Lab 2 – Interfacing to switches and LED's

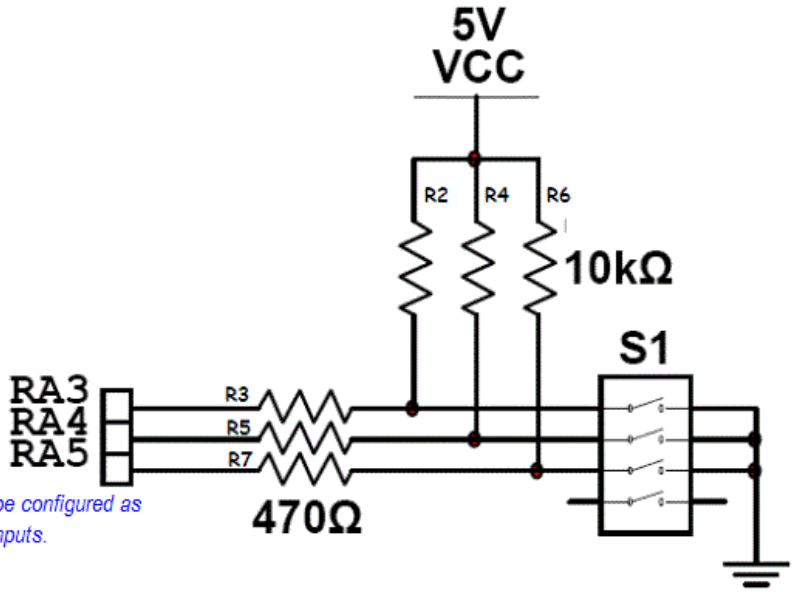
Objectives

- ☐ 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.



Should be configured as Digital Inputs. (Why?)

☐ Study the above diagram and answer the following questions:

Q1: How many dip switches are there? _____ *Dual In-line Package*

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):

ADCON1 = _____ // configure Port A as digital inp.

Pre-defined variable in the C-18 compiler, representing the corresponding control register in the PIC-18 microcontroller.

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

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 **Unimplemented:** Read as '0'

bit 5 **VCFG0:** Voltage Reference Configuration bit (VREF- source)
1 = VREF- (AN2)
0 = VSS

bit 4 **VCFG0:** Voltage Reference Configuration bit (VREF+ source)
1 = VREF+ (AN3)
0 = VDD

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

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	When PCFG3 = 0, AN4-0 are all analogue inputs. AN4 is also RA5, while AN3-0 are also RA3-0						A	A	A	A	A	A
1001	D							A	A	A	A	A	A
1010	D							D	A	A	A	A	A
1011	D							D	D	A	A	A	A

☐ 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):

Pre-defined variable representing the 8 bits in PORTA.

```
if ( PORTA & 0b00001000 == 0 ) // if dip switch @ RA3 is closed
    {                          // do something
    }
```

In binary *(In hex: 0x08)*

'Bit-wise' AND operation: Each of the 8 bits in PORTA will AND with the corresponding bit in the data.

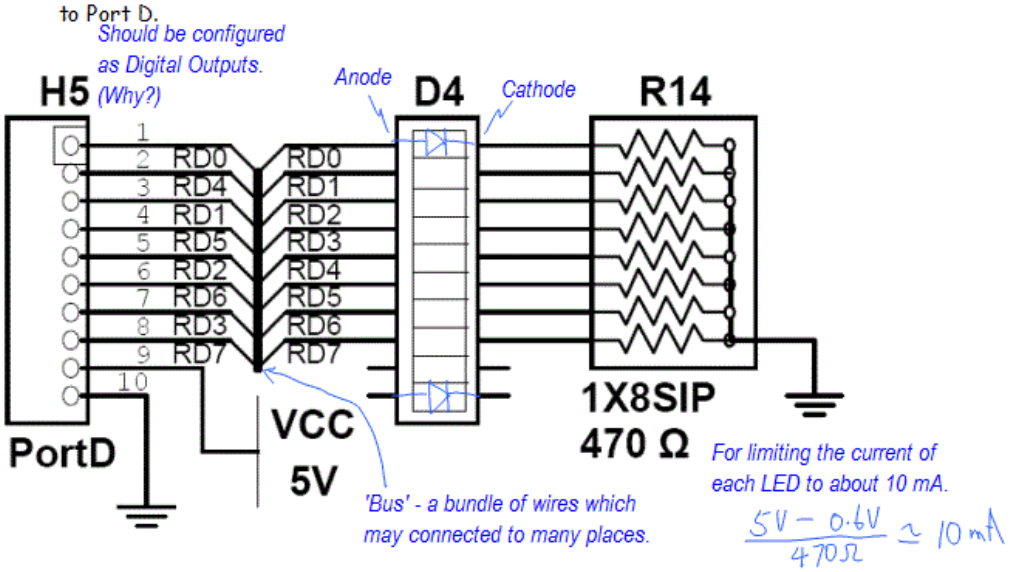
Alternatively:
if (PORTAbits.RA3 == 0)

Another pre-defined variable for PORTA which reads only 1 bit.

The pre-defined symbol for bit RA3.

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:

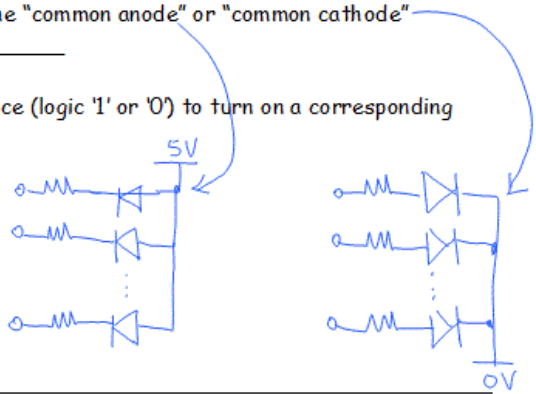
Q8: How many LED's are there in the LED bar? _____

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	Ni	RD7-0: Digital inputs.

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

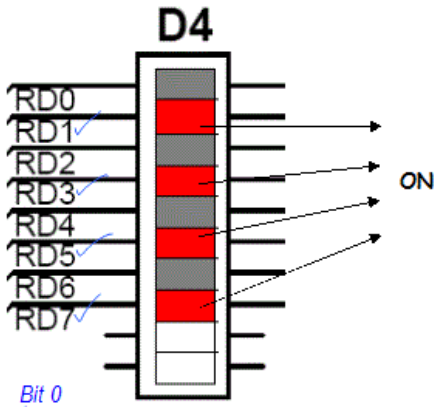
TRISD = 0; // configure Port D as digital outp.

(Alternatively: 0x00 or 0b00000000)

Recall each '0' in TRISD will configure the corresponding bit in PORTD to become Output.

☐ 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):



Bit 7 *Bit 0*

PORTD** = 0b**10101010 // turn alternate LED's on

*Alternatively: 0x**AA***
(Decimal value is not recommended in this case as it does not shows the bit pattern obviously.)

Activites:

Before you begin, ensure that the Micro-controller Board is connected to the General IO Board.

Using LED's to indicate switch status

1. Launch *MPLAB IDE*. Open Lab1 workspace by clicking *Project -> Open...* and selecting *ProjetA.mcp* from the *D:\PICProject* folder.

2. At the workspace window (*mcw*), click on *Source Files - TestSwLeds.c*, then right click to remove the file from the project.

Read switches, control LED's.

3. Next click on *Source Files*, then right click to add the file *OnOffLeds.c* to the project.

4. The above 3 steps are how you replace one C file with another. You will do this many times in this and subsequent labs. So remember the steps well.

5. Double click on *OnOffLeds.c* to study the code.

6. Describe what this program will do:

   _____

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. Replace *OnOffLeds.c* with *BlinkLeds.c*.

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

   _____

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

11. Note that the program uses the delay function *Delay10KTCYx(250)* and contains *#include <delays.h>*. To know more about this, refer to the last few pages of Lab1.

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
}
```

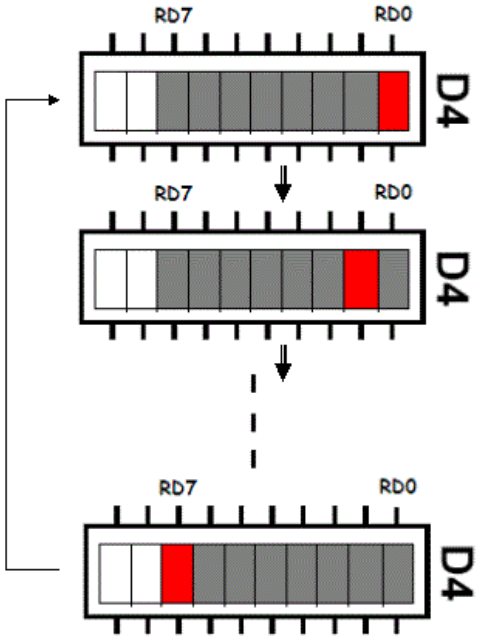
13. Describe what this NEW program will do:

_____

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 repeated 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. When your program is working, show it to your lecturer.

Lecturer's signature _____

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)
{
        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 scanning.

28. Debug until the program can work. When your program is working, show it to your lecturer.

Lecturer's signature _____

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. When your program is working, show it to your lecturer.

    Lecturer's signature _____

// OnOffLeds.c

```
// Program to use 3 switches to control 8 leds on General I/O Board

#include <delays.h>

// other lines not shown...

unsigned int i; /* 16 bit data type, range 0 to 65,535 */

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)    // If RA3 is 0, i.e. switch closed
                {
                PORTD=0xFF;              // turn on all 8 LEDs

                Delay10KTCYx(250);
                Delay10KTCYx(250);
                Delay10KTCYx(250);
                Delay10KTCYx(250);
                }
                else
                {
                PORTD=0x00;              // Else, turn off all 8 LEDs
```

Delay 4 x 250 x 10K clock cycles
(Max. value of the argument is 255.)

```
        Delay10KTCYx(250);
        Delay10KTCYx(250);
        Delay10KTCYx(250);
        Delay10KTCYx(250);
        }

        if (PORTAbits.RA4==0)   // If RA4 is 0, i.e. switch closed
        {
        PORTDbits.RD7 = 1;      // turn on LED7

        Delay10KTCYx(250);
        Delay10KTCYx(250);
        Delay10KTCYx(250);
        Delay10KTCYx(250);
        }
        else
        {
        PORTDbits.RD7 = 0;      // Else, turn off LED7

        Delay10KTCYx(250);
        Delay10KTCYx(250);
        Delay10KTCYx(250);
        Delay10KTCYx(250);
        }
    }
}
```

Delay

Delay

Delay

Alternatively:

```
while (1)
{
 if (PORTAbits.RA3 == 0)
   PORTD = 0X00;
 else
   PORTD = 0XFF;

 delay( ); // Assumed defined

 if (PORTAbits.RA4 == 0)
   PORTDbits.RD7 = 1;
 else
   PORTDbits.RD7 = 0;

 delay( );
}
```

// BlinkLeds.c

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

#include <delays.h>

// other lines not shown...

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;

        Delay10KTCYx(250);
        Delay10KTCYx(250);
        Delay10KTCYx(250);
        Delay10KTCYx(250);

        PORTD=0b01010101;

        Delay10KTCYx(250);
        Delay10KTCYx(250);
        Delay10KTCYx(250);
        Delay10KTCYx(250);
    }
}
```

// CountLeds.c

```
// Program to use 1 switch to control counting on 8 leds on General I/O // Board

#include <delays.h>

// other lines not shown...

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;                    / initialise j

        while(1)                // repeat
        {
                PORTD=j;                // output value of j to PORTD

                Delay10KTCYx(250);      // Delays
                Delay10KTCYx(250);
                Delay10KTCYx(250);
                Delay10KTCYx(250);

                j++;                    // increment j
        }
}
```