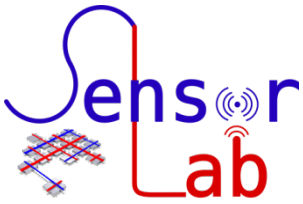
GEORG-AUGUST-UNIVERSITÄT
GÖTTINGEN



Georg-August-Universität Göttingen
Faculty of Mathematics and Computer Science
Institute of Computer Science
Telematics Group
Sensorlab

Lab 2 –Gathering Sensor Data

Chencheng Liang – University of Goettingen – 11603960

Introduction

In this lab we are going to get in touch with sensor mts420, and get some real environment data from the sensor, such as light, humidity, temperature and air pressure).

These data will be print on the screen, with the sensor node id. But before doing this we need to understand how to it works step by step. In the assignment2.1 and 2.2, we modify the Blink program to send a counter message according to its timer and LEDs. We use base station to listen this broadcast information and print on the screen.

After this is finished, we modify the code of mts400 to make the mts420 mote send back its node ID and other environment data. On another hand , we can control the LEDs on mts420 according to the light data, which means the sensor itself or base station can present some information come from the sensor.

When the base station listens to Blink program and mts400 program, it use different program, if the listen program is not matched, the message cannot be present properly.

Assignment 2.1

For finish this assignment, I adapt the code from mts400 to Blink, because in mts400, there are specific code of sending message.

First add a head file to define the message structure which is adapted from RadioCountToLeds. Include this head to both BlinkAppC and BlinkC.

In BlinkAppC, add one component(AMSenderC) and wire it with BlinkC (BlinkC.AMSend -> AMSenderC), and wire BlinkC.Packet -> AMSenderC;

In BlinkC, declare a message_t packet to load our message. declare a counter to count the Timers. The in event Timer0.fired(), add the counter. get the message structure in the payload, then use AMSend.send() to broadcast this message.

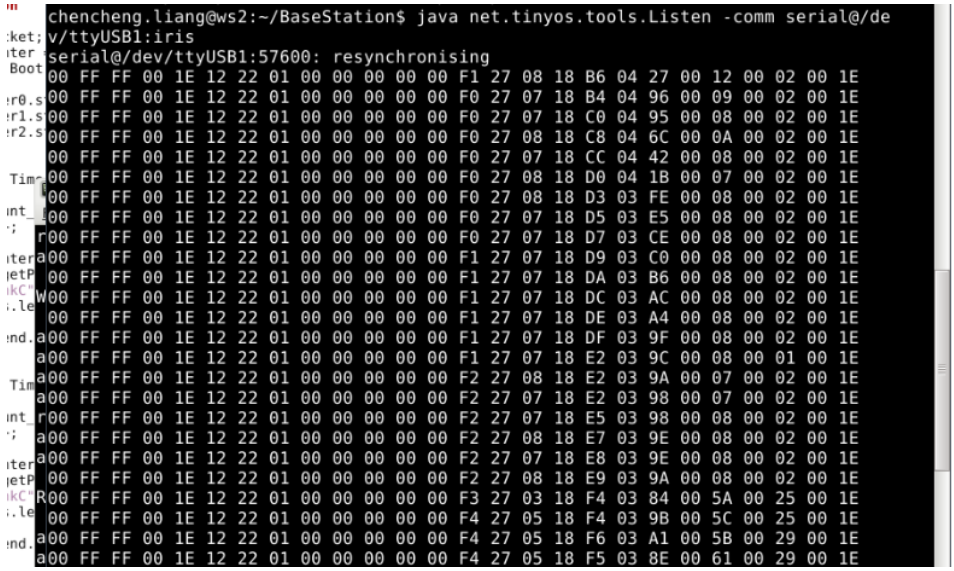
For each timer I did the same procedure.

After the code is finished. In stall this program to the mote with the command " make iris install.5 mib520,/dev/ttyUSB0". The remove the mote from programming board, then install a pair of battery and switch on the mote. It is blinking.

Assignment 2.2

In base station, use "make iris" compile the base station then use " make iris install.2 mib520,/dev/ttyUSB0" to install this base station. Under this directory use command " java net.tinyos.tools.Listen -comm serial@/dev/ttyUSB1:iris" to open the listening channel. Then we can get the broadcast message from the mote with BlinkC.

This is the result from the base station:



Assignment 2.3

In DataMsg.h add TOS_NODE_ID to send and receive the node ID.

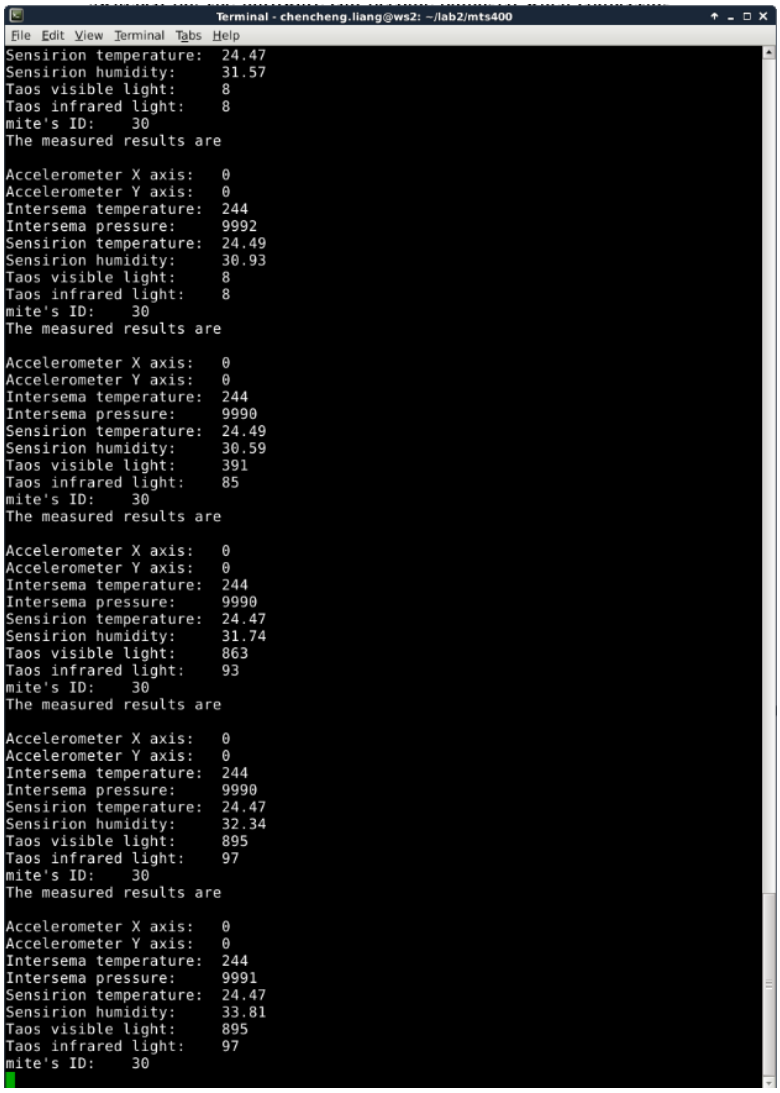
In Mts400Tester.java, add "out.println("mite's ID: "+results.get_TOS_NODE_ID())" under messageReceived() to print the node id.

In Mts400TesterP.nc under InfraredLight.readDone(),declare a variable for storing node id(uint16_t Sender_ID = call AMPacket.address()), then store it in the message structure(packet-> TOS_NODE_ID = Sender_ID) . Under VisibleLight.readDone() we can read the light and make LED change with it. Here if the light less than 200, switch on the LED3. When it larger than 200, switch off the LED3.

After the code is finished, we install this program to the programming board, then switch the programming board to MTS420 sensor, then give it a pair of battery, then switch on. And we can see that when I use hand to cover the sensor, which means the light is less than 200, then the LED3 switches on, and when I put this sensor to the sun light close to the window, then the LED3 switches off. We can verify this in the base station:

For the base station mote, first we I install a regular base station into the mote as assignment 2.2. The go to mts400 directory use command " javac Mts400Tester.java" to compile it and use command " java Mts400Tester -comm serial@/dev/ttyUSB1:iris" to open the base station to listen.

This is the result that I get:



Lessons Learned & Conclusion

In this lesson, I learned how to send message from a mote to the base station. First I learned to differentiate the devices including the programming USB module, programming mote(with battery), and the sensors(MTS420). I learned which program run on which device.

The idea of code is from RadioCountToLeds. When I send a message, the message should be get into the payload then we can send that message.
And be careful about the variable clamming, it can only be declared at the first of the function, it cannot be declared at where you want such as at the end of the function.

Also be careful about the command "install" and "reinstall". If you would like to rebuild your application, you can do so by using install instead of reinstall.
I leaned define a variable in the head file to transmit additional information in the message. The sensor send broadcast messages so we can receive other's message, so I need to specify my message or channel to distinguish the messages.

We have the same base station program, but when use different java code to listen, we can get different message from different devices. We cannot use one base station to listen different channel, for example, I cannot use one terminal to listen mts400, and another terminal to listen to the message from Blink program.

I saw many data such as coordinate and temperature sent from the mts420, so actually we can use mts420 to collect many data from the environment.

In program mts400 all the configurations are already finished, so we do not need to change it, we just use the defined interface , then the work can be done.

To send and receive specific message is a huge step for me and it makes me understand how these hardware been programmed and connected.

References