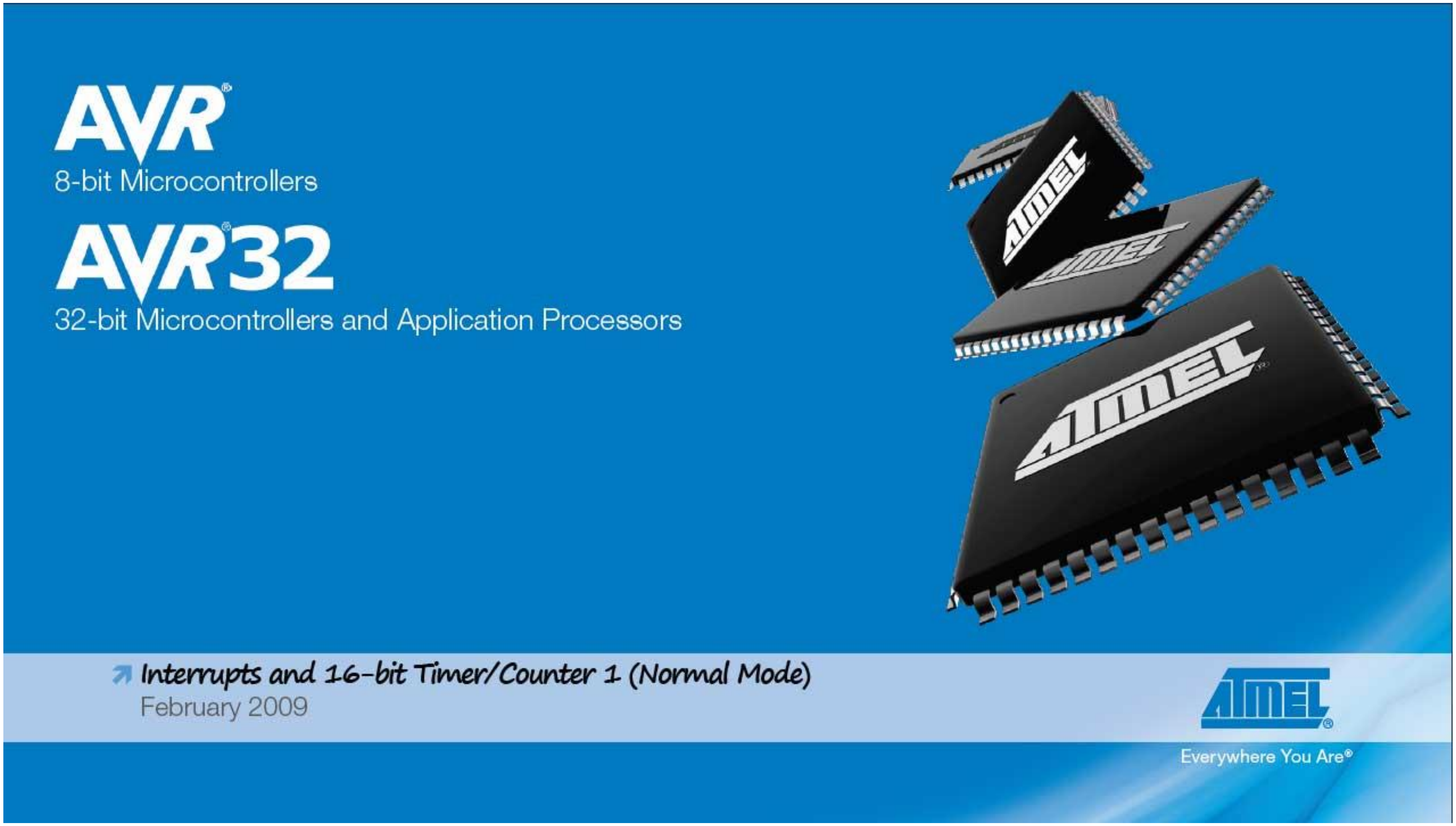


AVR®

8-bit Microcontrollers

AVR®32

32-bit Microcontrollers and Application Processors

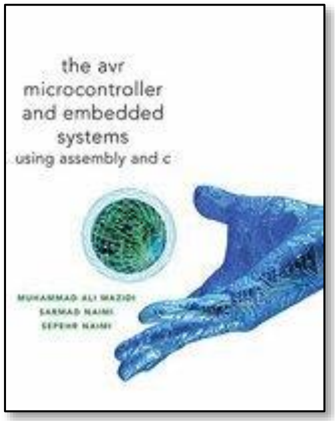
Interrupts and 16-bit Timer/Counter 1 (Normal Mode)

February 2009

Everywhere You Are®

Atmel AVR Timers and Interrupts

Reading



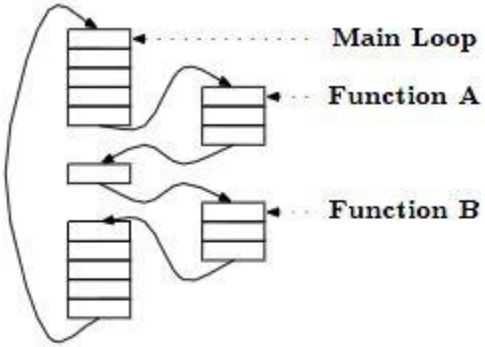
The AVR Microcontroller and Embedded Systems using Assembly and C)
by Muhammad Ali Mazidi, Sarmad Naimi, and Sepehr Naimi

Chapter 10: AVR Interrupt Programming in Assembly and C

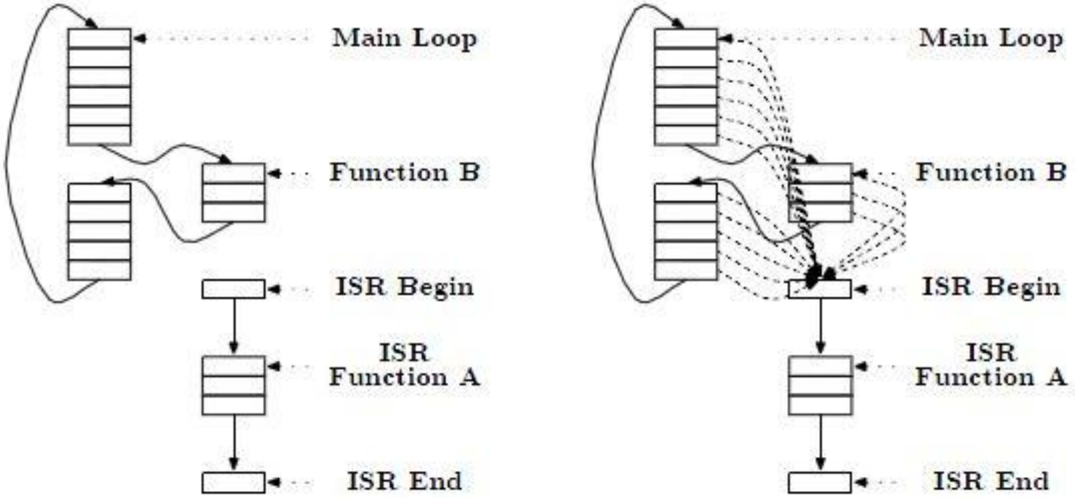
Section 10.2: Programming Timer Interrupts

Contents

INTERRUPT BASICS – REVIEW -



- A microcontroller normally executes instructions in an orderly fetch-execute sequence as dictated by a user-written program.
- However, a microcontroller must also be ready to handle unscheduled, events that might occur inside or outside the microcontroller.
- The interrupt system onboard a microcontroller allows it to respond to these internally and externally generated events. By definition we do not know when these events will occur.
- When an interrupt event occurs, the microcontroller will normally complete the instruction it is currently executing and then transition program control to an Interrupt Service Routine (ISR). These ISR, which handles the interrupt.
- Once the ISR is complete, the microcontroller will resume processing where it left off before the interrupt event occurred.



(a) Program with ISR (b) ISR called from anywhere

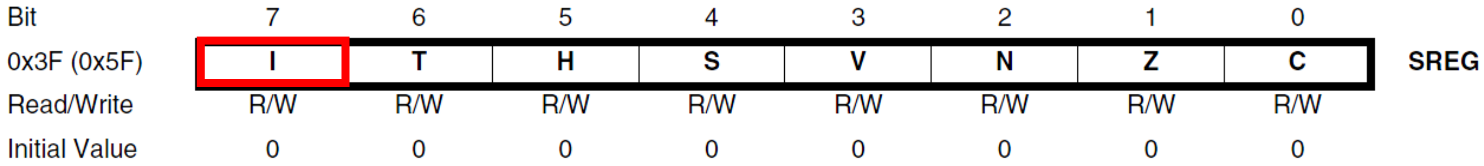
ATmega32U4 Interrupt Vector Table

Vector No	Program Address	Source	Interrupt Definition	Arduino/C++ ISR() Macro Vector Name
1	0x0000	RESET	Reset	
2	0x0002	INT0	External Interrupt Request 0 (pin D0)	(INT0_vect)
3	0x0004	INT1	External Interrupt Request 1 (pin D1)	(INT1_vect)
4	0x0006	INT2	External Interrupt Request 2 (pin D2)	(INT2_vect)
5	0x0008	INT3	External Interrupt Request 3 (pin D3)	(INT3_vect)
6	0x000A	Reserved	Reserved	
7	0x000C	Reserved	Reserved	
8	0x000E	INT6	External Interrupt Request 6 (pin E6)	(INT6_vect)
9	0x0010	Reserved		
10	0x0012	PCINT0	Pin Change Interrupt Request 0 (pins PB7 to PB0)	(PCINT0_vect)
11	0x0014	USB General	USB General Interrupt request	(USB_GENERAL_vect)
12	0x0016	USB Endpoint	USB Endpoint Interrupt request	(USB_ENDPOINT_vect)
13	0x0018	WDT	Watchdog Time-out Interrupt	(WDT_vect)
14	0x001A	Reserved	Reserved	
15	0x001C	Reserved	Reserved	
16	0x001E	Reserved	Reserved	
17	0x0020	TIMER1 CAPT	Timer/Counter1 Capture Event	(TIMER1_CAPT_vect)
18	0x0022	TIMER1 COMPA	Timer/Counter1 Compare Match A	(TIMER1_COMPA_vect)
19	0x0024	TIMER1 COMPB	Timer/Counter1 Compare Match B	(TIMER1_COMPB_vect)
20	0x0026	TIMER1 COMPC	Timer/Counter1 Compare Match C	(TIMER1_COMPC_vect)
21	0x0028	TIMER1 OVF	Timer/Counter1 Overflow (see note)	(TIMER1_OVF_vect)
↓				
41	0x0050	TIMER4 COMPD	Timer/Counter4 Compare Match D	(TIMER4_COMPD_vect)
42	0x0052	TIMER4 OVF	Timer/Counter4 Overflow	(TIMER4_OVF_vect)
43	0x0054	TIMER4 FPF	Timer/Counter4 Fault Protection Interrupt	(TIMER4_FPF_vect)

Note: Timer 1 not available when servos are attached to the 3DoT board.

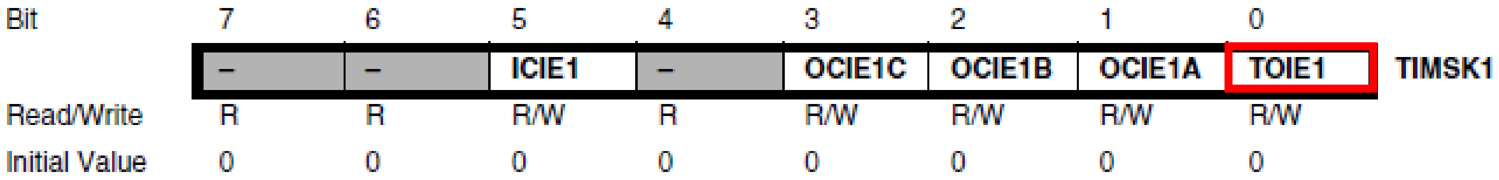
ATMEGA32U4 ENABLING AN INTERRUPT – TIMER/COUNTER 1

- All interrupts are assigned individual enable bits which must be written to logic one together with the Global Interrupt Enable bit in the Status Register (SREG) in order to enable the interrupt.



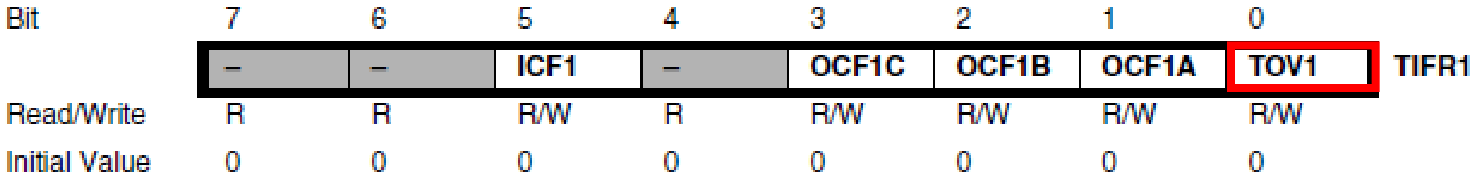
Bit	7	6	5	4	3	2	1	0	
0x3F (0x5F)	I	T	H	S	V	N	Z	C	SREG
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

- For example, to allow the Timer/Counter 1 Overflow flag (TOV1) to generate an interrupt you would set the Timer/Counter 1 Overflow Interrupt Enable (TOIE1) bit 0 in register TIMSK1 at extended I/O SRAM address 0x6F.



Bit	7	6	5	4	3	2	1	0	
	–	–	ICIE1	–	OCIE1C	OCIE1B	OCIE1A	TOIE1	TIMSK1
Read/Write	R	R	R/W	R	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

- When Timer/Counter 1 Overflows (0xFFFF → 0x0000) the TOV1 bit 0 in register TIFR1 at I/O address 0x16.

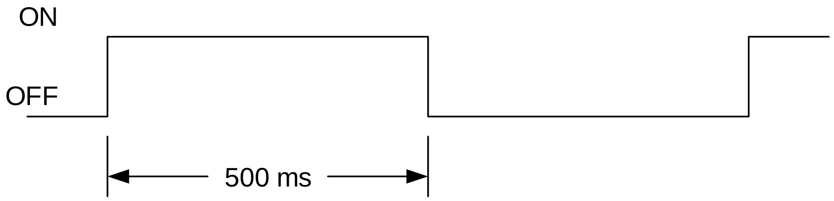


Bit	7	6	5	4	3	2	1	0	
	–	–	ICF1	–	OCF1C	OCF1B	OCF1A	TOV1	TIFR1
Read/Write	R	R	R/W	R	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

- With global interrupt I-bit set and Timer/Counter 1's Overflow Interrupt Enable TOIE1-bit set, when the Overflow TOV1-bit is set an interrupt will be generated and the Program Counter (PC) will be vectored to **Flash Program Memory address 0x0028** (see IVT Table on previous page). The AVR processor starts running the ISR.

- The TOV1 flag is automatically cleared at the beginning of the interrupt service routine. Alternatively, if you are polling the flag, it can be cleared by writing a logical one to it. The TIFR1 register is within the I/O address range (0x00 to 0x1F) of the Set Bit in I/O Register (SBI) Instruction.

Timer/Counter 1 Normal Mode – Design Example



- In this design example, we want to write a 500 msec delay routine assuming a system clock frequency of 8.000 MHz and a prescale divisor of 64.

Solution

1. Divide desired time delay by t_{clkT1} where $t_{clkT1} = 64/f_{clkI/O}$ = 64 / 8.000 MHz = 8 µsec/tic

   500msec / 8 µs/tic = 62,500 tics

   short-cut: TCNT1H = high(-62,500) and TCNT1L = low(-62,500)

2. Subtract 65,536 – step 1

   65,536 – 62,500 = 3,036

3. Convert step 2 to hexadecimal.

   3,036 = 0x0BDC

   For our example TCNT1H = 0x0B and TCNT1L = 0xDC

ATMEGA32U4 ENABLING TIMER/COUNTER 1 INTERRUPT

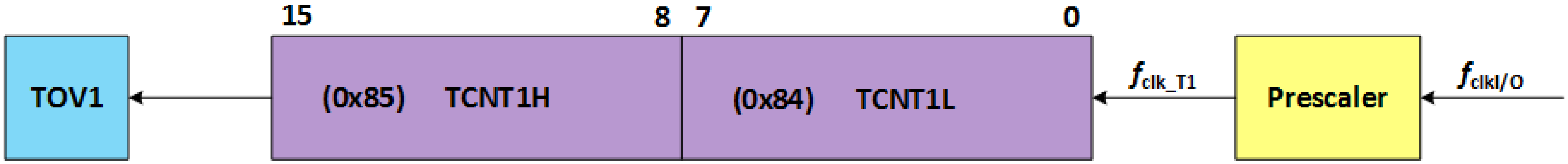
```
// Jump over and Setup the Interrupt Vector Table
RST_VECT:
  rjmp   reset

// TIMER1 OVF vector = 0x0028, Sect 9.1 Interrupt Vectors in ATmega32U4
.ORG OVF1addr
  jmp    TOVF1_ISR  // Section 4.8 Reset and Interrupt Handling

; Set prescale and start Timer/Counter1
   ldi   r16,(1<<CS11)|(1<<CS10) // prescale of 64 Sect 14.10.4
   sts   TCCR1B,r16   // Table 14-6 Clock Select Bit Description

   ldi   r16,0x0B     // load value high byte (Sect 14.10.7)
   sts   TCNT1H,r16
   ldi   r16,0xDC     // load value low byte
   sts   TCNT1L,r16
// Enable Local and Global Interrupts
   ldi   r16,(1<<TOIE1)  // Enable interrupts for Timer1 OVF
   sts   TIMSK1,r16      // TIMSK1 Bit 0 - TOIE1
   sei                   // Global Interrupt Enable
```

THE INTERRUPT SERVICE ROUTINE (ISR)

```
; -- Timer/Counter 1 Overflow Interrupt Service Routine --
; Called on Timer/Counter1 overflow TOV1
; TOV1 flag automatically cleared by AVR on interrupt
TOVF1_ISR:
  push  reg_F
  in    reg_F,SREG
  push  r16
; --- 250 msec ---
  ldi   r16,0x0B         // load value high byte 0x0B
  sts   TCNT1H,r16
  ldi   r16,0xDC         // load value low byte 0xDC
  sts   TCNT1L,r16
; --- Blink Discrete LED ---
  in    r16, PORTF
  ldi   r17,0b10000000
  eor   r16, r17         // toggle LED
  out   PORTF, r16
  pop   r16
  out   SREG,reg_F
  pop   reg_F
  reti
; -------------------------------------------------------
```

Normal Mode (WGM 1 bits 3:0 = 0000_2)

7	6	5	4	3	2	1	0	
COM1A1	COM1A0	COM1B1	COM1B0	COM1C1	COM1C0	WGM11	WGM10	TCCR1A
ICNC1	ICES1		WGM13	WGM12	CS12	CS11	CS10	TCCR1B
		ICF1		OCF1C	OCF1B	OCF1A	TOV1	TIFR1



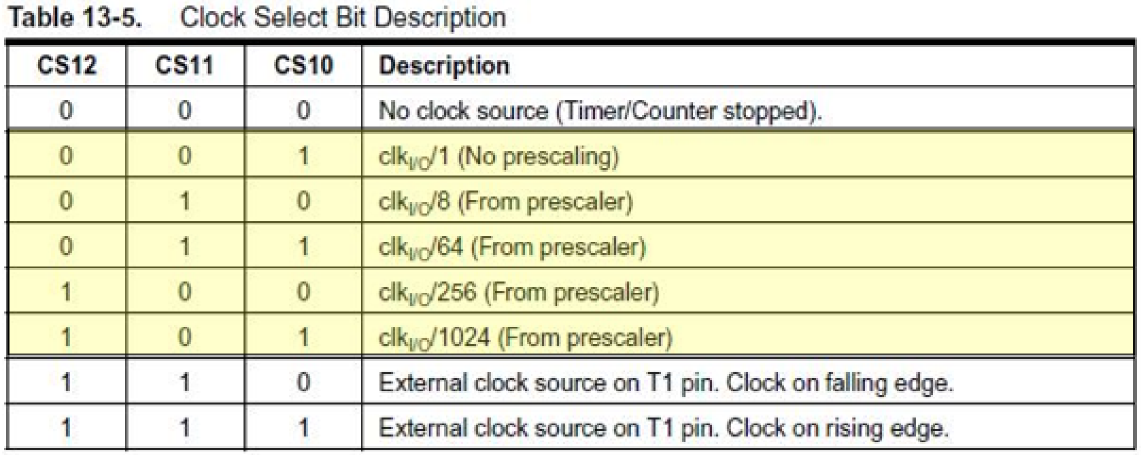


Table 13-5. Clock Select Bit Description

CS12	CS11	CS10	Description
0	0	0	No clock source (Timer/Counter stopped).
0	0	1	$clk_{I/O}$/1 (No prescaling)
0	1	0	$clk_{I/O}$/8 (From prescaler)
0	1	1	$clk_{I/O}$/64 (From prescaler)
1	0	0	$clk_{I/O}$/256 (From prescaler)
1	0	1	$clk_{I/O}$/1024 (From prescaler)
1	1	0	External clock source on T1 pin. Clock on falling edge.
1	1	1	External clock source on T1 pin. Clock on rising edge.