E C E 315: Introductory Microprocessor Laboratory

LAB #4

Serial Communication

1 Design Objectives

- Learn about SPI interface, SPI peripheral initialization and serial transmission of data
- Develop device drivers for communicating with display LCD
- Learn about UART interface, UART peripheral initialization and serial reception of data
- Develop code for communicating with ultrasonic sensor

1.1 Background

There are three parts to this lab. The first part would be focused on the UART peripheral, its initialization and communication with the ultrasonic sensor. Read the MaxBotix range finder's datasheet for more information about interfacing.

The second part would be focused on the SPI peripheral, its initialization and communication with the LCD display. Read the DOGS Graphic series LCD datasheet for more information about interfacing. This part will also involve writing device drivers using the basic SPI Tx routines to achieve the display functionality.

The third and the last part of this lab would be focused on displaying the distance measured by the range finder on the LCD with a resolution of 1 decimal position. (Eg: 12.1 inches).

1.2 Interfacing with Range Finder using UART

There are three MaxBotix range finders found on the ECE315 robot platform. One of the sensors (J8 or the left facing sensor) measures the distance and outputs the distance information serially via the UART interface. You will need to configure the GPIO pins connected to TIVA_SONAR_TX and TIVA_SONAR_RX as digital pins with alternate function behavior corresponding to UART.

In order to initialize the UART peripheral, follow the same implementation as that of `serialDebugInit()` defined within `boardUtil.c` and `boardUtil.h`. Notice how a `struct` has been defined with the UART's base address and passed as argument to the init function. Some of the high-level steps for interfacing are as follows:

1) Initialize the corresponding UART peripheral in `boardUtil.c`

2) Enable UART RX interrupt

3) Create a UART ISR in `interrupts.c` that alerts the main program when a valid distance has arrived (Refer to range finder's datasheet for more information parsing to obtain valid distance)

Circular Buffers need not be implemented but UART FIFOs should be enabled.

1.3 Interfacing with LCD Display using SPI

The display communicates to the microcontroller through an SPI interface. You will need to configure the GPIO pins connected to LCD_SPI_MOSI, LCD_SPI_CS, and LCD_SPI_CLK as digital pins with alternate function behavior corresponding to SPI. The GPIO pins connected to LCD_CD and N_RST_LCD needs to be configured only as digital outputs. You do not need to enable alternate function for these pins since they are going to act as normal outputs.

In this case, the initialization function `ece315_lcdInit()` has been provided to you inside `lcd.c`. A function named `spiTx()` is also made available to you from the compiled library. Your task will be to use this function and develop a higher level abstracted software layer that enables LCD initialization and character display. The exact functions to be developed are the six functions declared in `lcd.c`. Use the LCD's datasheet to learn and understand how this can be accomplished. Some of the high-level steps for interfacing are as follows:

1) Configure LCD_CD and N_RST_LCD as digital outputs and set them high / low as required. (Refer LCD's datasheet for this step).

2) Call the function `ece315_lcdInit()` to initialize the corresponding SPI peripheral

3) Complete the definitions for the six functions declared within `lcd.c`. Pay particular attention to how the pin LCD_CD must behave for different commands issued to the LCD. (Refer LCD's datasheet for this step)

4) Replace all `UNKNOWNS` inside `lcd.h` with proper definitions for peripheral bases and pins connected to LCD. (Refer the robot's schematics for this step)

5) Utilize the functions that you defined in step 3 to display characters on the LCD. `fonts.c` and `fonts.h` have been provided as a look-up table for character generation.

6) Fill out the worksheet at the end of the lab manual. Have the TA verify the worksheet prior to writing code for the LCD display. This worksheet will help you to determine the correct sequence of commands required to write to the LCD display.

1.4 What to Demo and Deliver

1) Display the distance measured from the range finder on the LCD with a resolution of 1 decimal position.

2) Display the direction of robot's movement on the LCD screen.

   The display should always read:

   ```
   DIST: xx.x
   DIR: yyyy
   ```

   (where 'x' is a number between 0 - 9 and 'y' is either FWD or REV or TURN)

3) Display the RX waveform for UART and MOSI-SCK waveforms for SPI. Include a screen-capture of the oscilloscope in your deliverable.

E C E 315: Introductory Microprocessor Laboratory

1.5 LCD Work Sheet

Make sure that you examine page 6 of the data sheet to properly initialize the LCD before writing any data to the display.

Bringing the LCD Out of Reset

What logic value does N_RST_LCD need to be set to bring the LCD out of reset?

Entering Command Mode

What logic value does CD need to be set to in order to enter command mode?

What GPIO pin is connected to CD?

Setting the Active Page

What is the logic value of CD when you set the active page?

If you wanted to set the page address to page 0, what would be the packet(s) that you would send over the SPI interface?

Setting Active Column

What is the logic value of CD when you set the active column?

If you wanted to set the column address to column 10, what would be the packet(s) that you would send over the SPI interface?

Writing Pixels to the LCD

What is the logic value of CD when you are writing data (pixels) to the LCD?

Write the code necessary to turn on all of the pixels for page 0, column 10. Assume that you have completed the functions `lcdSetPage` and `lcdSetColumn`.

When testing your LCD code, verify that you can turn on all of the pixels for page 0, column 10 before trying to implement writing entire characters to the LCD.