Microavionics Lab 3 (5067)

Sage Herrin
University of Colorado Boulder

Due 9/29/20

A. Lab Questions

warm-up questions

1.) using MOVF command w/ a+d op codes set to 0 to specify use of access bank & wreg, also using BCF, BSF, & BTG clear, set, & toggle

=> MOVF 0X0020, 0, 0

```
BCF WREG 0, 0
BSF WREG 2, 0
BSF WREG 3, 0
BTG WREG 5, 0
BTG WREG 6, 0
```

2.) copy_data subroutine

```
=> counter equ 0X09
   adr1    equ 0X200
   adr2    equ 0X2A0
loop1
   DecF  counter, 1
   MOVFF adr1, adr2
   BNZ   loop1
         return
```

Question 3

The X option for the LIST compiler directive changes the .lst file by either expanding out or listing the lines that are executed within a macro. For example if the LIST option is set to on, then the lines that a macro contains will be explicitly shown in the .lst file and a 'M' will be printed on the lines that contain macro commands. If the option is set to off then these lines are not shown in the .lst file for every time the macro is called in the code.

Question 4

Figure 1 below shows the time measured for a toggled LED set to toggle every 1 ms. According to the figure, the timing is off by approximately 0.029 milliseconds. This es quite close to the stated 1ms time of the loop. This error is not surprising given how finicky and challenging it can be, depending on the number of instruction in a loop and how many cycles it takes each instruction to execute, to get a certain delay loop to align with a desired value perfectly.

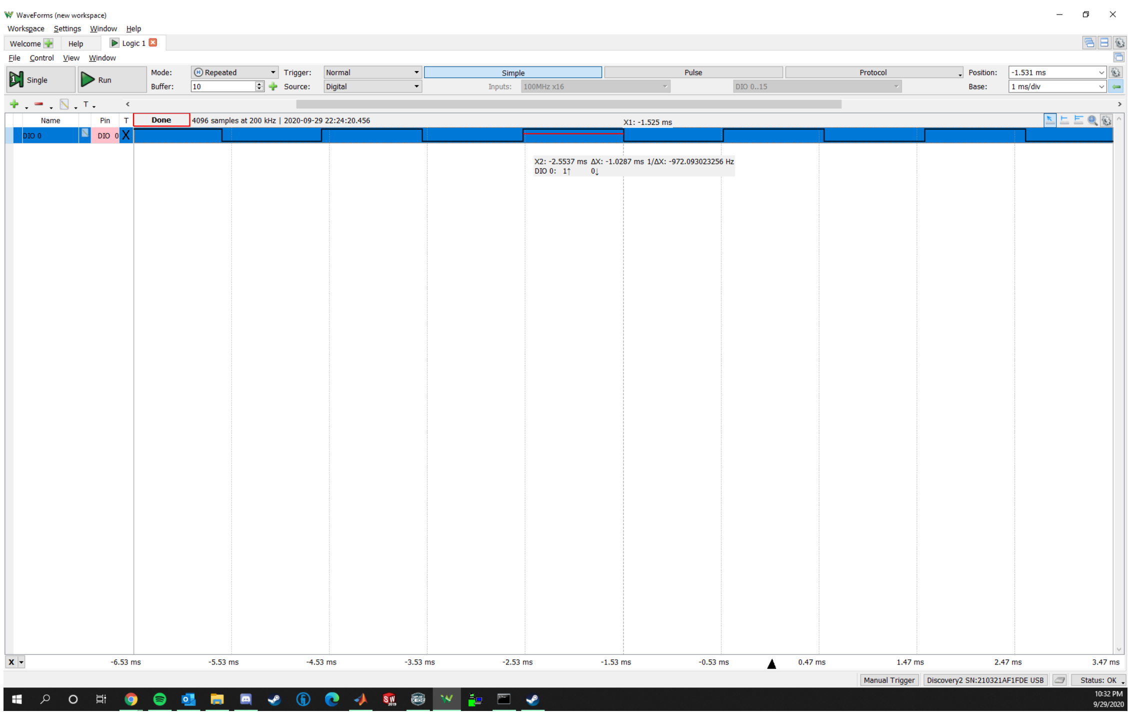

Fig. 1 This figure shows the elapsed time of the 1ms Delay loop within the given assembly code.

Question 5

Figure 2 below shows the elapsed time between a toggle of an LED when set to toggle every second. As can be seen in the figure, the elapsed time is shown to be 1.065 seconds, which lies within the $1 \pm 100ms$ bound given to us. This justifies that the ALIVE LED is being toggled by 1 second within the error bound, indicating the timing and structure of the source code delay loops is adequate/correct.

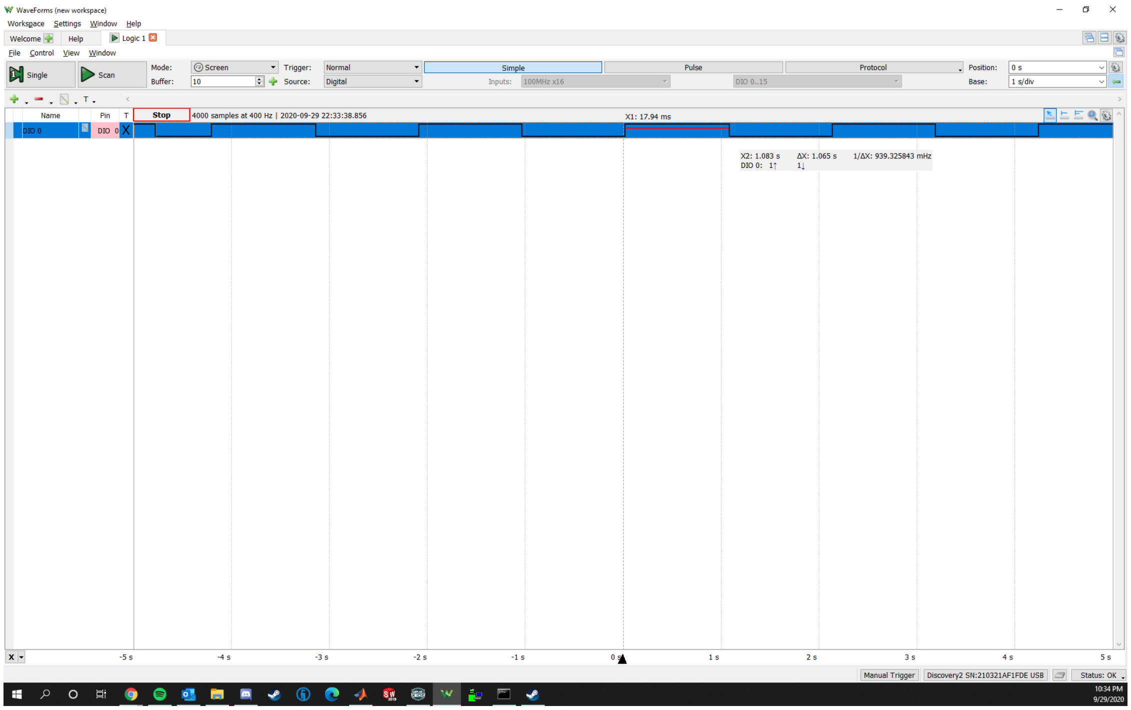

Fig. 2 This figure shows the elapsed time of the 1s Delay loop within the given assembly code.

Question 6

Switch bounce is essentially when the metal contact bounces momentarily in an electrical push button. This gives the illusions that the button was pressed, and then quickly released, on the order of milliseconds. If monitoring the status of this switch in a delay loop of some sort, and if the loop runs quickly enough, it may appear that the switch was pressed and released very quickly, when in fact it wasn't. In order to fix this issue, a small delay would be implemented directly after checking the status of the switch, where another status check would occur. This ensures the correct detection of a depressed switch that is still depressed. A delay of roughly 10-20 milliseconds is adequate enough, as this is how long switch bounce typically lasts for a mechanical switch.

Question 7 - Gaonkar 5.7

In order to set up PORTB as an input port and PORTC as an output port, as well as display 0x67 at PORTC, the following three instructions would work:

movlw B'11111111'
movwf TRISB
movlw B'00000000'
movwf TRISC
movlw 0x67
movwf LATC

Question 8 Gaonkar 5.22

According to the given code, because 50 is moved into reg2 and reg2 is decremented each iteration of the loop, the loop is executed 50 times.

Question 9 - Gaonkar 5.23

If BNZ is changed to BZ, then the loop will execute 1 time instead of 50 because after the first iteration, the the zero bit will not be '0', so the branch command will not be set, and the code will continue.

Question 10 - Gaonkar 5.29

For the loop given in question 2, not including the two commands before the loop, and ignoring the difference in the execution of the last cycle of the BNZ instruction, there are a total of $50 * 4 + 50 * 8$ cycles executed for the loop. Given a 10MHz = 10e6 Hz clock frequency, the delay time is $(50 * 4 + 50 * 8) * \frac{1}{10e6} = 6e - 5$ seconds delay for this loop.

Question 11

Question 12

The phase relationship between Ch A and Ch B is 90°, that is to say that they are 90 degrees out of phase. The binary sequence output when you turn it clockwise is 00 01 11 10 00. It is the reverse of this when you turn it counterclockwise, and is called the gray sequence. Since the RPG has 16 cycles per rotation and 4 different states per cycle, the cycle repeats 16 times per rotation and has 64 distinct states per rotation.

Question 13 - Gaonkar 5.26

Based on the given hex code given at 0020H and 0021H as E0 12, and using the data sheet for the PIC18F for the BZ command, the encoding takes the form of 1110 0000 nnnn nnnn, where after it is executed → PC+2+2n, where 2n is the 2's complement number. Given E0 12 for the problem, doing the calculation PC + 2 + 2n → 20 + 2 + 12 = 22H in memory location.

Question 14 - Gaonkar 5.30

If you account for the difference in the execution of the last cycle of the BNZ for the loop, there will be a difference of 4 cycles, which equates to $4 * \frac{1}{10e6} = 4e - 7$ seconds that the delay will change by.

Question 15 - Gaonkar 5.31

Taking in to account the single instruction for the last execution of the BRZ, as well as the two command before the loop, or order to get 150e-6 second delay → $(150 - e6 * 10e6) = 1.5e3$ instructions. Solving equation $2 + 3 * n - 1 = (150e - 6 * 10e6)$ for n yields a count of 500 to achieve this delay.