EE357 Fall 2012 Final Project
Interim Report

Due: 5 PM., 11/27/2012

Sangmook Johnny Jung and Sheldon Kwok
sangmooj@usc.edu and sheldonk@usc.edu

1. Project summary

For this project, we are creating a Wii Nunchuk-controlled implementation of the arcade game, Phoenix[1], released in 1980 by Atari, with the MCF52259 board. This is essentially a two-dimensional shooter game where the Nunchuk will control a ship that can shoot lasers at incoming spaceships coming from the top of a LCD screen. The objective of the game is to survive and kill incoming waves of spaceships using bullets and bombs. The main motivation for this project comes from our interest in Human Computer Interaction, and we felt that using the Wii Nunchuk would be natural interface for the game that we are implementing. The Wiimote[2] and Wii Nunchuk[3][4][5] peripherals have been used in the past often (especially with the Arduino board), but we have not seen any uses of them with the Coldfire MCF52259 board. We think that this will be an interesting challenge that can result in a fun game to play that uses a controller (the Nunchuk) that is familiar and natural to use.

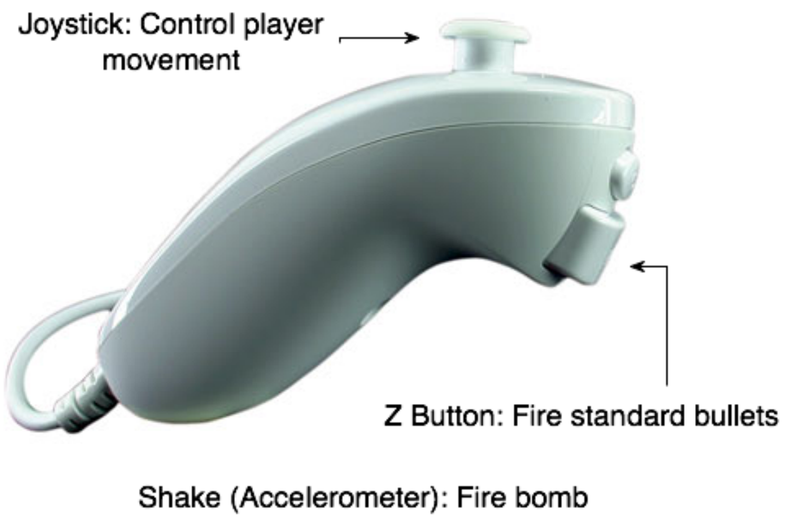
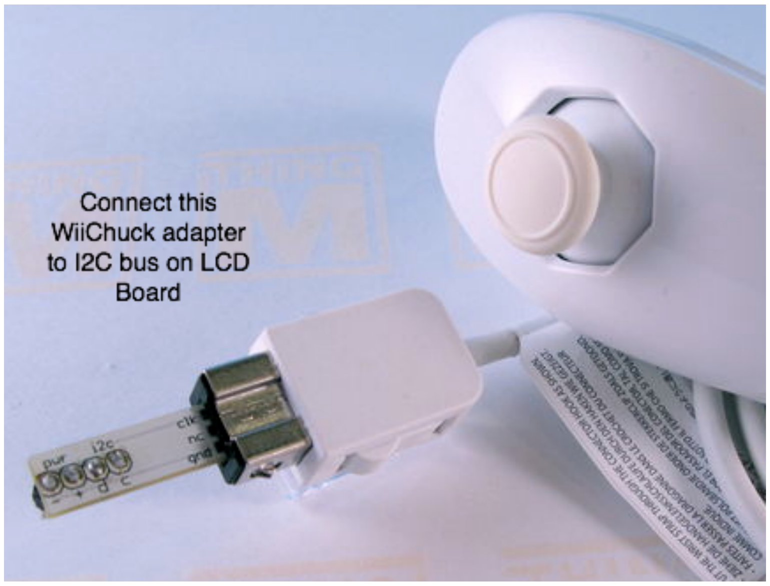

Joystick: Control player movement
Z Button: Fire standard bullets
Shake (Accelerometer): Fire bomb

Figure 1:
Nunchuk and Controls

The Wii Nunchuk will serve as a controller as shown in the above figure. The joystick on the front of the Nunchuk will control the ship's movement from left to right, and the large button on the back of the Nunchuk (the Z button) will be used for firing standard bullets. Additionally, using the built-in accelerometer in the Nunchuk, the player can vigorously shake the Nunchuk controller to activate a bomb. This Nunchuk will be connected to the LCD module using the I2C bus protocol[6][7], and the LCD monitor will be attached to the Coldfire board through the tower interface. The Nunchuk achieves its connection using a special WiiChuck adapter[8][9] as shown below. The I2C bus protocol will allow the Coldfire to read in the Nunchuk joystick, button and accelerator data and this will be used as input for the game. Our implementation of the game will then process this data, and generate output to the LCD with the correct game state and representation of all the individual pieces in the game.

Figure 2:
Nunchuk and
WiiChuck adapter



2. Contribution

Johnny will contribute by implementing the Wii Nunchuk hardware input interface to the Coldfire board and developing the proper back-end logic to handle and process this input (which may include writing device drivers). Additionally, Johnny will develop some of the back-end logic for the game engine. Sheldon will be in charge of designing and implementing the actual game itself. Sheldon will design the game engine and graphics for the game. Both of us will work to make sure the interface between the controller and the game will work to ensure that the proper input is sent to the game controller engine. We will both also work on documenting and testing the game as well.

3. Current progress

Johnny has researched the I2C bus protocol and how the Coldfire board can interact with the Nunchuk. He has started to test communication between the Wii Nunchuk and the Coldfire board. At the present moment, it hasn't been very successful, but the proposed plan is to use the I2C modules on the LCD board to connect the Wii Nunchuk to the Coldfire board. Additionally, Johnny has looked into the data format specification for the Nunchuk I2C data and has started to write backend code for the game to handle this input format.

Sheldon has currently designed the game engine and graphics. Knowledge of the LCD screen implementation has been acquired through completing the Timer lab. The skeleton for the game code has been written. The ship will be represented by the character 'A', bullets will be represented with '.', and the enemies will be represented with the character 'Y'.

4. Work to be done

Johnny will need to continue working on establishing the I2C bus protocol interface between the Nunchuk and the Coldfire board. This will most likely be achieved by utilizing existing code from the LCD lab, along with I2C code from previous labs and sample code given. Additionally, he'll need to write the appropriate code to process the given Nunchuk input for the game, and then write the back-end logic for the game that will use the Nunchuk input data. Sheldon must still implement the game functions within the skeleton code. This will involve handling the inputs from the Wii Nunchuk and displaying the graphics. Major features include drawing of the game objects onto the screen, movement of the game objects, and collision detection of game objects. Both Johnny and Sheldon must figure out how the game engine will process input from the game controller and accurately control the player's ship.

5. Challenge

Johnny's main challenge is actually getting the I2C connection between the Nunchuk and Coldfire board working. There are no online references to refer to for the Coldfire board, and the documentation is sparse at best. There are only a few resources for connecting the Nunchuk to the Arduino board[10][11][12][13]. This may require some clever hacks to get it work, and probably will require some use of sample code or writing a new device driver.

Sheldon's main challenge is creating a fluid game using the LCD module. From previous experiences, he has noticed that the LCD module has a limited frame rate of approximately 8 frames per second. This would severely hamper the gameplay experience since the user can see the individual frames being drawn. Creating an accurate timer will be a challenge due to the limited capabilities of the LCD module. We both also have the challenge of integrating the game engine with the actual Nunchuk input, by correctly utilizing the Nunchuk input data. It will be difficult to accurately interpret the controller input and make sure the game engine responds properly. User input may be delayed like the switches on the Coldfire board. Our goal is to minimize the effect this has on the users' experience, and create a game experience that is natural and fun to play.

References / Sources

[1] http://en.wikipedia.org/wiki/Phoenix_(video_game)

[2] http://johnnylee.net/projects/wii/

[3] http://www.windmeadow.com/node/42

[4] http://letsmakerobots.com/node/5684

[5] http://todbot.com/blog/2007/11/24/bionic-arduino-class-notes-3-4/

[6] http://wiibrew.org/wiki/Wiimote/Extension_Controllers/

[7] http://wiibrew.org/wiki/Wiimote/Extension_Controllers/Nunchuck

[8] https://www.sparkfun.com/products/9281

[9] http://todbot.com/blog/2008/02/18/wiichuck-wii-nunchuck-adapter-available/

[10] http://www.britishideas.com/2011/08/20/reading-a-wii-nunchuck-using-i2c/

[11] http://forum.allaboutcircuits.com/showthread.php?t=34422

[12] http://www.nerdkits.com/forum/thread/972/

[13] http://dangerousprototypes.com/2009/08/19/bus-pirate-wii-nunchuck-quick-guide/

Figure 1: http://express.howstuffworks.com/gif/autopsy-wii-nunchuk.jpg

Figure 2: http://todbot.com/blog/2008/02/18/wiichuck-wii-nunchuck-adapter-available/