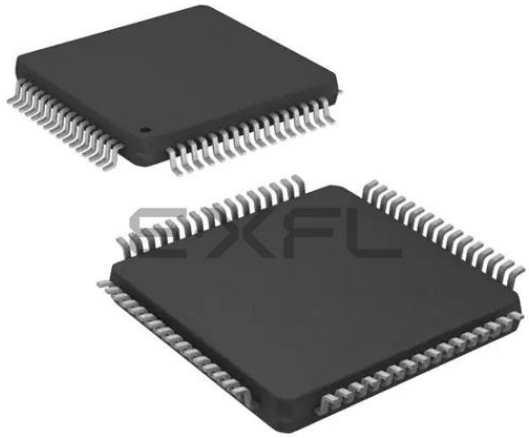


Welcome to E-XFL.COM

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	AVR
Core Size	8/16-Bit
Speed	32MHz
Connectivity	I²C, IrDA, SPI, UART/USART, USB
Peripherals	Brown-out Detect/Reset, DMA, POR, PWM, WDT
Number of I/O	50
Program Memory Size	384KB (384K x 8)
Program Memory Type	FLASH
EEPROM Size	4K x 8
RAM Size	32K x 8
Voltage - Supply (Vcc/Vdd)	1.6V ~ 3.6V
Data Converters	A/D 16x12b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	64-TQFP
Supplier Device Package	64-TQFP (14x14)
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/atxmega384c3-au

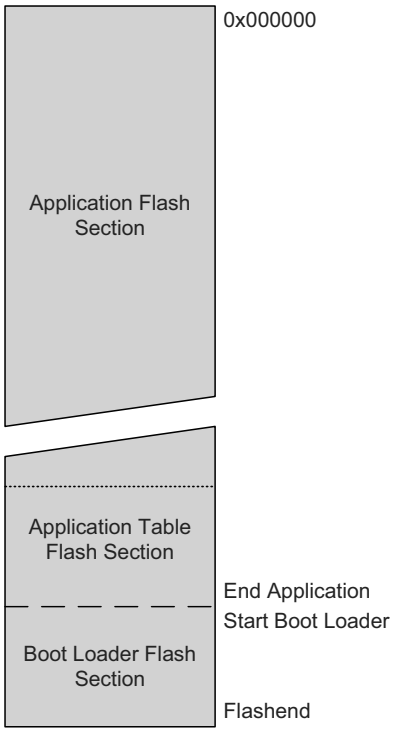
3.13 Fuse Lock

For some system-critical features, it is possible to program a fuse to disable all changes to the associated I/O control registers. If this is done, it will not be possible to change the registers from the user software, and the fuse can only be reprogrammed using an external programmer. Details on this are described in the datasheet module where this feature is available.

of the different sections are fixed, but device-dependent. These two sections have separate lock bits, and can have different levels of protection. The store program memory (SPM) instruction, which is used to write to the flash from the application software, will only operate when executed from the boot loader section.

The application section contains an application table section with separate lock settings. This enables safe storage of nonvolatile data in the program memory.

Figure 4-1. Flash Memory Sections



4.3.1 Application Section

The Application section is the section of the flash that is used for storing the executable application code. The protection level for the application section can be selected by the boot lock bits for this section. The application section can not store any boot loader code since the SPM instruction cannot be executed from the application section.

4.3.2 Application Table Section

The application table section is a part of the application section of the flash memory that can be used for storing data. The size is identical to the boot loader section. The protection level for the application table section can be selected by the boot lock bits for this section. The possibilities for different protection levels on the application section and the application table section enable safe parameter storage in the program memory. If this section is not used for data, application code can reside here.

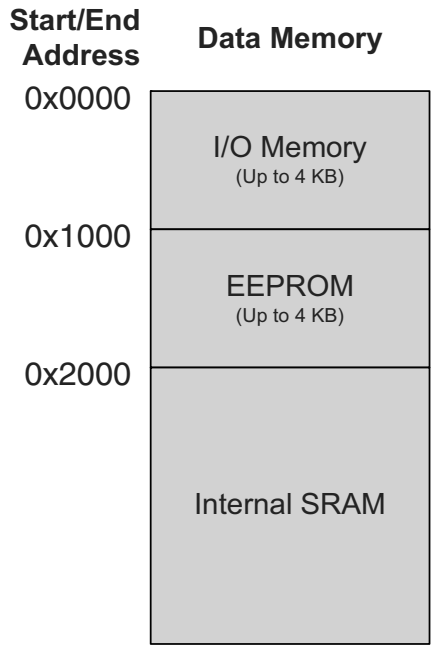
4.3.3 Boot Loader Section

While the application section is used for storing the application code, the boot loader software must be located in the boot loader section because the SPM instruction can only initiate programming when executing from this section. The SPM instruction can access the entire flash, including the boot loader section itself. The protection level for the boot loader section can be selected by the boot loader lock bits. If this section is not used for boot loader software, application code can be stored here.

4.5 Data Memory

The data memory contains the I/O memory, internal SRAM, optionally memory mapped and EEPROM. The data memory is organized as one continuous memory section, as shown in Figure 4-2.

Figure 4-2. Data Memory Map



I/O memory, EEPROM, and SRAM will always have the same start addresses for all XMEGA devices.

4.6 Internal SRAM

The internal SRAM always starts at hexadecimal address 0x2000. SRAM is accessed by the CPU using the load (LD/LDS/LDD) and store (ST/STS/STD) instructions.

4.7 EEPROM

All XMEGA devices have EEPROM for nonvolatile data storage. It is addressable in a separate data space (default) or memory mapped and accessed in normal data space. The EEPROM supports both byte and page access. Memory mapped EEPROM allows highly efficient EEPROM reading and EEPROM buffer loading. When doing this, EEPROM is accessible using load and store instructions. Memory mapped EEPROM will always start at hexadecimal address 0x1000.

4.8 I/O Memory

The status and configuration registers for peripherals and modules, including the CPU, are addressable through I/O memory locations. All I/O locations can be accessed by the load (LD/LDS/LDD) and store (ST/STS/STD) instructions, which are used to transfer data between the 32 registers in the register file and the I/O memory. The IN and OUT instructions can address I/O memory locations in the range of 0x00 to 0x3F directly. In the address range 0x00 - 0x1F, single-cycle instructions for manipulation and checking of individual bits are available.

4.8.1 General Purpose I/O Registers

The lowest four I/O memory addresses are reserved as general purpose I/O registers. These registers can be used for storing global variables and flags, as they are directly bit-accessible using the SBI, CBI, SBIS, and SBIC instructions.

4.15.14 COORDX1 – Wafer Coordinate X Register 1

Bit	7	6	5	4	3	2	1	0
0x13	COORDX1[7:0]							
Read/Write	R	R	R	R	R	R	R	R
Initial Value	x	x	x	x	x	x	x	x

- **Bit 7:0 – COORDX0[7:0]: Wafer Coordinate X Byte 1**

This byte contains byte 1 of wafer coordinate X for the device.

4.15.15 COORDY0 – Wafer Coordinate Y Register 0

Bit	7	6	5	4	3	2	1	0
0x14	COORDY0[7:0]							
Read/Write	R	R	R	R	R	R	R	R
Initial Value	x	x	x	x	x	x	x	x

- **Bit 7:0 – COORDY0[7:0]: Wafer Coordinate Y Byte 0**

This byte contains byte 0 of wafer coordinate Y for the device.

4.15.16 COORDY1 – Wafer Coordinate Y Register 1

Bit	7	6	5	4	3	2	1	0
0x15	COORDY1[7:0]							
Read/Write	R	R	R	R	R	R	R	R
Initial Value	x	x	x	x	x	x	x	x

- **Bit 7:0 – COORDY1[7:0]: Wafer Coordinate Y Byte 1**

This byte contains byte 1 of wafer coordinate Y for the device.

4.15.17 USBCAL0 – USB Calibration Register 0

USBCAL0 and USBCAL1 contain the calibration value for the USB pins. Calibration is done during production to enable operation without requiring external components on the USB lines for the device. The calibration bytes are not loaded automatically into the USB calibration registers, and so this must be done from software.

Bit	7	6	5	4	3	2	1	0
0x1A	USBCAL0[7:0]							
Read/Write	R	R	R	R	R	R	R	R
Initial Value	x	x	x	x	x	x	x	x

- **Bit 7:0 – USBCAL0[7:0]: USB Pad Calibration byte 0**

This byte contains byte 0 of the USB pin calibration data, and must be loaded into the USB CALL register.

4.15.18 USBCAL1 – USB Pad Calibration Register 1

Bit	7	6	5	4	3	2	1	0
0x1B	USBCAL1[7:0]							
Read/Write	R	R	R	R	R	R	R	R
Initial Value	x	x	x	x	x	x	x	x

- **Bit 7:0 – USBCAL1[7:0]: USB Pad Calibration byte 1**

This byte contains byte 1 of the USB pin calibration data.

interrupt, ERRIF will not be cleared when the interrupt vector is executed. This flag is cleared by writing a one to this location.

- **Bit 4 – TRNIF: Channel n Transaction Complete Interrupt Flag**

When a transaction on the DMA channel has been completed, the TRNIF flag will be set and the optional interrupt is generated. When repeat is not enabled, the transaction is complete and TRNIFR is set after the block transfer. When unlimited repeat is enabled, TRNIF is also set after each block transfer.

Since the DMA channel transaction n complete interrupt shares the interrupt address with the DMA channel error interrupt, TRNIF will not be cleared when the interrupt vector is executed. This flag is cleared by writing a one to this location.

- **Bit 3:2 – ERRINTLVL[1:0]: Channel Error Interrupt Level**

These bits enable the interrupt for DMA channel transfer errors and select the interrupt level, as described in “Interrupts and Programmable Multilevel Interrupt Controller” on page 112. The enabled interrupt will trigger for the conditions when ERRIF is set.

- **Bit 1:0 – TRNINTLVL[1:0]: Channel Transaction Complete Interrupt Level**

These bits enable the interrupt for DMA channel transaction completes and select the interrupt level, as described in “Interrupts and Programmable Multilevel Interrupt Controller” on page 112. The enabled interrupt will trigger for the conditions when TRNIF is set.

5.14.3 ADDRCTRL – Address Control Register

Bit	7	6	5	4	3	2	1	0
+0x02	SRCRELOAD[1:0]		SRCDIR[1:0]		DESTRELOAD[1:0]		DESTDIR[1:0]	
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Initial Value	0	0	0	0	0	0	0	0

- **Bit 7:6 – SRCRELOAD[1:0]: Channel Source Address Reload**

These bits decide the DMA channel source address reload according to Table 5-4. A write to these bits is ignored while the channel is busy.

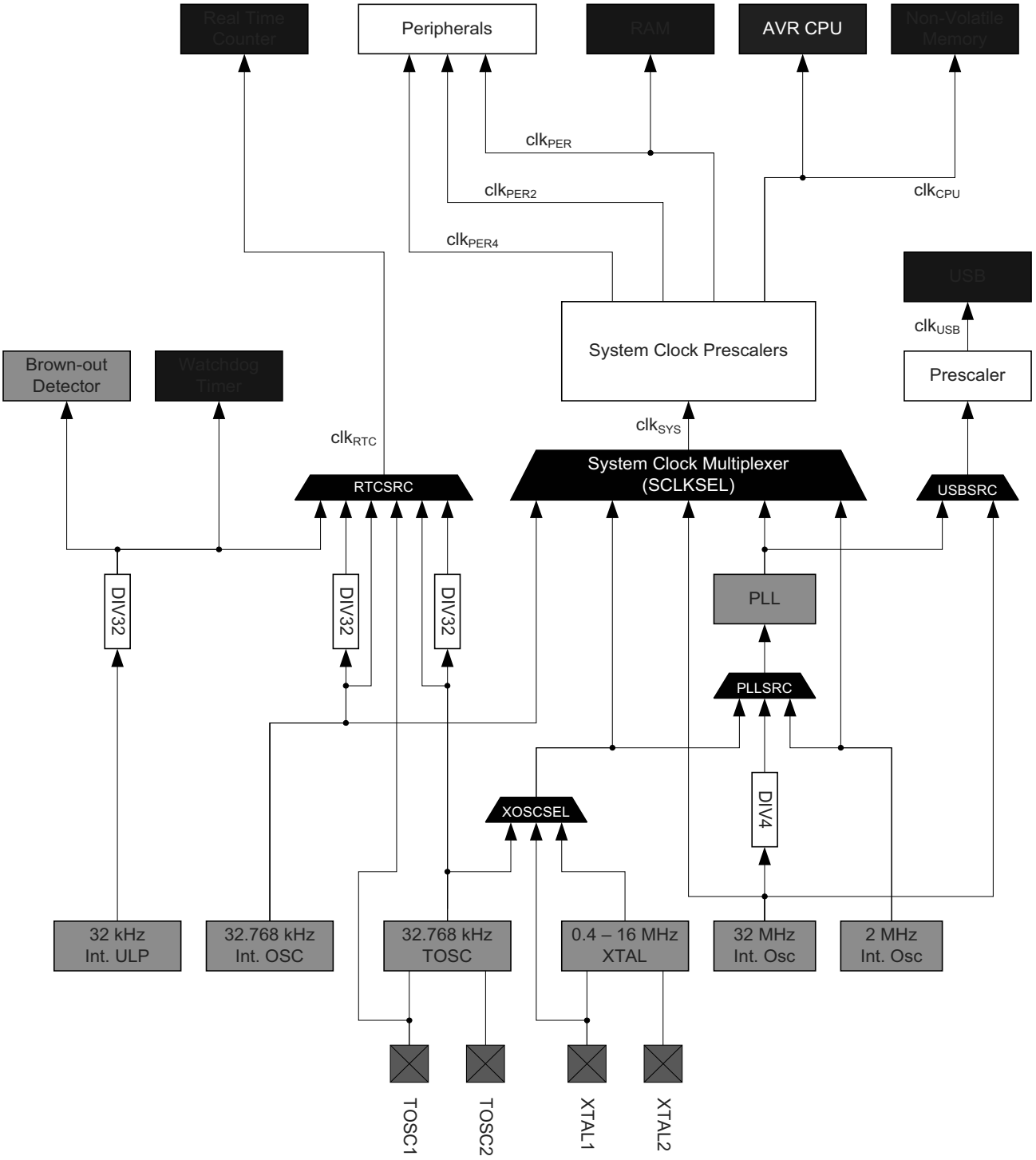
Table 5-4. DMA Channel Source Address Reload Settings

SRCRELOAD[1:0]	Group configuration	Description
00	NONE	No reload performed.
01	BLOCK	DMA source address register is reloaded with initial value at end of each block transfer.
10	BURST	DMA source address register is reloaded with initial value at end of each burst transfer.
11	TRANSACTION	DMA source address register is reloaded with initial value at end of each transaction.

- **Bit 5:4 – SRCDIR[1:0]: Channel Source Address Mode**

These bits decide the DMA channel source address mode according to Table 5-5. These bits cannot be changed if the channel is busy.

Figure 7-1. The Clock System, Clock Sources, and Clock Distribution



7.3 Clock Distribution

Figure 7-1 on page 74 presents the principal clock distribution system used in XMEGA devices.

7.3.1 System Clock - Clk_{SYS}

The system clock is the output from the main system clock selection. This is fed into the prescalers that are used to generate all internal clocks except the asynchronous and USB clocks.

7.3.2 CPU Clock - Clk_{CPU}

The CPU clock is routed to the CPU and nonvolatile memory. Halting the CPU clock inhibits the CPU from executing instructions.

7.3.3 Peripheral Clock - Clk_{PER}

The majority of peripherals and system modules use the peripheral clock. This includes the DMA controller, event system, interrupt controller, external bus interface and RAM. This clock is always synchronous to the CPU clock, but may run even when the CPU clock is turned off.

7.3.4 Peripheral 2x/4x Clocks - Clk_{PER2}/Clk_{PER4}

Modules that can run at two or four times the CPU clock frequency can use the peripheral 2x and peripheral 4x clocks.

7.3.5 Asynchronous Clock - Clk_{RTC}

The asynchronous clock allows the real-time counter (RTC) to be clocked directly from an external 32.768kHz crystal oscillator or the 32 times prescaled output from the internal 32.768kHz oscillator or ULP oscillator. The dedicated clock domain allows operation of this peripheral even when the device is in sleep mode and the rest of the clocks are stopped.

7.3.6 USB Clock - Clk_{USB}

The USB device module requires a 12MHz or 48MHz clock. It has a separate clock source selection in order to avoid system clock source limitations when USB is used.

7.4 Clock Sources

The clock sources are divided in two main groups: internal oscillators and external clock sources. Most of the clock sources can be directly enabled and disabled from software, while others are automatically enabled or disabled, depending on peripheral settings. After reset, the device starts up running from the 2MHz internal oscillator. The other clock sources, DFLLs and PLL, are turned off by default.

7.4.1 Internal Oscillators

The internal oscillators do not require any external components to run. For details on characteristics and accuracy of the internal oscillators, refer to the device datasheet.

7.4.1.1 32kHz Ultra Low Power Oscillator

This oscillator provides an approximate 32kHz clock. The 32kHz ultra low power (ULP) internal oscillator is a very low power clock source, and it is not designed for high accuracy.The oscillator employs a built-in prescaler that provides a 1kHz output, see “RTCCTRL – RTC Control Register” on page 83 for details. The oscillator is automatically enabled/disabled when it is used as clock source for any part of the device. This oscillator can be selected as the clock source for the RTC.

7.4.1.2 32.768kHz Calibrated Oscillator

This oscillator provides an approximate 32.768kHz clock. It is calibrated during production to provide a default frequency close to its nominal frequency. The calibration register can also be written from software for run-time calibration of the

- **Bit 0 – RC2MRDY: 2MHz Internal Oscillator Ready**

This flag is set when the 2MHz internal oscillator is stable and is ready to be used as the system clock source.

7.10.3 XOSCCTRL – XOSC Control Register

Bit	7	6	5	4	3	2	1	0
+0x02	FRQRANGE[1:0]		X32KLPM	XOSCPWR	XOSCSEL[3:0]			
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Initial Value	0	0	0	0	0	0	0	0

- **Bit 7:6 – FRQRANGE[1:0]: 0.4 - 16MHz Crystal Oscillator Frequency Range Select**

These bits select the frequency range for the connected crystal oscillator according to Table 7-7.

Table 7-7. 16MHz Crystal Oscillator Frequency Range Selection

FRQRANGE[1:0]	Group configuration	Typical frequency range [MHz]	Recommended range for capacitors C1 and C2 [pF]
00	04TO2	0.4 - 2	100-300
01	2TO9	2 - 9	10-40
10	9TO12	9 - 12	10-40
11	12TO16	12 - 16	10-30

- **Bit 5 – X32KLPM: Crystal Oscillator 32.768kHz Low Power Mode**

Setting this bit enables the low power mode for the 32.768kHz crystal oscillator. This will reduce the swing on the TOSC2 pin.

- **Bit 4 – XOSCPWR: Crystal Oscillator Drive**

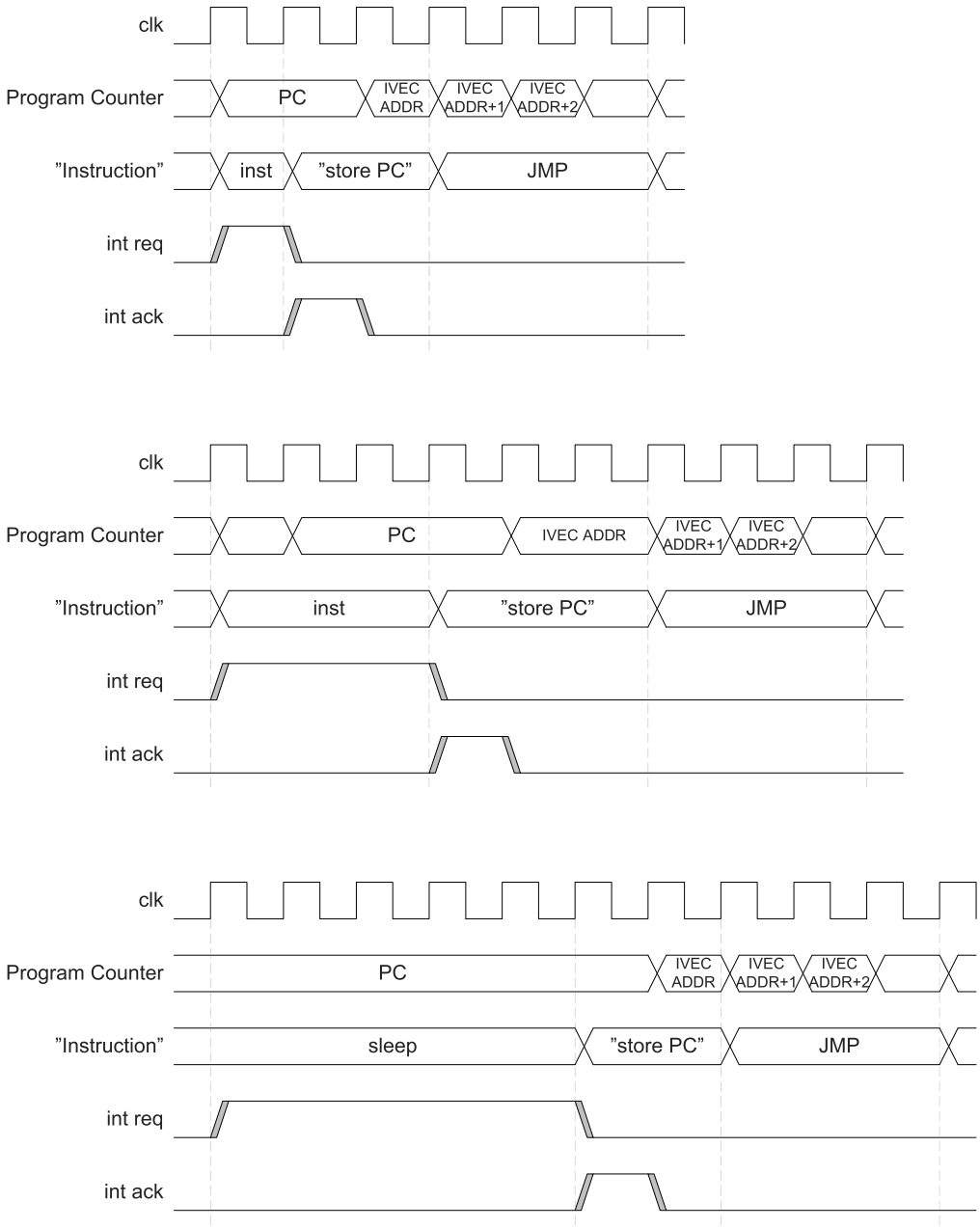
Setting this bit will increase the current in the 0.4MHz - 16MHz crystal oscillator and increase the swing on the XTAL2 pin. This allows for driving crystals with higher load or higher frequency than specified by the FRQRANGE bits.

- **Bit 3:0 – XOSCSEL[3:0]: Crystal Oscillator Selection**

These bits select the type and start-up time for the crystal or resonator that is connected to the XTAL or TOSC pins. See Table 7-8 on page 87 for crystal selections. If an external clock or external oscillator is selected as the source for the system clock, see “CTRL – Oscillator Control Register” on page 85. This configuration cannot be changed.

If an interrupt occurs during execution of a multicycle instruction, this instruction is completed before the interrupt is served. See Figure 11-2 for more details.

Figure 11-2. Interrupt Execution of a Multicycle Instruction



If an interrupt occurs when the device is in sleep mode, the interrupt execution response time is increased by five clock cycles. In addition, the response time is increased by the start-up time from the selected sleep mode.

A return from an interrupt handling routine takes four to five clock cycles, depending on the size of the program counter. During these clock cycles, the program counter is popped from the stack and the stack pointer is incremented.

12. I/O Ports

12.1 Features

- General purpose input and output pins with individual configuration
- Output driver with configurable driver and pull settings:
  - Totem-pole
  - Wired-AND
  - Wired-OR
  - Bus-keeper
  - Inverted I/O
- Input with synchronous and/or asynchronous sensing with interrupts and events
  - Sense both edges
  - Sense rising edges
  - Sense falling edges
  - Sense low level
- Optional pull-up and pull-down resistor on input and Wired-OR/AND configurations
- Asynchronous pin change sensing that can wake the device from all sleep modes
- Two port interrupts with pin masking per I/O port
- Efficient and safe access to port pins
  - Hardware read-modify-write through dedicated toggle/clear/set registers
  - Configuration of multiple pins in a single operation
  - Mapping of port registers into bit-accessible I/O memory space
- Peripheral clocks output on port pin
- Real-time counter clock output to port pin
- Event channels can be output on port pin
- Remapping of digital peripheral pin functions
  - Selectable USART, SPI, and timer/counter input/output pin locations

12.2 Overview

AVR XMEGA microcontrollers have flexible general purpose I/O ports. One port consists of up to eight port pins: pin 0 to 7. Each port pin can be configured as input or output with configurable driver and pull settings. They also implement synchronous and asynchronous input sensing with interrupts and events for selectable pin change conditions. Asynchronous pin-change sensing means that a pin change can wake the device from all sleep modes, included the modes where no clocks are running.

All functions are individual and configurable per pin, but several pins can be configured in a single operation. The pins have hardware read-modify-write (RMW) functionality for safe and correct change of drive value and/or pull resistor configuration. The direction of one port pin can be changed without unintentionally changing the direction of any other pin.

The port pin configuration also controls input and output selection of other device functions. It is possible to have both the peripheral clock and the real-time clock output to a port pin, and available for external use. The same applies to events from the event system that can be used to synchronize and control external functions. Other digital peripherals, such as USART, SPI, and timer/counters, can be remapped to selectable pin locations in order to optimize pin-out versus application needs.

Figure 12-1 on page 120 shows the I/O pin functionality and the registers that are available for controlling a pin.

12.18 Interrupt Vector Summary – Ports

Offset	Source	Interrupt description
0x00	INT0_vect	Port interrupt vector 0 offset
0x02	INT1_vect	Port interrupt vector 1 offset

13.13 Register Summary

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Page
+0x00	CTRLA	–	–	–	–	CLKSEL[3:0]				154
+0x01	CTRLB	CCDEN	CCCEN	CCBEN	CCAEN	–	WGMODE[2:0]			154
+0x02	CTRLC	–	–	–	–	CMPD	CMPC	CMPB	CMPA	155
+0x03	CTRLD	EVACT[2:0]			EVDLY	EVSEL[3:0]				155
+0x04	CTRLE	–	–	–	–	–	–	BYTEM		157
+0x05	Reserved	–	–	–	–	–	–	–	–	
+0x06	INTCTRLA	–	–	–	–	ERRINTLVL[1:0]		OVINTLVL[1:0]		157
+0x07	INTCTRLB	CCCINTLVL[1:0]		CCCINTLVL[1:0]		CCBINTLVL[1:0]		CCAINTLVL[1:0]		157
+0x08	CTRLFCLR	–	–	–	–	CMD[1:0]		LUPD	DIR	158
+0x09	CTRLFSET	–	–	–	–	CMD[1:0]		LUPD	DIR	158
+0x0A	CTRLGCLR	–	–	–	CCDBV	CCCBV	CCBBV	CCABV	PERBV	159
+0x0B	CTRLGSET	–	–	–	CCDBV	CCCBV	CCBBV	CCABV	PERBV	159
+0x0C	INTFLAGS	CCDIF	CCCIF	CCBIF	CCAIF	–	–	ERRIF	OVFIF	159
+0x0D	Reserved	–	–	–	–	–	–	–	–	
+0x0E	Reserved	–	–	–	–	–	–	–	–	
+0x0F	TEMP	TEMP[7:0]								160
+0x10 to	Reserved	–	–	–	–	–	–	–	–	
+0x20	CNTL	CNT[7:0]								160
+0x21	CNTH	CNT[15:8]								160
+0x22 to	Reserved	–	–	–	–	–	–	–	–	
+0x26	PERL	PER[7:0]								161
+0x27	PERH	PER[8:15]								161
+0x28	CCAL	CCA[7:0]								161
+0x29	CCAH	CCA[15:8]								161
+0x2A	CCBL	CCB[7:0]								161
+0x2B	CCBH	CCB[15:8]								161
+0x2C	CCCL	CCC[7:0]								161
+0x02D	CCCH	CCC[15:8]								161
+0x2E	CCDL	CCD[7:0]								161
+0x2F	CCDH	CCD[15:8]								161
+0x30 to	Reserved	–	–	–	–	–	–	–	–	
+0x36	PERBUFL	PERBUF[7:0]								162
+0x37	PERBUFH	PERBUF[15:8]								162
+0x38	CCABUFL	CCABUF[7:0]								162
+0x39	CCABUFH	CCABUF[15:8]								162
+0x3A	CCBBUFL	CCBBUF[7:0]								162
+0x3B	CCBBUFH	CCBBUF[15:8]								162
+0x3C	CCCBUFL	CCCBUF[7:0]								162
+0x3D	CCCBUFH	CCCBUF[15:8]								162
+0x3E	CCDBUFL	CCDBUF[7:0]								162
+0x3F	CCDBUFH	CCDBUF[15:8]								162

13.14 Interrupt Vector Summary

Offset	Source	Interrupt description
0x00	OVF_vect	Timer/counter overflow/underflow interrupt vector offset
0x02	ERR_vect	Timer/counter error interrupt vector offset
0x04	CCA_vect	Timer/counter compare or capture channel A interrupt vector offset
0x06	CCB_vect	Timer/counter compare or capture channel B interrupt vector offset
0x08	CCC_vect(1)	Timer/counter compare or capture channel C interrupt vector offset
0x0A	CCD_vect(1)	Timer/counter compare or capture channel D interrupt vector offset

Note: 1. Available only on timer/counters with four compare or capture channels.

The peripheral clock (clk_{PER}) is fed into the common prescaler (common for all timer/counters in a device). A selection of prescaler outputs from 1 to 1/1024 is directly available. In addition, the whole range of time prescalings from 1 to 2^{15} is available through the event system.

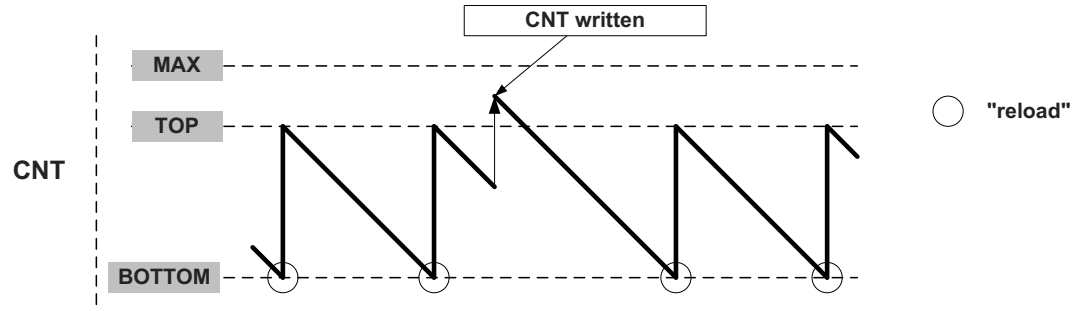
The clock selection (CLKSEL) selects one of the clock prescaler outputs or an event channel for the high-byte counter (HCNT) and low-byte counter (LCNT). By using the event system, any event source, such as an external clock signal, on any I/O pin can be used as the clock input.

By default, no clock input is selected, and the counters are not running.

14.5 Counter Operation

The counters will always count in single-slope mode. Each counter counts down for each clock cycle until it reaches BOTTOM, and then reloads the counter with the period register value at the following clock cycle.

Figure 14-3. Counter Operation

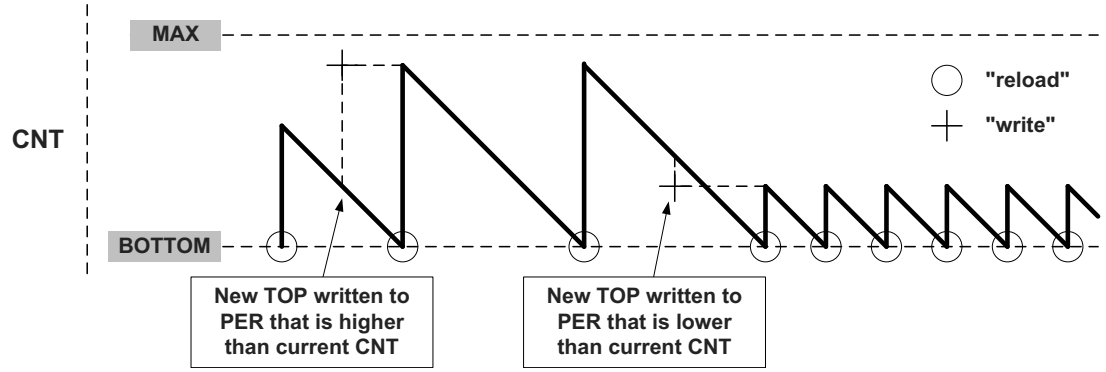


As shown in Figure 14-3, the counter can change the counter value while running. The write access has higher priority than the count clear, and reloads and will be immediate.

14.5.1 Changing the Period

The counter period is changed by writing a new TOP value to the period register. Since the counter is counting down, the period register can be written at any time without affecting the current period, as shown in Figure 14-4. This prevents wraparound and generation of odd waveforms.

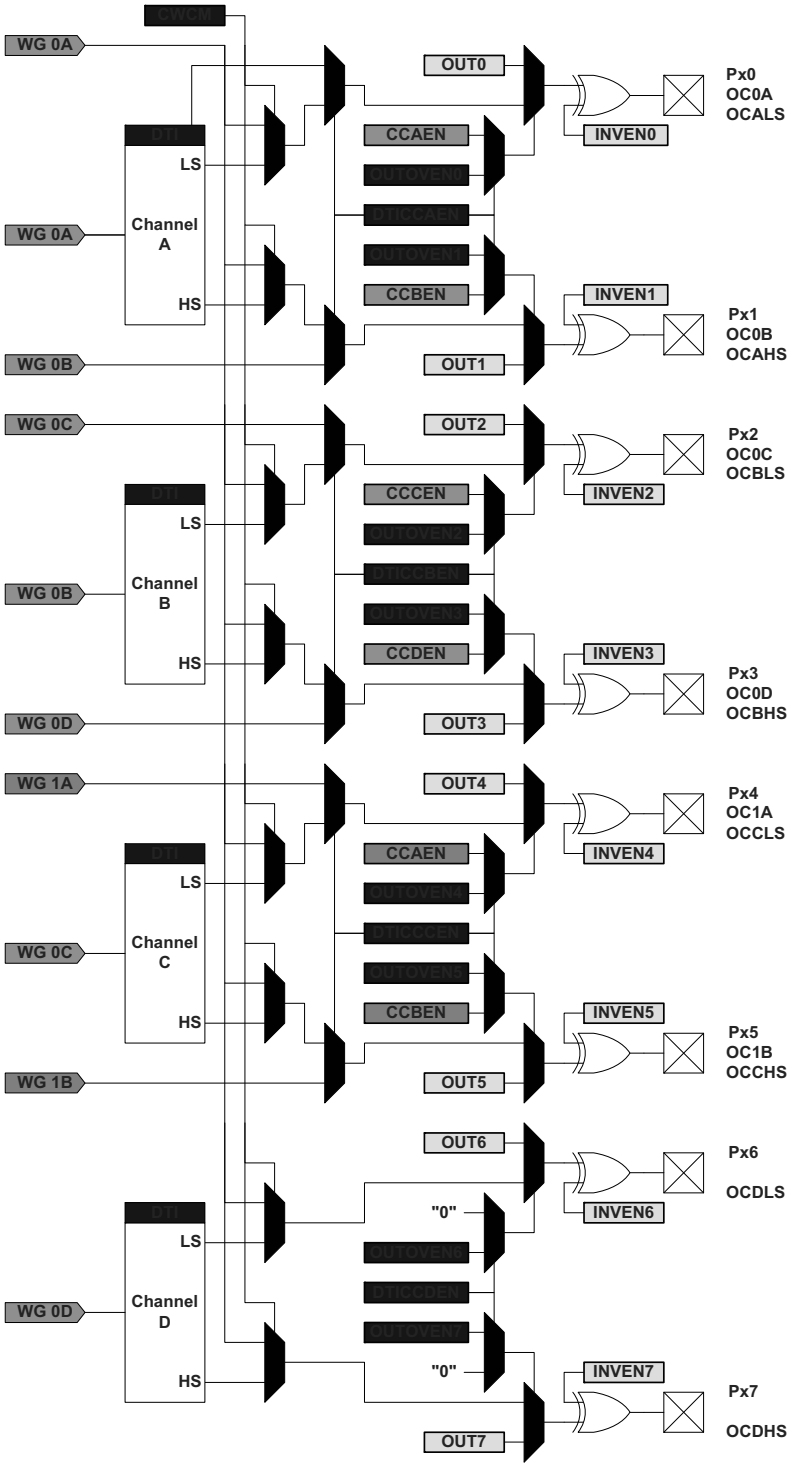
Figure 14-4. Changing the Period



14.6 Compare Channel

Each compare channel continuously compares the counter value with the CMPx register. If CNT equals CMPx, the comparator signals a match. For the low-byte timer/counter, the match will set the compare channel's interrupt flag at the next timer clock cycle, and the event and optional interrupt is generated. The high-byte timer/counter does not have compare interrupt/event.

Figure 15-2. Timer/counter Extensions and Port Override Logic



15.4 Dead-time Insertion

The dead-time insertion (DTI) unit generates OFF time where the non-inverted low side (LS) and inverted high side (HS) of the WG output are both low. This OFF time is called dead time, and dead-time insertion ensures that the LS and HS never switch simultaneously.

The DTI unit consists of four equal dead-time generators, one for each compare channel in timer/counter 0. Figure 15-3 on page 178 shows the block diagram of one DTI generator. The four channels have a common register that controls the

- **Bit 0 – ATTACH: Attach**

Setting this bit enables the internal D+ or D- pull-up (depending on the USB speed selection), and attaches the device to the USB lines. Clearing this bit disconnects the device from the USB lines.

18.13.3 STATUS – Status Register

Bit	7	6	5	4	3	2	1	0
+0x02	–	–	–	–	**URESUME**	**RESUME**	**SUSPEND**	**BUSRST**
Read/Write	R	R	R	R	R	R	R	R
Initial Value	0	0	0	0	0	0	0	0

- **Bit 7:4 – Reserved**

These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written.

- **Bit 3 – URESUME: Upstream Resume**

This flag is set when an upstream resume is sent.

- **Bit 2 – RESUME: Resume**

This flag is set when a downstream resume is received.

- **Bit 1 – SUSPEND: Bus Suspended**

This flag is set when the USB lines are in the suspended state (the bus has been idle for at least 3ms).

- **Bit 0 – BUSRST: Bus Reset**

This flag is set when a reset condition has been detected (the bus has been driven to SE0 for at least 2.5µs).

18.13.4 ADDR – Address Register

Bit	7	6	5	4	3	2	1	0
+0x03	–	**ADDR[6:0]**						
Read/Write	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Initial Value	0	0	0	0	0	0	0	0

- **Bit 7 – Reserved**

This bit is unused and reserved for future use. For compatibility with future devices, always write this bit to zero when this register is written.

- **Bit 6:0 – ADDR[6:0]: Device Address**

These bits contain the USB address the device will respond to.

18.13.5 FIFOWP – FIFO Write Pointer Register

Bit	7	6	5	4	3	2	1	0
+0x04	–	–	–	**FIFOWP[4:0]**				
Read/Write	R	R	R	R/W	R/W	R/W	R/W	R/W
Initial Value	0	0	0	0	0	0	0	0

- **Bit 7:5 – Reserved**

These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written.

- **Bit 4:0 – FIFOWP[4:0]: FIFO Write Pointer**

These bits contain the transaction complete FIFO write pointer. This register must be read only by the CPU or DMA controller. Writing this register will flush the FIFO write and read pointers.

19.6.1 Receiving Address Packets

When the TWI slave is properly configured, it will wait for a START condition to be detected. When this happens, the successive address byte will be received and checked by the address match logic, and the slave will ACK a correct address and store the address in the DATA register. If the received address is not a match, the slave will not acknowledge and store address, and will wait for a new START condition.

The slave address/stop interrupt flag is set when a START condition succeeded by a valid address byte is detected. A general call address will also set the interrupt flag.

A START condition immediately followed by a STOP condition is an illegal operation, and the bus error flag is set.

The R/$\overline{W}$ direction flag reflects the direction bit received with the address. This can be read by software to determine the type of operation currently in progress.

Depending on the R/$\overline{W}$ direction bit and bus condition, one of four distinct cases (S1 to S4) arises following the address packet. The different cases must be handled in software.

19.6.1.1 Case S1: Address packet accepted - Direction bit set

If the R/$\overline{W}$ direction flag is set, this indicates a master read operation. The SCL line is forced low by the slave, stretching the bus clock. If ACK is sent by the slave, the slave hardware will set the data interrupt flag indicating data is needed for transmit. Data, repeated START, or STOP can be received after this. If NACK is sent by the slave, the slave will wait for a new START condition and address match.

19.6.1.2 Case S2: Address packet accepted - Direction bit cleared

If the R/$\overline{W}$ direction flag is cleared, this indicates a master write operation. The SCL line is forced low, stretching the bus clock. If ACK is sent by the slave, the slave will wait for data to be received. Data, repeated START, or STOP can be received after this. If NACK is sent, the slave will wait for a new START condition and address match.

19.6.1.3 Case S3: Collision

If the slave is not able to send a high level or NACK, the collision flag is set, and it will disable the data and acknowledge output from the slave logic. The clock hold is released. A START or repeated START condition will be accepted.

19.6.1.4 Case S4: STOP condition received

When the STOP condition is received, the slave address/stop flag will be set, indicating that a STOP condition, and not an address match, occurred.

19.6.2 Receiving Data Packets

The slave will know when an address packet with R/$\overline{W}$ direction bit cleared has been successfully received. After acknowledging this, the slave must be ready to receive data. When a data packet is received, the data interrupt flag is set and the slave must indicate ACK or NACK. After indicating a NACK, the slave must expect a STOP or repeated START condition.

19.6.3 Transmitting Data Packets

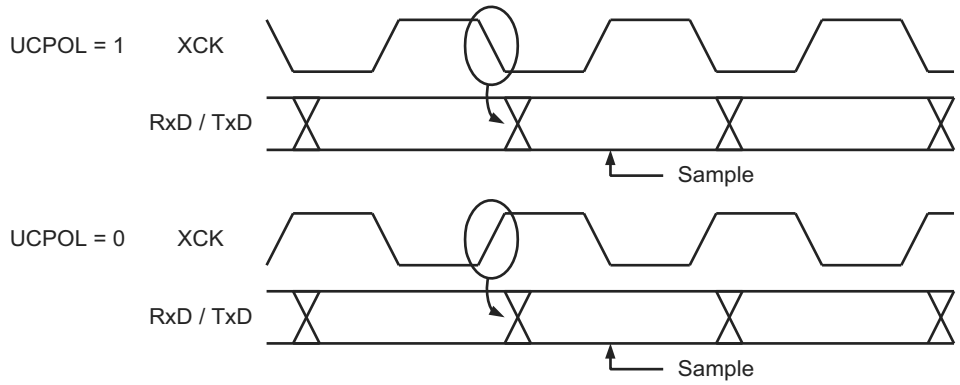
The slave will know when an address packet with R/$\overline{W}$ direction bit set has been successfully received. It can then start sending data by writing to the slave data register. When a data packet transmission is completed, the data interrupt flag is set. If the master indicates NACK, the slave must stop transmitting data and expect a STOP or repeated START condition.

19.7 Enabling External Driver Interface

An external driver interface can be enabled. When this is done, the internal TWI drivers with input filtering and slew rate control are bypassed. The normal I/O pin function is used, and the direction must be configured by the user software. When this mode is enabled, an external TWI compliant tri-state driver is needed for connecting to a TWI bus.

By default, port pins 0 (Pn0) and 1 (Pn1) are used for SDA and SCL. The external driver interface uses port pins 0 to 3 for the SDA_IN, SCL_IN, SDA_OUT, and SCL_OUT signals.

Figure 21-3. Synchronous Mode XCK Timing



Using the inverted I/O (INVEN) setting for the corresponding XCK port pin, the XCK clock edges used for data sampling and data change can be selected. If inverted I/O is disabled (INVEN=0), data will be changed at the rising XCK clock edge and sampled at the falling XCK clock edge. If inverted I/O is enabled (INVEN=1), data will be changed at the falling XCK clock edge and sampled at the rising XCK clock edge. For more details, see “I/O Ports” on page 119.

21.3.5 Master SPI Mode Clock Generation

For master SPI mode operation, only internal clock generation is supported. This is identical to the USART synchronous master mode, and the baud rate or BSEL setting is calculated using the same equations (see Table 21-1 on page 246).

There are four combinations of the SPI clock (SCK) phase and polarity with respect to the serial data, and these are determined by the clock phase (UCPHA) control bit and the inverted I/O pin (INVEN) settings. The data transfer timing diagrams are shown in Figure 21-4 on page 249. Data bits are shifted out and latched in on opposite edges of the XCK signal, ensuring sufficient time for data signals to stabilize. The UCPHA and INVEN settings are summarized in Table 21-2. Changing the setting of any of these bits during transmission will corrupt both the receiver and transmitter

Table 21-2. INVEN and UCPHA Functionality

SPI Mode	INVEN	UCPHA	Leading edge	Trailing edge
0	0	0	Rising, sample	Falling, setup
1	0	1	Rising, setup	Falling, sample
2	1	0	Falling, sample	Rising, setup
3	1	1	Falling, setup	Rising, sample

The leading edge is the first clock edge of a clock cycle. The trailing edge is the last clock edge of a clock cycle.

24. CRC – Cyclic Redundancy Check Generator

24.1 Features

- Cyclic redundancy check (CRC) generation and checking for
  - Communication data
  - Program or data in flash memory
  - Data in SRAM and I/O memory space
- Integrated with flash memory, DMA controller and CPU
  - Continuous CRC on data going through a DMA channel
  - Automatic CRC of the complete or a selectable range of the flash memory
  - CPU can load data to the CRC generator through the I/O interface
- CRC polynomial software selectable to
  - CRC-16 (CRC-CCITT)
  - CRC-32 (IEEE 802.3)
- Zero remainder detection

24.2 Overview

A cyclic redundancy check (CRC) is an error detection technique test algorithm used to find accidental errors in data, and it is commonly used to determine the correctness of a data transmission, and data present in the data and program memories. A CRC takes a data stream or a block of data as input and generates a 16- or 32-bit output that can be appended to the data and used as a checksum. When the same data are later received or read, the device or application repeats the calculation. If the new CRC result does not match the one calculated earlier, the block contains a data error. The application will then detect this and may take a corrective action, such as requesting the data to be sent again or simply not using the incorrect data.

Typically, an n-bit CRC applied to a data block of arbitrary length will detect any single error burst not longer than n bits (any single alteration that spans no more than n bits of the data), and will detect the fraction $1-2^{-n}$ of all longer error bursts. The CRC module in XMEGA devices supports two commonly used CRC polynomials; CRC-16 (CRC-CCITT) and CRC-32 (IEEE 802.3).

- **CRC-16:**

  Polynomial: $x^{16}+x^{12}+x^5+1$

  Hex value: 0x1021

- **CRC-32:**

  Polynomial: $x^{32}+x^{26}+x^{23}+x^{22}+x^{16}+x^{12}+x^{11}+x^{10}+x^8+x^7+x^5+x^4+x^2+x+1$

  Hex value: 0x04C11DB7

24.3 Operation

The data source for the CRC module must be selected in software as either flash memory, the DMA channels, or the I/O interface. The CRC module then takes data input from the selected source and generates a checksum based on these data. The checksum is available in the CHECKSUM registers in the CRC module. When CRC-32 polynomial is used, the final checksum read is bit reversed and complemented (see Figure 24-1 on page 275).

For the I/O interface or DMA controller, which CRC polynomial is used is software selectable, but the default setting is CRC-16. CRC-32 is automatically used if Flash Memory is selected as the source. The CRC module operates on bytes only.

25.15.10.1 12-bit Mode, Left Adjusted

- **Bit 7:0 – CHRES[11:4]: Channel Result High byte**

These are the eight msbs of the 12-bit ADC result.

25.15.10.2 12-bit Mode, Right Adjusted

- **Bit 7:4 – Reserved**

These bits will in practice be the extension of the sign bit, CHRES11, when the ADC works in differential mode, and set to zero when the ADC works in signed mode.

- **Bit 3:0 – CHRES[11:8]: Channel Result High byte**

These are the four msbs of the 12-bit ADC result.

25.15.10.3 8-bit Mode

- **Bit 7:0 – Reserved**

These bits will in practice be the extension of the sign bit, CHRES7, when the ADC works in signed mode, and set to zero when the ADC works in single-ended mode.

25.15.11 CH0RESL – Channel 0 Result Register Low

	Bit	7	6	5	4	3	2	1	0
12-/8-bit, right		CHRES[7:0]							
12-bit, left		CHRES[3:0]				–	–	–	–
	Read/Write	R	R	R	R	R	R	R	R
	Initial Value	0	0	0	0	0	0	0	0

25.15.11.1 12-/8-bit Mode

- **Bit 7:0 – CHRES[7:0]: Channel Result Low byte**

These are the eight lsbs of the ADC result.

25.15.11.2 12-bit Mode, Left Adjusted

- **Bit 7:4 – CHRES[3:0]: Channel Result Low byte**

These are the four lsbs of the 12-bit ADC result.

- **Bit 3:0 – Reserved**

These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written.

25.15.12 CMPH – Compare Register High

The CMPH and CMPL register pair represents the 16-bit value, CMP. For details on reading and writing 16-bit registers, refer to “Accessing 16-bit Registers” on page 13.

Bit	7	6	5	4	3	2	1	0
+0x19	CMP[15:0]							
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Initial Value	0	0	0	0	0	0	0	0

- **Bit 7:0 – CMP[15:0]: Compare Value High byte**

These are the eight msbs of the 16-bit ADC compare value. In signed mode, the number representation is 2's complement, and the msb is the sign bit.

25.17 Register Summary – ADC

This is the register summary when the ADC is configured to give standard 12-bit results. The register summaries for 8-bit and 12-bit left adjusted will be similar, but with some changes in the result registers, CH0RESH and CH0RESL.

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Page
+0x00	CTRLA	–	–	–	–	–	CH0STAR	FLUSH	ENABLE	290
+0x01	CTRLB	–	CURRLIMIT[1:0]		CONVMODE	FREERUN	RESOLUTION[1:0]		–	290
+0x02	REFCTRL	–	REFSEL[2:0]			–	–	BANDGAP	TEMPREF	291
+0x03	EVCTRL	–	–	–	EVSEL[1:0]		EVACT[2:0]			292
+0x04	PRESCALER	–	–	–	–	–	PRESCALER[2:0]			293
+0x05	Reserved	–	–	–	–	–	–	–	–	
+0x06	INTFLAGS	–	–	–	–	–	–	–	CH0IF	293
+0x07	TEMP	TEMP[7:0]								294
+0x08	SAMPCTRL	–	–	SAMPVAL[5:0]						294
+0x09	Reserved	–	–	–	–	–	–	–	–	
+0x0A	Reserved	–	–	–	–	–	–	–	–	
+0x0B	Reserved	–	–	–	–	–	–	–	–	
+0x0C	CALL	CAL[7:0]								294
+0x0D	Reserved	–	–	–	–	–	–	–	–	
+0x0E	Reserved	–	–	–	–	–	–	–	–	
+0x0F	Reserved	–	–	–	–	–	–	–	–	
+0x10	CH0RESL	CH0RES[7:0]								295
+0x11	CH0RESH	CH0RES[15:8]								294
+0x12	Reserved	–	–	–	–	–	–	–	–	
+0x13	Reserved	–	–	–	–	–	–	–	–	
+0x14	Reserved	–	–	–	–	–	–	–	–	
+0x15	Reserved	–	–	–	–	–	–	–	–	
+0x16	Reserved	–	–	–	–	–	–	–	–	
+0x17	Reserved	–	–	–	–	–	–	–	–	
+0x18	CMPL	CMP[7:0]								296
+0x19	CMPH	CMP[15:8]								295
+0x1A	Reserved	–	–	–	–	–	–	–	–	
+0x1B	Reserved	–	–	–	–	–	–	–	–	
+0x1C	Reserved	–	–	–	–	–	–	–	–	
+0x1D	Reserved	–	–	–	–	–	–	–	–	
+0x1E	Reserved	–	–	–	–	–	–	–	–	
+0x1F	Reserved	–	–	–	–	–	–	–	–	
+0x20	CH0 Offset	Offset address for ADC channel								
+0x28	Reserved	–	–	–	–	–	–	–	–	
+0x30	Reserved	–	–	–	–	–	–	–	–	
+0x38	Reserved	–	–	–	–	–	–	–	–	

25.18 Register Summary – ADC Channel

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Page
+0x00	CTRL	START	–	–	GAIN[2:0]			INPUTMODE[1:0]		296
+0x01	MUXCTRL	–	MUXPOS[3:0]				MUXNEG[2:0]			297
+0x02	INTCTRL	–	–	–	–	INTMODE[1:0]		INTLVL[1:0]		299
+0x03	INTFLAGS	–	–	–	–	–	–	–	IF	300
+0x04	RESL	RES[7:0]								301
+0x05	RESH	RES[15:8]								300
+0x06	SCAN	OFFSET				COUNT				301
+0x07	Reserved	–	–	–	–	–	–	–	–	

25.19 Interrupt Vector Summary

Offset	Source	Interrupt description
0x00	CH0	Analog-to-digital converter channel 0 interrupt vector