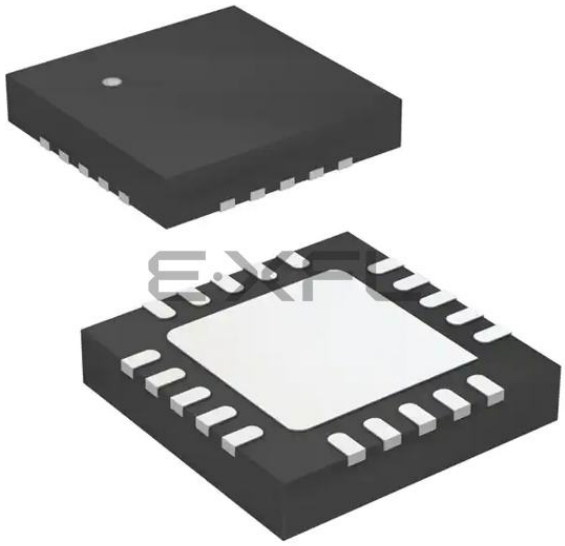


Welcome to E-XFL.COM

What is "Embedded - Microcontrollers"?

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

Applications of "Embedded - Microcontrollers"

Details	
Product Status	Active
Core Processor	AVR
Core Size	8-Bit
Speed	16MHz
Connectivity	USI
Peripherals	Brown-out Detect/Reset, POR, PWM, WDT
Number of I/O	16
Program Memory Size	4KB (2K x 16)
Program Memory Type	FLASH
EEPROM Size	256 x 8
RAM Size	256 x 8
Voltage - Supply (Vcc/Vdd)	2.7V ~ 5.5V
Data Converters	A/D 11x10b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 125°C (TA)
Mounting Type	Surface Mount
Package / Case	20-WFQFN Exposed Pad
Supplier Device Package	20-WQFN (4x4)
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/attiny461-15maz

5.3 Status Register

The status register contains information about the result of the most recently executed arithmetic instruction. This information can be used for altering program flow in order to perform conditional operations. Note that the status register is updated after all ALU operations, as specified in the instruction set reference. This will in many cases remove the need for using the dedicated compare instructions, resulting in faster and more compact code. The status register is not automatically stored when entering an interrupt routine and restored when returning from an interrupt. This must be handled by software.

5.3.1 SREG – AVR Status Register

The AVR® status register – SREG – is defined as:

Bit	7	6	5	4	3	2	1	0	
0x3F (0x5F)	**I**	**T**	**H**	**S**	**V**	**N**	**Z**	**C**	**SREG**
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

- **Bit 7 – I: Global Interrupt Enable**

The global interrupt enable bit must be set for the interrupts to be enabled. The individual interrupt enable control is then performed in separate control registers. If the global interrupt enable register is cleared, none of the interrupts are enabled independent of the individual interrupt enable settings. The I-bit is cleared by hardware after an interrupt has occurred, and is set by the RETI instruction to enable subsequent interrupts. The I-bit can also be set and cleared by the application with the SEI and CLI instructions, as described in the instruction set reference.

- **Bit 6 – T: Bit Copy Storage**

The bit copy instructions BLD (Bit LoaD) and BST (Bit STore) use the T-bit as source or destination for the operated bit. A bit from a register in the register file can be copied into T by the BST instruction, and a bit in T can be copied into a bit in a register in the register file by the BLD instruction.

- **Bit 5 – H: Half Carry Flag**

The half carry flag H indicates a half carry in some arithmetic operations. Half carry is useful in BCD arithmetic. See the “Instruction Set Description” for detailed information.

- **Bit 4 – S: Sign Bit, S = N ⊕ V**

The S-bit is always an exclusive or between the negative flag N and the two complement overflow flag V. See the “Instruction Set Description” for detailed information.

- **Bit 3 – V: Two’s Complement Overflow Flag**

The Two’s complement overflow flag V supports two’s complement arithmetics. See the “Instruction Set Description” for detailed information.

- **Bit 2 – N: Negative Flag**

The negative flag N indicates a negative result in an arithmetic or logic operation. See the “Instruction Set Description” for detailed information.

- **Bit 1 – Z: Zero Flag**

The zero flag Z indicates a zero result in an arithmetic or logic operation. See the “Instruction Set Description” for detailed information.

- **Bit 0 – C: Carry Flag**

The carry flag C indicates a carry in an arithmetic or logic operation. See the “Instruction Set Description” for detailed information.

9. System Control and Reset

9.1 Resetting the AVR

During reset, all I/O registers are set to their initial values, and the program starts execution from the reset vector. The instruction placed at the reset vector must be a 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. The circuit diagram in Figure 9-1 on page 39 shows the reset logic. See Section 23.5 “System and Reset Characteristics” on page 174 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 Section 7.2 “Clock Sources” on page 26.

9.2 Reset Sources

The Atmel® ATtiny261/461/861 has four sources of reset:

- Power-on reset. The MCU is reset when the supply voltage is below the power-on reset threshold (V_{POT}).
- External reset. The MCU is reset when a low level is present on the $\overline{\text{RESET}}$ pin for longer than the minimum pulse length.
- Watchdog reset. The MCU is reset when the watchdog timer period expires and the watchdog is enabled.
- Brown-out reset. The MCU is reset when the supply voltage V_{CC} is below the brown-out reset threshold (V_{BOT}) and the brown-out detector is enabled.

Figure 9-1. Reset Logic

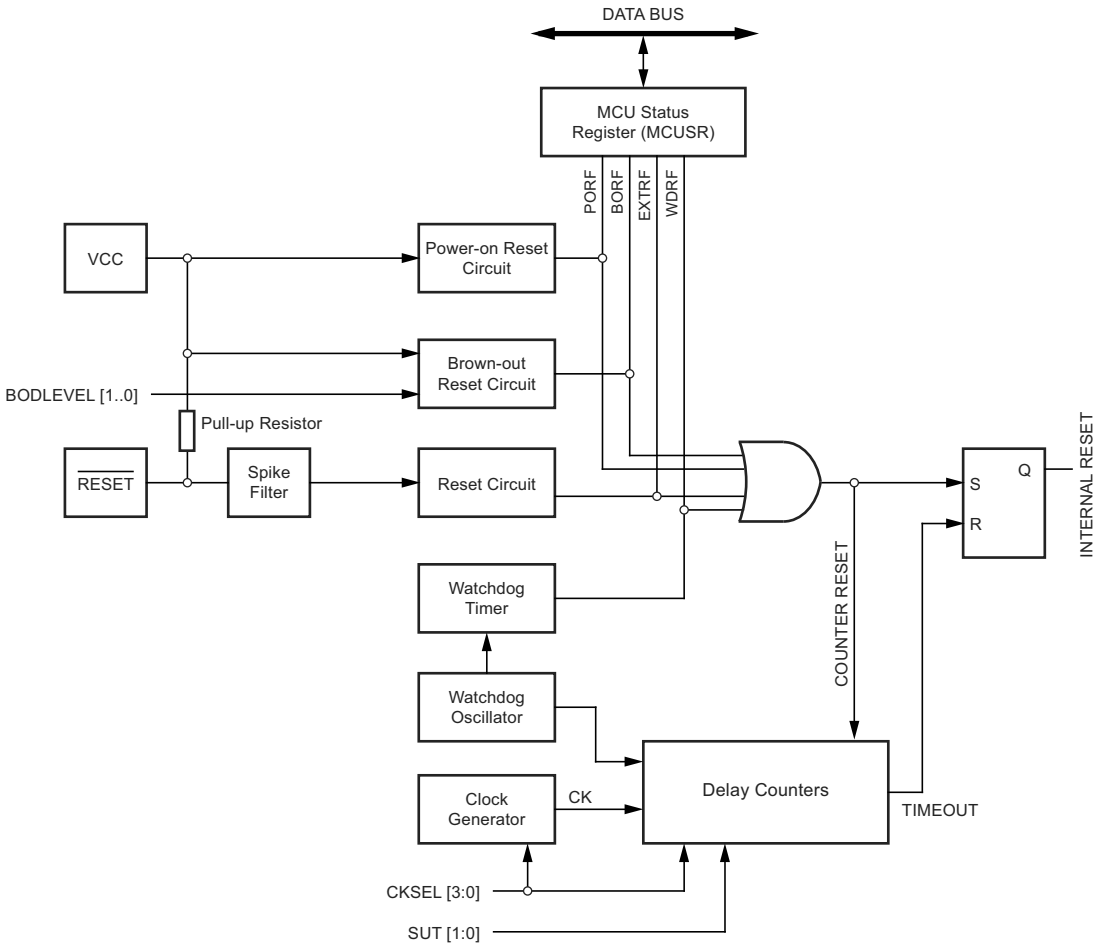
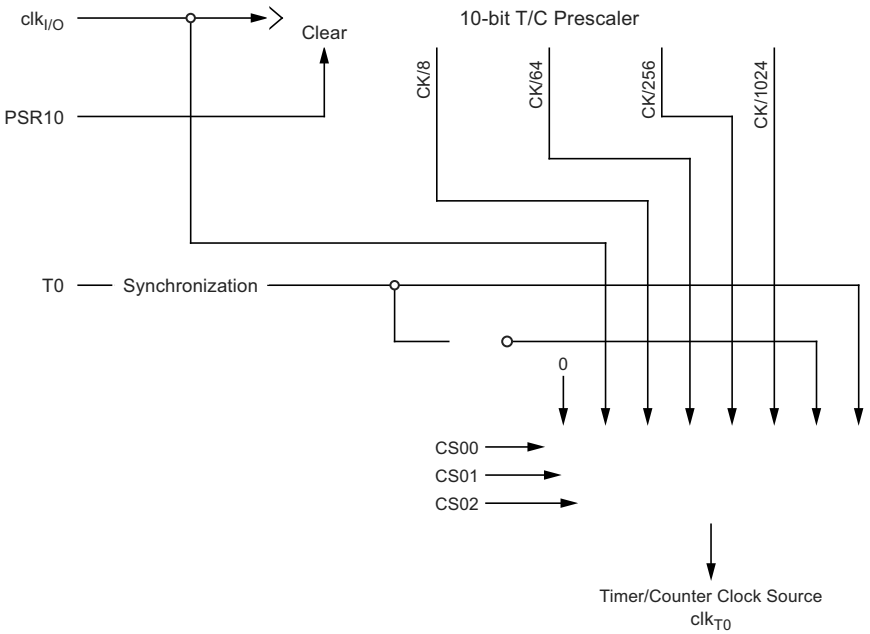


Figure 13-2. Prescaler for Timer/Counter0



Note: 1. The synchronization logic on the input pins (T0) is shown in Figure 13-1.

14.8 Timer/Counter Timing Diagrams

The Timer/Counter is a synchronous design and the timer clock (clk_{T0}) is therefore shown as a clock enable signal in the following figures. The figures include information on when interrupt flags are set. Figure 14-5 contains timing data for basic Timer/Counter operation. The figure shows the count sequence close to the MAX value.

Figure 14-5. Timer/Counter Timing Diagram, no Prescaling

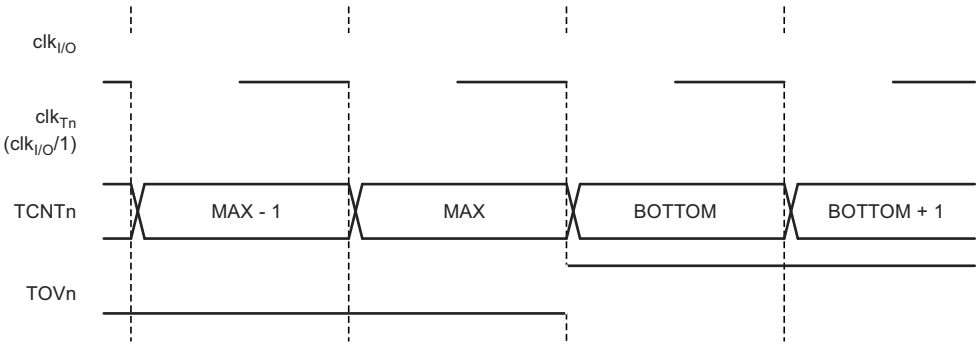


Figure 14-6 shows the same timing data, but with the prescaler enabled.

Figure 14-6. Timer/Counter Timing Diagram, with Prescaler ($f_{clk\_I/O}/8$)

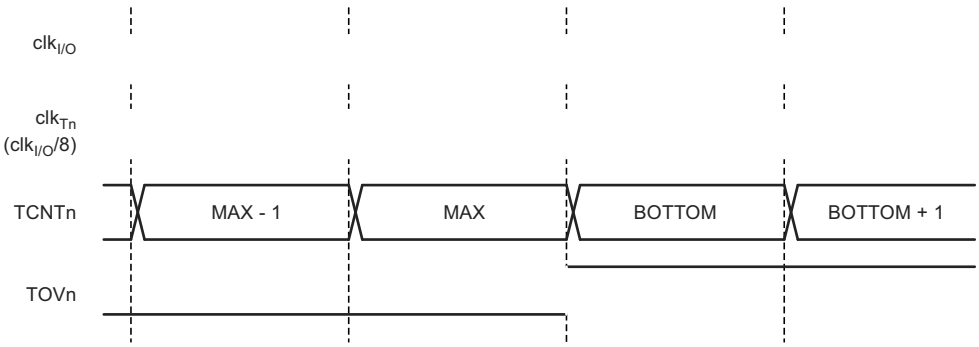


Figure 14-7 shows the setting of OCF0A and OCF0B in normal mode.

Figure 14-7. Timer/Counter Timing Diagram, Setting of OCF0x, with Prescaler ($f_{clk\_I/O}/8$)

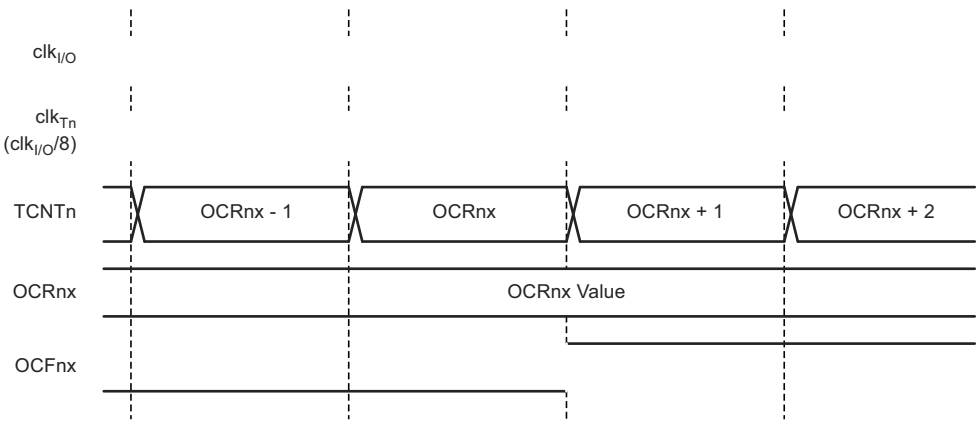
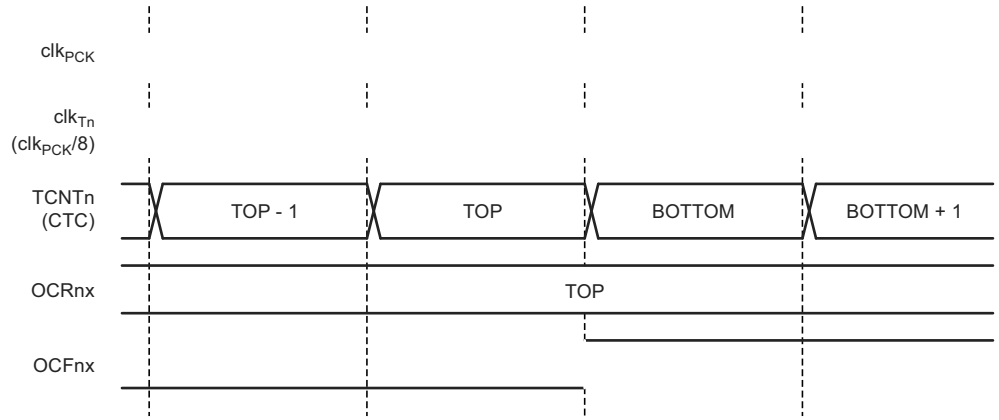


Figure 14-8 shows the setting of OCF0A and the clearing of TCNT0 in CTC mode.

Figure 14-8. Timer/Counter Timing Diagram, CTC mode, with Prescaler ($f_{clk\_I/O}$/8)



14.9 Accessing Registers in 16-bit Mode

In 16-bit mode (the TCW0 bit is set to one) the TCNT0H/L and OCR0A/B or TCNT0L/H and OCR0B/A are 16-bit registers that can be accessed by the AVR CPU via the 8-bit data bus. The 16-bit register must be byte accessed using two read or write operations. The 16-bit Timer/Counter has a single 8-bit register for temporary storing of the high byte of the 16-bit access. The same temporary register is shared between all 16-bit registers. Accessing the low byte triggers the 16-bit read or write operation. When the low byte of a 16-bit register is written by the CPU, the high byte stored in the temporary register, and the low byte written are both copied into the 16-bit register in the same clock cycle. When the low byte of a 16-bit register is read by the CPU, the high byte of the 16-bit register is copied into the temporary register in the same clock cycle as the low byte is read.

There is one exception in the temporary register usage. In the output compare mode the 16-bit output compare register OCR0A/B is read without the temporary register, because the output compare register contains a fixed value that is only changed by CPU access. However, in 16-bit Input Capture mode the ICR0 register formed by the OCR0A and OCR0B registers must be accessed with the temporary register.

To do a 16-bit write, the high byte must be written before the low byte. For a 16-bit read, the low byte must be read before the high byte.

14.10.7 TIFR – Timer/Counter0 Interrupt Flag Register

Bit	7	6	5	4	3	2	1	0	
0x38 (0x58)	**OCF1D**	**OCF1A**	**OCF1B**	**OCF0A**	**OCF0B**	**TOV1**	**TOV0**	**ICF0**	**TIFR**
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R	
Initial Value	0	0	0	0	0	0	0	0	

- **Bit 4– OCF0A: Output Compare Flag 0 A**

The OCF0A bit is set when a compare match occurs between the Timer/Counter0 and the data in OCR0A – output compare register0. OCF0A is cleared by hardware when executing the corresponding interrupt handling vector. Alternatively, OCF0A is cleared by writing a logic one to the flag. When the I-bit in SREG, OCIE0A (Timer/Counter0 compare match Interrupt Enable), and OCF0A are set, the Timer/Counter0 compare match interrupt is executed.

The OCF0A is also set in 16-bit mode when a compare match occurs between the Timer/Counter and 16-bit data in OCR0B/A. The OCF0A is not set in input capture mode when the output compare register OCR0A is used as an Input capture register.

- **Bit 3 – OCF0B: Output Compare Flag 0 B**

The OCF0B bit is set when a compare match occurs between the Timer/Counter and the data in OCR0B – output compare register0 B. OCF0B is cleared by hardware when executing the corresponding interrupt handling vector. Alternatively, OCF0B is cleared by writing a logic one to the flag. When the I-bit in SREG, OCIE0B (Timer/Counter compare B match interrupt enable), and OCF0B are set, the Timer/Counter Compare match interrupt is executed.

The OCF0B is not set in 16-bit output compare mode when the output compare register OCR0B is used as the high byte of the 16-bit output compare register or in 16-bit input capture mode when the output compare register OCR0B is used as the high byte of the input capture register.

- **Bit 1 – TOV0: Timer/Counter0 Overflow Flag**

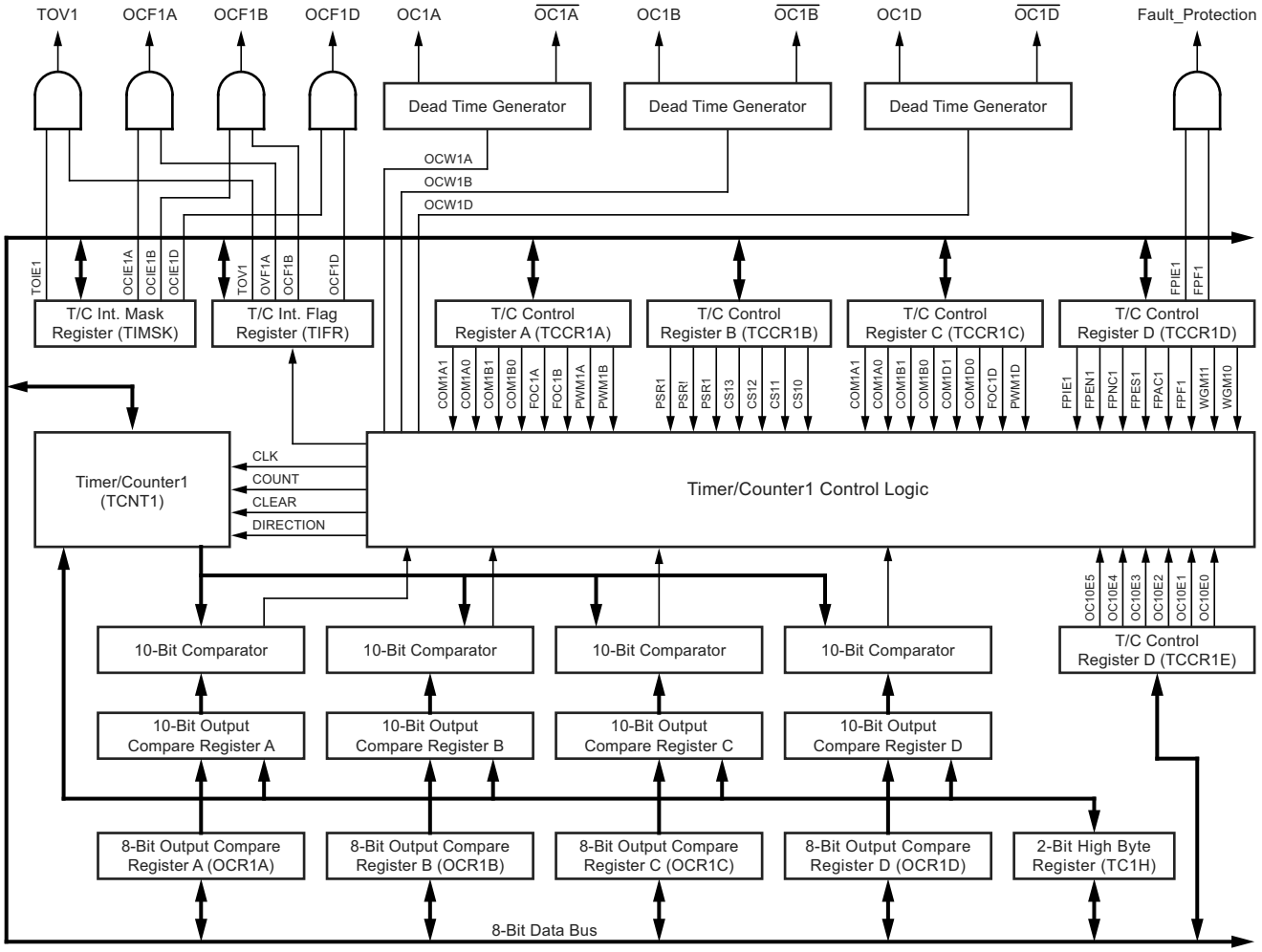
The bit TOV0 is set when an overflow occurs in Timer/Counter0. TOV0 is cleared by hardware when executing the corresponding interrupt handling vector. Alternatively, TOV0 is cleared by writing a logic one to the flag. When the SREG I-bit, TOIE0 (Timer/Counter0 overflow interrupt enable), and TOV0 are set, the Timer/Counter0 Overflow interrupt is executed.

- **Bits 0 – ICF0: Timer/Counter0, Input Capture Flag**

This flag is set when a capture event occurs on the ICP0 pin. When the input capture register (ICR0) is set to be used as the TOP value, the ICF0 flag is set when the counter reaches the TOP value.

ICF0 is automatically cleared when the input capture interrupt vector is executed. Alternatively, ICF0 can be cleared by writing a logic one to its bit location.

Figure 16-1. Timer/Counter1 Block Diagram



16.2.1 Speed

The maximum speed of the Timer/Counter1 is 64MHz. However, if a supply voltage below 2.7V is used, it is recommended to use the low speed mode (LSM), because the Timer/Counter1 is not running fast enough on low voltage levels. In the low speed mode the fast peripheral clock is scaled down to 32MHz. For more details about the low speed mode, see Section 15.3.1 “PLLCSR – PLL Control and Status Register” on page 85.

16.2.2 Accuracy

The Timer/Counter1 is a 10-bit Timer/Counter module that can alternatively be used as an 8-bit Timer/Counter. The Timer/Counter1 registers are basically 8-bit registers, but on top of that there is a 2-bit high byte register (TC1H) that can be used as a common temporary buffer to access the two MSBs of the 10-bit Timer/Counter1 registers by the AVR CPU via the 8-bit data bus, if the 10-bit accuracy is used. Whereas, if the two MSBs of the 10-bit registers are written to zero the Timer/Counter1 is working as an 8-bit Timer/Counter. When reading the low byte of any 8-bit register the two MSBs are written to the TC1H register, and when writing the low byte of any 8-bit register the two MSBs are written from the TC1H register. Special procedures must be followed when accessing the 10-bit Timer/Counter1 values via the 8-bit data bus. These procedures are described in Section 16.10 “Accessing 10-Bit Registers” on page 105.

16.7.4 PWM6 Mode

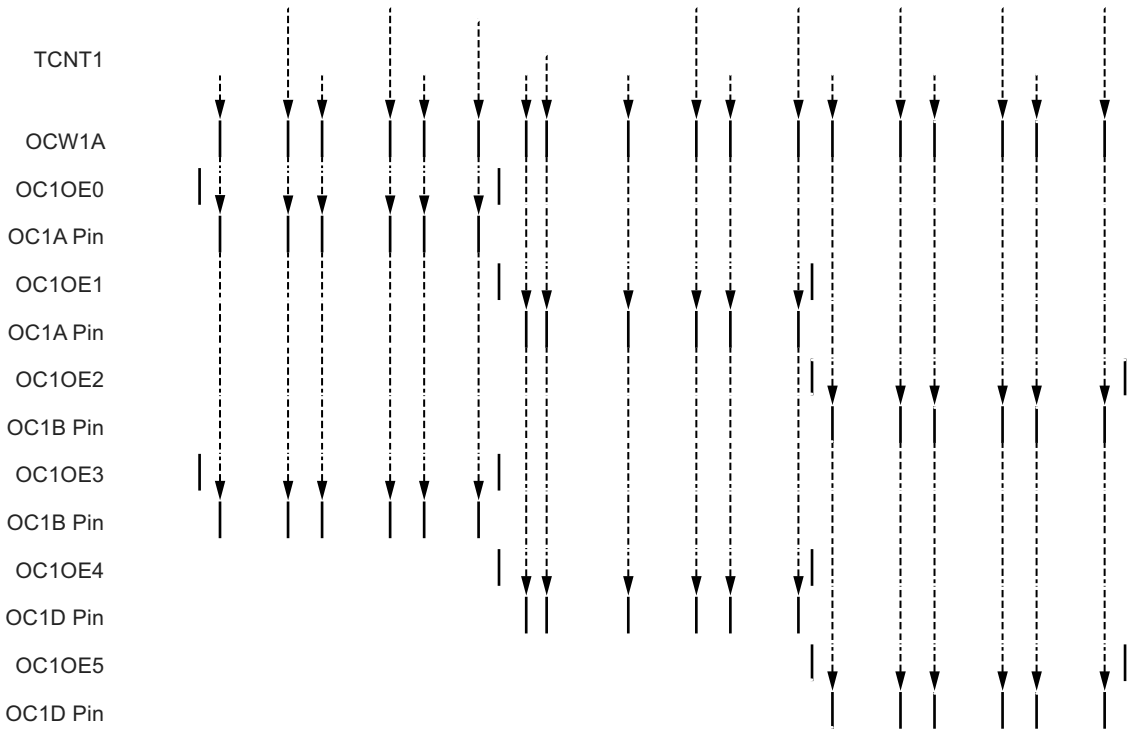
The PWM6 Mode (PWM1A = 1, WGM11 = 1 and WGM10 = x) provide PWM waveform generation option e.g. for controlling brushless DC (BLDC) motors. In the PWM6 mode the OCR1A register controls all six output compare waveforms as the same waveform output (OCW1A) from the waveform generator is used for generating all waveforms. The PWM6 mode also provides an output compare override enable register (OC1OE) that can be used with an instant response for disabling or enabling the output compare pins. If the output compare override enable bit is cleared, the actual value from the port register will be visible on the port pin.

The PWM6 mode provides two counter operation modes, a single-slope operation and a dual-slope operation. If the single-slope operation is selected (the WGM10 bit is set to 0), the counter counts from BOTTOM to TOP (defined as OCR1C) then restart from BOTTOM like in fast PWM Mode. The PWM waveform is generated by setting (or clearing) the waveform output (OCW1A) at the compare match between OCR1A and TCNT1, and clearing (or setting) the waveform output at the timer clock cycle the counter is cleared (changes from TOP to BOTTOM). The Timer/Counter overflow flag (TOV1) is set each time the counter reaches the TOP and, if the interrupt is enabled, the interrupt handler routine can be used for updating the compare value.

Whereas, if the dual-slope operation is selected (the WGM10 bit is set to 1), the counter counts repeatedly from BOTTOM to TOP (defined as OCR1C) and then from TOP to BOTTOM like in phase and frequency correct PWM mode. The PWM waveform is generated by setting (or clearing) the waveform output (OCW1A) at the compare match between OCR1A and TCNT1 when the counter increments, and clearing (or setting) the waveform output at the he compare match between OCR1A and TCNT1 when the counter decrements. The Timer/Counter overflow flag (TOV1) is set each time the counter reaches the BOTTOM and, if the interrupt is enabled, the interrupt handler routine can be used for updating the compare value.

The timing diagram for the PWM6 Mode in single-slope operation (WGM11 = 0) when the COM1A1:0 bits are set to “10” is shown in Figure 16-13. The counter is incremented until the counter value matches the TOP value. The counter is then cleared at the following timer clock cycle. The TCNT1 value is in the timing diagram shown as a histogram for illustrating the single-slope operation. The timing diagram includes output compare pins $\overline{OC1A}$ and OC1A, and the corresponding output compare override enable bits (OC1OE1..OC1OE0).

Figure 16-13. PWM6 Mode, Single-slope Operation, Timing Diagram



16.10 Accessing 10-Bit Registers

If 10-bit values are written to the TCNT1 and OCR1A/B/C/D registers, the 10-bit registers can be byte accessed by the AVR® CPU via the 8-bit data bus using two read or write operations. The 10-bit registers have a common 2-bit Timer/Counter1 high byte register (TC1H) that is used for temporary storing of the two MSBs of the 10-bit access. The same TC1H register is shared between all 10-bit registers. Accessing the low byte triggers the 10-bit read or write operation. When the low byte of a 10-bit register is written by the CPU, the high byte stored in the TC1H register, and the low byte written are both copied into the 10-bit register in the same clock cycle. When the low byte of a 10-bit register is read by the CPU, the high byte of the 10-bit register is copied into the TC1H register in the same clock cycle as the low byte is read.

To do a 10-bit write, the high byte must be written to the TC1H register before the low byte is written. For a 10-bit read, the low byte must be read before the high byte.

The following code examples show how to access the 10-bit timer registers assuming that no interrupts updates the TC1H register. The same principle can be used directly for accessing the OCR1A/B/C/D registers.

Assembly Code Example

```
...
; Set TCNT1 to 0x01FF
ldi    r17,0x01
ldi    r16,0xFF
out    TC1H,r17
out    TCNT1,r16
; Read TCNT1 into r17:r16
in     r16,TCNT1
in     r17,TC1H
...
```

C Code Example

```
unsigned int i;
...
/* Set TCNT1 to 0x01FF */
TC1H = 0x01;
TCNT1 = 0xFF;
/* Read TCNT1 into i */
i = TCNT1;
i |= ((unsigned int)TC1H << 8);
...
```

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.

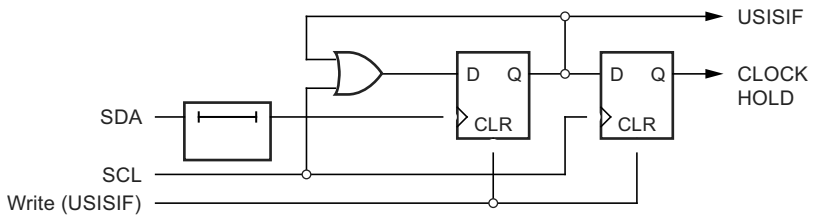
It is important to notice that accessing 10-bit registers are atomic operations. If an interrupt occurs between the two instructions accessing the 10-bit register, and the interrupt code updates the TC1H register by accessing the same or any other of the 10-bit timer registers, then the result of the access outside the interrupt will be corrupted. Therefore, when both the main code and the interrupt code update the TC1H register, the main code must disable the interrupts during the 16-bit access.

Referring to the timing diagram (Figure 17-5 on page 123), a bus transfer involves the following steps:

1. The a start condition is generated by the master by forcing the SDA low line while the SCL line is high (A). SDA can be forced low either by writing a zero to bit 7 of the shift register, or by setting the corresponding bit in the PORT register to zero. Note that the USI data register bit must be set to one for the output to be enabled. The slave device's start detector logic (Figure 17-6) detects the start condition and sets the USISIF flag. The flag can generate an interrupt if necessary.
2. In addition, the start detector will hold the SCL line low after the master has forced an negative edge on this line (B). This allows the slave to wake up from sleep or complete its other tasks before setting up the USI data register to receive the address. This is done by clearing the start condition flag and reset the counter.
3. The master set the first bit to be transferred and releases the SCL line (C). The slave samples the data and shift it into the USI data register at the positive edge of the SCL clock.
4. After eight bits are transferred containing slave address and data direction (read or write), the slave counter overflows and the SCL line is forced low (D). If the slave is not the one the master has addressed, it releases the SCL line and waits for a new start condition.
5. If the slave is addressed it holds the SDA line low during the acknowledgment cycle before holding the SCL line low again (i.e., the counter register must be set to 14 before releasing SCL at (D)). Depending of the R/W bit the master or slave enables its output. If the bit is set, a master read operation is in progress (i.e., the slave drives the SDA line) The slave can hold the SCL line low after the acknowledge (E).
6. Multiple bytes can now be transmitted, all in same direction, until a stop condition is given by the master (F). Or a new start condition is given.

If the slave is not able to receive more data it does not acknowledge the data byte it has last received. When the master does a read operation it must terminate the operation by force the acknowledge bit low after the last byte transmitted.

Figure 17-6. Start Condition Detector, Logic Diagram



17.3.5 Start Condition Detector

The start condition detector is shown in Figure 17-6 The SDA line is delayed (in the range of 50 to 300ns) to ensure valid sampling of the SCL line. The start condition detector is only enabled in two-wire mode.

The start condition detector is working asynchronously and can therefore wake up the processor from the power-down sleep mode. However, the protocol used might have restrictions on the SCL hold time. Therefore, when using this feature in this case the oscillator start-up time set by the CKSEL fuses (see Section 7.1 "Clock Systems and their Distribution" on page 24) must also be taken into the consideration. Refer to the USISIF bit description on page 126 for further details.

17.5.2 USIBR – USI Buffer Register

Bit	7	6	5	4	3	2	1	0	
0x10 (0x30)	MSB							LSB	USIBR
Read/Write	R	R	R	R	R	R	R	R	
Initial Value	0	0	0	0	0	0	0	0	

The content of the serial register is loaded to the USI buffer register when the transfer is completed, and instead of accessing the USI data register (the serial register) the USI data buffer can be accessed when the CPU reads the received data. This gives the CPU time to handle other program tasks too as the controlling of the USI is not so timing critical. The USI flags as set same as when reading the USIDR register.

17.5.3 USISR – USI Status Register

Bit	7	6	5	4	3	2	1	0	
0x0E (0x2E)	USISIF	USIOIF	USIPF	USIDC	USICNT3	USICNT2	USICNT1	USICNT0	USISR
Read/Write	R/W	R/W	R/W	R	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

The status register contains interrupt flags, line status flags and the counter value.

- **Bit 7 – USISIF: Start Condition Interrupt Flag**

When two-wire mode is selected, the USISIF flag is set (to one) when a start condition is detected. When output disable mode or three-wire mode is selected and (USICSx = 0b11 and USICLK = 0) or (USICS = 0b10 and USICLK = 0), any edge on the SCK pin sets the flag.

An interrupt will be generated when the flag is set while the USISIE bit in USICR and the global interrupt enable flag are set. The flag will only be cleared by writing a logical one to the USISIF bit. Clearing this bit will release the start detection hold of USCL in two-wire mode.

A start condition interrupt will wakeup the processor from all sleep modes.

- **Bit 6 – USIOIF: Counter Overflow Interrupt Flag**

This flag is set (one) when the 4-bit counter overflows (i.e., at the transition from 15 to 0). An interrupt will be generated when the flag is set while the USIOIE bit in USICR and the global interrupt enable flag are set. The flag will only be cleared if a one is written to the USIOIF bit. Clearing this bit will release the counter overflow hold of SCL in two-wire mode.

A counter overflow interrupt will wake up the processor from Idle sleep mode.

- **Bit 5 – USIPF: Stop Condition Flag**

When Two-wire mode is selected, the USIPF Flag is set (one) when a stop condition is detected. The flag is cleared by writing a one to this bit. Note that this is not an interrupt flag. This signal is useful when implementing two-wire bus master arbitration.

- **Bit 4 – USIDC: Data Output Collision**

This bit is logical one when bit 7 in the USI data register differs from the physical pin value. The flag is only valid when two-wire mode is used. This signal is useful when implementing two-wire bus master arbitration.

- **Bits 3:0 – USICNT3..0: Counter Value**

These bits reflect the current 4-bit counter value. The 4-bit counter value can directly be read or written by the CPU.

The 4-bit counter increments by one for each clock generated either by the external clock edge detector, by a Timer/Counter0 compare match, or by software using USICLK or USITC strobe bits. The clock source depends of the setting of the USICS1..0 bits. For external clock operation a special feature is added that allows the clock to be generated by writing to the USITC strobe bit. This feature is enabled by write a one to the USICLK bit while setting an external clock source (USICS1 = 1).

Note that even when no wire mode is selected (USIWM1..0 = 0) the external clock input (USCK/SCL) are can still be used by the counter.

- **Bit 3:2 – USICS1:0: Clock Source Select**

These bits set the clock source for the USI data register and counter. The data output latch ensures that the output is changed at the opposite edge of the sampling of the data input (DI/SDA) when using external clock source (USCK/SCL). When software strobe or Timer/Counter0 compare match clock option is selected, the output latch is transparent and therefore the output is changed immediately. Clearing the USICS1:0 bits enables software strobe option. When using this option, writing a one to the USICLK bit clocks both the USI data register and the counter. For external clock source (USICS1 = 1), the USICLK bit is no longer used as a strobe, but selects between external clocking and software clocking by the USITC strobe bit.

Table 17-2 on page 128 shows the relationship between the USICS1..0 and USICLK setting and clock source used for the USI data register and the 4-bit counter.

Table 17-2. Relations between the USICS1..0 and USICLK Setting

USICS1	USICS0	USICLK	USI Data Register Clock Source	4-bit Counter Clock Source
0	0	0	No Clock	No Clock
0	0	1	Software clock strobe (USICLK)	Software clock strobe (USICLK)
0	1	X	Timer/Counter0 compare match	Timer/Counter0 compare match
1	0	0	External, positive edge	External, both edges
1	1	0	External, negative edge	External, both edges
1	0	1	External, positive edge	Software clock strobe (USITC)
1	1	1	External, negative edge	Software clock strobe (USITC)

- **Bit 1 – USICLK: Clock Strobe**

Writing a one to this bit location strobes the USI data register to shift one step and the counter to increment by one, provided that the USICS1..0 bits are set to zero and by doing so the software clock strobe option is selected. The output will change immediately when the clock strobe is executed, i.e., in the same instruction cycle. The value shifted into the USI data register is sampled the previous instruction cycle. The bit will be read as zero.

When an external clock source is selected (USICS1 = 1), the USICLK function is changed from a clock strobe to a clock select register. Setting the USICLK bit in this case will select the USITC strobe bit as clock source for the 4-bit counter (see Table 17-2).

- **Bit 0 – USITC: Toggle Clock Port Pin**

Writing a one to this bit location toggles the USCK/SCL value either from 0 to 1, or from 1 to 0. The toggling is independent of the setting in the data direction register, but if the PORT value is to be shown on the pin the DDB2 must be set as output (to one). This feature allows easy clock generation when implementing master devices. The bit will be read as zero.

When an external clock source is selected (USICS1 = 1) and the USICLK bit is set to one, writing to the USITC strobe bit will directly clock the 4-bit counter. This allows an early detection of when the transfer is done when operating as a master device.

17.5.5 USIPP – USI Pin Position

Bit	7	6	5	4	3	2	1	0	
0x11 (0x31)	-	-	-	-	-	-	-	**USIPOS**	**USIPP**
Read/Write	R	R	R	R	R	R	R	R/W	
Initial Value	0	0	0	0	0	0	0	0	

- **Bits 7:1 – Res: Reserved Bits**

These bits are reserved bits in the ATtiny261/461/861 and always reads as zero.

- **Bit 0 – USIPOS: USI Pin Position**

Setting this bit to one changes the USI pin position. As default pins PB2..PB0 are used for the USI pin functions, but when writing this bit to one the USIPOS bit is set the USI pin functions are on pins PA2..PA0.

20.4 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.

20.5 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.

20.6 Register Description

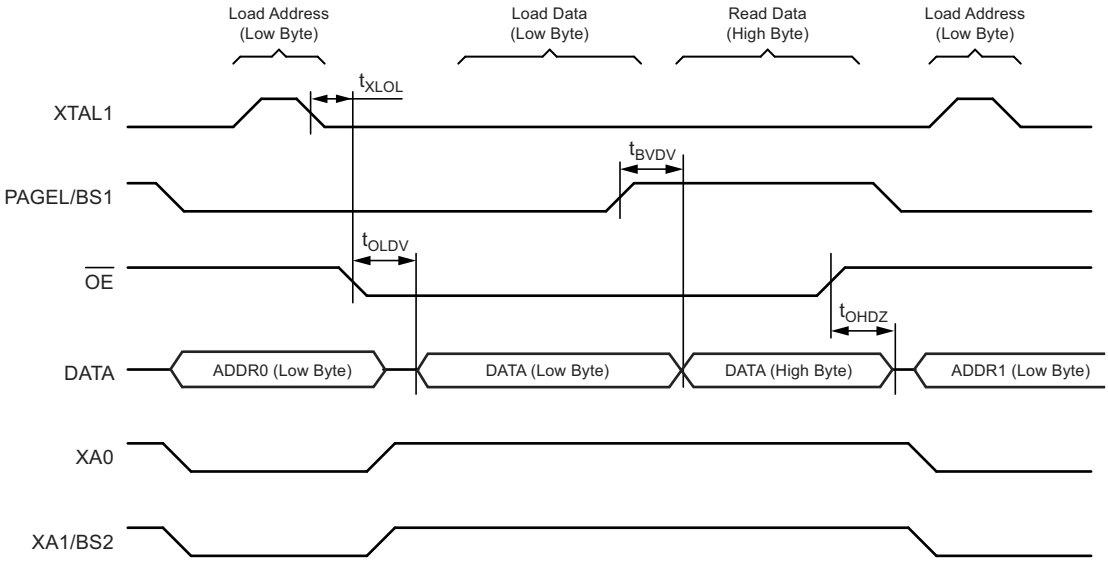
The following section describes the registers used with the debugWire.

20.6.1 DWDR – debugWire Data Register

Bit	7	6	5	4	3	2	1	0	
0x20 (0x40)	**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.

Figure 23-5. Parallel Programming Timing, Reading Sequence (within the Same Page) with Timing Requirements(1)



Note: 1. The timing requirements shown in Figure 23-3 on page 177 (i.e., t_{DVXH}, t_{XHXL}, and t_{XLDX}) also apply to reading operation.

Table 23-7. Parallel Programming Characteristics, V_{CC} = 5V ±10%

Parameter	Symbol	Min	Typ	Max	Unit
Programming enable voltage	V_{PP}	11.5		12.5	V
Programming enable current	I_{PP}			250	µA
Data and control valid before XTAL1 high	t_{DVXH}	67			ns
XTAL1 low to XTAL1 high	t_{XLXH}	200			ns
XTAL1 pulse width high	t_{XHXL}	150			ns
Data and control hold after XTAL1 low	t_{XLDX}	67			ns
XTAL1 low to $\overline{WR}$ low	t_{XLWL}	0			ns
BS1 valid before PAGEL high	t_{BVPH}	67			ns
PAGEL pulse width high	t_{PHPL}	150			ns
BS1 hold after PAGEL low	t_{PLBX}	67			ns
BS2/1 hold after $\overline{WR}$ low	t_{WLBX}	67			ns
PAGEL low to $\overline{WR}$ low	t_{PLWL}	67			ns
BS1 valid to $\overline{WR}$ low	t_{BVWL}	67			ns
$\overline{WR}$ pulse width low	t_{WLWH}	150			ns
$\overline{WR}$ low to RDY/$\overline{BSY}$ low	t_{WLRL}	0		1	µs
$\overline{WR}$ low to RDY/$\overline{BSY}$ high(1)	t_{WLRH}	3.7		4.5	ms
$\overline{WR}$ low to RDY/$\overline{BSY}$ high for chip erase(2)	t_{WLRH_CE}	7.5		9	ms
XTAL1 low to $\overline{OE}$ low	t_{XLOL}	0			ns
BS1 valid to DATA valid	t_{BVDV}	0		250	ns
$\overline{OE}$ low to DATA valid	t_{OLDV}			250	ns
$\overline{OE}$ high to DATA tri-stated	t_{OHDZ}			250	ns

Notes: 1. t_{WLRH} is valid for the write flash, write EEPROM, write fuse bits and write lock bits commands.
2. t_{WLRH_CE} is valid for the chip erase command.

26. Instruction Set Summary (Continued)

Mnemonics	Operands	Description	Operation	Flags	#Clocks
CLI		Global interrupt disable	I ← 0	I	1
SES		Set signed test flag	S ← 1	S	1
CLS		Clear signed test flag	S ← 0	S	1
SEV		Set twos complement overflow.	V ← 1	V	1
CLV		Clear twos complement overflow	V ← 0	V	1
SET		Set T in SREG	T ← 1	T	1
CLT		Clear T in SREG	T ← 0	T	1
SEH		Set half carry flag in SREG	H ← 1	H	1
CLH		Clear half carry flag in SREG	H ← 0	H	1
DATA Transfer Instructions					
MOV	Rd, Rr	Move between registers	Rd ← Rr	None	1
MOVW	Rd, Rr	Copy register word	Rd+1:Rd ← Rr+1:Rr	None	1
LDI	Rd, K	Load immediate	Rd ← K	None	1
LD	Rd, X	Load indirect	Rd ← (X)	None	2
LD	Rd, X+	Load indirect and post-inc.	Rd ← (X), X ← X + 1	None	2
LD	Rd, - X	Load indirect and pre-dec.	X ← X - 1, Rd ← (X)	None	2
LD	Rd, Y	Load indirect	Rd ← (Y)	None	2
LD	Rd, Y+	Load indirect and post-inc.	Rd ← (Y), Y ← Y + 1	None	2
LD	Rd, - Y	Load indirect and pre-dec.	Y ← Y - 1, Rd ← (Y)	None	2
LDD	Rd,Y+q	Load indirect with displacement	Rd ← (Y + q)	None	2
LD	Rd, Z	Load indirect	Rd ← (Z)	None	2
LD	Rd, Z+	Load indirect and post-inc.	Rd ← (Z), Z ← Z+1	None	2
LD	Rd, -Z	Load indirect and pre-dec.	Z ← Z - 1, Rd ← (Z)	None	2
LDD	Rd, Z+q	Load indirect with displacement	Rd ← (Z + q)	None	2
LDS	Rd, k	Load direct from SRAM	Rd ← (k)	None	2
ST	X, Rr	Store indirect	(X) ← Rr	None	2
ST	X+, Rr	Store indirect and post-inc.	(X) ← Rr, X ← X + 1	None	2
ST	- X, Rr	Store indirect and pre-dec.	X ← X - 1, (X) ← Rr	None	2
ST	Y, Rr	Store indirect	(Y) ← Rr	None	2
ST	Y+, Rr	Store indirect and post-inc.	(Y) ← Rr, Y ← Y + 1	None	2
ST	- Y, Rr	Store indirect and pre-dec.	Y ← Y - 1, (Y) ← Rr	None	2
STD	Y+q,Rr	Store indirect with displacement	(Y + q) ← Rr	None	2
ST	Z, Rr	Store indirect	(Z) ← Rr	None	2
ST	Z+, Rr	Store indirect and post-inc.	(Z) ← Rr, Z ← Z + 1	None	2
ST	-Z, Rr	Store indirect and pre-dec.	Z ← Z - 1, (Z) ← Rr	None	2
STD	Z+q,Rr	Store indirect with displacement	(Z + q) ← Rr	None	2
STS	k, Rr	Store direct to SRAM	(k) ← Rr	None	2
LPM		Load program memory	R0 ← (Z)	None	3
LPM	Rd, Z	Load program memory	Rd ← (Z)	None	3
LPM	Rd, Z+	Load program memory and post-inc	Rd ← (Z), Z ← Z+1	None	3
SPM		Store program memory	(z) ← R1:R0	None	

27. Ordering Information

Table 27-1. Available Product Offering

Ordering Code[2]	Speed (MHz)[3]	Power Supply (V)	Package[1]	Operation Range
ATtiny261-15SZ	16	2.7 - 5.5	TG	Automotive (–40° to +125°C)
ATtiny261-15MZ	16	2.7 - 5.5	PN	Automotive (–40° to +125°C)
ATtiny261-15XZ	16	2.7 - 5.5	6G	Automotive (–40° to +125°C)
ATtiny261-15MAZ	16	2.7 - 5.5	PC	Automotive (–40° to +125°C)
ATtiny461-15SZ	16	2.7 - 5.5	TG	Automotive (–40° to +125°C)
ATtiny461-15MZ	16	2.7 - 5.5	PN	Automotive (–40° to +125°C)
ATtiny461-15XZ	16	2.7 - 5.5	6G	Automotive (–40° to +125°C)
ATtiny461-15MAZ	16	2.7 - 5.5	PC	Automotive (–40° to +125°C)
ATtiny861-15SZ	16	2.7 - 5.5	TG	Automotive (–40° to +125°C)
ATtiny861-15MZ	16	2.7 - 5.5	PN	Automotive (–40° to +125°C)
ATtiny861-15XZ	16	2.7 - 5.5	6G	Automotive (–40° to +125°C)
ATtiny861-15MAZ	16	2.7 - 5.5	PC	Automotive (–40° to +125°C)

Notes:
1. This device can also be supplied in wafer form. Please contact your local Atmel sales office for detailed ordering information and minimum quantities.
2. Pb-free packaging, complies to the European Directive for Restriction of Hazardous Substances (RoHS directive). Also halide free and fully green.
3. For speed versus V_{CC}, see Figure 23.3 on page 173.

Table 27-2. Package Types

Package Type	
PN	32-pad, 5.0x5.0mm body, lead pitch 0.50mm, quad flat no lead package (QFN)
TG	20-lead, 0.300" wide body lead, plastic gull wing small outline package (SOIC)
6G	20-lead, 4.4x6.5mm body, 0.65mm pitch, lead length: 0.6mm Thin shrink small outline package (TSSOP)
PC	20-lead, 4.0x4.0mm body, 0.50mm pitch, quad flat no lead package (QFN)

30. Revision History

Please note that the following page numbers referred to in this section refer to the specific revision mentioned, not to this document.

Revision No.	History
7753G-AVR-06/14	• Put datasheet in the latest template
7753F-AVR-01/11	• Trigger reference levels of the analog comparator errata added
7753E-AVR-06/10	• Ordering Information updated • DC Characteristics table updated
7753D-AVR-11/09	• Ordering Information updated • QFN pinout added • Internal RC Oscillator Accuracy updated
7753C-AVR-07/09	• QFN package added • ADC characteristics updated • Temps sensor updated • Typical characteristics updated
7753B-AVR-08/08	• Added 6G product offering to Ordering Information.
7753A-AVR-11/07	• First datasheet draft - initial automotive version. Started from industrial datasheet doc2588 rev.B - 01/07