Microchip Technology - ATSAM4E16CB-AN Datasheet

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, Ethernet, IrDA, MMC/SD, SPI, UART/USART, USB
Peripherals	Brown-out Detect/Reset, DMA, POR, PWM, WDT
Number of I/O	79
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 ~ 105°C (TA)
Mounting Type	Surface Mount
Package / Case	100-LQFP
Supplier Device Package	100-LQFP (14x14)
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/atsam4e16cb-an

5. Power Considerations

5.1 Power Supplies

The SAM4E has several types of power supply pins:

- VDDCORE pins: power the core, the first flash rail, the embedded memories and the peripherals.
  Voltage ranges from 1.08V to 1.32V.
- VDDIO pins: power the peripheral I/O lines (Input/Output Buffers), the second flash rail, the backup part, the USB transceiver, 32 kHz crystal oscillator and oscillator pads.
  Voltage ranges from 1.62V to 3.6V.
- VDDIN pins: voltage regulator input, DAC and Analog Comparator power supply.
  Voltage ranges from 1.62V to 3.6V.
- VDDPLL pin: powers the PLL, the Fast RC and the 3 to 20 MHz oscillator.
  Voltage ranges from 1.08V to 1.32V.

5.2 Power-up Considerations

5.2.1 VDDIO Versus VDDCORE

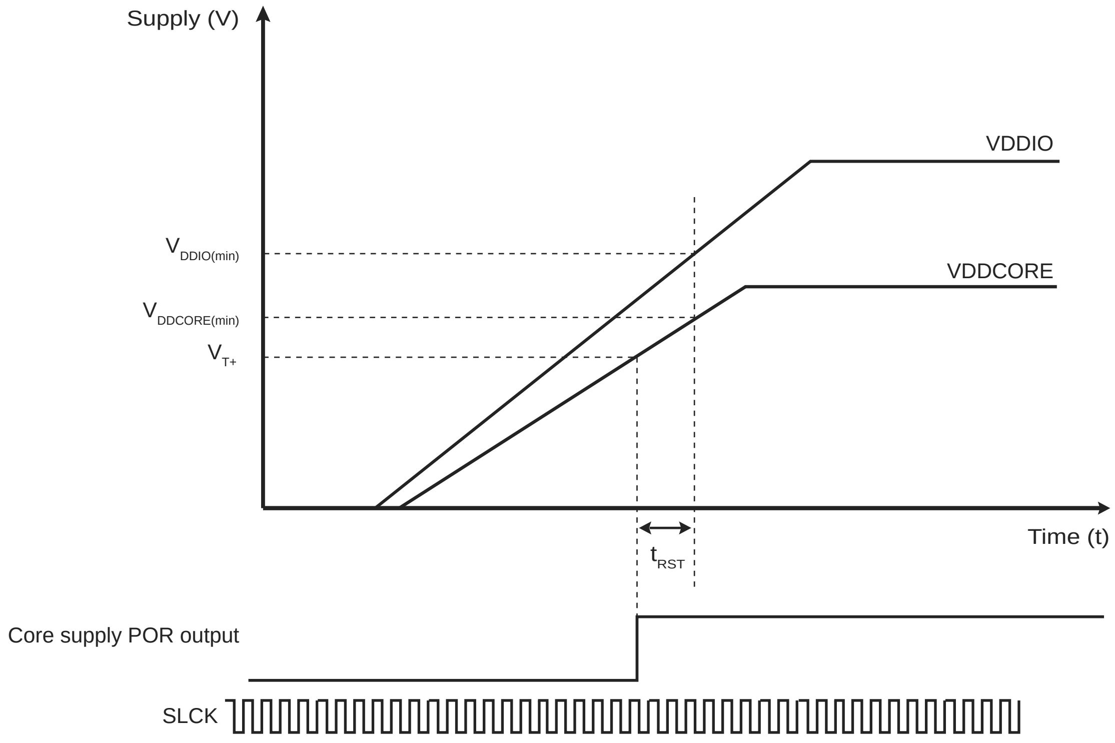
V_{DDIO} must always be higher than or equal to V_{DDCORE}.

V_{DDIO} must reach its minimum operating voltage (1.62 V) before V_{DDCORE} has reached $V_{DDCORE(min)}$. The minimum slope for V_{DDCORE} is defined by $(V_{DDCORE(min)} - V_{T+}) / t_{RST}$.

If V_{DDCORE} rises at the same time as V_{DDIO}, the V_{DDIO} rising slope must be higher than or equal to 8.8 V/ms.

If VDDCORE is powered by the internal regulator, all power-up considerations are met

Figure 5-1. VDDCORE and VDDIO Constraints at Startup



3. If the returned status bit from step 2 indicates that the Store-Exclusive instruction succeeded then the software has claimed the semaphore. However, if the Store-Exclusive instruction failed, another process might have claimed the semaphore after the software performed the first step.

The Cortex-M4 includes an exclusive access monitor, that tags the fact that the processor has executed a Load-Exclusive instruction. If the processor is part of a multiprocessor system, the system also globally tags the memory locations addressed by exclusive accesses by each processor.

The processor removes its exclusive access tag if:

- It executes a CLREX instruction
- It executes a Store-Exclusive instruction, regardless of whether the write succeeds.
- An exception occurs. This means that the processor can resolve semaphore conflicts between different threads.

In a multiprocessor implementation:

- Executing a CLREX instruction removes only the local exclusive access tag for the processor
- Executing a Store-Exclusive instruction, or an exception, removes the local exclusive access tags, and all global exclusive access tags for the processor.

For more information about the synchronization primitive instructions, see “LDREX and STREX” and “CLREX” .

11.4.2.8 Programming Hints for the Synchronization Primitives

ISO/IEC C cannot directly generate the exclusive access instructions. CMSIS provides intrinsic functions for generation of these instructions:

Table 11-8. CMSIS Functions for Exclusive Access Instructions

Instruction	CMSIS Function
LDREX	uint32_t __LDREXW (uint32_t *addr)
LDREXH	uint16_t __LDREXH (uint16_t *addr)
LDREXB	uint8_t __LDREXB (uint8_t *addr)
STREX	uint32_t __STREXW (uint32_t value, uint32_t *addr)
STREXH	uint32_t __STREXH (uint16_t value, uint16_t *addr)
STREXB	uint32_t __STREXB (uint8_t value, uint8_t *addr)
CLREX	void __CLREX (void)

The actual exclusive access instruction generated depends on the data type of the pointer passed to the intrinsic function. For example, the following C code generates the required LDREXB operation:

```
    __ldrex((volatile char *) 0xFF);
```

11.4.3 Exception Model

This section describes the exception model.

11.4.3.1 Exception States

Each exception is in one of the following states:

Inactive

The exception is not active and not pending.

Pending

The exception is waiting to be serviced by the processor.

11.6.5.24 USUB16 and USUB8

Unsigned Subtract 16 and Unsigned Subtract 8

Syntax

```
    op{cond}{Rd,} Rn, Rm
```

where

op	is any of:
	USUB16 Unsigned Subtract 16.
	USUB8 Unsigned Subtract 8.
cond	is an optional condition code, see “Conditional Execution” .
Rd	is the destination register.
Rn	is the first operand register.
Rm	is the second operand register.

Operation

Use these instructions to subtract 16-bit and 8-bit data before writing the result to the destination register:

The USUB16 instruction:

1. Subtracts each halfword from the second operand register from the corresponding halfword of the first operand register.
2. Writes the unsigned result in the corresponding halfwords of the destination register.

The USUB8 instruction:

1. Subtracts each byte of the second operand register from the corresponding byte of the first operand register.
2. Writes the unsigned byte result in the corresponding byte of the destination register.

Restrictions

Do not use SP and do not use PC.

Condition Flags

These instructions do not change the flags.

Examples

```
USUB16 R1, R0  ; Subtracts halfwords in R0 from corresponding halfword of R1
               ; and writes to corresponding halfword in R1USUB8  R4, R0, R5
               ; Subtracts bytes of R5 from corresponding byte in R0 and
               ; writes to the corresponding byte in R4.
```

11.6.6.7 SMMLA and SMMLS

Signed Most Significant Word Multiply Accumulate and Signed Most Significant Word Multiply Subtract

Syntax

```
        op{R}{cond} Rd, Rn, Rm, Ra
```

where:

op	is one of:
	SMMLA Signed Most Significant Word Multiply Accumulate.
	SMMLS Signed Most Significant Word Multiply Subtract.
	If the *X* is omitted, the multiplications are bottom × bottom and top × top.
R	is a rounding error flag. If *R* is specified, the result is rounded instead of being truncated. In this case the constant 0x80000000 is added to the product before the high word is extracted.
cond	is an optional condition code, see “Conditional Execution” .
Rd	is the destination register.
Rn, Rm	are registers holding the first and second multiply operands.
Ra	is the register holding the accumulate value.

Operation

The SMMLA instruction interprets the values from *Rn* and *Rm* as signed 32-bit words.

The SMMLA instruction:

- Multiplies the values in *Rn* and *Rm*.
- Optionally rounds the result by adding 0x80000000.
- Extracts the most significant 32 bits of the result.
- Adds the value of *Ra* to the signed extracted value.
- Writes the result of the addition in *Rd*.

The SMMLS instruction interprets the values from *Rn* and *Rm* as signed 32-bit words.

The SMMLS instruction:

- Multiplies the values in *Rn* and *Rm*.
- Optionally rounds the result by adding 0x80000000.
- Extracts the most significant 32 bits of the result.
- Subtracts the extracted value of the result from the value in *Ra*.
- Writes the result of the subtraction in *Rd*.

Restrictions

In these instructions:

- Do not use SP and do not use PC.

Condition Flags

These instructions do not affect the condition code flags.

11.6.7.2 SSAT16 and USAT16

Signed Saturate and Unsigned Saturate to any bit position for two halfwords.

Syntax

```
op{cond} Rd, #n, Rm
```

where:

op	is one of:
	SSAT16 Saturates a signed halfword value to a signed range.
	USAT16 Saturates a signed halfword value to an unsigned range.
cond	is an optional condition code, see "Conditional Execution" .
Rd	is the destination register.
n	specifies the bit position to saturate to:
n ranges from 1 to 16 for SSAT	n ranges from 0 to 15 for USAT.
Rm	is the register containing the value to saturate.

Operation

The SSAT16 instruction:

Saturates two signed 16-bit halfword values of the register with the value to saturate from selected by the bit position in *n*.

Writes the results as two signed 16-bit halfwords to the destination register.

The USAT16 instruction:

Saturates two unsigned 16-bit halfword values of the register with the value to saturate from selected by the bit position in *n*.

Writes the results as two unsigned halfwords in the destination register.

Restrictions

Do not use SP and do not use PC.

Condition Flags

These instructions do not affect the condition code flags.

If saturation occurs, these instructions set the Q flag to 1.

Examples

```
SSAT16    R7, #9, R2   ; Saturates the top and bottom highwords of R2
                       ; as 9-bit values, writes to corresponding halfword
                       ; of R7
USAT16NE  R0, #13, R5  ; Conditionally saturates the top and bottom
                       ; halfwords of R5 as 13-bit values, writes to
                       ; corresponding halfword of R0.
```

11.6.7.3 QADD and QSUB

Saturating Add and Saturating Subtract, signed.

Syntax

```
op{cond} {Rd}, Rn, Rm
op{cond} {Rd}, Rn, Rm
```

where:

op	is one of:
	QADD Saturating 32-bit add.
	QADD8 Saturating four 8-bit integer additions.
	QADD16 Saturating two 16-bit integer additions.
	QSUB Saturating 32-bit subtraction.
	QSUB8 Saturating four 8-bit integer subtraction.
	QSUB16 Saturating two 16-bit integer subtraction.
cond	is an optional condition code, see “Conditional Execution” .
Rd	is the destination register.
Rn, Rm	are registers holding the first and second operands.

Operation

These instructions add or subtract two, four or eight values from the first and second operands and then writes a signed saturated value in the destination register.

The QADD and QSUB instructions apply the specified add or subtract, and then saturate the result to the signed range $-2^{n-1} \leq x \leq 2^{n-1}-1$, where x is given by the number of bits applied in the instruction, 32, 16 or 8.

If the returned result is different from the value to be saturated, it is called *saturation*. If saturation occurs, the QADD and QSUB instructions set the Q flag to 1 in the APSR. Otherwise, it leaves the Q flag unchanged. The 8-bit and 16-bit QADD and QSUB instructions always leave the Q flag unchanged.

To clear the Q flag to 0, the MSR instruction must be used; see “MSR” .

To read the state of the Q flag, the MRS instruction must be used; see “MRS” .

Restrictions

Do not use SP and do not use PC.

Condition Flags

These instructions do not affect the condition code flags.

If saturation occurs, these instructions set the Q flag to 1.

Examples

```
QADD16   R7, R4, R2  ; Adds halfwords of R4 with corresponding halfword of
                     ; R2, saturates to 16 bits and writes to
                     ; corresponding halfword of R7
QADD8    R3, R1, R6  ; Adds bytes of R1 to the corresponding bytes of R6,
                     ; saturates to 8 bits and writes to corresponding
                     ; byte of R3
QSUB16   R4, R2, R3  ; Subtracts halfwords of R3 from corresponding
                     ; halfword of R2, saturates to 16 bits, writes to
                     ; corresponding halfword of R4
QSUB8    R4, R2, R5  ; Subtracts bytes of R5 from the corresponding byte
                     ; in R2, saturates to 8 bits, writes to corresponding
                     ; byte of R4.
```

- **PENDSVCLR: PendSV Clear-pending**

Write:

0: No effect.

1: Removes the pending state from the PendSV exception.

- **PENDSTSET: SysTick Exception Set-pending**

Write:

0: No effect.

1: Changes SysTick exception state to pending.

Read:

0: SysTick exception is not pending.

1: SysTick exception is pending.

- **PENDSTCLR: SysTick Exception Clear-pending**

Write:

0: No effect.

1: Removes the pending state from the SysTick exception.

This bit is Write-only. On a register read, its value is Unknown.

- **ISRPENDING: Interrupt Pending Flag (Excluding NMI and Faults)**

0: Interrupt not pending.

1: Interrupt pending.

- **VECTPENDING: Exception Number of the Highest Priority Pending Enabled Exception**

0: No pending exceptions.

Nonzero: The exception number of the highest priority pending enabled exception.

The value indicated by this field includes the effect of the BASEPRI and FAULTMASK registers, but not any effect of the PRIMASK register.

- **RETTOBASE: Preempted Active Exceptions Present or Not**

0: There are preempted active exceptions to execute.

1: There are no active exceptions, or the currently-executing exception is the only active exception.

- **VECTACTIVE: Active Exception Number Contained**

0: Thread mode.

Nonzero: The exception number of the currently active exception. The value is the same as IPSR bits [8:0]. See “Interrupt Program Status Register” .

Subtract 16 from this value to obtain the IRQ number required to index into the Interrupt Clear-Enable, Set-Enable, Clear-Pending, Set-Pending, or Priority Registers, see “Interrupt Program Status Register” .

Note: When the user writes to the SCB_ICSR, the effect is unpredictable if:
- Writing a 1 to the PENDSVSET bit and writing a 1 to the PENDSVCLR bit
- Writing a 1 to the PENDSTSET bit and writing a 1 to the PENDSTCLR bit.

11.9.1.17 Bus Fault Address Register

Name: SCB_BFAR

Access: Read/Write

31	30	29	28	27	26	25	24
ADDRESS							

23	22	21	20	19	18	17	16
ADDRESS							

15	14	13	12	11	10	9	8
ADDRESS							

7	6	5	4	3	2	1	0
ADDRESS							

The SCB_BFAR contains the address of the location that generated a bus fault.

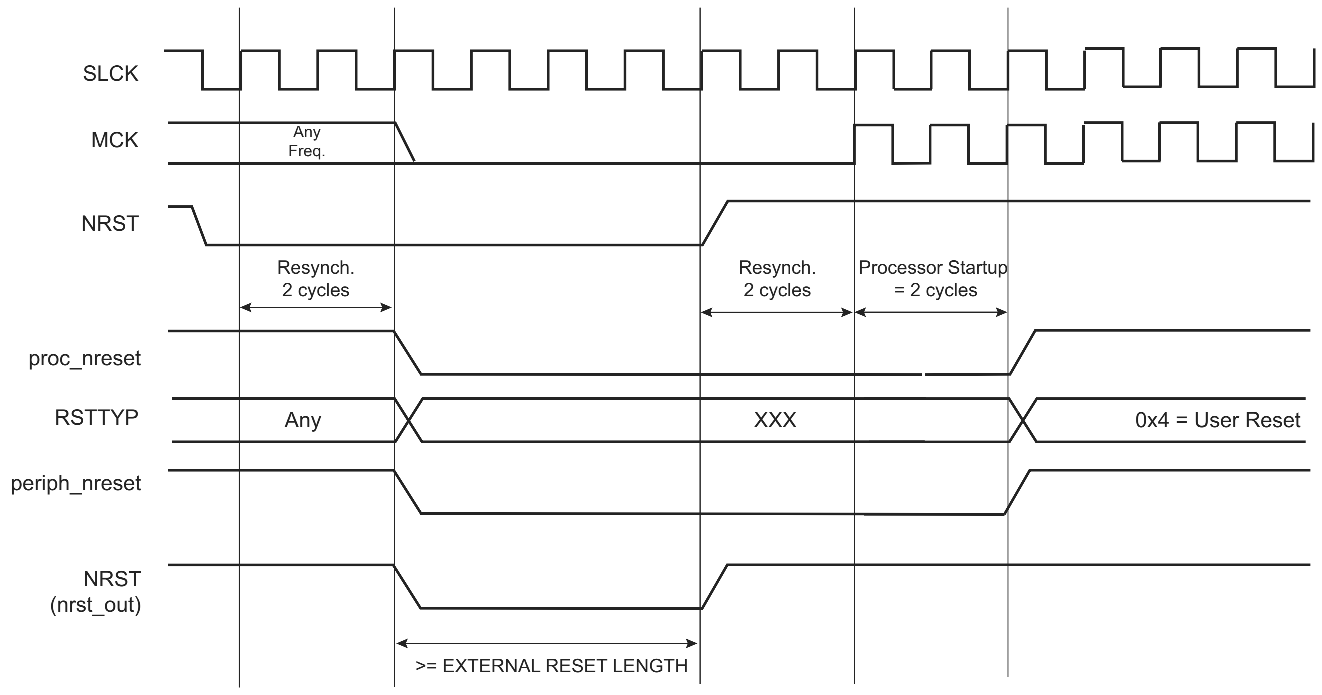
- **ADDRESS: Bus Fault Generation Location Address**

When the BFARVALID bit of the BFSR subregister is set to 1, this field holds the address of the location that generated the bus fault.

Notes: 1. When an unaligned access faults, the address in the SCB_BFAR is the one requested by the instruction, even if it is not the address of the fault.

2. Flags in the BFSR indicate the cause of the fault, and whether the value in the SCB_BFAR is valid. See “BFSR: Bus Fault Status Subregister” .

Figure 13-6. User Reset State



13.4.4 Reset State Priorities

The reset state manager manages the priorities among the different reset sources. The resets are listed in order of priority as follows:

1. General reset
2. Backup reset
3. Watchdog reset
4. Software reset
5. User reset

Particular cases are listed below:

- When in user reset:
  - A watchdog event is impossible because the Watchdog Timer is being reset by the proc_nreset signal.
  - A software reset is impossible, since the processor reset is being activated.
- When in software reset:
  - A watchdog event has priority over the current state.
  - The NRST has no effect.
- When in watchdog reset:
  - The processor reset is active and so a software reset cannot be programmed.
  - A user reset cannot be entered.

24.11 Write Protect Registers

To prevent any single software error that may corrupt the Bus Matrix behavior, the entire Bus Matrix address space can be write-protected by setting the WPEN bit in the Bus Matrix Write Protect Mode Register (MATRIX_WPMR).

If WPEN is at one and a write access in the Bus Matrix address space is detected, then the WPVS flag in the Bus Matrix Write Protect Status Register (MATRIX_WPSR) is set and the field WPVSRC indicates in which register the write access has been attempted.

The WPVS flag is reset by writing the Bus Matrix Write Protect Mode Register (MATRIX_WPMR) with the appropriate access key WPKEY.

The protected registers are:

"Bus Matrix Master Configuration Registers"

"Bus Matrix Slave Configuration Registers"

"Bus Matrix Priority Registers A For Slaves"

"Bus Matrix Master Remap Control Register"

"Write Protect Mode Register"

25.3 DMA Controller Peripheral Connections

The DMA Controller handles the transfer between peripherals and memory and receives triggers from the peripherals listed in the following table.

Table 25-1. DMA Channel Definition

Instance Name	Transmit/Receive	DMA Channel Number
HSMCI	Transmit/Receive	0
SPI	Transmit	1
SPI	Receive	2
USART0	Transmit	3
USART0	Receive	4
USART1	Transmit	5
USART1	Receive	6
AES	Transmit	11
AES	Receive	12
PWM	Transmit	13

33. Parallel Input/Output Controller (PIO)

33.1 Description

The Parallel Input/Output Controller (PIO) manages up to 32 fully programmable input/output lines. Each I/O line may be dedicated as a general-purpose I/O or be assigned to a function of an embedded peripheral. This ensures effective optimization of the pins of the product.

Each I/O line is associated with a bit number in all of the 32-bit registers of the 32-bit wide user interface.

Each I/O line of the PIO Controller features:

- An input change interrupt enabling level change detection on any I/O line.
- Additional Interrupt modes enabling rising edge, falling edge, low-level or high-level detection on any I/O line.
- A glitch filter providing rejection of glitches lower than one-half of peripheral clock cycle.
- A debouncing filter providing rejection of unwanted pulses from key or push button operations.
- Multi-drive capability similar to an open drain I/O line.
- Control of the pull-up and pull-down of the I/O line.
- Input visibility and output control.

The PIO Controller also features a synchronous output providing up to 32 bits of data output in a single write operation.

An 8-bit parallel capture mode is also available which can be used to interface a CMOS digital image sensor, an ADC, a DSP synchronous port in synchronous mode, etc.

Figure 38-6. Capture Mode

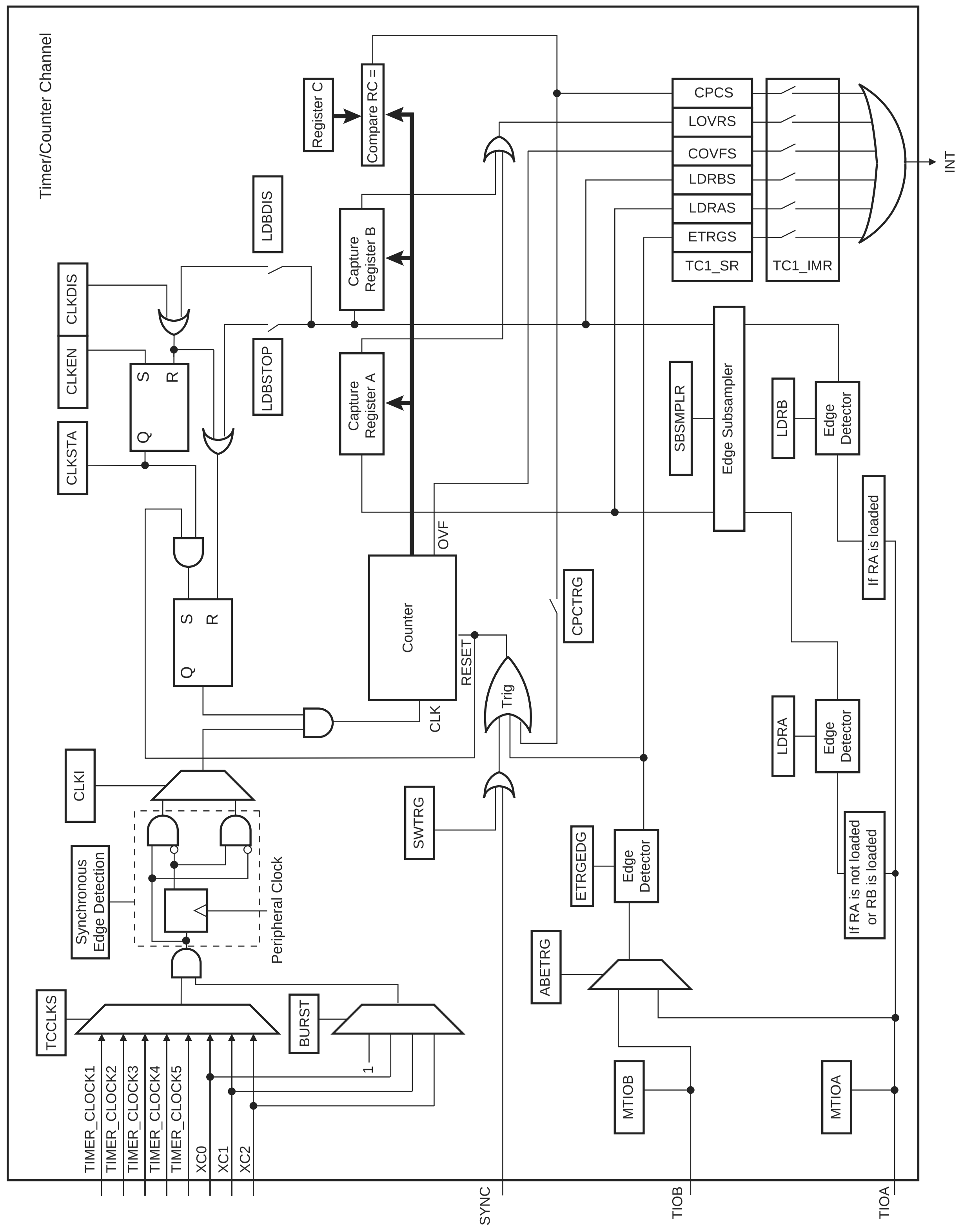
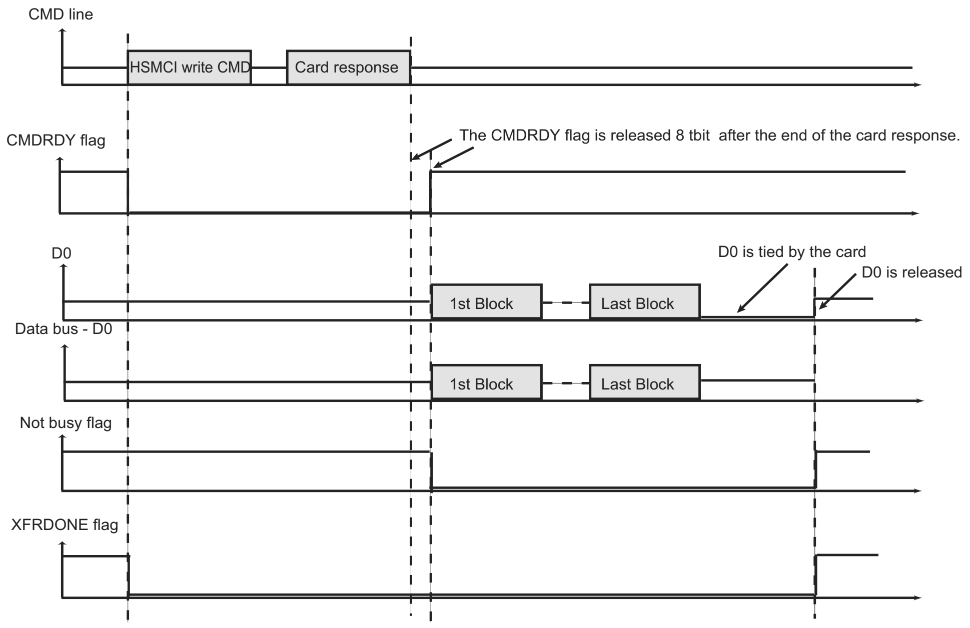


Figure 40-12. XFRDONE During a Write Access



42.8.14 GMAC PHY Maintenance Register

Name: GMAC_MAN

Address: 0x40034034

Access: Read/Write

31	30	29	28	27	26	25	24
WZO	CLTTO	OP		PHYA			

23	22	21	20	19	18	17	16
PHYA	REGA					WTN	

15	14	13	12	11	10	9	8
DATA							

7	6	5	4	3	2	1	0
DATA							

The PHY Maintenance Register is implemented as a shift register. Writing to the register starts a shift operation which is signalled as complete when bit 2 is set in the Network Status Register. 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. *See 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 0x0001. See Table 42-18.

Table 42-18. Clause 22/Clause 45 PHYs Read/Write Access Configuration (GMAC_MAN Bits 31:28)

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 Section 42.8.2 ”GMAC Network Configuration Register”.

- **DATA: PHY Data**

For a write operation this field is written with the data to be written to the PHY. After a read operation this field contains the data read from the PHY.

- **WTN: Write Ten**

Must be written to 10.

45. Analog Comparator Controller (ACC)

45.1 Description

The Analog Comparator Controller (ACC) configures the analog comparator and generates an interrupt depending on user settings. The analog comparator embeds two 8-to-1 multiplexers that generate two internal inputs. These inputs are compared, resulting in a compare output. The hysteresis level, edge detection and polarity are configurable.

The ACC also generates a compare event which can be used by the Pulse Width Modulator (PWM).

45.2 Embedded Characteristics

- Eight User Analog Inputs Selectable for Comparison
- Four Voltage References Selectable for Comparison: Temperature Sensor (TS), External Voltage Reference, DAC0 and DAC1
- Interrupt Generation
- Compare Event Fault Generation for PWM

45.3 Block Diagram

Figure 45-1. Analog Comparator Controller Block Diagram

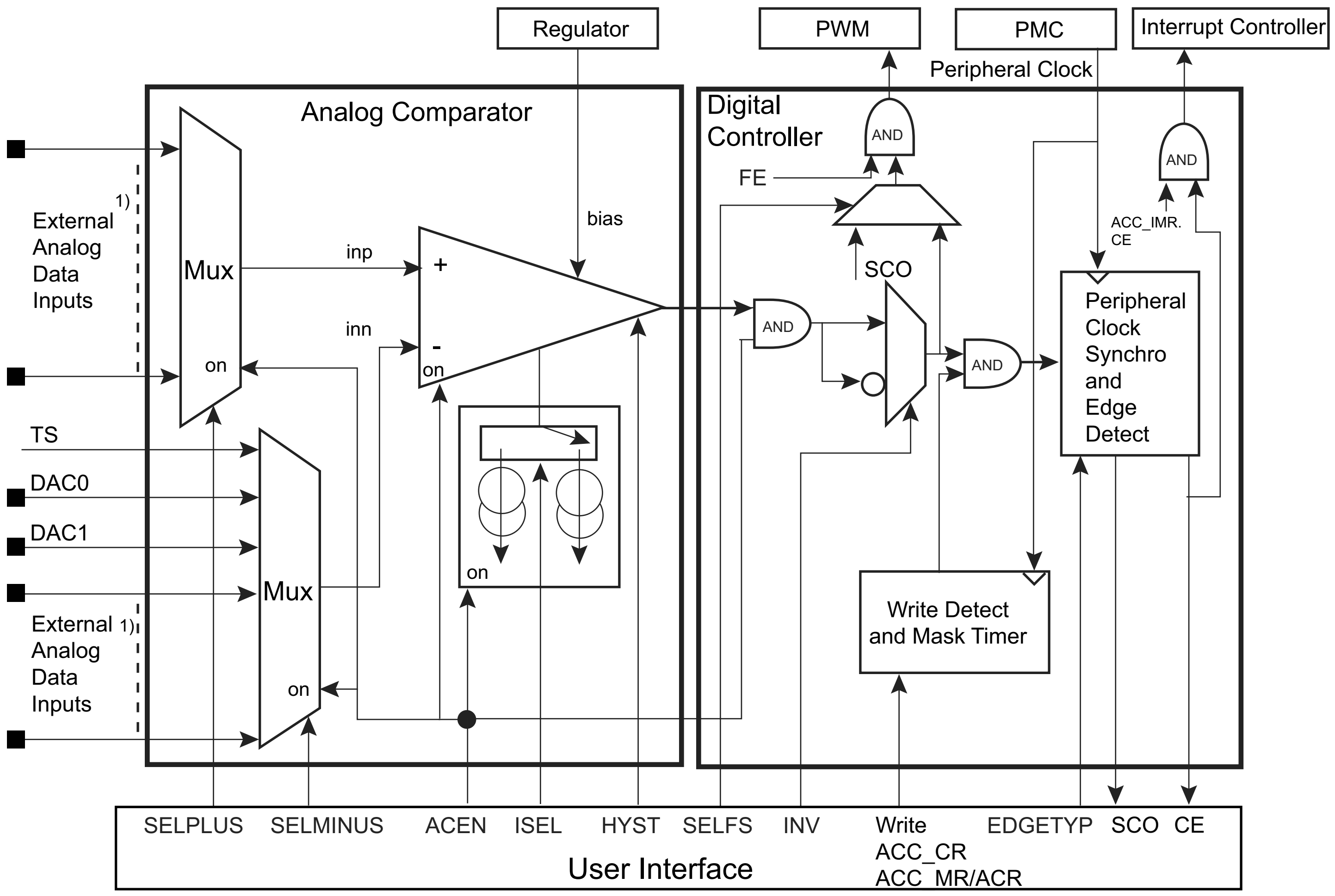
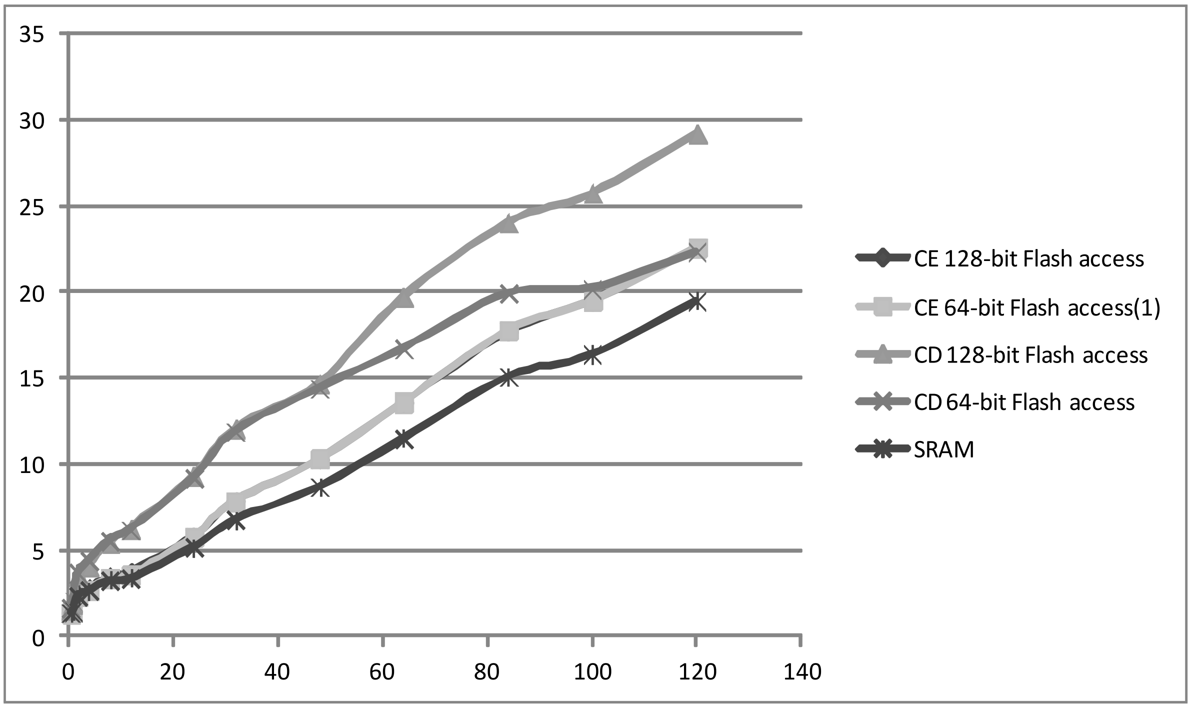


Figure 46-10. Active Total Power Consumption with VDDCORE @ 1.2V



46.4.8 Crystal Oscillator Design Considerations Information

46.4.8.1 Choosing a Crystal

When choosing a crystal for the 32.768 kHz Slow Clock Oscillator or for the 3–20 MHz oscillator, several parameters must be taken into account. Important parameters between crystal and SAM4E specifications are as follows:

- Load Capacitance

  $C_{crystal}$ is the equivalent capacitor value the oscillator must "show" to the crystal in order to oscillate at the target frequency. The crystal must be chosen according to the internal load capacitance (C_{LOAD}) of the on-chip oscillator. Having a mismatch for the load capacitance will result in a frequency drift.

- Drive Level

  Crystal Drive Level ≥ Oscillator Drive Level. Having a crystal drive level number lower than the oscillator specification may damage the crystal.

- Equivalent Series Resistor (ESR)

  Crystal ESR ≤ Oscillator ESR Max. Having a crystal with ESR value higher than the oscillator may cause the oscillator to not start.

- Shunt Capacitance

  Max. Crystal Shunt Capacitance ≤ Oscillator Shunt Capacitance (C_{SHUNT}). Having a crystal with ESR value higher than the oscillator may cause the oscillator to not start.

46.4.8.2 Printed Circuit Board (PCB)

SAM4E oscillators are low-power oscillators requiring particular attention when designing PCB systems.

46.5 PLLA Characteristics

Table 46-23. Supply Voltage Phase Lock Loop Characteristics

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
V_{DDPLLR}	Supply Voltage Range		1.08	1.2	1.32	V
$V_{rip(VDDPLL)}$	Allowable Voltage Ripple	RMS value 10 kHz to 10 MHz	—	—	20	mV
		RMS value > 10 MHz	—	—	10	

Table 46-24. PLLA Characteristics

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
f_{IN}	Input Frequency	—	3	—	32	MHz
f_{OUT}	Output Frequency	—	80	—	240	MHz
I_{PLL}	Current Consumption	Active mode @ 80 MHz @ 1.2V	—	0.94	1.2	mA
		Active mode @ 96 MHz @ 1.2V	—	1.2	1.5	
		Active mode @ 160 MHz @ 1.2V	—	2.1	2.5	
		Active Mode @ 240 MHz @ 1.2V	—	3.34	4	
t_s	Settling Time	—	—	60	150	µs

50.2.1 Watchdog

50.2.1.1 Watchdog Not Stopped in Wait Mode

When the Watchdog is enabled and the bit WAITMODE = 1 is used to enter Wait mode, the watchdog is not halted. If the time spent in Wait mode is longer than the Watchdog time-out, the device will be reset if Watchdog reset is enabled.

Problem Fix/Workaround

When entering Wait mode, the Wait For Event (WFE) instruction of the processor Cortex-M4 must be used with the SLEEPDEEP bit of the System Control Register (SCB_SCR) of the Cortex-M = 0.

50.2.2 Brownout Detector

50.2.2.1 Unpredictable Behavior if BOD is Disabled, VDDCORE is Lost and VDDIO is Connected

In Active mode or in Wait mode, if the Brownout Detector is disabled (SUPC_MR.BODDIS = 1) and power is lost on VDDCORE while VDDIO is powered, the device might not be properly reset and may behave unpredictably.

Problem Fix/Workaround

When the Brownout Detector is disabled in Active or in Wait mode, VDDCORE always needs to be powered.

Table 51-1. SAM4E Datasheet Rev. 11157H 31-Mar-2016 Revision History

Doc. Date	Changes
31-Mar-2016	Section 43. “Analog Front-End Controller (AFEC)”
	Section 43.7.13 “AFEC Interrupt Status Register”: defined EOCAL bit as ‘cleared on read’
	Section 43.6.1 “Analog Front-End Conversion”: updated section and added equations for AFE conversion time.
	Updated Section 43-2 “Sequence of AFE Conversions when Tracking Time > Conversion Time”
	Added sentence on write protection below the register table for:
	Section 43.7.21 “AFEC Channel Offset Compensation Register”
	Section 43.7.22 “AFEC Temperature Sensor Mode Register”
	Section 43.6.16 “Register Write Protection”: added ”AFEC Channel Differential Register” to the list of write-protected registers.
	Section 43.7.2 “AFEC Mode Register”: updated descriptions of fields TRACKTIM and TRANSFER
	Updated Warning in Section 43.6.10 “AFE Timings”
	Section 43.2 “Embedded Characteristics”: deleted bullet on conversion rate (redundant with Electrical Characteristics)
	Deleted Section 7.5 ”Conversion Results Format”.
	Section 43.6.7 “Comparison Window”: deleted paragraph on conversion sign.
	Section 43.7.3 “AFEC Extended Mode Register” bits 28 and 29 now reserved (were ‘SIGNMODE’)
	Section 43.7.21 “AFEC Channel Offset Compensation Register”: added note on configuration of AOFF.
	Section 43.7.19 “AFEC Channel Selection Register”: updated CSEL bit description.
	Section 43.6.9 “Input Gain and Offset”: updated information on AOFF field.
	Section 31. “Controller Area Network (CAN)”
	Updated MIDvA and MIDvB description in Section 31.9.15 “CAN Message Acceptance Mask Register”
	Updated Section 31.6.3 “Interrupt Sources” (replaced “Advanced Interrupt Controller” with ”interrupt controller”) and Figure 9-1, “Possible Initialization Procedure” (replaced “AIC” with “Interrupt Controller”)
	Section 31.8.3.2 “Transmission Handling”: in sixth paragraph, “CAN_MACR” remplaced with “CAN_ACR
	Section 46. “SAM4E Electrical Characteristics”
	Section 46.11.8.3 “MII Mode” : Removed note 1 below Table 46-66 “EMAC MII Timings”
	Deleted $t_{TRACKTIM}$ and t_s in Table 46.7.3 “ADC Timings”
	Modified Section 46.11.3 “SPI Characteristics”

Table 51-2. SAM4E Datasheet Rev. 11157G 12-Feb-2016 Revision History

Doc. Date	Changes
12-Feb-2016	Added MRLB (Rev. B) devices:
	- Updated Table 32-1 “SAM4E Chip ID Registers”.
	- Updated Section 50. “Errata on SAM4E Devices” (added Section 50.1.4 “Floating Point Unit (FPU)”, Section 50.2 “Errata SAM4E Rev.B Parts” and CHIPID information).
	- Table 49-1, “Ordering Codes for SAM4E Devices”.