Mazar Farran

Jacob Ley

EEC 172, Section A03

January 24, 2016

Lab 2 Report

**Introduction:**

For this lab, the CC3200 System on a Chip was used to drive an Adafruit display via SPI. Low level procedures *writeData()* and *writeCommand()* were implemented to allow for drawing on the display. Using this functionality, two CC3200s could communicate via UART and print received messages on the display. Each board is also connected to an individual pc (Console) through another UART to receive text input to draw on the opposite board.

**SPI Waveform Notes:**

1. The first byte of data is 0xFD, which is also the first command sent by the program (Macro SSD1351\_CMD\_COMMANDLOCK, line 123 of Adafruit\_OLED.c).

When the MISO (GPIO pin 15) is low, the signal is a command.

When it is high, the signal is data.

The enable signal defines the regions that each signal is being passed over.

1. The signal being sent over is for the board to print "1234." (Message2() function) The UART signals correspond to the ascii hex values of 1, 2, 3, 4 (31-34). The SPI signals then start up and represent the write commands/data, which is not immediately discernable as 1234. It is the drawchar() function being called from the interrupt ReadHandler().

**Difficulties:**

Setting up the display was the most difficult part of the lab. It required understanding the SPI API and determining the appropriate configurations by looking at timing diagrams on the LED schematic.

Setting up interrupts was also difficult. At first, the LED would not display characters until after a considerable delay. It turned out that the interrupt handler did not account for receive timeout interrupts.

If the string being passed is too big, the FIFO will fill up faster than the interrupt could handle the input. We had to delay the signals being sent because a letter could be sent quickly in 8-bit segments, while receiving and drawing the signals takes a multiple 8-bit commands and data segments.