ELEC-240 Lab5

Interfacing LCD Display to the STM32F429 Nucleo-144 Development Board

1 Introduction

The LCD display is an alphanumeric 16x2 (16 chars across by 2 lines down) interfaced via a parallel data bus.

1.1 Learning Outcomes

- ❖ By the end of this lab exercise you should be able to:

1. Demonstrate an understanding of how the LCD is controlled, specifically:
    a) Timing
    b) Control line functions
    c) Commands
2. Produce code to write single characters and strings to the LCD
3. Control the LCD in the most timing efficient manner

1.2 LCD Interface

The interface consists of 16 pins including:

- 8 data lines (D0 → D7)
- 3 control lines
    a) RS - Register Select signal
        - ‘1’ = Text command
        - ‘0’ = Instruction command
    b) R/W - Read/Write signal
        - ‘1’ = Read command
        - ‘0’ = Write command
    c) E - Enable signal, it idles low and needs to be pulsed high for at least 10µs to apply a command to the LCD.
- 2 power supply lines (VCC, GND)
- 1 contrast control line (VO)
- 2 backlight LED lines (A, K)

1.3 LCD Busy

The LCD runs considerably slower than the micro-controller, therefore before we can apply commands to the LCD we must first check if it is BUSY. This is done by sending a ***Read Command*** (R/W=`1') along with the ***Instruction command*** (RS=`0') and then applying a pulse to ***Enable*** (E=’0’ →’1’→ ‘0’) then monitor Bit 7 (Busy) on the data bus:

1. bit7 = `1', LCD is busy
2. bit7 = `0', LCD is not busy

The LCD can operate using either a 4Bit or 8Bit data bus.

Initially we will operate it in 8bit mode (default power-on mode) using GPIOD pins 0-7 as the 8-bit data bus and GPIOD pins 11,12,13 for the control lines RS, RW, E respectively. These are defined in ***LCD.h***

To set up the LCD we must first send a sequence of instructions to select the number of bits, number of lines, font, cursor mode, etc; making sure we check the busy flag each time. R/W and RS should both be ‘0’ during initialisation.

Task 1

1. Connect the LCD display to the Nucleo-144 development board using the information provided in Section 1 above along with the LCD datasheet available in Table 1.
2. Download ***LCD Example Code.zip*** from the DLE, extract and run the code. This example code prints a single character `A' to the display.

Task 2

Initially the ***WaitLCDBusy*** subroutine, uses a simple blocking delay to hold up the micro-controller and allow the LCD to process a command. This is very time inefficient as the LCD can take a variable time to process a command but the micro-controller is always being delayed for the longest possible time which slows the operations down considerably.

Modify the code so the micro-controller checks the busy flag instead of using a delay. The following steps should be performed:

1. Configures the data bus lines as digital inputs (call macro ***set_LCD_bus_input()*** to do this).
2. Apply read command instruction to LCD. (R/W = ‘1’ call macro ***set_LCD_RW()***, RS = ’0’ using ***clr_LCD_RS()***, See Section 1.2,)
3. Set Enable bit (use macro)
4. Read LCD port (***port= LCD_PORT->IDR***) and mask ***busy*** bit (bit 7) (See Section 1.3)
5. Reset Enable bit
6. Repeat steps 3-5 while ***busy*** (bit7) is high.
7. Configure data bus lines as digital outputs (call macro ***set_LCD_bus_output()***)

Task 3

Develop code to:

1. Clear the display
2. Write a message string “hello world” to the display
3. Select the top or bottom line of the LCD and select the print position on the line.

Task 4

Change the code so the LCD can be driven from a 4 Bit Data Bus

Task 5

Develop code that will display the value of a variable on the LCD screen both in decimal and hexadecimal.

Task 6

Integrate code developed in previous lab in ***delay.c*** into the project so that the LCD functions use a precise microsecond delay that uses a timer and not the blocking delay given in the example code.

Task 7

Develop code that will print user defined characters to the LCD screen.

2 Support Documentation

Document Name	Contained Information
UM1974 User manual	• Pin identification and the supported special functions • Circuit schematics • Jumper and component identification • Header pinouts
RM0090 Reference manual	• MCU memory and peripherals architecture • Peripheral control registers, addresses and bit-fields
LCD Datasheet	• Electrical Characteristics • Interface Pin Function • Timing Characteristics

Table 1: Table of relevant support documentation for Nucleo-144 development boards (The document names are hyperlinks, please click on them to access the documents)