Introduction:

The PIC18F4520 is a programmable microcontroller via the assembly language and an Integrated Development Environment (IDE). To better understand the intricacies of the microcontroller, the MPLAB IDE, and analysis tools, I will explore two test programs provided in class.

Overview & Analysis of P1 Program:

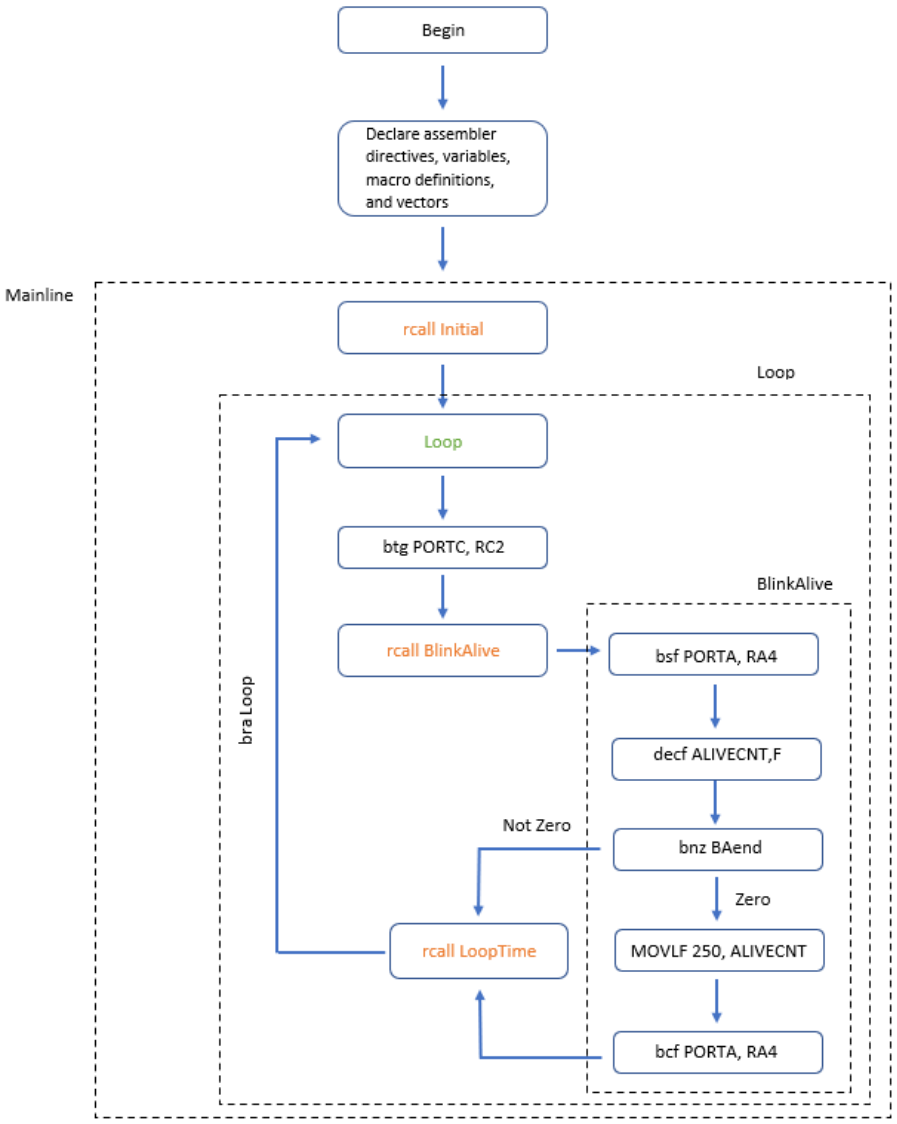

Figure 1: Flowchart of P1.asm

The purpose of the P1 program is to toggle an LED. Specifically, the LED is toggled ‘ON’ for 10ms every 2.5 seconds. The flowchart above provides an understanding on how the program runs. Upon taking care of some logistics involving declaring directives, initializing variables, and

defining macros, the focus is in `Mainline` which is where the "real" program lies. It is comprised of several subroutines (in orange) and in addition, several important instructions and variables that will dictate the flow of the program. Prior to examining these however, one thing must first be understood at the hardware level of the PIC18F4520: timers.

Function of Timer0

The PIC18F4520 encompasses four 16-bit timers: Timer0, Timer1, Timer2, Timer3 which are each suited for specific tasks. Timer0 is of particular interest because it serves as a counter (i.e. rollover time) for tasks. With no prescaler, Timer0 normally rolls over every $2^{16} =$ 65,536 clock cycles. If a task requires a rollover time of 10ms, we will need to remove some number x clock cycles. Given the microcontroller's internal clock period of $0.4\mu s$, $\frac{10ms}{0.4\mu s} =$ 2,5000 clock cycles are necessary to remove from the sequence. Note that it requires an additional 12+2 cycles to actually remove the cycles. Hence, the roll over time of Timer0 is determined by 65,536 – 2,500 + 12 + 2 stored in a variable `Bignum`.

LoopTime Subroutine:

This subroutine makes use of Timer0 to set the duration of the Loop in Mainline. In the P1 program, the variable `Bignum` tells us that this duration is 10ms. This is reflected on the RC2 pin of the microcontroller. If we were to observe the output of this pin, it is expected to be a pulse train with some duty cycle (more on that later).

BlinkAlive Subroutine:

It was previously mentioned that the LED lights up for 10ms as a result of the `LoopTime` Subroutine. The `BlinkAlive` subroutine determines how often the LED is toggled 'ON' via the RA4 pin. In other words, what is the delay between each toggle? The answer lies in the `ALIVECNT` variable. This variable is decremented by 1 each time the subroutine is called via the `decf` instruction (line 118). The instruction `bnz` (line 119) checks if `ALIVECNT` is zero or not. Upon reaching zero, `ALIVECNT` is reinitialized to 250 via the `MOVLF` instruction (line 120)and the counter is reset which in turn, toggles the LED for 10ms every $(10 \times 10^{-3}) \times (250) = 2.5$ seconds.

Simulation:

While this toggling can't be observed physically given the absence of the board, the signals produced can using the *MPLAB's Logic Analyzer* simulation tool. As mentioned earlier, the outputs at the RC2 and RA4 pins should produce a pulse train with some duty cycle which can be determined as such:

$$Duty\ Cycle = \frac{T_{HIGH}}{T_{HIGH} + T_{LOW}} \times 100\%$$

Where T_{HIGH} is the duration of the 'HIGH' time and T_{LOW} is the duration of the 'LOW' time.

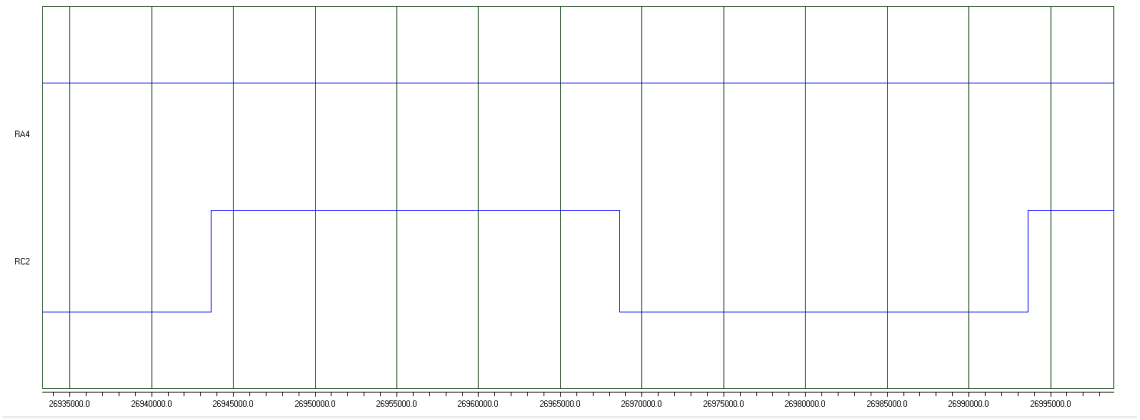

Figure 2: Output of RC2 and RA4 pins (P1.asm)

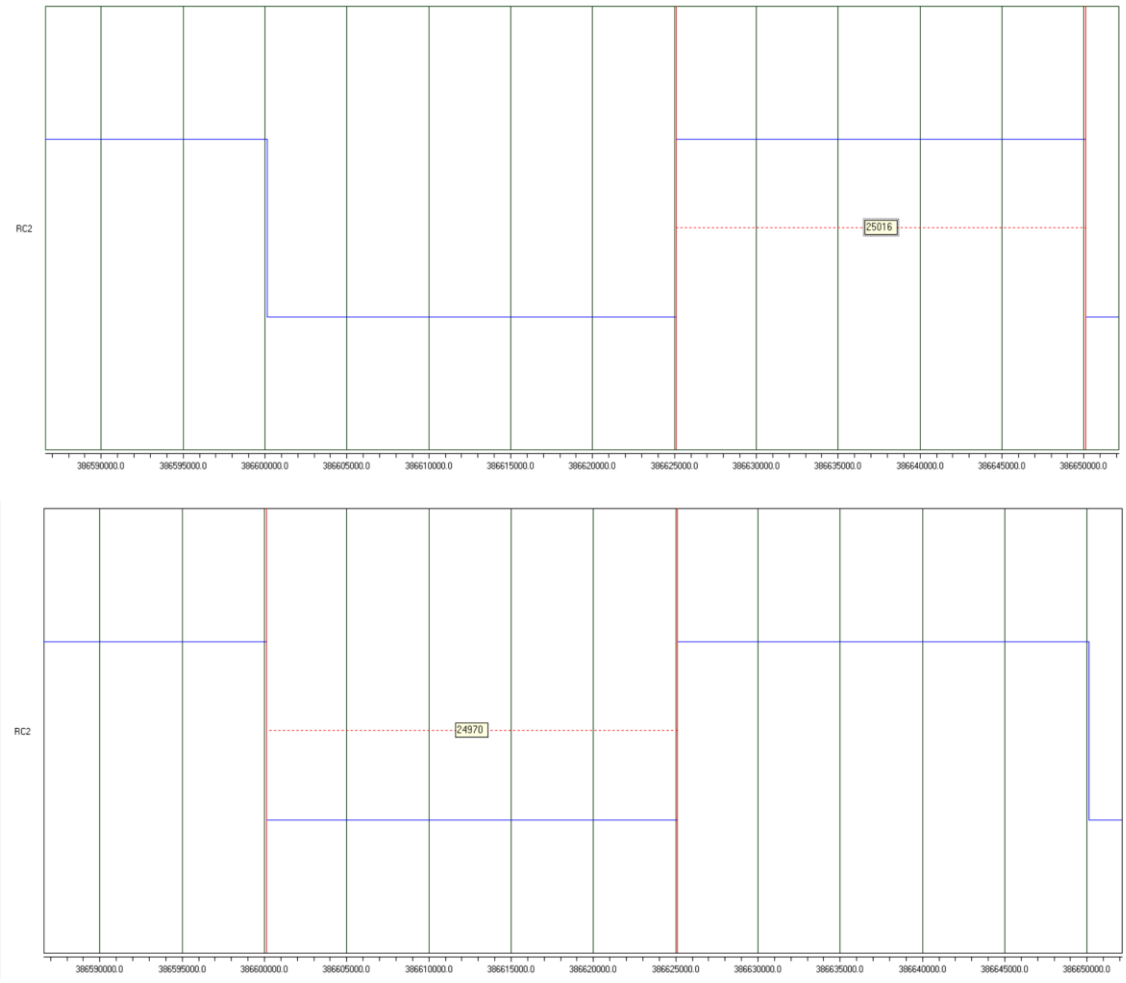

Figure 3: Measuring duty cycle of RC2 pin at Port C (P1.asm)

The parameters for the duty cycle of the pulse train at pin RC2 are $T_{HIGH} = 25016ms, T_{LOW} = 24970ms$ which yields a duty cycle of 50%.

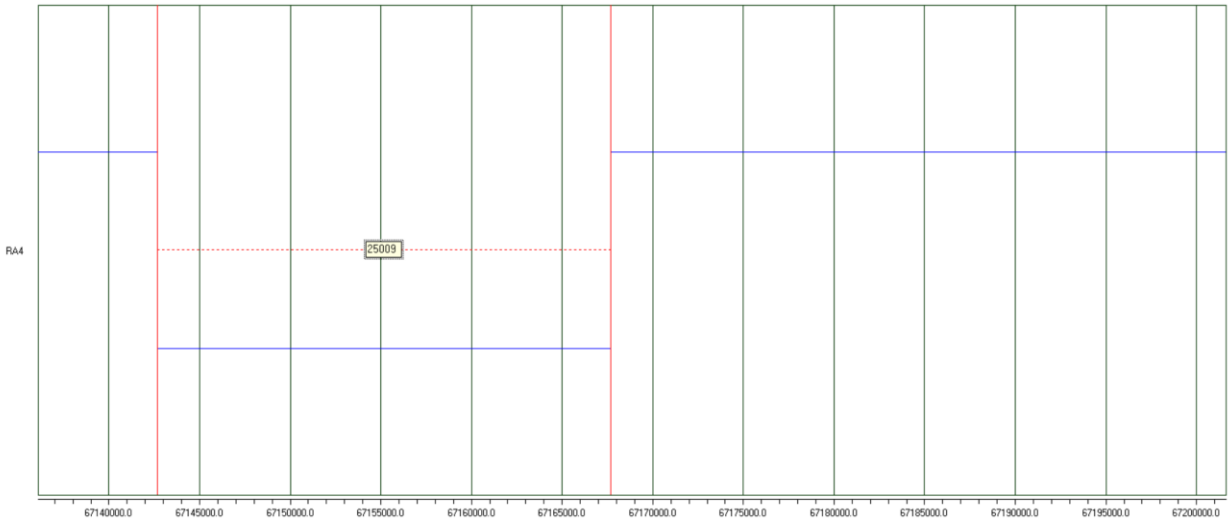

Figure 4: Measuring duty cycle of RA4 pin at Port A (P1.asm)

It's difficult to measure the RA4's pin HIGH time but its output is suspected to have a very low duty cycle.

Overview & Analysis of P0 Program:

The P0 program is very similar to the P1 program. In analyzing the `LoopTime` subroutine, the roll over time of Timer0 can be attributed to `Bignum` whose value is now 65,536 – 250 + 12 + 2. This means that the subroutine lasts just $250ms \times 0.4\mu s = 0.1ms$. In the `BlinkAlive` subroutine, the `ALIVECNT` is reinitialized to 4 (line 124) instead of 250 and so the LED is toggled 'ON' for 10ms every $(10 \times 10^{-3}) \times (4) = 4 \times 10^{-3}\ s = 4ms$. This means that the LED will toggle at a significantly faster rate than the P1 program.

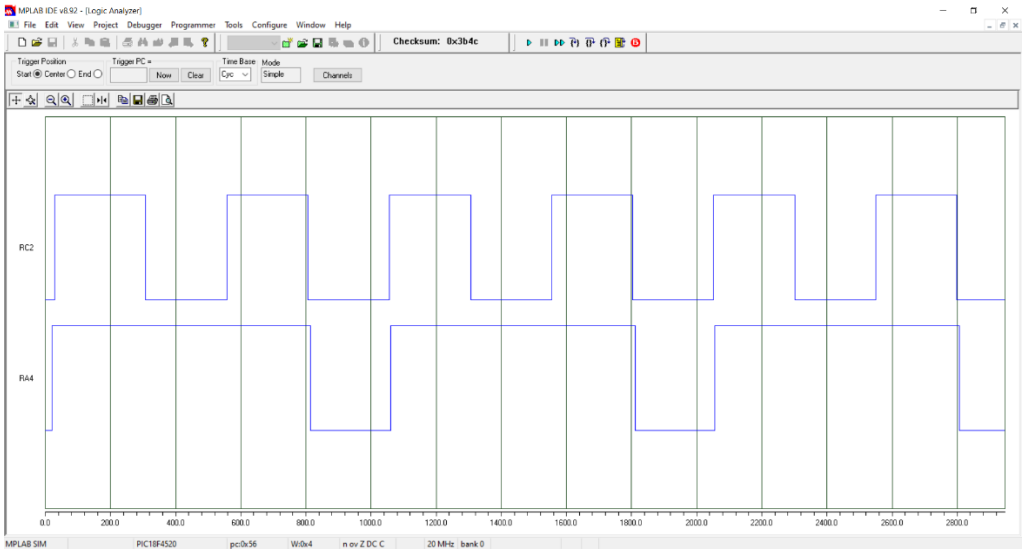

Figure 2: Output of RC2 and RA4 pins (P0.asm)

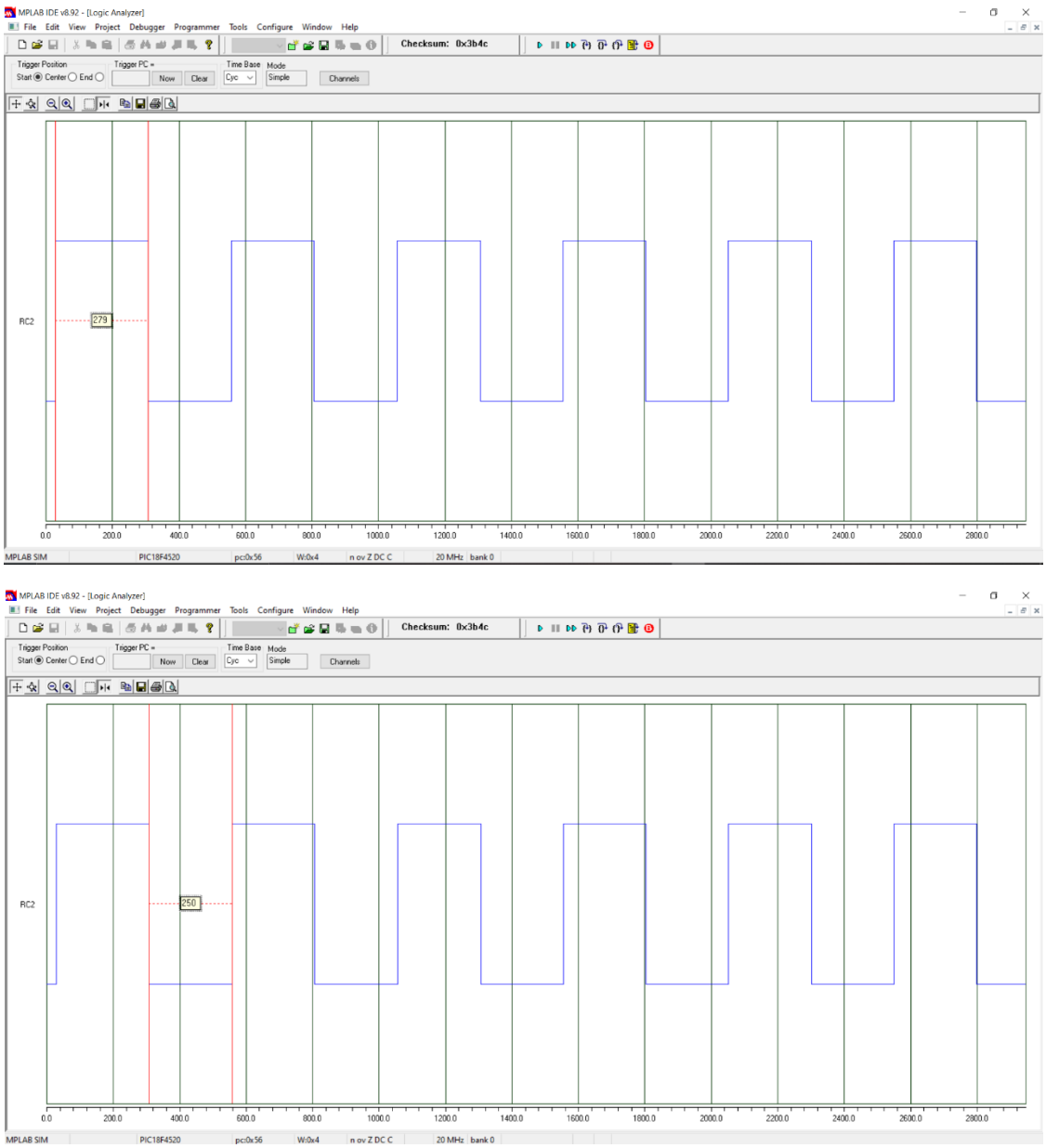

Figure 3: Measuring duty cycle of RC2 pin at Port C (P0.asm)

The parameters for the duty cycle of the pulse train at pin RC2 are $T_{HIGH} = 279ms, T_{LOW} = 250ms$ which yields a duty cycle of approximately 50%.

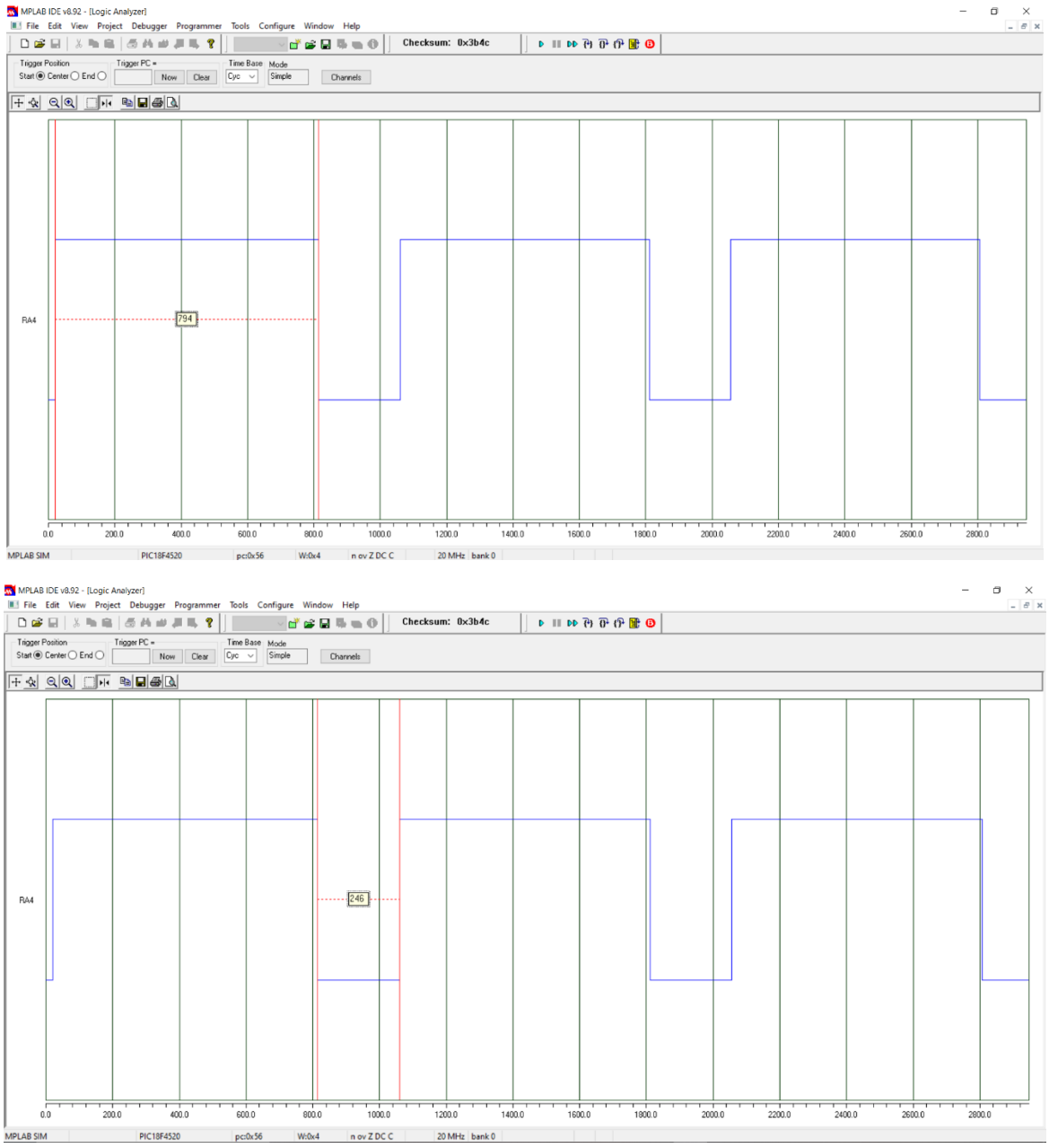

Figure: Measuring duty cycle of RA4 pin at Port A (P0.asm)

Likewise, $T_{HIGH} = 794ms, T_{LOW} = 246ms$ which results in a duty cycle of 76%. These two values correspond to what was stated in the tutorial given in the first lecture.

Conclusion:

This is just the beginning of the understanding the PIC18F4520 microcontroller as well as the MPLAB IDE. As we progress forward in the course, we will be able to perform more complicated tasks and make sense of meaningful results from these tasks.