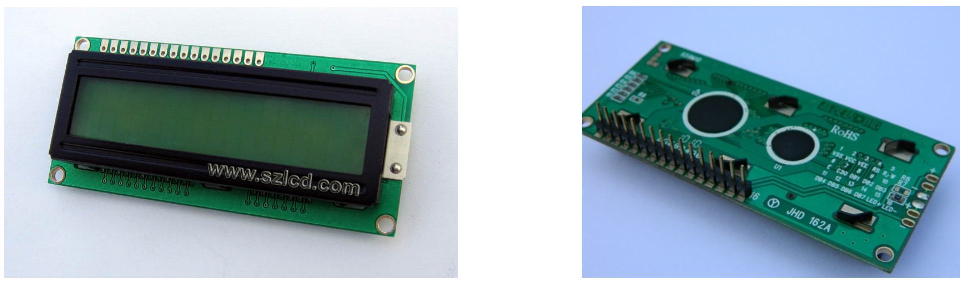
Experiment 9

Parallel Interfacing: Interfacing LCD Display

Objective

Frequently, a microcontroller program must interact with the outside world using input and output devices that communicate directly with a human being. One of the most common devices attached to a microcontroller is an LCD display. Some of the most common LCDs are 16x2 and 20x2 displays. This means 16 characters per line by 2 lines and 20 characters per line by 2 lines, respectively. Figure 9.1 shows the LCD display.



(a) Front View (b) Back View

Figure 9.1: Front and Back View of LCD Display

This manual explains how to interface Liquid Crystal Display (LCD) to LM4F120H5QR.

Introduction

A very popular standard exists which allows us to communicate with the vast majority of LCDs regardless of their manufacturer. The standard is referred to as HD44780U controller chip.

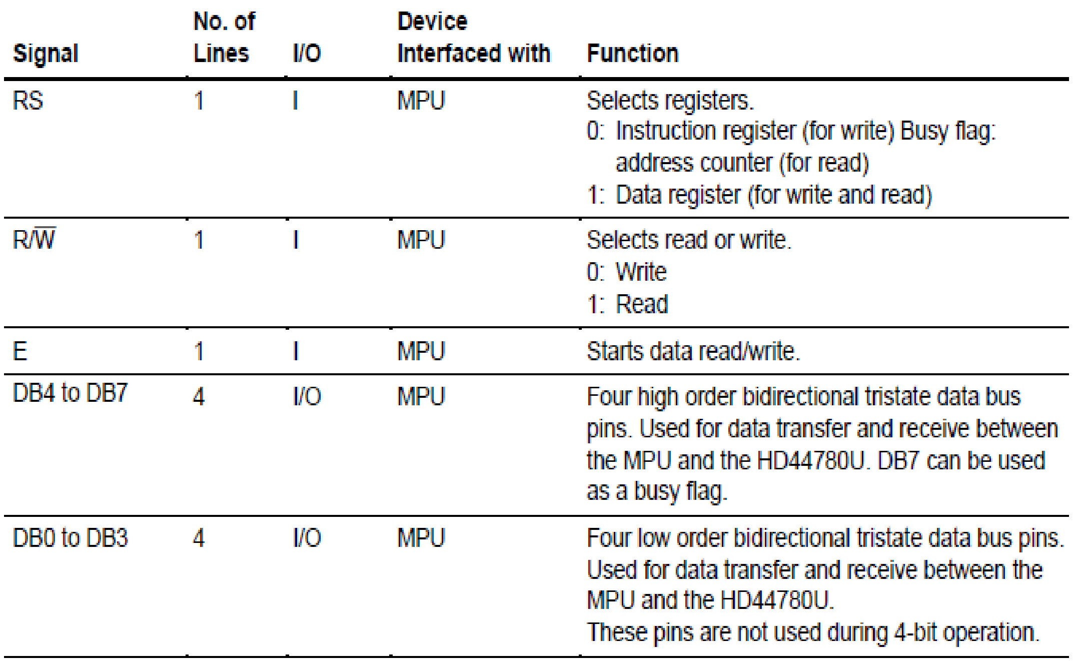
The interfacing of LCD is explained using this HD44780U as the Dot Matrix Liquid Crystal Display Controller/Driver. HD44780U is a programmable chip for various LCD display control functions. It drives the dot-matrix liquid crystal display. It can be configured under the control of a microprocessor.

In our example, we will connect the LCD controller to the LM4F120H5QR microcontroller. We will use the GPIO pins of LM4F120H5Qr for the purpose of connecting the LCD controller. We will then write a program to output some specific values to the GPIO pins (what values to write we will discuss about them shortly). This program will be executed in the ARM processor of the LM4F120H5QR microcontroller. The LCD controller, on receiving the bits sent over the GPIO pins, will display characters on the LCD and perform other control functions.

So what are the specific values that should be output to the GPIO pins? This is what we will learn in the next few sections.

LCD Controller Pins

The pins on the LCD controller being used for the purpose of interfacing are shown in Figure 9.2. Their explanation is given as follows.



Signal	No. of Lines	I/O	Device Interfaced with	Function
RS	1	I	MPU	Selects registers. 0: Instruction register (for write) Busy flag: address counter (for read) 1: Data register (for write and read)
R/$\overline{\text{W}}$	1	I	MPU	Selects read or write. 0: Write 1: Read
E	1	I	MPU	Starts data read/write.
DB4 to DB7	4	I/O	MPU	Four high order bidirectional tristate data bus pins. Used for data transfer and receive between the MPU and the HD44780U. DB7 can be used as a busy flag.
DB0 to DB3	4	I/O	MPU	Four low order bidirectional tristate data bus pins. Used for data transfer and receive between the MPU and the HD44780U. These pins are not used during 4-bit operation.

Figure 9.2: Pin Functions

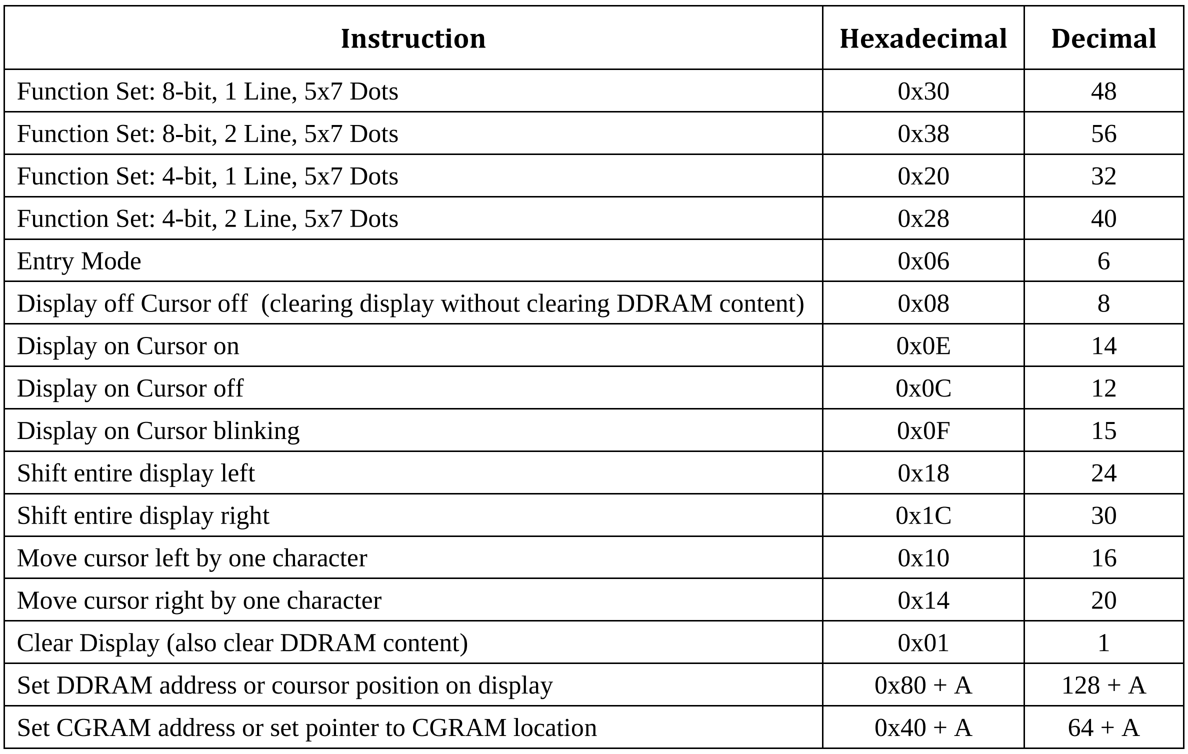
1. **DB0 to DB7** - the data bus pins. These signals can be used to send 8 bits of data from the microcontroller to the LCD controller or from the LCD controller to the microcontroller. DB7 can be used as busy flag also.
2. **R/W signal** - selects between a Read and Write operation. If R/W is equal to 0, a Write operation will be performed. For a Write operation, DB0 to DB7 will be used to send data from the microcontroller to the LCD controller. If R/W is equal to 1, a Read operation will be performed. For a Read operation, DB0 to DB7 will be used to send data from the LCD controller to the microcontroller.
3. **RS signal** - selects between instruction and data registers. RS equal to 0 selects the

Instruction register for a Write operation. This means that if RS=0 and R/W=0, the data sent over DB0 - DB7 will be put in the Instruction register. RS equal to 0 selects the Busy Flag for Read operation. This means that if RS=0 and R/W=1, the value in the Busy Flag will be sent over DB7. RS equal to 1 selects the Data register for Read and Write operation. This means that if RS=1 and R/W=0, the data sent over DB0 - DB7 will be put in the Data register. If RS=1 and R/W=1, the value in the Data register will be sent over DB0 - DB7 to microcontroller.

4. **E signal** - used to start data read or write. When the data is sent to the LCD, a high to low pulse must be applied to the E signal so that the LCD latches the data present at its pins. Similarly a high to low pulse must be provided to the E signal during a Read operation.

LCD Controller Commands

Although using the explanation in appendix A you can make your own commands and test them. Below is a breif list of useful commands which are used frequently while working on the LCD.



Instruction	Hexadecimal	Decimal
Function Set: 8-bit, 1 Line, 5x7 Dots	0x30	48
Function Set: 8-bit, 2 Line, 5x7 Dots	0x38	56
Function Set: 4-bit, 1 Line, 5x7 Dots	0x20	32
Function Set: 4-bit, 2 Line, 5x7 Dots	0x28	40
Entry Mode	0x06	6
Display off Cursor off (clearing display without clearing DDRAM content)	0x08	8
Display on Cursor on	0x0E	14
Display on Cursor off	0x0C	12
Display on Cursor blinking	0x0F	15
Shift entire display left	0x18	24
Shift entire display right	0x1C	30
Move cursor left by one character	0x10	16
Move cursor right by one character	0x14	20
Clear Display (also clear DDRAM content)	0x01	1
Set DDRAM address or coursor position on display	0x80 + A	128 + A
Set CGRAM address or set pointer to CGRAM location	0x40 + A	64 + A

Figure 9.3: LCD Commands

Interfacing the LCD Module

The microcontroller/microprocessor interface to HD44780 LCD modules is almost always 14 pins. You may find that some displays have additional pins for backlighting or other purposes, but the first 14 pins still serve as the interface.

The first three pins are used provide power to the LCD module. Pin 1 is GND and is connected to the GND available on Launchpad. Pin 2 is VCC and it is connected +5V VBUS available on the Launchpad. Pin 3 is the LCD Display Bias. A 10$k\omega$ potentiometer is connected to this pin to adjust the voltage. By adjusting the voltage or duty cycle of pin 3, the contrast of the display can be adjusted. Most character LCDs can achieve good display contrast with a voltage between 5V and 0V on pin 3. Note that greater contrast comes with lower voltage and you should never apply a VLCD higher than VCC. The control lines of the LCD, pins 4,5 and 6 are connected to the pins 1, 2 and 3 of GPIO Port D. Data Bus of the LCD (DB0 - DB7) is connected to GPIO port B (PB0 - PB7) of the Launchpad. These connections are shown in Figure 9.4 below

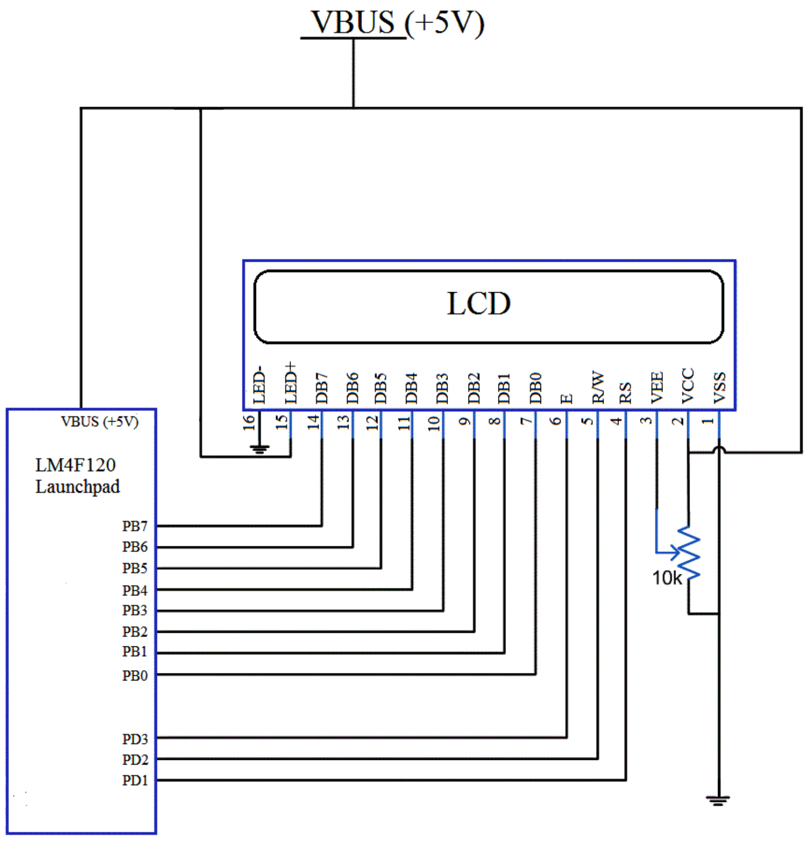


Figure 9.4: LCD Connections

LCD Initialization

Before displaying characters on the LCD display, it must be configured first. To configure an LCD display, four command words must be sent to LCD. The commands are:

1. Function set
2. Display On/Off control
3. Entry mode set
4. Display Clear

The flow chart in Figure 9.5 represents the initialization sequence of LCD display.

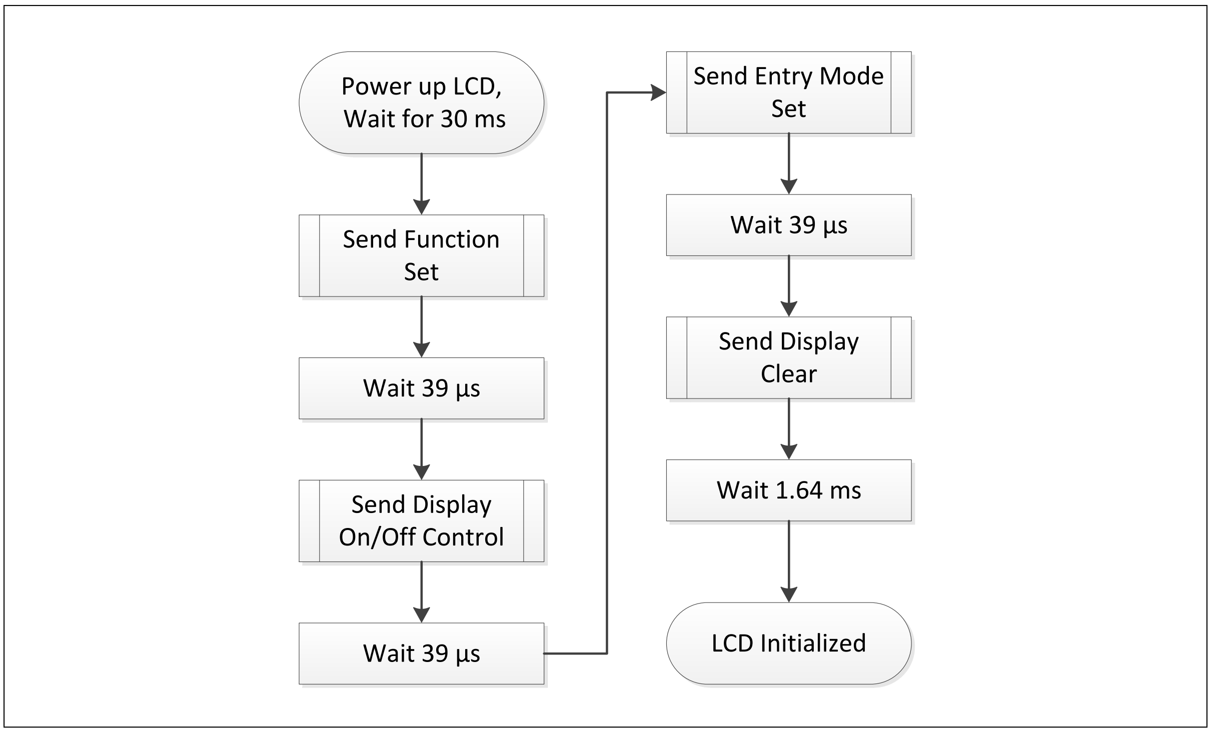


Figure 9.5: LCD Initialization Flowchart

Interfacing of LCD

To display data in the LCD or give command to it, first you put your data on the 8 bit bus, then put the command in the command bus and then pulse the enable signal. The flow chart in Figure 9.6 gives simple description of this interfacing.

Code Template

```
/*16x2 lcd with stellaris launchpad...
/-------------------------------------------/
CONNECTION TIPS:
RS: PD1
```

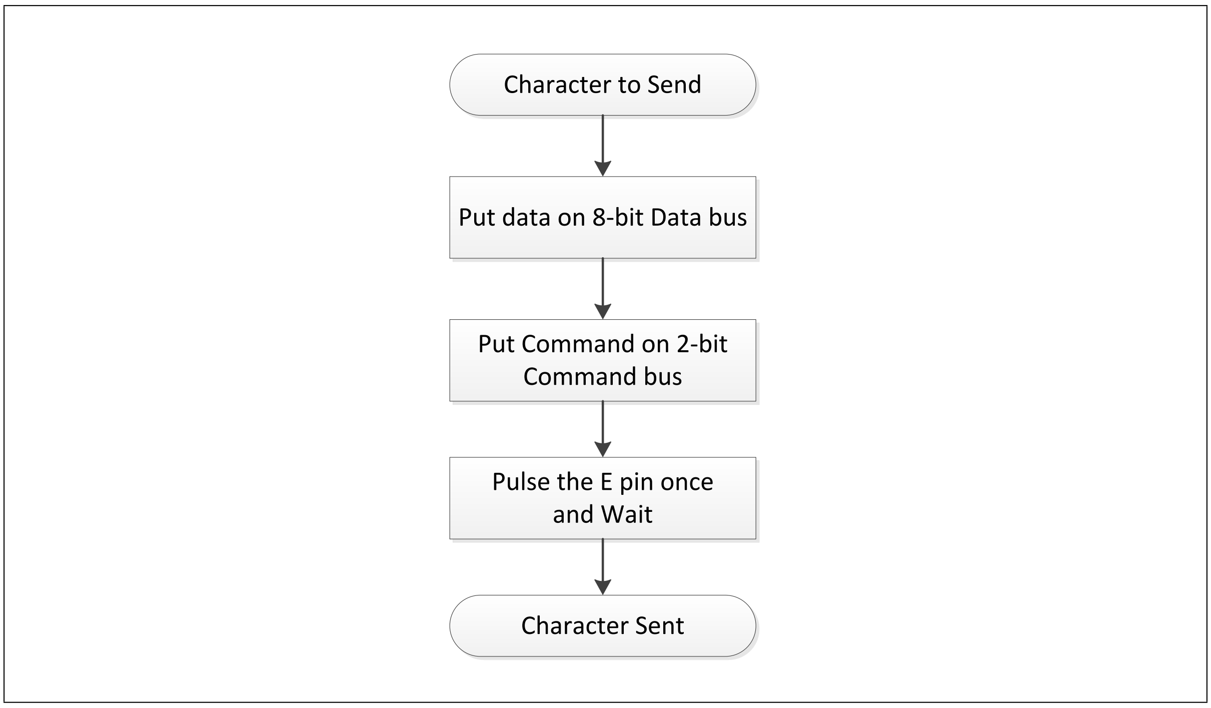

Character to Send
Put data on 8-bit Data bus
Put Command on 2-bit Command bus
Pulse the E pin once and Wait
Character Sent

Figure 9.6: Flowchart of 8 bit LCD Interfacing

```c
RW:  Connect to GND
EN:  PD3
DATA:  PB0 - PB7 to d0 to d7 in lcd respectively
VSS (GROUND: connect to ground on launchpad itself
VCC (POWER): connect 5v supply to LCD from the launchpad itself
VEE of LCD: Pull down using 10K resistor (connect to ground using 10K resistor)
/--------------------------------------------*/

#include "inc/lm4f120h5qr.h"

#define RS(X) GPIO_PORTD_DATA_R = ((GPIO_PORTD_DATA_R & ~(1<<1)) | (X<<1))
#define EN(X) GPIO_PORTD_DATA_R = ((GPIO_PORTD_DATA_R & ~(1<<3)) | (X<<2))

#define databits  GPIO_PORTB_DATA_R  // PB7 - D7, ....., PB0 - D0
#define LCD_STROBE  do{EN(1);EN(0);} while(0)

#define lcd_display_data_1   "Stellaris |====>"
#define lcd_display_data_2   "|====> Launchpad"

#define LINE1 lcd_cmd(0x80)
#define LINE2 lcd_cmd(0xc0)

volatile unsigned long ulLoop;

/*--------------------------------------------------------/
Name:                    setup
```

```
Description: Configure GPIO pins used to communicate
with LCD and clock setup for the experiment.
/-------------------------------------------------------------*/

void setup()
 {
  //clock setup
  SYSCTL_RCGCGPIO_R |= 0x0A;
  ulLoop = SYSCTL_RCGCGPIO_R;

  //lcd connections init
  GPIO_PORTB_AFSEL_R &= ~0xFF;
  GPIO_PORTB_PCTL_R &= 0x00000000;
  GPIO_PORTB_DIR_R = 0xFF;
  GPIO_PORTB_DEN_R = 0xFF;

  //add configuration for port D here

 }

/*-------------------------------------------------------------/
Name:                  lcd_cmd
Description: sends the command to the lcd controller
/-------------------------------------------------------------*/

void lcd_cmd(unsigned char c) {

  //add your code here
}

/*-------------------------------------------------------------/
Name:                  lcd_data
Description: sends the data to the lcd controller
/-------------------------------------------------------------*/

void lcd_data(unsigned char c) {
  //add your code here
}

/*-------------------------------------------------------------/
Name:                  clear
Description: clears the entire display and set DDRAM address to 0
/-------------------------------------------------------------*/

void clear(void)
{
  //add your code here
 }
```

```

/*------------------------------------------------------------/
Name:               lcd_init
Description: initializes the lcd controller
/------------------------------------------------------------*/

void lcd_init ()
{
  lcd_cmd(0x38);  // 8 bits, 2 lines and 5x8 dots font selected
  lcd_cmd(0x06);  // set entry mode (auto increment of cursor)
  lcd_cmd(0x0C);  // turn display on, make cursor invisible
  clear();        // clear screen
}

/*------------------------------------------------------------/
Name:               lcd_DisplayString
Description: displays character string on the lcd
/------------------------------------------------------------*/

void lcd_DisplayString(char str[], unsigned int length)
{
    unsigned int i;

    for(i=0; i<length; i++) {
       lcd_data( str[i] );
    }
}

/*------------------------------------------------------------/
                           main()
/------------------------------------------------------------*/
int main()
{
  while(1)
  {
    setup();
    lcd_init();
    LINE1;
    lcd_DisplayString(lcd_display_data_1, 16 );
    LINE2;
    lcd_DisplayString(lcd_display_data_2, 16 );
    RS(0);
  }
}
```

Exercise

Connect read or write (R/W) signal of LCD to the microcontroller Port D pin 2 and write a function to read busy flag (DB7) of LCD and modify the above code accordingly.