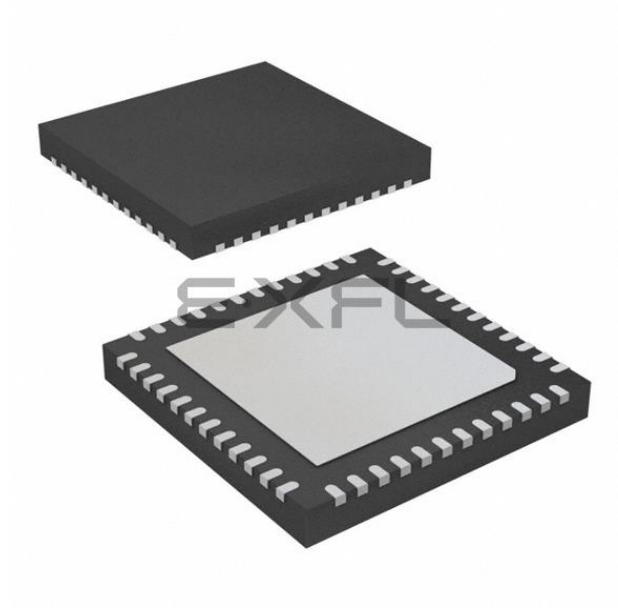


**Welcome to E-XFL.COM**

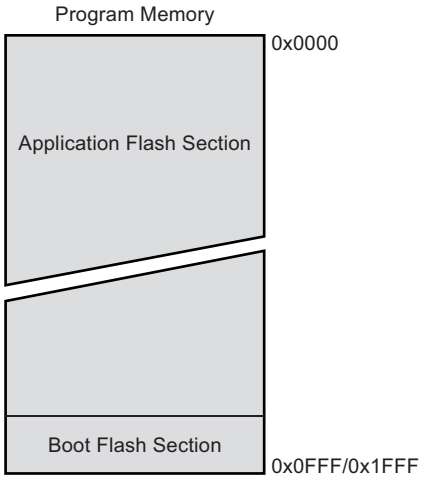
## What is "Embedded - Microcontrollers"?

"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 | Obsolete |
| Core Processor | AVR |
| Core Size | 8-Bit |
| Speed | 16MHz |
| Connectivity | I²C, LINbus, SPI, UART/USART |
| Peripherals | Brown-out Detect/Reset, POR, PWM, WDT |
| Number of I/O | 23 |
| Program Memory Size | 16KB (16K x 8) |
| Program Memory Type | FLASH |
| EEPROM Size | 512 x 8 |
| RAM Size | 1K x 8 |
| Voltage - Supply (Vcc/Vdd) | 2.7V ~ 5.5V |
| Data Converters | A/D 8x10b |
| Oscillator Type | Internal |
| Operating Temperature | -40°C ~ 125°C (TC) |
| Mounting Type | Surface Mount |
| Package / Case | 48-VFQFN Exposed Pad |
| Supplier Device Package | 48-QFN (7x7) |
| Purchase URL | https://www.e-xfl.com/product-detail/microchip-technology/ata6613c-plqw |

**Figure 5-8. Program Memory Map, Atmel ATA6612C/ATA6613C**




### 5.5.2 SRAM Data Memory

Figure 5-9 shows how the Atmel® ATA6612C/ATA6613C SRAM memory is organized.

The Atmel ATA6612C/ATA6613C is a complex microcontroller with more peripheral units than can be supported within the 64 locations reserved in the opcode for the IN and OUT instructions. For the extended I/O space from 0x60 - 0xFF in SRAM, only the ST/STS/STD and LD/LDS/LDD instructions can be used.

The lower 768/1280/1280 data memory locations address both the register file, the I/O memory, extended I/O memory, and the internal data SRAM. The first 32 locations address the register file, the next 64 location the standard I/O memory, then 160 locations of extended I/O memory, and the next 512/1024/1024 locations address the internal data SRAM.

The five different addressing modes for the data memory cover: Direct, indirect with displacement, indirect, indirect with pre-decrement, and indirect with post-increment. In the register file, registers R26 to R31 feature the indirect addressing pointer registers.

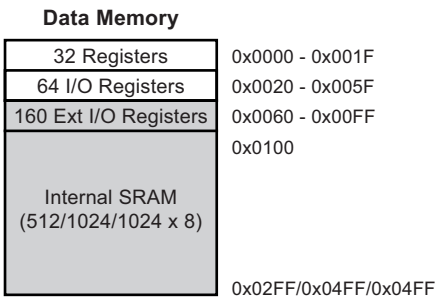
The direct addressing reaches the entire data space.

The indirect with displacement mode reaches 63 address locations from the base address given by the Y- or Z-register.

When using register indirect addressing modes with automatic pre-decrement and post-increment, the address registers X, Y, and Z are decremented or incremented.

The 32 general purpose working registers, 64 I/O registers, 160 extended I/O registers, and the 512/1024/1024 bytes of internal data SRAM in the Atmel ATA6612C/ATA6613C are all accessible through all these addressing modes. The register file is described in Section 5.4.5 “General Purpose Register File” on page 34.

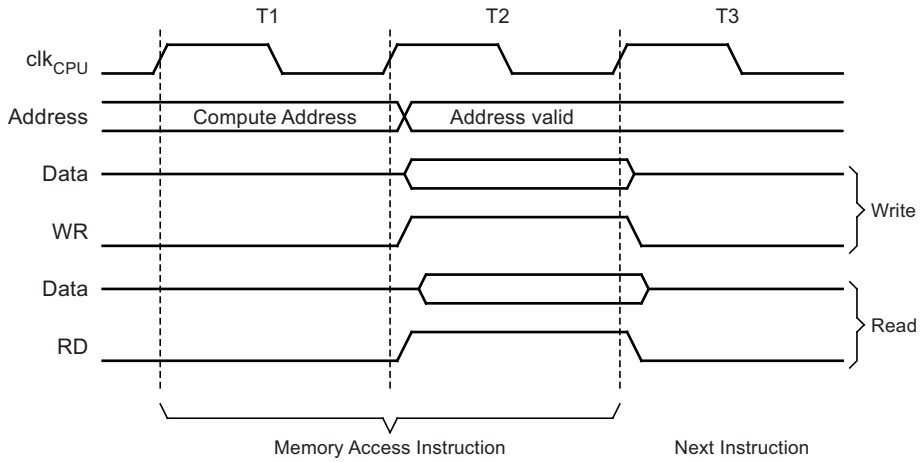
**Figure 5-9. Data Memory Map**



### 5.5.2.1 Data Memory Access Times

This section describes the general access timing concepts for internal memory access. The internal data SRAM access is performed in two $clk_{CPU}$ cycles as described in Figure 5-10.

**Figure 5-10. On-chip Data SRAM Access Cycles**




## 5.5.3 EEPROM Data Memory

The Atmel® ATA6612C/ATA6613C contains 512 bytes of data EEPROM memory. It is organized as a separate data space, in which single bytes can be read and written. The EEPROM has an endurance of at least 100,000 write/erase cycles. The access between the EEPROM and the CPU is described in the following, specifying the EEPROM address registers, the EEPROM data register, and the EEPROM control register.

The section Section 5.24 “Memory Programming” on page 253 contains a detailed description on EEPROM programming in SPI or parallel programming mode.

### 5.5.3.1 EEPROM Read/Write Access

The EEPROM access registers are accessible in the I/O space.

The write access time for the EEPROM is given in Table 5-3 on page 43. A self-timing function, however, lets the user software detect when the next byte can be written. If the user code contains instructions that write the EEPROM, some precautions must be taken. In heavily filtered power supplies, $V_{CC}$ is likely to rise or fall slowly on power-up/down. This causes the device for some period of time to run at a voltage lower than specified as minimum for the clock frequency used. See Section 5.5.3.5 “Preventing EEPROM Corruption” on page 44 for details on how to avoid problems in these situations.

In order to prevent unintentional EEPROM writes, a specific write procedure must be followed. Refer to the description of the EEPROM control register for details on this.

When the EEPROM is read, the CPU is halted for four clock cycles before the next instruction is executed. When the EEPROM is written, the CPU is halted for two clock cycles before the next instruction is executed.

### 5.6.5 Low Frequency Crystal Oscillator

The device can utilize a 32.768kHz watch crystal as clock source by a dedicated low frequency crystal oscillator. The crystal should be connected as shown in Figure 5-12. When this oscillator is selected, start-up times are determined by the SUT fuses and CKSEL0 as shown in Table 5-10.

**Table 5-10. Start-up Times for the Low Frequency Crystal Oscillator Clock Selection**

| Power Conditions | Start-up Time from Power-down and Power-save | Additional Delay from Reset ($V_{CC}$ = 5.0V) | CKSEL0 | SUT1..0 |
|---|---|---|---|---|
| BOD enabled | 1kCK | 14CK[1] | 0 | 00 |
| Fast rising power | 1kCK | 14CK + 4.1ms[1] | 0 | 01 |
| Slowly rising power | 1kCK | 14CK + 65ms[1] | 0 | 10 |
| Reserved | | | 0 | 11 |
| BOD enabled | 32kCK | 14CK | 1 | 00 |
| Fast rising power | 32kCK | 14CK + 4.1ms | 1 | 01 |
| Slowly rising power | 32kCK | 14CK + 65ms | 1 | 10 |
| Reserved | | | 1 | 11 |

Note: 1. These options should only be used if frequency stability at start-up is not important for the application.

### 5.6.6 Calibrated Internal RC Oscillator

The calibrated internal RC oscillator by default provides a 8.0MHz clock. The frequency is nominal value at 5V and 25°C. The device is shipped with the CKDIV8 fuse programmed. See Section 5.6.11 “System Clock Prescaler” on page 54 for more details. This clock may be selected as the system clock by programming the CKSEL fuses as shown in Table 5-11. If selected, it will operate with no external components. During reset, hardware loads the calibration byte into the OSCCAL register and thereby automatically calibrates the RC oscillator. At 5V and 25°C, this calibration gives a frequency of 8MHz ±1%. The tolerance of the internal RC oscillator remains better than ±10% within the whole automotive temperature and voltage ranges (4.5V to 5.5V, –40°C to +125°C). The oscillator can be calibrated to any frequency in the range 7.3 - 8.1MHz within ±1% accuracy, by changing the OSCCAL register. When this oscillator is used as the chip clock, the watchdog oscillator will still be used for the watchdog timer and for the reset time-out. For more information on the pre-programmed calibration value (see Section 5.24.4 “Calibration Byte” on page 256).

**Table 5-11. Internal Calibrated RC Oscillator Operating Modes[1][3]**

| Frequency Range[2] (MHz) | CKSEL3..0 |
|---|---|
| 7.3 - 8.1 | 0010 |

Notes:
1. The device is shipped with this option selected.
2. The frequency ranges are preliminary values. Actual values are TBD.
3. If 8MHz frequency exceeds the specification of the device (depends on $V_{CC}$), the CKDIV8 fuse can be programmed in order to divide the internal frequency by 8.

The following code example shows one assembly and one C function for changing the time-out value of the watchdog timer.

Assembly Code Example[1]

```
WDT_Prescaler_Change:
        ; Turn off global interrupt
        cli
        ; Reset Watchdog Timer
        wdr
        ; Start timed sequence
        lds r16, WDTCSR
        ori   r16, (1<<WDCE) | (1<<WDE)
        sts WDTCSR, r16
        ; -- Got four cycles to set the new values from here -
        ; Set new prescaler(time-out) value = 64K cycles (~0.5 s)
        ldi   r16, (1<<WDE) | (1<<WDP2) | (1<<WDP0)
        sts WDTCSR, r16
        ; -- Finished setting new values, used 2 cycles -
        ; Turn on global interrupt
        sei
        ret
```

C Code Example[1]

```
void WDT_Prescaler_Change(void)
{
        __disable_interrupt();
        __watchdog_reset();
        /* Start timed equence */
        WDTCSR |= (1<<WDCE) | (1<<WDE);
        /* Set new prescaler(time-out) value = 64K cycles (~0.5 s) */
        WDTCSR = (1<<WDE) | (1<<WDP2) | (1<<WDP0);
        __enable_interrupt();
}
```

Notes:
1. The example code assumes that the part specific header file is included.
2. The watchdog timer should be reset before any change of the WDP bits, since a change in the WDP bits can result in a time-out when switching to a shorter time-out period.

### 5.8.9.1 Watchdog Timer Control Register - WDTCSR

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|---|
| | **WDIF** | **WDIE** | **WDP3** | **WDCE** | **WDE** | **WDP2** | **WDP1** | **WDP0** | **WDTCSR** |
| Read/Write | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | |
| Initial Value | 0 | 0 | 0 | 0 | X | 0 | 0 | 0 | |

- **Bit 7 - WDIF: Watchdog Interrupt Flag**

This bit is set when a time-out occurs in the watchdog timer and the watchdog timer is configured for interrupt. WDIF is cleared by hardware when executing the corresponding interrupt handling vector. Alternatively, WDIF is cleared by writing a logic one to the flag. When the I-bit in SREG and WDIE are set, the watchdog time-out interrupt is executed.

- **Bit 6 - WDIE: Watchdog Interrupt Enable**

When this bit is written to one and the I-bit in the status register is set, the watchdog interrupt is enabled. If WDE is cleared in combination with this setting, the watchdog timer is in interrupt mode, and the corresponding interrupt is executed if time-out in the watchdog timer occurs.

If WDE is set, the watchdog timer is in interrupt and system reset mode. The first time-out in the watchdog timer will set WDIF. Executing the corresponding interrupt vector will clear WDIE and WDIF automatically by hardware (the watchdog goes to system reset mode). This is useful for keeping the watchdog timer security while using the interrupt. To stay in interrupt and system reset mode, WDIE must be set after each interrupt. This should however not be done within the interrupt service routine itself, as this might compromise the safety-function of the watchdog system reset mode. If the interrupt is not executed before the next time-out, a system reset will be applied.

### 5.10.2.2 Toggling the Pin

Writing a logic one to PINxn toggles the value of PORTxn, independent on the value of DDRxn. Note that the SBI instruction can be used to toggle one single bit in a port.

### 5.10.2.3 Switching Between Input and Output

When switching between tri-state ({DDxn, PORTxn} = 0b00) and output high ({DDxn, PORTxn} = 0b11), an intermediate state with either pull-up enabled {DDxn, PORTxn} = 0b01) or output low ({DDxn, PORTxn} = 0b10) must occur. Normally, the pull-up enabled state is fully acceptable, as a high-impedant environment will not notice the difference between a strong high driver and a pull-up. If this is not the case, the PUD bit in the MCUCR register can be set to disable all pull-ups in all ports.

Switching between input with pull-up and output low generates the same problem. The user must use either the tri-state ({DDxn, PORTxn} = 0b00) or the output high state ({DDxn, PORTxn} = 0b11) as an intermediate step.

Table 5-30 summarizes the control signals for the pin value.

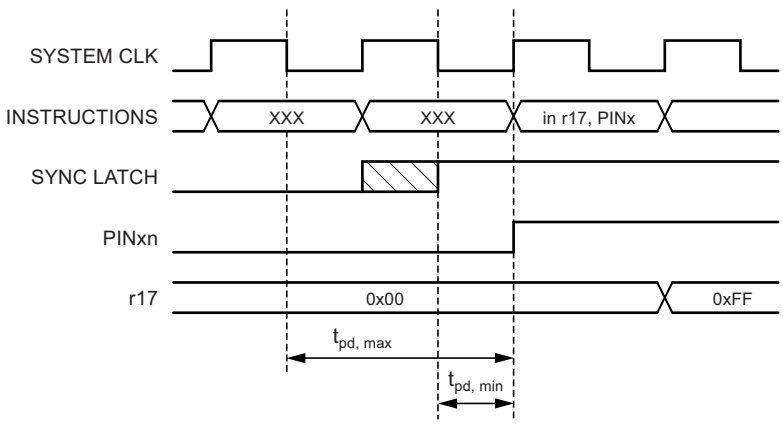
**Table 5-30. Port Pin Configurations**

| DDxn | PORTxn | PUD (in MCUCR) | I/O | Pull-up | Comment |
|---|---|---|---|---|---|
| 0 | 0 | X | Input | No | Tri-state (hi-Z) |
| 0 | 1 | 0 | Input | Yes | Pxn will source current if ext. pulled low. |
| 0 | 1 | 1 | Input | No | Tri-state (hi-Z) |
| 1 | 0 | X | Output | No | Output low (sink) |
| 1 | 1 | X | Output | No | Output high (source) |

### 5.10.2.4 Reading the Pin Value

Independent of the setting of data direction bit DDxn, the port pin can be read through the PINxn register bit. As shown in Figure 5-23 on page 78, the PINxn register bit and the preceding latch constitute a synchronizer. This is needed to avoid metastability if the physical pin changes value near the edge of the internal clock, but it also introduces a delay. Figure 5-24 shows a timing diagram of the synchronization when reading an externally applied pin value. The maximum and minimum propagation delays are denoted $t_{pd,max}$ and $t_{pd,min}$ respectively.

**Figure 5-24. Synchronization when Reading an Externally Applied Pin Value**




Consider the clock period starting shortly after the first falling edge of the system clock. The latch is closed when the clock is low, and goes transparent when the clock is high, as indicated by the shaded region of the “SYNC LATCH” signal. The signal value is latched when the system clock goes low. It is clocked into the PINxn register at the succeeding positive clock edge. As indicated by the two arrows tpd,max and tpd,min, a single signal transition on the pin will be delayed between ½ and 1½ system clock period depending upon the time of assertion.

The following code examples show how to do an atomic read of the TCNT1 register contents. Reading any of the OCR1A/B or ICR1 registers can be done by using the same principle.

Assembly Code Example[1]

```
TIM16_ReadTCNT1:
        ; Save global interrupt flag
        in      r18,SREG
        ; Disable interrupts
        cli
        ; Read TCNT1 into r17:r16
        in      r16,TCNT1L
        in      r17,TCNT1H
        ; Restore global interrupt flag
        out     SREG,r18
        ret
```

C Code Example[1]

```
unsigned int TIM16_ReadTCNT1( void )
{
        unsigned char sreg;
        unsigned int i;
        /* Save global interrupt flag */
        sreg = SREG;
        /* Disable interrupts */
        _CLI();
        /* Read TCNT1 into i */
        i = TCNT1;
        /* Restore global interrupt flag */
        SREG = sreg;
        return i;
}
```

Note: 1. The example code assumes that the part specific header file is included.
For I/O registers located in extended I/O map, “IN”, “OUT”, “SBIS”, “SBIC”, “CBI”, and “SBI” instructions must be replaced with instructions that allow access to extended I/O. Typically “LDS” and “STS” combined with “SBRS”, “SBRC”, “SBR”, and “CBR”.

The assembly code example returns the TCNT1 value in the r17:r16 register pair.

A frequency (with 50% duty cycle) waveform output in fast PWM mode can be achieved by setting OC1A to toggle its logical level on each compare match (COM1A1:0 = 1). This applies only if OCR1A is used to define the TOP value (WGM13:0 = 15). The waveform generated will have a maximum frequency of $f_{OC1A} = f_{clk\_I/O}/2$ when OCR1A is set to zero (0x0000). This feature is similar to the OC1A toggle in CTC mode, except the double buffer feature of the output compare unit is enabled in the fast PWM mode.

### 5.14.8.4 Phase Correct PWM Mode

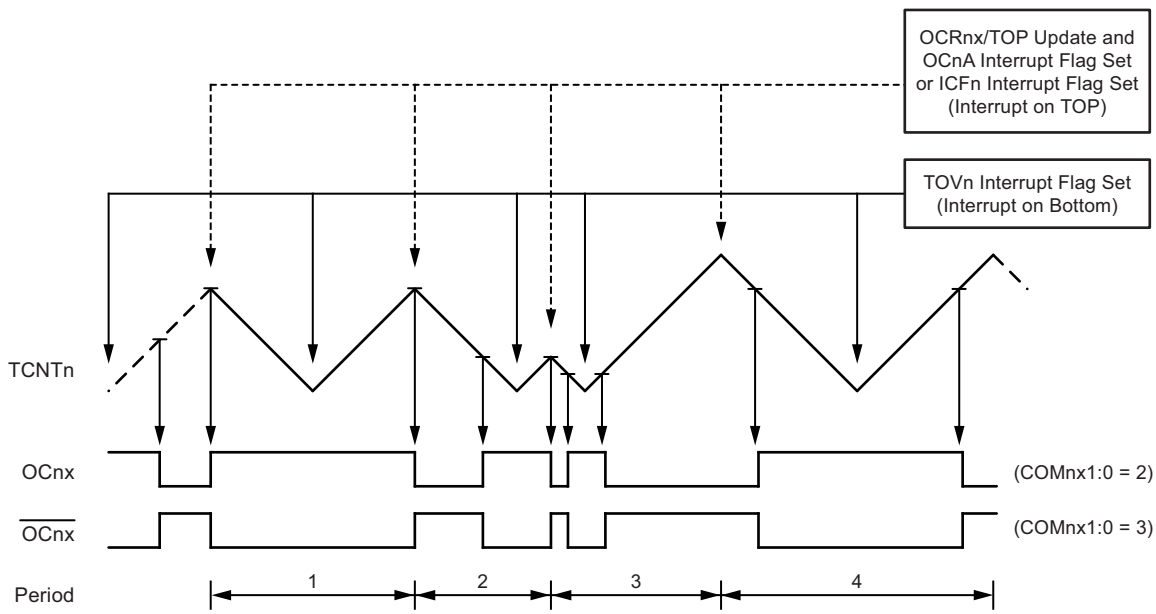
The phase correct pulse width modulation or phase correct PWM mode (WGM13:0 = 1, 2, 3, 10, or 11) provides a high resolution phase correct PWM waveform generation option. The phase correct PWM mode is, like the phase and frequency correct PWM mode, based on a dual-slope operation. The counter counts repeatedly from BOTTOM (0x0000) to TOP and then from TOP to BOTTOM. In non-inverting compare output mode, the output compare (OC1x) is cleared on the compare match between TCNT1 and OCR1x while upcounting, and set on the compare match while downcounting. In inverting output compare mode, the operation is inverted. The dual-slope operation has lower maximum operation frequency than single slope operation. However, due to the symmetric feature of the dual-slope PWM modes, these modes are preferred for motor control applications.

The PWM resolution for the phase correct PWM mode can be fixed to 8-, 9-, or 10-bit, or defined by either ICR1 or OCR1A. The minimum resolution allowed is 2-bit (ICR1 or OCR1A set to 0x0003), and the maximum resolution is 16-bit (ICR1 or OCR1A set to MAX). The PWM resolution in bits can be calculated by using the following equation:

$$R_{PCPWM} = \frac{\log(TOP + 1)}{\log(2)}$$

In phase correct PWM mode the counter is incremented until the counter value matches either one of the fixed values 0x00FF, 0x01FF, or 0x03FF (WGM13:0 = 1, 2, or 3), the value in ICR1 (WGM13:0 = 10), or the value in OCR1A (WGM13:0 = 11). The counter has then reached the TOP and changes the count direction. The TCNT1 value will be equal to TOP for one timer clock cycle. The timing diagram for the phase correct PWM mode is shown on Figure 5-47. The figure shows phase correct PWM mode when OCR1A or ICR1 is used to define TOP. The TCNT1 value is in the timing diagram shown as a histogram for illustrating the dual-slope operation. The diagram includes non-inverted and inverted PWM outputs. The small horizontal line marks on the TCNT1 slopes represent compare matches between OCR1x and TCNT1. The OC1x interrupt flag will be set when a compare match occurs.

**Figure 5-47. Phase Correct PWM Mode, Timing Diagram**



### 5.14.10 16-bit Timer/Counter Register Description

#### 5.14.10.1 Timer/Counter1 Control Register A – TCCR1A

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|---|
| | COM1A1 | COM1A0 | COM1B1 | COM1B0 | – | – | WGM11 | WGM10 | TCCR1A |
| Read/Write | R/W | R/W | R/W | R/W | R | R | R/W | R/W | |
| Initial Value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | |

- **Bit 7:6 – COM1A1:0: Compare Output Mode for Channel A**

- **Bit 5:4 – COM1B1:0: Compare Output Mode for Channel B**

The COM1A1:0 and COM1B1:0 control the output compare pins (OC1A and OC1B respectively) behavior. If one or both of the COM1A1:0 bits are written to one, the OC1A output overrides the normal port functionality of the I/O pin it is connected to. If one or both of the COM1B1:0 bit are written to one, the OC1B output overrides the normal port functionality of the I/O pin it is connected to. However, note that the *data direction register* (DDR) bit corresponding to the OC1A or OC1B pin must be set in order to enable the output driver.

When the OC1A or OC1B is connected to the pin, the function of the COM1x1:0 bits is dependent of the WGM13:0 bits setting. Table 5-53 shows the COM1x1:0 bit functionality when the WGM13:0 bits are set to a Normal or a CTC mode (non-PWM).

**Table 5-53. Compare Output Mode, non-PWM**

| COM1A1/COM1B1 | COM1A0/COM1B0 | Description |
|---|---|---|
| 0 | 0 | Normal port operation, OC1A/OC1B disconnected. |
| 0 | 1 | Toggle OC1A/OC1B on compare match. |
| 1 | 0 | Clear OC1A/OC1B on compare match (set output to low level). |
| 1 | 1 | Set OC1A/OC1B on compare match (set output to high level). |

Table 5-54 shows the COM1x1:0 bit functionality when the WGM13:0 bits are set to the fast PWM mode.

**Table 5-54. Compare Output Mode, Fast PWM(1)**

| COM1A1/COM1B1 | COM1A0/COM1B0 | Description |
|---|---|---|
| 0 | 0 | Normal port operation, OC1A/OC1B disconnected. |
| 0 | 1 | WGM13:0 = 14 or 15: Toggle OC1A on compare match, OC1B disconnected (normal port operation). For all other WGM1 settings, normal port operation, OC1A/OC1B disconnected. |
| 1 | 0 | Clear OC1A/OC1B on compare match, set OC1A/OC1B at TOP |
| 1 | 1 | Set OC1A/OC1B on compare match, clear OC1A/OC1B at TOP |

Note: 1. A special case occurs when OCR1A/OCR1B equals TOP and COM1A1/COM1B1 is set. In this case the compare match is ignored, but the set or clear is done at TOP. See Section 5.14.8.3 “Fast PWM Mode” on page 125 for more details.

Table 5-55 shows the COM1x1:0 bit functionality when the WGM13:0 bits are set to the phase correct or the phase and frequency correct, PWM mode.

**Table 5-55. Compare Output Mode, Phase Correct and Phase and Frequency Correct PWM(1)**

| COM1A1/COM1B1 | COM1A0/COM1B0 | Description |
|---|---|---|
| 0 | 0 | Normal port operation, OC1A/OC1B disconnected. |
| 0 | 1 | WGM13:0 = 8, 9, 10 or 11: Toggle OC1A on compare match, OC1B disconnected (normal port operation). For all other WGM1 settings, normal port operation, OC1A/OC1B disconnected. |
| 1 | 0 | Clear OC1A/OC1B on compare match when up-counting. Set OC1A/OC1B on compare match when downcounting. |
| 1 | 1 | Set OC1A/OC1B on compare match when up-counting. Clear OC1A/OC1B on compare match when downcounting. |

Note: 1. A special case occurs when OCR1A/OCR1B equals TOP and COM1A1/COM1B1 is set. See Section 5.14.8.4 “Phase Correct PWM Mode” on page 127 for more details.

- **Bit 1:0 – WGM11:0: Waveform Generation Mode**

Combined with the WGM13:2 bits found in the TCCR1B register, these bits control the counting sequence of the counter, the source for maximum (TOP) counter value, and what type of waveform generation to be used (see Table 5-56 on page 133). Modes of operation supported by the Timer/Counter unit are: Normal mode (counter), clear timer on compare match (CTC) mode, and three types of pulse width modulation (PWM) modes (see Section 5.14.8 “Modes of Operation” on page 124).

**Table 5-56. Waveform Generation Mode Bit Description(1)**

| Mode | WGM13 | WGM12 (CTC1) | WGM11 (PWM11) | WGM10 (PWM10) | Timer/Counter Mode of Operation | TOP | Update of OCR1x at | TOV1 Flag Set on |
|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | Normal | 0xFFFF | Immediate | MAX |
| 1 | 0 | 0 | 0 | 1 | PWM, phase correct, 8-bit | 0x00FF | TOP | BOTTOM |
| 2 | 0 | 0 | 1 | 0 | PWM, phase correct, 9-bit | 0x01FF | TOP | BOTTOM |
| 3 | 0 | 0 | 1 | 1 | PWM, phase correct, 10-bit | 0x03FF | TOP | BOTTOM |
| 4 | 0 | 1 | 0 | 0 | CTC | OCR1A | Immediate | MAX |
| 5 | 0 | 1 | 0 | 1 | Fast PWM, 8-bit | 0x00FF | TOP | TOP |
| 6 | 0 | 1 | 1 | 0 | Fast PWM, 9-bit | 0x01FF | TOP | TOP |
| 7 | 0 | 1 | 1 | 1 | Fast PWM, 10-bit | 0x03FF | TOP | TOP |
| 8 | 1 | 0 | 0 | 0 | PWM, phase and frequency correct | ICR1 | BOTTOM | BOTTOM |
| 9 | 1 | 0 | 0 | 1 | PWM, phase and frequency correct | OCR1A | BOTTOM | BOTTOM |
| 10 | 1 | 0 | 1 | 0 | PWM, phase correct | ICR1 | TOP | BOTTOM |
| 11 | 1 | 0 | 1 | 1 | PWM, phase correct | OCR1A | TOP | BOTTOM |
| 12 | 1 | 1 | 0 | 0 | CTC | ICR1 | Immediate | MAX |
| 13 | 1 | 1 | 0 | 1 | (Reserved) | – | – | – |
| 14 | 1 | 1 | 1 | 0 | Fast PWM | ICR1 | TOP | TOP |
| 15 | 1 | 1 | 1 | 1 | Fast PWM | OCR1A | TOP | TOP |

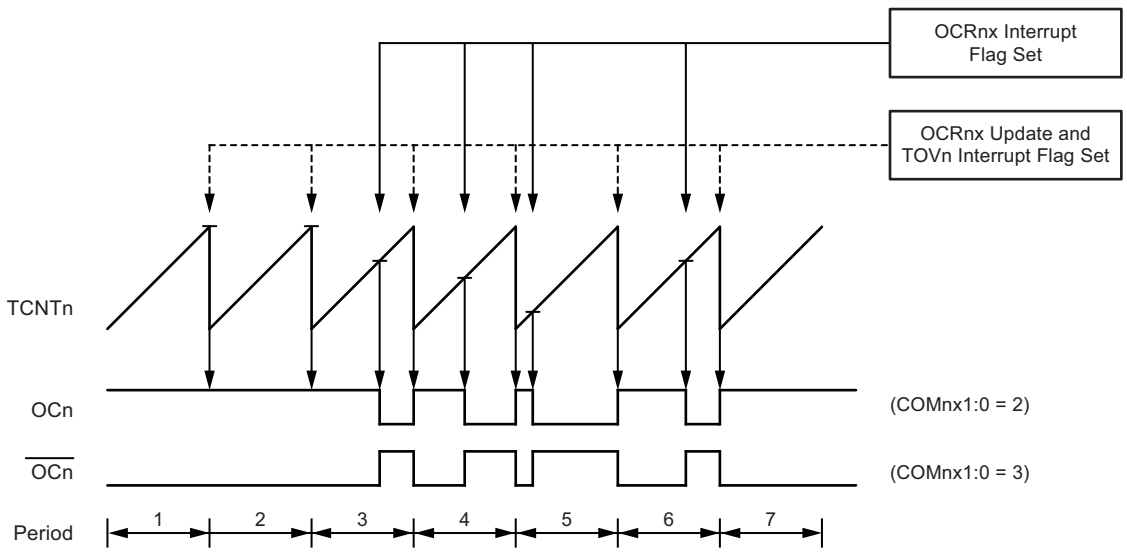
Note: 1. The CTC1 and PWM11:0 bit definition names are obsolete. Use the WGM12:0 definitions. However, the functionality and location of these bits are compatible with previous versions of the timer.

### 5.15.6.3 Fast PWM Mode

The fast pulse width modulation or fast PWM mode (WGM22:0 = 3 or 7) provides a high frequency PWM waveform generation option. The fast PWM differs from the other PWM option by its single-slope operation. The counter counts from BOTTOM to TOP then restarts from BOTTOM. TOP is defined as 0xFF when WGM2:0 = 3, and OCR2A when MGM2:0 = 7. In non-inverting compare output mode, the output compare (OC2x) is cleared on the compare match between TCNT2 and OCR2x, and set at BOTTOM. In inverting compare output mode, the output is set on compare match and cleared at BOTTOM. Due to the single-slope operation, the operating frequency of the fast PWM mode can be twice as high as the phase correct PWM mode that uses dual-slope operation. This high frequency makes the fast PWM mode well suited for power regulation, rectification, and DAC applications. High frequency allows physically small sized external components (coils, capacitors), and therefore reduces total system cost.

In fast PWM mode, the counter is incremented until the counter value matches the TOP value. The counter is then cleared at the following timer clock cycle. The timing diagram for the fast PWM mode is shown in Figure 5-58 on page 144. The TCNT2 value is in the timing diagram shown as a histogram for illustrating the single-slope operation. The diagram includes non-inverted and inverted PWM outputs. The small horizontal line marks on the TCNT2 slopes represent compare matches between OCR2x and TCNT2.

**Figure 5-58. Fast PWM Mode, Timing Diagram**




The Timer/Counter overflow flag (TOV2) is set each time the counter reaches TOP. If the interrupt is enabled, the interrupt handler routine can be used for updating the compare value.

In fast PWM mode, the compare unit allows generation of PWM waveforms on the OC2x pin. Setting the COM2x1:0 bits to two will produce a non-inverted PWM and an inverted PWM output can be generated by setting the COM2x1:0 to three. TOP is defined as 0xFF when WGM2:0 = 3, and OCR2A when MGM2:0 = 7 (see Table 5-60 on page 148). The actual OC2x value will only be visible on the port pin if the data direction for the port pin is set as output. The PWM waveform is generated by setting (or clearing) the OC2x register at the compare match between OCR2x and TCNT2, and clearing (or setting) the OC2x register at the timer clock cycle the counter is cleared (changes from TOP to BOTTOM).

The PWM frequency for the output can be calculated by the following equation:

$$f_{OCnxPWM} = \frac{f_{clk\_I/O}}{N \times 256}$$

The *N* variable represents the prescale factor (1, 8, 32, 64, 128, 256, or 1024).

The extreme values for the OCR2A register represent special cases when generating a PWM waveform output in the fast PWM mode. If the OCR2A is set equal to BOTTOM, the output will be a narrow spike for each MAX+1 timer clock cycle. Setting the OCR2A equal to MAX will result in a constantly high or low output (depending on the polarity of the output set by the COM2A1:0 bits.)

- **Bits 5:4 – COM2B1:0: Compare Match Output B Mode**

These bits control the output compare pin (OC2B) behavior. If one or both of the COM2B1:0 bits are set, the OC2B output overrides the normal port functionality of the I/O pin it is connected to. However, note that the data direction register (DDR) bit corresponding to the OC2B pin must be set in order to enable the output driver.

When OC2B is connected to the pin, the function of the COM2B1:0 bits depends on the WGM22:0 bit setting. Table 5-62 shows the COM2B1:0 bit functionality when the WGM22:0 bits are set to a normal or CTC mode (non-PWM).

**Table 5-62. Compare Output Mode, non-PWM Mode**

| COM2B1 | COM2B0 | Description |
|---|---|---|
| 0 | 0 | Normal port operation, OC2B disconnected. |
| 0 | 1 | Toggle OC2B on compare match |
| 1 | 0 | Clear OC2B on compare match |
| 1 | 1 | Set OC2B on compare match |

Table 5-63 shows the COM2B1:0 bit functionality when the WGM22:0 bits are set to fast PWM mode.

**Table 5-63. Compare Output Mode, Fast PWM Mode(1)**

| COM2B1 | COM2B0 | Description |
|---|---|---|
| 0 | 0 | Normal port operation, OC2B disconnected. |
| 0 | 1 | Reserved |
| 1 | 0 | Clear OC2B on compare match, set OC2B at TOP |
| 1 | 1 | Set OC2B on compare match, clear OC2B at TOP |

Note: 1. A special case occurs when OCR2B equals TOP and COM2B1 is set. In this case, the compare match is ignored, but the set or clear is done at TOP. See Section 5.15.6.4 “Phase Correct PWM Mode” on page 145 for more details.

Table 5-64 shows the COM2B1:0 bit functionality when the WGM22:0 bits are set to phase correct PWM mode.

**Table 5-64. Compare Output Mode, Phase Correct PWM Mode(1)**

| COM2B1 | COM2B0 | Description |
|---|---|---|
| 0 | 0 | Normal port operation, OC2B disconnected. |
| 0 | 1 | Reserved |
| 1 | 0 | Clear OC2B on compare match when up-counting. Set OC2B on compare match when down-counting. |
| 1 | 1 | Set OC2B on compare match when up-counting. Clear OC2B on compare match when down-counting. |

Note: 1. A special case occurs when OCR2B equals TOP and COM2B1 is set. In this case, the compare match is ignored, but the set or clear is done at TOP. See Section 5.15.6.4 “Phase Correct PWM Mode” on page 145 for more details.

- **Bits 3, 2 – Res: Reserved Bits**

These bits are reserved bits in the Atmel® ATA6612C/ATA6613C and will always read as zero.

- **Bit 0 – SPI2X: Double SPI Speed Bit**

When this bit is written logic one the SPI speed (SCK frequency) will be doubled when the SPI is in master mode (see Table 5-70 on page 162). This means that the minimum SCK period will be two CPU clock periods. When the SPI is configured as slave, the SPI is only guaranteed to work at fosc/4 or lower.

The SPI interface on the Atmel® ATA6612C/ATA6613C is also used for program memory and EEPROM downloading or uploading. See Section 5.24.8 “Serial Downloading” on page 267 for serial programming and verification.

### 5.16.1.5 SPI Data Register – SPDR

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|---|
| | MSB | | | | | | | LSB | SPDR |
| Read/Write | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | |
| Initial Value | X | X | X | X | X | X | X | X | Undefined |

The SPI data register is a read/write register used for data transfer between the register file and the SPI shift register. Writing to the register initiates data transmission. Reading the register causes the shift register receive buffer to be read.

## 5.16.2 Data Modes

There are four combinations of SCK phase and polarity with respect to serial data, which are determined by control bits CPHA and CPOL. The SPI data transfer formats are shown in Figure 5-67 and Figure 5-68 on page 164. Data bits are shifted out and latched in on opposite edges of the SCK signal, ensuring sufficient time for data signals to stabilize. This is clearly seen by summarizing Table 5-68 on page 161 and Table 5-69 on page 162, as done below.

**Table 5-71. CPOL Functionality**

| | Leading Edge | Trailing eDge | SPI Mode |
|---|---|---|---|
| CPOL = 0, CPHA = 0 | Sample (rising) | Setup (falling) | 0 |
| CPOL = 0, CPHA = 1 | Setup (rising) | Sample (falling) | 1 |
| CPOL = 1, CPHA = 0 | Sample (falling) | Setup (rising) | 2 |
| CPOL = 1, CPHA = 1 | Setup (falling) | Sample (rising) | 3 |

**Figure 5-67. SPI Transfer Format with CPHA = 0**

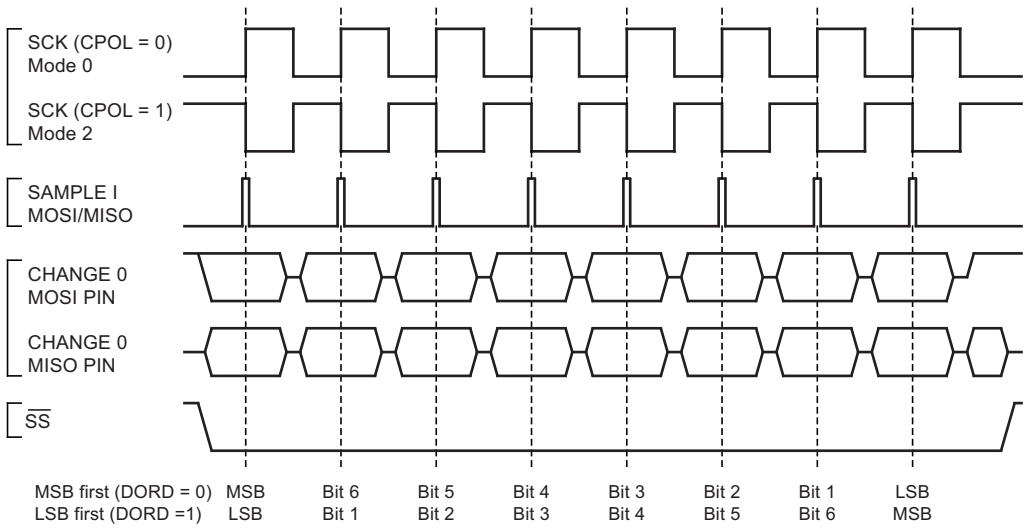
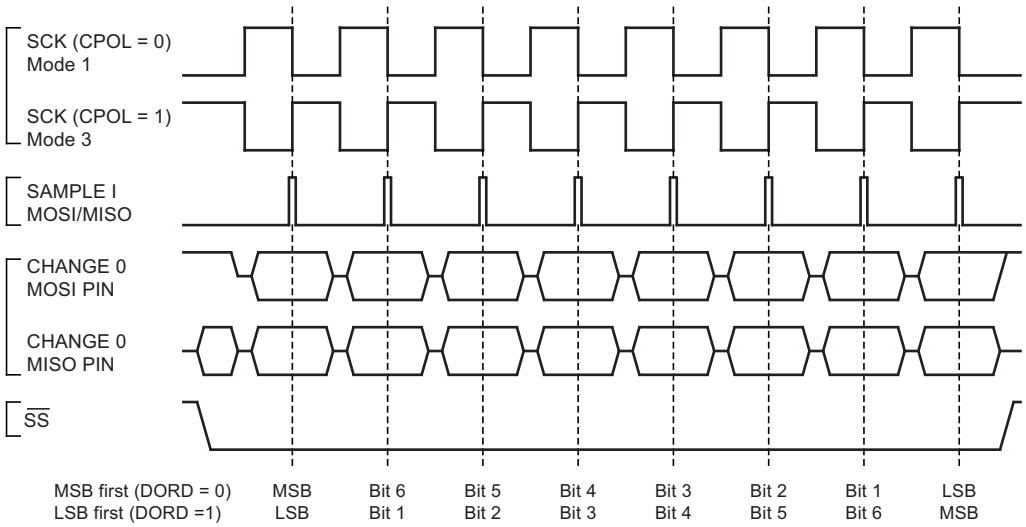


**Figure 5-68. SPI Transfer Format with CPHA = 1**




## 5.17 USART0

The universal synchronous and asynchronous serial receiver and transmitter (USART) is a highly flexible serial communication device. The main features are:

- Full duplex operation (independent serial receive and transmit registers)
- Asynchronous or synchronous operation
- Master or slave clocked synchronous operation
- High resolution baud rate generator
- Supports serial frames with 5, 6, 7, 8, or 9 data bits and 1 or 2 stop bits
- Odd or even parity generation and parity check supported by hardware
- Data overrun detection
- Framing error detection
- Noise filtering includes false start bit detection and digital low pass filter
- Three separate interrupts on TX complete, TX data register empty and RX complete
- Multi-processor communication mode
- Double speed asynchronous communication mode

The USART can also be used in master SPI mode (see Section 5.18 “USART in SPI Mode” on page 186. The power reduction USART bit, PRUSART0, in Section 5.7.7.1 “Power Reduction Register - PRR” on page 58 must be disabled by writing a logical zero to it.

Table 5-72 contains equations for calculating the baud rate (in bits per second) and for calculating the UBRRn value for each mode of operation using an internally generated clock source.

**Table 5-72. Equations for Calculating Baud Rate Register Setting**

| Operating Mode | Equation for Calculating Baud Rate[1] | Equation for Calculating UBRRn Value |
|---|---|---|
| Asynchronous normal mode (U2Xn = 0) | $BAUD = \frac{f_{OSC}}{16(UBRRn + 1)}$ | $UBRRn = \frac{f_{OSC}}{16BAUD} - 1$ |
| Asynchronous double speed mode (U2Xn = 1) | $BAUD = \frac{f_{OSC}}{8(UBRRn + 1)}$ | $UBRRn = \frac{f_{OSC}}{8BAUD} - 1$ |
| Synchronous master mode | $BAUD = \frac{f_{OSC}}{2(UBRRn + 1)}$ | $UBRRn = \frac{f_{OSC}}{2BAUD} - 1$ |

Note: 1. The baud rate is defined to be the transfer rate in bit per second (bps)

| | |
|---|---|
| **BAUD** | Baud rate (in bits per second, bps) |
| **$f_{osc}$** | System Oscillator clock frequency |
| **UBRRn** | Contents of the UBRRnH and UBRRnL Registers, (0-4095) |

Some examples of UBRRn values for some system clock frequencies are found in Table 5-80 on page 184 (see Section 5-80 “Examples of UBRRn Settings for Commonly Used Oscillator Frequencies” on page 184).

### 5.17.2.2 Double Speed Operation (U2Xn)

The transfer rate can be doubled by setting the U2Xn bit in UCSRnA. Setting this bit only has effect for the asynchronous operation. Set this bit to zero when using synchronous operation.Setting this bit will reduce the divisor of the baud rate divider from 16 to 8, effectively doubling the transfer rate for asynchronous communication.

Note however that the receiver will in this case only use half the number of samples (reduced from 16 to 8) for data sampling and clock recovery, and therefore a more accurate baud rate setting and system clock are required when this mode is used. For the transmitter, there are no downsides.

### 5.17.2.3 External Clock

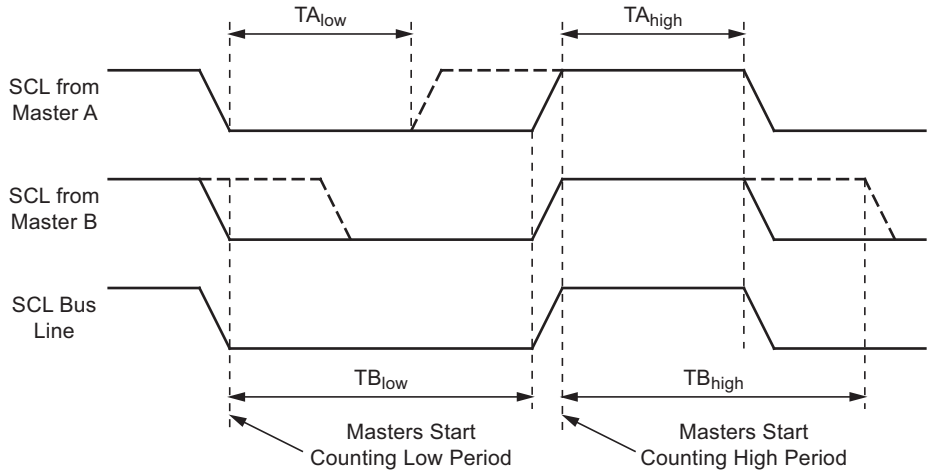
External clocking is used by the synchronous slave modes of operation. The description in this section refers to Figure 5-70 on page 166 for details.

External clock input from the XCKn pin is sampled by a synchronization register to minimize the chance of meta-stability. The output from the synchronization register must then pass through an edge detector before it can be used by the transmitter and receiver. This process introduces a two CPU clock period delay and therefore the maximum external XCKn clock frequency is limited by the following equation:

$$f_{XCK} < \frac{f_{OSC}}{4}$$

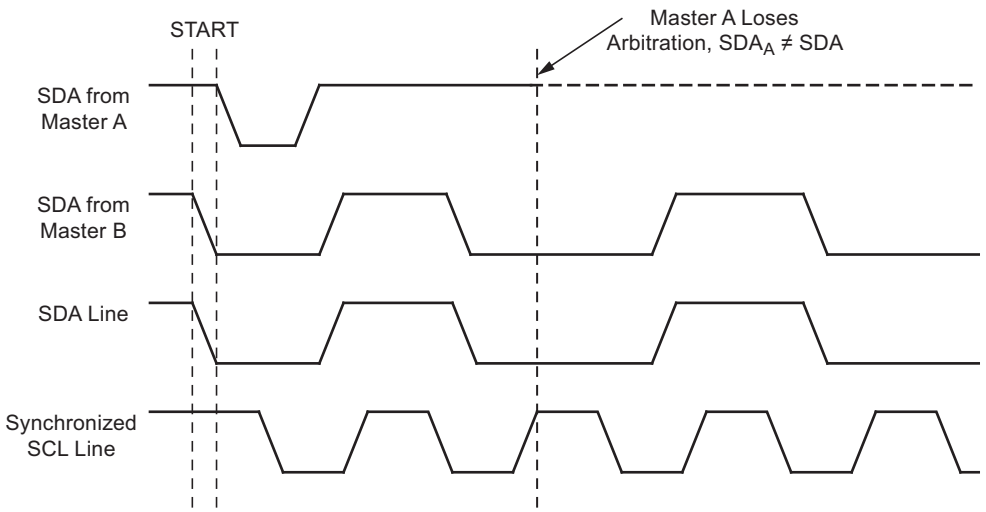
Note that $f_{osc}$ depends on the stability of the system clock source. It is therefore recommended to add some margin to avoid possible loss of data due to frequency variations.

**Figure 5-83. SCL Synchronization Between Multiple Masters**




Arbitration is carried out by all masters continuously monitoring the SDA line after outputting data. If the value read from the SDA line does not match the value the master had output, it has lost the arbitration. Note that a master can only lose arbitration when it outputs a high SDA value while another master outputs a low value. The losing master should immediately go to slave mode, checking if it is being addressed by the winning master. The SDA line should be left high, but losing masters are allowed to generate a clock signal until the end of the current data or address packet. Arbitration will continue until only one master remains, and this may take many bits. If several masters are trying to address the same slave, arbitration will continue into the data packet.

**Figure 5-84. Arbitration Between Two Masters**




Note that arbitration is not allowed between:

- A REPEATED START condition and a data bit.
- A STOP condition and a data bit.
- A REPEATED START and a STOP condition.

It is the user software's responsibility to ensure that these illegal arbitration conditions never occur. This implies that in multi-master systems, all data transfers must use the same composition of SLA+R/W and data packets. In other words: All transmissions must contain the same number of data packets, otherwise the result of the arbitration is undefined.

### 5.21.6.4 ADC Control and Status Register B – ADCSRB

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|---|
| | – | ACME | – | – | – | ADTS2 | ADTS1 | ADTS0 | ADCSRB |
| Read/Write | R | R/W | R | R | R | R/W | R/W | R/W | |
| Initial Value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | |

- **Bit 7, 5:3 – Res: Reserved Bits**

These bits are reserved for future use. To ensure compatibility with future devices, these bits must be written to zero when ADCSRB is written.

- **Bit 2:0 – ADTS2:0: ADC Auto Trigger Source**

If ADATE in ADCSRA is written to one, the value of these bits selects which source will trigger an ADC conversion. If ADATE is cleared, the ADTS2:0 settings will have no effect. A conversion will be triggered by the rising edge of the selected interrupt flag. Note that switching from a trigger source that is cleared to a trigger source that is set, will generate a positive edge on the trigger signal. If ADEN in ADCSRA is set, this will start a conversion. Switching to free running mode (ADTS[2:0]=0) will not cause a trigger event, even if the ADC interrupt flag is set.

**Table 5-102. ADC Auto Trigger Source Selections**

| ADTS2 | ADTS1 | ADTS0 | Trigger Source |
|---|---|---|---|
| 0 | 0 | 0 | Free running mode |
| 0 | 0 | 1 | Analog comparator |
| 0 | 1 | 0 | External interrupt request 0 |
| 0 | 1 | 1 | Timer/Counter0 compare match A |
| 1 | 0 | 0 | Timer/Counter0 overflow |
| 1 | 0 | 1 | Timer/Counter1 compare match B |
| 1 | 1 | 0 | Timer/Counter1 overflow |
| 1 | 1 | 1 | Timer/Counter1 capture event |

### 5.21.6.5 Digital Input Disable Register 0 – DIDR0

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|---|
| | – | – | ADC5D | ADC4D | ADC3D | ADC2D | ADC1D | ADC0D | DIDR0 |
| Read/Write | R | R | R/W | R/W | R/W | R/W | R/W | R/W | |
| Initial Value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | |

- **Bits 7:6 – Res: Reserved Bits**

These bits are reserved for future use. To ensure compatibility with future devices, these bits must be written to zero when DIDR0 is written.

- **Bit 5..0 – ADC5D..ADC0D: ADC5..0 Digital Input Disable**

When this bit is written logic one, the digital input buffer on the corresponding ADC pin is disabled. The corresponding PIN register bit will always read as zero when this bit is set. When an analog signal is applied to the ADC5..0 pin and the digital input from this pin is not needed, this bit should be written logic one to reduce power consumption in the digital input buffer.

Note that ADC pins ADC7 and ADC6 do not have digital input buffers, and therefore do not require digital input disable bits.

**Table 5-118. Fuse High Byte**

| High Fuse Byte | Bit No | Description | Default Value |
|---|---|---|---|
| RSTDISBL[1] | 7 | External reset disable | 1 (unprogrammed) |
| DWEN | 6 | debugWIRE enable | 1 (unprogrammed) |
| SPIEN[2] | 5 | Enable serial program and data downloading | 0 (programmed, SPI programming enabled) |
| WDTON[3] | 4 | Watchdog timer always on | 1 (unprogrammed) |
| EESAVE | 3 | EEPROM memory is preserved through the chip erase | 1 (unprogrammed), EEPROM not reserved |
| BODLEVEL2[4] | 2 | Brown-out detector trigger level | 1 (unprogrammed) |
| BODLEVEL1[4] | 1 | Brown-out detector trigger level | 1 (unprogrammed) |
| BODLEVEL0[4] | 0 | Brown-out detector trigger level | 1 (unprogrammed) |

Notes:
1. See Section 5.10.3.3 “Alternate Functions of Port C” on page 87 for description of RSTDISBL fuse.
2. The SPIEN fuse is not accessible in serial programming mode.
3. See Section 5.8.9.1 “Watchdog Timer Control Register - WDTCSR” on page 68 for details.
4. See Table 5-21 on page 63 for BODLEVEL fuse decoding.

**Table 5-119. Fuse Low Byte**

| Low Fuse Byte | Bit No | Description | Default Value |
|---|---|---|---|
| CKDIV8[4] | 7 | Divide clock by 8 | 0 (programmed) |
| CKOUT[3] | 6 | Clock output | 1 (unprogrammed) |
| SUT1 | 5 | Select start-up time | 1 (unprogrammed)[1] |
| SUT0 | 4 | Select start-up time | 0 (programmed)[1] |
| CKSEL3 | 3 | Select clock source | 0 (programmed)[2] |
| CKSEL2 | 2 | Select clock source | 0 (programmed)[2] |
| CKSEL1 | 1 | Select clock source | 1 (unprogrammed)[2] |
| CKSEL0 | 0 | Select clock source | 0 (programmed)[2] |

Notes:
1. The default value of SUT1..0 results in maximum start-up time for the default clock source. See Table 5-12 on page 52 for details.
2. The default setting of CKSEL3..0 results in internal RC Oscillator at 8 MHz. See Table 5-11 on page 51 for details.
3. The CKOUT fuse allows the system clock to be output on PORTB0. See Section 5.6.9 “Clock Output Buffer” on page 54 for details.
4. See Section 5.6.11 “System Clock Prescaler” on page 54 for details.

The status of the fuse bits is not affected by chip erase. Note that the fuse bits are locked if lock bit1 (LB1) is programmed. program the fuse bits before programming the lock bits.

#### 5.24.2.1 Latching of Fuses

The fuse values are latched when the device enters programming mode and changes of the fuse values will have no effect until the part leaves Programming mode. This does not apply to the EESAVE fuse which will take effect once it is programmed. The fuses are also latched on power-up in normal mode.

### 5.24.3 Signature Bytes

All Atmel® microcontrollers have a three-byte signature code which identifies the device. This code can be read in both serial and parallel mode, also when the device is locked. The three bytes reside in a separate address space.

**Figure 5-127. Serial Programming Waveforms**

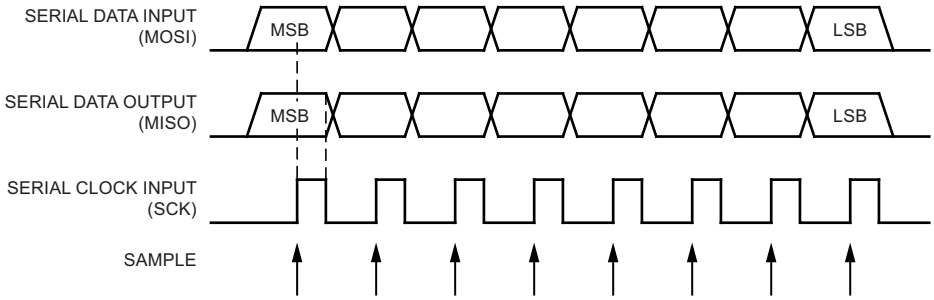



**Table 5-129. Serial Programming Instruction Set**

| Instruction | Instruction Format | | | | Operation |
|---|---|---|---|---|---|
| | Byte 1 | Byte 2 | Byte 3 | Byte4 | |
| Programming enable | 1010 1100 | 0101 0011 | xxxx xxxx | xxxx xxxx | Enable serial programming after RESET goes low. |
| Chip erase | 1010 1100 | 100x xxxx | xxxx xxxx | xxxx xxxx | Chip erase EEPROM and flash. |
| Read program memory | 0010 **H**000 | 000**a aaaa** | **bbbb bbbb** | **oooo oooo** | Read **H** (high or low) data **o** from Program memory at word address **a**:**b**. |
| Load program memory page | 0100 **H**000 | 000x xxxx | xx**bb bbbb** | **iiii iiii** | Write **H** (high or low) data **i** to program memory page at word address **b**. Data low byte must be loaded before data high byte is applied within the same address. |
| Write program memory page | 0100 1100 | 000**a aaaa** | **bb**xx xxxx | xxxx xxxx | Write program memory page at address **a**:**b**. |
| Read EEPROM memory | 1010 0000 | 000x xx**aa** | **bbbb bbbb** | **oooo oooo** | Read data **o** from EEPROM memory at address **a**:**b**. |
| Write EEPROM memory | 1100 0000 | 000x xx**aa** | **bbbb bbbb** | **iiii iiii** | Write data **i** to EEPROM memory at address **a**:**b**. |
| Load EEPROM memory Page (page access) | 1100 0001 | 0000 0000 | 0000 00**bb** | **iiii iiii** | Load data **i** to EEPROM memory page buffer. After data is loaded, program EEPROM page. |
| Write EEPROM memory Page (page access) | 1100 0010 | 00xx xx**aa** | **bbbb bb00** | xxxx xxxx | Write EEPROM page at address **a**:**b**. |
| Read lock bits | 0101 1000 | 0000 0000 | xxxx xxxx | xx**oo oooo** | Read Lock bits. “0” = programmed, “1” = unprogrammed. See Table 5-114 on page 253 for details. |
| Write lock bits | 1010 1100 | 111x xxxx | xxxx xxxx | 11**ii iiii** | Write lock bits. Set bits = “0” to program lock bits. See Table 5-114 on page 253 for details. |
| Read signature byte | 0011 0000 | 000x xxxx | xxxx xx**bb** | **oooo oooo** | Read signature byte **o** at address **b**. |
| Write fuse bits | 1010 1100 | 1010 0000 | xxxx xxxx | **iiii iiii** | Set bits = “0” to program, “1” to unprogram. See **Table XXX on page XXX** for details. |
| Write fuse high bits | 1010 1100 | 1010 1000 | xxxx xxxx | **iiii iiii** | Set bits = “0” to program, “1” to unprogram. See Table 5-98 on page 229 for details. |

Note: **a** = address high bits, **b** = address low bits, **H** = 0 - Low byte, 1 - High Byte, **o** = data out, **i** = data in, **x** = do not care

## 6.5 Instruction Set Summary

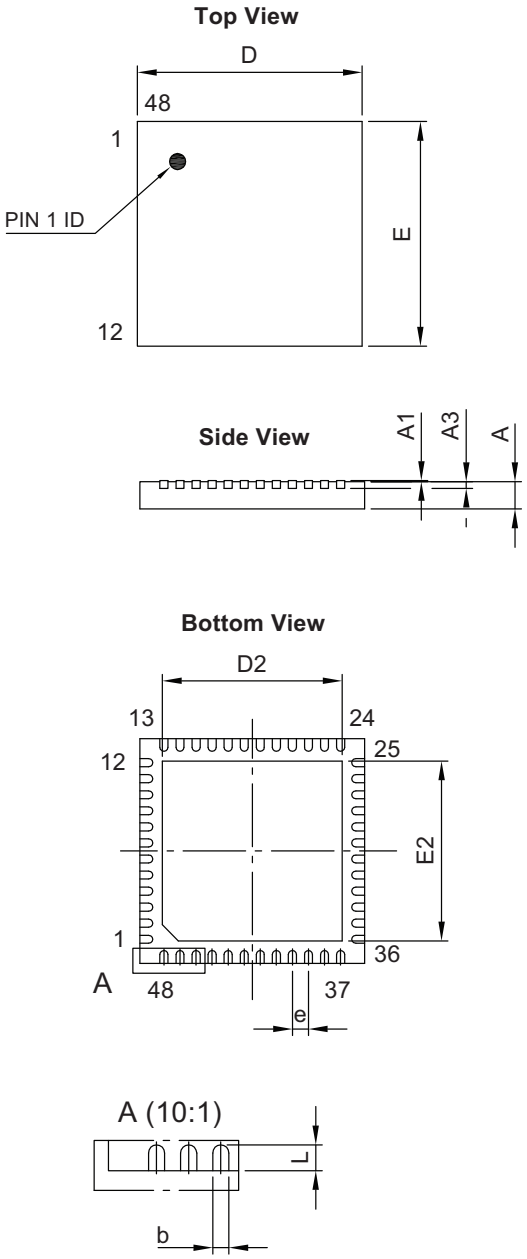
| Mnemonics | Operands | Description | Operation | Flags | #Clocks |
|---|---|---|---|---|---|
| Arithmetic and Logic Instructions | | | | | |
| ADD | Rd, Rr | Add two registers | Rd ← Rd + Rr | Z,C,N,V,H | 1 |
| ADC | Rd, Rr | Add with carry two registers | Rd ← Rd + Rr + C | Z,C,N,V,H | 1 |
| ADIW | Rdl,K | Add immediate to word | Rdh:Rdl ← Rdh:Rdl + K | Z,C,N,V,S | 2 |
| SUB | Rd, Rr | Subtract two registers | Rd ← Rd – Rr | Z,C,N,V,H | 1 |
| SUBI | Rd, K | Subtract constant from register | Rd ← Rd – K | Z,C,N,V,H | 1 |
| SBC | Rd, Rr | Subtract with carry two registers | Rd ← Rd – Rr – C | Z,C,N,V,H | 1 |
| SBCI | Rd, K | Subtract with carry constant from reg. | Rd ← Rd – K – C | Z,C,N,V,H | 1 |
| SBIW | Rdl,K | Subtract immediate from word | Rdh:Rdl ← Rdh:Rdl – K | Z,C,N,V,S | 2 |
| AND | Rd, Rr | Logical AND registers | Rd ← Rd × Rr | Z,N,V | 1 |
| ANDI | Rd, K | Logical AND register and constant | Rd ← Rd × K | Z,N,V | 1 |
| OR | Rd, Rr | Logical OR registers | Rd ← Rd v Rr | Z,N,V | 1 |
| ORI | Rd, K | Logical OR register and constant | Rd ← Rd v K | Z,N,V | 1 |
| EOR | Rd, Rr | Exclusive OR registers | Rd ← Rd ⊕ Rr | Z,N,V | 1 |
| COM | Rd | One’s complement | Rd ← 0xFF – Rd | Z,C,N,V | 1 |
| NEG | Rd | Two’s complement | Rd ← 0x00 – Rd | Z,C,N,V,H | 1 |
| SBR | Rd,K | Set bit(s) in register | Rd ← Rd v K | Z,N,V | 1 |
| CBR | Rd,K | Clear bit(s) in register | Rd ← Rd × (0xFF – K) | Z,N,V | 1 |
| INC | Rd | Increment | Rd ← Rd + 1 | Z,N,V | 1 |
| DEC | Rd | Decrement | Rd ← Rd – 1 | Z,N,V | 1 |
| TST | Rd | Test for zero or minus | Rd ← Rd × Rd | Z,N,V | 1 |
| CLR | Rd | Clear register | Rd ← Rd ⊕ Rd | Z,N,V | 1 |
| SER | Rd | Set register | Rd ← 0xFF | None | 1 |
| MUL | Rd, Rr | Multiply unsigned | R1:R0 ← Rd × Rr | Z,C | 2 |
| MULS | Rd, Rr | Multiply signed | R1:R0 ← Rd × Rr | Z,C | 2 |
| MULSU | Rd, Rr | Multiply signed with unsigned | R1:R0 ← Rd × Rr | Z,C | 2 |
| FMUL | Rd, Rr | Fractional multiply unsigned | R1:R0 ← (Rd × Rr) << 1 | Z,C | 2 |
| FMULS | Rd, Rr | Fractional multiply signed | R1:R0 ← (Rd × Rr) << 1 | Z,C | 2 |
| FMULSU | Rd, Rr | Fractional multiply signed with unsigned | R1:R0 ← (Rd × Rr) << 1 | Z,C | 2 |
| Branch Instructions | | | | | |
| RJMP | k | Relative jump | PC ← PC + k + 1 | None | 2 |
| IJMP | | Indirect jump to (Z) | PC ← Z | None | 2 |
| JMP[1] | k | Direct jump | PC ← k | None | 3 |
| RCALL | k | Relative subroutine call | PC ← PC + k + 1 | None | 3 |
| ICALL | | Indirect call to (Z) | PC ← Z | None | 3 |
| CALL[1] | k | Direct subroutine call | PC ← k | None | 4 |
| RET | | Subroutine return | PC ← STACK | None | 4 |
| RETI | | Interrupt return | PC ← STACK | I | 4 |
| CPSE | Rd,Rr | Compare, skip if equal | if (Rd = Rr) PC ← PC + 2 or 3 | None | 1/2/3 |
| CP | Rd,Rr | Compare | Rd – Rr | Z, N,V,C,H | 1 |
| CPC | Rd,Rr | Compare with carry | Rd – Rr – C | Z, N,V,C,H | 1 |

Note: 1. These instructions are only available in ATA6613C

## 8. Ordering Information

| Extended Type Number | Program Memory | Package | MOQ |
|---|---|---|---|
| ATA6612C-PLQW-1 | 8kB flash | QFN48, 7×7 | 4,000 pieces |
| ATA6613C-PLQW-1 | 16kB flash | QFN48, 7×7 | 4,000 pieces |

## 9. Package Information




| COMMON DIMENSIONS (Unit of Measure = mm) | | | | |
|---|---|---|---|---|
| Symbol | MIN | NOM | MAX | NOTE |
| A | 0.8 | 0.85 | 0.9 | |
| A1 | 0 | 0.035 | 0.05 | |
| A3 | 0.16 | 0.21 | 0.26 | |
| D | 6.9 | 7 | 7.1 | |
| D2 | 5.5 | 5.6 | 5.7 | |
| E | 6.9 | 7 | 7.1 | |
| E2 | 5.5 | 5.6 | 5.7 | |
| L | 0.35 | 0.4 | 0.45 | |
| b | 0.2 | 0.25 | 0.3 | |
| e | | 0.5 | | |

05/20/14

| Atmel | TITLE | GPC | DRAWING NO. | REV. |
|---|---|---|---|---|
| **Package Drawing Contact:** packagedrawings@atmel.com | **Package: QFN_7x7_48L** Exposed pad 5.6x5.6 | | 6.543-5188.03-4 | 1 |