ECE 409 Hardware Lab 2 – Blink

(Weeks 12, 13)

Introduction

The set of activities contained within this laboratory will acquaint you with the hardware and software associated with sensor networks. Upon successful completion one should understand the concepts of programming boards, and motes. Simple Mote programming techniques will also be covered.

Hardware Setup

1. The programming board contained within the setup kit will be the main focus of this activity. It serves a dual purpose, acting as a main programming tool for all Motes as well as being the gateway between the PC and a particular sensor network. The MICA2 Mote is the communication device which can be programmed to perform specific tasks. The Sensor Module is responsible for gathering requested data.

2. Connect the board to your PC (or laptop) via USB interface.

IMPORTANT: It is important to **remove batteries from the MICA2 Mote** prior to establishing a connection to the programming board, the power switch on the Mote should be in the off position. The MIB510 supplies power to the MICA2 while a connection is present and introducing another power source will damage both components.

4. Pick one of the supplied MICA2 Motes and attach it firmly to the MIB510 programming board via the 51-pin male to female connector.

Basic Code Upload

5. Start a Cygwin session and change to the below directory:

```
/opt/tinyos-1.x/apps/Blink
```

This is where the necessary components of the **Blink** application reside. As you have learned in lecture or readings, NesC requires a specific file structure. More information regarding this nesC can be obtained from http://nescc.sourceforge.net/ and http://www.tinyos.net/tinyos-1.x/doc/index.html. Try to familiarize yourself with the files present in this directory and their functions. Can you explain the purpose of each one of these files?

6. While in the `/Blink` directory type: `make mica2`

The build should complete successfully and create a binary .srec image of the program. This can be verified by changing to the below directory:

```
/opt/tinyos-1.x/apps/Blink/build/mica2
```

Note: Watch for any warnings or errors. The last two lines will provide you with actual numbers of memory usage. Save a screenshot for your report.

7. To upload the code, type the following command:

```
make mica2 install mib510,/dev/ttySX
```

Where X corresponds to the COM port assigned to the serial port connection minus 1(ex. COM5 = ttyS4). Ensure that you have returned to the Blink application directory, otherwise an error will occur.

8. The desired code should now be uploading to the Mote. During the upload process the red LED labeled as **ISP** will stay illuminated. Upon successful completion the last three lines of this operation should read:

```
Atmel AVR Atmega128 is found.
Uploading:  flash

Fuse Extended Byte set to 0xff
```

Save a screenshot for your report. Can you explain what these three lines mean?
Hint: Try looking for Atmel on the internet.

9. If the above completed successfully, a red LED on the MIB510 and the MICA2 Mote labeled as D2 should now be **blinking in 1 second intervals**. Make sure you understand the programming process, which could be summed up as having four major steps. Write desired code, compile, correct errors, and upload to motes.

10. Alter the Blink code enabling the red LED to blink in 5 second intervals. If you are unsure which files to edit refer to the documentation mentioned earlier.

Note: If problems are encountered during code uploads, try pressing the RST MOTE button located in the upper left corner of the MIB510. This will reset the processors for both the programming board and the mote but will not erase the uploaded code.

11. To view a graphical representation of the program structure one has the option to generate HTML based documentation for any application. To accomplish this while in the /Blink directory type:

```
make mica2 docs
```

The created documents will be placed in:

```
/opt/tinyos-1.x/doc/nesdoc/mica2
```

This document can be accessed through Windows by browsing to:

```
C:\tinyos\cygwin\opt\tinyos-1.x\doc\nesdoc\mica2\index.html
```

The overall structure of TinyOS can be better comprehended using this graphical interface.

- Show your progress to the instructor for a sign off.

Motes and Radio Communication

12. The Motes responsible for wireless communication are capable of transmission at multiple frequencies. The frequency of the Mote you are using should be clearly stated on its side. In order to establish a link between multiple MICA2 Motes they must all be operating on the same frequency and channel.

Note: A Mote will only operate on the frequency it was designed for (ex. 916, 433, etc.)

13. Change to `/opt/tinyos-1.x/apps/CntToLedsAndRfm`

14. Using **vi** editor open the `Makefile` contained within the directory. Ensure that the following line is present:

```
include ../MakeXbowlocal
```

15. Change back to `/opt/tinyos-1.x/apps` and using **vi** open `MakeXbowlocal(May not already exist).` Within the file find the frequency corresponding to your Motes. Pick one of the channels and uncomment the corresponding line.

Ex. `CFLAGS = -DCC1K_DEFAULT_FREQ=CC1K_915_998_MHZ`

Alter the group ID for your network, locate the below line and change the hex number to your bench number, converting it to hex of course:

```
DEFAULT_LOCAL_GROUP=0x7
```

Note: This option will ensure the separation of mote networks running on the same frequency and channel within the room. All 8-bit values are acceptable, except 126 and 255. Save your changes and exit. We will use this file through this course to aid in the process of choosing a frequency and group ID.

16. Ensuring that no batteries are present and the power switch is in the off position, place a MICA2 Mote on the MIB510.

17. Power on the MIB510. Compile and install the `CntToLedsAndRfm` application onto one of the Motes. Remember the commands you use and save a screenshot of the compile for your report.

18. Power off MIB510. Remove the newly programmed Mote from the MIB510, install batteries to the Mote and turn the power switch to the ON position. You should now observe the three LED's on the MICA2 labeled D2, D3 and D4, blinking. Additionally, the Mote is now transmitting values over the radio.

19. Place another Mote on the programming board this time outfitting it with a different application named `RfmToLeds`. Upon successful program upload, remove the mote from the programming board, install batteries and turn the device on. Try turning the first Mote on and off and watch what happens. The `README` file in both application directories provides a description of what should be occurring.

20. Now, program another mote with the `CntToLedsAndRfm` application and turn it on, following the procedures described earlier. Can you understand more clearly what is happening now?

Note: If you are unable to establish communication, be sure to check the MICA2 operating frequency. Try moving the devices and ensure the antennas are secured in place.

- Show your progress to the instructor for a sign off.

Laboratory Report

Instructions

The questions are to be answered based on your lab experience and are due 1 week after the scheduled lab time. The report questions must be accurately answered, typed, diagrams computer generated and this template used. Include the questions on your report.

Basic Code Upload

1. In activity number 5 you explored the file structure and an application called Blink. List all the files present in the directory and explain their purpose.

2. Provide a screenshot showing the result of a program compilation in activity 6. Can you describe what the process of programming a series of motes with this application? What are the ROM and RAM values?

3. In activity 8 you uploaded the compiled code to the MICA2 motes provide a screenshot of a successful code upload. Explain the last three lines of the resulting output.

4. Activity number 11 requested that you generate documentation for your newly compiled application, please provide a screenshot showing this accomplishment.

5. Explain what the Blink application is trying to accomplish.

6. Why is the LED on the MICA2 mote able to blink without any batteries?

7. Please draw a topology of equipment used up to this point. Be as specific as possible including names and model numbers, labeling connections etc.

Motes and Radio Communication

8. In activity 17 you were asked to compile and upload the application `CntToLedsAndRfm` **to one of the motes, please provide a screenshot of the compilation process. What are the ROM and RAM values?**

9. What were the commands used in the above process?

10. You used two applications to complete this section of the lab (step 12-20). Please explain the role of each one of those applications.

11. Upon completion of this section (step 12-20) you should have a working group of programmed MICA2 motes, please draw a topology of the utilized equipment. Be as specific as possible including names and model numbers, labeling connections etc.

Sensor Network Basics Sign-offs

Name: ___________________

Basic Code upload (step 1 - 11)

Code upload completed, Blink is operational	___________________________
Code change completed to 5 second intervals	___________________________
Documentation generated	___________________________

Motes and Radio Communication (step 12 - 20)

Code upload completed, applications operational	___________________________
Student has understanding of experiment concept	___________________________