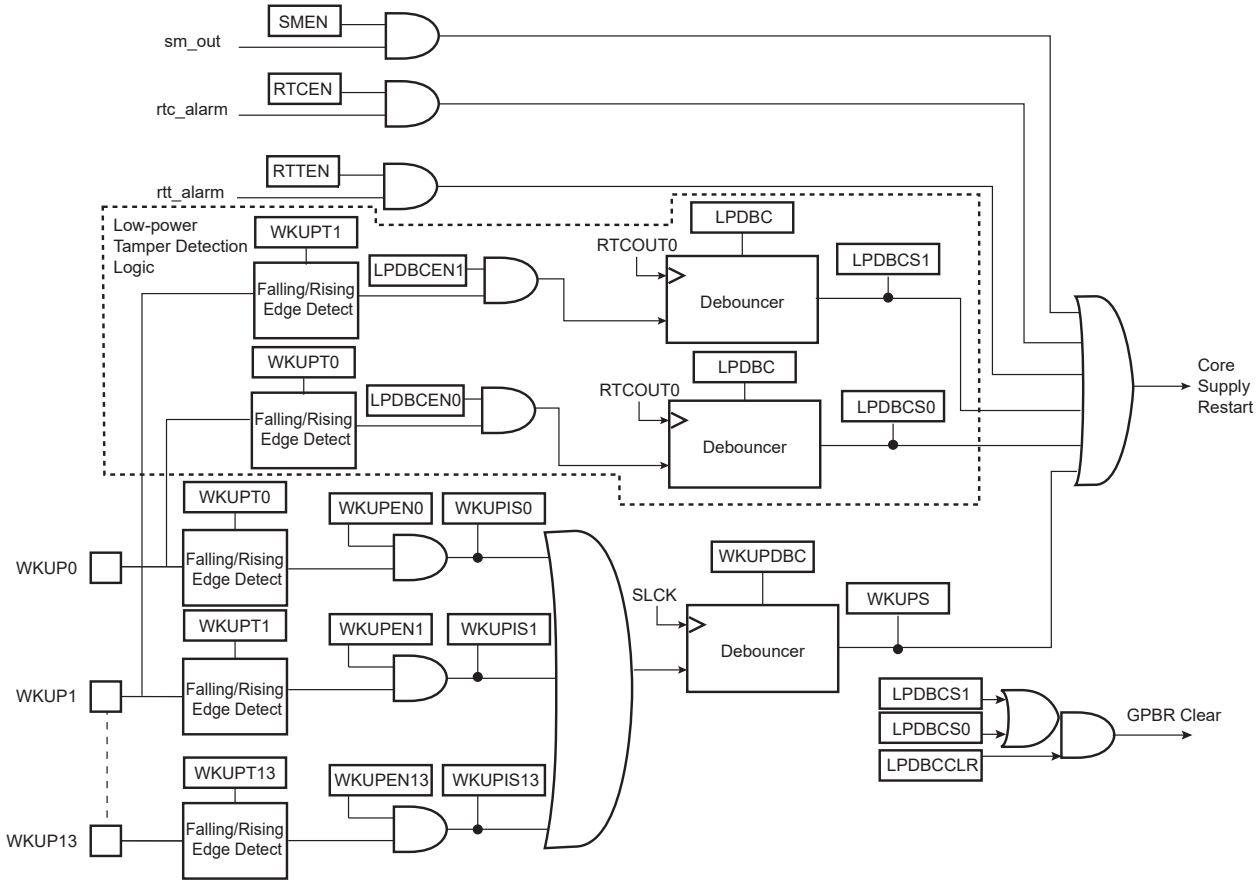
The SUPC can be used to switch on or off the power supply of the backup SRAM by opening or closing the SRAM power switch. This power switch is controlled by SUPC_MR.BKUPRETON. However, the battery backup SRAM is automatically switched on when the core power supply is enabled, as the processor requires the SRAM as data memory space.

- If SUPC_MR.BKUPRETON is written to ‘1’, there is no immediate effect, but the SRAM will be left powered when the SUPC enters Backup mode, thus retaining its content.
- If SUPC_MR.BKUPRETON is written to ‘0’, there is no immediate effect, but the SRAM will be switched off when the SUPC enters Backup mode. The SRAM is automatically switched on when Backup mode is exited.

23.4.9 Wakeup Sources

The wakeup events allow the device to exit Backup mode. When a wakeup event is detected, the SUPC performs a sequence that automatically reenables the core power supply.

Figure 23-7. Wakeup Sources



23.4.9.1 Wakeup Inputs

The wakeup inputs, WKUPx, can be programmed to perform a wakeup of the core power supply. Each input can be enabled by writing a ‘1’ to the corresponding bit, WKUPENx, in the Wakeup Inputs register (SUPC_WUIR). The wakeup level can be selected with the corresponding polarity bit, WKUPTx, also located in SUPC_WUIR.

32.6.1.52 PIO Parallel Capture Interrupt Disable Register

Name: PIO_PCIDR
Offset: 0x0158
Property: Write-only

The following configuration 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
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
Access					RXBUFF	ENDRX	OVRE	DRDY
Reset								

Bit 3 – RXBUFF Reception Buffer Full Interrupt Disable

Bit 2 – ENDRX End of Reception Transfer Interrupt Disable

Bit 1 – OVRE Parallel Capture Mode Overrun Error Interrupt Disable

Bit 0 – DRDY Parallel Capture Mode Data Ready Interrupt Disable

32.6.1.55 PIO Parallel Capture Reception Holding Register

Name: PIO_PCRHR
Offset: 0x0164
Reset: 0x00000000
Property: Read-only

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

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

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

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

Bits 31:0 – RDATA[31:0] Parallel Capture Mode Reception Data
If DSIZE = 0 in PIO_PCMR, only the 8 LSBs of RDATA are useful.

If DSIZE = 1 in PIO_PCMR, only the 16 LSBs of RDATA are useful.

37.6.10 ISI Control Register

Name: ISI_CR
Offset: 0x24
Reset: –
Property: Write-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
								ISI_CDC
Access								W
Reset								–

Bit	7	6	5	4	3	2	1	0
						ISI_SRST	ISI_DIS	ISI_EN
Access						W	W	W
Reset						–	–	–

Bit 8 – ISI_CDC ISI Codec Request
Write a one to this bit to enable the codec datapath and capture a full resolution frame. A new request cannot be taken into account while CDC_PND bit is active in the ISI_SR.

Bit 2 – ISI_SRST ISI Software Reset Request
Write a one to this bit to request a software reset of the module. Software must poll the SRST bit in the ISI_SR to verify that the software request command has terminated.

Bit 1 – ISI_DIS ISI Module Disable Request
Write a one to this bit to disable the module. If both ISI_EN and ISI_DIS are asserted at the same time, the disable request is not taken into account. Software must poll the DIS_DONE bit in the ISI_SR to verify that the command has successfully completed.

Bit 0 – ISI_EN ISI Module Enable Request
Write a one to this bit to enable the module. Software must poll the ENABLE bit in the ISI_SR to verify that the command has successfully completed.

38.8.59 GMAC Octets Received High Register

Name: GMAC_ORHI
Offset: 0x154
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
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

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 15:0 – RXO[15:0] Received Octets
Received octets in frame without errors [47:32]. 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.

- The host starts a SETUP transaction with a SET_ADDRESS (addr) request.
- The user writes this address to the USB Address (USBHS_DEVCTRL.UADD) field, and writes a zero to the Address Enable (USBHS_DEVCTRL.ADDEN) bit, so the actual address is still 0.
- The user sends a zero-length IN packet from the control endpoint.
- The user enables the recorded USB device address by writing a one to USBHS_DEVCTRL.ADDEN.

Once the USB device address is configured, the controller filters the packets to accept only those targeting the address stored in USBHS_DEVCTRL.UADD.

USBHS_DEVCTRL.UADD and USBHS_DEVCTRL.ADDEN must not be written all at once.

USBHS_DEVCTRL.UADD and USBHS_DEVCTRL.ADDEN are cleared:

- on a hardware reset,
- when the USBHS is disabled (USBHS_CTRL.USBE = 0),
- when a USB reset is detected.

When USBHS_DEVCTRL.UADD or USBHS_DEVCTRL.ADDEN is cleared, the default device address 0 is used.

39.5.2.7 Suspend and Wakeup

When an idle USB bus state has been detected for 3 ms, the controller sets the Suspend (USBHS_DEVISR.SUSP) interrupt bit. The user may then write a one to the USBHS_CTRL.FRZCLK bit to reduce power consumption.

To recover from the Suspend mode, the user should wait for the Wakeup (USBHS_DEVISR.WAKEUP) interrupt bit, which is set when a non-idle event is detected, then write a zero to USBHS_CTRL.FRZCLK.

As the USBHS_DEVISR.WAKEUP interrupt bit is set when a non-idle event is detected, it can occur whether the controller is in the Suspend mode or not. The USBHS_DEVISR.SUSP and USBHS_DEVISR.WAKEUP interrupts are thus independent, except that one bit is cleared when the other is set.

39.5.2.8 Detach

The reset value of the USBHS_DEVCTRL.DETACH bit is one.

It is possible to initiate a device re-enumeration by simply writing a one, and then a zero, to USBHS_DEVCTRL.DETACH.

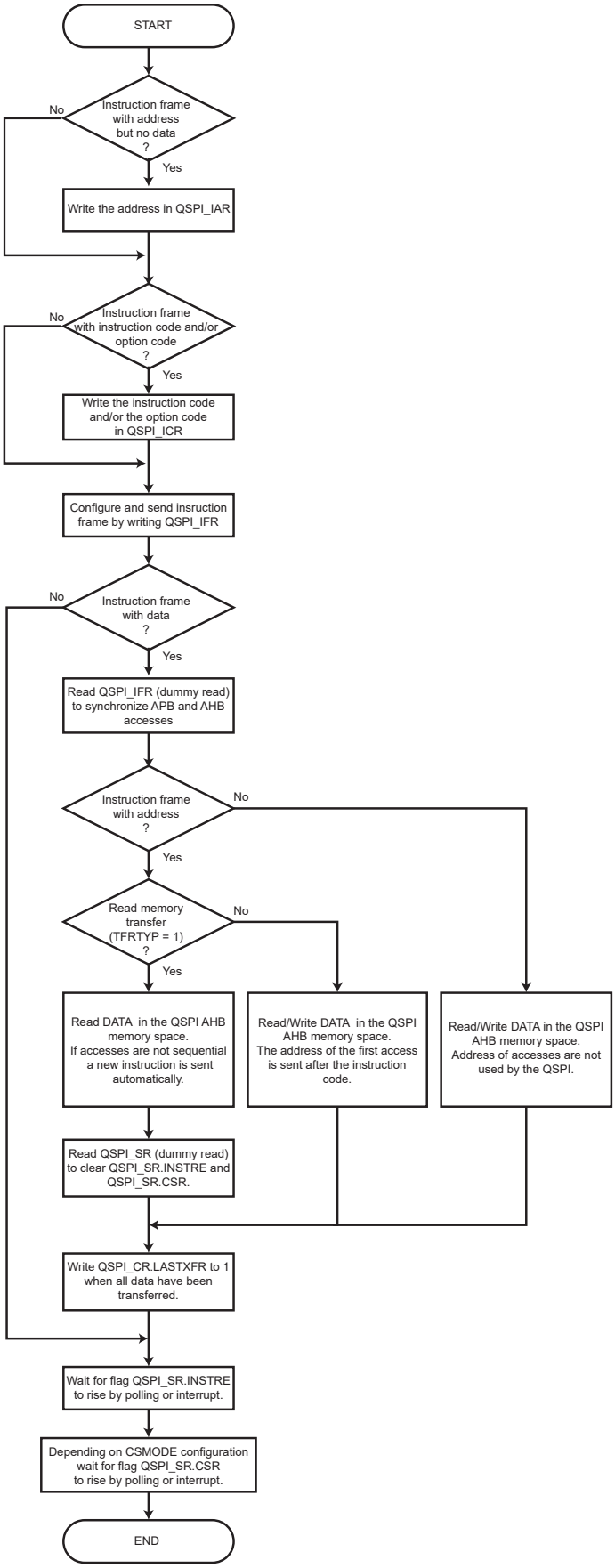
USBHS_DEVCTRL.DETACH acts on the pull-up connections of the D+ and D- pads. See “Device Mode” for further details.

39.5.2.9 Remote Wakeup

The Remote Wakeup request (also known as Upstream Resume) is the only one the device may send without a host invitation, assuming a host command allowing the device to send such a request was previously issued. The sequence is the following:

1. The USBHS must have detected a “Suspend” state on the bus, i.e., the Remote Wakeup request can only be sent after a USBHS_DEVISR.SUSP interrupt has been set.
2. The user writes a one to the Remote Wakeup (USBHS_DEVCTRL.RMWKUP) bit to send an upstream resume to the host for a remote wakeup. This will automatically be done by the controller after 5 ms of inactivity on the USB bus.
3. When the controller sends the upstream resume, the Upstream Resume (USBHS_DEVISR.UPRSM) interrupt is set and USBHS_DEVISR.SUSP is cleared.

Figure 42-9. Instruction Transmission Flow Diagram



1: When both holding register and internal shifter are empty and STOP condition has been sent.

TXCOMP behavior in Master mode can be seen in Master Write with One-Byte Internal Address and Multiple Data Bytes and in Master Read with Multiple Data Bytes.

- TXCOMP used in Slave mode:

0: As soon as a START is detected.

1: After a STOP or a REPEATED START + an address different from SADR is detected.

TXCOMP behavior in Slave mode can be seen in Clock Stretching in Read Mode, Clock Stretching in Write Mode, Repeated Start and Reversal from Read Mode to Write Mode and Repeated Start and Reversal from Write Mode to Read Mode.

46.7.42 USART LON Priority Register

Name: US_LONPRIO
Offset: 0x007C
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
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
		NPS[6:0]						
Access								
Reset		0	0	0	0	0	0	0

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

Bits 14:8 – NPS[6:0] LON Node Priority Slot

Value	Description
0-127	Node priority slot.

Bits 6:0 – PSNB[6:0] LON Priority Slot Number

Value	Description
0-127	Number of priority slots in the LON network configuration.

47.6.11 UART Write Protection Mode Register

Name: UART_WPMR
Offset: 0xE4
Reset: 0x00000000
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
	WPKEY[23:16]							
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	WPKEY[15:8]							
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	WPKEY[7:0]							
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
								WPEN
Access								R/W
Reset								0

Bits 31:8 – WPKEY[23:0] Write Protection Key

Value	Name	Description
0x554152	PASSWD	Writing any other value in this field aborts the write operation. Always reads as 0.

Bit 0 – WPEN Write Protection Enable
See Register Write Protection for the list of registers that can be protected.

Value	Description
0	Disables the write protection if WPKEY corresponds to 0x554152 (UART in ASCII).
1	Enables the write protection if WPKEY corresponds to 0x554152 (UART in ASCII).

2. Only valid for DMA pointers associated with the MediaLB block (Not valid for HBI block related pointers).

Isochronous Channel Descriptors

The format and field definitions for an isochronous CDT entry are shown in Table 48-16 and Table 48-17, respectively.

Table 48-16. Isochronous CDT Entry Format

Bit Offset	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	Reserved															
16	Reserved															
32	Reserved							BS[8:0]								
48	Reserved															
64	WSTS[2:0]			WPTR[12:0]												
80	RSTS[2:0]			RPTR[12:0]												
96	Reserved			BD[12:0]												
112	BF	rsvd	BA[13:0]													

Table 48-17. Isochronous CDT Entry Field Definitions

Field	Description	Details	Accessibility
BA	Buffer Base Address	- BA can start at any byte in the 16k DBR	r,w
BD	Buffer Depth	- BD = size of buffer in bytes - 1 - Buffer end address = BA + BD - Isochronous buffers must be large enough to hold at least 3 blocks (packets) of data - Buffer depth must be a integer multiple of blocks	r,w
BF	Buffer Full	- Software initializes to zero, hardware updates - DMA write hardware sets BF when the buffer is full - DMA read hardware clears BF when the buffer is empty - BF is valid only when the buffer is full or empty, otherwise ignore	r,w,u (1)
BS	Block Size	- BS defines when to begin the DMA to the data buffer - BS = buffer block size in bytes - 1 - For Rx channels, the DMA writes start when the number of empty bytes (SPACE) in the data buffer ≥ the block size - For Tx channels, the DMA reads start when the number of valid bytes (VALID) in the data buffer ≥ the block size	r,w,u (1)
RPTR	Read Pointer	- Software initializes to zero, hardware updates	r,w,u (1)

Table 49-9. Tx Event FIFO Element

	31			24	23						16	15	8	7	0
E0	ESI	XTD	RTR	ID[28:0]											
E1	MM[7:0]				ET [1:0]	FDF	BRS	DLC[3:0]				TXTS[15:0]			

• E0 Bit 31 ESI: Error State Indicator

0: Transmitting node is error active.

1: Transmitting node is error passive.

• E0 Bit 30 XTD: Extended Identifier

0: 11-bit standard identifier.

1: 29-bit extended identifier.

• E0 Bit 29 RTR: Remote Transmission Request

0: Data frame transmitted.

1: Remote frame transmitted.

• E0 Bits 28:0 ID[28:0]: Identifier

Standard or extended identifier depending on bit XTD. A standard identifier is stored into ID[28:18].

• E1 Bits 31:24 MM[7:0]: Message Marker

Copied from Tx Buffer into Tx Event FIFO element for identification of Tx message status.

• E1 Bit 23:22 ET[1:0]: Event Type

0: Reserved

1: Tx event

2: Transmission in spite of cancellation (always set for transmissions in DAR mode)

3: Reserved

• E1 Bit 21 FDF: FD Format

0: Standard frame format.

1: CAN FD frame format (new DLC-coding and CRC).

• E1 Bit 20 BRS: Bit Rate Switch

0: Frame transmitted without bit rate switching.

1: Frame transmitted with bit rate switching.

• E1 Bits 19:16 DLC[3:0]: Data Length Code

0-8: CAN + CAN FD: frame with 0-8 data bytes transmitted.

9-15: CAN: frame with 8 data bytes transmitted.

9-15: CAN FD: frame with 12/16/20/24/32/48/64 data bytes transmitted

• E1 Bits 15:0 TXTS[15:0]: Tx Timestamp

- SYNC: Each channel has a synchronization signal SYNC. When asserted, this signal has the same effect as a software trigger. The SYNC signals of all channels are asserted simultaneously by writing TC_BCR with SYNC set.
- Compare RC Trigger: RC is implemented in each channel and can provide a trigger when the counter value matches the RC value if TC_CMRx.CPCTRG is set .

The channel can also be configured to have an external trigger. In Capture mode, the external trigger signal can be selected between TIOAx and TIOBx. In Waveform mode, an external event can be programmed on one of the following signals: TIOBx, XC0, XC1 or XC2. This external event can then be programmed to perform a trigger by setting TC_CMRx.ENETRG.

If an external trigger is used, the duration of the pulses must be longer than the peripheral clock period in order to be detected.

50.6.7 Capture Mode

Capture mode is entered by clearing TC_CMRx.WAVE.

Capture mode allows the TC channel to perform measurements such as pulse timing, frequency, period, duty cycle and phase on TIOAx and TIOBx signals which are considered as inputs.

The figure Figure 50-6 shows the configuration of the TC channel when programmed in Capture mode.

50.6.8 Capture Registers A and B

Registers A and B (TC_RA and TC_RB) are used as capture registers. They can be loaded with the counter value when a programmable event occurs on the signal TIOAx.

TC_CMRx.LDRA defines the TIOAx selected edge for the loading of TC_RA, and TC_CMRx.LDRB defines the TIOAx selected edge for the loading of TC_RB.

The subsampling ratio defined by TC_CMRx.SBSMPLR is applied to these selected edges, so that the loading of Register A and Register B occurs once every 1, 2, 4, 8 or 16 selected edges.

TC_RA is loaded only if it has not been loaded since the last trigger or if TC_RB has been loaded since the last loading of TC_RA.

TC_RB is loaded only if TC_RA has been loaded since the last trigger or the last loading of TC_RB.

Loading TC_RA or TC_RB before the read of the last value loaded sets TC_SR.LOVRS. In this case, the old value is overwritten.

When DMA is used (on channel 0), the Register AB (TC_RAB) address must be configured as source address of the transfer. TC_RAB provides the next unread value from TC_RA and TC_RB. It may be read by the DMA after a request has been triggered upon loading TC_RA or TC_RB.

50.6.9 Transfer with DMAC in Capture Mode

The DMAC can perform access from the TC to system memory in Capture mode only.

The following figure illustrates how TC_RA and TC_RB can be loaded in the system memory without processor intervention.

52.4 Signal Description

Table 52-1. AFEC Signal Description

Pin Name	Description
VREFP	Reference voltage
VREFN	Reference voltage
AFE_AD0—AFE_AD11(1)	Analog input channels
AFE_ADTRG	External trigger

Note:

1. AFE_AD11 is not an actual pin but is connected to a temperature sensor.

52.5 Product Dependencies

52.5.1 I/O Lines

The digital input AFE_ADTRG is multiplexed with digital functions on the I/O line and the selection of AFE_ADTRG is made using the PIO Controller.

The analog inputs AFE_ADx are multiplexed with digital functions on the I/O lines. AFE_ADx inputs are selected as inputs of the AFEC when writing a one in the corresponding CHx bit of AFEC_CHER and the digital functions are not selected.

52.5.2 Power Management

The AFEC is not continuously clocked. The programmer must first enable the AFEC peripheral clock in the Power Management Controller (PMC) before using the AFEC. However, if the application does not require AFEC operations, the peripheral clock can be stopped when not needed and restarted when necessary.

When the AFEC is in Sleep mode, the peripheral clock must always be enabled.

52.5.3 Interrupt Sources

The AFEC interrupt line is connected on one of the internal sources of the Interrupt Controller. Using the AFEC interrupt requires the interrupt controller to be programmed first.

52.5.4 Temperature Sensor

The temperature sensor is connected to Channel 11 of the AFEC.

The temperature sensor provides an output voltage V_T that is proportional to the absolute temperature (PTAT).

52.5.5 Timer Triggers

Timer Counters may or may not be used as hardware triggers depending on user requirements. Thus, some or all of the timer counters may be unconnected.

52.5.6 PWM Event Lines

PWM event lines may or may not be used as hardware triggers, depending on user requirements.

55.6.9 ICM Descriptor Area Start Address Register

Name: ICM_DSCR
Offset: 0x30
Reset: 0x00000000
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
	DASA[25:18]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	DASA[17:10]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	DASA[9:2]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

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

Bits 31:6 – DASA[25:0] Descriptor Area Start Address
The start address is a multiple of the total size of the data structure (64 bytes).

3. Configure AES_AADLENR.AADLEN to 0x10 (16 bytes) and AES_CLENR.CLEN to '0'. This will allow running a single $GHASH_H$ on a 16-byte input data (see the figure below).
4. Fill AES_GHASHRx.GHASH with the state of the GHASH field stored at the end of the message processing.
5. Fill AES_IDATARx.IDATA according to the SMOD configuration used with 'len(*AAD*)64 || len(*C*)64' value as described in the NIST documentation and wait for DATRDY to be set; use interrupt if needed.
6. Read AES_GHASHRx.GHASH to obtain the current value of the hash.

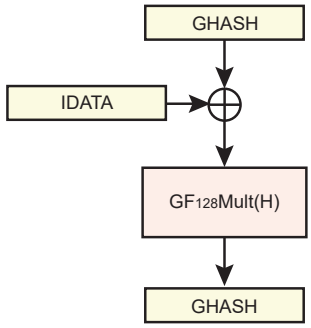
Processing T = GCTRK(J_0, S):

1. Set AES_MR.OPMOD to CTR.
2. Set AES_IVRx.IV with 'J_0' value.
3. Fill AES_IDATARx.IDATA with the GHASH value read at step 6 and wait for DATRDY to be set (use interrupt if needed).
4. Read AES_ODATARx.ODATA to obtain the GCM Tag value.

Note: Step 4 is optional if the GHASH field is to be filled with value '0' (0 length packet for instance).

57.4.4.3.5 Processing a Message with only AAD (GHASHH)

Figure 57-7. Single $GHASH_H$ Block Diagram (AADLEN ≤ 0x10 and CLEN = 0)



It is possible to process a message with only *AAD* setting the CLEN field to '0' in AES_CLENR, this can be used for J0 generation when len(*IV*) ≠ 96 for instance.

Example: Processing J_0 when len(*IV*) ≠ 96

To process $J_0 = GHASH_H(IV \,||\, 0^{s+64} \,||\, [len(IV)]64)$, the sequence is as follows:

1. Set AES_MR.OPMOD to GCM and AES_MR.GTAGEN to '0'.

1. 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.
2. Configure AES_AADLENR.AADLEN with 'len($IV \,||\, 0^{s+64} \,||\, [len(IV)]64$)' in and AES_CLENR.CLEN to '0'. This will allow running a $GHASH_H$ only.
3. Fill AES_IDATARx.IDATA with the message to process ($IV \,||\, 0^{s+64} \,||\, [len(IV)]64$) according to the SMOD configuration used. If Manual Mode or Auto Mode is used, the DATRDY bit indicates when a $GHASH_H$ step is over (use interrupt if needed).
4. Read AES_GHASHRx.GHASH to obtain the J_0 value.
   Note: The GHASH value can be overwritten at any time by writing the value of AES_GHASHRx.GHASH, used to perform a $GHASH_H$ with an initial value for GHASH (write GHASH field between step 3 and step 4 in this case).

57.5.6 AES Interrupt Status Register

Name: AES_ISR
Offset: 0x1C
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								R
Reset								0

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

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

Bit 16 – TAGRDY GCM Tag Ready

Value	Description
0	GCM Tag is not valid.
1	GCM Tag generation is complete (cleared by reading GCM Tag, starting another processing or when writing a new key).

Bits 15:12 – URAT[3:0] Unspecified Register Access (cleared by writing SWRST in AES_CR)
Only the last Unspecified Register Access Type is available through the URAT field.

Value	Name	Description
0	IDR_WR_PROCESSING	Input Data register written during the data processing when SMOD = 2 mode.
1	ODR_RD_PROCESSING	Output Data register read during the data processing.
2	MR_WR_PROCESSING	Mode register written during the data processing.
3	ODR_RD_SUBKGEN	Output Data register read during the sub-keys generation.
4	MR_WR_SUBKGEN	Mode register written during the sub-keys generation.
5	WOR_RD_ACCESS	Write-only register read access.

Bit 8 – URAD Unspecified Register Access Detection Status (cleared by writing SWRST in AES_CR)

Master Read Mode

Slave Read Mode

In slave mode, SPCK is the input clock for the SPI. The max SPCK frequency is given by setup and hold timings SPI_7/SPI_8(or SPI_{10}/SPI_{11}). Since this gives a frequency well above the pad limit, the limit in slave read mode is given by SPCK pad.

Slave Write Mode

58.13.1.6.2 SPI Timings

Timings are given in the following domains:

- 1.8V domain: V_{DDIO} from 1.7V to 1.95V, maximum external capacitor = 20 pF
- 3.3V domain: V_{DDIO} from 2.85V to 3.6V, maximum external capacitor = 40 pF

Table 58-56. SPI Timings

Symbol	Parameter	Conditions	Min	Max	Unit
SPI_0	MISO Setup time before SPCK rises (master)	3.3V domain	12.4	–	ns
		1.8V domain	14.6	–	ns
SPI_1	MISO Hold time after SPCK rises (master)	3.3V domain	0	–	ns
		1.8V domain	0	–	ns
SPI_2	SPCK rising to MOSI Delay (master)	3.3V domain	-3.7	2.2	ns
		1.8V domain	-3.8	2.7	ns
SPI_3	MISO Setup time before SPCK falls (master)	3.3V domain	12.6	–	ns
		1.8V domain	15.13	–	ns
SPI_4	MISO Hold time after SPCK falls (master)	3.3V domain	0	–	ns
		1.8V domain	0	–	ns
SPI_5	SPCK falling to MOSI Delay (master)	3.3V domain	-3.6	2.0	ns
		1.8V domain	-3.3	2.8	ns
SPI_6	SPCK falling to MISO Delay (slave)	3.3V domain	3.0	11.9	ns
		1.8V domain	3.5	13.9	ns
SPI_7	MOSI Setup time before SPCK rises (slave)	3.3V domain	1.2	–	ns
		1.8V domain	1.5	–	ns
SPI_8	MOSI Hold time after SPCK rises (slave)	3.3V domain	0.6	–	ns
		1.8V domain	0.8	–	ns
SPI_9	SPCK rising to MISO Delay (slave)	3.3V domain	3.0	12.0	ns
		1.8V domain	3.4	13.7	ns
SPI_{10}	MOSI Setup time before SPCK falls (slave)	3.3V domain	1.2	–	ns
		1.8V domain	1.5	–	ns
SPI_{11}	MOSI Hold time after SPCK falls (slave)	3.3V domain	0.6	–	ns