Welcome to E-XFL.COM

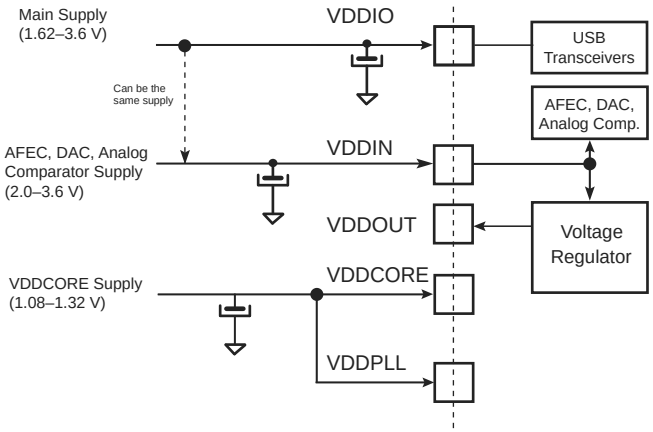
What is "Embedded - Microcontrollers"?

"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®-M4
Core Size	32-Bit Single-Core
Speed	120MHz
Connectivity	CANbus, EBI/EMI, Ethernet, IrDA, SD, SPI, UART/USART, USB
Peripherals	Brown-out Detect/Reset, DMA, POR, PWM, WDT
Number of I/O	117
Program Memory Size	1MB (1M x 8)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	128K x 8
Voltage - Supply (Vcc/Vdd)	1.62V ~ 3.6V
Data Converters	A/D 16x12b; D/A 2x12b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°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/atsam4e16ea-cu

Figure 5-3. Core Externally Supplied



Note: Restrictions:
- For USB, VDDIO needs to be greater than 3.0V
- For AFEC, DAC, and Analog Comparator, VDDIN needs to be greater than 2.4V

5.5 Active Mode

Active mode is the normal running mode with the core clock running from the fast RC oscillator, the main crystal oscillator or the PLLA. The power management controller can be used to adapt the frequency and to disable the peripheral clocks.

5.6 Low-power Modes

The SAM4E has the following low-power modes: Backup mode, Wait mode and Sleep mode.

Note: The Wait For Event instruction (WFE) of the Cortex-M4 core can be used to enter any of the low-power modes, however, this may add complexity in the design of application state machines. This is due to the fact that the WFE instruction goes along with an event flag of the Cortex core (cannot be managed by the software application). The event flag can be set by interrupts, a debug event or an event signal from another processor. Since it is possible for an interrupt to occur just before the execution of WFE, WFE takes into account events that happened in the past. As a result, WFE prevents the device from entering wait mode if an interrupt event has occurred.
Atmel has made provision to avoid using the WFE instruction. The workarounds to ease application design are as follows:
- For backup mode, switch off the voltage regulator and configure the VROFF bit in the Supply Controller Control Register (SUPC_CR).
- For wait mode, configure the WAITMODE bit in the PMC Clock Generator Main Oscillator Register of the Power Management Controller (PMC)
- For sleep mode, use the Wait for Interrupt (WFI) instruction.

Complete information is available in Table 5-1 “Low-power Mode Configuration Summary”.

5.6.1 Backup Mode

The purpose of Backup mode is to achieve the lowest power consumption possible in a system which is performing periodic wake-ups to perform tasks but not requiring fast startup time. Total current consumption is 1 µA typical (VDDIO = 1.8 V at 25°C).

The Supply Controller, zero-power power-on reset, RTT, RTC, backup registers and 32 kHz oscillator (RC or crystal oscillator selected by software in the Supply Controller) are running. The regulator and the core supply are off.

The SAM4E can be woken up from this mode using the pins WKUP0–15, the supply monitor (SM), the RTT or RTC wake-up event.

11.6.6.5 SMLAL and SMLALD

Signed Multiply Accumulate Long, Signed Multiply Accumulate Long (halfwords) and Signed Multiply Accumulate Long Dual.

Syntax

```
op{cond} RdLo, RdHi, Rn, Rm
op{XY}{cond} RdLo, RdHi, Rn, Rm
op{X}{cond} RdLo, RdHi, Rn, Rm
```

where:

op is one of:

MLAL Signed Multiply Accumulate Long.

SMLAL Signed Multiply Accumulate Long (halfwords, X and Y).

X and Y specify which halfword of the source registers *Rn* and *Rm* are used as the first and second multiply operand:

If *X* is B, then the bottom halfword, bits [15:0], of *Rn* is used.
If *X* is T, then the top halfword, bits [31:16], of *Rn* is used.

If *Y* is B, then the bottom halfword, bits [15:0], of *Rm* is used.
If *Y* is T, then the top halfword, bits [31:16], of *Rm* is used.

SMLALD Signed Multiply Accumulate Long Dual.

SMLALDX Signed Multiply Accumulate Long Dual Reversed.

If the *X* is omitted, the multiplications are bottom × bottom and top × top.

If *X* is present, the multiplications are bottom × top and top × bottom.

cond is an optional condition code, see “Conditional Execution” .

RdHi, RdLo are the destination registers.
RdLo is the lower 32 bits and *RdHi* is the upper 32 bits of the 64-bit integer.
For SMLAL, SMLALBB, SMLALBT, SMLALTB, SMLALTT, SMLALD and SMLALDX, they also hold the accumulating value.

Rn, Rm are registers holding the first and second operands.

Operation

The SMLAL instruction:

- Multiplies the two’s complement signed word values from *Rn* and *Rm*.
- Adds the 64-bit value in *RdLo* and *RdHi* to the resulting 64-bit product.
- Writes the 64-bit result of the multiplication and addition in *RdLo* and *RdHi*.

The SMLALBB, SMLALBT, SMLALTB and SMLALTT instructions:

- Multiplies the specified signed halfword, Top or Bottom, values from *Rn* and *Rm*.
- Adds the resulting sign-extended 32-bit product to the 64-bit value in *RdLo* and *RdHi*.
- Writes the 64-bit result of the multiplication and addition in *RdLo* and *RdHi*.

The non-specified halfwords of the source registers are ignored.

The SMLALD and SMLALDX instructions interpret the values from *Rn* and *Rm* as four halfword two’s complement signed 16-bit integers. These instructions:

- If *X* is not present, multiply the top signed halfword value of *Rn* with the top signed halfword of *Rm* and the bottom signed halfword values of *Rn* with the bottom signed halfword of *Rm*.
- Or if *X* is present, multiply the top signed halfword value of *Rn* with the bottom signed halfword of *Rm* and the bottom signed halfword values of *Rn* with the top signed halfword of *Rm*.

11.6.12.2 CPS

Change Processor State.

Syntax

```
CPSeffect iflags
```

where:

effect is one of:

- IE Clears the special purpose register.
- ID Sets the special purpose register.

iflags is a sequence of one or more flags:

- i Set or clear PRIMASK.
- f Set or clear FAULTMASK.

Operation

CPS changes the PRIMASK and FAULTMASK special register values. See “Exception Mask Registers” for more information about these registers.

Restrictions

The restrictions are:

- Use CPS only from privileged software, it has no effect if used in unprivileged software
- CPS cannot be conditional and so must not be used inside an IT block.

Condition Flags

This instruction does not change the condition flags.

Examples

```
CPSID i  ; Disable interrupts and configurable fault handlers (set PRIMASK)
CPSID f  ; Disable interrupts and all fault handlers (set FAULTMASK)
CPSIE i  ; Enable interrupts and configurable fault handlers (clear PRIMASK)
CPSIE f  ; Enable interrupts and fault handlers (clear FAULTMASK)
```

13.5.2 Reset Controller Status Register

Name: RSTC_SR

Address: 0x400E1804

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	SRCMP	NRSTL

15	14	13	12	11	10	9	8
–	–	–	–	–	RSTTYP		

7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	URSTS

- **URSTS: User Reset Status**

A high-to-low transition of the NRST pin sets the URSTS bit. This transition is also detected on the MCK rising edge. If the user reset is disabled (URSTEN = 0 in RSTC_MR) and if the interruption is enabled by the URSTIEN bit in the RSTC_MR, the URSTS bit triggers an interrupt. Reading the RSTC_SR resets the URSTS bit and clears the interrupt.

0: No high-to-low edge on NRST happened since the last read of RSTC_SR.

1: At least one high-to-low transition of NRST has been detected since the last read of RSTC_SR.

- **RSTTYP: Reset Type**

This field reports the cause of the last processor reset. Reading this RSTC_SR does not reset this field.

Value	Name	Description
0	GENERAL_RST	First power-up reset
1	BACKUP_RST	Return from Backup Mode
2	WDT_RST	Watchdog fault occurred
3	SOFT_RST	Processor reset required by the software
4	USER_RST	NRST pin detected low
5	–	Reserved
6	–	Reserved
7	–	Reserved

- **NRSTL: NRST Pin Level**

This bit registers the NRST pin level sampled on each Master Clock (MCK) rising edge.

- **SRCMP: Software Reset Command in Progress**

When set, this bit indicates that a software reset command is in progress and that no further software reset should be performed until the end of the current one. This bit is automatically cleared at the end of the current software reset.

0: No software command is being performed by the Reset Controller. The Reset Controller is ready for a software command.

1: A software reset command is being performed by the Reset Controller. The Reset Controller is busy.

15.6.2 RTC Mode Register

Name: RTC_MR

Address: 0x400E1864

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	TPERIOD		–	THIGH		
23	22	21	20	19	18	17	16
–	OUT1			–	OUT0		
15	14	13	12	11	10	9	8
HIGHPPM	CORRECTION						
7	6	5	4	3	2	1	0
–	–	–	NEGPPM	–	–	PERSIAN	HRMOD

This register can only be written if the WPEN bit is cleared in the System Controller Write Protection Mode Register (SYSC_WPMR).

- **HRMOD: 12-/24-hour Mode**

0: 24-hour mode is selected.

1: 12-hour mode is selected.

- **PERSIAN: PERSIAN Calendar**

0: Gregorian calendar.

1: Persian calendar.

- **NEGPPM: NEGative PPM Correction**

0: Positive correction (the divider will be slightly higher than 32768).

1: Negative correction (the divider will be slightly lower than 32768).

Refer to CORRECTION and HIGHPPM field descriptions.

Note: NEGPPM must be cleared to correct a crystal slower than 32.768 kHz.

- **CORRECTION: Slow Clock Correction**

0: No correction

1–127: The slow clock will be corrected according to the formula given in HIGHPPM description.

- **HIGHPPM: HIGH PPM Correction**

0: Lower range ppm correction with accurate correction.

1: Higher range ppm correction with accurate correction.

If the absolute value of the correction to be applied is lower than 30 ppm, it is recommended to clear HIGHPPM. HIGHPPM set to 1 is recommended for 30 ppm correction and above.

Formula:

If HIGHPPM = 0, then the clock frequency correction range is from 1.5 ppm up to 98 ppm. The RTC accuracy is less than 1 ppm for a range correction from 1.5 ppm up to 30 ppm.

The correction field must be programmed according to the required correction in ppm; the formula is as follows:

Depending on the number of bits to be programmed within the page, the EEFC adapts the write operations required to program the Flash.

When a ‘Write Page’ (WP) command is issued, the EEFC starts the programming sequence and all the bits written at 0 in the latch buffer are cleared in the Flash memory array.

During programming, i.e., until EEFC_FSR.FDRY rises, access to the Flash is not allowed.

Full Page Programming

To program a full page, all the bits of the page must be erased before writing the latch buffer and issuing the WP command. The latch buffer must be written in ascending order, starting from the first address of the page. See Figure 20-8 "Full Page Programming".

Partial Page Programming

To program only part of a page using the WP command, the following constraints must be respected:

- Data to be programmed must be contained in integer multiples of 64-bit address-aligned words.
- 64-bit words can be programmed only if all the corresponding bits in the Flash array are erased (at logical value ‘1’).

See Figure 20-9 "Partial Page Programming".

Programming Bytes

Individual bytes can be programmed using the Partial page programming mode.
In this case, an area of 64 bits must be reserved for each byte.

Refer to Figure 20-10 "Programming Bytes in the Flash".

22.5 Cortex-M Cache Controller (CMCC) User Interface

Table 22-1. Register Mapping

Offset	Register	Name	Access	Reset
0x00	Cache Controller Type Register	CMCC_TYPE	Read-only	–
0x04	Cache Controller Configuration Register	CMCC_CFG	Read/Write	0x00000000
0x08	Cache Controller Control Register	CMCC_CTRL	Write-only	–
0x0C	Cache Controller Status Register	CMCC_SR	Read-only	0x00000001
0x10–0x1C	Reserved	–	–	–
0x20	Cache Controller Maintenance Register 0	CMCC_MAINT0	Write-only	–
0x24	Cache Controller Maintenance Register 1	CMCC_MAINT1	Write-only	–
0x28	Cache Controller Monitor Configuration Register	CMCC_MCFG	Read/Write	0x00000000
0x2C	Cache Controller Monitor Enable Register	CMCC_MEN	Read/Write	0x00000000
0x30	Cache Controller Monitor Control Register	CMCC_MCTRL	Write-only	–
0x34	Cache Controller Monitor Status Register	CMCC_MSR	Read-only	0x00000000
0x38–0xFC	Reserved	–	–	–

25.8.18 DMAC Channel x [x = 0..3] Configuration Register

Name: DMAC_CFGx [x = 0..3]

Address: 0x400C0050 [0], 0x400C0078 [1], 0x400C00A0 [2], 0x400C00C8 [3]

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	FIFOCFG		–	AHB_PROT		

23	22	21	20	19	18	17	16
–	LOCK_IF_L	LOCK_B	LOCK_IF	–	–	–	SOD

15	14	13	12	11	10	9	8
–	–	DST_H2SEL	–	–	–	SRC_H2SEL	–

7	6	5	4	3	2	1	0
DST_PER				SRC_PER			

This register can only be written if the WPEN bit is cleared in “DMAC Write Protection Mode Register” on page 506

- **SRC_PER: Source with Peripheral identifier**

Channel x Source Request is associated with peripheral identifier coded SRC_PER handshaking interface.

- **DST_PER: Destination with Peripheral identifier**

Channel x Destination Request is associated with peripheral identifier coded DST_PER handshaking interface.

- **SRC_H2SEL: Software or Hardware Selection for the Source**

0 (SW): Software handshaking interface is used to trigger a transfer request.

1 (HW): Hardware handshaking interface is used to trigger a transfer request.

- **DST_H2SEL: Software or Hardware Selection for the Destination**

0 (SW): Software handshaking interface is used to trigger a transfer request.

1 (HW): Hardware handshaking interface is used to trigger a transfer request.

- **SOD: Stop On Done**

0 (DISABLE): STOP ON DONE disabled, the descriptor fetch operation ignores the DMAC_CTRLAx.DONE bit.

1 (ENABLE): STOP ON DONE activated, the DMAC module is automatically disabled if DMAC_CTRLAx.DONE bit is set.

- **LOCK_IF: Interface Lock**

0 (DISABLE): Interface Lock capability is disabled

1 (ENABLE): Interface Lock capability is enabled

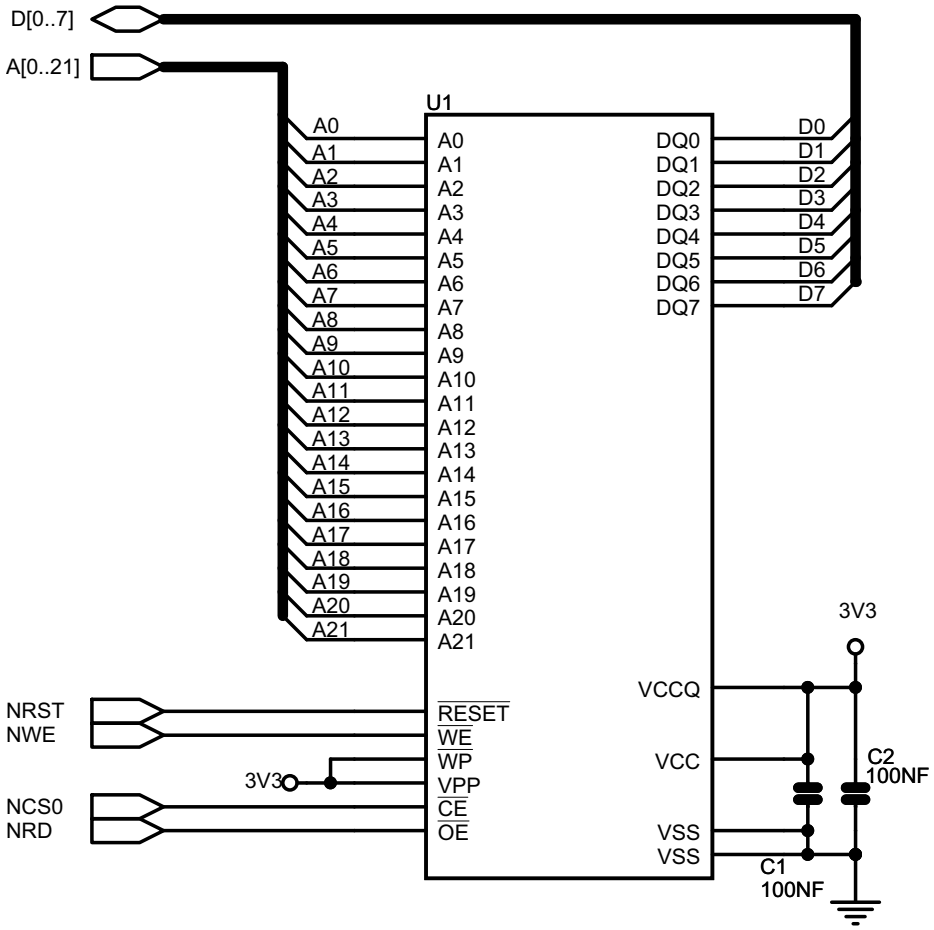
- **LOCK_B: Bus Lock**

0 (DISABLE): AHB Bus Locking capability is disabled.

1(ENABLE): AHB Bus Locking capability is enabled.

27.8.1.2 NOR Flash

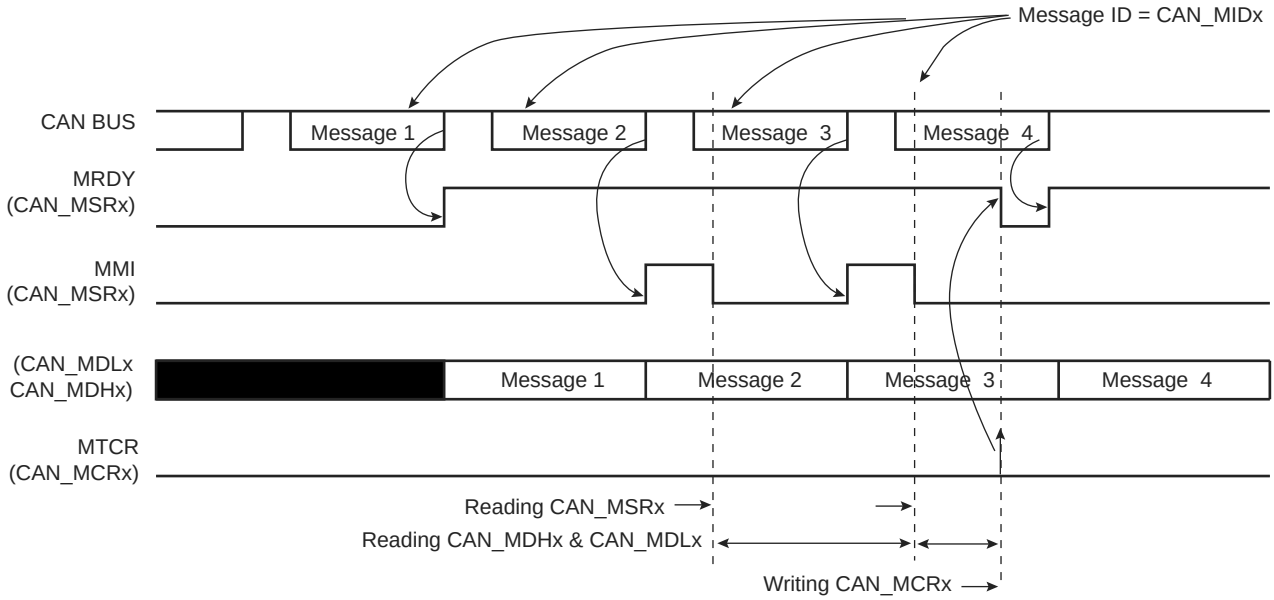
Hardware Configuration



Software Configuration

Configure the SMC CS0 Setup, Pulse, Cycle and Mode depending on Flash timings and system bus frequency.

Figure 31-12. Receive with Overwrite Mailbox



Chaining Mailboxes

Several mailboxes may be used to receive a buffer split into several messages with the same ID. In this case, the mailbox with the lowest number is serviced first. In the receive and receive with overwrite modes, the field PRIOR in the CAN_MMRx has no effect. If Mailbox 0 and Mailbox 5 accept messages with the same ID, the first message is received by Mailbox 0 and the second message is received by Mailbox 5. Mailbox 0 must be configured in Receive Mode (i.e., the first message received is considered) and Mailbox 5 must be configured in Receive with Overwrite Mode. Mailbox 0 cannot be configured in Receive with Overwrite Mode; otherwise, all messages are accepted by this mailbox and Mailbox 5 is never serviced.

If several mailboxes are chained to receive a buffer split into several messages, all mailboxes except the last one (with the highest number) must be configured in Receive Mode. The first message received is handled by the first mailbox, the second one is refused by the first mailbox and accepted by the second mailbox, the last message is accepted by the last mailbox and refused by previous ones (see Figure 31-13).

- **DRPT: Disable Repeat**

0: When a transmit mailbox loses the bus arbitration, the transfer request remains pending.

1: When a transmit mailbox loses the bus arbitration, the transfer request is automatically aborted. It automatically raises the MABT and MRDT flags in the corresponding CAN_MSRx.

31.9.20 CAN Message Data High Register

Name: CAN_MDHx [x=0..7]

Address: 0x40010218 (0)[0], 0x40010238 (0)[1], 0x40010258 (0)[2], 0x40010278 (0)[3], 0x40010298 (0)[4], 0x400102B8 (0)[5], 0x400102D8 (0)[6], 0x400102F8 (0)[7], 0x40014218 (1)[0], 0x40014238 (1)[1], 0x40014258 (1)[2], 0x40014278 (1)[3], 0x40014298 (1)[4], 0x400142B8 (1)[5], 0x400142D8 (1)[6], 0x400142F8 (1)[7]

Access: Read/Write

31	30	29	28	27	26	25	24
MDH							
23	22	21	20	19	18	17	16
MDH							
15	14	13	12	11	10	9	8
MDH							
7	6	5	4	3	2	1	0
MDH							

- **MDH: Message Data High Value**

When MRDY bit is set in the CAN_MSRx, the upper 32 bits of a received message are read or written by the software application. Otherwise, the MDH value is locked by the CAN controller to send/receive a new message.

In Receive with overwrite, the CAN controller may modify MDH value while the software application reads MDH and MDL registers. To check that MDH and MDL do not belong to different messages, the application has to check the MMI bit in the CAN_MSRx. In this mode, the software application must re-read CAN_MDH and CAN_MDL, while the MMI bit in the CAN_MSRx is set.

Bytes are received/sent on the bus in the following order:

1. CAN_MDL[7:0]
2. CAN_MDL[15:8]
3. CAN_MDL[23:16]
4. CAN_MDL[31:24]
5. CAN_MDH[7:0]
6. CAN_MDH[15:8]
7. CAN_MDH[23:16]
8. CAN_MDH[31:24]

38.7.1 TC Channel Control Register

Name: TC_CCRx [x=0..2]

Address: 0x40090000 (0)[0], 0x40090040 (0)[1], 0x40090080 (0)[2], 0x40094000 (1)[0], 0x40094040 (1)[1], 0x40094080 (1)[2], 0x40098000 (2)[0], 0x40098040 (2)[1], 0x40098080 (2)[2]

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	–	–	SWTRG	CLKDIS	CLKEN

- **CLKEN: Counter Clock Enable Command**

0: No effect.

1: Enables the clock if CLKDIS is not 1.

- **CLKDIS: Counter Clock Disable Command**

0: No effect.

1: Disables the clock.

- **SWTRG: Software Trigger Command**

0: No effect.

1: A software trigger is performed: the counter is reset and the clock is started.

38.7.12 TC Interrupt Disable Register

Name: TC_IDRx [x=0..2]

Address: 0x40090028 (0)[0], 0x40090068 (0)[1], 0x400900A8 (0)[2], 0x40094028 (1)[0], 0x40094068 (1)[1], 0x400940A8 (1)[2], 0x40098028 (2)[0], 0x40098068 (2)[1], 0x400980A8 (2)[2]

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	RXBUFF	ENDRX
7	6	5	4	3	2	1	0
ETRGS	LDRBS	LDRAS	CPCS	CPBS	CPAS	LOVRS	COVFS

- **COVFS: Counter Overflow**

0: No effect.

1: Disables the Counter Overflow Interrupt.

- **LOVRS: Load Overrun**

0: No effect.

1: Disables the Load Overrun Interrupt (if TC_CMRx.WAVE = 0).

- **CPAS: RA Compare**

0: No effect.

1: Disables the RA Compare Interrupt (if TC_CMRx.WAVE = 1).

- **CPBS: RB Compare**

0: No effect.

1: Disables the RB Compare Interrupt (if TC_CMRx.WAVE = 1).

- **CPCS: RC Compare**

0: No effect.

1: Disables the RC Compare Interrupt.

- **LDRAS: RA Loading**

0: No effect.

1: Disables the RA Load Interrupt (if TC_CMRx.WAVE = 0).

- **LDRBS: RB Loading**

0: No effect.

1: Disables the RB Load Interrupt (if TC_CMRx.WAVE = 0).

Figure 39-22. Synchronized Update of Update Period Value of Synchronous Channels

39.6.5.5 Changing the Comparison Value and the Comparison Configuration

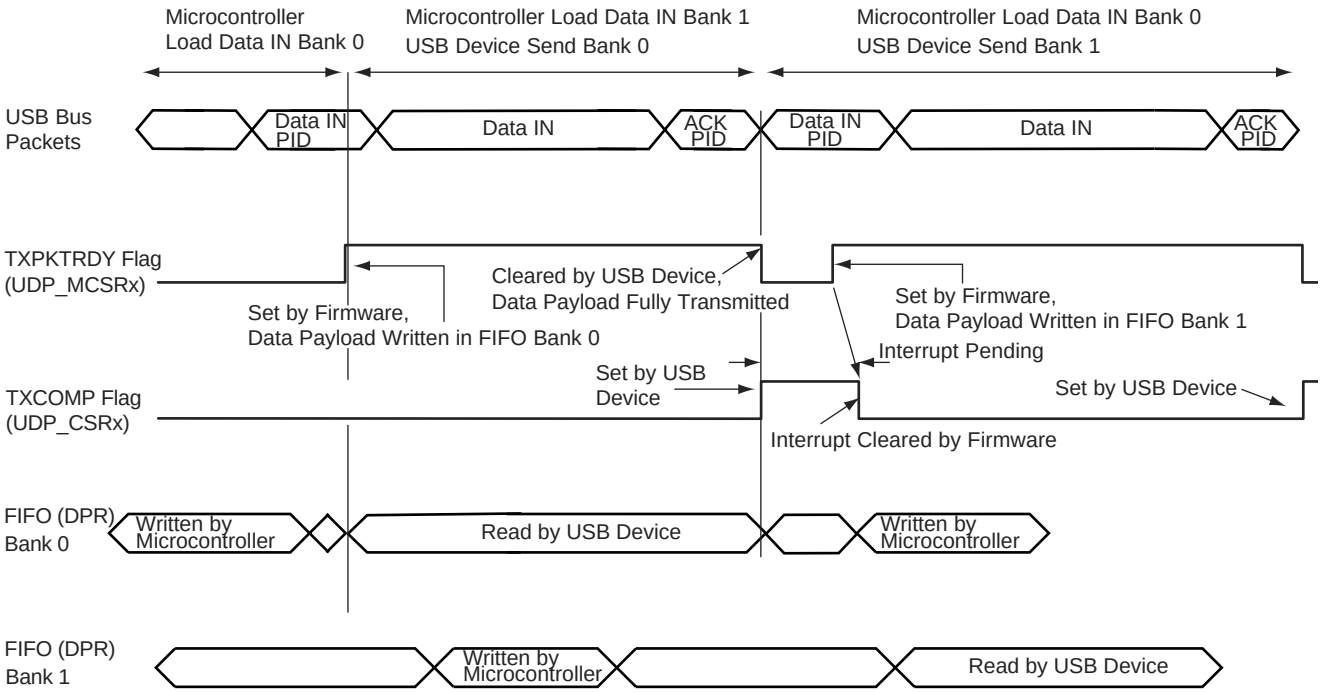
It is possible to change the comparison values and the comparison configurations while the channel 0 is enabled (see Section 39.6.3 “PWM Comparison Units”).

To prevent unexpected comparison match, the user must use the PWM Comparison x Value Update Register (PWM_CMPVUPDx) and the PWM Comparison x Mode Update Register (PWM_CMPMUPDx) to change, respectively, the comparison values and the comparison configurations while the channel 0 is still enabled. These registers hold the new values until the end of the comparison update period (when CUPRCNT is equal to CUPR in PWM Comparison x Mode Register (PWM_CMPMx) and the end of the current PWM period, then update the values for the next period.

CAUTION: The write of the register PWM_CMPVUPDx must be followed by a write of the register PWM_CMPMUPDx.

Note: If the update registers PWM_CMPVUPDx and PWM_CMPMUPDx are written several times between two updates, only the last written value are taken into account.

Figure 41-8. Data IN Transfer for Ping-pong Endpoint



Warning: There is software critical path due to the fact that once the second bank is filled, the driver has to wait for TX_COMP to set TX_PKTRDY. If the delay between receiving TX_COMP is set and TX_PKTRDY is set too long, some Data IN packets may be NACKed, reducing the bandwidth.

Warning: TX_COMP must be cleared after TX_PKTRDY has been set.

41.6.2.3 Data OUT Transaction

Data OUT transactions are used in control, isochronous, bulk and interrupt transfers and conduct the transfer of data from the host to the device. Data OUT transactions in isochronous transfers must be done using endpoints with ping-pong attributes.

Data OUT Transaction Without Ping-pong Attributes

To perform a Data OUT transaction, using a non ping-pong endpoint:

1. The host generates a Data OUT packet.
2. This packet is received by the USB device endpoint. While the FIFO associated to this endpoint is being used by the microcontroller, a NAK PID is returned to the host. Once the FIFO is available, data are written to the FIFO by the USB device and an ACK is automatically carried out to the host.
3. The microcontroller is notified that the USB device has received a data payload polling RX_DATA_BK0 in the endpoint's UDP_CSRx. An interrupt is pending for this endpoint while RX_DATA_BK0 is set.
4. The number of bytes available in the FIFO is made available by reading RXBYTECNT in the endpoint's UDP_CSRx.
5. The microcontroller carries out data received from the endpoint's memory to its memory. Data received is available by reading the endpoint's UDP_FDRx.
6. The microcontroller notifies the USB device that it has finished the transfer by clearing RX_DATA_BK0 in the endpoint's UDP_CSRx.
7. A new Data OUT packet can be accepted by the USB device.

For power-saving options, see Section 44.6.6 “DACC Timings”.

44.5.2 Interrupt Sources

The DACC interrupt line is connected to one of the internal sources of the interrupt controller. Using the DACC interrupt requires the interrupt controller to be programmed first.

Table 44-2. Peripheral IDs

Instance	ID
DACC	32

44.5.3 Conversion Performances

For performance and electrical characteristics of the DACC, see the product DC Characteristics section of the datasheet.

44.7.11 DACC Analog Current Register

Name: DACC_ACR

Address: 0x400B8094

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	IBCTLDACCORE	
7	6	5	4	3	2	1	0
–	–	–	–	IBCTLCH1		IBCTLCH0	

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

- **IBCTLCHx: Analog Output Current Control**

Used to modify the slew rate of the analog output (See the product Electrical Characteristics section for further details.)

- **IBCTLDACCORE: Bias Current Control for DAC Core**

Used to modify performance versus power consumption (See the product Electrical Characteristics section for further details.)

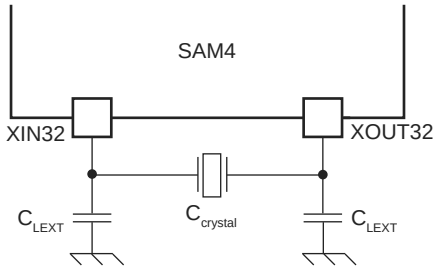
46.4.3 32.768 kHz Crystal Oscillator Characteristics

Table 46-18. 32.768 kHz Crystal Oscillator Characteristics

Symbol	Parameter	Conditions		Min	Typ	Max	Unit
f_{OSC}	Operating Frequency	Normal mode with crystal		—	—	32.768	kHz
$V_{rip(VDDIO)}$	Supply Ripple Voltage (on VDDIO)	RMS value, 10 kHz to 10 MHz		—	—	30	mV
—	Duty Cycle	—		40	50	60	%
t_{START}	Startup Time	R_S < 50 kΩ (1)	$C_{crystal}$ = 12.5 pF	—	—	900	ms
			$C_{crystal}$ = 6 pF	—	—	300	
		R_S < 100 kΩ (1)	$C_{crystal}$ = 12.5 pF	—	—	1200	
			$C_{crystal}$ = 6 pF	—	—	500	
I_{DDON}	Current Consumption	R_S < 50 kΩ (1)	$C_{crystal}$ = 12.5 pF	—	550	1150	nA
			$C_{crystal}$ = 6 pF	—	380	980	
		R_S < 100 kΩ (1)	$C_{crystal}$ = 12.5 pF	—	820	1600	
			$C_{crystal}$ = 6 pF	—	530	1350	
P_{ON}	Drive Level	—		—	—	0.1	µW
R_f	Internal Resistor	Between XIN32 and XOUT32		—	10	—	MΩ
$C_{crystal}$	Allowed Crystal Capacitance Load	From crystal specification		6	—	12.5	pF
C_{para}	Internal Parasitic Capacitance	—		0.6	0.7	0.8	pF

Note: 1. R_S is the series resistor.

Figure 46-11. 32.768 kHz Crystal Oscillator Schematics



$C_{LEXT} = 2 \times (C_{crystal} - C_{para} - C_{PCB})$

where:

C_{PCB} is the capacitance of the printed circuit board (PCB) track layout from the crystal to the SAM4 pin.

46.4.4 32.768 kHz Crystal Characteristics

Table 46-19. Crystal Characteristics

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
ESR	Equivalent Series Resistor (R_S)	Crystal @ 32.768 kHz	—	50	100	kΩ
C_m	Motional Capacitance	Crystal @ 32.768 kHz	0.6	—	3	fF
C_{SHUNT}	Shunt Capacitance	Crystal @ 32.768 kHz	0.6	—	2	pF

Table 51-4. SAM4E Datasheet Rev. 11157E Revision History (Continued)

Doc. Date	Changes
13-Feb-15	Section 46. “SAM4E Electrical Characteristics” (cont’d) Section 46.11.8.1 “Timing Conditions”: at end of section, deleted sentence “These values may be product dependent and should be confirmed by the specification” Section 46.11.9 “Embedded Flash Characteristics”: - inserted paragraph explaining that flash contents should be erased prior to programming an application - in second paragraph, corrected “field FWS of the MC_FMR” to “field FWS of the EEFC_FMR” - replaced four “Embedded Flash Wait State” tables with single Table 46-67 “Embedded Flash Wait State at 105°C”
	Section 49. “Ordering Information” Table 49-1 “Ordering Codes for SAM4E Devices”: removed “Package Type” column (package type information is provided on the Atmel website)