Microavionics Lab 5 (5067)

Sage Herrin
University of Colorado Boulder
Due 10/27/20

This lab continued to build the understanding of timers, CCPs, and ISRs in particular, with more work done with the LCD. It also included a brief introduction to C programming that tied in well with the concept of timers and interrupts, where this short task in C was given to toggle and LED using a low priority ISR and a timer.

A. Lab Questions

1

As no formal question were asked in this lab besides documenting the timing of certain events and PWM signals, the following screenshots taken using the logic analyzer AD2 tool show the progression of the lab events. The first several images show the increase in the output PWM as the RPG was continuously turned clockwise to increase the duty cycle, eventually reaching a maximum of 2 ms high, with some allowed tollerance.

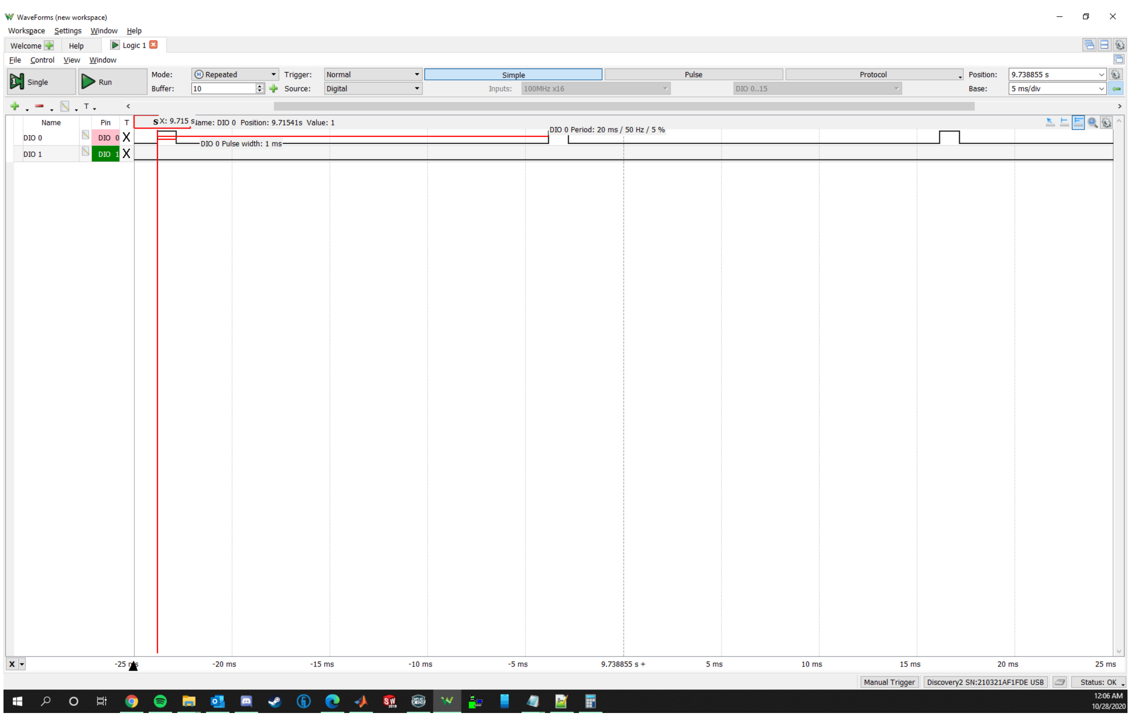

Fig. 1 1ms high PWM signal as lower limit.

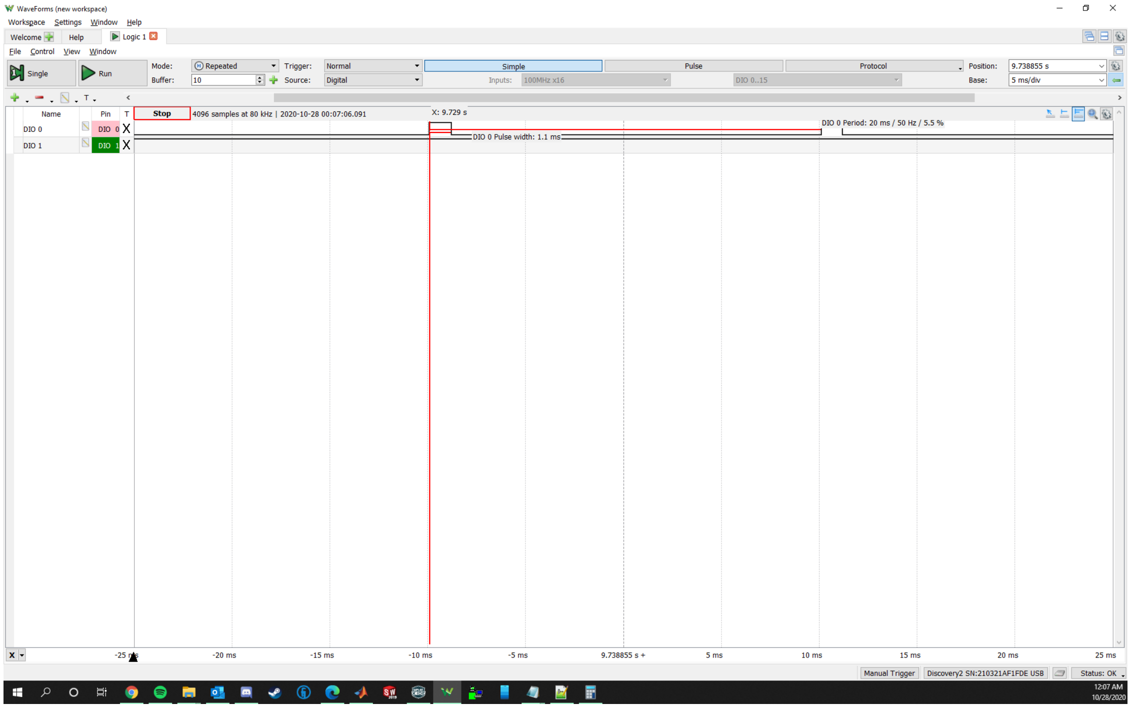

Fig. 2 1.1 ms high PWM signal

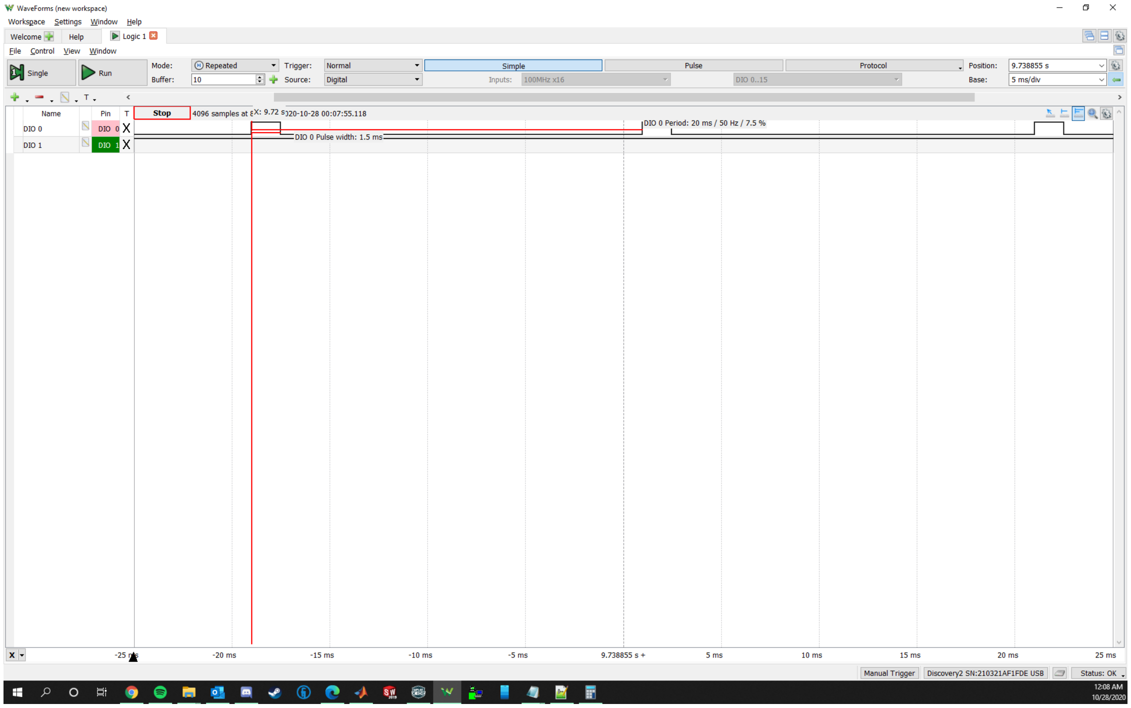

Fig. 3 1.5 ms high PWM signal.

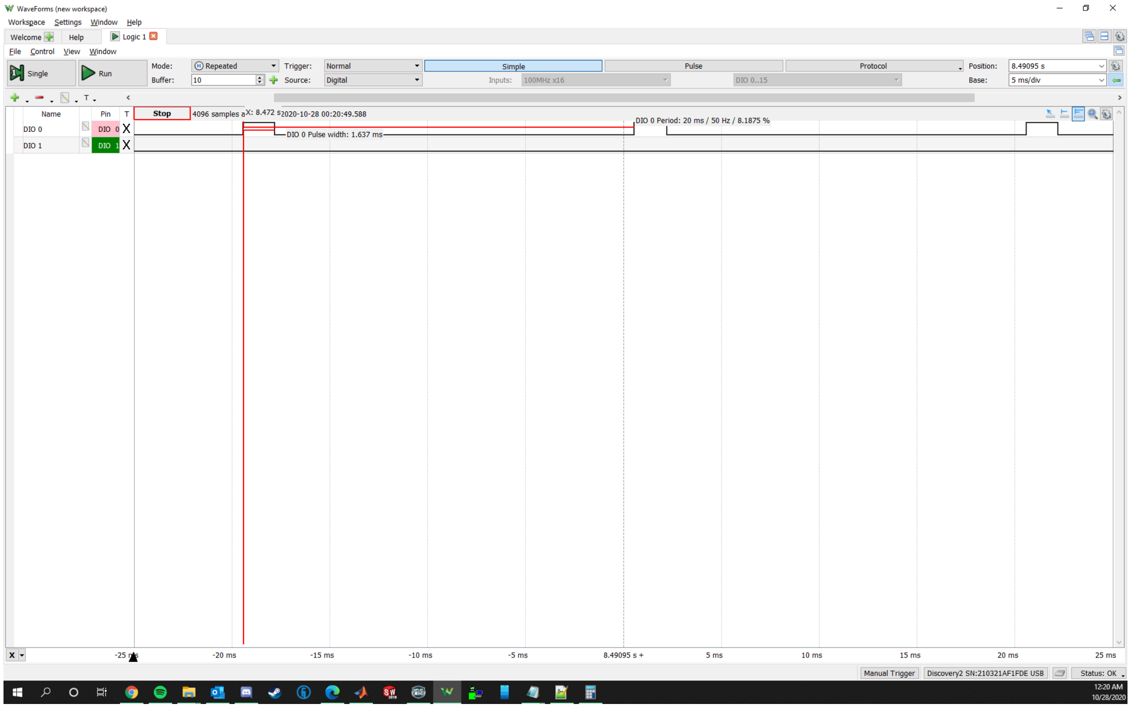

Fig. 4 1.64 ms high PWM signal.

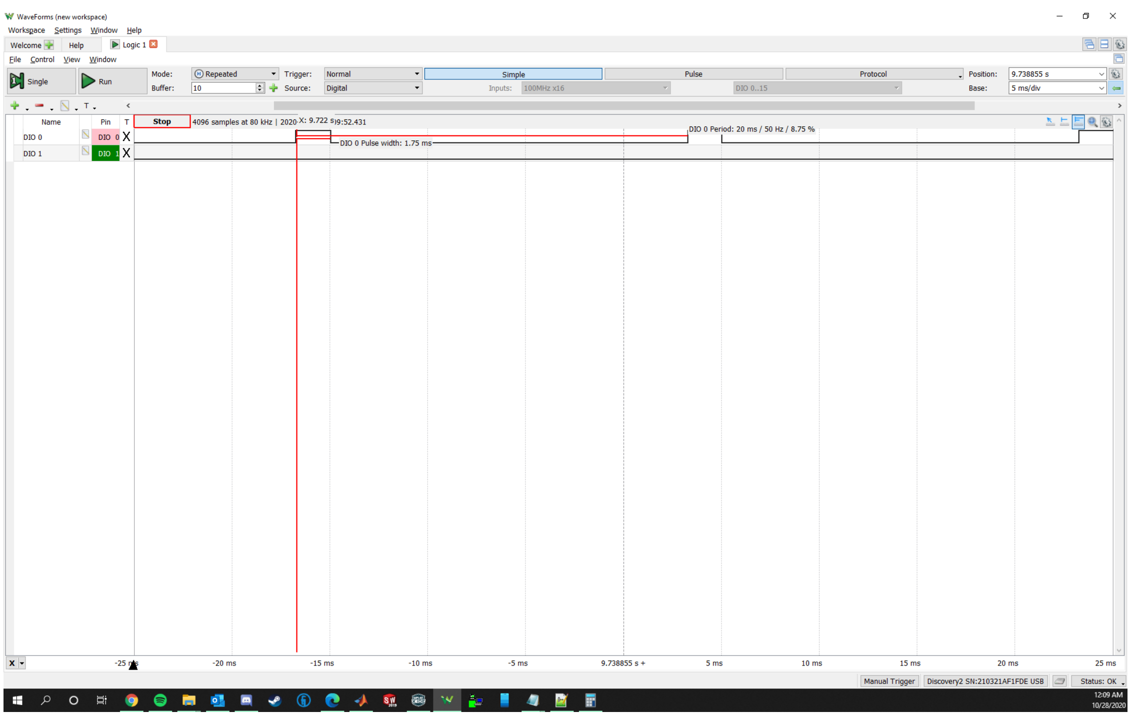

Fig. 5 1.75 ms high PWM signal.

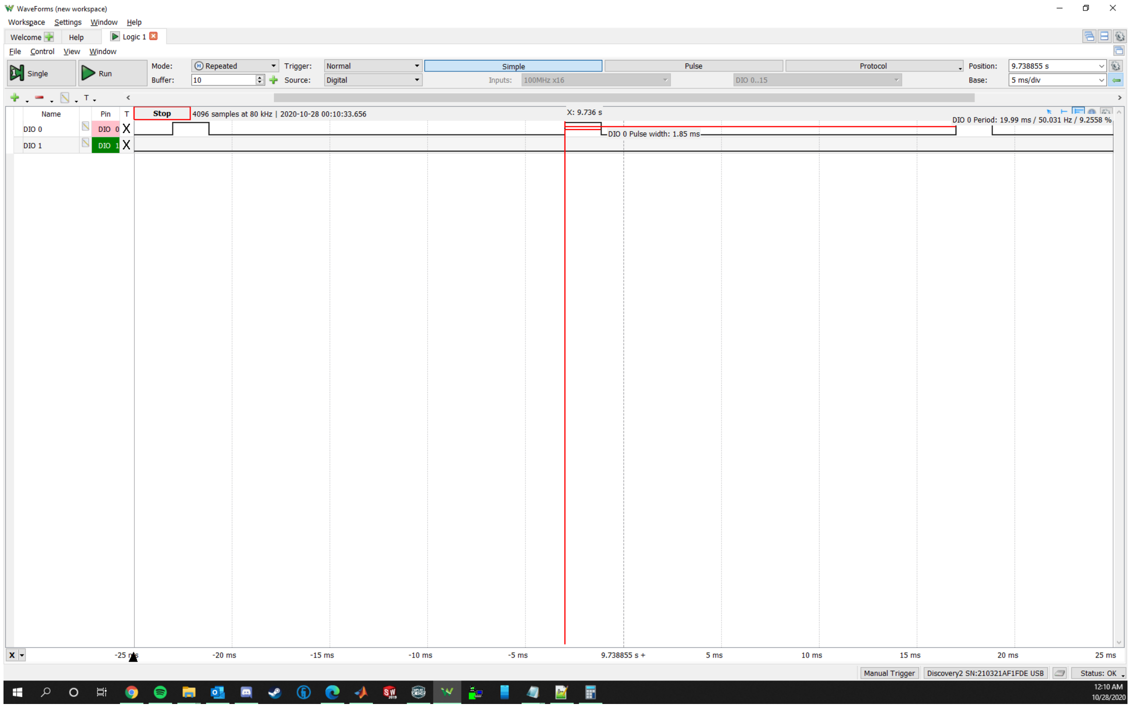

Fig. 6 1.85 ms high PWM signal as starting point.

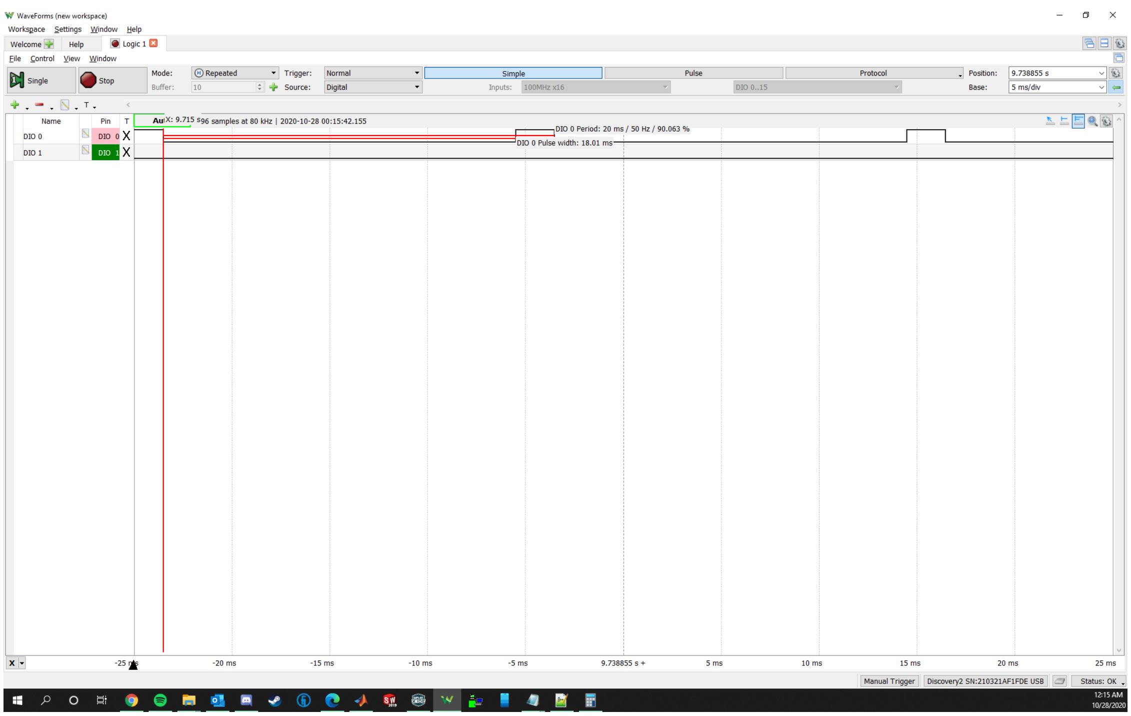

Fig. 7 2ms high PWM signal upper limit.

The following figure also shows the 1 Hz 50% duty cycle square wave generated using the provided C code template, which utilizes a low priority interrupt and timer to generate the signal, which blinks the LED RD4.

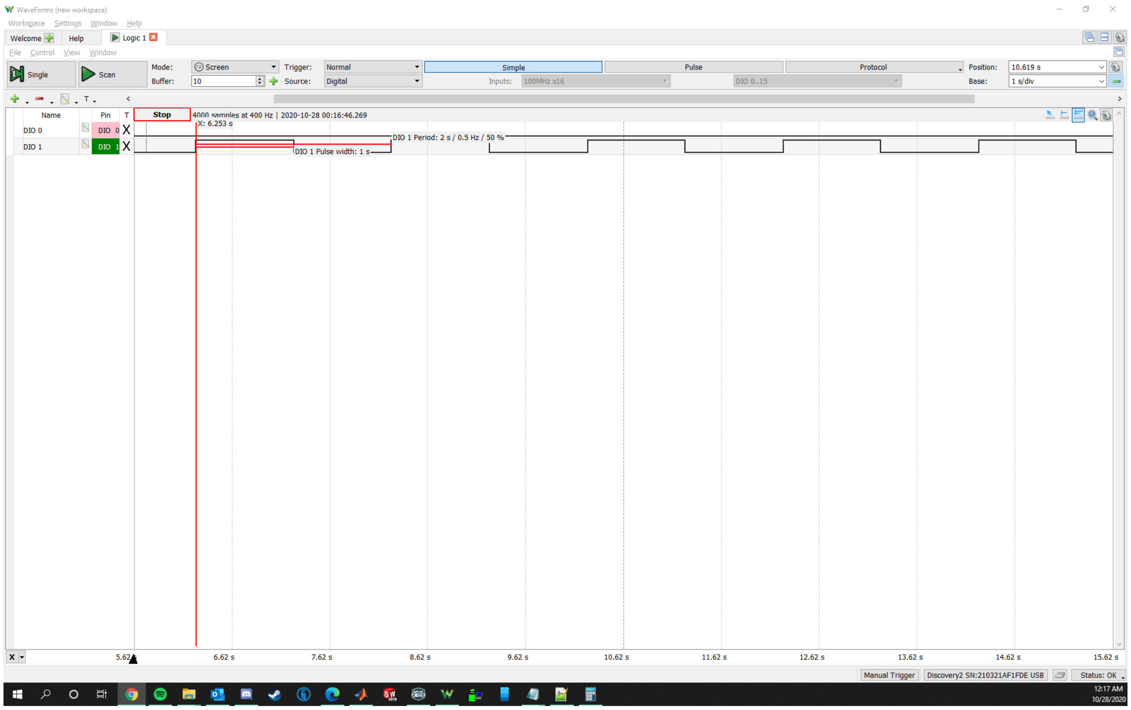

Fig. 8   1 s 50% duty cycle wave generated using C code and low priority interrupt.

The two figures below show the initialization which sets RD5,RD6, and RD7 high for .5 seconds, followed by the alive LED, which is high for 200ms and low for 800ms, for a full period of 1 second

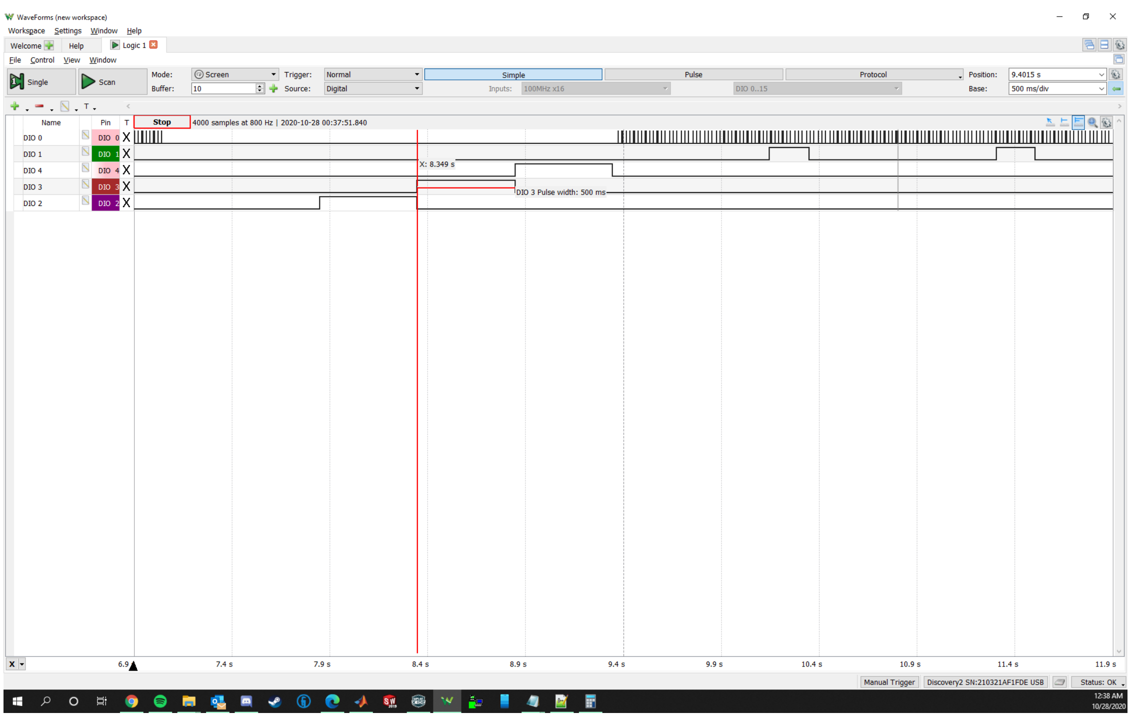

Fig. 9   Three LEDs going high for half a second each upon initialization

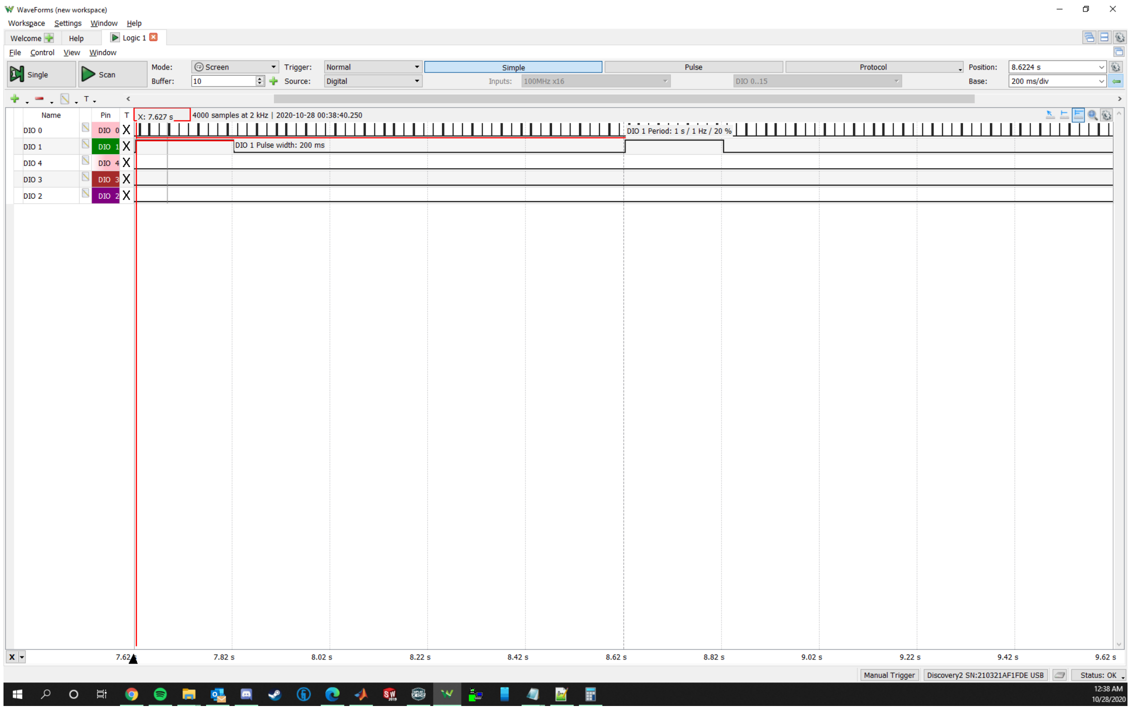

Fig. 10 Alive LED blinking 200 ms high then being set low for 800 ms.

I. Conclusion

This lab served as a terrific and fun way to to tie together how timers, CCPs, and interrupts work together to control events on board a microcontroller that also adhere to strict timing requirements. This also served as a gradual introduction to C programming, which will tie in heavily with the activites of lab 6, while also solidifying the concepts that have been taught up until this point in time.