Expertise
and insight
for the future

Smart Bois:
Lauri Marjanen, Taneli Voutilainen, Emil Besic, Patrik Winter, Niko Haapalainen, Daniel Eggleton

ZigBee Controller Unit

Metropolia University of Applied Sciences

Bachelor of Engineering

Smart Systems

Internet of Things (IoT) Project TX00CI65-300630

December 2020

Number of Pages Date	46 pages + 1 appendices 30 August 2018
Course	Internet of Things (IoT) Project TX00CI65-300630
Professional Major	Smart Systems Engineering

Keywords	ZigBee, Trådfri, Light Link, Light controller

Contents

List of Abbreviations

API	Application Programming Interface, a general term for application specific interfaces. These conventions will usually cover calls, requests and rules connected to using these commands.
BLE	BLE - Bluetooth low energy, a wireless personal area network technology
COM-port	Serial ports provided by devices. Refers to emulated and physicals ports alike. Includes USB and Bluetooth.
GUI	Graphic User Interface, essentially covers user interfaces that are non-console user interfaces.
ICT	Information and Communication Technology.
IoT	Internet of Things, a term referring to the ever-expanding mass of devices communicating with the internet and sharing data.
LED	Light-Emitting Diode. a small low power light source often used in circuits.
MAC	Media Access Control, a unique identifier for network devices.
PC	Personal Computer, covers laptops, desktops etc.
PIR	Passive Infrared Sensor. Covers all infrared detecting sensors.
RF payload	Refers to the actual payload of data being transferred to be used by receiving device.
RGB	Red, Green and Blue code. Each primary colour can be defined by 0-255 value and combines to include about 16,7 million possible colours.
UART	Universal Asynchronous Receiver/Transmitter. A two wire physical circuit used to transmit bits at a configurable rate between two devices.
USB	Universal Serial Bus, interface type

1 Introduction

Welcome to the documentation of our Light Controller project's implementation phase.

Our light controller is a ZigBee Light Link based light controller with integrated sensors for easy-of-use designed to directly control Trådfri lamps. Controller gets input from sensors and a website-based GUI and will control light based on configuration settings.

This is part of a two-course project that spanned a planning and concept phase, and an implementation phase. This documentation covers only implementation. The planning and concept phase has a separate document in a similar format.

Our group for the implementation phase is **Lauri Marjanen**, **Emil Besic**, **Taneli Voutilainen**, **Patrik Winter**, **Daniel Eggleton** and a new member since the planning phase **Niko Haapalainen**. All team members are 3rd or 4th year students from the Smart Systems Major in the ICT line.

1.1 The Goal

Project goal was to build an IoT device according to our planning and concept phase in the last course. The team has some initial doubts over the suitability of our planned project to challenge the group and help us grow as developers. Shifting plans and starting from scratch would've been possible, but we decided to keep ahead with our original light controller plan. Most doubts were connected to uncertainty to how challenging would getting our wireless communication protocols running.

1.2 Project Requirements

Project requirements were brief:

1. Device must have some connection to the Internet
2. Course teacher must approve the planned device to be suitable challenging
3. Planned device budget must be accepted by Metropolia. The school supports each group with a budget of about 50€ (negotiable based on project scope). Purchases must be made from select sites. Using existing school sensors and hardware is free and group's own investments are permitted.

1.3 The Project Vision

We wanted to build a device that could replace gateway devices that most smart bulb manufacturers produce. In our case we decided to work with Ikea Trådfri lamps. Since most manufacturers work with the ZigBee Light Link protocol, most bulbs could be integrated, with minor changes to ZigBee commands.

Being realistic with our limited 4 weeks of effective work timeframe we didn't expect to make an improved gateway. Instead we aimed at making an alternative, more sensor-based gateway, to handle certain logics internally.

Main goal in the project is to get experience in developing IoT systems and getting used to problem solving. These goals can be achieved without directly making groundbreaking improvements, if we understand the components we are using and attempt to build a device leveraging that knowledge. Also challenging the team members is important to grow as developers, so taking easy route which could be a guaranteed positive outcome could end up being less productive for us. For this reason, our main plan was to replace the gateway device instead of building on an existing gateway.

2 User Manual

2.1 Initial Setup

2.1.1 XBee Setup

After you have attached your XBee S2C chip to your XBee grove board, connect your grove board to a PC with XCTU installed with the provided micro USB to USB cable.

Then we need to discover the attached ZigBee chip.

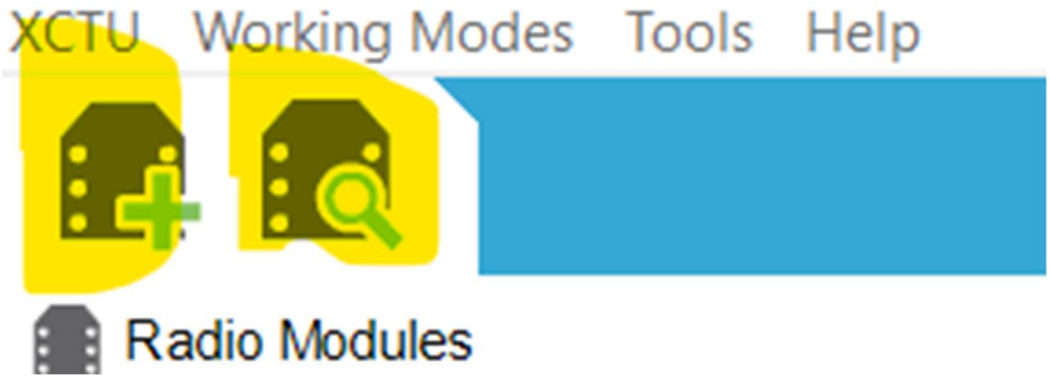

Figure 1 SCTU view buttons to start adding new ZigBee

You can discover attached ZigBee chips with either highlighted button. Button on the right in most cases will be easier to use, if your chip has basic specifications. With this method simple choose the COM-port to scan for the device (Check device manager if you are not sure which COM-port) and then assuming default chip parameters press *Finish*.

After that XCTU will take a few seconds scanning the device.

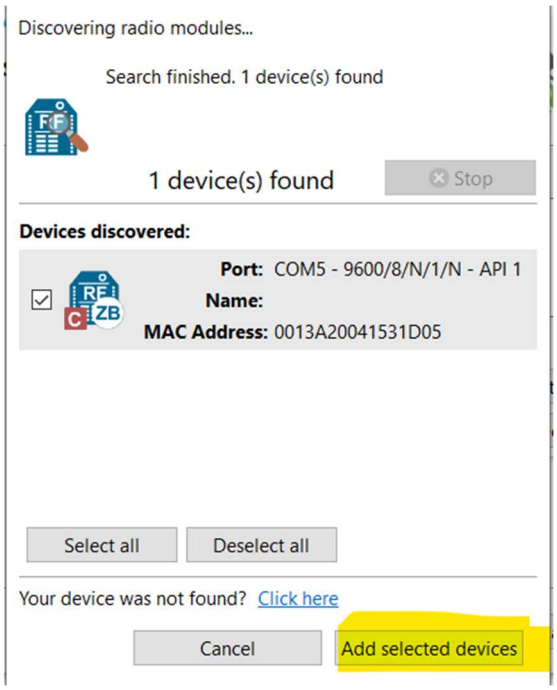

Figure 2 XCTU view, device discovering

If the correct device is discovered, press *Add selected device*.

Now your ZigBee chip has been added to XCTU and we can configure it for our needs.

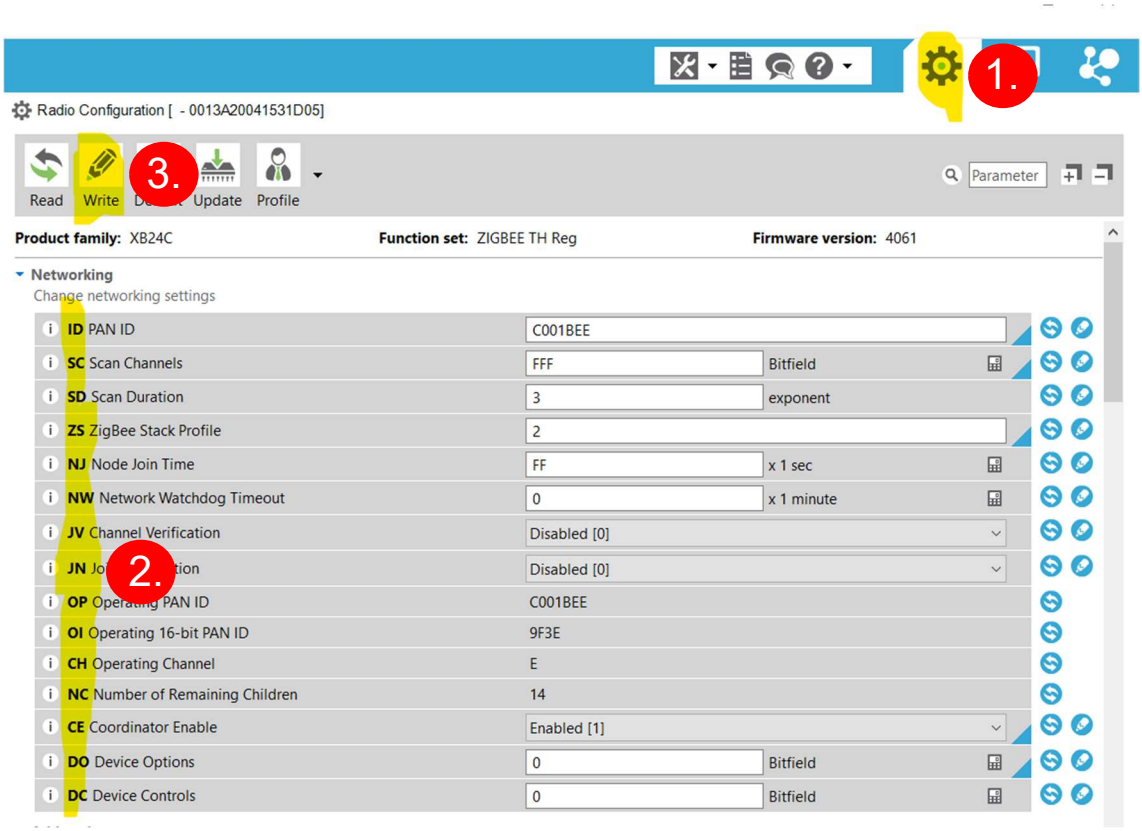

Figure 3 XCTU config window

Displayed in picture:

1. Use it to open configuration windows
2. Listed configuration parameters.
3. After changes to parameters we need to manually start the write by clicking write button

Values that need to be changed for project are as follows.

ID	Value
ZS	2
AP	1
AO	3
EE	1
EO	2
KY	5A6967426565416C6C69616E63653039
Baud Rate	115200

Figure 4 ZigBee chip values to config

NOTE.
KY is a secret key and is currently a default value defined by ZigBee Standard. This a security risk, but for testing purposes we didn't feel compelled to make changes.

Additional:

Check that device can be a set as a coordinator. (config)

After making changes remember to write changes to chip.

2.1.2 Trådfri Setup

Pairing Trådfri lamps can be done in the following manner

First factory reset the lamps according to instructions.

1. Have the light ON

2. Switch it OFF and back ON 6 times (in total, the light will be ON again 6 times without counting the initial ON state)

3. Wait in the 6th ON state, the bulb should increase brightness intensity to confirm reset

4. If the light didn't confirm reset, try to leave the bulb ON less than a second, and OFF longer. The timing of ON/OFF states may matter for some firmware versions

After factory resetting open up XCTU.

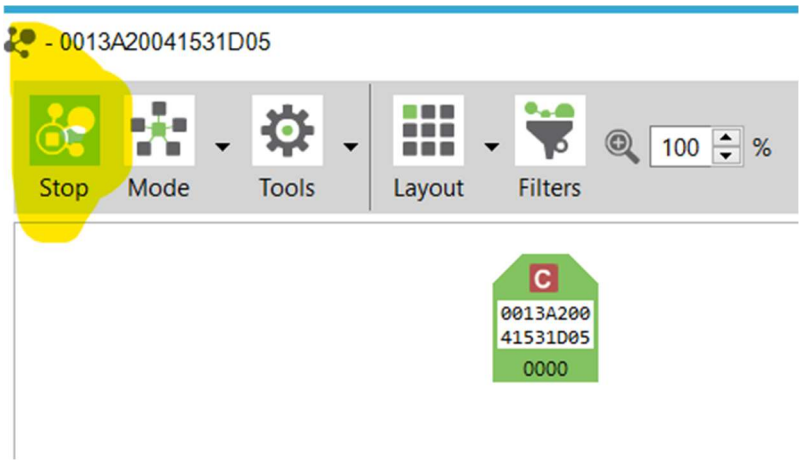

Figure 5 Scanning with only ZigBee chip displayed

With the XBee chip added to XCTU and connected with USB, start scanning the surrounding devices by pressing button shown above.

If Trådfri has been successfully reset, it should display as a similar looking icon, but with a R icon above. After this point the lamp should be paired and work in future until next reset has been done in the lamp, or ZigBee chip is switched.

2.1.3 ESP32 Setup

To flash the ESP32 with project use following manufacturer provided instructions.

https://docs.espressif.com/projects/esp-idf/en/latest/esp32/get-started/

Flash ESP32 with the supplied software and ESP32 should be ready to go.

2.1.4 Website setup

Website setup is also simple. Use following instructions to create github hosted website.

https://pages.github.com/

Then add HTML file to repository index.html.

Now the website should be up and running. Note Website must be HTTPS, if you host website on github this is included.

2.1.5 Wiring the device

Wire the device according to provided wiring diagram after all components have been set up.

If any alternative components are being used, then program will likely have issues working.

2.2 Everyday use

Currently we are hosting our website at tanelitv.github.io. Personal builds would course be on your own website.

Open a chrome browser session and navigate to the website.

Note. Must be chrome, edge or opera to function. (Check more specific compatibilities https://web.dev/bluetooth/)

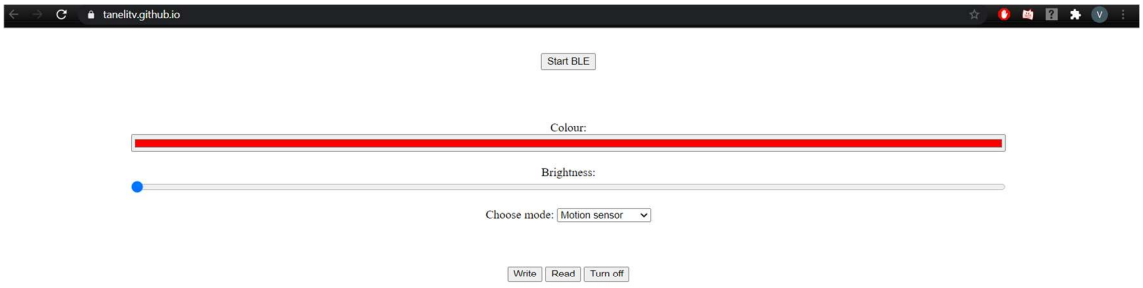

Figure 6 Website GUI

After opening website, click Start BLE.

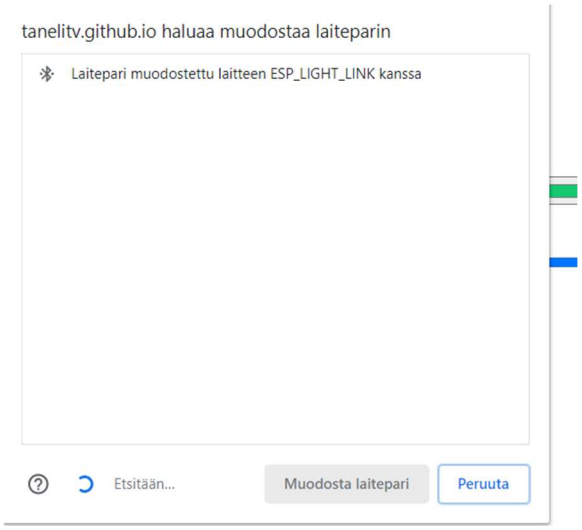

Figure 7 Bluetooth popup

After BLE pop up has shown, connect to the ESP32 that is managing the GATT server.

After this you can freely use the GUI to change colour, brightness, power and activate a single sensor to function. (currently limited to single sensor to save development time)

After changing values, press write to write to ESP32 hosted GATT server. This request will then be routed to the ZigBee network to control the lamps.

3 The Device

In the device section we discuss our final devices communications, components and sensors.

3.1 Device diagram

Up-to-date device diagram

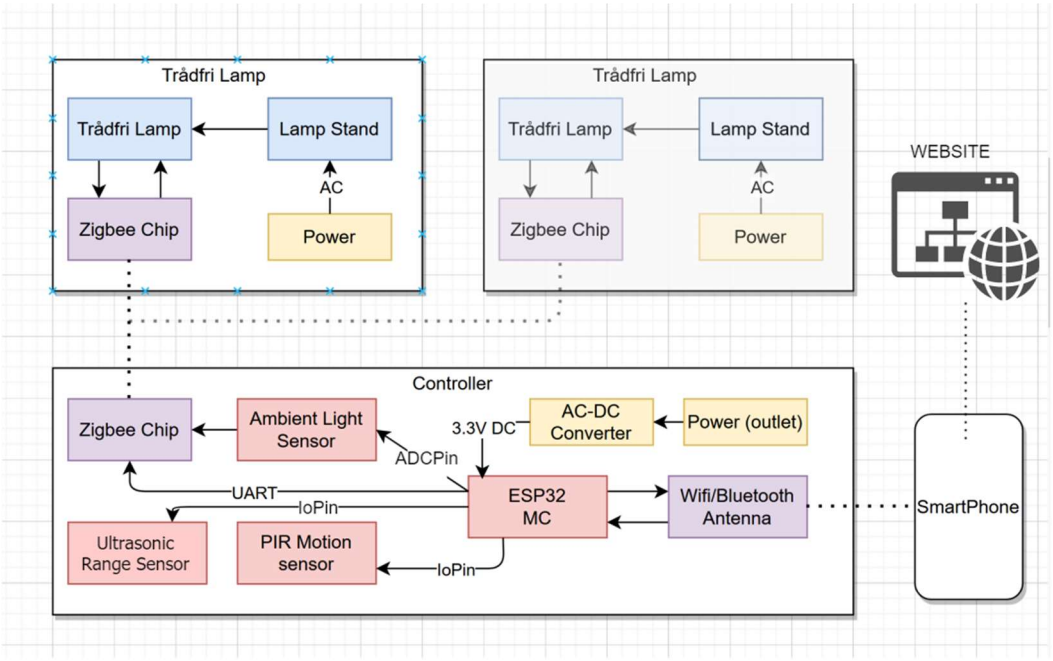

Figure 8 Final Device Diagram

Note: Trådfri ZigBee chips are integrated and not separate.

3.2 Wiring diagram

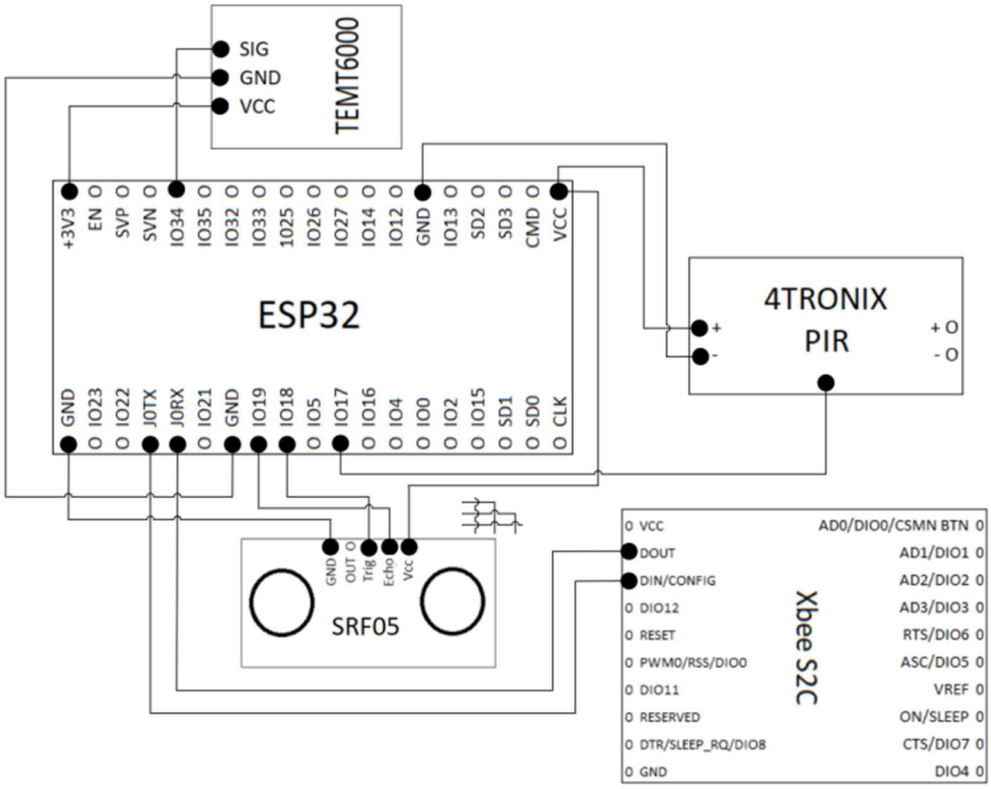

Figure 9 Wiring diagram of final device

3.3 Sequence chart

Here we visualize communication between GATT Client (Phone), GATT server/ESP32 (Controller) and ZigBee network (Trådfri lamps and controller XBee chip)

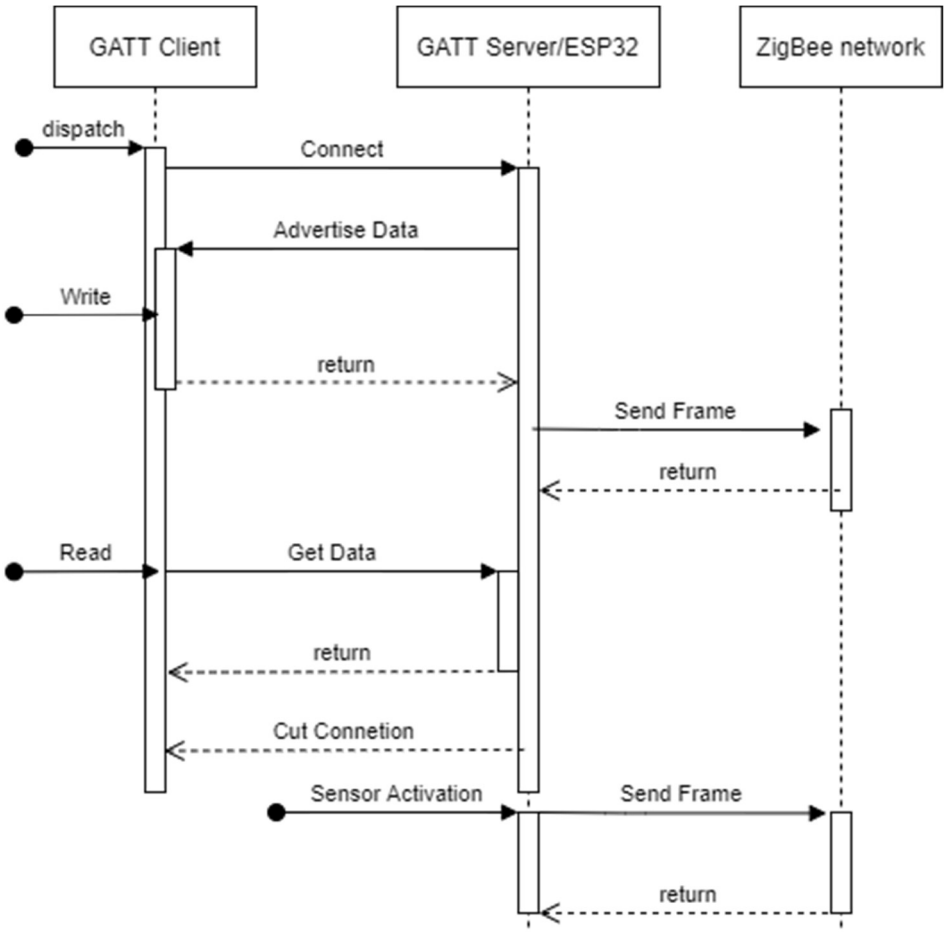

Figure 10 Flow of data between separate project components

3.4 Basic configuration

Project final device setup includes

1x ESP32 WIFI/BLE Board (CP2102) (27,90€ local price)

1x Crumble PIR-Motion Sensor (10,90€ local price)

1x Kitronik 5105 Light Sensor PCB (4,96€ local price)

1x Ultrasonic Range Sensor (3,29€ local price)

1x XBee S2C chip (27,90€ local price)

1x XBee Grove development board (26,17€ local price)

2x Ikea Led lamp 327 600 lm wireless dimmable rgb lamp (19,90€/per light local price)

1x Bluetooth enabled device with google browser (Assumed to be existing)

1x HTTPS protected website to host UI (free)

+ Miscellaneous components to connect devices (Breadboard, wiring etc.)

Setup is not by any means barebones and is based on some items already being available and some purchased items being easy to implement in the testing phase.

3.5 Components

3.5.1 ESP32

Manufacturer: Elecrow

Model: ARS01119B

ESP32 is a dual core development board. Most software logic for our program is stored and hosted on our ESP32. This board was chosen based on its wide range of interfaces (Bluetooth, Wifi and Micro USB). This would help us by circumventing any peripheral devices for these communicating methods. By the end the ESP32's role in the project extended past the original plan to host the GATT-server as well.

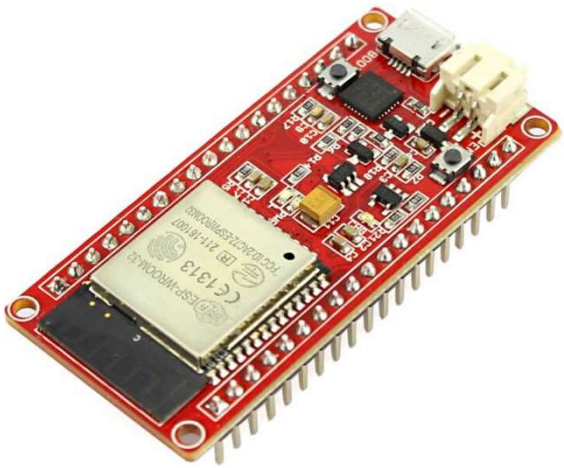

Figure 11 ESP32 development board

To harness the dual core properties our ESP32, our program was built with a Freertos operating system. This tasked based operating system allows all the cores to be utilized efficiently by spreading a single workload queue for both cores, instead of for example having 1 core working and the other idling.

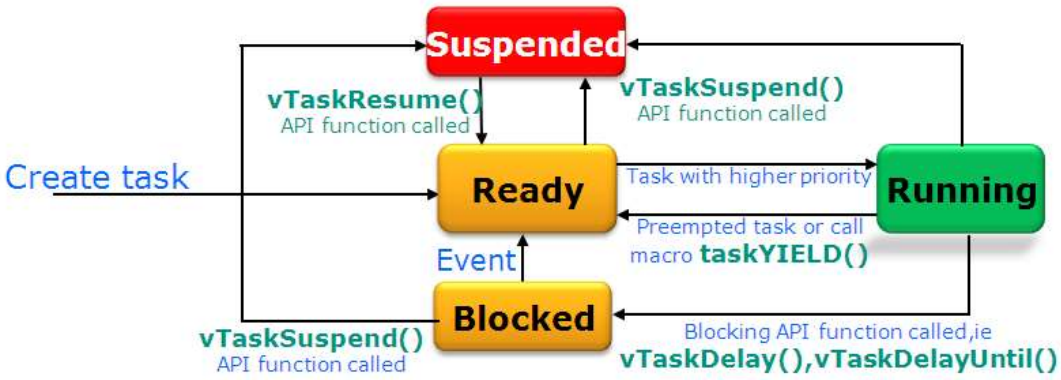

Figure 12 Freertos task states

3.5.2 XBee S2C chip

Manufacturer: Digi

Model: XBee S2C

Our ZigBee communication in the controller is handled by a XBee S2C chip. This chip is a higher end ZigBee chip good in testing phases. It has good supporting software and documentation thus minimizing integration issues with the ZigBee functionality itself.

This chip handles communication between the controller device and all Trådfri lamps. This is also the only extra ZigBee chip required as Trådfri lamps have an internal chip. The XBee acts as the coordinator in the ZigBee Mesh in all cases. Effectively acting as master device and defining certain network policies for each other connected chip.

ZigBee communication is explained in more detail in communication protocol section.

Project specific XBee configuration is detailed in user manual initial setup.

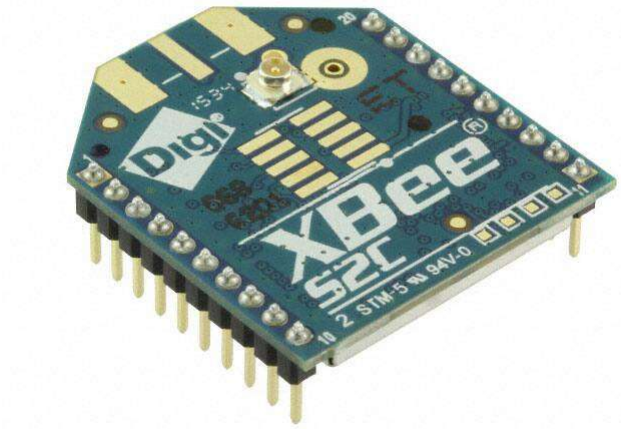

Figure 13 XBee S2C chip

In testing we used the supplied grove development board. This allowed quick startup of the testing with handy capabilities such as serial connection through Micro USB, a ready to go circuit, indicator LEDs and resetting chip with a button.

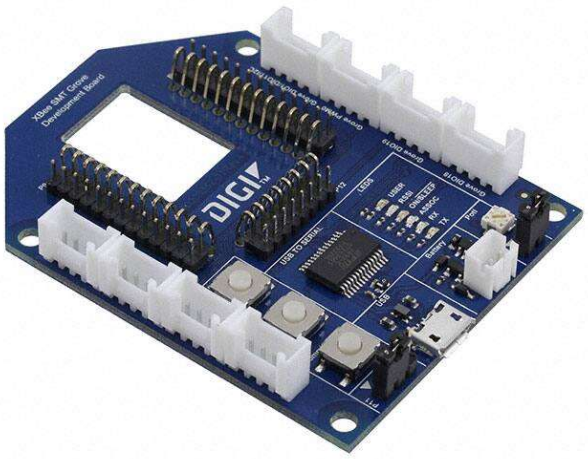

Figure 14 Grove Development Board similar to one used

3.5.3 PIR-motion sensor

Manufacturer: 4tronic

Model: PIR Sensor Crumb

The 4tronic PIR-motion sensor is an easy to approach movement sensor. The sensor is mounted on a breakout board to make placing easier and simplifying outputs to beginner level. Device can either output Analog or Digital signal based on what side of sensor power and ground are placed.

Our initial idea was for a PIR sensor that could act in as hall monitor fashion to turn on lights for a certain time after reading movement. Currently testing is still underway to check PIR range and judge its effectiveness at fulfilling this task.

PIR-Motion sensor is not an ideal sensor to fulfil our wishes as it is only activated by movement and is not too sensitive. Alternatives could've been a Grid-Eye sensor for active temperature recognition. This was unfortunately outside our budget range. We deemed the motion sensor to be a side objective and wanted to concentrate on wider range of sensor.

Figure 15 PIR Motion Sensor

3.5.4 Ultrasonic Range Sensor

Manufacturer: Unknown (School provided)

Model: SRF50

Ultrasonic range sensors function by sending ultrasonic waves and waiting for the wave bouncing back. Functionality is explained well in this article SRF05 functionality. Output functions by setting pin high when ultrasonic wave is sent and set pin low when wave has bounced back/wave waiting has time out. Starting a reading is activated by a trigger pulse inputted to sensor.

Our implementation of the sensor initially was unsure, but there were a lot of ideas of controlling the device like changing brightness by reading hand range going from close to far/far to close.

After some testing of the sensor, we concluded that it's not precise enough to function with fine hand gestures. So we continued with basic toggling feature to start with and would expand if possible.

In the final device setup, the sensor worked as a hand free light toggling sensor. Appropriate to current times, with people being more considerate with hygiene than for a long time. Even in this manner our setup needed some timing improvements to function as wanted.

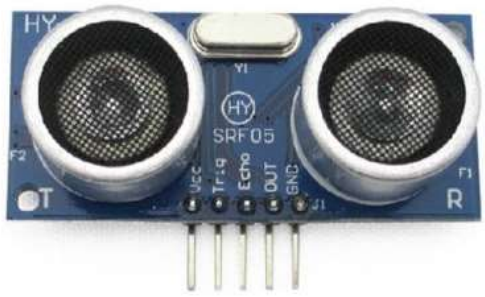

Figure 16 Ultrasonic Range Sensor

3.5.5 Ambient Light Sensor

Manufacturer: Kitronik

Model: 5105

The Kitronik ambient light sensor is a sensor that measures the current light level in its environment and returning an analog signal based on that level.

An ambient light sensor has quite limited uses due to feedback issues with lighting, yet a very helpful capability is keeping a minimum lighting inside a room. This would have the effect of making sure you would never have to walk in total darkness (at least based on sensor location) and not waste unnecessary power during the day when the room might get enough light from outside sources. Only issue is recognizing when to turn off during the day.

Feedback issues mentioned earlier are specifically connected to the issue of having a light sensor controlling the light level. In an absolute minimum the system would require extended testing to make sure levels do not change too often.

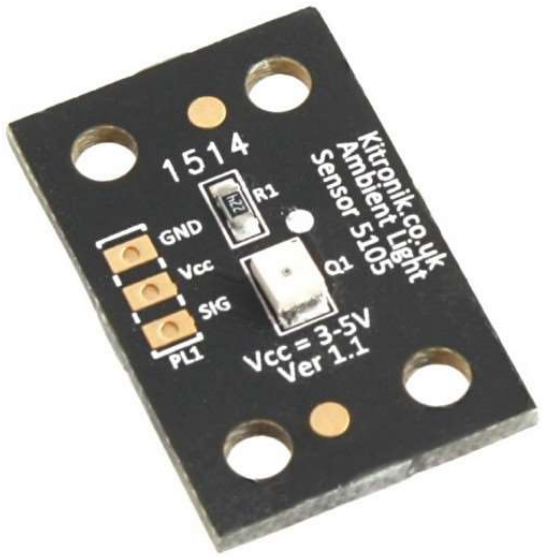

Figure 17 Ambient Light Sensor 5105

3.5.6 Trådfri Lamp

Manufacturer: Ikea

Model: LED1624G9

The Ikea Trådfri is smart lamp that is widely available at a competitive price. It is Light Link device meaning it is structured in a manner complying with ZigBee standards and at least to some level is compatible it's competitor's offerings complying to the same standards.

While working directly with Light Link lamps is common and quite accessible by mixing and matching manufacturers products or building your own controllers. Manufacturers generally do not give direct support to this activity as it can be counter productive to their business. Regardless there is an active community around such light controlling, giving a decent number of sources to work considering this is still a new standard on the grand scale of things.

Through research and testing we found the required clusters and commands to operate a Trådfri in our wished manner. More general specifications of structuring a command will be listed under communication protocols.

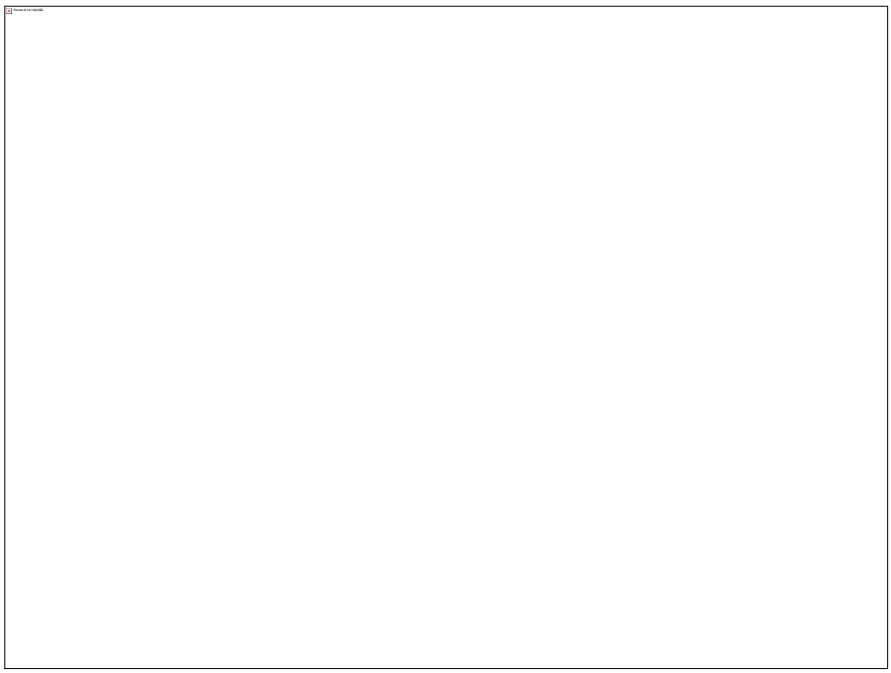

Figure 18 Our two Trådfri lamps and stands used for testing

Trådfri lamp commands	ZigBee cluster	Profile ID	Command	Additional parameters
Toggle light on/off	0x0006	0x0104	0x02	-
Light on	0x0006	0x0104	0x01	-
Light off	0x0006	0x0104	0x00	-
Scale brightness	0x0008	0x0104	0x04	0-255 brightness scale
Move to Colour	0x0300	0x0104	0x07	Colour XY, X:0-65535 Y:0-65535 Transition time: 0-65535 (1/10ths seconds)

Figure 19 Trådfri Tested commands

3.6 Communication Protocols

Here we talk about our communication protocols in a more general manner. For example, range, capabilities and important notes in its use.

3.6.1 ZigBee

ZigBee is a wireless low power, low rate mesh network designed to be used with battery powered devices. ZigBee devices primarily work in 2.4GHz band. This is a band reserved for FIXED, MOBILE, RADIOLOCATION, Amateur & Amateur-satellite service worldwide. The ZigBee Communication Standard is defined in IEEE 802.15.4.

ZigBee operating range is hardware and environment specific. In our case we would assume the range to be in the low end as our controller is designed for indoor use and our ZigBee chip does not include a substantial antenna. Low end maximum range is estimated as 10 meters. One of benefits of mesh network is that operating range can be extended by adding ZigBee chips as routers to extend the signal. With the example of

lighting, while 10 meters from the master device might not cover a full area of house, generally multiple other lights will exist inside the 10 meters. When these lights are added to the mesh, the range will be 10 meters around each of those lights. Building an extending this network in this manner can make the true operating range many times further than the master device independently could cover.

ZigBee mesh is structured around coordinators, routers and end devices. A ZigBee mesh will include a single coordinator and at least 1 end device or router. Routers can route data and enable the extension mesh. End devices on the other hand can not extend the mesh as they cannot route data. The main difference in a router and end device, is that only a end device can sleep. As such a battery device can save charge by sleeping, but routing in this case must be handled by another ZigBee device operating as a router. In our case no end devices are required as the lights are connected to a stable power supply. Lifetime could be possible expanded by sleeping.

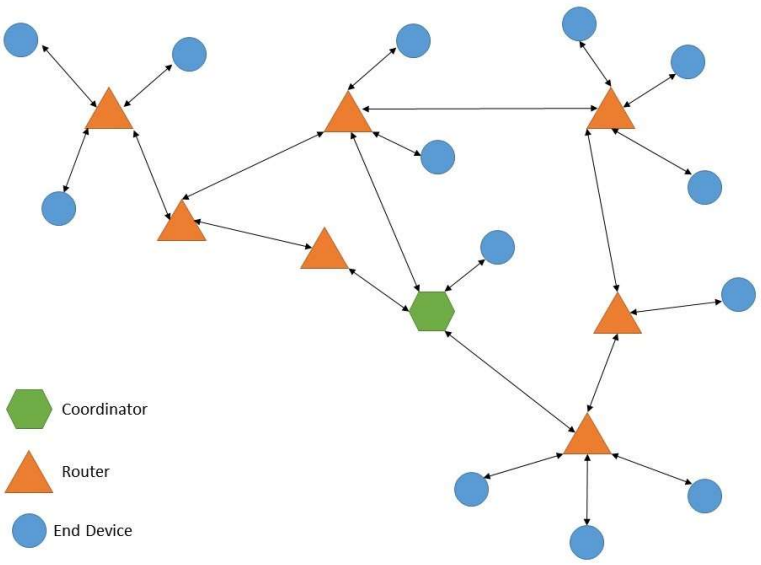

Figure 20 Example ZigBee mesh

3.6.2 Explicit Addressing Command Frame

All our built ZigBee commands use the Explicit Addressing Commands Frame. This frame is used to send a RF payload to a specific address, using specific source and destination endpoints, cluster ID, and profile ID.

Example frame scale light brightness

```
7E 00 1B 11 01 84 2E 14 FF FE A7 02 A1 FF FE E8 01 00 08 01 04 00 00 01 00 04
41 10 00 10 87
```

Hex	Description
7E	Start delimiter for a XBee API mode frame
001B	Number of bytes between length (here) and checksum
11	Frame type (Explicit Addressing Command Frame)
1	Frame ID Used to match responses with commands
84 2E 14 FF FE A7 02 A1	64 bit destination address, Mac Address of one of our lamps (Broadcast 000000000000FFFF)
FF FE	16 bit destination address, “Unknown” value as device is identify in 64 bit
E8	Source Endpoint, common Digi Data endpoint identifier
1	Destination Endpoint, Endpoint that works with Trådfri
00 08	Cluster ID, Cluster that includes brightness settings
01 04	Profile ID, Home Automation profile id, work with smart lamps
0	Broadcast radius, maximum number of hops, 0 lets mesh define max
00	Options, additional digi specific options
01 00	Unknow fixed value Seems like nesting (Single source explained, cannot be found again)
4	Command type, 4 effect brightness scale
41	Additional parameter, 0-255 brightness scale in this case.
10	Additional parameter 2, 0-255 transition time in this case
00 10	Unknow fixed value Seems like nesting (Single source explained, cannot be found again)
87	Checksum, all bytes of packet after length

3.6.3 Bluetooth and GATT

Bluetooth is a low-range, low-power wireless communication standard. This a communication method often utilized in PC, mobile device, audio and/or gaming console cross communication. Bluetooth has become one of the most widely spread consumer used communication methods. In our case most people will have the possibility to control our device, since most if not all modern mobiles phones are built with Bluetooth chips integrated.

Bluetooth range is typically less than 10 meters but enables higher data rate than ZigBee. Our system has a very light interface, meaning the higher rate is not critical.

Another limiting factor for Bluetooth communications is that communication happens between two devices and no functional network is created.

GATT refers to Generic Attribute Profile and defines how Profile, Service and Characteristic concepts are utilized to transfer data between two devices. GATT refers only to a post paired device relationship.

A GATT structure is built with a Server/Client relationship. Server stores characteristics that can be accessed by paired client devices.

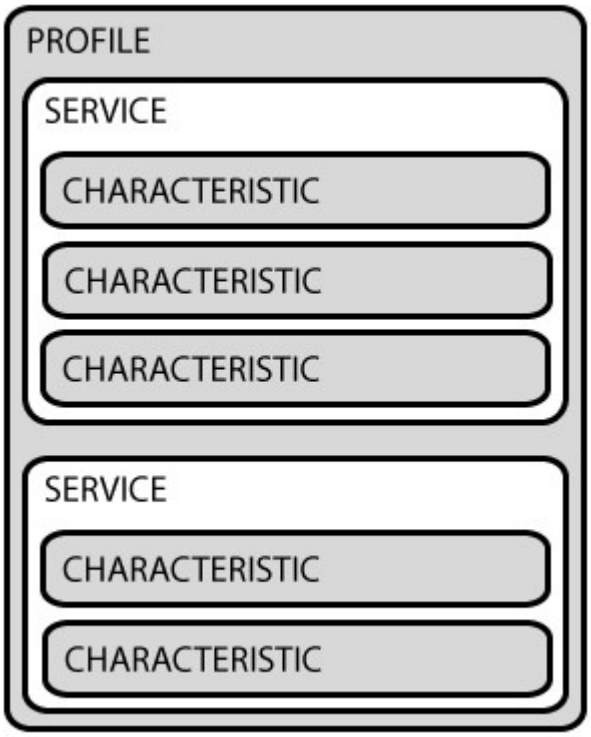

Figure 21 GATT Structure

Data is nested in GATT server with a manner displayed in the above figure. A GATT server can store a variable amount of each structure type.

A GATT server advertises characteristics to clients and clients will request changes in the reply. These changes will then be inserted in server characteristics if permittable.

4 Development process

Here we have documented our project's progress, meeting notes and more generally talk about how our project evolved from start to finish.

4.1 Initial Concept

Initial concept based on planning and concept phase from last course.

In brief concept includes 1 controller device, mobile phone and at least 1 Trådfri lamp. Controller device includes Wifi/Bluetooth receiver to connect with mobile phone. Phone has a GUI to control connected light/sensors. Controller device has sensors that can alter light states through software. Controller device communicates with Trådfri lights through the ZigBee chip.

For more detailed breakdown check planning and concept document.

Initial concept had yet to include connection to internet as this was not required in planning and concept phase.

This diagram is outdated and does not represent the final product. This diagram is only for reference and to compare initial to final device diagrams.

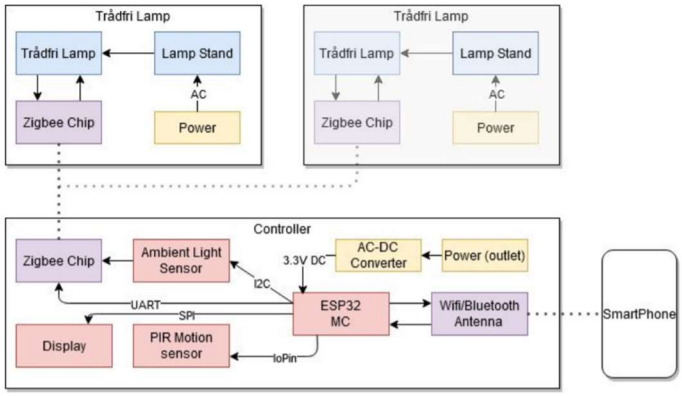

Figure 22, Initial device diagram (OUTDATED)

4.2 Diary

Here is visualized our task progress over the 4 weeks from the first meeting with the implementation phase with our physical components.

	Not started	Under work	Complete	Abandoned Path	
Goal	Week 1 >>	Week 2 >>	Week 3 >>	Week 4 >>	Overtime
ESP32 configuration					
Ambient light sensor orientation					
PIR motion sensor orientation					
Distance sensor orientation					
Zigbee orientation					
Raspberry Pi Web Server setup*					
GATT-Server setup					
Website Building					
Device Case**					
Final Device configuration					

*Plan altered to GATT-server path

**Printing possibility limited due to corona restrictions

4.2.1 Meeting 13.11.2020

Place of meeting: Myyrmäki Campus

Attending: Lauri, Emil, Taneli, Patrik, Daniel and Niko

Meeting notes:

The groups first face to face meeting and start of the implementation of the device was held 13.11.2020. No strict goals were set for this meeting prior.

To start off with we had a short session getting familiar with the hardware components (ESP32, ZigBee chips, PIR Sensor, Ambient Light Sensor and Ultrasonic sensor). Introductions were quick without digging into the datasheets. ESP32 was already setup with a hello world build as a team member was familiar with the microcontroller. IDE used for this step and planned for later is Virtual Studio Code

Knowing the PIR sensor would require having some distance from main board and lacking protruding pins. We as group went to the soldering classroom to solder some wires into the sensor's breakout board. While nobody was too experienced with soldering, we went through the basics. After some initial issues with soldering iron heat and solder wire, we managed to solder the wires in.

Next main issue was figuring out of to connect or "hijack" the ZigBee Light Link communication. After some initial searches we concluded that this wouldn't be a simple process. Since Light Link is encrypted communication, we would require some sort of secret key. At this early point (expecting issues to arise) we decided on a fallback plan. If hijacking the ZigBee communication would be problematic, we would purchase controller unit and let that handle the Light Link, while we drive the controller to get out commands out.
After some continued searching we found some threads that could help us with our original idea. For the time being we continued with the original plan.
We decided that for the future we would have 2 meetings a week face to face. One of Fridays at a Metropolia campus and another on the weekend at a public library. (excluding first week)

Finally, we made a rough division of work:

Jobs included:

- Using sensors to get output
- Connecting two ZigBee chips together
- Setting up a MQTT server with a raspberry pi and building a basic HTML file
- Documentation
- ESP32 development

4.2.2 Meeting 20.11.2020

Place of meeting: Myyrmäki Campus

Attending: Lauri, Emil, Taneli, Patrik, Daniel and Niko

Meeting notes:

Meeting was started with some initial testing while waiting for the group assemble.

ESP32 blinker test with an external LED succeeded after a couple broken LEDs were discarded. This was a simple resistor + LED being driven from ESP32 Digital to ESP32 ground.

Distance sensor test failed but was the expected outcome and this phase. Sensor testing lead to issues with Arduino examples being ported to LPC boards (that were used for testing). As Arduino handles most initial setup processes unlike the LPC boards, testing needed to be approached from a different angle. The new approach was using NXP's LPC example codes as a basis, instead of the specific Arduino codes. Arduino codes were only used to support the understanding of the logical process required to operate the components. Arduino libraries could be ported to ESP32 for more simplified coding, but we deemed that it would oversimplify our project. During the meeting we decided to purchase a second ESP32 to support sensor testing. Testing directly with ESP32 would avoid any bugs that are LPC specific.

ZigBee testing was at the point of reading into ZigBee documentation in preparation for hardware testing. In this meeting two ZigBee chips where connected to each other with XCTU software (designed by the XBee team). This was a simple test to check that the chips chosen were in operating condition. Communication from chip to chip succeeded and console communication was verified. Next step would be attempting to communicate with a ZigBee Light Link lamp.

Raspberry Pi was brought to the meeting, but after boot-up issues appeared we concluded that the Raspberry Pi brought was damaged. Our team had a second Raspberry Pi in reserve that we picked up during the meeting. This Raspberry Pi was booted with the same data and succeeded. A direct ESP32 based MQTT was suggested, but not

switched to yet since ESP32 microchips were in limited quantity. This would move the server handling to the existing ESP32.

A rescheduling was planned for the next few weeks as a Friday and Saturday meeting left no time between Friday and Saturday and an unnecessary long time between Saturday and Friday. A poll was held to check best times for meeting. Results will be checked later.

4.2.3 Meeting 21.11.2020

Place of meeting: Sello Library

Attending: Lauri, Emil, Daniel and Niko + the rest remotely

Meeting notes:

Since last meeting yesterday ZigBee has had a simple hello world python code tested on a ZigBee to ZigBee setup. This code communicated directly with ZigBee device and wasn't tied to the earlier .exe program (expect for confirming ZigBee chip output).

Also outputting data with the ambient light sensor was successful using ESP32 board. Directly using ESP32 was much faster than the earlier LPC testing and in addition no migration will be required to get the sensor to work in the project build. Moving to direct ESP32 development has thus give positive results. We continued with this success by setting up ESP32 development on another of the group's development PC. Only one PC at this point is still developing with the LPC Setup.

Today's goals for the face-to-face meeting group was to try to connect the ZigBee chip to the Trådfri device. After a quick recap on research done on the matter, the team concluded that we will start working with A ZigBee to MQTT repository by user Joonas to handle connection with Trådfri.

To start off we cloned the repository off GitHub. Then we attempted to config our ZigBee chips with this project, but ZigBee Herdsman errors started to appear. We believe that the issue is a configuration issue with the ZigBee to MQTT code. We attempted to run the code in ubuntu to avoid any possible linux to windows migration issues that could've existed since ZigBee to MQTT was natively built on linux platform. Work on this continued past meeting and goal of connecting to a Trådfri device with ZigBee chip was not met during the meeting.

4.2.4 Meeting 24.11.2020

Place of meeting: Remotely

Attending: Lauri, Taneli, Patrik, Daniel and Niko

Meeting notes:

We had a quick recap meeting today remotely to primarily advance the ZigBee communication.

To start off we talked about current state of assigned tasks. Sensors testers got the PIR motion sensor to return values. After some confusion with PIR motion functionality (specifically what triggers it) we concluded it was working as it should be. Our distance sensor on the other is suspected to be damaged from our last week's meeting. An unknown issue with our setup caused smoke to come from the test circuit. We will be attempting to get a replacement sensor from the school in the coming week.

Web server development on Raspberry Pi was put on hold for a few days. Some other ideas were put on the board how to communicate with the webserver and we didn't want to limit future possibilities by developing prematurely a server which limited out options. A Bluetooth API was specifically the new suggested method. Enabling the web server to directly send commands through the phone to the Bluetooth connected ESP32. This path will be checked for a few days to see if it compatible with out project. After that we will decide the path for the project.

To the main point of the meeting. No further development has happened with ZigBee since Saturday. We were having issues with the virtual ubuntu OS on windows, as it wouldn't recognize USB com-ports. Next step would be testing an installed ubuntu OS. This would be as close to the example set up of project as possible at this point. We found the our XBee chip was not directly mentioned to be a compatible device. This could be were our issues are coming from, yet we were confident in testing with the Ubuntu OS system first. We also recapped our fallback plans. We should be prepared to swap to that gateway fallback plan when we still have adequate time (if needed).

4.2.5 Meeting 27.11.2020

Place of meeting: Myyrmäki Campus

Attending: Lauri, Taneli, Patrik, Daniel and Niko

Meeting notes:

Before today's meeting Niko and Lauri met with our course teacher to discuss our current situation with ZigBee signal hijacking. Since our route of using the ZigBee to MQTT has yet to work out, we felt it necessary to discuss with him and ask for tips. According to his suggestion we should rethink our original method of connection. He said we should connect directly to the lamp with our chip. Direct connection could be done by factory resetting the lamp. This can be done by turning the lamp on and off 6 times within specific time constraints. When resetting the lamp to factory state it goes into pairing mode. During pairing mode pairing devices (like a ZigBee Chip and Trådfri lamp) should be possible according to this plan. In summary out ZigBee plan has now been diverted to this direct connection method.

Our website plan has been confirmed also. We will attempt to use a Bluetooth API to communicate from website to ESP32 through the device with website open. Bluetooth API can connect directly through the device to local Bluetooth devices so should suffice for this. During the meeting research was done into using Bluetooth API and a preliminary setup of HTTPS secured website was done. Github has a feature to setup a single website per user. This site is HTTPS secured and this is a requirement for using Bluetooth API. According to this plan a raspberry pi is obsolete from current plan as website is hosted by Github. According to this plan GATT-server structure would be built. ESP32 would act as the server and share its characteristics to peripheral device to alter (website). Characteristics would include our lamp RGB value, on/off state and brightness. ESP32 would then forward data to the ZigBee chip to handle.

All sensors are currently getting basic output, but coding is still underway and will continue.

A 3D-printed draft case was printed for our setup and the v.1 final case 3d-model is ready for printing. Modelling was done with a web-based 3D program Tinkercad.

4.2.6 Meeting 4.12.2020

Place of meeting: Myyrmäki Campus

Attending: Lauri, Emil, Taneli, Patrik, Daniel and Niko

Meeting notes:

Sensor development met some issues with software coding. The issue was with a software timer for the ultrasonic range sensor. An interrupt-based system was considered best practice by the group but coding it in was met with complications. Sensor group continued with resolving this issue.

Bluetooth API development was still underway. HTML page had the more complicated frontend components added to it.

ZigBee development continued as the main goal again for the meeting. During the meeting we confirmed that a ZigBee light can be communicated with through XCTU and the ZigBee chip. We had some uncertainty with connecting, as XCTU didn't directly support non-XBee devices. Being able to connect was a major step as XCTU enables an easy-to-use GUI for our use. We confirmed our lightbulb MAC-address and confirmed the Manufacturer MAC-prefix (Silicon Labs: 842E14FFFEA7). We made some probing attempts at making a correct command, but as these were fruitless, we needed to research into the topic more thoroughly.

Configuring our ZigBee chips also succeeded, and we connected to the communication with a default secret key and setup base configurations to be able to communicate with Light Link devices.

The biggest issue was finding the correct frame type as XCTU frame generator included 50+ frame types with a wide array of command parameters. Most project examples on the web used libraries so it was impossible to determine the frame type without digging deep into libraries. During the end of the meeting we discovered a Github project that used a more direct method and he used the frame type 0x11 Explicit Addressing Command Frame. This would prove to be a working frame for our need.

4.2.7 Meeting 5.12.2020

Place of meeting: Remotely

Attending: Lauri, Emil, Taneli, Patrik, Daniel and Niko

Meeting notes:

Before today's meeting we had finally managed to remotely control a Trådfri lamp through XCTU. We found our correct frame and managed to build 4 separate commands with the help of the project code we found last meeting. These commands included light toggling, off, on and brightness. The project had colour changing function, but this didn't work for Trådfri lamps.

Through some digging we found that Trådfri lamps didn't support the same colour changing command as HUE and various other smart lamps. These lamps used a Hue and Saturation function (cluster 0x0300, command 0x06), while Trådfri supported a colour XY function (cluster 0x0300, command 0x07). Since Trådfri isn't aimed directly to developers no real datasheet exist making finding these datapoints problematic. Most of these points are discovered by developers in our situation and shared to the community. By the end of the meeting we managed to control colour as well.

Meeting itself mostly included sharing our progress and some group effort into dealing with our interrupt-based system issue on our ultrasonic sensor.

4.2.8 Meeting 7.12.2020

Place of meeting: Remotely

Attending: Lauri, Emil, Taneli, Patrik, Daniel and Niko

Meeting notes:

Worked continued with our assigned work.
Sensor group continued work on an interrupt-based handling of ultrasonic sensor.
Our Bluetooth API/ GATT server setup and testing are currently our main goal, now that ZigBee communication has been reached. Currently we can pair a laptop to ESP32 and interact with GATT Service. An issue with GATT our setup is that we wanted a single service with multiple characteristics, but our outcome was multiple services with single characteristics. Fixing this structure was a goal for our meeting.

ZigBee communication's next goal is converting RGB colour to XY colour, building Checksum validation from custom frames and receiving commands to forward from UART.

4.2.9 Meeting 10.12.2020

Place of meeting: Myyrmäki campus

Attending: Lauri, Emil, Taneli, Patrik, Daniel and Niko

Meeting notes:

Work started on our final build.
Sensors were ready for integration with some slight modification for our main program's needs. Mostly working on Freertos structuring for each sensor.

Bluetooth API was ready for initial testing. We got read/writes moving between ESP32 and website, yet some issues appeared with keeping a smooth connection. Testing and debugging continued during the meeting.

This was the first meeting since successful ZigBee commands have been done and two Trådfri lamps were at our disposal. Initially we tested broadcast commands (which worked), then we tested ESP32 written commands (which worked). Now we only needed to code frame building and the logic behind pushing the messages out. Initial issues were met with ESP32 UART configuration but were fixed quickly.

Mostly from ZigBee code required were checksum generating and RGB to colour XY. Both of which were document well.

Final setup of the device was planned, and diagrams were put under construction.
Official deadline is tomorrow, so work will go to the wire.

4.2.10 Meeting 11.12.2020

Place of meeting: Myyrmäki campus

Attending: Lauri, Emil, Taneli, Patrik, Daniel and Niko

Meeting notes:

This was officially the final day of work and will be the final recorded group meeting. Overtime has been given and is required by our group to connect our final device build.

During this meeting we had two main goals. Firstly, work on the code. Mainly we worked on ZigBee frame building and combining sensor codes to a single program. Secondly, we worked of documentation. We built final device diagrams based on the two ESP32 builds (one with ZigBee and other with sensors). We assumed no issues would be met in combining these since no pin placements were shared by the builds. As expected, the project was not finished by the end of the meeting.

While overtime was defined as one week, we hoped to finish the project by Sunday based on team timetable issues. Documentation and final device building still required work and would be continued in our own time.

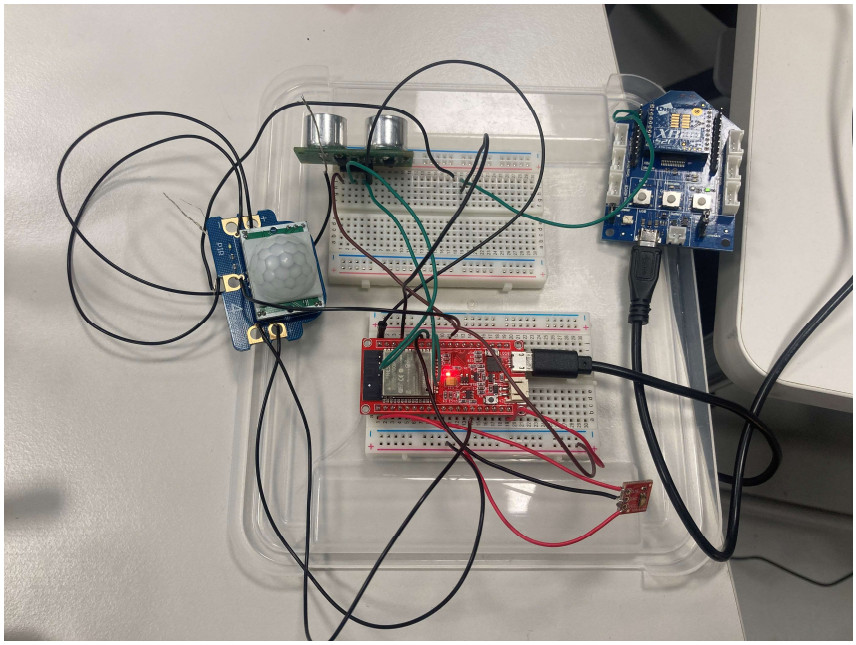

Figure 23 Final Device setup

4.3 Testing and device usability

Here we will show our findings of the final device in completing it's expected functions. Testing and functionality will be listed per component.

4.3.1 Controller

Goal: Easy to place, wall mountable, protects hardware.

This goal was not met, because of two reason. Firstly, our 3D printed case plan was left only as a model and couldn't be printed due to Corona restrictions. Secondly, we had a limited amount of time in the end to plan final device setup and sensor integration had issues. Planning the final device with a case would've required carefully considered sensor locations, since all the sensors activation that are not possible from inside the case.

A Draft Case was printed successfully before restrictions started. The draft case was surprisingly strong, so it would do fine protecting the internal components, but sensors had no plan made before working on the case was abandoned.

Summary: Adequate protecting for device not met. Requires further development. Suggested next steps would be printing the V.1 case, drilling/cutting required holes into case for sensors and power cable, and then mounting the device into the case.

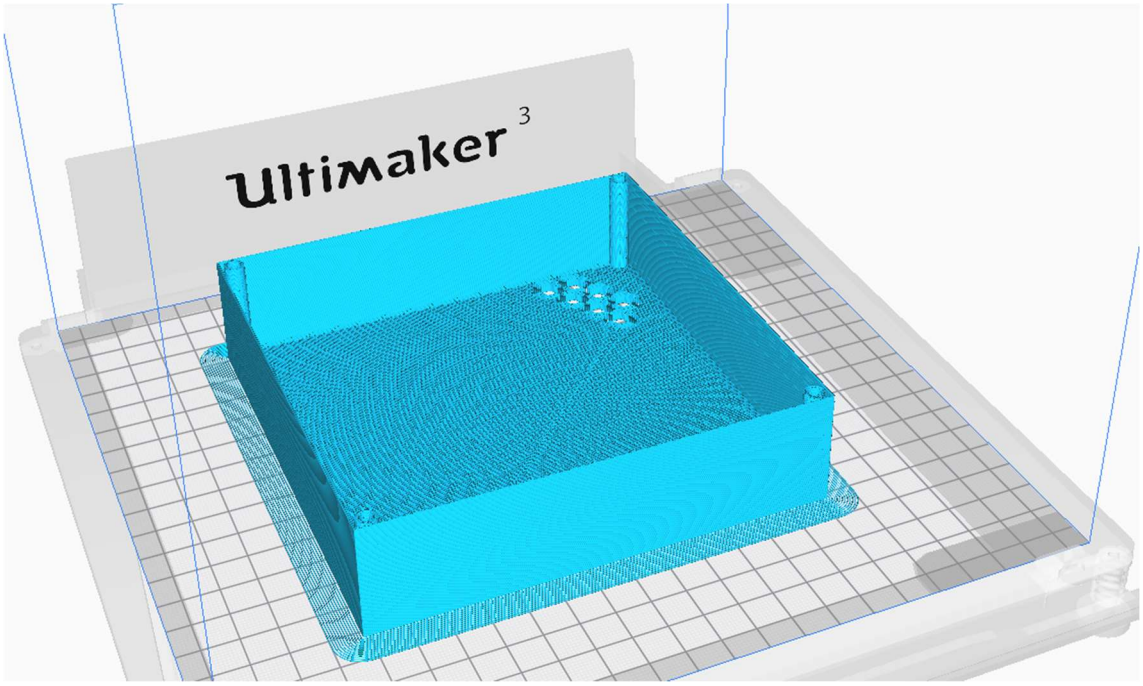

Figure 24 Case top, view from Ultimaker cura with gcode

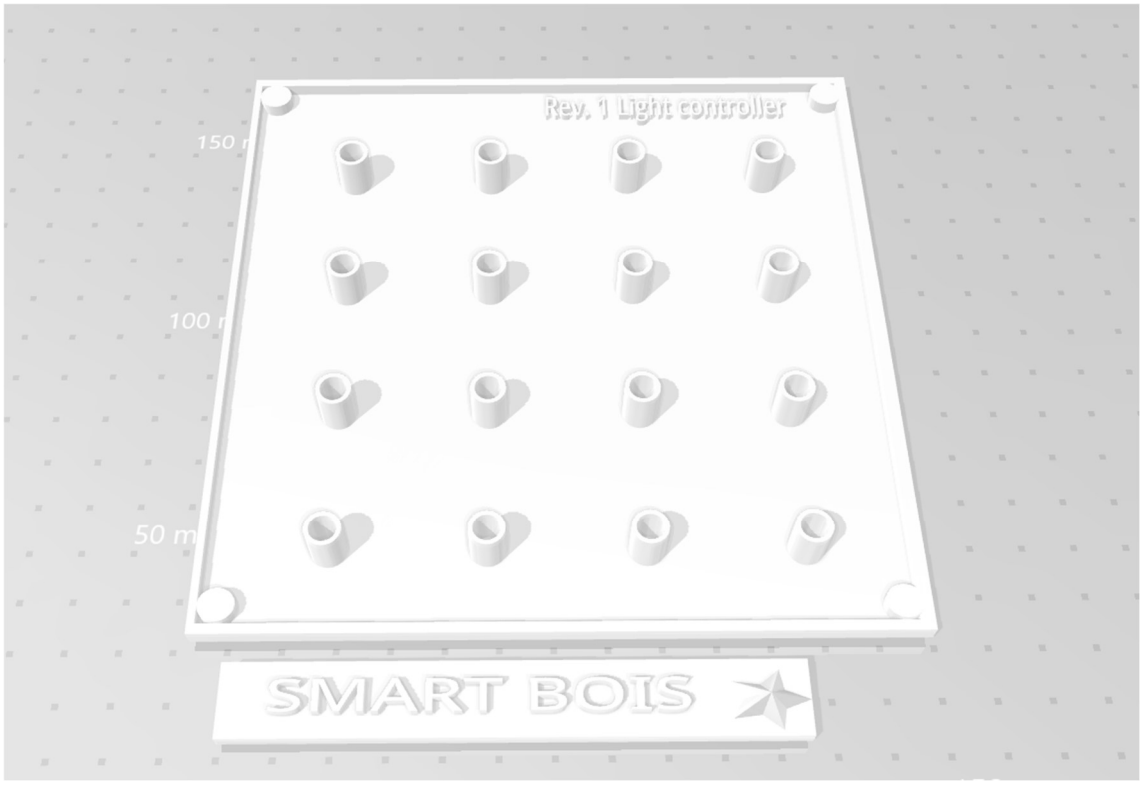

Figure 25 Case bottom and a logo, 3D model

4.3.2 PIR-sensor

Goal: Detects movement in a room with any level of lighting. Have a minimum range of 2 meters to cover a hallway from side to side.

This goal was met. Our PIR sensor had about a range of 2-3 meters. It reacted quite slowly to movement but, it would only require activation once to get the light on.

Minimum light setting would help in not requiring instant light reaction. Currently only 1 sensor can be active but could be changed as long and any conflicting action sensor points are dealt with.

Summary: Goal was met, but reaction time is sluggish. Future development would be dealing with sensor conflicts and allowing sensor to be activated all at once. Another point would be changing PIR movement sensor to a grid eye, this would take the controller to a different function and could be used in living rooms etc.

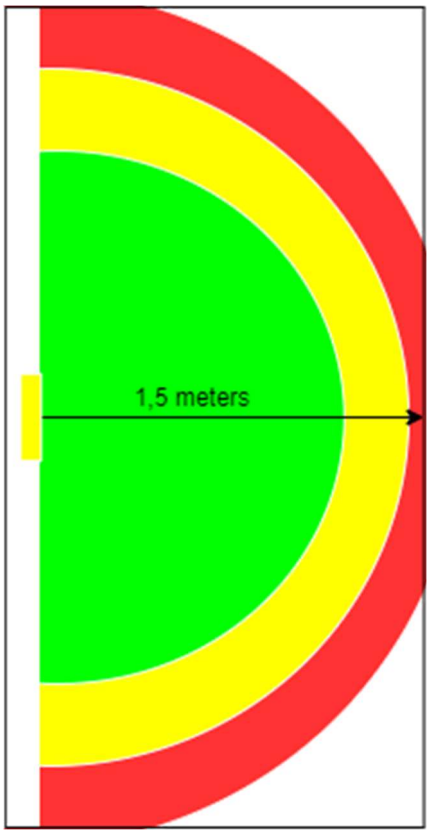

Figure 26 PIR functionality tested to workin this manner

4.3.3 Ambient Light Sensor

Goal: React to light levels in room, but avoid feedback issues with lamps themselves

Goal was met to some extent. No dynamic calibration was implemented, but from testing we found that the lights themself cause limited feedback in a dark room. Even from close range to lamps ambient light reads zero while in a dark room. Yet this is not fool proof and can be taken to its logical conclusion of placing the lamp right next to the sensor which will cause feedback.

Summary: Sensor works in most conditions as expected but will have issues function in certain situations and configuration. A software calibration would be ideal to avoid this issue.

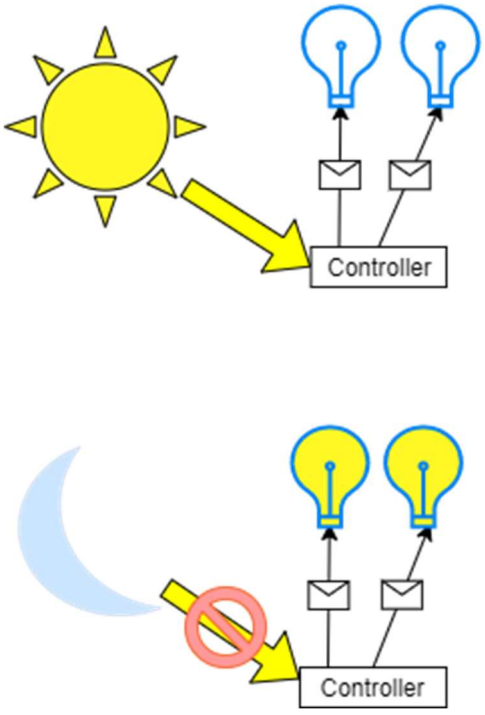

Figure 27 Ambient functionality

4.3.4 Ultrasonic sensor

Goal: Add gesturing functions to device.

Goal was met barebones, but after research on our sensor this might be as far as it could taken in a reliable manner. Actual gesture sensor exist that could replace this sensor to add functionality from directional swipes and push/pull movements.

Yet the toggling function worked as expected and worked with current sensor with reliability and ease of use. Having this manual setting is important since all other sensors are designed to act more independently reacting also to indirect interaction

Summary: In current setup sensor works in a toggling manner and is a good function to have. Suggested next step is to replace sensor with a gesture sensor or spending more time investigating ultrasonic sensor capabilities.

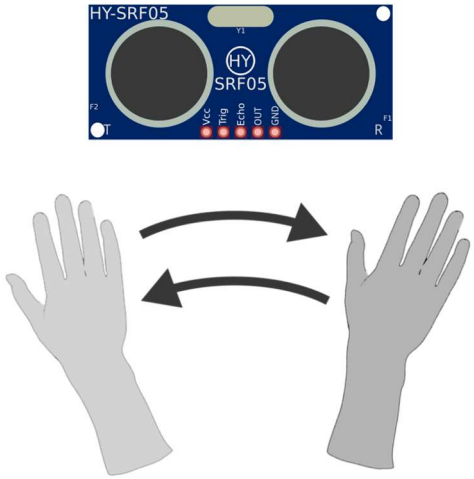

Figure 28 SRF05 toggling functionality

4.3.5 Website

Goal: Allows setting properties in a functional manner

Frontend development was not a goal for this project nor was there any special interest for any group member to make especially good-looking website. Its minimum functionality was met. The site works well on a laptop and desktop but is not responsive to mobile devices. This leads to some usability problems, but functionality is not lost on mobile devices.

Only major limiting factor is compatibility. Windows devices require chrome, edge or opera to connect, android devices require 6.1 version and chrome or opera app, ios devices are unsupported and can only be connected with workarounds.

Issues we met keeping the Bluetooth link active, but was not investigated, because some basic functionality was still being worked on.

Summary: Website works as expected, suggested future development would be making the site more secure, working on a responsive site for mobile browsing and checking the Bluetooth link expiration issue. Wider client device compatibility would require restructuring.

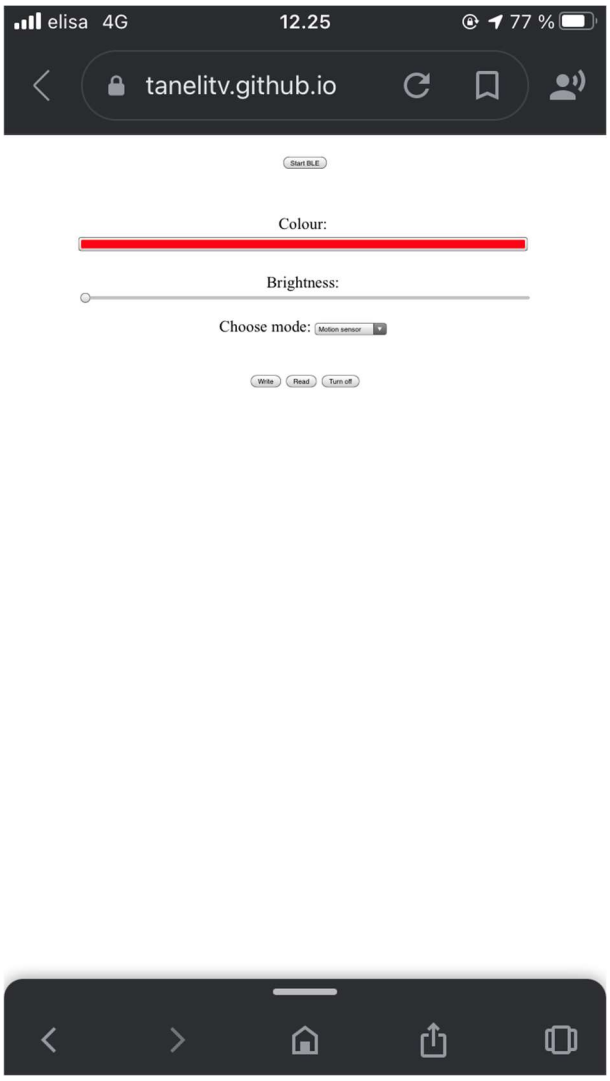

Figure 29 Mobile view

Appendix 1

Checksum

- Small data payload that is added to the frame and used to verify integrity of a command line by the receiving device.

Frame

- Term for a data packet used in XBee API mode.

Freertos

- An open source real-time operating system designed for microcontrollers. Currently under stewardship of Amazon but keeping to its open source roots.

GitHub

- Code hosting platform for version control and collaboration. Site is managed by github.inc a subsidiary of Microsoft.

Grid-Eye

- Infrared array sensor with a thermopile-type infrared sensor which detects quantity of infrared ray present. A registered trademark of Panasonic.

HTML

- HTML or Hypertext Markup Language is the standard language for documents designed to be displayed in a web browser. Current version designed by WHATWG steering group.

HTTPS

- Hypertext Transfer Protocol Secure (HTTPS) is an extension of the Hypertext Transfer Protocol (HTTP). HTTPS is used for secure communication and is widely used- Approximately 70% of websites use HTTPS. Managed by Netscape Communications.

Light Link

- ZigBee Light Link (ZLL) is a global standard for interoperable and easy-to-use consumer lighting and control products. Light Link covers home automation and most smart lightning companies have adhered to this standard enabling easy-to-integrate lightning.

LPC

- Microcontroller developer board range manufactured by NXP

MQTT

- Lightweight messaging protocol designed small sensor and devices. Designed to operate in suboptimal environments. Managed by Organization for the Advancement of Structured Information Standards.

Raspberry PI

- Miniature single board computer which has found popularity in IoT and Education. Designed by Raspberry Pi Foundation.

Trådfri

- Smart lighting range adhering to ZigBee Light Link standard. Manufactured by Ikea.

Ultimaker Cura

- 3D printing software to convert 3D files to a 3D printer compatible Gcode command file.

Wi-Fi

- wireless network protocols, which are commonly used for local area networking of devices and Internet access. Registered Non-profit trademark of Wi-Fi Alliance.

XBee

- Range of ZigBee chips manufactured by Digi.

XCTU

- Free to use desktop application designed by Digi for XBee chip configuration and testing.

ZigBee chip

- A physical chip that enables ZigBee communication in tandem with another chip/existing network. Doesn't strictly require extra peripherals to operate such as antennas etc but can improve operation

References

1 Figure 11, ESP32, https://www.partco.fi/fi/kehityskortit/esp/20179-esp32-wifible.html

2 Figure 12, Freertos, http://www.emcu.it/STM32F4xx/Exe2_FreeRTOS_on_STM32F4-Discovery/EXE2_FreeRTOS_on_STM32F4-Discovery.html

3 Figure 13, XBee chip, https://www.digikey.fi/product-detail/en/digi/XB24CAUIT-001/602-1893-ND/6010102#images

4 Figure 14, XBee Grove development board, https://www.digikey.com/en/products/detail/digi/76000979/7325219?s=N4IgjCBcoLQdID-GUAuAnArgUwDQgPZQDaIAbAAwBMIAugL517WQkDsF55AnK17XUA

5 Figure 15, PIR motion sensors, https://shop.4tronix.co.uk/collections/crumbs/products/pir-sensor-crumb-digital-input-for-crumble-controller

6 Figure 16, Ultrasonic range sensor, https://www.wish.com/product/553b1010bd460e305e36465d?hide_login_modal=true&share=web

7 Figure 17, Ambient light Sensor, https://www.partco.fi/fi/arduino/arduino-leikkikenttae/20363-ktr-5105.html

8 Figure 19, ZigBee Mesh, https://www.incibe-cert.es/en/blog/security-ZigBee-communications

9 Figure 20, GATT, https://learn.adafruit.com/introduction-to-bluetooth-low-energy/gatt

10 Figure 28, Gesturing, https://arduibots.wordpress.com/2015/10/12/component-hy-srf05-for-fritzing/