

EENG 460a Programming Assignment # 2

Professor Andreas Savvides
andreas.savvides@yale.edu

Teaching Assistant: Dimitrios Lymberopoulos
dimitrios.lymberopoulos@yale.edu

class url: <http://www.eng.yale.edu/enalab/courses/eeng460a/>

Due November 30, 2004

Introduction

This assignment builds on some of the basic embedded systems programming concepts introduced in PA#1 with the goal to familiarize you with a primitive sensor network. In this assignment you will use the SOS Operating System to write a basic wireless data collection application using the IEEE 802.15.4 protocol stack. This will familiarize you with the IEEE 802.15.4 packet types and packet format and it will expose you to the aspects involved with a complete system.

Application Scenario

Your application will collect a set of light and battery measurements from a small set of sensor nodes deployed inside the lab. A set of 3 pre-programmed sensor nodes will be placed at different places in the lab. These will be the data acquisition nodes and they will be programmed to respond to your requests (assuming that you use the correct packet format and protocol).

You will have to program a single sensor node to collect the value of the light sensor and the battery level of each one of the 3 pre-programmed nodes every 3 seconds for a total time of 2 minutes. In addition, your program should count all the transitions of the light sensor samples above or below a specified threshold. Your application should be running only for 2 minutes. In order to get the 3 seconds time interval you will have to use a software timer in the SOS operating system. Do NOT use a dedicated hardware timer to get the 3 seconds time interval.

The 3 data acquisition nodes will send you the required data only if you use the following protocol and packet formats.

Packet Format: All the packets sent and received must be IEEE 802.15.4 data packets and must be acknowledged. Short addressing modes must be used for both the source and the destination addresses. No encryption or data authentication should be used.

Communication Protocol: Your node will be able to communicate with the 3 data acquisition nodes by querying these nodes. Specifically, there are 3 different queries:

- QUERY_ALIVE (0x10): use this query to see if the data acquisition node is up and running. If the node is alive you should receive an ACK packet immediately after sending this query. The data acquisition nodes will not send any other information to you after receiving this query.
- QUERY_LIGHT (0x20): use this query to get the value of the light sensor on the data acquisition node. Upon the reception of this query, the data acquisition node will send you a packet with the value of its light sensor. You must acknowledge this packet.
- QUERY_VBAT (0x30): use this query to get the battery level on the data acquisition node. Upon the reception of this query, the data acquisition node will send you a packet containing its battery level. You must acknowledge this packet.

All the data sent and received by your node will have a 1 byte packet payload (without taking into account the ACK packets). In the case of the messages you send this byte will be the query (QUERY_ALIVE, QUERY_LIGHT or QUERY_VBAT) and in the case of the messages you receive this byte will be either the value of the light sensor or the battery level of the data acquisition node. Your program should do the following:

1. Wait for 3 seconds
2. Node 1: Send QUERY_ALIVE
3. Receive ACK
4. Node 1: Send QUERY_LIGHT
5. Receive ACK and a data packet
6. Node 1: Send QUERY_VBAT
7. Receive ACK and a data packet
8. Wait for 3 seconds
9. Node 2: Send QUERY_ALIVE
10. Receive ACK
11. Node 2: Send QUERY_LIGHT
12. Receive ACK and a data packet
13. Node 2: Send QUERY_VBAT
14. Receive ACK and a data packet
15. Wait for 3 seconds
16. Node 3: Send QUERY_ALIVE
17. Receive ACK
18. Node 3: Send QUERY_LIGHT
19. Receive ACK and a data packet
20. Node 3: Send QUERY_VBAT
21. Receive ACK and a data packet
22. Go to step 1 if the total running time of your application is less than 2 minutes.

If a data acquisition node does not acknowledge a QUERY_ALIVE packet then your program should not send the QUERY_LIGHT and QUERY_VBAT packets. If a data acquisition node receives a QUERY_LIGHT or a QUERY_VBAT packet without first receiving a QUERY_ALIVE packet then it will not send you the required data. Therefore, do not try to read data from a node before verifying that this node is alive.

For every QUERY_ALIVE packet that is acknowledged by a data acquisition node, your program should print the following line on the UART: “<node_id> IS ALIVE”, where <node_id> = 1, 2, or 3(read below). If a QUERY_ALIVE packet is not acknowledged then the following line should be printed on the UART: “<node_id> IS DEAD”, where <node_id> = 1, 2, or 3(read below). Every time that you receive a data packet containing the light sensor value or the battery level of a data acquisition node, your program should print the statement “<node_id: LIGHT=<value>” or “<node_id>VBAT=<value>” respectively on the UART. Each time that there is a transition (in the light sensor samples) above or below a specified threshold (defined in the next section) your program should also print the following statement on the UART: “<node_id>: Transition”. Your application after running for 2 minutes should print the total number of transitions for all the data acquisition nodes on the UART: “<node_id>: Total Number of Transitions: <transitions>”.

In addition, your application should keep track of the link quality indication of all the received data packets (ACK packets are not included). Specifically, when a data packet is received by your application the value of the link quality for this packet should be printed on the UART (“LQI=<value>”).

In order to print strings on the UART you will have to use the *ker_uart_send()* function of the SOS operating system. The baud rate of the serial port on the PC side should be set at: 57600bps.

Implementation Details

In order to communicate with the data acquisition nodes you need to know the following information:

- Data acquisition node 1 address: 0xDEAD
- Data acquisition node 2 address: 0x7231
- Data acquisition node 3 address: 0x35A3
- Network ID (PanId): 0x2420

You are free to choose the address of your node. Make sure that the address of your node is different from all the data acquisition nodes.

The threshold for a transition in the light sensor samples is the following HEX value: 0x70. Note that each data acquisition node sends you a byte containing the light sensor sample. If the previous byte received was higher (or lower) than 0x70 and the last byte received is lower (or higher) than 0x70 then a transition has occurred.

Compiling your code

The source code for this programming assignment will be posted on the following link:

<http://www.eng.yale.edu/enalab/XYZ/programming/sosPA2.tar>

All you have to do is to download the tar file and extract all the files included there in the directory you wish. The starting application for this assignment lies under the following directory: `<extract_directory>/apps/EENG460_PA2`. All the code that you have to modify lies under this directory. To compile your application type: `make xyz`.

Important notes

1. The grading for this assignment will be based on the quality and power efficiency of your design. Try to use as many of the XYZ low power features as you can to reduce the power consumption of your sensor node.
2. Before you start writing your own code, read the documentation of the Zigbee MAC protocol (<http://www.eng.yale.edu/enalab/XYZ/programming/ZIGBEE.pdf>) and try to understand the philosophy and the architecture of the SOS operating system.
3. Do NOT change the Zigbee MAC protocol. All you have to do in this assignment is to provide your own implementations of the Zigbee application interface functions (refer to the Zigbee documentation: <http://www.eng.yale.edu/enalab/XYZ/programming/ZIGBEE.pdf>).

Deliverables

The deliverables of this assignment are the following:

1. A demonstration of your implementation to the TA during his lab hours. During the demonstration you will be asked to explain in detail your implementation.
2. A single zip file with ALL your source code files (including the makefiles and a README file). The zip file should follow the following format: **lastname_PA2**, and should be sent to: dimitrios.lymberopoulos@yale.edu.

Getting help and answers to your questions

If you need help or you have questions about the assignment you can contact the teaching assistant of the course in the following ways:

1. Email: dimitrios.lymberopoulos@yale.edu
2. Stop by the C040 lab during the lab hours:
 - a. Tuesday: 4:00pm – 6:00pm
 - b. Wednesday: 2:00pm – 4:00pm