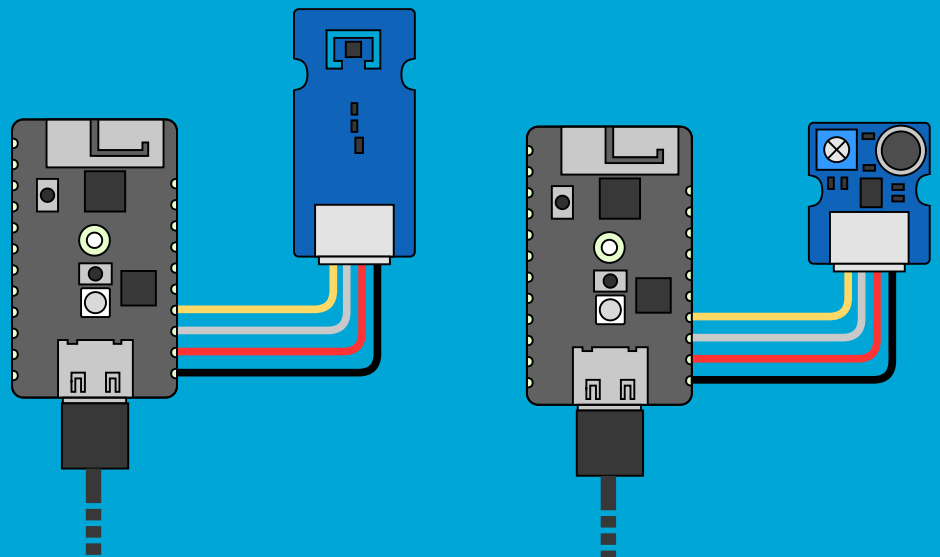
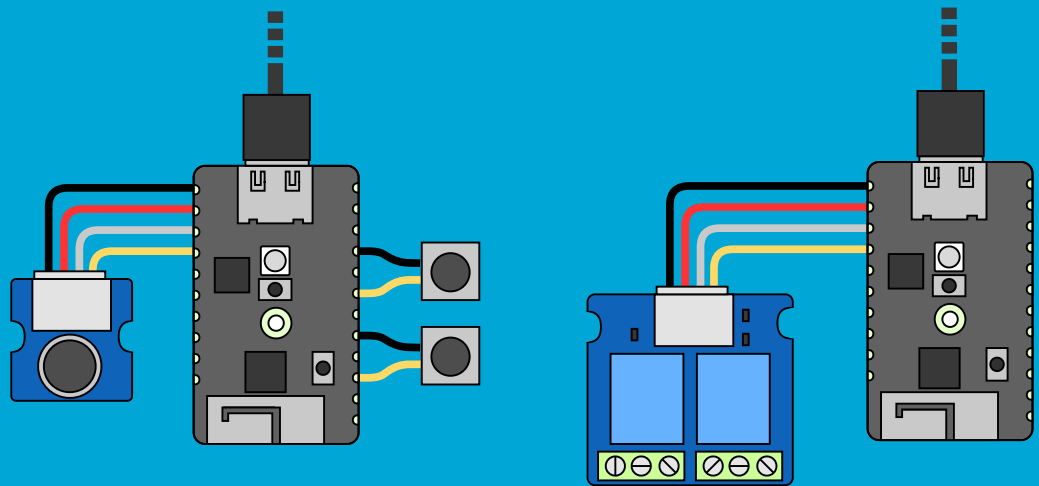


IoT-Based Smart Classroom

Subgroup: Hardware

R. Groenendijk

T. Goedegebuure



IoT-Based Smart Classroom

Subgroup: Hardware

by

R. Groenendijk
T. Goedegebuure

Student number:	4574567	Richard Groenendijk
Student number:	4893522	Tom Goedegebuure
Project duration:	April 11, 2022 – June 13, 2022	
Thesis committee:	Dr. B. Abdi,	TU Delft, supervisor
	Dr. J. Martinez,	TU Delft, supervisor

Abstract

In this work, we propose an IoT smart classroom framework which will be the base for smart classroom and hybrid teaching applications. The prototype introduces sensor readout capabilities, which can be used to monitor: temperature, humidity, and loudness in a classroom. Furthermore, the prototype introduces capabilities to collect and display measurements. Additionally, this prototype will assist an educator in a hybrid teaching setup by offering additional functionalities. With this prototype, the educator is able to control whether the students online are able to hear physically present students or not. The prototype also introduces a way to notify the educator of any questions from students attending online. The goal of this thesis is to make the first steps to improve the hybrid teaching environment by reducing the workload on the educator and by making it easier for online students to interact with the educator.

The project is divided into three submodules. Each submodule designs a different part of the total project. This thesis focuses on the hardware of the system. The hardware that is used for audio recording, audio controlling, question indicating and climate monitoring. The other two subgroups discuss IoT communication, data storing and graphical displaying of data.

Preface

This thesis describes the Bachelor Graduation Project of group C. The subject, IoT-based smart classroom, was given by TU Delft to be researched. This research is in relevance for future education for TU Delft, as for other educational instances. The designed system is a foundation for both a hybrid teaching system and a smart classroom system.

During the project we were guided by our supervisors dr. Jorge Martinez and dr. Bahareh Abdi. Therefore we would like to give special thanks to them. Secondly, we would like to thank dr.ing. Ioan Lager for coordinating The Bachelor Graduation Project. Finally, we are grateful to work with our colleagues: Mitchell Vuong, Bas Smeele, Ciarán Lichtenberg, and Mathijs Binnendijk. Together we showed cooperative teamwork to reach our goal during the project.

*R. Groenendijk
T. Goedegebuure
Delft, June 2022*

Contents

1	Introduction	1
1.1	Hybrid teaching issues and challenges	1
1.2	Smart classrooms issues and challenges	2
1.3	Problem definition	3
1.4	Product research	4
1.5	IoT-based smart classroom	5
1.6	Submodules of an IoT-based smart classroom	6
1.6.1	Hardware subgroup	6
1.6.2	Networking subgroup.	6
1.6.3	Server subgroup	6
1.7	Thesis structure.	6
2	Program of requirements	7
2.1	Main PoR takeaways.	7
2.2	General PoR	7
2.3	Hardware group PoR.	8
3	Hybrid education	9
3.1	Classroom audio system.	9
3.1.1	Design of the audio system	9
3.1.2	Implementation of the audio system.	12
3.1.3	Validation of the audio system	14
3.2	Notification System.	16
3.2.1	Design of the notification system	16
3.2.2	Implementation of the notification system.	18
3.2.3	Validation of the notification system	19
4	Smart classroom	20
4.1	Climate control	20
4.1.1	Temperature and humidity sensor	20
4.1.2	I ² C communication	21
4.1.3	Validation	23
4.2	Data collection of audience engagement	24
4.2.1	Loudness sensor	24
4.2.2	Validation	25
5	Discussion	27
6	Conclusion	29
A	Survey results	31
B	Temperature and humidity sensor code	34
C	Loudness sensor code	46
	Bibliography	51

Introduction

In the past two years, the Covid-19 virus had a major impact on the world and heavily altered various aspects of society. Specifically, the importance of remote solutions became apparent. In the education sector this resulted in either fully online courses through software such as Zoom or a mix of online and local courses (hybrid). Now that the pandemic has mostly subsided, lectures have returned to normal. However, the pandemic did show the potential of remote teaching solutions to accommodate the needs and preferences of students and educators. The combination of face-to-face lectures with remote teaching, called hybrid teaching, has remained after the pandemic has ended. Unfortunately, this can create a disconnect between the local classroom and the online environment, for which very few solutions are currently available.

Hybrid teaching is a functionality of a smart classroom application. Therefore researching the current trend of hybrid education as well as smart classrooms will form an important context for the design and development of a successful product. Since hybrid teaching is a larger focus point than other smart classroom applications, it will be discussed separately from smart classrooms. Section 1.1 discusses the issues and challenges of hybrid teaching. Section 1.2 discusses the issues and challenges of smart classrooms. These sections are followed by the general problem definition and research of existing products. Section 1.5 describes the solution that will be designed in this thesis. The design of this solution will be subdivided in three subgroups and discussed in 3 different theses as mentioned in section 1.6.

1.1. Hybrid teaching issues and challenges

To understand the problems teachers have using hybrid education and smart classroom technology, a survey was sent to all educators of the faculty of EEMCS at Delft University of Technology. The responses to the survey can be found in appendix A. For hybrid classrooms, a recurring problem was the interaction with the online students. Questions from online students are often missed as the attention of an educator is mostly on those physically present. They find it difficult to split their attention between both groups.

Another survey [1] showed that 62% of the educators feel that hybrid teaching should be adopted. Another 23% of educators want to adopt the hybrid teaching model but are untrained and unsure about it. This shows strong support for hybrid teaching models by educators and the importance of ease of use to convince the 23% that are still unsure. Another important result of the survey is that 73% of the students are willing to adopt the new model. 18% are willing but they don't have the proper resources or skills. This shows a strong support for hybrid teaching models by students as well.

This support for hybrid teaching is justified when looking at the pros [2]:

- **Ease of participation:** Students are able to participate both synchronous and asynchronous during courses allowing for great flexibility.
- **Choice in teaching format:** The ability to choose between classroom and e-learning possibilities allows students to choose their preferred learning style.

- **Cost effectiveness for the institution:** More students making use of hybrid education can reduce the opportunity costs for institutions as well as students. The institution can spend less on classroom space, utilities, and upkeep costs. Meanwhile, students can save on travel costs.

The hybrid teaching model does also have cons [2]:

- **Computer literacy:** Not every student is computer literate or has access to the resources needed for e-learning.
- **Course design difficulties:** Designing a course to effectively meet the needs of e-learning can be a sizeable challenge for the educator. Educators can also feel overwhelmed since they are responsible for more aspects such as technology, delays and dividing attention between online and face-to-face [3]. Teaching effectively while taking all these new aspects into account can introduce complexity.
- **Lacking face-to-face time:** Some students do best when physically present or need instructor face-to-face time to fully grasp a subject or to feel engaged in the learning process [4] [5]. Learning online is not the best environment for these students.
- **Teachers non-convinced by new teaching methods:** One of the interesting results based of the survey appendix A is that teachers are often non-convinced by new teaching techniques. One of the arguments presented is that the technology needed for hybrid teaching is not working properly. It also takes a lot of time and effort to understand the new technology. So this decreases the willingness to adapt to new teaching methods [6].

Regarding computer literacy, it is not much of an obstacle with the Netherlands ranking among the top EU countries in terms of digital skills according to the CBS [7]. This statistic could not be possible without overall access to the resources needed for e-learning, especially among young people.

Regarding course design difficulties, designing a course to effectively meet the needs of e-learning is difficult. This issue can be solved through education and experience. The division of students groups both online and offline can increase the amount of tasks teachers have to focus on during lecturing. *Care should always be taken to minimize the workload placed on teachers when hybrid technology is designed.*

Regarding face-to-face time, the fact that some students do better in physical based education can be taken into account within the education program. *It is important to allow for face-to-face teaching for those who prefer this style of education.*

And finally, regarding non-convinced teachers, it is understandable that not all teachers are convinced by hybrid teaching methods. It is important to take the stress and inconvenience introduced by hybrid teaching solutions into account, when they are not functioning as they are supposed to. This brings us to the last important takeaway, *The hybrid technology designed should be user friendly and intuitive.*

It can be concluded that, while hybrid teaching has its disadvantages, it can offer some significant benefits. This is why we would like to introduce a technological solution for a possible hybrid teaching application. This will be further elaborated on in section 1.5.

1.2. Smart classrooms issues and challenges

Within this project, smart classrooms are defined as classrooms equipped with technology providing some sort of benefit in an educational environment. This technology can help in hybrid education but also provide other uses in terms of security, environmental control, data collection, and visualisation. One interesting way of implementing such technologies is making use of the Internet of Things (IoT). IoT allows for an interconnected set of devices like sensors, actuators, and other technologies. All these devices are able to communicate with one another, thereby creating a wide range of possible functionalities. The following list gives some examples to get a general idea of the possibilities:

1. Security

- (a) Detecting persons in off-limit areas and notifying personnel.
- (b) Detecting emergencies like fires, and notifying authorities.
- (c) Wireless door locks.

2. Environmental control

- (a) Automated HVAC (heating, ventilation, and air-conditioning) control based on temperature and humidity measurements.
- (b) Controlling blinds based on light intensity.
- (c) Controlling lights based on occupancy.
- (d) Air quality measurement and control.
- (e) Automatically switch scenes when a different learning environment is required (like turning off lights when a projector is turned on).

3. Hybrid teaching

- (a) Giving educators visual feedback when there is a question online.
- (b) Automated recording of lecturers and students for online reviewing.

4. Data collection and visualization

- (a) Attendance tracking of students (for example by using student cards).
- (b) Using sound level measurements and machine vision to measure whether students are still paying attention.

IoT systems within education have a lot of advantages: reduction in cost, enhancement of comfort, saving time, enhancement in safety, exploring personalised learning, and increasing student collaboration [1] [8]. The need for specific applications will differ between institutions. Thus, *an IoT system should be easily customizable to fit the needs of differing institutions.*

IoT based systems also create a number of challenges. One difficulty is that teachers have problems with using the technology or technology is not working all the time. According to our survey A educators want smart teaching solutions to work automatically without having to set it up. As classrooms are used by multiple teachers and with multiple groups of students, any smart solution should be plug and play.

Furthermore, researchers agree that applying IoT in education could create a challenge in terms of security and privacy [1] [9]. This is because of an increase in data collection on things like credentials, location, and learning history of students. This is why *security and privacy should be taken into account when designing an IoT system.*

Another obstacle in implementing IoT technology in a classroom setting is cost [9]. Educational institutions could struggle with the budget for implementing a smart classroom IoT system. This is why *the cost should be minimized when designing an IoT system.*

1.3. Problem definition

In the sections above, the advantages and disadvantages of both hybrid teaching and smart classrooms are discussed. After this assessment of the challenges educators face, the problem that will be tackled in this project is defined. In this project a framework will be designed for a smart classroom using IoT solutions. Within this framework a useful application for hybrid teaching will be integrated in the system. The complete problem definition can be stated as followed:

How to design a complete and expandable IoT system that implements a smart classroom framework and improves hybrid education?

This problem encompasses two focus points. The first being the design of a complete and expandable IoT system for a smart classroom. The problem herein solved is the difficulty of using smart solutions

in a classroom. The second focus point is improving hybrid education. The main difficulty lecturers face is the interaction between online and physical audiences and not being able to focus on both (appendix A).

1.4. Product research

Hybrid teaching is a concept that has existed for a couple of years, during which multiple hybrid teaching systems have already been designed. However, no system has been able to resolve all identified issues in section 1.3. Some notable companies who have developed hybrid teaching systems are Logitech, Aver, Paramtech, and ViewSonic.

Logitech has designed a conferencing system that could be used for hybrid teaching. The system consists of at most one ultra high definition camera, two speakers, and seven microphone hubs. Since the system is not expandable, it can only be used in relatively small classrooms [10].

Aver also designs systems for conferencing purposes. This system has a person tracking camera, microphone range of 10 meters, and a single loudspeaker. This product is mainly used for conferencing since audio and video range is limited [11].

Paramtech is a company that creates solutions for hybrid teaching applications. Their hybrid teaching system is expandable up to one camera, twenty microphones, and two loudspeakers. They also offer a virtual microphone system that has two microphone bars. The audio range is for spaces up to 9.14 m by 15.24 m. The presenter in the room can walk freely because the camera has a tracking system, and the audio bars cover the whole room. But this system is applicable for a classroom specific. It can not be expanded to larger rooms as stated above. So the system cannot be used in a lecture room [12].

ViewSonic is developing a hybrid teaching system that focuses on both physical and online student engagement. The online students are projected on the wall and the physical students are audio and video recorded. At this moment ViewSonic designs systems for a standard classroom application since the amount of audio/video modules is limited. Due to the operational range of this equipment the product cannot be used for lecture halls [13].

Similar to hybrid teaching, the concept of smart classrooms is not new. With the advancements in technology, smart classroom tools have slowly crept into the education sector. For example, almost all classrooms are equipped with a projector, interactive whiteboard, or both. On another front, digital tools, such as the quizzing tool Kahoot!¹, are also being used more often in lectures. Additionally, uploading lecture slides, lecture recordings, and other learning resources through software such as Brightspace or Blackboard, has become the norm. More recent advancements are products such as ScanMarker², a scanner shaped like a marker which can read text and write it to a digital text editor. Alternatively, companies such as Magicard³ develop ID cards which can track student attendance, improve security through authentication and access control, or help regulate services such as printers or study rooms.

The future of smart classrooms looks promising, but still has a long way to go. Smart classroom products can be expensive, require extensive instructions, or be unreliable according to the responses of the survey in appendix A. All of this combined makes the education sector slow to adopt new technologies and educators hesitant to integrate new tools into their lectures (appendix A). Additionally, most tools work to enhance the already existing classroom environment and rarely try to connect the physical classroom with a remote teaching environment.

So there exist companies that offer hybrid teaching solutions. But these solutions are designed for small conferencing rooms or classrooms. A solution particular for a lecture room is not yet on the market. Secondly there are solutions for hybrid teaching and for smart education. But until now there does not exist a system that covers both.

¹Kahoot! is a game-based learning platform, used as educational technology in schools and other educational institutions. For more information: <https://kahoot.it/>

²For more information: <https://scanmarker.com/>

³For more information: <https://magicard.com/>

1.5. IoT-based smart classroom

When looking at existing products, see section 1.4, it can be observed that they all are quite limited in terms of functionality and/or expandability. The proposed product will address this problem, as well as the problems mentioned in section 1.3.

The core of the product is an expandable IoT network which will communicate between devices and a server. The IoT product will assist educators with hybrid teaching as well as offering smart classroom features. The smart classroom features include measuring the humidity, temperature, and loudness in a classroom to monitor the learning environment. For the hybrid teaching, a prototype will be created to make interacting in the lecture easier in a hybrid education setting for both the students and educators. See fig. 1.1 for the setup.

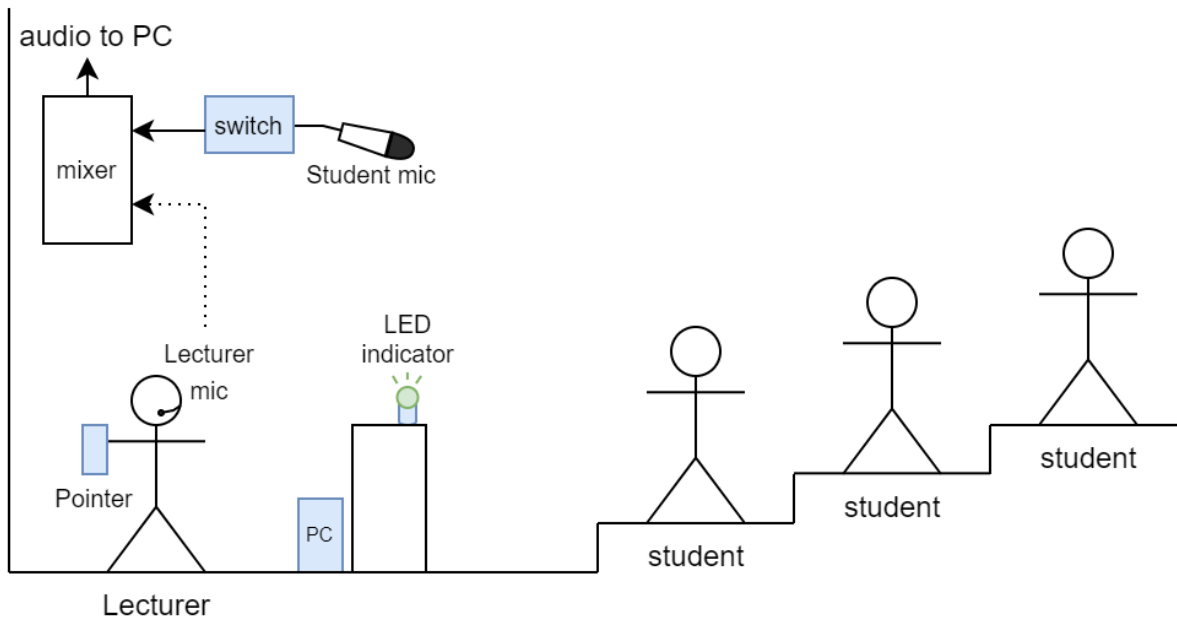


Figure 1.1: Hybrid teaching product setup

The important elements of the proposed product are: a computer running the online meeting software, an LED indicator, a laser pointer, and a student microphone. The computer running the online meeting software will have a program/app running on it, that will see if students online have a question. In turn, the computer will send a signal to indicate that a student has a question, this will light up an LED visible by the teacher. When the lecturer sees the LED indicator change color, they can then choose to unmute the online student(s) with a device attached to a laser pointer so everyone can hear the question in the class. This laser pointer can also be used to turn on the student microphone so that the students online will be able to hear the questions from the students attending the lecture in person. With this, the lecturer would not have to look for the online questions or need to repeat any questions, making the flow of the lecture smoother and decreases the work load on the lecturer.

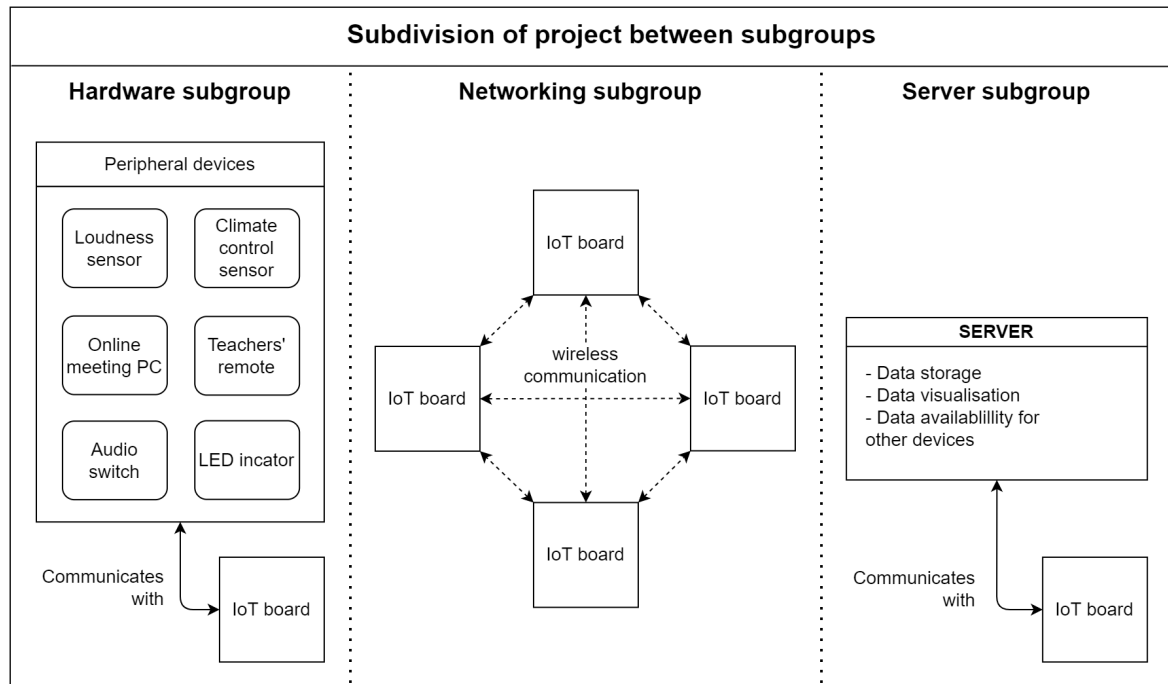


Figure 1.2: The subdivision between the subgroups

1.6. Submodules of an IoT-based smart classroom

The project is divided into three subgroups: hardware, network and server subgroup.

1.6.1. Hardware subgroup

The hardware subgroup designs and implements the peripheral devices needed for the system. These devices send include environmental sensors, audio equipment and actuators. The hardware devices communicate their data to an IoT development board.

1.6.2. Networking subgroup

The network subgroup is responsible for the wireless communication between IoT devices. The hardware subgroup will communicate their data with an IoT development board. The networking group will take this data and send it wirelessly between IoT boards. This means the networking group will have to look for a suitable wireless technology. Once a suitable wireless technology is found this technology must be implemented to fulfil the requirements of the prototype.

1.6.3. Server subgroup

The server subgroup designs a server which will store the acquired sensor data that will be received from the network and also provides the data that is requested to external and internal devices. This means that the server group will need to decide on a hardware platform to create the prototype on, on the software implementation of the server, and how the requirements of the prototype will be implemented.

1.7. Thesis structure

The thesis is structured as follows. In chapter 2 the program of requirements is discussed. In chapter 3 the design and implementation of the hybrid teaching part of the project are discussed. In chapter 4 the design and implementation of the smart classroom part of the project are discussed. After that, in chapter 5 the results and evaluation of the program of requirements are discussed. Finally, chapter 6 gives a conclusion together with future work.

2

Program of requirements

In this chapter, the requirements will be set, in order to develop our total IoT product. First, the main takeaways will be discussed, followed by the Program of Requirements (PoR) of the product, and then the PoR of the subgroup will be defined.

2.1. Main PoR takeaways

The context research in chapter 1 yielded some important points when making the PoR:

- TA1** Care should be taken to minimize the workload placed on teachers when hybrid technology is designed.
- TA2** It is important to allow for face-to-face teaching for those who prefer this style of education.
- TA3** The hybrid technology designed should be user friendly and intuitive.
- TA4** An IoT system should be easily customizable to fit the needs of differing institutions.
- TA5** Security and privacy should be taken into account when designing an IoT system.
- TA6** The cost should be minimized when designing an IoT system.

2.2. General PoR

The mandatory requirements, which were chosen in order to create a good working product, can be found below. The Trade-off requirements are chosen to improve the product as well. The boundary conditions are chosen to determine what will be left outside the scope of the project, and what will be included. The numbers following some of the requirements are references to the main PoR takeaways discussed in section 2.1.

Mandatory requirements

General

- M1** The collected data must be readable through a GUI (Graphical User Interface). [**TA1**, **TA3**]
- M2** The collected data must be available for external devices. [**TA4**]
- M3** The system must be able to log and save data.

IoT

- M4** The network must be able to accept new IoT devices while in operation. [**TA3**]
- M5** IoT devices must be able to send and receive data via a wireless network.
- M6** Sensors for the system must be applicable for a classroom.
- M7** The coverage of the network must be expandable. [**TA4**]
- M8** Losing an IoT device should not completely stop the operation of the network. [**TA3**]

Hybrid

- M9** The teacher must be notified of any questions from online students. [**TA1**, **TA3**]
- M10** The teacher must be able to switch between mute and unmute of the audience microphone and the students online. [**TA1**, **TA2**]
- M11** The online students must be able to hear the physically attending students. [**TA2**]

Trade-off requirements

- T1** The system should be affordable. [TA6]
- T2** The system should be customizable. [TA4]
- T3** The system should be able to be operated without extensive technical knowledge. [TA1, TA3]
- T4** The system hardware should have a small form factor.
- T5** The system should have basic event-based decision-making capabilities.

Boundary conditions

- B1** The security of the system will not be a main focus. [TA5]
- B2** The privacy of the system will not be a main focus. [TA5]
- B3** Only IoT devices applicable to a classroom will be discussed.

While security and privacy are important considerations, this topic deserves its own thesis. Therefore they are taken into consideration, but within the allotted time of this project, further research and design into this area is left outside of the scope of this thesis.

2.3. Hardware group PoR

The requirements above are relevant for the total system and therefore not all requirements are relevant for the hardware group. The requirements that are listed below are specific for the hardware subgroup. Mandatory requirements **M6**, **M9**, **M10**, and **M11** are applicable to the hardware group. In addition the trade-off requirements **T1**, **T2**, **T3**, and **T4** are all relevant. Finally, the following additional trade-off requirement and boundary conditions are specific for the hardware group.

Trade-off requirements

- HT1** Sensors and actuators should preferably communicate with the IoT device using a grove connector.

Boundary conditions

- HB1** Devices will not be battery powered, so power consumption will not be a main focus. [TA6]
- HB2** Already existing drivers for sensors will be used when available. [TA4]

3

Hybrid education

Technology can be the solution to improve hybrid education. This chapter discusses the design and implementation of two systems that will help improving hybrid education. The first system is a classroom audio system which is discussed in section 3.1. This system is able to record student voices in a room. The recorded audio can be broadcast to the students that are following the lessons from home. The purpose of this system is that students at home do not miss any information or questions that is mentioned by the audience during the lesson. The second system is a notification system which is discussed in section 3.2. This system is needed because teachers are unaware of questions that online students have. Therefore a notification system that recognizes a question and make the teacher aware of this question could be a solution.

3.1. Classroom audio system

According to the survey in appendix A online students have limited engagement in the physical lecture because the focus is on the students that are physically present. This problem is also mentioned in a feedback survey from King's college in London [3]. This survey contains the feedback of teachers and students after 104 hybrid sessions during the pandemic. One of the seven problems that was encountered is that online students do not feel engaged. Another problem is that the online students feel ignored during the lessons. Therefore, to improve hybrid education, these problems must be solved.

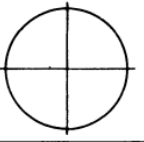
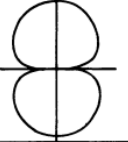
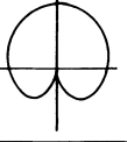
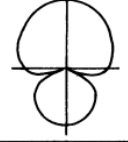
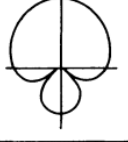
At this moment the audio of the teacher is recorded so that the online students are able to hear the teacher. But if a student in the classroom is talking or asking a question the online students do not hear this. So the online students are missing educational content and it therefore decreases their engagement compared to the physical students. This problem can be solved when there is a system that can record the audio of the physical students: a classroom audio system. This section describes the design and implementation of an audio system.

3.1.1. Design of the audio system

The sound of the students can be recorded with a microphone, or possibly multiple microphones. This microphone system must be able to record voices from everywhere a student can be present in that particular room. This room could be a standard classroom, but this could also be a lecture hall. The four requirements for this system are:

1. The audio system must be able to record in a lecture hall.
2. The audio system must be able to record the audience that is physically present.
3. The online students must be able to hear the recording of the audience.
4. The system should be as simple as possible.

There are several types of microphones such as omnidirectional, bidirectional, cardioid, hypercardioid and super-cardioid microphones. The characteristics of these are given in figure 3.1. The polar response gives the gain for the incoming sound wave for all angles. This gain is relative to an omnidirectional microphone. An omnidirectional microphone is a microphone that receives the audio strength in all directions.

CHARACTERISTIC	OMNIDIRECTIONAL	BIDIRECTIONAL	CARDIOID	HYPERCARDIOID	SUPER-CARDIOID
POLAR RESPONSE PATTERN					
POLAR EQUATION $F(\theta) \propto$	1	$\cos \theta$	$1/2(1+\cos \theta)$	$1/4(1+3\cos \theta)$	$.37+.63\cos \theta$
PICKUP ARC 3 dB DOWN (θ_3)	360°	90°	131°	105°	115°
PICKUP ARC 6 dB DOWN (θ_6)	360°	120°	180°	141°	156°
RELATIVE OUTPUT AT 90° (dB)	0	-∞	-6	-12	-8.6
RELATIVE OUTPUT AT 180° (dB)	0	0	-∞	-6	-11.7
ANGLE AT WHICH OUTPUT = 0 (θ_0)	—	90°	180°	110°	126°
RANDOM ENERGY EFFICIENCY (RE)	1 0 dB	.333 -4.8 dB	.333 -4.8 dB	.250 -6.0 dB	.268 -5.7 dB
DISTANCE FACTOR (DSF)	1	1.7	1.7	2	1.9

① MINIMUM RANDOM ENERGY EFFICIENCY FOR A FIRST ORDER CARDIOID

② MAXIMUM FRONT TO TOTAL RANDOM ENERGY EFFICIENCY FOR A FIRST ORDER CARDIOID

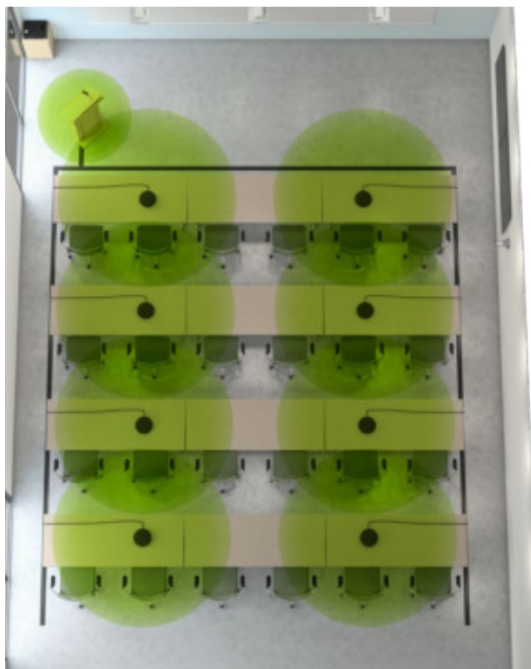
Figure 3.1: Polar responses and characteristics of different types of microphones [14]

Another characteristic of a microphone is the distance factor (DSF). This factor is the maximal gain if the microphone is pointed in the right direction. This means that a microphone with DSF of 2 can be placed 2 times farther than a microphone with DSF of 1 to receive the same audio strength. The bigger the DSF the bigger the directivity as seen in equation 3.1. Directivity is a measure which describes the sensitivity of the microphone in a single direction. So a microphone with a large directivity means that the microphone has a high gain >1 for particular angles, and a small gain <1 for all the other angles.

$$Directivity = 20 \log_{10}(DSF) \quad (3.1)$$

To conclude this there is a trade-off between the distance factor and the omnidirectionality. So you could have a microphone that is able to capture audio at long distances, but the microphone is very directional. Or you could have microphone that is able to capture audio at a wide spectrum of angles but this microphone has to be close to the audio source.

Looking at the literature, different possible solutions can be implemented. Parmetech [12] proposes two kinds of solutions for recording audio in larger rooms. The first solution is adding multiple microphones across the classroom so that they cover the whole classroom as seen in figure 3.2a. These microphones are connected via a mixer to a computer. The second solution is in cooperation with another company Nureva [15]. This company produces microphone bars that have 12 microphones integrated in. An example is shown in figure 3.2b. With the help of advanced signal processing, the



(a) Multiple microphone audio system from Parmetech [12]



(b) At the top the microphone bar from Nureva [15]

Figure 3.2: The two audio recording systems from Parmetech

bar is able to record all the audio in a room. To do so 25,000 million instructions per second (MIPS) need to be processed. This shows the complexity of the algorithms used.

Dynamount is a company that designs motorized microphone stands [16]. With a motorized stand, shown in figure 3.3, a microphone can be pointed in a desired direction. Their application is mainly sports like football or golf. Because they install a microphone with a large distance factor on the stand. With this stand they can record audio at long distances at dynamic places. This technology can also be used in a lecture hall. When a physical student has a question, the microphone will be controlled in the direction where the student is.



Figure 3.3: The motorized microphone stand including shotgun microphone from Dynamount [16].

So there are three different solutions to record the audio in a large classroom or lecture hall. The solution with the Nureva system is complex since the audio signal processing is complex and the range is only limited to spaces up to 9.14 m by 15.24 m [12]. thus this technology can not be used in large lecture rooms. The solution by Dynamount will introduce some main issues such as locating the student and rotating towards them. Determining the location of the student and pointing the microphone in the right direction take considerable time. Both due to the processing and the horizontal angular velocity of $120^\circ/\text{s}$ and vertical angular velocity of $30^\circ/\text{s}$. Therefore a small direction change could take hundreds of milliseconds. This means that the first few words of a question may be missed and not recorded since the microphone is not yet pointed in the right direction. So students at home will miss information. It can be concluded that the first solution with the multiple microphones would be the best option for our design.

Designing a microphone with multiple microphones is complex. Each room has its own acoustics and therefore its own setup with microphones. So the research for this topic will be classroom specific and not generalizable. However, a smaller audio system that represents the audio system will be discussed for the prototype in section 3.1.2.

3.1.2. Implementation of the audio system

This section describes the implementation of a prototype classroom audio system. The audio system that was mentioned in the design section 3.1.1 is not achievable within the scope of this project. Therefore a prototype will be implemented that corresponds with the design. The prototype consists of a relay, microphone, pushbutton, 2 Bluetooth modules and an audio system. A schematic of this prototype is shown in figure 3.4.

When there is a question in the lecture hall, the teacher will press a push button. This pushbutton together with the Bluetooth module is implemented in the teacher remote. The teacher remote is a device that teachers use to show next/previous slide. Thus an extra button can be added for the audio system. When this button is pressed the Bluetooth module will communicate that to the other Bluetooth module. This Bluetooth module is wired to the relay and can control this relay. The relay is normally open which causes an open circuit between the microphone and the audio system. As long as the button is pushed the relay and the audience microphone will be active.

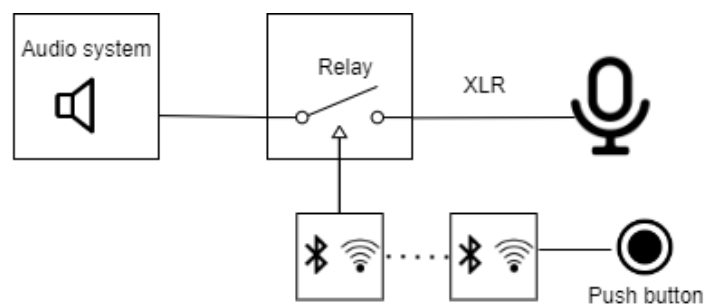


Figure 3.4: Schematic of classroom audio system prototype.

Microphone The microphone that is used for the prototype is a Røde NT1-A microphone. The microphone has a cardioid shape as seen in figure 3.5. This shape is useful for a classroom application when the microphone is placed in front of the students. This is because the microphone does not have to record sound from behind and only from the front, which a cardioid microphone can do. Furthermore the microphone has a frequency range of 20Hz-20.000Hz, suited for human speech. The microphone has a signal-to-noise ratio (SNR) of 87dB [17]. In comparison with other microphones this SNR is sufficient, and the signal is understandable with less noise [18].

Relay To switch the microphone on and off a relay is used. This relay is implemented in the audio signal path between the microphone and the audio system as showed in figure 3.4. The output of the microphone is XLR, which has 3 buses integrated in. Those 3 buses are: ground, positive and negative. So to toggle the microphone the positive and negative bus should be connected to the relay. Thus the relay must have 2 inputs. Switching of the relay will be toggled by the M5Stamp C3 board.

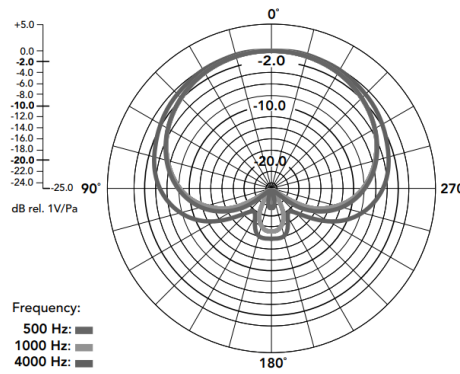


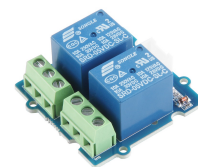
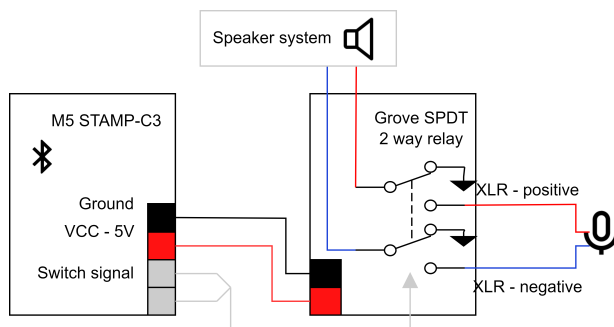
Figure 3.5: Polar plot of the NT1-A microphoned [17]

This board has a maximum of 5V output and the maximum output current is 40mA [19]. So this means that the relay should be able to operate at a voltage of 5V. The audio signal in the XLR cable has an 48V offset, this is called phantom voltage. This voltage is used for the microphone to operate since the microphone is an active object. The maximum voltage in the XLR line is 48V plus the audio voltage which is approximately a couple of volts [20]. The current that is travelling inside the XLR line are approximately a couple of milliamps [20]. We can conclude from this that the relay should be able to switch signals of around 50V and 10mA. The list beneath concludes all the requirements the relay should have.

Relay design requirements

1. Relay should switch signal voltages of 50V.
2. Relay should switch signal currents of 10mA.
3. Relay should operate at a voltage of 5V.
4. Relay should be normally open circuit, only activated when button is pressed.
5. Relay should have 2 inputs, for positive XLR and negative XLR.

According to the design requirements the Grove - 2-Channel SPDT Relay is used for the prototype. Figure 3.6b shows this relay. The relay meets all the requirements that are listed above.



(a) Implementation of the 2-way SPDT relay

(b) Grove 2-Channel SPDT Relay [21]

Figure 3.6

Relay specifications

1. Maximum allowable voltage is 110V DC.
2. Maximum allowable current is 10A DC.
3. Operating voltage of device is 5V DC.
4. Relay can be normally open.
5. 2 relays on one PCB and therefore 2 inputs.

Figure 3.6a shows the implementation of the relay between the audio system and the microphone. The two XLR lines are switched with this relay. When the relay is not activated, the audio system is connected to ground. So the audio system has a zero input. When the relay is activated, the microphone is directly connected to the audio system. The M5 Stamp-C3 board controls the relay with the signal connection. The pins on the board can be configured as output and set when the relay should be activated.

Button The push-button will be used to set the microphone on and off. This button is implemented in the teacher remote. The M5Stamp C3 board will also be implemented in the remote. When the button is pushed the M5Stamp C3 board will notice this via an input. The board will send this information via Bluetooth to the board that is connected to the relay. A schematic of the teacher remote is shown in figure 3.7.

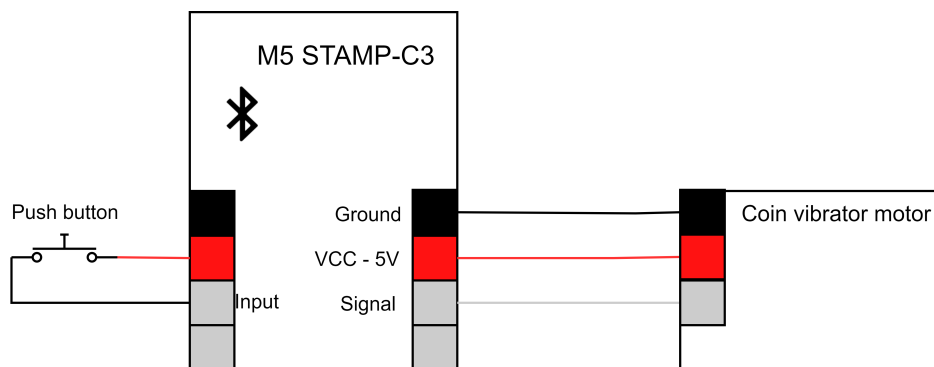


Figure 3.7: Schematic of the teacher remote including coin vibrating motor and push button.

3.1.3. Validation of the audio system

This section describes the validation of the audio system. The validation is performed using two setups. The first setup is including a relay, the second setup is excluding a relay and used as reference.

Setup including relay

The Røde NT1-A microphone is connected to the relay as in figure 3.6a. To do so the XLR cable is cut in half and the positive and negative lines are connected to the relay. The output (COM) of the relay is connected to the audio interface. This is the interface between the microphone and the computer. The Focusrite Scarlett 2i2 [22] is used as audio interface. This module also provides 48V phantom power to the microphone. The Scarlett 2i2 can be connected to a computer with USB and the microphone can be used as input for the computer. With the software: Audacity [23] audio can be recorded and analysed. The recordings can be used to validate the system.

Setup excluding relay

For this setup the microphone is directly connected to the Scarlett 2i2. So there is no relay between the audio path.

Audio recording using the two setups

Two audio recordings have been taken to validate the setup with the relay. A sentence is spoken 30cm from the microphones. The following sentence is said: 'Hello my name is Tom and I have a

question regarding the polar response of the microphone.’ This sentence is recorded with the two setups simultaneously.

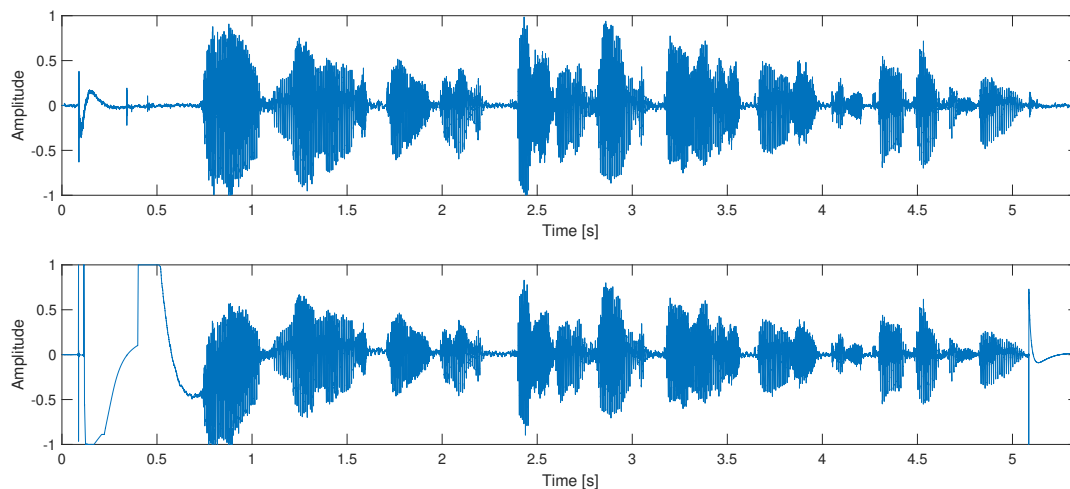
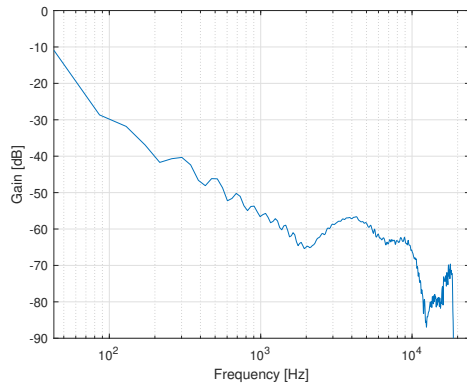


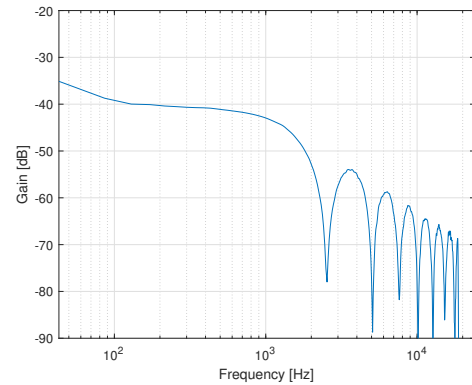
Figure 3.8: The upper graph shows audio that is recorded without the relay. The lower graph shows audio that is recorded with relay.

Figure 3.8 shows the recordings. The upper graph is the recording of the setup without relay, the lower graph is the recording with relay. The audio looks similar except for the first 0.7s and at 5s. The high amplitude signal at these points is caused by the relay due to its switching. The microphone will immediately be connected to the 48V phantom power which causes the maximum amplitude. To determine the power of this spike the spike can be analysed using the Audacity software. The frequency response is shown in figure 3.9. The power of the low frequencies in this spike is approximately 15dB higher than the power of the root notes of the actual audio which is shown in figure 3.10b. 15dB corresponds with 31x more power in the signal. When this audio gets broadcast to the online students they will hear a loud low frequency sound through their audio system, which is inconvenient. This should be avoided but a solution for this is not yet implemented.

Figure 3.10 shows the frequency response of the actual sentence. So the audio which is caused when switching is not selected for this plot. When comparing the frequency response of the audio with and without a relay, it can be seen that the relay has no significant influence on the power of the system. Because the frequency response has approximately the same power for all frequencies except for frequency's above 15,000Hz. The polar response shows a stronger decline in audio power with relay than without for frequencies larger than 15.000Hz. But this effect has minor effect for this application because these frequency's do not contribute to the intelligibility of the audio [24]. It can be concluded from these observations once the relay is switched that it has minimal impact on the audio signal and therefore it is useful for the design.

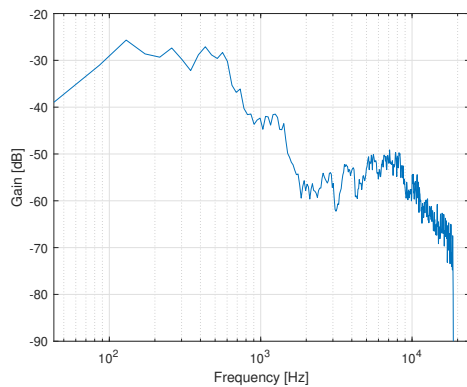


(a) Frequency response of the spike in the first 0.7s

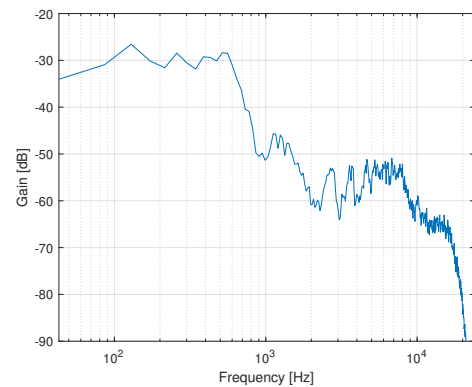


(b) Frequency response of the spike at 5s

Figure 3.9



(a) Frequency response of the recorded audio with relay implemented.



(b) Frequency response of the recorded audio without relay implemented.

Figure 3.10

3.2. Notification System

According to the survey in appendix A, teachers find it difficult to keep track of questions both in the classroom as online. So when an online student has a question, the teacher is not always aware of this question. Therefore the online students do not feel engaged with the lessons and eventually feel ignored [appendix A] [3]. So to improve hybrid education this problem must be solved.

Video communicating software such as: Microsoft Teams and Zoom, is used for online education. When online students have a question, they can raise a hand in the software or just type the question in the chat. Since it is difficult for teachers to keep track of these questions they should be notified about online questions. This section describes the design and implementation of this notification system.

3.2.1. Design of the notification system

To determine whether there is an online question a system has to be designed. This system must be capable of recognizing a question in the video communicating software. This recognizing system will be a software implementation and therefore not in the scope of this thesis. However the system also should have a notifier that alerts the teacher when there is a question. This notifier will be a hardware solution and therefore in the scope of this thesis.

To get someone's attention at least one of our five senses must be triggered. The five senses a human has are: seeing, hearing, tasting, touching, and smelling. So the notification system should at least trigger one of these five. Not all the ways to get someone's attention are applicable in a classroom. Listed below are the design requirements of the notification system.

1. The notification system should disturb the lesson as little as possible.
2. The notification system should be targeted to the teacher instead of the students
3. The notification system should be triggered via the M5Stamp-C3 board.
4. The notification system should be simple.

Not all possible solutions meet the requirements of this project. An alarm or other sound for example will disturb the lectures. Some system that triggers smell or taste will be too complex. So the two remaining senses than could be used as trigger for this application are touching and seeing. The notification system design will therefore be subdivided into two systems. One system that triggers the sense touching: Vibrating indicator. Another system that triggers the sense seeing: Led indicator.

Vibrating indicator An indicator that triggers human sensory cells should be minimal but enough to notify the teacher. The mobile phone industry uses such a system: a vibrating module. Such a module is not complex, it only targets the teacher, can be triggered via the Bluetooth mesh network and does not disturb the lesson.

A vibrating module is a module that uses a vibration motor. There are two different types of vibrating motors: eccentric rotating mass and linear resonant vibration motors [25].

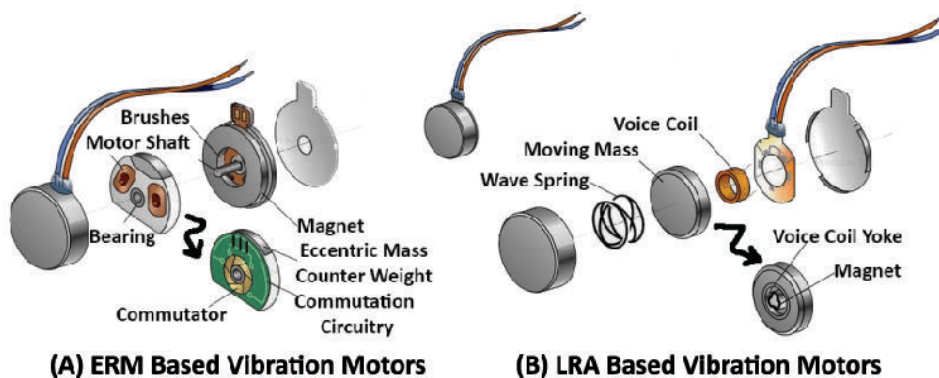


Figure 3.11: Two different types of vibration motors ERM and LRA based [25]

ERM based motors use a DC motor to rotate an eccentric mass around an axis. The mass is not symmetric with respect to its axis of rotation, so it causes the device to vibrate when rotating. The mass and other parts of the ERM based motor can be seen in figure 3.11. Both the amplitude and frequency of vibration depend on the rotational speed of the motor, which can in turn be controlled through an input DC voltage. With increasing input voltages, both amplitude and frequency increase almost linearly.

In LRA based motors, vibration is generated by the linear movement of a magnetic mass suspended near the voice coil as seen in figure 3.11. When an AC current is applied to the motor, the coil behaves like a magnet (due to the generated electromagnetic field) and causes the mass to be attracted or repelled, depending on the direction of the current. This generates vibration at the same frequency as the input AC signal, while the amplitude of vibration is dictated by the signal's peak-to-peak voltage. Thus, LRAs offer control on both the magnitude and frequency of vibration [25].

LRA vibration motors can operate at voltages of approximately 3V AC and ERM motors at voltages approximately 3V DC [26]. Since the M5Stamp C3 microcontroller has a 3V DC output it will be convenient to use the ERM motor instead of the LRA motor.

Led indicator Besides using the sense touching, the sense seeing could also be triggered. An indicator that triggers the eye should also be designed according to requirements listed in 3.2.1. Such a system is globally used: light indication. Think of traffic lights, flashing lights on police cars and much more applications. The light indication can also be used as notifier for a question at home.

There are a lot of different types of lights. For our application the most useful kind of light is an RGB led. This led does not disturb the lesson since it is not disturbingly bright. Secondly the led could only be targeted to the teacher. Thirdly the application of an led is not complex in practice. Lastly the led could be turned on and off with the M5Stamp C3 board. Because a typical RGB led has an input voltage of <5V, which is available on the board. When using an RGB led another useful functionality can be used. Because an RGB led can be set to all different colours. So besides indicating that there is a question other colours can be used for useful indications.

1. Question at home - yellow
2. Bad connection - red blinking
3. Audience microphone on - blue
4. On air - Green

3.2.2. Implementation of the notification system

This section describes the implementation of the vibrating indicator and the indicator led.

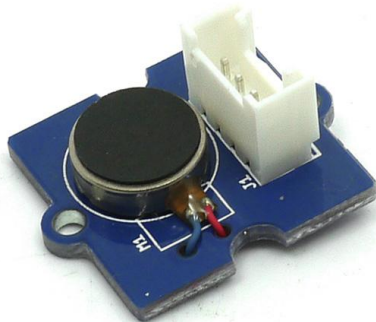
Vibrating indicator

As mentioned in section 3.1.2 the vibrating motor will be implemented in the teacher remote. The vibration motor should be connected to the M5Stamp C3 board. Therefore it should operate at 5V. Listed below are all the specifications the motor should have.

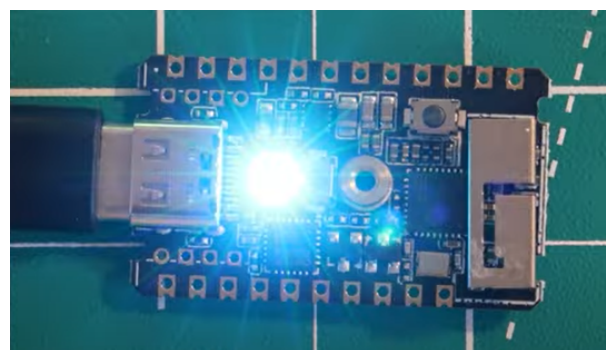
Coin motor specifications

1. Should operate at 5V DC.
2. Vibrating motor should be an ERM motor.
3. Motor should vibrate enough to get notified
4. Motor should not vibrate too hard that it is disturbing.

A vibrating module that meets all the specifications is the Grove vibration motor. This motor can be wired to the board, figure 3.7 shows the implementation of this motor. Figure 3.12a shows the motor.



(a) Grove vibration motor [27]



(b) The M5 Stamp-C3 board with RGB light that is on.

Figure 3.12

Indicator led The indicator led that will be used is an RGB led. The M5 Stamp-C3 has an RGB led integrated on the board as seen in figure 3.12b. This led can be used as indicator. If the RGB led is not sufficient visible, the design can be expanded to an RGB led with a better luminosity. But for prototype purposes the led on the board will be used.

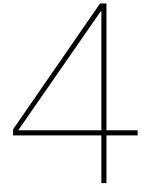
3.2.3. Validation of the notification system

The validation of the notification system will be done empirically. Because the system triggers human senses which hardly can be measured. For this validation we asked five educators their opinion about how they experience seeing the indicator led and touching the vibration motor. We asked if they would get notified by the indicators when giving a lecture. Furthermore we asked if this would not disturb them or the lesson.

The lecturers were all notified by the vibration motor. Four of them would not be disturbed by the vibration motor. One of them said it might be disturbing but not enough that he can not continue the lecture.

Three of the five lecturers had their doubts if they would get notified by the indicating led that is embedded on the M5Stamp C3 board. All the lectures said that they would not get disturbed by the RGB led.

According to the observations of the lecturers, the vibration motor is a useful solution for the problem. Since three of the five lecturers had doubts about the indicating led, it will be useful to expand this with an RGB led with higher luminosity. Nevertheless, for prototype purposes the RGB led on the board will be used.



Smart classroom

In this chapter the design and implementation of the hardware for the smart classroom is discussed. For this part two different sensors were selected. The first one is with regards to the climate control of a classroom and is discussed in section 4.1. The second sensor type is used to collect data to measure the student engagement in a lecture hall and is discussed in section 4.2. The climate control sensor uses a digital communication protocol that will be further elaborated on in section 4.1.2, while the other sensor will output analogue signals. These analogue signals will be converted to digital signals using the Analog-to-Digital Converter (ADC) on the ESP32-C3 chip. This is to show that the IoT system can handle both types of input signals, which are generally used by sensors. The system can therefore easily be expanded with other sensors.

4.1. Climate control

One of the goals of a smart classroom is to enhance the performance of students in the classroom. The climate in a classroom is strongly correlated with the performance. Temperature, humidity, fresh air flow and CO₂ concentration have an impact on the performance [28] [29]. According to Wyon (2004) the productivity can be increased with 6-9% when the air quality is controlled [30]. When environmental data can be collected the HVAC can be controlled resulting in an increase in productivity of the children [31]. Since temperature and humidity have a high impact on performance these measurements will be chosen first [28].

4.1.1. Temperature and humidity sensor

For the temperature and humidity sensor, four different types of I2C sensors were compared. These sensors are the DHT11 [32], DHT22 [33], TH02 [34] and SHT35 [35]. In Table 4.1 the performances of these four sensors according to their datasheets are compared. The accuracy data is from a typical environment, so these accuracies are worse for more extreme values.

Specification	DHT11 [32]	DHT22 [33]	TH02 [34]	SHT35 [35]
Temperature range [°C]	-20 to +60	-40 to +80	-40 to +85	-40 to +90
Temperature accuracy [°C]	±2	±0.5	±0.5	±0.3
Temperature drifting [°C/year]	<±0.5	unknown	<±0.05	<±0.03
Humidity range [%RH]	5-95	0-100	0-100	0-100
Humidity accuracy [%RH]	±5	±2	±3	±2
Humidity drifting [%RH/year]	<±0.5	<±0.5	<±0.25	<±0.25

Table 4.1: Comparison of four different temperature and humidity sensors with regards to their range, (typical) accuracy and long-term drifting

From this table it can be concluded that the SHT35 sensor from Sensirion AG is the best performing sensor of these four. A measurement from this sensor typically takes 2.5-12.5 ms, depending on the

repeatability condition. During measurements the sensor typically consumes 800 μA , while during the idle state it consumes 0.2 or 45 μA depending on the data acquisition mode that is selected.

4.1.2. I²C communication

As stated above in section 4.1.1 the communication of the SHT35 temperature and humidity sensor uses the I²C communication protocol. The protocol uses two wires for communication, the Serial Data (SDA) and Serial Clock (SCL) lines. I²C uses a master and slave device. In this application the ESP32-C3 chip is the master device and the temperature and humidity sensor is the slave device. The slave device only corresponds to a specific 7-bit address. The 8th bit is used to specify whether something will be read or written (from the perspective of the master). The slave device will only send data when the master device explicitly asks for it. All messages are sent in 8-bit frames, each frame being followed by an acknowledgement (ACK) bit from the other device.

For the SHT35 sensor each command consists of 16 bits [35]. The sensor will return every byte with with an ACK bit. For most generic commands the stop condition will be fulfilled after the last ACK and the communication stops. In fig. 4.1 a schematic display of such a command is presented. These commands include enabling periodic data acquisition mode, a soft reset, enabling and disabling the heater and clearing the status register. For all these commands no data is sent to the master by the slave and were thus all easily implemented.

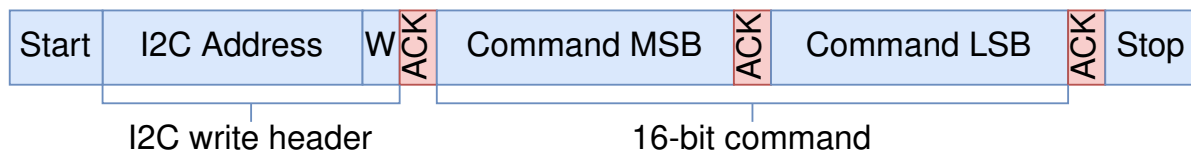


Figure 4.1: A schematic rendition of a generic 16-bit I²C command for the SHT35 sensor. Blue blocks are controlled by the ESP32-C3 microcontroller and red blocks by the sensor.

Acquiring and processing data

First the periodic data acquisition mode was implemented using a command like in fig. 4.1. The measurement frequency and repeatability can be changed by issuing a (slightly) different command. The exact chosen values for this will be elaborated on further below.

To read out the measurements from the sensor a different 16-bit command ($0\times\text{E}000$) is issued. In fig. 4.2 a schematic rendition of this I²C command is given. The start is similar to fig. 4.1, but to retrieve the data from the slave, the I²C read header is issued. If the slave responds with a not acknowledgement (NACK) bit, there is no data present to send to the master and the communication halts. When the slave responds with an ACK bit, it will send six bytes to the master. Both the temperature and humidity raw values are linearly scaled 16-bit unsigned integers and will thus take up four of those six bytes. The last two bytes are used for Cyclic Redundancy Check (CRC) to detect whether the transmitted data is correct. The CRC implemented in the SHT35 sensor uses a checksum calculation with a polynomial of $0\times\text{31}$. This check was also implemented in the code and it will return a checksum error when it is not correct.

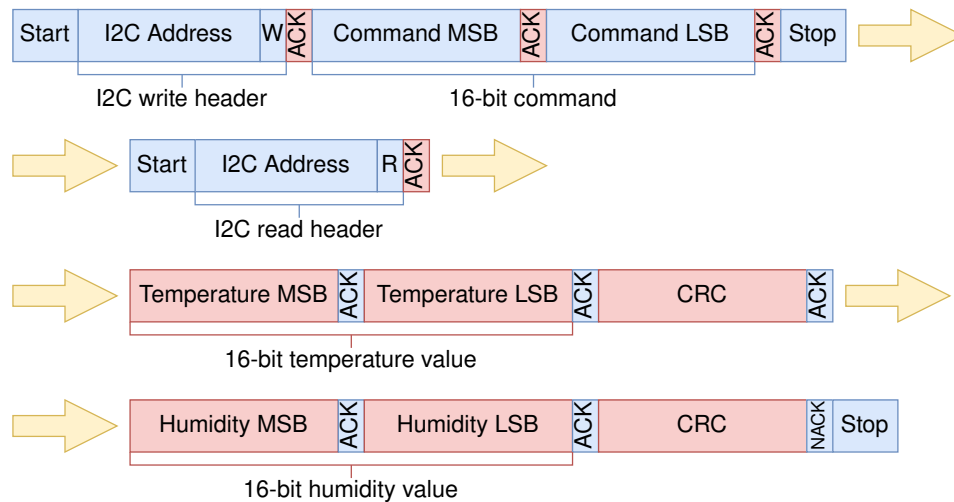


Figure 4.2: A schematic rendition of the I²C protocol for reading out the measurement results of the SHT35 sensor for periodic data acquisition mode. Blue blocks are controlled by the ESP32-C3 microcontroller and red blocks by the sensor.

The raw 16-bit unsigned integers will have to be converted onto a physical scale. Eq. (4.1) is used to calculate the temperature in °C and eq. (4.2) is used to calculate the relative humidity in %RH [35].

$$T = -45 + 175 \cdot \frac{S_T}{2^{16} - 1} [\text{°C}] \quad (4.1)$$

$$RH = 100 \cdot \frac{S_{RH}}{2^{16} - 1} [\%RH] \quad (4.2)$$

Measurement frequency

Using the periodic data acquisition mode the frequency can be selected at 0.5, 1, 2, 4, and 10 Hz. For the Smart Classroom application it is not necessary to measure the temperature and relative humidity this frequent. Taking into account the high accuracy of the SHT35 sensor a measuring frequency of once per minute makes more sense. In this case it is better to use the single shot mode instead of the periodic data acquisition mode.

In fig. 4.3 the I²C protocol for this single shot mode can be viewed. It looks very similar to reading out the measurement result in periodic data acquisition mode. The key difference is that the measurement still needs to be completed before the data can be sent. When the measurement is not yet finished the SHT35 sensor will respond with a NACK bit to the I²C read header. The SCL line will then remain free for a small period before the read header is sent again. When the measurement is completed the sensor will respond with an ACK bit to the read header and will then transmit the sensor data. It is also possible to use clock stretching in this mode, where the SCL line is pulled low by the slave until the measurement is completed. In the current implementation clock stretching is disabled.

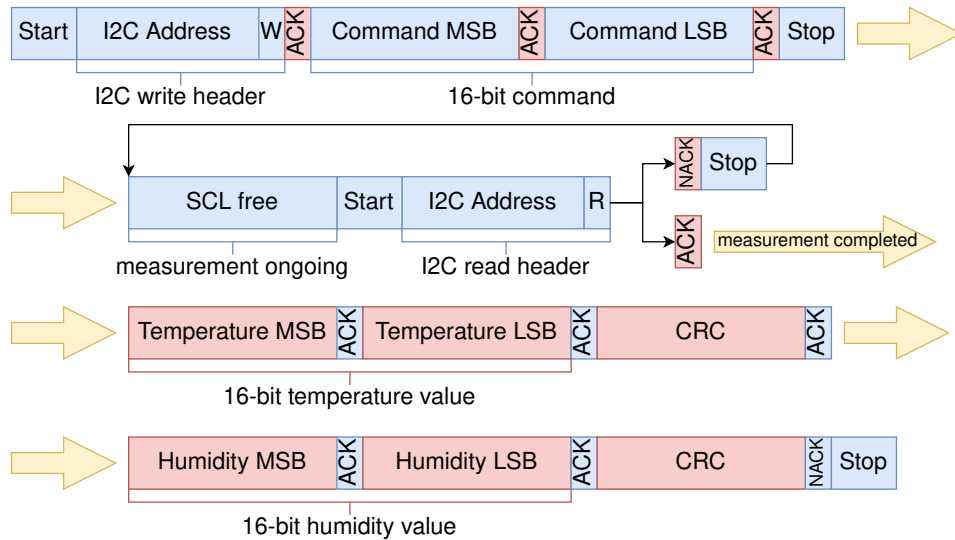


Figure 4.3: A schematic rendition of the I²C protocol for performing a single measurement of the SHT35 sensor using single shot mode. Blue blocks are controlled by the ESP32-C3 microcontroller and red blocks by the sensor.

Repeatability mode

The SHT35 sensor has three different repeatability modes that can be selected for measuring the temperature and humidity. For the high repeatability mode measurements will take longer and draw more energy. To determine the ideal setting for this application, the standard deviation of each mode was tested. This was done by completing 100 measurements at constant room temperature and at a measurement frequency of 4 Hz. Afterwards the standard deviation of this dataset was calculated. The results can be found in Table 4.2.

Repeatability mode	High	Medium	Low
$\sigma_{Temperature}$ [°C]	0.012	0.025	0.047
$\sigma_{Humidity}$ [%RH]	0.219	0.269	0.580

Table 4.2: Standard deviations of the SHT35 temperature and humidity sensor of 100 measurements at constant room temperature and at a measurement frequency of 4 Hz

As can be seen in Table 4.2, the standard deviation of the temperature is relatively low. Although it is significantly higher for the relative humidity, the sensor still yields good results. There is a notable difference between the different repeatability modes, especially the high repeatability mode has a very small standard deviation.

According to the datasheet [35], during measurements the SHT35 Temperature & Humidity sensor typically has a current draw of 800 μ A, with a maximum of 1500 μ A. In idle state this current draw is minimized to 0.2-2 μ A (in single shot mode). A high repeatability measurement typically takes 12.5 ms with a maximum of 15 ms. Although this is more than low- and medium repeatability measurements at 2.5 ms and 4.5 ms respectively, it is still a very short period. Using an oscilloscope the current draw of the entire M5Stamp C3 board including the SHT35 sensor was measured. At idle state (no wireless communication) this was found to be in the order of magnitude of 10 mA. This means that the current draw of the temperature and humidity sensor is negligible small compared to the board and a high accuracy is more preferable. Therefore from here on the high repeatability mode will be used. The code for the SHT35 sensor can be found in appendix B.

4.1.3. Validation

To test whether the sensor works correctly, the sensor performed measurements with a frequency of 1 Hz using periodic data acquisition mode for 250 seconds. After 33 seconds the sensor was put into a refrigerator. The result is plotted in fig. 4.4 and it can be seen that the temperature lowers gradually and the relative humidity increases because of the condensation.

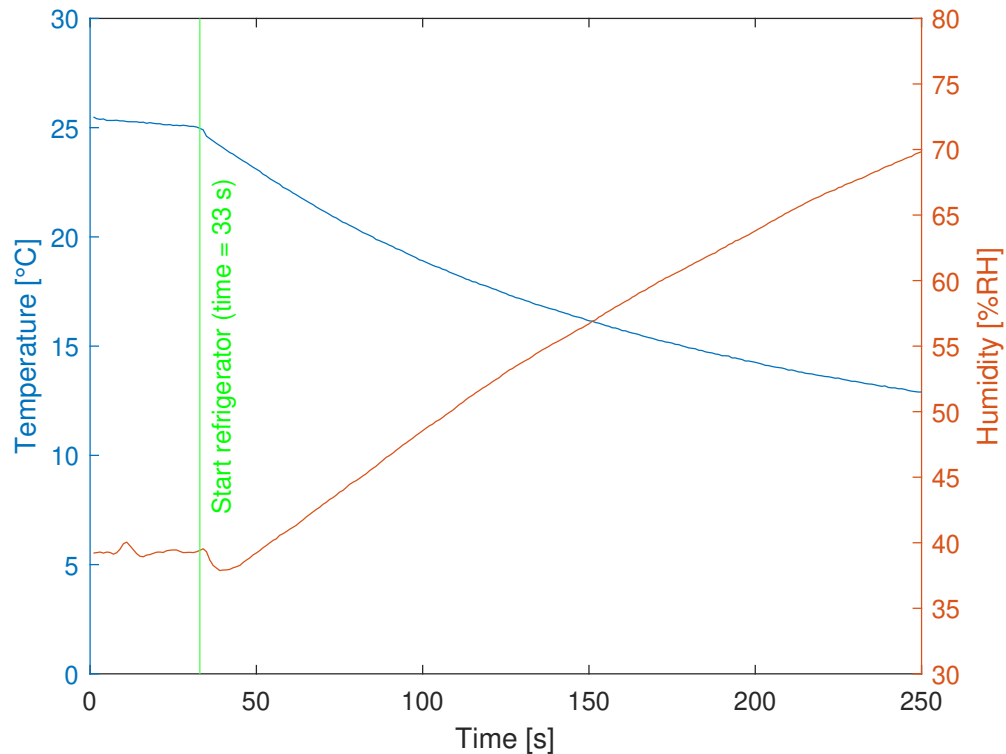


Figure 4.4: Temperature and humidity measurements after the SHT35 sensor was put in a refrigerator, with a frequency of 1 Hz

4.2. Data collection of audience engagement

Keeping the students engaged during lectures or instructions is an important part of teaching. When students are engaged they learn the material more effectively and perform better in academic environments [4] [36]. For teachers, it could be a useful tool to get some feedback on student engagement. With this information the lecturer can plan a break when necessary or make the lecture more interactive by starting a discussion or asking the audience questions.

For the smart classroom of this project, the audience engagement will be measured by the sound intensity of the background noise in the lecture hall. The interpretation of this data is not as clear cut, but a relatively silent classroom might indicate students being engaged in the lecture. The occasional loud sounds might be questions from students or simply students laughing at a joke of the lecturer. Continuously high volume background noise can indicate bored students talking and creating an overall buzz in the lecture hall [36] [37]. To get the most information out of these sensors, the data will have to be continuously monitored and logged. Also some basic data processing has been done by calculating the mean, median and standard deviation over a measurement period.

4.2.1. Loudness sensor

To measure the sound intensity several different sensors were considered. In section 3.1.2 the microphone for hybrid teaching is discussed and it could be possible to use that microphone to measure the sound intensity. However, from the program of requirements it follows that the sensors of the IoT network should be affordable and have a small form factor. So it was decided to go for a different route. For the implementation of the sound sensor the Grove - Loudness Sensor [38] from Seeed Studio was selected. In fig. 4.5 a picture of the loudness sensor can be found. It features a screw potentiometer that can be used for manual adjustment of the signal. It also filters the high frequencies of the signal and has a (theoretical) Signal-to-Noise Ratio (SNR) of 58 dB.

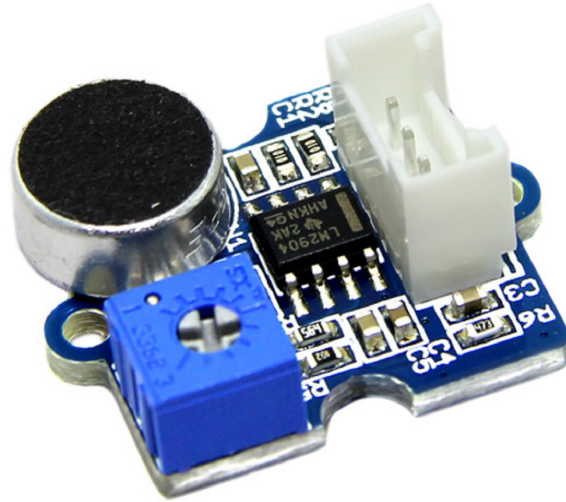


Figure 4.5: The Grove Loudness Sensor from Seeed studio. The blue box is the screw potentiometer for manual adjustment of the output gain and the white connector is the grove connector of the sensor [38].

Analogue readout of signal

The sensor features the convenient Grove connector also used in the SHT35 sensor, but in contrast does not use a digital communication protocol. The loudness sensor is an analog sensor, therefore the Analog-to-Digital Converter (ADC) of the ESP32-C3 chip is used. This chip features two Successive Approximation Register (SAR) ADC's. The ADC will output a 12-bit unsigned integer value with the measurement and multiple attenuation levels can be selected.

The ADC driver available for the ESP32-C3 chip was used. This driver includes a function that directly reads out the raw measurement of the ADC. In addition a built-in function that calibrates the ADC was used. Using another built-in function the output voltage can be computed from the raw ADC data.

Final implementation

Although it is possible to constantly measure the data, it may introduce extra challenges for the networking subgroup of the project. An interval of 10 ms is likely not enough for the wireless communication to successfully transmit the data. Therefore it was decided to do a measurement of one second and only send the important data points to the server. These data points would be the mean, median and standard deviation. In the future more advanced data processing could be implemented to send more useful data to the server. The code for the loudness sensor can be found in appendix C.

4.2.2. Validation

To test the functionality of the loudness sensor a testbench was written. It measures the sound intensity with a frequency of 100 Hz for 10 s. In fig. 4.6 the result of a test measurement can be found. The first three seconds of the measurement are in silence and from then on someone is speaking next to the sensor. Using Matlab the moving mean and moving standard deviation were also calculated, both with a window of 50 samples (or 0.5 s). The measurements were done with an attenuation of 6 dB, which is also used in the final implementation.

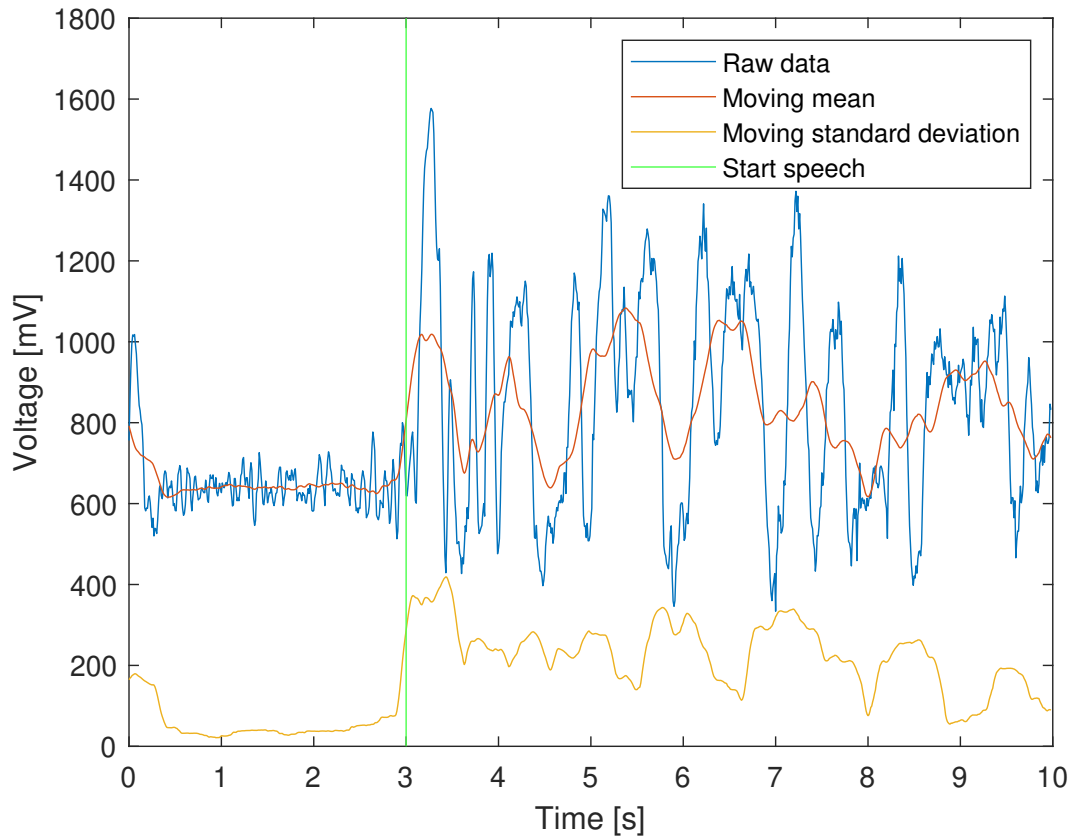


Figure 4.6: Measurements of the calculated voltage of the loudness sensor. The first three seconds are silent and from then on someone is speaking. The blue line is the raw data (in mV), while the red line is the moving mean and the yellow line the moving standard deviation (both with a window of 50 samples).

As can be seen in fig. 4.6, the larger amplitudes and variations in the recorded signals after $t=3s$, are indications of speech activities in the room. The voltage before speech is not necessarily much lower than the mean after. Nevertheless it can be concluded that in the current calibration a moving mean of larger than 800 mV are due to a person speaking and not due to random noise. In addition it was noted that (possibly due to the built-in filters) the signal also reaches lower peaks than during silence. Therefore the moving standard deviation was calculated, which correlates with the loudness of the person speaking.

5

Discussion

The goal of this project is to design a complete and expandable IoT system that implements a smart classroom framework and improves hybrid education. The prototype shows that the system supports various smart functionalities.

The temperature and humidity sensor can be used for environmental control in buildings. The system is able to read out the data using I²C. This protocol is generally used for sensors so the environmental control system can be easily expanded with more sensors that uses I²C. But other communication protocols such as UART, SPI, I²S and USB have not been implemented in the system.

The loudness sensor can be used for data collection to improve education with the help of student engagement tracking. To get the data the ADC on the M5Stamp C3 board is used. This implementation shows the system can be expanded with other analog sensors. There was a substantial amount of noise in the signal from the loudness sensor itself. The noise level is approximately 50% of the peaks of actual speech as shown in figure 4.6. Therefore it is inconvenient to use this data for student engagement tracking.

The smart functionality that toggles a classroom microphone is an example of an implementation that improves hybrid teaching. With the help of the IoT framework the microphone can get activated when desired. The teacher can use a remote to activate the audience microphone so that the online students can hear the students that are physically present. The implementation of the relay in the prototype is functional at a small scale. However the audio that is caused by switching on and off the relay must be filtered or suppressed.

The functionality that makes the lecturer aware of an online question is another example of an implementation that improves hybrid teaching. According to the survey that was held among five lecturers the prototype of the indicating system will be useful. The led should get elaborated better to be functional. All these functionalities shows that the system is a smart expandable IoT framework that improves hybrid teaching.

Program of requirements

In chapter 2 a program of requirements was devised. During the design and implementation phase decisions were made to accommodate those requirements. Mandatory requirement **M6**, which stated that sensors must be applicable to a classroom is achieved because all devices are designed with regards to a classroom. Mandatory requirements **M9**, **M10**, **M11** are all achieved by implementing the hybrid education prototype. The system notifies the teacher of online questions and an audience microphone is implemented that can be muted and unmuted by the teacher.

Trade-off requirement **T1**, which states that the system should be affordable is achieved by selecting cheap components and products for the design. Trade-off requirement **T2** states that the system should be customizable. All smart classroom sensors and future additions can operate independently, so the system could be customized for the customer's needs. Trade-off requirement **T3** states that the system

should be able to be operated without extensive technical knowledge. Any device designed in this thesis can simply be plugged in and does not require any additional setup. Requirement **T4**, which states that the hardware should have a small form factor, is achieved by using the small M5Stamp C3 development board and selecting small components like the relay and sensors. Trade-off requirement **HT1** states that sensors and actuators should preferably communicate with the IoT device using a grove connector. The relay, vibration motor, temperature and humidity sensor, and loudness sensor all use grove connectors.

The boundary conditions **HB1** and **HB2** stated that the devices will not be battery powered and existing drivers for the sensors will be used when available. The latter was done for both the temperature and humidity and loudness sensor, whose implementation required I²C or ADC drivers.

6

Conclusion

This thesis has described the design, implementation, and validation of the hardware part of the system. For improving hybrid education an audio system for the classroom was designed. This included a microphone for the audience and relay for toggling this microphone on and off. This relay can successfully turn on and off the audio from the microphone, although a spike can be heard when switching. The second part was a notification system for the teacher to be notified of online questions. A remote with a vibration motor and an indicator led were implemented to achieve this. The teacher can also use this remote to control the relay and thus the microphone.

The smart classroom was implemented by using a temperature and humidity sensor and a loudness sensor for audience engagement. The temperature and humidity sensor uses the digital communication protocol I²C to communicate with the IoT device. It will periodically measure the environment and send the measurements. The loudness sensor is an analog sensor that uses the ADC of the IoT device to convert the analog measurements to digital signals. Unfortunately the sensor has a lot of noise, but speech can still be detected. The important data points from short measurement intervals are sent by the IoT device.

To conclude, the hardware for both hybrid education and a smart classroom was successfully developed and can be used in the prototype for the final product. Within this IoT system is a framework for a smart classroom and improvements for hybrid education.

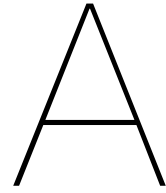
Recommendations And Future Work

An audio system that is able to record all the audio in a room is a complex system. As mentioned in section 3.1.1 every room has its own acoustics. So there is not a general solution for this problem. Except that multiple microphones will be best applicable for the design. But multiple microphones means that there must be some mixer or selection system that can handle all the microphones. This thesis does not describe such a system for multiple microphones. It only describes a selection system with one button and one microphone. But when multiple microphones are used, multiple buttons would be inconvenient. For future work it can be researched how this problem can be tackled. Perhaps using the loudness sensors to determine where the active speaker is and which microphone should be selected automatically.

The purpose of the relay in the audio path is to make sure that audio gets broadcast when desired. Therefore it is not necessary to shut down the microphone using a relay. This problem can also be tackled in the software. So the microphone is constantly recording and the audio system receives the audio. Then in the software will be implemented whether the audio gets broadcast or not. We have decided not to design this implementation since it is a software solution. So a hardware solution would be more suitable for this thesis. Furthermore a relay can be used for a lot more applications that can be useful for a classroom. For example lights, fan, heating system and other high power devices. So using a relay contributes to the design requirement that our system is easily expandable for multiple functionalities.

This thesis focuses on the audio in a classroom. However video is also needed for hybrid education. The design could be expanded with cameras to make the system more complete for hybrid education purposes.

When the relay switches a high amplitude of sound is generated. This sound will get broadcast to the online students. The sound is disturbing and should therefore be filtered. Such a system is not yet implemented and can be done in the future.



Survey results

What problems do you encounter with hybrid teaching?

1. That both groups expect to be the number 1 audience, while as teacher it is impossible to serve both groups equally. As an effect both groups are to some extent disappointed with the experience. It works fine if one audience is the target and the other group is either watching in or is 'audience'.
2. Chalkboards are not readable because of the light coming in. Often the entire board is filmed which makes text too small to read. The camera does not follow the teacher well.
3. Complications setting up cameras, smartboards and meetings and keeping an eye on the online participants during the lecture.
4. Lack of feedback, questions or participation from students connected online.
5. Students online do not engage. Focus is on students physically present.
6. Focus is to 1 group only, in most cases the ones that are present
7. Making the online people feel connected to the onsite group.
8. Hard to keep track of questions both in the room and online.
9. Serving two audiences at the same time.
10. Dividing my attention.

In what way could technology assist with or solve these problems?

1. A digital board that is streamed directly or writing on a tablet. A solution for following the teacher is unknown to me. I do not know if it is great to have the teacher stay in a fixed spot.
2. A different screen for projecting the student's camera that join online, a way to share the white board with the students online.
3. Work for one of the target groups only and improve the experience of the secondary group using technology.
4. Perhaps having a large screen (tv sized) in the audience to see the students at home could help.
5. If VR would be good enough that people online have the same perspective as in the room.
6. Less technology would be preferable (i.e. a proper blackboard) but a quality camera.
7. An up-voting mechanism for important questions.

What could be improved about hybrid teaching?

1. The quality of the sound and video streaming and setting it up. When managed from outside (for instance in aula) this works very well, but the quality is much worse when the lecturer has to set it up themselves.
2. Large smartboards which can be streamed directly (instead of via a camera) or good writing tablets of which the screen gets streamed and projected.
3. No smart boards, but traditional smart boards so you only need one technology (online meeting+camera) to function.
4. Balanced presence and attention of online and on campus students.
5. Participation of students joining online.
6. Connection with those online.
7. Interaction with each other.

In an ideal world with endless possibilities, what would be your vision on a hybrid classroom?

1. Either TV show setting where you do your class for the remote participants and the co-located ones are audience - as such they even have a lesser experience than the people at home, so we should give them all the tools and add ons that the people at home have too. OR we make co-located session where the people remote join in in such a way that all the things they miss from the co-location are compensated. I like the Getmibo townhall setting as example where remote environments bring co-located features.
2. A camera which follows the teacher automatically and a smartboard as big as an old school chalkboard with a second screen of the same size so you can shove the empty screen in front of the full screen.
3. All students turn on the cameras, a different screen for showing the students connected online so we 'feel' that also students joining online are actually there.
4. Just very high quality video and audio (but I still prefer the offline interaction, so I'd still prioritise those in the room).
5. Have the online people in the same line of vision as the onsite people. Like the holodeck.
6. If VR would be good enough that people online have the same perspective as in the room.
7. Blackboard + good camera + screen for feedback with online students.

What problems do you encounter in a smart classroom?

1. Every system is different and I use it not often. As such the UI always makes you wonder how it works.
2. Sometimes setting up the systems is really difficult.
3. Smart boards not working
4. Technology not working.

In what way could technology assist with or solve these problems?

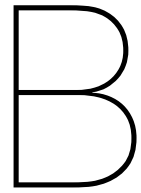
1. Replace smartboards with proper blackboards
2. Have less malfunctions.
3. Voice response

What could be improved about smart classrooms?

1. Often the controls are fixed to the desk and the desk as such needs a fixed spot. Before you know it the desk becomes THE central place in front, which it is not but cannot be moved away.

In an ideal world with endless possibilities, what would be your vision on a smart classroom?

1. Fully automatic - no display - maybe voice controlled
2. It is easier to set up all the systems.



Temperature and humidity sensor code

i2c_commands.h

```
1  /*
2  * i2c_commands.h
3  *
4  * Created on: 3 May 2022
5  * Author: Richard & Tom
6  */
7
8  #ifndef MAIN_I2C_COMMANDS_H_
9  #define MAIN_I2C_COMMANDS_H_
10
11 #include "freertos/FreeRTOS.h"
12 #include "driver/i2c.h"
13 #include "interface.h"
14
15 #define I2C_MASTER_SCL_IO 18 /*!< GPIO number used for I2C master clock
16 */
17 #define I2C_MASTER_SDA_IO 19 /*!< GPIO number used for I2C master data
18 */
19 #define I2C_MASTER_NUM 0 /*!< I2C master i2c port number, the
20 number of i2c peripheral interfaces available will depend on the chip */
21 #define I2C_MASTER_FREQ_HZ 100000 /*!< I2C master clock frequency */
22 #define I2C_MASTER_TX_BUF_DISABLE 0 /*!< I2C master doesn't need
23 buffer */
24 #define I2C_MASTER_RX_BUF_DISABLE 0 /*!< I2C master doesn't need
25 buffer */
26 #define I2C_MASTER_TIMEOUT_MS 1000
27
28 #define SHT35_SENSOR_ADDR 0x45
29
30 #define POLYNOMIAL 0x31
31
32 // Measurement repeatability
33 typedef enum{
34 HIGH_REPEATABLITY,
35 MEDIUM_REPEATABLITY,
36 LOW_REPEATABLITY,
37 }etRepeatability;
38
39 // Measurement frequency
40 typedef enum{
41 FREQUENCY_HZ5, // 0.5 measurements per seconds
42 FREQUENCY_1HZ, // 1.0 measurements per seconds
43 FREQUENCY_2HZ, // 2.0 measurements per seconds
44 FREQUENCY_4HZ, // 4.0 measurements per seconds
45 FREQUENCY_10HZ, // 10.0 measurements per seconds
46 }etFrequency;
47
48 esp_err_t i2c_master_init(void);
```



```

44 esp_err_t SHT35_single_measurement(int16_t *temp_ptr, uint16_t *hum_ptr);
45 esp_err_t SHT35_read_out_status_register(uint8_t *data, size_t read_size);
46 esp_err_t SHT35_read_and_print_status_register(void);
47 esp_err_t SHT35_single_shot_data_acquisition(uint8_t *data, size_t read_size, _Bool
    clock_stretching, etRepeatability repeatability);
48 esp_err_t SHT35_read_measurements_periodic_mode(uint8_t *data, size_t read_size);
49 esp_err_t SHT35_heater(_Bool heater_enabled);
50 esp_err_t SHT35_periodic_data_acquisition(etFrequency frequency, etRepeatability
    repeatability);
51 esp_err_t SHT35_soft_reset(void);
52 esp_err_t SHT35_break_command(void);
53 esp_err_t SHT35_ART_command(void);
54 esp_err_t SHT35_clear_status_register(void);
55
56 #endif /* MAIN_I2C_COMMANDS_H */

```

i2c_commands.c

```

1  /*
2  * i2c_commands.c
3  *
4  * Created on: 3 May 2022
5  * Author: Richard & Tom
6  */
7
8  #include "i2c_commands.h"
9
10 esp_err_t i2c_master_init(void)
11 /* Initialization function of the I2C protocol on the master side (ESP32) */
12 {
13     int i2c_master_port = I2C_MASTER_NUM;
14
15     i2c_config_t conf = {
16         .mode = I2C_MODE_MASTER,
17         .sda_io_num = I2C_MASTER_SDA_IO,
18         .scl_io_num = I2C_MASTER_SCL_IO,
19         .sda_pullup_en = GPIO_PULLUP_ENABLE,
20         .scl_pullup_en = GPIO_PULLUP_ENABLE,
21         .master.clk_speed = I2C_MASTER_FREQ_HZ,
22     };
23
24     i2c_param_config(i2c_master_port, &conf);
25
26     return i2c_driver_install(i2c_master_port, conf.mode, I2C_MASTER_RX_BUF_DISABLE,
        I2C_MASTER_TX_BUF_DISABLE, 0);
27 }
28
29 esp_err_t SHT35_single_measurement(int16_t *temp_ptr, uint16_t *hum_ptr)
30 /* Function to do a single measurement using single shot mode,
31 * the raw data is processed to integers ready for wireless transfer */
32 {
33     uint8_t raw_data[6] = {0};
34     size_t raw_data_len = sizeof(raw_data);
35     uint8_t *raw_data_ptr = raw_data;
36
37     esp_err_t err = SHT35_single_shot_data_acquisition(raw_data_ptr, raw_data_len, false,
        HIGH_REPEATABILITY);
38
39     if (err != ESP_OK)
40         return err;
41
42     err = process_raw_temp_hum_values(raw_data_ptr, raw_data_len, temp_ptr, hum_ptr);
43
44     return err;
45 }
46
47 static uint8_t SHT35_calculate_crc(uint8_t data[], uint8_t number_of_bytes)
48 {
49     uint8_t bit; // bit mask
50     uint8_t crc = 0xFF; // calculated checksum

```

```

51 uint8_t byte_ctr; // byte counter
52
53 // calculates 8-Bit checksum with given polynomial
54 for(byte_ctr = 0; byte_ctr < number_of_bytes; byte_ctr++)
55 {
56     crc ^= (data[byte_ctr]);
57     for(bit = 8; bit > 0; --bit)
58     {
59         if(crc & 0x80)
60             crc = (crc << 1) ^ POLYNOMIAL;
61         else
62             crc = (crc << 1);
63     }
64 }
65
66 return crc;
67 }
68
69 static esp_err_t SHT35_check_crc(uint8_t data[], uint8_t number_of_bytes, uint8_t checksum)
70 {
71     uint8_t crc; // calculated checksum
72
73     // calculates 8-Bit checksum
74     crc = SHT35_calculate_crc(data, number_of_bytes);
75
76     // verify checksum
77     if(crc != checksum)
78         return ESP_ERR_INVALID_CRC;
79     else
80         return ESP_OK;
81 }
82
83
84 esp_err_t SHT35_read_out_status_register(uint8_t *data, size_t read_size)
85 /* Function that reads out the status register and puts the data in 2 bytes + a crc byte ,
86 * read_size should be 3 bytes */
87 {
88     esp_err_t err;
89     uint8_t write_buffer[2] = {0xF3, 0x2D};
90     uint8_t *buffer_ptr = write_buffer;
91     size_t write_size = 2;
92
93     if(read_size > 3)
94         return ESP_ERR_INVALID_ARG;
95
96     err = i2c_master_write_read_device(I2C_MASTER_NUM, SHT35_SENSOR_ADDR, buffer_ptr,
97         write_size, data, read_size, I2C_MASTER_TIMEOUT_MS / portTICK_RATE_MS);
98
99     if(err != ESP_OK)
100         return err;
101
102     err = SHT35_check_crc(data, 2, *(data+2));
103
104     return err;
105 }
106
107 esp_err_t SHT35_read_and_print_status_register(void)
108 /* Function that reads out the status register and prints it using the functions
109 * SHT35_read_out_status_register and print_status_register */
110 {
111     uint8_t reg_data[3] = {0};
112     size_t reg_len = sizeof(reg_data);
113     uint8_t *reg_ptr = reg_data;
114
115     esp_err_t err = SHT35_read_out_status_register(reg_ptr, reg_len);
116     if(err != ESP_OK)
117         return err;
118
119     err = print_status_register(reg_ptr, reg_len);
120
121     return err;

```

```

121 }
122
123 esp_err_t SHT35_single_shot_data_acquisition(uint8_t *data, size_t read_size, _Bool
        clock_stretching, etRepeatability repeatability)
124 /* Function that acquires a single data point using single shot mode, read_size should be 6
        bytes */
125 {
126     esp_err_t err = ESP_OK;
127
128     uint8_t write_buffer[2] = {0};
129     uint8_t *buffer_ptr = write_buffer;
130     size_t write_size = 2;
131
132     if(read_size != 6)
133         err = ESP_ERR_INVALID_ARG;
134
135     if(clock_stretching)
136     {
137         write_buffer[0] = 0x2C;
138         switch(repeatability)
139         {
140             case HIGH_REPEATABILITY:
141                 write_buffer[1] = 0x06;
142                 break;
143             case MEDIUM_REPEATABILITY:
144                 write_buffer[1] = 0x0D;
145                 break;
146             case LOW_REPEATABILITY:
147                 write_buffer[1] = 0x10;
148                 break;
149             default:
150                 err = ESP_ERR_INVALID_ARG;
151         }
152     }
153     else
154     {
155         write_buffer[0] = 0x24;
156         switch(repeatability)
157         {
158             case HIGH_REPEATABILITY:
159                 write_buffer[1] = 0x00;
160                 break;
161             case MEDIUM_REPEATABILITY:
162                 write_buffer[1] = 0x0B;
163                 break;
164             case LOW_REPEATABILITY:
165                 write_buffer[1] = 0x16;
166                 break;
167             default:
168                 err = ESP_ERR_INVALID_ARG;
169         }
170     }
171
172     if(err != ESP_OK)
173         return err;
174
175     err = i2c_master_write_read_device(I2C_MASTER_NUM, SHT35_SENSOR_ADDR, buffer_ptr,
        write_size, data, read_size, I2C_MASTER_TIMEOUT_MS / portTICK_RATE_MS);
176
177     if(err == ESP_OK)
178         err = SHT35_check_crc(data, 2, *(data+2));
179     if(err == ESP_OK)
180         err = SHT35_check_crc(data+3, 2, *(data+5));
181
182     return err;
183 }
184
185 esp_err_t SHT35_read_measurements_periodic_mode(uint8_t *data, size_t read_size)
186 /* After enabling periodic mode the measurements can be read using this mode, read_size
        should be 6 bytes */
187 {

```

```

188 esp_err_t err;
189 uint8_t write_buffer[2] = {0xE0, 0x00};
190 uint8_t *buffer_ptr = write_buffer;
191 size_t write_size = 2;
192
193 if(read_size != 6)
194     return ESP_ERR_INVALID_ARG;
195
196 err = i2c_master_write_read_device(I2C_MASTER_NUM, SHT35_SENSOR_ADDR, buffer_ptr,
197     write_size, data, read_size, I2C_MASTER_TIMEOUT_MS / portTICK_RATE_MS);
198
199 if(err == ESP_OK)
200     err = SHT35_check_crc(data, 2, *(data+2));
201 if(err == ESP_OK)
202     err = SHT35_check_crc(data+3, 2, *(data+5));
203
204 return err;
205 }
206
207 esp_err_t SHT35_heater(_Bool heater_enabled)
208 /* Enabling/disabling the heater. NOT TESTED */
209 {
210     uint8_t write_buffer[2] = {0};
211     uint8_t *buffer_ptr = write_buffer;
212     size_t size = 2;
213
214     write_buffer[0] = 0x30;
215     if(heater_enabled)
216         write_buffer[1] = 0x6D;
217     else
218         write_buffer[1] = 0x66;
219
220     return i2c_master_write_to_device(I2C_MASTER_NUM, SHT35_SENSOR_ADDR, buffer_ptr, size,
221     I2C_MASTER_TIMEOUT_MS / portTICK_RATE_MS);
222 }
223
224 esp_err_t SHT35_periodic_data_acquisition(etFrequency frequency, etRepeatability
225     repeatability)
226 /* Function that enables periodic data acquisition, the sensor starts measuring frequently */
227 {
228     esp_err_t err = ESP_OK;
229
230     uint8_t write_buffer[2] = {0};
231     uint8_t *buffer_ptr = write_buffer;
232     size_t size = 2;
233
234     switch(frequency)
235     {
236     case FREQUENCY_HZ5:
237         write_buffer[0] = 0x20;
238         switch(repeatability)
239         {
240         case HIGH_REPEATABILITY:
241             write_buffer[1] = 0x32;
242             break;
243         case MEDIUM_REPEATABILITY:
244             write_buffer[1] = 0x24;
245             break;
246         case LOW_REPEATABILITY:
247             write_buffer[1] = 0x2F;
248             break;
249         default:
250             err = ESP_ERR_INVALID_ARG;
251         }
252         break;
253
254     case FREQUENCY_1HZ:
255         write_buffer[0] = 0x21;
256         switch(repeatability)
257         {
258         case HIGH_REPEATABILITY:

```

```
256     write_buffer[1] = 0x30;
257     break;
258     case MEDIUM_REPEATABILITY:
259     write_buffer[1] = 0x26;
260     break;
261     case LOW_REPEATABILITY:
262     write_buffer[1] = 0x2D;
263     break;
264     default:
265     err = ESP_ERR_INVALID_ARG;
266 }
267 break;
268
269 case FREQUENCY_2HZ:
270 write_buffer[0] = 0x22;
271 switch(repeatability)
272 {
273     case HIGH_REPEATABILITY:
274     write_buffer[1] = 0x36;
275     break;
276     case MEDIUM_REPEATABILITY:
277     write_buffer[1] = 0x20;
278     break;
279     case LOW_REPEATABILITY:
280     write_buffer[1] = 0x2B;
281     break;
282     default:
283     err = ESP_ERR_INVALID_ARG;
284 }
285 break;
286
287 case FREQUENCY_4HZ:
288 write_buffer[0] = 0x23;
289 switch(repeatability)
290 {
291     case HIGH_REPEATABILITY:
292     write_buffer[1] = 0x34;
293     break;
294     case MEDIUM_REPEATABILITY:
295     write_buffer[1] = 0x22;
296     break;
297     case LOW_REPEATABILITY:
298     write_buffer[1] = 0x29;
299     break;
300     default:
301     err = ESP_ERR_INVALID_ARG;
302 }
303 break;
304
305 case FREQUENCY_10HZ:
306 write_buffer[0] = 0x27;
307 switch(repeatability)
308 {
309     case HIGH_REPEATABILITY:
310     write_buffer[1] = 0x37;
311     break;
312     case MEDIUM_REPEATABILITY:
313     write_buffer[1] = 0x21;
314     break;
315     case LOW_REPEATABILITY:
316     write_buffer[1] = 0x2A;
317     break;
318     default:
319     err = ESP_ERR_INVALID_ARG;
320 }
321 break;
322
323 default:
324 err = ESP_ERR_INVALID_ARG;
325 }
326
```

```

327     if (err != ESP_OK)
328         return err;
329
330     err = i2c_master_write_to_device(I2C_MASTER_NUM, SHT35_SENSOR_ADDR, buffer_ptr, size,
331                                     I2C_MASTER_TIMEOUT_MS / portTICK_RATE_MS);
332
333     return err;
334 }
335
336 /* Simple single 16-bit commands */
337
338 esp_err_t SHT35_soft_reset(void)
339 /* Software reset command */
340 {
341     uint8_t write_buffer[2] = {0x30, 0xA2};
342     uint8_t *buffer_ptr = write_buffer;
343     size_t size = 2;
344
345     return i2c_master_write_to_device(I2C_MASTER_NUM, SHT35_SENSOR_ADDR, buffer_ptr, size,
346                                     I2C_MASTER_TIMEOUT_MS / portTICK_RATE_MS);
347 }
348
349 esp_err_t SHT35_break_command(void)
350 /* Stop periodic data acquisition mode */
351 {
352     uint8_t write_buffer[2] = {0x30, 0x93};
353     uint8_t *buffer_ptr = write_buffer;
354     size_t size = 2;
355
356     return i2c_master_write_to_device(I2C_MASTER_NUM, SHT35_SENSOR_ADDR, buffer_ptr, size,
357                                     I2C_MASTER_TIMEOUT_MS / portTICK_RATE_MS);
358 }
359
360 esp_err_t SHT35_ART_command(void)
361 /* Enable accelerated response time measurements */
362 {
363     uint8_t write_buffer[2] = {0x2B, 0x32};
364     uint8_t *buffer_ptr = write_buffer;
365     size_t size = 2;
366
367     return i2c_master_write_to_device(I2C_MASTER_NUM, SHT35_SENSOR_ADDR, buffer_ptr, size,
368                                     I2C_MASTER_TIMEOUT_MS / portTICK_RATE_MS);
369 }
370
371 esp_err_t SHT35_clear_status_register(void)
372 /* Clear the status register */
373 {
374     uint8_t write_buffer[2] = {0x30, 0x41};
375     uint8_t *buffer_ptr = write_buffer;
376     size_t size = 2;
377
378     return i2c_master_write_to_device(I2C_MASTER_NUM, SHT35_SENSOR_ADDR, buffer_ptr, size,
379                                     I2C_MASTER_TIMEOUT_MS / portTICK_RATE_MS);
380 }

```

interface.h

```

1  /*
2  * interface.h
3  *
4  * Created on: 6 May 2022
5  * Author: Richard & Tom
6  */
7
8  #ifndef MAIN_INTERFACE_H_
9  #define MAIN_INTERFACE_H_
10
11 #include "esp_err.h"
12 #include <math.h>

```

```

13
14 esp_err_t process_raw_temp_hum_values(uint8_t *data, size_t data_size, int16_t *temp_ptr,
    uint16_t *hum_ptr);
15 void print_sensor_values(int16_t *temp_ptr, uint16_t *hum_ptr);
16 esp_err_t print_status_register(uint8_t *data, size_t data_size);
17
18 #endif /* MAIN_INTERFACE_H */

```

interface.c

```

1 /*
2  * interface.c
3  *
4  * Created on: 6 May 2022
5  * Author: Richard & Tom
6  */
7
8 #include "interface.h"
9
10 esp_err_t process_raw_temp_hum_values(uint8_t *data, size_t data_size, int16_t *temp_ptr,
    uint16_t *hum_ptr)
11 /* Process the raw data from the sensor and put them into integers ready to transfer using
    the wireless protocol */
12 {
13     if(data_size < 5 || data_size > 6)
14         return ESP_ERR_INVALID_ARG;
15
16     float temperature = (*(data << 8) + *(data+1));
17     temperature = -45 + 175.0*temperature/(65536.0 - 1);
18
19     float humidity = (*(data+3) << 8) + *(data+4);
20     humidity = 100.0*humidity/(65536.0 - 1);
21
22     *temp_ptr = round(temperature * 100.0);
23     *hum_ptr = round(humidity * 100.0);
24
25     return ESP_OK;
26 }
27
28 void print_sensor_values(int16_t *temp_ptr, uint16_t *hum_ptr)
29 /* Print the sensor values, input should be pointers to 16 bit integers values for
    temperature and humidity */
30 {
31     printf("Temperature: %.2f\n", *temp_ptr / 100.0);
32     printf("Humidity: %.2f\n", *hum_ptr / 100.0);
33 }
34
35 esp_err_t print_status_register(uint8_t *data, size_t data_size)
36 /* Prints the status register to the terminal in a convenient format */
37 {
38     if(data_size < 2 || data_size > 3)
39         return ESP_ERR_INVALID_ARG;
40
41     _Bool alert_pending = *data & 0b10000000;
42     _Bool heater_status = *data & 0b00100000;
43     _Bool rh_tracking_alert = *data & 0b00001000;
44     _Bool t_tracking_alert = *data & 0b00000100;
45     _Bool system_reset_detected = *(data+1) & 0b00010000;
46     _Bool command_status = *(data+1) & 0b00000010;
47     _Bool write_data_checksum_status = *(data+1) & 0b00000001;
48
49     printf("Alert pending status\t\t- ");
50     if(alert_pending)
51         printf("'1': at least one pending alert\n");
52     else
53         printf("'0': no pending alerts\n");
54
55     printf("Heater status\t\t\t- ");
56     if(heater_status)
57         printf("'1': Heater ON\n");

```

```

58     else
59         printf(" '0': Heater OFF\n");
60
61     printf("RH tracking alert\t\t- ");
62     if(rh_tracking_alert)
63         printf(" '1': alert\n");
64     else
65         printf(" '0': no alert\n");
66
67     printf("T tracking alert\t\t- ");
68     if(t_tracking_alert)
69         printf(" '1': alert\n");
70     else
71         printf(" '0': no alert\n");
72
73     printf("System reset detected\t\t- ");
74     if(system_reset_detected)
75         printf(" '1': reset detected (hard reset, soft reset command or supply fail)\n");
76     else
77         printf(" '0': no reset detected since last 'clear status register' command\n");
78
79     printf("Command status\t\t\t- ");
80     if(command_status)
81         printf(" '1': last command not processed. It was either invalid, failed the integrated
82         command checksum\n");
83     else
84         printf(" '0': last command executed successfully\n");
85
86     printf("Write data checksum status\t- ");
87     if(write_data_checksum_status)
88         printf(" '1': checksum of last write transfer failed\n");
89     else
90         printf(" '0': checksum of last write transfer was correct\n");
91
92     return ESP_OK;
93 }

```

main.c

```

1  #include "freertos/FreeRTOS.h"
2  #include "esp_wifi.h"
3  #include "esp_system.h"
4  #include "esp_event.h"
5  #include "nvs_flash.h"
6  #include "driver/gpio.h"
7  #include <stdio.h>
8  #include "esp_log.h"
9  #include "driver/i2c.h"
10 #include "i2c_commands.h"
11 #include "interface.h"
12 #include <stdlib.h>
13 #include <math.h>
14
15 #define MEAS_CNT 250
16
17 static const char *TAG = "i2c";
18
19 static void SHT35_single_measurement_test();
20 static void periodic_measurements_test();
21 static void single_shot_measurements_test();
22 static void status_register_test();
23 static void SHT35_repeatability_test();
24 static void SHT35_measurement();
25
26 void app_main(void)
27 {
28     // periodic_measurements_test2();
29
30     // single_shot_measurements_test();
31

```



```

32 // status_register_test();
33
34 // SHT35_single_measurement_test();
35
36
37 // SHT35_repeatability_test();
38
39 SHT35_measurement();
40
41 }
42
43 static void SHT35_repeatability_test()
44 /* Function to test the different repeatability modes of the sensor for the thesis */
45 {
46 ESP_ERROR_CHECK_WITHOUT_ABORT(i2c_master_init());
47 ESP_LOGI(TAG, "I2C initialized successfully");
48
49 vTaskDelay(pdMS_TO_TICKS(100));
50
51 ESP_ERROR_CHECK_WITHOUT_ABORT(SHT35_periodic_data_acquisition(FREQUENCY_4HZ,
52     LOW_REPEATABILITY));
53
54 uint8_t raw_data[6] = {0};
55 size_t raw_data_len = sizeof(raw_data);
56 uint8_t *raw_data_ptr = raw_data;
57
58 int16_t temperature_raw = 0;
59 uint16_t humidity_raw = 0;
60
61 float temperature[MEAS_CNT] = {0};
62 float humidity[MEAS_CNT] = {0};
63
64 vTaskDelay(pdMS_TO_TICKS(250));
65
66 for (int i = 0; i < MEAS_CNT; i++)
67 {
68     ESP_ERROR_CHECK_WITHOUT_ABORT(SHT35_read_measurements_periodic_mode(raw_data_ptr ,
69         raw_data_len));
70
71     ESP_ERROR_CHECK_WITHOUT_ABORT(process_raw_temp_hum_values(raw_data_ptr , raw_data_len , &
72         temperature_raw , &humidity_raw));
73
74     temperature[i] = temperature_raw/100.0;
75     humidity[i] = humidity_raw/100.0;
76
77     ESP_LOGI(TAG, "Temp: %f" , temperature[i]);
78     ESP_LOGI(TAG, "Hum: %f" , humidity[i]);
79
80     vTaskDelay(pdMS_TO_TICKS(250));
81 }
82
83 float mean_temp = 0.0;
84 float mean_hum = 0.0;
85
86 for (int i = 0; i < MEAS_CNT; i++)
87 {
88     mean_temp += temperature[i];
89     mean_hum += humidity[i];
90 }
91
92 mean_temp /= MEAS_CNT;
93 mean_hum /= MEAS_CNT;
94
95 float sum_squared_diff_temp = 0.0;
96 float sum_squared_diff_hum = 0.0;
97
98 for (int i = 0; i < MEAS_CNT; i++)
99 {
100     sum_squared_diff_temp += (temperature[i] - mean_temp)*(temperature[i] - mean_temp);
101     sum_squared_diff_hum += (humidity[i] - mean_hum)*(humidity[i] - mean_hum);
102 }

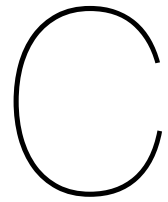
```

```

100 float std_dev_temp = sqrt(sum_squared_diff_temp/MEAS_CNT);
101 float std_dev_hum = sqrt(sum_squared_diff_hum/MEAS_CNT);
102
103
104 ESP_LOGI(TAG, "Standard deviation temp: %f", std_dev_temp);
105 ESP_LOGI(TAG, "Standard deviation hum: %f", std_dev_hum);
106 }
107
108 static void SHT35_measurement()
109 /* Measurement function for the thesis */
110 {
111 ESP_ERROR_CHECK_WITHOUT_ABORT(i2c_master_init());
112 ESP_LOGI(TAG, "I2C initialized successfully");
113
114 vTaskDelay(pdMS_TO_TICKS(100));
115
116 ESP_ERROR_CHECK_WITHOUT_ABORT(SHT35_periodic_data_acquisition(FREQUENCY_1HZ,
HIGH_REPEATABILITY));
117
118 uint8_t raw_data[6] = {0};
119 size_t raw_data_len = sizeof(raw_data);
120 uint8_t *raw_data_ptr = raw_data;
121
122 int16_t temperature_raw = 0;
123 uint16_t humidity_raw = 0;
124
125 float temperature[MEAS_CNT] = {0};
126 float humidity[MEAS_CNT] = {0};
127
128 vTaskDelay(pdMS_TO_TICKS(1000));
129
130 for (int i = 0; i < MEAS_CNT; i++)
131 {
132 ESP_ERROR_CHECK_WITHOUT_ABORT(SHT35_read_measurements_periodic_mode(raw_data_ptr ,
raw_data_len));
133
134 ESP_ERROR_CHECK_WITHOUT_ABORT(process_raw_temp_hum_values(raw_data_ptr , raw_data_len , &
temperature_raw , &humidity_raw));
135
136 temperature[i] = temperature_raw/100.0;
137 humidity[i] = humidity_raw/100.0;
138
139 vTaskDelay(pdMS_TO_TICKS(1000));
140 }
141
142 printf("Temperature:\n");
143 for (int i = 0; i < MEAS_CNT; i++)
144 printf("%.2f\n", temperature[i]);
145
146 printf("Humidity:\n");
147 for (int i = 0; i < MEAS_CNT; i++)
148 printf("%.2f\n", humidity[i]);
149 }
150
151
152
153 static void SHT35_single_measurement_test()
154 {
155 ESP_ERROR_CHECK_WITHOUT_ABORT(i2c_master_init());
156 ESP_LOGI(TAG, "I2C initialized successfully");
157
158 vTaskDelay(pdMS_TO_TICKS(10));
159
160 int16_t temperature;
161 uint16_t humidity;
162
163 ESP_ERROR_CHECK_WITHOUT_ABORT(SHT35_single_measurement(&temperature , &humidity));
164
165 print_sensor_values(&temperature , &humidity);
166 }
167

```

```
168 static void periodic_measurements_test()
169 {
170     ESP_ERROR_CHECK_WITHOUT_ABORT(i2c_master_init());
171     ESP_LOGI(TAG, "I2C initialized successfully");
172
173     vTaskDelay(pdMS_TO_TICKS(100));
174
175     ESP_ERROR_CHECK_WITHOUT_ABORT(SHT35_periodic_data_acquisition(FREQUENCY_10HZ,
176     HIGH_REPEATABILITY));
177
178     while(true)
179     {
180         vTaskDelay(pdMS_TO_TICKS(100));
181
182         uint8_t data[6] = {0};
183         size_t len=sizeof(data);
184         uint8_t *temp_hum = data;
185         ESP_ERROR_CHECK_WITHOUT_ABORT(SHT35_read_measurements_periodic_mode(temp_hum, len));
186
187         int16_t temperature;
188         uint16_t humidity;
189         ESP_ERROR_CHECK_WITHOUT_ABORT(process_raw_temp_hum_values(temp_hum, len, &temperature, &
190         humidity));
191
192         // print_sensor_values(&temperature, &humidity);
193     }
194 }
195
196 static void single_shot_measurements_test()
197 {
198     ESP_ERROR_CHECK_WITHOUT_ABORT(i2c_master_init());
199     ESP_LOGI(TAG, "I2C initialized successfully");
200
201     vTaskDelay(pdMS_TO_TICKS(10));
202
203     while(true)
204     {
205         vTaskDelay(pdMS_TO_TICKS(1000));
206
207         uint8_t data[6] = {0};
208         size_t len=sizeof(data);
209         uint8_t *temp_hum = data;
210         ESP_ERROR_CHECK_WITHOUT_ABORT(SHT35_single_shot_data_acquisition(temp_hum, len, false,
211         HIGH_REPEATABILITY));
212
213         int16_t temperature;
214         uint16_t humidity;
215         ESP_ERROR_CHECK_WITHOUT_ABORT(process_raw_temp_hum_values(temp_hum, len, &temperature, &
216         humidity));
217
218         print_sensor_values(&temperature, &humidity);
219     }
220 }
221
222 static void status_register_test()
223 {
224     ESP_ERROR_CHECK_WITHOUT_ABORT(i2c_master_init());
225     ESP_LOGI(TAG, "I2C initialized successfully");
226
227     vTaskDelay(pdMS_TO_TICKS(10));
228
229     ESP_ERROR_CHECK_WITHOUT_ABORT(SHT35_heater(false));
230
231     vTaskDelay(pdMS_TO_TICKS(10));
232
233     SHT35_read_and_print_status_register();
234 }
```



Loudness sensor code

loudness_sensor.h

```
1 /*
2  * loudness_sensor.h
3  *
4  * Created on: 13 May 2022
5  * Author: Richard & Tom
6  */
7
8 #ifndef MAIN_LOUDNESS_SENSOR_H_
9 #define MAIN_LOUDNESS_SENSOR_H_
10
11 #include "driver/adc.h"
12 #include "esp_adc_cal.h"
13 #include "esp_log.h"
14 #include "esp_event.h"
15 #include <stdlib.h>
16 #include <math.h>
17
18 #define ADC_ATTEN ADC_ATTEN_6db
19 #define ADC1_CHANNEL ADC1_CHANNEL_0
20 #define SAMPLES 100
21 #define THRESHOLD 700
22 #define MEASURE_PERIOD 10
23
24 esp_err_t adc_config_init(void);
25 bool adc_calibration_init(esp_adc_cal_characteristics_t *adc1_chars);
26 void print_results(uint16_t *raw_ptr, uint16_t *voltage_ptr);
27 void print_results_stat(float *mean_ptr, float *median_ptr, float *std_dev_ptr);
28 void single_one_sec_measurement(esp_adc_cal_characteristics_t *adc1_chars, float *mean, float
 *median, float *std_dev);
29 float mean_array(uint16_t *array, size_t array_size);
30 float std_dev_array(uint16_t *array, size_t size, float mean);
31 float median_array(uint16_t *array, size_t array_size);
32 int compare_func(const void * a, const void * b);
33
34 #endif /* MAIN_LOUDNESS_SENSOR_H_ */
```

loudness_sensor.c

```
1 /*
2  * loudness_sensor.c
3  *
4  * Created on: 13 May 2022
5  * Author: Richard & Tom
6  */
7
8 #include "loudness_sensor.h"
9
```

```

10 static const char *TAG = "ADC";
11
12 esp_err_t adc_config_init(void)
13 /* Cofiguration of the ADC */
14 {
15     esp_err_t err;
16
17     err = adc1_config_width(ADC_WIDTH_BIT_DEFAULT);
18
19     if (err == ESP_OK)
20         err = adc1_config_channel_atten(ADC1_CHANNEL, ADC_ATTEN);
21
22     return err;
23 }
24
25 bool adc_calibration_init(esp_adc_cal_characteristics_t *adc1_chars)
26 /* Initialization function for the calibration of the ADC */
27 {
28     esp_err_t ret;
29     bool cali_enable = false;
30
31     ret = esp_adc_cal_check_efuse(ESP_ADC_CAL_VAL_EFUSE_TP);
32     if (ret == ESP_ERR_NOT_SUPPORTED) {
33         ESP_LOGW(TAG, "Calibration scheme not supported, skip software calibration");
34     }
35     else if (ret == ESP_ERR_INVALID_VERSION) {
36         ESP_LOGW(TAG, "eFuse not burnt, skip software calibration");
37     }
38     else if (ret == ESP_OK) {
39         cali_enable = true;
40         esp_adc_cal_characterize(ADC_UNIT_1, ADC_ATTEN, ADC_WIDTH_BIT_DEFAULT, 0, adc1_chars);
41     }
42     else {
43         ESP_LOGE(TAG, "Invalid arg");
44     }
45
46     return cali_enable;
47 }
48
49 void print_results(uint16_t *raw_ptr, uint16_t *voltage_ptr)
50 {
51     ESP_LOGI(TAG, "Raw data: %d", *raw_ptr);
52     ESP_LOGI(TAG, "Calc voltage: %d mV", *voltage_ptr);
53 }
54
55 void print_results_stat(float *mean_ptr, float *median_ptr, float *std_dev_ptr)
56 {
57     ESP_LOGI(TAG, "Mean: %.2f", *mean_ptr);
58     ESP_LOGI(TAG, "Median: %.2f", *median_ptr);
59     ESP_LOGI(TAG, "Std dev: %.2f", *std_dev_ptr);
60 }
61
62 void single_one_sec_measurement(esp_adc_cal_characteristics_t *adc1_chars, float *mean, float
    *median, float *std_dev)
63 /* Final implementation of the loudness sensor. Outputs the mean, median and standard
    deviation */
64 {
65     uint16_t adc_raw[SAMPLES] = {0};
66     uint16_t voltage[SAMPLES] = {0};
67
68     for (int i = 0; i < SAMPLES; i++)
69     {
70         adc_raw[i] = adc1_get_raw(ADC1_CHANNEL);
71         voltage[i] = esp_adc_cal_raw_to_voltage(adc_raw[i], adc1_chars);
72         vTaskDelay(pdMS_TO_TICKS(MEASURE_PERIOD));
73     }
74
75     float raw_mean = mean_array(adc_raw, SAMPLES);
76
77     ESP_LOGI(TAG, "Raw mean: %.2f", raw_mean);
78

```

```

79  *mean = mean_array(voltage, SAMPLES);
80  *median = median_array(voltage, SAMPLES);
81  *std_dev = std_dev_array(voltage, SAMPLES, *mean);
82  }
83
84  float mean_array(uint16_t *array, size_t array_size)
85  /* Calculate the mean of an array */
86  {
87      float mean = 0.0;
88
89      for (int i = 0; i < array_size; i++)
90          mean += array[i];
91
92      return mean/array_size;
93  }
94
95  float std_dev_array(uint16_t *array, size_t size, float mean)
96  /* Calculate the standard deviation of an array */
97  {
98      float sum_squared_diff = 0;
99      for (int i = 0; i < size; i++)
100          sum_squared_diff += (array[i] - mean)*(array[i] - mean);
101
102      return sqrt(sum_squared_diff/size);
103  }
104
105  float median_array(uint16_t *array, size_t size)
106  /* Calculate the median of an array */
107  {
108      qsort(array, size, sizeof(uint16_t), compare_func);
109
110      if (size % 2 != 0)
111          return (float)array[size/2];
112      else
113          return (float)(array[(size-1)/2] + array[size/2])/2.0;
114  }
115
116  int compare_func(const void * a, const void * b)
117  {
118      // qsort() passes in 'void*' types because it can't know the actual types being sorted
119      // convert those pointers to pointers to int and deref them to get the actual int values
120
121      uint16_t val1 = *(uint16_t*)a;
122      uint16_t val2 = *(uint16_t*)b;
123
124      // qsort() expects the comparison function to return:
125      //
126      //   a negative result if val1 < val2
127      //   0 if val1 == val2
128      //   a positive result if val1 > val2
129
130      return ( val1 - val2 );
131  }

```

main.c

```

1  #include "freertos/FreeRTOS.h"
2  #include "esp_wifi.h"
3  #include "esp_system.h"
4  #include "esp_event.h"
5  #include "nvs_flash.h"
6  #include "driver/gpio.h"
7  #include "freertos/task.h"
8  #include "freertos/queue.h"
9
10 #include "loudness_sensor.h"
11
12 #define MEAS_CNT 250
13
14 void while_loop(esp_adc_cal_characteristics_t *adc1_chars)

```

```

15 {
16     uint16_t adc_raw;
17     uint16_t voltage;
18
19     while (true)
20     {
21         adc_raw = adc1_get_raw(ADC1_CHANNEL);
22         voltage = esp_adc_cal_raw_to_voltage(adc_raw, adc1_chars);
23
24         print_results(&adc_raw, &voltage);
25         vTaskDelay(pdMS_TO_TICKS(10));
26     }
27 }
28
29 void rolling_average(esp_adc_cal_characteristics_t *adc1_chars)
30 {
31     uint16_t adc_raw[SAMPLES] = {0};
32     uint16_t voltage[SAMPLES] = {0};
33     float rolling_average;
34
35     while (true)
36     {
37         for (int i = 0; i < (SAMPLES); i++)
38         {
39             adc_raw[i] = adc1_get_raw(ADC1_CHANNEL);
40             voltage[i] = esp_adc_cal_raw_to_voltage(adc_raw[i], adc1_chars);
41
42             rolling_average = mean_array(voltage, SAMPLES);
43
44             if (rolling_average > THRESHOLD)
45                 ESP_LOGI("ADC", "Rolling average: %f", rolling_average);
46
47             vTaskDelay(pdMS_TO_TICKS(MEASURE_PERIOD));
48         }
49     }
50 }
51
52 void continuous_measurements(esp_adc_cal_characteristics_t *adc1_chars)
53 {
54     float mean, median, std_dev;
55
56     while (true)
57     {
58         single_one_sec_measurement(adc1_chars, &mean, &median, &std_dev);
59
60         print_results_stat(&mean, &median, &std_dev);
61     }
62 }
63
64 void measurements(esp_adc_cal_characteristics_t *adc1_chars)
65 /* Function written for the validation part of the thesis */
66 {
67     uint16_t adc_raw = 0;
68     uint16_t voltage1[MEAS_CNT] = {0};
69     uint16_t voltage2[MEAS_CNT] = {0};
70     uint16_t voltage3[MEAS_CNT] = {0};
71     uint16_t voltage4[MEAS_CNT] = {0};
72
73     for (int i = 0; i < (MEAS_CNT); i++)
74     {
75         adc_raw = adc1_get_raw(ADC1_CHANNEL);
76         voltage1[i] = esp_adc_cal_raw_to_voltage(adc_raw, adc1_chars);
77
78         vTaskDelay(pdMS_TO_TICKS(MEASURE_PERIOD));
79     }
80     for (int i = 0; i < (MEAS_CNT); i++)
81     {
82         adc_raw = adc1_get_raw(ADC1_CHANNEL);
83         voltage2[i] = esp_adc_cal_raw_to_voltage(adc_raw, adc1_chars);
84
85         vTaskDelay(pdMS_TO_TICKS(MEASURE_PERIOD));

```

```
86 }
87 for (int i = 0; i < (MEAS_CNT); i++)
88 {
89     adc_raw = adc1_get_raw(ADC1_CHANNEL);
90     voltage3[i] = esp_adc_cal_raw_to_voltage(adc_raw, adc1_chars);
91
92     vTaskDelay(pdMS_TO_TICKS(MEASURE_PERIOD));
93 }
94 for (int i = 0; i < (MEAS_CNT); i++)
95 {
96     adc_raw = adc1_get_raw(ADC1_CHANNEL);
97     voltage4[i] = esp_adc_cal_raw_to_voltage(adc_raw, adc1_chars);
98
99     vTaskDelay(pdMS_TO_TICKS(MEASURE_PERIOD));
100 }
101
102 printf("Voltage:\n");
103 for (int i = 0; i < MEAS_CNT; i++)
104     printf("%u\n", voltage1[i]);
105 for (int i = 0; i < MEAS_CNT; i++)
106     printf("%u\n", voltage2[i]);
107 for (int i = 0; i < MEAS_CNT; i++)
108     printf("%u\n", voltage3[i]);
109 for (int i = 0; i < MEAS_CNT; i++)
110     printf("%u\n", voltage4[i]);
111 }
112
113 void app_main(void)
114 {
115     esp_adc_cal_characteristics_t adc1_chars;
116     bool cali_enable = adc_calibration_init(&adc1_chars);
117     if (cali_enable)
118         ESP_LOGI("ADC", "Calibrated correctly");
119
120     ESP_ERROR_CHECK_WITHOUT_ABORT(adc_config_init());
121
122     // rolling_average(&adc1_chars);
123
124     measurements(&adc1_chars);
125 }
126 }
```


Bibliography

- [1] Kshitij Shinghal et al. "IOT Based Modified Hybrid Blended Learning Model for Education". In: *2020 International Conference on Advances in Computing, Communication Materials (ICACCM)*. 2020, pp. 229–232. DOI: 10.1109/ICACCM50413.2020.9213049.
- [2] La Vonne Fedynich. "Teaching beyond the classroom walls: The pros and cons of cyber learning." In: *Journal of Instructional Pedagogies* 13 (2013).
- [3] V Giampietro M. Detyna R Sanchez Pizani and E.J. Dommett. "Hybrid flexible (HyFlex) teaching and learning: climbing the mountain of implementation challenges for synchronous online and face-to-face seminars during a pandemic". In: (2022). URL: <https://doi.org/10.1007/s10984-022-09408-y>.
- [4] P.C. Blumenfeld J.A. Fredricks and A.H. Paris. "School Engagement: Potential of the Concept, State of the Evidence." In: (2004). URL: <https://doi.org/10.3102/00346543074001059>.
- [5] Annelies Raes. "Exploring Student and Teacher Experiences in Hybrid Learning Environments: Does Presence Matter?" In: (2021). URL: <https://link.springer.com/content/pdf/10.1007/s42438-021-00274-0.pdf>.
- [6] Jo Ann Lee & Patricia Ellen Busch. "Factors Related to Instructors Willingness to Participate in Distance Education." In: *The Journal of Educational Research* (2005). URL: <https://doi.org/10.3200/JOER.99.2.109-115>.
- [7] *The Netherlands ranks among the EU top in Digital Skills*. Feb. 2020. URL: <https://www.cbs.nl/en-gb/news/2020/07/the-netherlands-ranks-among-the-eu-top-in-digital-skills>.
- [8] Dosheela Devi Ramlowat and Binod Kumar Pattanayak. "Exploring the Internet of Things (IoT) in Education: A Review". In: *Information Systems Design and Intelligent Applications*. Ed. by Suresh Chandra Satapathy et al. Singapore: Springer Singapore, 2019, pp. 245–255. ISBN: 978-981-13-3338-5.
- [9] Sciforce. "Internet of Things for the Classroom". In: (2019). URL: <https://www.ietfforall.com/internet-of-things-classroom>.
- [10] *Logitech Rally Kit(960-001218) Datasheet*. <https://cdn.webshopapp.com/shops/91456/files/319561589/logitech-rally-product-data-sheet.pdf>. Accessed: 31/05/2022.
- [11] *vc520pro2 datasheet*. <https://www.averusa.com/business/downloads/vc520pro2-datasheet.pdf>. Accessed: 30/05/2022.
- [12] *A BRIEF GUIDE TO AUDIO CONFERENCING IN HIGHER ED CLASSROOMS*. https://www.parmetech.com/wp-content/uploads/2021/09/Guide-to-Audio-Conferencing-Classrooms_-Education_-Cobranded.pdf. Accessed: 30/05/2022.
- [13] *Chung Yuan Christian University Establishes World's First ViewSonic Hybrid Teaching Classroom*. https://www.viewsonic.com/nl/education/pdf/Case_Study_Chung_Yuan_Christian_University_Establishes_Worlds_First_ViewSonic_Hybrid_Teaching_Classroom.pdf. Accessed: 30/05/2022.
- [14] John M. Eargle. *Handbook of recording engineering*. 2th ed. Van Nostrand Reinhold, 1992.
- [15] *Microphone Mist™ technology*. URL: <https://www.nureva.com/audio-conferencing/microphone-mist?hsCtaTracking=fda44768-b84e-42fa-a278-29cacadabc92%5C%7Cb2641c5a-e093-4d62-ba10-5acb039a9f8f>.
- [16] *echo-pt*. URL: <https://dynamount.com/products/echo-pt/>.
- [17] *NT1-A Instruction Manual*. https://www.bhphotovideo.com/lit_files/589217.pdf. Accessed: 4/06/2022.

- [18] Arthur Fox. *What Is A Good Signal-To-Noise Ratio For A Microphone?*
- [19] *ESP32C3 Family Datasheet*. URL: https://www.espressif.com/sites/default/files/documentation/esp32-c3_datasheet_en.pdf.
- [20] Chris Lyons. *Tell Me About: Mic and Line Level*. URL: <https://www.shure.com/en-MEA/conferencing-meetings/ignite/tell-me-about-mic-and-line-level>.
- [21] *Grove 2-Channel SPDT Relay*. URL: https://wiki.seeedstudio.com/Grove-2-Channel_SPDT_Relay/.
- [22] Focusrite. *Scarlett 2i2*. URL: <https://focusrite.com/nl/usb-audio-interface/scarlett/scarlett-2i2>.
- [23] Audacity. *Free, open source, cross-platform audio software*. URL: <https://www.audacityteam.org/>.
- [24] P.C. Loizou. "Mimicking the human ear". In: *IEEE Signal Processing Magazine* 15.5 (1998), pp. 101–130. DOI: 10.1109/79.708543.
- [25] *Haptic Motor Driver Hook-Up Guide*. URL: <https://learn.sparkfun.com/tutorials/haptic-motor-driver-hook-up-guide/erm-and-lra-motors>.
- [26] *small vibration motors*. URL: <https://www.nfpmotor.com/small-vibration-motors>.
- [27] *Grove vibration motor*. URL: https://wiki.seeedstudio.com/Grove-Vibration_Motor/.
- [28] Pawel Wargocki and David P. Wyon. "Ten questions concerning thermal and indoor air quality effects on the performance of office work and schoolwork". In: *Elsevier* (2017).
- [29] Hakpyeong Kim and Taehoon Hong. "Determining the optimal set-point temperature considering both labor productivity and energy saving in an office building". In: *Elsevier* (2020).
- [30] David P. Wyon. "The effects of indoor air quality on performance and productivity". In: *Indoor Air* (2004).
- [31] Pawel Wargocki and David P. Wyon. "Effects of HVAC On Student Performance". In: *ASHRAE Journal* (2006).
- [32] *Digital Temperature & Humidity Module DHT11 User Manual*. ASAIR. URL: <https://www.maritex.com.pl/product/attachment/147097/e3ba4b862ce252dac75137c2b42b983f>.
- [33] Thomas Liu. *Digital-output relative humidity & temperature sensor/module DHT22*. Aosong Electronics Co., Ltd. URL: <https://www.sparkfun.com/datasheets/Sensors/Temperature/DHT22.pdf>.
- [34] *Digital I2C Humidity and Temperature Sensor TH02*. Rev. 0.1. Seeed Technology Co., Ltd. URL: <https://www.digkey.co.il/htmldatasheets/production/3396943/0/0/1/th02.html#pdf>.
- [35] *Datasheet SHT3x-DIS Humidity and Temperature Sensor*. 3rd ed. Sensirion AG. Aug. 2016. URL: https://www.mouser.com/datasheet/2/682/Sensirion_Humidity_Sensors_SHT3x_Datasheet_digital-971521.pdf.
- [36] Daniel J. Exeter et al. "Student engagement in very large classes: the teachers' perspective". In: *Studies in Higher Education* 35.7 (2010), pp. 761–775. DOI: 10.1080/03075070903545058. eprint: <https://doi.org/10.1080/03075070903545058>. URL: <https://doi.org/10.1080/03075070903545058>.
- [37] Alessandro Vinciarelli et al. "Social signal processing: state-of-the-art and future perspectives of an emerging domain". In: *Proceedings of the 16th ACM international conference on Multimedia*. MM '08. Vancouver, British Columbia, Canada: Association for Computing Machinery, Oct. 2008, pp. 1061–1070. ISBN: 9781605583037. DOI: 10.1145/1459359.1459573. URL: <https://doi.org/10.1145/1459359.1459573>.
- [38] Seeed Studio. *Grove - Loudness Sensor*. URL: https://wiki.seeedstudio.com/Grove-Loudness_Sensor/.