**EGRE 427**

Advanced Digital Design

Lab 3

Due Date: 2/26/2015

Student: Allen Woods

On my honor, I have neither given nor received aid on this assignment, and I pledge that I am in compliance with the VCU Honor System.

1. **Introduction:** The purpose of this lab was to build a client program and server program in which the server program runs in Linux on the Zybo and outputs a sine or square wave based off of input from the client. Using the SPI bus interface on the Zybo, the server program communicates with a digital-to-analog converter chip inserted into one of the board’s PMOD connectors. Using an oscilloscope, the output of the DAC can be read to verify the output.
2. **Lab Contents:** The client and server programs were built to ensure compliance with the lab requirements. The client program begins by prompting the user for information about the signal. It asks for the signal type, frequency, and amplitude of the signal. Each signal parameter is stored in individual strings and concatenated to another string buffer to serve as a data packet between the client and server. The client sends the data packet with the parameters to the server using TCP sockets and the server outputs the desired signal.

The server program listens for communication from the client on its opened socket. When a connection is established, a thread is created to manage the signal output. The server’s main() function reads the data from the socket and parses out the signal parameters to determine how to produce the signal. These signal parameters are stored in shared global variables which can be seen by the output function and are updated by main() when the client sends data packets containing the information to the server. The shared variables are regulated via mutual exclusion so as to prevent either process from altering their contents until the other process is finished with it.

The Zybo’s SPI core was used to output signal data to the DAC. The SPI core’s memory addresses were mapped to Linux virtual memory using the mmap() function. Referring to the Zybo Technical Reference Manual, the SPI core’s memory addresses were set with the necessary values to commence operation. These memory addresses were represented in virtual memory with void pointers typecast as volatile integer pointers.

The sine wave amplitude was implemented using a sine wave look up table generated using an online tool. The values of the LUT were multiplied by the value of the user-input amplitude and divided by the maximum possible output voltage to produce a voltage that would correspond to the amplitude input. The square wave amplitude was implemented in the same manner but its LUT only had two values: 0x00 and 0xFF for on and off. The sine wave frequency was calculated by multiplying the signal’s period (based on user-input frequency) to an integer value and running that many iterations of an empty for loop to create a delay in processing the signal. The integer value was picked from a table of integers found from trial and error of outputting the signal. The value picked from the table was dependent on the input frequency. The square wave frequency was implemented in a similar way but only required two different integer values.

1. **Problems:** Completing this lab was incredibly difficult because of the difficulty of obtaining information on the SPI core and the use of virtual memory in Linux. Having examples of these components would have made things easier.
2. **Conclusion:** The client program can be run on a desktop connected to the Zybo and the signal output can be verified with an oscilloscope.