Wired sensor network for measuring fat deposition in extraction channels

A Power Supply and Communication Module

S. Garst, N. van Lith

4484320, 4305051

Bachelor Graduation Project

TU Delft
Delft University of Technology

Wired sensor network for measuring fat deposition in extraction channels

A Power Supply and Communication Module

Bachelor Graduation Project

S. Garst, N. van Lith

4484320, 4305051

June 18, 2018

Faculty of Electrical Engineering, Mathematics and Computer Science (EEMCS)
Delft University of Technology

Delft
University of
Technology

Abstract

The goal of this project is to identify the most promising measurement method and strategy for measuring the thickness of grease and to design a sensor module with electronic read-out and communication module. This project has been split into two parts: a sensor implementation and a communication system. This report describes the design and implementation of the communication module and power supply. The communication module has been designed to work conform a master-slave principle using MODBUS over an RS-485 databus. The used hardware for the communication module is rated to work within a desired 1 to 100m range. Furthermore the master node is connected to the internet to present the data on a small webserver. The power supply design depends on the implementation of the sensor, and contains the use of linear voltage regulators and inverters. The results show that the communication module works properly, whereas the power supply requires some further work.

Preface

The project that is being described in this report is part of the final project for the Bachelor Electrical Engineering of the Delft University of Technology. The project has been carried out over a period of eight weeks. The project has been carried out by a group of six students that are responsible for the design and production of a proof of concept. The goal of this project is to identify the most promising measurement method and strategy for real time monitoring of fat deposition in extraction channels. To test this method a sensor module with electronic read-out and communication module has to be designed.

The first weeks of the project have been used to work on literature research regarding the problem, the problem definition has been defined and the program of requirements has been set up. After the research possible options for measuring methods have been tested to see which were worth investigating in more depth. As a result of these investigations it was possible to decide which methods were worth further research and implementation. It was decided that there were two different measurement implementations that showed promise and that these would be investigated further. At this point in time, after the first three weeks, the project group was split in three different sub groups. A group that would look after an acoustic implementation of the sensor, a group that would investigate a capacitive implementation of the sensor and a group that would be responsible for the communication module and system design of the project.

We would like to thank our supervisors Andre Bossche and Jeroen Bastemeijer for the guidance, advice and support throughout the project. Also we would like to thank Michael and Dave from VETkanaal for their contributions to the project and the submission of the Bachelor Graduation proposal. Finally we would like to thank the Delft University of Technology for the available resources put at our disposal for working on this project.

Contents

Introduction

Preparing food is an essential part of our life. When cooking we usually use heat to prepare our food. To prevent the food from burning, most of the time a form of fat or grease is used. This fat or grease evaporates due to the heat and flows away when cooking outside. But when cooking inside this does not happen. One might want to use an extraction channel to suck away the evaporated grease and smells. Within the extraction channel the gases will flow towards the open air, but while flowing through the duct the gasses are able to cool down and to precipitate against the insides of the channel. This layer of grease itself is not a problem, however it is in case of a fire. The grease layer can be ignited and thus cause a fire, or even worse, fuel an already existing fire by allowing it to spread through the ducts, making the grease layer a potential safety threat.

When cooking at home this layer is not that much of a problem due to the slow precipitation of grease, short extraction channels, small amounts of used fats and the relatively few uses of the kitchen during the lifespan of a house. But when looking at the catering industry for example the used channels are much longer, the kitchen is used much more frequently and used for longer periods of time. The build-up of the grease layer then becomes something that has to be monitored and cleaned if necessary to prevent the grease from becoming a safety issue.

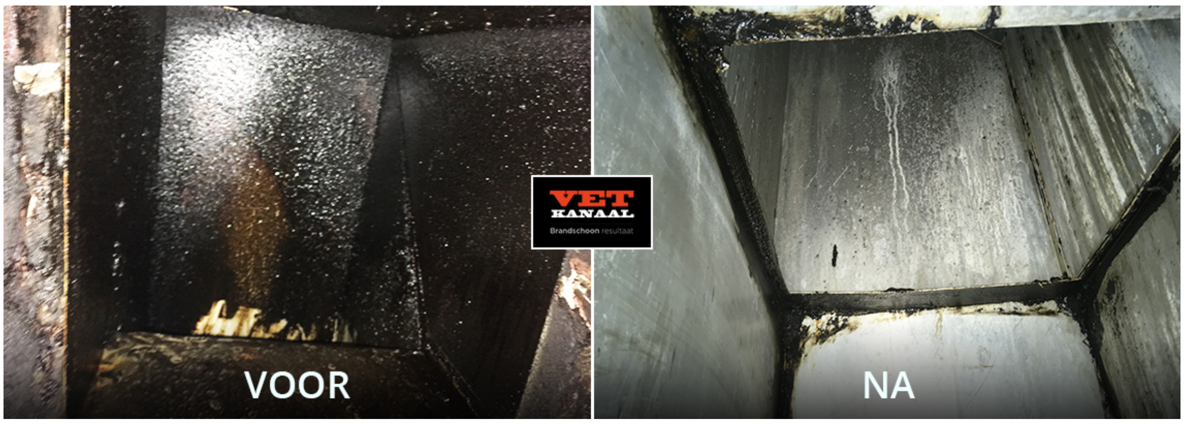


Figure 1-1: A comparison of an extraction channel before and after it is cleaned [1]

The thickness of the layer can be measured by the company that is responsible for cleaning the channels to check whether a certain limit has been reached or not. This can be done by hand, for example by using a device that uses a pulse-echo technique[2]. Since this measuring has to be done by hand it is quite resource intensive; It takes a lot of time and it requires a person to be physically present at a location to perform the measurements. Therefore we were asked by the cleaning company "VETkanaal"[1], hereafter referred to by as the client, to develop an online sensor system that is able to measure and monitor the thickness of the grease layer in the channel autonomously.

1-1 Current State of Affairs

At this moment in time there are no guidelines or regulations regarding the cleaning of extraction channels in the Netherlands. That being said, that is that there are none issued by the government. Some insurance companies do demand cleaning of the extraction ducts within a certain time span or at a maximum thickness of 500 micron of the grease layer. Since there are no regulations there are less constraints on the design of such a device.

There are different principles which can be used to measure the amount of grease in a channel. For example it is possible to measure the bulk physical properties (density, electrical conductivity and ultrasonic velocity)[3] [4], the absorption of radiation (UV-visible, infrared, Nuclear magnetic Resonance and X-ray absorption)[5] or the scattering of radiation (light scattering and ultrasonic scattering) [6]. Another way of measuring is via acoustic properties such as the resonance frequency or the attenuation due to the deposition [7] [8] [9]. Even though that there are many different principles that can be used, not all are equally suitable because of differences in accuracy, implementation or cost.

A study has been conducted with the intention of developing a low cost sensor to measure the fat thickness in kitchen extraction channels [10]. Models have been built to study the behaviour of grease particles in the ducts and to differentiate between cooking processes in which the ducts might be involved. This study has been done because currently available technologies are too expensive to be implemented in the development of a low cost sensor. This study explores the possibilities of sensors in the field of optical and mechanical solutions. The study showed that the mechanical approach to the measuring was not that promising, but that optical solutions are.

However at this moment there exists a patent by the company Halton which claims almost all variations of optical sensors for the use of measuring grease in ducts. Their patent uses light to estimate the thickness of the fat-layer within ducts [11]. It works by applying a surface in the duct on which the fat layer will form. Then a light will be shed onto the surface and based on certain properties the thickness can be determined. This patent claims both the measurement methods concerning measuring the light that goes through the surface as well as measuring light that is reflected from the surface. This patent has been developed towards an existing product [12].

Since the client wants to be able to sell the to be designed sensor network it is required to look at other options of measuring the grease thickness. As mentioned before the thickness measuring is now done by hand by using a pulse-echo technique. Before this technique a measuring comb was used. After literature research it became clear that there are many different solutions regarding the measuring of deposit in channels but almost none aiming at grease in kitchens or at grease itself. There has been done a lot of research aimed at

FOG's (Fats, Oils and Grease), but each of these have a different type of acid composition [13] making it difficult to rely on certain properties of the deposition.

1-2 Problem Definition

The developed sensor system should be able to reliably perform grease thickness measurements independently at several locations in the extraction channel. The results of these measurements should be reported and transported to a central location accessible by the cleaning company. The sensor system should be able to measure the grease deposition in extraction channels of kitchens (for now). The sensor should indicate when the thickness of the grease layer has reached certain levels and therefore indicate when the extraction channel should be cleaned. The measured information should be made available online at a central location thus making remote monitoring possible. However when looking at the grease layer formed in the extraction channels of the mentioned kitchens it is not only fat or grease that will precipitate against the insides. This layer will also contain moisture and other components. Therefore a measuring method has to be determined that is able to cope with these compositions and accurately measure the thickness of the formed layer. Concluding: the goal of this project is to identify the most promising measurement method and strategy for measuring the thickness of grease and to design a sensor module with electronic read-out and communication module.

This specific report will cover the proof of concept that will be designed regarding the communication and system design of the sensor network. It describes how the sensor network will deliver its information to the online platform, how the network will be interconnected, how it will transport the required data and how the system will be supplied with power.

1-3 Chapter Organisation

At first this report will describe the program of requirements of the to be designed sensor network as a whole. These are the requirements set by the client. After the general requirements, the specific requirements for the communication and system design will be discussed. When the requirements have been discussed the next chapter will describe the design process of the communication and power supply. After the design process the implementation of the design will be discussed. The final chapter will discuss the results, the conclusion and the recommendations towards producing a product ready to be sold.

Programme of Requirements

As mentioned in the introduction the goal of this project is to identify the most promising measurement method and strategy for measuring the thickness of grease in extraction channels. This method has to be designed and implemented in the way of a sensor module with electronic read-out and communication module. The sensor system should be able to reliably perform grease thickness measurements independently at several locations in extraction channels. The results of these measurements should be reported and transported to a central location accessible by the cleaning company. The aim is the development of a product that could be sold to other cleaning companies in such a way that the system will be implemented at members of for example the catering industry.

This description of the project and therefore the product define a number of requirements. These requirements can be divided into two different sets: a set of requirements for the complete system and requirements for the sub-system that will be discussed in this report. Most of these requirements are the result of conversations with the client, VETkanaal.

2-1 General System Requirements

The sensor system has certain demands that have to be fulfilled and requirements where trade-offs can be made. The following requirements are considered as the mandatory requirements:

- The system must be independent of orientation, length and shape of the extraction channel. The cross-section of the channel is at least 12.5 cm;
- The system has to be able to either withstand or not get in contact with the chemicals that are used for cleaning. The chemicals that are used are highly alkaline chemicals;
- The maintenance of the system has to be simplistic in such a way that it can be performed by anyone who was not involved in the design process;
- The sensor must be able to measure and quantify the thickness of grease in an extraction channel;
- The sensor must be able to measure accurately within the desired range of 0 to 600 micron;

- The sensor must be able to quantify a level of grease thickness that is considered dangerous. Usually a level of 500 micron is considered dangerous;
- The sensor must be able to withstand and operate in an internal temperature of the duct of 100 degrees and down to 20 degrees;
- The measurement results of the sensor must be independent of the different types of material that are used for the extraction channels;
- The acquired measurement results must be displayed on some form of online platform.

One of the requirements is that the results must be displayed on some form of online platform. This is because the client wants to be able to have an overview of all sensors. The client has agreed that for the prototype it is sufficient to present the sensor data on a small webserver. This is sufficient since designing an online platform has already been done by a third party. This means that presenting the data in the right format on a small webserver will be acknowledged as meeting the requirement. Anything more than that is not within the scope of this project.

The following requirements can be seen as trade-off requirements. They are desired by the client for the design but are not necessarily needed for the prototype.

- The measurement design should differ from already existing patents on grease measurements;
- The system should have a lifetime of at least one year;
- The sensor should work for every form of grease deposit since the consistency of grease deposit differs per site and per extraction channel;
- The sensor should not influence the even grease deposition in the extraction channel;
- The accuracy of the measurement results should be within ± 30 μm within the range of interest;
- The measurement results should be independent of the surrounding temperature;
- The production cost should be taken into account

2-2 Communication and System Design Requirements

Aside from requirements that are set upon the system as a whole there are certain requirements that apply to the communication and system design. These are as follows:

- The system must be able to measure from one up to a maximum of 8 different locations per channel (or system);
- The design must be able to measure at all sensor locations in the channel at least once a day;

- The system has to be able to reliably communicate with a maximum distance between the first and last sensor of 100 meters;
- The minimum distance between two sensors is 1 meter;
- The system has to supply the sensors and their read out circuits with power, either wired or by using batteries;
- The sensor data has to be reliably transported in some way. This can be done wireless or wired;
- The data of the sensors should not interfere with each other;
- The acquired sensor data has to be made available on an online platform.

Design Process

Now that the problem has been defined and that the design requirements are known it is possible to start with the design of the product. Before any specific communication or power supply design can be made, an overall system design needs to be created. This chapter will present the design choices and process related to the design. At first the general design will be discussed for the communication module and the power supply. After the general design the motivation behind the design choices per module will be discussed. The implementation of the design and choices made can be found in chapter 4.

3-1 System Overview

Based on the system requirements, design choices had to be made. In this section the current design will be presented.

The system is going to be split into two different types of nodes. There will be multiple sensor nodes that are responsible for the sensor and the read out circuit, and a single main node. Each node will be controlled by a microcontroller, in this case an Arduino. A master-slave type protocol is used, where the main node will act as the master of the system, while the sensor nodes can be seen as the slaves. The sensor nodes are placed on the extraction channel and measure the thickness of the fat layer at that location in the channel.

The read out circuit of the sensor will deliver the measurement as an analog output. Processing of the data is required for further actions. The microcontroller of the sensor nodes will be able to preform the analog to digital conversion as well as needed calculations to acquire the desired result. The result will then be transmitted to the main node. The main node is responsible for data collection, processing and management of the sensor nodes. The main node is connected to the internet and will transmit the data in the right format to a small web server.

3-1-1 Communication

The communication between the sensor nodes and the main node will be done by means of a wired connection using UTP cables. Using a wired connection, a serial bus communication between the nodes has been created. The MODBUS master-slave protocol was used to regulate the communication of all devices on this serial bus. The main node functions as the master, whereas all the sensor nodes will function as the slaves. To transmit data over the UTP cables the hardware protocol RS-485 was used. The UTP cables will also be used to supply power towards the sensor nodes and therefore the sensor read-out circuit.

figure 3-1 shows an overview of the designed system, the connections between the nodes and the connection towards the sensor. It shows a total of three nodes, one main and two sensor. For the proof of concept that will be designed it has been agreed upon with the client that this will be sufficient to demonstrate the correct operation of the communication system.

3-1-2 Shield Design

Figure 3-1 shows that a total of three different shields for the nodes were designed. Two of these shields are own design while the third one is prefabricated. The personally designed shields are necessary to connect the nodes to the databus. The end sensor node in the system needs only one connection with the databus, while the connection node needs two in order to maintain the databus. The third (prefabricated) shield is the shield needed for the connection with internet.

The design of the connection shield is quite straightforward. The shield is used to connect the pins of two RJ-45 sockets. These sockets are used to build the databus with UTP cables. Because of the sockets it is possible to tap a couple of the connections towards the Arduino. Wires 4 and 5 of the UTP cable are used to transmit the data, wire 6 to transmit the power and wire 8 is used as common ground.

The RS-485 protocol transmits data as a differential pair. A digital 1 means that the value on line 4 is higher than on line 5 and vice versa for a 0 (compared to the common ground on line 8). One problem does exist: The output of the serial port of the Arduino is not compatible with this way of transmitting. A dedicated IC is needed to convert the serial data from the Arduino to the differential pair on the databus and the other way around [14]. These IC's contain two enable signals to enable the read- and write functions. These enables are controlled by the Arduino. The enables are necessary because without these signals the IC would be sending and receiving data simultaneously. When this is happening the output becomes quite noisy and the data does not reach the other devices in a correct way. An overview of the shield is shown in figure 3-2

The design of the end shield is similar to the connection shield. The main difference is that this shield needs only one connector as they are either the starting or finishing node of the bus. Aside from one connector, the end shield contains a line terminator in the form of a resistor that is connected at both ends of the data lines. This is to protect the data on the bus from possible reflections. The value of this resistor should be matched to the line impedance, which is defined as 100 Ω [15] [16]. Figure 3-3 shows two extra resistors. These are only present on one of the two end shields and are used to polarise the data lines when there is no data present. The values of these resistors are specified by the MODBUS protocol, and should be between 450 Ω and 650 Ω[16].

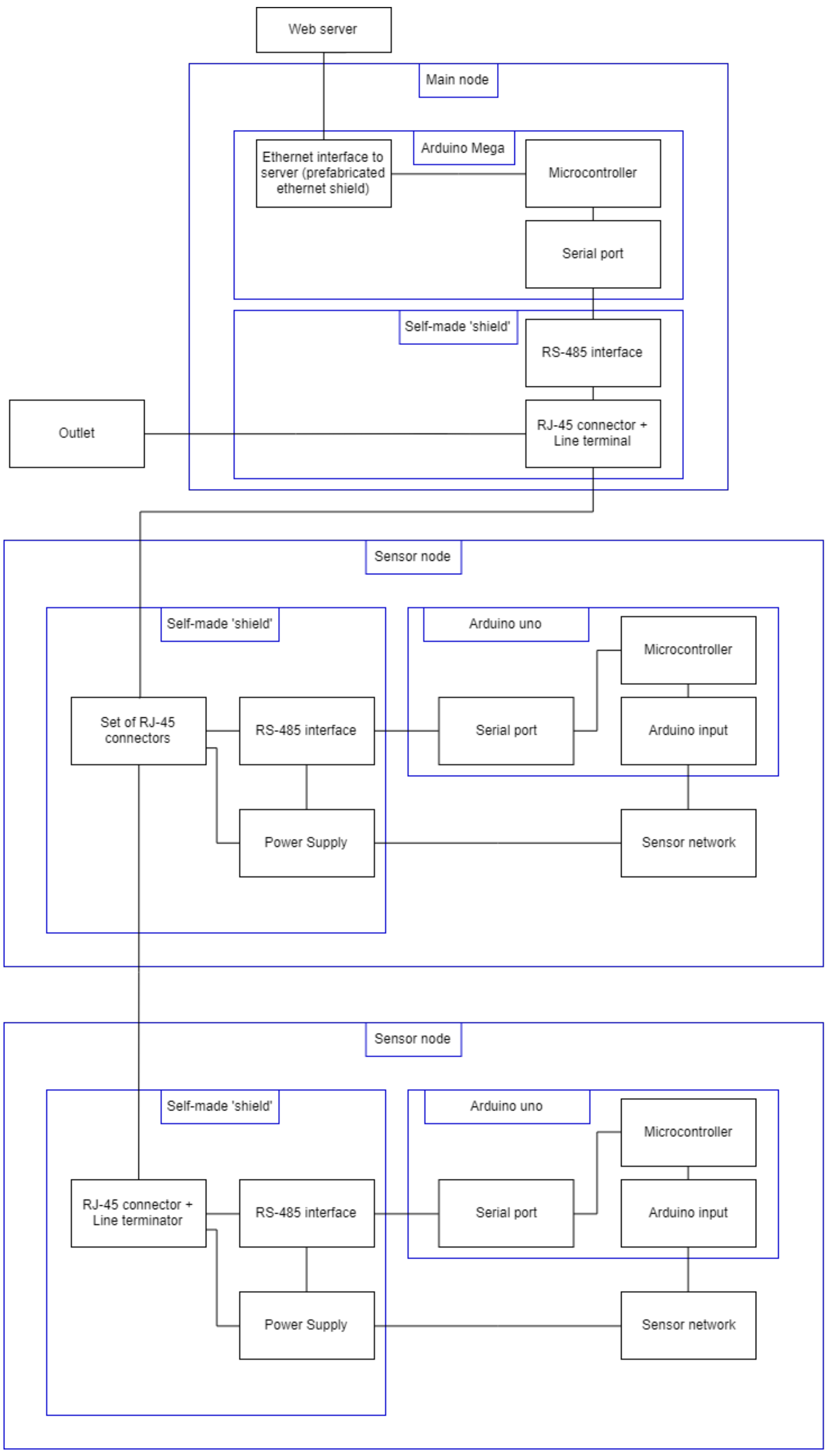


Figure 3-1: An overview of the entire communication system

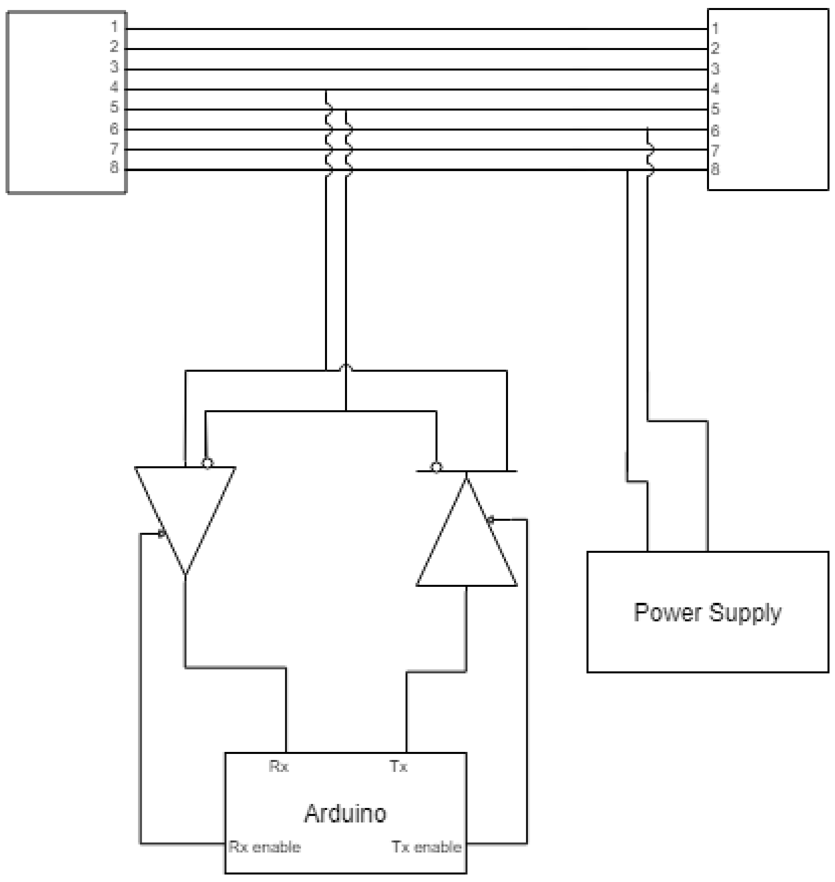


Figure 3-2: An overview of the connection shield

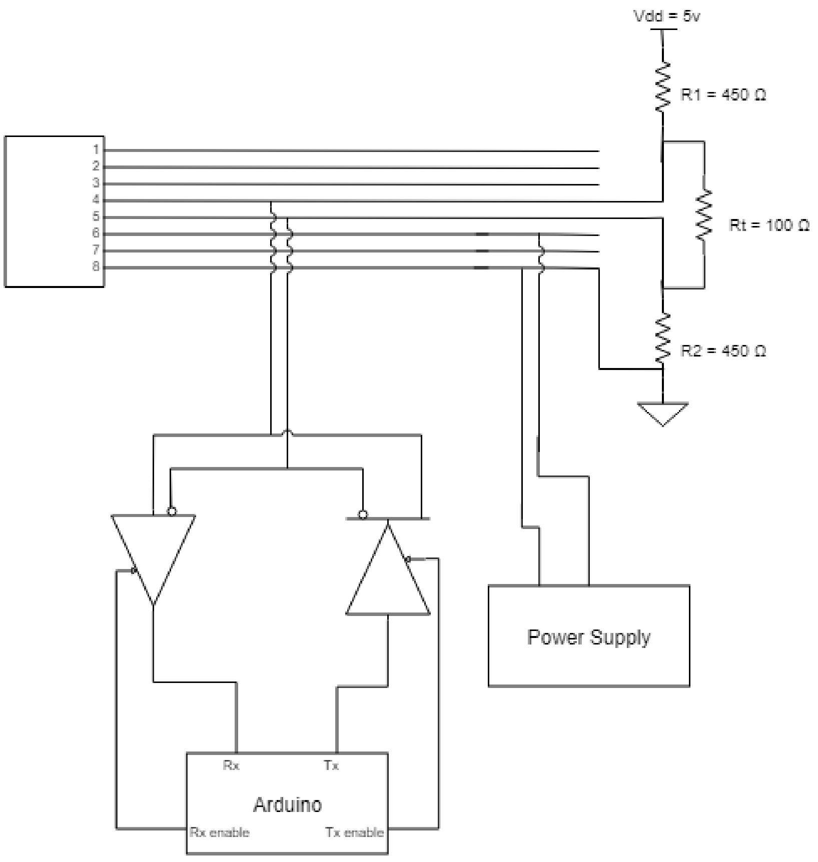


Figure 3-3: An overview of the end shields; R1 and R2 are only present at one of the two shields

3-1-3 Power Supply

Since there are two different implementations for the measuring method, the acoustic and capacitive way, there are also two different networks with each their own sets of power supply requirements. Therefore there are two different designs for the power supply. The general approach however remains the same. An adapter is connected to an outlet to convert the 230V AC from the mains towards a lower DC voltage. this DC voltage is then distributed over the UTP cable where line losses have to be taken into account. The voltage will be supplied towards a DC voltage regulator at the sensor node to convert the voltage to the required supply voltage. This implementation is shown in figure 3-4.

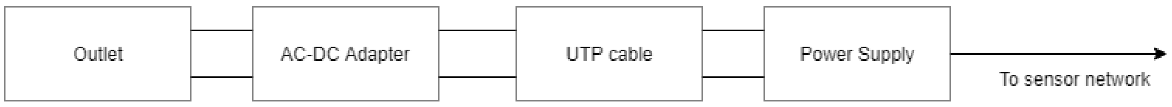


Figure 3-4: An overview of the power design which holds for both designs; They differ in implementation of the power supply

Capacitive Sensor Supply

The capacitive sensor needs a 5V DC supply with available power of approximately (or less than) 50mW. To accomplish this a simple voltage regulator will be implemented. Since Arduinos are used in the sensor nodes, they have a built-in 5V supply, it has been decided to use this connection for the prototype. The design behind this supply will be explained in this section in case the supply should have been implemented in a different way, as would be needed for a finalised product.

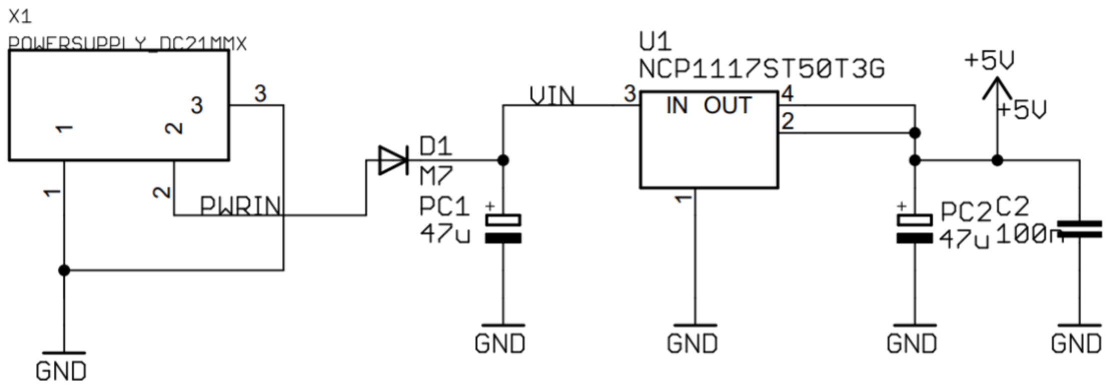


Figure 3-5: The voltage regulator from the datasheet of the arduino uno [17]

Figure 3-5 begins with a square block with three connections, denoting the power supplied to the arduino. In the design this would be power delivered by the UPT cables, and should be between 7 and 12V DC in order to be converted by the linear voltage regulator. If the supply voltage is not within this range the regulator is not able to operate properly. This regulator, in figure 3-5 denoted as U1, has an in-and output capacitor in order to keep the regulator stable. The diode D1 is used to protect the system from a negative supply. Arduinos are used in all kinds of projects and by people with a wide range of experience with electronic devices. Bearing this in mind, a safety diode makes sense. However, in the finalised system the supply voltage should be fixed and positive, therefore making this diode redundant.

Acoustic Sensor Supply

The acoustic sensor network requires a more complex power supply. This sensor requires a $\pm$ 15V supply and a connection to ground to operate properly. These voltages have been set as a requirement by the acoustic implementation group. The amount of required power by the sensor and the read out circuit were unknown at the time of the design. Therefore it has been decided that in this design all choices were made to supply as much power as possible. To deliver both a positive and a negative voltage a voltage regulator [18] was used in combination with a dc-dc inverter [19].

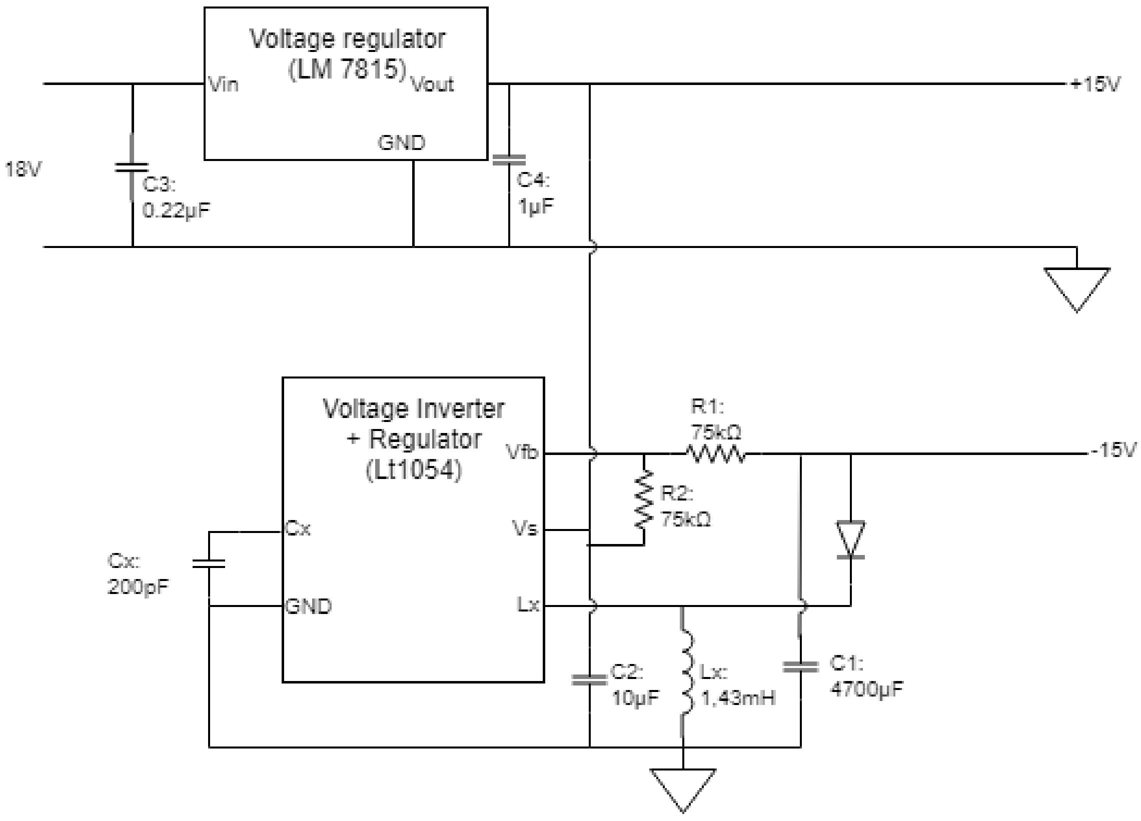


Figure 3-6: The converter used to create a ± 15V power supply

For this design an adapter delivers a 18V DC voltage to the UTP cable. This voltage is enough to cover for possible losses within the cable and The dropout voltage of the regulator. The voltage regulator delivers a positive 15V output. This voltage is connected as the 15V line and towards the inverter. This inverter outputs a -15V DC resulting in a total supply of ± 15V.

3-2 Design Motivation

At this point in the report the design has been presented. There have been made quite a lot of design choices regarding the communication module and the power supply. But these choices have not yet been motivated with respect to the program of requirements. This section shall motivate the design choices with respect to the requirements.

3-2-1 Communication System

The design choices for the communication system are choices related to the topology, the used protocols and the chosen hardware implementation of the entire system. The global thought process behind the design choices was to either meet the demands set in the programme of requirements or to keep the design simplistic because of time constraints.

At the time of designing the system it was undetermined in what format the sensor and it's read out circuit were going to deliver the measurements. It was unsure whether the data was going to be analogue or digital and if data processing was necessary. Due to this uncertainty it was decided to keep the design as flexible as possible by using a micro-controller in each sensor node (at least for the prototype). In this way more flexibility was available than when having to rely on analogue hardware to receive, process and transmit the data.

During the meetings with the client it became clear that the client did not have a preference towards the way of transmitting data. Wireless and wired communication were both an option. The wired option seemed as the more favourable option to go for. This because wired communication is more reliable than wireless as well as that the option to combine the data communication with power supplying is possible. In case of a wireless system it is needed to power the system in a different way. Batteries would be a alternative choice because of the reachability of the sensors. The downside of using batteries would be the need of having to store energy and to have optimised power dissipation within the design. This due to the fact that the system has to operate for at least a year before the batteries should be replaced.

Aside from choosing wireless or wired communication the size of the network was a big constraint during the protocol choices. The maximum distance between the first and final node in the system is set upon 100 meters. Therefore are popular protocols like I2C not suited for networks that have to cover that range. Because of this rather large range the RS-485 protocol has been decided upon to use as the hardware protocol. It was decided to implement this protocol using RJ-45 cables (UTP cables) as they are able to implement the RS-485 protocol. Furthermore these cables are not expensive, around 1 euro per meter, and were directly available for testing purposes. Another benefit of these cables is that they are available in lengths that would be needed for the system.

Although RS-485 specifies the hardware layer and guarantees quality for the desired distance a software protocol was needed to ensure all sensor nodes could communicate with the main node in a sufficient way. One of the main constraints regarding the protocol is that the system has to be scalable. A system with one sensor should work just as well as a system with eight sensors. Having a parallel connection to each slave would mean redesigning the entire system for every case, e.g. for one up to eight slaves. At the same time not all sensor nodes have to be able to communicate with the main node at the same time. Therefore a serial bus was used since all the sensor nodes should be doing is transmit information to the main node if the main node asks for it. For this a master-slave protocol seemed like the best fit. After a bit of research on rs-485 based master-slave protocols the choice fell upon the communication protocol MODBUS [16]. MODBUS is a well-documented and commonly used protocol making is easy and quick to implement.

Now that the data and communication protocols were decided it was possible to look at the way of communication between the nodes. For the micro-controllers it was decided to keep it simple for the prototype. This has been done by looking for the most flexible and optimised processor and keeping in mind that developing a whole development board around it would simply take up too much time. Therefore the choice was made to use Arduinos as the microcontrollers. Arduinos have a huge user community which means that there can be found may support in the form of libraries and tutorials online. Given the time constraints of the project this resulted in the best option for an easy and fast implementation.

Now that it had been decided that Arduinos would be used it was necessary to decide upon what type of Arduino would best fit the needs. The sensor nodes would not require a complex microcontroller. The sensor nodes only have to be able to perform simple calculations at best and are mainly used to transmit the data. This is why an Arduino Uno was chosen for the sensor nodes. The main node however had to have a bit more processing power, assuming that the main node should be able to perform more data processing on the sensor data. Aside from more processing this node should also be able to connect to the internet and transmit the data towards a small webserver. Therefore an Arduino Mega was chosen to be used as the microcontroller in the main node. If during the implementation of the system it would turn out that the system needs more processing power to process the sensor data a second microcontroller could be implemented. This second microcontroller could either be placed in the sensor nodes or in the main node and would then primarily be used for data processing.

3-2-2 Power Supply

Now that the motivations behind the communication design have been explained it is necessary to look at the power supply choices. As said in the communication motivation it was decided to use the UTP cable wiring to transmit power. This is possible since the MODBUS protocol uses only three wires for its communication (five in a more complex configuration, which is not necessary in this design). Since a UTP cable has eight wires this creates the option of using the remaining wires to supply power to the system. An adapter would be connected to an outlet at the main node which in turn would be connected to a power controller. This controller sets the voltage and current that are supplied to the sensors by making use of simple analogue hardware.

As explained in the design overview there are two types of sensors that can be connected to the system. Each sensor requires its own, different, power supply. The first one, the capacative sensors, needs a simple 5v connection that is created by using the internal voltage regulator and 5v connection on the Arduino Uno development board as explained earlier.

For the power supply of the acoustic sensor there were roughly three options to meet the requirements set by the acoustic implementation:

- Build and design an AC-DC converter and use a three tap transformer.
- Use a voltage regulator combined with an inverter and regulator.
- Have a dedicated DC-DC converter that transforms the input voltage to the right output voltage.

The design of the power supply has been postponed till the end of the project since it was not possible to start sooner. This because the read out circuits and the power requirements of the sensors were still undetermined since they depend on the implementation of the measuring concept. Since the design was postponed there was approximately one week left to design the power supply. Because of this it was decided that it was not realistic to properly design, implement and test an entire AC-DC converter due to the time constraint. Therefore it was decided to look at the other two options. Using an available dedicated DC-DC converter IC has the downside that a significantly lower maximum output power (0.5 W vs 2W) is available than with the other option. As it was not determined how much power the sensor and read out circuit was going to need the choice was made to implement the option that could ensure the largest amount of available power, the voltage regulator with inverter combination.

The implementation of this choice and the component values belonging to this design are shown in figure 3-6. The shown values are the ideal values. The components used in the implementation might differ from these values depending on availability and accuracy. All the values shown in figure 3-7 are derived from the available datasheets of the inverter and regulator. The motivation behind the component values will be explained below.

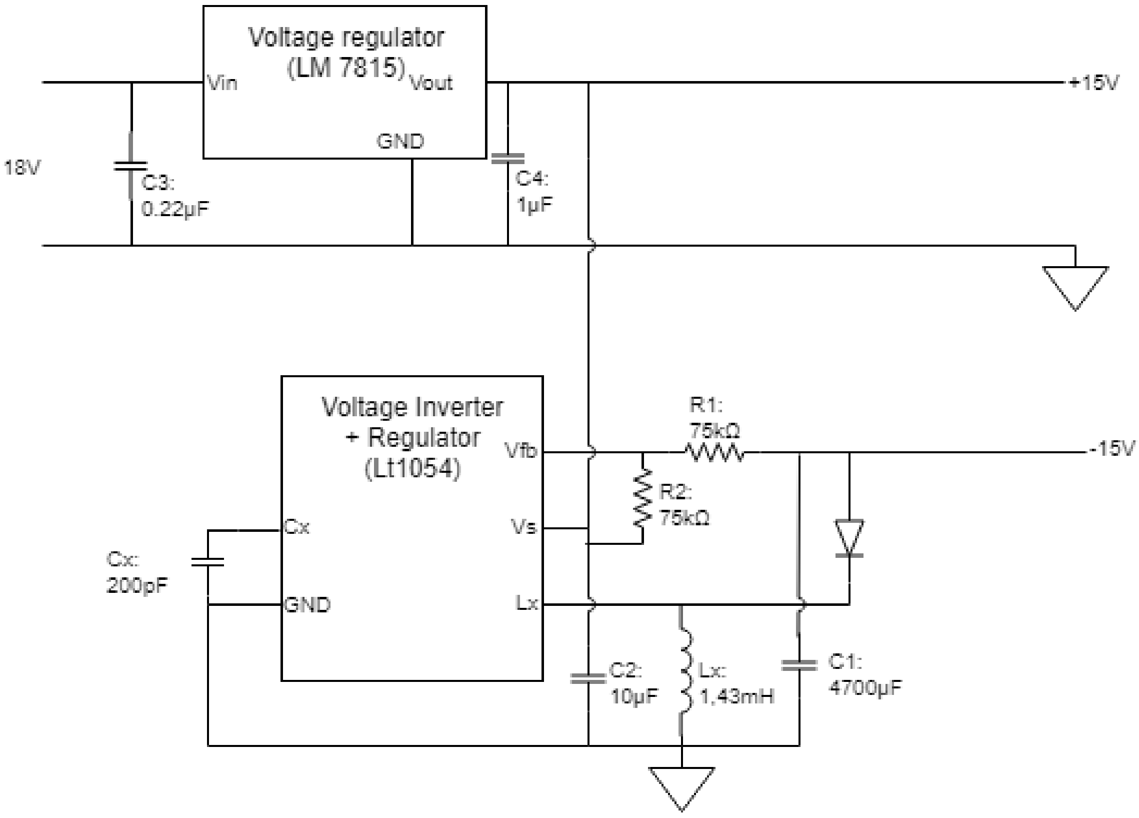


Figure 3-7: The converter used to create a ± 15V power supply

The two capacitors C_3 and C_4, located at the in-and output of the voltage regulator are used for stability. Capacitor C_2 is used for the same reason at the output of the inverter. Their recommended values are specified by their respective datasheets. The voltage inverter is a buck-boost converter and thus requires an oscillator. The frequency of this oscillator is set by capacitor C_x. According to the datasheet a capacitance of either 200pF or 47pF can be used in order to set the frequency to 10kHz or 40kHz accordingly. A lower frequency results in a higher output power, therefore it was decided to use a 200pF capacitor to set the oscillator frequency at 10kHz.

The value of inductor L_x determines the peak output current. The relation between the value of this inductor and the peak current is given by:

$$L_x = \frac{V_{in}T_{on}}{I_{pk}} \tag{3-1}$$

Equation 3-1 shows that a lower inductor value results in a higher peak current. This means that in this case the inductance of L_x should be chosen as small as possible and is limited by the maximum peak current that the inverter can handle. This is set upon 525mA. Furthermore V_{in}= 15V and $T_{on} = 0.5T = \frac{1}{2f} = 50\mu s$. This results in an ideal inductor value of:

$$L_x = \frac{15 * 50 * 10^{-6}}{525 * 10^{-3}} = 1.43mH \tag{3-2}$$

The actual value of the used inductor should by no means be lower than this value as that would overload the inverter.

Capacitor C_1 defines the ripple of the output voltage as it is used to supply current to the load while the inductor is not delivering any power. This ripple voltage has two components which have an approximate 90 deg phase difference. One component is caused by the effective series resistance (ESR) of the capacitor. The other component is caused by the change in stored charge of the capacitor with each output pulse.

The first component can be derived simply from ohm's law:

$$V_{ESR} = I_{pk} * R_{ESR} \tag{3-3}$$

This shows that the ESR of the used capacitor should be as low as possible, preferably as close to zero as possible. The second component follows from:

$$V_{dQ} = \frac{Q}{C} \tag{3-4}$$

where Q is the charge dissipated during the time where the capacitor delivers energy to the load. When combining the known results the following equation is found. The derivation of the formula can be found in appendix A-1.

$$V_{dQ} = \frac{V_{in} * T_{on} * T_{off}}{2LC} \tag{3-5}$$

Where $T_{on} = T_{off} = 0,5T$. From equation 3-5 it can be seen that to decrease the component of the ripple voltage the capacitor should be as big as possible.

The ratio between the resistors R1 and R2 set the output voltage according to the following equation:

$$V_{out} = -\frac{R1}{R2} * V_{in} \tag{3-6}$$

Since the desired relation between V_{in} and V_{out} is -1 in this case, the values of R1 and R2 should be taken to be equal. In order to reduce energy dissipation, these values should not be chosen too low. Typical values as shown in the datasheet were used to set these values at 75 kΩ

Implementation

This chapter describes the process related to the implementation of the design as discussed in chapter 3 and presents achieved results.

4-1 Implementation Process

This section will discuss the implementation and results per module in the same order as the previous chapter. At first the implementation of the communication module will be discussed and then the same for the power supply. After the implementation the achieved results will be presented.

4-1-1 Communication System

Once all design choices concerning the layout of the communication system were made the implementation of the communication module started. During the implementation a couple MODBUS libraries were found that could be used on the Arduinos. One of these libraries contained all the necessary functionalities to let an Arduino function as a master [20], while another library gave similar functionality in case an Arduino had to behave as a slave [21].

At first the functionalities of these libraries were tested by simply connecting one slave to the master with a couple of wires (so without use of the rs485 databus). This can be seen as shown in figure 4-1.

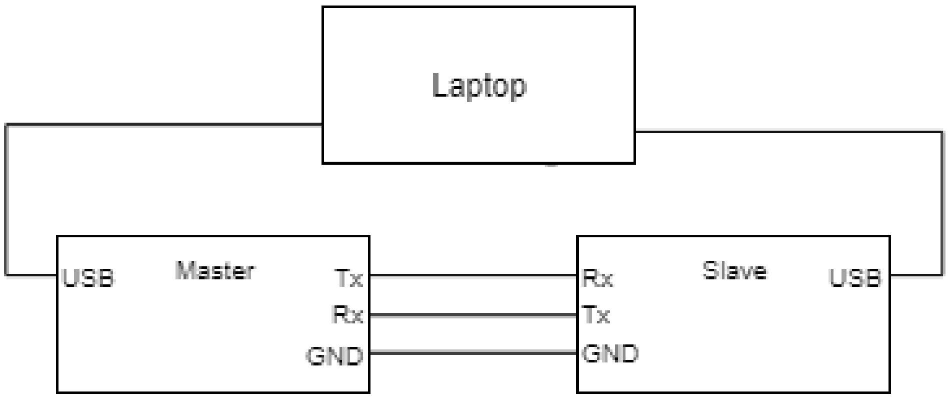


Figure 4-1: The first setup to test the software

By testing in this way insight in the library code was gathered. During the testing the databus was built at the same time. The test was executed as follow: the master requested data from a certain input register at the slave, which had that certain register connected to an analog data pin. On this data pin a potentiometer was connected. In this way the value of the data that had to be sent could be altered. Finally, the software of the slave was set up in such a way that it first printed the potentiometer value to a monitor on a PC and then initialised its communication line. The master could then request the data to be send and print this information on a different monitor (which was displayed on the same laptop). When the two values of the two monitors matched and when they would both change according to a change of the potentiometer the setup was deemed successful.

By the time this setup was working, the Rs485-bus was finished. This meant the start of the next phase of implementation, which was connecting one slave to the master over the Rs485-databus, as shown in figure 4-2:

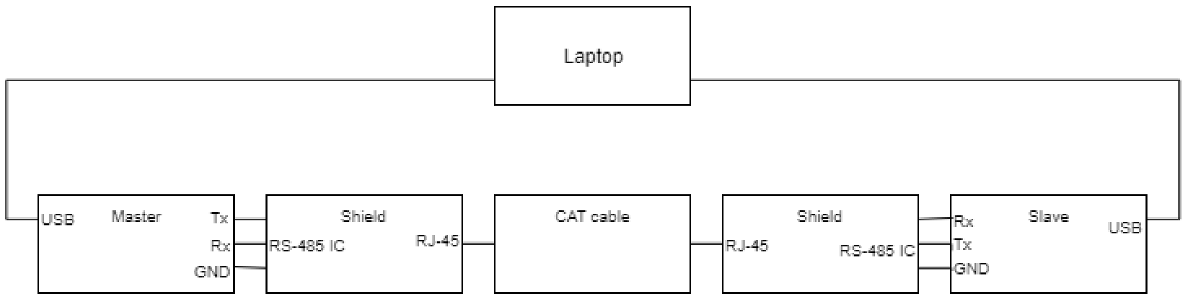


Figure 4-2: A first setup with the databus and only one slave

At the beginning, the results seemed quite unpromising; The slave could not see any request coming from the master. At this point an oscilloscope was connected to the setup to visualise the data signal at different places in the databus. Using the scope it could be seen that there was a significant amount of noise on the request at the slave's end. This noise caused the signal to be distorted and the master was unable to understand these signals.

At this point it was noted that both the receive-and write functionalities of the IC were always enabled in the current setup. After playing around with these enable signals, it became clear that this might have caused the issue. Fixing this required a slight alteration of some already existing enable-control of the slave-specific library. The master library did not have any of this functionality, but this could easily be implemented by following the example set in the slave library. After both the devices were set to only enable their write signal when they were actually trying to send something, and have their read signal disabled in the meantime, the communication between the master and one slave became successful.

The final step of the implementation involved having two slaves connected to the databus instead of one. This can be seen in figure 4-3.

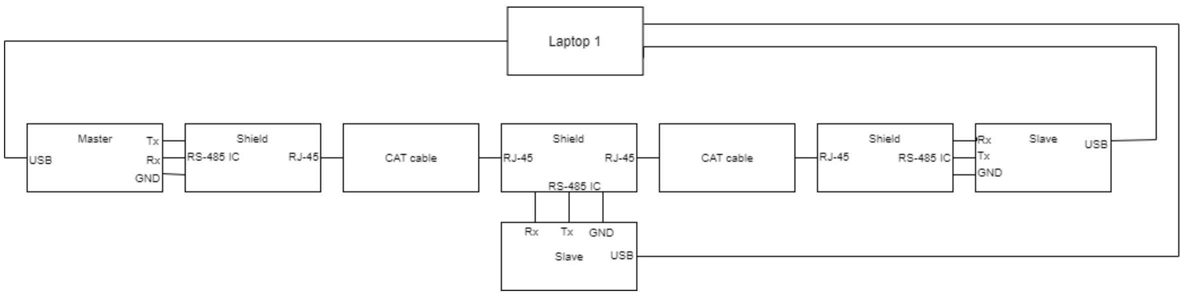


Figure 4-3: The final testing setup with both slaves connected

After some testing, it became clear that having enough time between receiving a message and sending another message to another slave was vital. If the master tries to send a new request directly after another slave has replied to an earlier request, the beginning of the new request and the end of the old reply are connected. This means that the new slave sees one big message. Since this message starts with the old reply, the new slave thinks there is still an ongoing communication between the master and the other slave and therefore does not respond. After implementing a small delay (1s) between the requests at the master, the complete data system was working properly.

4-1-2 Power Supply

At the moment of writing this report the implementation of the power supply has not been completed. While the power supply for the capacitive sensor is finished, the power supply for the acoustic sensor is not. This implementation still contains some flaws that currently have to be solved. It is still undetermined what causes the problems and therefore it is not possible to elaborate upon these flaws.

4-2 Results

The communication module can be divided into two parts, each with their own test results. There is the hardware layer of the RS-485 databus which is responsible for the correct transmission of the data and the software layer of the MODBUS protocol, which has to make sure this data is correct as well. First the results of the databus will be shown. This was tested by connecting the bus to a scope during transmission and looking at the waveforms at different positions.

In all measurements both signals were connected to a common ground. First off, the output of the master was compared to the input of the slave. This can be seen in figure 4-4. This figure shows that the data that is transmitted from master to slave is transmitted correctly.

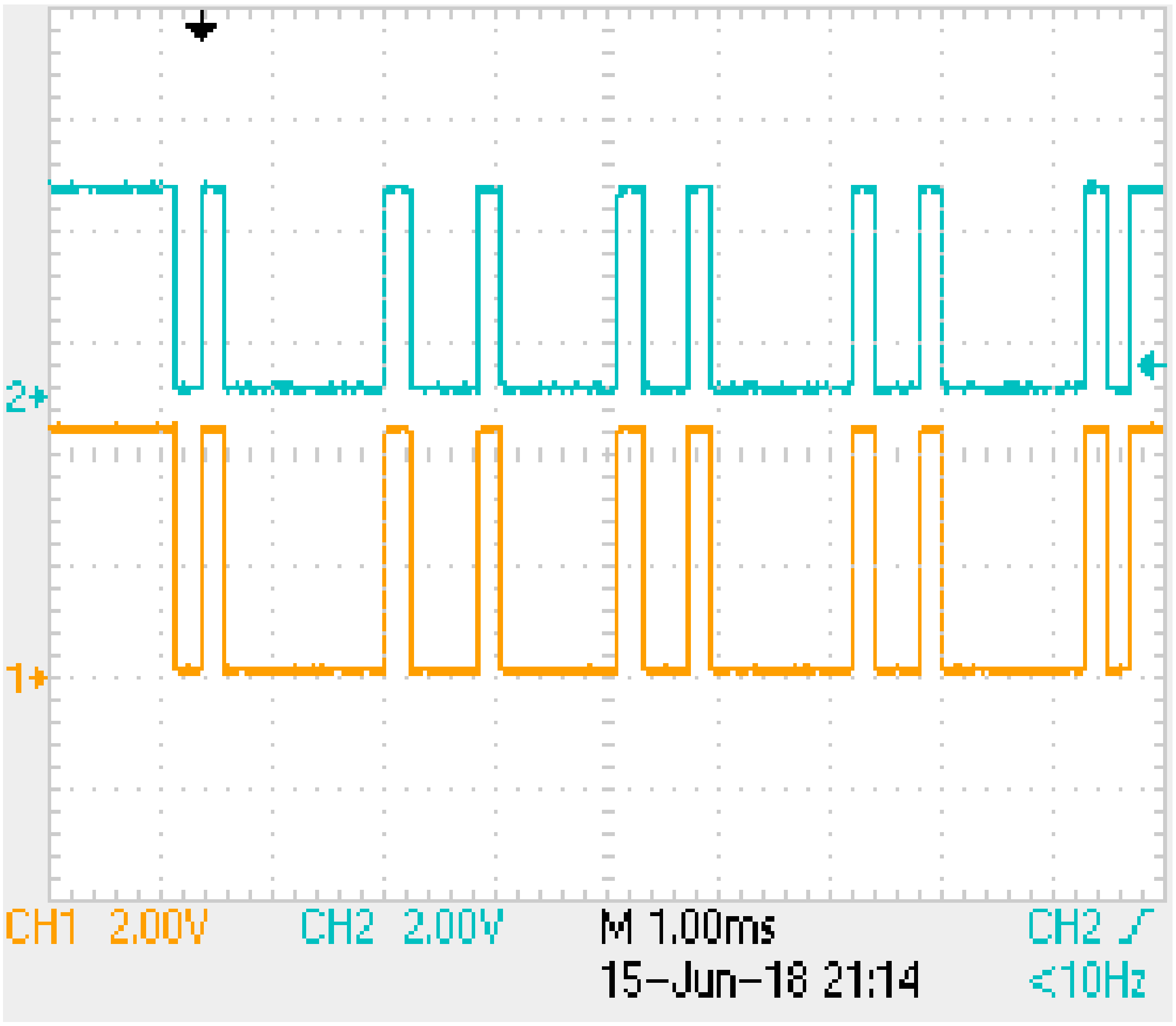
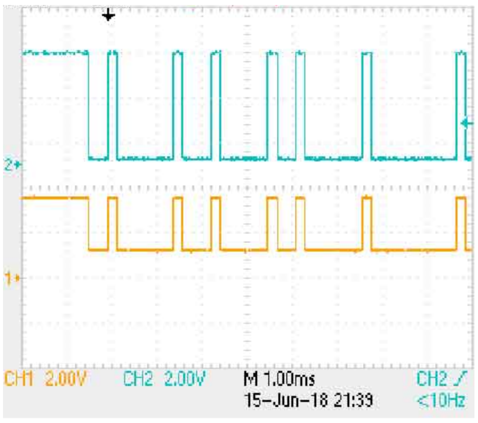


Figure 4-4: A comparison between the 'send' pin of the master and the 'receive' pin of the slave. The upper (blue) signal comes from the master, the bottom (orange) from the slave

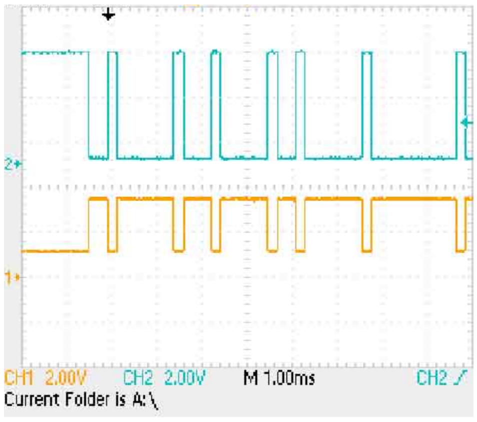
The question that remains is if the data also gets transformed conform the RS-485 protocol. To verify this, both data lines were connected to the scope in three different measurements, while being compared to the receive pin from the master. These results are shown in figure 4-5a, 4-5b and 4-6. Figure 4-5a shows a comparison between the 'send' pin of the master and the high line of the databus.

The databus signal in figure 4-5a resembles the transmitted signal, but with half the amplitude. This is due to the nature of the differential pair of the RS-485 line. the difference in amplitude between the two lines remains the same when converting by having a negative amplitude on the other line during a transmitted high signal.

Figure 4-5b shows that the signal on the dataline has half the amplitude of that on the output of the Arduino as well it has its values inverted. This is to be expected, as RS-485 uses a



(a) A comparison between the 'send' pin of the master and the high line of the databus



(b) A comparison between the 'send' pin of the master and the low line of the databus

differential pair. If transmitting a high value, one line is kept low while the other is high, meaning that a high signal at the Arduino results in a low signal at this line as explained above.

Figure 4-6 shows a comparison between the two datalines of the RS-485 bus, both compared to a common ground. This figure confirms that the databus is working as intended, by working as a differential pair.

Besides the hardware layer of the system, the software layer was tested as well. As mentioned before, the testing of the communication system was done by printing the received data value to the computer by using the built in serial monitor of the Arduino IDE. This data value was compared to the value of the requested data at the slave, which was printed to another monitor. This can be seen in figure 4-7.

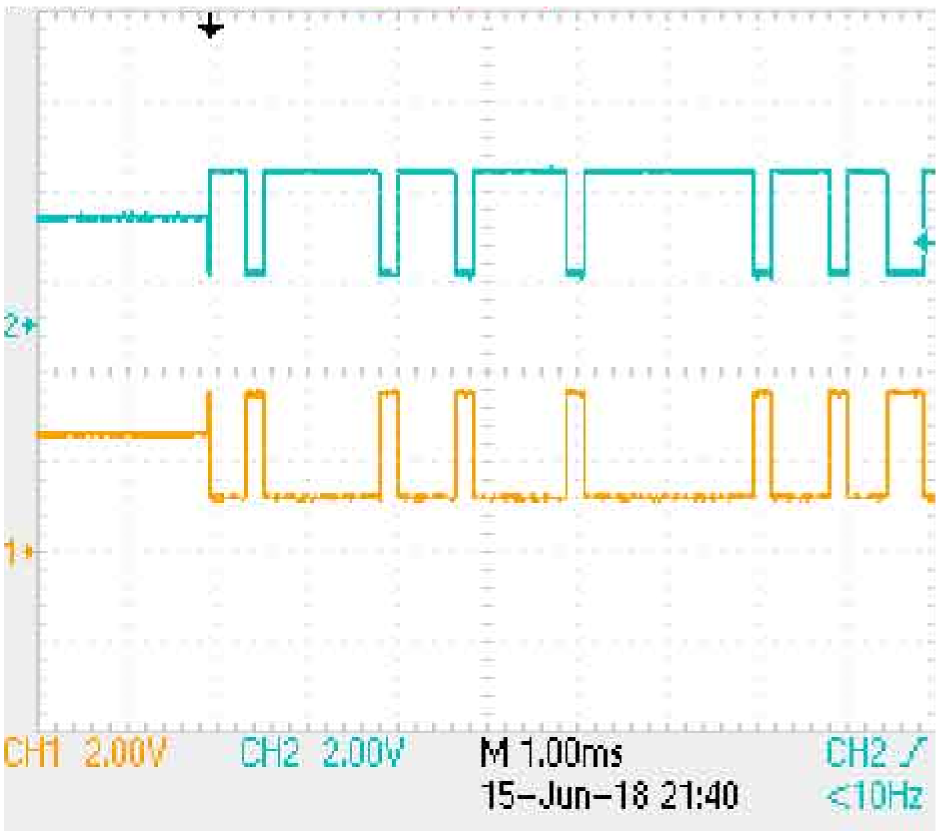
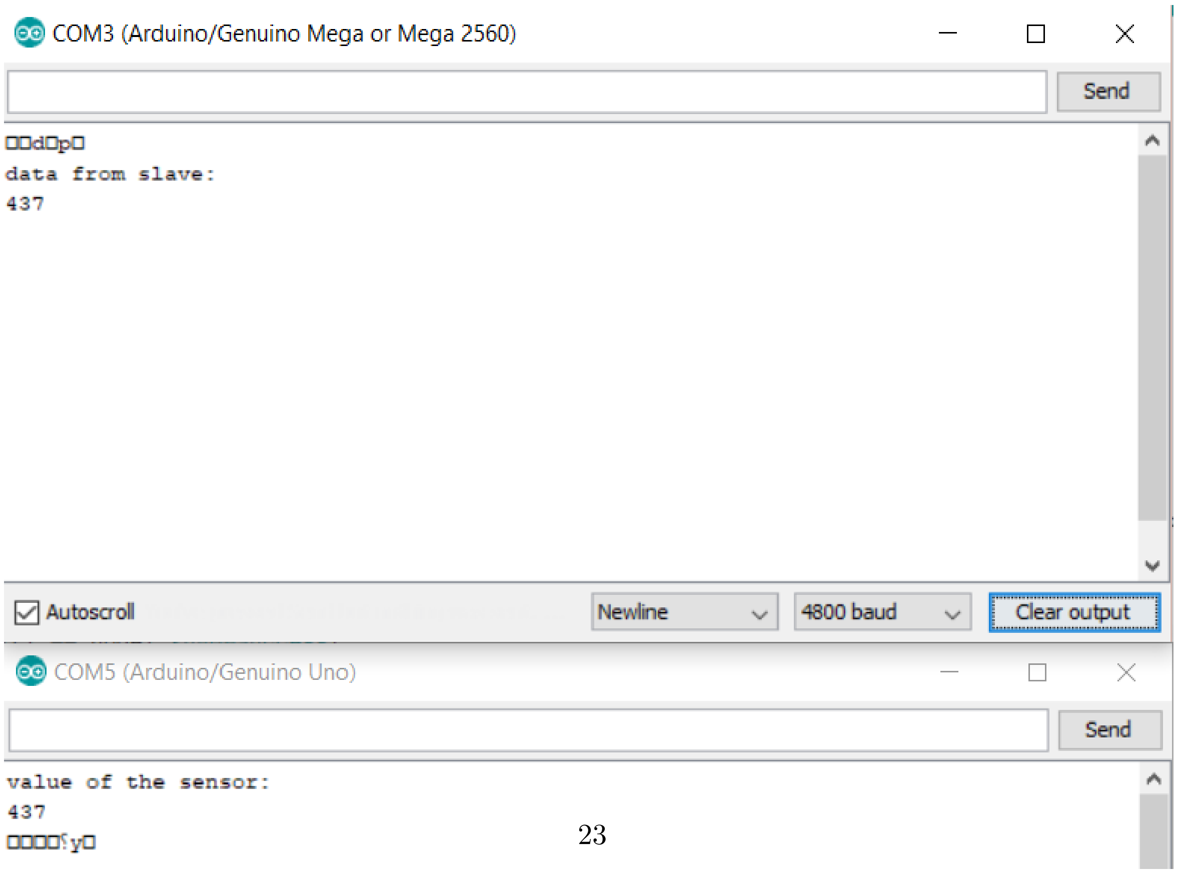


Figure 4-6: A comparison between the two lines of the databus

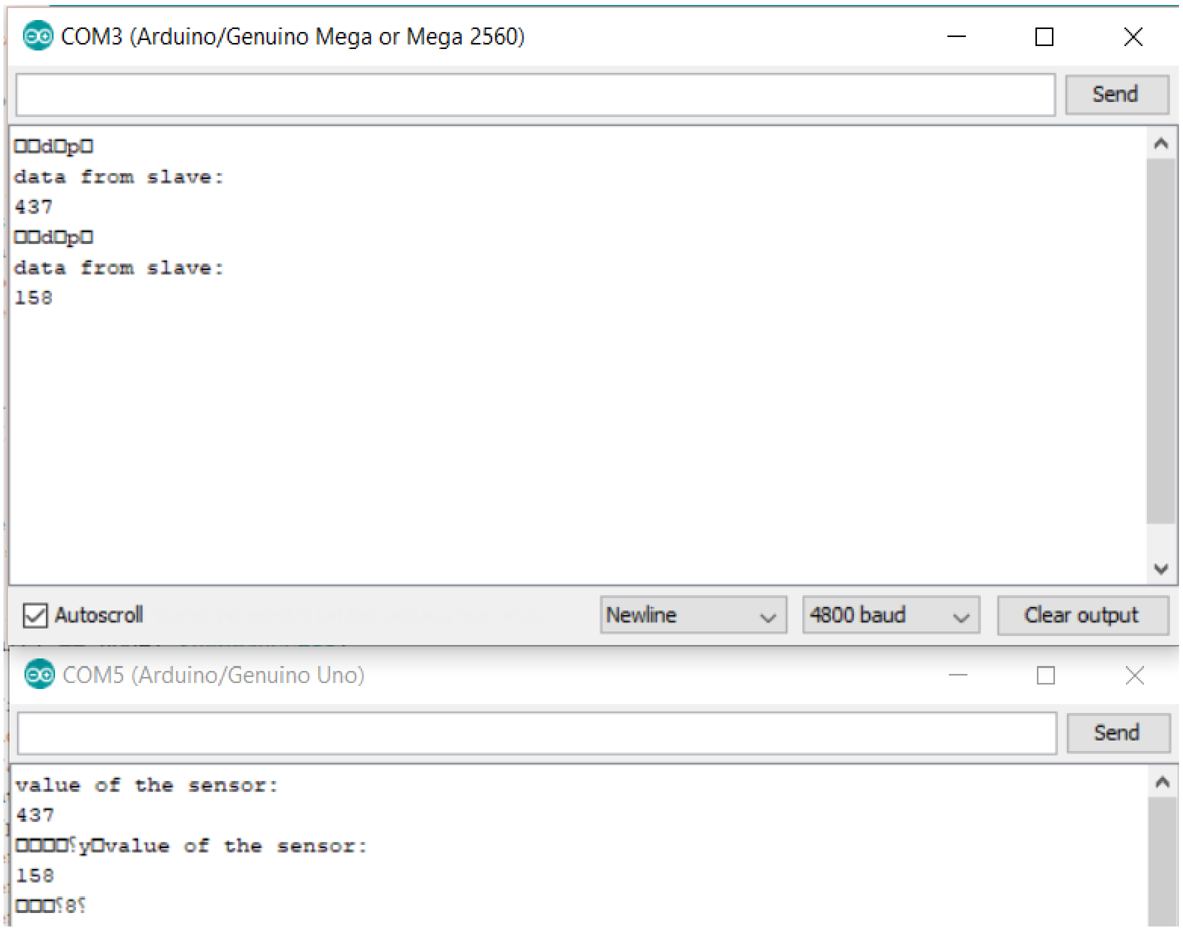

COM3 (Arduino/Genuino Mega or Mega 2560)
Send
Autoscroll
Newline
4800 baud
Clear output
COM5 (Arduino/Genuino Uno)
Send

```
□□d□p□
data from slave:
437
```

```
value of the sensor:
437
□□□□⸮y□
```

Figure 4-7: A first test of the communication system; The upper monitor is connected to the master, while the lower monitor is connected to a slave

Figure 4-7 shows that both the master and slave read the same value of the potentiometer, which was used to simulate a sensor. After the potmeter was turned a bit, another communication was executed. The result is displayed in figure 4-8.

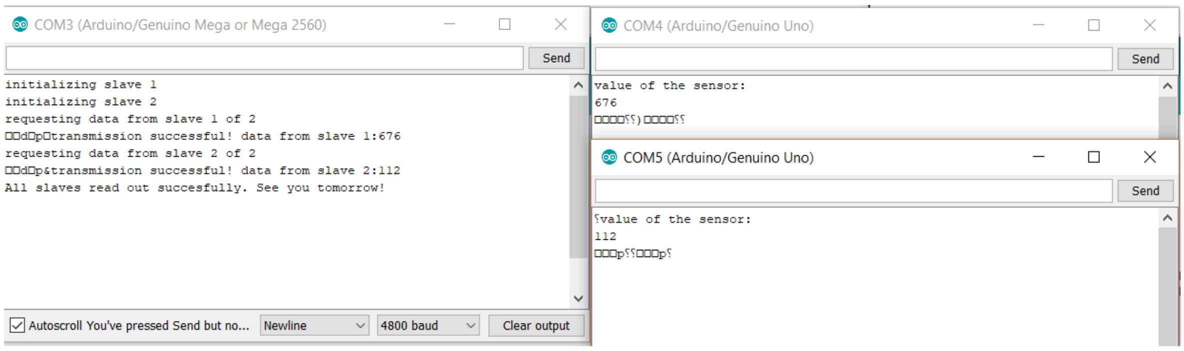

COM3 (Arduino/Genuino Mega or Mega 2560)
Send
Autoscroll
Newline
4800 baud
Clear output
COM5 (Arduino/Genuino Uno)
Send

```
□□d□p□
data from slave:
437
□□d□p□
data from slave:
158
```

```
value of the sensor:
437
□□□□ʕy□value of the sensor:
158
□□□ʕ8ʕ
```

Figure 4-8: A test where the potmeter was turned, showing that altering the value of the 'sensor' actually alters the value of the sent data

This test has been repeated for all three testing setups. For the final test, the test that has been done with two slaves, two laptops were used. These test results are shown in figure 4-9.


COM3 (Arduino/Genuino Mega or Mega 2560)
Send
COM4 (Arduino/Genuino Uno)
Send
COM5 (Arduino/Genuino Uno)
Send
Autoscroll You've pressed Send but no...
Clear output

COM3 (Arduino/Genuino Mega or Mega 2560):

```
initializing slave 1
initializing slave 2
requesting data from slave 1 of 2
□□d□p□transmission successful! data from slave 1:676
requesting data from slave 2 of 2
□□d□p&transmission successful! data from slave 2:112
All slaves read out succesfully. See you tomorrow!
```

Autoscroll [✓] Newline 4800 baud

COM4 (Arduino/Genuino Uno):

```
value of the sensor:
676
□□□□⸮⸮)□□□□⸮⸮
```

COM5 (Arduino/Genuino Uno):

```
⸮value of the sensor:
112
□□□p⸮⸮□□□p⸮
```

Figure 4-9: A screenshot of the final result considering the software; The left monitor (COM3) is connected to the master, the upper right monitor is connected to slave 1 and the lower right monitor is connected to slave 2

Discussion

This chapter discusses the side notes related to the achieved results and touches upon some unforeseen events during the project. Since the deliverables were divided between a communication module and a power supply the discussion will be divided in a similar way. However, aside from discussing the results of both systems, a general remark about the program of requirements will be made.

A lot of the requirements for the design, as given by the client and specified in the project requirements, are related to the design of the sensor specifically. For example the requirement that the sensor should be able to perform for every form and consistency of the grease. Although this is a major requirement towards the design of the sensor network, it has little to no influence on the design of the communication system. This is an example of one the requirements that is not discussed in this report during the design stage as described in chapter 3. Therefore are in this report only the requirements related to the communication module and power supply discussed.

5-1 Communication System

One of the first things to discuss are the results from the serial monitor of the Arduinos. This monitor shows the data that is send and thus the functioning of the communication protocol. Aside from the relevant info on the monitor there are also a couple strings of characters that seem to make no sense or are not supposed to be there. This is probably due to the way the Arduino serial monitor is built. When printing something on the monitor, it is also printed on the serial port used for communicating with the other nodes, and vice versa. This means that, when an Arduino receives or sends a message over the dataline, this message also shows up on the monitor in ASCII format. However, said message was never meant to show up on the monitor in the first place, let alone in ASCII form. This is shown for example in figure 4-8. This results in something that makes no sense on the monitor. Unfortunately there is no easy way around this, but it does not interfere with the validation of the results.

During the implementation of the communication module two different libraries were used: One to program the master and one for programming the slave. Although the libraries format their data similarly there are quite some differences between how a master and a slave react on received data. For example a master should check whether the data matches the format that it is expecting. This is determined by the request that is sent prior to receiving any data.

A slave on the other hand only has to check if a data package is addressed to his slave ID before doing anything with said package. Some applications only use an Arduino as master to control other MODBUS devices. Other applications use them as slaves, where they are controlled by an existing MODBUS master. There are hardly any MODBUS implementations where Arduinos are used both as master and as slave. Besides, Arduinos are catered towards people with little knowledge of or experience with these kind of systems. All these factors together result in the demand of high level, easy to use, either master- or slave Arduino libraries. As a result respective libraries could be found quite easily. However a library containing both master and slave functions did not show up. Eventhough it turned out to not be a problem in the final implementation difficulties surfaced when debugging. When an error occured it was not always clear at first whether this was the result of a mismatch between the two libraries or something else. Aside from the debugging it was also needed to understand two libraries instead of only one. Although if one library were to be used for both a slave and master implementation it would probably have been a bigger and more complex library than the used libraries. All these difficulties considered the use of both libraries still saved a lot of time compared to writing a new library, but being able to use a single library with both functionalities would have saved even more time.

At the start of the project the communication module was designed to have most data processing done in the main node. During the implementation of the sensor it became clear that controlling the sensor network was a much bigger task than expected, while processing the acquired data became easier due to choices made by the sensor subgroups. This meant a shift of required processing power from the main node towards the sensor nodes. This change should be taken into account when looking at micro-controllers for a finalised product.

5-2 Power Supply

The implementation of the two-way rail converter has somewhat different component values compared to the ones designed. This is due to error margins of the components itself or simply because of the fact that not all values were available either at the project hall or online. The values that would differ from the design were chosen conform the same reasoning behind the designed values. For example the resistors need to be of the same value for the right amplification, the inductance should be as small as possible, etc.

One of the components that needs more thought is the filter capacitor. This capacitor was designed to be 'as big as possible to reduce the ripple'. Although this statement still holds, having a bigger capacitor also means a larger discharge time. This should not be a problem within this application, as the capacitor never has to be discharged for the supply to work accordingly. However, having a lot of stored charge at the sensor node could be a possible safety hazard in combination with the grease. Therefore it is something to be considered during the product design.

Conclusion

During this project a proof of concept for a communication module and power supply were designed and implemented. The implementation of the communication system shows a network with two sensor nodes and a main node which could be extended to meet the eight sensor nodes requirement. The results show that the communication between the master and two slaves functions as required, both on a hardware level and on a software level. This means that the data is transported in a reliable and correct way. Furthermore, by letting the sensors send their data in turn their data will not interfere with each other. The systems data rate is sufficient to be able to communicate and collect data from eight sensors within a day ensuring that measuring the entire channel each day will be possible. The used hardware for the communication module is rated to work within the desired 1 to 100m range. To connect the main node to the internet a design has been made. Next the design to supply the capacitive sensor with power was motivated and could be used in a finalised product. Aside from the power supply for the capacitive sensor the supply for the acoustic senor has been designed. However, the implementation of the acoustic power supply and the connection to internet is unfinished due to time constraints and needs further work.

Recommendations

At the end of the project several recommendations can be made. These are recommendations concerning design choices for a product or how to improve the current design.

The current proof of concept consists of two end shields and a connection shield.The end shields have one connection to the bus and a line terminator, while the connection shield has two connections to the data bus. When more nodes are needed, the connection shield can be copied and placed in between the end nodes in order to preserve the data bus.

During this project there have not been any measurements at the location of measuring, i.e. the extraction channels. Although the design is made to operate in this environment experiments should be done to confirm this. It is highly possible that the nodes, especially the sensor nodes, will need a certain form of casing to ensure the right operation. This form of casting should be designed.

As said before, due to uncertainties during the design process it was decided to use Arduinos for the microcontrollers of the prototype. For a finalised product it would be advised to use microcontrollers optimised for the desired functions. The microcontrollers should be compatible with RS-485 in such a way that an IC between the controller and the data bus as used in this prototype becomes unnecessary. Furthermore, said microcontroller should be optimised for low-power consumption while keeping enough processing power to control the sensing circuit.

For the prototype existing libraries were used in order to have a quicker implementation of the communication protocol. It is likely, however, that there are no such libraries available for the microcontrollers that will be used in the final product. In this case such a library should be developed. It is advised to write one library that combines both the master and slave functions. A different option would be present if libraries for microcontrollers in combination with MODBUS-like protocols exist. In this case, it is highly recommendable to switch to such a protocol and use the already existing library.

During the project no printed circuit boards (PCB's) were designed for the shields. However it is advised to design a PCB for the slave and master nodes. In this way it becomes much easier to change the size of the network. Whether the power supply would fit onto the same PCB is undetermined and should be investigated when designing the product.

When designing the PCB for the slave nodes one thing should be kept in mind. That is, the PCB should also contain a slave ID next to implementing the current required hardware functionalities. At this moment the slave ID is set by the software, which results into eight different versions of software for each of the slaves. When trying to produce a product, having to use different versions of software for every slave is not efficient. Besides, the problem of which slave has which ID will arise, which is of high importance when assembling a system. This could lead to possible confusion. The slave ID could be engraved onto the PCB at production making it easier to ensure that when assembling a system no duplicate slave ID's are used.

During the design of the power supply it was unclear what the power requirements of the sensor circuits were; Only the required voltages and an estimate of the currents were known. In order to optimise the power supply, a clear analysis of the power requirements of the sensor network should be done. It also has to be taken into account that the micro-controllers should also be powered by this power supply. Right now the Arduinos are powered by the supply of laptops. A power design that also contains the microcontrollers should be made for the final product.

References

[1] *Vetkanaal.nl | de specialist in het reinigen van vetkanalen*, https://www.vetkanaal.nl/, Jun. 2018.

[2] Elcometer, *Mtg4 material thickness gauge*, http://www.elcometerndt.com/en/precision-thickness/ptg8-precision-thickness-gauge/ptg8-precision-thickness-gauge.html, Accessed on April 25th 2018.

[3] B.Kesil, D.Margulis, and E.Gershenzon, "Method and apparatus for measuring thickness of conductive films with the use of inductive and capacitive sensors", Granted Patent US6593738B2, Jul. 15, 2003. [Online]. Available: https://patentimages.storage.googleapis.com/d1/59/24/c2593f3976a979/US6593738.pdf.

[4] R. Igreja and C. Dias, "Analytical evaluation of the interdigital electrodes capacitance for a multi-layered structure", *Sensors and Actuators A: Physical*, vol. 112, no. 2, pp. 291–301, 2004, ISSN: 0924-4247. DOI: https://doi.org/10.1016/j.sna.2004.01.040. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0924424704000779.

[5] T. Yasui, T. Yasuda, K.-i. Sawanaka, and T. Araki, "Terahertz paintmeter for non-contact monitoring of thickness and drying progress in paint film", *Appl. Opt.*, vol. 44, no. 32, pp. 6849–6856, Nov. 2005. DOI: 10.1364/AO.44.006849. [Online]. Available: http://ao.osa.org/abstract.cfm?URI=ao-44-32-6849.

[6] P. D. J. McClements, "Analysis of lipids", *Department of Food Science*, section 5, 2003. [Online]. Available: http://people.umass.edu/~mcclemen/581Lipids.html.

[7] J. J. da Silva, A. M. N. Lima, F. H. Neff, and J. S. da Rocha Neto, "Non-invasive fast detection of internal fouling layers in tubes and ducts by acoustic vibration analysis", *IEEE Transactions on Instrumentation and Measurement*, vol. 58, no. 1, pp. 108–114, Jan. 2009, ISSN: 0018-9456. DOI: 10.1109/TIM.2008.927206.

[8] C. E. Brown and M. F. Fingas, "Development of airborne oil thickness measurements", *Marine Pollution Bulletin*, vol. 47, no. 9, pp. 485–492, 2003, ISSN: 0025-326X. DOI: https://doi.org/10.1016/S0025-326X(03)00203-0. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0025326X03002030.

[9] R. M. Bielemann, M. C. Gonzalez, T. G. Barbosa-Silva, S. P. Orlandi, M. O. Xavier, R. B. Bergmann, and M. C. F. Assuncao, "Estimation of body fat in adults using a portable a-mode ultrasound", *Nutrition*, vol. 32, no. 4, pp. 441–446, 2016, ISSN: 0899-9007. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0899900715004153.

[10] N. Ramesh, "Grease particle deposition measurements in a kitchen exhaust duct for the development of low cost grease sensors", 2010. [Online]. Available: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.834.6849&rep=rep1&type=pdf.

[11] L. A. V and S. D. W, "Duct grease deposit detection devices, systems, and methods", Granted Patent US 8487776 B2, Jul. 16, 2013. [Online]. Available: https://lens.org/061-952-189-984-841.

[12] *Kgs duct safety system*, Halton, Oct. 2017.

[13] K. M. Keener, J. Ducoste, and L. Holt, "Properties influencing fat, oil and grease deposit formation", *Ingenta connect*, vol. 80, pp. 2244–2245, 2008. DOI: https://doi.org/10.2175/193864708X267441. [Online]. Available: http://www.ingentaconnect.com/content/wef/wer/2008/00000080/00000012/art00007#expand/collapse.

[14] *Sn65176b, sn75176b differential bus transceivers datasheet*, Jun. 2018. [Online]. Available: http://www.produktinfo.conrad.com/datenblaetter/150000-174999/152828-da-01-en-SN_75176_BP.pdf.

[15] *Cat5e utp cable*, Jun. 2018. [Online]. Available: https://www.dlinkmea.com/upload/downloadable/7498-NCB-5EUBLUR-305.pdf.

[16] Modicon, "Modbus over serial line specification & implementation guide", Modicon, Tech. Rep., 2002.

[17] *Arduino uno datasheet*, Jun. 2018. [Online]. Available: https://www.arduino.cc/en/uploads/Main/Arduino%5C_Uno%5C_Rev3-schematic.pdf.

[18] *Lm340, lm340a and lm7805 family wide vin 1.5-a fixed voltage regulators datasheet*, Jun. 2018. [Online]. Available: http://www.ti.com/lit/ds/symlink/lm7800.pdf.

[19] *Maxim max634/max4391 cmos micropower inverting switching regulator datasheet*, Jun. 2018. [Online]. Available: https://docs-emea.rs-online.com/webdocs/12b1/0900766b812b1239.pdf.

[20] *Github- modbusmaster*, Jun. 2018. [Online]. Available: https://github.com/4-20ma/ModbusMaster.

[21] *Github- modbus-arduino*, Jun. 2018. [Online]. Available: https://github.com/andresarmento/modbus-arduino.

Appendix

A-1 Derivation of the Ripple Voltage

As noted in chapter 3, the ripple voltage is known as:

$$V_{dQ} = \frac{Q}{C} \tag{A-1}$$

where Q is the charge dissipated during the time where the capacitor delivers energy to the load.This energy is determined by:

$$Q = t_{off} * \frac{I_{peak}}{2} \tag{A-2}$$

Since I_{peak} is determined by the value of L_x, it can be derived from equation 3-1:

$$I_{peak} = \frac{T_{on} * V_{in}}{2} \tag{A-3}$$

Combining equations A-3 and A-2 into equation A-4 results in:

$$V_{dQ} = \frac{V_{in} * T_{on} * T_{off}}{2LC} \tag{A-4}$$

A-2 Arduino Code

Listing A.1: Code for first slave

```

#include <Modbus.h>
#include <ModbusSerial.h>
#include <ModbusMaster.h>


//Modbus Registers Offsets (0-9999)
const int SENSOR_IREG = 100;
//Used Pins
const int sensorPin = A0;
const int txPin = 2; //digital pin used for the drive enable
const int rxPin = 3; //digital pin used for the receive enable
byte* frame;


// ModbusSerial object
ModbusSerial mb;


void setup() {

    pinMode(LED_BUILTIN, OUTPUT);
    // Config Modbus Serial (port, speed, ,tx enable pin, rx enable pin)
    mb.config(&Serial, 4800, txPin, rxPin);
    // Set the Slave ID (1-247)
    mb.setSlaveId(1);

    // Add SENSOR_IREG register
    mb.addIreg(SENSOR_IREG);

    //print the value of the sensor (to be able to compare with the ...
        received value at the master)
    Serial.println("value of the sensor: ");
    Serial.println(analogRead(sensorPin), DEC);

}

void loop()
{
   //couple the sensor register to the pin of the sensor
   mb.Ireg(SENSOR_IREG, analogRead(sensorPin));

   //listen for a request, and respond accordingly
   mb.task();
}
```

Listing A.2: Code for second slave

```

#include <Modbus.h>
#include <ModbusSerial.h>
#include <ModbusMaster.h>


//Modbus Registers Offsets (0−9999)
const int SENSOR_IREG = 100;
//Used Pins
const int sensorPin = A0;
const int txPin = 2; //digital pin used for the drive enable
const int rxPin = 3; //digital pin used for the receive enable
byte∗ frame;


// ModbusSerial object
ModbusSerial mb;


void setup() {

    pinMode(LED_BUILTIN, OUTPUT);
    // Config Modbus Serial (port, speed, ,tx enable pin, rx enable pin)
    mb.config(&Serial, 4800, txPin, rxPin);
    // Set the Slave ID (1−247)
    mb.setSlaveId(2);

    // Add SENSOR_IREG register
    mb.addIreg(SENSOR_IREG);

    //print the value of the sensor (to be able to compare with the ...
        received value at the master)
    Serial.println("value of the sensor: ");
    Serial.println(analogRead(sensorPin), DEC);

}

void loop()
{
   //couple the sensor register to the pin of the sensor
   mb.Ireg(SENSOR_IREG, analogRead(sensorPin));

   //listen for a request, and respond accordingly
   mb.task();
}
```

Listing A.3: First test code for the master to communicate with one slave

```
#include <ModbusMaster.h>

// instantiate ModbusMaster object
ModbusMaster node1;

void setup()
{

  uint8_t result1, result2, j;
  uint16_t data1, data2;
  //initialize communication with the slave
  Serial.begin(4800);
  node1.begin( 1, Serial, 2, 3);


  // initialize digital pin LED_BUILTIN as an output; used for debugging
  pinMode(LED_BUILTIN, OUTPUT);


  // request data from one slave. during test phase this command was given ...
      in the setup part to make sure the request was only sent once, which ...
      was easier while debugging
  result1 = node1.readInputRegisters(0x0064, 1);

  //check if the request got a successful responce
  if (result1 == node1.ku8MBSuccess)
    {
        //if so, turn on the led (debugging)
        digitalWrite(LED_BUILTIN, HIGH);
        //acquire the data from the response of the slave
        data1 = node1.getResponseBuffer(0);
        //print this data to the serial monitor
        Serial.println();
        Serial.println("data from slave: ");
        Serial.println(data1, DEC);
  }
  else
  {
      //if the transmission was not succesful, print the error code as ...
          defined by the library.
      Serial.println();
      Serial.println(result1, HEX);
  }



}


void loop()
{
  //as stated before, the loop part of the program was not used during ...
      early debugging.
}
```

Listing A.4: Second test code for the master to communicate with two slaves consecutively

```
#include <ModbusMaster.h>

// instantiate ModbusMaster objects
ModbusMaster node1;
ModbusMaster node2;

void setup()
{

  uint8_t result1, result2, j;
  uint16_t data1, data2;
  //initialize communication with the slave
  Serial.begin(4800);
  node1.begin( 1, Serial, 2, 3);
  node2.begin( 2, Serial, 2, 3);


  // initialize digital pin LED_BUILTIN as an output; used for debugging
  pinMode(LED_BUILTIN, OUTPUT);


  // request data from one slave. during test phase this command was given ...
      in the setup part to make sure the request was only sent once, which ...
      was easier while debugging
  result1 = node1.readInputRegisters(0x0064, 1);

  //check if the request got a successful responce
  if (result1 == node1.ku8MBSuccess)
    {
        //if so, turn on the led (debugging)
        digitalWrite(LED_BUILTIN, HIGH);
        //acquire the data from the response of the slave
        data1 = node1.getResponseBuffer(0);
        //print this data to the serial monitor
        Serial.print("data from slave 1: ");
        Serial.println(data1, DEC);
  }
  else
  {
      //if the transmission was not succesful, print the error code as ...
          defined by the library.
      Serial.println();
      Serial.println(result1, HEX);
  }
  //wait a second and execute the exact same steps for slave 2
  delay(1000);

   // request data from one slave. during test phase this command was ...
       given in the setup part to make sure the request was only sent once, ...
       which was easier while debugging
  result2 = node2.readInputRegisters(0x0064, 1);

  //check if the request got a successful responce
  if (result2 == node2.ku8MBSuccess)
    {
        //if so, turn on the led (debugging)
        digitalWrite(LED_BUILTIN, HIGH);
        //acquire the data from the response of the slave
        data2 = node2.getResponseBuffer(0);
        //print this data to the serial monitor
        Serial.print("data from slave 2: ");
        Serial.println(data2, DEC);
  }
  else
  {
      //if the transmission was not succesful, print the error code as ...
         defined by the library.
      Serial.println();
      Serial.println(result2, HEX);
  }



}


void loop()
{
  //as stated before, the loop part of the program was not used during ...
      early debugging.
}
```

Listing A.5: Final code for the master can communicate with any number of slaves automatically once per day by just filling in the amount of slaves at the top

```
#include <ModbusMaster.h>
//specify the amount of nodes; maximum 8
int NodeCount = 2;

int j = 0;

//instantiate array of MobusMaster objects
ModbusMaster* nodes = new ModbusMaster[NodeCount];

void setup()
{
  //initialize communication with the slaves
  Serial.begin(4800);
  for (int i = 0; i < NodeCount; i++)
  {
    Serial.print("initializing slave ");
    Serial.println(i+1);
    nodes[i].begin(i+1, Serial, 2, 3);
  }
}


void loop()
{
  int result[NodeCount], data[NodeCount];
  int t = 0;
  int T = 0;

  // request data from slave i.
  Serial.print("requesting data from slave ");
  Serial.print(j+1, DEC);
  Serial.print(" of ");
  Serial.println(NodeCount);
  delay(1000);
  result[j] = nodes[j].readInputRegisters(0x0064, 1);

  //check if the request got a successful response
  if (result[j] == nodes[j].ku8MBSuccess)
  {
    //acquire the data from the response of the slave
    data[j] = nodes[j].getResponseBuffer(0);
    //print the value (to confirm results)
    Serial.print("transmission successful! data from slave ");
    Serial.print(j+1);
    Serial.print(":");
    Serial.println(data[j], DEC);
    //increment i to acquire data from the next slave during the next ...
        iteration. if the transmission was not succesfull, i will not be ...
        incremented and data from the same slave is requested
    j = j+1;
    //debug
    //Serial.print("incremented j; now going to request data from slave ");
    //Serial.println(j +1);
      //check if all slaves delivered their data

    //wait a certain time before requesting the next set of data
    delay(1000);

  }
  else
  {
    Serial.print("Transmission failed! Error code: ");
    Serial.println(result[j], HEX);
    delay(1000);
  }

  if (j + 1 > NodeCount)
    {
      j = 0;
      //send data to online platform
      //wait for the next time all data has to be gathered (next day)
      Serial.println("All slaves read out succesfully. See you tomorrow!");
      delay(1000000000);
    }


}
```