Experiment No: 7

Title: Interfacing of LCD with PIC

Course Name:	**Microprocessor and Microcontrollers**	**Semester:**	**IV**
Date of Performance:	**22 / 04 / 2024**	**Batch No:**	**A - 2**
Faculty Name:	**Prof. Kirti Sawlani**	**Roll No:**	**16014022050**
Faculty Sign & Date:		**Grade/Marks:**	**__ / 25**

Aim and Objective of the Experiment:

Aim: Write a C Program to Interface 16X2 LCD with 8051 microcontroller using 8-BIT mode.

Objectives:

- Study of Port operation of 8051
- Study of LCD interfacing with microcontroller

COs to be achieved:

CO 3. Understand the internal design of 8051 microcontrollers along with its features
CO 4. Build applications using 8051 and various I/O devices

Theory:

- ***The Key features of the 8051 Microcontroller –***
- 4 KB on-chip ROM (Program memory).
- 128 bytes on-chip RAM (Data memory).
- The 8-bit data bus (bidirectional).
- 16-bit address bus (unidirectional).
- Two 16-bit timers.
- Instruction cycle of 1 microsecond with 12 MHz crystal.
- Four 8-bit input/output ports.
- 128 user-defined flags.
- Four register banks of 8 bit each.
- 16-byte bit-addressable RAM.
- The general purpose registers are 32 each is 8-bit.
- 8051 has two external and three internal interrupts.
- 8051 microcontroller specifies some special function features like UARTs, ADC, Op-amp, etc.
- It has a 16-bit program counter and data pointer.

As it was designed to perform simple control applications thus called peripheral interface controller. But now-a-days it is commonly known as programmable intelligent computer.

A 16×2 LCD means it can display 16 characters per line and there are 2 such lines. In this LCD each character is displayed in 5×7 pixel matrix. This LCD has two registers, namely, Command and Data. The command register stores the command instructions given to the LCD. A command is an instruction

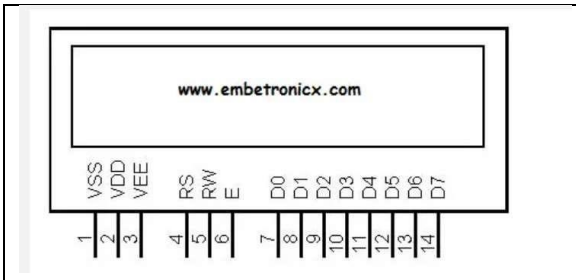
given to LCD to do a predefined task like initializing it, clearing its screen, setting the cursor position, controlling display etc. The data register stores the data to be displayed on the LCD. The data is the ASCII value of the character to be displayed on the LCD.

Code (Hex)	Command to LCD Instruction Register
1	Clear Display screen
2	Return home
4	Decrement cursor (Shift cursor to left)
6	Increment cursor (Shift cursor to Right)
5	Shift display right
7	Shift display left
8	Display off, cursor off
A	Display on, cursor off
C	Display on, cursor off
E	Display on, cursor blinking
F	Display on, cursor blinking
10	Shift cursor position to left
14	Shift cursor position to right
18	Shift the entire display to the left
1C	Shift the entire display to the right
80	Force cursor to beginning of 1st line
0C0	Force cursor to beginning of 2nd line

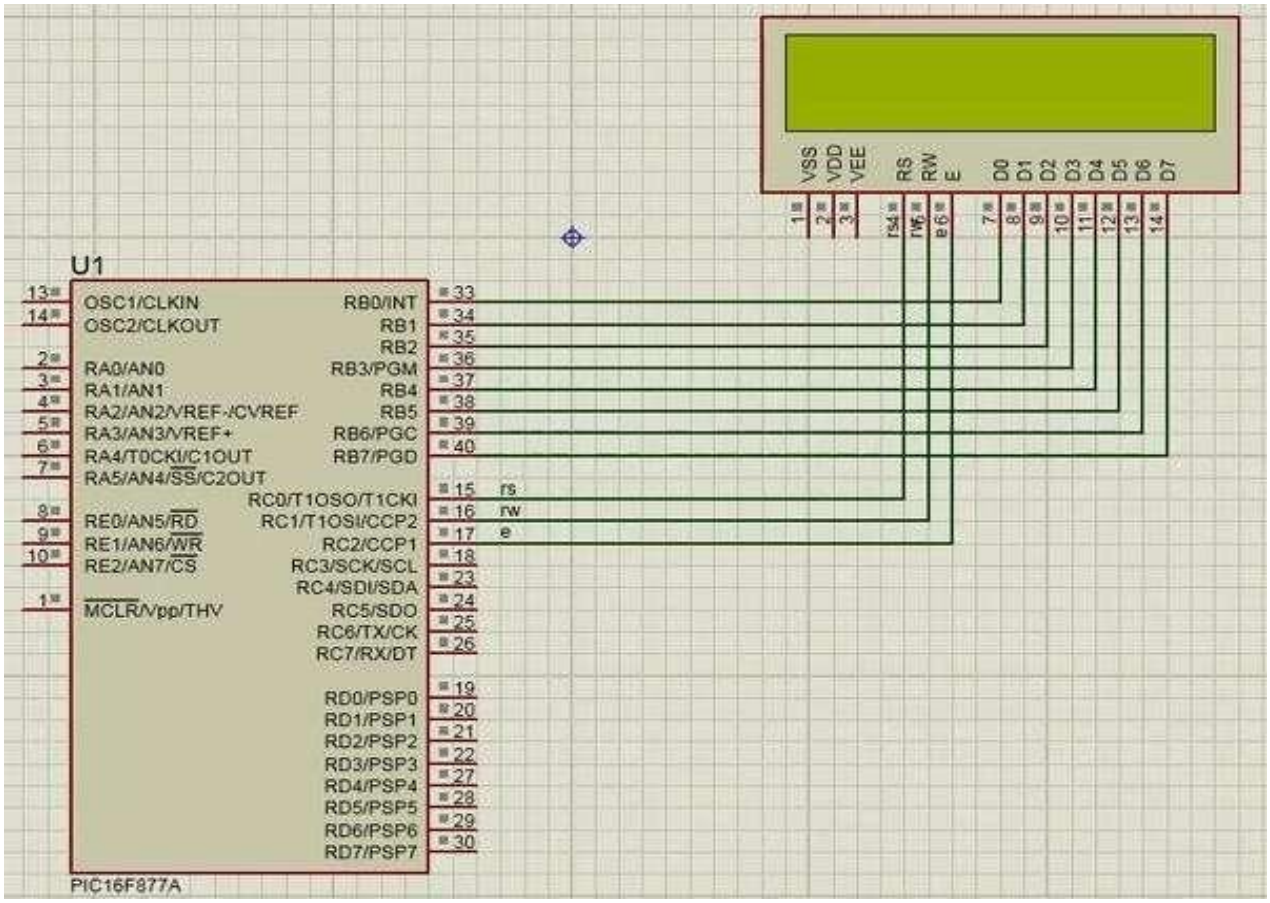
38 5x7 matrix

- **RS** – Port C .0 (RC0)
- **RW** – Port C.1 (RC1)
- **EN** – Port C.2 (RC2)

- **Data lines** – Port B



Circuit Diagram/ Block Diagram:



Stepwise-Procedure:

1. Design circuit and connect it using Proteus simulator.
2. Write an assembly program to achieve the aim.
3. Compile the program and generate HEX file using MPLAB IDE.
4. Run the hardware and take screen shot of it to attach in the output.

Assembly program:

```
sfr lcd_data_port=0x90;     /* P1 port as data port */
sbit rs=P2^0;               /* Register select pin */
sbit rw=P2^1;               /* Read/Write pin */
sbit en=P2^2;               /* Enable pin */

void delay(unsigned int count) {
      int i,j;
      for(i=0;i<count;i++)
      for(j=0;j<112;j++);
}

void LCD_Command (unsigned char cmd) {
      lcd_data_port = cmd;
      rs=0;             /* command reg. */
      rw=0;             /* Write operation */
      en=1;
      delay(1);
      en=0;
      delay(5);
}

void LCD_Char (unsigned char char_data) {
      lcd_data_port=char_data;
      rs=1;             /* Data reg.*/
      rw=0;             /* Write operation*/
      en=1;
      delay(1);
      en=0;
      delay(5);
}

void LCD_String (unsigned char *str) {
      int i;
      for(i=0;str[i]!=0;i++) {
            LCD_Char (str[i]);  /* Call LCD data write */
      }
}

void LCD_Init (void) {
      delay(20);
      LCD_Command (0x38);
```

```
      LCD_Command (0x06);
      LCD_Command (0x01);
      LCD_Command (0x80);
}

void main() {
      LCD_Init();
      LCD_String("VRISHANK");
      LCD_Command(0xC0);
      LCD_String("WARRIER");
      while(1);

}
```

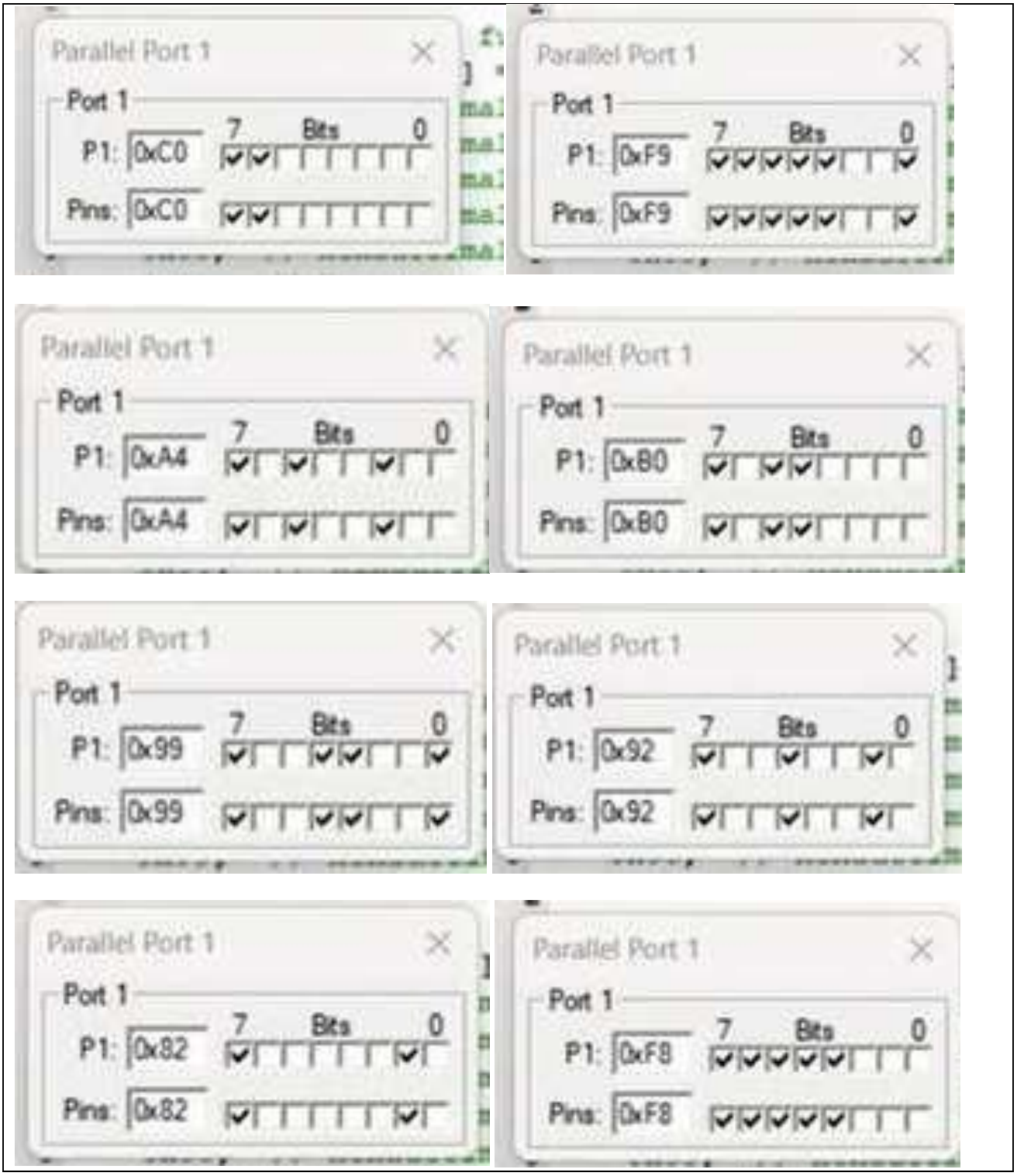
Observation Table/Output of program:

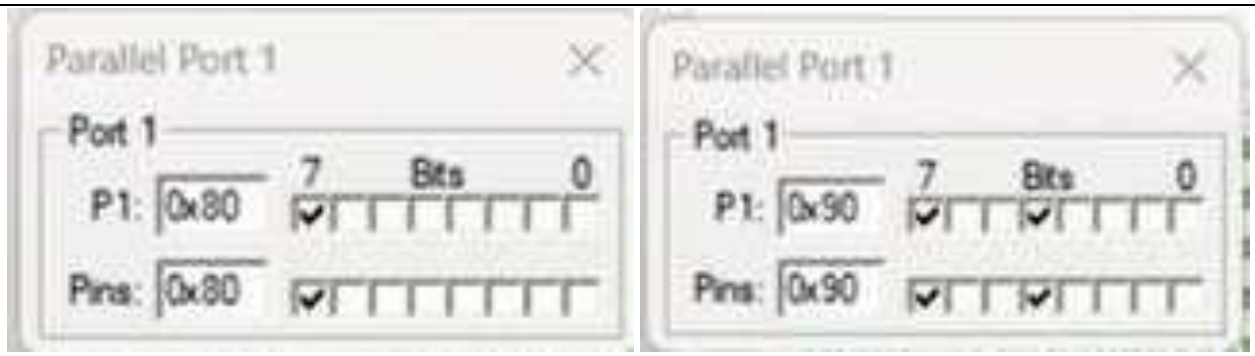
Post Lab Subjective/Objective type Questions:

Write a C Program to **interface keypad to 8051. (**Attach C file along with snapshot of result)

```
#include <reg51.h>  //Include header file for 8051 microcontroller to accessits register definitions and functions.

void main()// Main function where execution starts.
{
        // Array to hold segment patterns for digits 0 to 9 for a common cathode 7-segment display.
        unsigned char seg[10] = {
        0xc0,  // Hexadecimal pattern for displaying '0' on a 7-segment display.
        0xf9,  // Hexadecimal pattern for displaying '1' on a 7-segment display.
        0xa4,  // Hexadecimal pattern for displaying '2' on a 7-segment display.
        0xb0,  // Hexadecimal pattern for displaying '3' on a 7-segment display.
        0x99,  // Hexadecimal pattern for displaying '4' on a 7-segment display.
        0x92,  // Hexadecimal pattern for displaying '5' on a 7-segment display.
        0x82,  // Hexadecimal pattern for displaying '6' on a 7-segment display.
        0xf8,  // Hexadecimal pattern for displaying '7' on a 7-segment display.
        0x80,  // Hexadecimal pattern for displaying '8' on a 7-segment display.
        0x90    // Hexadecimal pattern for displaying '9' on a 7-segment display.
        };
        unsigned char x;          // Variable to loop through the array of segmentpatterns.
        unsigned int i;           // Variable for delay loop.P1 = 0X00;  // Initialize port P1 as output.
        while(1)  // Infinite loop to continuously display digits on the 7-segment display.
        {
                for(x = 0; x < 10; x++) // Loop through each digit in thesegment array.
                {
                        P1 = seg[x]; // Output the segment pattern for the currentdigit to port P1 to display it on the 7-segment display.
                        for (i = 0; i < 60000; i++);      // Delay loop to control thespeed of digit display.
                }
        }
}
```


Parallel Port 1
Port 1
P1: 0xC0
Pins: 0xC0
7 Bits 0
Parallel Port 1
Port 1
P1: 0xF9
Pins: 0xF9
7 Bits 0
Parallel Port 1
Port 1
P1: 0xA4
Pins: 0xA4
7 Bits 0
Parallel Port 1
Port 1
P1: 0x80
Pins: 0x80
7 Bits 0
Parallel Port 1
Port 1
P1: 0x99
Pins: 0x99
7 Bits 0
Parallel Port 1
Port 1
P1: 0x92
Pins: 0x92
7 Bits 0
Parallel Port 1
Port 1
P1: 0x82
Pins: 0x82
7 Bits 0
Parallel Port 1
Port 1
P1: 0xF8
Pins: 0xF8
7 Bits 0



Conclusion:

The experiment successfully interfaced a 16x2 LCD with an 8051microcontroller using 8-bit mode, achieving the objectives of studying port operation and LCD interfacing. This hands-on exercise deepened understanding of 8051's internal design and facilitated practical application development with various I/O devices.

Signature of faculty in-charge with Date: