

Timer Lab

Example 1:

Write Embedded C code using ATmega328P μ C to control led by Timer2.

Requirements:

- 1- The LED is connected to pin 0 in PORTC.
- 2- Connect the LED using **Positive Logic** configuration.
- 3- Configure the timer clock to $F_{CPU}/1024$.
- 4- Timing should be count using Timer2 in Normal Mode.
- 5- Toggle the LED every half second.

Steps Main:

- 1- Configure the LED pin to be output pin.
- 2- Make LED off at the beginning(Positive Logic).
- 3- Enable global interrupts in MC by setting the I-Bit.
- 4- Start the timer.

Steps Timer init:

- 1- initial the value of timer counter register (TCNT) to zero.
- 2- Enable overflow interrupt.
- 3- Configure the timer control register
 - ✓ Non PWM mode FOC0=1
 - ✓ Normal Mode WGM01=0 & WGM00=0
 - ✓ Normal Mode COM00=0 & COM01=0
 - ✓ clock = $F_{CPU}/1024$ CS00=1 CS01=0 CS02=1

ISR:

Increase the count of a global or static variable in ISR and check the count if it reach the required counts to obtain the required time (0.5 sec).

Example2:


Write Embedded C code using ATmega328P μ C to control a 7-segment using Timer2.

Requirements:

- 1- 7-segment connected to **PORTC**.
- 2- Configure the timer clock to $F_{CPU}/256$.
- 3- Timing should be count using Timer2 in Normal Mode.
- 4- Every second the 7-segment should be incremented by one. If the first 7-segment reached 9 overflow will occur.

Example 3:

Repeat Example 1 but use Timer2 Clear timer on compare (CTC) Mode.

 In this example CTC mode will act exactly as timer normal mode in Example 1.

Example 4:

Write Embedded C code using ATmega328P μ C to generate a 15.625 KHz clock using Timer2 CTC Mode.

Requirements:

1. Use Timer2 in CTC Mode with clock equals to $F_{CPU}/1024$ clock.
2. Clock duty cycle is 50%.
3. Connect the output to buzzer

Steps Timer init:

1. initial the value of timer counter register (TCNT) to zero.
2. Set Compare value (OCR=250)
3. Set OC2A (PB3) Pin as output pin
- 4- Configure the timer control register
 - ✓ Non PWM mode FOC0=1
 - ✓ CTC Mode WGM01=1 & WGM00=0
 - ✓ Toggle OC2A on compare match COM00=1 & COM01=0
 - ✓ clock = CPU/1024 clock CS00=1 CS01=1 CS02=1
 - ✓ clock = $F_{CPU}/128$ CS20=1 CS21=0 CS22=1.