Interfacing STM32 with 16×2 LCD

G V V Sharma*

Contents

***Abstract*—This manual shows how to interface the 16×2 HD44780-controlled LCD using STM32F103C8T6.**

1 Components

Component	Value	Quantity
Breadboard		1
STM32F103C8T6		1
LCD	16 × 2 HD44780	1
Jumper Wires	F-F	20
Jumper Wires	M-F	20

TABLE 1.0: Components

Rpi 3	STM32	LCD Pins	LCD Pin Label	LCD Pin Description
GND		1	GND	
5V		2	Vcc	
GND		3	Vee	Contrast
	A0	4	RS	Register Select
GND		5	R/W	Read/Write
	A1	6	EN	Enable
	A2	11	DB4	Serial Connection
	A3	12	DB5	Serial Connection
	A4	13	DB6	Serial Connection
	A5	14	DB7	Serial Connection
5V		15	LED+	Backlight
GND		16	LED-	Backlight

TABLE 2.1: Pin Connections

2 Hardware

Problem 2.1. Make connections as shown in Table **??**.

3 Software

Problem 3.1. Execute the following program

```
https://github.com/gadepall/
    STM32F103C8T6/blob/master/
    examples/lcd/lcd_example.c
```

The following problems explain how to display the string **0** on the screen using the above code.

Problem 3.2. Write the ASCII code for the 0 character.

Solution: The code for 0 is **48 =0b00110000 = 0x30**.

Problem 3.3. How is 0 written by the STM32 to the LCD controller.

Solution: For the number 0, the upper nibble 0011 is first written followed by the lower nibble 0000. This is done by

```
void SendByte (byte data)
{

 SendNibble(data >> 4); // send
    upper 4 bits 0011
 SendNibble(data); // send lower 4
     bits 0000
}
```

Problem 3.4. How is the nibble 0011 written to the LCD.

*The author is with the Department of Electrical Engineering, Indian Institute of Technology, Hyderabad 502285 India e-mail: gadepall@iith.ac.in. All content in this manual is released under GNU GPL. Free and open source.

Solution: This is done by the following function where data = 0011.

```
void SendNibble(byte data)
{
GPIOA->BRR = ~(data << 2) & 0
    b00111100;
GPIOA->BSRR = (data << 2) & 0
    b00111100;

 PulseEnableLine(); // clock 4
     bits into controller
}
```

The expression

$$GPIOA->BSRR = (data << 2)\&0b00111100 = 0b00001100. \quad (3.4.1)$$

This ensures that 11 is written to the pins A2-A3. Note that << indicates 2 left shifts. Similarly,

$$GPIOA->BRR = (data << 2)\&0b00111100 \quad (3.4.2)$$

ensures that 00 is written to the pins A4-A5. PulseEnableLine() provides a clock pulse used to write the nibble 0011 to the LCD.

Problem 3.5. Which pins of the STM32 are used for what purpose?

Solution: The A2-A5 pins of the STM32 are used for pushing the upper/lower data nibble to the DB4-DB7 pins of the LCD using the BRR and BSRR registers. The A0-A1 pins are used for Register Select and EN for the LCD.

Problem 3.6. What is Register Select?

Solution: Register Select = 0 implies that LCD configuration commands are being written. For example, cursor on/off, clearing display, number of lines, etc... Register Select = 1 implies that characters are being writen to the LCD.

4 Project

Develop an arithmetic calculator using the STM32 along with the LCD.