Andhra Pradesh State Skill Development Corporation

Learn Anytime Anywhere

Andhra Pradesh State Skill Development Corporation

Embedded systems

LCD Display interfacing with ARM7

LCD Interfacing with LPC 2148

AIM: To print the required string in LCD Module.

Software Required: Keil IDE and Magic Flash Tool

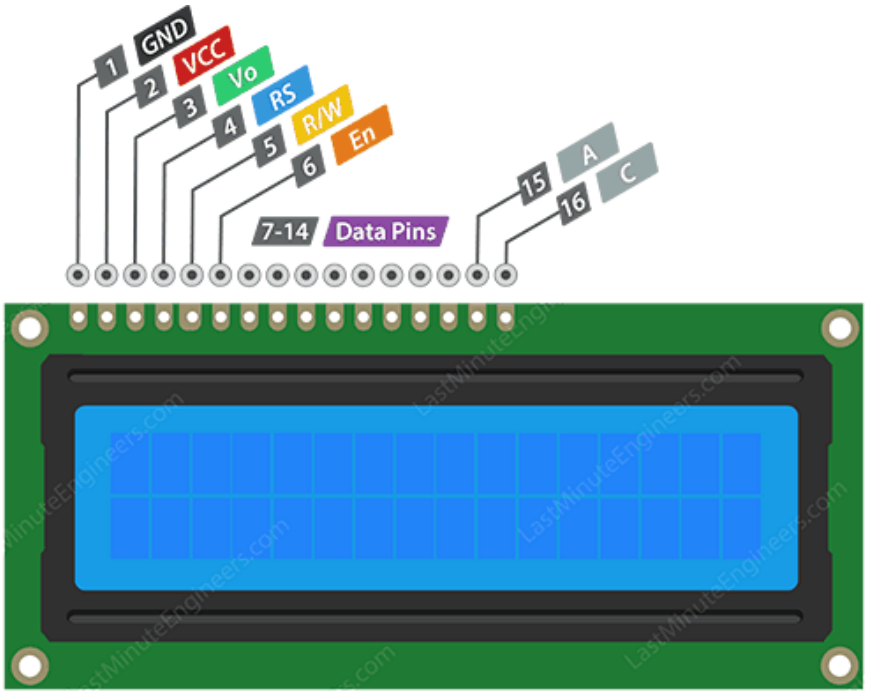
Components Required:

1. System -1
2. ARM7-LPC2148 Microcontroller board
3. LCD (16X2)
4. Potentiometer
5. Breadboard
6. Connecting Wires
7. Micro USB cable

Theory:-

LCD (liquid crystal display) is the technology used for displays in notebook and other smaller computers. Like light-emitting diode (LED) and gas-plasma technologies, LCDs allow displays to be much thinner than cathode ray tube (CRT) technology. This LCD controller can be operated in 4-bit or 8-bit mode. You can easily buy this cheap china made LCD in almost every supplier shop. Let's first try to understand its pins and related functions. I recommend you to keep datasheet in-hand, to download click here:

LCD 16x2 Module



LCD Pins Desciption:

Before wiring up circuit between JHD162A LCD and LPC2148 Microcontroller. Let's understand function of each pin provided on JHD162A LCD Module and are given as below:

PIN1 (VSS): Ground (GND) Pin of JHD162A LCD Module

PIN2 (VCC): +5V supply should be given to this pin as VCC

PIN3 (VEE): This pin usually used to adjust a contrast. This is usually done by connecting 10K potentiometer to +5V and ground and then connecting slider pin to VEE of LCD Module. This voltage across VEE pin defines the contrast. In general case this voltage is between 0.4V to 0.9V.

PIN4 (RS): This pin referred as Register Select (RS). The JHD126A LCD has two registers called command register & data register. Logic HIGH ('1') at Pin **RS** selects data register and Logic LOW ('0') at Pin RS will select command register. When we make **RS** Pin HIGH and put data on data lines (DB0-DB7). It will be recognized as data. And if we make **RS** Pin LOW and put any value on data lines, then it will recognizes as a command.

PIN5(R/W): This Pin is used for selecting pin function between read and write mode. Logic HIGH ('1') at this pin activates read mode and Logic LOW at this pin activates write mode.

PINs (DB0-DB7): These are data pins. The commands and data can be transferred through these pins in either 8-bit or 4-bit mode.

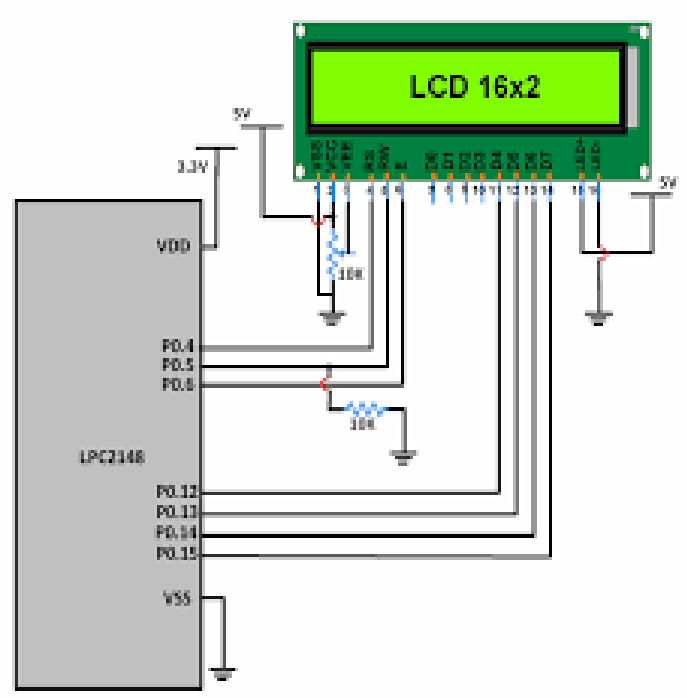
PIN15 (A or LED+): Anode of the backlight LED. When operated on 5V a 10E resistor should be connected in series to this pin.

PIN16 (K or LED-): Cathode of the backlight LED.

You might be wondering why we are using 4-bit and not 8-bit mode to Interface LCD with LPC2148 ARM7 Microcontroller. Here I have presented some differences. It actually depends on requirements, availability of pins and timing requirement etc. etc. 4-bit mode uses 4 I/O Port Pins for data and two or three additional I/O Pins for control. 8-bit mode uses 8 I/O Port Pins for data and two or three additional I/O Pins for control. 4-bit mode requires two 4-bit transfers for each instruction and character that is sent to the display. 8-bit mode requires only one 8-bit transfer for each instruction and character that is sent to the display. The implication is that, 4-bit data transfers will take twice as much time to transfer as 8-bit data transfer. 4-bit data transfers use 4 I/O lines less than 8-bit data transfer. It is a trade off! We save up to 4 I/O lines using 4-bit mode over that of 8-bit mode but, the data transfer takes twice as long in 4-bit mode as it does in 8-bit mode. If I/O is limited, 4-bit mode might preferred. If I/O is enough in number available and timing is important then 8-bit mode may be the way to go. 4-bit data transfers also require a bit more code as, the lower nibble will need to be shifted into the upper nibble with each command and character transfer. As we all aware that almost everything in this field is trade-off! We have to decide on an application by application basis, which is best approach to take. To establish proper communication and interface between JHD162A LCD with LPC2148 Microcontroller. We need

to supply commands in a given order to the data pins with small amount of delay in between to initialize LCD properly. These commands are listed in given table. We' will use these commands in our program.

LCD Interfacing with ARM7



Procedure:-

1. Open Keil µVision from the icon created on your desktop.
2. Create a new project on Kiel with the appropriate name and destination.
3. Take a new text file and write the code in a text editor.
4. Save the text file with ".c "extension.
5. add ".c " file to source group and check errors and warnings.
6. change the target options and create a hex file.
7. Now open flash magic to burn hex file into the development board.
8. Connect the hardware circuit and Connect your development Board to the USB port of your computer.
9. In the flash-magic window select the target device, serial port, board rate, and hex file.
10. Click on the start button to burn the hex file to the development board.
11. after uploading press the reset button and check the output.

Code:-

```
#include<lpc21xx.h>

// pin declearation
#define rs 0x00000040 //p0.10
#define en 0x00000080  //p0.11
#define d0 0x00000100        // P0.12
#define d1 0x00000200        //P0.13
#define d2 0x00000400        //P0.14
#define d3 0x00000800        //P0.15
#define d4 0x00001000        //P0.16
#define d5 0x00002000        //P0.17
#define d6 0x00004000        //P0.18
#define d7 0x00008000        //P0.19

// fuctions declaration
void lcd_delay(unsigned int);
void lcd_cmd(unsigned char);
void lcd_init(void);
void lcd_data(unsigned char);
void lcd_string(unsigned char []);
void lcd_number(unsigned long);

// main fuction
int main(){
        lcd_init();
        while(1){
                lcd_cmd(0x80);
                lcd_data('E');
                lcd_cmd(0x82);
                lcd_string("Skill APSSDC");
                lcd_cmd(0xc0);
                lcd_number(20201006);
        }
}
// function definition
void lcd_delay(unsigned int ch){
        int x,y;
        for(x=0;x<ch;x++)
        for(y=0;y<1000;y++);
}

void lcd_cmd(unsigned char ch){
        IOCLR0|=(d0|d1|d2|d3|d4|d5|d6|d7);
        IOCLR0|=rs;
        IOSET0|=(ch<<8);
        IOSET0|=en;
        lcd_delay(10);
        IOCLR0|=en;
}
```

```
void lcd_init(void){
        IODIR0|=(rs|en|d0|d1|d2|d3|d4|d5|d6|d7);
        lcd_cmd(0x38);
        lcd_cmd(0x02);
        lcd_cmd(0x06);
        lcd_cmd(0x0c);
        lcd_cmd(0x01);
}

void lcd_data(unsigned char ch){
        IOCLR0|=(d0|d1|d2|d3|d4|d5|d6|d7);
        IOSET0|=rs;
        IOSET0|=(ch<<8);
        IOSET0|=en;
        lcd_delay(10);
        IOCLR0|=en;
}

void lcd_string(unsigned char ch[]){
        int x;
        for(x=0;ch[x]!='\0';x++)
        lcd_data(ch[x]);
}

void lcd_number(unsigned long ch){
        unsigned int x=0,y[10];
        while(ch>0){
                y[x]=ch%10;
                ch=ch/10;
                x++;
        }
        while(x>0){
                x--;
                lcd_data(y[x]+48);
        }
}
```

Result: The required Data displayed in LCD Module.