Section-8 TC1 (Timer/Counter 1) Module in Normal Mode Operation

There are 3 TC Modules inside the ATmega328P MCU — 8-bit TC0, 16-bit TC1, and 8-bit TC2. These TCn (n = 0, 1, 2) Modules can be used to generate various timing functions, events, and waveforms. In this Section, we will discuss the normal mode operation of TC1 Module

8.1 Introduction

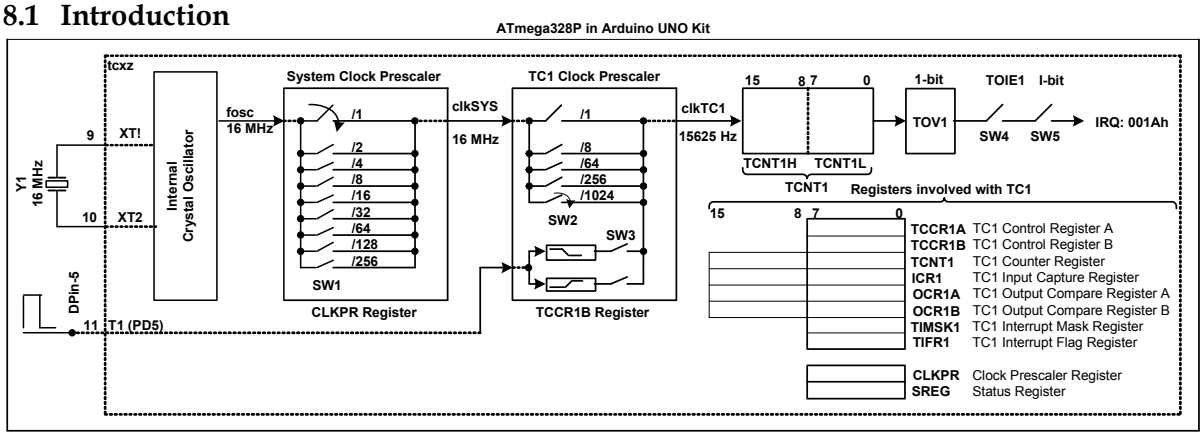


Figure-8.1: *Internal architecture of TC1 Module in Normal Mode operation*

(1) The TC1 or TCNT1 is a 16-bit register composed of two 8-bit registers — the lower TCNT1L and the higher TCNT1H. The driving clock (clkTC1) for TCNT1 may come either from internal oscillator via SW1/SW2 or from external source via Pin-11/SW3. The SW1, SW2, and SW3 are software switches whose open/close conditions are controlled by software instructions. Let us note that SW1 is at position /1 in Arduino UNO; hence, the clkSYS is 16 MHz.

(2) When TCNT1 counts clock pulses coming from internal oscillator, we say that the TCNT1 is working as **Timer**. We say that the TCNT1 is working a **Counter** when it receives the clocking (driving) pulses from external source. As a counter, the TCNT1 can respond to either rising edge or the falling edge of the incoming pulses. Once the TCNT1 gets started by the MCU, the TCNT1 keeps counting incoming pulses on its own; there is no intervention from the MCU side. However, we will use the words Timer and Counter interchangeably to mean the same thing.

(3) **(a)** **BOTTOM:** The TCNT1 reaches the BOTTOM when its content becomes 0x0000.

(b) **MAX:** The counter reaches MAX when its content becomes 0xFFFF.

(c) **TOP:** The counter reaches the TOP when its content becomes equal to the highest value in the count sequence. The TOP value can be assigned one of the fixed values: 0xFF, 0x01FF, or 0x03FF or to the value stored in the OCR1A or ICR1 register. The assignment is dependent on the mode of operation.

(3) To do a 16-bit write into TCNT1, the high byte must be written first into TCNT1H and then the low byte into TCNT`L. To do a 16-bit read, the low byte must be read first from TCNT1l and then the high byte must be read from TCNT1H. For example:

(a) Storing 0x1234 into TCNT1

```
TCNT1H = 0x12;
TCNT1L = 0x34;
```

Or

```
TCNT1 = 0x1234;     //high byte is stored first and then low byte
```

(b) Reading content of TCNT1

```
byte x0 = TCNT1L;
byte x1 = TCNT1H;
```

Or

```
int x = TCNT1; //low byte is retrieved first and then high byte
```

8.2 Normal Mode Operation of TC1/TCNT1 Module

(1) In this mode of operation, the TCNT1 always works as an up counter register. The counting may begin from 0x0000 (called BOTTOM) and reaches to 0xFFFF (called MAX) without any counter clear operation. When TCNT1 counts one more pulse after reaching at MAX value, the content will change from all 1s to all 0s (FFFF + 1 = 1 0000). This event is known as **rollover** or **overflow.** The time required to arrive at the overflow event is known as 'overflow time'. The counter can be loaded at any time without disturbing the counting operation.

(2) When the overflow event occurs, the TOV1 bit/flag (TC1 Overflow Flag) assumes HIGH state. The user program may continuously monitor this flag to detect the occurrence of overflow event. This method of knowing the value of TOV1 flag is known as **polling** process. TOV1 flag is brought into reset state by writing HIGH into this bit position.

(3) The occurrence of overflow event can also be known through interrupt process. If SW4/SW5 (Fig-8.1) are kept at closed positions in the setup() function, the HIGH value of TOV1 flag will automatically interrupt the MCU; as a result, the MCU will jump to the following ISR (interrupt subroutine). The TOV1 flag automatically comes into reset state when the MCU jumps to the ISR.

```
ISR(TIMER1_OVF_vect)
{

}
```

(4) It is possible that we pre-load some value into TCNT1 register and then command the TC1 to count up from that pre-set value. Under this condition, the overflow time will be shorter than what we would get if the TC1 would start counting from 0x0000. The overflow time can be changed by changing the values of both clkTC1 frequency and pre-set quantity.

8.3 Example - 1

Estimate values of clkTC1 and preset quantity so that the overflow time of TCNT1 is 1 sec.

1. overflowCount = presetCount + actualCount.
2. Choose a division factor 1024 for TC1 Clock Prescaler to get clkTC1 = 16 MHz/1204 = 15625 Hz.
3. Overflow time = 1 sec.
4. From Step-1:
==> actualCount = overflowCount – presetCount
==> presetCount = overflowCount – actualCount
==> presetCount = 0x10000 – 15625 (pulses counted by TCNT1 in 1 sec time)
==> presetCount = 0x10000 – 0x3D09
==> presetCount = 0xC2F7

8.4 Registers involved in the Normal Mode Operation of TC1 Module

SREG – AVR Status Register

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	

TCCR1A – Timer/Counter1 Control Register A

Bit	7	6	5	4	3	2	1	0	
(0x80)	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	

TCCR1B – Timer/Counter1 Control Register B

Bit	7	6	5	4	3	2	1	0	
(0x81)	ICNC1	ICES1	–	WGM13	WGM12	CS12	CS11	CS10	TCCR1B
Read/Write	R/W	R/W	R	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

TCNT1H and TCNT1L – Timer/Counter1

Bit	7	6	5	4	3	2	1	0	
(0x85)	TCNT1[15:8]								TCNT1H
(0x84)	TCNT1[7:0]								TCNT1L
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	

TIMSK1 – Timer/Counter1 Interrupt Mask Register

Bit	7	6	5	4	3	2	1	0	
(0x6F)	–	–	ICIE1	–	–	OCIE1B	OCIE1A	TOIE1	TIMSK1
Read/Write	R	R	R/W	R	R	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

TIFR1 – Timer/Counter1 Interrupt Flag Register

Bit	7	6	5	4	3	2	1	0	
0x16 (0x36)	–	–	ICF1	–	–	OCF1B	OCF1A	TOV1	TIFR1
Read/Write	R	R	R/W	R	R	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

8.5 Exercises

1 CS12 – CS10 bits of TCCR1B Register determine the division factor of SW2 of Fig-8.1. Consult data sheets of ATmega328P MCU and fill up the following Table.

CS12	CS11	CS10	Division Factor	clkADC
0	0	0	--	No clock source; TC1 is stopped
0	0	1	1	clkADC = f16 MHz/1 = 16 MHz
0	1	0	8	clkADC = 16 MHz/8 = 2 MHz
0	1	1	64	clkADC = 16 MHz/64 = 250 kHz
1	0	0	256	clkADC = 16 MHz/256 = 62500 Hz
1	0	1	1024	clkADC = 15 MHz/1024 = 15625 Hz

2 Data sheets say that the TOV1-bit of TIFR1 Register can be cleared (assumes LOW) by writing HIGH at this bit position. Write all possible codes to clear the TOV1 bit.

3 Under what condition will TOV1 bit assume HIGH state?
Assume that TCNT1 is counting the driving pulses and has reached to full count of 0xFFFF (all 1s). Now, the arrival of the next driving pulse will turn the contents of TCNT1 from all 1s to all 0s. This event will deposit HIGH value into TOV1 bit indicating that the rollover/overflow event has occurred.

4 Under what conditions, the transition of TOV1 bit from LOW to HIGH state will interrupt the MCU.
The active state of TOV1 bit/flag will automatically interrupt the MCU if the following conditions are satisfied:

(1) Switch SW4 of Fig-8.1 must be brought into closed condition by software codes in the setup() function of the sketch. This is done by putting HIGH at the TOIE1 bit of TIMSK register.

(2) Switch SW5 of Fig-8.1 must be brought into closed condition by software codes in the setup() function of the sketch. This is done by putting HIGH at the I-bit of SREG register.

5 Create sketch to blink L of UNO at 1-sec interval which is to be generated using TC1 and based on the pseudo codes of Section-8.3.

```
void setup()
{
     A:   L initialization
     1.  Set direction of DPin-13 as output

     B:   TC1 initialization in Normal Mode at clkADC = 15625 Hz
     1.  Set normal mode via TCCR1A Register
     2.  Keep TC1 at OFF condition with the help of TCCR1B Register
     3.  Load presetCount into TCNT1 for 1-sec time delay
     4.  Start TC1 as Timer with clkADC = 15625 Hz
}

void loop()
{
     digitalWrite(13, !digitalRead(13));
     timeDelay();
}

void timeDelay()
{
     1.  Wait until 1-sec has elapsed
     2.  Clear the TOV1 flag
     3.  Reload presetCount
}
```

6 Create sketch to blink L of UNO at 1-sec interval; where, the 1-sec time interval will be generated by TC1 on interrupt basis.

1. Initialize everything as needed

2. Ignite L, enter into loop() function, and then wait for interrupt.

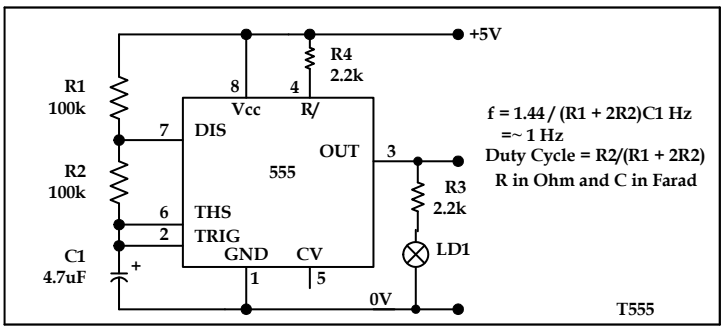
3. When at the elapse of 1-sec time TOV1 flag assumes HIGH state, the MCU is interrupted; it goes to ISR() and carry out the following tasks:

a. Change the state of L from ON to OFF.

b. Reload TCNT1 with pre-set count.

c. Return to loop() function and wait for interrupt.

7 Build the following 555-based ~1 Hz oscillator circuit on the breadboard and connect its output at DPin-6 of Fig-8.1. Create sketch so that the MCU will just flash L of UNO when it finished counting each set of 16 pulses of the oscillator.



8 The duty cycle (ON period) of the output signal of the oscillator of Q7 is about 330 ms. Create sketch to measure the duty cycle of the signal and show it on the Serial Monitor at 1-sec interval.

9 Change the value of R1 of the oscillator circuit of Q7. Now, create sketch to measure the frequency of the signal and show it on Serial Monitor at 5-sec interval.

10 Write down the symbolic names of the registers that are involved with TC1.