lab 2

1. FSR usage
2. bubble sort
3. binary search

Some SFRs related to pointer FSRx (x for 0~2)

- INDFx：指針不變，對指向的記憶體位置進行操作
- POSTINCx：對指向的記憶體位置進行操作後，指針 + 1
- POSTDECx：對指向的記憶體位置進行操作後，指針 - 1
- PREINCx：指針 + 1 後，對指向的記憶體位置進行操作
- PLUSWx：指針 + WREG = 新的記憶體位置後，對指向的記憶體位置進行操作
  Sample code:

```
#INCLUDE <p18f4520.inc>
 CONFIG OSC = INTIO67
 CONFIG WDT = OFF
 org 0x00 ;PC = 0x00
setup1:
 LFSR 0, 0x000 ; FSR0 point to 0x000
 LFSR 1, 0x010 ; FSR1 point to 0x010
 LFSR 2, 0x020 ; FSR2 point to 0x020
 MOVLW 0x10 ; WREG = 0x10
start:
 INCF POSTINC0
 ; [0x000] += 1; FSR0 point to 0x001

 INCF PREINC1
 ; FSR1 point to 0x011 ;[0x011] += 1

 INCF POSTDEC2
 ; [0x020] += 1 ; FSR2 point to 0x01F

 INCF INDF2
 ; [0x01F] += 1 ;
 ; FSR2 point to 0x01F(unchanged)

 INCF PLUSW2
 ; [0x01F+0x10] += 1
 ; FSR2 point to 0x01F(unchanged)
end
```

BSR

```
#INCLUDE <p18f4520.inc>
        CONFIG OSC = INTIO67
        CONFIG WDT = OFF
        org 0x10 ;PC = 0x10
start:
        MOVLW 0x99 ; WREG = 0x99
        MOVLB 0x4 ; BSR = 4
        MOVWF 0x10, 1 ; use BSR select bank ; [0x410] = 0x99
end
```

Lab3

1.shift

2.multiple 16 bit*16 bit

3.log2(x)

REGISTER 5-2: STATUS REGISTER

U-0	U-0	U-0	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
—	—	—	N	OV	Z	DC(1)	C(2)
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7-5 **Unimplemented:** Read as '0'

bit 4 **N:** Negative bit
This bit is used for signed arithmetic (2's complement). It indicates whether the result was negative (ALU MSB = 1).
1 = Result was negative
0 = Result was positive

bit 3 **OV:** Overflow bit
This bit is used for signed arithmetic (2's complement). It indicates an overflow of the 7-bit magnitude which causes the sign bit (bit 7) to change state.
1 = Overflow occurred for signed arithmetic (in this arithmetic operation)
0 = No overflow occurred

bit 2 **Z:** Zero bit
1 = The result of an arithmetic or logic operation is zero
0 = The result of an arithmetic or logic operation is not zero

bit 1 **DC:** Digit Carry/$\overline{\text{borrow}}$ bit(1)
For ADDWF, ADDLW, SUBLW and SUBWF instructions:
1 = A carry-out from the 4th low-order bit of the result occurred
0 = No carry-out from the 4th low-order bit of the result

bit 0 **C:** Carry/$\overline{\text{borrow}}$ bit(2)
For ADDWF, ADDLW, SUBLW and SUBWF instructions:
1 = A carry-out from the Most Significant bit of the result occurred
0 = No carry-out from the Most Significant bit of the result occurred

Lab4 macro & subroutine

1. macro
2. cross
3. fib(iter)

   Macro 裡面不能有label,會redefine,要goto 的話只能用sub routine

   除非這樣：

```
DELAY macro num1, num2
    local LOOP1         ; Inner loop
    local LOOP2         ; Outer loop

    ; 2 cycles
    MOVLW num2          ; Load num2 into WREG
    MOVWF L2            ; Store WREG value into L2

    ; Total_cycles for LOOP2 = 2 cycles
    LOOP2:
    MOVLW num1
    MOVWF L1

    ; Total_cycles for LOOP1 = 8 cycles
    LOOP1:
    NOP                 ; busy waiting
    NOP
    NOP
    NOP
    NOP
    DECFSZ L1, 1
    BRA LOOP1           ; BRA instruction spends 2 cycles

    ; 3 cycles
    DECFSZ L2, 1        ; Decrement L2, skip if zero
    BRA LOOP2
endm
```

Lab5: mixing with c

1. sqrt(x)
2. gcd
3. signed mul, (no `mul`)
   int 的大小為 16 bit，char 的大小則為 8 bit
   return type 1Byte 以內是回傳Wreg(並且會同時將Wreg內的值也放進0x001)

Function Parameters

如果第一個 parameter大小在1Byte 以內是放入Wreg, 其他parameters 則是根據type依序放入0x001,0x002

第一個 parameter超過1Byte的話,則是根據type大小放入0x001 (lb) 和:0x002 (hb) ..., 其他parameters就接著後面放入

return

return type 超過1Byte 是回傳到0x001 (lb) 和:0x002 (hb) ...，依據你的return type的大小，並且遵循XC8所使用之little endian存放方式。

lab06

燈泡:長+短-

delay:

instruction frequency = 1 MHz / 4 = 0.25 MHz

instruction time = 1/0.25 = 4 μs

Total_cycles = 2 + (2 + 8 * num1 + 3) * num2 cycles

num1 = 111, num2 = 70, Total_cycles = 62512 cycles

Total_delay ~= Total_cycles * instruction time = 0.25 s

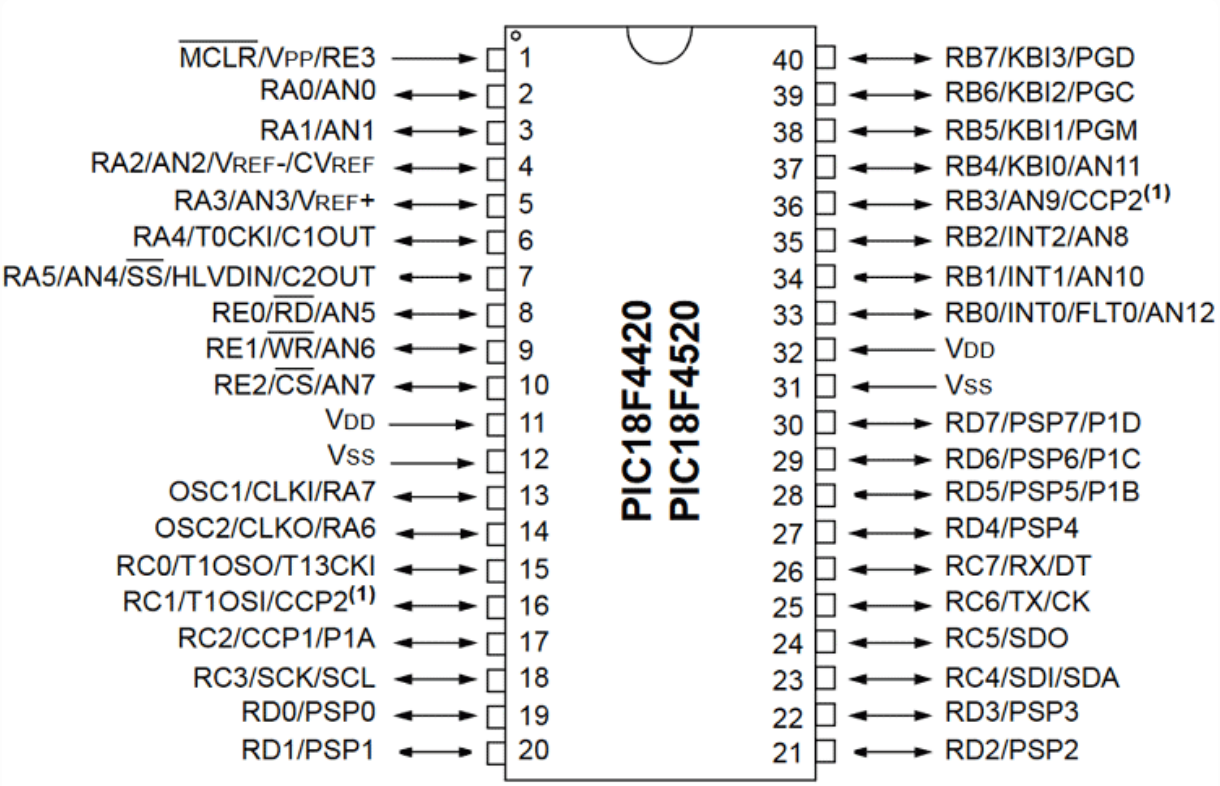
```
TRISA: set 0 as out
TRISB: set bit 0 as in

LATA: 對應TRISA

check_process:
;    BTFSC PORTB, 0      ; Check if PORTB bit 0 is low (button pressed)
;       BRA check_process    ; If button is not pressed, branch back to checl
;    BRA lightup         ; If button is pressed, branch to lightup
```

lab07

```
; 離開ISR，回到原本程式執行的位址，同時會將GIE設為1，允許之後的interrupt能夠觸發
RETFIE
```



Enable: 看此interupt 有無開，有才執行

> 不能用MOVFF改interuput control register

Interrupt用

Register名稱	在第幾頁	用途
RCON	第44頁	IPEN: 設定 Interrupt 優先度
INTCON	第95頁	GIE、INT0 的 [Flag bit, Enable Bit]
ADCON1	第226頁	設定數位類比

sample:

```
#include "p18f4520.inc"

; CONFIG1H
  CONFIG  OSC = INTIO67         ; Oscillator Selection bits (Internal oscillator
  CONFIG  FCMEN = OFF           ; Fail-Safe Clock Monitor Enable bit (Fail-Safe (
  CONFIG  IESO = OFF            ; Internal/External Oscillator Switchover bit (Os

; CONFIG2L
  CONFIG  PWRT = OFF            ; Power-up Timer Enable bit (PWRT disabled)
  CONFIG  BOREN = SBORDIS       ; Brown-out Reset Enable bits (Brown-out Reset en
  CONFIG  BORV = 3              ; Brown Out Reset Voltage bits (Minimum setting)
```

```
; CONFIG2H
  CONFIG  WDT = OFF             ; Watchdog Timer Enable bit (WDT disabled (contr
  CONFIG  WDTPS = 32768         ; Watchdog Timer Postscale Select bits (1:32768)

; CONFIG3H
  CONFIG  CCP2MX = PORTC        ; CCP2 MUX bit (CCP2 input/output is multiplexed
  CONFIG  PBADEN = ON           ; PORTB A/D Enable bit (PORTB<4:0> pins are conf
  CONFIG  LPT1OSC = OFF         ; Low-Power Timer1 Oscillator Enable bit (Timer1
  CONFIG  MCLRE = ON            ; MCLR Pin Enable bit (MCLR pin enabled; RE3 inp

; CONFIG4L
  CONFIG  STVREN = ON           ; Stack Full/Underflow Reset Enable bit (Stack f
  CONFIG  LVP = OFF             ; Single-Supply ICSP Enable bit (Single-Supply I
  CONFIG  XINST = OFF           ; Extended Instruction Set Enable bit (Instructi

; CONFIG5L
  CONFIG  CP0 = OFF             ; Code Protection bit (Block 0 (000800-001FFFh) n
  CONFIG  CP1 = OFF             ; Code Protection bit (Block 1 (002000-003FFFh) n
  CONFIG  CP2 = OFF             ; Code Protection bit (Block 2 (004000-005FFFh) n
  CONFIG  CP3 = OFF             ; Code Protection bit (Block 3 (006000-007FFFh) n

; CONFIG5H
  CONFIG  CPB = OFF             ; Boot Block Code Protection bit (Boot block (00
  CONFIG  CPD = OFF             ; Data EEPROM Code Protection bit (Data EEPROM n

; CONFIG6L
  CONFIG  WRT0 = OFF            ; Write Protection bit (Block 0 (000800-001FFFh)
  CONFIG  WRT1 = OFF            ; Write Protection bit (Block 1 (002000-003FFFh)
  CONFIG  WRT2 = OFF            ; Write Protection bit (Block 2 (004000-005FFFh)
  CONFIG  WRT3 = OFF            ; Write Protection bit (Block 3 (006000-007FFFh)

; CONFIG6H
  CONFIG  WRTC = OFF            ; Configuration Register Write Protection bit (C
  CONFIG  WRTB = OFF            ; Boot Block Write Protection bit (Boot block (0
  CONFIG  WRTD = OFF            ; Data EEPROM Write Protection bit (Data EEPROM n

; CONFIG7L
  CONFIG  EBTR0 = OFF           ; Table Read Protection bit (Block 0 (000800-001
  CONFIG  EBTR1 = OFF           ; Table Read Protection bit (Block 1 (002000-003
  CONFIG  EBTR2 = OFF           ; Table Read Protection bit (Block 2 (004000-005
  CONFIG  EBTR3 = OFF           ; Table Read Protection bit (Block 3 (006000-007

; CONFIG7H
  CONFIG  EBTRB = OFF           ; Boot Block Table Read Protection bit (Boot blo

    L1 EQU 0x14
    L2 EQU 0x15
    org 0x00

DELAY macro num1, num2
    local LOOP1
    local LOOP2
    MOVLW num2
    MOVWF L2
    LOOP2:
        MOVLW num1
        MOVWF L1
    LOOP1:
```

```
        NOP
        NOP
        NOP
        NOP
        NOP
        NOP
        DECFSZ L1, 1
        BRA LOOP1
        DECFSZ L2, 1
        BRA LOOP2
endm

; 程式邏輯：會一直卡在main裡面做無限迴圈，按下RB0的按鈕後會觸發interrupt，跳到ISR執行
; ISR裡的內容會亮起所有在RA上的燈泡，Delay約0.5秒後熄滅。

goto Initial                    ; 避免程式一開始就會執行到ISR這一段，要跳過。
ISR:                            ; Interrupt發生時，會跳到這裡執行。
    org 0x08
    BSF LATA, 2
    BSF LATA, 0
    DELAY  d'350' , d'180'      ; 約500_000cycles數， 在1MHz的情況下大約會Delay0.5秒
    CLRF LATA
    BCF INTCON, INT0IF
    RETFIE                     ; 離開ISR，回到原本程式執行的位址，同時會將GIE設為1，允許之後


Initial:                                ; 初始化的相關設定
    MOVLW 0x0F
    MOVWF ADCON1                ; 設定成要用數位的方式，Digitial I/O
    a
    CLRF TRISA
    CLRF LATA
    BSF TRISB,  0
    BCF RCON, IPEN
    BCF INTCON, INT0IF          ; 先將Interrupt flag bit清空
    BSF INTCON, GIE             ; 將Global interrupt enable bit打開
    BSF INTCON, INT0IE          ; 將interrupt0 enable bit 打開 (INT0與RB0 pin腳位置相

main:
    bra main
end
```

Timer用

```
#include "p18f4520.inc"

; CONFIG1H
  CONFIG  OSC = INTIO67         ; Oscillator Selection bits (Internal oscil
  CONFIG  FCMEN = OFF           ; Fail-Safe Clock Monitor Enable bit (Fail-
  CONFIG  IESO = OFF            ; Internal/External Oscillator Switchover b

; CONFIG2L
  CONFIG  PWRT = OFF            ; Power-up Timer Enable bit (PWRT disabled)
  CONFIG  BOREN = SBORDIS       ; Brown-out Reset Enable bits (Brown-out Re
  CONFIG  BORV = 3              ; Brown Out Reset Voltage bits (Minimum set

; CONFIG2H
  CONFIG  WDT = OFF             ; Watchdog Timer Enable bit (WDT disabled (
  CONFIG  WDTPS = 32768         ; Watchdog Timer Postscale Select bits (1:3

; CONFIG3H
  CONFIG  CCP2MX = PORTC        ; CCP2 MUX bit (CCP2 input/output is multip
  CONFIG  PBADEN = ON           ; PORTB A/D Enable bit (PORTB<4:0> pins are
  CONFIG  LPT1OSC = OFF         ; Low-Power Timer1 Oscillator Enable bit (T
  CONFIG  MCLRE = ON            ; MCLR Pin Enable bit (MCLR pin enabled; RE

; CONFIG4L
  CONFIG  STVREN = ON           ; Stack Full/Underflow Reset Enable bit (St
  CONFIG  LVP = OFF             ; Single-Supply ICSP Enable bit (Single-Sup
  CONFIG  XINST = OFF           ; Extended Instruction Set Enable bit (Inst

; CONFIG5L
  CONFIG  CP0 = OFF             ; Code Protection bit (Block 0 (000800-001F
  CONFIG  CP1 = OFF             ; Code Protection bit (Block 1 (002000-003F
  CONFIG  CP2 = OFF             ; Code Protection bit (Block 2 (004000-005F
  CONFIG  CP3 = OFF             ; Code Protection bit (Block 3 (006000-007F

; CONFIG5H
  CONFIG  CPB = OFF             ; Boot Block Code Protection bit (Boot bloc
  CONFIG  CPD = OFF             ; Data EEPROM Code Protection bit (Data EEP

; CONFIG6L
  CONFIG  WRT0 = OFF            ; Write Protection bit (Block 0 (000800-001
  CONFIG  WRT1 = OFF            ; Write Protection bit (Block 1 (002000-003
  CONFIG  WRT2 = OFF            ; Write Protection bit (Block 2 (004000-005
  CONFIG  WRT3 = OFF            ; Write Protection bit (Block 3 (006000-007

; CONFIG6H
  CONFIG  WRTC = OFF            ; Configuration Register Write Protection b
  CONFIG  WRTB = OFF            ; Boot Block Write Protection bit (Boot blo
  CONFIG  WRTD = OFF            ; Data EEPROM Write Protection bit (Data EE

; CONFIG7L
  CONFIG  EBTR0 = OFF           ; Table Read Protection bit (Block 0 (00080
  CONFIG  EBTR1 = OFF           ; Table Read Protection bit (Block 1 (00200
  CONFIG  EBTR2 = OFF           ; Table Read Protection bit (Block 2 (00400
  CONFIG  EBTR3 = OFF           ; Table Read Protection bit (Block 3 (00600

; CONFIG7H
  CONFIG  EBTRB = OFF           ; Boot Block Table Read Protection bit (Boo

    org 0x00

```

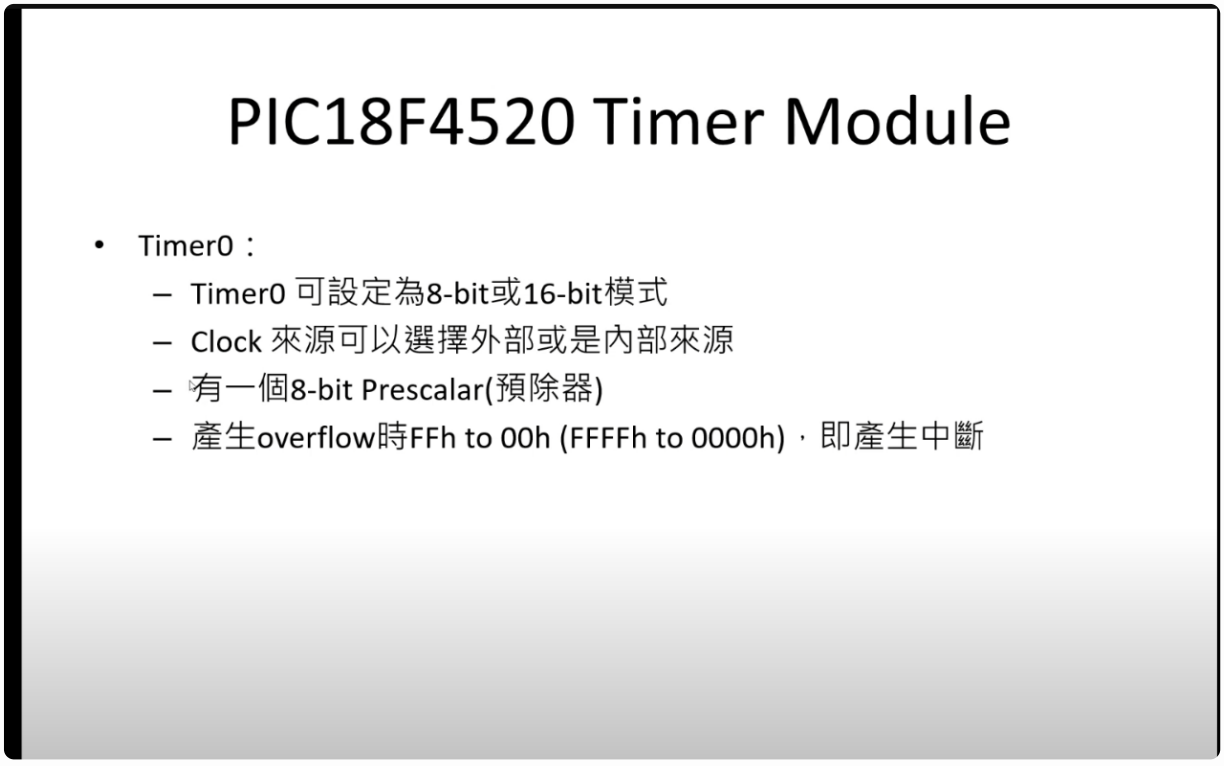
```
goto Initial
ISR:
    org 0x08                ; 大致效果: 每0.5秒會進入一次interrupt
    COMF LATA               ; interrupt會開關LATA一次
    BCF PIR1, TMR2IF        ; 離開前記得把TMR2IF清空 (清空flag bit)
    RETFIE

Initial:
    MOVLW 0x0F
    MOVWF ADCON1
    CLRF TRISA
    CLRF LATA
    BSF RCON, IPEN
    BSF INTCON, GIE
    ; 為了使用TIMER2，所以要設定好相關的TMR2IF、TMR2IE、TMR2IP。
    BCF PIR1, TMR2IF
    BSF IPR1, TMR2IP
    BSF PIE1 , TMR2IE       ;enable tmr2
    MOVLW b'11111111'          ; 將Prescale與Postscale都設為1:16，意思是之後每25
    MOVWF T2CON                ; 而由於TIMER本身會是以系統時脈/4所得到的時脈為主
    MOVLW D'122'               ; 因此每256 * 4 = 1024個cycles才會將TIMER2 + 1
    MOVWF PR2                  ; 若目前時脈為250khz，想要Delay 0.5秒的話，代表每經
                               ; 因此PR2應設為 125000 / 1024 = 122.0703125， 約
    MOVLW D'00100000'
    MOVWF OSCCON               ; 記得將系統時脈調整成250kHz

main:
    bra main


end
```

timer2好處:別人都只能用prescaler除玩overflow才interrupt，他可以設定值，when couner == 就interrupt



Register名稱	在第幾頁	用途
OSCCON	第32頁	調整時脈 (可以玩看看)
T0CON	第125頁	設定Timer0的啟動、預除器後除器
T2CON	第135頁	設定Timer2的啟動、預除器後除器
PIR1	第98頁	TMR2IF、TMR1IF等
PIE1	第100頁	TMR2IE、TMR1IE等
IPR1	第102頁	TMR2IP、TMR1IP等的priority
RCON	44	IPEN設定優先度enable

prescaler

1:8 代表8個clock 才會increase 一次timer

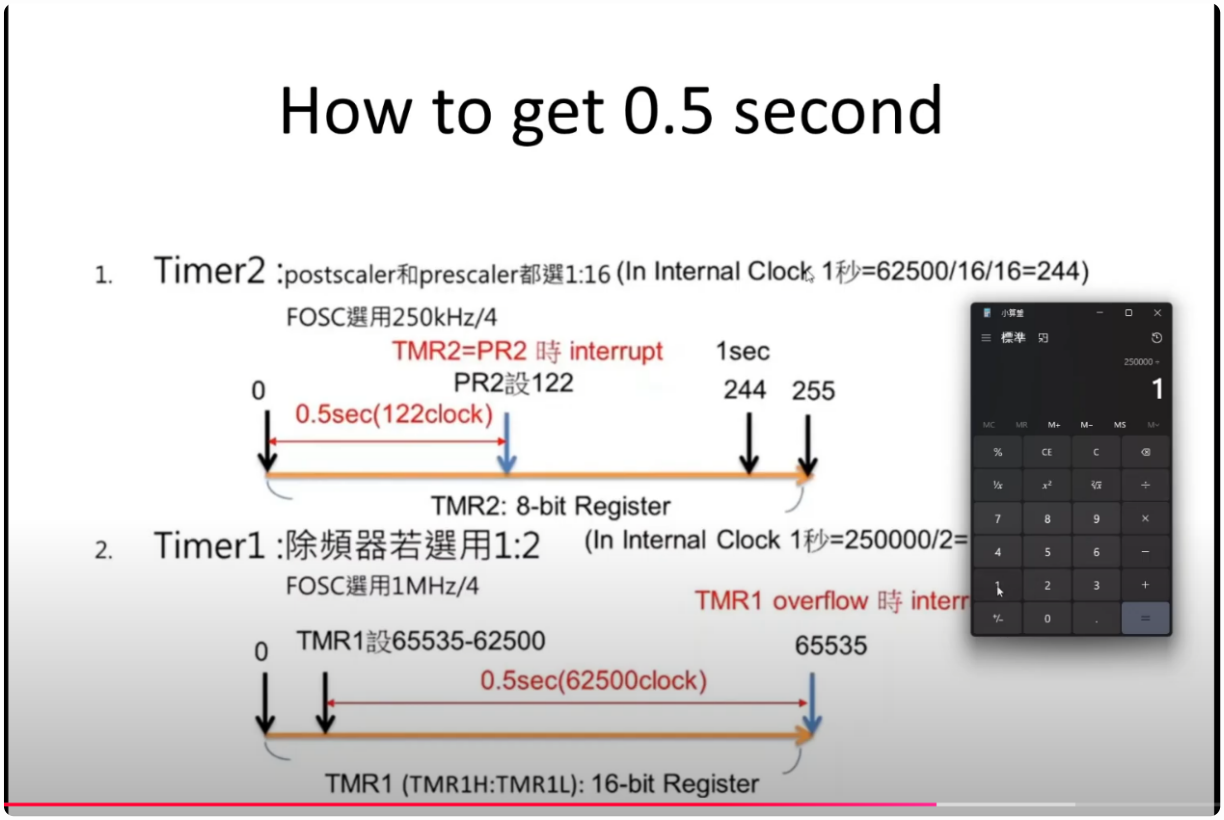
example

how to get 0.5s?

- Timer2:
  - assume 250khz，post、prescaler = 1:16
  - PR2 = 250k/4/16/16 = 244(to get 1 s)
  - PR2 = 122 to get 0.5s

> 在選FOSC(時脈)if6 選太大，會無法用pr2表示，因為只有8bit會無法表示

- Timer1:
  - assume 1Mhz，prescaler = 1:2
  - 1M/4/2 = 125000(1s)
  - 62500 to get 0.5s
  - no pr1 so select 起始點 TMR1= 65535-62500 = 3035



> 在設定timer時長(PR2)改變，要先暫停 `BCF PIE1 , TMR2IE` 再改，然後再啟用BCF PIE1 , TMR2IE
>
> 頻率62可以61不行? 操

Lab 08 ccp馬達

接線

咖啡ground
橘色 vout

reference

Each Capture/Compare/PWM module is associated with a control register (generically, CCPxCON) and a data register (CCPRx).

CCP/ECCP Mode	Timer Resource
Capture	Timer1 or Timer3
Compare	Timer1 or Timer3

PWM	Timer2

Mode Selection

CCP1CON p149

REGISTER 16-1: CCP1CON: ECCP CONTROL REGISTER (40/44-PIN DEVICES)

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
P1M1	P1M0	DC1B1	DC1B0	CCP1M3	CCP1M2	CCP1M1	CCP1M0
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7-6 **P1M<1:0>:** Enhanced PWM Output Configuration bits

If CCP1M3:CCP1M2 = 00, 01, 10:

xx = P1A assigned as capture/compare input/output; P1B, P1C, P1D assigned as port pins

If CCP1M3:CCP1M2 = 11:

00 = Single output, P1A modulated; P1B, P1C, P1D assigned as port pins
01 = Full-bridge output forward, P1D modulated; P1A active; P1B, P1C inactive
10 = Half-bridge output, P1A, P1B modulated with dead-band control; P1C, P1D assigned as port pins
11 = Full-bridge output reverse, P1B modulated; P1C active; P1A, P1D inactive

bit 5-4 **DC1B<1:0>:** PWM Duty Cycle bit 1 and bit 0

Capture mode:
Unused.

Compare mode:
Unused.

PWM mode:
These bits are the two LSbs of the 10-bit PWM duty cycle. The eight MSbs of the duty cycle are found in CCPR1L.

bit 3-0 **CCP1M<3:0>:** Enhanced CCP Mode Select bits

0000 = Capture/Compare/PWM off (resets ECCP module)
0001 = Reserved
0010 = Compare mode, toggle output on match
0011 = Capture mode
0100 = Capture mode, every falling edge
0101 = Capture mode, every rising edge
0110 = Capture mode, every 4th rising edge
0111 = Capture mode, every 16th rising edge
1000 = Compare mode, initialize CCP1 pin low; set output on compare match (set CCP1IF)
1001 = Compare mode, initialize CCP1 pin high; clear output on compare match (set CCP1IF)
1010 = Compare mode, generate software interrupt only, CCP1 pin reverts to I/O state
1011 = Compare mode, trigger special event (ECCP resets TMR1 or TMR3, sets CCP1IF bit)
1100 = PWM mode, P1A, P1C active-high; P1B, P1D active-high
1101 = PWM mode, P1A, P1C active-high; P1B, P1D active-low
1110 = PWM mode, P1A, P1C active-low; P1B, P1D active-high
1111 = PWM mode, P1A, P1C active-low; P1B, P1D active-low

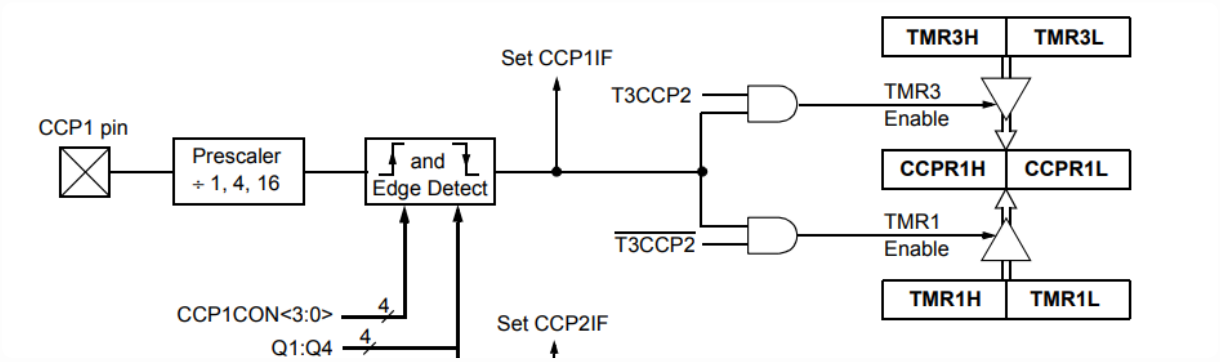
Capture mode(input)

Event definition

We capture 16-bits value of the TMR1 or TMR3 register when an event occurs on the corresponding CCPx pin.

An event is defined as one of the following:

every falling edge
every rising edge
every 4th rising edge
every 16th rising edge
在特定時間內看有無發生EVENT，if有會將timer值寫到CCPRxH、CCPRxL中



Compare mode(out)

When a match occurs, the CCPx pin can be:

driven high
driven low
toggled (high-to-low or low-to-high)
remain unchanged (that is, reflects the state of the I/O latch)

PWM mode(out)

週期性輸出訊號，透過 PWM period = (PR2 + 1) * 4 * Tosc * (TMR2 prescaler)，決定多久發出一次訊號

setup 過程:

```
    // Timer2 -> On, prescaler -> 4
    T2CONbits.TMR2ON = 0b1;
    T2CONbits.T2CKPS = 0b01;

    // Internal Oscillator Frequency, Fosc = 125 kHz, Tosc = 8 µs
    OSCCONbits.IRCF = 0b001;

    // PWM mode, P1A, P1C active-high; P1B, P1D active-high
    CCP1CONbits.CCP1M = 0b1100;

    // CCP1/RC2 -> Output
    TRISC = 0;
    LATC = 0;

    // Set up PR2, CCP to decide PWM period and Duty Cycle
    /**
     * PWM period
     * = (PR2 + 1) * 4 * Tosc * (TMR2 prescaler)
     * = (0x9b + 1) * 4 * 8µs * 4
     * = 0.019968s ~= 20ms
     */
    PR2 = 0x9b;

    /**
     * 500 ~ 2400 µs (-90 ~ 90, 1450 us = 0)
     * Duty cycle
     * = (CCPR1L:CCP1CON<5:4>) * Tosc * (TMR2 prescaler)
     * = (0x0b*4 + 0b01) * 8µs * 4
     * = 0.00144s ~= 1450µs
     *
     */
    //initial motor to -90 deg
    CCPR1L = 4;
    CCP1CONbits.DC1B = 0;
```

LAB 9 adc旋鈕輸入

參考頁數

Register名稱	在第幾頁	用途
ADCON2	第p.229 set ASCS(ADCON2 p.225)頁	set ASCS (Tad 、ACQT)
T0CON	第125頁	設定Timer0的啟動、預除器後除器
T2CON	第135頁	設定Timer2的啟動、預除器後除器
PIR1	第98頁	TMR2IF、TMR1IF等

PIE1	第100頁	TMR2IE、TMR1IE等
IPR1	第102頁	TMR2IP、TMR1IP等

什麼是ADC

主要功能：把輸入的類比訊號轉成數位數值
本次Lab會把可變電阻輸入的電壓轉成數值形式

接法:左邊接 5V，右邊接地，中間接 Analog 輸入

VREF與resolution

VREF+：上界的參考電壓
VREF-：下界的參考電壓
Resolution：ADC的解析度

> e.g., VREF- = 0V, VREF+ = 10V, Resolution : 10bits (range = [0,1023])
> 0V-> 0
> 5V -> 511
> 10V -> 1023

TAD

越小越好，但要>0.7us
可以設定成n *Tos, 看
p.229 set ASCS(ADCON2 p.225)

TABLE 19-1: TAD vs. DEVICE OPERATING FREQUENCIES

AD Clock Source (TAD)		Maximum Device Frequency	
Operation	ADCS<2:0>	PIC18F2X20/4X20	PIC18LF2X2X/4X20(4)
2 TOSC	000	2.86 MHz	1.43 kHz
4 TOSC	100	5.71 MHz	2.86 MHz
8 TOSC	001	11.43 MHz	5.72 MHz
16 TOSC	101	22.86 MHz	11.43 MHz
32 TOSC	010	40.0 MHz	22.86 MHz
64 TOSC	110	40.0 MHz	22.86 MHz
RC(3)	x11	1.00 MHz(1)	1.00 MHz(2)

Note 1: The RC source has a typical TAD time of 1.2 µs.
2: The RC source has a typical TAD time of 2.5 µs.
3: For device frequencies above 1 MHz, the device must be in Sleep for the entire conversion or the A/D accuracy may be out of specification.
4: Low-power (PIC18LFXXXX) devices only.

再去設定ACQT設定採樣時間，>2.4us

ADRESH、ADRESL

result of conversion

ADFM設定解析度

When ADFM = 0 (LEFT JUSTIFIED)

ADRESH register

ADRES9	ADRES8	ADRES7	ADRES6	ADRES5	ADRES4	ADRES3	ADRES2

ADRESL register

ADRES1	ADRES0	-	-	-	-	-	-

When ADFM = 1 (RIGHT JUSTIFIED)

ADRESH register

techetrx.com

-	-	-	-	-	-	ADRES9	ADRES8

ADRESL register

ADRES7	ADRES6	ADRES5	ADRES4	ADRES3	ADRES2	ADRES1	ADRES1

REGISTER 19-3: ADCON2: A/D CONTROL REGISTER 2

R/W-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
ADFM	—	ACQT2	ACQT1	ACQT0	ADCS2	ADCS1	ADCS0
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7 **ADFM:** A/D Result Format Select bit
1 = Right justified
0 = Left justified

bit 6 **Unimplemented:** Read as '0'

bit 5-3 **ACQT<2:0>:** A/D Acquisition Time Select bits
111 = 20 TAD
110 = 16 TAD
101 = 12 TAD
100 = 8 TAD
011 = 6 TAD
010 = 4 TAD
001 = 2 TAD
000 = 0 TAD(1)

bit 2-0 **ADCS<2:0>:** A/D Conversion Clock Select bits
111 = FRC (clock derived from A/D RC oscillator)(1)
110 = FOSC/64
101 = FOSC/16
100 = FOSC/4
011 = FRC (clock derived from A/D RC oscillator)(1)
010 = FOSC/32
001 = FOSC/8
000 = FOSC/2

Note 1: If the A/D FRC clock source is selected, a delay of one TCY (instruction cycle) is added before the A/D clock starts. This allows the SLEEP instruction to be executed before starting a conversion.

work flow

step-1

Configure the ADC module:

Select VREF (ADCON1.VCFG0, ADCON1.VCFG1)
Select A/D port control(ADCON1.PCFG)
Select A/D input channel (ADCON0.CHS)
Select A/D conversion clock (ADCON2.ADCS)
Select A/D acquisition time (ADCON2.ACQT)
Select justified method (ADCON2.ADFM)
Turn on A/D module (ADCON0.ADON)

Note : The port pins needed as analog inputs must have their corresponding TRIS bits set (input).

step-2

Configure the ADC interrupt:

Enable A/D interrupt (PIE1.ADIE)
Clear A/D interrupt flag bit (PIR1.ADIF)
Enable peripheral interrupt (INTCON.PEIE)
Set GIE bit (INTCON.GIE)

step-3

Start conversion:

Set GO/DONE bit (ADCON0.GO)

step-4

Conversion completed:

Go to ISR
Read value of ADRES register
Do things you want
Clear ADC interrupt flag bit (PIR1.ADIF)

step-5

Next conversion(if required) :

You need to have a minimum wait of 2 TAD

before next acquisition start, then go back to step 3.

Ex

```c

#include <xc.h>
#include <pic18f4520.h>
#include <stdio.h>

#pragma config OSC = INTIO67 // Oscillator Selection bits
#pragma config WDT = OFF     // Watchdog Timer Enable bit
#pragma config PWRT = OFF    // Power-up Enable bit
#pragma config BOREN = ON    // Brown-out Reset Enable bit
#pragma config PBADEN = OFF  // Watchdog Timer Enable bit
#pragma config LVP = OFF     // Low Voltage (single -supply) In-Circute Ser
#pragma config CPD = OFF     // Data EEPROM?Memory Code Protection bit (Dat

void __interrupt(high_priority)H_ISR(){

    //step4
    int value = ADRESH;


    //do things


    //clear flag bit
    PIR1bits.ADIF = 0;


    //step5 & go back step3
    /*
    delay at least 2tad
    ADCON0bits.GO = 1;
    */

    return;
}

void main(void)
{
    //configure OSC and port
    OSCCONbits.IRCF = 0b100; //1MHz
    TRISAbits.RA0 = 1;       //analog input port

    //step1
    ADCON1bits.VCFG0 = 0;
    ADCON1bits.VCFG1 = 0;
    //互相搭配------------------
    ADCON1bits.PCFG = 0b1110; //AN0 為analog input,其他則是 digital
    ADCON0bits.CHS = 0b0000;  //AN0 當作 analog input
    //-----------------------
    ADCON2bits.ADCS = 0b000;  //查表後設000(1Mhz < 2.86Mhz)
    ADCON2bits.ACQT = 0b001;  //Tad = 2 us acquisition time設2Tad = 4 > 2.4
    ADCON0bits.ADON = 1;
    ADCON2bits.ADFM = 0;    //left justified


    //step2
    PIE1bits.ADIE = 1;
    PIR1bits.ADIF = 0;
    INTCONbits.PEIE = 1;
    INTCONbits.GIE = 1;
```

```
60
61
62      //step3
63      ADCON0bits.GO = 1;
64
65      while(1);
66
67      return;
68  }
```

basic

- if left justfied 要shift : `int value = (ADRESH<<2)|(ADRESL>>6);`
- 要在ISR中加入if(step != pre_step)才更新LATB,才不會更新太快。

advance

基本上一樣，但變成不能在state更新，要用value更新，但會閃所以要用THRESH 決定大於才更新

```
1  if(abs(value - pre_value) > THRESH){
2              if(pre_value > value){
3                  LATB = seq_neg[step];
4              }
5              else if(pre_value < value) {
6                  LATB = seq_pos[step];
7              }
8              pre_value = value;
9  }
```

bonus

結合PWM duty cycle，recall 前面有10 bit resolution(CCPR1L:CCP1CON<5:4>)，這邊也是過設定這10 bit 決定亮度 :

- 1111111111 max
- 0000000000 min

> 馬達是特定區間，如 500 to 2400，沒有吃滿

Lab 10

if 用TXIE,TXIF會一直觸發isr導致無法跑主程式

```
1      //interrupt ENABLE
2      PIE1bits.TXIE = 0;    .//  set as zero
3      PIE1bits.RCIE = 1;
```

bounding rate

p206 to p208可以直接查表，也有公式直接看即可此lab 4Mhz

```
TXSTAbits.SYNC = 0;
BAUDCONbits.BRG16 = 0;
TXSTAbits.BRGH = 0;
SPBRG = 51;
```

timer

看timer.c

final

要動頻率: CCP(TMR2) ,TMR1,ADC(Tad), portrate

> 125khz uart也可以做，這樣馬達就好辦了，but adc有點慢
> OSCCONbits.IRCF = 0b001;
> ADCON2bits.ADCS = 0b000; // ADC clock = Fosc/2(Tad)
> ADCON2bits.ACQT = 0b001; // Acquisition time = 2 TAD (>= 2.45 μs)
> PR2 = 0x9b
> T2CONbits.T2CKPS = 0b01; // Prescaler = 4
> //portrate
> TXSTAbits.SYNC = 0;
> BAUDCONbits.BRG16 = 1;
> TXSTAbits.BRGH = 0;
> SPBRG = 26;

> 500khz可work，ADC不會慢,portrate有點慢(射程1200可以)
> OSCCONbits.IRCF = 0b011;
> ADCON2bits.ADCS = 0b000; // ADC clock = Fosc/2(Tad)>0.7us
> ADCON2bits.ACQT = 0b001; // Acquisition time = 2 TAD (>= 2.45 μs)
> PR2 = 155
> T2CONbits.T2CKPS = 0b11; // Prescaler = 16
>
> //set 1200 br
> TXSTAbits.SYNC = 0;
> BAUDCONbits.BRG16 = 1;
> TXSTAbits.BRGH = 0;
> SPBRG = 25;

ISR ADC 會一職觸發

final project

延遲兩秒問題

2023考古

bourd rate = 300太慢，閃燈要等他寫完：改成1200

- 2:
  - 要把按鈕射程下緣觸發 `INTCON2bits.INTEDG0 = 0;` 不然會一開始就觸發一次。
  - 在UART如果輸入過程有刪除字元，最後string會錯，但print出來是正常string: don't known why?
- 3:
  - 馬達在 0 會一直抖動，不知道為啥，有時候不會，單獨轉ADC正常，加入CCP才壞掉：重燒一次有機率會好。