ISSN 2229-5518

Interfacing of LCD Module with ARM Processor

P.S.S SUSHAMA, C.NAGARAJA, K. NAGABHUSHAN RAJU and K. MALAKONDAIAH

ABSTRACT---- Many microcontroller based instruments and machines need to display alpha numerals to give directions or data values to the user. In instruments where only a small amount of data needs to be displayed, simple digit type displays are often used. Liquid crystal displays are used mostly in portable and battery operated instruments because of their low power consumption. In the present study, a dot matrix liquid crystal display module (JHD 162A) which is an output module is interfaced with an ARM processor LPC2366. The hardware and software features are described.

1. Introduction:

All Instrumentation systems have to possess either a display that human operator can read out from and interpret, or an output device that enables the transfer of information from the instrumentation system to a general purpose or a dedicated micro computer.[1] Various types of displays are available today for the presentation of outputs of digital systems in visual form. In system where a large amount of data needs to be displayed, a cathode ray tube is generally used. In systems where only a small amount of data is to be displayed, a simple digital type displays are used. Liquid crystal displays are one such type of simple displays that are widely used in portable and battery operated instruments because of their low power consumption. Many formats such as segments and dot matrix are used for the representation of alpha numeric characters[2] The embedded processors/systems are widely used in many industrial, laboratory and domestic applications in view of their advanced architecture and other excellent features. Hence, the need to interface the liquid crystal displays with the embedded processors. In the present study, a dot matrix display module JHD 162A is interfaced with an ARM processor LPC 2366.

2. Experimental:

ARM Processor LPC2366 posses many features. Some of its salient features are mentioned below.

(i) 16/32 bit CPU.
(ii) Five 32- bit ports with 70 GPIO pins.
(iii) Single power supply 3.3V.
(iv) Four timers/ counters.
(v) 10-bit ADC & 10- bit DAC.
(vi) Four serial ports etc.

The block diagram of dot matrix liquid crystal display JHD 162A module is shown in Fig.1. and its pin configuration [3] is shown in Fig.2 . The Fig. 3 shows the picture of JHD 162A LCD module.

The salient features of the display module [3] are:

i. Display construction -> 16 Characters * 2 Lines.
ii. Display mode -> Positive Transflective.
iii. Display type -> TN\STN.
iv. Backlight -> LED(B/5.0V)
v. Viewing direction -> 6 o'clock.
vi. Operating temperature -> Indoor.
vii. Driving voltage -> single power.
viii. Driving method ->1/16 duty,1/5 bias.
ix. Type-> COB (Chip On Board).
x. Number of data line-> 8-bit parallel.
xi. Connector -> pin type.

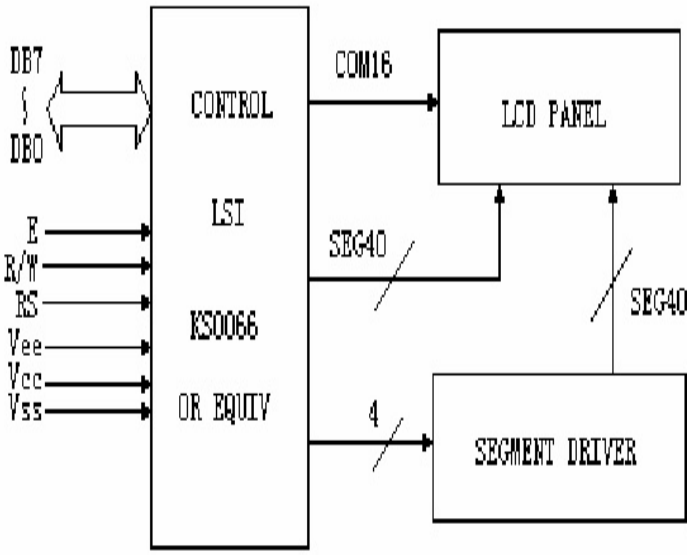

Figure1. Block diagram of LCD display module.

-

PIN CONFIGURATION

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
VSS	Vcc	VEE	RS	R/W	E	DB0	DB1	DB2	DB3	DB4	DB5	DB6	DB7	LED+	LED-

Figure 2. Pin configuration of LCD module.

Figure3. LCD module display.

3. Hardware details:

Fig 4.shows the circuit details of interfacing the JHD 162A with ARM processor. The power supply pins VDD, VSS of the module are connected between +5V and ground pins, and wiper terminal of the potentiometer is connected to VL terminal of the display for adjusting the display contrast. To read the status of the display the R/W pin is made high. To write the data in to the display R/W pin is made to be Low. As we are not reading any data from the display in the present study, the R/W pin of the display is connected to the ground. The RS (register select), E(enable) pins of the display are connected to the P1.28, P1.29 pins of ARM controller. The data pins D4-D7 of the display are connected to port P1.24-P1.27 of the microcontroller.

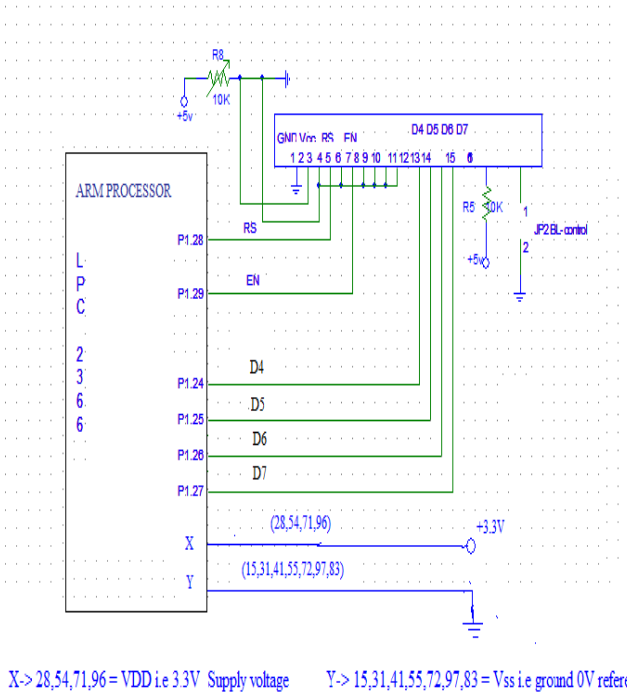

Figure4. Interfacing of LCD Module with ARM processor:

Software details :

Algorithim:

LCD Algorithm:-

1. Call Initialization subroutine.
2. Call print LCD subroutine.

Initialization subroutine:

1. Set the direction of the pins from 24 to 29 as output pins for LCD by using register .
2. IODIR1 for port1.
3. Get data to a register.
4. Call command write subroutine for enable 4-bit mode .
5. Call delay1 subroutine.
6. Call command write subroutine to initialize LCD in 4-bit mode and two line 5*7dots display.
7. Call delay1 subroutine.
8. Call command write subroutine to clears all display and returns the cursor to the home position.
9. Call delay1 subroutine.
10. Call command write subroutine to sets the cursor move direction and specifies not to shift the display.
11. Call delay1 subroutine.

12. Call command write subroutine to sets ON of all display, cursor ON, and blink of cursor position character.
13. Call delay1 subroutine.
14. Call command write subroutine to sets the CGRAM, data is sent and recived after this setting.
15. Call delay1 subroutine.

Print LCD subroutine:

1. Take a character and send to data write subroutine.
2. Call delay subroutine.
3. Repeate from step 1 to 2 write regsiter untill data pointer reaches null charecter.

Command write subroutine:

1. Take command data in temperory variable.
2. Clear data on pins P1.24 to P1.29.
3. Mask higher nibble with 0xF0 of temperory variable and left shift with 20.
4. Send the data in the temporary variable to P1.24 to P1.29.
5. Set RS pin 0.
6. Set enable pin .
7. Call delay subroutine.
8. Clear enable pin.
9. Take command data in temperory variable.
10. Clear data on pins P1.24 to P1.29.
11. Mask lower nibble with 0x0F of temperory variable and left shift with 24.
12. Send the data in the temporary variable to P1.24 to P1.29.
13. Set enable pin.
14. Call delay subroutine.
15. Clear enable pin.

Data write subroutine:

1. TakeLCD data in temperory variable.
2. Clear data on pins P1.24 to P1.29.
3. Mask higher nibble with 0xF0 of temperory variable and left shift with 20.
4. Send the data in the temporary variable to P1.24 to P1.29.
5. Set RS pin to 1.
6. Set enable pin .
7. Call delay subroutine.
8. Clear enable pin.
9. Take LCD data in temperory variable.
10. Clear data on pins P1.24 to P1.29.
11. Mask lower nibble with 0x0F of temperory variable and left shift with 24.
12. Send the data in the temporary variable to P1.24 to P1.29.
13. Set RS pin to 1.
14. Set enable pin.
15. Call delay subroutine.
16. Clear enable pin.

Some of the registers of LPC 2366 are used in the program as mentioned in the user manual[4].

The program, in detail, written in Embedded C (Keil IDE vision V4.00)[5] as follows.

```
//--------------Main file-------------//

#include <LPC23xx.H>

#include "lcd.h"

int main()

{

LCD_INIT();//LCD Initialization

PRINTLCD ("S.K . University");

}

//-------LCD initialization-------//

#include <LPC23xx.H>

#include "lcd.H"

void LCD_CMD_WRT(unsigned char c)//send commands to LCD

{

 long int temp_lcd_value;

 temp_lcd_value=c;

 IOCLR1=0x3F000000;

 temp_lcd_value=temp_lcd_value&0xF0

 temp_lcd_value=temp_lcd_value<<20;

 IOSET1=temp_lcd_value;

 IOCLR1=0x10000000; //RS clr,cmd

 IOSET1=0x20000000;//EN set

 LCD_DELAY(delay);

 IOCLR1=0x20000000;//EN clr

 temp_lcd_value=c;

 IOCLR1=0x3F000000;
```

```
 temp_lcd_value=temp_lcd_value & 0x0F;
 IOSET1=temp_lcd_value<<24;
 IOCLR1=0x10000000;//RS clr,cmd
 IOSET1=0x20000000;//EN set
 LCD_DELAY(delay);
 IOCLR1=0x20000000;//EN clr
}
//=========================//
void LCD_DATA_WRT(unsigned char c) //Function to send data to LCD
{
 long int temp_lcd_value;
 temp_lcd_value=c;
 IOCLR1=0x3F000000;
 temp_lcd_value=temp_lcd_value & 0xF0;
 temp_lcd_value=temp_lcd_value<<20;
 IOSET1=temp_lcd_value;
 IOSET1=0x10000000;  //RS set,data
 IOSET1=0x20000000;//EN set
 LCD_DELAY(delay);
 IOCLR1=0x20000000;  //EN CLR
 temp_lcd_value=c;
 IOCLR1=0x3F000000;
 temp_lcd_value=temp_lcd_value & 0x0F;
 IOSET1=temp_lcd_value<<24;
 IOSET1=0x10000000;//RS set
 IOSET1=0x20000000;//EN set
 LCD_DELAY(delay);
 IOCLR1=0x20000000;//EN CLR
}
//=====================//
VoidLCD_INIT(void)   //Function to initialise LCD
{
  IODIR1=0x3F000000;
  LCD_CMD_WRT(0x20);
  LCD_DELAY1(delay1);
  LCD_CMD_WRT(0x28);
  LCD_DELAY1(delay1);
  LCD_CMD_WRT(0x01);
  LCD_DELAY1(delay1);
  LCD_CMD_WRT(0x06);
//don't move after display
  LCD_DELAY1(delay1);
  LCD_CMD_WRT(0x0f);
  LCD_DELAY1(delay1);
  LCD_CMD_WRT(0x80);
  LCD_DELAY1(delay1);
}
//=========================//
void PRINTLCD(unsigned char *ptr)
 //Function to print string on LCD
 {
   while(*ptr!='\0')
   {
   LCD_DATA_WRT(*ptr++);
   LCD_DELAY(delay);
   }
}
```

```
//=========================//
void LCD_DELAY(unsigned int i )          //Delay Function
{
  unsigned int j;
  for(j=0 ;j < i;j++);
}
//=========================//
VoidLCD_DELAY1(unsignedinti) //Delay Function
{
  unsigned int j;
  for(j=0 ;j < i;j++);
}
//----Initialisations .h file ----------//
#include <LPC23xx.H>
#ifndef  _LCD_H
#define  _LCD_H
#define  delay  60000
#define  delay1 60000
voidLCD_CMD_WRT(unsigned char);
voidLCD_DATA_WRT(unsigned char);
void LCD_INIT(void);
void LCD_DELAY(unsigned int );
void LCD_DELAY1(unsigned int );
void PRINTLCD(unsigned char *);
#endif
```

By the execution of the above program the following data is displayed on Lcd display module.

Data displayed on LCD module is

S.K. University

Conclusions:

The hardware and software features of interfacing JHD 162A dot matrix display module with an ARM Processor LPC2366 are described. The necessary software is developed in Embedded C language. The data written in the embedded C code is displayed on LCD module in Line1.

Acknowledgements:

The authors are thankful to the University Grants Commission, New Delhi for providing financial assistance through UGC Major Research Project.

References

[1] S.Bhasker Reddy, K. Malakondaiah and S. Raja Ratnam, Interfacing a Dot Matrix Liquid Crystal Display Module with Microcontroller, J. Physics Education, April-June, P. 41-50 (2000).
[2] Douglas V. Hall, Microprocessors and Interfacing Programming and Hardware, Tata McGraw-Hill, New Delhi (1993).
[3] JHD 162A user's manual.
[4] www.keil.com/dd/docs/datashts/philips/lpc23xx_um.pdf.
[5] http://www.keil.com/arm/mdk.asp