Lab Report 7

IR-Remote Peripheral for Linux System

Christopher Padilla

ECEN 449 - 504

Purpose/Introduction

- The purpose of this lab is to guide students through the process creating pulse demodulation hardware for the IR signal of a television remote control. The IR detection hardware is built on a breadboard, and from that we use a custom peripheral IP and Verilog HDL to see the messages generated by the remote control on a terminal.

Procedure

- The first part of the lab was to build the physical IR receiver circuit on a breadboard and observe the signal on an oscilloscope. This is done by using proper resistor values, an IR phototransistor and an LM339 comparator.
- The last part of the lab was to create a custom hardware IP peripheral and interface it with software using the Xilinx Standard Development Kit. Similarly to lab 3, I was given templated Verilog files for my implementation and added the required active low IR_signal that we will receive from the breadboard and wrote the demodulation user logic that the ARM Processor in the Zynq chip performs.

Results

- After building the circuit and hooking it up to the oscilloscope, I was able to see the active low IR signal. Pulse-width modulation means the message depends on the width of the individual segments. The TV remote initially sends a start signal; then from there, I was able to determine whether the signals that follow were 0 or 1 depending on how wide they were (how long the IR signal remained low) and verify the results using the sample command codes provided in the lab manual.
- To display show the messages, in the test software I used an infinite loop to continuously poll the value of register 0 and detect any changes. After creating the software to show the demodulated message, I was able to see the correct values display on the PICOCOM terminal for the volume up, channel up, channel 1 and channel 2 buttons as shown in the lab manual. This means that the my FPGA's demodulation hardware is fully functional and ready to decode any of the TV remote's IR signals.

Conclusion

- The overall concept of this lab was pretty simple: we create the hardware (on the breadboard and on the FPGA) to demodulate a message from an IR signal of a TV remote and display it on a terminal. However, if someone (me) is not careful, debugging can be tough, as small errors (like errors in storing a full message and counting signal values) can weigh heavily in the time it takes to implement the system.
- In this lab, I was able to develop a means of demodulating an IR signal using hardware and software to display it in a way that can be understood externally. Throughout the lab, I became deeply familiar with pulse-width modulation and am now able to see how it is applicable to many systems.

Questions

-

Button Pressed	**Demodulated Message**
Volume Up	0x490
Volume Down	0xc90
Channel Up	0x90
Channel Down	0x890
Stop	0x7b0
Play	0xfb0
Channel 1	0x10
Channel 2	0x810
Channel 3	0x410
Channel 4	0xc10

- On the press of a button by the average person, the same message is approximately sent anywhere from 3 to 5 times (see PICOCOM output for sampling). Multiple messages are sent to allow the hardware that's demodulating to receive the signal at least once in cases of high noise or error. Also, it may be used to continuously perform the same action (in a similar manner that volume up continuously raises the volume of a TV as long as the button is held).
- I did this using two blocking assignments that execute one after the other; the newMessage signal goes high and then quickly goes back to zero. The processor may use this signal to send a signal to other components of the operating system that a new message was received and can be the basis for an interrupt-driven system (next lab).