Temple University

College of Engineering

Department of Electrical and Computer Engineering (ECE)

Student Lab Report Cover Page

Course Number : 3613

Course Section : 001 / 002

Experiment # : Lab # 11

Student Name (print) : Von Kaukeano

TUid# : 915596703

Date : 11/14/19

Grade : _____ **/100**

TA Name : Sung Choi

Introduction

This lab is a demonstration of timers and counters in both C and assembly. This is useful because not only does it generate PWM signals with time delays but it can also can count events from external sources. In order to generate a time delay you set the timer register to a value that will begin there and increment by 1 until it reaches the max value of 255. You use the control register to set the mode and pre-scale value. Once the value is greater than the max value the timer overflow flag will raise high. Once that bit is set high, reset it by writing a one to the bit and start over. This creates the on/off of the signal. Other methods are using the output compare register to reach the signal you would like to achieve.

Procedure

For **both** activities we began by doing some simple calculations so we know what to initialize the timer register at.

To achieve the 3k and 1k PWM.

1/3k = .33ms

Tsig = .167ms

Tclk = 1/16000000 = .0605microseconds

Tsig / Tclk = 2672

This needs to be prescaled.

2672 / 64 = 41.75

256 – 41 = 215

256 is the highest a value an 8-bit register can be.

The initial value of the timer register should be is 215 or in hex D7.

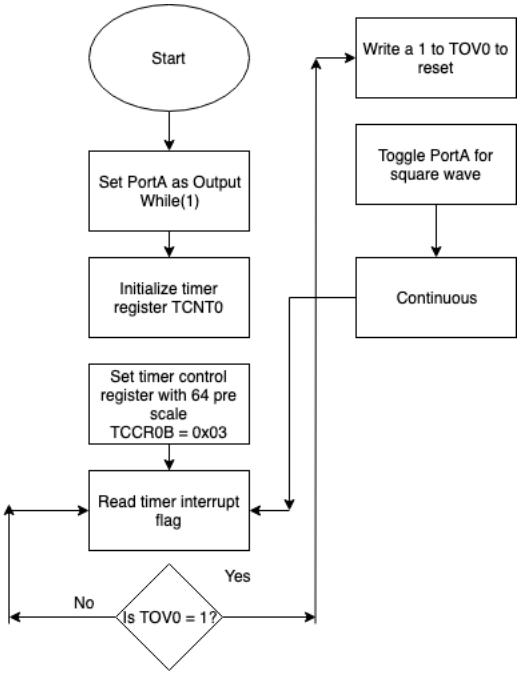
The same calculations can be made for the 1k signal and it is found that the prescale is the same and that the initial value for the timer register should be hex 83.

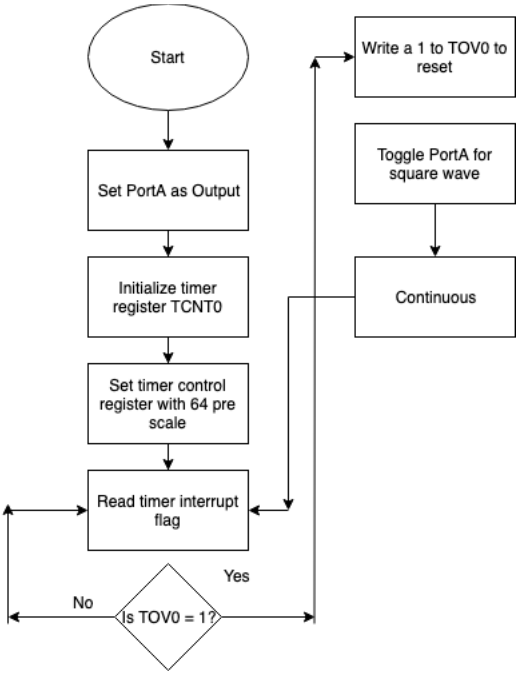
By looking at the control register datasheet, the correct value for pre scale and normal mode is hex 03.It can be seen in the code that all four have a pre scale of 64 and the initial values are the only ones that need to be changed per reading.

The register is incremented while the TOV0 is zero and reset once it is raised high.

The values needed to be changed in the C code is TCNT0 and TCCR0B, and register R20.

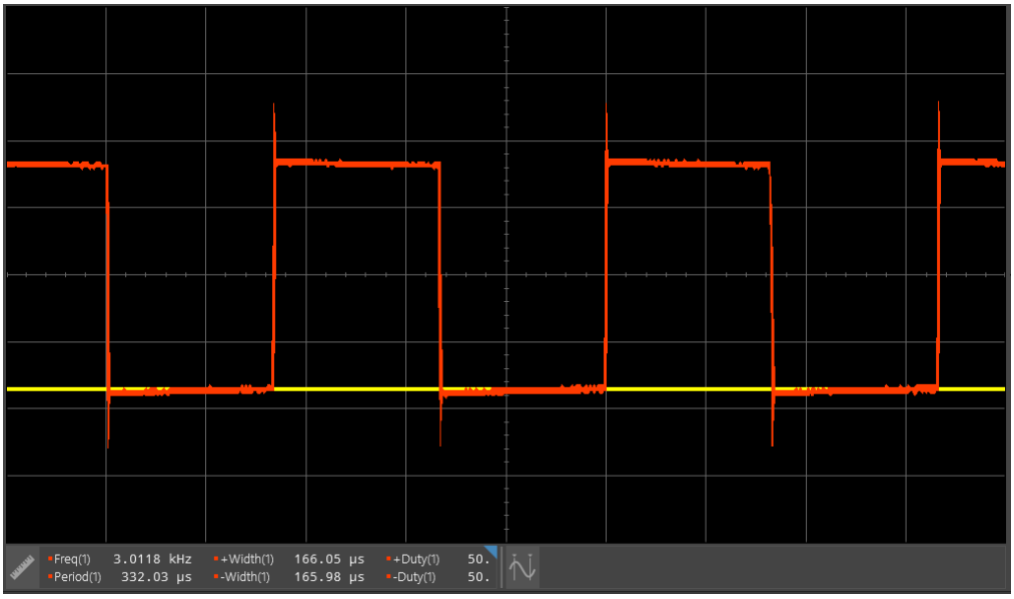
I do not have screenshots of the timer and cannot get back into the lab before the due date and finished the lab before I left, but the calculations are correct and verified with the screenshots below.

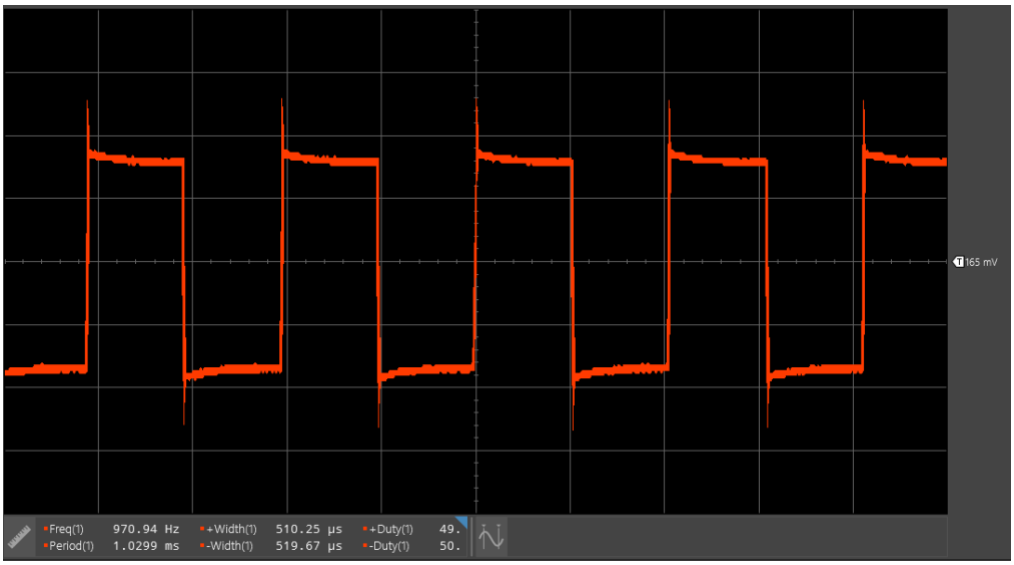




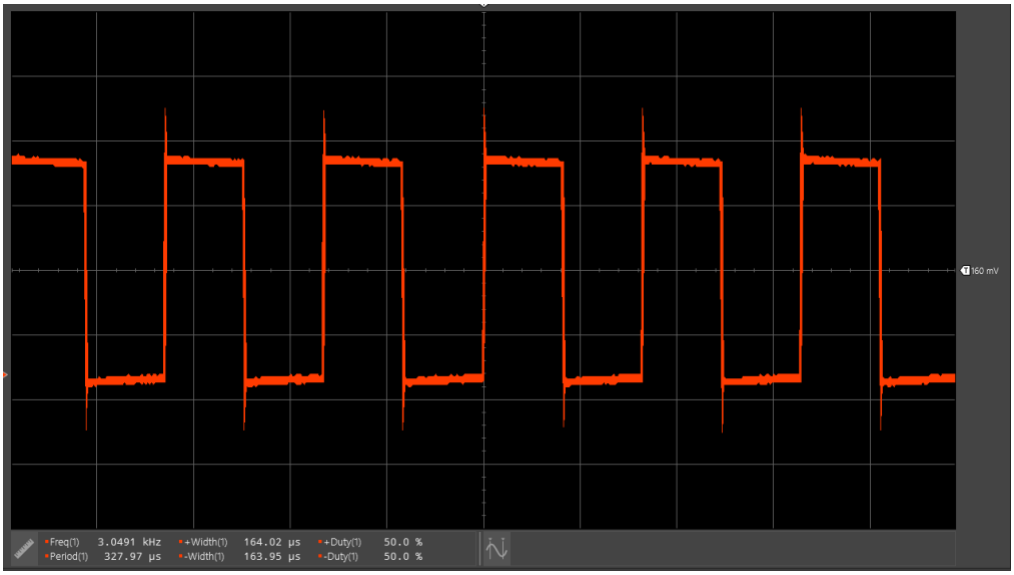
Result and analysis

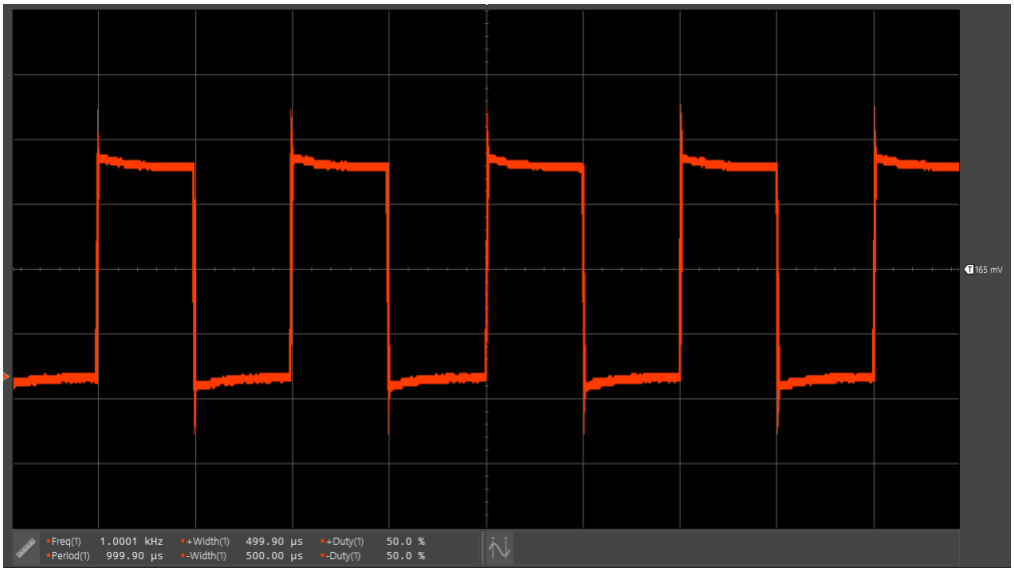
Activity 1


Freq(1) 3.0118 kHz +Width(1) 166.05 µs +Duty(1) 50.
Period(1) 332.03 µs -Width(1) 165.98 µs -Duty(1) 50.


165 mV
Freq(1) 970.94 Hz +Width(1) 510.25 µs +Duty(1) 49.
Period(1) 1.0299 ms -Width(1) 519.67 µs -Duty(1) 50.

Activity 2





Just as expected, two 1kHz signals and two 3kHz signals.

Discussion and Conclusion

After the calculations the desired output signal was the expectation and that was the result. We could have done this for any signal that we wanted that respected the max value of the register and frequency of the microcontroller. At first I did not understand the difference between a delay in assembly and a timer delay, but after speaking to Dr. Helferty we will be implementing the counter for the final processors project and that the timer does not tie up the processor as much as the regular delay.

Code

1.

```
#define F_CPU 16000000UL
```

```
#include <util/delay.h>
#include <avr/io.h>
int main(void)
{
   DDRA = 0xFF;

   while (1)
   {
                    TCNT0 = 0x83; //D7
                    TCCR0B = 0x03;
                    while ((TIFR0&(1 <<TOV0))==0);
                    TCCR0B = 0;
                    TIFR0 = (1<<TOV0);
                    PORTA = ~PORTA;
   }
```

2.

```
        LDI R16,0xFF
        OUT DDRA,R16
        BEGIN:
        LDI R20,0X83     ; D7
        OUT TCNT0,R20
        LDI R20,03
        OUT TCCR0B,R20
        AGAIN:
        IN R20,TIFR0
        SBRS R20,0
        RJMP AGAIN
        LDI R20,0X00
        OUT TCCR0B,R20
        LDI R20,(1<<TOV0)
        OUT TIFR0,R20
        EOR R17,R16
        OUT PORTA,R17
        RJMP BEGIN
```