Interfacing LCD with Pt-51 kit

EE319: Microprocessor and Microcontroller Laboratory

Department of Electrical Engineering
Indian Institute of Technology Dharwad

Overview

JHD 162A LCD Display

- The display provided with the Pt51 kit (JHD 162A)
- JHD 162A can display up to 16 characters per line in two lines (hence its name – 16 2A).

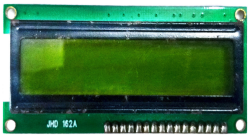
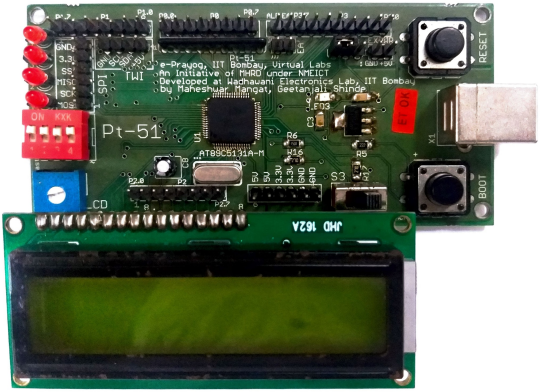


Figure: JHD 162A Display

- Controlling the LCD display is done by sending **commands** and **data** from the micro-controller on Pt51 (AT 89C5131A) to the micro-controller on the display (HD44780U).
- HD44780U has a local memory which stores the characters to display and a programmable memory which stores fonts.

LCD connection to Pt-51 kit



Hardware and Software Interfacing

Hardware

- Three control lines from *Port P0* to HD44780
  - P0.0 - Register Select (RS)
  - P0.1 - Read or Write (RW)
  - P0.2 - Enable (EN)
- RS - decides whether a **command** or **data** is sent
  - RS = 0 for command transfer
  - RS = 1 for data transfer
- RW - decides whether a **read** or **write** operation is desired
  - RW = 0 for write operation
  - RW = 1 for read operation

- EN - provides hand shake for command/data transfer
  - EN = 0 at start of command/data transfer
  - EN = 1 to look for command/data transfer and set back to EN = 0
  - HD44780 acts on the command/data only when EN has a downward transition.
- 8 lines from *Port P2* to HD44780 for data

Software

- Following equates can be defined in program for control and data lines:

```
RS EQU P0.0
RW EQU P0.1
EN EQU P0.2
Data EQU P2
```

Port definitions for signal transfer

```
LCD_data equ P2    ;LCD Data port
LCD_rs   equ P0.0  ;LCD Register Select
LCD_rw   equ P0.1  ;LCD Read/Write
LCD_en   equ P0.2  ;LCD Enable
```

```c
/* @section  I N C L U D E S */
#include <AT89C513xA.h>
#define LCD_data P2
#define LCD_rs P0_0
#define LCD_rw P0_1
#define LCD_en P0_2


/*Function Declarations*/
void LCD_Init();
void LCD_DataWrite(unsigned char dat);
void LCD_CmdWrite(unsigned char cmd);
void sdelay(unsigned int delay);
```

Figure: Port definitions in ASM and C language

Command Set for 44780

- The position of the first non-zero bit (from MSB) determines what kind of command.
- If DB7 = 1, it is the *display data RAM address set command*. The rest of the bits specify the address in the display RAM from where the next read/write will be carried out.
- If DB6 = 1 is the first non zero bit, it is the *character generator address set command*. The remaining bits will be character generator RAM address.
- If DB7, DB6 are 0 and DB5 = 1, it is the *function set command*. Then, DB4 specifies the number of data bits (NDB) used by the interface, DB3 gives the number of display lines (NDL) in the display and DB2 picks the font (F).
- If DB4 is the first non zero bit, it is the *shift command*. DB3 gives whether the display or the cursor will be shifted, while DB2 signifies whether a right or left shift is desired.

Command Set for 44780 (Cont...)

- If DB3 is the first non zero bit, it is the *display switch command*. Then, value at DB2 determines if the whole display will be turned on/off (D), DB1 is for turning the cursor on/off (C), while DB0 turns blinking of the cursor on or off (B).
- If DB2 is the first non-zero bit, it is the *input set command*. Here, DB1 is interpreted as auto increment/decrement mode of cursor position (I/D) while DB0 signifies whether the display should shift (S) after entering the new character by one position.
- If DB1 is the first non-zero bit, it *returns the cursor to the first position*.
- If DB0 is the only non-zero bit, it *clears the screen*.

Configuring the LCD

Step 1: Function set command

- To configure the LCD, we need to send a function set command first.
- Function command sets interface data length or the number of data bits (NDB), number of display line (NDL), and character font (F).
- Without this command, the display will not know whether to interpret the rest of the communication as 8 bit data or 4 bit data.
- Function set has the following format:

DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
0	0	1	NDB	NDL	F	-	-

- DB 765 = 001 signifies that it is a function set command.
- NDB is 1 for 8 bit data interface and 0 for 4 bit interface, NL is 1 for a 2 line display, 0 for a single line display and F is 1 for a 5×10 font and 0 for a 5×7 font.
- *For a 2 line display with 5×7 font on an 8 bit interface, we need to send 00111000 = 38H.*

Configuring the LCD (cont..)

Step 2: Display switch command

- The LCD display is turned ON/OFF by sending the display switch command.
- Display switch command sets on/off of all display (D), cursor on/off (C), and blink of cursor on/off (B).
- Display switch has the following format:

DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
0	0	0	0	1	D	C	B

- D is 0 for display OFF and 1 for display ON, C is 0 for cursor OFF and 1 for cursor ON and B is 0 for cursor blink OFF and 1 for cursor blink ON.
- *To turn on the display with a cursor which is not blinking, we send 00001110 or 0EH.*

Step 3: Input set command

- Finally, program the display in such a way that every time we send a character, the cursor automatically shifts to the right by one position.
- Here I/D stands for auto increment/decrement of cursor position while S is for shift control. These operations are performed during data read/write.
- This command has the following format:

DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
0	0	0	0	0	1	I/D	S

- *For just incrementing the cursor position, we send 00000110 or 06H.*

- Thus, configuring/initializing of the display involves sending 38H, 0EH and 06H to the display to set it to the mode described above.
- Each command should be sent after ensuring through a status read that the display μC is no more busy.

Step 4: Clear screen command (optional)

- *One may optionally clear the screen after this by sending a clear screen command 00000001 or 01H.*

LCD initilization (in ASM)

```
;-----------------------LCD Initialisation routine-------------------------------------------------
lcd_init:

        mov   A,#38H  ;Function set: 2 Line, 8-bit, 5x7 dots
        acall lcd_command

        mov   A,#0CH  ;Display on, Curson off
        acall lcd_command

        mov   A,#01H  ;Clear LCD
        acall lcd_command

        mov   A,#06H  ;Entry mode, auto increment with no shift
        acall lcd_command
        ret                 ;Return from routine
```

Command writing subroutine (in ASM)

```
;----------------------command sending routine-----------------------------------
 lcd_command:
        mov   LCD_data,A     ;Move the command to LCD port
        clr   LCD_rs         ;Selected command register
        clr   LCD_rw         ;We are writing in instruction register
        setb  LCD_en         ;Enable H->L
                acall delay
        clr   LCD_en
                acall delay

        ret
```

LCD initilization (in C)

```
void LCD_Init()
{
  sdelay(100);
  LCD_CmdWrite(0x38);          /* LCD 2lines, 5*7 matrix*/
  LCD_CmdWrite(0x0C);          /* Display on, Curson off*/
  LCD_CmdWrite(0x01);          /* Clear the LCD*/
  LCD_CmdWrite(0x06);          /* Entry mode, auto increment with no shift*/
}
```

Command writing subroutine (in C)

```
void LCD_CmdWrite(unsigned char cmd)
{
        LCD_data = cmd;                 /* Send the command to LCD*/
        LCD_rs=0;                       /* Select the Command Register by pulling LCD_rs LOW*/
        LCD_rw=0;                       /* Select the Write Operation  by pulling RW LOW*/
        LCD_en=1;                       /* Send a High-to-Low Pusle at Enable Pin*/
        sdelay(5);
        LCD_en=0;
        sdelay(5);
}
```

Writing character to the display in data mode.

- After initialization, for displaying text, write each character to the display in data mode.
- Before sending each character, one must ensure that:
  - LCD is not busy
  - Position the text at a particular place in the display
    - To position text/cursor, the data address must be set accordingly **(as a command)**
    - The sixteen characters in the first line reside at data address 80H to 8FH.
    - Addresses for the second line span the address range C0H to CFH.
- Send each character to be shown on LCD **as data**

Writing data (in ASM)

```
;----------------------data sending routine-----------------------------------
lcd_senddata:
        mov   a, #'p'
        mov   LCD_data, a     ;Move the command to LCD port
        setb  LCD_rs          ;Selected data register
        clr   LCD_rw          ;We are writing
        setb  LCD_en          ;Enable H->L
                acall delay
        clr   LCD_en
        acall delay
                acall delay
        ret                   ;Return from busy routine
```

Writing data (in C)

```
void LCD_DataWrite(unsigned char dat)
{
  LCD_data=dat;                 /* Send the data to LCD*/
  LCD_rs=1;                     /* Select the Data Register by pulling LCD_rs HIGH*/
  LCD_rw=0;                     /* Select the Write Operation by pulling RW LOW*/
  LCD_en=1;                     /* Send a High-to-Low Pusle at Enable Pin*/
  sdelay(5);
  LCD_en=0;
        sdelay(5);
}
```

Main Program

Main program section in ASM and C

```
ORG 0000H
ljmp start

org 200h
start:
        mov P2,#00h
        acall delay ;initial delay for lcd power up
        acall delay
        acall lcd_init      ;initialise LCD
        acall delay
        acall delay
        acall delay
        mov a,#085h             ;Put cursor on first row,5 column. For second row third column, 0C3h
        acall lcd_command       ;send command to LCD
        acall delay
        acall lcd_senddata        ;call text strings sending routine
        acall delay
here: sjmp here                       ;stay here
```

```
void main(void)
{
        P2 = 0x00;              /* Make Port 2 output*/
        LCD_Init();
        while(1)                /* endless  */
  {
                sdelay(500);
                LCD_CmdWrite(0x85);/* Put cursor on first row, fifth column, for second row use C instead of 8*/
                sdelay(10);             /* Delay*/
                LCD_DataWrite('p');  /* Write "Hello" in the first line*/
        }
}
```

Additional subroutines

Delay subroutine in ASM and C

```
;·····delay routine·····
delay:  mov r0,#10
loop2:  mov r1,#255
        loop1:  djnz r1, loop1
        djnz r0, loop2
        ret
```

```c
void sdelay(unsigned int delay)
{
        char d=0;
        while(delay>0)
        {
                for(d=0;d<5;d++);
                delay--;
        }
}
```

Lab Assignment

Problem Statement:

1. One line display: Verify the sample program provided, both in C and ASM, by displaying
   - a particular character on first line at the right end.
   - a particular integer on left end of the second line.

2. Two line display: Modify the sample code so as to display your batch and student name in two lines of lcd display (in both C and ASM).
   Eg:'EE2018-2022' on first line and 'ABC' on second line. Make sure to align the text to the middle of the display.