Lab Assignment 4

Controlling the Wiimote with

Object-Oriented Programming

Saul Reyna, Yuheng Dong

[reyna.s@husky.neu.edu](mailto:reyna.s@husky.neu.edu)

dong.yuh@husky.neu.edu

Submit date: 2/19/2020

Due Date: 2/19/2020

**Abstract**

This lab consisted introduced a new piece of equipment: a Wii remote and a Bluetooth adapter. The adapter was connected to the ZedBoard, which allowed a Bluetooth connection between the board and the remote control. After being familiarized with the controller and how to read the information given by the remote, that information was used to control the lights of the board, dependent on the acceleration of the remote.

# Introduction

Using a Bluetooth dongle, a connection between a Wii remote and a ZedBoard is established. With the board’s implementation of Bluetooth, the BlueZ stack, the connection is established by syncing the devices while the remote is in pairing mode. Since the board doesn’t have a pairing button like the remote, the MAC address of the Wii remote is added to a list of trusted devices in the board. Once pairing is complete, information is read from the connection and formatted to a readable form. The readable form of the information is then used to control I/O of the board.

# Lab Discussion

Equipment used in this lab include

* Zedboard (ZYNQ SoC)
  + Dual ARM Cortex – A9 MPCore with CoreSight
  + Zynq-7000 AP SoC XC7Z020-1CLG484
* Xillinux OS
* CoE lab desktop
* Wii remote
* Bluetooth adpater

# Results and Analysis

**Assignment 4.1**

The program for this assignment involved reading button events from the Wii remote and printing the code of the button and its associated value. The value in this program represents the ON and OFF state of the button, or whether the button was pressed or not. The codes for each of the buttons

**Assignment 4.2**

This assignment consisted of converting a program that read the hex dump from the Wii remote and printed out the code values for whenever a button was pressed into an object-oriented program. The program has an object with a constructor that opens the file for the hex dump for button pressed and assigns its file descriptor member. The object also has two methods, one that is a infinite loop that is constantly looking for any button presses and the other that prints out the relevant information from that event, such as the state of the button (pressed or not) and the code value for that given button.

**Assignment 4.3**

This assignment is similar to the previous assignment, only that instead of looking for button events, the program is listening for any acceleration information from the remote and printing the dimension of the acceleration and the value of said acceleration.

**Assignment 4.4**

The program for this assignment involved creating a new function to the WiimoteAccel class that would light up the LEDs on the board depending on the acceleration of the remote. For example, if the remote was still on the table, meaning no acceleration is occurring, no LEDs would be turned on. If a max acceleration of 100 was occurring, all LEDs would be one. To know how many LED’s had to be turned on, the acceleration would be divided by 12.5 and each light would be turned on, one by one, until it reached the value calculated by the division done previously.

# Conclusion

By establishing a Bluetooth connection between a Wii remote and a ZedBoard, information read from the connection, which contained information such as button events and acceleration events occurring on the Wii remote, was used to control the I/O of the board. By reading the values of buttons or acceleration, those values could now be passed unto previously created ZedBoard classes as inputs.

# References

1. Michael Benjamin, “*Lab Report Guide*”, Northeastern University, January 18 2006.