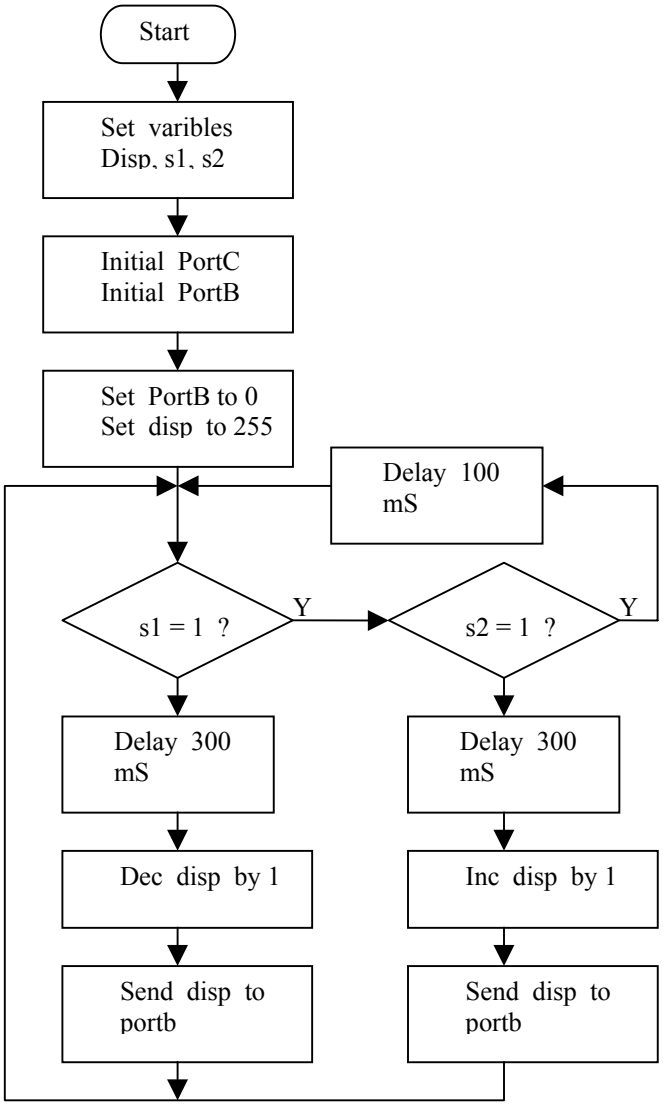
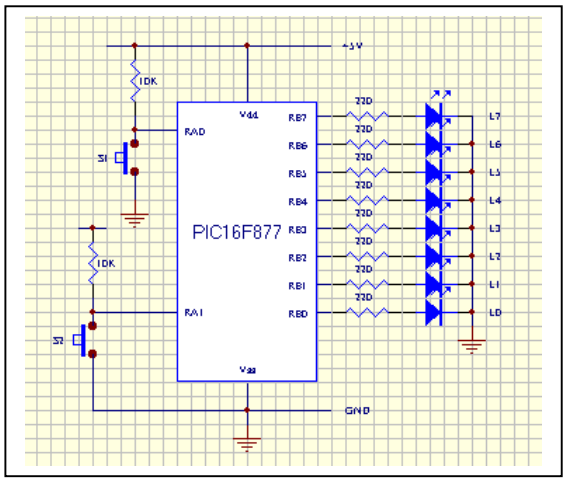
Assignment 5 Increment and Decrement Value 5

จุดประสงค์ เพื่อเรียนรู้หลักการเขียนโปรแกรม เพิ่มค่าและลดค่าตัวแปร(Variable) ที่อยู่ภายในตัวชิพไมโครคอนโทรลเลอร์ โดยการใช้สวิตช์ S1 และ S2 กดควบคุมจากภายนอก เพื่อนำไปประยุกต์ใช้งานต่าง ๆ เช่น ทำปุ่มปรับเพิ่ม – ลดอุณหภูมิ เป็นต้น **ในใบงานนี้ เมื่อ กด S1 หรือ S2 ค้างไว้ ค่าจะเพิ่มและลดโดยต่อเนื่องปล่อยมือจะหยุด**

```
disp        VAR         BYTE
s1          VAR         PORTA.0
s2          VAR         PORTA.1
            ADCON1 =  7
            TRISA    =  %11111111
            TRISB    =  %00000000
            PORTB    =  0
            disp     =  255

start:      IF s1 = 1 Then
                GoTo chk_s2
            Else
                Pause 300
                disp  = disp - 1
                PORTB = disp
            EndIF
chk_s2:     IF s2 = 1 Then
                Pause 100
                GoTo start
            Else
                Pause 300
                disp  = disp + 1
                PORTB = disp
            EndIF
            GoTo start
            End
```

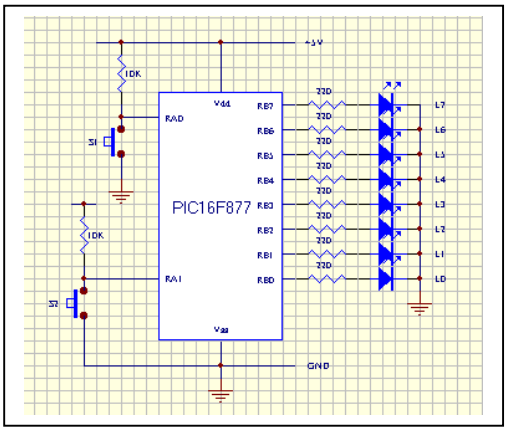




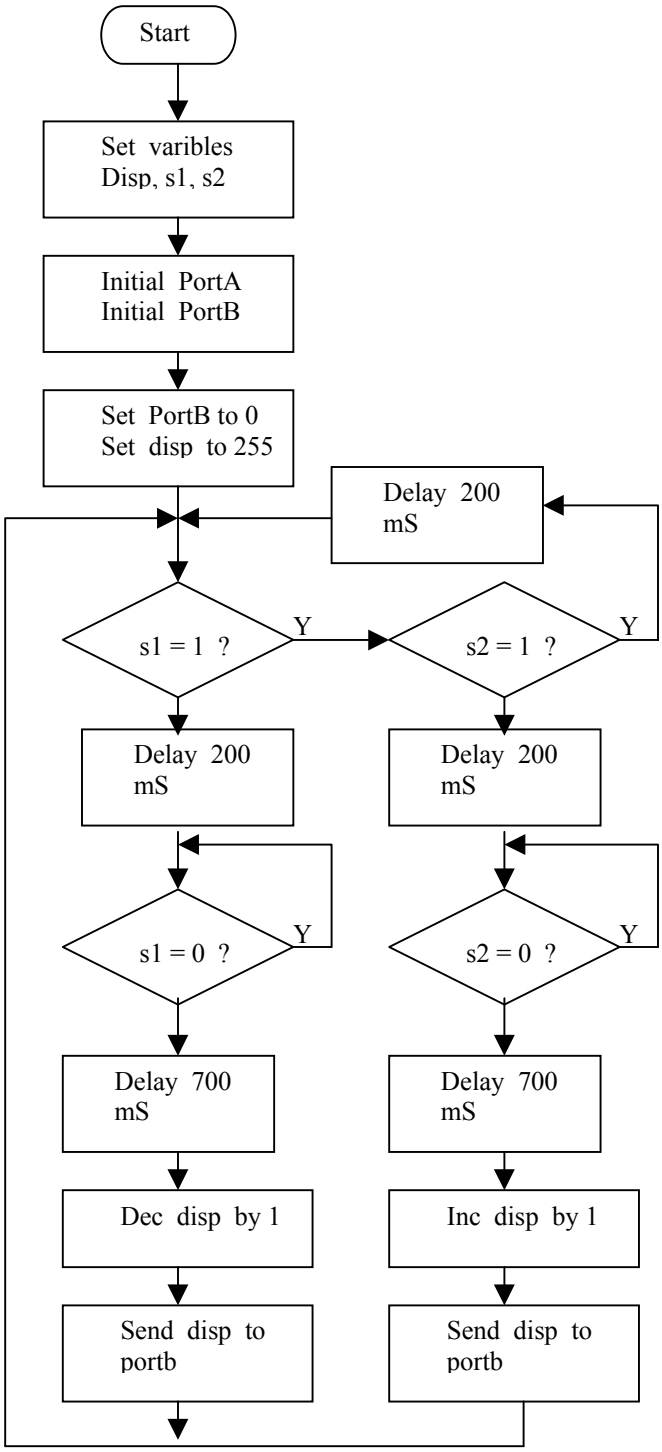
Assignment 6 Increment and Decrement Value 6

(เมื่อกด S1 หรือ S2 จะลดค่าและเพิ่มค่าทีละหนึ่งค่าไม่ต่อเนื่อง)

จุดประสงค์ เพื่อเรียนรู้หลักการเขียนโปรแกรม เพิ่มค่าและลดค่าตัวแปร(Variable) ที่อยู่ภายในตัวชิพไมโครคอนโทรลเลอร์ โดยการใช้สวิตช์ S1 และ S2 กดควบคุมจากภายนอก เพื่อนำไปประยุกต์ใช้งานต่าง ๆ เช่น ทำปุ่มปรับเพิ่ม – ลดอุณหภูมิ เป็นต้น **แต่ใบงานนี้จะเพิ่ม-ลดไม่ต่อเนื่อง จะต้องกดทีละครั้ง**




```
disp        VAR         BYTE
s1          VAR         PORTA.0
s2          VAR         PORTA.1
            ADCON1 = 7
            TRISA = %11111111
            TRISB = %00000000
            PORTB = 0
            disp  = 255

start:      IF s1 = 1 Then
                GoTo chk_s2
            Else
                Pause 200
                IF s1 = 1 Then
                    Pause 200
                    disp  = disp - 1
                    PORTB = disp
                EndIF
            EndIF
chk_s2:     IF s2 = 1 Then
                Pause 100
                GoTo start
            Else
                Pause 200
                IF s2 = 1 Then
                    Pause 200
                    disp  = disp + 1
                    PORTB = disp
                EndIF
            EndIF
            GoTo start
            End
```

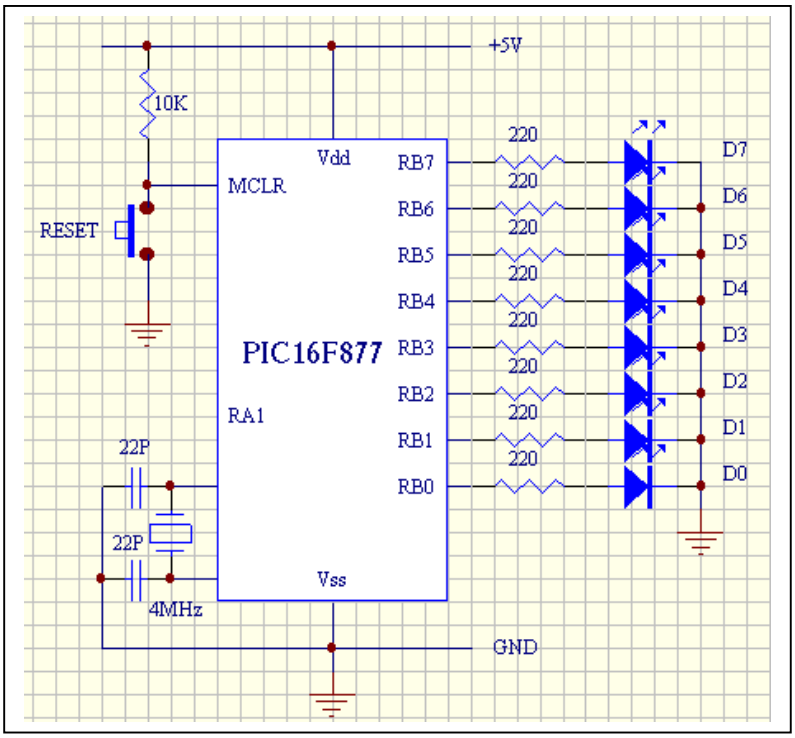

Start
Set varibles
Disp, s1, s2
Initial PortA
Initial PortB
Set PortB to 0
Set disp to 255
Delay 200
mS
s1 = 1 ?
Y
s2 = 1 ?
Y
Delay 200
mS
Delay 200
mS
s1 = 0 ?
Y
s2 = 0 ?
Y
Delay 700
mS
Delay 700
mS
Dec disp by 1
Inc disp by 1
Send disp to
portb
Send disp to
portb

Assignment 7 Shifting the content in register 8

จุดประสงค์ เพื่อเรียนรู้หลักการเขียนโปรแกรม เลื่อน หรือขยับตำแหน่งบิทที่เป็นค่าของตัวแปร ที่อยู่ในรีจิสเตอร์ (Register)ของตัวไมโครคอนโทรลเลอร์ เพื่อนำไปประยุกต์เป็นพื้นฐานในการทำไฟวิ่ง หรืองานประมวลผลอื่น ๆ เป็นต้น

การทำงานของโปรแกรม ในโปรแกรมนี้ค่าที่อยู่ในตัวแปร คือ 10000000 เมื่อทำให้ข้อมูลขยับไปทีละบิท จะทำให้เลข 1 วิ่งขยับจากซ้าย ไปขวา เมื่อสุดแล้วจะเริ่มใหม่อย่างต่อเนื่อง

```
'8 LED running light
delay       CON         100
light       VAR         BYTE
reload      CON         %10000000
            TRISB = %00000000
            light = reload
main:  Pause delay
            PORTB = light
            light = light >> 1
            IF light = %00000000 Then
                  light = reload
            EndIF
            GoTo  main
            End
```

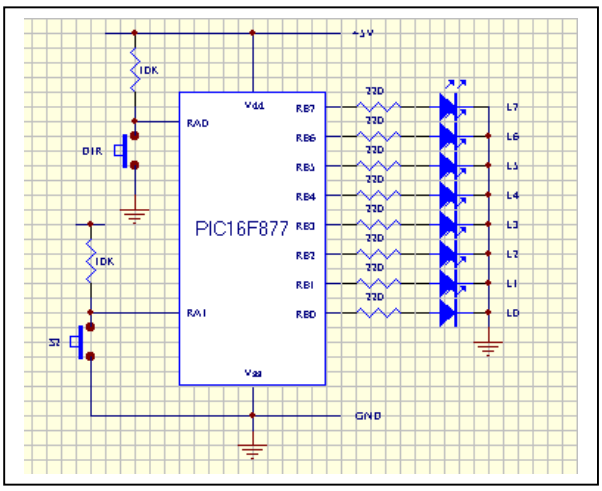


Assignment 8 Shifting the content in register 9

จุดประสงค์ เพื่อเรียนรู้หลักการเขียนโปรแกรม เลื่อนหรือขยับตำแหน่งบิทที่เป็นค่าของตัวแปร ที่อยู่ในรีจิสเตอร์ (Register) ให้ไปทางซ้าย หรือขวา ได้โดยการควบคุมจากสวิตช์จากภายนอกของตัวไมโครคอนโทรลเลอร์ เพื่อเป็นพื้นฐานนำไปประยุกต์ในการทำไฟวิ่งหรืองานประมวลผลอื่น ๆ แบบควบคุมได้ เป็นต้น

การทำงานของโปรแกรม ในโปรแกรมนี้ ค่าที่อยู่ในตัวแปร คือ 10000000 และ 00000001 เมื่อทำให้ข้อมูลขยับไปทีละบิท จะทำให้เลข 1 วิ่งขยับจากซ้าย ไปขวา หรือขวา ไปซ้าย เมื่อสุดแล้วจะเริ่มใหม่อย่างต่อเนื่อง ทิศทางการเลื่อนโดยการใช้สวิตช์ ควบคุมจากภายนอก

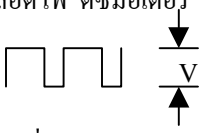
```
delay       CON       300
light       VAR       BYTE
direct      VAR       PORTA.0
reload1     CON       %10000000
reload2     CON       %00000001
            ADCON1  = 7
            TRISB = %00000000
            light = reload1
main:       Pause   delay
            PORTB = light
            Input   direct
            IF direct = 1 Then
                    light = light >> 1
                    IF light = %00000000 Then
                            light = reload1
                    EndIF
            Else
                    light = light << 1
                    IF light = %00000000 Then
                            light = reload2
                    EndIF
            EndIF
            GoTo  main
            End
```



Assignment 9 Sending Pulse Width Modulation (PWM) 10

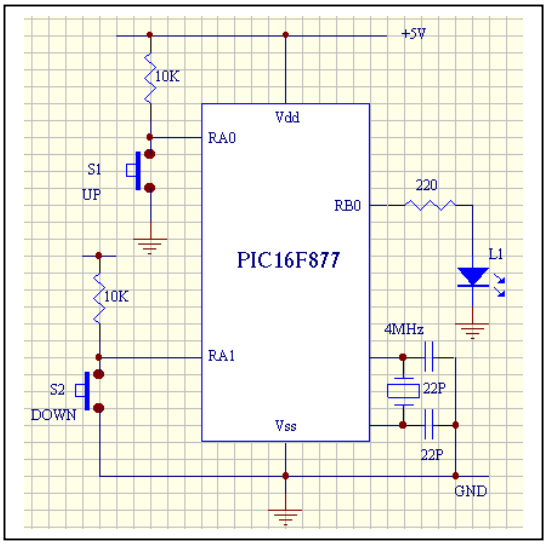
จุดประสงค์ เพื่อเรียนรู้หลักการเขียนโปรแกรม ผลิตสัญญาณ PWM (PWM: Pulse Width Modulation) เพื่อนำไปประยุกต์ใช้ควบคุมการเร่ง – หรี่กำลังอุปกรณ์ไฟดีซีต่าง ๆ ได้แก่หลอดไฟ ดีซีมอเตอร์

การทำงานของโปรแกรม สัญญาณ PWM มีลักษณะเป็น Pulse บวกและลบ สลับกันไป



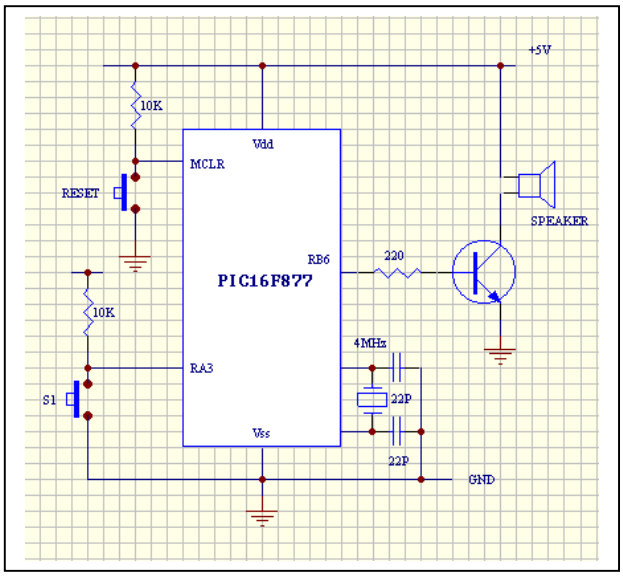
ไซเกิลจะมีทั้ง Pulse บวกและลบ โดยการกำหนดความถี่ของทุก ๆไซเกิลไว้คงที่ หากมีการทำให้ความกว้างของ Pulse บวก เปลี่ยนแปลงให้กว้างหรือแคบภายในแต่ละไซเกิลได้ ก็จะทำให้ค่าเฉลี่ยของแรงเคลื่อนด้านเอ้าท์เปลี่ยนแปลงได้ เท่ากับเราสามารถควบคุมกำลังอุปกรณ์ไฟดีซีต่าง ๆ ได้ ความกว้างของ Pulse บวกนี้เราเรียกว่า ดิวตี้ไซเกิล (Duty Cycle) คิดเป็นเปอร์เซ็นต์ของทั้งไซเกิล

```
LED             VAR     PORTB.0
duty            VAR     BYTE
cycle           CON     50
sw_up           VAR     PORTA.0
sw_dn           VAR     PORTA.1
        adcon1  =   7
        duty    =   127
chk_up:     Input sw_up
        IF sw_up = 1 Then
            GoTo chk_down
        Else
            Pause 10
            duty  =  duty +1
            if  duty >=254  then  duty = 254
            GoTo send_pwm
        EndIF

chk_down: Input sw_dn
        IF sw_dn = 0 Then
            Pause 10
            duty  =  duty –1
            if  duty <= 1  then  duty = 1
            GoTo send_pwm
        EndIF
send_pwm: PWM LED,duty,cycle
        GoTo  chk_up
        End
```



จุดประสงค์ เพื่อเรียนรู้หลักการเขียนโปรแกรม ผลิตสัญญาณ ความถี่ออกทางขา I/Oของไมโครคอนโทรลเลอร์ ด้วยการใช้คำสั่ง FREQOUT เพื่อนำไปประยุกต์ใช้ในการกำเนิดเสียงเตือน หรือเสียงคลิกไป พร้อมกับการการกดสวิตช์

```
'Two-tone  chime
spk          VAR          PORTB.6
duration1    VAR          WORD
duration2    VAR          WORD
freq1        VAR          WORD
freq2        VAR          WORD
switch       VAR          PORTA.3
       ADCON1     =      7
       duration1  =      5
       duration2  =      1250
       freq1      =      1250
       freq2      =      950

main:  IF switch  =  1   Then main
loop:  FreqOut       spk,duration1,freq1
       IF switch = 0   Then    loop
       FreqOut spk,duration2,freq2
       GoTo  main
       End
```

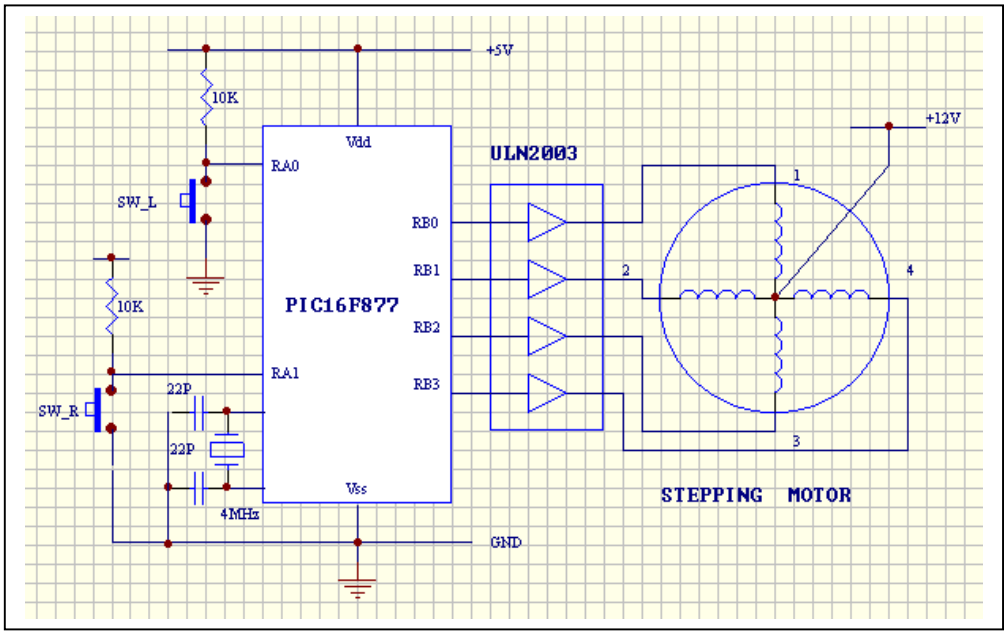


Assignment 11 **Stepping Motor Drives Control 1** **12**

จุดประสงค์ เพื่อเรียนรู้หลักการเขียนโปรแกรม ผลิตสัญญาณ การขับ Stepping Motor แบบ 1 เฟส เพื่อนำไปประยุกต์ใช้เป็นพื้นฐานการควบคุมตำแหน่งของการเคลื่อนที่กลไก

การทำงานของโปรแกรม ใช้หลักการเช่นเดียวกับโปรแกรมควบคุมการเลื่อนขยับบิทข้อมูลเหมือนไฟวิ่งทีละ 1 หลอด

```
'1- Phase Stepping  Motor  Controller
'By  Somboon  Niamglam
'17 Jan  2001
'
'---------------------------------
SW_L        VAR                 PORTA.0
SW_R        VAR                 PORTA.1
delay       CON                 200
TRISB       =       %00000000
TRISA       =       %11111111
PORTB       =       %00010000
ADCON1      =       7

main:  IF SW_L = 0 Then ccw
       IF SW_R = 0 Then cw
       GoTo main

cw:    Pause delay
       IF PORTB  = %10000000 Then reload_cw
       PORTB  = PORTB << 1
       GoTo main

reload_cw:  PORTB = %00010000
            GoTo main

ccw:        Pause delay
            IF PORTB = %00010000 Then reload_ccw
            PORTB = PORTB >> 1
            GoTo main

reload_ccw: PORTB = %B10000000
            GoTo main
            End
```



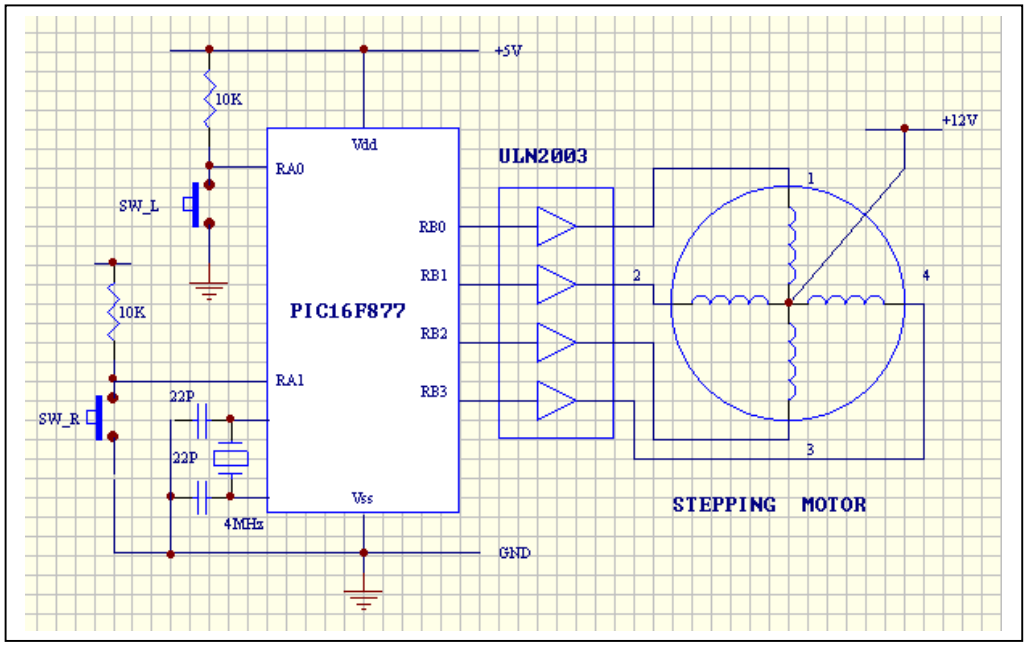
วงจรควบุมการหมุนของ Stepping Motor ตาม Assignment 11 Switch SW_L และ SW_R สำหรับกดควบคุมให้หมุนซ้าย และหมุนขวา ทีละสเต็ป หากกดค้างจะหมุนต่อเนื่อง

Assignment 12 Stepping Motor Drives Control 2 14

จุดประสงค์ เพื่อเรียนรู้หลักการเขียนโปรแกรม ผลิตสัญญาณ การขับ Stepping Motor แบบ 2 เฟส เพื่อนำไปประยุกต์ใช้เป็นพื้นฐานการควบคุมตำแหน่งของการเคลื่อนที่กลไก

การทำงานของโปรแกรม ใช้หลักการเช่นเดียวกับโปรแกรมควบคุมการเลื่อนขยับบิทข้อมูลเหมือนไฟวิ่ง. ทีละ 2 หลอดนั่นหมายถึงการขับทีละ 2 เฟส ทำให้มีกำลังบิดมากขึ้นด้วย

```
'2- Phase Stepping  Motor  Controller
'By  Somboon  Niamglam
'17 Jan  2001
'
'---------------------------------
SW_L        VAR             PORTA.0
SW_R        VAR             PORTA.1
stepp       VAR             BYTE
delay       CON             300

            TRISB = %00000000
            TRISA = %11111111
            ADCON1  =  7
            stepp = 0
            GoSub drive
main:       IF SW_L = 0 Then ccw
            IF SW_R = 0 Then cw
            GoTo main

cw:         stepp = (stepp+1)//4
            GoSub drive
            GoTo main

ccw:        stepp = (stepp-1)//4
            GoSub drive
            GoTo main

drive:      LookUp stepp,[%00110000,%01100000,_
                    %11000000,%10010000],PORTB
            Pause delay
            Return
            End
```



วงจรควบุมการหมุนของ Stepping Motor ตาม Assignment 12 Switch SW_L และ SW_R สำหรับกดควบคุมให้หมุนซ้าย และหมุนขวาอย่างต่อเนื่อง

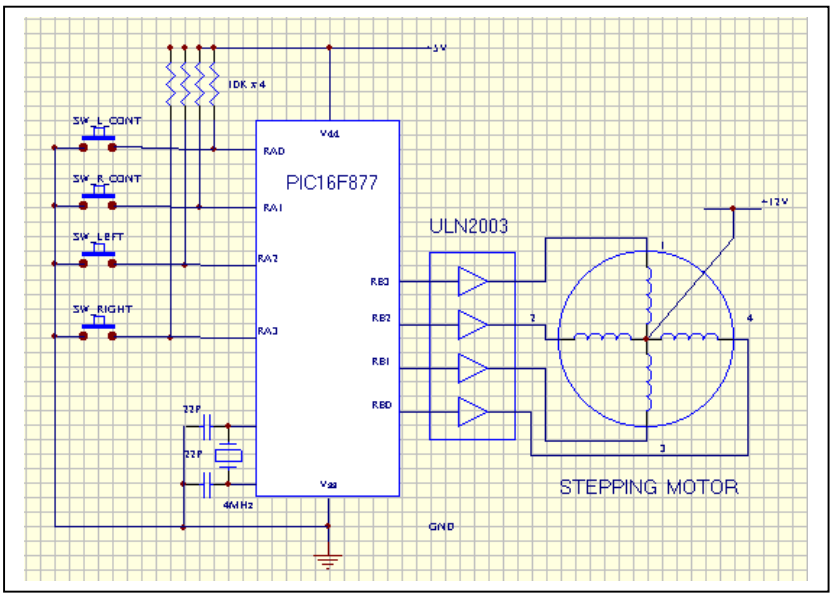
Assignment 13 Stepping Motor Drives Control 3 16

จุดประสงค์ เพื่อเรียนรู้หลักการเขียนโปรแกรม ผลิตสัญญาณ การขับ Stepping Motor แบบ Half – Step เพื่อนำไปประยุกต์ใช้เป็นพื้นฐานการควบคุมตำแหน่งของการเคลื่อนที่กลไกที่มีความละเอียดขึ้น

การทำงานของโปรแกรม ใช้หลักการเช่นเดียวกับโปรแกรมควบคุมการเลื่อนขยับบิทข้อมูลเหมือนไฟวิ่ง. ทีละ 2 หลอด และ 1 หลอด สลับต่อเนื่องกัน เพื่อแบ่งสเต็ปการหมุนให้เป็นทีละครึ่งสเต็ป ทำให้ค่า Step angle หรือองศาการหมุนลดลงครึ่งหนึ่ง ของการขับแบบ Full-Step จากใบงานที่ 11 และ 12 การขับในลักษณะนี้จะทำให้ตำแหน่งสเต็ปการหมุนละเอียดขึ้น แต่แรงบิดจะน้อยลงในบางสเต็ป ตามโปรแกรมนี้สามารถกำหนดการหมุนให้เป็นแบบกดทีละสเต็ป และการหมุนต่อเนื่องได้

```
DELAY       CON   100          'Define DELAY as 100
SW_L_CONT   VAR   PORTA.0
SW_R_CONT   VAR   PORTA.1
SW_LEFT     VAR   PORTA.2
SW_RIGHT    VAR   PORTA.3
STAGE       VAR   BYTE         'Define STAGE as byte variable
MOTOR       VAR   BYTE
L_CONT      VAR   BIT          'Define L_CONT as flag
R_CONT      VAR   BIT          'Define R_CONT as flag

            TRISB  =  %00000000        ‘Port B  as  Output
            STAGE=0                    'Set initial STAGE as 0
            L_CONT=0                   'Clear left continue flag
            R_CONT=0                   'Clear right continue flag
            GOSUB DRIVE                'Drive stepper motor

MAIN:       IF (SW_LEFT) AND (SW_RIGHT) THEN MAIN_1
                                       'Check any switch are presed?
            L_CONT=0 : R_CONT=0        'Clear both flags if not
MAIN_1:     IF SW_LEFT=0  THEN CCW     'CCW when SW_LEFT pressed
            IF SW_RIGHT=0 THEN  CW     'CW when SW_RIGHT pressed
            IF SW_L_CONT=1  THEN MAIN_2:_  'Check left continue switch
            L_CONT=1:R_CONT=0          'Set left cont. flag, clear right cont.

MAIN_2:     IF SW_R_CONT=1 THEN MAIN_3:_   'Check right cont.  switch
            L_CONT=0 : R_CONT=1        'Set right cont. flag, clear left cont.

MAIN_3:     IF L_CONT=1 THEN CCW       'Drive left continue if L_CONT=1
            IF R_CONT=1 THEN CW        'Drive right continue if R_CONT=1
            GOTO MAIN                  'Jump to main

CW:         STAGE=(STAGE+1)/ /8        'Increase stage within 0-8
            GOSUB DRIVE                'Drive stepper motor
            GOTO MAIN                  'Jump to main

CCW:        STAGE=(STAGE-1)/ /8        'Increase stage within 0-8
            GOSUB DRIVE                'Drive stepper motor
            GOTO MAIN                  'Jump to main
```

```
DRIVE:          LOOKUP  STAGE, [%0001, %0011, %0010, %0110, %0100,_
%1100, %1000, %1001], MOTOR
                PORTB   =  MOTOR            'Get data
                PAUSE DELAY                 'Delay
                RETURN                      'Return
                END
```



วงจรตาม Assignment 13 มีสวิทช์ควบคุมการหมุนของสเต็ปปิ้งมอเตอร์ 4 ตัวทำงานดังนี้คือ

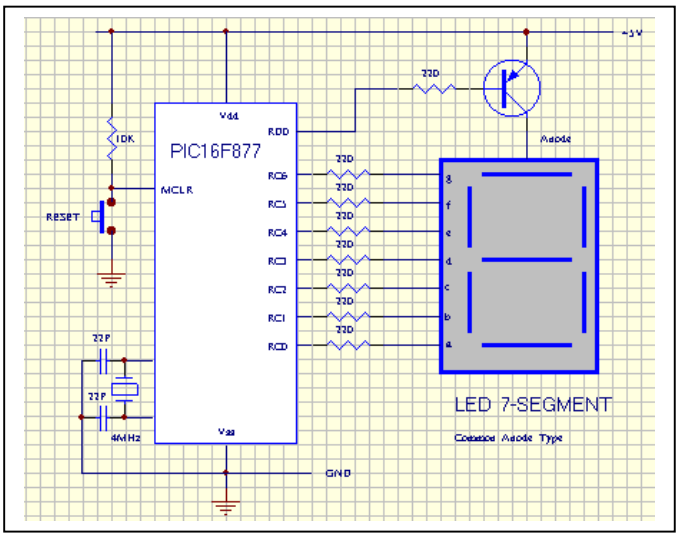
- SW_L_CON และ SW_R_CON สำหรับกดควบคุมการหมุนซ้าย และ ขวา อย่างต่อเนื่อง
- SW_LEFT และ SW_RIGHT สำหรับควบคุมการหมุนซ้าย และ ขวา ทีละสเต็ป

Assignment 14 1 – Digit LED 7-Segment Test 18

จุดประสงค์ เพื่อเรียนรู้หลักการเขียนโปรแกรมขับหลอด LED แบบตัวเลข 7 ส่วน เพื่อเป็นพื้นฐานการแสดงผลค่าต่าง ๆ เช่น จำนวน นาฬิกา ค่าอุณหภูมิ เป็นต้น

การทำงานของโปรแกรม ตามวงจรในใบงานนี้จะใช้ LED 7-Segment แบบอะโหนดร่วม (Common Anode) คือ ขาอะโหนดของ LED ที่ฝังอยู่ในตัวเรือนของ 7-Segment ต่อร่วมกันหมด รวมถึง LED ส่วนที่เป็นจุดทศนิยมด้วย เนื่องจากการต่อร่วมกันทั้ง 8 LED ดังกล่าวนี้ หาก LED ติดหมดทุกดวงเป็นเลข 8 จะทำให้มีกระแสไหลที่จุดต่อร่วมอะโหนดมากถึงประมาณ 100 mA ดังนั้นตามในวงจรจะต้องนำเอาทรานซิสเตอร์มาขับ เพื่อลดกระแสควบคุม ในการเชื่อมต่อกับขา I/O ของไมโครคอนโทรลเลอร์ ตามในวงจรข้างล่างนี้ใช้ทรานซิสเตอร์แบบ PNP ซึ่งขา I/O ต้องส่งลอจิก 0 ออกมาขับ และรวมถึงขา Segment a, b, c, จนถึง g ด้วย จึงจะทำให้หลอดที่เป็น Segment ติดสว่างได้ ตามโปรแกรมนี้ เพียงทำให้หลอดทุก Segment ติดและดับสลับกัน

```
' LED 7-segment connected  at portc
'
            TRISC = %00000000
            LOW   PORTD.0
            PORTC = 0
start:      Pause 1000
            PORTC = 255
            Pause 1000
            PORTC =0
            GoTo  start
            End
```

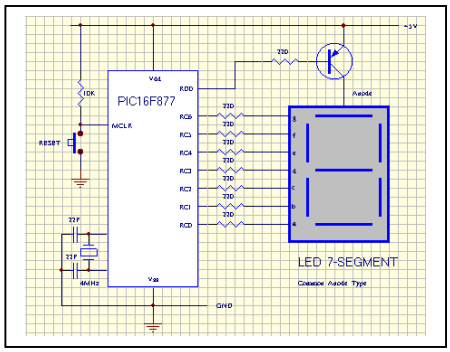


Assignment 15 1 – Digit LED 7-Segment Display 19

จุดประสงค์ เพื่อเรียนรู้หลักการเขียนโปรแกรมขับหลอด LED แบบตัวเลข 7 ส่วน เพื่อเป็นพื้นฐานการแสดงผลค่าต่าง ๆ เช่น จำนวน นาฬิกา ค่าอุณหภูมิ เป็นต้น

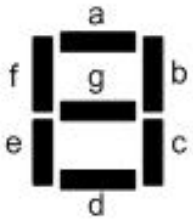
การทำงานของโปรแกรม ตามโปรแกรมในใบงานนี้ ไมโครคอนโทรลเลอร์จะขับหลอด LED แบบตัวเลข 7 ส่วน ให้แสดงตัวเลข 0 ถึง 9 วนต่อเนื่องกันไป ข้อมูลที่เป็นรหัสตัวเลข 0 ถึง 9 จะถูกกำหนดไว้ในคำสั่ง LookUp ในรูปแบบของเลขฐาน 16 โดยเรียงลำดับตั้งแต่เลข 0 คือ $40 ไปเรื่อย ๆ จนถึงเลข 9 คือ $10 กระบวนการนับวน 10 ครั้งจะใช้คำสั่ง FOR.. NEXT กำกับ ในแต่ละครั้ง ค่าตัวแปร num จะเป็นตัวชี้ตำแหน่งข้อมูลตัวเลขที่เป็นรหัสฐาน 16 ในวงเล็บของคำสั่ง LookUp แล้วมากำหนดให้เป็นค่าของตัวแปร disp แล้วส่งค่าออกแสดงผลเป็นค่า Segment ของตัวเลขที่ Port C

```
'---------------------------
TRISC = %00000000
PORTC = 255
num     VAR             BYTE
disp    VAR             BYTE
LOW   PORTD.0
loop:   num = 0
        For num = 0 to 9
                LookUp   num,[$40,$79,$24,$30,$19,_
$12,$02,$78,$00,$10],disp
                PORTC = disp
                Pause 1000
        Next num
        GoTo loop
        End
```



<u>หมายเหตุ</u> ข้อมูลตัวเลข 0 ถึง 9 ที่เป็นรหัสตัวเลขฐาน 16 หามาได้จากตารางข้างล่างนี้

DEC	g	f	e	d	c	b	a	HEX
0	1	0	0	0	0	0	0	$40
1	1	1	1	1	0	0	1	$79
2	0	1	0	0	1	0	0	$24
3	0	1	1	0	0	0	0	$30
4	0	0	1	1	0	0	1	$19
5	0	0	1	0	0	1	0	$12
6	0	0	0	0	0	1	0	$02
7	1	1	1	1	0	0	0	$78
8	0	0	0	0	0	0	0	$00
9	0	0	1	0	0	0	0	$10



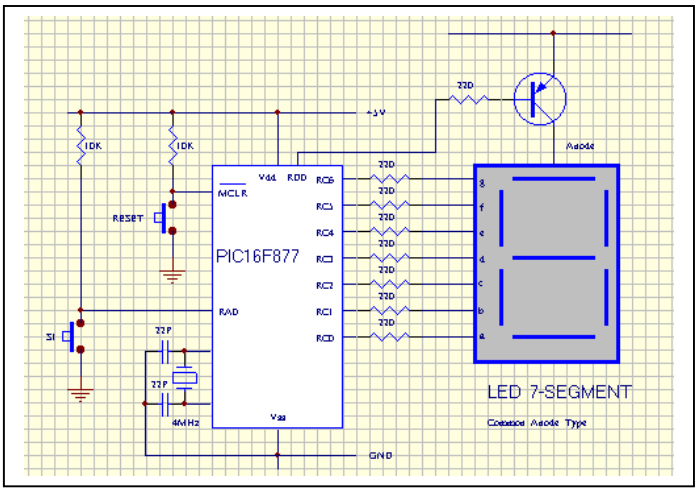
ตามตาราง หากต้องการให้ Segment ใดติด ต้องส่งลอจิก 0 ขับที่ Segment นั้น เนื่องจากเป็นชนิด Common Anode

Assignment 16 1 – Digit LED 7-Segment Display 20

จุดประสงค์ เพื่อเรียนรู้หลักการเขียนโปรแกรมขับหลอด LED แบบตัวเลข 7 ส่วน เพื่อเป็นพื้นฐานการแสดงผลค่าต่าง ๆ เช่น จำนวน นาฬิกา ค่าอุณหภูมิ เป็นต้น

การทำงานของโปรแกรม ตามโปรแกรมในใบงานนี้ ไมโครคอนโทรลเลอร์จะขับหลอด LED แบบตัวเลข 7 ส่วน ให้แสดงตัวเลข 0 ถึง 9 วนต่อเนื่องกันไป ข้อมูลที่เป็นรหัสตัวเลข 0 ถึง 9 จะถูกกำหนดไว้ในคำสั่ง LookUp ในรูปแบบของเลขฐาน 16 เช่นเดียวกันกับใบงานที่ 15 แต่ในใบงานนี้ เราจะต้องใช้สวิตช์ที่ต่ออยู่กับ PORTA.0 เป็นตัวควบคุมว่าจะให้ตัวเลขที่แสดงเพิ่มขึ้นก็ต่อเมื่อเรากดสวิตช์เท่านั้น หากไม่กดตัวเลขจะไม่เปลี่ยนแปลง

```
'RC0=a RC1=b RC2=c RC3=d RC4=e RC5=f RC6=g
'
        TRISC   = %00000000
        TRISA   = %11111111
        PORTC = 255
        s1      VAR     PORTA.0
        num     VAR     BYTE
        disp    VAR     BYTE
        ADCON1 = 7                      'set PORTA to digital mode
        num = 0
        LOW   PORTD.0

chk_close:      IF s1 = 0 Then PAUSE 100: GOTO chk_open
                        GoTo chk_close
chk_open:       IF s1 = 0 Then chk_open
                        Pause 200
                        LookUp num,[$40,$79,$24,$30,$19,_
$12,$02,$78,$00,$10],disp
                        PORTC = disp
                        num = num + 1
                        IF num >= 10 Then num = 0
                GoTo chk_close
                End
```

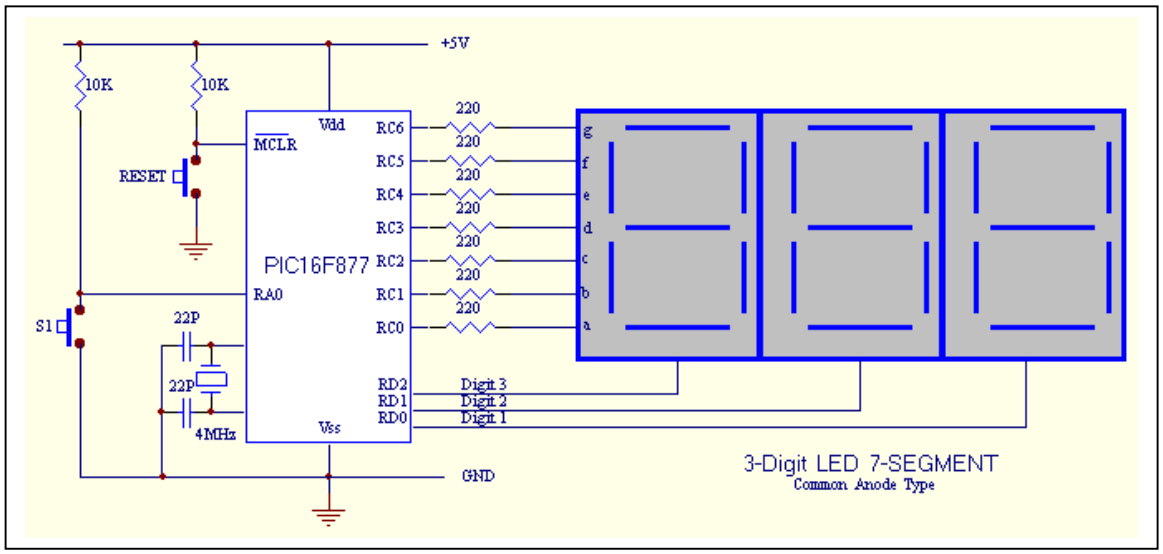


signment 17 3 – Digit LED 7-Segment Display 21

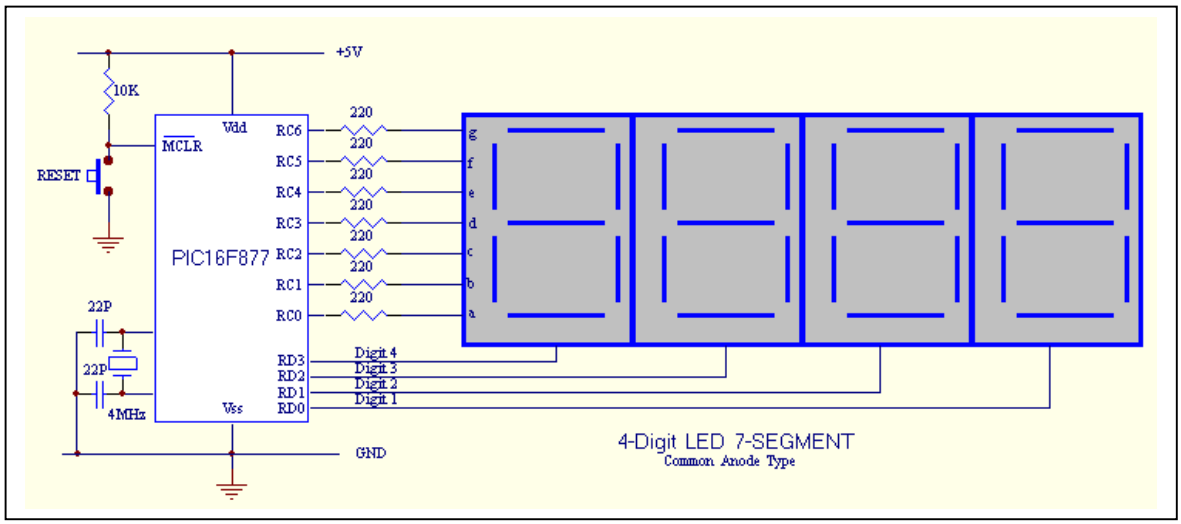
จุดประสงค์ เพื่อเรียนรู้หลักการเขียนโปรแกรมขับหลอด LED แบบตัวเลข 7 ส่วน ที่มากกว่า 1 หลัก เพื่อเป็นพื้นฐานการแสดงผลค่าต่าง ๆ เช่น จำนวน นาฬิกา ค่าอุณหภูมิ เป็นต้น

การทำงานของโปรแกรม ตามโปรแกรมในใบงานนี้ ไมโครคอนโทรลเลอร์จะขับหลอด LED แบบตัวเลข 7 ส่วน โดยให้แสดงเป็น 3 หลัก แบบค่าคงที่ไม่เปลี่ยนแปลง คือ 123 เพื่อให้เข้าใจหลักการแสดงผลตัวเลขหลายหลักแบบมัลติเพล็กซ์ หลักการของโปรแกรมนี้ คือ เราต้องแบ่งหลักของตัวเลขที่อยู่ในรูปของตัวแปร num ออกเป็นหลักหน่วย หลักสิบ และหลักร้อย โดยใช้คำสั่ง digit = num DIG 0 แยกหลักหน่วย digit = num DIG 1แยกหลักสิบ และ digit = num DIG 2 แยกหลักร้อยตามลำดับ จากนั้นเอาตัวเลขที่แยกหลักแล้วมาชี้ตำแหน่งรหัสตัวเลขฐาน 16 ในคำสั่ง LookUp ขับออกแสดงผลเรียงลำดับตั้งแต่ หลักหน่วย หลักสิบ และหลักร้อยตามลำดับ โดยแสดงวนต่อเนื่องเร็ว ๆเกินกว่า 30 ครั้งต่อวินาที จนสายตาจะแยกไม่ออก เสมือนแสดงพร้อม ๆ กันทั้ง 3 หลัก

```
'PortC connect to 7-segment
'RC0=a RC1=b RC2=c RC3=d RC4=e RC5=f RC6=g
'digit0=RD0  digit1=RD1  digit2=RD2
'
        TRISC   = %00000000
        TRISD   = %00000000
        TRISA   = %11111111
        PORTC  = 255
        PORTD  = 255
        digit   VAR     BYTE
        s1      VAR     PORTA.0
        num     VAR     BYTE
        disp    VAR     BYTE
                ADCON1 = 7    'set PORTA to digital mode
                num = 123
loop:           digit = num DIG 0
                LookUp digit,[$40,$79,$24,$30,$19,$12,$02,$78,$00,$10],disp
                PORTC = disp
                PORTD = %11111110
                Pause 5
                '
                digit = num DIG 1
                LookUp digit,[$40,$79,$24,$30,$19,$12,$02,$78,$00,$10],disp
                PORTC = disp
                PORTD = %11111101
                Pause 5
                '
                digit = num DIG 2
                LookUp digit,[$40,$79,$24,$30,$19,$12,$02,$78,$00,$10],disp
                PORTC = disp
                PORTD = %11111011
                Pause 5
                GoTo loop
                End
```



วงจรตาม Assignment 17 แสดงตัวเลข 3 หลักค่าคงที่ “123”



วงจรตาม Assignment 18 แสดงการนับตัวเลข 0 – 9999

Assignment 18 4– Digit 0-9999 Counter Display 23

จุดประสงค์ เพื่อเรียนรู้หลักการเขียนโปรแกรมขับหลอด LED แบบตัวเลข 7 ส่วน ที่มากกว่า 1 หลัก เพื่อเป็นพื้นฐานการแสดงผลค่าต่าง ๆ เช่น จำนวน นาฬิกา ค่าอุณหภูมิ เป็นต้น

การทำงานของโปรแกรม ตามโปรแกรมในใบงานนี้ ไมโครคอนโทรลเลอร์จะขับหลอด LED แบบตัวเลข 7 ส่วน โดยให้แสดงเป็น 4 หลัก โดยแสดงตั้งแต่เลข 0 - 9999โดยมีสวิตช์ที่ต่ออยู่กับ PortA.0 เป็นตัวควบคุมการนับ หากกดแช่ทิ้งไว้ ตัวเลขที่นับไว้จะค้างคงที่อยู่จนกว่าจะถอนนิ้ว การทำงานของโปรแกรมนี้ใช้หลักการแสดงเรียงทีละหลัก และแสดงซ้ำ ๆ เช่นเดียวกัน โดยมีคำสั่ง FOR..NEXT ลูปในสุดเป็นตัวกำกับการแยกหลักและแสดงผลเรียงลำดับ FOR .. NEXT ลูปกลางกำกับการแสดงผลซ้ำ ๆ กันทั้ง 4 หลัก 5 ครั้ง ส่วน FOR .. NEXT ลูปนอกสุดทำหน้าที่เพิ่มค่าตัวแปร num ทีละค่าจาก 0 - 9999

```
        TRISC   =   %00000000
        TRISD   =   %00000000
        TRISA   =   %11111111
        PORTC =   255
        PORTD =   255
        digit     VAR     BYTE
        s1        VAR     PORTA.0
        num       VAR     WORD
        disp      VAR     BYTE
        i         VAR     BYTE
        j         VAR     BYTE
        adcon1  =   7                    'set porta to digital mode
loop:

        For       num = 0 to 9999
                  For i = 0 to 5
                            For     j = 0 to 3
                                    digit = num DIG j
                                    LookUp digit,[$c0,$f9,$a4,$b0,$99,$92,$82,$f8,$80,$90],disp
                                    PORTC = disp
                                    LookUp j,[$fe,$fd,$fb,$f7],PORTD
                                    Pause 5
                                    PORTD=$ff
                            Next j
                  Next i
                  Input s1
                  IF s1=0 Then num=num-1
        Next num
        GoTo loop
        End
```

<u>หมายเหตุ</u> วงจรตามหน้า 23 เมื่อศึกษาและต่อวงจรทดลองจนเข้าใจดีแล้ว จงดัดแปลงวงจรนี้ไปใช้เป็นเครื่องนับจำนวน ที่สามารถรับสัญญาณการนับจากภายนอกได้ และศึกษาไปถึงการตั้งจำนวนการนับได้ และสามารถส่งค่าเอ้าท์พุทออกมาได้ เมื่อนับไปถึงจำนวนที่ตั้งไว้

Assignment 19 4– Digit 24-Hour Clock Display 24

จุดประสงค์ เพื่อเรียนรู้หลักการเขียนโปรแกรมขับหลอด LED แบบตัวเลข 7 ส่วน ที่มากกว่า 1 หลัก เพื่อนำไปประยุกต์เป็นนาฬิกา ตั้งเวลา และจับเวลา

การทำงานของโปรแกรม ตามโปรแกรมในใบงานนี้ เช่นเดียวกันกับใบงานที่ 18 เพียงแต่ใช้คำสั่ง FOR .. NEXT กำกับวงรอบการประมวลผลซ้ำในแต่ละหลัก ให้อยู่ในรูปของ วินาที นาที และชั่วโมง

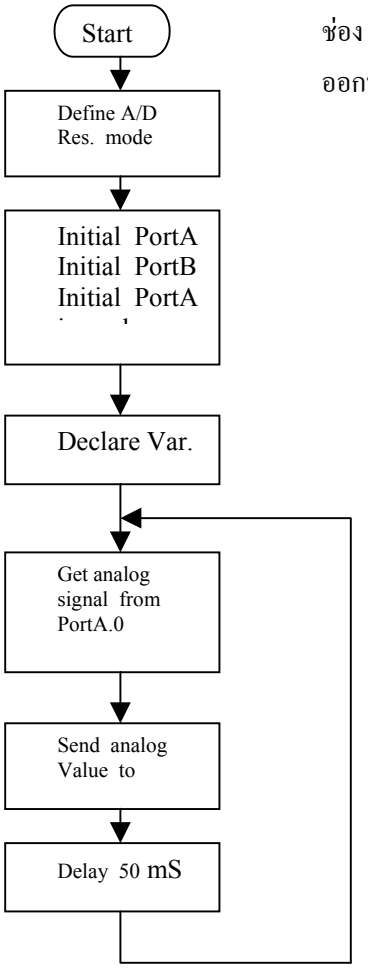
```
'portc connect to 7 segment
'rc0=a rc1=b rc2=c rc3=d rc4=e rc5=f rc6=g
'digit0=rd0 digit1=rd1 digit2=rd2 digit3=rd3

        TRISC   =  %00000000
        TRISD   =  %00000000
        TRISA   =  %11111111
        PORTC = 255
        PORTD = 255
        digit     VAR    BYTE
        num1      VAR    BYTE
        num2      VAR    BYTE
        disp      VAR    BYTE
        i         VAR    BYTE
        j         VAR    BYTE
        k         VAR    BYTE
        adcon1  =   7                          'set porta to digital mode
loop:
        For num2 = 0 to 23                     ‘set for  24 hrs  in  a  day
                For      num1 = 0 to 59        ‘set  for  60  min. in  1  hour
                         For i = 1 to 20       ‘reflesh  display  20  times
                                  For      j = 0 to 1
                                           digit = num1 DIG j
                                           LookUp _
digit,[$c0,$f9,$a4,$b0,$99,$92,$82,$f8,$80,$90],disp
                                           PORTC = disp
                                           LookUp j,[$fe,$fd,$fb,$f7],PORTD
                                           Pause 5
                                           PORTD=$ff
                                           digit = num2 DIG j
                                           LookUp  _
digit,[$c0,$f9,$a4,$b0,$99,$92,$82,$f8,$80,$90],disp
                                           PORTC = disp
                                           LookUp j+2,[$fe,$fd,$fb,$f7],PORTD
                                           Pause 5
                                           PORTD=$ff
                                  Next j
                         Next i
                Next num1
        Next num2
        GoTo loop
        End
```

<u>หมายเหตุ</u> ใช้วงจรตาม Assignment 18 หน้า 23 เมื่อศึกษาเข้าใจดีแล้ว ให้ดัดแปลงไปใช้เป็นนาฬิกา จับเวลา และนาฬิกาปลุก

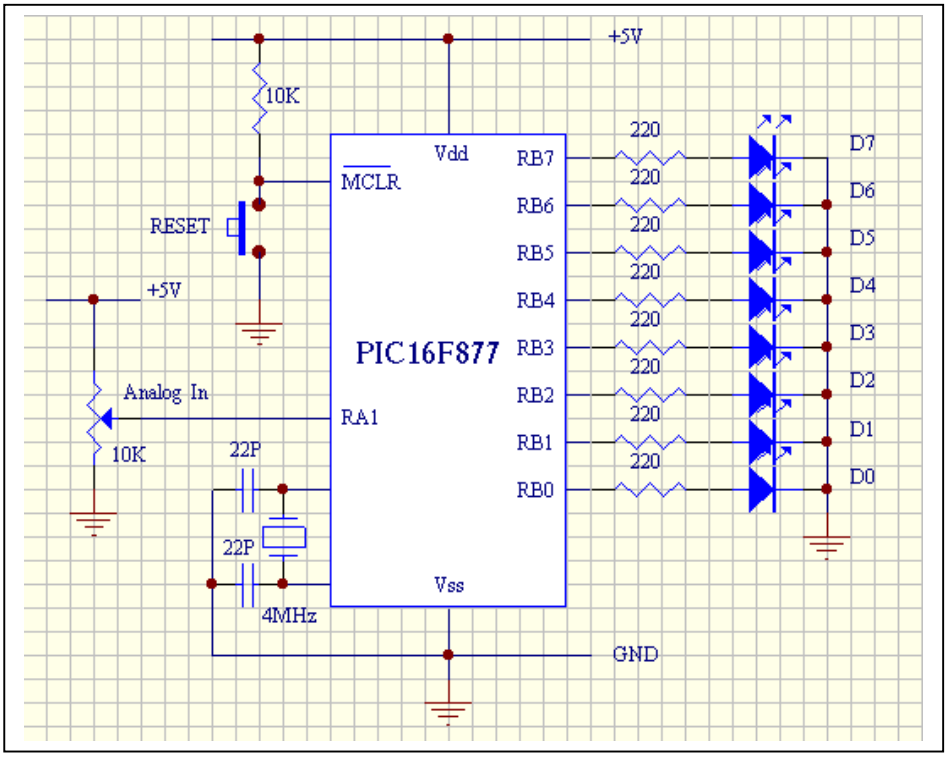
Assignment 20 Analog – to – Digital Converter (1) 25

จุดประสงค์ เพื่อเรียนรู้หลักการเขียนโปรแกรมในการรับสัญญาณอะนาล็อก และแปลงเป็นค่าดิจิตอล เพื่อเป็นพื้นฐานในการเรียนรู้และการประยุกต์ใช้งาน A / D

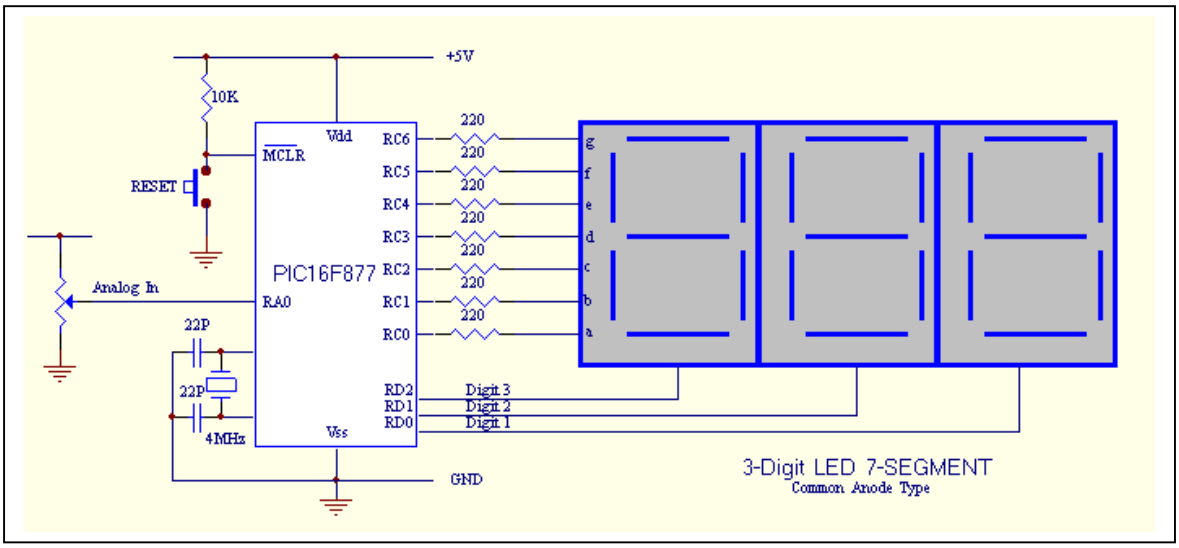
การทำงานของโปรแกรม ในฮาร์ดแวร์ของตัวชิพโครคอนโทรลเลอร์ตระกูล PIC ที่ใช้ตามบอร์ดทดลองนี้ PortA และ Port Eจะถูกออกแบบให้สามารถรับสัญญาณอินพุทแบบอะนาล็อกในระดับ 0–5 V รวมกันได้ถึง 8 ช่อง ในระดับความระเอียดของการแปลงที่ 8 บิท และ 10 บิท การทำงานโปรแกรมนี้ จะรับค่าอะนาล็อกจากช่อง 1 (PortA.1) แล้วส่งค่าที่แปลงเป็นดิจิตอลออกที่ PortB ในรูปแบบของรหัสไบนารี่




```
        analog    VAR   BYTE
        TRISA      =   255                 'Initial PORTA as input
        TRISB      =   0                   'Initial PORTB as output
        ADCON1  =     0                    'Set PORTA as in analog mode
main:   ADCIN 1,analog              'Get analog signal to variable
        PORTB    =      analog      'Send analog to display at portb
        Pause     50
        GoTo      main
        End
```



วงจรตาม Assignment 20



วงจรตาม Assignment 21

Assignment 21 Analog – to – Digital Converter (2) 27

จุดประสงค์ เพื่อเรียนรู้หลักการเขียนโปรแกรมในการรับสัญญาณอะนาล็อก และแปลงเป็นค่าดิจิตอล ในลักษณะเป็นตัวเลขฐาน 10 เพื่อเป็นพื้นฐานในการเรียนรู้และการประยุกต์ใช้งาน A / D

การทำงานของโปรแกรม เช่นเดียวกันกับใบงานที่ 20 แต่ใบงานนี้นำเอาค่าดิจิตอลที่แปลงมาได้มาแสดงผลผ่านทาง LED 7-Segment ในรูปแบบของตัวเลขฐาน 10 จำนวน 3 หลัก

ในการเขียนโปรแกรมสำหรับแปลงค่า A/D ต้องลำดับขั้นตอนดังต่อไปนี้

1. กำหนดให้ PortA และ PortE ที่รับสัญญาณอะนาล็อก เป็นอินพุท
2. ใช้คำสั่งสำหรับแปลงค่า A/D เก็บค่าไว้ในตัวแปร

```
'
        TRISC = %00000000
        TRISD = %00000000
        TRISA = %11111111
        DEFINE ADC_BITS     8
        ADCON1      =       0
        analog      VAR         WORD
        digit       VAR         BYTE
        disp        VAR         BYTE
main:   ADCIN 1,analog
        Pause 5
        digit = analog DIG 0
        LookUp digit,[$40,$79,$24,$30,$19,$12,$02,$78,$00,$10],disp
        PORTC = disp
        PORTD = %11111110
        Pause 5
        '
        digit = analog DIG 1
        LookUp digit,[$40,$79,$24,$30,$19,$12,$02,$78,$00,$10],disp
        PORTC = disp
        PORTD = %11111101
        Pause 5
        '
        digit = analog DIG 2
        LookUp digit,[$40,$79,$24,$30,$19,$12,$02,$78,$00,$10],disp
        PORTC = disp
        PORTD = %11111011
        Pause 5
        GoTo main
        End
```

หมายเหตุ ต่อวงจรทดลองตามหน้า 26

Assignment 22 4–Digits 0–5 Vdc Digital Voltmeter 28

จุดประสงค์ เพื่อเรียนรู้หลักการเขียนโปรแกรมในการรับสัญญาณอะนาล็อก และแปลงเป็นค่าดิจิตอล เพื่อเป็นพื้นฐานในการเรียนรู้และการประยุกต์ใช้งานด้านการสร้างโวลท์มิเตอร์

การทำงานของโปรแกรม เช่นเดียวกันกับใบงานที่ 21 แต่ใบงานนี้นำเอาค่าดิจิตอลที่แปลงมาแล้วนำค่ามาคำนวน เพื่อให้ได้ผลลัพท์ Full sacle ที่ 5 โวลท์พอดี แล้วนำออกแสดงผลทาง LED 7-Segment แบบ 4 หลัก

‘

```
        DEFINE ADC_BITS 8
        TRISA  =  255
        TRISB  =  0
        TRISC  =  0
        TRISD  =  0
        PORTC = 255
        PORTD = 255
        digit      VAR   BYTE
        disp       VAR   BYTE
        i          VAR   BYTE
        c          VAR   BYTE
        analog     VAR   WORD
        ADCON1  =  0
        analog     =  0
loop:
        For     c = 0 to 3
                digit = analog DIG c
                LookUp _
digit,[$c0,$f9,$a4,$b0,$99,$92,$82,$f8,$80,$90],disp
                IF c=2 Then disp = disp & $7f
                PORTD = disp
                LookUp c,[$fe,$fd,$fb,$f7],PORTC
                ADCIN 1,analog
                analog   =  (analog*100)/51
                Pause 1
                PORTC=$ff
        Next c
                GoTo loop
                End
```

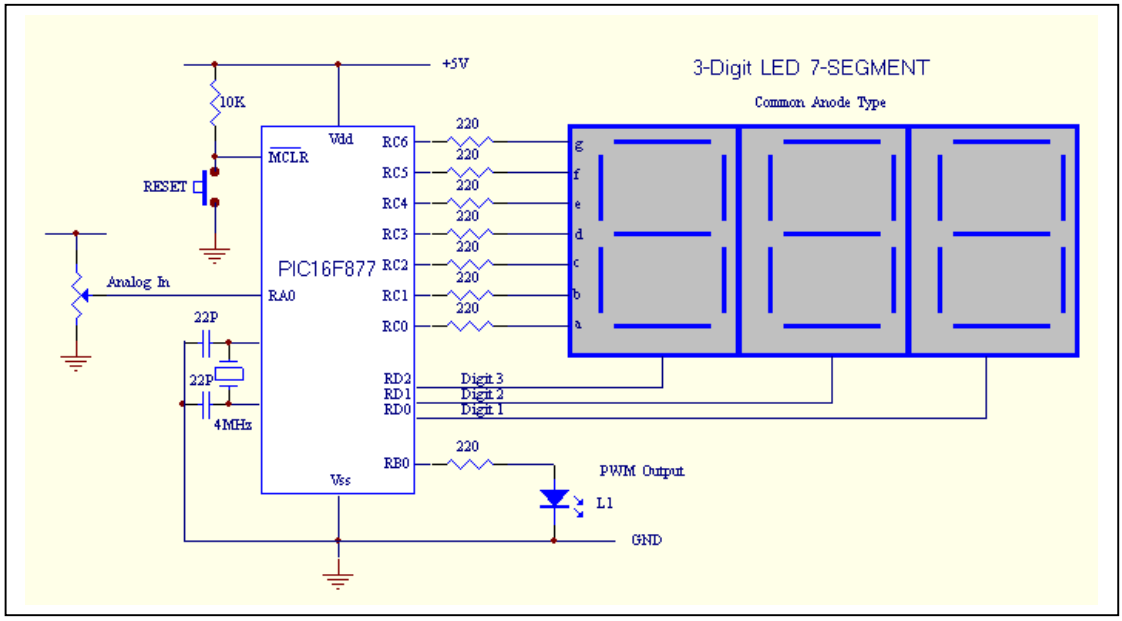
<u>หมายเหตุ</u> ต่อวงจรทดลองตามวงจรของ Assignment 21 โดยต่อ LED 7-Segment Display เพิ่มเป็น 4 หลัก

Assignment 23 Pulse Width Modulation (PWM) (2) 29

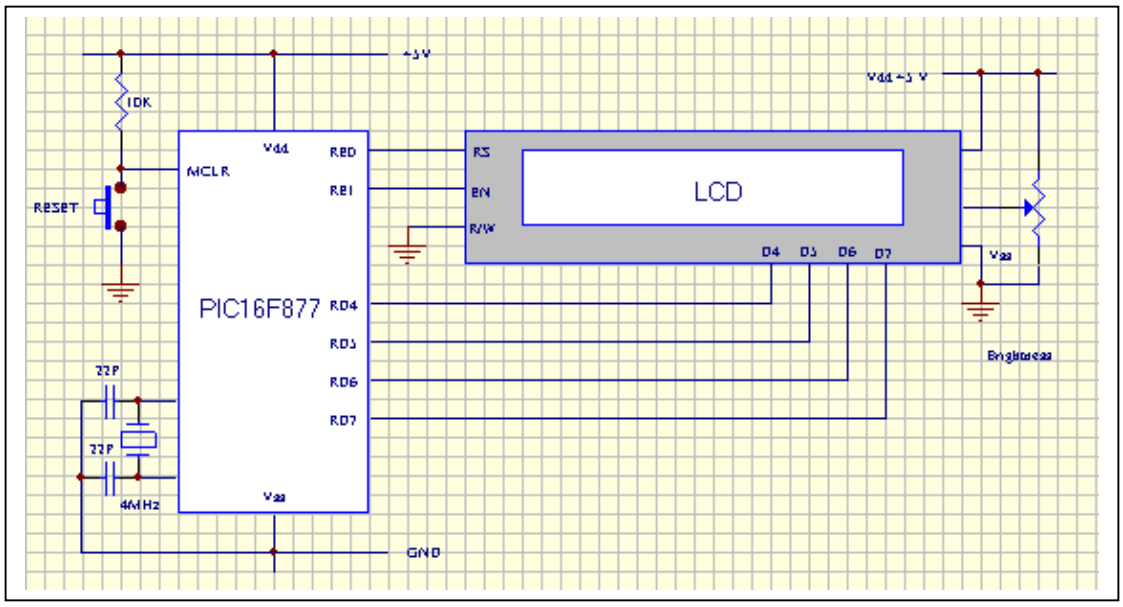
จุดประสงค์ เพื่อเรียนรู้หลักการเขียนโปรแกรมในการรับสัญญาณอะนาล็อก และแปลงเป็นค่าดิจิตอล เพื่อนำไปประยุกต์ใช้ควบคุมกำลังไฟดีซี ได้แก่มอเตอร์และหลอดไฟด้วยวิธีการปรับแรงดัน A/D

การทำงานของโปรแกรม เช่นเดียวกันกับใบงานที่ 21 แต่ใบงานนี้นำเอาค่าดิจิตอลที่แปลงมาได้มาแสดงผลผ่านทาง LED 7-Segment และกำหนดเป็นค่าดิวตี้ไซเกิล (Duty Cycle) ของคำสั่งสร้างสัญญาณ PWM เพื่อนำไปขับโหลดภายนอกที่ต่ออยู่กับ PortB.2

```
        DEFINE ADC_BITS     8
        TRISC       =       %00000000
        TRISD       =       %00000000
        TRISA       =       %11111111
        ADCON1      =       0
        analog      VAR             WORD
        digit       VAR             BYTE
        disp        VAR             BYTE
        M1          VAR             PORTB.2
main:   ADCIN 1,analog
                Pauseus  50
                digit = analog DIG 0
                LookUp digit,[$40,$79,$24,$30,$19,$12,$02,$78,$00,$10],disp
                PORTC = disp
                PORTD = %11111110
                Pauseus  50
                digit = analog DIG 1
                LookUp digit,[$40,$79,$24,$30,$19,$12,$02,$78,$00,$10],disp
                PORTC = disp
                PORTD = %11111101
                Pauseus  50
                digit = analog DIG 2
                LookUp digit,[$40,$79,$24,$30,$19,$12,$02,$78,$00,$10],disp
                PORTC = disp
                PORTD = %11111011
                Pauseus  50
                PWM M1,analog,2
                GoTo main
                End
```



วงจรทดลอง ตาม Assignment 23



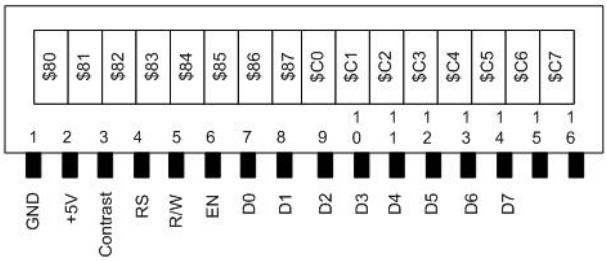
วงจรทดลองตาม Assignment 24

Assignment 24 LCD Module Display 31

จุดประสงค์ เพื่อเรียนรู้หลักการเขียนโปรแกรมแสดงผลออกทางจอ LCD เพื่อเป็นพื้นฐานในการประยุกต์ใช้งานจริงในระบบแสดงผลของอุปกรณ์ต่าง ๆ และในอุตสาหกรรม

การทำงานของโปรแกรม ในโครงสร้างของจอ LCD จะประกอบด้วย

1. หน้าจอที่แสดงตัวอักขระ โดยทั่วไปจะมีตั้งแต่ 1 บรรทัด 16 ตัวอักขระ ไปจนถึง 4 บรรทัด 16 ตัวอักขระต่อ บรรทัด โดยมี MCU เป็นตัวควบคุมการแสดงผล
2. ขั้วต่อ โดยทั่วไปจะมี 16 ขาตามรูปข้างล่างนี้



จากรูป แสดงตำแหน่งขา และตำแหน่งแอดเดรสของตัวอักขระของจอ LCD แบบ 1 บรรทัด 16 ตัวอัขระ

การเขียนโปรแกรมภาษา PIC BASIC PRO ต้องประกอบด้วยขั้นตอนต่อไปนี้

1. ต้องกำหนดตำแหน่ง ขาของจอ LCD ที่จะต่อกับขา I/O ของ MCU ด้วยคำสั่ง DEFINE ทั้ง 6 บรรทัด
2. ใช้คำสั่งแสดงผล คือ LCDOUT และหากต้องการจะส่ง Command ให้ CPU ของจอ จะต้องมี $FE, นำหน้าเสมอ เช่น $FE, 1 คือลบหน้าจอ $FE, $83 คือไปเริ่มแสดงผลที่ตำแหน่งแอดเดรสของตัวอักขระที่ $83 เป็นต้น

```
        DEFINE  LCD_DREG    PORTD
        DEFINE  LCD_DBIT    4
        DEFINE  LCD_RSREG   PORTE
        DEFINE  LCD_RSBIT   0
        DEFINE  LCD_EREG    PORTE
        DEFINE  LCD_EBIT    1
        ADCON1  = 7
start:          LCDOut $FE,1,$83,"Hello.."
                Pause 2000
                LCDOut $FE,1,$83,"My name is"
                Pause 2000
                LCDOut $FE,1,$83,"Mr.PIC16F877"
                Pause 2000
                LCDOut $FE,1,$C5,"Staff Development Institute"
                Pause 2000
                For I = 1 to 5
                        LCDOut $FE,1,$83,"Hello..!"
                        Pause 200
                        LCDOut $FE,1,$83,"           "
                        Pause 200
                Next I
                Pause 2000
                GoTo start
End
```

Assignment 25 Sending Frequency Output signal 32

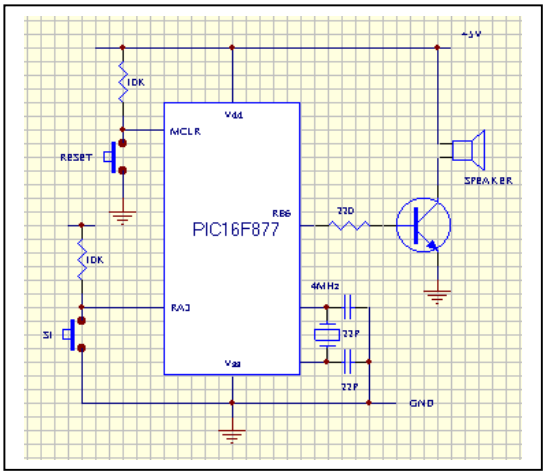
จุดประสงค์ เพื่อเรียนรู้หลักการเขียนโปรแกรมขับความถี่สัญญาณเสียงออกทางขา I/O ของ MCU เพื่อเป็นพื้นฐานนำไปประยุกต์ใช้งาน เช่นสัญญาณเตือน เสียงคลิกเมื่อกดสวิตช์ เป็นต้น

การทำงานของโปรแกรม ใช้คำสั่ง FREQOUT โปรแกรมแรก เมื่อกดสวิตช์ โปรแกรมจะสร้างสัญญาณความถี่ 3000 Hz เป็นเวลา 1500 mSec ออกที่ขา PortB.6 ซึ่งต่อกับลำโพง

ส่วนโปรแกรมที่สอง จะส่งออกมา 2 ความถี่

```
'Program 1
'Single tone frequency
'
        SPK     VAR     PORTB.6
        S1      VAR     PORTA.3
                PORTB.0 = 1
LOOP1:  IF  (S1 = 1)  THEN  LOOP1
            FREQOUT  SPK, 1500, 3000

            END
```

```
'Program 2
'Double  Frequency  Output

        SPK     VAR     PORTB.6
        S1      VAR     PORTA.3
        i       VAR    BYTE
loop1:  IF  (S1 = 1)  THEN  LOOP1
        for   i =   1   to   5
                FREQOUT  SPK,1500,500,3500
                LOW  SPK
                pause  400
        next  i
                goto  loop1
                END
```



Assignment 26 Stop MCU working for a while 33

จุดประสงค์ เพื่อเรียนรู้หลักการเขียนโปรแกรมให้ไมโครคอนโทรลเลอร์หยุดพักการประมวลผลชั่วขณะเพื่อเข้าสู่โหมดการประหยัดพลังงาน

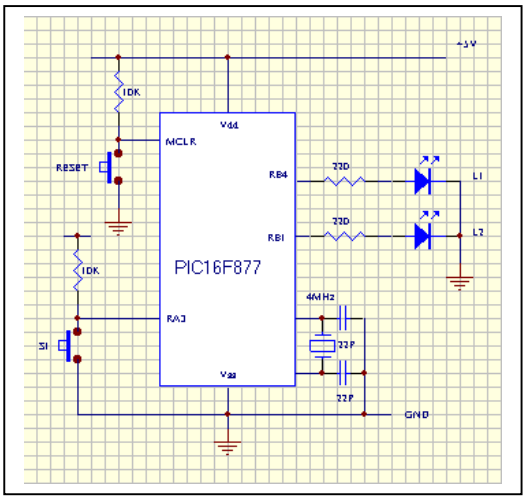
การทำงานของโปรแกรม คำสั่งที่ใช้มี 2 คำสั่งได้แก่ NAP และ SLEEP

คำสั่ง NAP เป็นคำสั่งให้ MCU หยุดพักการทำงานในช่วงเวลาสั้น ๆ โดยจะไม่รับคำสั่งใด ๆ และจะใช้พลังงานน้อยที่สุด ประมาณ 20 ไมโครแอมป์

คำสั่ง SLEEP เป็นคำสั่งให้ MCU หยุดการประมวลผลครั้งละ 2.3 วินาทีโดยจะไม่รับคำสั่งใด ๆ และจะใช้พลังงานน้อยที่สุด ประมาณ 20 ไมโครแอมป์เช่นกัน แต่คำสั่ง SLEEP จะมีตัวคูณเวลาการหยุดพัก ซึ่งค่ามากที่สุดจะได้นานถึง 18 ชั่วโมง ถ้าจะต้องการให้ตื่นต้องปลุกด้วยการกดปุ่ม Reset รายละเอียดคำสั่งทั้งสองให้ศึกษาเพิ่มเติมจากคู่ประกอบการใช้โปรแกรม PIC BASIC PRO COMPILER

```
'Program 1
            L1      VAR     PORTB.4
            L2      VAR     PORTB.1
            S1      VAR     PORTA.3
LOOP1:              INPUT   S1
            IF  S1 = 1  THEN  LOOP1
LOOP2:              HIGH  L1
            PAUSE  5000
            LOW  L1
            NAP 7
LOOP3:              HIGH  L2
            PAUSE 5000
            LOW  L2
            NAP 6
            GOTO  LOOP1
            END
```

```
'Program 2

      S1    VAR     PORTA.3
      L1    VAR     PORTB.1
LOOP1:      INPUT  S1
            IF  S1 = 1  THEN  LOOP1
LOOP2:      HIGH  L1
            PAUSE  2000
            LOW  L1
            SLEEP  10
            GOTO  LOOP1
            END
```



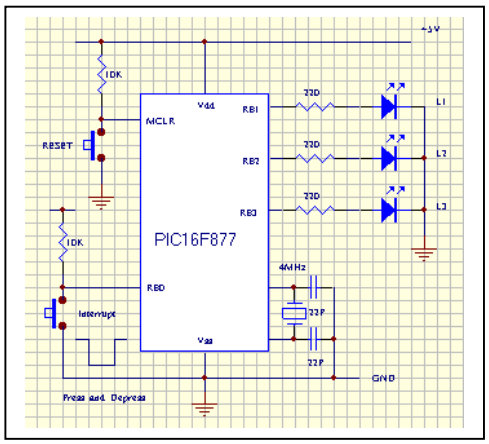
Assignment 27 **Interrupt the MCU working**

จุดประสงค์ เพื่อเรียนรู้หลักการเขียนโปรแกรมให้ไมโครคอนโทรลเลอร์รับสัญญาณจากการขอขัดจังหวะการทำงาน(Interrupt) และให้บริการ Interrupt เพื่อเป็นพื้นฐานในการนำไปประยุกใช้ออกแบบสร้างปุ่มฉุกเฉินของเครื่องมือ เครื่องจักรต่าง ๆ

การทำงานของโปรแกรม การใช้คำสั่งให้ทำงานให้บริการ Interrupt มี 3 ขั้นตอน คือ

1. คำสั่ง ON INTERRUPT มีไว้เพื่อบอกว่าหากเกิด Interrupt จะไปทำงานที่ใด
2. INTCON = %10010000 เป็นรีจิสเตอร์ที่ต้องกำหนดค่า เพื่อขอใช้บริการ Interrupt
3. DISABLE มีไว้คั่นส่วนของ Main Program กับ Interrupt routine และยกเลิกหลังจากให้บริการ Interrupt ครั้งก่อน
4. ส่วน INTCON = %10010000
   RESUME
   ENABLE
   มีไว้ใน Interrupt Routine เพื่อเริ่มการให้บริการใหม่หลังจากเกิด Interrupt แล้ว

```
L2      VAR     PORTB.2
L3      VAR     PORTB.3
'
ON INTERRUPT GOTO LOOP2
INTCON = %10010000

LOOP1:          HIGH  L2
                PAUSE
                LOW  L1
                PAUSE  1000
                HIGH  L2
                PAUSE  1000
                LOW  L2
                PAUSE  1000
                GOTO  LOOP1
                DISABLE

LOOP2:          HIGH  L3
                PAUSE  5000
                LOW  L3
                PAUS  2000
                INTCON = %10010000
                RESUME
                ENABLE
                END
```



Assignment 28 Scan key (1) 35

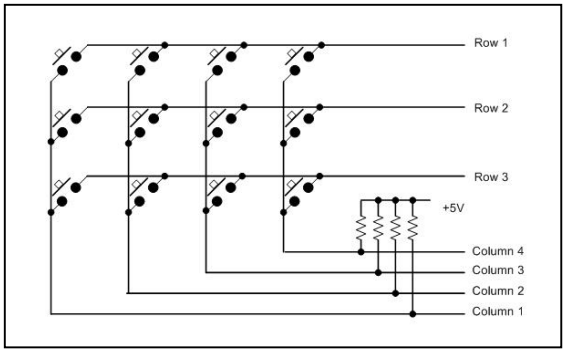
จุดประสงค์ เพื่อเรียนรู้หลักการเขียนโปรแกรมให้ไมโครคอนโทรลเลอร์รับสัญญาณจาก Key Pad หรือ Key matrix ที่นำมาเชื่อมต่อกับไมโครคอนโทรลเลอร์ เพื่อนำไปประยุกต์ใช้งานเกี่ยวกับการออกแบบแผงคอนโซลที่มีจำนวนสวิตช์อินพุทมาก ๆ โดยมีขา I/O ของไมโครคอนโทรลเลอร์จำนวนไม่เพียงพอ

การทำงานของโปรแกรม Matrix Switches โดยทั่วไปออกแบบต่อสวิตช์ให้เป็นลักษณะที่เป็น Row กับ Column ตัดกัน โดยสวิตช์ทุกตัวขาข้างหนึ่งจะต่อกับ Row และอีกข้างหนึ่งจะต่อกับ Column ขา I/O ของ MCU ที่ต่อกับ Row ทุกขาจะถูกกำหนดให้เป็น Output ส่วนขา I/O ของ MCU ที่ต่อกับ Column ทุกขาจะถูกกำหนดให้เป็น Input โดยต่อ R Pull-up ไว้เพื่อให้เป็นลอจิก 1 ไว้ก่อน

หลักการสแกนรับอินพุท เริ่มต้นที่ส่งลอจิก 0 ไปที่ Row 1 ก่อน แล้วเขียนโปรแกรมเช็คลอจิก 0 ที่ขา Column ทั้ง 4 ขาว่ามีหรือไม่ หากมีแสดงว่ามีการกดสวิตช์ตัวใดตัวหนึ่งที่ต่อกับ Row 1 เสร็จแล้วให้ส่งลอจิก 1 ไปที่ Row 1 เพื่อคืนสถานะให้เป็นลอจิก 1 เหมือนเดิม

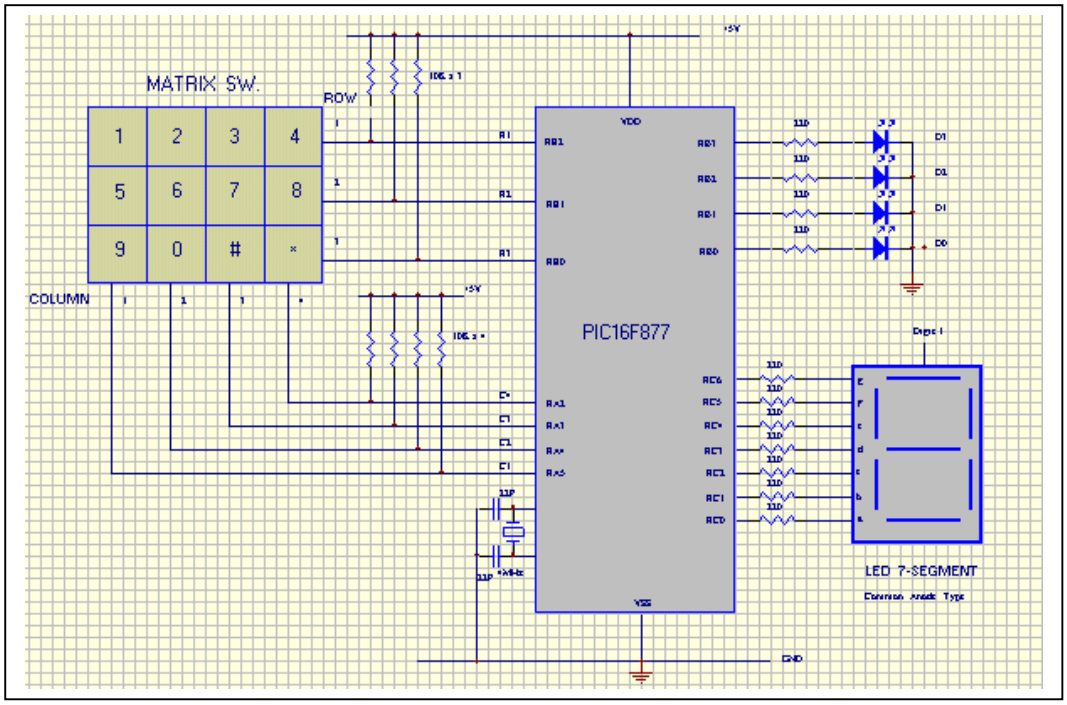
ในทำนองเดียวกัน ให้ทำกับ Row 2 และ Row 3 เมื่อเสร็จแล้วให้วนกลับไปทำกับ Row 1 ใหม่ไปเรื่อย ๆ กระบวนการนี้เรียกว่า “Scan Key” การใช้ Scan Key มีข้อดีคือ หากมีขา I/O ต่อกับ Row เพียง 8 ขา และColumn 8 ขา จะสามารถต่อสวิตช์รับอินพุทได้ถึง 64 ตัว

```
TRISB = %00000000
TRISC = %00000000
TRISD = %00000000
TRISE =  %00000000
TRISA =  %11111111
'
R1      VAR     PORTE.2
R2      VAR     PORTE.1
R3      VAR     PORTE.0
'
C1      VAR     PORTA.5
C2      VAR     PORTA.4
C3      VAR     PORTA.3
C4      VAR     PORTA.2
'
        PORTB = 0
        LOW  PORTD.0
        ADCON1  =  7
        High    R1
        High    R2
        High    R3
```



```
start:
loop1:  Low     R1
                IF C1 = 0 Then disp1
                IF C2 = 0 Then disp2
                IF C3 = 0 Then disp3
                IF C4 = 0 Then disp4
        High    R1

loop2:  Low     R2
                IF C1 = 0 Then disp5
                IF C2 = 0 Then disp6
                IF C3 = 0 Then disp7
                IF C4 = 0 Then disp8
        High    R2


loop3:  Low     R3
                IF C1 = 0 Then disp9
                IF C2 = 0 Then disp0
                IF C3 = 0 Then dispa
                IF C4 = 0 Then dispb
        High    R3
                Pause   100
                GoTo    start

disp1:  PORTB = 1
                PORTC = $79
                GoTo    loop1
disp2:  PORTB = 2
                PORTC = $24
                GoTo    loop1
disp3:  PORTB = 3
                PORTC = $30
                GoTo    loop1
disp4:  PORTB = 4
                PORTC = $19
                GoTo    loop1
disp5:  PORTB = 5
                PORTC = $12
                GoTo    loop2
disp6:  PORTB = 6
                PORTC = $02
                GoTo    loop2
disp7:  PORTB = 7
                PORTC = $78
                GoTo    loop2
disp8:  PORTB = 8
                PORTC = $00
                GoTo    loop2
disp9:  PORTB = 9
                PORTC = $10
                GoTo    loop3
disp0:  PORTB = 0
                PORTC = $40
                GoTo    loop3
dispa:  PORTB = 10
```

```
                PORTC = $08
                GoTo    loop3
dispb:   PORTB = 11
                PORTC = $03
                GoTo    loop3
                End
```



วงจรตาม Assignment 28, 29

Assignment 29 Scan key (2) 38

จุดประสงค์ เพื่อเรียนรู้หลักการเขียนโปรแกรมให้ไมโครคอนโทรลเลอร์รับสัญญาณจาก Key Pad หรือ Key matrix ที่นำมาเชื่อมต่อกับไมโครคอนโทรลเลอร์ เพื่อนำไปประยุกต์ใแสดงผลเป็นตัวเลข

```
'
        TRISB = %00000000
        TRISC = %00000000
        TRISD = %00000000
        TRISE =  %00000000
        TRISA =  %11111111
        '
        R1      VAR     PORTE.2
        R2      VAR     PORTE.1
        R3      VAR     PORTE.0
        '
        C1      VAR     PORTA.5
        C2      VAR     PORTA.4
        C3      VAR     PORTA.3
        C4      VAR     PORTA.2
        '
                PORTB = 0
                LOW  PORTD.0
                ADCON1  =  7
                High    R1
                High    R2
                High    R3
start:
loop1:  Low     L1
                IF C1 = 0 Then disp1
                IF C2 = 0 Then disp2
                IF C3 = 0 Then disp3
                IF C4 = 0 Then disp4
        High    L1

loop2:  Low     L2
                IF C1 = 0 Then disp5
                IF C2 = 0 Then disp6
                IF C3 = 0 Then disp7
                IF C4 = 0 Then disp8
        High    L2

loop3:  Low     L3
                IF C1 = 0 Then disp9
                IF C2 = 0 Then disp0
                IF C3 = 0 Then dispa
                IF C4 = 0 Then dispb
        High    L3
                Pause   100
                GoTo    start

disp1:  PORTC = 1
```

```
            PORTD = $79
            GoTo   loop1
disp2:  PORTC = 2
            PORTD = $24
            GoTo   loop1
disp3:  PORTC = 3
            PORTD = $30
            GoTo   loop1
disp4:  PORTC = 4
            PORTD = $19
            GoTo   loop1
disp5:  PORTC = 5
            PORTD = $12
            GoTo   loop2
disp6:  PORTC = 6
            PORTD = $02
            GoTo   loop2
disp7:  PORTC = 7
            PORTD = $78
            GoTo   loop2
disp8:  PORTC = 8
            PORTD = $00
            GoTo   loop2
disp9:  PORTC = 9
            PORTD = $10
            GoTo   loop3
disp0:  PORTC = 0
            PORTD = $40
            GoTo   loop3
dispa:  PORTC = 10
            PORTD = $08
            GoTo   loop3
dispb:  PORTC = 11
            PORTD = $03
            GoTo   loop3
            End
```

งานที่มอบหมาย เมื่อเข้าใจการทำงานของโปรแกรมดีแล้ว ให้ประยุกต์โดยใส่เสียงคลิกขณะกดแป้นคีย์ และเขียนโปรแกรมเพิ่มเติมเพื่อนำผลของการกดคีย์แต่ละคีย์ไปใช้งาน เช่น เปิด-ปิดหลอดไฟ หรือควบคุมการทำงานของมอเตอร์ เป็นต้น

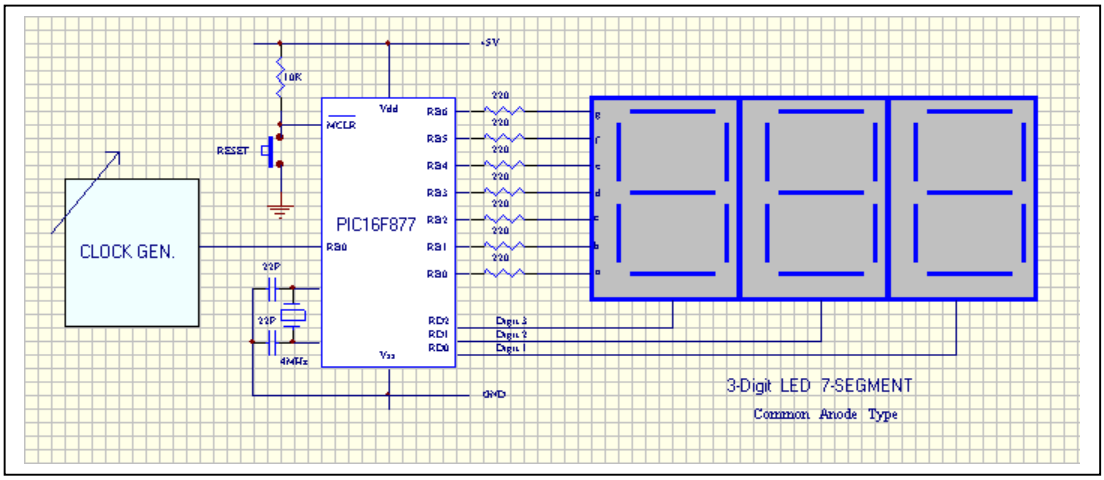
Assignment 30 Counting Pulses From I/O Port

จุดประสงค์ เพื่อเรียนรู้หลักการเขียนโปรแกรมให้ไมโครคอนโทรลเลอร์นับจำนวนลูก Pulse ที่มาปรากฏที่ขา I/O เพื่อเป็นพื้นฐานในการนำไปประยุกต์ใช้ในระบบการนับจำนวน หรือความถี่ต่าง ๆ ได้

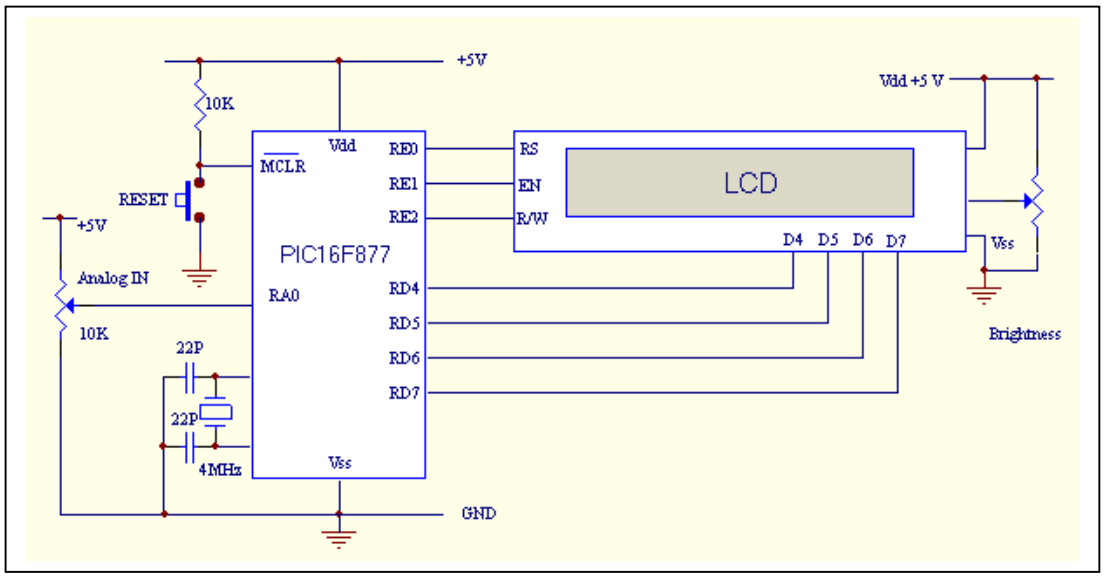
หลักการทำงานของโปรแกรม โดยการใช้คำสั่ง COUNT เพื่อนับจำนวนลูก Pulse บวกในจำนวนคาบเวลาที่กำหนด แล้วเก็บไว้ที่ตัวแปร เพื่อนำค่าไปใช้งานต่อไป

```
'PortC connect to 7-segment
'RC0=a RC1=b RC2=c RC3=d RC4=e RC5=f RC6=g
'digit0=RD0  digit1=RD1  digit2=RD2
'count input = PORTB.0
'
        TRISC = %00000000
        TRISD = %00000000
        TRISA = %11111111
        ADCON1  =     7
        num     VAR     BYTE
        digit   VAR     BYTE
        disp    VAR     BYTE
main:   COUNT  PORTB.0,100,num
                Pauseus  5
                digit = num DIG 0
                LookUp digit,[$40,$79,$24,$30,$19,$12,$02,$78,$00,$10],disp
                PORTC = disp
                PORTD = %11111110
                Pauseus  5
                '
                digit = num DIG 1
                LookUp digit,[$40,$79,$24,$30,$19,$12,$02,$78,$00,$10],disp
                PORTC = disp
                PORTD = %11111101
                Pauseus  5
                '
                digit = num DIG 2
                LookUp digit,[$40,$79,$24,$30,$19,$12,$02,$78,$00,$10],disp
                PORTC = disp
                PORTD = %11111011
                GoTo main
                End
```

มอบหมายงาน เมื่อเข้าใจการใช้คำสั่งดีแล้ว ให้ประยุกต์การแสดงผลออกทางจอ LCD



วงจรตาม Assignment 30



วงจรตาม Assignment 31

Assignment 31 Read 8 – Bit A/D and Display on LCD 42

จุดประสงค์ เพื่อเรียนรู้หลักการเขียนโปรแกรมให้ไมโครคอนโทรลเลอร์รับสัญญาณอะนาล็อก และแปลงเป็นค่าดิจิตอลขนาดความละเอียด 8 บิท เพื่อเป็นพื้นฐานการประยุกต์ใช้ในระบบแสดงผลการวัดค่าอะนาล็อกต่าง ๆ ได้แก่ อุณหภูมิ ความดัน ระดับ ฯลฯ แล้วนำค่ามาแสดงผลทางจอ LCD

```
' Define LCD registers and bits
DEFINE      LCD_DREG    PORTD
DEFINE      LCD_DBIT    4
DEFINE      LCD_RSREG   PORTE
DEFINE      LCD_RSBIT   0
DEFINE      LCD_EREG    PORTE
DEFINE      LCD_EBIT    1

        adval     VAR BYTE          ' Create adval to store result
        TRISA   = %11111111         ' Set PORTA to all input
        ADCON1 = %00000010          ' Set PORTA analog
        Low PORTE.2                 ' LCD R/W line low (W)

        Pause 500                   ' Wait .5 second

loop:   ADCIN 0, adval              ' Read channel 0 to adval

        LCDOut $fe, 1               ' Clear LCD
        LCDOut "Value: ", DEC adval ' Display the decimal value

        Pause 100                   ' Wait .1 second

        GoTo loop                   ' Do it forever
        End
```

มอบหมายงาน เมื่อศึกษาการทำงานของโปรแกรมเข้าใจดีแล้ว ให้พัฒนาโปรแกรมที่สามารถใช้สวิตช์กดตั้งค่าที่สมมุติเป็นอุณหมิ แล้วนำมาเปรียบเทียบกันค่าของ A/D ที่รับเข้ามา แล้วสั่งการให้เปิดปิดรีเลย์ควบคุมคอมเพรสเซอร์ (Portb.5) พร้อมทั้งเพิ่มเสียงคลิกขณะที่กดสวิตช์เพิ่ม – ลดอุณหภูมิ

Assignment 32 Read 10 – Bit A/D and Display on LCD 43

จุดประสงค์ เพื่อเรียนรู้หลักการเขียนโปรแกรมให้ไมโครคอนโทรลเลอร์รับสัญญาณอะนาล็อก และแปลงเป็นค่าดิจิตอลขนาดความละเอียด 10 บิท เพื่อเป็นพื้นฐานการประยุกต์ใช้ในระบบแสดงผลการวัดค่าอะนาล็อกต่าง ๆ ได้แก่ อุณหภูมิ ความดัน ระดับ ฯลฯ แล้วนำค่ามาแสดงผลทางจอ LCD '

```
Define LCD registers and bits
DEFINE      LCD_DREG     PORTD
DEFINE      LCD_DBIT     4
DEFINE      LCD_RSREG    PORTE
DEFINE      LCD_RSBIT    0
DEFINE      LCD_EREG     PORTE
DEFINE      LCD_EBIT     1

' Define ADCIN parameters
DEFINE      ADC_BITS     10            ' Set number of bits in result
DEFINE      ADC_CLOCK    3             ' Set clock source (3=rc)
DEFINE      ADC_SAMPLEUS      50       ' Set sampling time in uS

adval   VAR   WORD                     ' Create adval to store result

        TRISA = %11111111              ' Set PORTA to all input
        ADCON1 = %10000010             ' Set PORTA analog and right justify result
        Low PORTE.2                    ' LCD R/W line low (W)

        Pause 500                      ' Wait .5 second

loop:   ADCIN 0, adval                 ' Read channel 0 to adval

        LCDOut $fe, 1                  ' Clear LCD
        LCDOut "Value: ", DEC adval    ' Display the decimal value

        Pause   100                    ' Wait .1 second

        GoTo    loop                   ' Do it forever
        End
```

มอบหมายงาน เมื่อศึกษาการทำงานของโปรแกรมเข้าใจดีแล้ว ให้พัฒนาโปรแกรมที่สามารถใช้สวิตช์กดตั้งค่าที่สมมุติเป็นอุณหมิ แล้วนำมาเปรียบเทียบกันค่าของ A/D ที่รับเข้ามา แล้วสั่งการให้เปิดปิดรีเลย์ควบคุมคอมเพรสเซอร์ (Portb.5) พร้อมทั้งเพิ่มเสียงคลิกขณะที่กดสวิตช์เพิ่ม – ลดอุณหภูมิในระดับค่าที่มีความละเอียดขึ้น