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	Obsolete
Core Processor	AVR
Core Size	32-Bit Single-Core
Speed	50MHz
Connectivity	I²C, SPI, UART/USART
Peripherals	Brown-out Detect/Reset, DMA, POR, PWM, WDT
Number of I/O	36
Program Memory Size	16KB (16K x 8)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	8K x 8
Voltage - Supply (Vcc/Vdd)	1.62V ~ 3.6V
Data Converters	A/D 8x12b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	48-TQFP
Supplier Device Package	48-TQFP (7x7)
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/at32uc3l016-aur

If TCR is zero when writing to TCRR, the TCR and MAR are automatically updated with the value written in TCRR and MARR.

7.5.5 Ring Buffer

When Ring Buffer mode is enabled the TCRR and MARR registers will not be cleared when TCR and MAR registers reload. This allows the PDCA to read or write to the same memory region over and over again until the transfer is actively stopped by the user. Ring Buffer mode is enabled by writing a one to the Ring Buffer bit in the Mode Register (MR.RING).

7.5.6 Peripheral Selection

The Peripheral Select Register (PSR) decides which peripheral should be connected to the PDCA channel. A peripheral is selected by writing the corresponding Peripheral Identity (PID) to the PID field in the PSR register. Writing the PID will both select the direction of the transfer (memory to peripheral or peripheral to memory), which handshake interface to use, and the address of the peripheral holding register. Refer to the Peripheral Identity (PID) table in the Module Configuration section for the peripheral PID values.

7.5.7 Transfer Size

The transfer size can be set individually for each channel to be either byte, halfword or word (8-bit, 16-bit or 32-bit respectively). Transfer size is set by writing the desired value to the Transfer Size field in the Mode Register (MR.SIZE).

When the PDCA moves data between peripherals and memory, data is automatically sized and aligned. When memory is accessed, the size specified in MR.SIZE and system alignment is used. When a peripheral register is accessed the data to be transferred is converted to a word where bit n in the data corresponds to bit n in the peripheral register. If the transfer size is byte or halfword, bits greater than 8 and16 respectively are set to zero.

Refer to the Module Configuration section for information regarding what peripheral registers are used for the different peripherals and then to the peripheral specific chapter for information about the size option available for the different registers.

7.5.8 Enabling and Disabling

Each DMA channel is enabled by writing a one to the Transfer Enable bit in the Control Register (CR.TEN) and disabled by writing a one to the Transfer Disable bit (CR.TDIS). The current status can be read from the Status Register (SR).

While the PDCA channel is enabled all DMA request will be handled as long the TCR and TCRR is not zero.

7.5.9 Interrupts

Interrupts can be enabled by writing a one to the corresponding bit in the Interrupt Enable Register (IER) and disabled by writing a one to the corresponding bit in the Interrupt Disable Register (IDR). The Interrupt Mask Register (IMR) can be read to see whether an interrupt is enabled or not. The current status of an interrupt source can be read through the Interrupt Status Register (ISR).

The PDCA has three interrupt sources:

- Reload Counter Zero - The TCRR register is zero.
- Transfer Finished - Both the TCR and TCRR registers are zero.
- Transfer Error - An error has occurred in accessing memory.

7.7.22 Performance Channel 0 Write Stall Cycles

Name: PWSTALL0

Access Type: Read-only

Offset: 0x814

Reset Value: 0x00000000

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

- **STALL: Stall Cycles Counted Since Last Reset**

  Clock cycles are counted using the CLK_PDCA_HSB clock

12.7.20 Version Register

Name: VERSION

Access Type: Read-Only

Offset: 0x3FC

Reset Value: -

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

- **VARIANT: Variant Number**
  Reserved. No functionality associated.
- **VERSION: Version Number**
  Version number of the module. No functionality associated.

13.6.9 32 KHz Oscillator Control Register

Name: OSCCTRL32

Access Type: Read/Write

Reset Value: 0x00000004

31	30	29	28	27	26	25	24
RESERVED	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	-	STARTUP[2:0]		

15	14	13	12	11	10	9	8
-	-	-	-	-	MODE[2:0]		

7	6	5	4	3	2	1	0
-	-	-	-	EN1K	EN32K	PINSEL	OSC32EN

Note: This register is only reset by Power-On Reset

- **RESERVED**

  This bit must always be written to zero.

- **STARTUP: Oscillator Start-up Time**

  Select start-up time for 32 KHz oscillator

Table 13-3. Start-up Time for 32 KHz Oscillator

STARTUP	Number of RCSYS Clock Cycle	Approximative Equivalent Time (RCOSC = 115 kHz)
0	0	0
1	128	1.1 ms
2	8192	72.3 ms
3	16384	143 ms
4	65536	570 ms
5	131072	1.1 s
6	262144	2.3 s
7	524288	4.6 s

14.6.18 Calendar Value

Name:	CALV
Access Type:	Read/Write
Offset:	0x54
Reset Value:	0x00000000

<table>
<tr><td>31</td><td>30</td><td>29</td><td>28</td><td>27</td><td>26</td><td>25</td><td>24</td></tr>
<tr><td colspan="6">YEAR</td><td colspan="2">MONTH[3:2]</td></tr>
</table>

<table>
<tr><td>23</td><td>22</td><td>21</td><td>20</td><td>19</td><td>18</td><td>17</td><td>16</td></tr>
<tr><td colspan="2">MONTH[1:0]</td><td colspan="5">DAY</td><td>HOUR[4]</td></tr>
</table>

<table>
<tr><td>15</td><td>14</td><td>13</td><td>12</td><td>11</td><td>10</td><td>9</td><td>8</td></tr>
<tr><td colspan="4">HOUR[3:0]</td><td colspan="4">MIN[5:2]</td></tr>
</table>

<table>
<tr><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td></tr>
<tr><td colspan="2">MIN[1:0]</td><td colspan="6">SEC</td></tr>
</table>

When the SR.BUSY bit is set writes to this register will be discarded and this register will read as zero.

- **YEAR: Year**

  Current year. The year is considered a leap year if YEAR[1:0] = 0.

- **MONTH: Month**

  1 = January

  2 = February

  ...

  12 = December

- **DAY: Day**

  Day of month, starting with 1.

- **HOUR: Hour**

  Hour of day, in 24-hour clock format.

  Legal values are 0 through 23.

- **MIN: Minute**

  Minutes, 0 through 59.

- **SEC: Second**

  Seconds, 0 through 59.

Figure 16-2. Timing Diagram, Synchronous Interrupts, High Level or Rising Edge

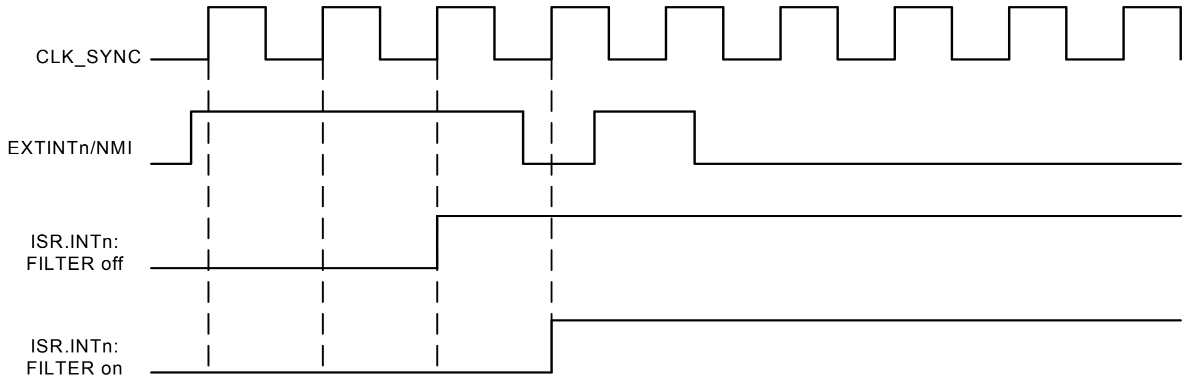
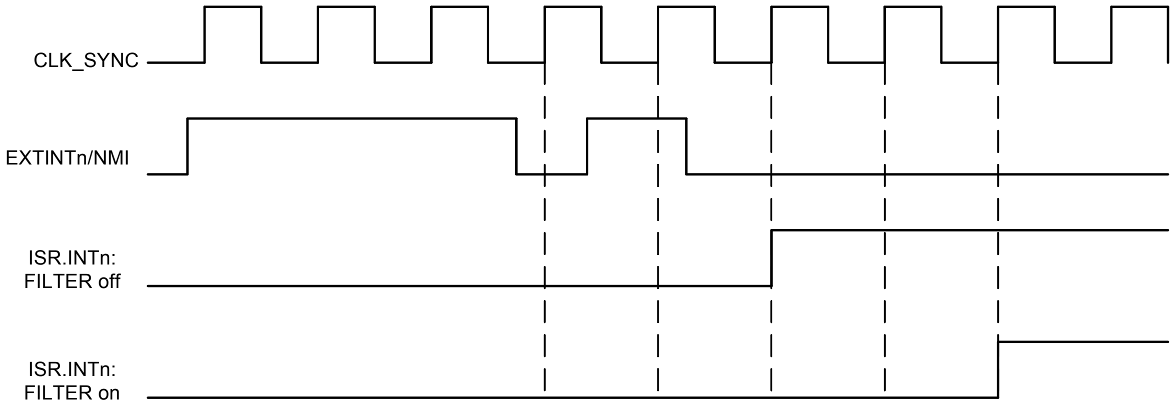


Figure 16-3. Timing Diagram, Synchronous Interrupts, Low Level or Falling Edge



16.6.3 Non-Maskable Interrupt

The NMI supports the same features as the external interrupts, and is accessed through the same registers. The description in Section 16.6.1 should be followed, accessing the NMI bit instead of the INTn bits.

The NMI is non-maskable within the CPU in the sense that it can interrupt any other execution mode. Still, as for the other external interrupts, the actual NMI input can be enabled and disabled by accessing the registers in the EIC.

16.6.4 Asynchronous Interrupts

Each external interrupt can be made asynchronous by writing a one to INTn in the ASYNC register. This will route the interrupt signal through the asynchronous path of the module. All edge interrupts will be interpreted as level interrupts and the filter is disabled. If an interrupt is configured as edge triggered interrupt in asynchronous mode, a zero in EDGE.INTn will be interpreted as low level, and a one in EDGE.INTn will be interpreted as high level.

EIC_WAKE will be set immediately after the source triggers the interrupt, while the corresponding bit in ISR and the interrupt to the interrupt controller will be set on the next rising edge of CLK_SYNC. Please refer to Figure 16-4 on page 297 for details.

17.6.9 Interrupt Clear Register

Name: ICR

Access Type: Write-only

Offset: 0x020

Reset Value: 0x00000000

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
-	-	-	-	-	-	RCLKRDY	DONE

Writing a zero to a bit in this register has no effect.

Writing a one to a bit in this register will clear the corresponding bit in ISR and the corresponding interrupt request.

18.7.14 Interrupt Mode Register 1

Name:	IMR1
Access:	Read/Write, Set, Clear, Toggle
Offset:	0x0B0, 0x0B4, 0x0B8, 0x0BC
Reset Value:	-

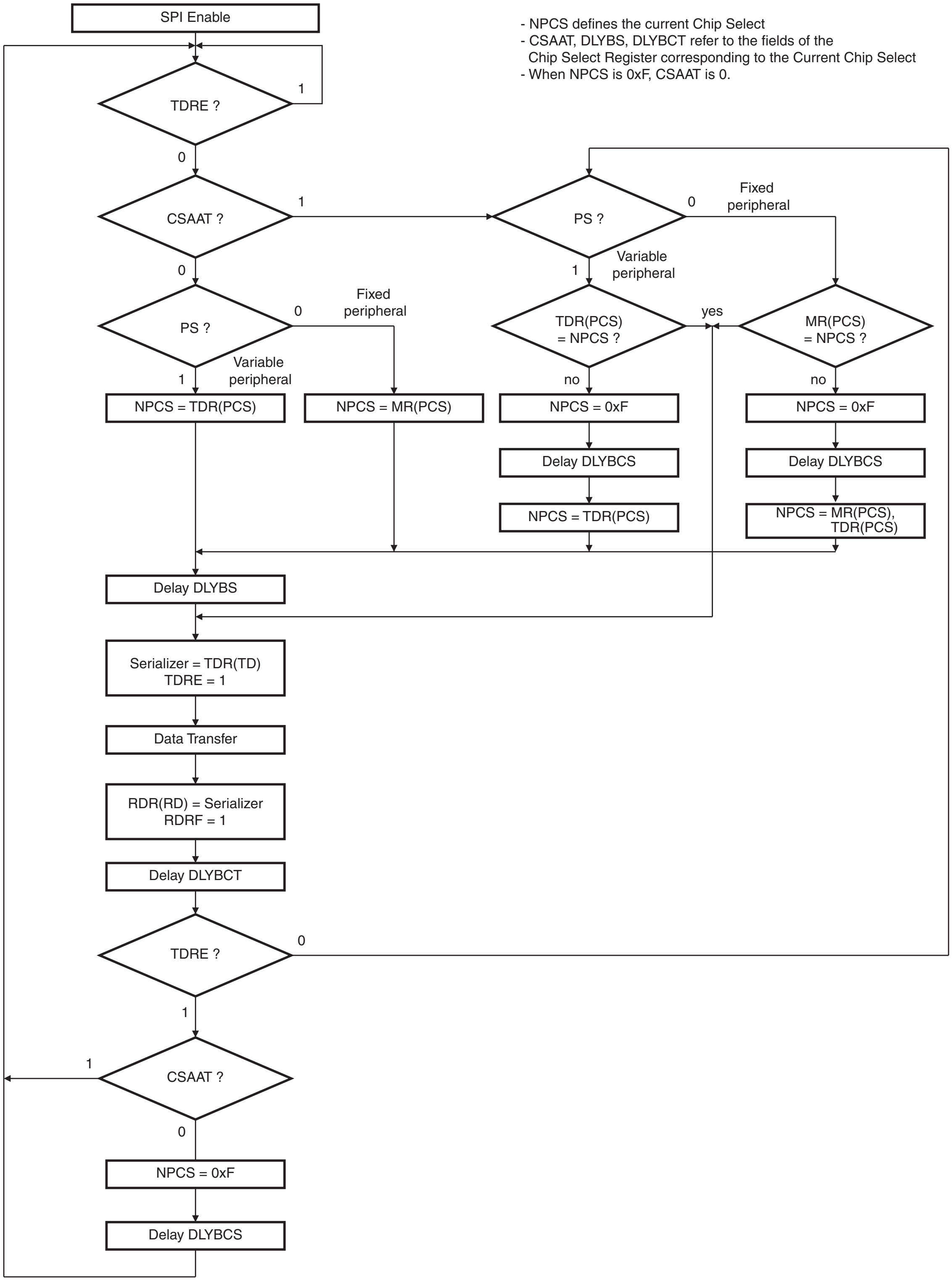
31	30	29	28	27	26	25	24
P31	P30	P29	P28	P27	P26	P25	P24
23	22	21	20	19	18	17	16
P23	P22	P21	P20	P19	P18	P17	P16
15	14	13	12	11	10	9	8
P15	P14	P13	P12	P11	P10	P9	P8
7	6	5	4	3	2	1	0
P7	P6	P5	P4	P3	P2	P1	P0

- **P0-31: Interrupt Mode Bit 1**

{IMR1, IMR0}	Interrupt Mode
00	Pin Change
01	Rising Edge
10	Falling Edge
11	Reserved

20.7.3.2 Master mode flow diagram

Figure 20-6. Master Mode Flow Diagram



TWI transfers require the slave to acknowledge each received data 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. The master polls the data line during this clock pulse and sets the Data Acknowledge bit (DNACK) in the Status Register if the slave does not acknowledge the data byte. As with the other status bits, an interrupt can be generated if enabled in the Interrupt Enable Register (IER).

TXRDY is used as Transmit Ready for the Peripheral DMA Controller transmit channel.

The end of a command is marked when the TWIM sets the SR.CCOMP bit. See Figure 21-6 and Figure 21-7.

Figure 21-6. Master Write with One Data Byte

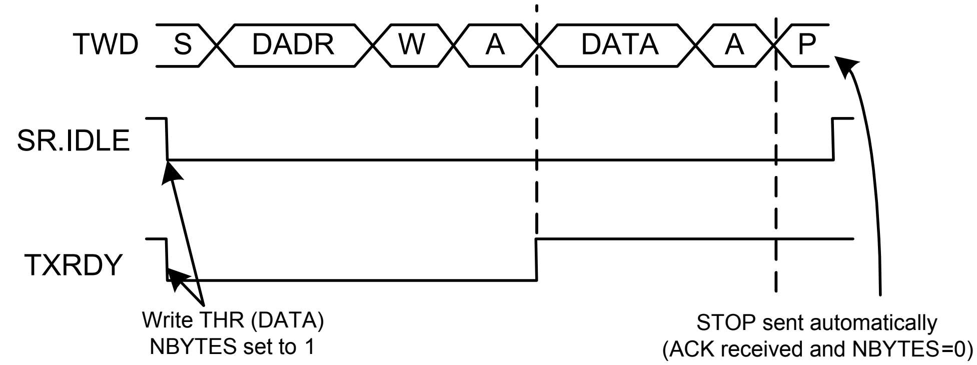
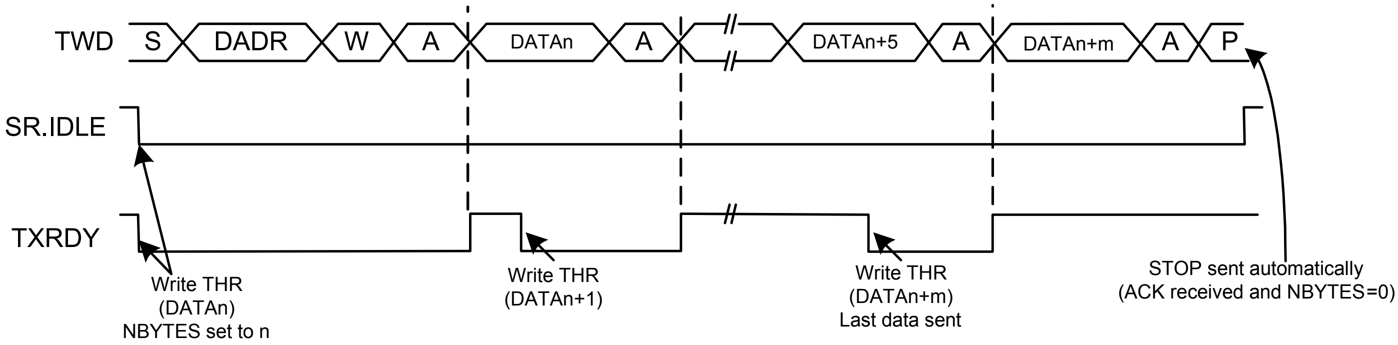


Figure 21-7. Master Write with Multiple Data Bytes



21.8.4 Master Receiver Mode

A START condition is transmitted and master receiver mode is initiated when the bus is free and CMDR has been written with START=1 and READ=1. START and SADR+R will then be transmitted. During the address acknowledge clock pulse (9th pulse), the master releases the data line (HIGH), enabling the slave to pull it down in order to acknowledge the address. The master polls the data line during this clock pulse and sets the Address Not Acknowledged bit (ANAK) in the Status Register if no slave acknowledges the address.

After the address phase, the following is repeated:

while (NBYTES>0)

21.8.8 Ten Bit Addressing

Writing a one to CMDR.TENBIT enables 10-bit addressing in hardware. Performing transfers with 10-bit addressing is similar to transfers with 7-bit addresses, except that bits 9:7 of CMDR.SADR must be written appropriately.

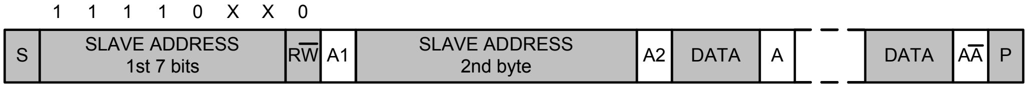
In Figure 21-14 and Figure 21-15, the grey boxes represent signals driven by the master, the white boxes are driven by the slave.

21.8.8.1 Master Transmitter

To perform a master transmitter transfer:

1. Write CMDR with TENBIT=1, REPSAME=0, READ=0, START=1, STOP=1 and the desired address and NBYTES value.

Figure 21-14. A Write Transfer with 10-bit Addressing



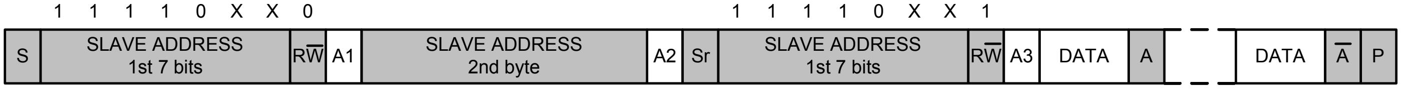
21.8.8.2 Master Receiver

When using master receiver mode with 10-bit addressing, CMDR.REPSAME must also be controlled. CMDR.REPSAME must be written to one when the address phase of the transfer should consist of only 1 address byte (the 11110xx byte) and not 2 address bytes. The I²C standard specifies that such addressing is required when addressing a slave for reads using 10-bit addressing.

To perform a master receiver transfer:

1. Write CMDR with TENBIT=1, REPSAME=0, READ=0, START=1, STOP=0, NBYTES=0 and the desired address.
2. Write NCMDR with TENBIT=1, REPSAME=1, READ=1, START=1, STOP=1 and the desired address and NBYTES value.

Figure 21-15. A Read Transfer with 10-bit Addressing



21.8.9 SMBus Mode

SMBus mode is enabled and disabled by writing to the SMEN and SMDIS bits in CR. SMBus mode operation is similar to I²C operation with the following exceptions:

- Only 7-bit addressing can be used.
- The SMBus standard describes a set of timeout values to ensure progress and throughput on the bus. These timeout values must be written into SMBTR.
- Transmissions can optionally include a CRC byte, called Packet Error Check (PEC).
- A dedicated bus line, SMBALERT, allows a slave to get a master's attention.
- A set of addresses have been reserved for protocol handling, such as Alert Response Address (ARA) and Host Header (HH) Address.

22.8.7 Wakeup from Sleep Modes by TWI Address Match

The TWIS is able to wake the device up from a sleep mode upon an address match, including sleep modes where CLK_TWIS is stopped. After detecting the START condition on the bus, The TWIS will stretch TWCK until CLK_TWIS has started. The time required for starting CLK_TWIS depends on which sleep mode the device is in. After CLK_TWIS has started, the TWIS releases its TWCK stretching and receives one byte of data on the bus. At this time, only a limited part of the device, including the TWIS, receives a clock, thus saving power. The TWIS goes on to receive the slave address. If the address phase causes a TWIS address match, the entire device is wakened and normal TWIS address matching actions are performed. Normal TWI transfer then follows. If the TWIS is not addressed, CLK_TWIS is automatically stopped and the device returns to its original sleep mode.

22.8.8 Identifying Bus Events

This chapter lists the different bus events, and how these affects the bits in the TWIS registers. This is intended to help writing drivers for the TWIS.

Table 22-5. Bus Events

Event	Effect
Slave transmitter has sent a data byte	SR.THR is cleared. SR.BTF is set. The value of the ACK bit sent immediately after the data byte is given by CR.ACK.
Slave receiver has received a data byte	SR.RHR is set. SR.BTF is set. SR.NAK updated according to value of ACK bit received from master.
Start+Sadr on bus, but address is to another slave	None.
Start+Sadr on bus, current slave is addressed, but address match enable bit in CR is not set	None.
Start+Sadr on bus, current slave is addressed, corresponding address match enable bit in CR set	Correct address match bit in SR is set. SR.TRA updated according to transfer direction (updating is done one CLK_TWIS cycle after address match bit is set) Slave enters appropriate transfer direction mode and data transfer can commence.
Start+Sadr on bus, current slave is addressed, corresponding address match enable bit in CR set, SR.STREN and SR.SOAM are set.	Correct address match bit in SR is set. SR.TRA updated according to transfer direction (updating is done one CLK_TWIS cycle after address match bit is set). Slave stretches TWCK immediately after transmitting the address ACK bit. TWCK remains stretched until all address match bits in SR have been cleared. Slave enters appropriate transfer direction mode and data transfer can commence.
Repeated Start received after being addressed	SR.REP set. SR.TCOMP unchanged.
Stop received after being addressed	SR.STO set. SR.TCOMP set.

The RA Loading Selection field in CMRn (CMRn.LDRA) defines the TIOA edge for the loading of the RA register, and the RB Loading Selection field in CMRn (CMRn.LDRB) defines the TIOA edge for the loading of the RB register.

RA is loaded only if it has not been loaded since the last trigger or if RB has been loaded since the last loading of RA.

RB is loaded only if RA has been loaded since the last trigger or the last loading of RB.

Loading RA or RB before the read of the last value loaded sets the Load Overrun Status bit in SRn (SRn.LOVRS). In this case, the old value is overwritten.

24.6.2.2 Trigger conditions

In addition to the SYNC signal, the software trigger and the RC compare trigger, an external trigger can be defined.

The TIOA or TIOB External Trigger Selection bit in CMRn (CMRn.ABETRG) selects TIOA or TIOB input signal as an external trigger. The External Trigger Edge Selection bit in CMRn (CMRn.ETREDG) defines the edge (rising, falling or both) detected to generate an external trigger. If CMRn.ETRGEDG is zero (none), the external trigger is disabled.

- **CPCSTOP: Counter Clock Stopped with RC Compare**

  1: Counter clock is stopped when counter reaches RC.

  0: Counter clock is not stopped when counter reaches RC.

- **BURST: Burst Signal Selection**

BURST	Burst Signal Selection
0	The clock is not gated by an external signal.
1	XC0 is ANDed with the selected clock.
2	XC1 is ANDed with the selected clock.
3	XC2 is ANDed with the selected clock.

- **CLKI: Clock Invert**

  1: Counter is incremented on falling edge of the clock.

  0: Counter is incremented on rising edge of the clock.

- **TCCLKS: Clock Selection**

TCCLKS	Clock Selected
0	TIMER_CLOCK1
1	TIMER_CLOCK2
2	TIMER_CLOCK3
3	TIMER_CLOCK4
4	TIMER_CLOCK5
5	XC0
6	XC1
7	XC2

28.7.3 Pin Mode Registers 0 and 1

Name: PINMODE0 and PINMODE1

Access Type: Read/Write

Offset: 0x08, 0x0C

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-							

23	22	21	20	19	18	17	16
-							SP[16]

15	14	13	12	11	10	9	8
SP[15:8]							

7	6	5	4	3	2	1	0
SP[7:0]							

- **SP: Sense Pair Mode Selection**

  Each SP[n] bit determines the operation mode of sense pair n (CSAn and CSBn pins). The (PINMODE1.SP[n] PINMODE0.SP[n]) bits have the following definitions:

  00: Sense pair n disabled.

  01: Sense pair n is assigned to QTouch Group A.

  10: Sense pair n is assigned to QTouch Group B.

  11: Sense pair n is assigned to the QMatrix Group.

30.7.6 Interrupt Mask Register

Name: IMR

Access Type: Read-only

Offset: 0x14

Reset Value: 0x00000000

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

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

15	14	13	12	11	10	9	8
-	-	TRMIS	-	-	OVERRUN	DREADYINT	READYINT

7	6	5	4	3	2	1	0
-	-	-	-	-	-	-	-

0: The corresponding interrupt is disabled.

1: The corresponding interrupt is enabled.

A bit in this register is cleared when the corresponding bit in IDR is written to one.

A bit in this register is set when the corresponding bit in IER is written to one.

30.7.8 Transmit Holding Register

Name:	THR
Access Type:	Read/Write
Offset:	0x1C
Reset Value:	0x00000000

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
TXDATA							

- **TXDATA: Transmit Data**

  The data to send.

30.7.11 Clock Request Register

Name: CLKR

Access Type: Read/Write

Offset: 0x28

Reset Value: 0x00000000

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
-	-	-	-	-	-	-	CLKEN

- **CLKEN: Clock Enable**

  0: The aWire clock is disabled.

  1: The aWire clock is enabled.

  Writing a zero to this bit will disable the aWire clock.

  Writing a one to this bit will enable the aWire clock.

Table 31-59. STATUS_INFO Details

Response	Details
Response value	0xC4
Additional data	2 status bytes

31.6.8.8 MEMORY_SPEED

Counts the number of RC120M clock cycles it takes to sync one message to the SAB interface and back again. The SAB clock speed (f_{sab}) can be calculated using the following formula:

$$f_{sab} = \frac{3f_{aw}}{CV-3}$$

Table 31-60. MEMORY_SPEED Details

Response	Details
Response value	0xC5
Additional data	Clock cycle count (MS)

31.6.9 Security Restrictions

When the security fuse in the Flash is programmed, the following aWire commands are limited:

- MEMORY_WRITE
- MEMORY_READ

Unlimited access to these instructions is restored when the security fuse is erased by the CHIP_ERASE aWire command.

Note that the security bit will read as programmed and block these instructions also if the Flash Controller is statically reset.

The VERSION register reads 0x100 instead of 0x101.
Fix/Workaround
None.

35.4.18 ACIFB

1. **Generic clock sources in sleep modes.**
The ACIFB should not use RC32K or CLK_1K as generic clock source if the chip uses sleep modes.
Fix/Workaround
None.

2. **Negative offset**
The static offset of the analog comparator is approximately -50 mV
Fix/Workaround
None.

3. **CONFW.WEVSRC and CONFW.WEVEN are not correctly described in the user interface**
CONFW.WEVSRC is only two bits instead of three bits wide. Only values 0, 1, and 2 can be written to this register. CONFW.WEVEN is in bit position 10 instead of 11.
Fix/Workaround
Only write values 0, 1, and 2 to CONFW.WEVSRC. When reading CONFW.WEVSRC, disregard the third bit. Read/write bit 10 to access CONFW.WEVEN.

4. **VERSION register reads 0x200**
The VERSION register reads 0x200 instead of 0x212.
Fix/Workaround
None.

35.4.19 CAT

1. **Switch off discharge current when reaching 0 V**
The discharge current will switch off when reaching MGCFG1.MAX, not when reaching 0 V.
Fix/Workaround
None.

2. **CAT external capacitors are not clamped to ground when CAT is idle**
The CAT module does not clamp the external capacitors to ground when it is idle. The capacitors are left floating, so they could accumulate small amounts of charge.
Fix/Workaround
None.

3. **DISHIFT field is stuck at zero**
The DISHIFT field in the MGCFG1, TGACFG1, TGBCFG1, and ATCFG1 registers is stuck at zero and cannot be written to a different value. Capacitor discharge time will only be determined by the DILEN field.
Fix/Workaround
None.

4. **MGCFG2.CONSEN field is stuck at zero**
The CONSEN field in the MGCFG2 register is stuck at zero and cannot be written to a different value. The CAT consensus filter does not function properly, so termination of QMatrix data acquisition is controlled only by the MAX field in MGCFG1.