ElCoM
ELECTRICAL COMPUTER MECHATRONICS

تقدم لجنة ElCoM الاكاديمية

تلخيص لمختبر:

معالجات و متحكمات دقيقة

جزيل الشكر للطالبة:

سارة آبو سارة

PIC Lab. Summary

Lab(1) PIC18F452 Assembly Instructions

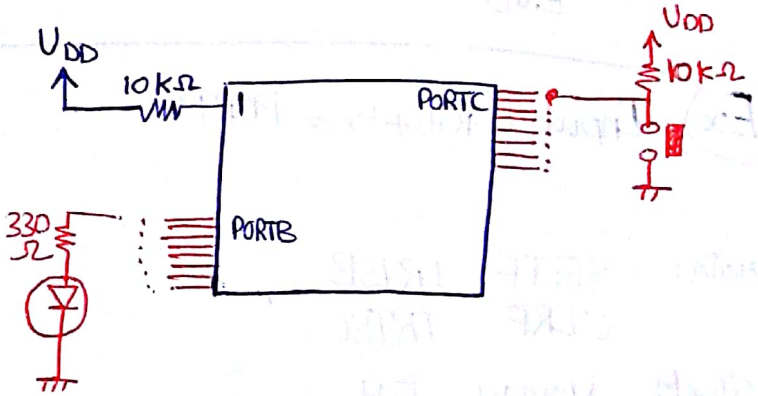
Instructions → Byte-Oriented fileReg Operation, [يتعامل مع 8 bits من ال fileReg العادي]
→ Bit-Oriented fileReg Operation, [يتعامل مع كل bit لحالها]
→ Literal Operations
→ Control Operations

خطوات كتابة برنامج

1) معرفة I/O ports
2) تحديد TRISX
3) كتابة الكود ← من ال start
4) عمل Endless Loop ← [start GOTO start]
5) END

في ال main / PORTA is output / CLRF TRISA
PORTA is input / SETF TRISA

Ex) Write a program that adding the lower 4 bits of the input data to the higher 4 bits, and moving the result to PORTB register? at PORTC




```
Lbyte   EQU   0x06
Hbyte   EQU   0x07
Main:   SETF  TRISC
        CLRF  TRISB
Start:  MOVLW 0FH
        ANDWF PORTC,W
        MOVWF Lbyte
        MOVLW 0F0H
        ANDWF PORTC,W
        MOVWF Hbyte
        ADDWF Lbyte
        MOVFF PORTC,PORTB
        GOTO  Start
        END
```

Ex) Write a program that multiply the lower 4bits of the input data at PORTC to the higher 4bits and moving the result to PORTB :-

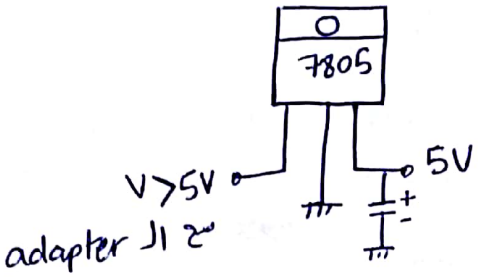
في حالة الضرب نتعامل مع (PRODH : PRODL) register

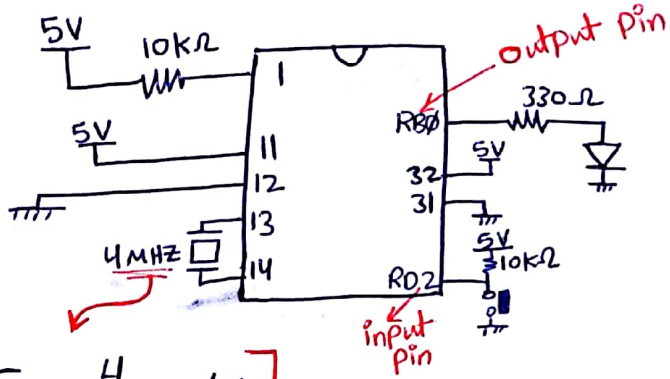
```
Lbyte   EQU   0x06
Hbyte   EQU   0x07
Main:   SETF  TRISC
        CLRF  TRISB
Start:  MOVLW  0FH
        ANDWF  PORTC,W
        MOVWF  Lbyte
        MOVLW  0F0H
        ANDWF  PORTC,W
        MOVWF  Hbyte
        SWAPF  Hbyte,W
        MULLWF Lbyte
        MOVFF  PRODL,PORTB
        GOTO   start
        END
```

Ex) Input PORTB+5 = PORTC

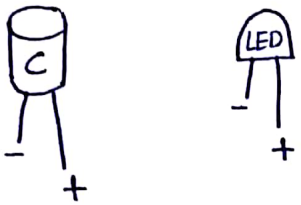
```
Main:   SETF  TRISB
        CLRF  TRISC
Start:  MOVLW  5H
        ADDWF  PORTB
        MOVFF  PORTB,PORTC
        GOTO   start
        END
```

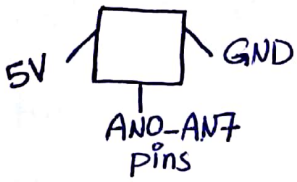
Hardware Implementation





$$T_{IC} = \frac{4}{4M} = 1\mu s$$





Lab(2) Wait Code Implementation

* كل الـ Loop ←

```
        COUNT   EQU   0x04
Main:   MOVLW   D'5'      ← قيمة العداد
        MOVWF   COUNT
start:  ________
        ________
        ________
        DECFSZ  COUNT , f
        GOTO    start     ← [خط شوبه يصير بس يخلص العدّ]
        END
```

* أكبر قيمة عدّ ممكن نخلها باللوب الواحد أعلاه هي OFFH = 255

* For more than 255 times ⇒

Ex) Load the PORTB with 55H and complement PORTB 700times?

700 = 70*10

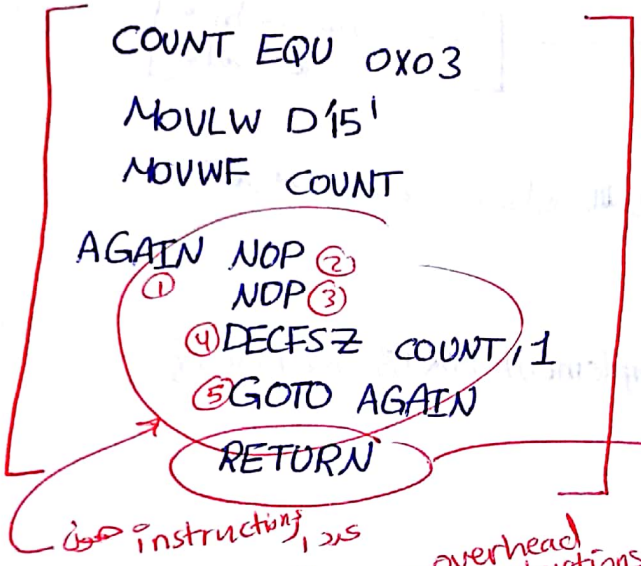
```
        R1      EQU   0x01
        R2      EQU   0X02

Main:   L1      MOVLW   D'70'
                MOVWF   R1
        L2      MOVLW   D'10'
                MOVWF   R2
Start:          MOVLW   55H
                MOVWF   PORTB
                COMF    PORTB
                DECFSZ  L2 , f
                GOTO    L2
                DECFSZ  L1 , f
                GOTO    L1
                END
```

*Delay Program :-

delay value depends on → crystal frequency (4MHz)
→ No. of instructions in the delay

① تعريف مكان خاص للكاونتر
② ادخال قيمة العداد على لى ثم لكاونتر
③ استخدام الـ delay



يرجعنا الـ instructions الى بعد (Call Delay)

عدد الـ instructions جوه

overhead instructions

$$D = [(15 * 5) + 3] * 1\mu = 78\,\mu s$$

$$D| = 15 * 5 * 1\mu = 75\,\mu s$$
approx

counter value

Ex Write a code that increasing the data at PORTB by 1 and if the PORTB value was odd then, it will wait for 1 sec. and if it's even, it will wait 2sec. to do the next increment

```
        R1      EQU     0x01
        R2      EQU     0x02
        R3      EQU     0x03
DELAY   MOVLW   D'100'
        MOVWF   R1
L2      MOVLW   D'100'
        MOVWF   R2
L3      MOVLW   D'20'
        MOVWF   R3
AGAIN   NOP
        NOP
        DECFSZ  R3,1
        GOTO    AGAIN
        DECFSZ  R2,1
        GOTO    L2
        DECFSZ  R1,1
        GOTO    DELAY
        RETURN
Main:   CLRF    TRISB
Start:  CLRF    PORTB
        INCF    PORTB
        BTFSS   PORTB,0
        GOTO    even
        CALL    DELAY
        GOTO    start
        END
```

Lab (3): ADC

التحويل من الـ analog signal إلى digital signal (Binary) أو من الـ Sampling وبحولها لـ digital (binary) , وذلك يتم خلال 3 مراحل :-

① Discretization / (Sampling Time) عملية تقطيع السيجنال إلى

② Quantitization / عملية تحويل السيجنال إلى Binary Value بتكافئ هاي السيجنال

$$RES = \frac{V^+ - V^-}{2^n} \quad [V/bit]$$

(عن طريق حساب الـ Resolution)

- n : no. of bits
- V^+ : قيمة الـ power للسنسور
- V^- : " " "

5V على أعلى

0V

(أقل analog increment) بتزيد الـ binary value ① بمقدار

تكون عملية التحويل من أي رقم إلى 10bit أو 8bit مثل ← 1V : 0000 1111

مثال: سنسور شغال على 5V,0 وكان عدد الـ bits (8) جاي قيمة الـ Res. ؟

$$Res = \frac{5-0}{2^8} = 0.01953 \ V/bit$$

يعني لحتى تزيد قيمة الـ binary الناتجة بمقدار ① لازم القيمة المدخلة عالسيجنال أكبر أو تساوي 0.01953 (من مضاعفاتها)

* للتحويل من Voltage ← binary Res. تقسم على

Res. تضرب بالـ Voltage ← binary

③ Knowing the gain sensor, and the resolution for ADC

ex temp. gain = 10 mV/C° يزداد 10 mV لكل 1°C

Res. قد من binary معمول من مراحل لفولتية وصفها لـ

In PIC, 8 pins (analog pins) → AN0 – AN7

→ ADC allows conversion of an analog input signal to a corresponding 10bit digital number.

→ ADC has 4 registers, each is 8bit :-

result ← * ADRESH, ADRESL] لعرض القيمة الي تم تحويلها 10bit وبتكون

ADCON0, ADCON1] للتحكم في وظائف ADC → Controls the operation of ADC

↳ Functions of the Port pins

[ADCON0]

7	6	5	4	3	2	1	0
ADCS1	ADCS0	CHS2	CHS1	CHS0	GO/DONE	—	ADON

- ADCS1, ADCS0: إلها علاقة بال crystal
- [Fosc/8] / 01 For 5MHZ — لأنه قيمة crystal الـ 4MHZ
- CHS2, CHS1, CHS0: إلها علاقة بال pins اللي بدي أضيء الإشارة منها — حسب الـ Channel
- GO/DONE: 0 — 0 — وبشغلها من الكود
- ADON: للتشغيل 1

```
BSF   ADCON0,2
L1 BTFSC ADCON0,2
   GOTO L1
```

* [CHS0_CHS2]

000	=	channel 0 (AN0)
001	=	channel 1 (AN1)
010	=	2 (AN2)
011	=	3 (AN3)
100	=	4 (AN4)
101	=	5 (AN5)
110	=	6 (AN6)
111	=	7 (AN7)

← نقوم بتحويل الـ channel اللي بدي قراءة السيجنال منها
أي تحويل الـ pin للوصول مع جسم

[ADCON1]

7	6	5	4	3	2	1	0
ADFM	ADCS2	—	—	PCFG3	PCFG2	PCFG1	PCFG0

- ADFM: A/D Result format select bit — لـ علاقة بكيفية عرض القيمة المحولة على ADRESH ADRESL
- ADCS2: 0 إذا كان 4MHZ
- 0
- PCFG3 PCFG2 PCFG1 PCFG0: لتحديد نوع الإشارة الداخلة على الـ pin (من الجدول الكبير) analog أو digital

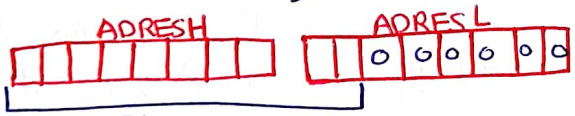
ثم تتم عملية الـ 10bits من الـ 16bits

* if bit7 = 1, right Justified

هنا يتم عرض القيمة

عندما يكون التغير صغير جداً mV كل مرة

* IF bit7 = 0, left Justified



هنا يتم عرض القيمة

لما يكون التغير كبير 1V في كل مرة

$\frac{V^+ - V^-}{2^n \rightarrow 8 bits}$ ← ADRESL أو ADRESH نتعامل مع

$\frac{V^+ - V^-}{2^{10}}$ ← مع بعضهم

Ex Write a program to turn the LED at RB0 on and the LED at RB1 off when the voltage at AN0 is greater than the voltage at AN1, and turn the LED at RB0 off and the LED at RB1 on when the voltage at AN1 is greater than the voltage at AN0 and if the 2 voltages are equal than the 2 LED's at RB0 and RB1 are off.

```
        COUNT EQU 0X02
main:   CLRF  TRISB
        CLRF  PORTB
        CLRF  ADCON1
        MOVLW B'01000001'      ← لتعريف AN0
        MOVWF ADCON0
Start:  BSF   ADCON0,2
L1:     BTFSC ADCON0,2
        GOTO  L1
        MOVFF ADRESH,COUNT
        MOVLW B'01001001'      ← لتعريف AN1
        MOVWF ADCON0
        BSF   ADCON0,2
TEST:   BTFSC ADCON0,2
        GOTO  TEST
        MOVF  ADRESH,0         ← AN1
        CPFSGT COUNT           ← AN0
        GOTO  L2
        BSF   PORTB,0
        BCF   PORTB,1
        GOTO  start
L2:     CPFSEQ COUNT
        GOTO  L4
        BCF   PORTB,0
        BCF   PORTB,1
L4:     BCF   PORTB,0
        BSF   PORTB,1
        GOTO  start
```

Ex:

```
Main :  CLRF   TRISB
        MOVLW  B'01010001'      ; AN2
        MOVWF  ADCON0
        CLRF   ADCON1
start:  BSF    ADCON0,2
TEST    BTFSC  ADCON0,2
        GOTO   TEST
        MOVLW  D'32'
        CPFSLT ADRESH
        GOTO   L1
        MOVLW  B'1'
        MOVWF  PORTB
        GOTO   start
L1      MOVLW  D'64'
        CPFSLT ADRESH
        GOTO   L2
        MOVLW  B'10'
        MOVWF  PORTB
        GOTO   start
L2      MOVLW  D'96'
        CPFSLT ADRESH
        GOTO   L3
        MOVLW  B'100'
        MOVWF  PORTB
        GOTO   start
L3      :
        :
        .
```

8 bits

$\frac{5-0}{2^8} = 0.01953$ V/bit

left Justified

(mV) تغير حسب مطلوب

$\frac{5}{8} = 0.625$ V

بدي أحوله binary (Res) يعتمد على

$\frac{0.625}{0.01953} = 32$

32 → Region 1 (x<32)

[32 ≤ x < 64] region(2)

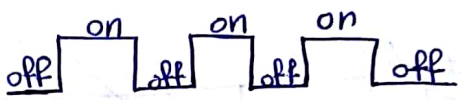
[64 ≤ x < 96] region(3)

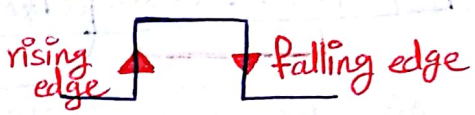
:

Lab(4): Pulse Width Modulation

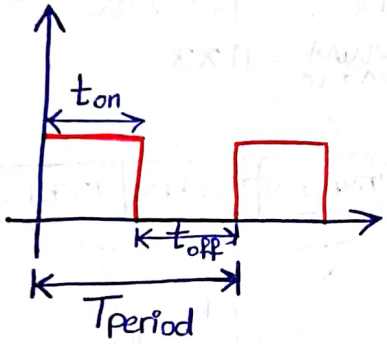
modulation ≡ تحميل [data] معينة على مكان معين للتحكم بال (Freq. / Pulse width / amplitude)

Pulse [on-off signal]





*



$$\text{Duty Cycle [D.C]} = \frac{T_{on}}{T_{period}} \%$$

اضافة مقدم حسب المطلوب في السؤال

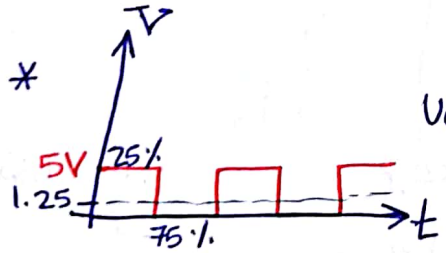
* مقدار Ton يعتمد على ال Res. أي أنه يعتمد على no. of bits ال data التي يتم تحميلها على ال Ton

[00000000 _ 11111111] ≡ [0-255] Ton يمكن التحكم بقيمة ← # of bits = 8 خطأ

[00000000 - 11111111] ≡ [0-1023]$_D$ Ton = = = ← # of bits = 10

---

* الفائدة من ال PWM ← تحويل ال digital signal إلى Analog signal يتم متغيرة ويمكن التحكم بال (amplitude) لل (analog signal) كسبب ال D.C



Voltage Amplitude (A) = Voltage * Ton
= 5 * 25%
= 1.25 V

ولتحويل الاشارة لـ (analog) نغير على Freq. ← $F = \frac{1}{T}$ ← Hz

T ← sec

T = 1 μsec.
F = 1 MHz
لتم كبيرة جدًا

في ال data sheet يعتبر ال Ton هي في D.C

① PR2 ② CCPR1L ③ T2CON ④ CCP1CON ⑤ TRISC

→ موال الـ output للـ PWM نعرّف كـ output من PORTC

BCF TRISC,2

* CCP1CON →

7	6	5	4	3	2	1	0
0	0	DCxB1	DCxB0	CCPxM3	CCPxM2	CCPxM1	CCPxM0

DCxB1, DCxB0: تأتي بها عن طريق حساب قيمة الـ D.C.

x من 2bits أول

$D.C = T_{on} = x * T_{osc} * P.S$

CCPxM3–CCPxM0: تستخدم لتشغيل خاصية الـ PWM

PWM mode = 11XX

* T2CON →

7					2	1	0
0	TOUTPS3	TOUTPS2	TOUTPS1	TOUTPS0	TMR2ON	T2CKPS1	T2CKPS0

الباقي ما مفهوم إله ←

TMR2ON → 1 لتشغيل الـ timer وعملية الـ PWM

أو منخليها zero ومنفعلها بالكود ⇐ **BSF T2CON,2**

T2CKPS1, T2CKPS0 → إلهم علاقة بالـ Prescalar Value

00 = prescalar is 1
01 = 〃 〃 4
1X = 〃 〃 16

وهذه القيم يتم تحديدها عن طريق معادلة الـ (T_{period})

$$T_{period} = (PR2+1) * 4 * (T_{osc}) * P.S$$

إذا كانت معنا قيمها لازم نوجد الـ P.S والـ PR2

PR2 → [0-255]

$T_{osc} = \frac{1}{F_{osc}} = \frac{1}{4MHz} = 0.25\mu s$

Ex) $T_{period} = 1000\mu s$, 4MHZ, find P.S and PR2?

* خطوات الحل

منفرض إنه **P.S = 1**

$1000\mu = (PR2+1) * 4 * \frac{1}{4}\mu * (1)$

PR2 = 999 No way

$1000 = (PR2+1) * 4$ → **PR2 = 249** → **P.S = 4**

* CCPR1L → "8 bit"   تحديد قيمة ال duty cycle

من معرفة قيمة x بأخذ عدد 8bits منها $D.C = T_{on}$

ويوجد 2bit أيضاً في ال CCP1CON

$$T_{on} = D.C = [CCPR1L\langle 7:0\rangle, \langle 5:4\rangle CCP1CON] * T_{osc} * P.S$$

(8bit: CCPR1L ⟨7:0⟩ ; 2bit: ⟨5:4⟩ CCP1CON ; x نعرف ; $T_{osc} = \frac{1}{4MHz}$; P.S من معادلة ال T_{period})

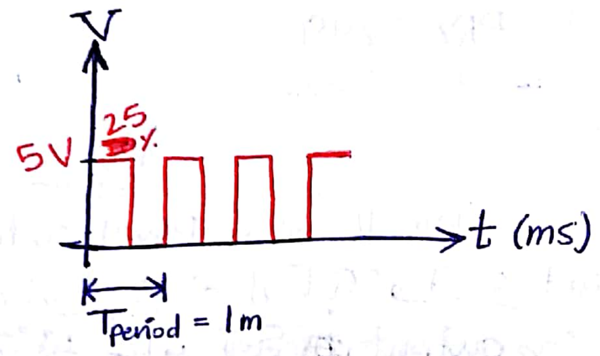
Ex If $T_{on} = 200\mu s$, find CCPR1L and CCP1CON and P.S = 4 ?

$$200\mu = x * 4 * \frac{1}{4MHz}$$

$x = 200$ → نحول إلى binary → 0011001000

- 00110010 → CCPR1L (8 bits)
- 00 → CCP1CON ⟨5:4⟩

Ex



$$T_{on} = D.C = x * 4 * \frac{1}{4MHz}$$

$$25\% \; ms = x * 1\mu s$$

$$250\mu = x * 1\mu$$

250	
125	0
62	1
31	0
15	1
7	1
3	1
1	1
0	1

$x = 250$ → 0011111010

- 00111110 → CCPR1L
- 10 → CCP1CON

$$T_{period} = (PR2+1) * 4 * \frac{1}{4MHz} * P.S$$

$$1000\mu = (PR2+1) * \frac{1\mu}{4} * 4 * 4$$

$PR2 = 249$

CCP1CON → 00101100 (قيم ال 250 ، ثابت)

CCPR1L → 00111110 → قيم وأخر 8 bits بال 250

خطوات كتابة الكود :-

① تعريف BCF TRISC,2

② حل المعادلات Tperiod و D.C لإيجاد قيمة CCPR1L, PR2, CCP1CON, P.S

③ نقل قيمة CCP1CON ← 1111(--)00 لاحظ الـ 2bits من

④ " " CCPR1L ← داخل 8bits من x عشر منازل Ton

⑤ " " PR2 , اللي طلعتها معنا من قانون Tperiod

⑥ تفعيل T2CON , حسب الـ (prescalar value) اللي اعتمدناها

D.C ثابت

```
main:  BCF    TRISC,2
       MOVLW  B'00101111'
       MOVWF  CCP1CON
       MOVLW  B'00111110'
       MOVWF  CCPR1L
       MOVLW  D'249'
       MOVWF  PR2
       MOVLW  B'101'
       MOVWF  T2CON
start: GOTO   start      ← ممكن أعطي برنامج
       END
```

if P.S = 4 = 01

then T2CON = 101 → 5

CCP1CON = 00101111

CCPR1L = 0011 1110

PR2 = 249

ملاحظة

PWM code موجود بالـ main

عادي يعطي فرصة لكتابة كود آخر في start

لتنفيذ برنامج تحكم بـ output معين يستخدم PWM

D.C متغير

```
main:  BCF    TRISC,2
       MOVLW  B'00001111'
       MOVWF  CCP1CON
       MOVLW  B'11111101'
       MOVWF  CCPR1L
       MOVLW  D'249'
       MOVWF  PR2
       MOVLW  B'101'
       MOVWF  T2CON
Start: Call   delay
       MOVLW  B'101111'
       MOVWF  CCP1CON
       MOVLW  B'111110'
       MOVWF  CCPR1L
       GOTO   start
       END
```

$T_{on} = 500\mu s$

$T_{on} = 500\mu$

$500\mu = x * \frac{1}{4MHz} * 4$

$x = 500$

$= 11111010\underline{00}$

$T_{on} = 250\mu s$

$x = 250$

$= 11111010$

* سنبلغ LED على RC2 بحيث تزداد الفولتية من 0 إلى 5 ثم تعود من 5 إلى 0

* مثال 5sec.

$5 = n * 10^{-6} * 5$

$n = 1000000$

$= 100 * 100 * 100$

$255 + 255 = 510$

$\frac{5}{510 * 10^{-6} * 5} = 1961 \rightarrow 109$*

* بدنا نتحكم بال D.C وبدنا نوصله Tperiod و P.S

$T_{period} = (PR2+1) * 4 * T_{osc} * P.S$

$255\mu = (PR2+1) * 1\mu * 1$ (فرض) (فرض)

$PR2 = 254$

$T_{on} = x * 4 * \frac{1}{4MHz}$

```
DELAY : MOVLW  D'1**'
        MOVWF  R1
L2 :    MOVLW  D'39'
        MOVWF  R2
L1 :    MOVLW  D'10*'
        MOVWF  R3
L0 :    NOP
        NOP
        DECFSZ R3,1
        GOTO   L0
        DECFSZ R2,1
        GOTO   L1
        DECFSZ R1,1
        GOTO   DELAY
        RETURN

Main:   CLRF   TRISC
        CLRF   PORTC
        MOVLW  B'100'
        MOVWF  T2CON
        MOVLW  D'254'
        MOVWF  PR2
        MOVLW  B'00001111'
        MOVWF  CCP1CON
        CLRF   CCPR1L
start:  MOVLW  D'254'
        CPFSGT CCPR1L
        GOTO   L0
        GOTO   L1
L0      CALL   DELAY
        INCF   CCPR1L
        GOTO   start
L1      DECF   CCPR1L
        Call   DELAY
        MOVLW  D'0'
        CPFSEQ CCPR1L
        GOTO   L1
        GOTO   start
        END
```

Lab Sheet:: وبناءً على قيمتها يعدّل AO بدنا نستقبل بوتنشومتر على قوة الـ pulse

```
Main:  CLRF   TRISC        ← out. الـ PWM
       MOVLW  B'11111001'
       MOVWF  PR2          ← نضعزم بالأول
       MOVLW  B'00001100'
       MOVWF  CCP1CON
       MOVLW  B'101'
       MOVWF  T2CON
       MOVLW  B'01000001'  ← AND
       MOVWF  ADCON0

       CLRF   ADCON1
Start: BSF    ADCON0,2     ← دائماً منخطها
TEST   BTFSC  ADCON0,2
       GOTO   TEST
       MOVFF  ADRESH,CCPR1L   ← نقلنا القيمة اللي تحولت لـ CCPR1L عند digital
       BTFSC  ADRESL,7
       GOTO   L1
       BCF    CCP1CON,5
       GOTO   L2
L1:    BSF    CCP1CON,5
L2:    BTFSC  ADRESL,6
       GOTO   L3
       BCF    CCP1CON,4
       GOTO   L4
L3:    BSF    CCP1CON,4
L4:    GOTO   start
       END
```

$$* T_{period} = (PR2+1) * 4 * \frac{1}{4MHz} * 4 \quad \text{(P.S)}$$

$$1000\,\mu = (PR2+1) * 4$$

$$PR2 = 249 = 11111001$$

$$* T_{on} = (x) * 4 * \frac{1}{4MHz}$$

← متغيرة حسب اللي بدخل من الأنالوج فأول شي نضعزها

* Interrupts : هي الـ Subroutine الذي يتم استدعاء بـ Call لشيء معين.
لكن الفرق أن الـ interrupts طبيعتها Call وتحدث فجأة عند تفعيل

one type of it → External Interrupt (RB0)

```
BSF   TRISB , 0
```

مميزاتها: ① يتم تفعيلها عن طريق الـ hardware بدون call داخل البرنامج
② يستخدم لتشغيل 2 events مع بعض في نفس البرنامج
③ مختصر في البرنامج

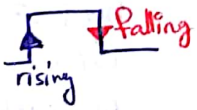
← مبدأ عملها: يتم تفعيل الـ interrupts اذا حدث HW event معين حيث يقوم بتوقيف البرنامج بغض النظر عن مكان تنفيذ البرنامج الـ " ثم يعود إلى البرنامج الأصلي ويكمل من مكان التوقف.

** External Interrupts (RB0) Registers :

① INTCON ↓ 3 bits من بتات 7 4 1
② INTCON2 ↓ bit 6 من بتات
③ RCON ↓ bit 7 من بتات

1 INTCON : ① 7 → GIE/GIEH : Global Interrupt Enable bit / وصول عن تشغيل كل الـ interrupts
① 4 → INT0E : External Interrupt " = " = /RB0, INT وصول عن تشغيل نوع معين من
⓪ 1 → INT0IF : " = " = Flag = / interrupt يعطي إشارة على حدوث الـ
subroutine داخل ① حيث لا يصير ⓪ ولا بد من CLRF بعد

2 INTCON2 : 6 → INTEDG1 : External Interrupt to edges
1 = Interrupt rising edge (0→1)
0 = " falling " (1→0)



3 RCON : 7 → IPEN : Interrupt priority Enable bit / ⓪ اذا خليناها 1 معناته يمكن تقسيم جميع الـ interrupt الخاصة بال low & high الأولوية

```
BCF   RCON , 7
```

Ex برنامج يقوم بعمل زيادة على قيمة PORTC اذا تم الضغط على RB0 (RB0 input, PORTC output)

```
HighInt: BTFSS  INTCON , 1
         GOTO   L1
         INCF   PORTC , 1
         BCF    INTCON , 1
L1 :     retfie fast
```

(HighInt موجود في main)

```
main:  BCF    RCON , 7
       MOVLW  B'10010000'
       MOVWF  INTCON
       BCF    INTCON2 , 6      falling pull up
       CLRF   PORTC
       CLRF   TRISC
       SETF   TRISB
Start: GOTO   Start             لا نهاية لها
       END
```

Ex

برنامج تقوم بعدّ الكبسات التي تم ضغطها على RBØ كل 3sec. وعرضها على PORTC ؟

$3 = 5 * 1\mu * x$

$x = 600,000$

```
HighInt:
        BTFSS   INTCON,1
        GOTO    T1
        INCF    Ali
        BCF     INTCON,1
T1 :    retfie  fast
```

فوق ال instruction

بدل [BCF INTCON,1]

```
delay:  MOVLW   D'100'
        MOVWF   R1
L2      MOVLW   D'100'
        MOVWF   R2
L1      MOVLW   D'60'
        MOVWF   R3
AGAIN   NOP
        NOP
        DECFSZ  R3,1
        GOTO    AGAIN
        DECFSZ  R2,1
        GOTO    L1
        DECFSZ  R1,1
        GOTO    L2
        RETURN

R1      EQU     0x01
R2      "       0x02
R3      "       0x03

main:   SETF    TRISB
        CLRF    PORTC
        CLRF    TRISC
        MOVLW   B'10010000'     ; H'90'
        MOVWF   INTCON
        BCF     INTCON2,6
        BCF     RCON,7

Start:  call    delay
        MOVFF   Ali, PORTC
        CLRF    Ali
        GOTO    Start
        END
```

فوق

Interrupts

* جو ال main :-

```
BCF     RCON,7
MOVLW   B'10010000'
MOVWF   INTCON
BCF     INTCON2,6   → falling
SETF    TRISB           pull up
```

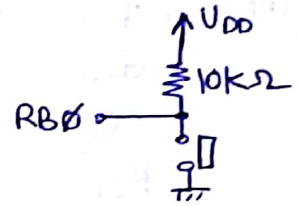
* قبل ال main :

```
HighInt:
        BTFSS   INTCON,1
        GOTO    L1

        BCF     INTCON,1
L1 :    retfie  fast
```

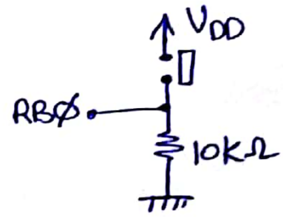
* Pull up (Falling) → [BCF INTCON2,6].

دائماً بتعطي 1 ، إذا كبسنا بتعطي 0



* Pull down (rising) → [BSF INTCON2,6]

دائماً صفر إلا إذا كبسنا الكبسة بتعطي 1



سنوات فاينل

Q1

05 → ~~GOTO~~ DECF COUNT1,1

11 → CLRF TRISC → PWM الـ (output) الـ

16 → MOVWF INTCON , [H'90' = B'10010000']

19 → MOVWF T2CON , [H'07' = B'111' ⇒ (P.S ← 11 قبل الـ)]

23 → MOVWF CCP1CON [H'0F' = B'0000 1111']

32 → MOVWF PORTD

PR2 = 249

Q2

(P2) Program and build a CCT to change the frequency of the pulses generated of PWM module from 1.25 kHZ to 2.5 kHZ using Push buttons at RB1 to increase the frequency, and RB0 to decrease it, with constant duty cycle of 25% ?

Crystal freq. = 4MHZ

* T_{period} → 800µ → 400µ (1/1.25 KHZ ... 1/2.5 KHZ)

25%

* $250\mu = 4 * \frac{1}{4MHZ} * X \rightarrow X = 250$

$= B'00111110'$ (10)

CCPR1L

```
main :  CLRF   TRISC   ← output لأن PWM
        SETF   TRISB
        MOVLW  B'00101111'
        MOVWF  CCP1CON
        MOVLW  B'00111110'
        MOVWF  CCPR1L
        MOVLW  B'101'
        MOVWF  T2CON
```

لأن duty cycle

```
start:  BTFSS  PORTB,1  ← RB1
          GOTO  L1
        MOVLW  B'01100011'
        MOVWF  PR2
L1      BTFSS  PORTB,0  ← RB0
          GOTO  start
        MOVLW  B'11000111'
        MOVWF  PR2
        GOTO   start
        END
```

* at $T_p = 800\mu s$:

$800\mu = (PR2+1) * 4 * \frac{1}{4MHZ} * P.S$ ← 4

if P.S = 4 → PR2 = 199 = B'11000111'

199	
99	1
49	1
24	1
12	0
6	0
3	0
1	1
0	1

T2CON [10]

* at $T_p = 400\mu$:

$400 = (PR2+1) * 4$

PR2 = 99 = B' 1100011'

99	
49	1
24	1
12	0
6	0
3	0
1	1
0	1

T2CON [10]

$$T_{period} = \frac{1}{f}$$

Summary

* ADC : [Analog to digital converter]

ADCON0 → '01---001' → [AN0_AN7]

ADCON1 → 'I000----'

→ Left Justified = 0 / ADRESH, 1V تعبر كبير
right Justified = 1 / ADRESL, تعبر صغير mV بال

---

* PWM : [Pulse Width Modulation]

main:

T2CON → [100, 101, 111] ← if P.S = 16 (P.S=1, P.S=4)

CCP1CON, CCPR1L → [T_{on} = D.C (ms) = 4 * T_{osc} * X] → مثلاً X = 00100111 01

P.S = 1, 4, 16

CCPR1L: 00100111

CCP1CON: 00 11 1111

PR2 → [T_{period} = (PR2+1) * 4 * T_{osc} * P.S]

CLRF TRISC → PWM output, (pin 2) كتبنا

PORTC : PWM output

---

* Interrupts :

INTCON [10010000] → Bit 1

INTCON 2 ↔ rising, falling

RCON

High Int:

```
BTFSS  INTCON, 1
GOTO   L1
       BCF  INTCON, 1
L1:    retfie fast
```

PORTB : Interrupt Input

main

main:

BSF TRISB, 0 ← أو [SETF TRISB]

كتبنا قيم كلاً من ⇐

INTCON [1001 0000]

BCF RCON, 7

BCF INTCON2, 6 ← (pull up) falling

إذا كتبناه بصفر =

Start:

GOTO Start

END