# Lab 4 Timers/Counters in ATmega328P

## Objectives:

1. Understand the operation modes of the timers/counters in ATmega328P.
2. Learn to program timers to generate time delays and waveforms .
3. Learn to program counters to count the external event.

## Lab Activities:

## 1. Generate a square waveform using Timer1

In this part we will use Timer1 Channel A to generate a 0.5 Hz square waveform to blink the LED. Connect OC1A (pin9 in the Arduino board) to an LED and run the following program. The LED should be on for 1 second every 2 seconds. If we choose to use Phase-Correct PWM mode, toggle output and pre-scaler 1024 to generate the output, the OCR1A should be 1sec/(2*(1/(16MHz/1024)))=7812 , as shown in Figure L4.1.

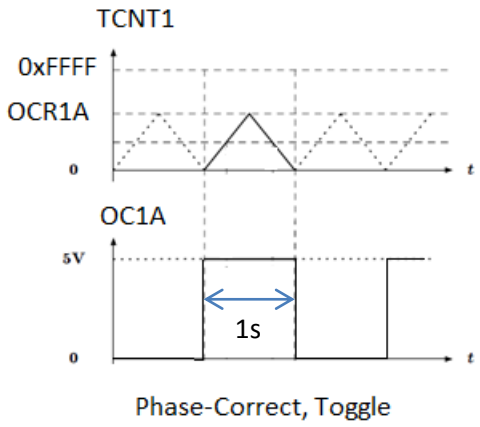


Figure L4.1

```
/* Lab 4 Example 1 code. Timer 1 Channel A output a 0.5 Hz square waveform */
#include "avr/io.h"

void setup()
{
  DDRB = 0x02;   //set pin 9 as output;  pin 9 (OC1A) is connected to an LED

  /*set operation Toggle mode; WGM13:0 = 1011 phase correct PWM
      CS12:0 = 101 prescaler = 1024*/
  TCCR1A = 0x43;
  TCCR1B = 0x15;
```

```
  OCR1A = 0x1E84; //set the TOP value 1sec/(2*(1/(16MHz/1024)))=7812 = 0x1E84
  TCNT1 = 0;  //clear counter and update OCR1A
}

void loop()
{ }
```

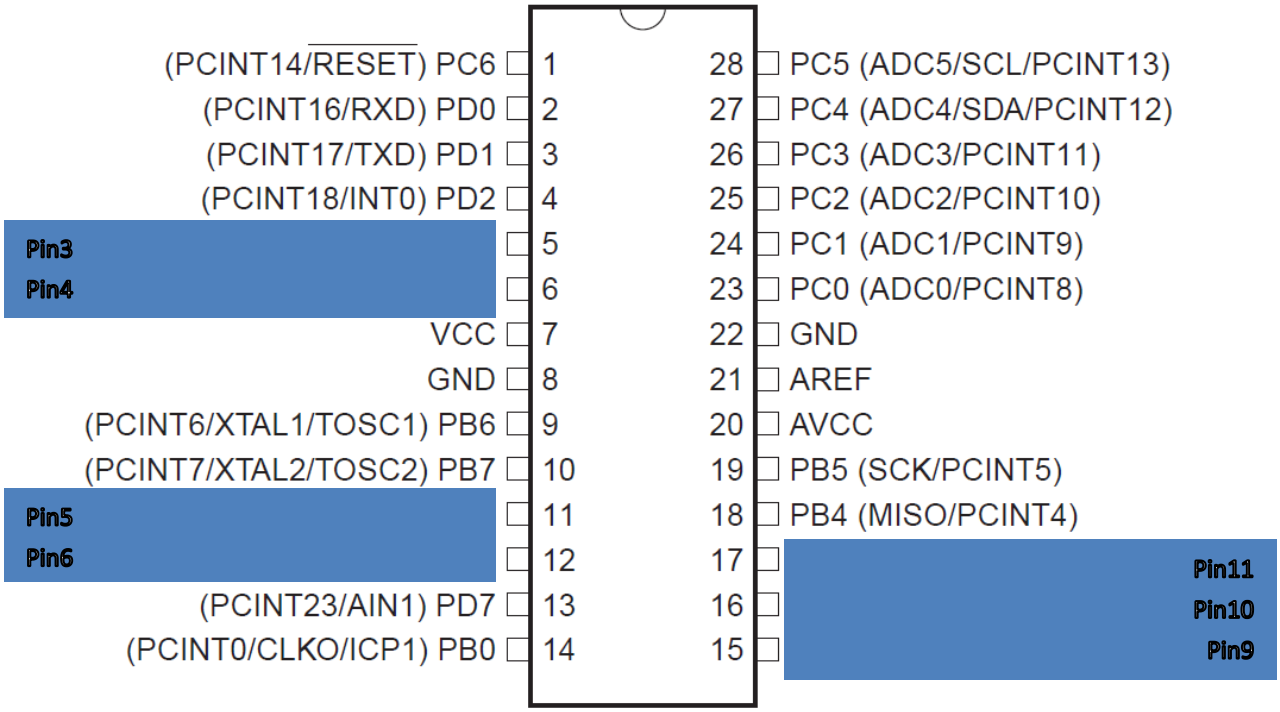


Figure L4.2 Pin layout of ATmega328P

**Lab Requirement 1:**

Please determine the value of TCCR1A, TCCR1B, and OCR1A to generate a 10 Hz square waveform according to the following specific:

- use channel B for the output waveform,
- use phase-correct PWM mode,
- use OCR1A as the top.
- use clear on up-counting, set on down counting,

Select proper prescaler. Set duty ratio as 50%. Modify the above program and observe the output. Note: You need to connect to OC1B to the LED instead. You should observe the LED blinks quickly. Please show the demo to the instructor. Please include the fully commented code for your report.

## 2. Use PWM waveform to obtain desired analog voltage

In this part, we will use PWM to get analog signal that we need. First, use Timer/Counter 1 to generate a 200 KHz square waveform similar to part 1. Then Use a RC as the LPF to observe the output analog voltage. Modify the duty ratio by adjusting the OCR1B for 20% and 80% and observe the analog output. Remember in order for the pwm to work, we need $f_{analog} << f_c << f_{pwm}$.

**Lab Requirement 2:**

1) Please determine the value of TCCR1A, TCCR1B, and OCR1A to generate a 200 KHz square waveform according to the same specific in Lab Requirement 1. You may use a pre-scaler of 1. Set duty ratio as 50%. Modify the program and observe the output in oscilloscope.
2) Please determine the RC values for a proper cutoff frequency (e.g. 10 KHz.). You may adjust the PWM frequency and cutoff frequency to get an analog output with a small ripple.
3) Prepare a function to adjust the duty ratio automatically to get desired level of analog signal according to the duty ratio parameter. The function prototype should look like:

    **void OutputVoltage(float dutyRatio);**

The parameter dutyRatio is a float point number between 0 to 1. The function should be able to calculate the appropriate OCR1B value to get the waveform with the request duty ratio.
The initialization of the Timer1 can be put into another separate function, such as:
**void PWMInit()**.

Please show the demo to the instructor. You should be able to observe the analog output voltage changes when *dutyRatio* is adjusted. Please include the process how you determine the Timer1 registers and RC values, the connect circuit and fully commented code for your report.

## 3. Generate a triangle waveform

In this part, you are required to generate a triangle waveform of approximately 500 Hz, as shown in Figure 4.3. You may use N different duty ratios (for example N= 10, Duty ratio should be in sequence of 0, 0.1, 0.2,…., 0.9, 1, 0.9, 0.8, …0.1, 0 and repeats) to get the needed analog voltage levels (0V, 0.5V, 1V, 1.5V, …4.5V, 5V) for your triangle waveform. Please reuse the functions you implemented in previous parts.

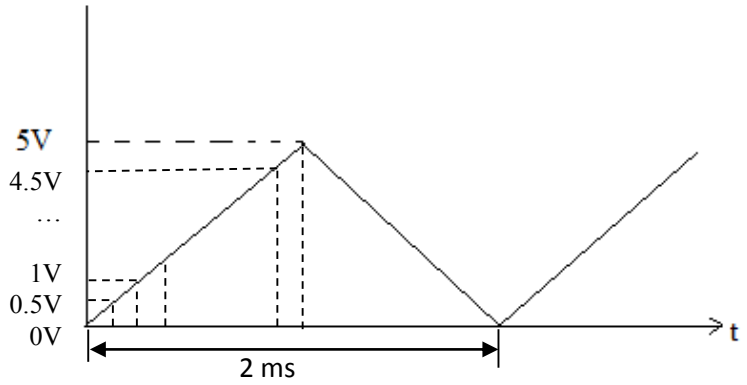


Figure L4.3 Triangle Waveform

Note that in order to get the 500 Hz triangle wave, you need to wait for an interval of 2ms/(N*2) =100 μs between updates when N = 10. You may use Timer0 to generate the delay of 100 microsecond and use the function prototype as follows:

```
void My100uSecDelay();
```

Inside the 100uSecDelay() function, you can use a while loop to wait till the TOV0 flag is set to obtain the desired delay, for example:

```
do {} while ((TIFR0 & (0x1 << TOV0)) ==0);   //wait till TOV0 to be set
```

You may then obtain the waveform by repeating calling the outputVoltage() function using different duty ratio. Use the following code snips as a reference.

```
static const float m_triangleWaveform[] =
{  0.000,  0.100,  0.200,  0.300,  0.400,  0.500,  0.600,  0.700,  0.800,   0.900,
   1.000,  0.900,  0.800,  0.700,  0.600,  0.500,  0.400,  0.300,  0.200,  0.100
};
Unsigned char currentIndex;
```

Inside the loop

```
sample = m_triangleWaveform[currentIndex];
currentIndex = (currentIndex+1)%20;
outputVoltage(sample);
myMicroSecDelay();
```

**Lab Requirement 3:**

Please determine the Timer 0 and Timer 1 register values needed for your program. Complete the program and observe the triangle waveform in the oscilloscope. Show the demo to the instructor. Please include a flow chart and the fully commented program for your report.

## 4. Event counter using Timer/Counter0

Use Timer0 as an event counter. Connect T0 (PD4/pin4) to a pushbutton. The number of pulses counted will be displayed on the Serial Monitor window.

```
#include "avr/io.h"
unsigned int count; //the count of events

void setup()
{
  PORTD |= 0x10;  //activate pull-up of PD4
  DDRD &= 0xEF;  //set pin 4 (PD4) as input
  TCCR0A = 0x00; //normal mode
  TCCR0B = 0x06; //output clock source, falling edge
  TCNT0 = 0;
  Serial.begin(9600);
}

void loop()
{
  char message[80];

  count = TCNT0;
  sprintf(message, "\nThe number of events: %d", count);
  Serial.write(message);
}
```

**Lab Requirement 4**

Observe what is happening when you push the pushbutton. Will only one event or multiple event be counted when you only press the pushbutton one time? Explain the result.
If the rising edge is used to trigger the event counter, how should you change the program?
Please include the circuit diagram and your explanation for your report.

## Report Requirements:

Please follow the general requirements for the lab report. Fully commented programs for Lab requirements 1-3 are required. Please follow the instructions in each individual part.