**Welcome to E-XFL.COM**

## What is "Embedded - Microcontrollers"?

"Embedded - Microcontrollers" refer to small, integrated circuits designed to perform specific tasks within larger systems. These microcontrollers are essentially compact computers on a single chip, containing a processor core, memory, and programmable input/output peripherals. They are called "embedded" because they are embedded within electronic devices to control various functions, rather than serving as standalone computers. Microcontrollers are crucial in modern electronics, providing the intelligence and control needed for a wide range of applications.

## Applications of "Embedded - Microcontrollers"

| Details | |
|---|---|
| Product Status | Active |
| Core Processor | ARM® Cortex®-M7 |
| Core Size | 32-Bit Single-Core |
| Speed | 300MHz |
| Connectivity | I²C, IrDA, LINbus, SPI, SSC, UART/USART |
| Peripherals | Brown-out Detect/Reset, DMA, I²S, POR, PWM, WDT |
| Number of I/O | 44 |
| Program Memory Size | 1MB (1M x 8) |
| Program Memory Type | FLASH |
| EEPROM Size | - |
| RAM Size | 384K x 8 |
| Voltage - Supply (Vcc/Vdd) | 1.7V ~ 3.6V |
| Data Converters | A/D 5x12b; D/A 1x12b |
| Oscillator Type | Internal |
| Operating Temperature | -40°C ~ 105°C (TA) |
| Mounting Type | Surface Mount |
| Package / Case | 64-LQFP |
| Supplier Device Package | 64-LQFP (10x10) |
| Purchase URL | https://www.e-xfl.com/product-detail/microchip-technology/atsams70j20b-an |

| Value | Name | Description |
|---|---|---|
| 4 | 16BEAT_BURST | 16-beat Burst—The undefined length burst or bursts sequence is split into 16-beat bursts or less, allowing re-arbitration every 16 beats. |
| 5 | 32BEAT_BURST | 32-beat Burst —The undefined length burst or bursts sequence is split into 32-beat bursts or less, allowing re-arbitration every 32 beats. |
| 6 | 64BEAT_BURST | 64-beat Burst—The undefined length burst or bursts sequence is split into 64-beat bursts or less, allowing re-arbitration every 64 beats. |
| 7 | 128BEAT_BURST | 128-beat Burst—The undefined length burst or bursts sequence is split into 128-beat bursts or less, allowing re-arbitration every 128 beats.<br>Note: Unless duly needed, the ULBT should be left at its default 0 value for power saving. |

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | 31:24 | | | | | | | | |
| 0x2C | RTC_VER | 7:0 | | | | | NVCALALR | NVTIMALR | NVCAL | NVTIM |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |

**Figure 35-33. Clock Rate Transition Occurs while the SMC is Performing a Write Operation**

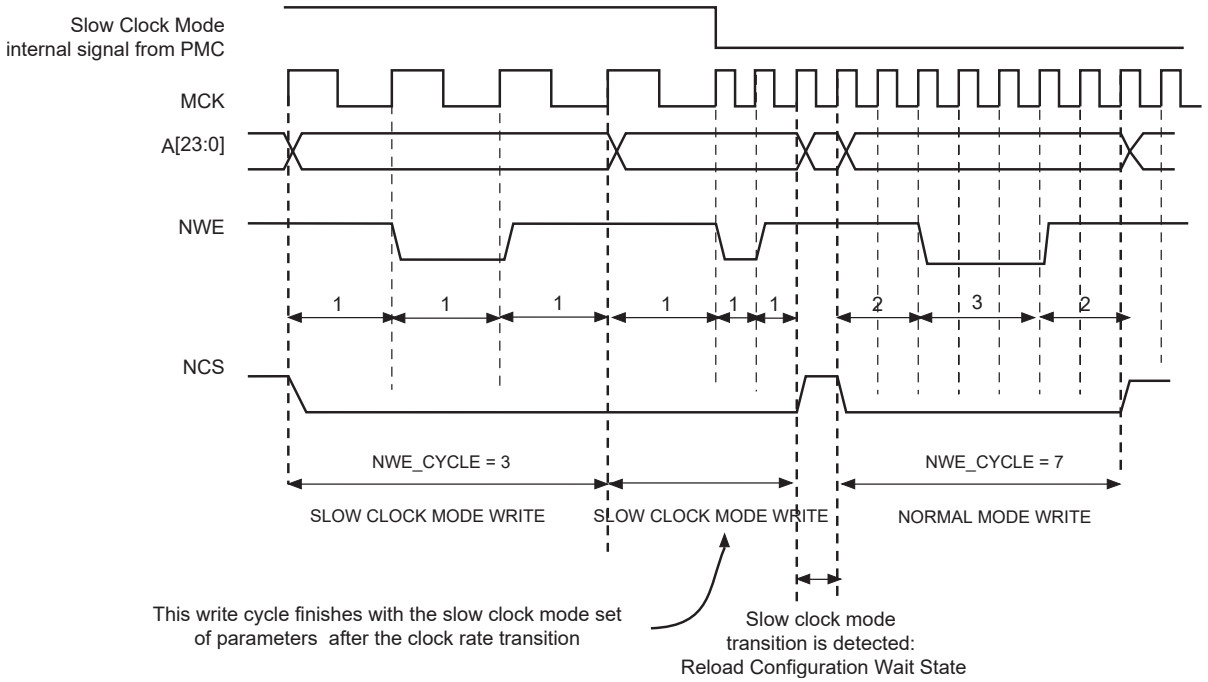
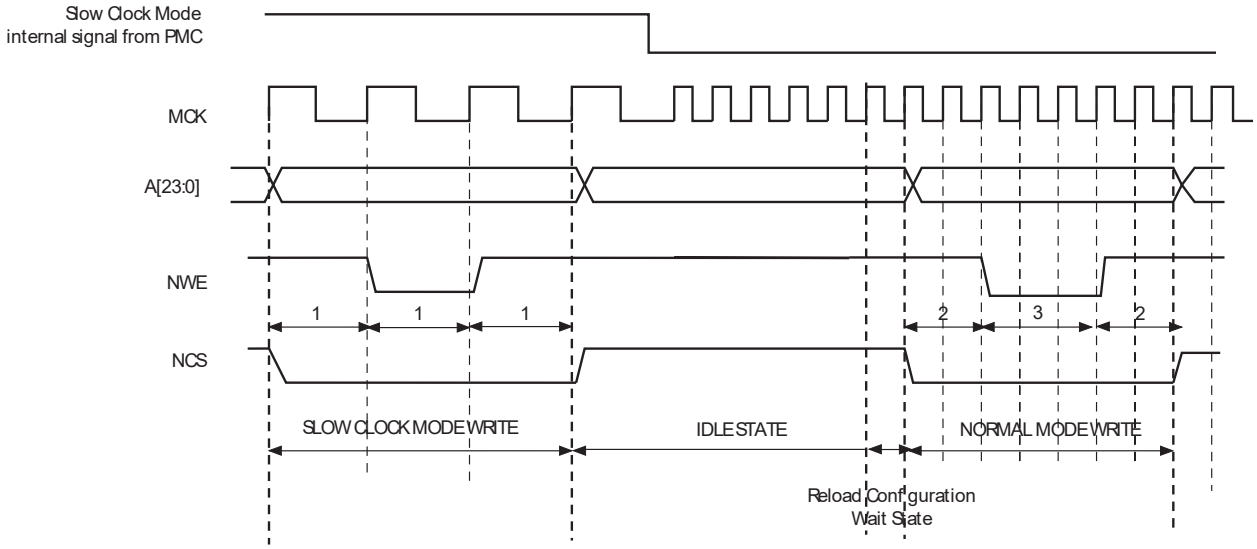



**Figure 35-34. Recommended Procedure to Switch from Slow Clock Mode to Normal Mode or from Normal Mode to Slow Clock Mode**




## 35.15 Asynchronous Page Mode

The SMC supports asynchronous burst reads in Page mode, provided that the Page mode is enabled (SMC_MODE.PMEN =1). The page size must be configured in the SMC_MODE register (PS field) to 4, 8, 16 or 32 bytes.

The page defines a set of consecutive bytes into memory. A 4-byte page (resp. 8-, 16-, 32-byte page) is always aligned to 4-byte boundaries (resp. 8-, 16-, 32-byte boundaries) of memory. The MSB of data address defines the address of the page in memory, the LSB of address define the address of the data in the page as detailed in the following table.

### 35.16.1.9 SMC Write Protection Status Register

**Name:** SMC_WPSR
**Offset:** 0xE8
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | WPVSRC[15:8] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | WPVSRC[7:0] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | WPVS |
| Access | | | | | | | | |
| Reset | | | | | | | | 0 |

**Bits 23:8 – WPVSRC[15:0]** Write Protection Violation Source
When WPVS = 1, WPVSRC indicates the register address offset at which a write access has been attempted.

**Bit 0 – WPVS** Write Protection Violation Status

| Value | Description |
|---|---|
| 0 | No write protection violation has occurred since the last read of the SMC_WPSR register. |
| 1 | A write protection violation has occurred since the last read of the SMC_WPSR register. If this violation is an unauthorized attempt to write a protected register, the associated violation is reported into field WPVSRC. |

### 36.9.22 XDMAC Channel x Source Address Register [x = 0..23]

**Name:** XDMAC_CSA
**Offset:** 0x60 + n*0x40 [n=0..23]
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | SA[31:24] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | SA[23:16] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | SA[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

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

**Bits 31:0 – SA[31:0]** Channel x Source Address
Program this register with the source address of the DMA transfer.

A configuration error is generated when this address is not aligned with the transfer data size.

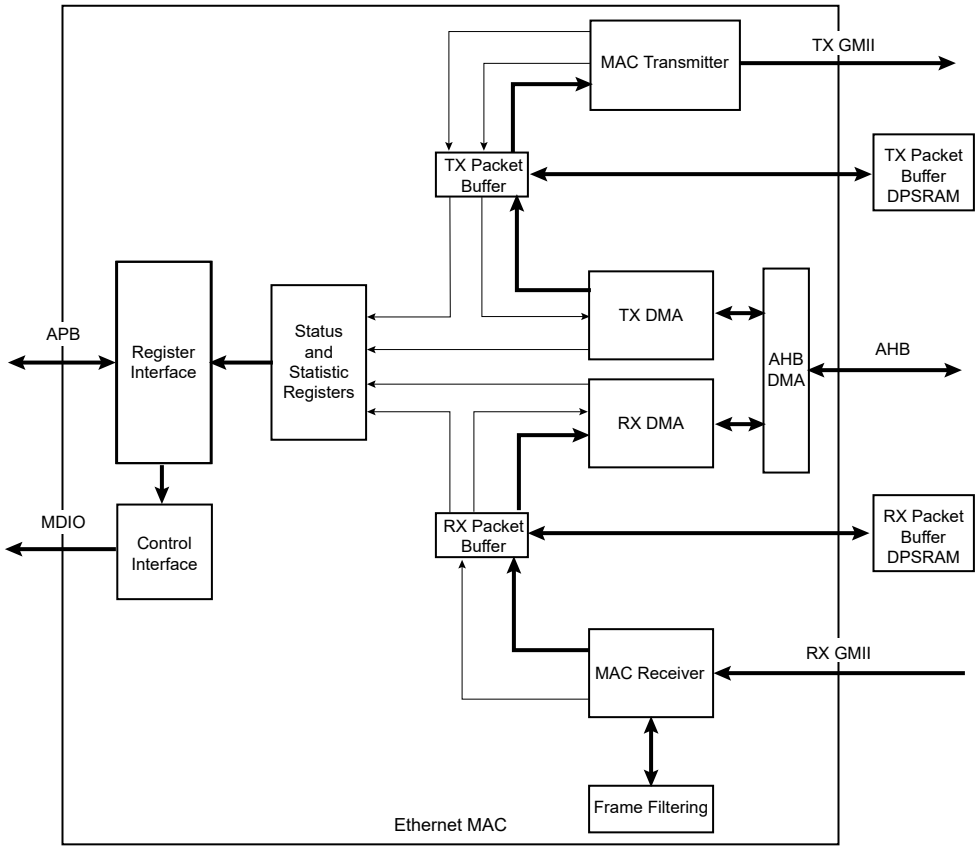
As described above, the DMA can be programmed into a low latency mode, known as Partial Store and Forward. For further details of this mode, see the related Links.

When the DMA is in full store and forward mode, full packets are buffered which provides the possibility to:

- Discard packets with error on the receive path before they are partially written out of the DMA, thus saving AHB bus bandwidth and driver processing overhead,
- Retry collided transmit frames from the buffer, thus saving AHB bus bandwidth,
- Implement transmit IP/TCP/UDP checksum generation offload.

With the packet buffers included, the structure of the GMAC data paths is shown in this image:

**Figure 38-2. Data Paths with Packet Buffers Included**




### 38.6.3.7 Transmit Packet Buffer

The transmitter packet buffer will continue attempting to fetch frame data from the AHB system memory until the packet buffer itself is full, at which point it will attempt to maintain its full level.

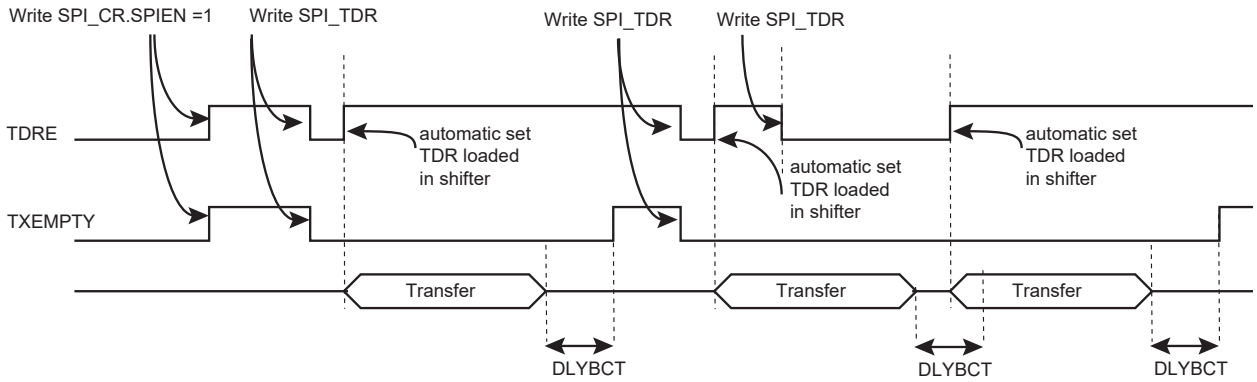
To accommodate the status and statistics associated with each frame, three words per packet (or two if the GMAC is configured in 64-bit data path mode) are reserved at the end of the packet data. If the packet is bad and requires to be dropped, the status and statistics are the only information held on that packet. Storing the status in the DPRAM is required in order to decouple the DMA interface of the buffer from the MAC interface, to update the MAC status/statistics and to generate interrupts in the order in which the packets that they represent were fetched from the AHB memory.

If any errors occur on the AHB while reading the transmit frame, the fetching of packet data from AHB memory is halted. The MAC transmitter will continue to fetch packet data, thereby emptying the packet

The end of transfer is indicated by the TXEMPTY flag in SPI_SR. If a transfer delay (DLYBCT) is greater than 0 for the last transfer, TXEMPTY is set after the completion of this delay. The peripheral clock can be switched off at this time.

**Note:** When the SPI is enabled, the TDRE and TXEMPTY flags are set.

**Figure 41-5. TDRE and TXEMPTY Flag Behavior**




The transfer of received data from the internal shift register to SPI_RDR is indicated by the Receive Data Register Full (RDRF) bit in SPI_SR. When the received data is read, SPI_SR.RDRF is cleared.

If SPI_RDR has not been read before new data is received, the Overrun Error (OVRES) flag in SPI_SR is set. As long as this flag is set, data is loaded in SPI_RDR. The user has to read SPI_SR to clear OVRES.

The following figures show, respectively, a block diagram of the SPI when operating in Master mode and a flow chart describing how transfers are handled.

### 42.7.6 QSPI Interrupt Enable Register

**Name:** QSPI_IER
**Offset:** 0x14
**Reset:** –
**Property:** Write-only

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

0: No effect.

1: Enables the corresponding interrupt.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | INSTRE | CSS | CSR |
| Access | | | | | | W | W | W |
| Reset | | | | | | – | – | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | OVRES | TXEMPTY | TDRE | RDRF |
| Access | | | | | W | W | W | W |
| Reset | | | | | – | – | – | – |

**Bit 10 – INSTRE** Instruction End Interrupt Enable

**Bit 9 – CSS** Chip Select Status Interrupt Enable

**Bit 8 – CSR** Chip Select Rise Interrupt Enable

**Bit 3 – OVRES** Overrun Error Interrupt Enable

**Bit 2 – TXEMPTY** Transmission Registers Empty Enable

**Bit 1 – TDRE** Transmit Data Register Empty Interrupt Enable

**Bit 0 – RDRF** Receive Data Register Full Interrupt Enable

| Value | Name | Description |
|---|---|---|
| 2 | TRSFR_WRITE | Write transfer into the serial memory.<br>Scrambling is not performed. |
| 3 | TRSFR_WRITE_MEMORY | Write data transfer into the serial memory.<br>If enabled, scrambling is performed. |

**Bit 10 – ADDRL** Address Length
The ADDRL bit determines the length of the address.

0 (24_BIT): The address is 24 bits long.

1 (32_BIT): The address is 32 bits long.

**Bits 9:8 – OPTL[1:0]** Option Code Length
The OPTL field determines the length of the option code. The value written in OPTL must be consistent with the value written in the field WIDTH. For example, OPTL = 0 (1-bit option code) is not consistent with WIDTH = 6 (option code sent with QuadSPI protocol, thus the minimum length of the option code is 4 bits).

| Value | Name | Description |
|---|---|---|
| 0 | OPTION_1BIT | The option code is 1 bit long. |
| 1 | OPTION_2BIT | The option code is 2 bits long. |
| 2 | OPTION_4BIT | The option code is 4 bits long. |
| 3 | OPTION_8BIT | The option code is 8 bits long. |

**Bit 7 – DATAEN** Data Enable

| Value | Description |
|---|---|
| 0 | No data is sent/received to/from the serial Flash memory. |
| 1 | Data is sent/received to/from the serial Flash memory. |

**Bit 6 – OPTEN** Option Enable

| Value | Description |
|---|---|
| 0 | The option is not sent to the serial Flash memory. |
| 1 | The option is sent to the serial Flash memory. |

**Bit 5 – ADDREN** Address Enable

| Value | Description |
|---|---|
| 0 | The transfer address is not sent to the serial Flash memory. |
| 1 | The transfer address is sent to the serial Flash memory. |

**Bit 4 – INSTEN** Instruction Enable

| Value | Description |
|---|---|
| 0 | The instruction is not sent to the serial Flash memory. |
| 1 | The instruction is sent to the serial Flash memory. |

**Bits 2:0 – WIDTH[2:0]** Width of Instruction Code, Address, Option Code and Data

### 43.7.6 TWIHS Status Register

**Name:** TWIHS_SR
**Offset:** 0x20
**Reset:** 0x03000009
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | SDA | SCL |
| Access | | | | | | | R | R |
| Reset | | | | | | | 1 | 1 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | SMBHHM | SMBDAM | PECERR | TOUT | | MCACK |
| Access | | | R | R | R | R | | R |
| Reset | | | 0 | 0 | 0 | 0 | | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | EOSACC | SCLWS | ARBLST | NACK |
| Access | | | | | R | R | R | R |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | UNRE | OVRE | GACC | SVACC | SVREAD | TXRDY | RXRDY | TXCOMP |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 1 |

**Bit 25 – SDA** SDA Line Value

| Value | Description |
|---|---|
| 0 | SDA line sampled value is ‘0’. |
| 1 | SDA line sampled value is ‘1’. |

**Bit 24 – SCL** SCL Line Value

| Value | Description |
|---|---|
| 0 | SCL line sampled value is ‘0’. |
| 1 | SCL line sampled value is ‘1.’ |

**Bit 21 – SMBHHM** SMBus Host Header Address Match (cleared on read)

| Value | Description |
|---|---|
| 0 | No SMBus Host Header Address received since the last read of TWIHS_SR. |
| 1 | An SMBus Host Header Address was received since the last read of TWIHS_SR. |

**Bit 20 – SMBDAM** SMBus Default Address Match (cleared on read)

| Value | Description |
|---|---|
| 0 | No SMBus Default Address received since the last read of TWIHS_SR. |
| 1 | An SMBus Default Address was received since the last read of TWIHS_SR. |

### 44.9.1 SSC Control Register

**Name:** SSC_CR
**Offset:** 0x0
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | SWRST | | | | | | TXDIS | TXEN |
| Access | W | | | | | | W | W |
| Reset | – | | | | | | – | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | RXDIS | RXEN |
| Access | | | | | | | W | W |
| Reset | | | | | | | – | – |

**Bit 15 – SWRST** Software Reset

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Performs a software reset. Has priority on any other bit in SSC_CR. |

**Bit 9 – TXDIS** Transmit Disable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Disables Transmit. If a character is currently being transmitted, disables at end of current character transmission. |

**Bit 8 – TXEN** Transmit Enable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Enables Transmit if TXDIS is not set. |

**Bit 1 – RXDIS** Receive Disable

### 45.8.4 I2SC Status Clear Register

**Name:** I2SC_SCR
**Offset:** 0x0C
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | TXURCH[1:0] | | | | | |
| Access | | | W | W | | | | |
| Reset | | | – | – | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | RXORCH[1:0] | |
| Access | | | | | | | W | W |
| Reset | | | | | | | – | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | TXUR | | | | RXOR | | |
| Access | | W | | | | W | | |
| Reset | | – | | | | – | | |

**Bits 21:20 – TXURCH[1:0]** Transmit Underrun Per Channel Status Clear
Writing a ’0’ has no effect.

Writing a ’1’ to any bit in this field clears the corresponding bit in the I2SC_SR and the corresponding interrupt request.

**Bits 9:8 – RXORCH[1:0]** Receive Overrun Per Channel Status Clear
Writing a ’0’ has no effect.

Writing a ’1’ to any bit in this field clears the corresponding bit in the I2SC_SR and the corresponding interrupt request.

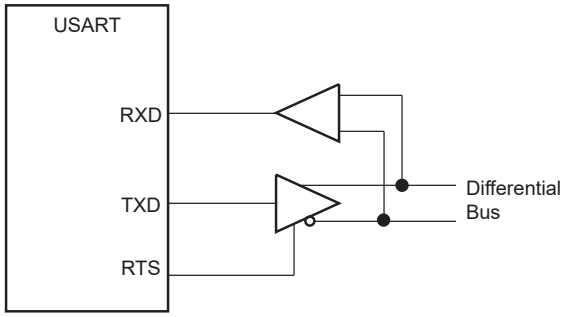
**Bit 6 – TXUR** Transmit Underrun Status Clear
Writing a ’0’ to this bit has no effect.

Writing a ’1’ to this bit clears the status bit.

**Bit 2 – RXOR** Receive Overrun Status Clear
Writing a ’0’ to this bit has no effect.

Writing a ’1’ to this bit clears the status bit.

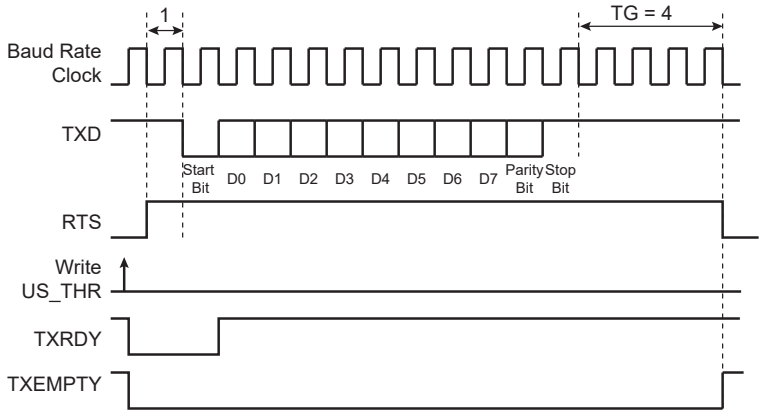
**Figure 46-35. Typical Connection to a RS485 Bus**




RS485 mode is enabled by writing the value 0x1 to the US_MR.USART_MODE.

The RTS pin is at a level inverse to the TXEMPTY bit. Significantly, the RTS pin remains high when a timeguard is programmed so that the line can remain driven after the last character completion. Figure 46-36 gives an example of the RTS waveform during a character transmission when the timeguard is enabled.

**Figure 46-36. Example of RTS Drive with Timeguard**




### 46.6.7 Modem Mode

The USART features the Modem mode, which enables control of the signals DTR (Data Terminal Ready), DSR (Data Set Ready), RTS (Request to Send), CTS (Clear to Send), DCD (Data Carrier Detect), and RI (Ring Indicator). While operating in Modem mode, the USART behaves as a DTE (Data Terminal Equipment) as it drives DTR and RTS and can detect level change on DSR, DCD, CTS, and RI.

Modem mode is enabled by writing the value 0x3 to US_MR.USART_MODE. While operating in Modem mode, the USART behaves as though in Asynchronous mode and all the parameter configurations are available.

The following table provides the correspondence of the USART signals with modem connection standards.

**Table 46-11. Circuit References**

| USART Pin | V24 | CCITT | Direction |
|---|---|---|---|
| TXD | 2 | 103 | From terminal to modem |
| RTS | 4 | 105 | From terminal to modem |
| DTR | 20 | 108.2 | From terminal to modem |

**Table 46-12. SPI Bus Protocol Mode**

| SPI Bus Protocol Mode | CPOL | CPHA |
|---|---|---|
| 0 | 0 | 1 |
| 1 | 0 | 0 |
| 2 | 1 | 1 |
| 3 | 1 | 0 |

**Figure 46-37. SPI Transfer Format (CPHA = 1, 8 bits per transfer)**

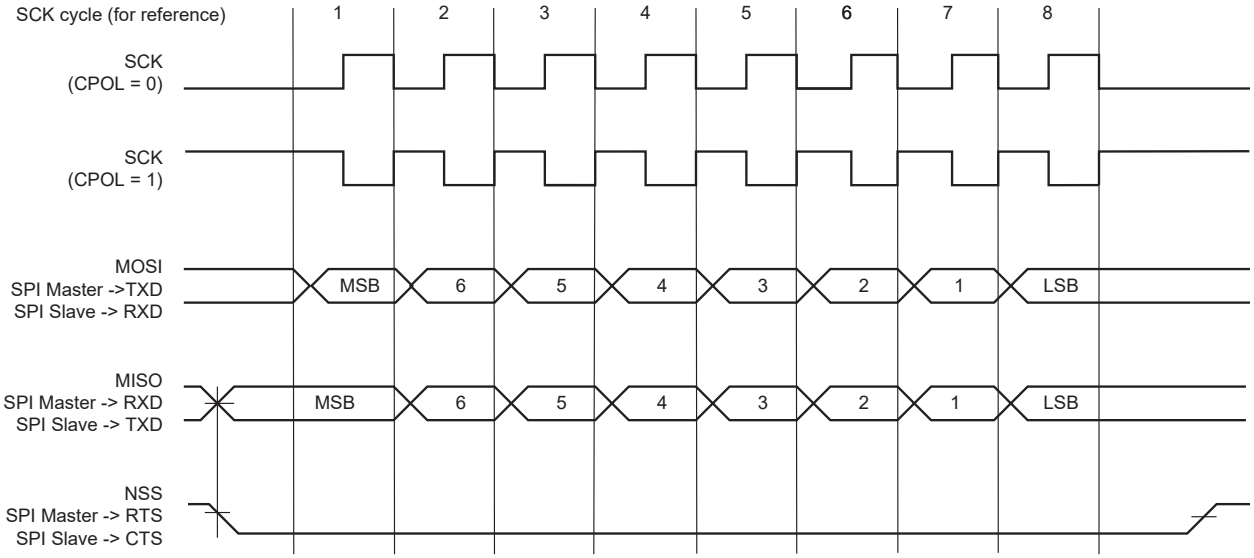
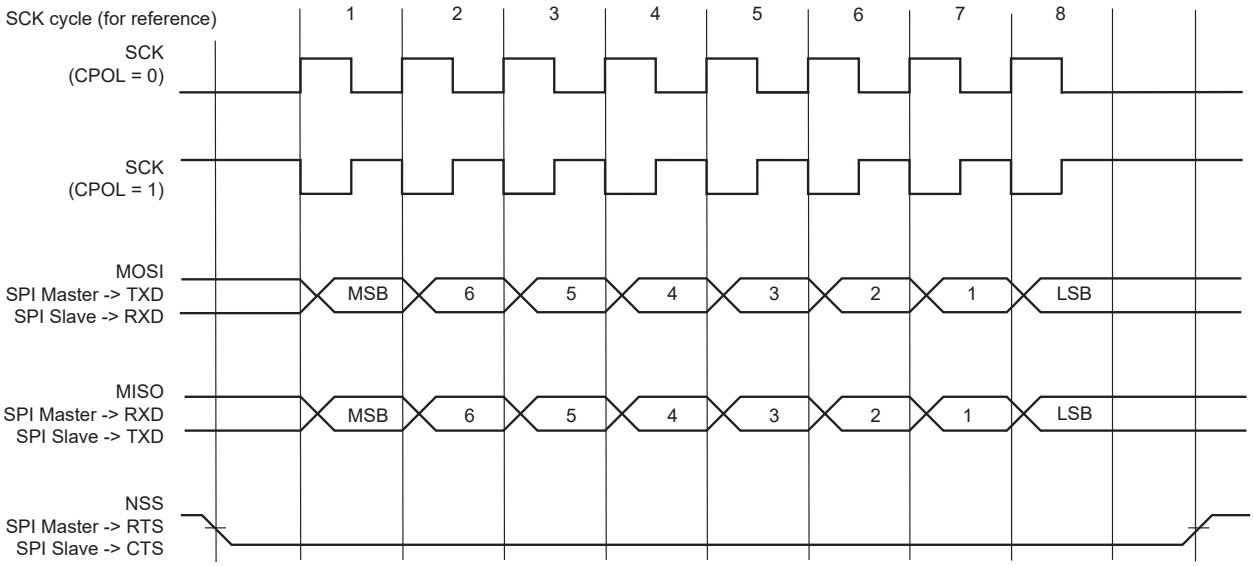



**Figure 46-38. SPI Transfer Format (CPHA = 0, 8 bits per transfer)**




#### 46.6.8.4 Receiver and Transmitter Control

See “Receiver and Transmitter Control”.

### 49.5.6 FIFO Acknowledge Handling

The Get Indices of Rx FIFO 0, Rx FIFO 1, and the Tx Event FIFO are controlled by writing to the corresponding FIFO Acknowledge Index in the registers MCAN_RXF0A, MCAN_RXF1A and MCAN_TXEFA. Writing to the FIFO Acknowledge Index will set the FIFO Get Index to the FIFO Acknowledge Index plus one and thereby updates the FIFO Fill Level. There are two use cases:

When only a single element has been read from the FIFO (the one being pointed to by the Get Index), this Get Index value is written to the FIFO Acknowledge Index.

When a sequence of elements has been read from the FIFO, it is sufficient to write the FIFO Acknowledge Index only once at the end of that read sequence (value: Index of the last element read), to update the FIFO's Get Index.

Due to the fact that the processor has free access to the MCAN's Message RAM, special care has to be taken when reading FIFO elements in an arbitrary order (Get Index not considered). This might be useful when reading a High Priority Message from one of the two Rx FIFOs. In this case the FIFO's Acknowledge Index should not be written because this would set the Get Index to a wrong position and also alters the FIFO's Fill Level. In this case some of the older FIFO elements would be lost.

**Note:** The application has to ensure that a valid value is written to the FIFO Acknowledge Index. The MCAN does not check for erroneous values.

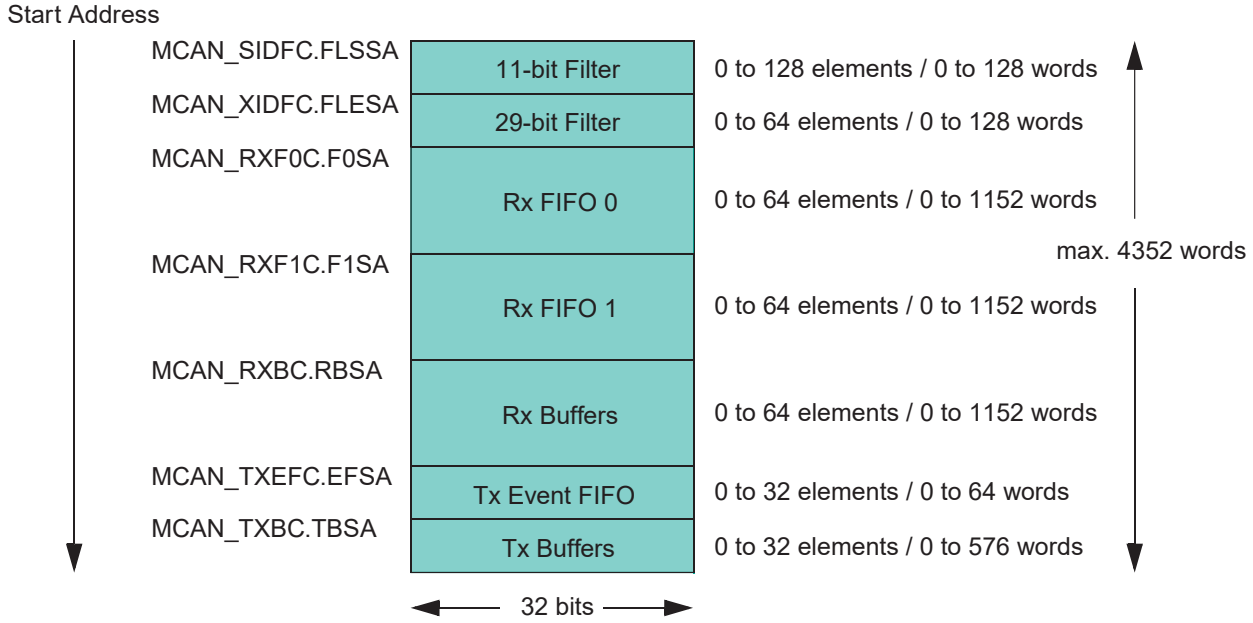
### 49.5.7 Message RAM

#### 49.5.7.1 Message RAM Configuration

The Message RAM has a width of 32 bits. The MCAN module can be configured to allocate up to 4352 words in the Message RAM. It is not necessary to configure each of the sections listed in the figure below, nor is there any restriction with respect to the sequence of the sections.

When operated in CAN FD mode, the required Message RAM size depends on the element size configured for Rx FIFO0, Rx FIFO1, Rx Buffers, and Tx Buffers via MCAN_RXESC.F0DS, MCAN_RXESC.F1DS, MCAN_RXESC.RBDS, and MCAN_TXESC.TBDS.

**Figure 49-12. Message RAM Configuration**



- Read of the Capture Registers by the DMAC
- Compare Event Fault Generation for PWM
- Register Write Protection

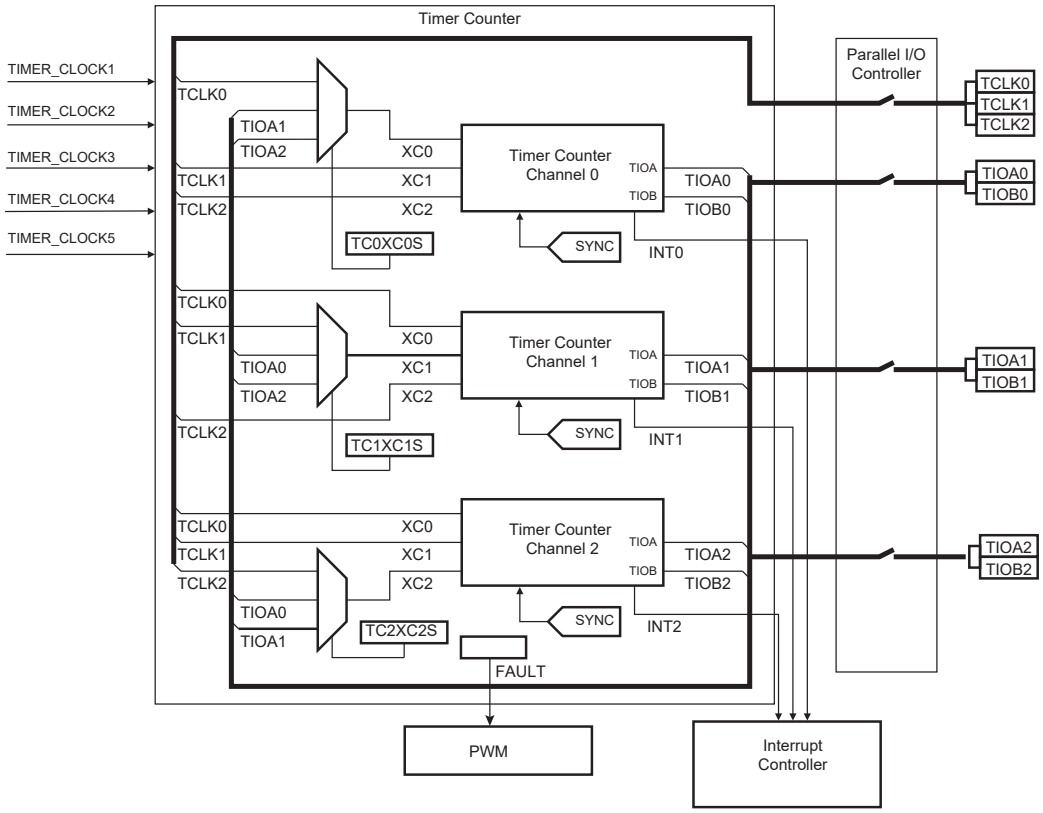
## 50.3 Block Diagram

**Table 50-1. Timer Counter Clock Assignment**

| Name | Definition |
|---|---|
| TIMER_CLOCK1 | PCK6 or PCK7 (TC0 only) |
| TIMER_CLOCK2 | MCK/8 |
| TIMER_CLOCK3 | MCK/32 |
| TIMER_CLOCK4 | MCK/128 |
| TIMER_CLOCK5 (1) | SLCK |

1. When SLCK is selected for Peripheral Clock (CSS = 0 in PMC Master Clock register), SLCK input is equivalent to Peripheral Clock.
2. The PCK6 or PCK7 (TC0 only) frequency must be at least three times lower than peripheral clock frequency.

**Figure 50-1. Timer Counter Block Diagram**




**Note:**
The QDEC connections are detailed in Predefined Connection of the Quadrature Decoder with Timer Counters.

### 50.7.20 TC QDEC Interrupt Status Register

**Name:** TC_QISR
**Offset:** 0xD4
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

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

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | MPE | QERR | DIRCHG | IDX |
| Access | | | | | R | R | R | R |
| Reset | | | | | 0 | 0 | 0 | 0 |

**Bit 8 – DIR** Direction
Returns an image of the current rotation direction.

**Bit 3 – MPE** Consecutive Missing Pulse Error

| Value | Description |
|---|---|
| 0 | The number of consecutive missing pulses has not reached the maximum value specified in MAXCMP since the last read of TC_QISR. |
| 1 | An occurrence of MAXCMP consecutive missing pulses has been detected since the last read of TC_QISR. |

**Bit 2 – QERR** Quadrature Error

| Value | Description |
|---|---|
| 0 | No quadrature error since the last read of TC_QISR. |
| 1 | A quadrature error occurred since the last read of TC_QISR. |

**Bit 1 – DIRCHG** Direction Change

| Value | Description |
|---|---|
| 0 | No change on rotation direction since the last read of TC_QISR. |
| 1 | The rotation direction changed since the last read of TC_QISR. |

### 51.7.42 PWM Channel Duty Cycle Update Register

**Name:** PWM_CDTYUPDx
**Offset:** 0x0208 + x*0x20 [x=0..3]
**Reset:** –
**Property:** Write-only

This register acts as a double buffer for the CDTY value. This prevents an unexpected waveform when modifying the waveform duty-cycle.

Only the first 16 bits (channel counter size) are significant.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | CDTYUPD[23:16] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | CDTYUPD[15:8] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CDTYUPD[7:0] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | – |

**Bits 23:0 – CDTYUPD[23:0]** Channel Duty-Cycle Update
Defines the waveform duty-cycle. This value must be defined between 0 and CPRD (PWM_CPRDx).

**Table 58-59. SMC Read Signals - NCS Controlled (READ_MODE = 0)**

| Symbol | VDDIO Supply | 1.8V Domain | 3.3V Domain | 1.8V Domain | 3.3V Domain | Unit |
|---|---|---|---|---|---|---|
| | Parameter | Min | | Max | | |
| NO HOLD Settings (NCS_RD_HOLD = 0) | | | | | | |
| $SMC_8$ | Data Setup before NCS High | 24.9 | 21.4 | – | – | ns |
| $SMC_9$ | Data Hold after NCS High | 0 | 0 | – | – | ns |
| HOLD Settings (NCS_RD_HOLD ≠ 0) | | | | | | |
| $SMC_{10}$ | Data Setup before NCS High | 13.4 | 11.7 | – | – | ns |
| $SMC_{11}$ | Data Hold after NCS High | 0 | 0 | – | – | ns |
| HOLD or NO HOLD Settings (NCS_RD_HOLD ≠ 0, NCS_RD_HOLD = 0) | | | | | | |
| $SMC_{12}$ | A0–A22 valid before NCS High | (NCS_RD_SETUP + NCS_RD_PULSE) × $t_{CPMCK}$ - 4.0 | (NCS_RD_SETUP + NCS_RD_PULSE) × $t_{CPMCK}$ - 3.9 | – | – | ns |
| $SMC_{13}$ | NRD low before NCS High | (NCS_RD_SETUP + NCS_RD_PULSE - NRD_SETUP) × $t_{CPMCK}$ - 2.8 | (NCS_RD_SETUP + NCS_RD_PULSE - NRD_SETUP) × $t_{CPMCK}$ - 4.2 | – | – | ns |
| $SMC_{14}$ | NCS Pulse Width | NCS_RD_PULSE length × $t_{CPMCK}$ - 0.9 | NCS_RD_PULSE length × $t_{CPMCK}$ - 0.2 | – | – | ns |

#### 58.13.1.9.2 Read Timings

**Table 58-60. SMC Read Signals - NRD Controlled (READ_MODE = 1)**

| Symbol | | VDDIO Supply | | Unit |
|---|---|---|---|---|
| | Parameter | Min | Max | |
| NO HOLD Settings (NRD_HOLD = 0) | | | | |
| $SMC_1$ | Data Setup before NRD High | 14.3 | – | ns |
| $SMC_2$ | Data Hold after NRD High | 0 | – | ns |
| HOLD Settings (NRD_HOLD ≠ 0) | | | | |
| $SMC_3$ | Data Setup before NRD High | 12.1 | – | ns |
| $SMC_4$ | Data Hold after NRD High | 0 | – | ns |
| HOLD or NO HOLD Settings (NRD_HOLD ≠ 0, NRD_HOLD = 0) | | | | |

| Symbol | Parameter | Condition | Min | Max | Unit |
|---|---|---|---|---|---|
| $C_i$(1) | Capacitance for each I/O Pin | – | – | 10 | pF |
| $f_{TWCK}$ | TWCK Clock Frequency | – | 0 | 400 | kHz |
| $R_P$ | Value of Pull-up resistor | $f_{TWCK} \le 100$ kHz | ($V_{DDIO}$ - 0.4V) ÷ 3mA | 1000ns ÷ $C_b$ | Ω |
| | | $f_{TWCK} > 100$ kHz | | 300ns ÷ $C_b$ | Ω |
| $t_{LOW}$ | Low Period of the TWCK clock | $f_{TWCK} \le 100$ kHz | (3) | – | µs |
| | | $f_{TWCK} > 100$ kHz | (3) | – | µs |
| $t_{HIGH}$ | High period of the TWCK clock | $f_{TWCK} \le 100$ kHz | (4) | – | µs |
| | | $f_{TWCK} > 100$ kHz | (4) | – | µs |
| $t_{HD;STA}$ | Hold Time (repeated) START Condition | $f_{TWCK} \le 100$ kHz | $t_{HIGH}$ | – | µs |
| | | $f_{TWCK} > 100$ kHz | $t_{HIGH}$ | – | µs |
| $t_{SU;STA}$ | Set-up time for a repeated START condition | $f_{TWCK} \le 100$ kHz | $t_{HIGH}$ | – | µs |
| | | $f_{TWCK} > 100$ kHz | $t_{HIGH}$ | – | µs |
| $t_{HD;DAT}$ | Data hold time | $f_{TWCK} \le 100$ kHz | 0 | 3 × $t_{CPMCK}$(5) | µs |
| | | $f_{TWCK} > 100$ kHz | 0 | 3 ×$t_{CPMCK}$(5) | µs |
| $t_{SU;DAT}$ | Data setup time | $f_{TWCK} \le 100$ kHz | $t_{LOW}$ - 3 × $t_{CPMCK}$(5) | – | ns |
| | | $f_{TWCK} > 100$ kHz | $t_{LOW}$ - 3 × $t_{CPMCK}$(5) | – | ns |
| $t_{SU;STO}$ | Setup time for STOP condition | $f_{TWCK} \le 100$ kHz | $t_{HIGH}$ | – | µs |
| | | $f_{TWCK} > 100$ kHz | $t_{HIGH}$ | – | µs |
| $t_{HD;STA}$ | Hold Time (repeated) START Condition | $f_{TWCK} \le 100$ kHz | $t_{HIGH}$ | – | µs |
| | | $f_{TWCK} > 100$ kHz | $t_{HIGH}$ | – | µs |

**Note:**

1. Required only for $f_{TWCK} > 100$ kHz.
2. $C_b$ = capacitance of one bus line in pF. Per $I^2C$ standard, $C_b$ max = 400pF.
3. The TWCK low period is defined as follows: $t_{LOW} = ((CLDIV \times 2^{CKDIV}) + 4) \times t_{MCK}$.
4. The TWCK high period is defined as follows: $t_{HIGH} = ((CHDIV \times 2^{CKDIV}) + 4) \times t_{MCK}$.
5. $t_{CPMCK}$ = MCK bus period