lab 4

- I/O
  - PORTA
    - RA0 → RA4
    - 4 bit / pin
  - PORTB
    - RB1 → RB7
    - 7 bit / pin
    - RB0 → interrupt
    - RB0 / INT
- interrupt

is it output or input?

Both are Both

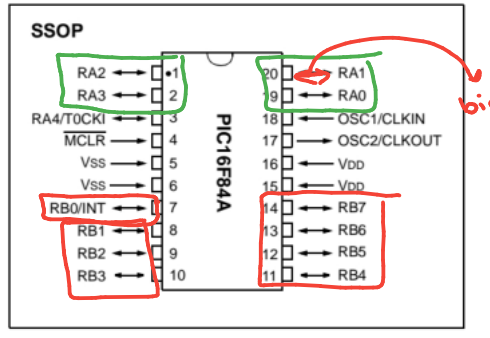
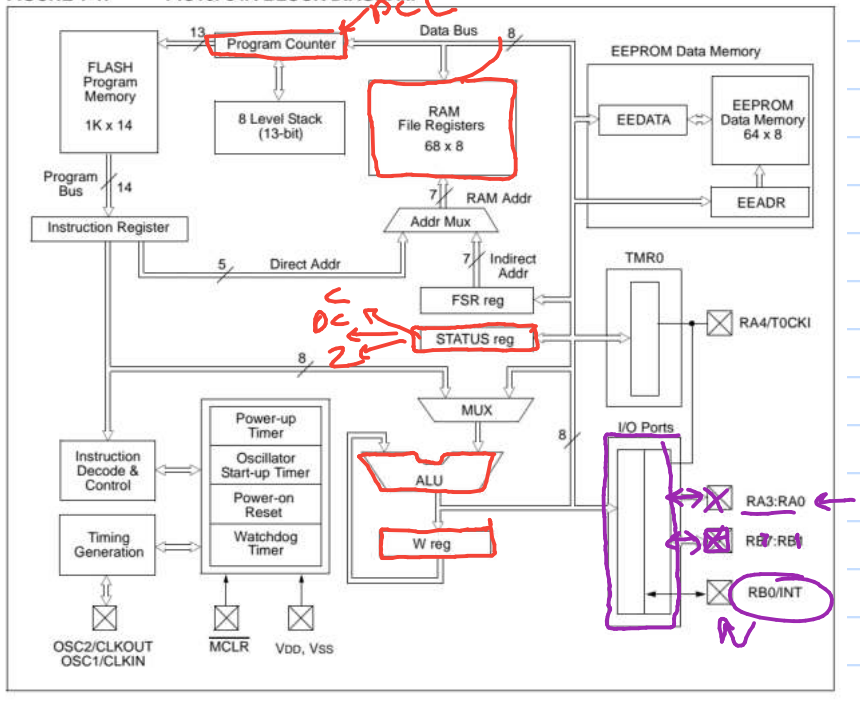


FIGURE 1-1: PIC16F84A BLOCK DIAGRAM



How can I define if PORTA or PORTB is an input or an output?

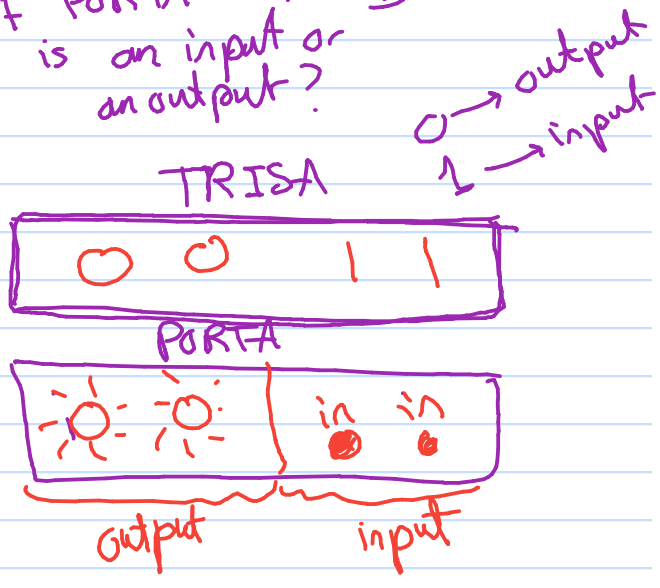


FIGURE 2-2: REGISTER FILE MAP - PIC16F84A

File Address			File Address
00h	Indirect addr.(1)	Indirect addr.(1)	80h
01h	TMR0	OPTION_REG	81h
02h	PCL	PCL	82h
03h	STATUS	STATUS	83h
04h	FSR	FSR	84h
05h	PORTA	TRISA	85h
06h	PORTB	TRISB	86h
07h	—	—	87h
08h	EEDATA	EECON1	88h
09h	EEADR	EECON2(1)	89h
0Ah	PCLATH	PCLATH	8Ah
0Bh	INTCON	INTCON	8Bh
0Ch	68 General Purpose Registers (SRAM)	Mapped (accesses) in Bank 0	8Ch
4Fh / 50h			CFh / D0h
7Fh	Bank 0	Bank 1	FFh

☐ Unimplemented data memory location, read as '0'.
Note 1: Not a physical register.

direction indicator

TRISB

0000 1111

PORTB

n bit | 4 bit

output inputs

status Reg

RP0

de RP0 = 0 / 1

RP0

0 Bank 0

1 Bank 1

BSF Status, RP0

bit set = 1

= access 1 TRISA TRISB

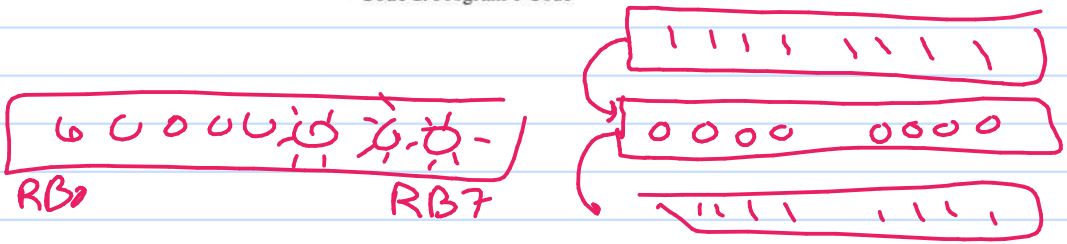
Bit clear = 0

BCF status RP0

Program Example

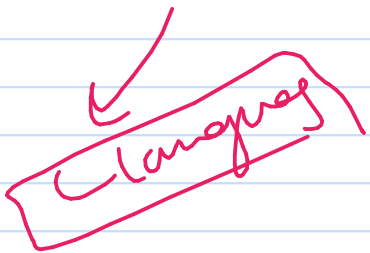
```
        BSF     STATUS, RP0     ; Go to bank 1
        CLRF    TRISB           ; Setup all pins of PORTB as outputs
        BCF     STATUS, RP0     ; Go to bank 0
        MOVLW   0xFF            ; W = 0xFF
        MOVWF   PORTB           ; PORTB = W
LOOP    CALL    delay500ms      ; Call a subroutine resulting in a 0.5s delay
        COMF    PORTB           ; Complement all pins of PORTB
        GOTO    LOOP
```

Code 1. Program 1 Code

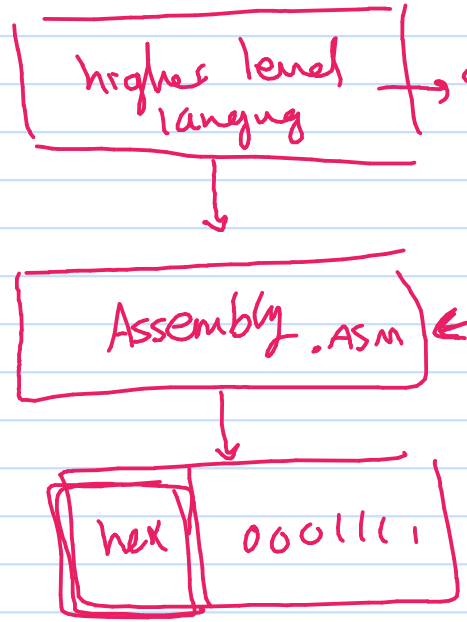


Program C programming

*Program 2



MPLAB → Assembly code



simulator ✗ → PICkit3

mpasm ✗ → XC8 → clan

assembly

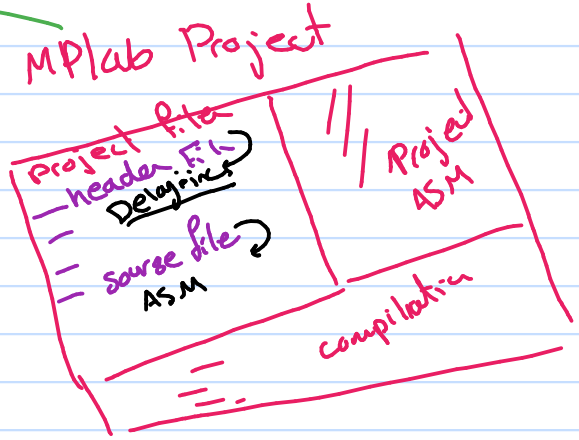
Assembly in Moodle
File.zip

Delay.inc

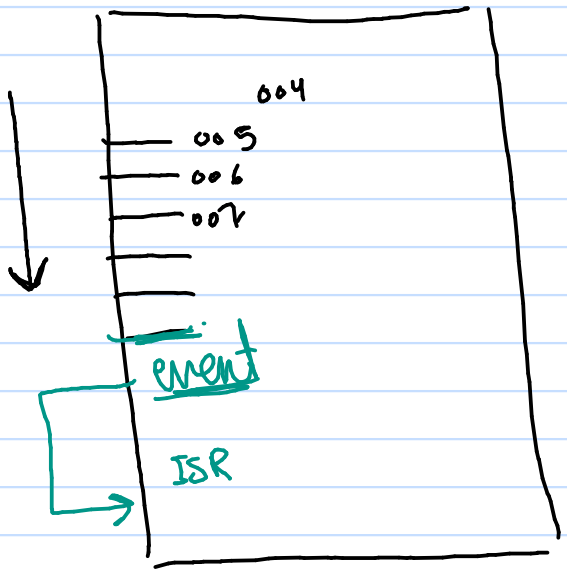
Project File
"lab4.X"
copy the file inside

Before main
#include "Delay.inc"
inside the code file

MPlab Project



« Interrupt » → interrupts your normal sequence of events



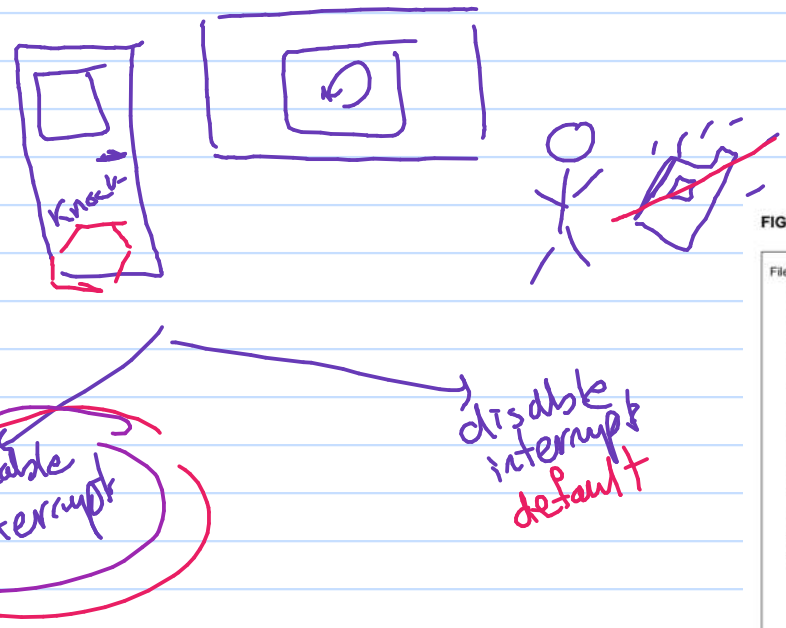


FIGURE 2-2: REGISTER FILE MAP - PIC16F84A

File Address	Indirect addr.(1)	Indirect addr.(1)	File Address
00h			80h
01h	TMR0	OPTION_REG	81h
02h	PCL	PCL	82h
03h	STATUS	STATUS	83h
04h	FSR	FSR	84h
05h	PORTA	TRISA	85h
06h	PORTB	TRISB	86h
07h	—	—	87h
08h	EEDATA	EECON1	88h
09h	EEADR	EECON2(1)	89h
0Ah	PCLATH	PCLATH	8Ah
0Bh	INTCON	INTCON	8Bh
0Ch	68 General Purpose Registers (SRAM)	Mapped (accesses) in Bank 0	8Ch
4Fh / 50h			CFh / D0h
7Fh	Bank 0	Bank 1	FFh

☐ Unimplemented data memory location, read as '0'.
Note 1: Not a physical register.

Responsible for interrupts

REGISTER 2-3: INTCON REGISTER (ADDRESS 0Bh, 8Bh)

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-x
GIE	EEIE	T0IE	INTE	RBIE	T0IF	INTF	RBIF
bit 7							bit 0

bit 7 **GIE**: Global Interrupt Enable bit → GIE = 1
1 = Enables all unmasked interrupts
0 = Disables all interrupts

bit 6 **EEIE**: EE Write Complete Interrupt Enable bit
1 = Enables the EE Write Complete interrupts
0 = Disables the EE Write Complete interrupt

bit 5 **T0IE**: TMR0 Overflow Interrupt Enable bit
1 = Enables the TMR0 interrupt
0 = Disables the TMR0 interrupt

bit 4 **INTE**: RB0/INT External Interrupt Enable bit ← enable external interrupt "RB0"
1 = Enables the RB0/INT external interrupt
0 = Disables the RB0/INT external interrupt

bit 3 **RBIE**: RB Port Change Interrupt Enable bit
1 = Enables the RB port change interrupt
0 = Disables the RB port change interrupt

bit 2 **T0IF**: TMR0 Overflow Interrupt Flag bit
1 = TMR0 register has overflowed (must be cleared in software)
0 = TMR0 register did not overflow

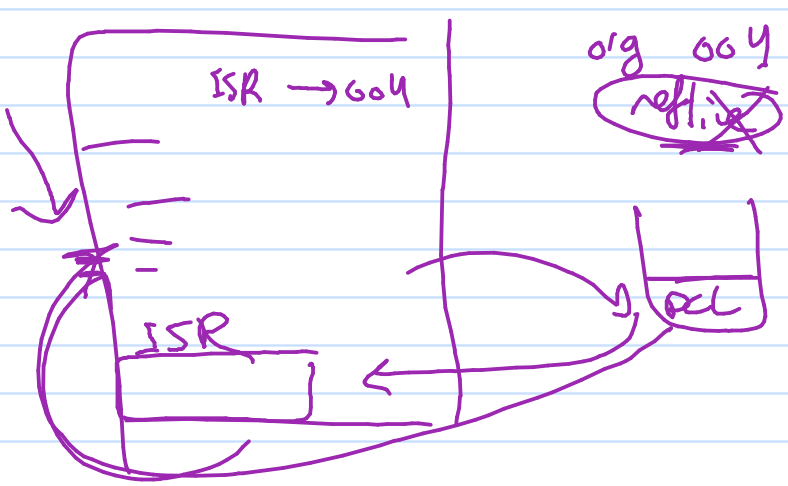
bit 1 **INTF**: RB0/INT External Interrupt Flag bit
1 = The RB0/INT external interrupt occurred (must be cleared in software)
0 = The RB0/INT external interrupt did not occur

bit 0 **RBIF**: RB Port Change Interrupt Flag bit
1 = At least one of the RB7:RB4 pins changed state (must be cleared in software)
0 = None of the RB7:RB4 pins have changed state

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

```
BSF   INTCON, GIE
BSF   INTCON, INTE
```

ISR → interrupt service Routine



org 004
reti
GoTo ISR

```
ISR   BCF   INTCON, INTF
      =
      =
      =
      =
      RETFIN
```