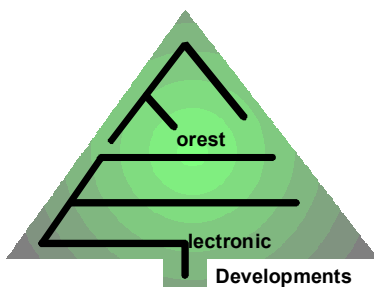


The FED PIC Development Board

THE FED Development board offers a host for 40 pin devices and includes interfaces to LCD, Keypad, LED's, serial ports and includes an In Circuit Programmer

This manual is copyright (C) Forest Electronic Developments 2000. It may not be copied, transmitted to 3rd parties in any form, or altered without express permission of Forest Electronic Developments. Forest Electronic Developments can accept no responsibility for the consequences of any errors or omissions in this introductory manual



Forest Electronic Developments

60 Walkford Road
Christchurch
Dorset
BH23 5QG
Sales : +44 - (0)1425 - 274068

info@fored.co.uk

Or see the **Forest Electronic Developments** home page on the world wide web at the following URL:

<http://www.foresd.co.uk>

Contents

THE FED PIC DEVELOPMENT BOARD		1
CONTENTS		2
1 16F877 DEVELOPMENT BOARD		3
1.2 Table 1 - Connections for board:	8	
1.3 Component List	9	
2 EXAMPLE PROGRAMS		11
2.1 LED's	11	
2.2 LCD	11	
2.3 FullDemo	11	
3 USING THE IN-CIRCUIT PROGRAMMER		12
3.1 Power Supply	12	
3.2 Serial cable	12	
3.3 WINDOWS installation	12	
3.4 Using the programmer	12	
3.5 Programming when an application is running	12	
3.6 In Circuit Programmer Menu Reference	13	
4 BUILDING THE KIT		15
5 COMPLETE DEMONSTRATION PROGRAMME		17
6 RUNNING BASIC ON THE BOARD		22

1 40 pin PIC (including 16F877) development board

FED have a capable development/evaluation board for 40 pin devices which is suitable for use with all the examples and the tutorial and full example program. The board has the following features:

- Supports all 40 pin 16 and 18 series PIC's (examples use 16F877).
- Will run FED PIC BASIC interpreter/development system (supplied with CD-ROM)
- Has on board 5V, 1A, regulator
- 2 Serial interfaces for standard 3 wire serial communications connected to PIC serial interface and CCP pins
- Supports 8 pin EEPROM/RAM devices with IIC interface
- 20MHz crystal oscillator
- 32 I/O pins (all except E2) available on standard IDC connectors
- Connector for LCD module - 1:1 pin out including brightness control
- Key pad connector, resistors and diodes for 4x4 hex keypad
- Full In Circuit Serial programmer - connect to the PC and program hex files direct to any of the 40 pin PIC's in the 16Cxxx, 16Fxxx and 18Cxxx series.
- 4 LED's
- LCD, Hex keypad and LED's all connect to PORTD/PORTE leaving all other ports free for other I/O functions

1.1.1 Circuit Diagram

The circuit diagram is shown in figure 1. The peripherals are mainly connected to Ports D and E, all of ports A,B,C and D are available on IDC connectors, and bits E0 and E1 are also available on external connectors.

1.1.1.1 Connections

The boards external connectors are shown in table 1.

1.1.1.2 Power Supply

The power supply is from a 1A, 5V regulator. There is no heat sink supplied, and applications using more than 200mA MUST provide an external heat sink for the voltage regulator. The board may be driven by power supplies of 9V or greater, however if the In-Circuit programmer is required then the supply should be greater than 17V - cheap unregulated 12V battery eliminators work very well in this case.

The board may be driven directly from a 5V supply if D3 and IC4 are removed and replaced by wire links, in-circuit programming will not work in this case.

1.1.1.3 Input & Outputs

+5V Power and 32 of the PIC I/O's are available on 20 pin IDC connectors J3 and J4 as shown in table 1. The only PIC I/O not available is port E, bit 2. If required this may taken by a flying lead from one end of R5.

1.1.1.4 Running FED PIC BASIC

If you wish to run FED PIC BASIC then SKT1 should hold a 24LC65 EEPROM. The F877 should be programmed with BASIC and the board can be connected to a PC on J2. Consult the later section in this manual for further details.

1.1.1.5 IIC EEPROM/RAM

SKT 1 will hold an 8 pin EEPROM or RAM device with an IIC interface. The EEPROM programmer application shown in the FED PIC C manual and also in the WIZPIC manual may be made to operate with these devices. Note that it may only be driven by a software IIC interface.

1.1.1.6 LED's

The LED's are driven from bits D4 to D7, they are all turned on or off from bit E2 to TR2. When the LCD is being driven, or the keypad is read then bit E2 must be turned off. Here is an example to turn on LED 1 :

```

bsf STATUS,RP0
bcf STATUS,RP1
bcf TRISE,2           ; Turn on transistor drive
movlw 0x0f
andwf TRISD          ; Drive LED outputs
bcf STATUS,RP0
movlw 0x0f
andwf PORTD          ; Turn off all LED outputs
bsf PORTD,4          ; Turn on LED1
bsf PORTE,2          ; Turn on Drive transistor

```

Now whenever a routine to drive the LCD is called, or whenever a routine to read the Keypad is called the LED's must be turned off and turned on again after the keypad or display function.

In these examples there is a variable called LEDPat which holds the pattern of bits for Port D to turn on LED's (if bit 4 of LEDPat is high then LED 1 will be illuminated, bit 5 for LED 2 etc). The first function LEDOn turns on the LED's. The second LEDOff turns them off again. LEDOff should be called before all LCD or Keypad functions, LEDOn may be called afterwards. Normally Keypad and LCD accesses will be so quick that no effect will be noticed on the LED's.

```

LEDOn
    bcf STATUS,RP1
    bcf STATUS,RP0
    movlw 0x0f
    andwf PORTD          ; Turn off all LED drive bits

    bsf STATUS,RP0
    movlw 0x0f
    andwf TRISD          ; Make Upper bits of D outputs
    bcf TRISE,2          ; Drive Transistor
    bcf STATUS,RP0
    bsf PORTE,2          ; Turn on drive transistor
    movlw LEDPat
    iorwf PORTD          ; Turn on all requested LED's
    return

LEDOFF
    bcf STATUS,RP1
    bsf STATUS,RP0
    movlw 0xf0
    iorwf TRISD          ; Make LED pins inputs
    bcf STATUS,RP0
    bcf PORTE,2          ; Must turn off transistor
    return

```

1.1.1.7 LCD

The LCD is connected to CONN 2. This is a 14 pin connector - most LCD modules have 14 pin connectors and should be connected directly pin for pin with CONN2, only pins 1 to 6 and


```

Wait(1000);
for(i=15; i; i--)
{
    Wait(200);
    LCDShift(0,0);
}
LCDClear();
for(i=-5; i<=5; i++)
{
    LCDNum(i);
    Wait(512);
    LCDPrintAt(0,0);
}
LCDClear();
for(i=1; i<10000; i+=i)
{
    LCDNum(i);
    Wait(512);
    LCDPrintAt(0,0);
}
Wait(2000);
LCDClear();
LCDString("Complete");
while(1);
}

void LCDClear()
{
    LCD(0x108);    // Display off
    LCD(0x101);    // Clear display
    LCD(0x10C);    // Display on
    LCD(0x180);    // Print at 0,0
}

void LCDNum(int number)
{
    BYTE numflag=0;
    if (number<0) {LCD('-'); number=-number;}

    if (number>9999)
        {LCD(number/10000+'0'); number=number%10000; numflag=1;}
    if ((number>999) || numflag)
        {LCD(number/1000+'0'); number=number%1000; numflag=1;}
    if ((number>99) || numflag)
        {LCD(number/100+'0'); number=number%100; numflag=1;}
    if (number>9)
        {LCD(number/10+'0'); number=number%10;}
    LCD(number+'0');
}

```

1.1.1.8 Keypad

The keypad is a 4 x 4 matrix. It is driven from bits D0 to D3, and the inputs are read from bits D4 to D7 which are pulled up by 47K resistors. When a key is pressed the input will go low. Diodes are provided to ensure that if two keys are pressed together they will not short out pins of PORT D.

The keypad is compatible with the PIXE and WIZPIC keypad elements and the FED PIC C KeyScan() library function.

The keypad rows 1 to 4 should be connected to Port D outputs D0 to D3 (on CONN1 pins 5, 6, 7 and 8). The columns should be connected to PORT D inputs D4 to D7, columns 1 to 4 connect to CONN1 pins 1,2,3 and 4.

Here is the set up for the FED PIC C keyscan() function.

```

const int Row1Port=&PORTD;           // Set up keypad
const int Row2Port=&PORTD;
const int Row3Port=&PORTD;
const int Row4Port=&PORTD;

```

```

const int Col1Port=&PORTD;
const int Col2Port=&PORTD;
const int Col3Port=&PORTD;
const int Col4Port=&PORTD;
const int Row1Bit=0;
const int Row2Bit=1;
const int Row3Bit=2;
const int Row4Bit=3;
const int Col1Bit=4;
const int Col2Bit=5;
const int Col3Bit=6;
const int Col4Bit=7;

```

For PIXIE and WIZPIC the keypad element can be connected to the correct pins of Port D using the application designer.

1.1.1.9 Variable resistor

The board includes space for a variable resistor which is connected to A/D converter input 0, and varies from 0 to 5V. The resistor should be removed if the A/D input is not required. This resistor is not supplied with the kit or board by FED.

1.1.1.10 Serial Port 1

Serial port 1 is connected to J2 and to the asynchronous serial hardware on the PIC port pins C6 and C7. Only Ground, Tx and Rx are connected to the 9 pin socket on the board. This hardware is compatible with the FED asynchronous interrupt driven serial communications library functions included with WIZPIC, PIXIE and FED PIC C. The port is fully powered and can be connected to a PC, or to another development board.

1.1.1.11 Serial Port 2

Serial port 2 is connected to J1. It is shared with the on board programmer which will not be enabled until a specific long byte sequence is received from the PC, and therefore it can be safely used when the unit is not being programmed by a PC.

It is connected by resistors to the CCP 1 and CCP 2 pins (Port pins C2 and C1) of the PIC. It is also connected to the In Circuit Serial Programming chip IC2. IC2 will hold its outputs in high impedance until it is enabled by a byte sequence from the port, it can be safely ignored if the application is using the serial port. IC2 and associated circuitry may be removed if the final application does not require ICSP.

Only Ground, Tx and Rx are connected to the 9 pin socket on the board.

The software serial port library routines may be used to drive this port (These are supplied with WIZPIC, PIXIE and FED PIC C). Alternatively interrupt driven communications may be used with the CCP functions.

1.2 Table 1 - Connections for board:

J1 - Serial Connector 1,
Programming+CCP

2	Tx Data (from board)
3	Rx Data (to board)
5	Ground

J2 - Serial Connector 2,
PIC Asynchronous serial port

2	Tx Data (from board)
3	Rx Data (to board)
5	Ground

Conn 1 - Keypad connector

1	Keypad column 0
2	Keypad column 1
3	Keypad column 2
4	Keypad column 3
5	Keypad row 0
6	Keypad row 1
7	Keypad row 2
8	Keypad row 3

Conn 2 - LCD Module

1	Ground	LCD Module pin 1
2	+5V	LCD Module pin 2
3	Contrast	LCD Module pin 3
4	RS	LCD Module pin 4
5	R/W	LCD Module pin 5
6	E	LCD Module pin 6
7	NC	
8	NC	
9	NC	
10	NC	
11	D4	LCD Module pin 11
12	D5	LCD Module pin 12
13	D6	LCD Module pin 13
14	D7	LCD Module pin 14

P1 - Power

1	Ground
2	NC
3	+9-30V

J3 - PIC Ports A, C & E

1	Ground
2	Ground
3	RA0
4	RC0
5	RA1
6	RC1
7	RA2
8	RC2
9	RA3
10	RC3
11	RA4
12	RC4
13	RA5
14	RC5
15	RE0
16	RC6
17	RE1
18	RC7
19	+5V
20	+5V

J4 - PIC Ports B & D

1	Ground
2	Ground
3	RB7
4	RD7
5	RB6
6	RD6
7	RB5
8	RD5
9	RB4
10	RD4
11	RB3
12	RD3
13	RB2
14	RD2
15	RB1
16	RD1
17	RB0
18	RD0
19	+5V
20	+5V

1.3 Component List

1.3.1 Development Board

C1,16	10uF (2)	RN1	47K (1)
C2,C11-14	100nF (5)	R1-4	4K7 (4)
C3,5,6,7	1uF (4)	R5,20,21	10K (3)
C9,10	22pF (2)	R6-9	300 (4)
CONN1	8 pin SIL (hex keypad)*	R10,14	1K (2)
CONN2	14 way SIL LCD module *	R13	100K
D3	1N4001	R16-17,22-23	1K5 (4)
D4,5,6,7	1N4148 (4)	LED1-4	Red LED (4)
IC1	MAX232	P1	2.1mm Power Socket
IC3	16F877, or other 40 pin PIC	XL1	20MHz
J2	9 pin D connector (serial)	U1	7805
J3	20 way IDC socket	VR1	1K (or 4K7)
J4	20 way IDC socket	VR2	100K *
SKT1	8 pin DIL socket (24LC65)		
TR2	BC548		

* Not supplied with Kit/Module

1.3.2 In-Circuit Serial Programming

C4,8	22pF (2)
C15	10uF
D1	11V Zener Diode
D2	1.2V Zener Diode, or 2x 1N4148 (see notes)
IC4	12V regulator 100mA
J1	9 pin D connector (serial)
R11	47R
R12	10K
R15	1K
TR1	BC559
XL2	4MHz

2 Example programs

Three example programs are provide on the CD in the Projects Sub-directory. The examples are all compiled to run on a 16F877 device at 20MHz. They are all written in FED PIC C and the source code and projects are included, however the hex files are also included, and these may be used to program a 16F877 directly without using PIC C.

2.1 LED's

This program simply flashes the 4 LED's in sequence, and is useful for checking the basic operation of the board.

2.2 LCD

This program operates with an LCD module connected to CONN2. It sends a series of text patterns to a 2 line display and is useful for checking out a display.

2.3 FullDemo

This project is described fully further in this manual. It provides a demonstration of the LED's, LCD, keypad, A/D converter and serial port all operating together in the same program. If no LCD is connected the program will stall.

- Every time that timer 0 overflows the analogue input on pin A0 is read and printed to the display as a value from 0 to 1023.
- If a key on the keypad is pressed then it is shown on the LCD display as a hex value from 0 to F. The key value is also transmitted on the serial port as a text string, and the current value on the analogue input pin A0 is also transmitted to the serial port as a text string. The port operates at 9600bps.
- If a byte is received on the serial port from the PC then the character is printed on the display.
- The LED's flash in a binary pattern and change every 63 overflows of Timer 0.

3 Using the in-circuit programmer

The programmer will handle most 40 pin PIC's including 16F87x, 16C7x, 16c6x and the 18Cxx series.

3.1 Power Supply

To use the programmer then the power supply to the module should be greater than 17 VDC and less than 28VDC at around 150mA. (Two 9v batteries or an unregulated battery eliminator work well). Use a 2.1mm DC power plug so that the centre terminal is positive. Wire a serial cable as shown below (or use a commercially available cable as the connections on the programmer are standard).

3.2 Serial cable

The programmer operates on connector J1 of the board. A standard commercial serial cable is the best

Programmer	PC end connector type :			
	9 way Male	9 way Female	25 way Female	25 way Male
PL101	3	2	3	2
2	3	2	3	2
3	2	3	2	3
5	5	5	7	7

3.3 WINDOWS installation

The host software is installed from Windows 3.1 or Windows '95/98 by running the program "SETUP.EXE" within the Programmer sub-directory of the supplied disk, or from the start menu on CD-ROM. The software may be installed in any directory on your hard disk, but defaults to installation in a directory called "C:\Program Files\FED\PIC Programmer". Once installed then double click the icon called PICPROG to start the program. Occasionally communication problems occur if the Windows 3.1 Installation is not correct. Check that COMnFIFO=True (where n is your comm port) line is in the [386Enh] section of the SYSTEM.INI file.

On power up LED 5 should flash three times to indicate that the programmer chip is working OK.

3.4 Using the programmer

The host software will not allow any operation with the programmer before it has detected that the programmer is present. The OPTIONS, COMMUNICATIONS menu option should be used to set the serial port parameters - set 19200bps, and the port number that you are using. If the host software is started before the programmer is connected or powered up, then use the OPTIONS, CHECK PROGRAMMER menu option to establish communication with the programmer.

3.5 Programming when an application is running

Occasionally when an application is running the programmer will be unable to clear an EEPROM chip, in this case simply power the circuit down and up again and the programmer should clear it successfully. Alternatively use the programmer menu option **PIC | Erase EEPROM**.

3.6 In Circuit Programmer Menu Reference

File Menu

The file menu consists of the following items:

Load	This command loads a file which is in any of the formats supported by the PIC programmer. The file is loaded to the Buffer and then may be saved or downloaded to the programmer.
Save As	This command saves the Buffer to a file selected by the user.
Clear Buffer	This command clears the Buffer, the EEPROM buffer, and sets the PIC fuses to their default values.
Fuses/ID locations	This command brings up the FUSE/ID Dialog box, allowing the user to configure the fuses and ID type of the PIC to be programmed, verified or read.
PRID	This command brings up the PRID Dialog box which allows the user to set a PRID area in the PIC program.

PIC Menu

The PIC menu contains those items which affect the PIC. Most of them will be disabled if the program running on the host PC is unable to detect a programmer on the configured serial link. If the program has been unable to detect a programmer which is subsequently connected or turned on, then the Check Programmer command of the Options menu should be used to force the program to look for the programmer. Please note that for the DOS version some of these items are under the Program menu. The command items are as follows:

Read (F4 key)	This command reads the memory contents of the PIC, and writes them to the Buffer.
Blank Check (F5 key)	This command checks whether the PIC which is currently installed into the programmer is blank - that is all locations are set to FFF Hex (or 3FFF Hex for 14 bit devices).
Verify	This command compares the program held in the PIC which is currently installed in the programmer with the contents of the Buffer, and the currently set Fuse and User ID locations, and reports if they are the same
Checksum PIC	This command calculates and displays the checksum of the PIC currently installed in the programmer. This is useful for comparison with the value displayed for the current Buffer.
Program PIC(F2 key)	This command programs the PIC with the contents of the Buffer. If the PIC is not blank a warning is displayed before programming begins. If the PIC fuses do not match the fuse configuration being programmed for a

	16C5X series device then a warning is also displayed before programming can begin. Note that for EEPROM devices the data memory is automatically programmed at the same time as the main program memory.
Program Fuses	This command just programs the FUSE word of the PIC currently installed in the programmer.
Program User ID	This command just programs the User ID locations of the PIC currently installed in the programmer.
Erase EEPROM (F3 key)	This command erases the program and user memory areas of a PIC which uses EEPROM technology (e.g. the PIC16C84). This command may be used to reset the code protect fuse of an EEPROM device.
Select PIC	This command allows the user to select the PIC to be used. It is essential that the correct PIC is selected before commencing any operation using the programmer.
Examine EEPROM	This command is only available for EEPROM devices. It switches the Buffer to show the contents of EEPROM data memory which may then be read or programmed in the same way as program memory. Note that the EEPROM data memory is programmed when the main program is written. Also when a program is loaded into the main buffer then if a data file was previously loaded into the EEPROM data memory at the same time as the main program then it will automatically be loaded again. This enables rapid programming of EEPROM main and data memory.

Options Menu

Communications	This command brings up a dialog box which allows the user to set the communications port (from 1 to 4), and the bit rate to be used with the programmer. This information is automatically saved when the program exits. The bit rate should normally be set to 9600.
Check Programmer	This command checks that the programmer is present. If the programmer is not present then an error is displayed and few of the PIC menu items will be enabled. If the programmer is connected to the PC, or turned on after the main host program is run then the Check Programmer command should be used to enable all the functions of the program.

3.6.1 Other Programmer Information

3.6.1.1 Other operational details

The host software is fairly self explanatory in operation and the Windows version includes help facilities (press F1). The PIC, SELECT PIC menu option allows the PIC device to be selected. Use the down/up arrow keys, or hold the mouse button down and push it off the top or bottom of the box to display the full range of PIC's. The FILE, LOAD menu option is used to load a file. The MPASM program assembles to an Intel hex format file with an extension of .HEX. The fuses should be set to the required values and then the PIC, PROGRAM menu option may be used to program the device. The other menu options allow devices to be read and verified, and allow fuses, PRID and user locations to be set and programmed.

3.6.1.2 Upgrading the programmer

To upgrade the programmer to handle new device types please contact Forest Electronic Developments.

3.6.1.3 PIC Information

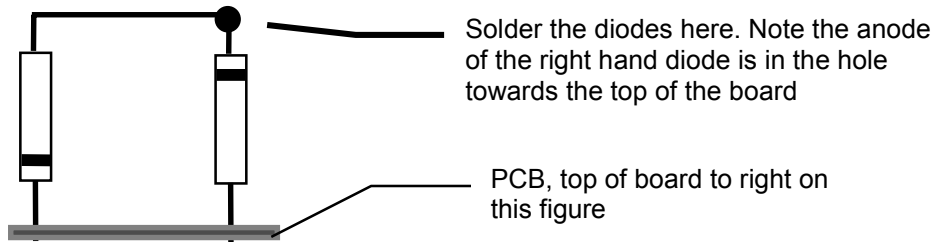
The microchip PIC devices are well documented in the MICROCHIP data book which is available from Maplin. The data book includes full descriptions of the operation of the devices, and their instruction sets. Data sheets, and the latest versions of the assembler and simulator can be downloaded from the MICROCHIP bulletin board, this can be contacted by using a terminal program with 8 bits, no parity, one stop bit. The access number is 0171-490-8881, then press return and in response to the prompt HOST type MCHIPBBS. Alternatively use the MICROCHIP web pages which can be obtained at URL <http://www.microchip.com/>.

4 Building the kit

If you have the kit version then please read the following instructions carefully.

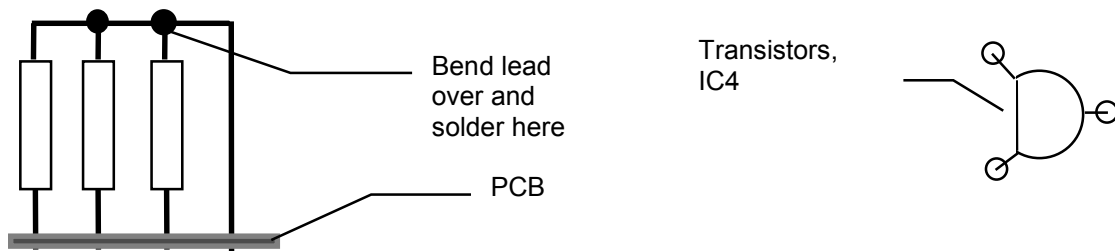
The component list is shown in table 1. There are two lists, one for the main board, and one for the ICSP components. Check that you have all the components on the list and identify them before you start construction. Note that VR2, Conn1 and Conn2 are not supplied with the kit as they are a matter for the specific devices connected by the user.

D2 is a 1.2V zener diode, it may be replaced by two 1N4148 diodes in the kit connected as shown below:



There are no wire links on the programmer. Insert and solder the horizontally inserted resistors and diodes first. The IC sockets are next.

Insert RN1, this is the resistor network which is mounted to the right of IC3, Pin 1 is labelled with a square on the component outline. RN1 may be supplied as a resistor network, or as 4 discrete resistors. In the latter case insert the bottom resistor first, bend the lead upwards across the holes for the other three resistors and into the common hole, solder. Insert the other resistors and solder to the common lead. See the figure below:



Insert and solder VR1, the capacitors and crystal(s) (Note that XL2 for the programmer may be a ceramic resonator), be sure to get the polarity of the electrolytic capacitors correct. Insert and solder the LED's which mount flush with the board, the flat on all 5 LED's is towards the bottom of the board. LED 5 is only needed for the programmer. Insert and solder the transistors and IC2 and IC3 which are mounted as shown in the diagram above.

U1 is the power regulator. It should be mounted to a heatsink if peripheral devices will consume too much current. The IDC connectors (J3 and J4) are mounted with the keyslot to the right of the board.

Finally mount the remaining components and connectors. Check your work, particularly for short circuits and dry joints.

Before inserting any of the IC's connect the module to the power supply, consult the circuit diagram and check that the voltage on the power pins of IC1 is correct (at 5VDC +/- 0.25V). If using the programmer then check the voltage on the output terminal of IC4, on R11 say, it should be 13.5V +/- 0.5V. Power down, insert the IC's and then the board is ready for use.

05 December 2000

WARNING - initial programming for all device types should be performed with erasable devices in case of problems with the kit construction.

5 Board

Please see the picture below:



6 Complete demonstration programme

The following programme is written in FED PIC C and is included in the Projects sub-directory of the CD ROM under the FULLDEMO directory.

- Every time that timer 0 overflows the analogue input on pin A0 is read and printed to the display.
- If a key on the keypad is pressed then it is shown on the LCD display as a hex value from 0 to F. The key value is also transmitted on the serial port as a text string, and the current value on the analogue input pin A0 is also transmitted to the serial port as a text string. The port operates at 9600bps.
- If a byte is received on the serial port from the PC then the character is printed on the display.
- The LED's flash in a binary pattern and change every 63 overflows of Timer 0.

Note how LED's are disabled every time the LCD is used or that the keypad is read. The program and hex file are included on the CD ROM.

It is an interesting exercise to re-write this program using PIXIE which considerably reduces the amount of user effort.

```

#include <P16F877.h>
#include <DataLib.h>
#include <Delays.h>
#include <Displays.h>
#include <Strings.h>

const int LCDPORT=&PORTD; // Connect LCD module to port D
const int LCDEPORT=&PORTE;
const int LCDEBIT=1;
const int LCDRSPORT=&PORTD;
const int LCDRSBIT=2;
const int LCDRWPORT=&PORTD;
const int LCDRWBIT=3;

const int Row1Port=&PORTD; // Set up keypad
const int Row2Port=&PORTD;
const int Row3Port=&PORTD;
const int Row4Port=&PORTD;
const int Col1Port=&PORTD;
const int Col2Port=&PORTD;
const int Col3Port=&PORTD;
const int Col4Port=&PORTD;
const int Row1Bit=0;
const int Row2Bit=1;
const int Row3Bit=2;
const int Row4Bit=3;
const int Col1Bit=4;
const int Col2Bit=5;
const int Col3Bit=6;
const int Col4Bit=7;
const int KeyPadRelease=0;

#define LCDPrintAt(x,y) LCD(0x100+0x80+x+y*0x40)
#define LCDOnOff(display,cursor,blink) LCD(0x108+display*4+cursor*2+blink)
#define LCDShift(cursor,right) LCD(0x110+(!cursor)*8+right*4)

const int TXBUFSZ=8;
const int RXBUFSZ=8; // Must be at least 32 bytes if used at full rate
const int SERINTRATE=9600;
const int USEXON=1; // Use XON/XOFF signalling
unsigned char TxTab[TXBUFSZ];
unsigned char RxTab[RXBUFSZ];

//
// Functions
//
void DisableLEDs(); // Disable LED's
void DoAToD(); // Scan A/D port, print result
void DoCheckRx(); // Check received chars from serial port
void TDoKeyScan(); // Scan the keyboard and display
void DoLED(); // Next LED
void EnableLEDs(); // Enable LED's
void LCDClear(); // Clear display
void main();
void WriteSerString(char ram *ws); // Write string to serial i/f

//
// Global variables
//
int ADRes; // Analogue conversion result
BYTE ActionFlags; // useful flags byte
BYTE Key; // Value of key read from hex keypad
BYTE LED; // LED pattern
BYTE T0Count; // Counts T0 overflows
char ws[17]; // String big enough for print to line

//
// Values for Flags
//
enum FlagValues {SendVolts=1};

void main()
{
    ADCON0=0xC1; // A/D converter enabled on channel 0 (RA0)

```

```

ADCON1=0x84;          // A/D converter input on RA0,RA1,RA3, ADFM=1
LCD(-2);             // 2 line display

LCDClear();          // Various demonstrations of module capability
LCDString("FED PIC C");
LCDPrintAt(0,1);
LCDString("Development Kit");
OPTION_REG=7;        // Overflow every 13mS
Wait(2000);
LCDClear();          // Clear LCD Module
SerIntInit();        // Interrupt driven Serial port
AddTx('K');          // Transmit a 'K' to tell system that we're here

while(1)             // Main loop
{
  if (INTCON&(1<<T0IF)) // Test for Timer 0 overflow
  {
    T0Count++;
    INTCON&=~(1<<T0IF); // Clear T0 flag
    DoAToD();
    TDoKeyScan();
    if (!(T0Count&63)) DoLED();
  }
  DoCheckRx();
}

//
// Interrupt handler - used only for interrupt driven serial port
//

const int QuickInt=1; // Quick interrupts

void Interrupt() // Interrupt handler
{
  SerIntHandler();
}

void LCDClear()
{
  LCD(0x108); // Display off
  LCD(0x101); // Clear display
  LCD(0x10C); // Display on
  LCD(0x180); // Print at 0,0
}

//
// Do an A/D conversion - display result on 1st line at 0,0
//

void DoAToD()
{
  ADCON0|= (1<<GO); // Start A/D conversion
  while(ADCON0&(1<<GO)); // Wait for end of conversion
  ADRes=((int)ADRESH<<8)+ADRESL; // 10 bit result
  riPrtString(ws,ADRes); // Print to ws
  DisableLEDs();
  LCDPrintAt(0,0); LCDString(ws); LCDString(" "); // Print Result
  if (ActionFlags&SendVolts)
  {
    WriteSerString(ws); // Write value to
    ActionFlags&=~SendVolts;
  }
  EnableLEDs();
}

//
// Write a string to the serial interface
//
void WriteSerString(char ram *ws)
{
  char ram *wsp=ws;
  while(*wsp) AddTx(*wsp++);
}

//
// Scan the keypad - display key at line 1, position 0

```

```

//

void TDoKeyScan()
{
  DisableLEDs();
  BYTE NewKey=KeyScan();
  BYTE Asc;

  if (NewKey==Key || NewKey==0xff) {EnableLEDs(); return;}
  Key=NewKey;
  Asc=Key+'0';
  if (Asc>'9') Asc+='A'-'9'-1; //Hex display
  LCDPrintAt(0,1); LCD(Asc);
  AddTx('\r'); AddTx('\n'); AddTx(Asc); AddTx(' '); // New line & key value
  ActionFlags|=SendVolts;
  EnableLEDs();
}

//
// Read from serial port - display characters at 0,7
//

void DoCheckRx()
{
  if (!GetRxSize()) return;
  DisableLEDs();
  LCDPrintAt(8,0);
  LCD(WaitRx());
  EnableLEDs();
}

//
// Note - as keyscan and LCD drivers upset the values of PORTD then we
// must
// Keep a record of the LED's being driven and enable/disable them around
// LCD & Keypad accesses
//
void DoLED()
{
  LED++; // LED's count in binary
  EnableLEDs();
}

void EnableLEDs()
{
  PORTD&=0x0f; // All LED's off
  TRISD&=0x0f; // Drive upper 4 bits of PORTD
  PORTD|=(LED<<4); // Move LED pattern to upper 4 bits of PORT D
  TRISE&=~(1<<2); // Now turn on transistor
  PORTE|=(1<<2);
}

void DisableLEDs()
{
  PORTE&=~(1<<2); // Transistor off
  TRISD|=0xf0; // PORT D drivers to read
}

```

7 Running BASIC on the board

7.1.1 FED PIC BASIC

FED PIC BASIC is provided on the CD ROM in the BASIC sub-directory. The document "Introduction to FED PIC BASIC.pdf" provides full details of the system. The document "BASIC Compiler Info.pdf" adds to the previous document to describe the compiler. Both the interpreted and compiled versions are provided.

7.1.2 Installing PIC BASIC.

PIC BASIC is supplied on the CD ROM in the same form as on floppy disks. Either install from the main menu, or run SETUP.EXE on the CD ROM in the directory \BASIC\INSTALL\Disk1. When the installation program prompts for disk 2 then simply change the name DISK1 to DISK2 in the dialog box.

7.1.3 Programming the PIC for the interpreter.

BASIC will run with the 16C74 or 16F877 chips at either 4MHz or 20MHz only. Hex files are provided for each which must be programmed into a chip using the development board.

In the "Hex" subdirectory of the BASIC directory on CD the following files are provided:

File	Purpose	Set the fuse values:
74_04.hex	16C74, 4MHz crystal	XT oscillator, no watchdog
74_20.hex	16C74, 20MHz crystal	HS oscillator, no watchdog
877_04.hex	16F877, 4MHz crystal	XT oscillator, no watchdog, Low voltage programming disabled
877_20.hex	16F877, 20MHz crystal	HS oscillator, no watchdog, Low voltage programming disabled

The fuse values must be checked in the programmer.

7.1.4 Running BASIC

Please ensure that a 24LC65 EEPROM is inserted into SKT 1 of the development board to run the interpreted version of the program.

7.1.5 Connecting to the PC

For the development board connection to the PC use J1 (the upper socket) for the programmer, and J2 (the lower socket) for communication between the BASIC interpreter on the PC and the development board.

7.1.6 Running compiled BASIC

Compiled BASIC will only run with the 16C74. It is possible to run a program compiled for the 16C74 on the 16F877 provided that variables are limited to a total size of 52 bytes. A compiled BASIC program is in hex file format (a .hex extension) which may be used to program a 16C74 or 16F877 device on the development board.