Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

1. February 2008

Nicolas Burri, Pascal von Rickenbach,
Roger Wattenhofer

Exam Ad Hoc and Sensor Networks HS 2007

- Please write your **Name and Legi-Number** on all sheets you hand in.
- You have **60 minutes**.
- There are **4 questions** with a total of **60 points**.
- Put your Legi on the table, we will check them.
- **No auxiliary material** is allowed.
- You may answer the questions in **German or English.**
- Where indicated stick to the expected answer length.

→ **Do not open or turn until told to do so by the supervisor!**

Name:	**Legi-Nr:**

Question	**max**	**achieved**
1	10	
2	15	
3	21	
4	14	
Total	**60**	

Grade:	

Question 1 (10 Points): Data Gathering

We want to build a wireless sensor network monitoring the occurrence of lightning (= "Blitze") by means of light sensors. Sensor nodes are thereby placed on roof tops of different buildings around the ETH main building. Readings are propagated to a base station using multi-hop communication. Not all nodes are equipped with a light sensor; some solely serve as communication relays. Furthermore, all devices are battery driven. The system should run for several months without human intervention. Message delay is not of primary importance but should be relatively small (no long term buffering).

a) (6 points) You are asked to design an energy efficient MAC protocol for this scenario. Identify the requirements of the application and discuss which of the approaches to optimize MAC layers presented in the lecture are most suitable. Describe your proposed solution in a few sentences.

b) (4 points) After a couple of months of successful operation the lightning detection system is extended to additionally measure the ozone concentration in the air. Thus, all nodes are enhanced with custom ozone-sensors which are sampled once every two minutes. Would you stick with your MAC protocol form a) and tweak it to better cope with the new situation? If so, what changes would you propose? Or would you rather choose another MAC layer approach? If so, what would your new system look like? Explain your decision.

Question 2 (15 Points): Time Synchronization

a) (3 points) In the lecture you have seen the *Reference-Broadcast Synchronization (RBS)* for clock synchronization. Briefly describe the basic idea of this mechanism.

b) (3 points) Can you think of a connected topology where RBS fails? Give a simple example.

c) (4 points) Consider the following scenario: A multi-hop network is used to detect events (e.g. people entering different rooms in a building). Whenever an event is detected at a node in the network a message is generated and immediately time-stamped with the node's current clock value. This message is then sent towards the base station. Assume all nodes in the network frequently execute RBS to determine their clock offset to each neighboring node. Is it possible to order all incoming events at the sink according to their actual occurrence in time? If yes: How? If no: Why?
(1-2 sentences)

d) (5 points) Consider a second scenario analogous to the one in c). All nodes execute RBS every minute. Detected events are immediately time-stamped with the local clock value. However, to save communication overhead, upon event detection a node stores the timestamp in local memory instead of directly sending a message. Once every 12 hours each node transmits all logged events and their timestamps in one go. Is the base station in this scenario able to order the events? Discuss in 2-3 sentences.

Question 3 (21 Points): Dominating Sets

a) (2 points) What is the definition of a *Minimum Dominating Set*?

b) (2 points) Flooding is an example where a dominating set may be helpful. Explain why.

c) (4 points) Given several algorithms computing *Dominating Sets*, you are asked to select the “best” of them. Which properties of the algorithms do you take into consideration when choosing?

d) (13 points) In the lecture the *Greedy Algorithm for Dominating Sets* was described. In each round, the algorithm greedily chooses a node that dominates the most neighbors which are so far neither dominated nor part of the dominating set themselves. The approximation ratio for this algorithm is claimed to be $\log(\Delta)$ if Δ is the maximum node degree of the graph. Give a proof of this claim.

Hints:

- The proof is similar to the one for the “Tree Growing” algorithm in the script.
- The Harmonic Function is defined as: $H(n) = 1 + 1/2 + 1/3 + \ldots + 1/n$
- $H(n) \approx \log(n) + 0.7$

Question 4 (14 Points): TinyOS

a) (6 points) The following code of a module and two interface definitions contains some errors. Mark the problems directly in the code and add a short comment indicating what is wrong (e.g. missing return value).

```
***************** File MyTestApp.nc ************************

  module MyTestApp{
    provides {
        interface StdControl as Control;
        interface Detection;
    }
  }
  implementation{

    uint8_t count;

    result_t StdControl.init(){
        count = 0;
        return SUCCESS;
    }

    command uint8_t countEvents(){
        return count;
    }

    event result_t Detection.somethingHasHappened(){
        count++;
        return SUCCESS;
    }
  }



***************** File Detection.nc ************************

  interface Detection {
    command uint8_t countEvents();
    event result_t somethingHasHappened();
  }



***************** File StdControl.nc ***********************

  interface StdControl {
    command result_t init();
    command result_t start();
    command result_t stop();
  }
```

b) (5 points) A simplified "Send" interface in TinyOS to transmit messages over the radio looks as follows:

```
interface Send {
   command result_t send(TOSMsgPtr msg);
   event TOSMsgPtr sendDone(TOSMsgPtr msg, result_t result);
}
```

The following code snippet of an application tries to send out all numbers from 1 to 200.

```
uint8_t i;
TOSMsg msg;
TOSMsgPtr mPtr = &msg;
.
.
.
initializeMessage(mPtr); // a black box function
                         // initializing all necessary
                         // fields of the message
.
.
.
for (i = 1; i<=200; i++){
    mPtr->data = i;
    call Send.send(mPtr);
}
```

What happens if this application is executed? Do you see (potential) problems? (2-3 sentences)

c) (3 points) TinyOS is a single threaded operating system. Nevertheless, with the "atomic" keyword the system provides a mechanism to guarantee the consecutive execution of multiple commands. Why is this necessary? (1-2 sentences)