Lab 2 – Interfacing to switches and LED's

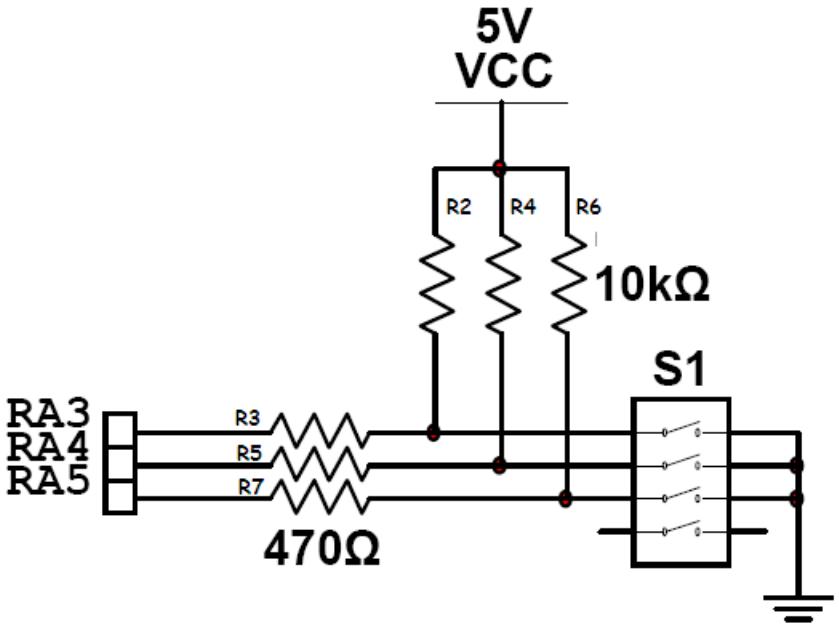
Objectives

- ☐ To learn to configure PIC18F4550's I/O ports as inputs or outputs.
- ☐ To learn to read status of switches – open or closed.
- ☐ To learn to turn on / off a number of LED's, in various sequences.
- ☐ To learn to use the delay functions.

Introduction / Briefing

Switches at Port A

- ☐ In this experiment, you will be reading the status of the dip switches connected to Port A.



Below are the ways in which the switches are named and used in software.
Switch at RA3 has the name PORTAbits.RA3
Switch at RA4 has the name PORTAbits.RA4
Switch at RA5 has the name PORTAbits.RA5
PORTA on its own refers to all 8 bits and not individual bits.

☐ Study the above diagram and answer the following questions:

Q1: How many dip switches are there? _____

Q2: How many are connected to Port A? _____

Q3: What is the purpose of the 470 ohm resistors?

_____

(The above is a tough question. Hint: imagine someone making the mistake of configuring Port A as output AND producing a logic '1' at one of RA3:RA5 AND the corresponding dip switch is closed.)

Q4: Are the switches connected in the "active high" or "active low" manner? (An "active high" switch gives a logic '1' when closed.)

_____

Q5: What will a Port A pin read (logic '1' or '0') when a corresponding dip switch is closed? _____

☐ To allow Port A to read the dip switches, it must be configured as a digital input port. However, Port A is a partially analogue/partially digital input port by default (after power on reset):

[Refer to the "insert" on the next page to understand the last column.]

Port	Available pins	Not available as general purpose I/O (- reasons)	After power on reset
A	RA6-0	RA6 (– oscillator)	RA5, 3-0: Analogue inputs (*). RA4: Digital input.

Q6: Give the C-command to configure Port A as a digital input port (hint: ADCON1):

_____ // configure Port A as digital inp.

REGISTER 21-2: ADCON1: A/D CONTROL REGISTER 1

U-0	U-0	R/W-0	R/W-0	R/W-0(1)	R/W(1)	R/W(1)	R/W(1)
—	—	VCFG0	VCFG0	PCFG3	PCFG2	PCFG1	PCFG0
bit 7							bit 0

bit 3-0 **PCFG3:PCFG0:** A/D Port Configuration Control bits:

Default value of PCFG3 is 0, making AN4-0 all analogue inputs.

AN4 is also RA5, while AN3-0 are also RA3-0.

The C code ADCON1 = 0x0F; puts binary 1111 into PCFG3-0, making AN4-0 i.e. RA5, RA3-0 all digital.

PCFG3: PCFG0	AN12	AN11	AN10	AN9	AN8	AN7(2)	AN6(2)	AN5(2)	AN4	AN3	AN2	AN1	AN0
0000(1)	A	A	A	A	A	A	A	A	A	A	A	A	A
0001	A	A	A	A	A	A	A	A	A	A	A	A	A
0010	A	A	A	A	A	A	A	A	A	A	A	A	A
0011	D	A	A	A	A	A	A	A	A	A	A	A	A
0100	D	D	A	A	A	A	A	A	A	A	A	A	A
0101	D	D	D	A	A	A	A	A	A	A	A	A	A
0110	D	D	D	D	A	A	A	A	A	A	A	A	A
0111(1)	D	D	D	D	D	A	A	A	A	A	A	A	A
1000	D	D	D	D	D	D	A	A	A	A	A	A	A
1001	D	D	D	D	D	D	D	A	A	A	A	A	A
1010	D	D	D	D	D	D	D	D	A	A	A	A	A
1011	D	D	D	D	D	D	D	D	D	A	A	A	A
1100	D	D	D	D	D	D	D	D	D	D	A	A	A
1101	D	D	D	D	D	D	D	D	D	D	D	A	A
1110	D	D	D	D	D	D	D	D	D	D	D	D	A
1111	D	D	D	D	D	D	D	D	D	D	D	D	D

A = Analog input D = Digital I/O

☐ When a dip switch is closed, the corresponding Port A pin will read a logic '0'.

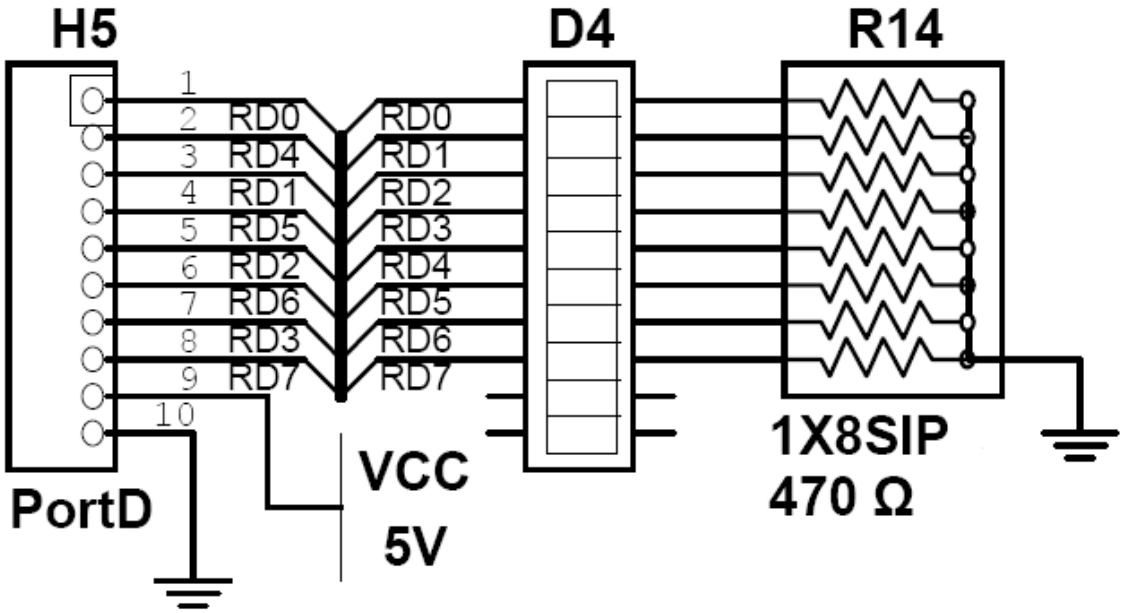
Q7: Give the C-command to check if the dip switch connected to RA3 is closed (hint: PORTA):

```
if _____ // if dip switch @ RA3 is closed
        {
        // do something

        }
```

LED bar at Port D

☐ In this experiment, you will also be turning on and off an LED bar connected to Port D.



☐ Study the above diagram and answer the following questions:

Q8: How many LED's are there in the LED bar? _____

Q9: How many are connected to Port D? _____

Q10: What is the purpose of the 470 ohm resistors in the SIP (Single-In-Line package)? _____

Q11: Are the LED's connected in the "common anode" or "common cathode" manner? _____

Q12: What must a Port D pin produce (logic '1' or '0') to turn on a corresponding LED? _____

☐ To allow Port D to control the LED bar, it must be configured as a digital output port. However, Port D is a digital input port by default (after power on reset):

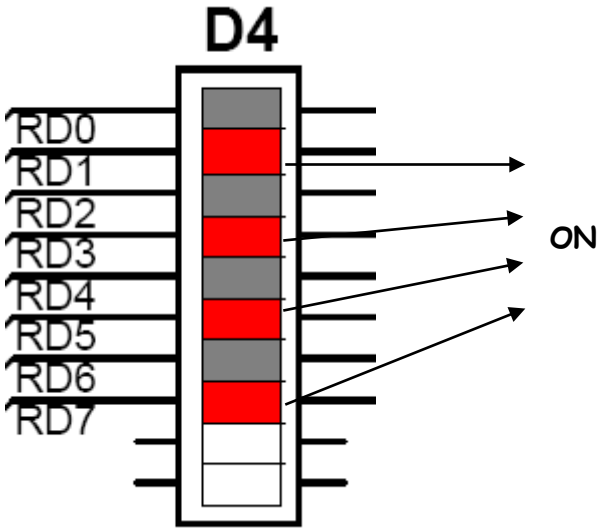
Port	Available pins	Not available as general purpose I/O (- reasons)	After power on reset
D	RD7-0		RD7-0: Digital inputs.

Q13: Give the C-command to configure Port D as a digital output port (hint: TRISD):

_____ // configure Port D as digital outp.

☐ To turn on a particular LED, the corresponding Port D pin must produce a logic '1'.

Q14: Give the C-command to turn on the LED's as follows (hint: PORTD):



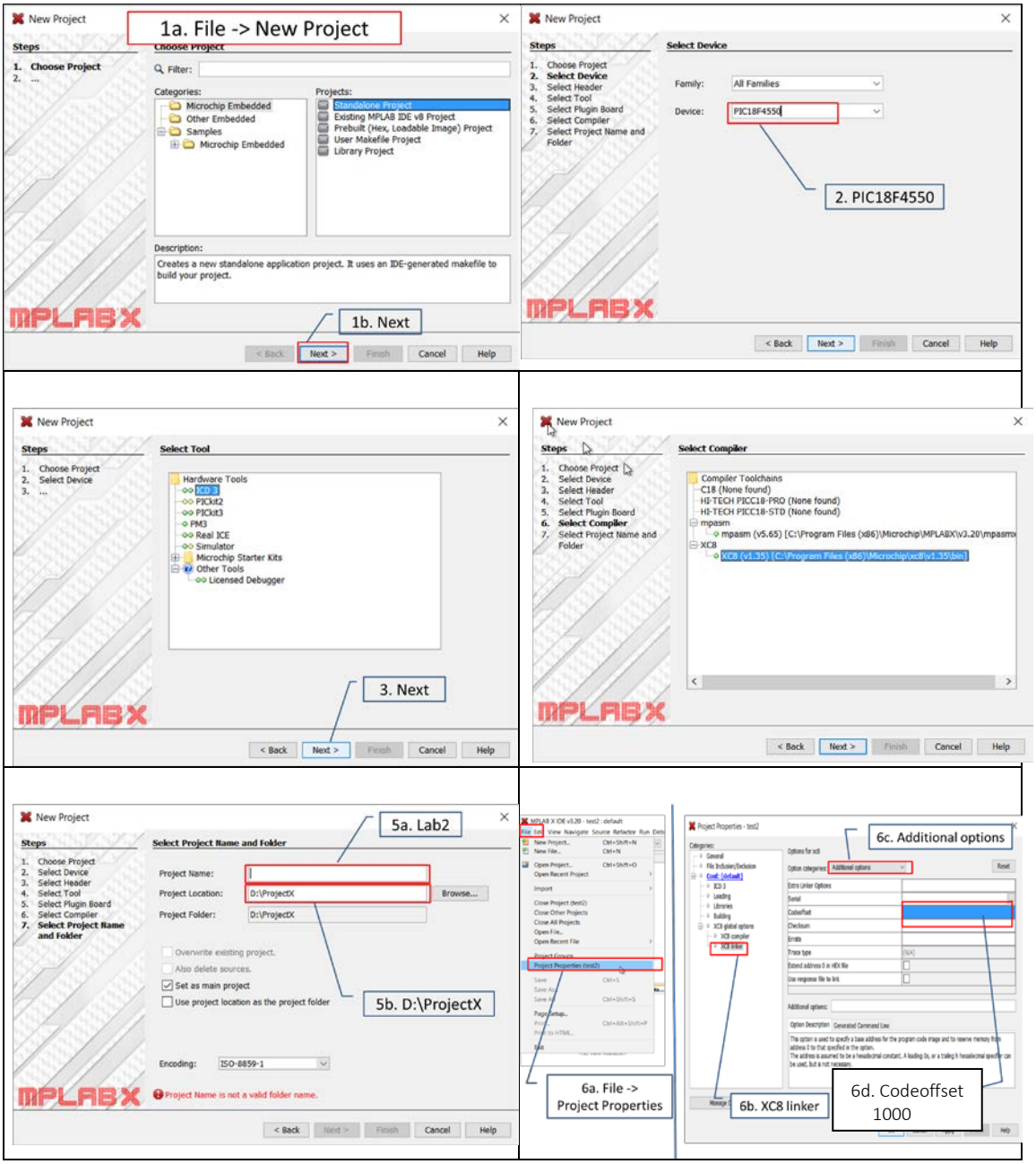
_____ // turn alternate LED's on

Activities:

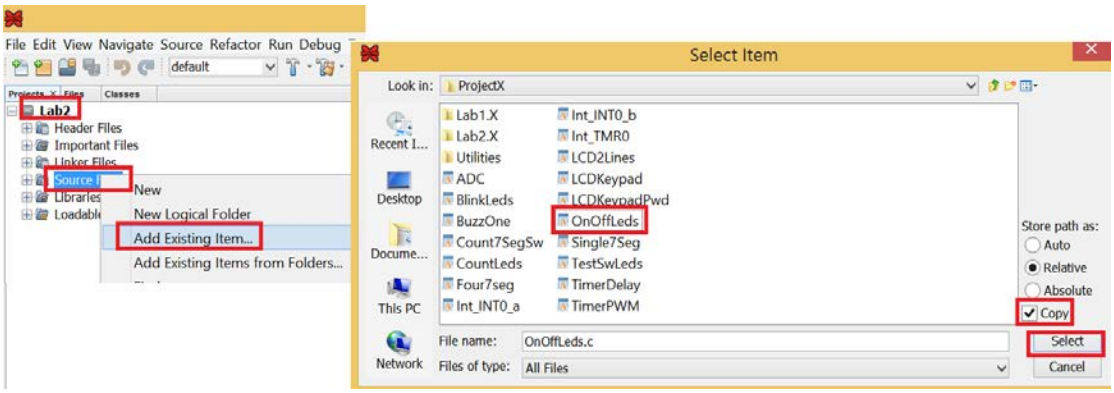
Using LEDs to indicate switch status

Create a New Project – Lab2

1. Launch the *MPLABX IDE* and create a new project called Lab2.
   Below is a summary of the steps needed or you can refer to Lab1.



2. Use "*Add Existing Item*" to add the file *OnOffLeds.c* to the *Lab2* project *Source File* folder, as follows. Make sure *Copy* is ticked before you click *Select*.



3. Double click on *Source Files – OnOffLeds.c* to look at the program.

```
// OnOffLeds.c
// Program to use switches to control 8 leds on General I/O Board

#include <xc.h>

void main(void) {
   ADCON1 = 0x0F; //make Port A digital

   TRISA = 0b11111111; //RA5 to RA3 are connected to On/Off switches

   TRISD = 0b00000000; //RD7 to RD0 are connected to LEDs

   while (1) //repeat
   {
      if (PORTAbits.RA3 == 0) //_____
      {
         PORTD = 0xF0;     // PORTD = 0b_____
      }
      else
      {
         PORTD = 0x0F;     // PORTD = 0b_____
      }
   }
}
```

4. Describe what this program will do:

_____

_____

5. Build, download and execute the program. Observe the result and see if it is as expected.

6. Change the program to use the switch at RA5, such that if RA5 is on, all the LEDs should be on and if RA5 is off, all the LEDs should be off.

7. Build, download and execute the program. Observe the result and see if it is as expected.

LED's blinking / "scanning"

Use delays, toggle LED's.

8. Right click on *OnOffLeds.c* and click "*Remove From Project*". Then add the file *BlinkLeds.c* to the project. ("*Remove From Project*" only removes the file from the project, it is not deleted.)

9. Study the code and describe what this program will do:

_____

10. Note that the program uses the delay function *delay_ms()* and contains *#include "delays.h"*. The file *delays.h* needs to be added to *Header Files* while the file *delays_utilities.c* needs to be added to *Source Files*.

11. Build, download and execute the program. Observe the result and see if it is as expected.

Pause an action

12. Add the following line to the *while(1)* loop:

```
while(1)
{
        while (PORTAbits.RA3 == 0); // loop here when switch is on

        PORTD=0b10101010;
        ……. // other existing lines – don't touch
}
```

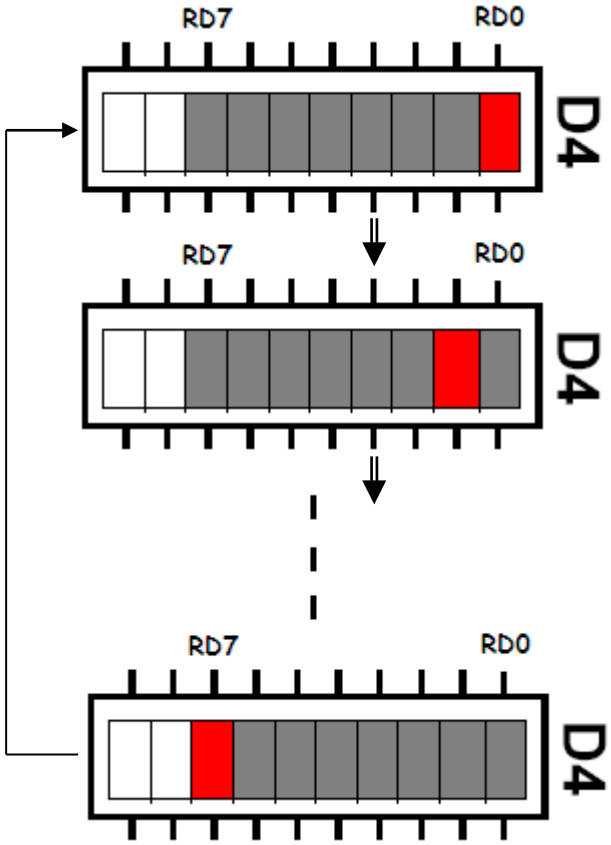
13. Describe what this NEW program will do:

_____

14. Build, download and execute the program. Observe the result and see if it is as expected.

Right to left "scan"

15. Modify the program to do "scanning", such that one LED light repeatedly moves from right to left (after a short delay).



Pause the "scan"

16. Add in a switch control line such that when the switch connected to RA3 is closed, the "scanning" is paused.

17. Debug until the program can work.

2-way "scan"

18. Modify the program to do a "right to left scan", followed by a "left to right scan" repeatedly. Include the switch control line to pause the scanning with a closed switch at RA3.

19. Debug until the program can work.

Slow down the "scan"

20. Modify the program such that a closed switch at RA4 slows down the scanning (while a closed switch at RA3 pauses the scanning). (Hint: add delay if switch at RA4 is closed.)

21. Debug until the program can work.

LED's "counting"

Counting

22. Replace *BlinkLeds.c* with *CountLeds.c*.

23. Study the code and describe what this program will do:

_____

24. Build, download and execute the program. Observe the result and see if it is as expected.

Counting Up/down

25. Modify the program such that a closed switch at RA5 causes counting up while an opened switch causes counting down. Here are some hints:

```
while (1)
{
        ..... // other lines unchanged
        if (PORTAbits.RA5 == 0) // if switch is closed
                _____ // count up
        else
                _____ // count down
}
```

Slow down the counting

26. Add in a line such that a closed switch at RA4 slows down the counting.

Pause the counting

27. Add in another line such that a closed switch at RA3 pauses the counting.

28. Debug until the program can work.

Extra Exercise

29. A left-shift is equivalent to multiplication by 2 while a right-shift is equivalent to division by 2.

2-way scan, with pausing & slowing down

30. If you still have time at the end of this Lab, try to write a LED scanning program (you can modify any existing file – OnOffLeds.c or BlinkLeds.c or CountLeds.c) such that the scanning is normally from right to left, but

    A closed switch at RA5 causes a left to right scan.
    A closed switch at RA4 causes scanning to slow down.
    A closed switch at RA3 causes scanning to pause.

31. Debug until the program can work.

```
// OnOffLeds.c
// Program to use 1 switch to control 8 leds on General I/O Board

#include <xc.h>

void main(void) {
    ADCON1 = 0x0F;      //make Port A digital

    TRISA = 0b11111111; //RA5 to RA3 are connected to On/Off switches

    TRISD = 0b00000000; //RD7 to RD0 are connected to LEDs

    while (1) //repeat
    {
        if (PORTAbits.RA3 == 0) //_____________________
        {
            PORTD = 0xF0;     // PORTD = 0b_____________________
        }
        else
        {
            PORTD = 0x0F;     // PORTD = 0b_____________________
        }
    }
}
```

// BlinkLeds.c

```
// Program to light up alternate leds on General I/O Board

#include <xc.h>
#include "delays.h"

void main(void) {
    ADCON1 = 0x0F;      //make Port A digital

    TRISA = 0b11111111; //RA5 to RA3 are connected to On/Off switches

    TRISD = 0b00000000; //RD7 to RD0 are connected to LEDs

    while (1) //repeat
    {
        PORTD = 0b10101010;
        delay_ms(1000);

        PORTD = 0b01010101;
        delay_ms(1000);

    }
}
```

// CountLeds.c

```
// Program to make 8 leds on General I/O Board do binary up counting

#include <xc.h>
#include "delays.h"

unsigned char j;  // 8 bit data type, range 0 to 255

void main(void)
{
      ADCON1 = 0x0F;    //make Port A digital

      TRISA=0b11111111; //RA5 to RA3 are connected to On/Off switches

      TRISD=0b00000000; //RD7 to RD0 are connected to LEDs

      j=0;                    //beginning

      while(1)                //repeat
      {

            PORTD = j;        // Output value of j to PORTD

            delay_ms(500);

            j++;              // Increment j
      }
}
```