Chloe Norris

MoteWorks Summary

Sections 6 and 7

Section 6 of the MoteWorks guide shows how to create a Mote firmware application that reads light sensor data from a sensor board, sending message containing sensor data through the Mote serial port, sending messages containing sensor data over the Mote radio to another Mote plugged into the programming board, using *XServe* to parse packets on a PC, and using *XSniffer* to display the sensor data messages on a PC.

The following are hardware requirements for the sensing applications in this summary: two standard edition Motes, one sensor or data acquisition board, one gateway board, and a Windows PC with *MoteWorks* installed.

A simple sensing application would take light readings using the MTX300/310 or MDA100 sensing boards, use the More serial port (UART) and radio to send sensor data to the base station, and compile and debug if necessary. To get started in making this application, the first step would be to create the application folder (directory) where all of the needed application code and other files will be stored.

The applications configuration is located in the MyApp.nc file. The Photo component is used to send messages over the serial port and radio. To create an application's configuration, certain code would need to be saved into a new document in Programmer's Notepad. When the application is installed and running on the Mote you should see the red, green and yellow LED's blinking every second. Each LED indicated progression of firing the timer, sampling the light sensor and then sending the message back to the base station.

XServe is used to display the sensor message packet contents as they arrive on the PC over the serial port. *XServe* has many different unique features. *XServe* is a program that runs within a Cygwin command prompt window.

To send sensor data over the radio, only one change would be needed in the code of the MyAppM.nc file: (the changed code would read to this) if (call SendMsg.send(TOS BCAST ADDR,sizeof(XDataMsg), &msg buffer) != SUCCESS). The SendMsg.send command decides where the message packet should be sent.TOS_BCASE_ADDR tells the communications component to send the message through the radio. This sends the message to any Mote within the range. If we want to send the message to one specific base station we can set this parameter value to 0. The Mote plugged into the base station always has a node id of 0.

XSniffer can be used to eavesdrop on messages sent over the Mote radios. It also is used to monitor the messages sent from out modified sensing application. To view sensor data sent over the radio with *XSniffer*, we first need to modify the sensing application in the ***/lesson_3*** folder onto a Mote. This can be done by loading the *MyApp.nc* file from /lesson_3 into the programmer's notepad. Then select Tools>shell. When prompted for parameters, type in **make mica2 install, 1 mib510, com1**. Remove the Mote from the programming board, plug one of the sensorboards onto the Mote and turn it on. You should see three LEDs blinking ever second. Then install the *XSniffer* application onto another Mote that remains plugged into your programming board (base station). Install this application with a node id of 2 using Programmer's Notepad. Start the *XSniffer* application by double clicking on the icon

located on your desktop. Click on the "Options" tap and select "General Packet Type" radio button. Go back to the Log tab, select the COM port that is connected to the programming board and then click on Start to begin "sniffing" the radio traffic. Eventually, you should see message packets displayed in *XSniffer*. Each time the LEDs blink, you should see a new message captured by XSniffer.

There are two additional features in the *MyApp* application form section 5. First, we are sampling the sensorboard light sensor. Second, we are building a message packet that includes this light sensor value and sending it back to the base station. The first thing we need to do when building a sensing application is to specify the sensorboard we went to use. In order to sample the light sensor on the sensor board we need to include a component named Photo in our configuration file. In order to send a message containing the sensor data back to the base station we need access to the TinyOS communication component named GenericComm, which is used to send messages through the UTART port over to the radio depending on the destination node address specified.

All of Crossbow's sensor and data acquisition boards are supported with *XSensor* enabled applications. *XSensor* applications are test applications for Crossbow's sensor data acquisition boards. They allow users to quickly and easily test sensor and data acquisition boards when attached to Mote. *XSensor* applications send data over one hop so all test Motes must be within RF range of the base station.

Section 7 discusses the *XMesh* enabled sensing application. The hardware requirements are as follows: three motes, one sensor acquisition board, one gateway/programming board, and a PC with *MoteWorks & Moteview* installed.

Simple sensing application using the *XMesh* multi-hop networking service would use the Mote radio to send sensor data to the base station and compile and debug if necessary. To get started, you would first create the application folder (directory) where code and files would be stored. The first step in creating an application is to type in the *Makefile*. You can copy and paste this file from the subdirectory /***lesson_4*** into ***/myapp***. When finished save the file.

The next step is to verify messages are being received at the base station by running the *XServe* application. The following summarizes how to use XSniffer to view sensor data sent through the network. The *XSniffer* can now be used to monitor the messages being sent from the sensor node. Remove the *XMeshBase* programmed Mote from the programming board and set aside before continuing. Install the *XSniffer* application onto a third Mote that you will plug into your programming board (base station). Install this application with a node id of 2 using the Programmer's Notepad. Then, keep the Mote you just programmed plugged into the programming board and start the *XSniffer* application by double clicking on the icon on your desktop. Click on the options tab and select the *XMesh* packet Type radio button. Go back to the Log tab, select the COM port that is connected to the programming board and then click on Start to begin "sniffing" the radio traffic. Eventually, you should see message packets displayed in *XSniffer*. Messages are sent about 1 second apart.

To view your sensor network with MoteView, double click on the desktop icon, remove the *XSniffer* Mote from the programming board and plug the *XMeshBase* Mote back into the programming board. From the *MoteView* main menu select **File>Connect>Connect to Database**. Select **mts310_results** and click Apply. From the MoteView main menu select **File>Connect>Connect to MIB510/MIB520/MIB600/Stargate**. Set the COM port value to the correct value for your setup. Select the **XMTS310** application from the *XMesh* Application drop down list. Select the **Advanced** tab. In **Data Logging Options** menu, check the box for **Spawn Separate Shell**. Click on Start. You will now see *XServe*

start-up enabled to log sensor data to the database. Now move back to the main *MoteView* window and you should see sensor data from node 1. All of Crosbow's sensor and data acquisition boards are supported with *XMesh* enabled applications.