Program 3 {Modified April 29, 2011} **55 Points**

A Network of TelosB Sub-Networks
Due: Monday, May 2, 2011 at 10 a.m.

Introduction

The goal of this project is to have each team manage communications between two motes in their own sub-network and also interact with a base station and a target node. Optimistically, the bigger objective is to run an experiment in class on May 2nd involving all the sub-networks in the class.

Assignment

Figure 1 is a characterization of the setup that your team would use to test your implementation where the nodes labeled **Base Station** and **Target Node** will be implemented by a team **for extra credit** and the other two nodes represent your team's two-node sub-network. For this assignment, your subnetID is **defined on the Programming Teams' pdf file.** For sub-net communication use the assigned, dedicated frequency **defined in the same pdf file**. The initial action for your two motes is to turn off their Blue LEDs. Once they receive BeaconMsg's, the motes should turn on their Blue LEDs to indicate they are connected to the **Base Station**. The **Base Station** and the **Target Node** will only use the broadcast channel. The broadcast channel is **DEFAULT_FREQ_CHANNEL** (namely, 26).

Your subnet test experiment is as follows. The **Target Node** sends out a TargetMsg every **TARGETPERIOD** (with a constant transmit power) to the **TOS_BCAST_ADDR** over the broadcast channel. Both your sub-network motes should detect these messages because all sub-net motes are listening on their private channel and the broadcast channel and use the messages to 'find' the **Target Node.** Specifically, your motes need to communicate with each other on their private channel to determine which of the two motes is closest to the **Target Node** by using received signal strength (RSSI). The node in your sub-net closest to the target will be identified as the **Near Node (**the near node needs to be unique. Devise your own RSSI tie-breaker). Once this determination has been made, the **Near Node** turns on its Red LED. The other node, henceforth known as the **Far Node**, turns on its Green LED. Since the **Target node** is a mobile node, the identity of the **Near Node** and the **Far Node** may change over time.

The **Near Node** must communicate its nodeID occasionally to the **Base Station** over the broadcast channel via a ReportMsg. This occurs in two situations:

1. A ReportMsg is required every time the identity of the **Near Node** is switched to the other mote.

2. The **Base Station** may inject a REQUEST for a ReportMsg on the broadcast channel.

The ReportMsg should **be sent to** the **Base Station** after either of the two possible events. The ReportMsg contains a TimeStamp which is the time the message was generated according to the **Base Station** clock. Since each node keeps its own local time, you might need a time synchronization protocol **{Note – given the lack of time, this synchronization is beyond the scope of program 3}**. The request field of the ReportMsg is explained below.

The **Base Station** sends out BeaconMsg's every **BEACONPERIOD** over the broadcast channel using the **TOS_BCAST_ADDR**. Simple beacons advertise the presence of the **Base Station** and its local clock. Complex beacon messages additionally serve as a REQUEST for a ReportMsg from an individual sub-network. The request field in the BeaconMsg identifies by subnetID the specific sub-net to whom the request is being sent. Hence, the resulting ReportMsg must return the value of the request field.

BeaconMsg's are also employed to evaluate connectivity. The objective is to have every mote be able to directly communicate with the **Base Station**. If a node does not hear the Base Station for five **BEACONPERIODs**, it should it is disconnected from the **Base Station** and should indicate this by turning its .Blue LED off. Since all motes start with their Blue LED's off, once a mote successfully receives a BeaconMsg of either type, it should turn its Blue LED on. Hence Blue LED on is visible signal that the mote is connected to the **Base Station**.

You will have to test your design against the working implementation of the **Base Station** and **Target Node**. Thus the actual value of **BEACONPERIOD** and **TARGETPERIOD** might change and should be used as constants in your design.

You are permitted and probably should define new message types. When creating new active message IDs for your internal subnet, use numbers that indicate your subnetID in the first digit (e..g, if you are subnet 3, your AM IDs could be 31, 32, 33, …) Place new messages and new definitions in a new.h file. **MsgProgram3.h has been** provided by the TA to identify the other message types (TargetMsg, ReportMsg and BeaconMsg).

To control the radio settings, you must use the RadioControl interfaces as in Program 2. **The Base Station and Target Node have already been implemented, but they cannot be fully tested until one team completes its sub-net functionality. Once we have confirmed the Base Station and Target Node working, we will make this code available so that teams with four motes can download this code to two of their motes. Note, the 'drop-dead' deadline for this assignment is the beginning of class on Monday, May 2nd.**

Deliverables

Your team will need to provide a concise report detailing your design decisions. This should explain protocols chosen, tradeoffs considered and rationale for your key decisions. As with the previous programs, your team needs to turn in a tarred/zipped file that contains source code, make files and a README file. The README is critically important for receiving partial credit on this assignment.

While we will attempt a live demonstration in class on May 2nd, each team will need to arrange a private demo slot with the TA on May 2nd or 3rd to go over your implementation and to give the TA all your motes.

Figure 1 - Group Experimental Set Up