Microprocessors and Microcontrollers (EE-231)

Lab-13

Objective

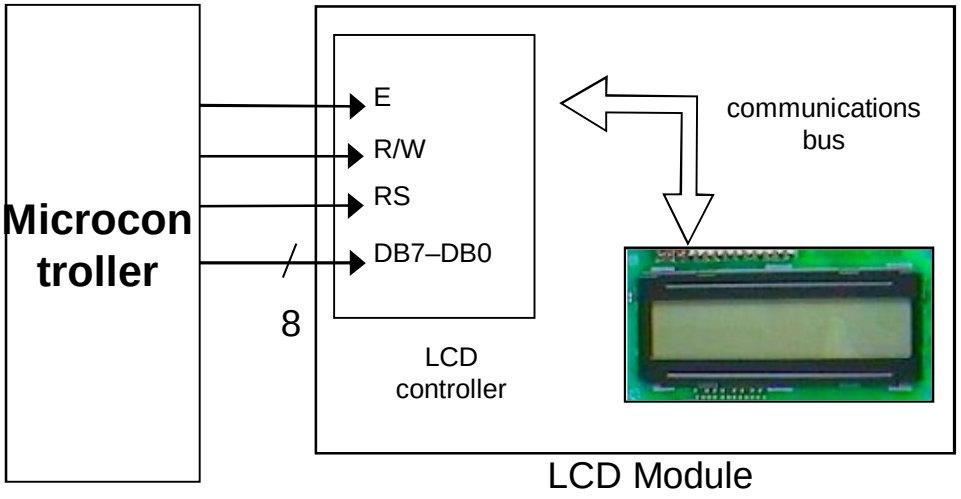
- LCD interfacing and Programming in C
  - LCDs Commands and Registers
  - LCD in 4-bit Mode

LCD Interfacing

- Liquid Crystal Displays (LCDs)
- cheap and easy way to display text
- Various configurations (1 line by 20 X char upto 8 lines X 80).
- Integrated controller(HD44780U)
- The display has two register
  - command register
  - data register
- By RS you can select register
- Data lines (DB7-DB0) used to transfer data and commands

Alphanumeric LCD Interfacing

- Pinout
  - 8 data pins D7:D0
  - RS: Data or Command Register Select
  - R/W: Read or Write
  - E: Enable (Latch data)



- RS – Register Select
  - RS = 0 → Command Register
  - RS = 1 → Data Register
- R/W = 0 → Write , R/W = 1 → Read
- E – Enable
  - Used to latch the data present on the data pins.

- D0 – D7
  - Bi-directional data/command pins.
  - Alphanumeric characters are sent in ASCII format.

LCD Commands

- The LCD's internal controller can accept several commands and modify the display accordingly. These commands would be things like:
  - Function Select
  - Display on/off
  - Decrement/Increment cursor

- After writing to the LCD, it takes some time for it to complete its internal operations. During this time, it will not accept any new commands or data.
  - We need to insert time delay between any two commands or data sent to LCD

Pin Description

Pin	Symbol	I/O	Descriptions
1	VSS	--	Ground
2	VCC	--	+5V power supply
3	VEE	--	Power supply to control contrast
4	RS	I	RS=0 to select command register, RS=1 to select data register
5	R/W	I	R/W=0 for write, R/W=1 for read
6	E	I/O	Enable
7	DB0	I/O	The 8-bit data bus
8	DB1	I/O	The 8-bit data bus
9	DB2	I/O	The 8-bit data bus
10	DB3	I/O	The 8-bit data bus
11	DB4	I/O	The 8-bit data bus
12	DB5	I/O	The 8-bit data bus
13	DB6	I/O	The 8-bit data bus
14	DB7	I/O	The 8-bit data bus

used by the LCD to latch information presented to its data bus

Commands

Function	LCD_RS	LCD_RW	Upper Nibble				Lower Nibble			
			DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
Clear Display	0	0	0	0	0	0	0	0	0	1
Return Cursor Home	0	0	0	0	0	0	0	0	1	-
Entry Mode Set	0	0	0	0	0	0	0	1	I/D	S
Display On/Off	0	0	0	0	0	0	1	D	C	B
Cursor and Display Shift	0	0	0	0	0	1	S/C	R/L	-	-
Function Set	0	0	0	0	1	0	1	0	-	-
Set CG RAM Address	0	0	0	1	A5	A4	A3	A2	A1	A0
Set DD RAM Address	0	0	1	A6	A5	A4	A3	A2	A1	A0
Read Busy Flag and Address	0	1	BF	A6	A5	A4	A3	A2	A1	A0
Write Data to CG RAM or DD RAM	1	0	D7	D6	D5	D4	D3	D2	D1	D0
Read Data from CG RAM or DD RAM	1	1	D7	D6	D5	D4	D3	D2	D1	D0

Command Codes

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 off, cursor on
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 to 1st line
C0	Force cursor to beginning to 2nd line
38	2 lines and 5x7 matrix

LCD Memory Map

- LCD has three types of internal storages

1. CG ROM
2. DD RAM
3. CG RAM

- **CG ROM:**
- The Character Generator ROM (CG ROM) contains the font bitmap for each of the predefined characters that the LCD screen can display.
- **CG RAM:**
- The Character Generator RAM (CG RAM) provides space to create eight custom character bitmaps. Each custom character location consists of a 5-dot by 8-line bitmap

LCD Memory Map

LCD Memory Map

- **DD RAM:**
- The Display Data RAM (DD RAM) stores the character code to be displayed on the screen. Most applications interact primarily with DD RAM. The character code stored in a DD RAM location references a specific character bitmap stored either in the predefined CG ROM character set or in the user-defined CG RAM character set.
- Physically, there are 80 total character locations in DD RAM with 40 characters available per line.
- After First 16 characters , more characters can only be displayed using controller's display shifting functions.

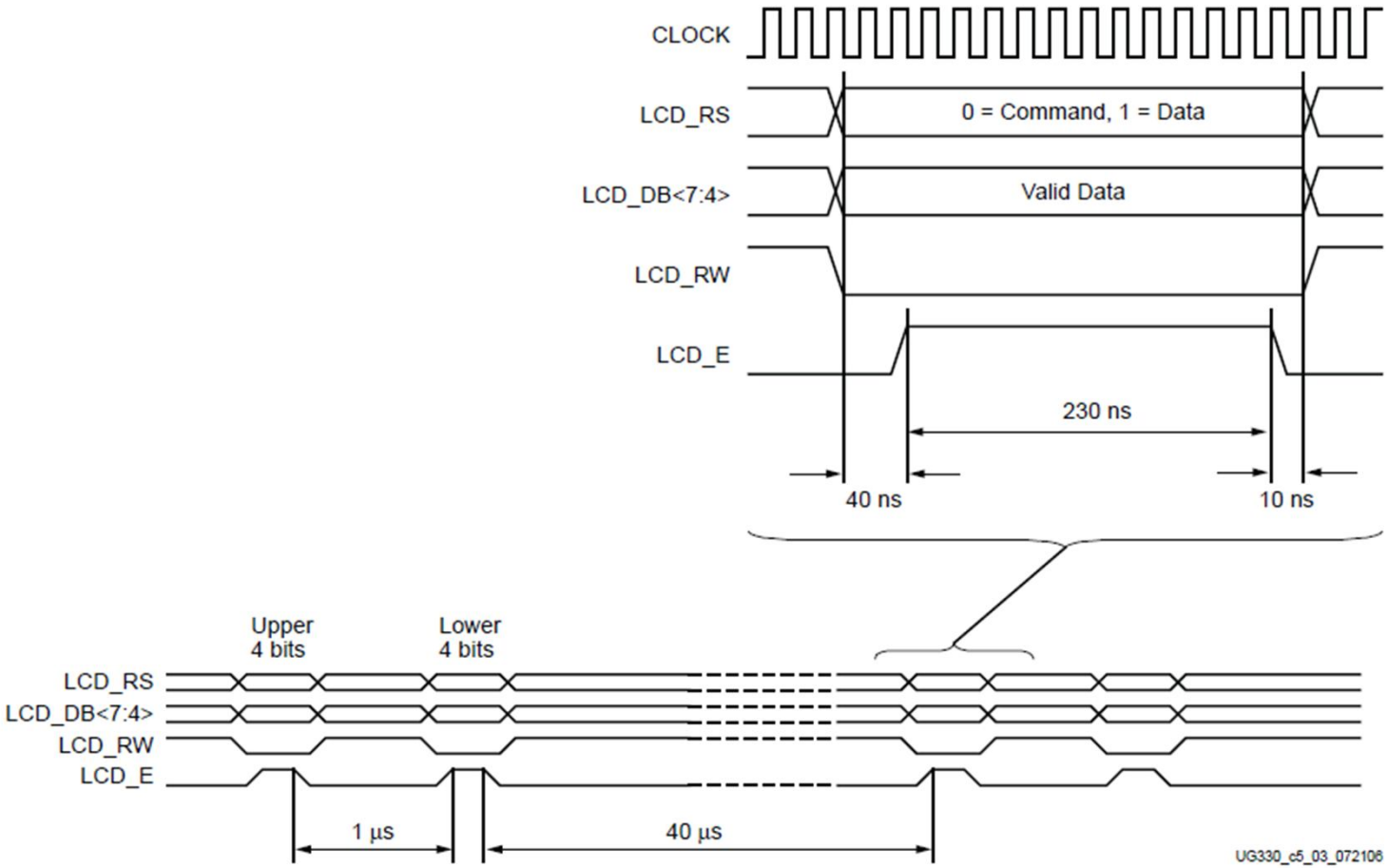
	Character Display Addresses																Undisplayed Addresses		
1	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	10	…	27
2	40	41	42	43	44	45	46	47	48	49	4A	4B	4C	4D	4E	4F	50	…	67
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	…	40

LCD in 4-bit Mode

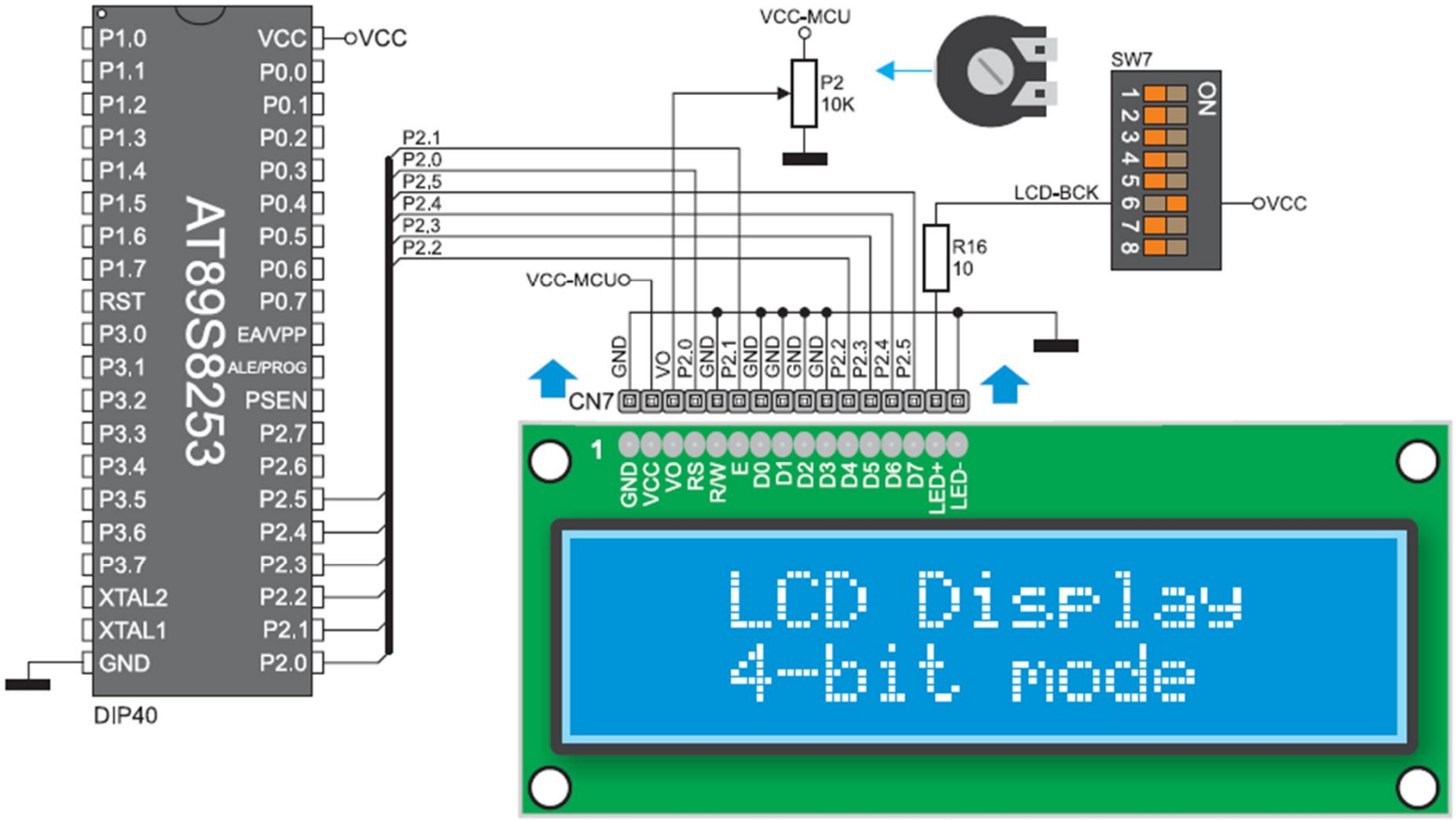
- In 4-bit mode
- Only data pins D4-D7 are connected.
- D0-D3 are grounded.
- This mode saves Microcontrollers 4 precious I/O Pins.
- Data is sent as High nibble first followed by lower nibble.

LCD Timing

- All instructions except ‘return cursor home’ and ‘clear display’ take 40 us to execute. These two take 1.6 ms.



4-bit LCD interfaced to 8051



Example Program

Example:

Display some characters on LCD using 4-bit Mode.

```
1 #include<reg51.h>
2 #define lcd P2
3 sbit RS=P2^0;
4 sbit E =P2^1;
5
6 void LCD_CMD(unsigned char);
7 void LCD_Data(unsigned char);
8 void delay_ms(unsigned int);
9
10 void main(void)
11 {
12   lcd=0;
13   lcd=lcd|0x08;
14   E=1;
15   E=0;
16   delay_ms(1);
17   LCD_CMD(0x28);// Function Set Command
18   LCD_CMD(0x06);// Entry Mode Set
19   LCD_CMD(0x0C);// Display on/off Control
20   LCD_CMD(0x01);// Clear Display
21   delay_ms(1);
22   LCD_Data('U');
23   LCD_Data('E');
24   LCD_Data('T');
25   while(1);
```

```
27 void LCD_CMD(unsigned char command)
28 {
29 lcd=0;
30 RS=0;
31 //First higher nibble
32 lcd=lcd|((command>>2) & 0x3C) ; // Lcd data pins are conected to P2.2 to P2.5
33 E=1;
34 E=0;
35 lcd=0;
36 RS=0;
37 //Then lower nibble
38 lcd=lcd|((command<<2) & 0x3C);
39 E=1;
40 E=0;
41 delay_ms(1);
42 }
```

```
44 void LCD_Data(unsigned char Data)
45 {
46 lcd=0;
47 RS=1;
48 //First higher nibble
49 lcd=lcd|((Data>>2) & 0x3C) ; // Lcd data pins are conected to P2.2 to P2.5
50 E=1;
51 E=0;
52 lcd=0;
53 RS=1;
54 //Then lower nibble
55 lcd=lcd|((Data<<2) & 0x3C) ;
56 E=1;
57 E=0;
58 delay_ms(1);
59 }
```

Todays Task 1

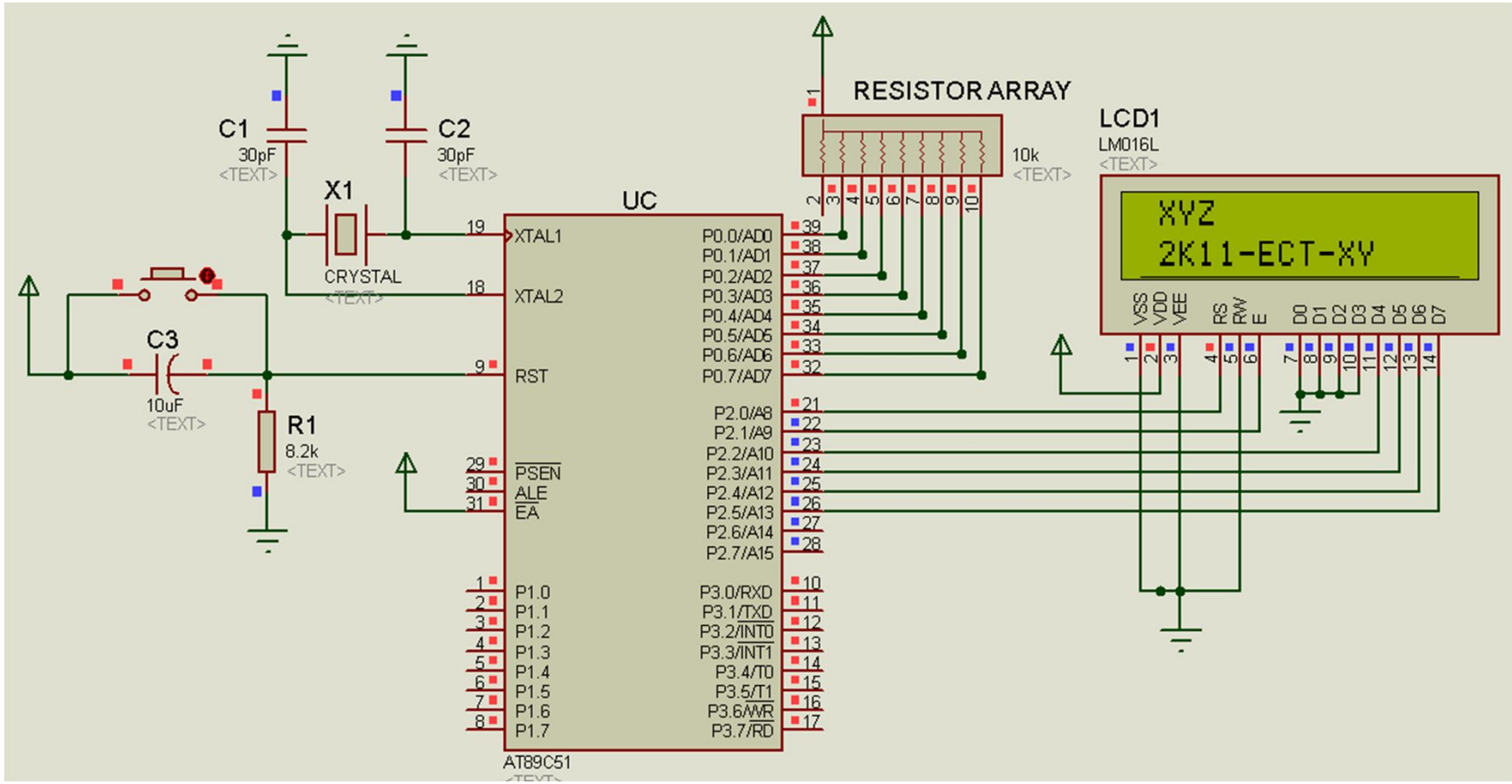
- Implement this on easy 8051 Kit and Proteus
- Make a function that takes a character string as input argument and displays it on LCD.
- Using the abovementioned function
- Write your name on the LCD's First line
- Write your roll number on the LCD's second line

Task Code

```
1  #include<reg51.h>
2  #define lcd P2
3  sbit RS=P2^0;
4  sbit E =P2^1;
5
6  void LCD_CMD(unsigned char);
7  void LCD_Data(unsigned char);
8  void delay_ms(unsigned int);
9  void Display_String(unsigned char*);
10
11 void main(void)
12 {
13   lcd=0;
14   lcd=lcd|0x08;
15   E=1;
16   E=0;
17   delay_ms(1);
18   LCD_CMD(0x28);// Function Set Command
19   LCD_CMD(0x06);// Entry Mode Set
20   LCD_CMD(0x0C);// Display on/off Control
21   LCD_CMD(0x01);// Clear Display
22   delay_ms(1);
23   Display_String(" XYZ ");
24   LCD_CMD(0xC0);//Take Cursor to Second Line , First Character Positior
25   Display_String(" 2K11-ECT-XY ");
26   while(1);
27 }
```

```
62 void Display_String(unsigned char *str)
63 {
64   unsigned char a=0;
65   while(str[a]!=0)
66   {
67     LCD_Data(str[a]);
68     a++;
69   }
70 }
```

Proteus Simulation



Todays Task 2

- Implement this on easy 8051 Kit.
- Read P1 and Display its hexadecimal value on LCD after converting it into ASCII.

Task Code

```
11 void main(void)
12 {
13   unsigned char x,a,b;
14   lcd=0;
15   lcd=lcd|0x08;
16   E=1;
17   E=0;
18   delay_ms(1);
19   LCD_CMD(0x28);// Function Set Command
20   LCD_CMD(0x06);// Entry Mode Set
21   LCD_CMD(0x0C);// Display on/off Control
22   LCD_CMD(0x01);// Clear Display
23   delay_ms(1);
24   Display_String("The Count is");
25   while(1)
26   {
27     LCD_CMD(0xC4);
28     x=P1;
29     a=((x&0xF0)>>4);
30     b=(x&0x0F);
31     //Display First Character
32     if(a<10)
33     LCD_Data(0x30|a);
34     else
35     LCD_Data(0x40|(a-9));
36     //Display Second Character
37     if(b<10)
38     LCD_Data(0x30|b);
39     else
40     LCD_Data(0x40|(b-9));
41   }
42 }
```

Proteus Simulation