CMPE121L

Xingyu Zhou

Anujan Varma

2019/5/17

**Lab5 Report**

**Introduction:**

This lab was basically the same as lab4, but instead of using UART, we would be using USB instead of UART to transmit/receive data. Lab5 is consisted of two parts, the first part was to control the LED brightness with raspberry Pi and the second part was to generate a sinewave using PSoC board and the Pi together.

**Part1: Remote Control of LED brightness with PWM**

A close up of a piece of paper

Description generated with high confidence

**Figure 1: Part1 Top Design**

The set up process was almost the same as lab4, but we were asked to use USB instead of UART to do the transmission. To achieve the 1 millisecond delay, I just added a CyDelay at the end of my infinite loop. For the Pi code, I used the example code as a template and modified it to receive the potentiometer reading and write them to the PWM. This time however, instead of using a scalar to scale up my potentiometer reading, I connected it directly to the 5v VDD on the PSoC (it was connected to VDDIO before, which was 3.3v, that was the reason I was getting a reading around 691). When I was testing the program, the potentiometer reading could only go up to 1020, so I added a condition to forced the reading to 1024 when it was greater or equal to 1020.

**Result for part1:**

I was able to toggle the LED from completely off to completely on.

**Part 2: Analog Loopback through the Raspberry Pi**

A screenshot of a cell phone

Description generated with high confidence

**Figure 2: Part2 Top Design**

Again it was mostly the same as part2 of lab4 with the UART being switched to USB. The basic idea was to convert the analog signal to digital signal, transmit the data to Pi through USB, receive the data back from Pi and convert the digital signal back to analog. Since Rx and Tx interrupt were unavailable this time, everything was done inside of the infinite for loop. Two DMA ISRs (DMA1 and DMA2) were used only for channel switching (because we had ping pong buffer). For the coding part, I was following the example code on the PSoC creator and added some modification. For the Pi code, I was following the example code given by the professor as well. When I was testing the program, I noticed that the Pi was actually receiving and sending out the data, but nothing was showing up on the oscilloscope. After running the debugger, I realized that the condition OUT\_BUFFER\_FULL could never be satisfied, thus the data was never sent to the VDAC. After changing the condition to BUFFER\_EMPTY, the program was able to enter the if loop and execute the data transfer portion.

**Results for part2:**

I was having a lot of issues with my PSoC board this time. I kept getting “Reset failed” “Get configuration failed” as well as “cannot claim interface”. The professor said it might be the blocking calls in the OUT transfer so I added an if loop to check the buffer condition before transferring the data to VDAC. It was working fine but at the time of check off I was getting the same error again. (I didn’t check for buffer condition at the first time and it was working fine).

At the end I had to use my friend’s PSoC to demonstrate my part 2. I believe it was the hardware issue since the green LED on my board was gradually turning on and off when connected to the Pi while my friend’s LED stayed on all the way.

Anyway, I was able to display the waveform without data loss at a frequency at 10kHz for a long time.

**Questions for part2:**

I started to get data looses when the input frequency reached 800Hz. The data loss became more and more serious as I continued increasing the frequency and eventually failed to generate a sinewave at 2.1k Hz.

**Conclusion:**

The hardest part of this lab was to read and understand how to configure USB and how to use it for data manipulation. Once I had a basic understating of what was going on, I was able to follow the code example and design each part. It’s nice to learn another new way to manipulate memory as well as data transmission.

A screenshot of a cell phone

Description generated with very high confidence

**Figure 3: External Diagram for Lab5**