Chapter 4

Inputs

In order to use a digital input we need to ask the question, is the input high or low? Is it a logic 1 or 0?

If it is a 1 then we execute some code, if it is a 0 then we execute some other code. So we are asking the question, if the input is a certain value then execute the code. Let us look at the C code for this If statement.

IF STATEMENT

The if statement looks like this:

```
if (condition)
{
    •
    Code
    •
}
```

As an example if x is equal to 2 then make y = 6 and z = 4 would be:

```
if (x==2)
{
y=6;
z=4;
}
```

NB. The expression x==2 means if x is equal to 2.
The expression x=2 in C code means make x=2.

If there is only one line of code, say y = 6, then a shorthand way of writing this without the brackets {} is

```
if (x==2) y=6;
```

IF–ELSE

There is an extension to this statement called if–else written:

```
if (condition)
{
    •
    Code
    •
}
else
{
    •
    Code
    •
}
```

As an example of this:

```
if (x==2)
{
    y=6;
    z=4;        // if x==2 then make y=6 and z=4

}else
{
    y=1;        // if x is not equal to 2 then make y=1 and z=3
    z=3;
}
```

If we only have one line of code we could write this again without the {} brackets as:

```
if (x==2) y=6;
else y=1;
```

We are now in a position to apply this to our hardware.

Suppose as a simple example we want LED1 connected to PORTB bit 4 shown in Figure 4.1 to turn on when input RA4 is a logic 1 and turn off when it is a logic 0.

```
// input1.c by DW Smith, 22 September 2011
#include <p18f1220.h>
#pragma config WDT=OFF, OSC=INTIO2, PWRT=ON, LVP=OFF, MCLRE=OFF
#include <delays.h>

void main (void)
{
```

```
            //SET UP
            // OSCCON defaults to 31 kHz. So no need to alter it.
            ADCON1=0x7F;                //all IO are digital or 0b01111111 in binary
            TRISA=0b11111111;           //sets PORTA as all inputs
            PORTA=0b00000000;           //turns off PORTA outputs
            TRISB=0b00000000;           //sets PORTB as all outputs
            PORTB=0b00000000;           //turns off PORTB outputs, good start position

            while (1)
    {
            if (PORTAbits.RA4==1)       // if RA4 is equal to 1
            {
            PORTBbits.RB0=1;            // make RB0=1
            }
            else
            {
            PORTBbits.RB0=0;            // make RB0=0
            }
      }
    }
```

Open the file header.c, include the new lines and save the program as **input1.c**

Use the Project Wizard to make the project, or follow the steps of Figures 3.3 and 3.4 Opening a new project and adding a file.

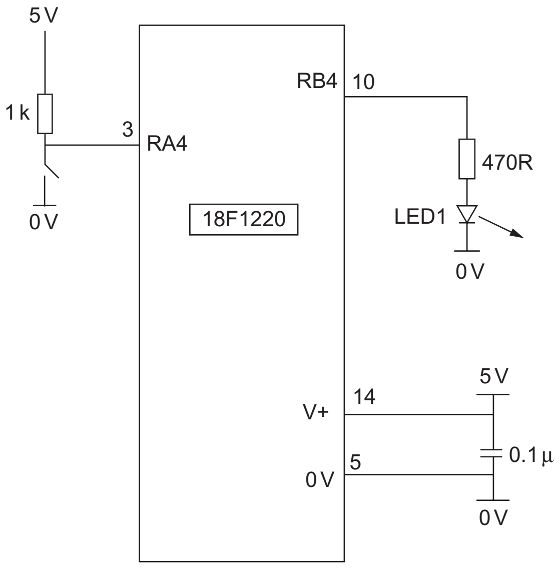

FIGURE 4.1 The input circuit.

The comments in the code explain what is going on. I have mentioned before about the curly brackets { } but let us just recap, they do look a little confusing until you are used to them.

The main function line 6 has its code written between the brackets on lines 7 and 27 (in the same column for clarity).

The while (1) loop has its code between lines 17 and 26.

The if statement has its code written between the brackets on lines 19 and 21.

The else statement has its code written between the brackets on lines 23 and 25.

If you miss a bracket or include an extra one then the program will not compile and it will display an error.

Line 4 is not required because we are not using delays.

Can you modify the code so that the LED will flash when the switch is open and extinguish when the switch is closed.

You could add more LEDs and use the switch to start the disco lights running.

USING SEVERAL INPUTS

Suppose we wish to use four momentary switches, SW0–SW3, to control four LEDs, LED0–LED3, as shown in Figure 4.2.

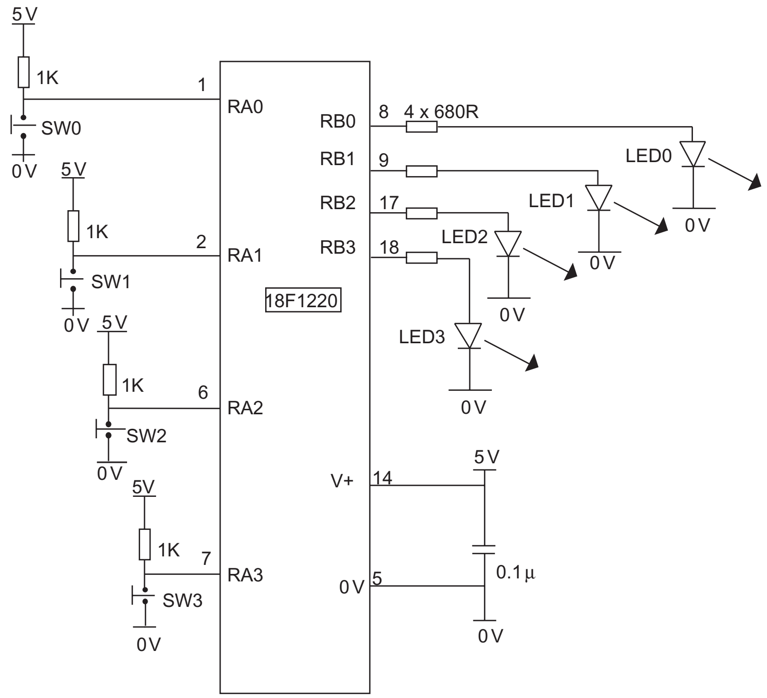

FIGURE 4.2 Using several inputs.

```
// inputs.c by DW Smith, 21 September 2011
#include <p18f1220.h>
#pragma config WDT=OFF , OSC=INTIO2 , PWRT = ON, LVP=OFF,  MCLRE = OFF
#include <delays.h>

void main (void)
{
        //SET UP
        // OSCCON defaults to 31kHz. So no need to alter it.
  ADCON1 = 0x7F;  //all IO are digital  or 0b01111111 in binary
  TRISA = 0b11111111;   //sets PORTA as all inputs
  PORTA = 0b00000000;  //turns off PORTA outputs
  TRISB = 0b00000000;   //sets PORTB as all outputs
  PORTB = 0b00000000;  //turns off PORTB outputs, good start position

while (1)
 {
        if (PORTAbits.RA0==0)   PORTB=0b00000001;

        if (PORTAbits.RA1==0)   PORTB=0b00000011;

        if (PORTAbits.RA2==0)   PORTB=0b00000111;

        if (PORTAbits.RA3==0)   PORTB=0b00001111;
 }
}
```

FIGURE 4.3 inputs.c code.

There are several example programs we could write here. Suppose we use SW0 to turn on LED0; SW1 turns on LED0 and LED1; SW2 turns on LED0, LED1, and LED2; and SW3 turns on LED0, LED1, LED2, and LED3.

Open MPLAB, Open the file header.c and add the code lines 18–24 as shown in Figure 4.3 and put the correct title in line1 then save file as **inputs.c**

Make a new project called inputs.

Add file inputs.c to the inputs project.

Notice the if statements lines 18–24 do not have any {} brackets with them. The lines shown are a shortcut because there is only one outcome for each if statement and it is written on the same line.

Can you alter the code so that when a switch is pressed the corresponding LED flashes a number of times. Give this program a name, say inputs2.c

Writing your own programs will aid your understanding of the topic. Have a play with inputs and LEDs, there are numerous programs you can write.

Let's continue our discussion of inputs by considering the use of the keypad.