A Centrally Monitored, Cost Effective Wireless Sensor Platform

Tim Åberg
Sebastian Själín

Master of Science in Engineering Technology
Electrical Engineering

Luleå University of Technology
Department of Computer Science, Electrical and Space Engineering
Todays hospitals have a lot of samples; tissue, saliva and blood to mention a few. These samples are stored in refrigerators with little or no monitoring; temperatures being written down by hand twice a day to get a backlog are not an uncommon event. This report deals with that problem by providing a central monitored cost effective wireless sensor platform using ZigBee, which is easy to maintain and expand. The thesis was made for the clinical physiology lab at Sunderbyn hospital.

The internal supervisor was Professor Per Lindgren at EISLAB and the external supervisor was Joackim Barkman at Länsteknik.
The project mostly went smooth, except for some problems with deliveries and some hardware design errors. We would like to send a big thank to all the people we got in touch with at Sunderbyns Hospital, without you this thesis would not have been possible.

Tim Åberg and Sebastian Själin
# CONTENTS

## Chapter 1 – Introduction
1.1 Actors in the project ........................................... 1
1.2 Introduction .................................................... 1

## Chapter 2 – Evaluation of Problem
2.1 Specifying the needs of the system .......................... 3
2.2 User requirements .............................................. 3
2.3 Technician/service personnel requirements ................. 4

## Chapter 3 – Evaluation of Methods
3.1 Different solutions to the smaller problems ................. 5
3.1.1 Temperature measurement .................................. 5
3.1.2 Wireless transfer ............................................ 6
3.1.3 Data storage .................................................. 7
3.1.4 User interface on node ...................................... 8

## Chapter 4 – Selected Solution
4.1 Chosen solutions ............................................... 11
4.1.1 Temperature measurement ................................. 11
4.1.2 Wireless transfer ............................................ 11
4.1.3 Microcontroller (MCU) ..................................... 13
4.1.4 Programming languages ................................... 13
4.1.5 Data storage .................................................. 14
4.1.6 Display on node ............................................. 14
4.1.7 Data presentation ............................................ 14

## Chapter 5 – Implementation
5.1 Software .......................................................... 15
5.1.1 Server ........................................................ 15
5.1.2 Sensor node .................................................. 16
5.1.3 USB gateway ............................................... 18
5.2 Hardware .......................................................... 18
5.2.1 Microcontroller units ....................................... 18
5.2.2 Temperature sensor ........................................ 19
CHAPTER 1

Introduction

1.1 Actors in the project

Sunderbyn hospital is the county hospital in Norrbotten; it is also the local hospital for Luleå and Boden. During one year the hospital has about 130 000 visits and treats 21 000 inpatients.

Länsteknik is responsible for Norrbotten county councils (NLL) technical infrastructure, they also maintain and develop functional and patient safe IT- MT\textsuperscript{1}-systems and equipment.

The clinical physiology Lab at Sunderbyn hospital is responsible for all the medical samples that are collected from patients. They are also responsible for lab storage and testing the samples, giving back the result to a doctor.

Luleå University of Technology (LTU) is the northern most university in Scandinavia, it yearly conducts research within the technical faculty as well as the humanist and social sciences faculty. The annual turnover for the research is 700 million SEK and covers 70 research subjects divided into 6 prioritized research areas and 6 development areas.

1.2 Introduction

This masters thesis is made on behalf of the clinical physiology Lab as a collaboration between Länsteknik and LTU, and will deal with the problem of monitoring and backlogging temperature dependent environments e.g. refrigerators. Backlogging is very important in a hospital environment, and the demand that everything needs to be traceable is rising so new solutions are a hot topic. The logging today at Sunderbyn hospital is done by hand. The most advanced monitoring system in the county is located in Kalix, in chapter 6.1 our system will be compared to that system.

To do the thesis we were given a budget of 20 000 SEK to develop a new platform and

\textsuperscript{1}Medical Technology
come up with a working prototype.
CHAPTER 2

Evaluation of problem

2.1 Specifying the needs of the system

The main idea the clinical physiology lab had for their new system was that it should consist of some kind of nodes/units mounted on every refrigerator. On these units there would be a display for showing the current temperature. It should also communicate with a server that stores all values and could present a graphical representation of the recorded temperatures. The server should also have some kind of warning system for when a node does not report a new temperature in time or when the temperature is outside the given span, i.e. too hot or too cold. The final goal was to make both the core system and the ability to add new nodes as cheap as possible.

From this outline of the system we had to make a list of requirements. The requirements were divided into two separate parts:

2.2 User requirements

To compile this list the staff at the laboratory at Sunderby hospital were consulted and they stressed the following points:

- Cheap, both installation and migration from current system
- Units should be as cheap as possible
- Units mounted on the refrigerators, and driven by battery i.e. no extra cords to the refrigerator
- Capable of calibration
- One central monitoring unit/server
• Ability to store temperatures and show them over time

• Alarm of some kind that gets triggered once a node reports a value outside of a given temperature range

• Determine if some node is malfunctioning

2.3 Technician/service personnel requirements

The other part that will be involved with the equipment is the technicians; they will take care of the maintenance and calibration but also following up on the accuracy of the system. The points that they stressed were:

• Pass an electrical safety test according to [1], [2] and [3]

• Satisfy the rules in [4] and [5]

• An error less than $\pm 2 \, ^\circ C$

• Battery lifetime should be calculated to know when to make battery changes on the nodes

• Easy battery change

• Uses standard batteries

• Solder less battery replacement

• Easy calibration

These two lists form the basis for all our subsequent decisions. We also strived to make the system as flexible as possible, so different sensors could be connected in the future.
CHAPTER 3

Evaluation of methods

3.1 Different solutions to the smaller problems

3.1.1 Temperature measurement

Choosing the right temperature sensor was very important, the whole system could be excellent, but if the temperature data is off it would all have been for nothing. Today there are many components that use different ways of measuring temperature; the following are the ones we took into consideration:

**Analog temperature sensors**

One of the most common temperature sensor types. They change their resistance as temperature changes, and knowing this relation one can calculate a temperature from the resistance. The drawback of analog temperature sensors is that everything that changes the resistance will affect the reading. For example: resistance in wires, bad soldering or glitching connectors.

Pros:
- Low cost

Cons:
- Inadequate accuracy
- Must be calibrated and translated to yield valid temperature readings

**Digital temperature sensors**

The temperature is measured by an analog sensor as above, but is processed on chip and the data is sent digitally. This is a more accurate way of reading the temperature, as cable resistance can be ignored since the signal is digital.
Pros:

- Better accuracy
- Calibration is done as part of the chip manufacturing process

Cons:

- More expensive

### 3.1.2 Wireless transfer

The choice of a wireless standard was a central part of the development process; it would set different demands and restrictions on the hardware and functionality. Three different standards were evaluated and from those we picked the one that best satisfied the demands in Chapter 2. The standards were Wi-Fi, a more lightweight standard for short range wireless communication, and some kind of mobile solution via the mobile phone network (GPRS/3G).

The lightweight solution that was chosen for the comparison was ZigBee, because the technology has been used in similar situations.

The only solution that connects directly to NLLnet (the internal network of NLL) and does not need any other gateway is Wi-Fi; ZigBee need some kind of a gateway to forward data to the network and the mobile solution also needs to interface with systems on NLLnet in some way.

**Wi-Fi (IEEE 802.11)**

Wi-Fi or IEEE 802.11 is one of the most persuasive standards when it comes to wireless communication. NLL has Wi-Fi or has plans to build Wi-Fi in all of their buildings so all the infrastructure needed is already in place. Unfortunately Wi-Fi-connections are strictly regulated by the administrators of NLLnet.

Pros:

- Available through most of the NLL facilities
- A widely used standard
- Does not need any gateway to traverse the firewall

Cons:

- Higher power consumption
- Larger data overhead
- Strict rules for NLLnet
3.1. Different solutions to the smaller problems

**ZigBee**

ZigBee is a higher layer of the IEEE 802.15.4-2003 physical layer protocol. It is designed to be both power efficient and lightweight. It is developed and maintained by the ZigBee Alliance\(^1\).

Pros:

- Power efficient
- Low data overhead
- Can sleep between transfers

Cons:

- Must use a gateway to traverse the firewall and relay information to the server
- Coverage is limited to the range of the gateways

**GPRS/3G**

If any kind of solution that sends traffic over the mobile network is going to be used, it has to be certain that the security of NLLnet is no compromised in any way.

Pros:

- Only need mobile network coverage
- Can mobile e.g. used in ambulances

Cons:

- Needs a mobile phone carrier subscription
- Must use a gateway to traverse the firewall and relay information to the server

**3.1.3 Data storage**

The data needs to be available for auditing purposes, hence a way of storing the data is needed.

\(^1\)http://www.zigbee.org/
A custom solution

This solution could have been made successful or unsuccessful depending on the time spent on it, so it is hard to predict the outcome of this option.

Pros:

- No third party application needed
- Full control over the system

Cons:

- Hard to predict the success of the system

MySQL

MySQL is a very popular relational database management system (RDBMS) available under the GNU General Public License. MySQL is running as a server and handles multiple connections from users as well as handling different databases.

Pros:

- Easily integrated with Sunderbys existing MySQL server
- Fast and reliable access, when server is up and running

Cons:

- Relies on a third party, if the server is down the logging will not work

3.1.4 User interface on node

A display on the node is vital for showing the user the current temperature of the refrigerator. It could also be needed for configuring and calibrating the node.

Alphanumeric display

Alphanumeric displays often have a built in display controller that controls a dot matrix LCD. The information to be displayed is sent from the microcontroller (MCU) to the built in controller of the display.

Pros:

- Dot matrix displays allow nice printouts

Cons:

- Occupies a data bus on the MCU
7-Segment display

As the name suggest, 7 segment displays use 7 separate segments to display numbers from 0–9. This display type does not usually have a display controller of its own, so the MCU has to do everything.

Pros:

• Does not occupy a serial bus
• Low power consumption
• Easy interfacing

Cons:

• The MCU must have some kind of display-controller
• Occupies many more pins on the MCU

No display at all

Despite the options above, the display could be completely left out, the temperature would then need to be read from the computer displaying all the information from the refrigerators.

Pros:

• No power consumption
• Would result in a smaller and less expensive unit

Cons:

• Hard to tell the temperature away from the computer
• Hard to set up a node or calibrate it
CHAPTER 4

Selected solution

4.1 Chosen solutions

4.1.1 Temperature measurement

The chosen temperature sensor was a digital one, because it is vital that the temperature is correctly measured. The analog sensor where sloped even considered the low price because they were not good enough. Since the digital temperature sensor is designed to have a linear behavior, the only calibration needed is the temperature offset.

4.1.2 Wireless transfer

ZigBee was chosen since both Wi-Fi and GPRS/3G could possibly compromise the security of NLLnet, and demand more work from the IT department checking and verifying our solution. It fitted our needs and was an interesting technology to work with. In the long run the best way of dealing with the problem might be a Wi-Fi solution but due to all the regulations it was estimated that the development of such a system would take more time than we had available, and did not have the mandate to involve the IT department of Länsteknik in such a degree.

Due to little experience in working with ZigBee, a development platform was bought for verifying that the specifications fulfilled the project needs. The development equipment bought was the Atmel Raven kit described in [6] and [7] and the Atmel STK600. Using this equipment, we could verify that ZigBee fit our needs.

In experiments at Porsögård 26, Luleå, the range of the Raven kit was measured at around 10 meters, a value deemed sufficient for the task at hand. The measurements were taken with the washing machines and kitchen as obstacles between the sender and receiver to emulate lab equipment with big metal casings. Using a set-up with lower frequencies

\[1\text{Atmel Software Development Kit 600,}\]
\[\text{http://www.atmel.com/dyn/products/tools_card.asp?tool_id=4254}\]
could improve the range, but there was no time to do testing regarding interference with any of the other lab equipment. The chip on the Raven is using a frequency of 2.4 GHz, which is the same as Wi-Fi, so all the equipment that could function near a Wi-Fi network would function near the system.

The chip chosen for transmitting on the Atmel ATZB-24-A2 described in [8]. This chip has an integrated antenna and is using the 2.4 GHz band.

Since there was limited time and money to develop custom PCBs it was decided that the ZigBee to NLLnet gateway needed for the ZigBee solution should be the same USB-dongle that was included in the Raven kit. It had sufficient functionality needed and had already proven its capabilities.
4.1.3 Microcontroller (MCU)

The last major hardware decision for the sensor node was the architecture and model to use for the MCU. The test platform was an ATRaven with the ATMega3290P MCU. The ATMega3290P has support for quite big LCDs, many GPIOs, and a lot of other nice features like ADC and DAC. Since it worked well for the test platform and fit the needs of the project, it was the MCU of choice.

4.1.4 Programming languages

The choice of programming languages was C for the embedded devices and Python for all Windows applications.

C was chosen because of prior experience in embedded programming with it, and it is one of the most commonly used languages for embedded programming. It also has excellent support for targeting Atmel AVR5, the architectures we are using.

---

2 General Purpose Input/Output
3 Analog to Digital Converter
4 Digital to Analog Converter
Python was chosen for its user-friendliness, in contrast with C, where you have full control over memory etc. In Python you exchange a big portion of that control for convenient syntax and high level primitives. We thought it would be nice to get more experience in both ends of the spectrum.

4.1.5 Data storage

We choose MySQL for data storage mainly because Sunderbyn uses it. Should the server go down, it would be brought back up with very little delay.

4.1.6 Display on node

A 7-segment display was chosen because it is easy to work with and to get a working prototype with little effort, it does not require any timings or data transfers.

4.1.7 Data presentation

For the data presentation many different techniques were used to make it as easy as possible for the user to understand. A graph was used for displaying temperature over time, with every node showing as a separate curve. To avoid a cluttered graph the user can choose which nodes to show; this is easily done by just selecting the node in a list. To highlight nodes that are too warm, too cold, or too unresponsive, three boxes representing these are shown. The graphical user interface (GUI) can be seen in Figure 5.2.
CHAPTER 5

Implementation

5.1 Software

The software comprises four main parts; the server that presents the data and handles the database connection. Application side of the gateway receives data from the USB port and forwards it to the server via NLLnet. USB side of the gateway receives the radio signals from the sensor node and transmits them via USB to the other part of the gateway. Lastly there is the sensor node, which gets the data from the temperature sensor and transmits it via ZigBee to the gateway. A model of the software can be seen in Figure 5.1. It shows the complete data path from the refrigerator node to both the GUI and the database. Data can also be read from the database to access previous readings for a monitored unit, e.g. a refrigerator.

5.1.1 Server

The server was programmed using python. The goal was to make the server as platform independent as possible, so all the libraries used are platform agnostic. We used PyQt 4\(^1\) as a base for the GUI.

Every sensor node has a unique id (1 – 4095), which identifies them when sending data. The server keeps track of the type of unit the node monitors, e.g. refrigerator, freezers, low temp storage etc. based on their id and monitors that node according to the settings of that node. The message can also be an error code from some part of the system which the server will process and present to the user in a way they can comprehend.

To present the temperature data to the user, Matplotlib\(^2\) was used, it creates nice graphs in a MATLAB\(^\text{TM}\)-like fashion. The communication between the server and the MySQL server was supported by PyMySQL\(^3\).

\(^{1}\)PyQt is a Python port of Nokias Qt. Information about Qt can be found at http://qt.nokia.com/

\(^{2}\)http://matplotlib.sourceforge.net/

\(^{3}\)http://pymysql.sourceforge.net/
5.1.2 Sensor node

The heart of the sensor node is the ATMeaga3290P, it controls the display and handles the temperature sensor. By default the sensor node shows the previously measured temperature. It also have a very simple user interface, the user can only do the following:

- Show the nodes id for 10 seconds.
- Increment the id by one
- Decrement the id by one
- Show the node temperature calibration offset for 10 seconds. More on this in the Section 5.2.2
- Increase the calibration offset by 0.1 °C
- Decrease the calibration offset by 0.1 °C

The interface is only accessible by removing the back cover of the sensor node, this is because the id should only be changed or set at installation, and calibration will only be done during the accuracy test, once per year.

It will send its id and current temperature at a preset interval to the ZigBee chip via the USART\(^4\) interface. The ZigBee forwards it towards the server.

\(^4\)Universal Serial Asynchronous Receiver/Transmitter
5.1. Software

Figure 5.2: Server GUI

Scheduling on the ATmega3290P was achieved by porting TinyTimber\(^5\) to the device, and the TinyTimber version used was the same as the one used for the Baldos 2009\(^6\) project.

Apart from porting it to the MCU some code and files had to be rearranged, since AVRstudio\(^7\) does not support relative paths to subdirectories, and the development was done using different computers. This solution was easier than redefining all the paths every time the code was compiled on a different computer from last time.

The code in the ZigBee module is a slimmed, and modified version of the Atmel Radio Evaluation Software (RES) described in [9]. It is stripped to the point where only basic sending and receiving works, since development of a custom ZigBee stack was deemed too time consuming. The RES is built in such way that only its back-end has to be changed if you change the ZigBee chip. This was useful when migrating from the ATRaven which uses an ATmega1284 to the PCB designed for the project, since the ATZB-24-A2 has an integrated ATmega1281.

The RES has built in error handling of package loss and to that we added the safeguard of sending information more frequently than needed. If the server needs an information

---

\(^5\) TinyTimer is a small real-time kernel for embedded systems. It is based on the same theories as Timber - http://www.timber-lang.org/

\(^6\) http://09.baldos.se/

\(^7\) The development environment provided by Atmel, http://www.atmel.com/dyn/products/tools_card.asp?tool_id=2725
package once every 15 minutes to know that a sensor node is up and running, we sent one every five minutes instead. This is done since we do not have full control of the inner workings of the RES, and to get some error handling if RES is changed to something without such features.

### 5.1.3 USB gateway

The gateway consists of two parts. The first is the program on the USB dongle. This also, as in the sensor node, uses a slimmed and modified version of the Atmel RES. It simulates a serial port on the computer it is attached to, and forward all the ZigBee messages there to be processed by the other part of the gateway.

The second part of the gateway is the serial to NLLnet bridge, it consists of a driver for the simulated serial port and the actual gateway. The gateway verifies that the message follows the message standard with id and information in the right place. It then repackages the information and sends it to the server.

<table>
<thead>
<tr>
<th>Settings for the simulated serial port.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Baudrate 38400</td>
</tr>
<tr>
<td>Databits 8</td>
</tr>
<tr>
<td>Parity None</td>
</tr>
<tr>
<td>Stop bits 1</td>
</tr>
<tr>
<td>Flow control None</td>
</tr>
</tbody>
</table>

### 5.2 Hardware

The USB transceiver was purchased from Atmel. There was no casing available for this component and an open circuit board was not an option, so a casing was made. The sensor node was designed from scratch and consists of the components described in the following subsections. The communication between the hardware components in illustrated in Figure 5.1.

### 5.2.1 Microcontroller units

**ATMega3290P**

The ATMega3290P is a low-power 8-bit CMOS microcontroller, based on an enhanced RISC architecture. This MCU is used for most tasks on the node, except for the data communication via ZigBee. Some information of the ATMega3290P can be found below.

---

8RS-232

9There is one for both IntelX86- and AMD64 standard
5.2. Hardware

<table>
<thead>
<tr>
<th>Hardware</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Flash</td>
<td>32 KB</td>
</tr>
<tr>
<td>EEPROM</td>
<td>1 KB</td>
</tr>
<tr>
<td>RAM</td>
<td>2KB</td>
</tr>
<tr>
<td>Freq.</td>
<td>up to 20 MHz</td>
</tr>
<tr>
<td>GPIO</td>
<td>69</td>
</tr>
<tr>
<td>Serial USART</td>
<td>1</td>
</tr>
<tr>
<td>LCD Segments</td>
<td>4 * 40</td>
</tr>
<tr>
<td>Packages</td>
<td>TQFP100</td>
</tr>
</tbody>
</table>

**ATMega1281**

This MCU is integrated in the Atmel ATZB-24-A2 module. The ATZB-24-A2 consists of a AT86RF230 radio transceiver and the Atmel ATMega1281 MCU. Together they form a good base for sending wireless data over ZigBee. The main properties of the ATZB-24-A2 and the ATMega1281 are described in the tables below.

<table>
<thead>
<tr>
<th>ATZB-24-A2</th>
<th>ATMega1281</th>
</tr>
</thead>
<tbody>
<tr>
<td>MCU</td>
<td>ATMega1281</td>
</tr>
<tr>
<td>Transceiver</td>
<td>AT86RF230</td>
</tr>
<tr>
<td>GPIO</td>
<td>9</td>
</tr>
<tr>
<td>Serial USART</td>
<td>1</td>
</tr>
<tr>
<td>Band</td>
<td>2.4 GHz</td>
</tr>
<tr>
<td>Size</td>
<td>24 * 13.5 * 2 mm</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>ATMega1281</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Flash</td>
<td>128 KB</td>
</tr>
<tr>
<td>EEPROM</td>
<td>4 KB</td>
</tr>
<tr>
<td>RAM</td>
<td>8 KB</td>
</tr>
<tr>
<td>Freq.</td>
<td>up to 16 MHz</td>
</tr>
<tr>
<td>GPIO</td>
<td>54</td>
</tr>
<tr>
<td>Serial USART</td>
<td>1</td>
</tr>
</tbody>
</table>

5.2.2 Temperature sensor

The SMT16030 digital temperature sensor described in [10] was used. This sensor comes with casing and cable, and has a static absolute accuracy of ±0.7°C (-30°C to +100°C) and a resolution better than 0.005 °C. The output from this sensor is a duty cycle (D.C.), by using the equation (5.1) and (5.2), equation (5.3) was derived and the temperature \( t \) could be calculated.

\[
D.C. = 0.320 + 0.00470 \times t \quad (5.1)
\]
Figure 5.3: ATZB-24-A2 chip

\[ D.C. = \frac{\text{pulse width}}{\text{period}} \]  \hspace{1cm} (5.2)

\[ t = \frac{D.C. - 0.320}{0.00470} \]  \hspace{1cm} (5.3)

Figure 5.4: Temperature sensor
5.3 Division of responsibilities

5.2.3 Display

The display DE 130-RS-20/7,5 described in [11] is a five digit module. It is driven and controlled by the ATMega3290P.

5.2.4 Circuit board

The circuit board was designed with the above components in mind. The majority of unused pins on the MCUs\textsuperscript{10} are connected to pins on the board for future additional sensors. Circuit board, schematic and layout can be seen in appendix B.

![Figure 5.5: PCB (LCD side) unpopulated](image)

5.3 Division of responsibilities

Since both of us wanted to learn and develop both in Python and C the programs were broken down to smaller parts that were continuously integrated. This is only valid for software; the hardware was a cooperative effort.

\textsuperscript{10}Both from the ATMega3290P and the ATMega1281
5.3.1 Server

For the server, the responsibilities were divided as follows.

Sebastian:

- Network and communication
- GUI
5.3. **Division of responsibilities**

Figure 5.8: PCB (component side) populated

- Configuration management

Tim:
- Data management (MySql)
- Data graph

5.3.2 **Sensor node**

For the sensor node the responsibilities were divided in the following way.

Sebastian:
- Data gateway
- Tuning and reworking of Atmels RES
- Communication from the ATMega3290P to the ATBZ-24-A2
- Porting and restructuring TinyTimber

Tim:
- Temperature sensor
- LCD
- Buttons
Chapter 6

Comparison with another system

6.1 Comparison with existing systems under Norrbottens Läns Landsting

There are two different ways to monitor and log the temperature in the hospital environment in Norrbotten. The first way is to manually check the temperature every day and write it down. This is the analogue method they want to switch from. The other way is a digital solution used only in Kalix hospital. Since the analog way is by default inferior, they have said that all digital solutions that handle the logging by itself is better, hence a comparison has not been made with this system.

6.1.1 Comparison with the system in Kalix

The system in Kalix is monitoring 14 refrigerators all in close proximity of the server/graph computer. This system was developed by the technicians at Kalix hospital in 2009. Their system is not connected to NLLnet and is therefore not forced to follow [5].

All the sensors was connected directly to and driven by the server so connecting sensors in other parts of the building would be difficult and impossible at other locations using their current setup. This could be solved by using some serial-port-to-network-to-serial-port solution, but as there are no plans to make the system monitor any refrigerators other than those in adjacent laboratories this solution has not been studied by them or included in this comparison.

When a node reports temperatures outside their present range, an SMS will be sent to a predefined telephone number over a GPRSmodem\(^1\) connected to it. The server is not connected to NLLnet and is therefore is forced to store all data locally.

\(^1\)This is strictly in violation of [5] if the system was connected to NLLnet.
The data is processed and stored using the freeware program LogTemp\textsuperscript{2}.

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{logtemp.png}
\caption{LogTemp}
\end{figure}

\subsection{Comparison with our system}

Pros for our system:
\begin{itemize}
\item Can monitor temperatures all over NLL
\item Stores data in a separate database, which is operated and backed up by the IT department
\end{itemize}

\textsuperscript{2}http://mrsoft.fi/ohj01en.htm
6.1. Comparison with existing systems under Norrbottens Läns Landsting

- Wireless
- Easy to install new sensors
- Can print on any printer connected to NLLnet
- Temperatures visible on the refrigerator node display
- Can detect malfunctioning sensors

Pros for their system:

- Cheaper
- More functionality in the server program
- Notifications by SMS
- No batteries that need to be changed
- Less complex
This system is the only system that fulfills requirements given by NLL that we could find in the county. It offers great opportunities for improvements and for monitor different environment variables, since it has unused pads for connecting new sensors. There is also much unused computational power in the sensor nodes that could be exploited, and if the MCU is clocked to its maximum of 20MHz instead of the 8MHz it is running at now it would be able to compute even more, at a cost of reducing battery life.

If one were to continue working on this project, the first thing we would recommend is to order new circuit boards, since the ones produced have some major flaws, flaws that could be fixed by patching them but that is not feasible for large scale deployment.

The server GUI is satisfactory in the sense that it is minimalistic. It is hard to do anything that would compromise the view and the staff at Sunderby hospital thought it was intuitive.

The technical staff was also satisfied with the solutions and the technical documentation.
8.1 List of possible changes/improvements

Although the system works fine there is always room for improvement. This chapter lists possible improvements to the system.

**Smaller gateway**

Make a custom gateway for transferring data directly to NLLnet without a computer in-between, a whole ZigBee to Ethernet gateway on one PCB.

**Smaller sensor node**

Redesign the node for smaller, lighter, and better looking casings.

**Better display**

The display was chosen for its low energy consumption and the ease of interfacing, but in order to display more information simultaneously, it need to be bigger and have higher resolution.

**Reduce power consumptions**

The device was designed with low power consumption in mind, but there are still ways to lower it, by adding a power source, eg. a solar panel, using the temperature difference between the inside and the outside of the refrigerator etc.

**Custom ZigBee stack/protocols**

Make a more lightweight ZigBee solution that only has the functionality that the system utilizes. This could save power and would give full control over the system on all levels.
Use Wi-Fi instead of ZigBee

Wi-Fi could be used instead of ZigBee. This would include involving the IT department in getting through the rules and regulation that concerns their wireless network. This would make the use of a gateway redundant, but would drain significantly more power.

Other frequencies for ZigBee

If using a ZigBee chip that operates at a lower frequency the range might be increased and therefore reduce the need for so many gateways. The problem with this is guaranteeing that it does not interfere with any other equipment in the vicinity of the sensor nodes.
APPENDIX A

Schematic
Figure A.1: The sensor node schematic
Figure B.1: PCB (both sides)
Figure B.2: PCB (component side)

Figure B.3: PCB (LCD side)


