วัตถุประสงค์ของการทดลองปฏิบัติการ

1. แสดงปัญหาที่เกิดขึ้นเมื่อทำการเชื่อมต่อระหว่างสวิทช์หรือปุ่มกดและไมโครคอนโทรลเลอร์
2. แสดงการแก้ไขปัญหาการกระเด้งที่หน้าสัมผัสของสวิทช์โดยใช้กระบวนการทางซอฟแวร์
3. ปฏิบัติการทดลองใช้อัลกอริธึ่มในการแก้ปัญหาการกระเด้งของหน้าสัมผัสโดยใช้ภาษาแอสเซมบลี้บน PIC16F628A

---

1 ทฤษฎีที่เกี่ยวข้อง

การกระเด้งของหน้าสัมผัสของสวิทช์เป็นปัญหาที่มีจุดกำเนิดจากคุณลักษณะทางกลศาสตร์ของกลไกในสวิทช์แบบปุ่มกด พิจารณารูปที่ 1 ซึ่งเป็นวงจรอย่างง่ายของการเชื่อมต่อสวิทช์แบบปุ่มกด (Push button switch) โดยใช้ตัวทาน Pull-up resistor โดยที่สถานะทางตรรกะของ Vout จะมีค่าเป็น 1 หรือ 5 V เมื่อสวิทช์ไม่โดนกด (เปิดวงจร) และมีค่าเป็น 0 หรือ 0 V เมื่อสวิทช์โดนกด (ปิดวงจร) เมื่อใช้เครื่องวัดเช่น ออสซิลโลสโคปวัดค่าแรงดัน Vout ปัญหาการกระเด้งของสวิทช์จะปรากฏขึ้นอย่างชัดเจนดังแสดงในรูปที่ 2 จะเห็นได้ว่าเมื่อสวิทช์โดนกด แรงดัน Vout จะไม่เปลี่ยนสถานะแรงดันจาก 5 V เป็น 0 V แบบครั้งเดียวจบ แต่จะมีลักษณะกระโดดกลับไปมาระหว่าง 0-5V แทน ปรากฏการณ์นี้มีสาเหตุจากสปริงของสวิทช์และความไม่แน่นอนของแรงกดจากนิ้วของมนุษย์ (ข้อสังเกต ปรากฏการณ์กระเด้งนี้จะเกิดขึ้นทั้งเมื่อสวิทช์โดนกดและโดนปล่อย) เมื่อสัญญาณ Vout ที่มีการกระโดดไปมานี้ถูกเชื่อมต่อเข้ากับระบบดิจิทัลเช่น ตัวนับ (digital counter) หรือไมโครคอนโทรลเลอร์ การกดปุ่มสวิทช์เพียงครั้งเดียวจะถูกตีความโดยระบบดิจิทัลว่าเป็นการกดปุ่มจำนวนหลายครั้ง ซึ่งจะทำให้เกิดความผิดพลาดของการทำงานของระบบดิจิทัลขึ้น

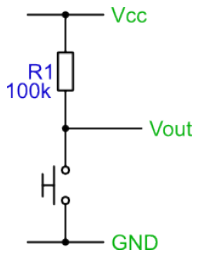

Figure 1: วงจรเชื่อมต่อสวิทช์โดยใช้ตัวต้านทาน Pull-up resistor

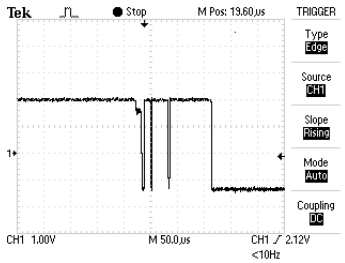

Figure 2: สัญญาณ Vout ที่วัดโดยใช้ออสซิลโลสโคป

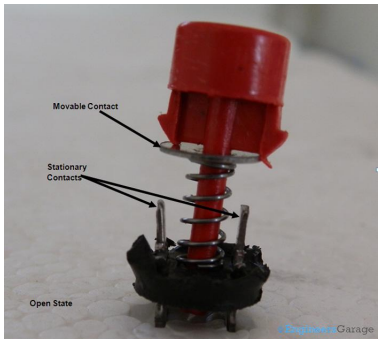

Figure 3: สวิทช์แบบปุ่มกด (สถานะเปิดวงจร)

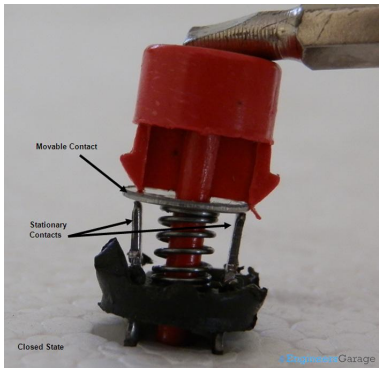

Figure 4: สวิทช์แบบปุ่มกด (สถานะปิดวงจร)

การแก้ปัญหานี้สามารถทำได้หลายวิธีเช่น การใช้วงจรดังแสดงในรูปที่ 5 ไดโอด, ตัวเก็บประจุและตัวต้านทานในวงจรนี้จะประกอบตัวเป็นวงจรกรองสัญญาณอย่างง่ายเพื่อกรองการกระโดดไปมาของ Vout ออกไปทำให้การเปลี่ยนแปลงระดับตรรกะจาก 5 V ไป 0 V เรียบขึ้น การแก้ปัญหานี้ยังสามารถทำได้โดยใช้กรรมวิธีทางซอฟแวร์ (Software debounce)

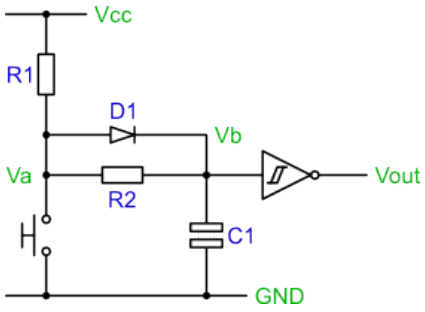

Figure 5: การแก้ปัญหาการกระเด้งของหน้าสัมผัสของสวิทช์โดยใช้วงจรอิเล็กทรอนิคส์

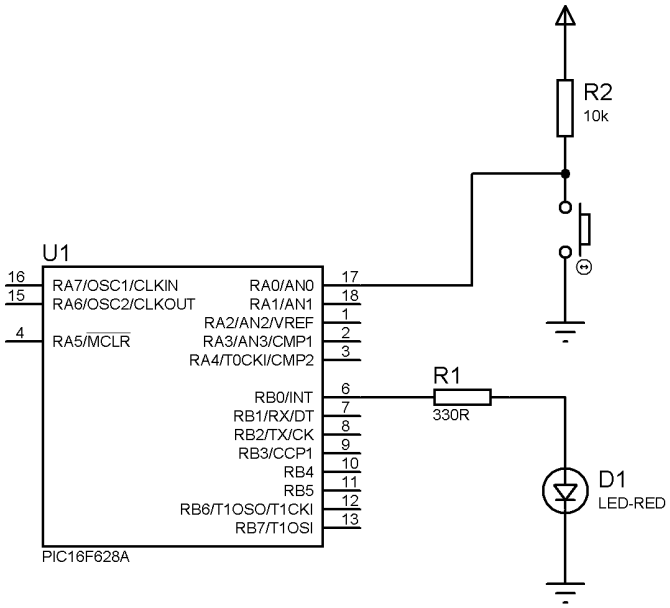

Figure 6: วงจรแสดงการเชื่อมต่อสวิทช์และ LED ผ่านไมโครคอนโทรลเลอร์ PIC16F628A

1.1 Software Debounce

การแก้ปัญหาการกระเด้งของสวิทช์ด้วยซอฟแวร์สามารถทำได้อย่างง่ายๆ โดยใช้การหน่วงเวลา Time delay เพื่อหลบเลี่ยงช่วงเวลาของการกระเด้ง โดยปรกติแล้ว ช่วงเวลาของการกระเด้งจะกินเวลาประมาณ 10 มิลลิวินาที เราสามารถอธิบายการทำงานของโปรแกรม Switch debouncing ได้โดยใช้วงจรของการเชื่อมโยงไมโครคอนโทรลเลอร์ PIC16F628A กับสวิทช์และ LED ดังแสดงในรูปที่ 6 และอัลกอริธึ่มของการแก้ปัญหาการกระเด้งของสวิทช์ดังแสดงด้วยโฟลวชาตในรูปที่ 7 จากรูปทั้งสองสามารถสรุปได้ว่า เราสามารถหลีกเลี่ยงช่วงการกระเด้งของสวิทช์ได้โดย ตรวจสอบว่ามีการกดสวิทช์หรือไม่ ถ้ามีการกดสวิทช์ ก็จะใช้การหน่วงเวลาประมาณ 10 มิลลิวินาที เพื่อหลีกเลี่ยงช่วงความไม่แน่นอนของสถานะของหน้าสัมผัสของสวิทช์ แล้วจากนั้นจึงตรวจสอบการกดสวิทช์อีกครั้งหนึ่ง ถ้าการตรวจพบการกดสวิทช์อีกครั้งแสดงว่าเป็นการกดสวิทช์ที่ถูกต้อง การกดสวิทช์นี้ก็จะถูกยอมรับและไมโครคอนโทรลเลอร์ก็จะปฏิบัติตามภาระงานเช่น ทำให้ LED สว่างในกรณีตัวอย่างนี้ แต่ถ้าไม่สามารถตรวจพบการกดสวิทช์หลังจากการหน่วงเวลา 10 มิลลิวินาที แสดงว่าหน้าสัมผัสของสวิทช์ยังคงกระเด้งอยู่ การกดสวิทช์นี้จะถูกละเลยดังแสดงในรูปที่ 7

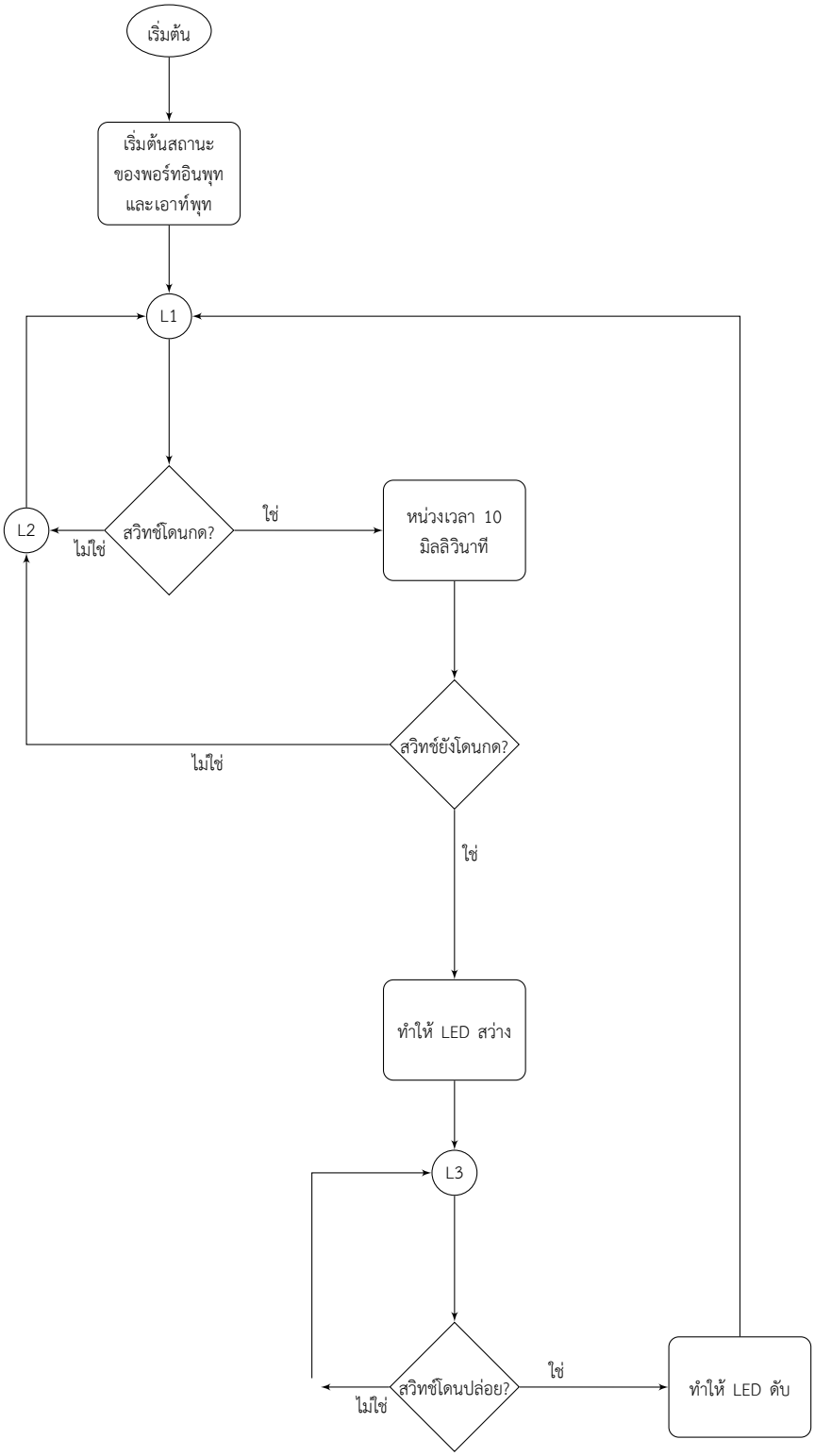

Figure 7: Flow-chart แสดงการแก้ปัญหาการกระเด้งของสวิทช์

โปรแกรมแอสเซมบลี้ต่อไปนี้เป็นโปรแกรมที่ถูกเขียนขึ้นตามโฟลวชาตในรูปที่ 7

```
;****************************************
;**** Program title:  Switch deboune  ******
;**** the switch is connected to port RA0 **
;**** the LED is connected to port RB0  ****
;**** Programmer: Mr. Chatchai   ***********
;****************************************

                PROCESSOR PIC16F628
                #include <P16F628.INC>
                __CONFIG _CP_OFF & _MCLRE_OFF & _INTRC_OSC_NOCLKOUT & _LVP_OFF & _WDT_OFF

;****** Define general purpose registers for temporary variables
                cblock  0x20
                        temp
                        count
                        count0
                        count1
                        count2
                endc
;*********************************************************
                ORG     0x00            ; Reset Vector
                goto    Main            ; vector to main program

Main:                                   ; Main program begins here

                call    Init            ; Calling initialization subroutine

L1:                                     ; Begining for the infinite loop

L2:
                btfsc   PORTA,0         ; Check if the switch is pressed?
                goto    L2              ; No the switch is not pressed

                movlw   .10             ; Yes the switch is pressed then delay for 10 mS
                call    DelayMS         ; by calling the time delay subroutine

                btfsc   PORTA,0         ; Check again if the switch is still pressed?
                goto    L2              ; No then go back to check the switch again

                bsf     PORTB,0         ; Yes the switch is sitll pressed
                                        ; then turn-on the LED
L3:
                btfss   PORTA,0         ; Check if the switch is released?
                goto    L3              ; No the switch is still pressed

                bcf     PORTB,0         ; Yes the switch is released
                                        ; then turn-off the LED

                goto    L1              ; Go back and repeat this loop again

;******************** Subroutines ****************************

;=====================================================
;* Initialization subroutine
;=====================================================
Init:
                movlw   .7
                banksel CMCON
                movwf   CMCON           ; Disable analog comparator
```

```
60                  banksel TRISB
61                  movlw   0x00
62                  movwf   TRISB           ; Set PORTB as an output port
63                  movlw   0xFF
64                  movwf   TRISA           ; Set PORTA as an input port
65                  banksel PORTB           ; Back to Bank0
66                  clrf    PORTB           ; Turn-off LEDs connected to PORTB
67                  return
68
69  ;*****************************************************
70  ;* Delay 2 Routine - Decrement delay loop in Milisecond
71  ;* 1 instruction cycle is 1 micro-second
72  ;* at 4 Mhz X'tal frequency, 1MS = 1000 uS = 100x10
73  ;* where 100 iterations for inner loops, 10 iterations for
74  ;* outter loops
75  ;*****************************************************
76  DelayMS:
77                  movwf   count2
78                  incf    count2,f
79                  decfsz  count2,f
80                  goto    $+2
81                  goto    $+3
82                  call    Delay1MS
83                  goto    $-4
84                  return
85
86  Delay1MS:
87                  movlw   .50             ; 1 cyc
88                  movwf   count1          ; 1 cyc
89  outterloop:
90                  movlw   .5              ; 1 cyc * count1
91                  nop                     ; 1 cyc * count1
92                  movwf   count0          ; 1 cyc * count1
93  innerloop:
94                  decfsz  count0,F        ; 1 cyc * count1 * count0
95                  goto    innerloop       ; 2 cyc * count1 * count0
96                  decfsz  count1,F        ; 1 cyc * count1
97                  goto    outterloop      ; 2 cyc * count1
98                  return                  ; 1 cyc
99                  ; total = 3 + (6+3.count0).count1
100                 ; count0 = 5 , count1 = 50, total = 1053 cyc ??
101
102                 END
```

2 การทดลองปฏิบัติการ

เขียนโปรแกรมภาษาแอสเซมบลี้เพื่อควบคุมวงจรในรูปที่ 9 โดยกำหนดให้ เมื่อกดสวิทช์ SW1 จะทำให้ LED1 สว่างและเมื่อกดสวิทช์ SW2 จะทำให้ LED2 สว่าง โดยนักศึกษาจะต้องประยุกต์ใช้การแก้ปัญหาการกระเด้งของสวิทช์ดังที่ได้แสดงมาแล้วก่อนหน้านี้กับการควบคุมสวิทช์ทั้งสองนี้ เพื่อให้ได้คะแนนเต็มนักศึกษาจะต้อง

1. เขียนโปรแกรมแอสเซมบลี้แบบสมบูรณ์ในพื้นที่ว่างต่อไปนี้ (เขียนโฟลวชาตประกอบถ้าเป็นไปได้)
2. ทำการจำลองการทำงานของโปรแกรมบน Proteus ISIS
3. ทำการดาวน์โหลดโปรแกรมที่ได้ลงสู่บอร์ด PCK-1000 ต่อวงจรจาก PortA (RA0, RA1) ไปที่สวิทช์จำนวนสองตัว (สวิทช์อยู่บนบอร์ด PCK-1000) ต่อวงจรจาก PortB (RB0, RB1) เข้ากับ LED 2 ดวงและสาธิตการทำงานของวงจรจริงให้กับอาจารย์ผู้สอน

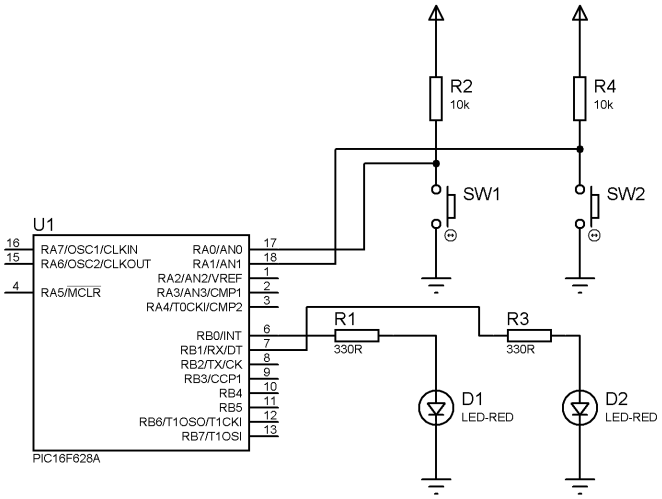

Figure 8: วงจรเชื่อมโยงไมโครคอนโทรลเลอร์ PIC16F628A กับสวิทช์จำนวนสองตัวและ LED สองตัว

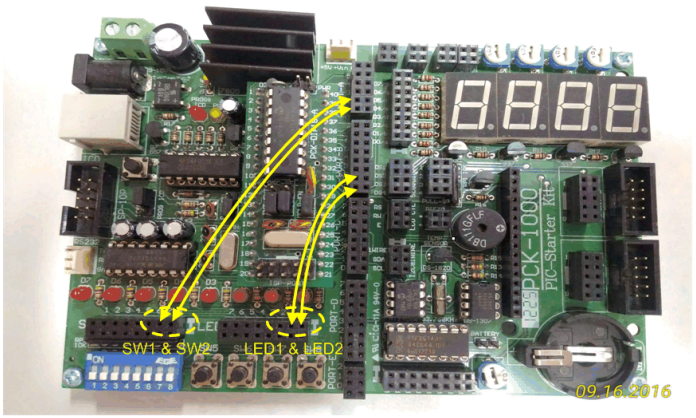

Figure 9: การต่อวงจรบนบอร์ด PCK-1000

(เขียนโปรแกรมในพื้นที่ว่างต่อไปนี้)

Laboratory #4

Switch debouncing

Name _____ ID _____

8 / 8

(เขียนโปรแกรมในพื้นที่ว่างต่อไปนี้)

Score _____ Date __/__/____