Jori-Pekka Harju

Designing and Building a Device for Monitoring Office Conditions + IoT

Thesis
Spring 2020
Seinäjoki University of Applied Sciences
Automation

SeAMK
SEINÄJOEN AMMATTIKORKEAKOULU
SEINÄJOKI UNIVERSITY OF APPLIED SCIENCES

SEINÄJOKI UNIVERSITY OF APPLIED SCIENCES

Thesis abstract

Faculty: School of Technology

Degree Programme: Automation Engineering

Specialisation: Machine Automation

Author: Jori-Pekka Harju

Title of thesis: Designing and Building a Device for Monitoring Office Conditions + IoT

Supervisor: Hietamäki Marko

Year: 2019 Number of pages: 38

The focus of the thesis project was to design and build a prototype for an inexpensive, but reliable device, which monitors office conditions. This device should be easy to move, and it should have an internet connection.

The office conditions to be measured were illuminance, temperature, relative humidity and noise level. The noise level measuring was left out for possible future development.

This thesis studied the phases of designing and building a prototype device for monitoring office conditions, why the device is necessary and what are the recommendations for offices. The possible future development options were also considered in the end of the thesis. The project was designed for Alte Oy and the device will be put in use at their offices.

Keywords: Arduino, office conditions, IoT, indoor weather station, prototyping

SEINÄJOEN AMMATTIKORKEAKOULU

Opinnäytetyön tiivistelmä

Koulutusyksikkö: Seinäjoen Ammattikorkeakoulu

Tutkinto-ohjelma: Automaatiotekniikka

Suuntautumisvaihtoehto: Koneautomaatio

Tekijä: Jori-Pekka Harju

Työn nimi: Laitteen suunnittelu ja rakentaminen toimisto-olosuhteiden tarkkailuun + IoT

Ohjaaja: Hietamäki Marko

Vuosi: 2019 Sivumäärä: 38

Tämä opinnäytetyöprojekti keskittyi edullisen, mutta luotettavan laitteen prototyypin suunnitteluun ja rakentamiseen. Laitteen oli tarkoitus seurata toimisto-olosuhteita. Laitteen tulisi olla helposti liikutettava ja siinä piti olla internetyhteys.

Mitatut toimisto-olosuhteet olivat valonmäärä, lämpötila, suhteellinen kosteus ja äänenvoimakkuus. Äänenvoimakkuus jätettiin kuitenkin odottamaan mahdollista myöhempää kehittelyä.

Tässä opinnäytetyössä tutkittiin toimisto-olojen seurantalaitteen suunnittelu- ja rakentamisvaiheita, ja selvitettiin myös, miksi laite on tarpeellinen ja mitä suosituksia on olemassa toimistoille. Mahdollisia tulevia jatkokehittelyvaihtoehtoja mietittiin opinnäytetyön lopussa. Projekti suunniteltiin Alte Oy:lle ja laite tulee heidän toimistoihinsa käytettäväksi.

Asiasanat: Arduino, toimisto-olosuhteet, IoT, sisätilan sääasema, tuotekehittely

TABLE OF CONTENTS

Tables, Figures and Pictures

Terms and Abbreviations

%RH	Relative humidity percentage. Unit for measuring relative humidity.
3G	Third generation, the 3rd generation of mobile network technology.
AES	Advanced Encryption Standard. AES is used to encrypt and protect classified data.
Bluetooth	Short-range radio technology.
DHCP	Dynamic Host Configuration Protocol. A network management protocol.
DSE	Display Screen Equipment.
E_m	Maintained illuminance.
Flash memory	Flash memory is the most popular non-volatile, rewritable storage chip.
GHz	Gigahertz. Unit of frequency.
I2C	Inter-Integrated Circuit.
IDE	Integrated development environment.
IoT	Internet of Things. a system where devices communicate with each other over the network.
IP	Internet Protocol. Communication protocol.
kg	kilogram. Unit of mass.
kohm	kilo ohm. Unit of electrical resistance.
LTE	Long Term Evolution, the 4th generation of mobile network technology.
lx	lux. Derived unit of illuminance.

mA	milli Amperes. Unit of electrical current.
MB/s	Megabytes per second. Unit of download and upload speed
mm	millimetre. Unit of length.
OLED	Organic Light Emitting Diode.
R_a	Minimum colour rendering indices.
RAM	Random Access Memory.
RTC	Real-time clock.
s	second. Unit of time.
SCL	Serial clock.
SD	Secure Digital. A memory card format.
SDA	Serial data.
TKIP	Temporal Key Integrity Protocol. A security protocol for wireless networks
UART	Universal Asynchronous Receiver Transmitter. A circuit for transmitting and receiving serial data
UGR_L	Unified Glare Rating limit.
U_o	Minimum illuminance uniformity.
V	Voltage. Unit of potential difference in electric circuits.
VCC	Voltage common collector.
W	Watt. Unit of power.
WEP	Wired Equivalent Privacy. A security algorithm for wireless networks.
Wi-Fi	Wireless networking based on radio waves.

WPA/WPA2	Wi-Fi Protected Access. Security certification programs to secure wireless computer networks.

1 Introduction

When technology develops, more and more work is transferred from field to offices, thus increasing work in the offices. This phenomenon will bring new challenges to companies. One of these challenges is office conditions which are being developed and enhanced all the time. In bad work conditions work atmosphere is poor, and people get stressed and sick more often. When office conditions are implemented well, work is pleasant and there is less work wearing. This will raise people's work morale and help companies to work more efficiently. New methods are constantly being implemented to improve and measure office conditions. Lots of effort is being put into improving work ergonomics and keeping the temperature and lighting in the office as pleasant as possible, but less attention is paid to measuring actual values and comparing how they change in different conditions.

New technology makes it easy for companies to measure values of lighting, temperature, noise and humidity but there is no low-cost device in the market meant for measuring all these values to monitor and improve office conditions. Monitoring these conditions in different circumstances will give valuable data, which can be analysed and used for enhancing office conditions. These different circumstances can be, for example, the number of people in the office, weather outside, the time of the day and how many computers there are turned on. One of the technologies that can be used for these measurements is Arduino microcontroller with necessary sensors, a network expansion and a connection to an IoT cloud.

This thesis is done to see if it is possible to develop an inexpensive high-end monitoring device for improving conditions in offices. The device will monitor temperature, humidity, noise and lighting in the office. This information is then sent to an IoT-cloud, which will deduce if the office conditions are good or not. Arduino was chosen to be the backbone of this project and free Thingboard IO IoT-cloud was chosen to be the platform for analysing and visualising the data. The thesis is done in collaboration with Alte Oy.

1.1 Arduino

Arduino is an inexpensive open source programmable microcontroller which can be programmed to get data from sensors and to process, save and share it to different outputs, for

example, to a cloud or to a screen or even changed into a movement by using different actuators. Arduino was originally a research project at the Interaction Design Institute of Ivrea in the early 2000s. Its main purpose was to help students to learn about programming and electronics, but later engineers around the world noticed its potential. They started using Arduino in their own projects and started developing their own codes for it. This led it to be widely known and developed the Arduino project even further. (Arduino 2019)

1.2 Alte Oy

In 1969 Tarmo Soralahti founded a company in the city of Raahe. The company was called Plan Verks which was a predecessor for Alte Oy. The name comes from words Planera och verkstella which means planning and manufacturing. In 1970 the company was registered and in 1973 the name was changed to Alte and the company form was changed to a joint-stock company. The founders of this new joint-stock company were Tarmo and Seppo Soralahti. The name Alte comes from words Allaskoneisto ja teräsrakenne which means tank machinery and steel structure.

In 1980 Alte founded its first branch office in Hyvinkää. Five employees moved from Raahe to Hyvinkää and five more employees were recruited from Hyvinkää. Later Alte started to buy other companies and established more branch offices.

In 2008 Alte bought a company called TSS Group Oy. TSS Group's specialty was in electrical planning and it was located in Tampere. Alte was sold to a French company called Alten in 2014. Alte kept its name but is now widely known as Alten Finland Oy. In 2019 Alte had 11 branch offices and the company is still growing steadily. Alten group is a world leader in Technology and Engineer consulting. It was formed 30 years ago and is based over 25 countries. (Alten Finland 2019)

2 Related office conditions and guidelines

In this chapter we are going through some guidelines and standards regarding the office conditions related to this thesis. These conditions are mostly planned just in theory and sometimes measured after building but they are rarely measured after the offices have been taken in use.

2.1 Illuminance

Illuminance is a value which measures the amount of light on a surface area. This is used to observe and plan the sufficient lighting for each given task. Different illuminance areas are represented in a figure 1.

> "The average illuminance for each task shall not fall below the value given in table 2, regardless of the age and condition of the installation." (SFS-EN 12464-1:2011, 17).

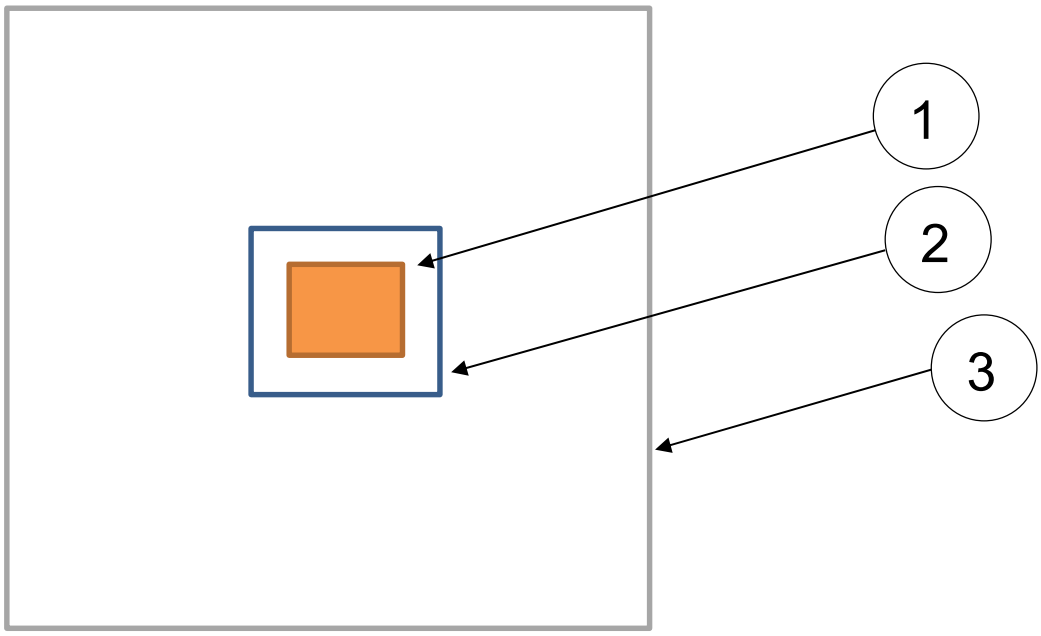


Figure 1: Illuminance areas

Key:

1. task area
2. immediate surrounding

3. background area

> “Large spatial variations in illuminances around the task area can lead to visual stress and discomfort.” (SFS-EN 12464-1:2011, 21).

> “The illuminance of the immediate surrounding area shall be related to the illuminance of the task area and should provide a well-balanced luminance distribution in the visual field.” This is shown on table 1. (SFS-EN 12464-1:2011, 21).

Table 1: Relationship of Illuminances on immediate surrounding to the illuminance on the task area

Illuminance on the task area E_{task} lx	Illuminance on immediate surrounding areas lx
≥ 750	500
500	300
300	200
200	150
150	E_{task}
100	E_{task}
≤ 50	E_{task}

> “In indoor [workplaces], particularly those devoid of daylight, a large part of the area surrounding an active and occupied task are needs to be illuminated. This area known as the “background area” should be a band at least 3m wide adjacent to the immediate surrounding area within the limits of the space and it shall be illuminated with a maintained illuminance of 1/3 of the value of the immediate surrounding area.” (SFS-EN 12464-1:2011, 23).

Table 2: Recommended illuminances depending on the task (SFS-EN 12464-1:2011, 55)

Type of area, task or activity	Illuminance E_m lx	Unified glare rating UGR_L –	Uniformity U_O –	colour rendering index Ra –	Specific requirements
Filing, copying, etc.	300	19	0,40	80	

Writing, typing, reading, data processing	500	19	0,60	80	DSE-work
Technical drawing	750	16	0,70	80	
CAD workstations	500	19	0,60	80	DSE-work
Conference and meeting rooms	500	19	0,60	80	Lighting should be controllable
Reception desk	300	22	0,60	80	
Archives	200	25	0,40	80	

On DSE areas lighting should be selected and located in a way that it does not cause any bright reflections from screens or other reflective surfaces. Reflections can be discomforting and cause fatigue to eyes. (SFS-EN 12464-1:2011, 31.)

2.2 Temperature

When assessing temperatures, the heat production of bodies should be taken into consideration. This means clothing, how heavy the work is and what kind of physical conditions there are. These physical conditions are air temperature, heat radiation, air velocity and humidity. (Työturvallisuuskeskus 2019)

Table 3: Recommended working temperatures (Työturvallisuuskeskus 2019)

Recommended temperatures in workspaces	
Work description	**Recommended temperature**
Very light work	21-25°C
light work	19-23°C

heavy work	17-21°C
Very heavy work	12-17°C

An employer's responsibility to make sure that the temperature of a working environment does not rise above +28°C when outside temperature is under +25°C. When the outside temperature is more than +25°C and the working temperature rises above +28°C, 10 – 15 minute breaks should be allowed every hour. (Työturvallisuuskeskus 2019)

2.3 Humidity

Humidity inside buildings should not be high enough to cause any damage or to let bacteria or fungus grow. Relative humidity of a room should be monitored with a reliable humidity meter. In winter humidity is never too high, unless caused by the actions of a human or a malfunctioning air conditioner. Relative humidity is usually under 40% but it can be even under 20% on extremely cold days. The suitable relative inside humidity for buildings in winter is 20-40%. In summer relative humidity varies between 50 - 70% and it is affected by the humidity coming from outside. (Hengitysliitto 2019)

3 Objective of the thesis

The objective of this thesis was to design and create a working prototype for measuring different kind of environmental data and sending this data to an IoT-cloud. The device would be used to measure office conditions either in Alte Oy's offices or in their server room. The idea for the thesis formed when I noticed that there was dry air in offices and the room temperature was too high. After talking about this idea with my supervisors, the plan for building the device got in motion.

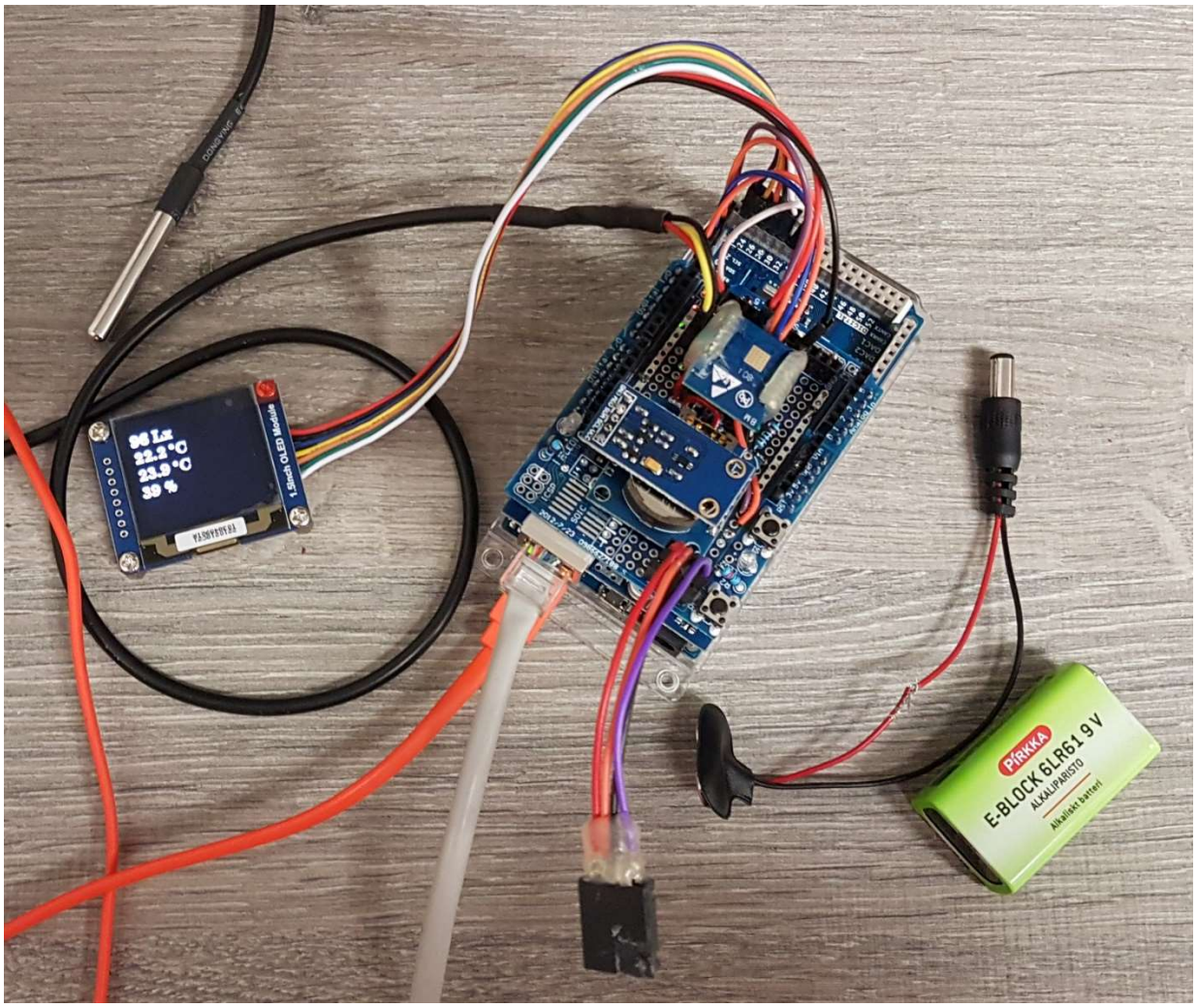
3.1 Device description

The device measures three different environmental sensor values, temperature, relative humidity and light intensity level. After measuring, the values are displayed on an OLED-display and on graphs on IoT-cloud. In addition to this, the device will save data every minute to a micro SD-card which can be read later by using an SD-card reader and a computer or any device, which supports reading .txt-files. The device can be connected by using an Ethernet cable or Wi-Fi and it can be powered through a 5V micro USB port or a 7V - 12V input port. Using a micro USB port is recommended for saving energy.

Table 4: Device specifications

Technical Spec.	Details	Value	Unit
Power usage:	Battery (Ethernet)	2,97	W
	Battery (Wi-Fi)	3,6	W
	USB (Ethernet)	not measured	W
	USB (Wi-Fi)	not measured	W
Input voltage:	Battery	7-12	V
	USB	5	V
Operating temperature		-40 ~ 85	°C
Micro-SD(HC) card size	Fat32	up to 32GB	
	Fat16	up to 4GB	
Ethernet	Speed	100	MB/s

Wi-Fi	Protocols	802.11 b/g/n	
	Frequency	2.4	GHz
	Security*	WPA/WPA2	
	Protection	WEP/TKIP/AES	
Update interval to IoT-cloud	Ethernet	1	s
	Wi-Fi	1.5	s
Temperature measurement accuracy	AM2320	± 0.5	°C
	DS18B20	± 0.5	
Temperature measurement resolution	AM2320	0.1	°C
	DS18B20	0.1	°C
Humidity measurement accuracy		± 3	%RH
Humidity measurement resolution		0.1	%RH
Light intensity measurement accuracy		± 20	%
Light intensity measurement resolution		1	lx
Weight (without USB or Battery)		0,153	kg
Dimensions (L x W x H)	Without wires	100 x 55 x 60	mm



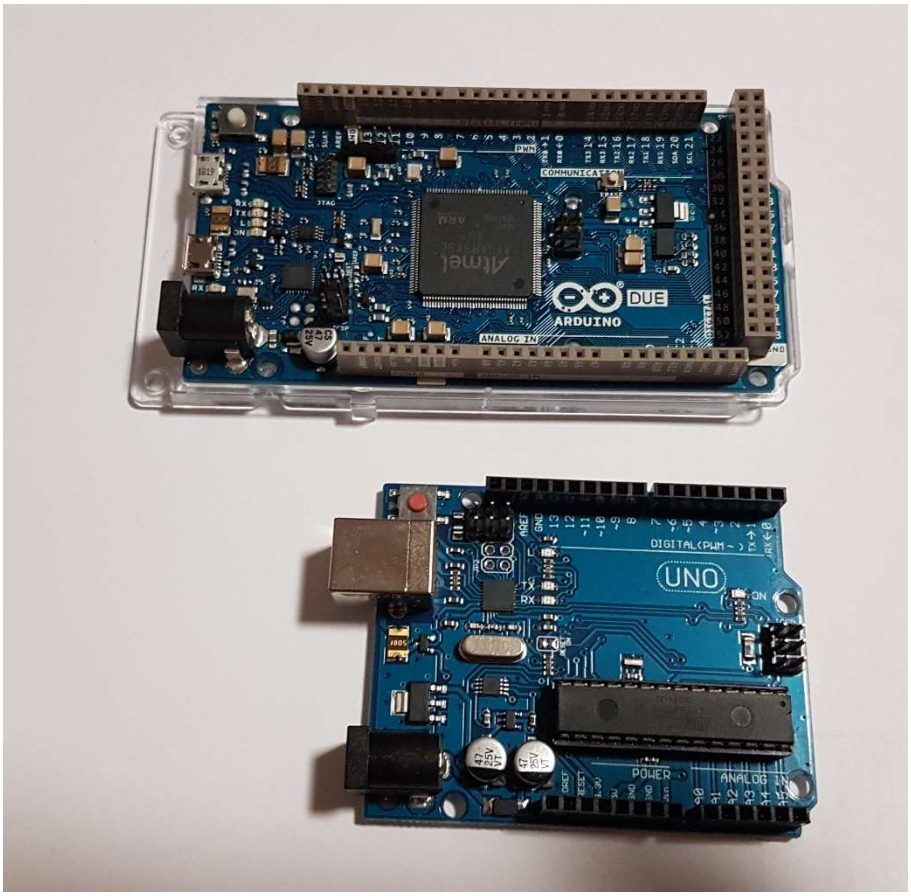
Picture 1: Finished prototype without an enclosure

3.2 Choosing the components and programs

Components were chosen carefully based on their availability, price and delivery time. The suppliers were chosen to be from Finland due their fast delivery time and trustworthiness. In the end, two suppliers were chosen for their large selection: Ahaa Elektroniikka and Triopak. Later the Arduino Due was ordered from an official Arduino online store. This was done to avoid any problems with combability. The product being on sale was another reason for choosing the official store.

3.2.1 Microcontroller

Arduino Uno was chosen to be the base and brains, but after a while it was obvious that Arduino Uno did not have enough memory for this project, and it was replaced by a more powerful unit, Arduino Due.



Picture 2: Arduino Due and Arduino Uno copy

Arduino based boards were chosen because of their popularity, price and easiness for further developing. Other candidates for the project were Raspberry Pi and ESP8266, but Raspberry Pi did not have same developing possibilities to bring down the costs of future devices, and ESP8266 was still a less popular and less used microcontroller, meaning there is not a big community behind it and it does not have that many sample projects. These facts lead for choosing Arduino.

3.2.2 Sensors

Sensors chosen for the project were Aosong Electronics' AM2320 for measuring temperature and relative humidity, ROHM Semiconductor's BH1750FVI for measuring light intensity levels, and DS18B20 for measuring temperature. These sensors were chosen based on their availability, popularity and price. Later noise sensor was also purchased along Arduino Due, but this sensor did not have sensitivity adjustment and was discarded from the project.

3.2.3 Other components

The project also needed other components to work. A 1,5" 128x128 monochrome OLED-display was chosen for displaying data without using any connection to the internet or to a computer. It was chosen for its price, size and ability to display lots of information on same screen.

The Wi-Fi-module Wroom02 was chosen to connect the device to wireless connections. It was chosen for its price and size. An ethernet-shield with a W5100 chip was chosen for connecting the device to the internet through an ethernet cable and to save data to an SD-card. It was chosen because how easy it is to use and to implement to the Arduino.

A prototype-shield was chosen to help creating the prototype with a little effort and tools. The real-time clock DS1307 was chosen for recording time. It was chosen for its price and popularity but also because it was easy to implement to the project.

3.2.4 Accessories

Multiple different accessories were needed to test the device but also to make it work. A breadboard and wires with Dupont-heads were needed to connect components together and test them.

A battery CR2032 for the real-time clock was needed to keep the clock running after the input power was cut. A micro SD-card was needed to back-up the data in case the connection to an IoT-server disconnects. Resistors were needed to make pull-up and pull-down connections between data lines and a VCC line or a ground line. Other accessories for testing were a 9V battery and a connector between the battery and the Arduino.

3.2.5 Program for programming

A free Arduino IDE was chosen for programming for its combability for Arduino boards and for the Wroom-02 Wi-Fi-module. The IDE user community size also affected to a decision making. Later the Arduino IDE was found out to be perfect for this project because it was simple and clear.

3.2.6 IoT-cloud

After making a list of applicable IoT-clouds, which were made for monitoring, they were compared by making a list of their disadvantages and advantages. After making a list of their disadvantages and advantages, the grade was given for all of them and the free Thingboard IO was chosen as a platform for this thesis. Advantages for this IoT-cloud were that it is completely free, it has good manuals and is tested with a similar Wi-Fi-module and a microcontroller, also the visualization was made easy to make, and the graphic quality was great.

Picture 3: Real-time monitors

4 Sections of the project

The project can be divided to six major sections. These sections are: Planning, coding, soldering, documenting, creating a manual and making an enclosure. Documenting took most of the time and coding took the second most. Creating an enclosure for the device by 3D modelling and printing took least of the time.

4.1 Planning

Planning started after getting a permission from the school and the client. First thing was to make a schedule for the project and then thinking what criteria there are for the device, suppliers and components. Criteria for the components, suppliers and device were price, fast delivery and easy to develop further if decided so. The next step was to think what components are needed for the device. This was done by browsing different websites to see what components there are available online and thinking, which of those are needed.

After all that, two suppliers were chosen based on their product selection and delivery time. After ordering products, it was time for searching data and example codes for products. For all products there were an example program made by either the manufacturer or the user community. This helped a lot for designing and making the code for the device.

Finding how to wire the components to the Arduino was the next thing, and online there were lots of information regarding wiring. Some information were found from manufacturers' websites and some were made by user community, but some wirings were not documented well enough in anywhere and those wirings were needed to be tested without guides.

4.2 Coding

Coding started by trying example codes for every component separately and selecting code pieces from examples that were needed for the project but also writing some new code. After there was a code for almost every component, combining them started.

Combining was started by using the code of OLED-display as a base because it was the easiest way to start building up the code. Every time you added a piece of code to the base code, you could test if it works or not by printing it to the OLED-display and to the Arduino IDE's serial monitor. After combining the OLED-display's and the SD-card's code pieces together, it was obvious that the memory in the Arduino Uno was not going to be enough. This led for ordering the Arduino Due.

Using the Due created new problems and some changes were needed to do inside of some libraries. These problems are explained later in the chapter "Problems that occurred in the project and solutions to them". After everything else was combined to the base code it was time to connect the Wi-Fi module, which was left out because it needed some soldering done before it could be tested or used.

Coding the Wi-Fi module proved to be a lot more difficult than it was anticipated. Getting the module to connect to a Wi-Fi as a standalone and with the Arduino was easy but getting it to connect the IoT-cloud was not easy by giving AT commands using the Arduino and this never worked. The module was needed to program with a custom program. Using a basic library for MQTT-connection did not keep the connection connected to the IoT-cloud for unknown reason. This was a common problem acknowledged by the user community and the employers of the IoT-cloud company and they had made a specific library and a code for their IoT-cloud.

After the Wi-Fi module kept the connection to the IoT-cloud connected, trying to write the data through serial ports from the Arduino to the Wi-Fi module was done. When all of these were done, error messages and descriptions were created by coding so the user would know when there is an error on and what the error is. Testing the device and fixing coding errors that had been missed while writing the code was done after this. Creating error messages took surprisingly long time because there were so many different situations to test. For example, unexpected power loss, sensor missing, etc.

4.3 Soldering

In the project there was just a little soldering done and the first thing which was soldered was wires to the Wi-Fi module so it could be tested and programmed by using the Arduino Due, the

Arduino IDE and the breadboard with some wires and resistors. After testing and programming the Wi-Fi module it was time to solder it to a prototype shield.

Only few components were soldered to the prototype shield. These were the RTC, the Wi-Fi module and the temperature & humidity sensor. Later it was found out that the temperature & humidity sensor should be unsoldered and solder it to wires with a connector to the prototype shield. In addition, connectors for the temperature sensor and the light sensor were soldered to the prototype shield.

4.4 Documenting

The documentation was done throughout the project. The first documentations done were a research plan and a time schedule. Some documentation was done alongside the project to make sure nothing is forgotten.

Documents done alongside were pieces of example codes, wiring layouts, an assembly picture, wiring tables, a diary, an error list and a list of improvement ideas. Then there were documentation done after the device was ready. These documents are a Thesis report, a user manual and a report for actual time usage.

4.4.1 Research plan and time schedule

The research plan and the time schedule were made for all parties, the client, the thesis writer and the school. This was done to give an idea what is researched, why it is researched, how it is researched, how long it will take and how it will be used.

4.4.2 Wiring tables, wiring layouts and list for improvement ideas

Wiring tables, wiring layouts, a list for improvement ideas were done to keep a future development and projects in mind. This way a person doing development can check how things are wired and what ideas came up while doing the project.

4.4.3 An assembly picture, an error list and a user manual

An assembly picture, an error list and a user manual were done to keep the user in mind. The error list is written in a manual so user can read what error codes mean if there is any. The assembly picture is made for a case that user needs to disassembly the device and put it together again.

4.4.4 The thesis report

The thesis report was the last document to write and was done by using the knowledge got from the project, Standards and recommendations from online. Documents done before made it easier to write the thesis report.

4.4.5 Other documents

Other documents were made for helping to make the device and later to help making the thesis report. For example documents for sample codes for each device were made, and comparison documents were also created.

4.5 Creating a manual

The manual is part of the project. It is meant to be simple enough for a basic user to understand but thorough to go through all basic operations. The manual goes through technical specifications, use of the device and what to do when problems occur.

Making the manual was not easy. There were many things which needed to be taken into consideration and most of the things are obvious in building phase but not for the basic end user, who does not know anything about the device. Creating the manual was done by thinking what information the end user might need.

The manual writing was done after the device was complete for making it easier to make. After completing the device, all variables are known and the knowledge from making the device made it easier to write more thorough manual.

4.6 Creating an enclosure

After the final revision from the device has been made, it was time to design a housing for the device. This housing should take sensors into account for example, not covering incoming light from the light sensor. Nowadays it is easy to make and design a housing by using 3D-modeling and printing. This is inexpensive and easy way to make prototypes to your liking.

The enclosure was decided to be done by 3D modelling and printing. This was decided, to make a better fit for the device. The program used for the 3D modelling was the Autodesk's Autocad. The Autocad was chosen because there was already a license for it. Enclosures were printed by a third party from given 3D models. Two models were created during the thesis. The first model was just for testing the 3D printing quality and accuracy without any connector holes. Connector holes were made afterwards by drilling and cutting to see how the device was fitting to the enclosure model. Second model was a finished model with holes and fixes in measurements.

Picture 4: Finished prototype

5 Problems that occurred in the project and solutions to them

Throughout the project multiple problems occurred and all of them took their own time to solve. For most of them, there was a solution to be found online but some of them needed to be solved without any help. Most of the problems were with the code but also many other problems occurred.

5.1 Problems with codes

This chapter goes through some problems caused by the code and how those were solved. With some of the coding problems it took a lot of time to find out why the problem occurred and many trials and errors and some research online to solve the problem. Sometimes there were no hints online for the problem and then everything had to be found out by trying.

5.1.1 Display library

When the project had advanced to the point of testing the OLED-display and the OLED manufacturer's example code was tested, it was obvious that it used a way too large library for this project, and that it would thus take most of the memory in Arduino Uno. This led to searching for other libraries, but there were not many libraries made for the resolution of 128x128 which would not take too much space from the microcontroller's memory. One of these libraries was Adafruit's Adafruit_SSD1351. This library did not work with our monochrome display and it only printed random lines. This was because it was made for colour displays. Another option was to use Olikrauses' U8g2 library. This library worked well with our OLED-display.

5.1.2 Connecting to network

The problem in connecting to the client's network was that the router only accepted connections through DHCP-requests. This was unexpected and for not knowing that network used a DHCP-protocol only, took a lot of time and effort. This was found out by driving different example programs and the only program that worked was getting the IP through a DHCP-request.

By using this knowledge, it was easy to make a connection to Ethernet using the DHCP-protocol.

5.1.3 Real-time clock was not responding

Real-time clock worked like a dream with Arduino Uno but when the microcontroller was changed to Arduino Due, the real-time clock stopped working. The reason for this was in the library used for the clock. It was trying to form a connection through the SCL1 and SDA1 ports but in this project, it was more reasonable to connect to other I2C-ports: 20 and 21. This was solved by commenting one line out from the library by using // in front of the line, in case there will be a need to change it back to the SCL1 and SDA1 ports.

5.1.4 Light sensor was not working properly

The testing light sensor worked fine alone, and it also worked fine before the real-time clock was added. After the real-time clock had been added the light sensor stopped working and froze every time to the first value measured when the device was turned on. For making sure that this was caused by the code of the real-time clock, the code of the real-time clock was commented out from rest of the project code by using /* in front and */ after the real-time clock code and the program was tested again. This time the light sensor worked again. The solution to this problem was to use JeeLabs' RTC library instead of Arduino's Wire library.

5.1.5 Ethernet connection timeout was too slow

When the program tried to connect to the internet, it waited over a minute before it timed out. This slowed the whole program unnecessarily. This was solved by copying Arduino's Ethernet library, changing its name and modifying all other libraries to refer to this copy instead of the original library. After doing all this, the next thing was to change a few lines in the copy. These lines were found from Ethernet.h file and were the following:

```
static int begin(uint8_t *mac, unsigned long timeout = 5000, unsigned long responseTimeout = 4000);
int beginWithDHCP(uint8_t *, unsigned long timeout = 5000, unsigned long responseTimeout = 4000);
```

original values for both lines were "timeout = 60000" and "responseTimeout = 4000"

5.1.6 Maintaining MQTT-connection through Wi-Fi

There was a problem when using a general pubSubClient library for connecting to the IoT-cloud through Wi-Fi. By using this library, it was possible to connect the Thingsboard IoT-server, but it could not keep the connection alive. This was also noted by other users in the Arduino community. The pubSubClient library worked with some other IoT-servers but not with the Thingsboard server. The solution to this was to use their own Thingsboard library, which was specifically tailored for Thingsboard IoT-servers.

5.1.7 Arduino freezing with the Wi-Fi module

Arduino froze after a while of using, when the Wi-Fi module did not get a connection. This was due to the Wi-Fi module creating too long print through a serial connection to Arduino. The solution to this was to remove the whole line, which could be done as it was completely unnecessary for our application.

5.2 Problems with the hardware

This chapter goes through the problems there were when using the chosen components and how the problems were solved. Hardware was causing the second most of the problems. These problems were easier to find out but often solving them required changing the hardware or making modifications to the hardware.

5.2.1 Wi-Fi module incompatible with Arduino Uno

Wroom-02 Wi-Fi module uses 3.3V and requires 500 mA current. Arduino Uno microcontroller has a 3.3V line rated for 50mA output current. This is not enough since an average current drawing with Wroom-02 is 80mA. Arduino Uno can supply more than the rated 50mA but still not nearly enough for 500mA, and this will cause stability issues. There were a few solutions to this problem. The first solution was to make a voltage divider by using three resistors but this would be too unstable for the project because the current Wroom-02 is drawing is not a constant but it variates according to the Wi-Fi usage. The second solution for getting the 3.3V

was to buy a step-down voltage regulator, which was a viable option. The third solution was to make a new step-down circuit by using a voltage regulator, resistors and capacitors, which also was a possible solution. The fourth solution was to buy another Wi-Fi module, which also was a possible option. The fifth solution was to buy a new microcontroller and use it as a base. In the end, the fifth option was chosen because it also solved the memory problem explained in the next subchapter.

5.2.2 Uno is low in memory and space

Arduino Uno's RAM and Flash memory sizes became a problem after a while of coding. This was tried to be solved by making the code smaller but in the end, it was not enough. An SD-card and OLED-display eat a lot of space and memory together. The first thing that was tried to solve this, was to change the OLED's code from a full buffer mode to a page buffer mode, which saved a lot of space and memory but not enough. The second try was to use a smaller Bill Greiman's SDfat library, but it did not save enough space and memory. The third try was to use loops in the OLED's code, but it did not save much space and made the refresh rate slow due to using the Page buffer option from before. The final solution was to get another microcontroller and Arduino Due was chosen for its memory and output voltage.

5.2.3 Temperature sensor showing -127°C

The DS18B20 temperature sensor gave a value of -127°C when read by using the example codes. Some of the DS18B20 sensors need a 4.7kΩ pull-up resistor to work and some do not. The solution to this problem was to add a 4.7kΩ pull-up resistor between the data wire and V_{cc} wire to ensure the known state for the sensor.

5.2.4 Problem with real-time clock designed for a 5 Volts and for rechargeable backup battery

After changing Arduino Uno to Arduino Due one of the problems was that the real-time clock had been designed for a 5 Volts. It was changed to work with a 3.3 Volts by removing two pull-up resistors from it. Another problem with the clock was that it was designed to recharge a

battery in it, which would damage non-rechargeable batteries like CR2032. This was solved by removing more resistors from the clock. Also, one diode was removed, and one jump wire needed to be added. This way the recharging capabilities were disabled from the real time clock.

5.2.5 Wi-Fi module not connecting all networks

The Wi-Fi module refuses to connect some Wi-Fi networks. This could be caused by an old firmware. The Wi-Fi module was never flashed to the newest version in this project, but as for prototype it was enough that it connected to most of the networks. The security protocol was not the problem since all the networks were tested without any security. Solution for this problem was not found.

5.2.6 Problems with wirings and programming the Wi-Fi module

There are not any good guides online how to wire and program a Wroom-02 Wi-Fi module. The manufacturer's datasheet tells you how pins should be connected to make a UART-download and a flash boot, this helped a lot to figure out how the Wroom-02 should be connected.

5.2.7 Problem with a sound level measurement

The sound level sensor which was ordered was not good enough for measuring sound levels. It was designed for recognizing noises and thus it was insufficient for this project. The measure range was way too small and unreliable for measuring sound levels.

5.3 Other problems

This clause goes through some problems which was faced, and which are not related to a hardware and coding. The most of these problems came by a surprise and there was no information online about them and how to fix them. Some of the problems are related to choices

made in the beginning when tools were chosen and to solve them would mean changing some of these tools.

5.3.1 Values not showing on client's network (unsolved)

For some reason values could not be read from the client's network. The reason for this might be that there is a firewall blocking the incoming data in the client's server. The problem was discussed with client's IT personnel, but solution was not found.

5.3.2 Letters not printing correctly in some browsers

Some letters did not print correctly from the IoT-cloud to some web browsers and the reason for this was the font, which is done for Internet Explorer browser. By changing this font to Arial, the site printed every letter and character correctly.

5.3.3 The device disconnected from the IoT shows the last measured values

The IoT-cloud is programmed not to allow changes when the connection is lost and thus it does not let the values to change to zero or anything else before the connection is made again. This phenomenon is not good because this way the user can read wrong data from graphs when the connection is established again. The IoT cloud will draw a straight line between two measured values and does not let the user know if the connection was lost between. This can be solved by changing IoT-cloud to a different one.

6 Ideas for improvements and further development

6.1 Changing the microcontroller or building an own circuit board

The next step for making the project on level would be changing the Arduino Due to a completely different controller called ESP8266. Choosing this controller would make the device cheaper and as ESP8266 has more memory also the performance of the device would improve. This would involve some hardware connection changes but mostly change of coding.

After the device has been tested with all the new parts, making a new circuit board would make sense and using only the required components would make the device smaller and cheaper. The ESP8266 would work as a model where you could choose all the necessary components and leave the rest out.

6.2 Booting from online

If the device needs to be booted due to some errors, it would be easier for a user to do it online. For example, if the device is far away or in a difficult place, or if booting locally would disturb the employees nearby, then booting online would save time or the nerves of the employees.

6.3 Reading errors from an IoT-cloud

Now errors are only read on the screen, but in future it would be possible to bring them online where a user can also see if something is wrong, what is the problem and react accordingly without going to the device. This would save time, for example, if the device just needs a booting, which could be done online.

6.4 Changing the CR2032 battery to a rechargeable LIR2032 battery

Changing the CR2032 to a LIR2032 in the real-time clock would remove the need for possible time adjusting after 10-20 years when the CR2032 runs out of energy. Changing the CR2032

to a LIR2032 would require some hardware changes to the real-time clock but no software changes.

6.5 Adding a way to change settings

Now settings are hardcoded, and programming is needed to change them. For example, changing a Wi-Fi network is an important setting, which changes depending on the user's network. Different options for changing the settings are presented in the subsections below.

6.5.1 Bluetooth connection

Using the Bluetooth connection would be one option, but it has its own advantages and disadvantages. The advantages are that the Bluetooth connection would be versatile and quite simple to use, and it could be used from a couple of meters away without touching the actual device. The disadvantages are that it needs an external device which is connected to device through a Bluetooth connection. This device would require an app and coding is needed to make this app. another disadvantage is bigger current consumption.

6.5.2 Replacing Wi-Fi by using LTE/3G technology

The advantages of using a mobile internet connection instead of the Wi-Fi connection are that it would be connected even when outside the Wi-Fi connection and that it would not require setting it up. The disadvantages of this option are that it would require a connection from a telephone operator, and this usually would mean a monthly fee to a customer. Another disadvantage is that some offices are out of the reach of mobile connections due to their location or construction material.

6.5.3 Adding a keyboard to the device

Adding a keyboard would be the best option for this device. The advantages of this option are that you do not need any external devices, the power consumption hardly increases, and it is

simple to use. The disadvantages of this option are that you would need a physical contact to the device to change the settings and it might be a bit slower to change settings compared to a Bluetooth connection with an external device.

7 Conclusions

The objective of the thesis was to create a prototype for a device which measures office conditions. The conditions were temperature, humidity, illuminance and sound level. The prototype was meant to be inexpensive and easily movable, have an internet connection and have an enclosure. Another objective was to make reporting tools to an IoT server, these tools communicate with the prototype using the internet connection.

All the objectives were achieved, except for the sound level measurement. The prototype measures temperature, humidity and illuminance correctly. The values were checked by commercial meters and the prototype sends data to the internet either through an ethernet cable or through Wi-Fi. This data was successfully analyzed and visualized on the IoT-platform.

If the project would have been implemented after getting the knowledge from the first prototype, then microcontroller would be changed, and a different IoT-server would have been used. The new microcontroller would have an internal Wi-Fi circuit, more memory, and it would be smaller.

BIBLIOGRAPHY

Alten Finland. 2019. Leader in Engineering and Technology Consulting. [Web page]. alten.com. [Ref. 10 July 2019]. Available at: https://www.alten.com/alten-group/

Arduino. 2019. About Us. [Web page]. arduino.cc. [Ref. 10 July 2019]. Available at: https://www.arduino.cc/en/Main/AboutUs

Hengitysliitto. 2019. Sisäilman kosteus ja lämpötila. [Web page]. hengitysliitto.fi. [Ref. 20 October 2019]. Available at: https://www.hengitysliitto.fi/fi/sisailma/sisailma-asiat-sisailmaongelmat/sisailman-kosteus-ja-lampotila

SFS-EN 12464-1. 2011. Light and lighting – Lighting of work places – Part 1: Indoor. Helsinki: Indoor work places. European committee for standardization (CEN).

Työturvallisuuskeskus. Lämpötila. [Web page]. ttk.fi. [Ref. 20 October 2019]. Available at: https://ttk.fi/tyoturvallisuus_ja_tyosuojelu/tyoturvallisuuden_perusteet/tyoymparisto/lampoolot_ja_sisailma

Waveshare. 9.2.2018. 1.5inch OLED Module User Manual EN. [Manual]. [Referred 25.6.2019]. Available in: https://www.waveshare.com/w/upload/8/80/1.5inch_OLED_Module_User_Manual_EN.pdf