Chapter 13

EEPROM Data Memory

In previous examples we have stored data in user files called variables such as TEMPERATURE. The reading of the temperature would have been lost if the power was switched off to the circuit. However if we store the data in the EEPROM Data Memory then the information is retained in the micro when the power is switched off or lost.

The 18F1220 has 256 bytes of EEPROM data memory and the 18(L) F26K22 has 1024 bytes.

Access enabling Reading and Writing to and from the EEPROM is via the four registers:

- EECON1
- EECON2
- EEDATA
- EEADR

When reading we identify the address from 0 to 0xFF (for the 18F1220) using the address register EEADR. The data is then available in register EEDATA. When writing to the EEPROM data memory we specify the data in the register EEDATA and the location in the register EEADR.

Two other files are used to enable the process, they are EECON1 and EECON2, two EEPROM control registers.

The Register EECON1 is shown below in Figure 13.1.

Bit 0: RD is set to a 1 to perform a read. It is cleared by the micro when the read is finished.

Bit 1: WR is set to a 1 to perform a write. It is cleared by the micro when the write is finished.

Bit 2: WREN, WRite ENable a 1 allows the write cycle, a 0 prohibits it.

Bit 3: WRERR reads a 1 if a write is not completed, reads a 0 if the write is completed successfully.

Bit 4: FREE, Flash Row Erase Bit, 1=Erase the program memory row addressed by TBLPTR. 0=perform write only.

Bit 7: EEPGD, Program/Data EEPROM Select Bit. This bit allows either the program memory or the data memory to be selected. 0 selects Data, 1 selects Program Memory.

EEPGD	CFGS	-	FREE	WRERR	WREN	WR	RD
bit 7							bit 0

FIGURE 13.1 The EECON1 register.

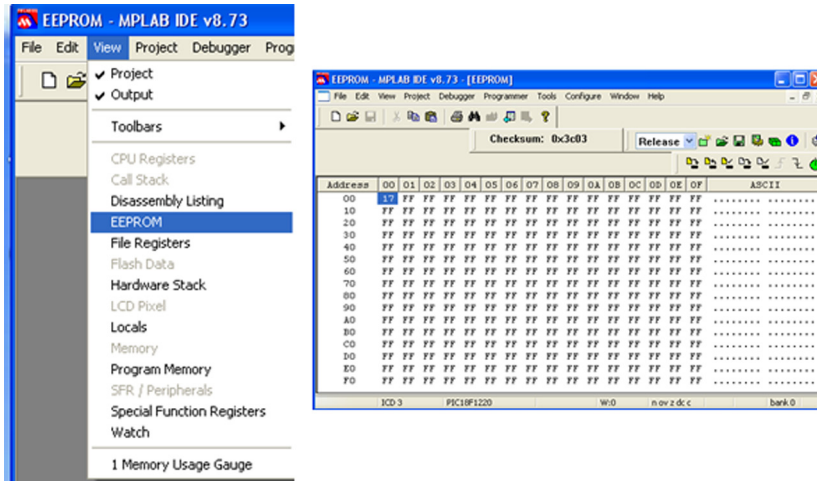

FIGURE 13.2 Selecting EEPROM data memory.

Data can be initially programmed into the EEPROM data memory from MPLAB using View, EEPROM, as shown in Figure 13.2 and writing the hex data into the relevant EEPROM address.

In order to see how to write and read from the EEPROM memory consider the example where we count the number of times a switch is thrown and display the result on eight LEDs connected to PORTB. The circuit for this application is shown in Figure 13.3.

Suppose we set the data initially to 0×17 in EEPROM Address 0 with MPLAB in order to start counting from there.

The program to show this count and store it in EEPROM address 0 is shown in **EEPROM.C**:

```
1.    //EEPROM.C by DW Smith 11-6-12
2.    #include <p18f1220.h>
3.    #pragma config WDT=OFF, OSC=INTIO2, PWRT=ON, LVP=OFF, MCLRE=OFF
4.    #include <delays.h>

5.    // put variables here
6.    unsigned char COUNT=0;

7.    void Read(void)
8.    {
9.    EEADR=0;          //points to EE Address 0
```

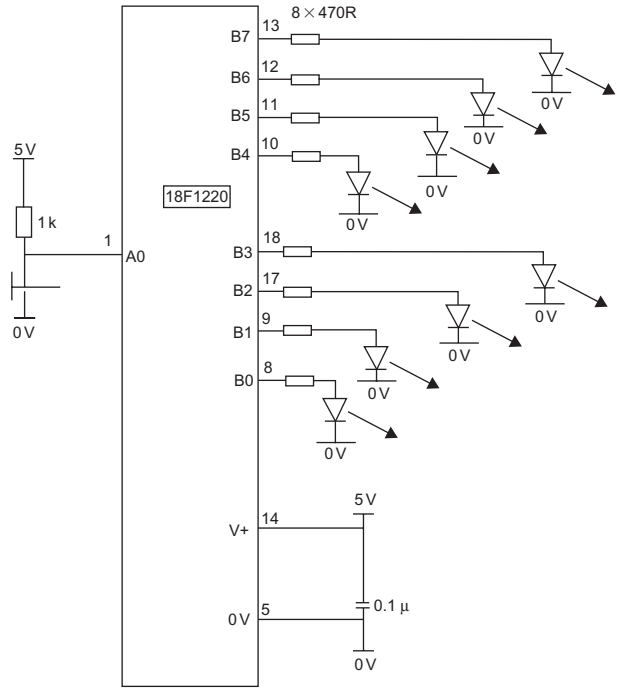

FIGURE 13.3 Counting switch presses.

```
10.   EECON1bits.EEPGD=0;                  //point to DATA memory
11.   EECON1bits.RD=1;                     //do EEPROM read
12.   COUNT=EEDATA;                        //read DATA into COUNT.
13.   EECON1bits.RD=0;                     //reset read
14.   }

15.   void Write(void)
16.   {
17.   EECON1bits.EEPGD=0;                  //point to DATA memory
18.   EECON1bits.WREN =1;                  //enable write
19.   EEDATA=COUNT;                        //move COUNT to EEDATA
20.   EECON2=0×55;
21.   EECON2=0xAA;
22.   EECON1bits.WR=1;                     //do EEPROM write
23.   while (EECON1bits.WR == 1);          //wait until WR=0, write completed.
24.   EECON1bits.WREN =0;                  //disable write
25.   }

26.   void main (void)
27.   {
28.   //SET UP
29.   // OSCCON defaults to 31 kHz. So no need to alter it. 128 ms timing
30.   ADCON1=0x7F;                         //all IO are digital or 0b01111111 in binary
31.   TRISA=0b11111111;                    //sets PORTA as all inputs
```

```
32.   PORTA=0b00000000;          //turns off PORTA outputs, not required, no outputs
33.   TRISB=0b00000000;          //sets PORTB as all outputs
34.   PORTB=0b00000000;          //turns off PORTB outputs, good start position

35.   Read();
36.   PORTB=COUNT;

37.   while(1)
38.     {
39.   while (PORTAbits.RA0==1);
40.   Delay10TCYx(78);           //0.1 s wait for bounce to stop.
41.   while (PORTAbits.RA0==0);
42.   Delay10TCYx(78);           //0.1 s
43.   COUNT=COUNT+1;
44.   PORTB=COUNT;
45.   Write();
46.       }
47.   }
```

- Lines 7–14 read data from EEPROM address 0. Line 9 sets the EEPROM address to 0.
- Lines 15–25 are the Write routine to write EEDATA into EEADR. The Write bit WR cannot be set if the WREN bit is clear.
- Lines 20–21 write 2 bytes of data to EECON2 to execute the write cycle.
- Line 35 reads the data from address 0 into COUNT and line 36 sends it to the output.
- Lines 37–46 repeatedly count switch presses and store the data.
- Lines 39–42 wait for the switch to be pressed and released with delays for the bouncing to stop.
- Line 43 increments the count.
- Line 44 moves the count to the output, PORTB.
- Line 45 Writes COUNT to EEADR 0, to save it.

When the program is run it starts the count at 0×17, after several switch presses the power can be removed. When the power is reconnected the program runs from the last count.