**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 | 512KB (512K 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-LQFP |
| Supplier Device Package | 144-LQFP (20x20) |
| Purchase URL | https://www.e-xfl.com/product-detail/microchip-technology/atsam4e8ea-aur |

#### 12.4.2.1 Memory Regions, Types and Attributes

The memory map and the programming of the MPU split the memory map into regions. Each region has a defined memory type, and some regions have additional memory attributes. The memory type and attributes determine the behavior of accesses to the region.

*Memory Types*

- **Normal**
  The processor can re-order transactions for efficiency, or perform speculative reads.
- **Device**
  The processor preserves transaction order relative to other transactions to Device or Strongly-ordered memory.
- **Strongly-ordered**
  The processor preserves transaction order relative to all other transactions.

The different ordering requirements for Device and Strongly-ordered memory mean that the memory system can buffer a write to Device memory, but must not buffer a write to Strongly-ordered memory.

*Additional Memory Attributes*

- **Shareable**
  For a shareable memory region, the memory system provides data synchronization between bus masters in a system with multiple bus masters, for example, a processor with a DMA controller.
  Strongly-ordered memory is always shareable.
  If multiple bus masters can access a non-shareable memory region, the software must ensure data coherency between the bus masters.
- **Execute Never (XN)**
  Means the processor prevents instruction accesses. A fault exception is generated only on execution of an instruction executed from an XN region.

#### 12.4.2.2 Memory System Ordering of Memory Accesses

For most memory accesses caused by explicit memory access instructions, the memory system does not guarantee that the order in which the accesses complete matches the program order of the instructions, providing this does not affect the behavior of the instruction sequence. Normally, if correct program execution depends on two memory accesses completing in program order, the software must insert a memory barrier instruction between the memory access instructions, see “Software Ordering of Memory Accesses” .

However, the memory system does guarantee some ordering of accesses to Device and Strongly-ordered memory. For two memory access instructions A1 and A2, if A1 occurs before A2 in program order, the ordering of the memory accesses is described below.

**Table 12-3. Ordering of the Memory Accesses Caused by Two Instructions**

| A2 / A1 | Normal Access | Device Access | | Strongly-ordered Access |
|---|---|---|---|---|
| | | Non-shareable | Shareable | |
| Normal Access | – | – | – | – |
| Device access, non-shareable | – | < | – | < |
| Device access, shareable | – | – | < | < |
| Strongly-ordered access | – | < | < | < |

Where:

– Means that the memory system does not guarantee the ordering of the accesses.

< Means that accesses are observed in program order, that is, A1 is always observed before A2.

#### 12.6.11.11 VLDR

Loads a single extension register from memory

Syntax

```
VLDR{cond}{.64} Dd, [Rn{#imm}]
VLDR{cond}{.64} Dd, label
VLDR{cond}{.64} Dd, [PC, #imm}]
VLDR{cond}{.32} Sd, [Rn {, #imm}]
VLDR{cond}{.32} Sd, label
VLDR{cond}{.32} Sd, [PC, #imm]
```

where:

| | |
|---|---|
| cond | is an optional condition code, see “Conditional Execution” . |
| 64, 32 | are the optional data size specifiers. |
| Dd | is the destination register for a doubleword load. |
| Sd | is the destination register for a singleword load. |
| Rn | is the base register. The SP can be used. |
| imm | is the + or - immediate offset used to form the address.<br>Permitted address values are multiples of 4 in the range 0 to 1020. |
| label | is the label of the literal data item to be loaded. |

Operation

This instruction:

- Loads a single extension register from memory, using a base address from an ARM core register, with an optional offset.

Restrictions

There are no restrictions.

Condition Flags

These instructions do not change the flags.

### 24.5.8 Cache Controller Monitor Enable Register

**Name:** CMCC_MEN

**Address:** 0x400C402C

**Access:** Write-only

**Reset:** 0x00002000

| 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 |
| – | – | – | – | – | – | – | MENABLE |

- **MENABLE: Cache Controller Monitor Enable**

0: When set to 0, the monitor counter is disabled.

1: When set to 1, the monitor counter is activated.

**Figure 28-14. Early Read Wait State: Write with No Hold Followed by Read with No Setup**

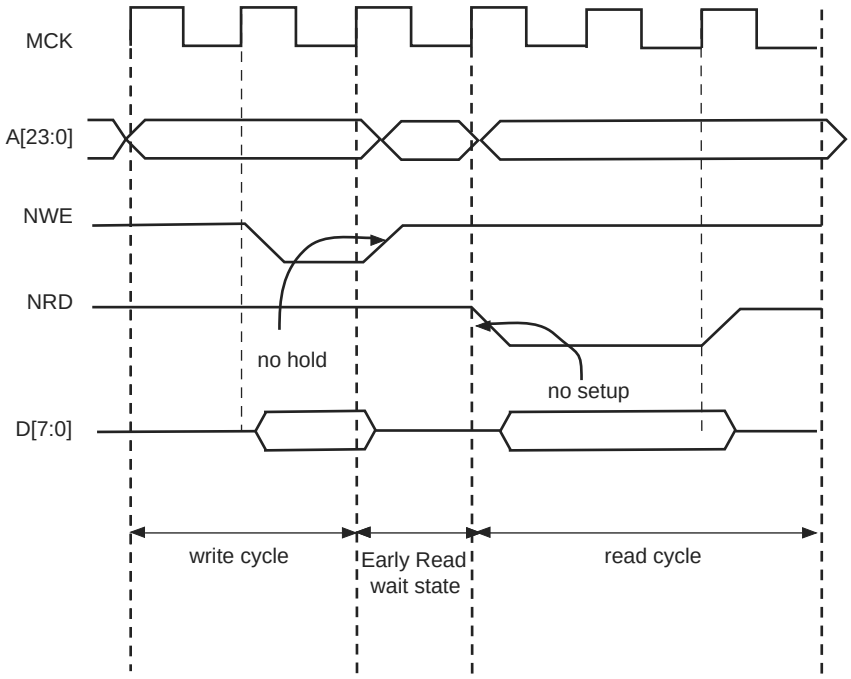
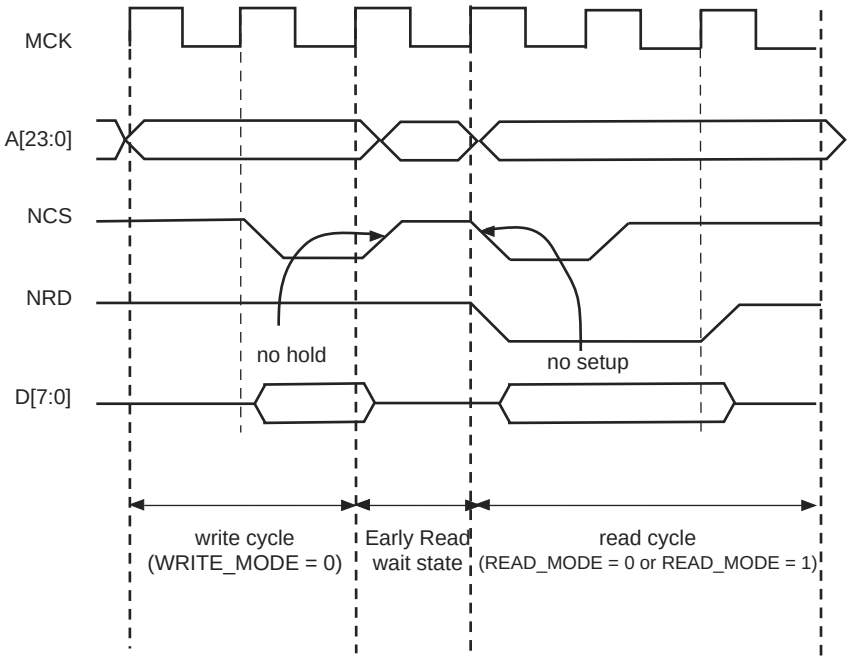



**Figure 28-15. Early Read Wait State: NCS Controlled Write with No Hold Followed by a Read with No NCS Setup**



## 28.11 Data Float Wait States

Some memory devices are slow to release the external bus. For such devices, it is necessary to add wait states (data float wait states) after a read access:

- before starting a read access to a different external memory
- before starting a write access to the same device or to a different external one.

The Data Float Output Time ($t_{DF}$) for each external memory device is programmed in the TDF_CYCLES field of the SMC_MODE register for the corresponding chip select. The value of TDF_CYCLES indicates the number of data float wait cycles (between 0 and 15) before the external device releases the bus, and represents the time allowed for the data output to go to high impedance after the memory is disabled.

Data float wait states do not delay internal memory accesses. Hence, a single access to an external memory with long $t_{DF}$ will not slow down the execution of a program from internal memory.

The data float wait states management depends on the READ_MODE and the TDF_MODE fields of the SMC_MODE register for the corresponding chip select.

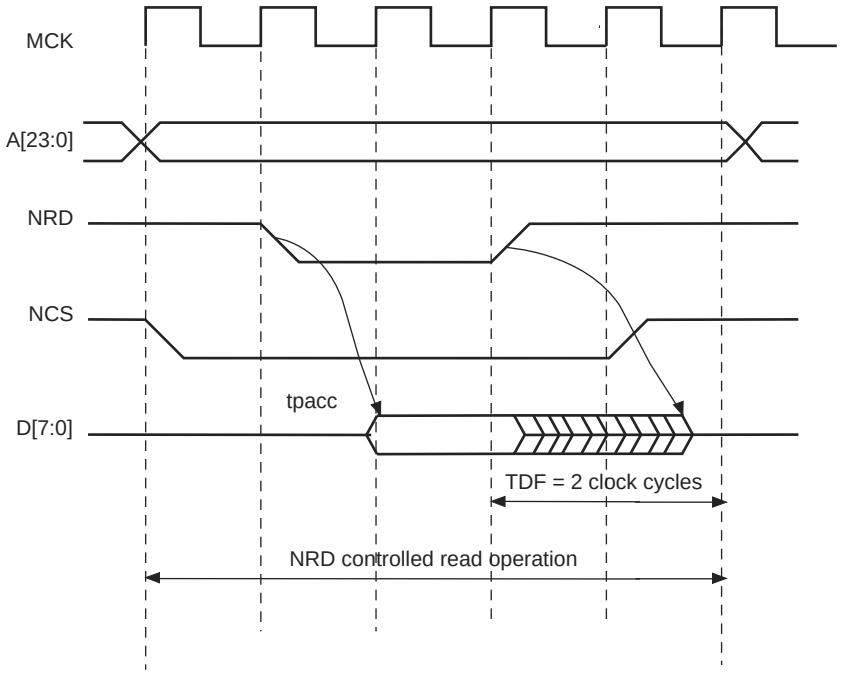
### 28.11.1 READ_MODE

Setting the READ_MODE to 1 indicates to the SMC that the NRD signal is responsible for turning off the tri-state buffers of the external memory device. The Data Float Period then begins after the rising edge of the NRD signal and lasts TDF_CYCLES MCK cycles.

When the read operation is controlled by the NCS signal (READ_MODE = 0), the TDF field gives the number of MCK cycles during which the data bus remains busy after the rising edge of NCS.

Figure 28-17 illustrates the Data Float Period in NRD-controlled mode (READ_MODE =1), assuming a data float period of 2 cycles (TDF_CYCLES = 2). Figure 28-18 shows the read operation when controlled by NCS (READ_MODE = 0) and the TDF_CYCLES parameter equals 3.

**Figure 28-17. TDF Period in NRD Controlled Read Access (TDF = 2)**



In page mode, the programming of the read timings is described in Table 28-6:

**Table 28-6. Programming of Read Timings in Page Mode**

| Parameter | Value | Definition |
|---|---|---|
| READ_MODE | 'x' | No impact |
| NCS_RD_SETUP | 'x' | No impact |
| NCS_RD_PULSE | $t_{pa}$ | Access time of first access to the page |
| NRD_SETUP | 'x' | No impact |
| NRD_PULSE | $t_{sa}$ | Access time of subsequent accesses in the page |
| NRD_CYCLE | 'x' | No impact |

The SMC does not check the coherency of timings. It will always apply the NCS_RD_PULSE timings as page access timing ($t_{pa}$) and the NRD_PULSE for accesses to the page ($t_{sa}$), even if the programmed value for $t_{pa}$ is shorter than the programmed value for $t_{sa}$.

### 28.14.2 Page Mode Restriction

The page mode is not compatible with the use of the NWAIT signal. Using the page mode and the NWAIT signal may lead to unpredictable behavior.

### 28.14.3 Sequential and Non-sequential Accesses

If the chip select and the MSB of addresses as defined in Table 28-5 are identical, then the current access lies in the same page as the previous one, and no page break occurs.

Using this information, all data within the same page, sequential or not sequential, are accessed with a minimum access time ($t_{sa}$). Figure 28-32 illustrates access to an 8-bit memory device in page mode, with 8-byte pages. Access to D1 causes a page access with a long access time ($t_{pa}$). Accesses to D3 and D7, though they are not sequential accesses, only require a short access time ($t_{sa}$).

If the MSB of addresses are different, the SMC performs the access of a new page. In the same way, if the chip select is different from the previous access, a page break occurs. If two sequential accesses are made to the page mode memory, but separated by an other internal or external peripheral access, a page break occurs on the second access because the chip select of the device was deasserted between both accesses.

### 33.7.49 PIO I/O Delay Register

**Name:** PIO_DELAYR

**Address:** 0x400E0F10 (PIOA), 0x400E1110 (PIOB), 0x400E1310 (PIOC), 0x400E1510 (PIOD), 0x400E1710 (PIOE)

**Access:** Read-write

**Reset:** See Table 33-3

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|
| Delay7 | | | | Delay6 | | | |
| 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| Delay5 | | | | Delay4 | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| Delay3 | | | | Delay2 | | | |
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Delay1 | | | | Delay0 | | | |

- **Delayx [x=0..7]: Delay Control for Simultaneous Switch Reduction**

Gives the number of elements in the delay line associated to pad x.

### 35.12.13TWI Write Protection Status Register

**Name:** TWI_WPSR

**Address:** 0x400A80E8 (0), 0x400AC0E8 (1)

**Access:** Read-only

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

- **WPVS: Write Protect Violation Status**

0: No Write Protect Violation has occurred since the last read of the TWI_WPSR register.

1: A Write Protect Violation has occurred since the last read of the TWI_WPSR register. If this violation is an unauthorized attempt to write a protected register, the associated violation is reported into field WPVSRC.

- **WPVSRC: Write Protect Violation Source**

When WPVS is active, this field indicates the write-protected register (through address offset or code) in which a write access has been attempted.

Note: Reading TWI_WPSR automatically clears all fields.

- **CLKO: Clock Output Select**

0: The USART does not drive the SCK pin.

1: The USART drives the SCK pin if USCLKS does not select the external clock SCK.

- **OVER: Oversampling Mode**

0: 16x Oversampling.

1: 8x Oversampling.

- **INACK: Inhibit Non Acknowledge**

0: The NACK is generated.

1: The NACK is not generated.

- **DSNACK: Disable Successive NACK**

0: NACK is sent on the ISO line as soon as a parity error occurs in the received character (unless INACK is set).

1: Successive parity errors are counted up to the value specified in the MAX_ITERATION field. These parity errors generate a NACK on the ISO line. As soon as this value is reached, no additional NACK is sent on the ISO line. The flag ITERATION is asserted.

- **INVDATA: Inverted Data**

0: The data field transmitted on TXD line is the same as the one written in US_THR register or the content read in US_RHR is the same as RXD line. Normal mode of operation.

1: The data field transmitted on TXD line is inverted (voltage polarity only) compared to the value written on US_THR register or the content read in US_RHR is inverted compared to what is received on RXD line (or ISO7816 IO line). Inverted Mode of operation, useful for contactless card application. To be used with configuration bit MSBF.

- **VAR_SYNC: Variable Synchronization of Command/Data Sync Start Frame Delimiter**

0: User defined configuration of command or data sync field depending on MODSYNC value.

1: The sync field is updated when a character is written into US_THR register.

- **MAX_ITERATION: Maximum Number of Automatic Iteration**

0 - 7: Defines the maximum number of iterations in mode ISO7816, protocol T= 0.

- **FILTER: Infrared Receive Line Filter**

0: The USART does not filter the receive line.

1: The USART filters the receive line using a three-sample filter (1/16-bit clock) (2 over 3 majority).

- **MAN: Manchester Encoder/Decoder Enable**

0: Manchester Encoder/Decoder are disabled.

1: Manchester Encoder/Decoder are enabled.

- **MODSYNC: Manchester Synchronization Mode**

0:The Manchester Start bit is a 0 to 1 transition

1: The Manchester Start bit is a 1 to 0 transition.

- **ONEBIT: Start Frame Delimiter Selector**

0: Start Frame delimiter is COMMAND or DATA SYNC.

1: Start Frame delimiter is One Bit.

### 38.6.11 Waveform Operating Mode

Waveform operating mode is entered by setting the WAVE parameter in TC_CMR (Channel Mode Register).

In Waveform Operating Mode the TC channel generates 1 or 2 PWM signals with the same frequency and independently programmable duty cycles, or generates different types of one-shot or repetitive pulses.

In this mode, TIOA is configured as an output and TIOB is defined as an output if it is not used as an external event (EEVT parameter in TC_CMR).

Figure 38-7 shows the configuration of the TC channel when programmed in Waveform Operating Mode.

### 38.6.12 Waveform Selection

Depending on the WAVSEL parameter in TC_CMR (Channel Mode Register), the behavior of TC_CV varies.

With any selection, RA, RB and RC can all be used as compare registers.

RA Compare is used to control the TIOA output, RB Compare is used to control the TIOB output (if correctly configured) and RC Compare is used to control TIOA and/or TIOB outputs.

### 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.

### 38.7.18 TC QDEC Interrupt Disable Register

**Name:** TC_QIDR

**Address:** 0x400900CC (0), 0x400940CC (1), 0x400980CC (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 |
| – | – | – | – | – | QERR | DIRCHG | IDX |

- **IDX: InDeX**

0 = No effect.

1 = Disables the interrupt when a rising edge occurs on IDX input.

- **DIRCHG: DIRection CHanGe**

0 = No effect.

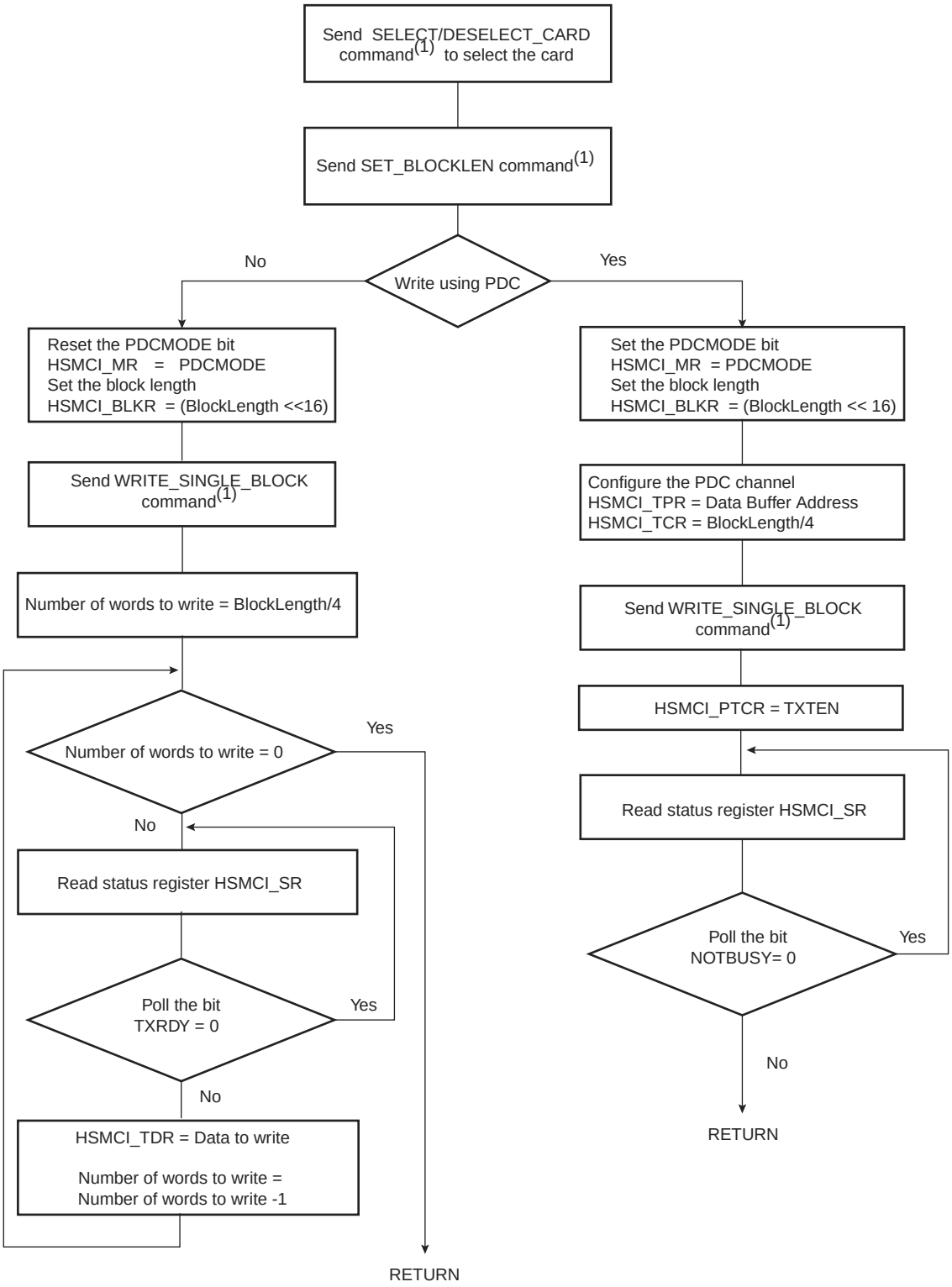
1 = Disables the interrupt when a change on rotation direction is detected.

- **QERR: Quadrature ERRor**

0 = No effect.

1 = Disables the interrupt when a quadrature error occurs on PHA, PHB.

**Figure 39-9. Write Functional Flow Diagram**




Note: 1. It is assumed that this command has been correctly sent (see Figure 39-7).

The following flowchart (Figure 39-10) shows how to manage a multiple write block transfer with the PDC. Polling or interrupt method can be used to wait for the end of write according to the contents of the Interrupt Mask Register (HSMCI_IMR).

## 39.13 Write Protection Registers

To prevent any single software error that may corrupt HSMCI behavior, the entire HSMCI address space from address offset 0x000 to 0x00FC can be write-protected by setting the WPEN bit in the “HSMCI Write Protect Mode Register” (HSMCI_WPMR).

If a write access to anywhere in the HSMCI address space from address offset 0x000 to 0x00FC is detected, then the WPVS flag in the HSMCI Write Protect Status Register (HSMCI_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 HSMCI Write Protect Mode Register (HSMCI_WPMR) with the appropriate access key, WPKEY.

The protected registers are:

- “HSMCI Mode Register” on page 974
- “HSMCI Data Timeout Register” on page 976
- “HSMCI SDCard/SDIO Register” on page 977
- “HSMCI Completion Signal Timeout Register” on page 982
- “HSMCI Configuration Register” on page 996

## 39.14 High Speed MultiMedia Card Interface (HSMCI) User Interface

**Table 39-8. Register Mapping**

| Offset | Register | Name | Access | Reset |
|---|---|---|---|---|
| 0x00 | Control Register | HSMCI_CR | Write | – |
| 0x04 | Mode Register | HSMCI_MR | Read-write | 0x0 |
| 0x08 | Data Timeout Register | HSMCI_DTOR | Read-write | 0x0 |
| 0x0C | SD/SDIO Card Register | HSMCI_SDCR | Read-write | 0x0 |
| 0x10 | Argument Register | HSMCI_ARGR | Read-write | 0x0 |
| 0x14 | Command Register | HSMCI_CMDR | Write | – |
| 0x18 | Block Register | HSMCI_BLKR | Read-write | 0x0 |
| 0x1C | Completion Signal Timeout Register | HSMCI_CSTOR | Read-write | 0x0 |
| 0x20 | Response Register(1) | HSMCI_RSPR | Read | 0x0 |
| 0x24 | Response Register(1) | HSMCI_RSPR | Read | 0x0 |
| 0x28 | Response Register(1) | HSMCI_RSPR | Read | 0x0 |
| 0x2C | Response Register(1) | HSMCI_RSPR | Read | 0x0 |
| 0x30 | Receive Data Register | HSMCI_RDR | Read | 0x0 |
| 0x34 | Transmit Data Register | HSMCI_TDR | Write | – |
| 0x38 - 0x3C | Reserved | – | – | – |
| 0x40 | Status Register | HSMCI_SR | Read | 0xC0E5 |
| 0x44 | Interrupt Enable Register | HSMCI_IER | Write | – |
| 0x48 | Interrupt Disable Register | HSMCI_IDR | Write | – |
| 0x4C | Interrupt Mask Register | HSMCI_IMR | Read | 0x0 |
| 0x50 | Reserved | – | – | – |
| 0x54 | Configuration Register | HSMCI_CFG | Read-write | 0x00 |
| 0x58-0xE0 | Reserved | – | – | – |
| 0xE4 | Write Protection Mode Register | HSMCI_WPMR | Read-write | – |
| 0xE8 | Write Protection Status Register | HSMCI_WPSR | Read-only | – |
| 0xEC - 0xFC | Reserved | – | – | – |
| 0x100-0x128 | Reserved for PDC registers | – | – | – |
| 0x12Cx1FC | Reserved | – | – | – |
| 0x200 | FIFO Memory Aperture0 | HSMCI_FIFO0 | Read-write | 0x0 |
| ... | ... | ... | ... | ... |
| 0x5FC | FIFO Memory Aperture255 | HSMCI_FIFO255 | Read-write | 0x0 |

Notes: 1. The Response Register can be read by N accesses at the same HSMCI_RSPR or at consecutive addresses (0x20 to 0x2C). N depends on the size of the response.

### 39.14.3 HSMCI Data Timeout Register

**Name:** HSMCI_DTOR

**Address:** 0x40080008

**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 |
| – | – | – | – | – | – | – | – |
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| – | DTOMUL | | | DTOCYC | | | |

This register can only be written if the WPEN bit is cleared in “HSMCI Write Protect Mode Register” on page 997.

- **DTOCYC: Data Timeout Cycle Number**

These fields determine the maximum number of Master Clock cycles that the HSMCI waits between two data block transfers. It equals (DTOCYC x Multiplier).

- **DTOMUL: Data Timeout Multiplier**

Multiplier is defined by DTOMUL as shown in the following table:

| Value | Name | Description |
|---|---|---|
| 0 | 1 | DTOCYC |
| 1 | 16 | DTOCYC x 16 |
| 2 | 128 | DTOCYC x 128 |
| 3 | 256 | DTOCYC x 256 |
| 4 | 1024 | DTOCYC x 1024 |
| 5 | 4096 | DTOCYC x 4096 |
| 6 | 65536 | DTOCYC x 65536 |
| 7 | 1048576 | DTOCYC x 1048576 |

If the data time-out set by DTOCYC and DTOMUL has been exceeded, the Data Time-out Error flag (DTOE) in the HSMCI Status Register (HSMCI_SR) rises.

### 39.14.4 HSMCI SDCard/SDIO Register

**Name:** HSMCI_SDCR

**Address:** 0x4008000C

**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 |
| – | – | – | – | – | – | – | – |
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| SDCBUS | | – | – | – | – | SDCSEL | |

This register can only be written if the WPEN bit is cleared in “HSMCI Write Protect Mode Register” on page 997.

- **SDCSEL: SDCard/SDIO Slot**

| Value | Name | Description |
|---|---|---|
| 0 | SLOTA | Slot A is selected. |
| 1 | SLOTB | – |
| 2 | SLOTC | – |
| 3 | SLOTD | – |

- **SDCBUS: SDCard/SDIO Bus Width**

| Value | Name | Description |
|---|---|---|
| 0 | 1 | 1 bit |
| 1 | – | Reserved |
| 2 | 4 | 4 bit |
| 3 | 8 | 8 bit |

0 = No UDP Resume Interrupt pending.

1 =UDP Resume Interrupt has been raised.

The USB device sets this bit when a UDP resume signal is detected at its port.

After reset, the state of this bit is undefined, the application must clear this bit by setting the RXRSM flag in the UDP_ICR register.

- **SOFINT: Start of Frame Interrupt Status**

0 = No Start of Frame Interrupt pending.

1 = Start of Frame Interrupt has been raised.

This interrupt is raised each time a SOF token has been detected. It can be used as a synchronization signal by using isochronous endpoints.

- **ENDBUSRES: End of BUS Reset Interrupt Status**

0 = No End of Bus Reset Interrupt pending.

1 = End of Bus Reset Interrupt has been raised.

This interrupt is raised at the end of a UDP reset sequence. The USB device must prepare to receive requests on the endpoint 0. The host starts the enumeration, then performs the configuration.

- **WAKEUP: UDP Resume Interrupt Status**

0 = No Wake-up Interrupt pending.

1 = A Wake-up Interrupt (USB Host Sent a RESUME or RESET) occurred since the last clear.

After reset the state of this bit is undefined, the application must clear this bit by setting the WAKEUP flag in the UDP_ICR register.

Moreover, it is possible to raise a flag only if there is predefined change in the temperature measure. The user can define a range of temperature or a threshold in the AFEC_TEMPCWR register and the mode of comparison that can be programmed into the AFEC_TEMPMR register by means of TEMPCMPMOD bitfield. These values define the way the TEMPCHG flag will be raised in the AFEC_IS1R register.

The TEMPCHG flag can be used to trigger an interrupt if there is something to update/modify in the system resulting from a temperature change.

In any case, if temperature sensor measure is configured, the temperature can be read at anytime in AFEC_CDR (AFEC_CSELR must be programmed accordingly prior to read AFEC_CDR) without any specific software intervention.

### 43.6.12 Enhanced Resolution Mode and Digital Averaging Function

The Enhanced Resolution Mode is enabled if RES field is set to 13-bits resolution or more in the AFEC Extended Mode Register (AFEC_EMR). FREERUN mode is not supported if Enhanced Resolution Mode is used.

There is no averaging on the temperature sensor channel if the temperature sensor measure is triggered on RTC event (see “Temperature Sensor” on page 1153).

In this mode the AFE Controller trades conversion performance for accuracy by averaging multiple samples, thus providing a digital low-pass filter function.

If 1-bit enhancement resolution is selected (RES=2 in the AFEC_EMR register), the AFEC effective sample rate is maximum AFEC sample rate divided by 4, therefore the oversampling ratio is 4.

When 2-bit enhancement resolution is selected (RES=3 in the AFEC_EMR register), the AFEC effective sample rate is maximum AFEC sample rate divided by 16 (oversampling ratio is 16).

When 3-bit enhancement resolution is selected (RES=4 in the AFEC_EMR register), the AFEC effective sample rate is maximum AFEC sample rate divided by 64 (oversampling ratio is 64).

When 4-bit enhancement resolution is selected (RES=5 in the AFEC_EMR register), the AFEC effective sample rate is maximum AFEC sample rate divided by 256 (oversampling ratio is 256).

The selected oversampling ratio applies to all enabled channels (except temperature sensor channel if triggered by RTC event).

The average result is valid into an internal register (read by means of the AFEC_CDR register) only if EOCn (n corresponding to the index of the channel) flag is set in AFEC_ISR and OVREn flag is cleared in the AFEC_OVER register. The average result is valid for all channels in the AFEC_LCDR register only if DRDY is set and GOVRE is cleared in the AFEC_ISR register.

The samples can be defined in different ways for the averaging function depending on the configuration of the STM bit in the Extended Mode register (AFEC_EMR) and USEQ bit in the Mode register (AFEC_MR).

When USEQ is cleared, there are 2 possible ways to generate the averaging through the trigger event. If the STM bit is cleared in the AFEC_EMR register, every trigger event generates one sample for each enabled channel as described in Figure 43-11, on page 1157. Therefore 4 trigger events are requested to get the result of averaging if RES=2.

### 45.7.2 DACC Mode Register

**Name:** DACC_MR

**Address:** 0x400B8004

**Access:** Read-write

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|
| – | – | STARTUP | | | | | |
| 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| – | CLKDIV | MAXS | TAG | – | – | USER_SEL | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| REFRESH | | | | | | | |
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| – | FASTWKUP | SLEEP | WORD | TRGSEL | | | TRGEN |

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

- **TRGEN: Trigger Enable**

| Value | Name | Description |
|---|---|---|
| 0 | DIS | External trigger mode disabled. DACC in free running mode. |
| 1 | EN | External trigger mode enabled. |

- **TRGSEL: Trigger Selection**

| TRGSEL | | | Selected TRGSEL |
|---|---|---|---|
| 0 | 0 | 0 | External trigger |
| 0 | 0 | 1 | TIO Output of the Timer Counter Channel 0 |
| 0 | 1 | 0 | TIO Output of the Timer Counter Channel 1 |
| 0 | 1 | 1 | TIO Output of the Timer Counter Channel 2 |
| 1 | 0 | 0 | PWM Event Line 0 |
| 1 | 0 | 1 | PWM Event Line 1 |
| 1 | 1 | 0 | Reserved |
| 1 | 1 | 1 | Reserved |

- **WORD: Word Transfer**

| Value | Name | Description |
|---|---|---|
| 0 | HALF | Half-Word transfer |
| 1 | WORD | Word Transfer |

- **SLEEP: Sleep Mode**

| SLEEP | Selected Mode |
|---|---|
| 0 | Normal Mode |
| 1 | Sleep Mode |