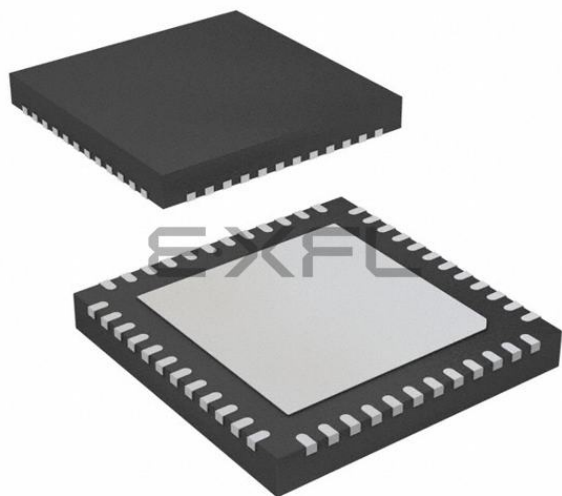




Welcome to [E-XFL.COM](https://www.e-xfl.com)

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	I ² C, 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-VQFN Exposed Pad
Supplier Device Package	48-QFN (7x7)
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/ata6613p-plqw

5. Electrical Characteristics (Continued)

5V < V_S < 27V, -40°C < T_{case} < 125°C, -40°C < T_j < 150°C, unless otherwise specified. All values refer to GND pins

No.	Parameters	Test Conditions	Pin	Symbol	Min.	Typ.	Max.	Unit	Type*
16.4	High-level leakage current	V _S = 27V V _{WAKE} = 27V	WAKE	I _{WAKEL}	-5		+5	μA	A
16.5	Time of low pulse for wake-up via WAKE pin	V _{WAKE} = 0V	WAKE	I _{WAKEL}	30	70	150	μs	A
18	VCC Voltage Regulator, PVCC = VCC								
18.1	Output voltage VCC	5.5V < V _S < 18V (0mA to 50mA)	VCC	VCC _{nor}	4.9		5.1	V	A
18.2	Output voltage VCC at low V _S	4V < V _S < 5.5V	VCC	VCC _{low}	V _S - V _D		5.1	V	A
18.3	Regulator drop voltage	V _S > 4V I _{VCC} = -20mA	V _S , VCC	V _{D1}			250	mV	A
18.4	Regulator drop voltage	V _S > 4V I _{VCC} = -50mA	V _S , VCC	V _{D2}		400	600	mV	A
18.5	Regulator drop voltage	V _S > 3.3V I _{VCC} = -15mA	V _S , VCC	V _{D3}			200	mV	A
18.6	Line regulation	5.5V < V _S < 18V	VCC	VCC _{line}			1	%	A
18.7	Load regulation	5mA < I _{VCC} < 50mA	VCC	VCC _{load}		0.5	2	%	A
18.8	Power supply ripple rejection	10Hz to 100kHz C _{VCC} = 10μF V _S = 14V, I _{VCC} = -15mA	VCC		50			dB	A
18.9	Output current limitation	V _S > 5.5V	VCC	I _{VCClim}	-240	-130		mA	A
18.10	Load capacity	0.2Ω < ESR < 5Ω at 100kHz	VCC	V _{thunN}	1.8	10		μF	D
18.11	VCC undervoltage threshold	Referred to VCC V _S > 5.5V	VCC	V _{thunN}	4.2		4.8	V	A
18.12	Hysteresis of undervoltage threshold	Referred to VCC V _S > 5.5V	VCC	V _{hys} _{thun}		250		mV	A
18.13	Ramp-up time V _S > 5.5V to V _{CC} = 5V	C _{VCC} = 2.2μF I _{load} = -5mA at VCC	VCC	t _{VCC}		130	300	μs	A

*) Type means: A = 100% tested, B = 100% correlation tested, C = Characterized on samples, D = Design parameter

Note that the Status Register is not automatically stored when entering an interrupt routine, nor restored when returning from an interrupt routine. This must be handled by software.

When using the CLI instruction to disable interrupts, the interrupts will be immediately disabled. No interrupt will be executed after the CLI instruction, even if it occurs simultaneously with the CLI instruction. The following example shows how this can be used to avoid interrupts during the timed EEPROM write sequence.

Assembly Code Example
<pre> in r16, SREG ; store SREG value cli ; disable interrupts during timed sequence sbi EECR, EEMPE ; start EEPROM write sbi EECR, EEPE out SREG, r16 ; restore SREG value (I-bit) </pre>
C Code Example
<pre> char cSREG; cSREG = SREG; /* store SREG value */ /* disable interrupts during timed sequence */ _cli(); EECR = (1<<EEMPE); /* start EEPROM write */ EECR = (1<<EEPE); SREG = cSREG; /* restore SREG value (I-bit) */ </pre>

When using the SEI instruction to enable interrupts, the instruction following SEI will be executed before any pending interrupts, as shown in this example.

Assembly Code Example
<pre> sei ; set Global Interrupt Enable sleep; enter sleep, waiting for interrupt ; note: will enter sleep before any pending interrupt(s) </pre>
C Code Example
<pre> __enable_interrupt(); /* set Global Interrupt Enable */ __sleep(); /* enter sleep, waiting for interrupt */ /* note: will enter sleep before any pending interrupt(s) */ </pre>

6.4.8.1 Interrupt Response Time

The interrupt execution response for all the enabled AVR® interrupts is four clock cycles minimum. After four clock cycles the program vector address for the actual interrupt handling routine is executed. During this four clock cycle period, the Program Counter is pushed onto the Stack. The vector is normally a jump to the interrupt routine, and this jump takes three clock cycles. If an interrupt occurs during execution of a multi-cycle instruction, this instruction is completed before the interrupt is served. If an interrupt occurs when the MCU is in sleep mode, the interrupt execution response time is increased by four clock cycles. This increase comes in addition to the start-up time from the selected sleep mode.

A return from an interrupt handling routine takes four clock cycles. During these four clock cycles, the Program Counter (two bytes) is popped back from the Stack, the Stack Pointer is incremented by two, and the I-bit in SREG is set.

6.7.8.6 Port Pins

When entering a sleep mode, all port pins should be configured to use minimum power. The most important is then to ensure that no pins drive resistive loads. In sleep modes where both the I/O clock ($\text{clk}_{\text{I/O}}$) and the ADC clock (clk_{ADC}) are stopped, the input buffers of the device will be disabled. This ensures that no power is consumed by the input logic when not needed. In some cases, the input logic is needed for detecting wake-up conditions, and it will then be enabled. Refer to the section [“Digital Input Enable and Sleep Modes” on page 94](#) for details on which pins are enabled. If the input buffer is enabled and the input signal is left floating or have an analog signal level close to $V_{\text{CC}}/2$, the input buffer will use excessive power.

For analog input pins, the digital input buffer should be disabled at all times. An analog signal level close to $V_{\text{CC}}/2$ on an input pin can cause significant current even in active mode. Digital input buffers can be disabled by writing to the Digital Input Disable Registers (DIDR1 and DIDR0). Refer to [“Digital Input Disable Register 1 – DIDR1” on page 264](#) and [“Digital Input Disable Register 0 – DIDR0” on page 281](#) for details.

6.7.8.7 On-chip Debug System

If the On-chip debug system is enabled by the DWEN Fuse and the chip enters sleep mode, the main clock source is enabled and hence always consumes power. In the deeper sleep modes, this will contribute significantly to the total current consumption.

6.8 System Control and Reset

6.8.1 Resetting the AVR

During reset, all I/O Registers are set to their initial values, and the program starts execution from the Reset Vector. For the Atmel® ATA6613, the instruction placed at the Reset Vector must be a JMP – Absolute Jump – instruction to the reset handling routine. For the Atmel ATA6612, the instruction placed at the Reset Vector must be an RJMP – Relative Jump – instruction to the reset handling routine. If the program never enables an interrupt source, the Interrupt Vectors are not used, and regular program code can be placed at these locations. This is also the case if the Reset Vector is in the Application section while the Interrupt Vectors are in the Boot section or vice versa (Atmel ATA6612/ATA6613 only). The circuit diagram in [Figure 6-15 on page 69](#) shows the reset logic. [Table 6-20 on page 70](#) defines the electrical parameters of the reset circuitry.

The I/O ports of the AVR® are immediately reset to their initial state when a reset source goes active. This does not require any clock source to be running.

After all reset sources have gone inactive, a delay counter is invoked, stretching the internal reset. This allows the power to reach a stable level before normal operation starts. The time-out period of the delay counter is defined by the user through the SUT and CKSEL Fuses. The different selections for the delay period are presented in [“Clock Sources” on page 52](#).

6.8.7 MCU Status Register – MCUSR

The MCU Status Register provides information on which reset source caused an MCU reset.

Bit	7	6	5	4	3	2	1	0	
	–	–	–	–	WDRF	BORF	EXTRF	PORF	MCUSR
Read/Write	R	R	R	R	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	See Bit Description				

- **Bit 7..4: Res: Reserved Bits**

These bits are unused bits in the Atmel® ATA6612/ATA6613, and will always read as zero.

- **Bit 3 – WDRF: Watchdog System Reset Flag**

This bit is set if a Watchdog System Reset occurs. The bit is reset by a Power-on Reset, or by writing a logic zero to the flag.

- **Bit 2 – BORF: Brown-out Reset Flag**

This bit is set if a Brown-out Reset occurs. The bit is reset by a Power-on Reset, or by writing a logic zero to the flag.

- **Bit 1 – EXTRF: External Reset Flag**

This bit is set if an External Reset occurs. The bit is reset by a Power-on Reset, or by writing a logic zero to the flag.

- **Bit 0 – PORF: Power-on Reset Flag**

This bit is set if a Power-on Reset occurs. The bit is reset only by writing a logic zero to the flag.

To make use of the Reset Flags to identify a reset condition, the user should read and then Reset the MCUSR as early as possible in the program. If the register is cleared before another reset occurs, the source of the reset can be found by examining the Reset Flags.

6.8.8 Internal Voltage Reference

Atmel ATA6612/ATA6613 features an internal bandgap reference. This reference is used for Brown-out Detection, and it can be used as an input to the Analog Comparator or the ADC.

6.8.8.1 Voltage Reference Enable Signals and Start-up Time

The voltage reference has a start-up time that may influence the way it should be used. The start-up time is given in [Table 6-23 on page 74](#). To save power, the reference is not always turned on. The reference is on during the following situations:

1. When the BOD is enabled (by programming the BODLEVEL [2..0] Fuses).
2. When the bandgap reference is connected to the Analog Comparator (by setting the ACBG bit in ACSR).
3. When the ADC is enabled.

Thus, when the BOD is not enabled, after setting the ACBG bit or enabling the ADC, the user must always allow the reference to start up before the output from the Analog Comparator or ADC is used. To reduce power consumption in Power-down mode, the user can avoid the three conditions above to ensure that the reference is turned off before entering Power-down mode.

When the BOOTRST Fuse is programmed, the Boot section size set to 2Kbytes and the IVSEL bit in the MCUCR Register is set before any interrupts are enabled, the most typical and general program setup for the Reset and Interrupt Vector Addresses in ATA6613 is:

```

Address  Labels Code          Comments
;
.org 0x1C00
0x1C00      jmp    RESET      ; Reset handler
0x1C02      jmp    EXT_INT0    ; IRQ0 Handler
0x1C04      jmp    EXT_INT1    ; IRQ1 Handler
...         ...      ...      ;
0x1C32      jmp    SPM_RDY     ; Store Program Memory Ready Handler
;
0x1C33  RESET: ldi    r16,high(RAMEND); Main program start
0x1C34      out    SPH,r16     ; Set Stack Pointer to top of RAM
0x1C35      ldi    r16,low(RAMEND)
0x1C36      out    SPL,r16
0x1C37      sei                      ; Enable interrupts
0x1C38      <instr> xxx

```

6.9.2.1 Moving Interrupts Between Application and Boot Space, Atmel ATA6612 and ATA6613

The MCU Control Register controls the placement of the Interrupt Vector table.

6.9.2.2 MCU Control Register – MCUCR

Bit	7	6	5	4	3	2	1	0	
	–	–	–	PUD	–	–	IVSEL	IVCE	MCUCR
Read/Write	R	R	R	R/W	R	R	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

• Bit 1 – IVSEL: Interrupt Vector Select

When the IVSEL bit is cleared (zero), the Interrupt Vectors are placed at the start of the Flash memory. When this bit is set (one), the Interrupt Vectors are moved to the beginning of the Boot Loader section of the Flash. The actual address of the start of the Boot Flash Section is determined by the BOOTSZ Fuses. Refer to section [“Boot Loader Support – Read-While-Write Self-Programming, Atmel ATA6612 and ATA6613” on page 284](#) for details. To avoid unintentional changes of Interrupt Vector tables, a special write procedure must be followed to change the IVSEL bit:

- Write the Interrupt Vector Change Enable (IVCE) bit to one.
- Within four cycles, write the desired value to IVSEL while writing a zero to IVCE.

Interrupts will automatically be disabled while this sequence is executed. Interrupts are disabled in the cycle IVCE is set, and they remain disabled until after the instruction following the write to IVSEL. If IVSEL is not written, interrupts remain disabled for four cycles. The I-bit in the Status Register is unaffected by the automatic disabling.

Note: If Interrupt Vectors are placed in the Boot Loader section and Boot Lock bit BLB02 is programmed, interrupts are disabled while executing from the Application section. If Interrupt Vectors are placed in the Application section and Boot Lock bit BLB12 is programmed, interrupts are disabled while executing from the Boot Loader section. Refer to section [“Boot Loader Support – Read-While-Write Self-Programming, Atmel ATA6612 and ATA6613” on page 284](#) for details on Boot Lock bits.

6.12.1.1 Definitions

Many register and bit references in this section are written in general form. A lower case “n” replaces the Timer/Counter number, in this case 0. A lower case “x” replaces the Output Compare Unit, in this case Compare Unit A or Compare Unit B. However, when using the register or bit defines in a program, the precise form must be used, i.e., TCNT0 for accessing Timer/Counter0 counter value and so on.

The definitions in [Table 6-43](#) are also used extensively throughout the document.

Table 6-43. General Counter Definitions

BOTTOM	The counter reaches the BOTTOM when it becomes 0x00.
MAX	The counter reaches its MAXimum when it becomes 0xFF (decimal 255).
TOP	The counter reaches the TOP when it becomes equal to the highest value in the count sequence. The TOP value can be assigned to be the fixed value 0xFF (MAX) or the value stored in the OCR0A Register. The assignment is dependent on the mode of operation.

6.12.1.2 Registers

The Timer/Counter (TCNT0) and Output Compare Registers (OCR0A and OCR0B) are 8-bit registers. Interrupt request (abbreviated to Int.Req. in the figure) signals are all visible in the Timer Interrupt Flag Register (TIFR0). All interrupts are individually masked with the Timer Interrupt Mask Register (TIMSK0). TIFR0 and TIMSK0 are not shown in the figure.

The Timer/Counter can be clocked internally, via the prescaler, or by an external clock source on the T0 pin. The Clock Select logic block controls which clock source and edge the Timer/Counter uses to increment (or decrement) its value. The Timer/Counter is inactive when no clock source is selected. The output from the Clock Select logic is referred to as the timer clock (clk_{T0}).

The double buffered Output Compare Registers (OCR0A and OCR0B) are compared with the Timer/Counter value at all times. The result of the compare can be used by the Waveform Generator to generate a PWM or variable frequency output on the Output Compare pins (OC0A and OC0B). See [“Using the Output Compare Unit” on page 143](#) for details. The compare match event will also set the Compare Flag (OCF0A or OCF0B) which can be used to generate an Output Compare interrupt request.

6.12.2 Timer/Counter Clock Sources

The Timer/Counter can be clocked by an internal or an external clock source. The clock source is selected by the Clock Select logic which is controlled by the Clock Select (CS02:0) bits located in the Timer/Counter Control Register (TCCR0B). For details on clock sources and prescaler (see [“Timer/Counter0 and Timer/Counter1 Prescalers” on page 131](#)).

The general I/O port function is overridden by the Output Compare (OC0x) from the Waveform Generator if either of the COM0x1:0 bits are set. However, the OC0x pin direction (input or output) is still controlled by the Data Direction Register (DDR) for the port pin. The Data Direction Register bit for the OC0x pin (DDR_OC0x) must be set as output before the OC0x value is visible on the pin. The port override function is independent of the Waveform Generation mode.

The design of the Output Compare pin logic allows initialization of the OC0x state before the output is enabled. Note that some COM0x1:0 bit settings are reserved for certain modes of operation (see [“8-bit Timer/Counter Register Description” on page 125](#)).

6.12.5.1 *Compare Output Mode and Waveform Generation*

The Waveform Generator uses the COM0x1:0 bits differently in Normal, CTC, and PWM modes. For all modes, setting the COM0x1:0 = 0 tells the Waveform Generator that no action on the OC0x Register is to be performed on the next compare match. For compare output actions in the non-PWM modes refer to [Table 6-44 on page 125](#). For fast PWM mode, refer to [Table 6-45 on page 125](#), and for phase correct PWM refer to [Table 6-46 on page 126](#).

A change of the COM0x1:0 bits state will have effect at the first compare match after the bits are written. For non-PWM modes, the action can be forced to have immediate effect by using the FOC0x strobe bits.

6.12.6 **Modes of Operation**

The mode of operation, i.e., the behavior of the Timer/Counter and the Output Compare pins, is defined by the combination of the Waveform Generation mode (WGM02:0) and Compare Output mode (COM0x1:0) bits. The Compare Output mode bits do not affect the counting sequence, while the Waveform Generation mode bits do. The COM0x1:0 bits control whether the PWM output generated should be inverted or not (inverted or non-inverted PWM). For non-PWM modes the COM0x1:0 bits control whether the output should be set, cleared, or toggled at a compare match (see [“Compare Match Output Unit” on page 117](#)).

For detailed timing information refer to [“Timer/Counter Timing Diagrams” on page 123](#).

6.12.6.1 *Normal Mode*

The simplest mode of operation is the Normal mode (WGM02:0 = 0). In this mode the counting direction is always up (incrementing), and no counter clear is performed. The counter simply overruns when it passes its maximum 8-bit value (TOP = 0xFF) and then restarts from the bottom (0x00). In normal operation the Timer/Counter Overflow Flag (TOV0) will be set in the same timer clock cycle as the TCNT0 becomes zero. The TOV0 Flag in this case behaves like a ninth bit, except that it is only set, not cleared. However, combined with the timer overflow interrupt that automatically clears the TOV0 Flag, the timer resolution can be increased by software. There are no special cases to consider in the Normal mode, a new counter value can be written anytime.

The Output Compare unit can be used to generate interrupts at some given time. Using the Output Compare to generate waveforms in Normal mode is not recommended, since this will occupy too much of the CPU time.

6.12.8.2 Timer/Counter Control Register B – TCCR0B

Bit	7	6	5	4	3	2	1	0	
	FOC0A	FOC0B	–	–	WGM02	CS02	CS01	CS00	TCCR0B
Read/Write	W	W	R	R	R	R	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

- Bit 7 – FOC0A: Force Output Compare A

The FOC0A bit is only active when the WGM bits specify a non-PWM mode.

However, for ensuring compatibility with future devices, this bit must be set to zero when TCCR0B is written when operating in PWM mode. When writing a logical one to the FOC0A bit, an immediate Compare Match is forced on the Waveform Generation unit. The OC0A output is changed according to its COM0A1:0 bits setting. Note that the FOC0A bit is implemented as a strobe. Therefore it is the value present in the COM0A1:0 bits that determines the effect of the forced compare.

A FOC0A strobe will not generate any interrupt, nor will it clear the timer in CTC mode using OCR0A as TOP.

The FOC0A bit is always read as zero.

- Bit 6 – FOC0B: Force Output Compare B

The FOC0B bit is only active when the WGM bits specify a non-PWM mode.

However, for ensuring compatibility with future devices, this bit must be set to zero when TCCR0B is written when operating in PWM mode. When writing a logical one to the FOC0B bit, an immediate Compare Match is forced on the Waveform Generation unit. The OC0B output is changed according to its COM0B1:0 bits setting. Note that the FOC0B bit is implemented as a strobe. Therefore it is the value present in the COM0B1:0 bits that determines the effect of the forced compare.

A FOC0B strobe will not generate any interrupt, nor will it clear the timer in CTC mode using OCR0B as TOP.

The FOC0B bit is always read as zero.

- Bits 5:4 – Res: Reserved Bits

These bits are reserved bits in the Atmel® ATA6612/ATA6613 and will always read as zero.

- Bit 3 – WGM02: Waveform Generation Mode

See the description in the [“Timer/Counter Control Register A – TCCR0A” on page 125](#).

- Bits 2:0 – CS02:0: Clock Select

The three Clock Select bits select the clock source to be used by the Timer/Counter.

The following code examples show how to do an atomic write of the TCNT1 Register contents. Writing any of the OCR1A/B or ICR1 Registers can be done by using the same principle.

Assembly Code Example⁽¹⁾

```
TIM16_WriteTCNT1:
    ; Save global interrupt flag
    in r18,SREG
    ; Disable interrupts
    cli
    ; Set TCNT1 to r17:r16
    out TCNT1H,r17
    out TCNT1L,r16
    ; Restore global interrupt flag
    out SREG,r18
    ret
```

C Code Example⁽¹⁾

```
void TIM16_WriteTCNT1( unsigned int i )
{
    unsigned char sreg;
    unsigned int i;
    /* Save global interrupt flag */
    sreg = SREG;
    /* Disable interrupts */
    _CLI();
    /* Set TCNT1 to i */
    TCNT1 = i;
    /* Restore global interrupt flag */
    SREG = sreg;
}
```

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 “SBR”, “SBRC”, “SBR”, and “CBR”.

The assembly code example requires that the r17:r16 register pair contains the value to be written to TCNT1.

6.14.2.1 Reusing the Temporary High Byte Register

If writing to more than one 16-bit register where the high byte is the same for all registers written, then the high byte only needs to be written once. However, note that the same rule of atomic operation described previously also applies in this case.

The OCR1x Register is double buffered when using any of the twelve *Pulse Width Modulation* (PWM) modes. For the Normal and *Clear Timer on Compare* (CTC) modes of operation, the double buffering is disabled. The double buffering synchronizes the update of the OCR1x Compare Register to either TOP or BOTTOM of the counting sequence. The synchronization prevents the occurrence of odd-length, non-symmetrical PWM pulses, thereby making the output glitch-free.

The OCR1x Register access may seem complex, but this is not case. When the double buffering is enabled, the CPU has access to the OCR1x Buffer Register, and if double buffering is disabled the CPU will access the OCR1x directly. The content of the OCR1x (Buffer or Compare) Register is only changed by a write operation (the Timer/Counter does not update this register automatically as the TCNT1 and ICR1 Register). Therefore OCR1x is not read via the high byte temporary register (TEMP). However, it is a good practice to read the low byte first as when accessing other 16-bit registers. Writing the OCR1x Registers must be done via the TEMP Register since the compare of all 16 bits is done continuously. The high byte (OCR1xH) has to be written first. When the high byte I/O location is written by the CPU, the TEMP Register will be updated by the value written. Then when the low byte (OCR1xL) is written to the lower eight bits, the high byte will be copied into the upper 8-bits of either the OCR1x buffer or OCR1x Compare Register in the same system clock cycle. For more information of how to access the 16-bit registers refer to [“Accessing 16-bit Registers” on page 135](#).

6.14.6.1 *Force Output Compare*

In non-PWM Waveform Generation modes, the match output of the comparator can be forced by writing a one to the *Force Output Compare* (FOC1x) bit. Forcing compare match will not set the OCF1x Flag or reload/clear the timer, but the OC1x pin will be updated as if a real compare match had occurred (the COM11:0 bits settings define whether the OC1x pin is set, cleared or toggled).

6.14.6.2 *Compare Match Blocking by TCNT1 Write*

All CPU writes to the TCNT1 Register will block any compare match that occurs in the next timer clock cycle, even when the timer is stopped. This feature allows OCR1x to be initialized to the same value as TCNT1 without triggering an interrupt when the Timer/Counter clock is enabled.

6.14.6.3 *Using the Output Compare Unit*

Since writing TCNT1 in any mode of operation will block all compare matches for one timer clock cycle, there are risks involved when changing TCNT1 when using any of the Output Compare channels, independent of whether the Timer/Counter is running or not. If the value written to TCNT1 equals the OCR1x value, the compare match will be missed, resulting in incorrect waveform generation. Do not write the TCNT1 equal to TOP in PWM modes with variable TOP values. The compare match for the TOP will be ignored and the counter will continue to 0xFFFF. Similarly, do not write the TCNT1 value equal to BOTTOM when the counter is downcounting.

The setup of the OC1x should be performed before setting the Data Direction Register for the port pin to output. The easiest way of setting the OC1x value is to use the Force Output Compare (FOC1x) strobe bits in Normal mode. The OC1x Register keeps its value even when changing between Waveform Generation modes. Be aware that the COM1x1:0 bits are not double buffered together with the compare value. Changing the COM1x1:0 bits will take effect immediately.

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.

6.14.8.4 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 6-47 on page 150](#). 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.

6.15.4.2 *Compare Match Blocking by TCNT2 Write*

All CPU write operations to the TCNT2 Register will block any compare match that occurs in the next timer clock cycle, even when the timer is stopped. This feature allows OCR2x to be initialized to the same value as TCNT2 without triggering an interrupt when the Timer/Counter clock is enabled.

6.15.4.3 *Using the Output Compare Unit*

Since writing TCNT2 in any mode of operation will block all compare matches for one timer clock cycle, there are risks involved when changing TCNT2 when using the Output Compare channel, independently of whether the Timer/Counter is running or not. If the value written to TCNT2 equals the OCR2x value, the compare match will be missed, resulting in incorrect waveform generation. Similarly, do not write the TCNT2 value equal to BOTTOM when the counter is downcounting.

The setup of the OC2x should be performed before setting the Data Direction Register for the port pin to output. The easiest way of setting the OC2x value is to use the Force Output Compare (FOC2x) strobe bit in Normal mode. The OC2x Register keeps its value even when changing between Waveform Generation modes.

Be aware that the COM2x1:0 bits are not double buffered together with the compare value. Changing the COM2x1:0 bits will take effect immediately.

6.15.5 **Compare Match Output Unit**

The Compare Output mode (COM2x1:0) bits have two functions. The Waveform Generator uses the COM2x1:0 bits for defining the Output Compare (OC2x) state at the next compare match. Also, the COM2x1:0 bits control the OC2x pin output source. [Figure 6-56 on page 167](#) shows a simplified schematic of the logic affected by the COM2x1:0 bit setting. The I/O Registers, I/O bits, and I/O pins in the figure are shown in bold. Only the parts of the general I/O Port Control Registers (DDR and PORT) that are affected by the COM2x1:0 bits are shown. When referring to the OC2x state, the reference is for the internal OC2x Register, not the OC2x pin.

The following code examples show how to initialize the SPI as a Slave and how to perform a simple reception.

Assembly Code Example⁽¹⁾

```

SPI_SlaveInit:
    ; Set MISO output, all others input
    ldi r17, (1<<DD_MISO)
    out DDR_SPI, r17
    ; Enable SPI
    ldi r17, (1<<SPE)
    out SPCR, r17
    ret

SPI_SlaveReceive:
    ; Wait for reception complete
    sbis SPSR, SPIF
    rjmp SPI_SlaveReceive
    ; Read received data and return
    in r16, SPDR
    ret

```

C Code Example⁽¹⁾

```

void SPI_SlaveInit(void)
{
    /* Set MISO output, all others input */
    DDR_SPI = (1<<DD_MISO);
    /* Enable SPI */
    SPCR = (1<<SPE);
}

char SPI_SlaveReceive(void)
{
    /* Wait for reception complete */
    while(!(SPSR & (1<<SPIF)))
    ;
    /* Return Data Register */
    return SPDR;
}

```

Note: 1. The example code assumes that the part specific header file is included.

6.17.7.3 Asynchronous Operational Range

The operational range of the Receiver is dependent on the mismatch between the received bit rate and the internally generated baud rate. If the Transmitter is sending frames at too fast or too slow bit rates, or the internally generated baud rate of the Receiver does not have a similar (see [Table 6-72](#)) base frequency, the Receiver will not be able to synchronize the frames to the start bit.

The following equations can be used to calculate the ratio of the incoming data rate and internal receiver baud rate.

$$R_{slow} = \frac{(D + 1)S}{S - 1 + D \times S + S_F} \qquad R_{fast} = \frac{(D + 2)S}{(D + 1)S + S_M}$$

D	Sum of character size and parity size (D = 5 to 10 bit)
S	Samples per bit. S = 16 for Normal Speed mode and S = 8 for Double Speed mode.
S_F	First sample number used for majority voting. S _F = 8 for normal speed and S _F = 4 for Double Speed mode.
S_M	Middle sample number used for majority voting. S _M = 9 for normal speed and S _M = 5 for Double Speed mode.
R_{slow}	is the ratio of the slowest incoming data rate that can be accepted in relation to the receiver baud rate. R _{fast} is the ratio of the fastest incoming data rate that can be accepted in relation to the receiver baud rate.

[Table 6-72](#) and [Table 6-73 on page 211](#) list the maximum receiver baud rate error that can be tolerated. Note that Normal Speed mode has higher toleration of baud rate variations.

Table 6-72. Recommended Maximum Receiver Baud Rate Error for Normal Speed Mode (U2Xn = 0)

D # (Data+Parity Bit)	R _{slow} (%)	R _{fast} (%)	Max Total Error (%)	Recommended Max Receiver Error (%)
5	93.20	106.67	+6.67/-6.8	±3.0
6	94.12	105.79	+5.79/-5.88	±2.5
7	94.81	105.11	+5.11/-5.19	±2.0
8	95.36	104.58	+4.58/-4.54	±2.0
9	95.81	104.14	+4.14/-4.19	±1.5
10	96.17	103.78	+3.78/-3.83	±1.5

Table 6-83. Equations for Calculating Baud Rate Register Setting

Operating Mode	Equation for Calculating Baud Rate ⁽¹⁾	Equation for Calculating UBRRn Value
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)

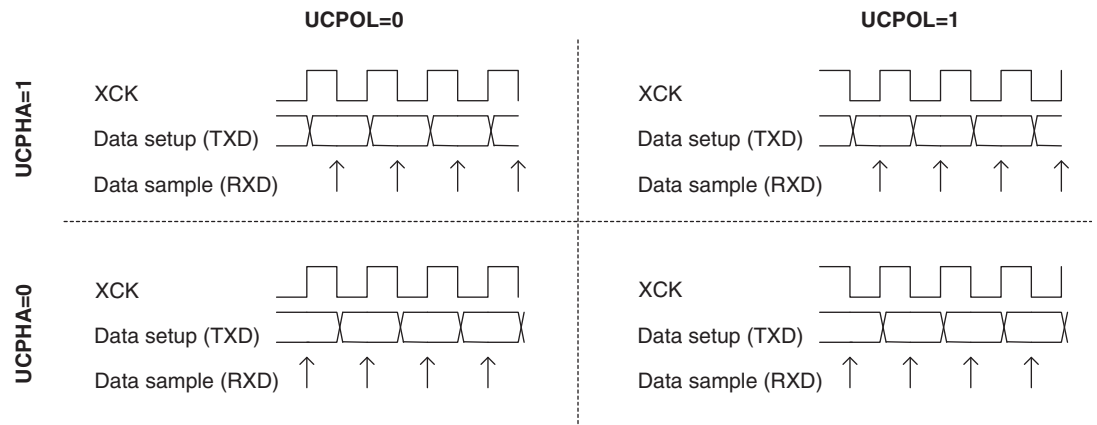
6.18.3 SPI Data Modes and Timing

There are four combinations of XCKn (SCK) phase and polarity with respect to serial data, which are determined by control bits UCPHAN and UCPOLn. The data transfer timing diagrams are shown in [Figure 6-76 on page 221](#). Data bits are shifted out and latched in on opposite edges of the XCKn signal, ensuring sufficient time for data signals to stabilize. The UCPOLn and UCPHAN functionality is summarized in [Table 6-84](#). Note that changing the setting of any of these bits will corrupt all ongoing communication for both the Receiver and Transmitter.

Table 6-84. UCPOLn and UCPHAN Functionality-

UCPOLn	UCPHAn	SPI Mode	Leading Edge	Trailing Edge
0	0	0	Sample (Rising)	Setup (Falling)
0	1	1	Setup (Rising)	Sample (Falling)
1	0	2	Sample (Falling)	Setup (Rising)
1	1	3	Setup (Falling)	Sample (Rising)

Figure 6-76. UCPHAN and UCPOLn Data Transfer Timing Diagrams



6.18.6.3 USART MSPIM Control and Status Register n B - UCSRnB

Bit	7	6	5	4	3	2	1	0	
	RXCIE_n	TXCIE_n	UDRIE	RXEN_n	TXEN_n	-	-	-	UCSRnB
Read/Write	R/W	R/W	R/W	R/W	R/W	R	R	R	
Initial Value	0	0	0	0	0	1	1	0	

- Bit 7 - RXCIE_n: RX Complete Interrupt Enable

Writing this bit to one enables interrupt on the RXC_n Flag. A USART Receive Complete interrupt will be generated only if the RXCIE_n bit is written to one, the Global Interrupt Flag in SREG is written to one and the RXC_n bit in UCSRnA is set.

- Bit 6 - TXCIE_n: TX Complete Interrupt Enable

Writing this bit to one enables interrupt on the TXC_n Flag. A USART Transmit Complete interrupt will be generated only if the TXCIE_n bit is written to one, the Global Interrupt Flag in SREG is written to one and the TXC_n bit in UCSRnA is set.

- Bit 5 - UDRIE: USART Data Register Empty Interrupt Enable

Writing this bit to one enables interrupt on the UDRE_n Flag. A Data Register Empty interrupt will be generated only if the UDRIE bit is written to one, the Global Interrupt Flag in SREG is written to one and the UDRE_n bit in UCSRnA is set.

- Bit 4 - RXEN_n: Receiver Enable

Writing this bit to one enables the USART Receiver in MSPIM mode. The Receiver will override normal port operation for the RxD_n pin when enabled. Disabling the Receiver will flush the receive buffer. Only enabling the receiver in MSPI mode (i.e. setting RXEN_n=1 and TXEN_n=0) has no meaning since it is the transmitter that controls the transfer clock and since only master mode is supported.

- Bit 3 - TXEN_n: Transmitter Enable

Writing this bit to one enables the USART Transmitter. The Transmitter will override normal port operation for the TxD_n pin when enabled. The disabling of the Transmitter (writing TXEN_n to zero) will not become effective until ongoing and pending transmissions are completed, i.e., when the Transmit Shift Register and Transmit Buffer Register do not contain data to be transmitted. When disabled, the Transmitter will no longer override the TxD_n port.

- Bit 2:0 - Reserved Bits in MSPI mode

When in MSPI mode, these bits are reserved for future use. For compatibility with future devices, these bits must be written to zero when UCSRnB is written.

Figure 6-105. ADC Timing Diagram, Auto Triggered Conversion

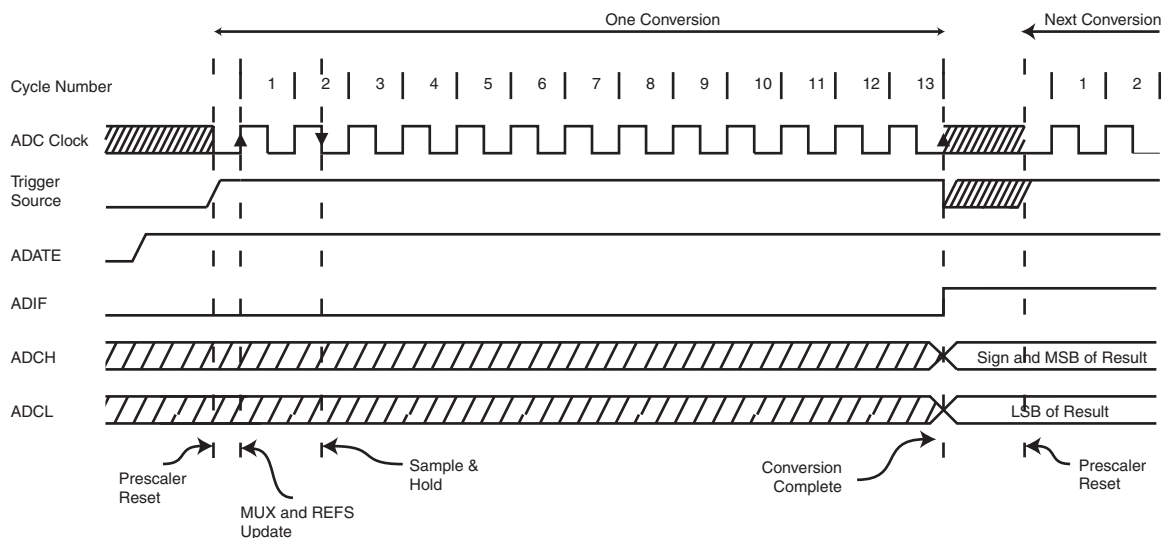


Figure 6-106. ADC Timing Diagram, Free Running Conversion

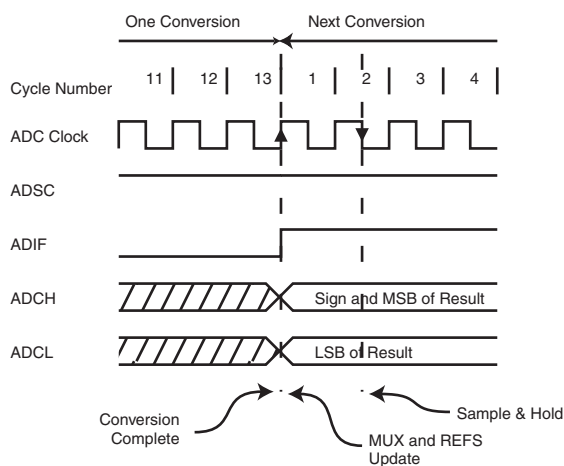


Table 6-97. ADC Conversion Time

Condition	Sample & Hold (Cycles from Start of Conversion)	Conversion Time (Cycles)
First conversion	13.5	25
Normal conversions, single ended	1.5	13
Auto Triggered conversions	2	13.5

When designing a system where debugWIRE will be used, the following observations must be made for correct operation:

- Pull-up resistors on the dW/(RESET) line must not be smaller than 10kΩ. The pull-up resistor is not required for debugWIRE functionality.
- Connecting the RESET pin directly to V_{CC} will not work.
- Capacitors connected to the RESET pin must be disconnected when using debugWIRE.
- All external reset sources must be disconnected.

6.22.4 Software Break Points

debugWIRE supports Program memory Break Points by the AVR® Break instruction. Setting a Break Point in AVR Studio® will insert a BREAK instruction in the Program memory. The instruction replaced by the BREAK instruction will be stored. When program execution is continued, the stored instruction will be executed before continuing from the Program memory. A break can be inserted manually by putting the BREAK instruction in the program.

The Flash must be re-programmed each time a Break Point is changed. This is automatically handled by AVR Studio through the debugWIRE interface. The use of Break Points will therefore reduce the Flash Data retention. Devices used for debugging purposes should not be shipped to end customers.

6.22.5 Limitations of debugWIRE

The debugWIRE communication pin (dW) is physically located on the same pin as External Reset (RESET). An External Reset source is therefore not supported when the debugWIRE is enabled.

The debugWIRE system accurately emulates all I/O functions when running at full speed, i.e., when the program in the CPU is running. When the CPU is stopped, care must be taken while accessing some of the I/O Registers via the debugger (AVR Studio).

A programmed DWEN Fuse enables some parts of the clock system to be running in all sleep modes. This will increase the power consumption while in sleep. Thus, the DWEN Fuse should be disabled when debugWire is not used.

6.22.6 debugWIRE Related Register in I/O Memory

The following section describes the registers used with the debugWire.

6.22.6.1 debugWire Data Register – DWDR

Bit	7	6	5	4	3	2	1	0	
	DWDR[7:0]								DWDR
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	

The DWDR Register provides a communication channel from the running program in the MCU to the debugger. This register is only accessible by the debugWIRE and can therefore not be used as a general purpose register in the normal operations.

Table 6-115. Lock Bit Protection Modes⁽¹⁾⁽²⁾. Only Atmel® ATA6612/ATA6613.

BLB0 Mode	BLB02	BLB01	
1	1	1	No restrictions for SPM or LPM accessing the Application section.
2	1	0	SPM is not allowed to write to the Application section.
3	0	0	SPM is not allowed to write to the Application section, and LPM executing from the Boot Loader section is not allowed to read from the Application section. If Interrupt Vectors are placed in the Boot Loader section, interrupts are disabled while executing from the Application section.
4	0	1	LPM executing from the Boot Loader section is not allowed to read from the Application section. If Interrupt Vectors are placed in the Boot Loader section, interrupts are disabled while executing from the Application section.
BLB1 Mode	BLB12	BLB11	
1	1	1	No restrictions for SPM or LPM accessing the Boot Loader section.
2	1	0	SPM is not allowed to write to the Boot Loader section.
3	0	0	SPM is not allowed to write to the Boot Loader section, and LPM executing from the Application section is not allowed to read from the Boot Loader section. If Interrupt Vectors are placed in the Application section, interrupts are disabled while executing from the Boot Loader section.
4	0	1	LPM executing from the Application section is not allowed to read from the Boot Loader section. If Interrupt Vectors are placed in the Application section, interrupts are disabled while executing from the Boot Loader section.

Notes: 1. Program the Fuse bits and Boot Lock bits before programming the LB1 and LB2.
2. “1” means unprogrammed, “0” means programmed

6.24.2 Fuse Bits

The Atmel ATA6612/ATA6613 has three Fuse bytes. [Table 6-116](#) to [Table 6-118 on page 303](#) describe briefly the functionality of all the fuses and how they are mapped into the Fuse bytes. Note that the fuses are read as logical zero, “0”, if they are programmed.

9. Ordering Information

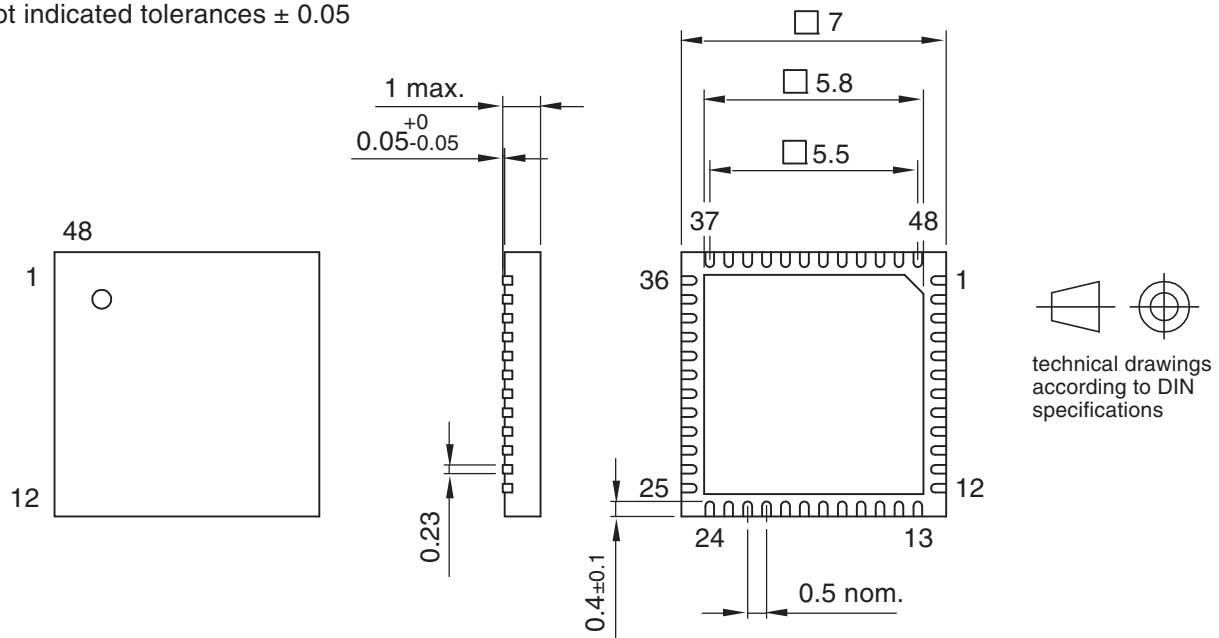
Extended Type Number	Program Memory	Package	MOQ
ATA6612P-PLQW	8kB flash	QFN48, 7×7	4,000 pieces
ATA6612P-PLPW	8kB flash	QFN48, 7×7	1,000 pieces
ATA6613P-PLQW	16kB flash	QFN48, 7×7	4,000 pieces
ATA6613P-PLPW	16kB flash	QFN48, 7×7	1,000 pieces

10. Package Information

Package: QFN 48 - 7 x 7
Exposed pad 5.8 x 5.8

Dimensions in mm

Not indicated tolerances ± 0.05



Drawing-No.: 6.543-5089.03-4

Issue: 1; 22.01.03