CS 3468
Lab 3

Jared Wallace

Objectives

1. Understand communication among sensors
2. Learn to read and use APIs (application programming interface) at `http://www.tinyos.net/tinyos-2.x/doc/nesdoc/micaz`

Tasks

Radio communication

Radio communication is used by sensors to exchange information. At least two parties are involved in communication: the sender and the receiver. Communication is done via the transmission of packets. Accordingly, a sensor that can send and receive via radio communication needs three components: *AMSenderC* (in sender), *AMReceiverC* (in receiver), and *ActiveMessageC* (as radio control). TinyOS provides a number of interfaces to abstract the underlying communications services and a number of components that provide (implement) these interfaces. All of these interfaces and components use a common message buffer abstraction, called message_t, which is implemented as a nesC struct (similar to a C struct).

1. System component MainC
2. System component new TimerMilliC() as TimerC
3. System component LedsC
4. System component new AMSenderC(RADIOID) as SenderC
5. System component new AMReceiverC(RADIOID) as RecverC
6. System component ActiveMessageC as AmsgC
7. Application component AppC

Reading and understanding an API

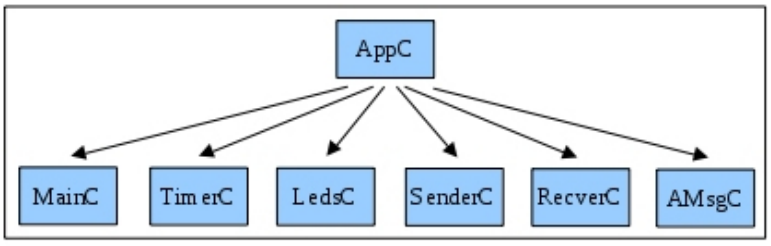
Now, open the API link `http://www.tinyos.net/tinyos-2.x/doc/nesdoc/micaz/`. Find the API documents of the first six system components.

Name	Description	Provides Interfaces
MainC		
TimerMilliC		
LedsC		
AMSenderC		
AMReceiverC		
ActiveMessageC		

Table 1: Fill in this component table as your reference for further programming.

Name	Components	Event functions	Description of events

Table 2: In this table, summarize all interfaces according to the component table. Use the API documents as reference.



The diagram shows a sensor application that can send and receive commands and also light LEDs according to commands. The application has seven components.

Read through and understand the source code of an application that communicates

Make sure you understand the following:

- How the seven components are wired through the interfaces
- When the events are called and how the events are handled
- How the AppC component makes use of system functions.
- How an AM message is defined.

Compile and run the example application

- Get two sensor IDs from the lab instructor. Use one for the sender and the other for the receiver. Modify Sender.h to include the receiver's ID. Note: You need to modify the command you use to install the program onto the mote! Use the following syntax:

```
su
(enter password)
make micaz install,[insert appropriate id] mib510,/dev/ttyUSB0
```

  If your id numbers are 3 and 4, where 3 is your sender id, then when installing the sender program you would write:

```
make micaz install,3 mib510,/dev/ttyUSB0
```

- Modify the given code of the sender to send the following commands periodically and light the LEDs according to the following table.
- Modify the code of the receiver to receive the commands and light the LEDs according to the table.
- Use the assigned sensor IDs when loading the executable into sensors. For example:

```
su
make micaz install,1 mib510,/dev/ttyUSB0
```

  for the sender (assuming *your* sender id is 1)

Command	Yellow LED	Green LED	Red LED
0	off	off	off
2	on	off	off
4	off	on	off
6	off	off	on
8	on	on	on

Convert your app to be more reliable

The problem with the source code you have now is that it does not ensure reliable communication. Modify the source code to enforce reliable communication with per-packet acknowledgment. In another words, modify your code such that the transmission of a packet is successful only if the sender gets the acknowledgment from the receiver. Read over the APIs on the website to discover how to accomplish this.

Next, design an experiment to check whether the radio communication is more reliable with the addition of the acknowledgment requirement. Collect data for analysis and use data as evidence for your findings.

Lab Report

1. Please demonstrate your program to the lab instructor and let him check your code at the end of the current lab project.
2. Your project report is due at the beginning of the next lab.
3. Grading criteria
    - Demonstration, 15 percent
    - Code, 15 percent
    - Report, 70 percent

Report instructions

Format:

1. Include your name and ID in the first page
2. Font size of at least 10pt
3. Single spaced
4. Maximum of 5 pages (I will take points off for exceeding this without any good reason)
5. Please submit as PDF online, and turn in a hard copy

Content:

1. Introduction (10 percent of the report grade) Please summarize the task of this lab and what you have learned in the lab
2. Implementation (30 percent of the report grade) Please describe in detail how your made your program. Show what components you used, how they are wired, and through what interfaces.
3. Experiment (30 percent of the report grade) Please describe in detail what can be observed from your program and explain how said observed behavior is a result of your code (and not happy coincidence).

More generally, with great power comes great dEnergy/dt