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/M4F
Core Size	32-Bit Dual-Core
Speed	120MHz
Connectivity	EBI/EMI, I²C, IrDA, SPI, UART/USART
Peripherals	Brown-out Detect/Reset, DMA, LCD, POR, PWM, WDT
Number of I/O	52
Program Memory Size	2MB (2M x 8)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	256K x 8
Voltage - Supply (Vcc/Vdd)	1.62V ~ 3.6V
Data Converters	A/D 6x10b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°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/atsam4cmp32cb-aur

The SAM-BA Boot provides an interface with SAM-BA Graphic User Interface (GUI).

The SAM-BA Boot is in ROM and is mapped in Flash at address 0x0 when GPNVM bit 1 is set to 0.

8.1.4.10 GPNVM Bits

The SAM4CM features two (SAM4CM16/SAM4CM8/SAM4CM4) or three (SAM4CM32) GPNVM bits. These bits can be cleared or set respectively through the commands “Clear GPNVM Bit” and “Set GPNVM Bit” of the EEFC User Interface (refer to Section 22. “Enhanced Embedded Flash Controller (EEFC)”).

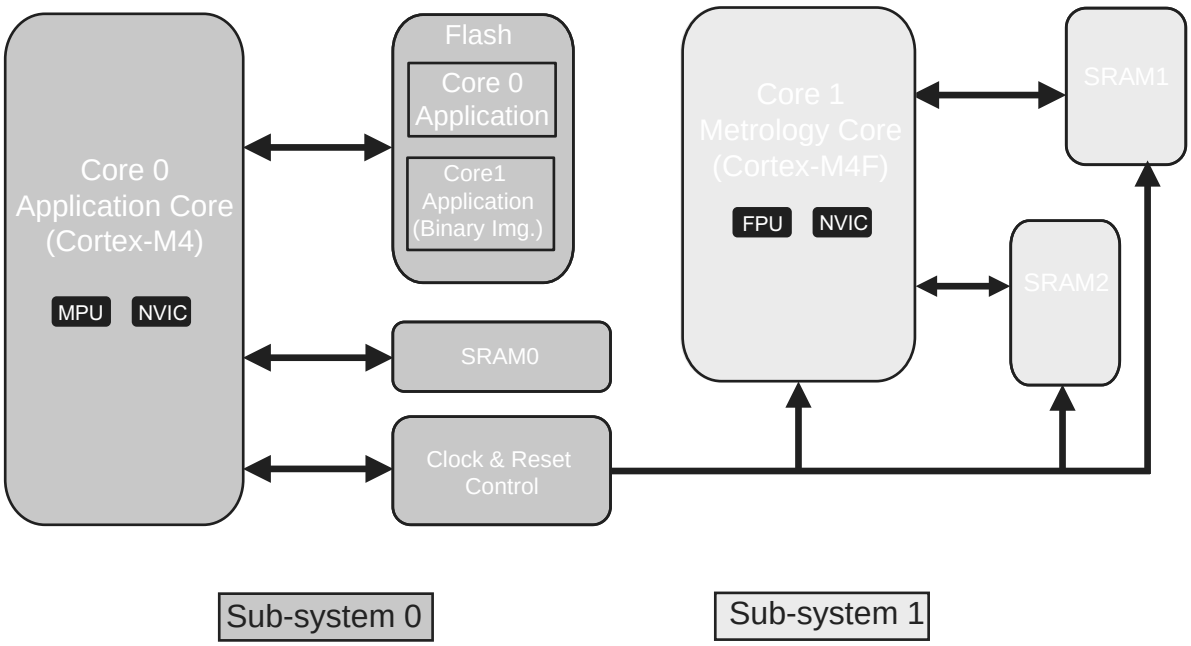
Table 8-3. General-purpose Nonvolatile Memory Bits

GPNVM Bit	Function
0	Security bit
1	Boot mode selection
2	Memory Plane Boot Selection (Plane 0 or Plane 1) (SAM4CM32 only)

8.1.5 Boot Strategy

Figure 8-5 below shows a load view of the memory at boot time.

Figure 8-5. Simplified Load View at Boot Time



Note: Matrices, AHB and APB Bridges are not represented.

8.1.5.1 Application Core (Core 0) Boot Process

The application processor (CM4P0) always boots at the address 0x0. To ensure maximum boot possibilities, the memory layout can be changed using a General-purpose NVM (GPNVM) bit. A GPNVM bit is used to boot either on the ROM (default) or from the Flash. The GPNVM bit can be cleared or set through the commands “Clear General-purpose NVM Bit” and “Set General-purpose NVM Bit” of the EEFC User Interface. Setting GPNVM Bit 1 selects the boot from Flash whereas clearing this bit selects the boot from ROM. Asserting ERASE clears the GPNVM Bit 1 and thus selects the boot from the ROM by default.

12.6.6.9 SMUAD and SMUSD

Signed Dual Multiply Add and Signed Dual Multiply Subtract

Syntax

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

where:

op — is one of:

SMUAD Signed Dual Multiply Add.

SMUADX Signed Dual Multiply Add Reversed.

SMUSD Signed Dual Multiply Subtract.

SMUSDX Signed Dual Multiply Subtract Reversed.

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

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

The SMUAD instruction interprets the values from the first and second operands as two signed halfwords in each operand. This instruction:

- Optionally rotates the halfwords of the second operand.
- Performs two signed 16 × 16-bit multiplications.
- Adds the two multiplication results together.
- Writes the result of the addition to the destination register.

The SMUSD instruction interprets the values from the first and second operands as two’s complement signed integers. This instruction:

- Optionally rotates the halfwords of the second operand.
- Performs two signed 16 × 16-bit multiplications.
- Subtracts the result of the top halfword multiplication from the result of the bottom halfword multiplication.
- Writes the result of the subtraction to the destination register.

Restrictions

In these instructions:

- Do not use SP and do not use PC.

Condition Flags

Sets the Q flag if the addition overflows. The multiplications cannot overflow.

12.6.10.4 TBB and TBH

Table Branch Byte and Table Branch Halfword.

Syntax

```
TBB [Rn, Rm]
TBH [Rn, Rm, LSL #1]
```

where:

Rn is the register containing the address of the table of branch lengths.

If *Rn* is PC, then the address of the table is the address of the byte immediately following the TBB or TBH instruction.

Rm is the index register. This contains an index into the table. For halfword tables, LSL #1 doubles the value in *Rm* to form the right offset into the table.

Operation

These instructions cause a PC-relative forward branch using a table of single byte offsets for TBB, or halfword offsets for TBH. *Rn* provides a pointer to the table, and *Rm* supplies an index into the table. For TBB the branch offset is twice the unsigned value of the byte returned from the table. and for TBH the branch offset is twice the unsigned value of the halfword returned from the table. The branch occurs to the address at that offset from the address of the byte immediately after the TBB or TBH instruction.

Restrictions

The restrictions are:

- *Rn* must not be SP
- *Rm* must not be SP and must not be PC
- When any of these instructions is used inside an IT block, it must be the last instruction of the IT block.

Condition Flags

These instructions do not change the flags.

12.6.11.10 VLDM

Floating-point Load Multiple

Syntax

```
VLDM{mode}{cond}{.size} Rn{!}, list
```

where:

mode — is the addressing mode:
- *IA* Increment After. The consecutive addresses start at the address specified in *Rn*.
- *DB* Decrement Before. The consecutive addresses end just before the address specified in *Rn*.

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

size — is an optional data size specifier.

Rn — is the base register. The SP can be used

! — is the command to the instruction to write a modified value back to *Rn*. This is required if mode == DB, and is optional if mode == IA.

list — is the list of extension registers to be loaded, as a list of consecutively numbered doubleword or singleword registers, separated by commas and surrounded by brackets.

Operation

This instruction loads:

- Multiple extension registers from consecutive memory locations using an address from an ARM core register as the base address.

Restrictions

The restrictions are:

- If *size* is present, it must be equal to the size in bits, 32 or 64, of the registers in *list*.
- For the base address, the SP can be used.
  In the ARM instruction set, if *!* is not specified the PC can be used.
- list must contain at least one register. If it contains doubleword registers, it must not contain more than 16 registers.
- If using the *Decrement Before addressing* mode, the write back flag, *!*, must be appended to the base register specification.

Condition Flags

These instructions do not change the flags.

- **MEMFAULTPENDED: Memory Management Fault Exception Pending**

Read:

0: The exception is not pending.

1: The exception is pending.

Note: The user can write to these bits to change the pending status of the exceptions.

- **USGFAULTPENDED: Usage Fault Exception Pending**

Read:

0: The exception is not pending.

1: The exception is pending.

Note: The user can write to these bits to change the pending status of the exceptions.

- **SYSTICKACT: SysTick Exception Active**

Read:

0: The exception is not active.

1: The exception is active.

Note: The user can write to these bits to change the active status of the exceptions.
- Caution: A software that changes the value of an active bit in this register without a correct adjustment to the stacked content can cause the processor to generate a fault exception. Ensure that the software writing to this register retains and subsequently restores the current active status.
- Caution: After enabling the system handlers, to change the value of a bit in this register, the user must use a read-modify-write procedure to ensure that only the required bit is changed.

- **PENDSVACT: PendSV Exception Active**

0: The exception is not active.

1: The exception is active.

- **MONITORACT: Debug Monitor Active**

0: Debug monitor is not active.

1: Debug monitor is active.

- **SVCALLACT: SVC Call Active**

0: SVC call is not active.

1: SVC call is active.

- **USGFAULTACT: Usage Fault Exception Active**

0: Usage fault exception is not active.

1: Usage fault exception is active.

- **BUSFAULTACT: Bus Fault Exception Active**

0: Bus fault exception is not active.

1: Bus fault exception is active.

- **MEMFAULTACT: Memory Management Fault Exception Active**

0: Memory management fault exception is not active.

1: Memory management fault exception is active.

12.9.1.15 Hard Fault Status Register

Name: SCB_HFSR

Access: Read/Write

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

The SCB_HFSR gives information about events that activate the hard fault handler. This register is read, write to clear. This means that bits in the register read normally, but writing a 1 to any bit clears that bit to 0.

- **DEBUGEVT: Reserved for Debug Use**

When writing to the register, write a 0 to this bit, otherwise the behavior is unpredictable.

- **FORCED: Forced Hard Fault**

It indicates a forced hard fault, generated by escalation of a fault with configurable priority that cannot be handles, either because of priority or because it is disabled:

0: No forced hard fault.

1: Forced hard fault.

When this bit is set to 1, the hard fault handler must read the other fault status registers to find the cause of the fault.

- **VECTTBL: Bus Fault on a Vector Table**

It indicates a bus fault on a vector table read during an exception processing:

0: No bus fault on vector table read.

1: Bus fault on vector table read.

This error is always handled by the hard fault handler.

When this bit is set to 1, the PC value stacked for the exception return points to the instruction that was preempted by the exception.

Note: The HFSR bits are sticky. This means that, as one or more fault occurs, the associated bits are set to 1. A bit that is set to 1 is cleared to 0 only by writing a 1 to that bit, or by a reset.

19.5.1 Reinforced Safety Watchdog Timer Control Register

Name: RSWDT_CR

Address: 0x400E1500

Access: Write-only

31	30	29	28	27	26	25	24
KEY							

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

15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–

7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	WDRSTT

- **WDRSTT: Watchdog Restart**

0: No effect.

1: Restarts the watchdog.

- **KEY: Password**

Value	Name	Description
0xC4	PASSWD	Writing any other value in this field aborts the write operation.

19.5.2 Reinforced Safety Watchdog Timer Mode Register

Name: RSWDT_MR

Address: 0x400E1504

Access: Read-write Once

31	30	29	28	27	26	25	24
–	–	WDIDLEHLT	WDDBGHLT	WDD			
23	22	21	20	19	18	17	16
WDD							
15	14	13	12	11	10	9	8
WDDIS	WDRPROC	WDRSTEN	–	WDV			
7	6	5	4	3	2	1	0
WDV							

Notes:
1. The first write access prevents any further modification of the value of this register; read accesses remain possible.
2. The WDD and WDV values must not be modified within three slow clock periods following a restart of the watchdog performed by means of a write access in the RSWDT_CR, else the watchdog may trigger an end of period earlier than expected.

- **WDV: Watchdog Counter Value**

Defines the value loaded in the 12-bit watchdog counter.

- **WDRSTEN: Watchdog Reset Enable**

0: A Watchdog fault (underflow or error) has no effect on the resets.

1: A Watchdog fault (underflow or error) triggers a watchdog reset.

- **WDRPROC: Watchdog Reset Processor**

0: If WDRSTEN is 1, a watchdog fault (underflow or error) activates all resets.

1: If WDRSTEN is 1, a watchdog fault (underflow or error) activates the processor reset.

- **WDD: Watchdog Delta Value**

Defines the permitted range for reloading the RSWDT.

If the RSWDT value is less than or equal to WDD, writing RSWDT_CR with WDRSTT = 1 restarts the timer.

If the RSWDT value is greater than WDD, writing RSWDT_CR with WDRSTT = 1 causes a Watchdog error.

- **WDDBGHLT: Watchdog Debug Halt**

0: The RSWDT runs when the processor is in debug state.

1: The RSWDT stops when the processor is in debug state.

- **WDIDLEHLT: Watchdog Idle Halt**

0: The RSWDT runs when the system is in Idle mode.

1: The RSWDT stops when the system is in idle state.

Figure 20-4. SAM4CM16/8/4 Wakeup Sources

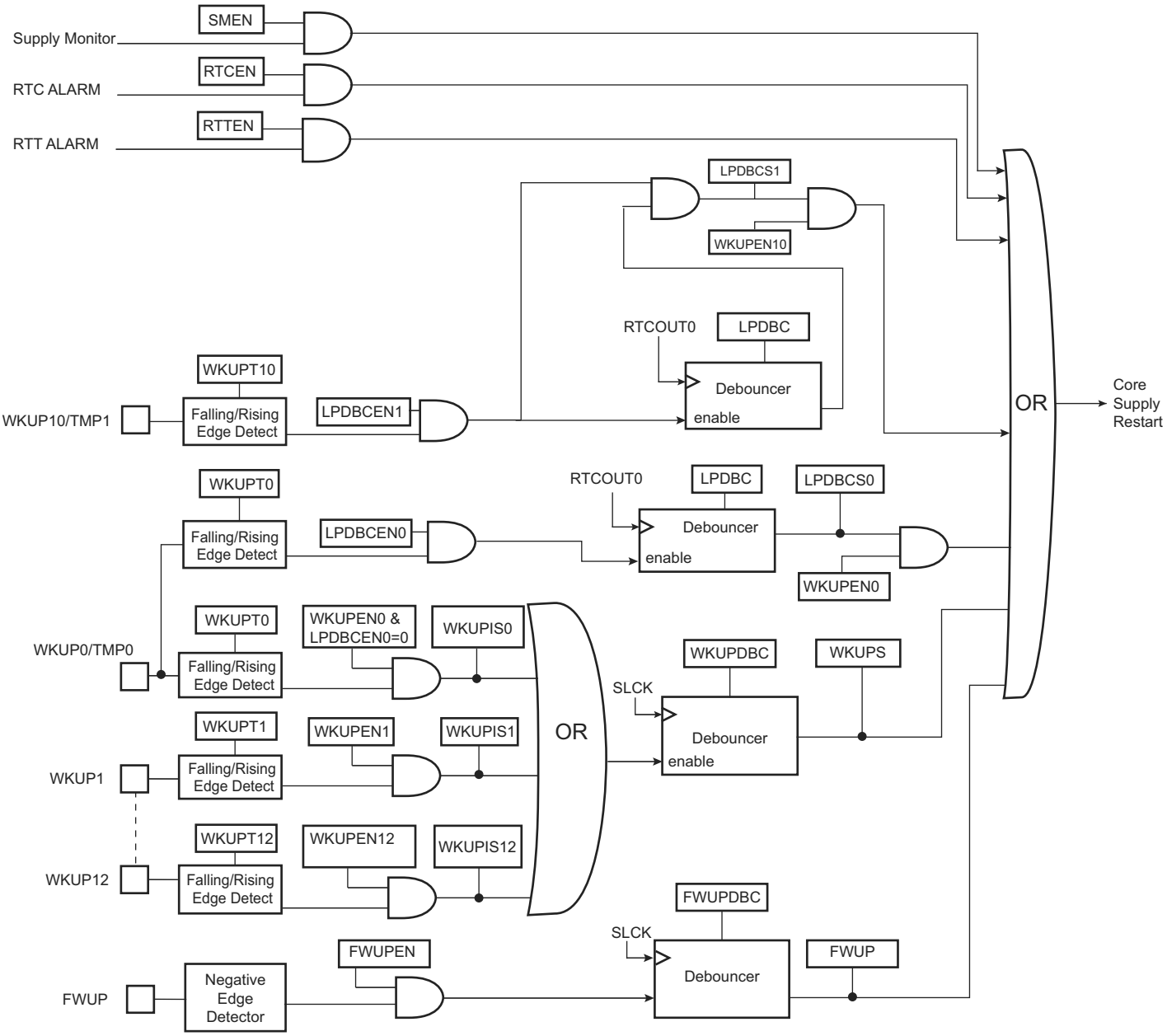
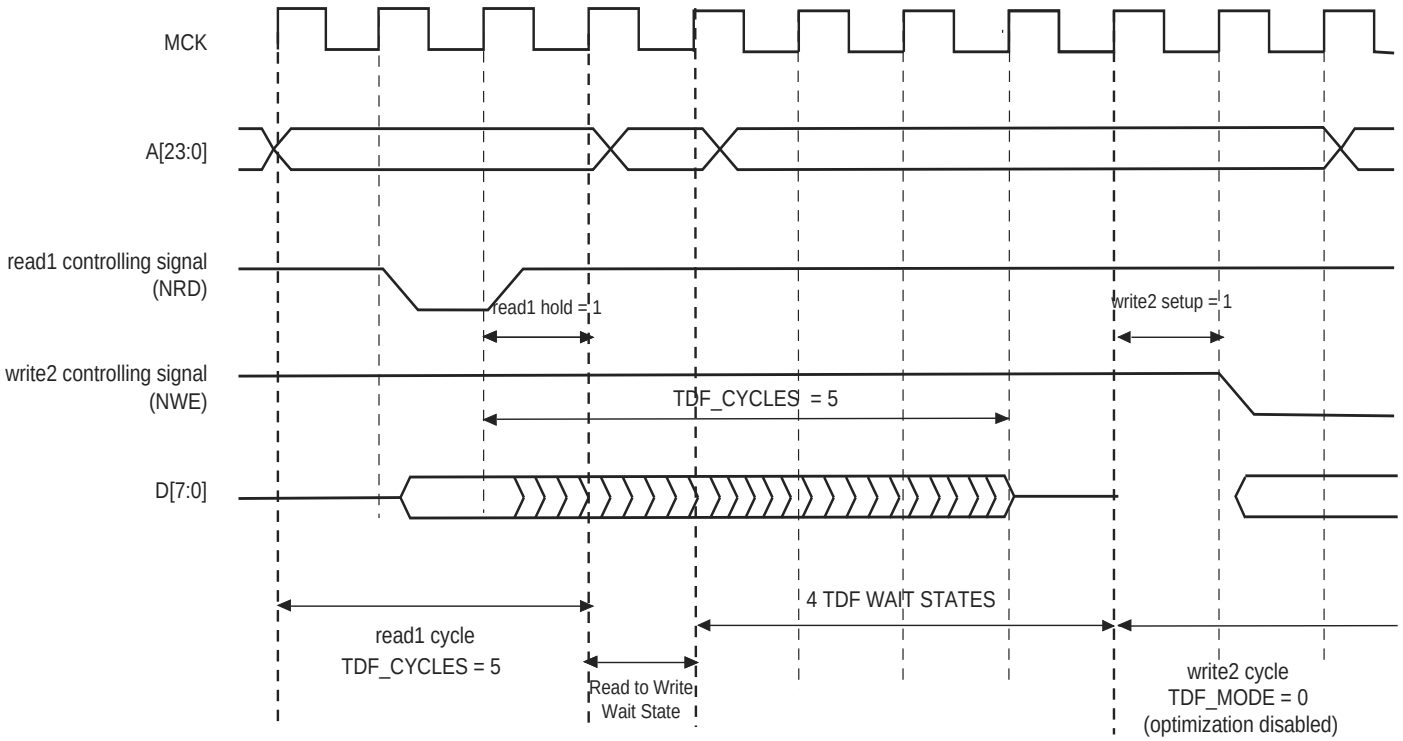


Figure 27-24. TDF Mode = 0: TDF wait states between read and write accesses on the same chip select



27.13 External Wait

Any access can be extended by an external device using the NWAIT input signal of the SMC. The EXNW_MODE field of the SMC_MODE register on the corresponding chip select must be set either to “10” (Frozen mode) or “11” (Ready mode). When the EXNW_MODE is set to “00” (disabled), the NWAIT signal is simply ignored on the corresponding chip select. The NWAIT signal delays the read or write operation in regards to the read or write controlling signal, depending on the Read and Write modes of the corresponding chip select.

27.13.1 Restriction

When one of the EXNW_MODE is enabled, **it is mandatory to program at least one hold cycle for the read/write controlling signal. For that reason, the NWAIT signal cannot be used in Page mode (Section 27.15 ”Asynchronous Page Mode”), or in Slow clock mode (Section 27.14 ”Slow Clock Mode”).**

The NWAIT signal is assumed to be a response of the external device to the read/write request of the SMC. Then NWAIT is examined by the SMC only in the pulse state of the read or write controlling signal. The assertion of the NWAIT signal outside the expected period has no impact on SMC behavior.

27.16.5 SMC OCMS Mode Register

Name: SMC_OCMS

Address: 0x400E0080 (0), 0x4801C080 (1)

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	CS3SE	CS2SE	CS1SE	CS0SE
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	SMSE

- **CSxSE: Chip Select (x = 0 to 3) Scrambling Enable**

0: Disable scrambling for CSx.

1: Enable scrambling for CSx.

- **SMSE: Static Memory Controller Scrambling Enable**

0: Disable scrambling for SMC access.

1: Enable scrambling for SMC access.

33.8.3 SPI Receive Data Register

Name: SPI_RDR

Address: 0x40008008 (0), 0x48000008 (1)

Access: Read-only

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

- **RD: Receive Data**

Data received by the SPI Interface is stored in this register in a right-justified format. Unused bits are read as zero.

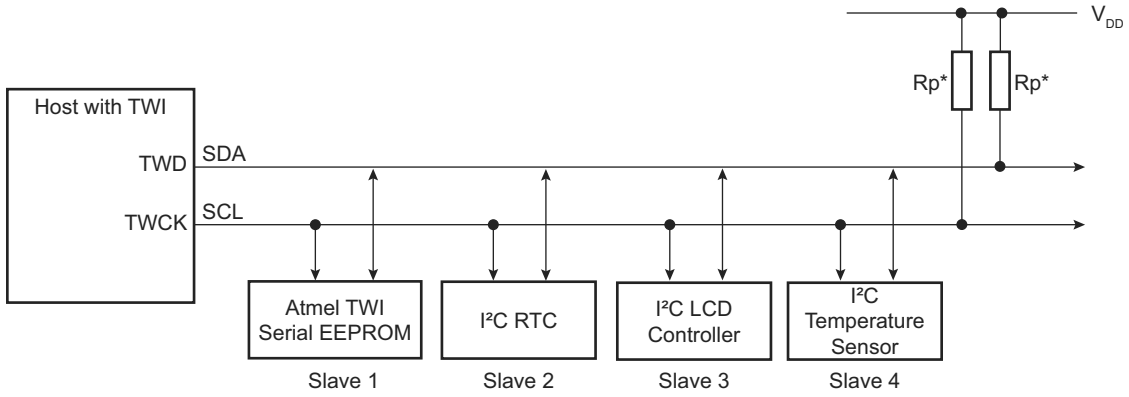
- **PCS: Peripheral Chip Select**

In Master mode only, these bits indicate the value on the NPCS pins at the end of a transfer. Otherwise, these bits are read as zero.

Note: When using Variable peripheral select mode (PS = 1 in SPI_MR), it is mandatory to set the SPI_MR.WDRBT bit to 1 if the PCS field must be processed in SPI_RDR.

34.7.3.2 Application Block Diagram

Figure 34-4. Master Mode Typical Application Block Diagram



* Rp: Pull-up value as given by the I²C Standard

34.7.3.3 Programming Master Mode

The following fields must be programmed before entering Master mode:

1. TWI_MMR.DADR (+ IADRSZ + IADR if a 10-bit device is addressed): The device address is used to access slave devices in Read or Write mode.
2. TWI_CWGR.CKDIV + CHDIV + CLDIV: Clock waveform.
3. TWI_CR.SVDIS: Disables the Slave mode
4. TWI_CR.MSEN: Enables the Master mode

Note: If the TWI is already in Master mode, the device address (DADR) can be configured without disabling the Master mode.

34.7.3.4 Master Transmitter Mode

After the master initiates a START condition when writing into the Transmit Holding register (TWI_THR), it sends a 7-bit slave address, configured in the Master Mode register (DADR in TWI_MMR), to notify the slave device. The bit following the slave address indicates the transfer direction—0 in this case (MREAD = 0 in TWI_MMR).

The TWI transfers require the slave to acknowledge each received byte. During the acknowledge clock pulse (9th pulse), the master releases the data line (HIGH), enabling the slave to pull it down in order to generate the acknowledge. If the slave does not acknowledge the byte, then the Not Acknowledge flag (NACK) is set in the TWI Status Register (TWI_SR) of the master and a STOP condition is sent. The NACK flag must be cleared by reading the TWI Status Register (TWI_SR) before the next write into the TWI Transmit Holding Register(TWI_THR). As with the other status bits, an interrupt can be generated if enabled in the Interrupt Enable register (TWI_IER). If the slave acknowledges the byte, the data written in the TWI_THR is then shifted in the internal shifter and transferred. When an acknowledge is detected, the TXRDY bit is set until a new write in the TWI_THR.

TXRDY is used as Transmit Ready for the PDC transmit channel.

While no new data is written in the TWI_THR, the serial clock line (SCL) is tied low. When new data is written in the TWI_THR, the TWCK/SCL is released and the data is sent. Setting the STOP bit in TWI_CR generates a STOP condition.

After a master write transfer, the SCL is stretched (tied low) as long as no new data is written in the TWI_THR or until a STOP command is performed.

See Figure 34-5, Figure 34-6, and Figure 34-7.

34.8.7 TWI Interrupt Enable Register

Name: TWI_IER

Address: 0x40018024 (0), 0x4001C024 (1)

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
TXBUFE	RXBUFF	ENDTX	ENDRX	EOSACC	SCL_WS	ARBLST	NACK
7	6	5	4	3	2	1	0
–	OVRE	GACC	SVACC	–	TXRDY	RXRDY	TXCOMP

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

0: No effect.

1: Enables the corresponding interrupt.

- **TXCOMP: Transmission Completed Interrupt Enable**
- **RXRDY: Receive Holding Register Ready Interrupt Enable**
- **TXRDY: Transmit Holding Register Ready Interrupt Enable**
- **SVACC: Slave Access Interrupt Enable**
- **GACC: General Call Access Interrupt Enable**
- **OVRE: Overrun Error Interrupt Enable**
- **NACK: Not Acknowledge Interrupt Enable**
- **ARBLST: Arbitration Lost Interrupt Enable**
- **SCL_WS: Clock Wait State Interrupt Enable**
- **EOSACC: End Of Slave Access Interrupt Enable**
- **ENDRX: End of Receive Buffer Interrupt Enable**
- **ENDTX: End of Transmit Buffer Interrupt Enable**
- **RXBUFF: Receive Buffer Full Interrupt Enable**
- **TXBUFE: Transmit Buffer Empty Interrupt Enable**

35.6.1 UART Control Register

Name: UART_CR

Address: 0x400E0600 (0), 0x48004000 (1)

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
–	–	–	–	–	–	–	RSTSTA
7	6	5	4	3	2	1	0
TXDIS	TXEN	RXDIS	RXEN	RSTTX	RSTRX	–	–

- **RSTRX: Reset Receiver**

0: No effect.

1: The receiver logic is reset and disabled. If a character is being received, the reception is aborted.

- **RSTTX: Reset Transmitter**

0: No effect.

1: The transmitter logic is reset and disabled. If a character is being transmitted, the transmission is aborted.

- **RXEN: Receiver Enable**

0: No effect.

1: The receiver is enabled if RXDIS is 0.

- **RXDIS: Receiver Disable**

0: No effect.

1: The receiver is disabled. If a character is being processed and RSTRX is not set, the character is completed before the receiver is stopped.

- **TXEN: Transmitter Enable**

0: No effect.

1: The transmitter is enabled if TXDIS is 0.

- **TXDIS: Transmitter Disable**

0: No effect.

1: The transmitter is disabled. If a character is being processed and a character has been written in the UART_THR and RSTTX is not set, both characters are completed before the transmitter is stopped.

- **RSTSTA: Reset Status**

0: No effect.

1: Resets the status bits PARE, FRAME and OVRE in the UART_SR.

Table 37-4. I/O Lines (Continued)

TC1	TIOB3	PB25	A
TC1	TIOB4	PA16	B
TC1	TIOB5	PA20	B

37.5.2 Power Management

The TC is clocked through the Power Management Controller (PMC), thus the programmer must first configure the PMC to enable the Timer Counter clock of each channel.

37.5.3 Interrupt Sources

The TC has an interrupt line per channel connected to the interrupt controller. Handling the TC interrupt requires programming the interrupt controller before configuring the TC.

Table 37-5. Peripheral IDs

Instance	ID
TC0	23
TC1	24

37.6 Functional Description

37.6.1 Description

All channels of the Timer Counter are independent and identical in operation except when the QDEC is enabled. The registers for channel programming are listed in Table 37-6 “Register Mapping”.

37.6.2 16-bit Counter

Each 16-bit channel is organized around a 16-bit counter. The value of the counter is incremented at each positive edge of the selected clock. When the counter has reached the value 2^{16}-1 and passes to zero, an overflow occurs and the COVFS bit in the TC Status Register (TC_SR) is set.

The current value of the counter is accessible in real time by reading the TC Counter Value Register (TC_CV). The counter can be reset by a trigger. In this case, the counter value passes to zero on the next valid edge of the selected clock.

37.6.3 Clock Selection

At block level, input clock signals of each channel can either be connected to the external inputs TCLK0, TCLK1 or TCLK2, or be connected to the internal I/O signals TIOA0, TIOA1 or TIOA2 for chaining by programming the TC Block Mode Register (TC_BMR). See Figure 37-2.

Each channel can independently select an internal or external clock source for its counter:

- External clock signals(1): XC0, XC1 or XC2
- Internal clock signals: MCK/2, MCK/8, MCK/32, MCK/128, SLCK

This selection is made by the TCCLKS bits in the TC Channel Mode Register (TC_CMR).

The selected clock can be inverted with the CLKI bit in the TC_CMR. This allows counting on the opposite edges of the clock.

The burst function allows the clock to be validated when an external signal is high. The BURST parameter in the TC_CMR defines this signal (none, XC0, XC1, XC2). See Figure 37-3.

37.7.9 TC Status Register

Name: TC_SRx [x=0..2]

Address: 0x40010020 (0)[0], 0x40010060 (0)[1], 0x400100A0 (0)[2], 0x40014020 (1)[0], 0x40014060 (1)[1], 0x400140A0 (1)[2]

Access: Read-only

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

- **COVFS: Counter Overflow Status (cleared on read)**

0: No counter overflow has occurred since the last read of the Status Register.

1: A counter overflow has occurred since the last read of the Status Register.

- **LOVRS: Load Overrun Status (cleared on read)**

0: Load overrun has not occurred since the last read of the Status Register or TC_CMRx.WAVE = 1.

1: RA or RB have been loaded at least twice without any read of the corresponding register since the last read of the Status Register, if TC_CMRx.WAVE = 0.

- **CPAS: RA Compare Status (cleared on read)**

0: RA Compare has not occurred since the last read of the Status Register or TC_CMRx.WAVE = 0.

1: RA Compare has occurred since the last read of the Status Register, if TC_CMRx.WAVE = 1.

- **CPBS: RB Compare Status (cleared on read)**

0: RB Compare has not occurred since the last read of the Status Register or TC_CMRx.WAVE = 0.

1: RB Compare has occurred since the last read of the Status Register, if TC_CMRx.WAVE = 1.

- **CPCS: RC Compare Status (cleared on read)**

0: RC Compare has not occurred since the last read of the Status Register.

1: RC Compare has occurred since the last read of the Status Register.

- **LDRAS: RA Loading Status (cleared on read)**

0: RA Load has not occurred since the last read of the Status Register or TC_CMRx.WAVE = 1.

1: RA Load has occurred since the last read of the Status Register, if TC_CMRx.WAVE = 0.

- **LDRBS: RB Loading Status (cleared on read)**

0: RB Load has not occurred since the last read of the Status Register or TC_CMRx.WAVE = 1.

1: RB Load has occurred since the last read of the Status Register, if TC_CMRx.WAVE = 0.

41. Energy Metering Analog Front End (EMAFE)

41.1 Description

The Energy Metering Analog Front End peripheral (EMAFE) embeds four or seven high-resolution Sigma-Delta Analog-to-Digital Converters followed by SINC decimation filters running at an output data rate of 16kS/s. The two or four current measurement channels feature a low noise programmable gain amplifier to accommodate any type of current sensor configured in any type of IEC/ANSI-C application. One of these channels is dedicated to neutral current measurement to implement anti-tamper functions.

The EMAFE also embeds a high-performance voltage reference and a die temperature sensor. The temperature characteristics of these functions are measured during manufacturing and stored in an internal read-only memory. A low-cost and efficient voltage reference temperature correction can then be implemented at software level.

41.2 Embedded Characteristics

- Single-phase, Two-phase or Three-phase Energy Metering Analog Front End
- Works with the Atmel MCU Metrology Library
- Compliant with Class 0.2 standards (ANSI C12.20-2002 and IEC 62053-22)
- Acquisition Channels
  - Four or Seven Sigma-Delta ADC Measurement Channels: Two or Three Voltages, Two or Four Currents, 20-bit Resolution - 102 dB Dynamic Range
  - Current Channels with Pre-Gain (x1, x2, x4, x8)
  - Supports Shunt, Current Transformer and Rogowsky Coils
  - Direct Connection of Sensors Without External Preamplifier
  - Dedicated Current Channel for Anti-tamper Measurement
  - Integrated SINC Decimation Filters. Output Data Rate: 16kSps
- Precision Voltage Reference
  - Standard 1.2V Output Voltage With Possible External Bypass
  - Temperature Drift: 10 ppm Typical With Software Correction
- Factory-measured Temperature Drift and On-board Temperature Sensor to Perform Software Correction
- Integrated 2.8V LDO Regulator To Supply Analog Functions
- 3.0V to 3.6V Operation, Ultra-Low-Power at < 2.5mW per Channel @3.3V
- Specified Over T_J: -40°C to +100°C

42.5.3 AES Interrupt Enable Register

Name: AES_IER

Address: 0x40000010

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	TAGRDY
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	URAD
7	6	5	4	3	2	1	0
–	–	–	TXBUFE	RXBUFF	ENDTX	ENDRX	DATRDY

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

0: No effect.

1: Enables the corresponding interrupt.

- **DATRDY: Data Ready Interrupt Enable**
- **ENDRX: End of Receive Buffer Interrupt Enable**
- **ENDTX: End of Transmit Buffer Interrupt Enable**
- **RXBUFF: Receive Buffer Full Interrupt Enable**
- **TXBUFE: Transmit Buffer Empty Interrupt Enable**
- **URAD: Unspecified Register Access Detection Interrupt Enable**
- **TAGRDY: GCM Tag Ready Interrupt Enable**

45.6 True Random Number Generator (TRNG) User Interface

Table 45-2. Register Mapping

Offset	Register	Name	Access	Reset
0x00	Control Register	TRNG_CR	Write-only	_
0x00–0x0C	Reserved	–	_	–
0x10	Interrupt Enable Register	TRNG_IER	Write-only	–
0x14	Interrupt Disable Register	TRNG_IDR	Write-only	–
0x18	Interrupt Mask Register	TRNG_IMR	Read-only	0x0000_0000
0x1C	Interrupt Status Register	TRNG_ISR	Read-only	0x0000_0000
0x20–0x4C	Reserved	–	_	–
0x50	Output Data Register	TRNG_ODATA	Read-only	0x0000_0000
0x54–0xFC	Reserved	–	_	–