Thomas Gibbons

November 29, 2016

ECE 3220

LAB 10

Objective

The objective of this lab was to become familiar with TS-7250 Embedded Board. We then also learned to develop applications using Eclipse or the terminal and using cross-compilation to compile the program for the server or the embedded board.

Discussion

In this lab we first learned how to setup the board to use. It required that we log into the server using the ssh command with –X and the corresponding email with our pawprint. From the nfs1 server, we are able to run some commands to work the boards. We used boardsetup # to setup the specific board we were using as all of the boards in the lab were numbered. We then reset the board and it rebooted in a couple minutes. Next, we used the boardlogin # command to access the file directories on that particular board. From the directory we can compile a program on the board since we are in the terminal of the board. This program will not be executable on the server or linux computer. It is made directly for the embedded system. Same applies if the program is compiled on the server it will only run on the server. This is why Eclipse can be used to cross-compile from the computer straight to the board using the special form of g++ with arm-linux. After the board is done being used we use the boardrelease command to let others use the board and we use the shutdown –h now command to properly shut down the board to cause no corruption.

We then used the template program given to us to show how to access memory locations within the board and simply light up the 3 LEDs. This was utilized in our coding assignment for the lab. Each word got translated to morse code and that morse code was then sent to the LEDs. So we wrote a simple if else statement method that had 3 options: dot, dash, or end of word.

```
if(a == '.'){//red
    *PBDDR |= 0x20;

    *PBDR |= 0x20;      // ON: write a 1 to port
    sleep(1);
    *PBDR &= ~0x20;     // OFF: write a 0 to port
    sleep(1);

}
```

```
else if(a == '-'){//yellow
    *PBDDR |= 0x40;

    *PBDR |= 0x40;  // ON: write a 1 to port
    sleep(1);
    *PBDR &= ~0x40; // OFF: write a 0 to port
    sleep(1);
}
```

If the character we sent was a dot it would access the red light. It would write a 1 to the port for the red light to turn on, then wait a second, then write a 0 to the port to turn it off, and then wait a

second. If the method was called by sending a dash it would do the same as before except by accessing the yellow light. And then if any other character is sent it would signify the end of the word. So while the translating is going on from word to morse code letter by letter we can also send those letters to the LED method to display. Then after each word we can send any sort of character to set off the green LED.

The output of this lab has morse code to the screen as well as a display of LEDs. Since the display of morse code is the same as in lab 8 and the LEDs cannot be displayed there is not really any extra output screens that I can display for this lab.

As always a link to my github account is https://github.com/tgibbons95/Lab10.