OULUN AMMATTIKORKEAKOULU

Selina Kallio

LAB MONITORING IN A MODULAR DATA CENTER

Implementing a Microcontroller Management System

LAB MONITORING IN A MODULAR DATA CENTER

Implementing a Microcontroller Management System

Selina Kallio
Bachelor's thesis
Spring 2024
Degree in Information Technology
Oulu University of Applied Sciences

ABSTRACT

Oulu University of Applied Sciences
Degree Programme in Information Technology

Author: Selina Kallio
Title of thesis: Laboratory Monitoring
Supervisor: Olli Himanka
Term and year when the thesis was submitted: Spring 2024
Number of pages: 39 pages and 0 appendices

Due to the ever-growing highlight on environmental factors, the conservation of energy is receiving focus. Data centers can be a huge consumer of electricity in companies. Electricity use in cooling these data centers reflects straight to the used company resources, remarkably monetary savings can be found. The project was done for Nokia, who is currently building a new site to Linnanmaa, Oulu. The energy efficiency at the new site will be state of the art – an element coming into effect will be the harvesting of heat generated by the many devices in the data centers.

The implementation of hot/cold aisle functions can save over a third of the energy that would normally be used. To make functional aisles, the air must be truly separated. This is done with modular ceiling panels and automatically functioning laboratory doors. Open doors or ceilings lead to loss of heat and therefore loss of energy. This project took focus on monitoring the ceiling panels. They are placed on rails on top of the rack bodies many data centers consist of. Some devices are located above the panels. Also, power outlets and cabling are done outside the aisle formation. Human errors lead to forgetting to put the panels back into place after modifications, even being left open for multiple days. In small spaces this can cause the temperature to rise which creates its own risks for the environment, such as negatively impacting the longevity of device parts.

The data center ceiling panel monitoring in this minor proof-of-concept project was completed using microcontrollers and creating a circuit with copper tape. Despite the copper tape not being a durable solution, it was the easiest and most time and cost-effective solution. The circuit was formed to the top of the data centers, consisting of pieces of tape on both the ceiling panels and the rails in a specific formation. The layout of the tape defines how the microcontroller will react. The microcontroller connected to the circuit sends the circuit's current status to its superior microcontroller. The superior microcontroller in turn alerts users of the data center's situation.

The goal of the monitoring software was to notify users after a specific period of the ceiling panels being open. This enabled the faculty to go and confirm the laboratory status and fix the faults causing the failed status. The concept worked and based off the research, a solution such as this one could generate electricity savings. However, some flaws were discovered in the physical implementation side – the tape degraded under long-term stress and abrasion. This could be prevented in progressed models using harder metal embeds or microswitches instead of copper tape.

Keywords: Microcontroller, Embedded system, Raspberry Pi, GPIO, Micropython, MQTT, SNMP

CONTENTS

ABBREVIATIONS AND TERMS

IoT	Internet of Things
PoC	Proof of Concept
GUI	Graphical User Interface
HVAC	Heating, Ventilation, and Air Conditioning
GPIO	General Purpose Input Output
Wi-Fi	Wireless Fidelity
HTTP	HyperText Transfer Protocol
HTTPS	HTTP (Hypertext Transfer Protocol) Secure
CoAP	Constrained Application Protocol
MQTT	Message Queuing Telemetry Transport
SNMP	Simple Network Management Protocol
Raspberry Pi	Microcontroller brand, producer of Raspberry Pi 4 and Pico W
ESP32	Microcontroller
CPU	Central Processing Unit, a processor
(Micro-)python	Programming languages
DUT	Device Under Test
WLAN	Wireless Local Area Network
ESD	Electrostatic-sensitive device/ electrostatic discharge
EL	Environmental level
EPA	ESD Protected Area
ESDS	Electronic Component, Assembly, Module, or System, Sensitive to ESD Damage from 20,000 volts or less
ILM	Interactive Laboratory Maps
DNS	Domain Name System
DHCP	Dynamic Host Configuration Protocol

PREFACE

This bachelor's thesis was made for Nokia in the Spring of 2024. It aimed to find a way to monitor laboratory ceiling panels that were being forgotten open, resulting in energy loss.

I started as a trainee at Nokia in my first year of my studies. New to the world of information technology, I was eager to learn. Since then, I have been taught so much about networks, cybersecurity, coding and how to be a better engineer and person. Thank you, Juha-Matti Malila for seeing the potential in me. Thank you, Lassi Myöhänen, for being my buddy at Nokia from the moment I stepped inside. You gave me the idea for this thesis and an immense amount of support during it.

My team, Nokia LabOps Oulu, has become like a family to me these past few years. Special thanks to my work-dad Niko Kelloniemi. Niko provided help throughout my thesis despite his busy days and implemented his expertise in his guidance.

Finally, thank you, Olli Himanka, for being my thesis supervisor. Thank you for your input and comments! You are a pleasure to work with and a great teacher.

I would like to extend my thanks to everyone involved in my thesis for your invaluable help, I really could not have done it without you.

1 INTRODUCTION

With an ever-growing need to save energy and live within the boundaries of the Earth, energy saving, and frugal use of components has become a priority for many enterprises and a factor that consumers are looking for in a company. This makes air monitoring critical in corporate data centers. Excess heat can do significant damage to electronic devices, which is why most devices have standard levels of heat they can tolerate. So: hot air is not a good solution but cooling the air too much is not energy efficient either. Keeping the input air of the devices at an optimal temperature can lengthen their life span, saving components and energy in the process.

As devices in data centers are laid out in average room-height rows, to optimize data center energy efficiency in terms of electricity consumption, hot and cold aisles have been developed to optimize device performance and efficiency. In the human world, mistakes are possible and not rare. By adding mechanical monitoring to the mix, human errors – such as forgetting the ceiling panel of an aisle accidentally open – can be erased from the equation. Normal procedures such as opening the panels can lead to wasting energy if they are not taken care of. The goal of this project is to monitor whether the aisles are closed off properly – an idea that can later be automated even further, for example, even to close the panels. This project will aim at recognizing a gap in energy efficiency and alert users.

Saving energy and thus caring about the environment is a huge selling point and the reason for this project – by making sure to be energy efficient the project can bring fiscal savings to Nokia and be implemented around the world resulting in an array of savings. This thesis ponders the implementation of an IoT (Internet of Things) system to data center aisles that monitors temperature status as well as panel placement, a critical factor to sustain stable data center conditions. The documentation begins with assessing and evaluating optimal materials and ways of working and conducting small PoC (Proof of Concept) experiments to weigh different options.

Energy saving remains a core aspect throughout all the elements of the project from selecting low-powered microcontrollers to combining the monitoring GUI (graphical user interface) to an existing server and website. The documentation continues to system design, exploring solutions to the basic criteria of the project: how to convey the monitoring results and how the metaphysical virtual side

of the project will be implemented. The documentation ends with the results of a larger scale PoC implementation.

2 THEORETICS: MATERIAL AND SOLUTION SELECTION

Planning this project to be modular and with a possibility to easily expand and modify elements, it is important to compare and weigh in on the benefits and limitations of solutions. The documentation contains an implementation to one to two lab aisles. The aisles are rows of cool/warm air, surrounded by electronic devices such as radios and switches on both sides as shown in FIGURE 1. This implementation of a hot/cold aisle system can lead to relatively big savings in cooling expenses – even up to 35% (2). The devices are placed in rack cabinets, a few meters high. These HVAC (Heating, ventilation, air conditioning) control systems are implemented in data centers around the world (3). As some of the electrical laboratories are ESD-protected, electrostatically approved materials must be selected or available for future implementations.

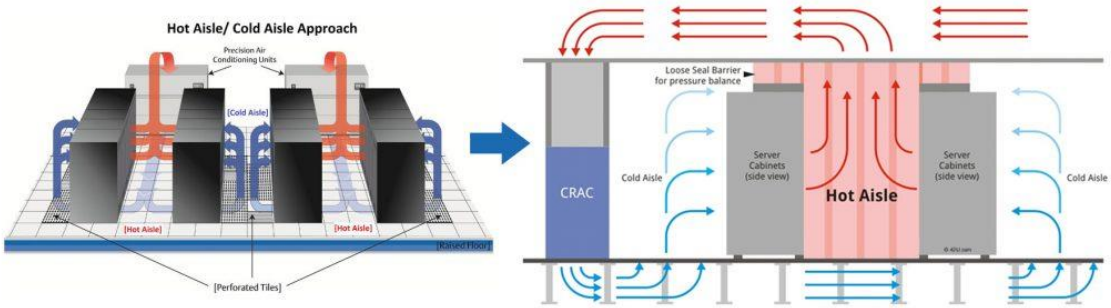

FIGURE 1. Hot/Cold aisle functionality (1.)

The physical part of the project begins with creating a circuit by applying conductive material around the plates or by ensuring their placement with an alternative method. With any conductive material or switch alternative, the method is very similar with same physics standards at the foundation: creating a circuit. When all the plates are down and correctly placed, according to electric physics, a circuit should be formed here (4). Other ways of creating a circuit will be weighed within the next few chapters.

Many modern microcomputers and microcontrollers have GPIO (General Purpose Input Output) pins. If one pin is set to “1” (HIGH) and another pin receives its digital input. If the pins are connected, the second one should also produce an output of “1” – HIGH – receiving the input from the first pin (5). Applying this to the project: if the pins are connected via circuit (the panes in the ceiling of the data center) if all the plates are in position, the second pin will confirm it. (6.)

2.1 Comparing microcontrollers

The microcontrollers allocated for this project do not need to be high performance, as they theoretically only have one task. They monitor the status of the second pin, sending the results to a database. Two good producer alternatives are Arduino and Raspberry Pi. Arduino has many microcontrollers, some considerable examples being Uno, Tiny and ESP32. Raspberry has models such as Nano and Pico, both available with multiple models with a selection of features such as Wi-Fi (Wireless Fidelity) connection and either sockets or pins (7.). The overall price of the device and the electricity consumption are critical factors to consider in comparing producers. The model selection is varied to provide a wide range of alternatives for different purposes (8). Things to consider when selecting a microcontroller are for example: how many and how challenging (CPU-heavy and complex) tasks will it be required to complete? How many devices will the implementation require, and with what kind of budget? Does it require network access – Ethernet or Wi-fi? How easy is the development environment; are there many libraries, tools, and pre-existing assets?

Alternatives are considered in a blog article titled *Exploring ESP32 Alternatives: A Comprehensive Guide for Newbies*. Many of the aforementioned devices are considered and their feature lists are compared. ESP32, a rather "overkill" microcontroller is a great alternative for projects and tasks requiring an efficient and highly interoperable device. However, the learning curve may be rather steep as the Arduino-based device has very advanced functionalities and is written in C alternatively to for example, Raspberry Pi Pico, which supports both C/C++ toolchain and Micropython. In addition, the ESP32 is not nearly as energy efficient as the compared devices. (9.)

A top contender mentioned in the article listing ESP32 alternatives is the Raspberry Pi Pico (FIGURE 2). Pico has many models, with possibilities of sockets, pins, Wi-Fi and more. The Pico is comparatively budget friendly. A key point in the project is modularity, which is Raspberry Pi's selling point for their Pico. This makes Pico an ideal competitor and option/alternative for the final execution/implementation. As already mentioned, Pico reads Micropython scripts, which are simpler to write than C-based languages, especially for a use-case like this where functionality and simplicity are key elements. As Python is a widely known, fast and renowned language, the Raspberry options are easier and more time efficient.

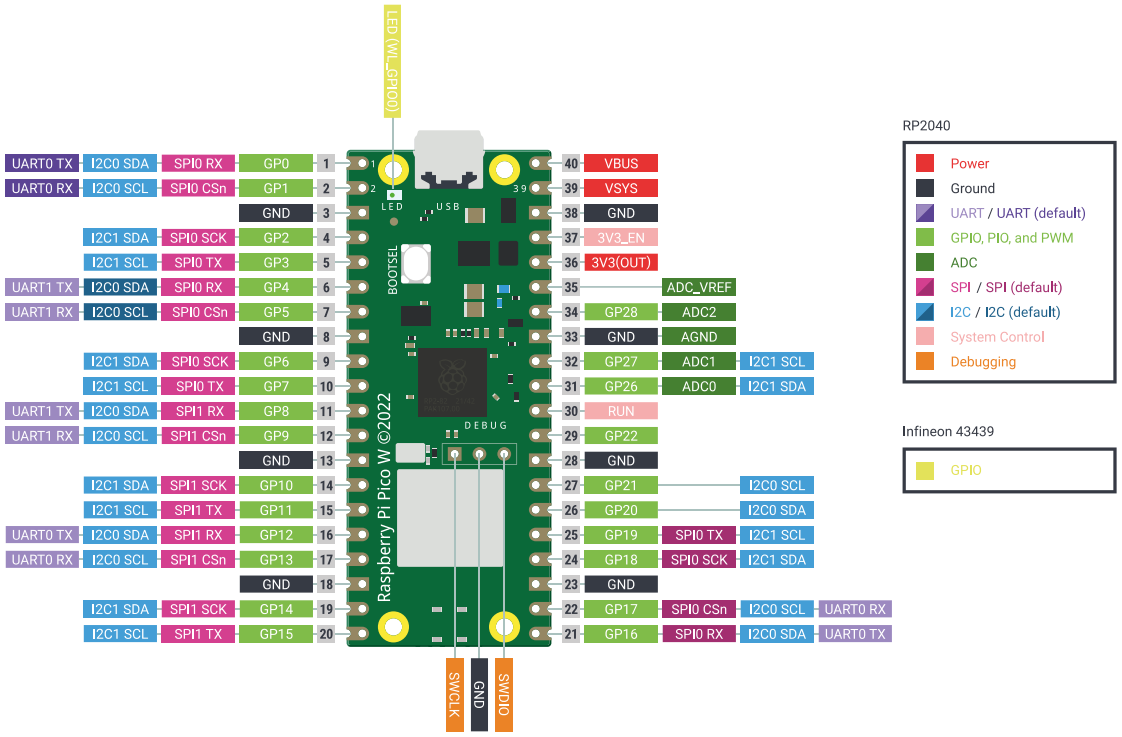

FIGURE 2. Raspberry Pi Pico W pinout

Arduino's Nano 33 IoT, like the Raspberry Pi Pico, is an efficient small device with low-power operating skills in addition to modularity and computational power. It is also beginner-friendly compared to the ESP32 with lots of quick guides that can be found online. (10.) Another competitor is the Particle Argon. The Particle Argon relies on Nordic Semiconductor's chipset, making it a more European option especially compared to the ESP32, from Espressif, a China-based corporation. In addition to being a more local alternative, the Particle Argon offers many of the advantages of the previous devices. (11;12;13). The only issue: it is designed to work in Particle's ecosystem, which could complicate future modularity matters. Nordic Semiconductor's other microcontrollers, i.e., from the nRF52-series would also be considerable options, however they were opted out of the list due to budget constraints in a corporate project.

As the project is to take place in a corporate environment, and the project may possibly be scaled up in the future, the budget is a vital part. That is why, comparing all the alternatives for microcontrollers, the Raspberry Pi Pico W was finally selected for use. The Pico W is cost effective, small, easily implemented, and adaptable. The main "hub" – an action controlling intermediary unit – is a Raspberry Pi 4, with great compatibility to the Pico.

2.1.1 Communication protocol and server compatibility

Many recognize HTTP and HTTPS from web-URLs. They are communication protocols widely used in transferring data. Communication protocol is required to send data between two or more devices (14). HTTP (Hypertext Transfer Protocol) and HTTPS (HTTP Secure) are just a few of the possibilities for communication protocol alternatives. CoAP (Constrained Application Protocol) and MQTT (Message Queuing Telemetry Transport) are also considerable options – MQTT is the industry standard for many IoT implementations (15).

The communication protocols must accommodate both the needs of the microcontroller and the server. LibreNMS, a device monitoring software can only get data by SNMP polls. The Raspberry Pi Pico cannot handle SNMP (Simple Network Management Protocol), but usually uses MQTT. Therefore, a middleman is required and all the pieces in the communication must be well thought through.

MQTT is lightweight and efficient, making it a perfect match for the Raspberry Pi Pico with limited resources (16). MQTT implements a pub/sub (publish/subscribe) model, which allows for efficient communication architecture and a flexible and scalable approach to server contacting, making it the ideal solution for communication from the small Raspberry Pi Pico units to the more central Raspberry Pi 4 units. (15;16.)

As the data still needs to be transferred to a server for analysis and storage, an addition to MQTT is required. A considerable option is to use a preexistent monitoring solution, such as LibreNMS. It uses SNMP (Simple Network Management Protocol) to fetch data to its server where it can be processed more (17). SNMP is a protocol used for network management and monitoring. It allows devices on a network to be monitored and controlled from a central location. SNMP sends messages to different parts of a network. An open-source network monitoring and management system like LibreNMS that uses SNMP to gather and organize data from network devices can fulfill the rest of the communication needs. (17;18.)

LibreNMS supports a wide range of devices and can be particularly useful for monitoring large and diverse networks, making it eligible also for harvesting the data from the central Raspberry Pi 4 – units. Raspberry Pi Pico and its limited resources do not inherently support SNMP (19). SNMP is more commonly used with network devices like routers, switches, and servers that run full-fledged operating systems. SNMP messages can be used to retrieve information from network devices, set configurations, and receive notifications about specific events; in this case the messages can be used to collect sensor/circuit data. (18.)

2.2 Panel status from circuit integrity

In a massive data center environment with busy engineers focusing only on their own issues, ease of maintenance and low failure rates are critical points of focus. With a small project such as this one, it will be hard to find a successor with time and energy to maintain the work and upkeep after a few years. This has been seen in many siloed projects at larger companies – a task is just in the upkeep of one person and as that person is unable to continue it, the project gets cut-off, discontinued or worse (20; 21). Analyzing the reliability of different parts of this project, the circuit-forming steps seem the least reliable and most vulnerable to problems, incidents, and wear over time.

The documentation and physical part of the process revolve around tracking the exclusion of the hot/cold aisle from outside air. This can be done in multiple ways, but a common theme is the creation and following of a circuit. If the circuit is cut, a message is sent. The ceiling panels must form a circuit, as mentioned in the previous chapters.

The first alternative to circuit measurement considered was copper tape. Affordable, easy to source, easy to install and relatively easy to upkeep in laboratory conditions. Consisting of a thin copper layer on adhesive, the tape is easily applied to necessary parts shown in FIGURE 3, where it is depicted on a 3D (3 Dimensional) model of the data center ceiling. (22.)

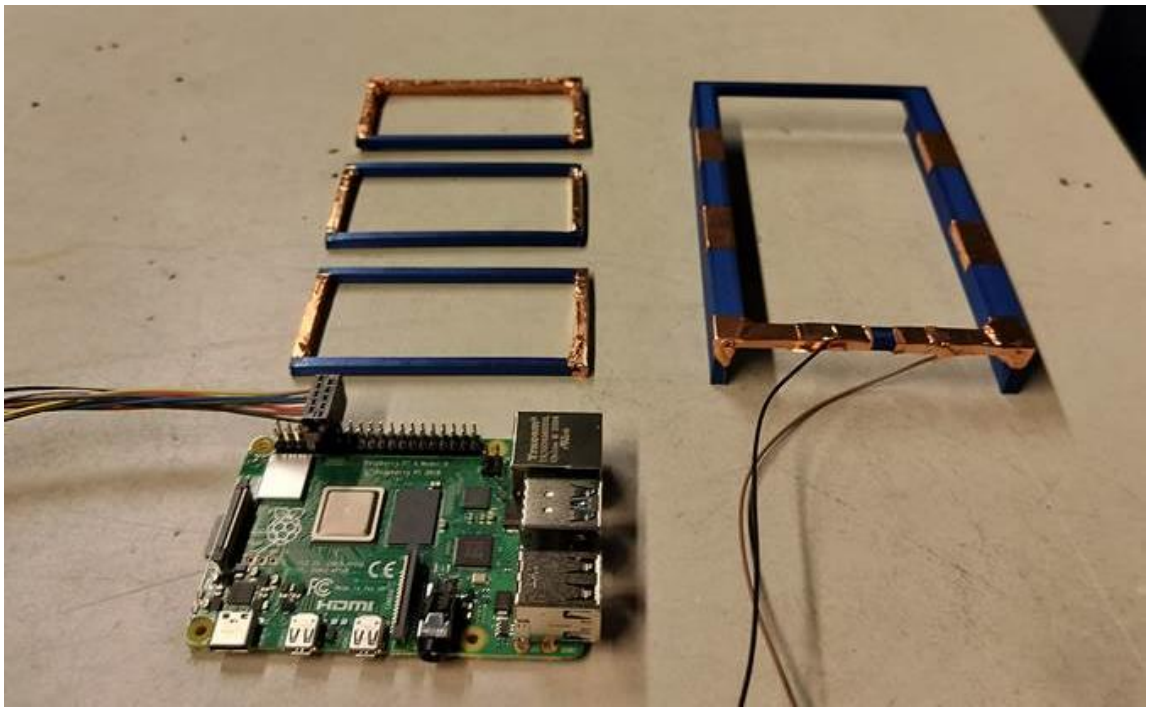

FIGURE 3. A rough draft of the project configuration with a 3D-printed lab model and a Raspberry Pi 4

The physical part of the project begins with applying conductive material to the bottom of the top plates. The conductive material will be placed also along the meeting points of the plates, but on the rack side. With the low moisture levels in professional device spaces, copper tape has a low oxidization rate which eases maintenance. The tape must, however, be cleaned and inspected at intervals. According to preliminary tests concerning conductive material durability tests, broken/affected parts can easily be spotted and replaced. However, with multiple panels and only minor contact loss, finding the issue may prove more difficult. This can however be solved by dividing aisles into smaller sections. Copper is very conductive, which in addition to the ease-of-use – especially compared to more mechanical circuit-forming alternatives – is why copper tape is often found in electronics prototyping and temporary circuit setups. (22.)

Considerable alternatives to copper foil tape are conductive fabric tape and adhesive strips. They unfortunately do not offer such high levels of reliability and durability as copper tape does. Other types of tape may exhibit higher resistance compared to copper tape. As demonstrated in experiments, copper tape has very low resistance. However, tapes with higher resistance can lead to the voltage dropping so much that the pin at the other end would fail to receive the high input. (22; 24.)

In lieu of conductive tape conductive paint can be regarded. The paint is applied in a thin layer to the places where the tape would also be placed (23). As also the tape alternatives, conductive paint does not do well in high-stress environments. Reapplication could also be required due to wear.

The paint and tape act as switches informing the Raspberry Pi Pico that the circuit is formed. Alternately, a physical switch or spring-loaded contact can be used. Switches add a more interactive element: when the plates are connected to the racks, the switch is closed, signaling continuity of the circuit. Switches – especially high-quality ones – are very reliable and can last for years even with heavier use (25.). However, the maintenance cleaning process of switches is a bit more complex than wiping smooth copper-taped surfaces. The switches also require replacement time-to-time, which requires knowledgeable staff. Microswitches, displayed in FIGURE 4, could be the best solution for an accurate result in this kind of environment and situation, albeit requiring a more thought-through installation. The small pins on the side of the microswitch would be pinned down by the weight of the panels, and this would be signaled to the current.

FIGURE 4: Microswitches (SparkFun Electronics/CC-BY-SA-2.0)

Spring-loaded contacts/connectors can be used to track the connection between the plates and racks. Physically, they work in a similar way to the microswitches in Figure 4. When the plates are pressed against the racks, the spring-loaded contacts ensure a continuous circuit. Spring-loaded contacts and pogo pins are used commonly in testing purposes to alert devices of proximity to another object and to ensure reliable electrical contact to a DUT (Device Under Test). As with the switches, reliability can be counted on if the contacts are well-designed and maintained (26; 27).

Maintenance is also parallel to switches: a bit more intricate cleaning is required, and personnel require education on the topics – again, making the project harder to maintain as a legacy.

Also, non-circuit alternatives to laboratory measuring exist in the form of sensors and can be implemented for nearly as low prices as circuit alternatives however requiring more work – a tedious task in a big physical data center location.

As the goal is to make sure that the top of the rack center area is closed, the area should be hotter/colder than normal (due to the hot/cold aisle formation), radical temperature change could implicate that a ceiling plate has been left open. However, this system is also vulnerable to other changes: i.e., in case a device has been shut down the aisle cools down.

Other options include measuring changes between the plates and the ceiling. This can be done via ultrasonic distance sensors, light sensors, capacitive sensors, or force sensors. Ultrasonic distance sensors, little boombox-looking electric devices, can detect the distance between plates from around 20mm up to a few meters (if the sensor result is over 50mm, most likely the plate is out of position). Overall, changes in distance may indicate misalignment or movement of the plates. (28.)

These – switches and electrical components in general – require more effort in installation. Especially in a small-scale project such as this one, it is simply too much work. With the amount necessary for larger scale implementation, the price will get very high, ultimately not making these sensors the best possible option.

2.2.1 ESD concerns

In laboratories with sensitive testing equipment, ESD (electrostatic discharge) is a matter of concern. ESD can mean either electrostatic discharge or electrostatic-sensitive device. The Oulu premises of Nokia have both non-EL and ESD-sensitive laboratories ranging from EL1 to EL3 (environmental level 1 and 3, respectively). ELA ratings are defined by ESD-sensitivity, EL1 being the most ESD-sensitive area. EL refers to a classification of environments based on their sensitivity to electrostatic discharge (ESD). This classification is part of a broader set of standards that helps ensure the safety and integrity of electronic components in various working spaces.

The ESD concerns of the project can later be assessed as they are not of high criticality in the PoC phase and preliminary data shows that in current laboratory conditions they do not pose a high ESD risk. Electrostatic Level 3 (the ELA-level in the test lab, home of the PoC), means an environment where electronic components and assemblies are not highly sensitive to electrostatic discharge. The requirements for grounding and protective measures are less strict compared to higher EL levels. Normal protocol in place in EL3 areas are basic ESD control measures like anti-static mats, wrist straps, and grounding points. ESD shoes are a good commonly used grounding point for persons working in EL areas. ESD-sensitive devices must be marked with signs. FIGURE 5 shows the symbol that is visible in both ESD shoes and ESDS. A Similar yellow-black sign with a hand symbol an EPA.

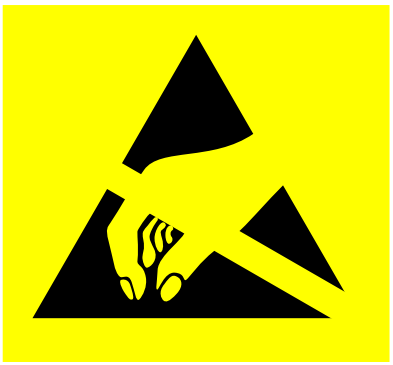

FIGURE 5: The ESD susceptibility symbol, used on devices with a sensitivity to ESD

Despite being less sensitive, EL2 and EL3 environments can still face issues if basic ESD control measures are not maintained. For example, with improper grounding (i.e., not using ESD shoes) problems can arise and equipment can be damaged.

The main ranking component when deciding the environmental level in a laboratory is component sensitivity. An EL3 environment may be suitable for components with a higher threshold for electrostatic discharge. For example, components that can withstand 2000V ESD without damage can be safely handled in an EL3 space. Regular audits are held to ensure the correct ELs are in place.

Laboratories with open devices are much more susceptible to damage from ESD, resulting in higher environmental levels. Highly sensitive components like advanced microprocessors, semiconductor

devices, and other intricate electronics are highly sensitive to even small ESD events. These components might be damaged by voltages as low as 100V or less, necessitating very strict ESD controls.

FIGURE 6: Cling caused by static electricity (Sean McGrath / CC-BY-SA-2.0)

Static electricity is automatically produced with the contact of surfaces. The charge of static electricity is held until release, bringing a threat to electricity sensitive devices (ESDs). The shock of charge neutralization is an electrostatic discharge and can be even up to tens of kilovolts in normal conditions. Typical voltage from friction and contact between clothes and varying items can range from 7 to 21 kilovolts. The cat in FIGURE 6 is carrying a lot of static electricity – a charge that attracts the elements with the opposite charge, the packing peanuts in the photo. With proper grounding the static electricity can be released. Most likely the wooden floor in the figure is too insulating that the static charge cannot be released. If the floor was conductive and grounded, the electricity would be released, and the packing peanuts would fall off. (29.)

2.2.2 Durability testing

The durability and performance of components, i.e., copper tape and alternatives, within the data center environment can be assessed through tests covering factors ranging from temperature-related durability, abrasion resistance, to environmental conditions such as moisture and heat levels.

Temperature-related tests evaluate the ability of components to endure thermal stress without compromising conductivity or structural integrity. As previously mentioned, copper tape has a lower resistance making it good for preliminary and testing use. High frequency switching tests can be conducted on switches to simulate repeated use, however, for the PoC, switches were discarded as an idea. Also, abrasion resistance tests can be performed on copper tape and switches to get information on their performance over time and their resistance to wear and potential damage.

Despite not doing formal stress testing, wear in the copper tape was noticed over time, as illustrated in FIGURE 7. While wear in the copper tape may be noted, it is important to acknowledge that this is not a currently a concern, because the copper tape is utilized only in the Proof of Concept (PoC) phase and will be replaced with full metal plates or something stronger than copper tape in the final version. Overall, this wear is not a real issue as the copper tape will be replaced with a better solution later if the project goes into mass production.

FIGURE 7. Copper tape tearing in PoC data center

There are many other topics that could later be considered, including the copper oxidation due to humidity and temperature levels in the laboratory. Also, tests could be conducted to determine the level of impact of those factors (humidity and temperature) to the functioning of copper tape alternatives, such as switches. More information on the overall lifespan of switches and other components could also be gathered from hardware installation/maintenance teams and analyzing data from the manufacturers.

3 SYSTEM DESIGN AND IMPLEMENTATION

3.1 Core functionalities

The core functionalities of the project have three main aspects: monitoring ceiling panel status, timely alerting of any issues, and making the locating of open panels easy. Regarding the first aspect, the status of ceiling panels, concisely: the status will be visible in real-time in an internal tool, showing the open ceiling area on a map of the premises, enabling efficient monitoring and management. This topic will also be comprehensively addressed in Chapter 4, titled Web Interface and Virtual Monitoring.

The second core functionality involves alerting to ensure prompt response and reaction to any deviations in panel status, achieved through the utilization of LibreNMS, a network monitoring system already integrated into the data center environment. By configuring LibreNMS to send email alerts in response to predefined triggers, relevant lab personnel will be promptly notified of any issues requiring attention. This is discussed more in Chapter 5, Monitoring

The third core functionality focuses on helping to locate open panels within the facility by implementing lights outside laboratory aisles and the laboratories themselves. These lights will change color in accordance with the status of each panel, providing a clear visual cue to the status of rooms and aisles and assisting personnel in locating open panels quickly and efficiently. A more detailed cross-section of these core functionalities, including their implementation and integration into the overall system architecture, will be covered more thoroughly in Chapter 5.

3.2 Communication design

A script running on the Raspberry Pi Pico W collects data from GPIO pins and publishes it to an MQTT broker as shown in FIGURE 8.This PoC utilized Mosquitto as the MQTT broker. A Raspberry Pi 4 serves as an intermediary, running the Mosquitto broker and subscribing to the MQTT topics and translating the received data into a format compatible with SNMP. The Raspberry Pi 4 has additional libraries added for dealing with LibreNMS polling, such as WiringPI and rpigpiomonitor

(30). The Raspberry Pi 4 ensures that the further links of the chain receive the correct data in correct form, removing erroneous messages and sending out only appropriate data. As it is normal for the panel to be open for several minutes, the Pi ensures that alerts are only put out on the web interface at a delay – after the panels have been open for over 60 minutes.

```
# MQTT configuration
CLIENT_ID = "pico"
MQTT_BROKER = "192.168.1.15"
MQTT_PORT = 1883
MQTT_TOPIC = "test"
```

```
# Initialize MQTT client
def init_mqtt():
    client = MQTTClient(CLIENT_ID, MQTT_BROKER, port=MQTT_PORT)
    return client

# Publish status via MQTT
def publish_status(client, status):
    try:
        client.connect()
        client.publish(MQTT_TOPIC, status)
        print("Status published:", status)
    except Exception as e:
        print(f"Failed to publish status: {e}")
    finally:
        client.disconnect()
```

FIGURE 8: The configuration for sending MQTT messages from the Raspberry Pi Pico W

LibreNMS is configured to poll the Raspberry Pi 4. The SNMP polling from LibreNMS retrieves the translated data from the Raspberry Pi 4 and adds it to the database. Finally, the status data from the Raspberry Pi 4 is accessed by ILM (Interactive Lab Maps), an interactive map service for the data centers where the data is visually displayed.

These steps of the process required additional attention due to the nature of the environment: confirming proxies and DNS (Domain Name System) addresses prolonged the process and brought challenges which were eventually solved. To download MQTT and SNMP (and SNMPD – an SNMP daemon) handlers, external sites had to be accessible to the Raspberry Pis.

3.3 Network design

As shown in FIGURE 9, the Raspberry Pi 4 must have access to both the lab network and the Raspberry WLAN at the same time. This is so it can be reachable for the SNMP queries from LibreNMS and able to receive MQTT messages from the Pico. However, for the Raspberry Pi Picos, it is enough that they are only in the WLAN.

In this project, the router providing the WLAN managed to supply an entry into the Raspberry Pi 4 from the normal lab network. This way, by contacting the router, the Raspberry would answer.

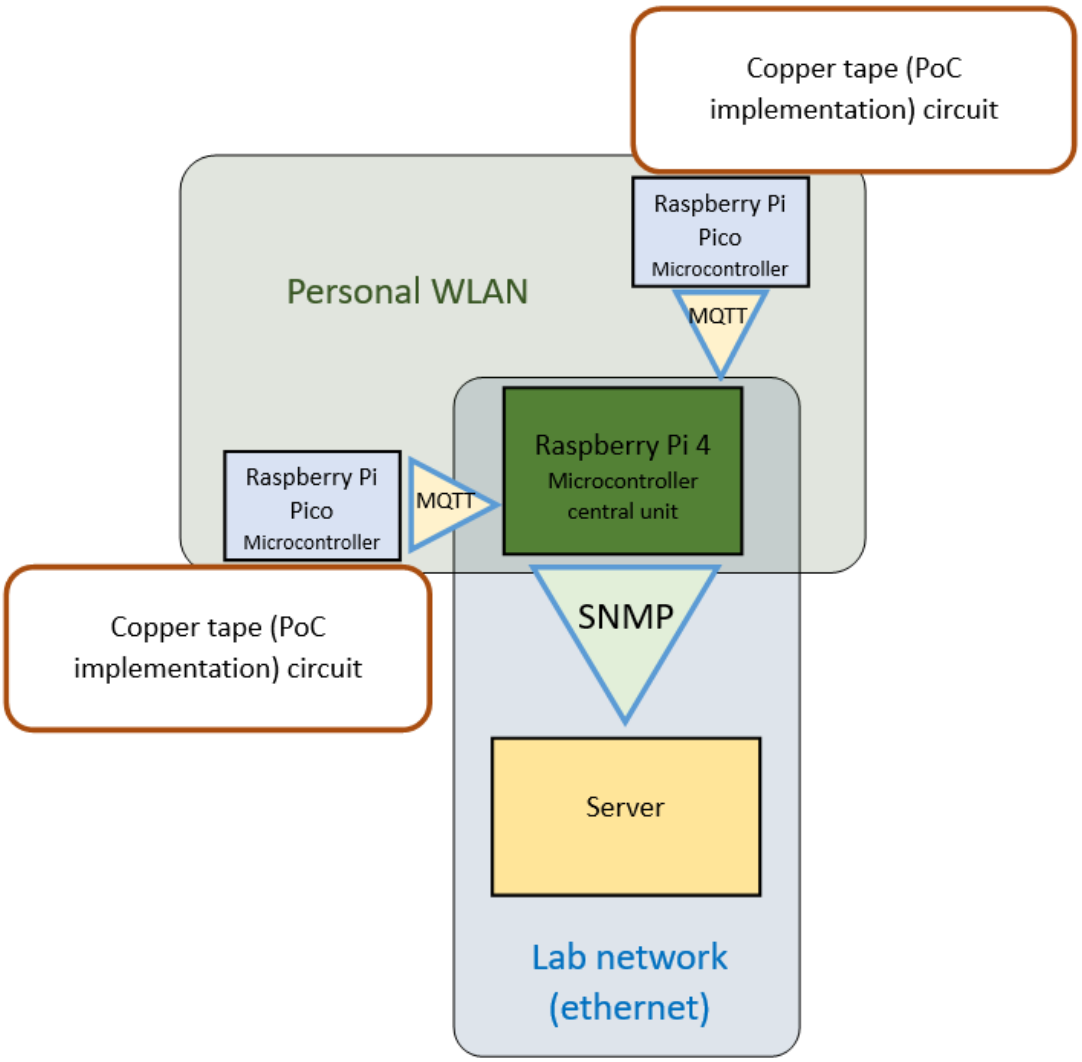

FIGURE 9: The simplified PoC laboratory setup: devices, networks, communication

A router was setup in a small data center area functioning as a base for the PoC, providing a way for the Raspberry Pi family to communicate via MQTT – also visible in FIGURE 9 and with more information found in chapter 3.2. The SNMP polls and contact to the ILM software require the Raspberry Pi 4 to have access to the laboratory network. The lab network is only available with an Ethernet cable, and Pico Ws do not have the capability for non-Wi-Fi (ethernet) network connections. The project was modified to accommodate the WLAN for the Picos and FIGURE 10 shows the process of how the Raspberry Pi Pico W connects to the lab WLAN network.

```
# Wi-Fi configuration
SSID = 'xxxx'
PASSWORD = 'xxxx'

# Connect to Wi-Fi
def connect_wifi():
    wlan = network.WLAN(network.STA_IF)
    wlan.active(True)
    wlan.connect(SSID, PASSWORD)

    while not wlan.isconnected():
        print('Connecting to WiFi...')
        time.sleep(1)

    print('Connected to WiFi')
    print(wlan.ifconfig())
```

FIGURE 10: Forming a network connection on the Pico

After connecting the Raspberry Pi 4 to both networks it should have IP addresses assigned from each network, allowing it to communicate with devices on both networks simultaneously.

Without wireless local area network (WLAN) functionality in a possible future iteration of the product, the utilization of Raspberry Pi devices equipped solely with Ethernet ports is important. Of course, other alternative microcontroller brands could also be reconsidered at this point. Relying on only Ethernet connectivity obsoletes the need for dual-homed configurations.

The absence of WLAN functionality eliminates potential security vulnerabilities associated with wireless communication protocols, strengthening the integrity and therefore safety of the network infrastructure.

3.4 Database design

As previously mentioned, the database is stored in the LibreNMS server, which takes care of the polling. The standard polling rate is once every five minutes. This can however be adjusted. For data storage and lab use monitoring reasons it is important the ceiling panel data is stored, not just shown on demand. Long-term data storage allows to identify trends and patterns over time, such as the frequency of panel misplacements or recurring issues in specific areas. This information can be relayed to laboratory users, emphasizing the weak points in data centers and extra-attention-needing zones for optimal energy efficiency.

Database data storage allows for the creation of visualizations that make it easier to understand and communicate data insights to users and highlight key areas requiring attention. Some lab ceilings panels do not sit straight, and other data centers are in more remote locations, so with the information of the frequency of panel issues, active 'hot zones' can be spotted, and actions can be taken to minimize the mistakes.

3.5 System implementation

Copper tape makes the most affordable solution and is the simplest to implement in the PoC phases, as stated in Chapter 2, Theoretics: Material and Solution Selection. The ceiling panels were taped according to the design displayed in the miniature model of FIGURE 3. Tape was also applied to pieces on top of the ceiling frame. Already a few days after the application, weaknesses could be spotted. FIGURE 7 shows the copper tape ripping from the conjoining point of two ceiling panels due to abrasion and the sharp ends of the panels.

The life-size model of FIGURE 3 was implemented with a Raspberry Pi Pico W in the place of the Raspberry Pi 4. A slight modification also had to be made to the copper tape pieces at the joining points of panels. As the panels were not perfectly flat, some failed to contact the tape, leading to discontinuity in the circuit. Due to this, multiple pieces of tape had to be layered to create higher towers for the panels to touch and ensure continuity.

4 WEB INTERFACE AND VIRTUAL MONITORING

There are many options for displaying the status of the ceiling panels. Overall, integration to a web interface allows to streamline virtual monitoring and enhance the management of various systems within a data center environment. While many projects typically use their own standalone websites for monitoring purposes, a pre-existing platform such as LibreNMS which is already being used at Nokia is a great option from the already numerous tools. Integrating this aspect of lab management via LibreNMS supports efficiency and centralization which are key points. Another software already in use is an in-house tool, ILM, for the displaying of an interactive lab map structure.

ILM is a comprehensive device monitoring system already in use for overseeing device temperature, locations, and overall laboratory status, and with the code available to customize it, there are possibilities for customizations such as ceiling panel status displaying. The LibreNMS database would most likely be where the microcontroller data would be sent, as it is easier and more logical to use a pre-existent database instead of creating a new one for each project. This database and all the information displayed on the GUI provides a centralized hub, where laboratory users can view real-time insights into the status and performance of the electronic systems and soon, the status of the ceiling panels.

The Pico sends MQTT updates to a topic every 10 seconds, which the Raspberry Pi 4 subscribes to. The process for this is shown in FIGURE 11. The Raspberry Pi 4 further analyses this and relays the information to ILM and LibreNMS (as explained more in Chapter 5, Monitoring). A script on the ILM server gets a file containing the status of the panels in csv-formal (comma-separated value) every 5 minutes. The csv-file is written by the Raspberry Pi 4 with a script shown in FIGURE 12, and updates every time the Raspberry Pi 4 receives a new MQTT post from the Pico. LibreNMS polls the Raspberry Pi 4 by SNMP at 5-minute intervals and gets the data that way.

```
# Main function to connect, publish, and detect current
def main():
    connect_wifi()
    client = init_mqtt()

    output_pin.value(1)

    while True:
        # Check if current is detected
        input_state = input_pin.value()
        if input_state:
            status = f"Current in circuit, green!"
        else:
            status = f"No current detected on input pin, red!"

        print(status)

        # Publish status via MQTT
        publish_status(client, status)

        # Delay before next iteration
        utime.sleep(10)

if __name__ == '__main__':
    main()
```

FIGURE 11: Pico's script for detecting and reporting status of the current

```
def log_status(roof_id, status):
    try:
        with open(LOG_FILE, 'w') as log_file:  # Open the log file in write mode
            log_file.write("roof_id,status\n")  # Write the header line
            log_file.write(f"{roof_id},{status}\n")  # Write the current status
        print("Log written:", f"{roof_id},{status}")
    except Exception as e:
        print(f"Failed to write log: {e}")
```

FIGURE 12: Writing roof status to file for ILM to analyze

If the panels are closed, the ILM visual interface shows green, as shown in FIGURE 13. If the panels are open, but for less than 30 minutes, it shows orange, with a state of ‘pending’. If the panels have been open for more than that, they turn red.

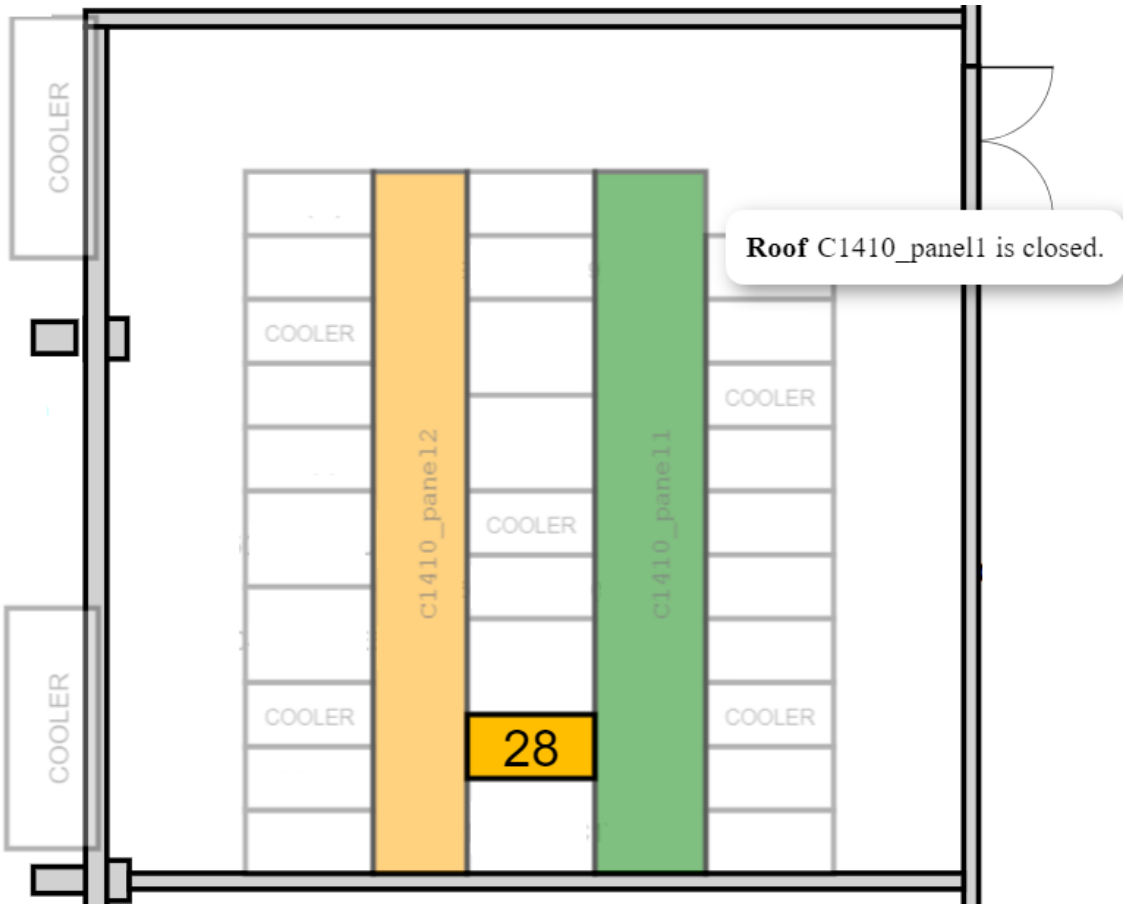

FIGURE 13. Laboratory monitoring visual implementation displaying ceiling status

Considering monitoring solutions, a widely known challenge is in the fragmentation. Bigger corporations struggle with a multitude of different applications, often with overlapping functions. This fragmentation not only complicates the monitoring process but also adds additional burdens on users, who must navigate multiple interfaces to perform their daily work.

As mentioned in Chapter 3, System Design and Implementation, LibreNMS polls the Raspberry Pi 4 at steady intervals. This data is then held in LibreNMS for data analysis and storage purposes.

5 MONITORING DATABASE AND LEDS

5.1 Color changing lights

As previously displayed in FIGURE 11, the Raspberry Pi Pico W publishes MQTT messages containing a color code, either ‘red’ or ‘green’. The messages are received by another Raspberry Pi, which in turn responds by changing color of LEDs (light emitting diodes) to indicate the roof status. The goal of the lights was to have LEDs at the doors of each laboratory, so that users could walk along corridors and easily see if something was wrong. Another position where the lights were supposed to be set up was at each data center aisle. Ideally, the LEDs would have been in representable-looking wooden boxes or something of the type. Nonetheless, only a barebones model of the LED structure was implemented. It can be seen in FIGURE 14. Two green lights turn on at the signal of ‘green’ and two red lights turn on at the signal of ‘red’.

With long enough cables, a Pico could be connected to two LED groups, for example for neighboring aisles. This rendition of the data center ceiling panel monitoring project failed to implement the room-centric LEDs due to material constraints.

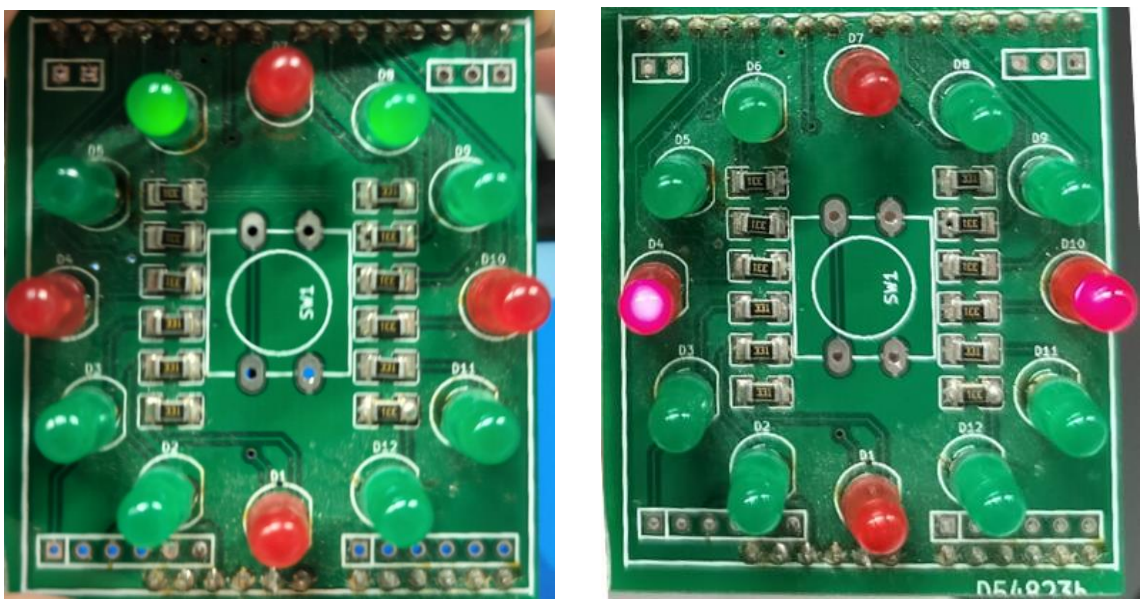

FIGURE 14: The corridor lights, not yet in their boxes, displaying the status of the ceiling panels with either green LEDs or red

The LED color management was conducted by the Raspberry Pi 4 due to simplicity in the PoC stage. FIGURE 15 shows how the GPIO pins are set up for the wires leading to the red pins and the green ones. There is also a wire going from the LED set to a grounding pin on the Raspberry. The two bottom lines on the figure show that the pins are set to GPIO.OUT. The Raspberry Pi 4 is a pin-model (there are both pin and port Raspberry Pis). With the lines of code, the pins were set to 3.3V. The Raspberry Pi 4 also has 3.3V output pins but those cannot be turned off.

```
import RPi.GPIO as GPIO
import time
import paho.mqtt.client as mqtt

# GPIO setup
RED_PIN = [17, 27]
GREEN_PIN = [23, 24]

GPIO.setmode(GPIO.BCM)
GPIO.setup(RED_PIN, GPIO.OUT)
GPIO.setup(GREEN_PIN, GPIO.OUT)
```

FIGURE 15: GPIO imports and pin setup on the Raspberry Pi 4

With the GPIO pins set up, the code progressed to change the LED colors according to messages on the topic the Pico was publishing to. FIGURE 16 shows how for each scenario (panels open or closed), the pins attached to both LED colors have to be edited to change to appropriate colors.

The lights turn red as soon as a panel is open, whereas the ILM panels first turn orange and after a long enough break, red.

```
# Time tracking
last_red_time = None
status = "pending"  # Default initial status

def on_connect(client, userdata, flags, rc):
    print("Connected with result code " + str(rc))
    client.subscribe(MQTT_TOPIC)

def on_message(client, userdata, msg):
    global last_red_time, status
    message = msg.payload.decode()
    print(f"Message received: {message}")
    current_time = time.time()

    if 'green' in message:
        for pin in RED_PIN:
            GPIO.output(pin, GPIO.LOW)
        for pin in GREEN_PIN:
            GPIO.output(pin, GPIO.HIGH)
        print("Trying to turn LED green")
        status = "closed"
        last_red_time = None  # Reset the red signal timer
    elif 'red' in message:
        for pin in GREEN_PIN:
            GPIO.output(pin, GPIO.LOW)
        for pin in RED_PIN:
            GPIO.output(pin, GPIO.HIGH)
        print("Trying to turn LED red")
        if last_red_time is None:
            last_red_time = current_time
            status = "pending"
        else:
            elapsed_time = current_time - last_red_time
            if elapsed_time > 20 * 60:  # 20 minutes in seconds
                status = "open"
            else:
                status = "pending"
    log_status("C1410_panel2", status)
```

FIGURE 16: Continuation of FIGURE 15, responding to the MQTT posts by the Pico by changing LED color

5.2 LibreNMS database and email alerts

To be able to poll the Raspberry Pi 4 from LibreNMS via SNMP and get necessary data, some extensions had to be downloaded onto the Raspberry. The extensions ‘rpigpiomonitor’ and ‘raspberry’ are visible in FIGURE 17. They are provided by LibreNMS, and the documentation and examples helped implement them (30).

```
extend rpigpiomonitor '/usr/bin/php /etc/snmp/rpigpiomonitor.php'
extend raspberry '/bin/sh /etc/snmp/raspberry.sh'
```

FIGURE 17: Extension libraries provided by LibreNMS for more advanced functionality

LibreNMS comes with a readiness to send alerts to predetermined groups – such as one with the contacts of whole teams or just to individuals. For the PoC, an alert transport was created to send an email to the appropriate personnel, visible in FIGURE 18.

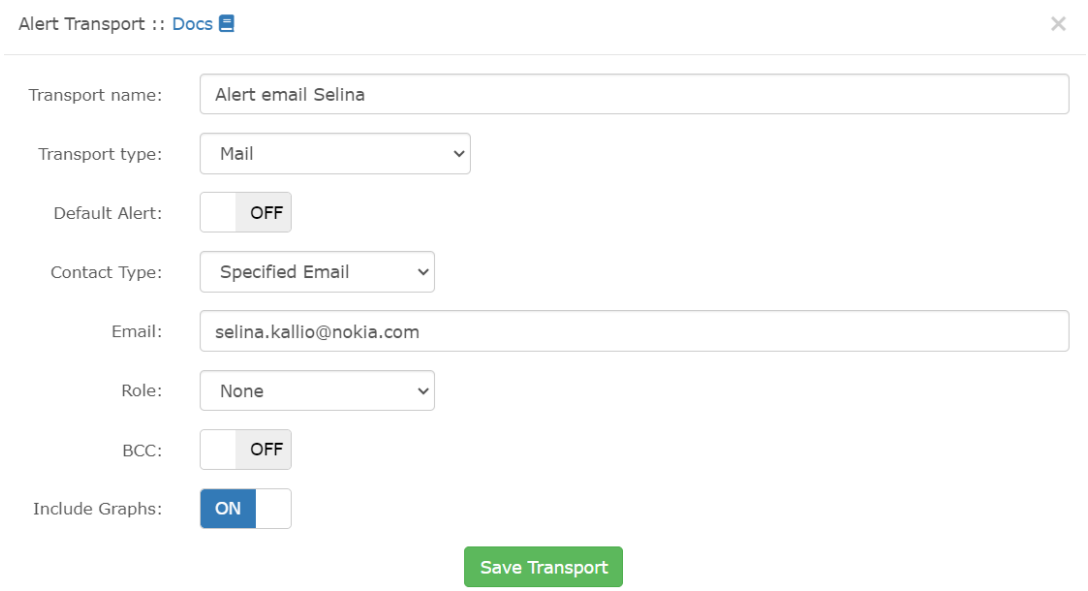

FIGURE 18: Creating a LibreNMS alert transport to alert the appropriate personnel via chosen method (email)

The previously mentioned rpigpiomonitor package had .ini and .php files that could be edited to meet the needs of each use case. FIGURE 19 shows how rpigpiomonitor.ini was configured. The configuration included the type of data/monitor (for example, temperature, humidity, voltage, or state), names of the values, a description, and a name in addition to where the rpigpiomonitor.php was to get the actual data from – a GPIO pin or a file. State was chosen as the data type since the panels were either open or closed (0 or 1). The file check_panel_status_from_file.sh checked the file going to ILM, and if it said open or pending, printed 1 (to indicate open panels) and if the file said closed, it printed 2.

```
[Raspberry Pi 4 LED monitoring]
type = state
description = Ceiling panels
states.Panel out of place.value = 1
states.Panel out of place.generic = 2
states.Panels closed.value = 0
states.Panels closed.generic = 0
external_gpio_reader = /etc/snmp/check_panel_status_from_file.sh
```

FIGURE 19: Configuration of the rpigpiomonitor.ini, setting up a sensor for the status of the panels

The data extracted from the rpigpiomonitor extension is visible alongside other data from the Raspberry Pi 4, as is visible in FIGURE 20. If the panels are open, the alternative status is seen as in FIGURE 21.

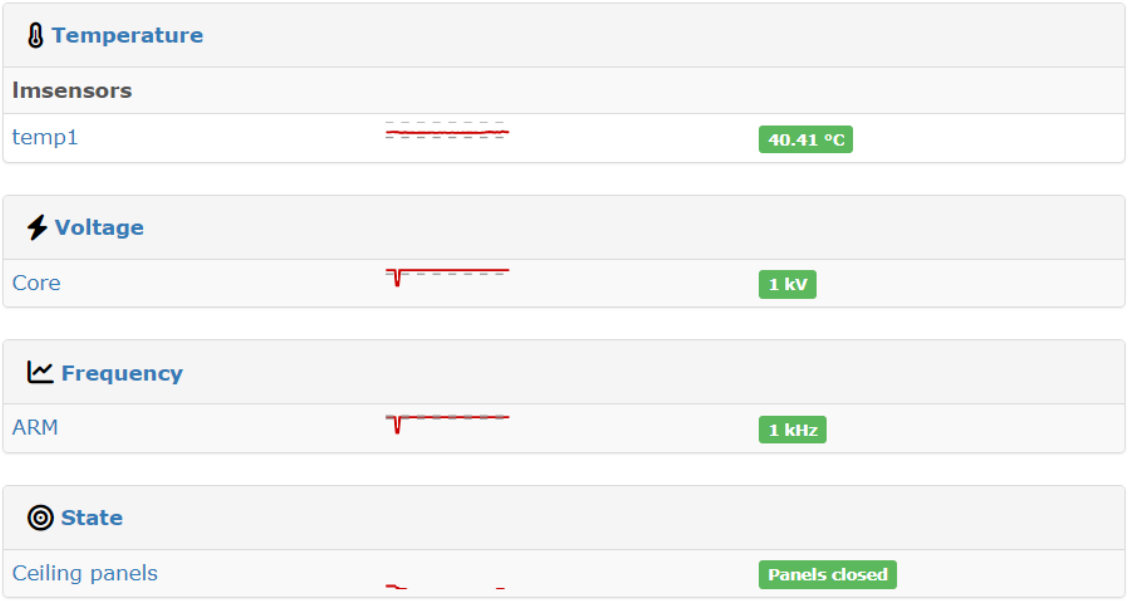

FIGURE 20: Raspberry Pi 4 statistics displayed in LibreNMS

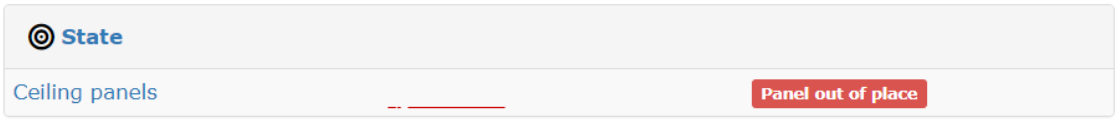

FIGURE 21: Alternate status of panels in LibreNMS

LibreNMS also provides a visual interface for tracking data over time. Values such as temperature, traffic and CPU use can be inspected in graphs and alarms can be set to notify users of situations. A graph of the status of the panels is visible in FIGURE 22. Alerts can be made from a device's page, or from the main 'Alerts' control. FIGURE 23 shows the popup window for creating an alert

if the Raspberry Pi 4 sensor data claims that the panels have been open for 60 minutes. After the initial alert email, emails are sent every three hours two more times unless the issue is fixed.

FIGURE 22: Graph with ceiling panel data

The alerts can be configured to send SMS messages for even more urgent situations. An alert rule can be created to poll the SQL database and in case of a specific event, use a preexistent alert transport rule.

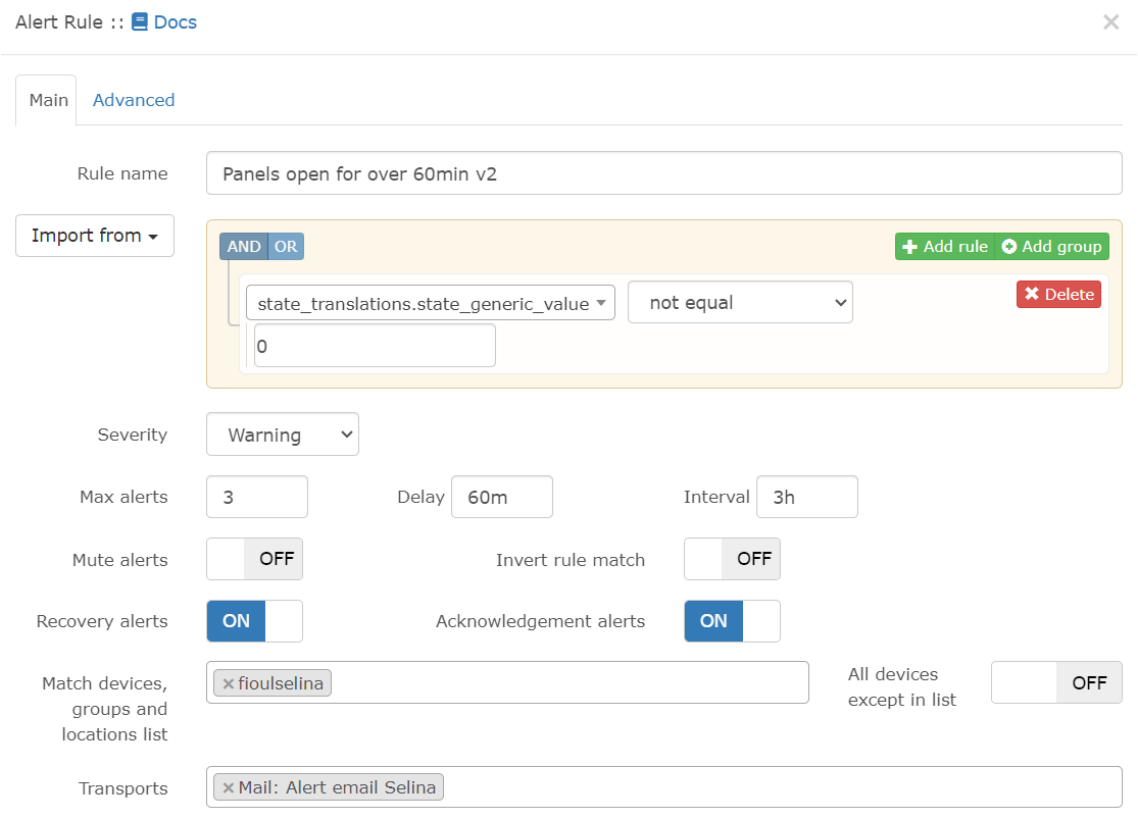

FIGURE 23: Creating the alert for when panel has been open for over sixty minutes

6 REFLECTION, RESULTS AND FUTURE

I think that energy efficiency is growing increasingly critical. We should be constantly developing towards a better future and not getting comfortable in the now. This thesis provided a good base for data center monitoring to build from: it showed that panel monitoring can be done and listed the alternatives to copper tape. If this goes to further development as I hope, most likely full-metal bars will be used to replace it as a more durable option. In any case, it would be best to implement the conductive elements/switches/whatever already in their production phase, as that would be most time and cost efficient. Also, a similar solution could be implemented onto doors, to ensure they are also being closed.

In future iterations it would best to choose ethernet-capable microcontrollers because of the extra safety they bring. They are also simpler to implement, as they do not require extra elements such as routers. If the data center aisles are long, the aisles can also be split into sections for easier identification of misplaced panels. I would like to see more experimentation and research done on the ESD side, since that was only very slightly covered in this thesis.

The LED light implementation was cut a bit short by restrictions in time, and I wish I could have made them look prettier by laser cutting wooden boxes for them or using more sophisticated-looking lights. They, however, worked and, in the end, a PoC does not have to be pretty – or even working, as long as something is learned. However, I am happy that it did work.

To optimize further use, it would also be good to implement some sort of remote access on the Raspberry Pi Pico W. I did some research on WebREPL and similar solutions, but finally decided not to implement it in the PoC phase. With OTA (Over The Air) updates or something similar, the devices could easily be updated in the future.

With the new Nokia campus being built and planned, I still have strong hopes that some sort of panel monitoring will be implemented. I hope the research I have done and the experience I have on the matter will be useful and help inspire others to identify problems and create solutions for them.

WORKS CITED

1. Angara, Jamie 2021. Data Center Aisle Containment. AKCP. Search Date 11.2.2024. https://www.akcp.com/articles/data-center-aisle-containment/.
2. Move to a Hot Aisle/Cold Aisle Layout. Energy Star. Search Date 11.2.2024. https://www.energystar.gov/products/move_hot_aislecold_aisle_layout.
3. Kirvan, Paul 2022. Hot/Cold Aisle. TechTarget. Search date 12.2.2024. https://www.techtarget.com/searchdatacenter/definition/hot-cold-aisle.
4. Electric Circuits. Victoria State Government. Search Date 14.2.2024. https://www.education.vic.gov.au/school/teachers/teachingresources/discipline/science/continuum/Pages/electriccircuit.asp.
5. Schappi, Marcus 2024. Digital Inputs with Raspberry Pi. Little Bird Electronics. Search Date 14.2.2024. https://littlebirdelectronics.com.au/guides/92/digital-inputs-with-raspberry-pi.
6. [Nickname] SLR 2022. Understanding the Microcontroller GPIO – GPIO Working Explained. EmbeTronicX. Search Date 15.2.2024. https://embetronicx.com/tutorials/tech_devices/understanding-the-microcontroller-gpio-gpio-working-explained/nderstanding-the-microcontroller-gpio-gpio-working-explained/.
7. Karonji, Wahome 2023. Raspberry Pi Pico vs. Arduino: Which Microcontroller Should You Use?. Make Use Of. Search Date 28.2.2024. https://www.makeuseof.com/raspberry-pi-pico-vs-arduino-which-microcontroller/.
8. Raspberry Pi Microcontrollers Unveiled: From Basics to Beyond. Indian Institute of Embedded Systems. Search Date 28.2.2024. https://iies.in/blog/raspberry-pi-microcontrollers-basics-to-beyond/.
9. CircuitMonster. Exploring ESP32 Alternatives: A Comprehensive Guide for Newbies. Search Date 29.2.2024. https://circuitmonster.co.in/exploring-esp32-alternatives-a-comprehensive-guide-for-newbies/.
10. [Nickname] drmpf. Arduino NANO 33 Made Easy BLE, Sense and IoT : 11 Steps. Search Date 29.2.2024. https://www.instructables.com/Arduino-NANO-33-Made-Easy-BLE-Sense-and-IoT/.
11. Nordic Semiconductor Infocenter. Nordic Semiconductor. Search Date 30.2.2024. https://infocenter.nordicsemi.com/index.jsp?topic=%2Fstruct_nrf52%2Fstruct%2Fnrf52.html

12. Particle Argon. Particle. Search date 30.2.2024. https://docs.particle.io/argon/.
13. Espressif Systems. Espressif. Search date 30.2.2024. https://www.espressif.com/.
14. What Is a Network Protocol, and How Does It Work?. CompTIA. Search date 7.4.2024. https://www.comptia.org/content/guides/what-is-a-network-protocol.
15. Building a Specification on top of MQTT to Meet Your Industry Requirements. HiveMQ. Search Date 8.4.2024. https://www.hivemq.com/resources/defining-mqtt-specification-to-meet-iot-iiot-industry-implementation/.
16. What is MQTT? A practical introductiomn. Inray. Search Date 8.4.2024. https://www.opc-router.com/what-is-mqtt/
17. Configuring LibreNMS Network Management System. Network Startup Resource Center. Search Date 12.4.2024. nsrc.org/workshops/2016/tznog4-nmm/netmgmt/LibreNMS/netmgmt/librenms-lab.md.html.
18. Jiaojiao, Li 2021. What Is SNMP? Huawei. Search date 12.4.2024. https://info.support.huawei.com/info-finder/encyclopedia/en/SNMP.html.
19. Raspberry PI and LibreNMS: Powerful Monitor for Home Network. peppe8o. Search Date 13.4.2024. https://peppe8o.com/librenms-raspberry-pi/.
20. Preston, Julia 2017. Transversality: Breaking Down Work Silos for a Better Team Experience. Medium. Search Date 14.4.2024. https://projekt202.medium.com/transversality-breaking-down-work-silos-for-a-better-team-experience-b952517c15d5
21. Bhatia, Ravi 2020. The “Silo” Effect in Large Project-Based Organizations. PLG. https://www.getplg.com/post/the-silo-effect-in-large-project-based-organizations
22. Conductive Tapes – A Selection Guide. Robert McKeown Company Inc. Search date 14.3.2024. https://robertmckeown.com/blog/conductive-tapes-guide/.
23. Electric Paint. Bare Conductive. Search date 14.3.2024. https://www.bareconductive.com/collections/electric-paint.
24. Mu-copper tape. Holland Shielding Systems BV. Search date 14.3.2024. https://holland-shielding.com/en/mu-copper-tape
25. Coreray 2023. Choosing the Right Switch: Optical or Electrical? A Comprehensive Guide. Search Date 14.3.2024. https://corerayoptical.weebly.com/blogs/choosing-the-right-switch-optical-or-electrical-a-comprehensive-guide.
26. Mill-Max 2024. Frequently asked questions about spring-loaded pogo pins and connectors. Search date 17.3.2024. https://www.mill-max.com/engineering-notebooks/spring-loaded-pogo-pins-connectors/faqs

27. Mill-Max 2024. Introduction to spring loaded pogo pins and connectors. Search date 17.3.2024 https://www.mill-max.com/engineering-notebooks/spring-loaded-pogo-pins-connectors.
28. MaxBotix 2023. How Ultrasonic Sensors Work. Search date 17.4.2024. https://max-botix.com/blogs/blog/how-ultrasonic-sensors-work.
29. Beaty, William 1999. "Static Electricity" means "High Voltage" – Measuring your body-voltage. http://amasci.com/emotor/voltmeas.html.
30. LibreNMS. Applications. Search date 9.5.2024. https://docs.librenms.org/Extensions/Applications.