Assignment #3 – TinyOS programming

Due 17/01, at 17:00
Please put your assignment answer in Philippe Bonnets mailbox

Overall description
You are given 5 sensor nodes with a USB hub that allows you to power them. Two of the nodes are equipped with a sensor board. One of the nodes acts as a gateway to a PC. Using the USB hub to power the nodes forces them to be in communication range with each other. This is the sensor network you are to program for this assignment. Your task is to program the sensor network to accomplish a given task with the most efficient duty cycling.

Prerequisites
You must install TinyOS on a PC of your choice, running either Windows or Linux with an USB port. The PC must have access to the Internet, more specifically it must be able to make HTTP requests against www.distlab.dk on port 80. If you do not have access to a PC, you can use one in the projektlab.
We will provide you with the following:

- 5 13192EVB platform with 2 sensor boards.
- A boot loader host program for Linux and Windows on each node.
- A TinyOS 13192EVB tree that you must use instead or alongside your installed TinyOS tree. This code distribution contains the SimpleMac components that will allow you to operate the radio.
- For Windows: A set of drivers for the FTDI chip.
- For Linux: Source code to a simple terminal program that you can use instead of minicom

All software will be provided at http://www.distlab.dk/v04336/ – you must use username: student, and password: sNs42 to access the files. Please do not distribute these files to people outside your group! The files includes a README file that provides more information about the compilation process. Please consult this file as soon as possible.
Assignment

You should program the sensor network for the following task:

- Both nodes equipped with sensors sample data at 1 Hz.
- If light is off and temperature is greater than 22 degrees C then both measurements should be transmitted to the gateway with the shortest delay\(^1\) possible
- Otherwise both measurements should be sent to the gateway throughout the sensor network (in exactly 4 hops),

Your program should respect the two following constraints:

1. No two consecutive 4 hop sequences should be identical when data is transmitted throughout the sensor network
2. The maximum acceptable delay for the data to reach the gateway is 20 seconds

Your assignment is to implement this program with the lowest possible node duty cycling (we consider that a duty cycle of 10%, i.e., the nodes/radios are turned off 90% of the time, as pretty good).

Your report should include, your design (how do you plan to meet the delay requirements? What is your duty cycling strategy? How do they impact the design of your program), the design of your TinyOS programs, the testing (how do you make sure that your program respect the constraints? How do you measure duty cycling?)

We expect you to hand in a 1) report that includes at least a printed copy of your source code, component graph, and 2) the platforms programmed with your application. (Please put the nodes, usb hub, and cables in a plastic bag or similar, with a written note that has the group members names. To ease identification of the nodes, your program should start out by writing the names of your group members to the UART on startup.)

Hints

- You only have two weeks. Do not be overly ambitious with your design. We recommend an approach where you program first something that works, and then try and optimize it. Note that this approach requires a pretty sound design. Do not spend 2 weeks designing though. We will favour the groups that have a working implementation.
- Again, if you believe you have found a problem, do not hesitate to mail madsdyd@diku.dk about it. Include all the information you can think of, inclusive the source code to the application you are trying to compile, etc. The discovery of bugs/flaws and the submission of informative bug reports will add positively to the evaluation of your report. Only the first group that submits a bug will be credited, of course. Bugs will be fixed/reported on the homepage during the assignment period, and the software repository updated as needed.
- Discuss with other students on the newsgroup diku.sensor-networks.

\(^1\) Note that we define delay as the difference between the time measurements are obtained on the sensor node and the time those measurements are received on the gateway.
The network interface we expect you to use is called 
"SimpleMac", and it is documented in the interface file.

When you use the radio on the sensorboard, the UART has a raw transmission rate of 38400 bps, because it is able to take advantedge of the higher precision clock of the radio. We expect to be able to make sense of the information by attaching the sensor node to a standard terminal emulation program such as hyperterm (Windows), minicom or miniterm. Note, that you can use the UART for debugging purposes, and perhaps even give the nodes (debugging) commands through the UART interface. Also, note that the UART interface have changed slightly to be more compatible with the main trunk of TinyOS. The put command now only takes a single byte. The put2 command is identical to the put command you used in the last assignment.

Note that recent NesC distributions include editing modes for emacs and vi, as well as for the popular kde applications such as kate, kwrite and kdevelop.

The platforms you will be provided with, that do not have a sensor board mounted are programmed with a program that you may use to verify that your setup is working: Connect the board to an usb port, it should start flashing leds. Connect using a terminal program (38400 bps, 8N1), and you should get output that indicates wheter the nodes can reach the radio of eachother. The platforms that do have a sensor board mounted, contains a random program, using 300 bps (8N1) communication.