Stephen Hall and Razik Ahmed
TA: Harshad Desai

Lab 11 Report

A) Objectives

- **Overview**
  - **Objective**: To desgin, layout, construct, and test an embedded system.
  - **Roles and Responsibilities**: Razik will do everything related to software, which includes the drivers, test code, and final game code. Stephen will do the hardware, which includes designing the printed circuit board, modifying the box, ordering parts, and soldering. The clients are people that are bored of board games, people who want to play Battleship without al the hassle of keeping track of the pegs and ships.
- **Function Description**
  - **Functionality**: To play Battleship against a computer.
  - **Performance**: Current will be determined using the bench power supply which will help decide the battery. Initial breadboarding with the DP512 suggests that the system will take around 50 mA with 120 mA spikes.
  - **Usability**: There will be two different modules each with a C32 microcontroller powered by a battery, 128 x 64 pixel graphical LCD screen, six buttons for user input, 6 LEDs to signify a hit, and a speaker to play sound. A switch will also be on the outside of the box to turn power from the battery on or off. The two modules will be connected via a serial cable.
- **Deliverables**
  - **Reports**: The reports for Labs 8 and 11 will be written.

Outcomes: Objectives, hardware and software design, and measurement data for Lab 8 and Lab 11

B) Hardware Design

Schmatic Design is on page 3.

C) Software Design

The software initializes all the components first. After the game is initializied, the main goes into a do nothing for loop. The game operates on a finite state machine. The states are changed when an input capture interrupt triggers. Once the input capture is handled, the do nothing loop takes over again. This way, the power usage is dropped when nothing is being input. Some states advance after a certain

amount of time which is handled with a busy wait.Six of the IC/OC interrupts are used for switch interrupts. One OC interrupt is used for switch debouncing and the other OC is used for the music.

D) Measurement Data

Three test points were designed into the board to allow easy checks that there is 5 volts into the chip, 3.3 volts into the Xbee, and 2.5 volts into the DAC chip. A green LED is also on the board to tell if the power switch is on or off. We found that the current requirement for our system is 42 mA, so with a 9 volt battery that has 1200 mAhr our system will last 28.5 hours.

E) Analysis and Discussion (1 page maximum)

Once we received the board, we soldered it quickly so that we could ahve more time to test it. One problem we found was that the power pins are labeled red and black but they are backwards. After we changed those, all of our voltages were fine. The green power LED came on, the 3.3 V line was 3.28 V, the 2.5 V was 2.49 V, and the 5 V line was 4.96 V. Since everything was good in those repsects, we continued to test the code on the breadboarded version before porting DP512 code to C32 code. Initially, we wanted to have one module be able to talk to another by either the SCI port or the Xbee. We soon found out that our serial port didn't work correctly. We tried outputting 0x55 to the serial port at 9600 baud rate. On the scope we saw 0x55 but at 1200. So we changed the registers to multiply the baud rate by 8. Instead of getting 9600 on the next test, we got 5500 so we decided not to use the serial port anymore. We then focused only on single player mode.

We also found out that we have a problem with programming. Although we flashed the BDM, we were unable to to program without it. We could load a program only if both the serial port and the BDM were connected. If one or the other was not connected, the CodeWarrior wouldn't be able to find the board. Luckily our programs stayed in ROM so we just made sure to have a BDM whenever we programmed.

After all these problems, we worked on the box and porting the code. In the end we got single player working. The LCD would display the board. Six switches were inputs for up, left, down, right, select, and back. When an attack was made, the speaker would output a doppler-like whistle. If the attack was a hit, an explosion would play, otherwise nothing. If your attack hit, the six LEDs would flash, but this would not occur if an opponent hit you. A power switch was at the top for turning power on and off. Overall, we tried to make our box look as clean as possible. The holes were drilled in pre-determined spots and the squares were dremeled out so that everything fit perfectly. In the end everything worked well, besides the serial port.