PIC Microcontroller (Lab Assignment)

1-second UP/DOWN-counter

Leonardo Fusser, 1946995

Experiment Performed on **27 November 2020**
Report Submitted on **13 December 2020**

Department of Computer Engineering Technology
Microcontroller & Microprocessor systems
Day Yann Fong

VANIER
CÉGEP / COLLEGE
Learning today Leading tomorrow

TABLE OF CONTENTS

1.0 Purpose

- Configure Timer0 module to have 1-second delay for UP/DOWN-counter.
- Implement UP/DOWN-counter functionality in ISR.
- Implement switch to select between UP/DOWN-counter functionality.
- Learn how to use and understand 7-segment display functionality.
- Learn how to use and understand 7448 7-segment driver/decoder functionality.
- Learn how to drive 7-segment display with 7448 7-segment driver/decoder.
- Implement 7-segment display with 7448 to PIC 16F887.
- Implement conditional check/evaluation statements using Assembly language.
- Use of MPLAB X IDE Simulator for debugging purposes.

2.0 Materials Used

- (1x) 7448 7-segment display driver/decoder
- (1x) common-cathode red 7-segment display
- (4x) Yellow LEDs
- (1x) PIC 16F887 RISC 8-bit microcontroller
- (1x) MPLAB PICKIT 4
- (1x) Push-button
- (1x) 4x dip-switch
- (7x) 220Ω resistors
- (2x) 10kΩ resistors
- (3x) 10V 100nF non-polarized capacitors
- (1x) Breadboard
- (?x) Wire

3.0 Original Design

Part A assembly code

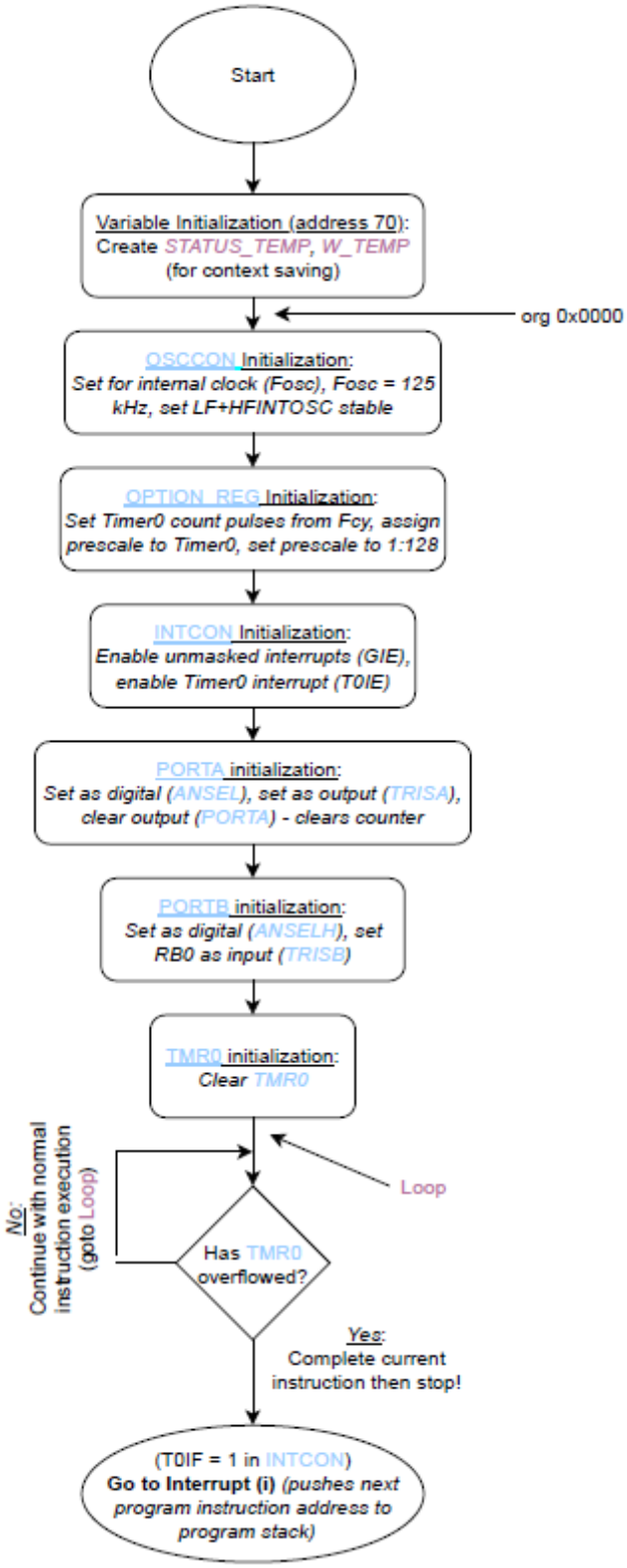
Part B assembly code

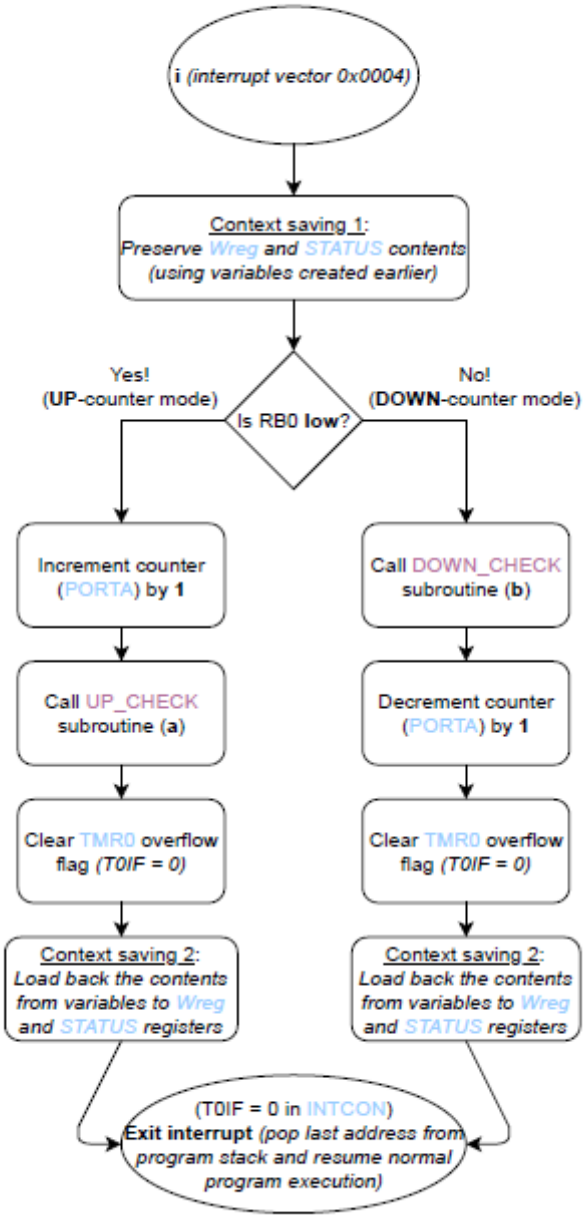
Part C assembly code

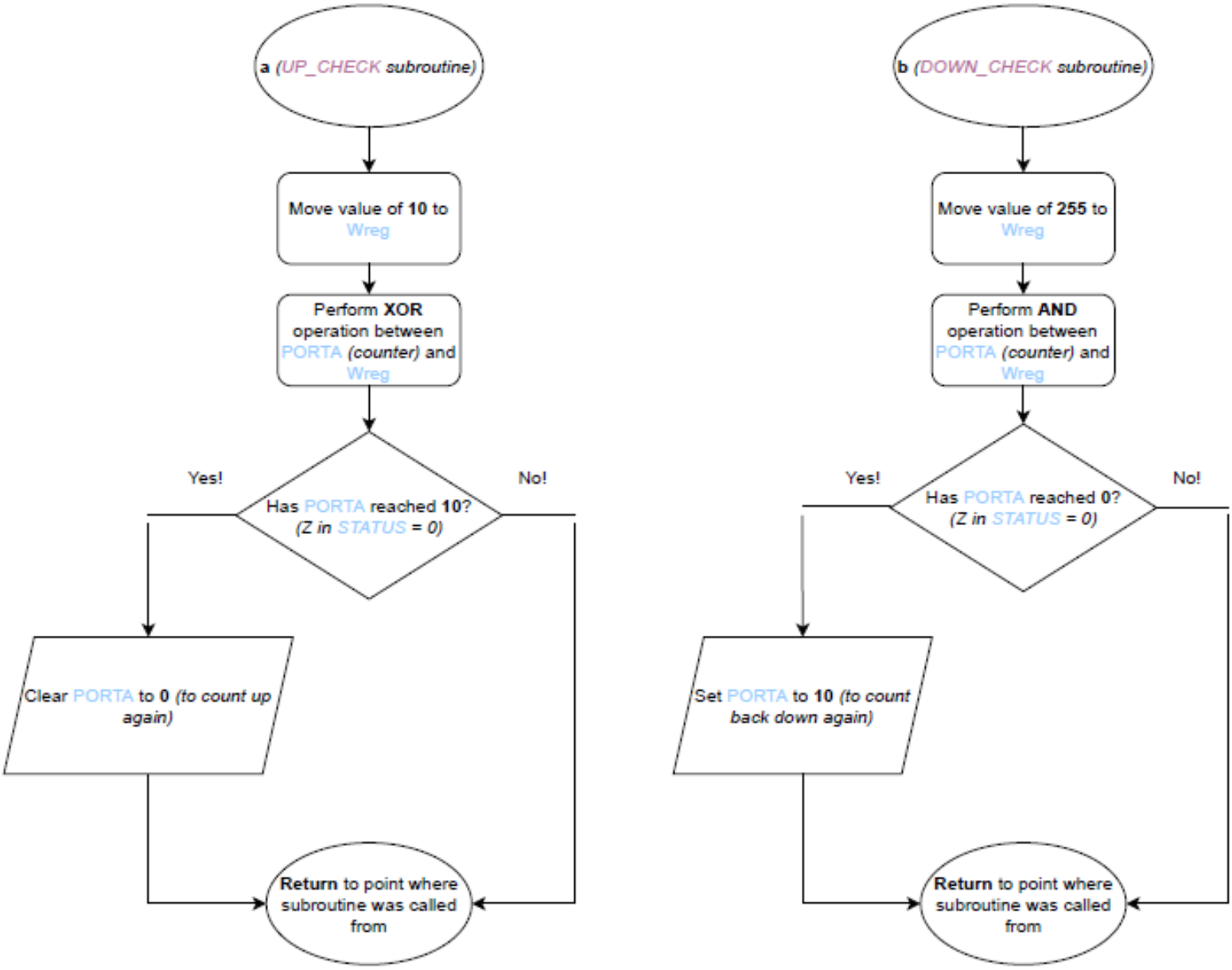
<u>Note</u>: *.asm* source files for Part A, B and C are attached to this submission for your convenience. They are not shown here due to complexity and length.

Microcontroller & Microprocessor systems
Day Yann Fong
Lab Assignment - Part C Flowchart (I)

Leonardo Fusser (1946995)

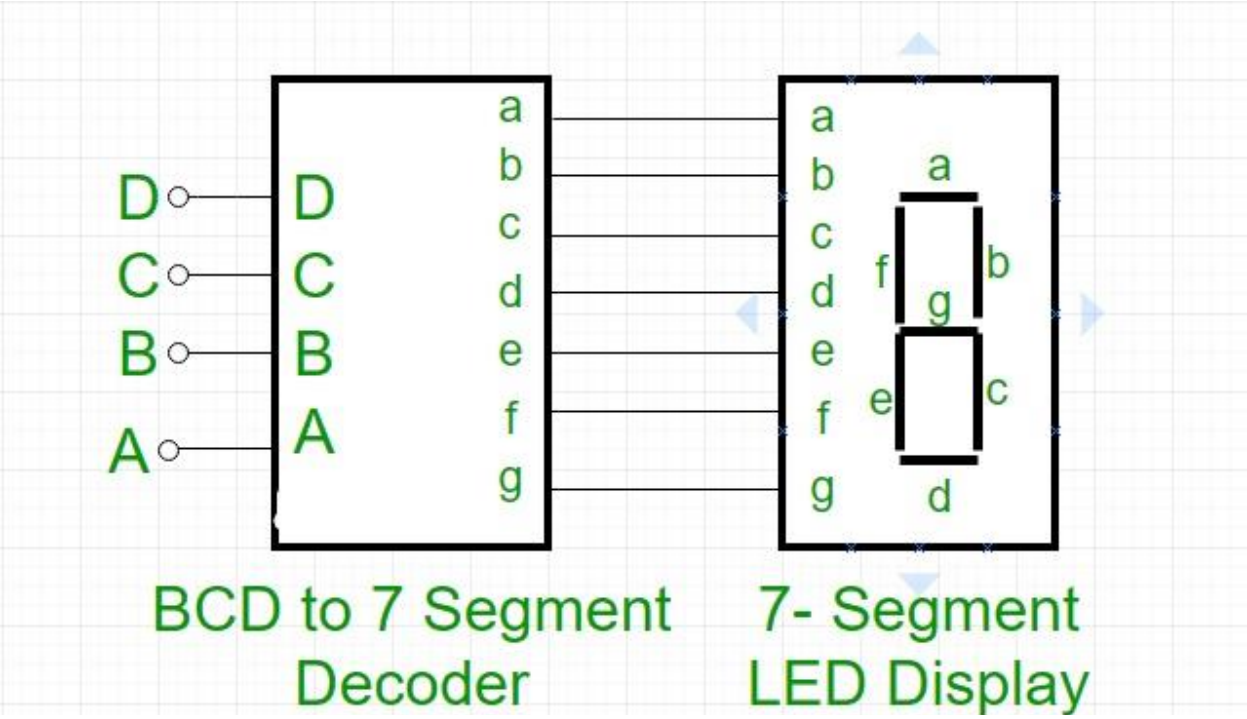




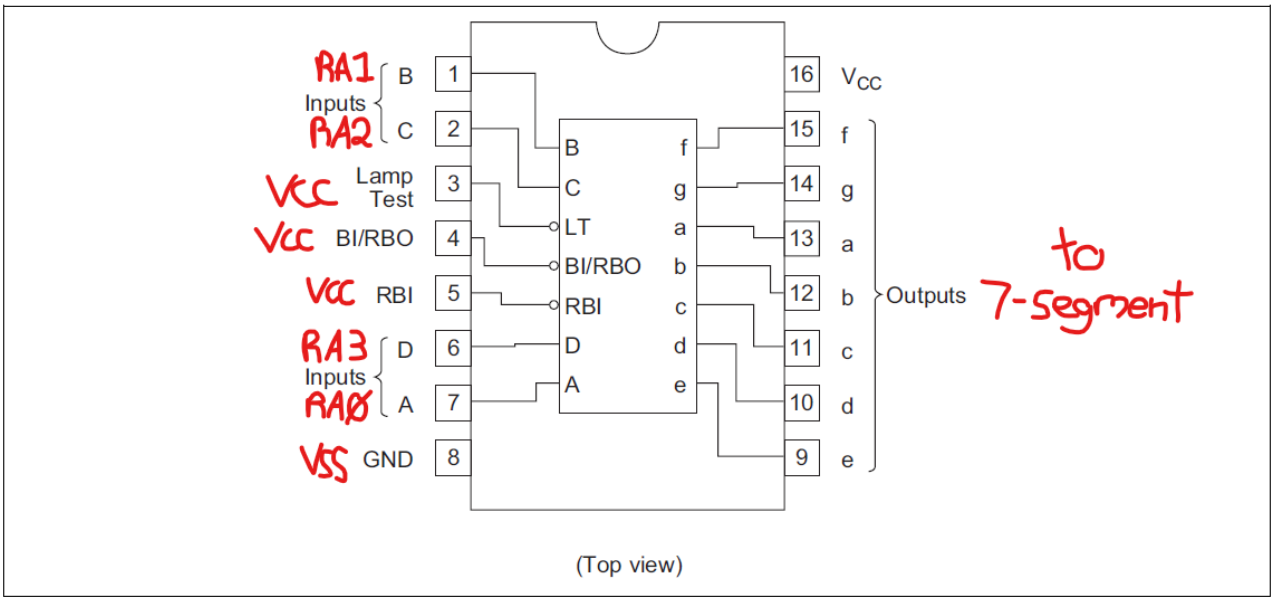


Note: ***Separate flowchart is attached to this report submission for your convenience. It is labeled "LabAssignment_LeonardoFusser_Flowchart.pdf".***

7-segment display and 7448 display driver wiring

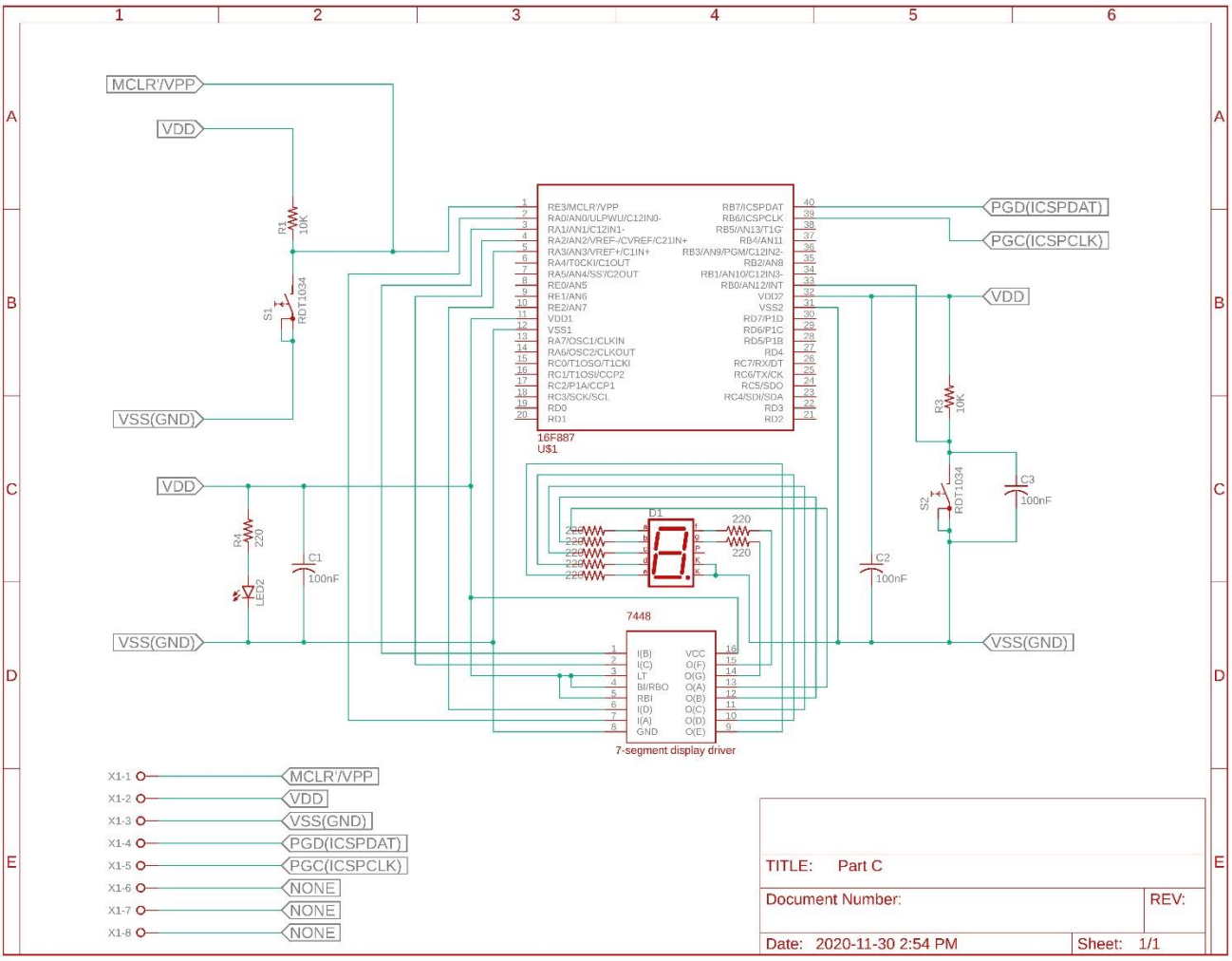


Pin Arrangement



Pins A to D (input) on 7448 are connected to pins RA0 to RA3 respectively on PIC 16F887. Pins a to g (output) on 7448 are connected to pins a to g (input) on 7-segment display. Pins 3 to 5 on 7448 connected to VDD (on PICKIT 4), this is shown later why in the truth table found under the "Questions" section of this report. 220Ω current-limiting resistors are placed between the outputs of the 7448 (a to g) and the inputs on the 7-segment display (a to g).

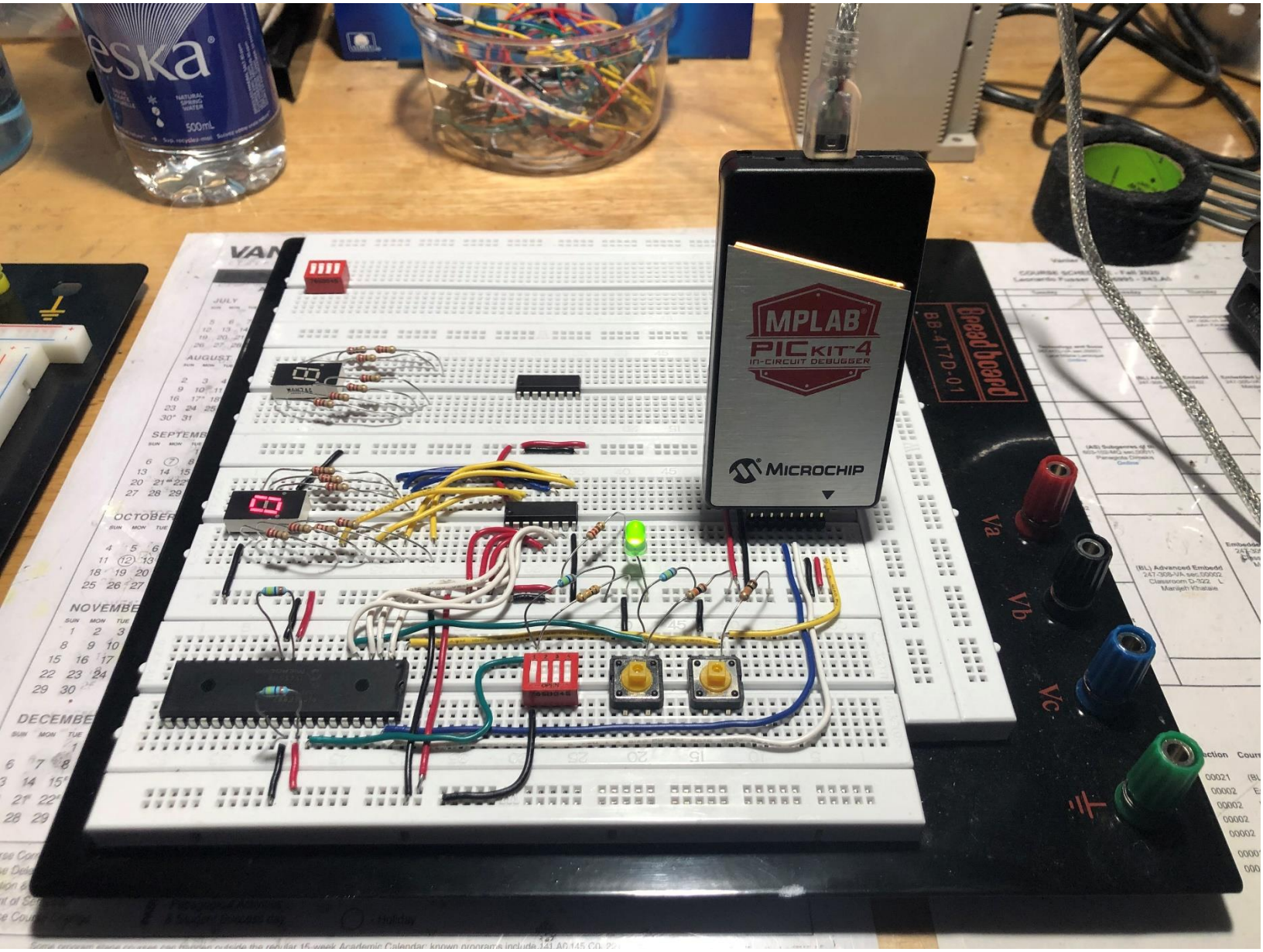
Final Circuit Schematic



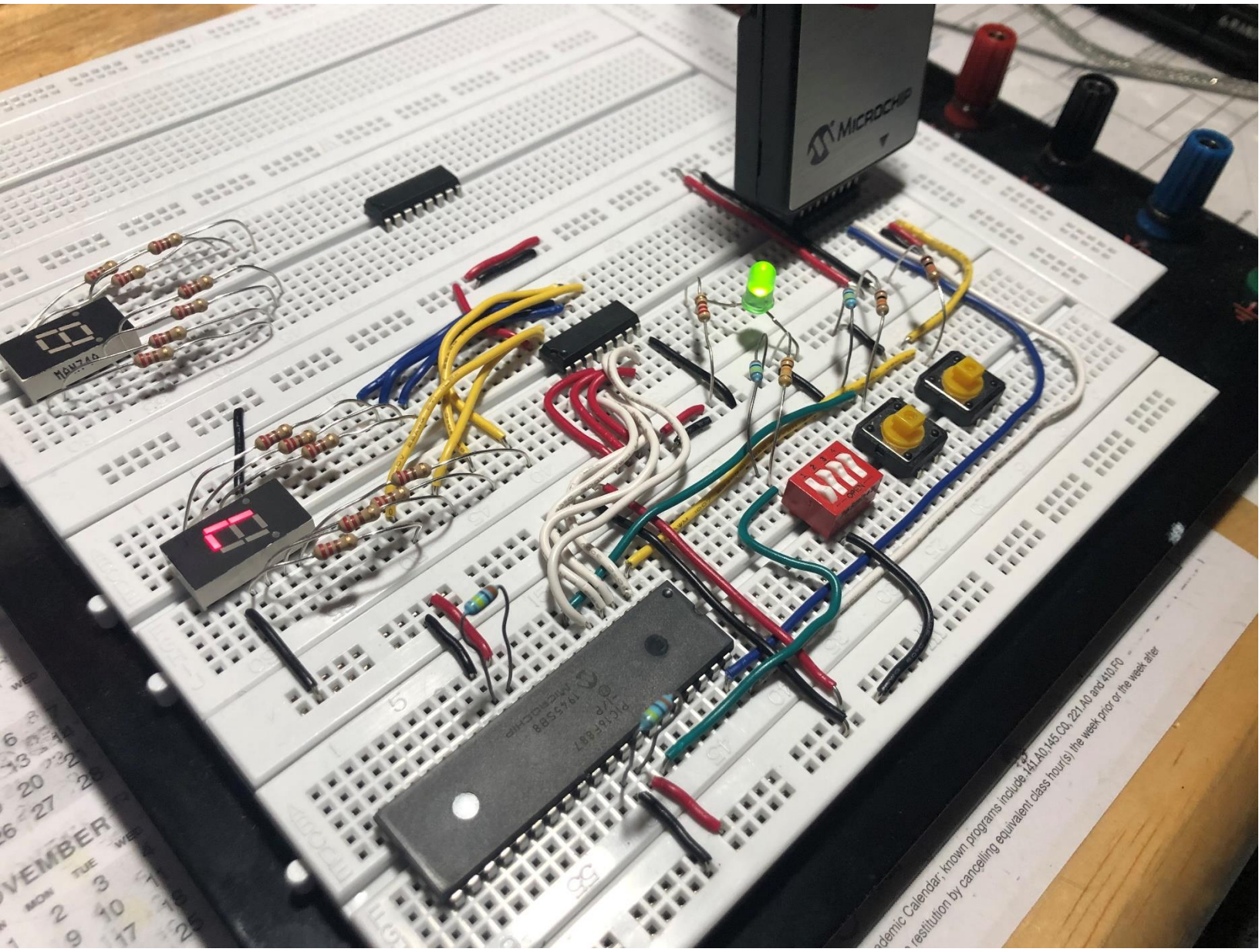
Complete final circuit schematic shown above. As mentioned from before, pins RA0 to RA3 interface to pins A to D (input) on the 7448. Pins a to g (output) on the 7448 connect to pins a to g (input) on the 7-segment display with 220Ω current-limiting resistors placed in between the two. A switch to select between UP/DOWN-count functionality is connected to pin RB0 on PIC 16F887. A 100nF capacitor is placed across the connections of the switch to ensure no switch noise during operation (filter). 100nF decoupling capacitors are placed across the VDD and VSS pins on the PIC 16F887. Reset circuitry (other switch) and power-up LED maintain the same design from previous labs. Essential connections are made to the PICKIT 4.

<u>Note</u>: ***Separate schematic is attached to this report submission for your convenience. It is labeled as "LabAssignment_LeonardoFusser_Schematic – Final.pdf".***

Final Circuit Prototype



Final circuit prototype shown above. Design taken directly from schematic shown before. Second 7-segment display and 7448 7-segment display driver/decoder shown above not used for this lab. First dip switch selects between UP/DOWN-count functionality, green LED is the power-up LED, push-button directly below green LED not being utilized and push-button next to other push-button is the reset button. Utilized 7-segment display is located directly above PIC 16F887 and utilized 7448 IC is located to the right of the 7-segment display (with 220Ω current limiting resistors in between).

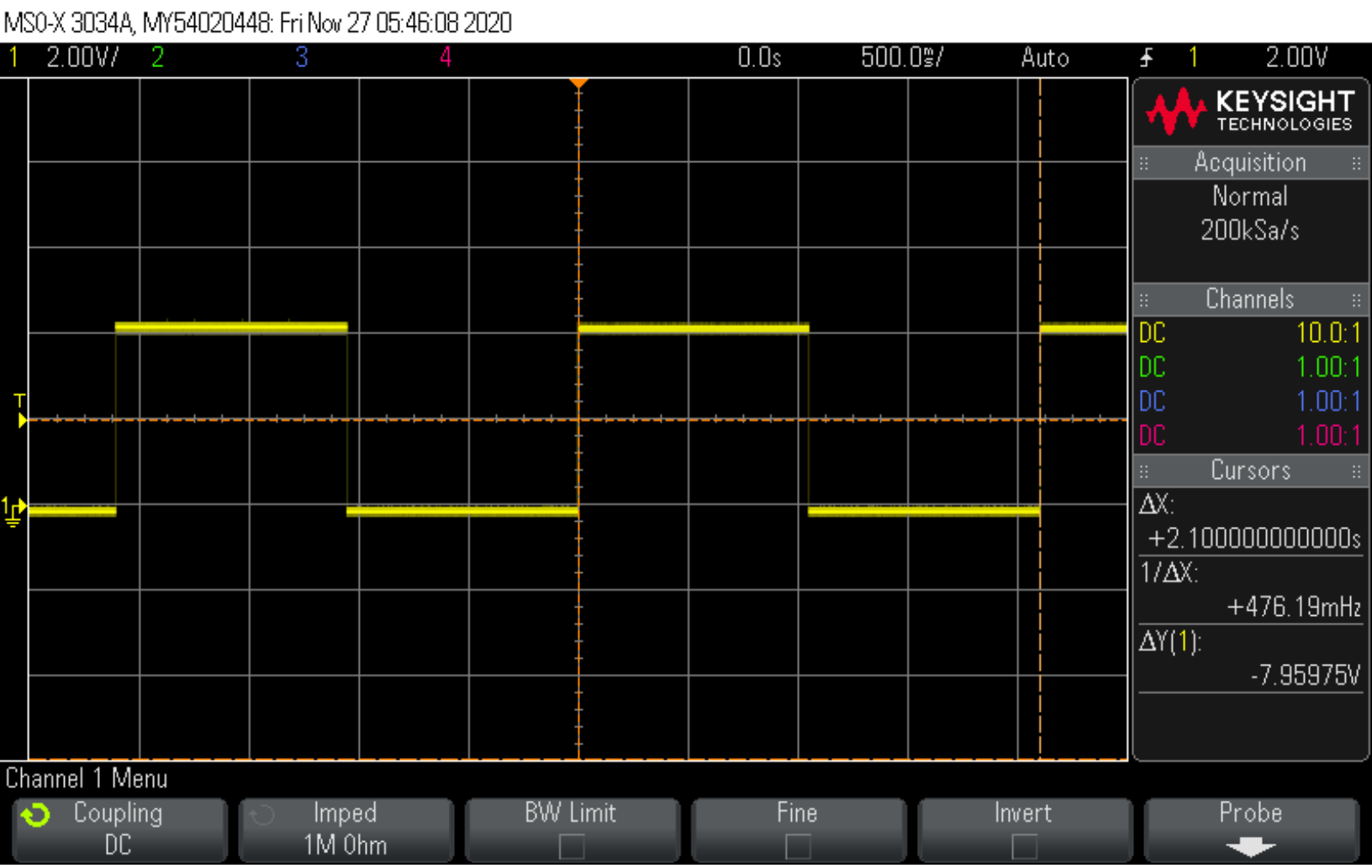


Final circuit prototype shown above. Design taken directly from schematic shown before. Second 7-segment display and 7448 7-segment display driver/decoder shown above not used for this lab. First dip switch selects between UP/DOWN-count functionality, green LED is the power-up LED, push-button directly below green LED not being utilized and push-button next to other push-button is the reset button. Utilized 7-segment display is located directly above PIC 16F887 and utilized 7448 IC is located to the right of the 7-segment display (with 220Ω current limiting resistors in between).

3.0 OBSERVATIONS

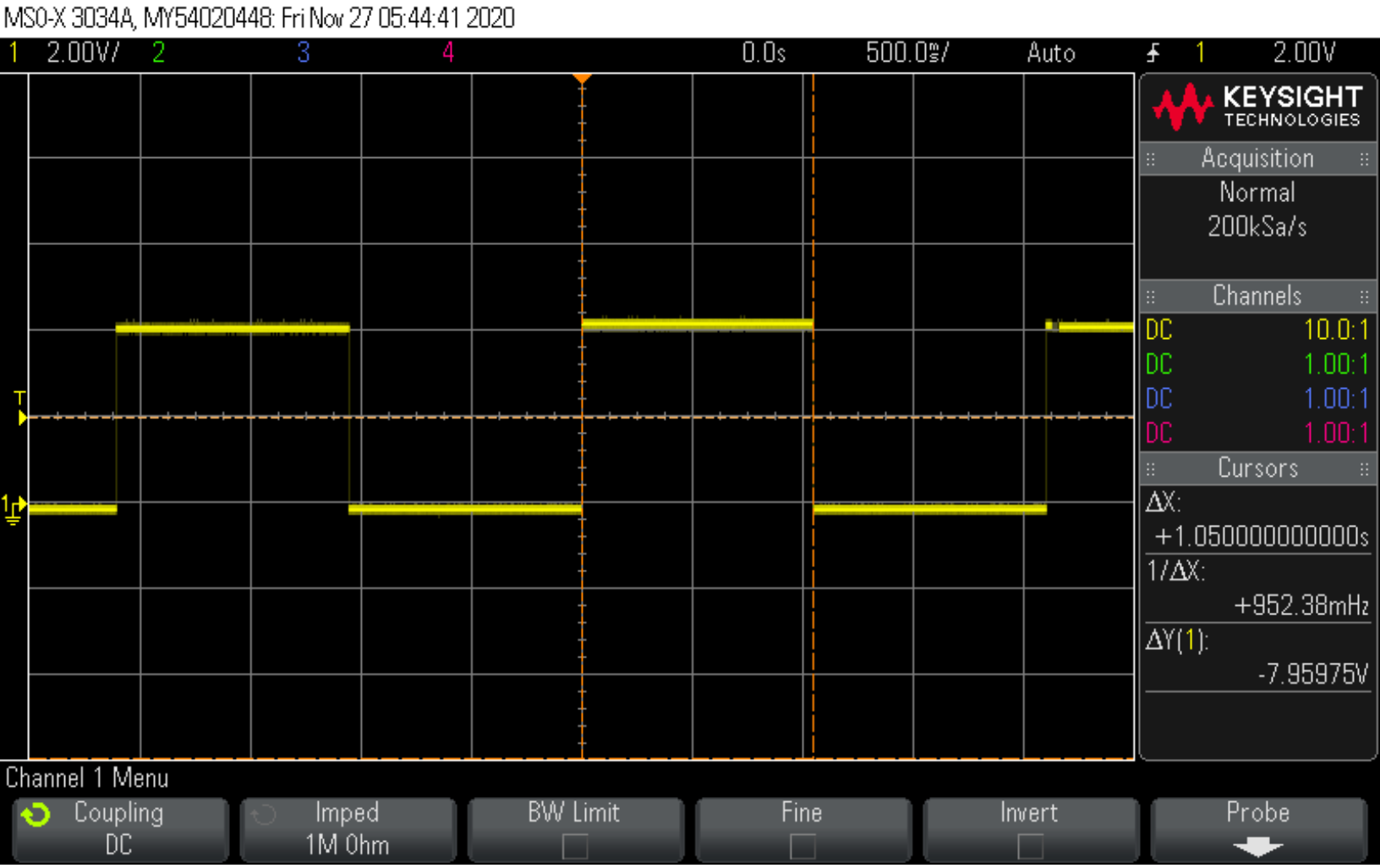
Observations from the lab

Part A, 2:



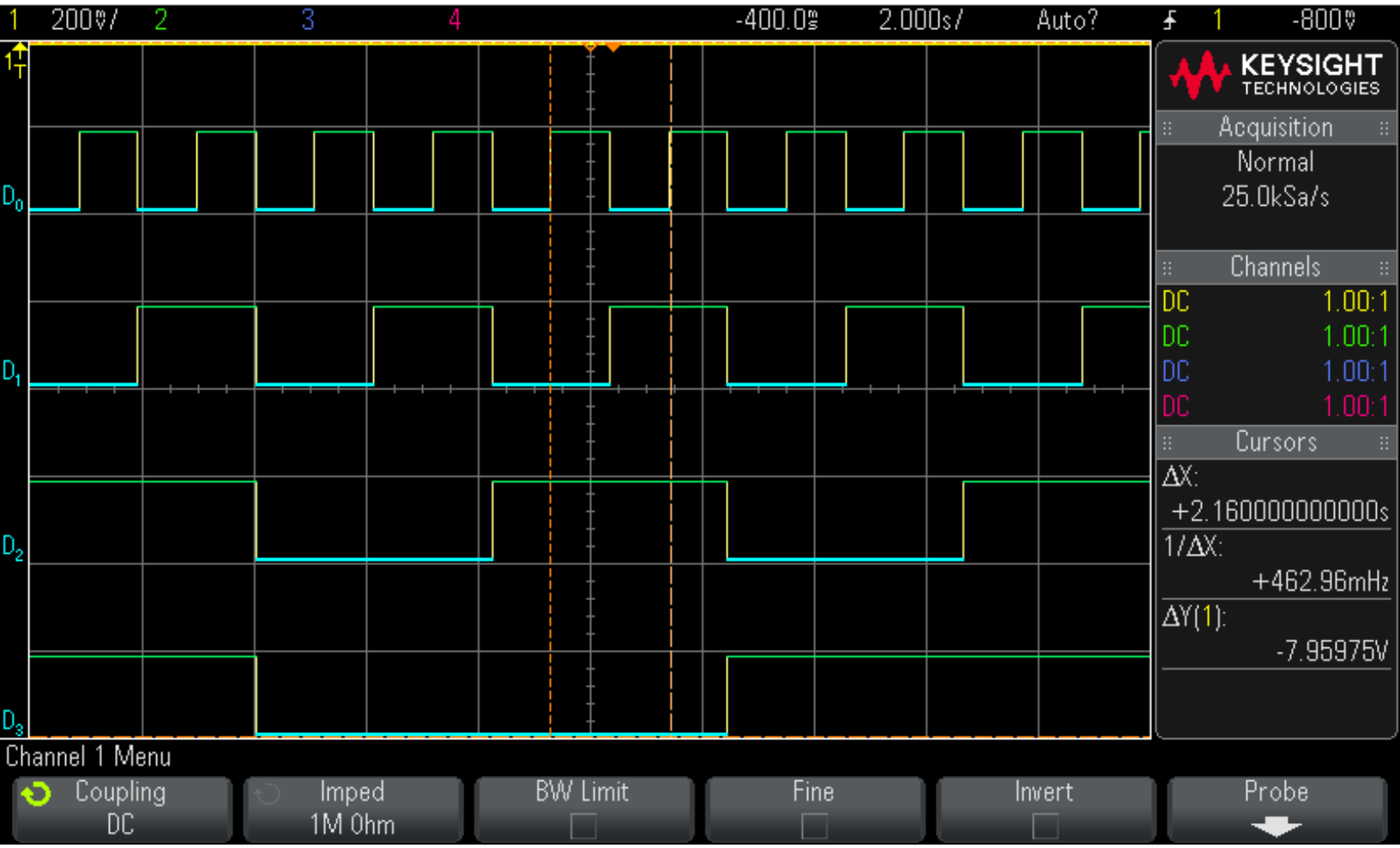
Scope settings: 2V per-division (vertical) and 500mS per-division (horizontal)

Oscilloscope screen-shot capture for Part A circuit. Probe is connected to pin RA0 on PIC 16F887. Vpp is almost 5V and Frequency is almost 500mHz (based on cursor setting). Period is around 2-seconds and pulse width is around 1-second. Counter is being incremented by almost 1-second intervals (as shown by LED connected to pin RA0 on PIC 16F887 and oscilloscope capture from above). The same 1-second intervals can be observed for Parts B and C of this Lab. (Cursors above show 2-second delay period)



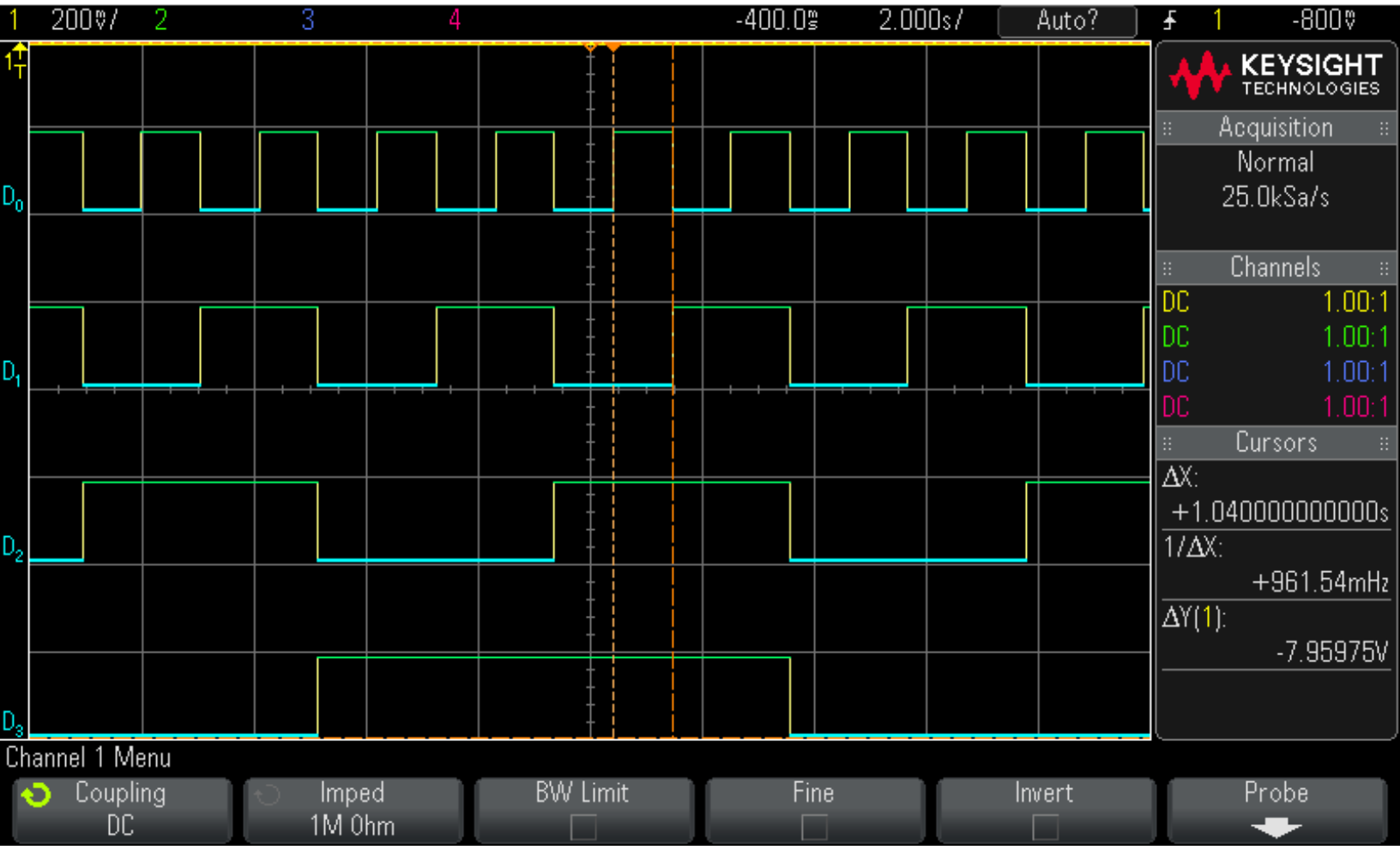
Scope settings: 2V per-division (vertical) and 500mS per-division (horizontal)

Oscilloscope screen-shot capture for Part A circuit. Probe is connected to pin RA0 on PIC 16F887. Vpp is almost 5V and Frequency is almost 1Hz (based on cursor setting). Period is around 2-seconds and pulse width is around 1-second. Counter is being incremented by almost 1-second intervals (as shown by LED connected to pin RA0 on PIC 16F887 and oscilloscope capture from above). The same 1-second intervals can be observed for Parts B and C of this Lab. (Cursors above show 1-second delay pulse width)



Scope settings: using logic analyzer

Oscilloscope screen-shot capture for Part A circuit. Logic analyzer probes are connected to RA0 to RA3 on PIC 16F887. D0 is RA0, D1 is RA1, D2 is RA2 and D3 is RA3 shown above. Period is around 2-seconds and pulse width is around 1-second. Counter is being incremented by almost 1-second intervals (as shown by LED connected to pin RA0 on PIC 16F887 and capture from logic analyzer above). Other timings are shown for pins RA1 to RA3 (D1 to D3 on logic analyzer). The same 1-second intervals can be observed for Parts B and C of this Lab. Frequency remains around 500mHz (based on cursor setting). (Cursors above show 2-second delay period)



Scope settings: using logic analyzer

Oscilloscope screen-shot capture for Part A circuit. Logic analyzer probes are connected to RA0 to RA3 on PIC 16F887. D0 is RA0, D1 is RA1, D2 is RA2 and D3 is RA3 shown above. Period is around 2-seconds and pulse width is around 1-second. Counter is being incremented by almost 1-second intervals (as shown by LED connected to pin RA0 on PIC 16F887 and capture from logic analyzer above). Other timings are shown for pins RA1 to RA3 (D1 to D3 on logic analyzer). The same 1-second intervals can be observed for Parts B and C of this Lab. Frequency remains around 1Hz (based on cursor setting). (Cursors above show 1-second delay pulse width)

Part B, 5c) / Part C, 6:

Observation table for 4-bit binary UP counter (0 – 9)				
Decimal value	*(Value of 00001010 is moved to Wreg when subroutine is called)*		Z flag in STATUS	PORTA goes back to 0?
	Value of counter (PORTA)	Result after xorwf PORTA,w		
0	00000000	00001010	0	No
1	00000001	00001011	0	No
2	00000010	00001000	0	No
3	00000011	00001001	0	No
4	00000100	00001110	0	No
5	00000101	00001111	0	No
6	00000110	00001100	0	No
7	00000111	00001101	0	No
8	00001000	00000010	0	No
-> 9	00001001	00000011	0	No
10**	00001010	00000000	1	Yes!

*Note: for Z flag in STATUS register, 0 -> not set (no zero) & 1 -> is set (is zero)!

**Although not shown on 7-segment, it does count up to 10 in background.

For my UP-counter, this table reflects the observed behavior from my code. During normal program execution, an XOR operation is performed between the working register (value of 10) and the current value of the counter after the counter has incremented by 1. If the result of the XOR operation yields true, meaning that my counter has reached 10, the zero flag (Z) is set in the STATUS register indicating that the counter has reached 10 and that the program should count back up from 0 again. The reason why the program waits until the UP-counter reaches 10 is because there is small delay (1-2 cycles) when displaying the result to the 7-segment display. The "->" indicates this is the last value that is seen on the 7-segment display before restarting counting from 0 again and again. What is not shown here is the behavior of my switch connected to pin RB0 on the PIC 16F887. Basically, when the switch is in the "closed position", the program will run in UP-counter mode. When the switch is in the "open position", the program will run in the DOWN-counter mode. The user has the ability to toggle between the two modes during program execution and the UP/DOWN-counter will not reset itself either to 0 (for UP-counter) or 9 (for DOWN-counter).

Observation table for 4-bit binary DOWN counter (9 - 0)				
Decimal value	*(Value of 11111111 is moved to Wreg when subroutine is called)*		Z flag in STATUS	PORTA goes back to 9?
	Value of counter (PORTA)	Result after andwf PORTA,w		
10**	00001010	00001010	0	No
-> 9	00001001	00001001	0	No
8	00001000	00001000	0	No
7	00000111	00000111	0	No
6	00000110	00000110	0	No
5	00000101	00000101	0	No
4	00000100	00000100	0	No
3	00000011	00000011	0	No
2	00000010	00000010	0	No
1	00000001	00000001	0	No
0	00000000	00000000	1	Yes!

*Note: for Z flag in STATUS register, 0 -> not set (no zero) & 1 -> is set (is zero)!

**Although not shown on 7-segment, it does count down from 10 in background.

For my DOWN-counter, this table reflects the observed behavior from my code. During normal program execution, an AND operation is performed between the working register (value of 255) and the current value of the counter before the counter has decremented by 1. If the result of the AND operation yields true, meaning that my counter has reached 0, the zero flag (Z) is set in the STATUS register indicating that the counter has reached 0 and that the program should count back down from 9 again. The reason why the program loads a value of 10 for the DOWN-counter when counting down to 0 again is because there is small delay (1-2 cycles) when displaying the result to the 7-segment display. The "->" indicates this is the first value that is seen on the 7-segment display before counting down from 9 again and again. What is not shown here is the behavior of my switch connected to pin RB0 on the PIC 16F887. Basically, when the switch is in the "closed position", the program will run in UP-counter mode. When the switch is in the "open position", the program will run in the DOWN-counter mode. The user has the ability to toggle between the two modes during program execution and the UP/DOWN-counter will not reset itself either to 0 (for UP-counter) or 9 (for DOWN-counter).

TABLE 15-2: PIC16F882/883/884/886/887 INSTRUCTION SET

Mnemonic, Operands		Description	Cycles	14-Bit Opcode MSb LSb	Status Affected	Notes
BYTE-ORIENTED FILE REGISTER OPERATIONS						
ADDWF	f, d	Add W and f	1	00 0111 dfff ffff	C, DC, Z	1, 2
ANDWF	f, d	AND W with f	1	00 0101 dfff ffff	Z	1, 2
CLRF	f	Clear f	1	00 0001 lfff ffff	Z	2
CLRW	–	Clear W	1	00 0001 0xxx xxxx	Z	
COMF	f, d	Complement f	1	00 1001 dfff ffff	Z	1, 2
DECF	f, d	Decrement f	1	00 0011 dfff ffff	Z	1, 2
DECFSZ	f, d	Decrement f, Skip if 0	1(2)	00 1011 dfff ffff		1, 2, 3
INCF	f, d	Increment f	1	00 1010 dfff ffff	Z	1, 2
INCFSZ	f, d	Increment f, Skip if 0	1(2)	00 1111 dfff ffff		1, 2, 3
IORWF	f, d	Inclusive OR W with f	1	00 0100 dfff ffff	Z	1, 2
MOVF	f, d	Move f	1	00 1000 dfff ffff	Z	1, 2
MOVWF	f	Move W to f	1	00 0000 lfff ffff		
NOP	–	No Operation	1	00 0000 0xx0 0000		
RLF	f, d	Rotate Left f through Carry	1	00 1101 dfff ffff	C	1, 2
RRF	f, d	Rotate Right f through Carry	1	00 1100 dfff ffff	C	1, 2
SUBWF	f, d	Subtract W from f	1	00 0010 dfff ffff	C, DC, Z	1, 2
SWAPF	f, d	Swap nibbles in f	1	00 1110 dfff ffff		1, 2
XORWF	f, d	Exclusive OR W with f	1	00 0110 dfff ffff	Z	1, 2

Picture taken directly from page 227 of the PIC 16F887 datasheet. The first red circle (to the left) shows the XOR operation that was used for the conditional evaluation in my UP-counter. The other red circle (to the right) shows the status flag that is affected by this operation (Z flag in STATUS register). As mentioned before, when this flag is set, it will tell the program that the counter has reached its threshold and that certain action must occur (for UP-counter, it was to restart up counting from 0).

TABLE 15-2: PIC16F882/883/884/886/887 INSTRUCTION SET

Mnemonic, Operands		Description	Cycles	14-Bit Opcode MSb LSb	Status Affected	Notes
BYTE-ORIENTED FILE REGISTER OPERATIONS						
ADDWF	f, d	Add W and f	1	00 0111 dfff ffff	C, DC, Z	1, 2
ANDWF	f, d	AND W with f	1	00 0101 dfff ffff	Z	1, 2
CLRF	f	Clear f	1	00 0001 lfff ffff	Z	2
CLRW	–	Clear W	1	00 0001 0xxx xxxx	Z	
COMF	f, d	Complement f	1	00 1001 dfff ffff	Z	1, 2
DECF	f, d	Decrement f	1	00 0011 dfff ffff	Z	1, 2
DECFSZ	f, d	Decrement f, Skip if 0	1(2)	00 1011 dfff ffff		1, 2, 3
INCF	f, d	Increment f	1	00 1010 dfff ffff	Z	1, 2
INCFSZ	f, d	Increment f, Skip if 0	1(2)	00 1111 dfff ffff		1, 2, 3
IORWF	f, d	Inclusive OR W with f	1	00 0100 dfff ffff	Z	1, 2
MOVF	f, d	Move f	1	00 1000 dfff ffff	Z	1, 2
MOVWF	f	Move W to f	1	00 0000 lfff ffff		
NOP	–	No Operation	1	00 0000 0xx0 0000		
RLF	f, d	Rotate Left f through Carry	1	00 1101 dfff ffff	C	1, 2
RRF	f, d	Rotate Right f through Carry	1	00 1100 dfff ffff	C	1, 2
SUBWF	f, d	Subtract W from f	1	00 0010 dfff ffff	C, DC, Z	1, 2
SWAPF	f, d	Swap nibbles in f	1	00 1110 dfff ffff		1, 2
XORWF	f, d	Exclusive OR W with f	1	00 0110 dfff ffff	Z	1, 2

Picture taken directly from page 227 of the PIC 16F887 datasheet. The first red circle (to the left) shows the AND operation that was used for the conditional evaluation in my DOWN-counter. The other red circle (to the right) shows the status flag that is affected by this operation (Z flag in STATUS register). As mentioned before, when this flag is set, it will tell the program that the counter has reached its threshold and that certain action must occur (for DOWN-counter, it was to restart down counting from 9).

4.0 QUESTIONS

Questions from the lab

Part A:

1) In order to implement a 1-second delay, I needed to slow down the TMR0 counter overflow time so that it will show around a 1-second delay on the LEDs (after each incrementation). To do this, I had to assign a prescale to the Timer0 module. Since the PIC 16F887 has multiple options for prescale, some calculations were needed to get the correct settings for our Timer0 prescale bits. I knew that the delay has to be 1-second so that means my delay period will be around 2-seconds and the frequency (f = 1/T) should be around 500mHz (if you look at the scope screenshots under "Observations", you'll see that this is true). Furthermore, I had to play around with the PIC's internal clock rate (Fosc) in order to find one that will work correctly to get a 1-second delay. Here are the calculations needed to be done in order to obtain the 1-second delay:

$$F_{OSC} = 1MHz\ (F_{CY} = 250kHz)$$

$$Time\ needed\ to\ complete\ 1\ instruction\ cycle = \frac{1}{F_{OSC}} * 4 = \frac{1}{1MHz} * 4 = 1uS$$

$$Time\ needed\ to\ generate\ 1\ second\ delay = \frac{1\ second}{1uS} = 1'000'000uS$$

$$Prescale\ settings\ needed = \frac{1'000'000}{x} = 255 \rightarrow x = 3'921\ (1{:}\ 3'921\ assigned\ to\ Timer0)$$

These settings above will not work because PIC 16F887 doesn't have a 1:3'921 prescale ratio for the Timer0 module. Another Fosc must be used. To shorten the possible combinations, here is the one I found that worked for this lab:

$$F_{OSC} = 125kHz\ (F_{CY} = 31.25kHz)$$

$$Time\ needed\ to\ complete\ 1\ instruction\ cycle = \frac{1}{F_{OSC}} * 4 = \frac{1}{125kHz} * 4 = 32uS$$

$$Time\ needed\ to\ generate\ 1\ second\ delay = \frac{1\ second}{32uS} = 31'250uS$$

$$Prescale\ settings\ needed = \frac{31'250}{x} = 255 \rightarrow x = 122.54\ (1{:}\ 122.54\ assigned\ to\ Timer0)$$

These settings above will work because PIC 16F887 has a 1:128 prescale (closest value is this from 122.54) ratio for the Timer0 module. Although the prescale isn't set at exactly at 1:122.54, a delay of around 1-second will still be produced. In order to get exactly a 1-second delay, a value must be pre-loaded into the TMR0 register.

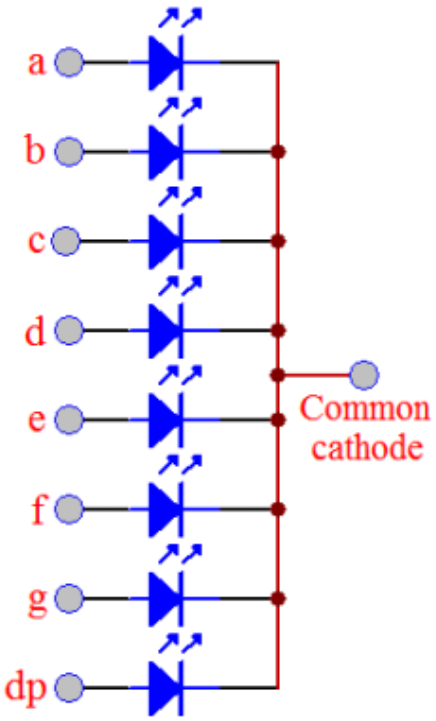
3) As mentioned briefly before, in order to attempt to achieve exactly a 1-second delay, a value must be pre-loaded into the TMR0 register. Without any value being loaded into the TMR0 register, a delay of around 1-second will still be generated. As it is shown in the scope screenshots under "Observations", the generated delay was 1.05 seconds. This is not noticeable unless being viewed with an oscilloscope. To implement an approximate 1-second delay, a value will need to be pre-loaded into the TMR0 during initialization of the PIC 16F887 and upon leaving the ISR. Although it wasn't implemented in my code, below is an example how a value can be pre-loaded into the TMR0:

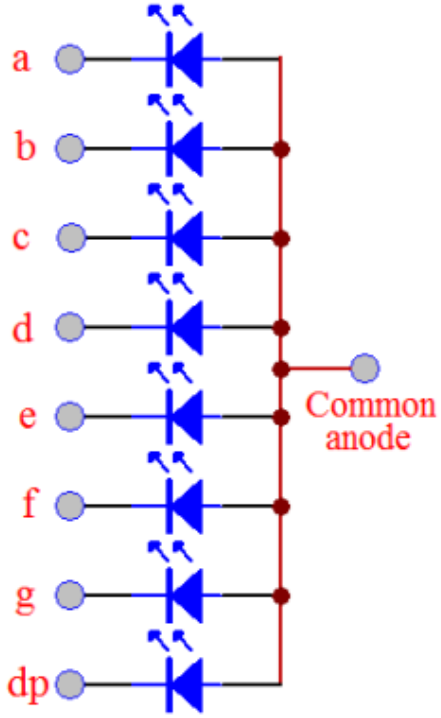
```
;Pre-loading a value into TMR0 to achieve aprox. 1-second delay
banksel     TMR0            ; Selects the TMR0 register.
clrf        TMR0            ; Clears TMR0 register (initialization).
movlw       h'F0'           ; Moves .240 to the Wreg.
movwf       TMR0            ; .240 is loaded into the TMR0 register (from Wreg).
clrw                        ; Clears Wreg.
```

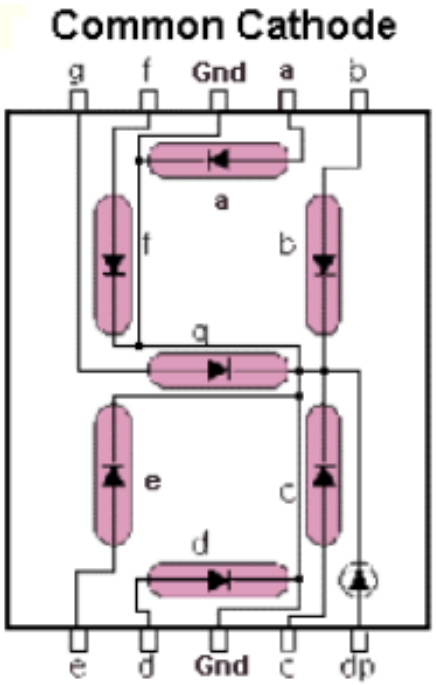
How to preload a value in the TMR0 register to achieve approximately a 1-second delay

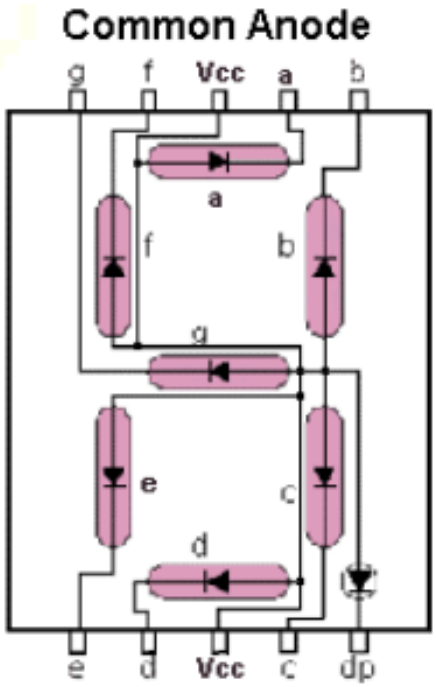
Part B:

5a)


a
b
c
d
e
f
g
dp
Common cathode


a
b
c
d
e
f
g
dp
Common anode


Common Cathode
g f Gnd a b
a
f
b
g
e
c
d
e d Gnd c dp


Common Anode
g f Vcc a b
a
f
b
g
e
c
d
e d Vcc c dp

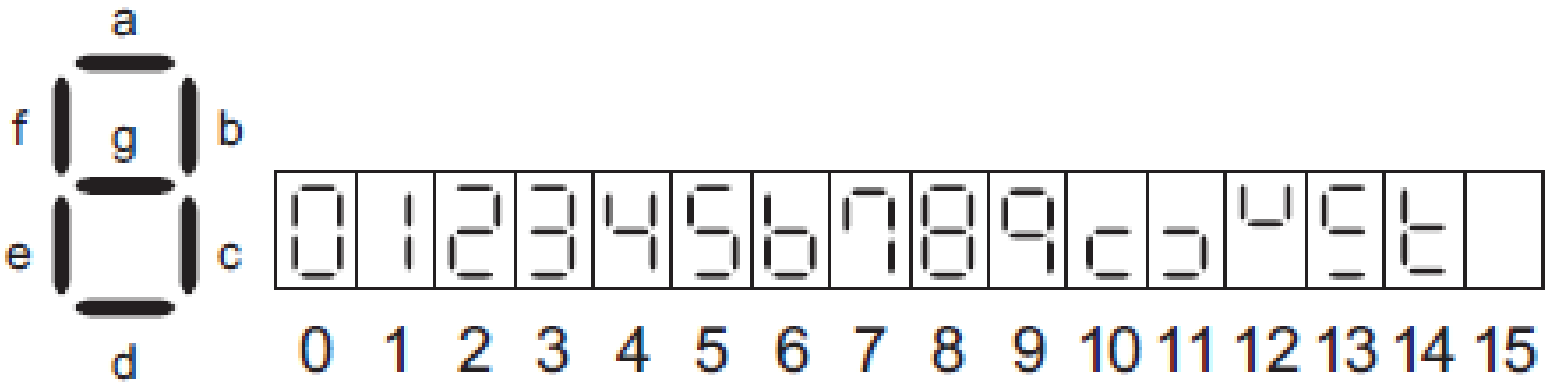
Explanation: Upon receiving my 7-segment display unit, I first looked up the datasheet for it. After looking at the datasheet, it was concluded that the unit I received was a common cathode unit. The two pictures shown above are how a common cathode and a common anode 7-segment's internal wiring is configured. In a common cathode configuration, the ground connection is "common" (shared) among all the segments on the display. In a common anode configuration, the VCC connection (power) is "common" (shared) among all the segments on the display. In our circuitry, since we have a common cathode unit, a logical HIGH (1) output from one of the outputs on the 7448 7-segment display driver/decoder connected to one of the segments on the 7-segment display will light up that specific segment. If we were using a common anode unit, a logical LOW (0) output from one of the outputs on the 7448 7-segment display driver/decoder connected to one of the segments on the 7-segment display will light up that specific segment (this is due to the common anode configuration). In that case, the 7448 would have to sink the current of the specific segment.

b)

Function Table

Decimal or Function	Inputs						BI / RBO	Outputs							Note
	LT	RBI	D	C	B	A		a	b	c	d	e	f	g	
0	H	H	L	L	L	L	H	H	H	H	H	H	H	L	1
1	H	X	L	L	L	H	H	L	H	H	L	L	L	L	
2	H	X	L	L	H	L	H	H	H	L	H	H	L	H	
3	H	X	L	L	H	H	H	H	H	H	H	L	L	H	
4	H	X	L	H	L	L	H	L	H	H	L	L	H	H	
5	H	X	L	H	L	H	H	H	L	H	H	L	H	H	
6	H	X	L	H	H	L	H	L	L	H	H	H	H	H	
7	H	X	L	H	H	H	H	H	H	H	L	L	L	L	
8	H	X	H	L	L	L	H	H	H	H	H	H	H	H	
9	H	X	H	L	L	H	H	H	H	H	L	L	H	H	
10	H	X	H	L	H	L	H	L	L	L	H	H	L	H	
11	H	X	H	L	H	H	H	L	L	H	H	L	L	H	
12	H	X	H	H	L	L	H	L	H	L	L	L	H	H	
13	H	X	H	H	L	H	H	H	L	L	H	L	H	H	
14	H	X	H	H	H	L	H	L	L	L	H	H	H	H	
15	H	X	H	H	H	H	H	L	L	L	L	L	L	L	
BI	X	X	X	X	X	X	L	L	L	L	L	L	L	L	2
RBI	H	L	L	L	L	L	L	L	L	L	L	L	L	L	3
LT	L	X	X	X	X	X	H	H	H	H	H	H	H	H	4

H; high level, L; low level, X, irrelevant



Explanation: First thing that was done to determine the behavior of the 7448 7-segment display driver/decoder was to look up the corresponding datasheet. For our specific display driver, the numerical designations are from 0 to 15 (second picture shown above). The resultant displays from the 7448 are shown under the "outputs" of the truth table (first picture shown above). A 4-bit binary sequence from 0000 to 1111 (these are the "inputs") will cause the numerical designations 0 to 15 to show up on the 7-segment display (first picture shown above). For our circuit behavior, we will only see the first 9 numbers (0 to 9) to show up on the display (shown in red in first picture above).

5.0 Conclusion

- Purpose of this lab has been achieved.
- Understood how to get 1-second delay from Timer0 module.
- Understood how to implement UP/DOWN-counter functionality in ISR.
- Understood how to implement a switch to select between UP/DOWN-counter functionality.
- Understood how to use a 7-segment display and theory behind it.
- Understood how to use a 7448 7-segment driver/decoder and theory behind it.
- Understood how to drive a 7-segment display using a 7448 7-segment display driver and theory behind it.
- Understood how to implement 7-segment circuity with 7448 to PIC 16F887 microcontroller.
- Understood how to implement a conditional evaluation statement in Assembly using byte-oriented file register operations (ANDF, XORWF) and bit-oriented file register operations (BTFSC).
- Understood how to use MPLAB X IDE simulator to debug and find any logical problems.
- Problem: UP and DOWN-counter were not counting up/down to correct values causing 7-segment to display to go beyond expected seen values (ex: 0 to 9 was more like 0 to 10 and 9 to 0 was more like 9 to 0-1).
- Solution: carefully analyze circuit behavior using MPLAB X IDE simulator to debug logical problems. Found multiple problems thanks to debugging with simulator, fixed them and circuit behavior now works as intended (0 to 9 UP count now works and 9 to 0 DOWN count now works).