Leonardo Fusser (1946995) November 15th, 2020

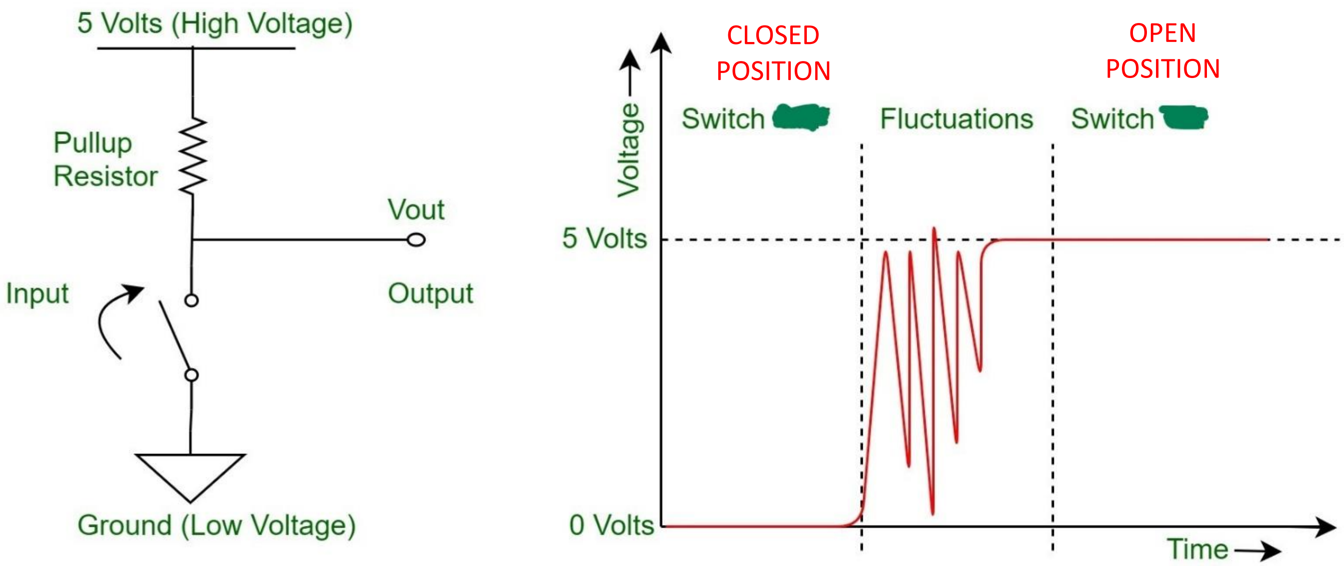
Microcontroller & Microprocessor Systems

Lab 5

Day Yann Fong

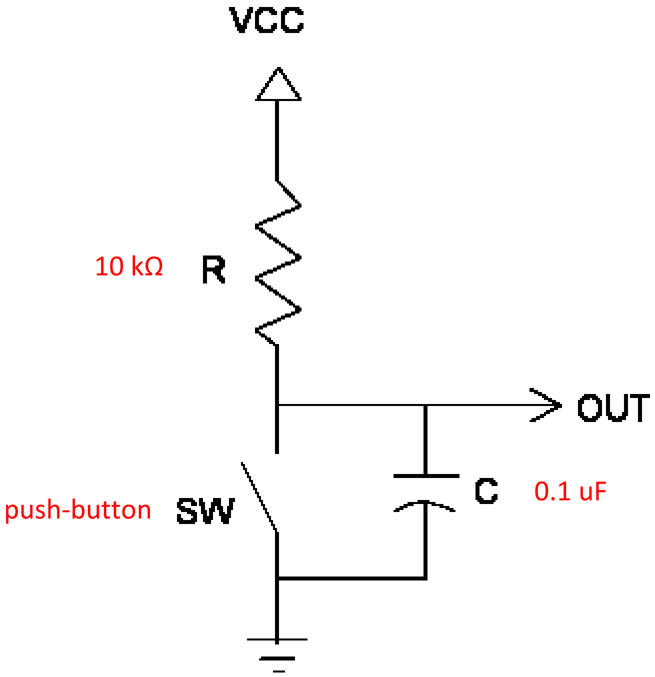
Part A:

3a) The counter doesn’t always operate as expected. This is because my switch does transition smoothly from HIGH to LOW signal (there is quite some noise left behind after switch is pressed). This causes some erratic behaviour from the circuit and counter doesn’t quite work as expected. This behaviour is known as “switch bouncing”.



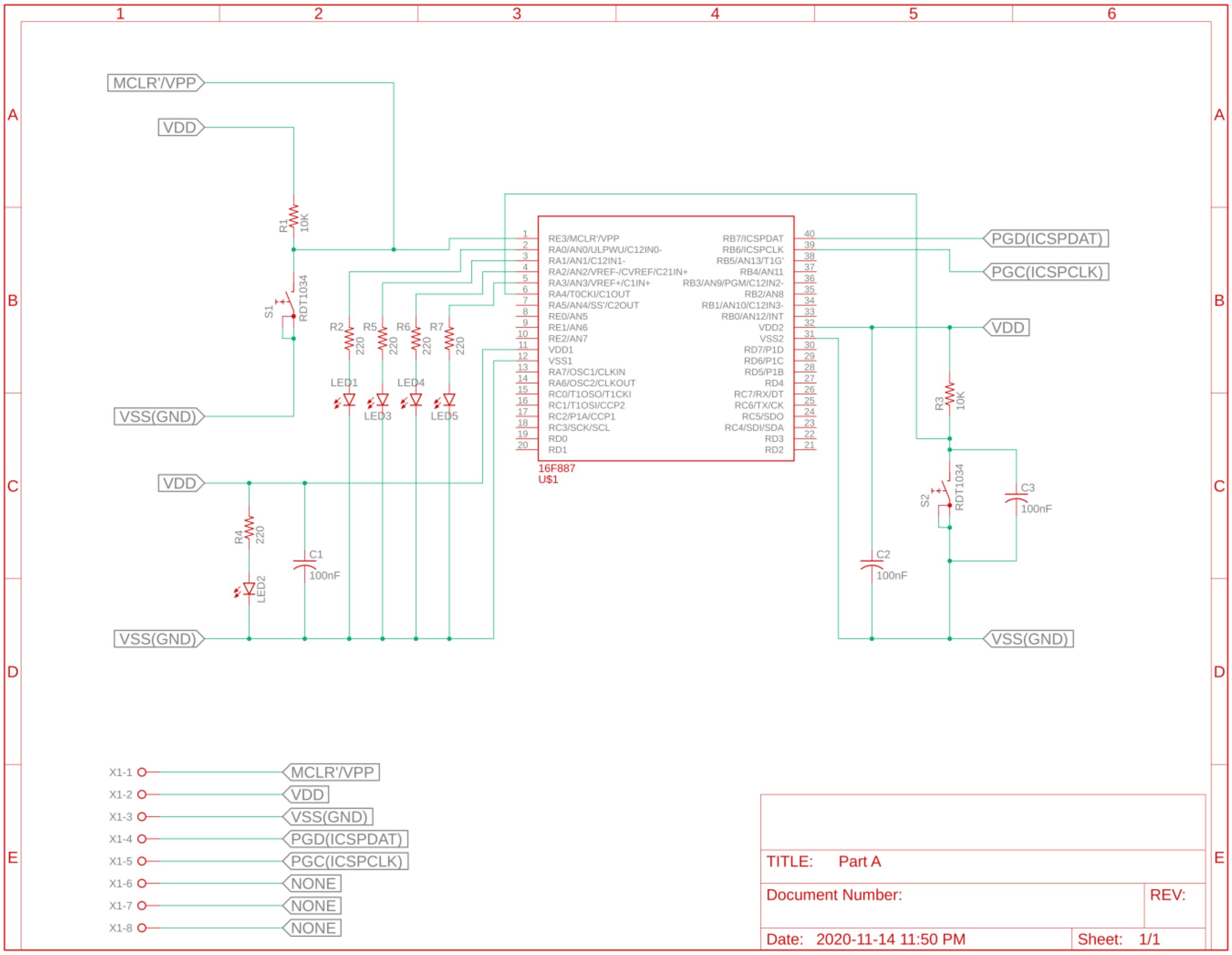
Typical behaviour observed from “switch bouncing” shown above

b) The solution to the problem mentioned above is to implement a "switch debouncing" circuit. Basically, the only change that has to be made is the addition of a 0.1 uF capacitor across the push-button switch to filter out the switch bouncing that might be created during circuit operation.

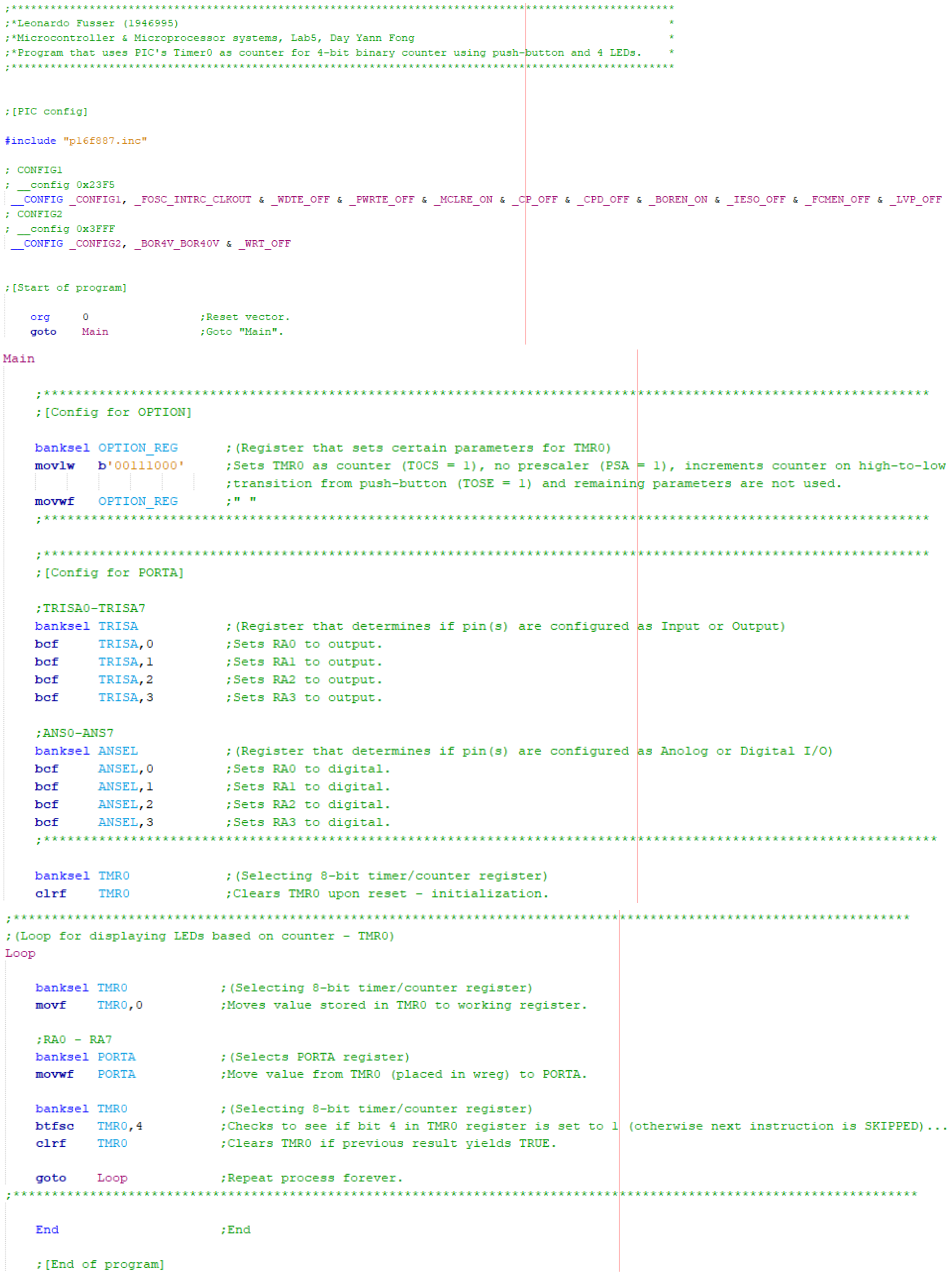


"Switch debouncing" circuit used to solve problem mentioned before

Original Design:

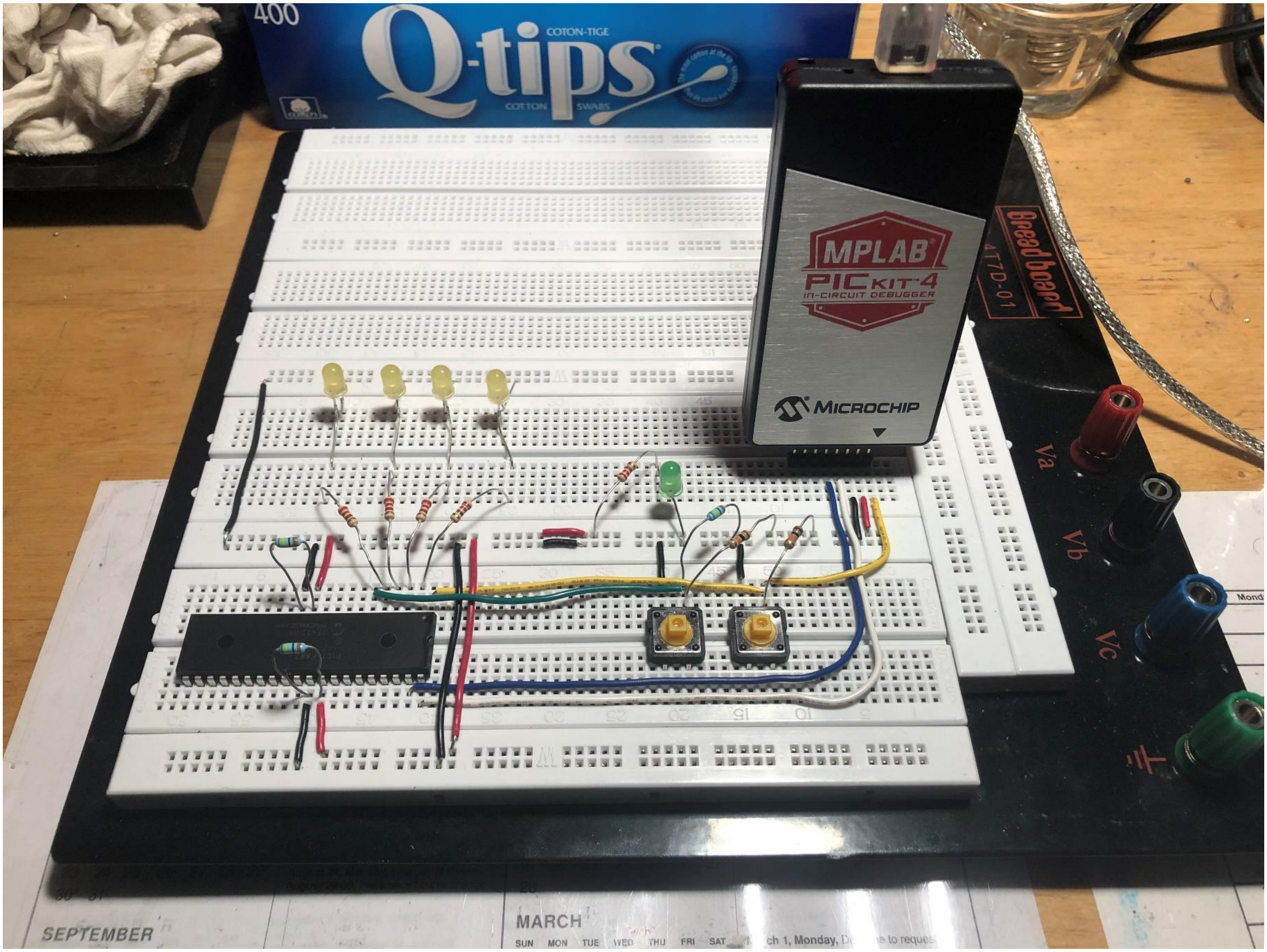


Note: *complete schematic “Lab5_LeonardoFusser_Schematic – PartA.pdf” is included with this lab submission.*

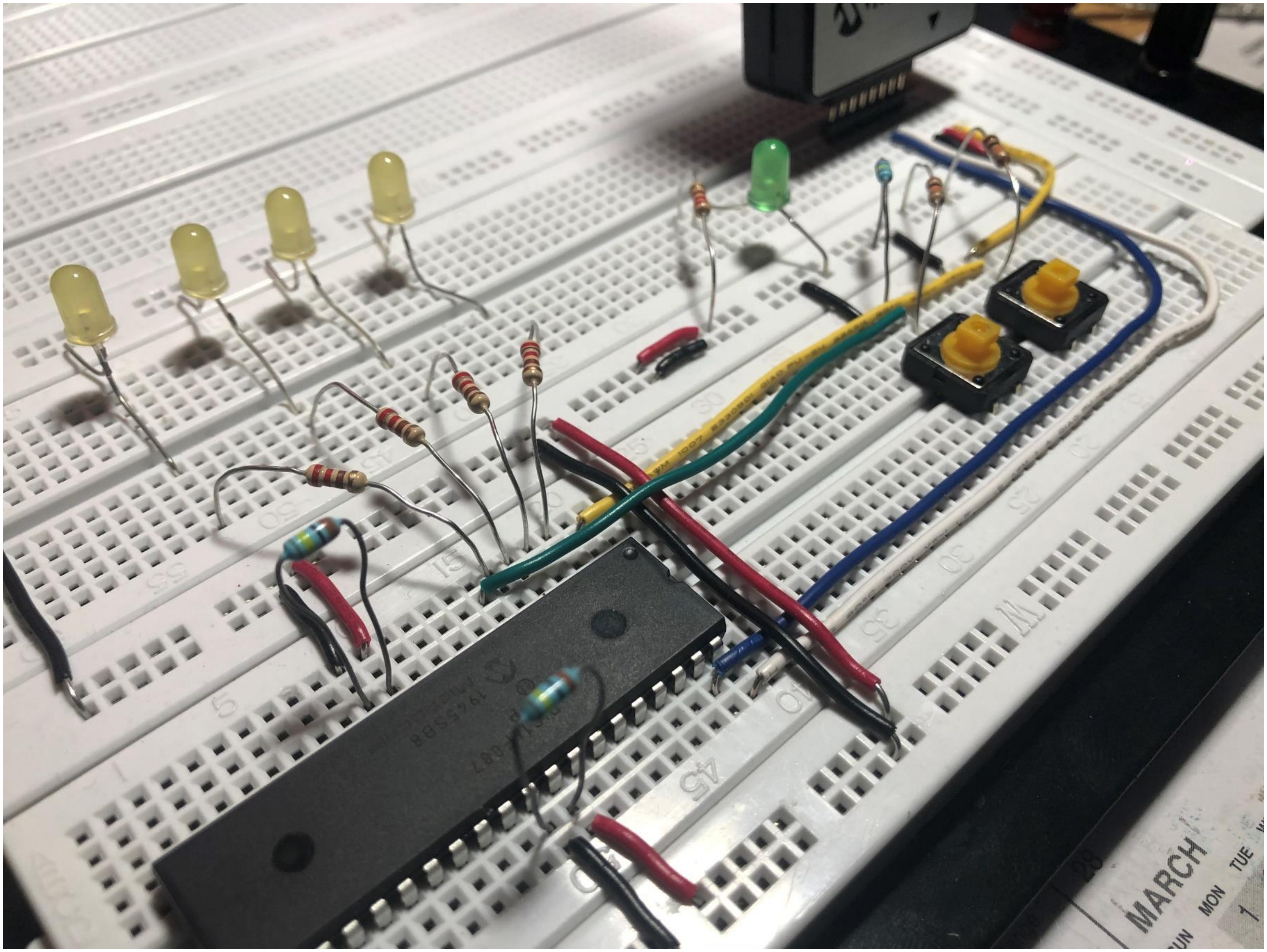


```
;*****************************************************************************************
;*Leonardo Fusser (1946995)                                                              *
;*Microcontroller & Microprocessor systems, Lab5, Day Yann Fong                          *
;*Program that uses PIC's Timer0 as counter for 4-bit binary counter using push-button and 4 LEDs.    *
;*****************************************************************************************


;[PIC config]

#include "p16f887.inc"

; CONFIG1
; __config 0x23F5
 __CONFIG _CONFIG1, _FOSC_INTRC_CLKOUT & _WDTE_OFF & _PWRTE_OFF & _MCLRE_ON & _CP_OFF & _CPD_OFF & _BOREN_ON & _IESO_OFF & _FCMEN_OFF & _LVP_OFF
; CONFIG2
; __config 0x3FFF
 __CONFIG _CONFIG2, _BOR4V_BOR40V & _WRT_OFF


;[Start of program]

    org     0                   ;Reset vector.
    goto    Main                ;Goto "Main".
Main

    ;*********************************************************************************************************
    ;[Config for OPTION]

    banksel OPTION_REG      ;(Register that sets certain parameters for TMR0)
    movlw   b'00111000'     ;Sets TMR0 as counter (T0CS = 1), no prescaler (PSA = 1), increments counter on high-to-low
                            ;transition from push-button (T0SE = 1) and remaining parameters are not used.
    movwf   OPTION_REG      ;" "
    ;*********************************************************************************************************

    ;*********************************************************************************************************
    ;[Config for PORTA]

    ;TRISA0-TRISA7
    banksel TRISA           ;(Register that determines if pin(s) are configured as Input or Output)
    bcf     TRISA,0         ;Sets RA0 to output.
    bcf     TRISA,1         ;Sets RA1 to output.
    bcf     TRISA,2         ;Sets RA2 to output.
    bcf     TRISA,3         ;Sets RA3 to output.

    ;ANS0-ANS7
    banksel ANSEL           ;(Register that determines if pin(s) are configured as Anolog or Digital I/O)
    bcf     ANSEL,0         ;Sets RA0 to digital.
    bcf     ANSEL,1         ;Sets RA1 to digital.
    bcf     ANSEL,2         ;Sets RA2 to digital.
    bcf     ANSEL,3         ;Sets RA3 to digital.
    ;**********************************************************************************************************

    banksel TMR0            ;(Selecting 8-bit timer/counter register)
    clrf    TMR0            ;Clears TMR0 upon reset - initialization.
;************************************************************************************************************
;(Loop for displaying LEDs based on counter - TMR0)
Loop

    banksel TMR0            ;(Selecting 8-bit timer/counter register)
    movf    TMR0,0          ;Moves value stored in TMR0 to working register.

    ;RA0 - RA7
    banksel PORTA           ;(Selects PORTA register)
    movwf   PORTA           ;Move value from TMR0 (placed in wreg) to PORTA.

    banksel TMR0            ;(Selecting 8-bit timer/counter register)
    btfsc   TMR0,4          ;Checks to see if bit 4 in TMR0 register is set to 1 (otherwise next instruction is SKIPPED)...
    clrf    TMR0            ;Clears TMR0 if previous result yields TRUE.

    goto    Loop            ;Repeat process forever.
;************************************************************************************************************

    End                     ;End

    ;[End of program]
```

<u>Note</u>: *.asm source code “Lab5_LeonardoFusser.asm” is included with this submission*



Final circuit prototype shown above. LEDs are connected to pins 2 – 5 on PIC 16F887 (RA0 – RA3). Push-button under green power-up LED is connected to pin 6 on PIC 16F887 (RA4 T0CKI). Push-button next to it is for reset circuit.



Final circuit prototype shown above. LEDs are connected to pins 2 – 5 on PIC 16F887 (RA0 – RA3). Push-button under green power-up LED is connected to pin 6 on PIC 16F887 (RA4 T0CKI). Push-button next to it is for reset circuit.

Observation:

- Circuit behaviour was not working initially. Problem: output of LEDs was not being produced correctly and push-button was not triggering counter. Reason: TRISA and ANSEL configuration was conflicting with RA4 (T0CKI) pin. Solution: isolate configuration of setting only first 4 bits of TRISA as output and ANSEL as digital.

- After solving problem mentioned above, a new problem occurred. Problem: output of LEDs was not consistent with every push of push-button switch at times. Reason: switch was producing "bouncing" effect at random times causing the counter to increment or jump more then it was expected to do. Solution: implement "switch debounce" circuit; add 100 nF capacitor across push-button connected to RA4 to filter any bouncing the push-button might create during operation. Circuit worked as intended afterwards without any problems.

Discussion & Conclusion:

- Learned how to configure Timer0 module in PIC 16F887.
- Learned how to configure Timer0 module as counter in PIC16F887.
- Learned how to create a 4-bit binary counter (using Timer0 module as counter) to count on every falling-edge of push-button and display result to LEDs.