Carolina Andreia da Silva Martins de Pinho

Comunicações sem fios para sistemas de deteção de incêndio de próxima geração

Universidade de Aveiro
2022

Carolina Andreia da Silva Martins de Pinho

Comunicações sem fios para sistemas de deteção de incêndio de próxima geração

Dissertação apresentada à Universidade de Aveiro para cumprimento dos requisitos necessários à obtenção do grau de Mestre em Engenharia Eletrónica e Telecomunicações, realizada sob a orientação científica do Doutor Paulo Bacelar Reis Pedreiras, Professor auxiliar do da Universidade de Aveiro, e do Doutor Luis Emanuel Moutinho da Silva (co-orientador), Professor Adjunto Convidado da Escola Superior de Gestão e Tecnologia de Águeda
.

o júri / the jury

presidente / president

Professor Doutor Mário José Neves de Lima
Professor Auxiliar, Universidade de Aveiro

vogais / examiners committee

Prof. Doutor Luis Miguel Pinho de Almeida
Professor Associado, Universidade do Porto - Faculdade de Engenharia

Prof. Doutor Paulo Bacelar Reis Pedreiras
Professor Auxiliar, Universidade de Aveiro

agradecimentos /
acknowledgements

No final deste ciclo resta-me agradecer a todos os que direta ou indiretamente fizeram parte do meu percurso académico. Foram 5 anos cheios de emoção e aprendizagem que sem dúvida vão deixar saudades.
Começar por agradecer ao meu professor e orientador Paulo Pedreiras pelo suporte incomparável e orientação neste trabalho, pelas suas palavras sábias e disponibilidade.
Ao meu co-orientador Luís Moutinho, um grande obrigada pelos momentos de companheirismo e orientação constante. Agradeço-lhe ainda a oportunidade de desenvolver este trabalho na BOSCH, não esquecendo todas as boleias e almoços que me permitiram estabelecer laços com a equipa incrível de FIRE da BOSCH Ovar, aos quais devo também um agradecimento especial pela simpatia e amizade.
A todos os meus colegas de curso e amigos por estarem presentes nos bons e maus momentos e por todas as aprendizagens.
A todos os amigos do associativismo que tanto me ajudaram a crescer, foi sem dúvida das melhores coisas do percurso.
Aos meus amigos "da terrinha" pelas gargalhadas e chatices, sem vocês não teria sido tão divertido. Um obrigado especial à Beatriz pela constante presença e amizade que já celebra mais de uma década. Que venham muitas mais.
À minha mãe e ao meu irmão pelos conselhos, apoio e compreensão.
Por fim mas não menos importante, ao meu namorado pelo amor incondicional. Sem ti nada disto seria possível. Obrigada.

Palavras Chave

Internet das coisas, WSN, sistemas tempo real, Comunicações sem fios, Alarme de incêndio, TSCH, IEEE 802.15.4e, Contiki, 6TiSCH, Indústria 4.0.

Resumo

A quarta revolução industrial trouxe consigo um dos temas mais investigados atualmente, a Internet das Coisas (IoT). O objetivo geral é interligar o maior número possível de dispositivos no máximo de aplicações. A indústria de alarmes de incêndio está agora a perceber a necessidade de atualizar os seus protocolos de comunicação para acompanhar este novo conceito industrial. A maioria dos sistemas de alarme de incêndio comercializados atualmente são cablados, com protocolos de comunicação antigos. A evolução das tecnologias de comunicação sem fios, trouxe novas funcionalidades que permitem o suporte para aplicações com requisitos críticos de tempo real e de consumo energético. Desta forma, é cada vez mais vantajoso procurar atualizar sistemas deste tipo para tirar proveito da flexibilidade que as comunicações sem fios trazem, uma vez que estas já permitem satisfazer os requisitos desta classe de aplicações. Este trabalho tem como objetivo explorar as tecnologias de comunicações sem fios existentes e avaliar a correspondente adequação à implementação de sistemas de alarme de incêndio. Após a escolha da tecnologia IEEE802.15.4e com TSCH MAC *behavior*, o foco passou para desenvolver uma prova de conceito baseada em CC2650 Launchpads. Para tal, foi necessário estudar e adaptar o sistema operativo Contiki NG para criar uma *micro-network* para testes de laboratório de forma a avaliar a implementação.

Keywords

Internet of Things, Real-Time Systems, WSN, Wireless Communication Networks, Fire Alarms, TSCH,IEEE 802.15.4e, Contiki, 6TiSCH, Industry 4.0.

Abstract

The fourth industrial revolution brought with it one of the most researched topics today, the Internet of Things (IoT). Its overall goal is to interconnect as many devices as possible in as many applications as possible. The fire alarm industry is now realizing the need to upgrade their communication protocols to keep up with this new industrial concept. Most fire alarm systems marketed today are wired and use legacy communication protocols. The evolution of wireless communication technologies has brought new features that allow the support for applications with critical real-time and power consumption requirements. Thus, it is becoming increasingly advantageous to seek to upgrade systems of this type to take advantage of the flexibility that wireless communications bring, since it is already possible to guarantee the requirements posed by fire alarm systems. This work aims to explore existing wireless communications and evaluate which ones can be used to support fire alarm systems. After choosing the IEEE802.15.4e technology with TSCH MAC behavior, the focus shifted to developing a proof of concept based on CC2650 Launchpad. To do this, it was necessary to study and adapt the Contiki NG operating system to create a micro-network for laboratory testing in order to verify the correctness and feasibility of the implementation.

Contents

List of Figures

List of Tables

Glossary

6LoWPAN	IPv6 over Low Power Wireless Personal Area Network
6Top	6TiSCH Operation Sublayer
API	Application Programming Interface
AES	Advanced Encryption Standard
ASN	Absolute Slot Number
ACK	Acknowledgement
AMCA	Aynchronous Multi-Channel Adaptation
ARM	Advanced RISC Machine
BLE	Bluetooth Low Energy
BTN	Button
CAP	Contention Access Period
CSMA/CA	Carrier-Sense Multiple-Access with Collision Avoidance
CCA	Clear Channel Assesment
CSS	Chirp Spread Spectrum
CO2	Carbon Dioxide
CoAP	Constrained Application Protocol
DAG	Directed Acyclic Graph
DODAG	Destination-Oriented DAG
DSME	Deterministic and Synchronous Multi-Channel Extension
EB	Enhanced Beacon
FFD	Full Function Device
GTS	Guaranteed Time Slots
GACK	Group Acknowledgement
HART	Highway Addressable Remote
IoT	Internet of Things
IP	Internet Protocol
IEEE	Institute of Electrical and Electronics Engineers
IPv4	Internet Protocol version 4
IPv6	Internet Protocol version 6
ISA	International Society of Automation
LLN	Low-Power and Lossy Networks
LLDN	Low Latency Deterministic Networks
LPWAN	Low Power Wide Area Networks
LR-WPAN	Low Rate Wireless Personal Area Networks
MQTT	Message Queuing Telemetry Transport
MAC	Media Access Control
MCU	MicroController Unit
MU	Multiuser Transmission
NB-IoT	Narrow Band IoT
OS	Operating System
OFDMA	Orthogonal Frequency Division Multiplexing Access
PC	Personal Computer
PHY	Physical
PAN	Personal Area Network
QoS	Quality-of-Service
RAM	Random Access Memory
RPL	Routing Protocol for Low-Power and Lossy Networks
RFD	Reduced-Function Device
RFID	Radio Frequency Identification
RX	Reception
ScF	Scheduling Functions
SDK	Software Development Kit
SF	Spreading Factor
SR	Spatial Reuse
TSCH	Time Slotted Channel Hopping
TCP	Transmission Control Protoco
TI	Texas Instrument
TX	Transmission
TWT	Target Wake Time
UDP	User Datagram Transport
USB	Universal Serial Bus
MCU	microcontroller unit
NG	Next Generation
Wi-Fi	Wireless Fidelity
WSN	Wireless Sensor Network
WLAN	Wireless Local Area Network
WPAN	Wireless Personal Area Network

CHAPTER 1

Introduction

This dissertation's topic and scope are introduced globally in this first Chapter, which is divided into four sections. The overview and motivations for this dissertation are stated in Section 1.1. The goals of this work are listed in Section 1.2, and the document's structure is described in Section 1.3.

1.1 Overview and motivation

During the last few years, the evolution of technology and the various changes that have occurred within it have led to new ways of collecting, sharing, transmitting, and representing information. All of these advancements have resulted in an improvement in human life quality by introducing more convenience, guarantees, efficiency, and security.

The present era is characterized by a technical-industrial revolution, commonly called Industry 4.0 [1], in which there is a common objective of improving process efficiency and productivity. Another famouse concept of this era is the Internet of Things (IoT) [2] that promises to connect billions of smart devices over a massive and collaborative network infrastructure. Adopting IoT is a springboard toward the development of smart homes, smart buildings, and, eventually, smart cities. The growing number of smart products used on a daily basis, such as phones, assistants, automobiles, and even smart lights, demonstrates the direction technology is taking. However, this concept brings all kinds of requirements in the field of microelectronics, such as memories, batteries, energy scavenging, and hardware designs, hence the need for large-scale, low-power networks [3]. This is one of the key facts that motivated the exploration of the already known Wireless Sensor Networks (WSN) for this type of applications [4]. Wireless Sensor Networks support a wide range of real-time applications, including industrial automation, environmental control, disaster management, smart buildings, personal health care, and even security applications. This type of systems normally features a considerable number of low-power and low-cost devices that are integrated with limited computation capacity and radio communication capabilities for sensing purposes [4]. Therefore, it can be highly suitable for communication between fire alarm detectors.

Until now, there are few companies with a wireless fire alarm solution on the market. Most of the available products for fire alarms are hardwired. Hardwired solutions have been linked to fire alarm systems since they became automated, effectively responding to the time-based requirements imposed by this type of system. However, wireless communications allow a large number of devices to be connected to the Internet in a simple, flexible and cost-effective manner. This type of solution can cover a wide scope of applications based on range: wide area networks, known as Low Power Wide Area Network (LPWAN), in both private (LoRa, LoRaWAN, IEEE 802.11ah) and public (SigFox, NB-IoT, 5G) infrastructure versions; or personal area networks, known as Low Rate Wireless Personal Area Network (LR-WPAN), such as IEEE 802.15.4 and IEEE 802.15.1 standards [3].

1.2 Objectives

The main goal of this project is to explore the current wireless communication technologies and see which ones could be used to implement a wireless fire alarm system. The difficulty here is that this type of system has strict time and energy requirements that not all wireless technologies can respond to. For this reason, there are main requirements that can be seen as objectives for the project:

- Research for wireless communication that are low power and can respond to real time requirements and scalability
- Design a system architecture that enables the extension of current fire detection systems with a view to integrating devices with the wireless communication technology identified as most promising;
- Develop prototype devices (e.g., hand-held alarm detectors and buttons) capable of communicating with the identified wireless communication technology;

1.3 Structure

This report starts with an introduction stating the main motivation that led to the necessity of this project and a presentation of its objectives.

Then, in Chapter 2, it is presented an overview of what alarm systems are, how they operate and why are they essential to the quality of life of everyone. Then, a market survey presents some current solutions that support wireless communication in their fire alarm systems. In particular, it is further examined one popular fire alarm system from Siemens, in order to provide a real-world example of what is the intended architecture.

In Chapter 3, an introduction is given to the types of wired fire alarm systems currently implemented, as well as an overview of which are the most commonly used communication technologies in this field. After presenting the main disadvantages of these current systems, the types of wireless technologies are presented together with a survey of those that could be suitable for the implementation of a fire alarm system with real-time and low-power requirements. In the end, a reflection is conducted to choose which of the technologies

presented would be the best to proceed with the project implementation to further explore its potential.

In Chapter 4, an overview of the chosen technology addresses its working principles and main features that can respond to the requisites imposed. IEEE 802.15.4e was the technology chosen and it presents many (Media Access Control)MAC behaviors that are also explored and explained in this Chapter in order to choose which of them is better for the implementation.

Having the technologies settled, a survey of the possible development platforms is made in Chapter 5. After all the considerations, the Contiki-NG operating system over CC2650 from Texas is the selected framework. Then, a presentation of the characteristics and advantages of Contiki is detailed, presenting their unique contribution to Wireless Sensor Networks.

In Chapter 6, a characterization of the implementation is presented from an hardware and software perspective. Here are shown the procedures, developed code, and difficulties faced in order to complete the project development in a positive way.

Chapter 7 addresses system tests and verifications. Here, a description of the procedure of how the tests were made and the respective results are presented, as well as a critical analysis of the system with parameters like latency, interarrival time, packet loss rate, and coverage.

Finally, in Chapter 8, the results of this project are summarized, and possible future work is discussed.

CHAPTER 2

Alarm Systems: An overview

Fire is one of the greatest achievements of mankind, having been an essential tool since its discovery. However, it can cause severe devastation when left unchecked. The history of how people dealt with firefighting is reviewed in this section, which is also the evolutionary path of the fire alarm systems known today. After this first introduction, the general architecture of fire alarm systems will be discussed in section 2.2. Then, section 2.3 addresses some regulations that bind all fire alarm system solutions, and the requirements that wireless communications have to meet. Finally, section 2.4 gives an overview of some companies that have already started to implement wireless fire alarm systems.

2.1 History and evolution of fire response

The evolution of the way in which a fire situation was dealt with is represented in Figure 2.1. For thousands of years, the detection and later fighting of fires were made as they erupted because there was no other way. Thus, in general, fires could only be countered once a

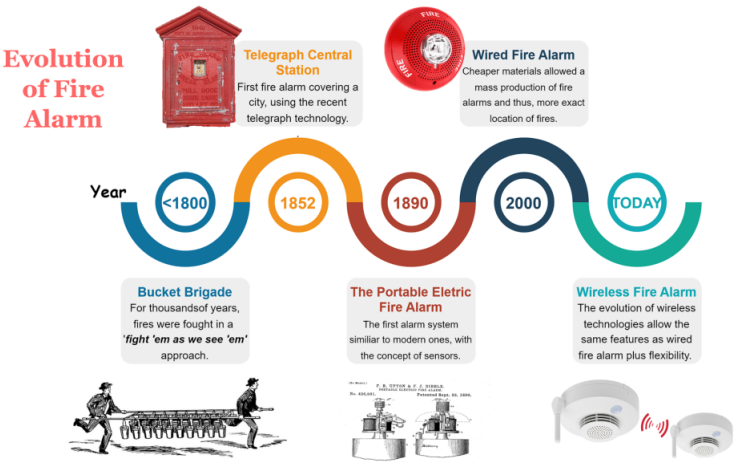

Figure 2.1: Evolution of fire Alarms

building was already highly consumed due to the slow reaction time. The fact that urban areas were more and more overcrowded and full of wooden structures has only compounded the constant calamity situations created by fires. In 1800, central bell towers were constructed in order to alert as many people as possible that there was a fire nearby. There were even ringing patterns to signal where in the city it was happening. However, the lack of an exact location kept firefighting difficult and still slow [5]. With the appearance of technology such as long-distance communications, in 1852, the first fire alarm covering an entire city was spearheaded with alarm boxes and telegraphic keys. Thus, the concept of the "central tower bell" was transformed into a "central station" that, by means of a telegraph, received notification of a nearby box and allowed the communication of this closer location to the fire response team [5].

The first automatic electric fire alarm appeared in 1890 named “The Portable Electric Fire Alarm” which checked for heat due to smoke or fire with a “thermostatic coil” [6]. During the early 20th century, other detection devices became available, including the smoke detector and carbon monoxide detector, which are now essential components of modern fire alarm systems.

Nowadays, fire alarm systems are more complex and designed to alert the authorities right away when there is a fire by incorporating sensors and the latest technologies. This way the time of intervention in the fire can be reduced in order to have better control of its propagation.

The evolution of the fire response does not depend only on improving the response part when the fire has already occurred, but also on its prevention, which over the years has adopted new measures. It is obvious that good prevention is the fundamental key to avoid fires and their propagation and it is multi-faceted. It goes from the core of building designs, including passive and active fire prevention systems such as sprinklers and smoke alarms, means of escape etc., to the proper maintenance of the building or even strict regulation on social issues related to fire risks (alcohol abuse, drugs, type of furniture,etc.) [5].

Figure 2.2 represents the amount of dwelling fires and deaths related to fire events as firefighting solutions are reviewed and became more rigid and complex in England between 1981 and 2016. This graphic shows that the effort put into fire alarm security and prevention has compensated over the years. However, in this interval of time not only were new prevention measures implemented but also the fire detection systems became more and more complex and reliable, which certainly influenced the reduction in the number of fatalities due to fires. The narrower line across the graph has a more accentuated reduction than the bars regarding the number of fires, i.e., as much as the regulations and prevention help to reduce the number of fires, the safety of individuals depends largely on the rapid detection to increase the possible time of escape and subsequent alert of the competent authorities. As alarm systems evolve, their advantages are reflected in the safety not only of individuals but also of material goods which is reflected all over the world.

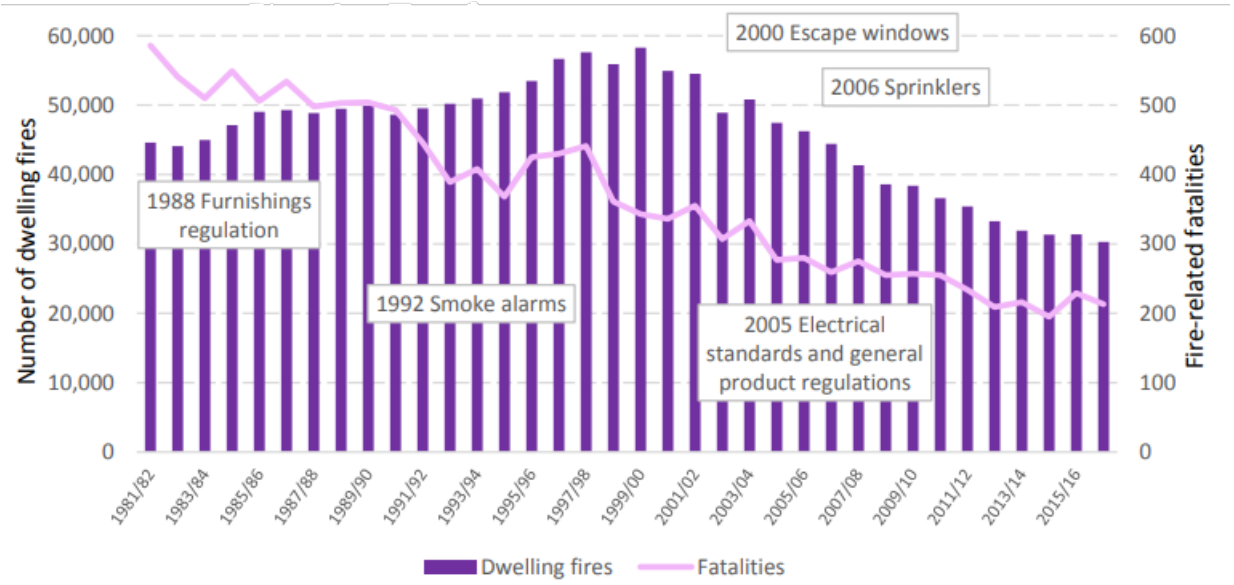

Figure 2.2: Dwelling fires and fire-related fatalities, shown against regulations aimed to increase fire safety, England; 1981-82 to 2016-17 [7]

2.2 General Fire alarm systems architecture

As will be seen below, fire alarm systems today follow strict requirements in order to be approved by the competent authorities and to be subsequently marketed and used on an European and worldwide level. Therefore, regardless of their manufacturer, fire alarm systems follow a similar structure for the same type of application. For example, to cover large scale buildings the alarm systems can be represented in a black box like in Figure 2.3 and are usually composed by: fire detectors that can have several sensors incorporated (smoke, carbon dioxide, heat, etc.), manual call points or emergency buttons, passive safety devices (sounders and strobes), control and indicating equipment (CIE) to manage the system and that serves as a bridge to communicate the system status (alarm, autonomy, errors...) to the respective competent entities. The distribution of these devices throughout the buildings depends on the specifications of each system, but the ideal is that there is a control panel in the center managing all the information of the remaining system devices spread throughout the respective space.

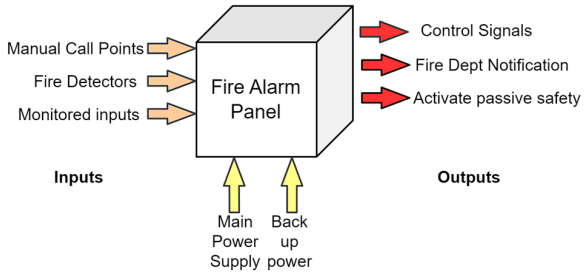

Figure 2.3: Black box diagram of a basic fire alarm system

2.3 Regulatory requirements

Fire safety measures are extremely important when it comes to reducing the consequences that a fire has on people's lives, material goods, and properties. As presented before, when efficient systems had not yet been developed and technology could not meet the demands of safety in case of fire, what managed to control the impact of fires was the introduction of regulations. Thus, regulation has always followed the security measures of fire alarms.

Nowadays, the production of fire alarm systems has to follow specific regulations in order to be approved on an European level and be sealed with the CE mark. CE marking guarantees that a product meets the necessary safety, health, and environmental protection requirements set by the EU to be sold and have free circulation in the European market.

In the 70's, the CEN (from French, Comité Européen de Normalisation), started the European standardization for fire alarms, the EN 54 series [8]. This regulation is a mandatory standard with strict laboratory tests and requirements that have to be followed for every component of fire detection and fire alarm systems. Nowadays, EN 54 gives the CE marking, being widely recognized in other countries outside the European Union, such as Brazil and American and Asian Countries. Since its start, this standard has been constantly updated, following the technical evolution of fire alarm systems. Currently, EN 54 series covers some of the following topics [9]:

- Control panels and power supply
- Optical and acoustic alarm devices
- Manual call points and detectors (smoke detectors, heat detectors, fire gas detectors)
- Compatibility of the system components of fire detection and fire alarm system
- Guidelines for the application of fire alarm systems
- Wireless fire alarms

Regarding the part of wireless systems, there are some important general requirements to be taken from the standard[10]:

- *"The system shall have the capacity to allow the installation of up to 240 detectors, warning devices, call points and modules on each communication loop."*
- *"The system shall consist of analog addressable fire detection, wired or wireless smoke and heat detectors (EN54 part 5 and 7), manual call points(EN54 part 11), sounders, and visual alarm indicators to communicate with the control and indicating equipment (CIE)."*
- *"The power source shall give the device an operational life of at least three years in normal conditions. "*
- *"All wireless devices shall have two independent battery supplies. A primary cell to power the device under normal conditions and a secondary cell to maintain the operation of the device should the primary fail."*
- *"The control and indicating equipment (CIE) shall be modular in design to allow for future expansion with a central processing unit capable of receiving and analyzing signals from both wired and wireless fire devices."*

- *"Alarm Transmission Time: the maximum time allowed between detection of fire to signalization at the control panel is 10 s."* [11]
- *"Monitoring: missing or malfunctioning devices have to be reported to the user at the control panel within at most 300 s after the error occurs."* [11]

2.4 Market Survey

Most modern fire alarm systems still use wired communication buses, as is the case of the current BOSCH system, shown in Figure 2.4. This and other similar wired communication systems will be discussed in section 3.1, after the state of art review, for a better understanding on the underlying protocols. There are several disadvantages related on having a wired fire

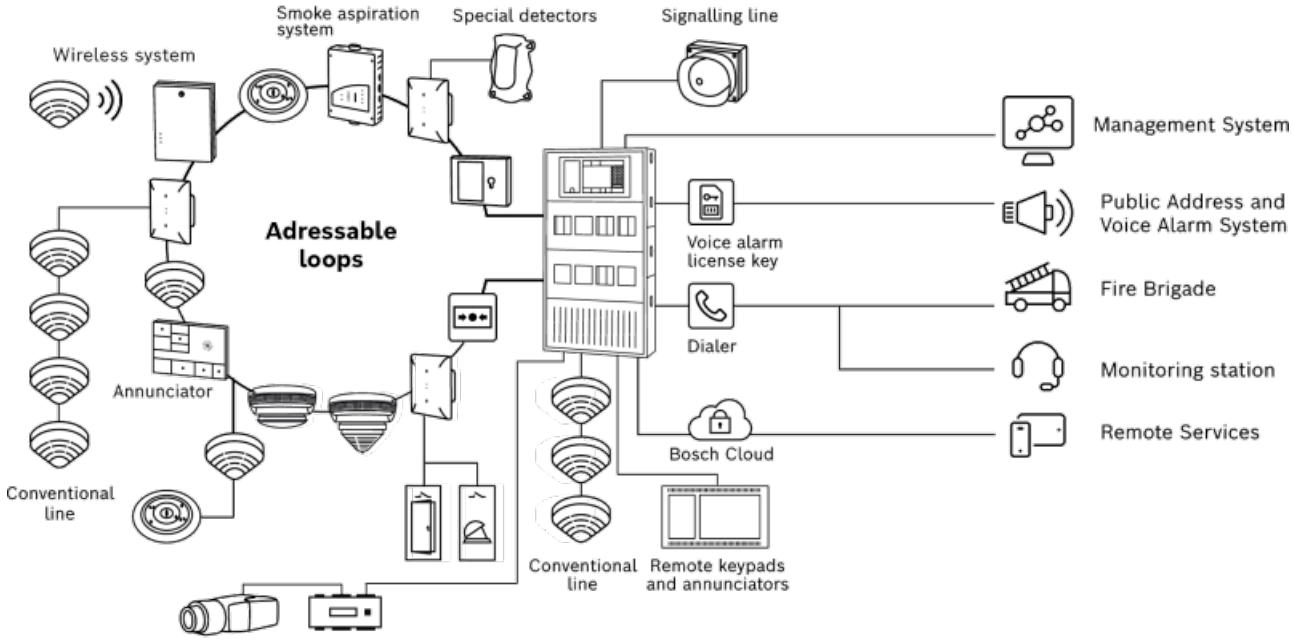

Figure 2.4: BOSCH fire alarm system overview [12]

alarm implementation, which is the most used in the area, namely the fact that it makes installation and maintenance difficult and limits the users' freedom for possible retrofits. Thus, it is becoming more and more important for developers to focus on wireless solutions in order to reduce costs and add flexibility while maintaining the benefits of reliability and robustness of wired systems.

The evolution of wireless technologies has opened a new horizon in the fire alarm systems field. The possibility of having fire alarm systems with the same advantages and response times as the wired ones, but with all the advantages of wireless solutions, has raised considerable interest in companies in the area that are researching more and more solutions of this type. One of the most significant advantages of wireless technologies is that devices can be positioned and repositioned quickly and freely. It simplifies planning, enables cost-effective installation, and lends itself to a high degree of flexibility when it comes to changing the use of the space

or the structure of the building. The pioneering companies in this area have started selling these solutions without any competition and even to their competitors. Some of the recent wireless solutions for fire alarm systems are presented below:

- Wi-Fyre [13]: a development from Eurofyre that allows cohesive integration of wired and wireless detection technologies in the proper ratios.
- Wi-Fyre Xenos [14]: Full wireless solution
- Swing from Siemens [11]: Full wireless solution

One of the most popular wireless alarm systems is SWING from Siemens, and section 2.4.1 focuses on understanding its features and technologies.

2.4.1 SWING system

Figure 2.5: SWING by Siemens [11]

SWING system (Figure 2.5) was developed by Siemens to improve existing wireless fire detection systems based on the following general requirements:

- Robustness: The system must overcome as many obstacles as possible ingeniously.
- Reliability: All incidents must be reported in real-time.
- Low-Power: It should be energy efficient since the detection devices are battery-powered.

SWING is intended for indoor applications like companies' buildings, houses, hospitals, etc. Figure 2.6 represents the SWING system in a building where a central gateway can communicate with all the network devices, which can be detectors or call points.

2.4.2 Concepts behind SWING

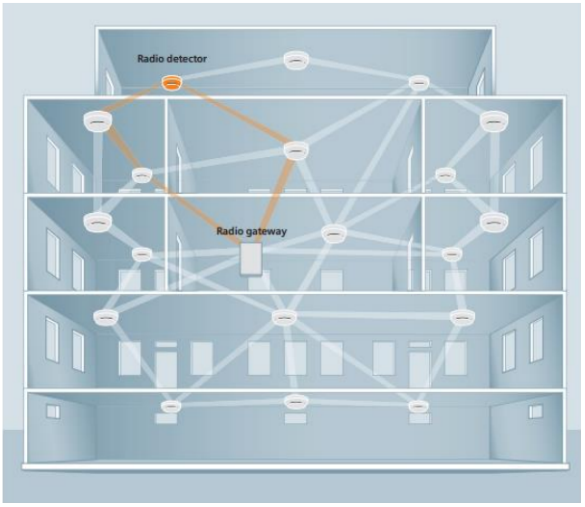

Figure 2.6: Network structure of SWING in a building [11]

SWING follows every requirement imposed by EN54 for wireless fire alarm systems.

The Siemens company uses a the multi-hop mesh network topology and the WiseMAC protocol [15] in their SWING product. Here all devices are battery-powered except the gateway [11]. For fire detection, the SWING system uses a heat detector and ASAtechnology (Advanced Signal Analysis) dual optical technology to reduce power consumption, reducing up to 10 percent of consumption compared to a wired system [7].

2.4.2.1 ASAtechnology

ASAtechnology is a software-based solution from Siemens that compares signals from sensors placed at strategic positions in the device with sophisticated algorithms to distinguish harmless smoke signals from fire signals. ASAtechnology operates by converting the signals coming from the detectors into mathematical components that are then compared in real time to programmed values using intelligent algorithms. The parameter sets are also dynamically adapted to the actual situation. Because of its efficiency and reliability, this technology can be used in critical locations like hospitals, where every second counts to get everyone out of the large building.

2.4.2.2 Additional features

SWING can add new devices (for example, a new smoke detector) without using an extra device like a computer to change the gateway settings. This is possible by switching the gateway to a "receptive" mode which will speed up the network maintenance process by allowing a new device to enter seamlessly, creating the proper links to the other nodes. This makes SWING a modular system. In order to cope with the presence of other users in the same frequency band, SWING can use up to 47 channels in 2 frequency bands. SWING can support a maximum of 30 nodes in a network that can cover up to 60 meters and span up to 5 floors.[16] In SWING 5s of the 10s required for alarm notifications, are intended for the wired transmission between the gateway and the fire control panel. This product also features a simple installation.

CHAPTER 3

Fire Alarm Communications: State of art

When it comes to fire alarm systems, most buildings and companies still use hardwired solutions because they are efficient and reliable. However, these solutions have some disadvantages, such as the high price and difficulty of installing the devices, especially in buildings with finished interiors, which require breaking through walls. Nowadays, there is a wide variety of wireless technologies, from low-power solutions to cellular deployments, designed to connect all devices. These technologies are increasingly able to provide benefits over wired alternatives while still meeting the demands of rigorous systems like fire alarm systems. Some of these technologies, wired (section 3.1) and wireless (section 3.2), will be discussed in this chapter along with their benefits and drawbacks. Before that, some background on possible network topologies will be provided.

3.1 Wired Technologies

As previously discussed, fire alarm systems have kept up with the evolution of technology, so when studying this type of system it is necessary to understand its origins. Until a few years ago, the only way to communicate fire alarm signals was through wires. Although this is a safe way, it is also, as one might imagine, very expensive. Originally, fire alarm systems were wired from the available point in the cities right to the fire department [17].

3.1.1 Types of hard wired systems

Hard wired fire alarm systems can be divided into two types: conventional and addressable.

In conventional systems, responders have only a general idea of which part of the building the fire or device issue is coming from. This happens because these systems are programmed to divide the area into zones, and when a device is activated with an alarm, only the name of the zone will appear on the screen. By this logic, these zone circuits that can be seen in Figure

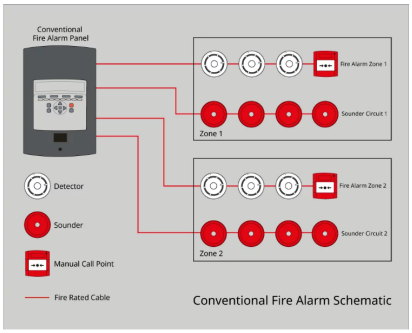

Figure 3.1: Conventional Fire Alarm Schematic [18]

3.1, can accommodate multiple detectors and call points on each radial circuit. Despite being simple to use, the fact that it uses separate radial circuits for both detectors and sounders makes the installation cost sometimes more expensive than the addressable system.

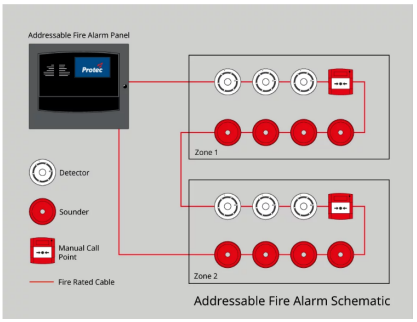

Figure 3.2: Addressable Fire Alarm Schematic [18]

In addressable systems (Figure 3.2), all devices, including sounders and beacons or VADs (Visual Alarm Devices) are organized in a loop configuration that uses less cable compared to a more extensive conventional system [18]. Here, each device utilizes a system-wide protocol to communicate with the control panel. The panel allocates each of the devices with its own unique number/code, which is the device address that needs to be logged by the commission engineer in each device [18]. This way, it is possible to know the exact location of the fire because devices are identified individually. Addressable systems can support a wide variety of fire alarm devices and fire zones. Due to this, an addressable system is perfect for medium-sized to large-scale projects.

3.1.2 Hard wired communications

Regarding the actual communication between devices of fire alarm systems, until a few years back, there were only two ways to communicate fire alarm signals. Obviously, both initial solutions were based on wired systems, which have always been a secure way to transmit information but, at the same time, very expensive because of the amount of cabling that is required.

The first method was reverse polarity [19]. This was nothing less than a turn on/turn off circuit. When the control panel received information that an alarm was happening, it would reverse polarity, lighting a bulb or a horn at the local dispatch center. That was the only

information delivered, a trouble signal that a fire was happening somewhere in the city, no other information was supplied.

The second method was a digital alarm communication system that utilized a DACT (Digital Alarm Communicator Transmitter) connected to a phone line to send signals to the supervising station [20]. At the beginning of this implementation, two analog phone lines were required, and it was a very secure method of transmission. However, with a revision of NFPA 72 (National Fire Protection Association)[21] in 2013, DACTs had to be connected to a single telephone line, with a different technology serving as the second channel (one-way radio, or two-way radio), as can be seen in Figure 3.3.

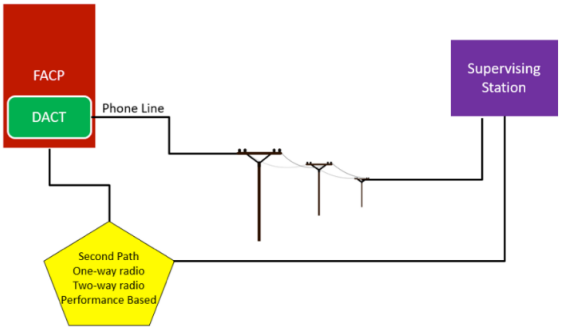

Figure 3.3: Digital alarm communication system [20]

3.1.3 Most used communication buses

As time went by, communication protocols started to improve in terms of efficiency and cost effectiveness. Today, companies developing fire alarm systems eventually choose to develop their own communication protocol for their systems and BOSCH is one such example.

3.1.3.1 Local Security Network

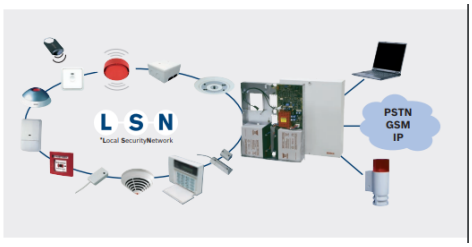

Figure 3.4: BOSCH System example with Local Security Network LSN [22]

Local Security Network LSN [22] is a2-wire bus used by BOSCH for fire and intrusion peripherals, as can be seen in Figure 3.4. It is an addressable solution that, in the case of fire alarm systems, interconnects all the devices involved. It is referred to by the developers as a simple technology that provides a better overview of the system's operation by having the control panel displaying precise, easy-to-read information on each detector. At the same time it is robust in order to maximize security with short circuit isolators, and so a short circuit or a wire break in the loop does not affect the functioning of the bus and connected devices [22].

It incorporates bidirectional digital transmission that allows the detectors and control panels to continuously exchange data between parties, resulting in flexibility in network topologies. One important aspect is that both information and power are transmitted in a single two-wire cable.

3.2 Wireless technologies

As can be witnessed, wireless technologies have had an exponential growth in the last few years. More areas like healthcare, industrial automation, environmental control, security systems, and others are taking advantage of wireless benefits.

3.2.1 Wireless network topologies

It is essential to understand that, besides choosing the best technology for the system, it is also important to consider which topology is the most appropriate.

3.2.1.1 Star Topology

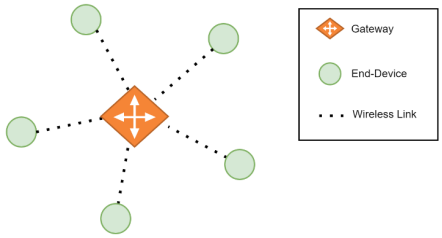

Figure 3.5: Star Topology

As presented in Figure 3.5, a star topology follows a point-to-point architecture, also called "line-of-sight", where there is a central "hub" or gateway that communicates directly with all the devices in the network. On one side, this topology is the one that has the lowest energy consumption. However, it limits the distance that data can be transmitted—often 30 to 100 meters. This makes the star topology applicable for relatively simple and low-power solutions.

3.2.1.2 Mesh Topology

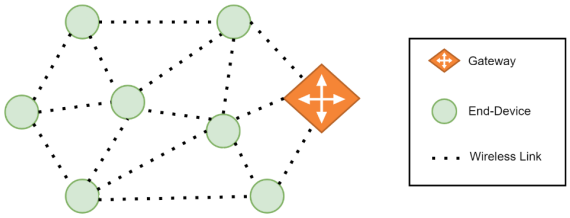

Figure 3.6: Mesh Topology

In a mesh topology, all the devices can communicate with other nodes in the network besides the gateway, establishing multiple links, as presented in Figure 3.6. This topology

brings a multi-hopping capability to the messages that can "hop" between nodes, having multiple possible paths to the gateway. A topology like this introduces easier network expansion, providing wider coverage and higher fault tolerance at the cost of potentially higher power consumption and increased network latency [23].

3.2.1.3 Cluster-tree Topology

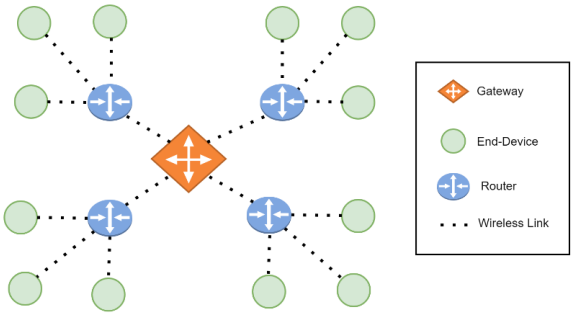

Figure 3.7: Cluster-Tree Topology

A cluster-tree topology is a hybrid of star and mesh topologies where a new element is introduced, a router. These routers communicate with the devices in a star-topology and with the gateway and the other routers in a mesh topology, as can be seen in figure 3.7. In this approach, the advantages of both topologies are combined: the potentially low power concept of the parts in a "star" and the wider coverage and fault tolerance of the mesh portions. However, the disadvantages of the star topology part still apply as there are no backup links on the end devices to each router [23].

3.2.2 Types of Wireless Networks

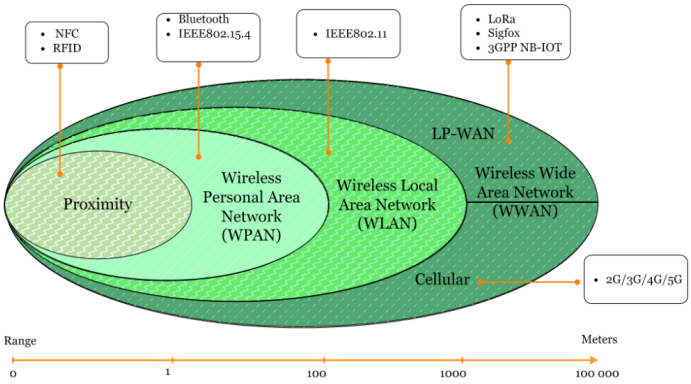

Figure 3.8: Wireless Networks Classification

Figure 3.8 presents a diagram of wireless technologies classification regarding their coverage area.

Below is a brief description of each group of technologies. The following section presents a more detailed view of interesting technologies in each group for the proposed project.

3.2.2.1 Wireless Personal Area Network

WPAN is a wireless area network designed to interconnect devices around an individual's workspace, like earphones, watches, etc. Many of these technologies are managed by the IEEE 802.15 working group, generally having communication range of tens of meters. However, it is possible to create mesh networks, increasing the expected coverage. Despite the coverage gap, these technologies can usually be implemented cost-effectively and easily. Some examples of WPAN are Bluetooth [24] and IEEE 802.15.4 [25] standard.

3.2.2.2 Wireless Local Area Network

With WLAN the purpose is to provide wireless connectivity throughout a limited area such as a school, a home, or a small building. So the coverage area exceeds the WLAN one, being in the hundreds of meters range. These types of technologies often use unlicensed frequency bands, such as 2.4 GHz. They were originally designed to achieve high data rates, as is the case of the well-known Wi-fi.

3.2.2.3 Wireless Wide Area Network

In this group of wireless technologies are those that have a long range, exceeding tens of kilometers. Among these technologies, there are well-known cellular communications such as 3G, 4G, 5G that operate in the licensed spectrum. They generally achieve more significant data rates but are more expensive to implement, and their devices have a short battery life. On the other hand, LPWAN (Low-Power Wide Area Network) is a long-range and low-power technology in the unlicensed spectrum designed for low data rate transmissions, being a cheaper solution than cellular communication. LoRa [26] is a famous example of this group of technologies.

3.2.3 Technologies and protocols

3.2.3.1 LoRa

Short for Long Range, LoRa[26] uses spread spectrum modulation derived from the Chirp Spread Spectrum (CSS) technique, which makes its signals robust and highly resilient to in-band and out-of-band interference mechanisms. Here, chirps, also known as symbols, are responsible for carrying the data. Its main features are extended range, which theoretically covers up to 10 kilometers for wireless communication, low power consumption, and low data rate [26].

This technology has some good features that make it suitable for many application. One example of that is the Spreading Factor (SF) that controls the chirp rate and the speed of data transmission. As can be seen if Figure 3.1, this parameter allows the network to increase or decrease data rate at the cost of range and power consumption.

Despite all the advantages LoRa can bring to wireless systems, there are some limitations. For example, the available frequencies are in the 863 MHz to 870 MHz ISM unlicensed band. This band is available to anyone, which means it is widely used and prone to external interference.

Spreading Factor (For UL at 125 kHz)	Bit Rate (bps)	Range (Depends on Terrain)	Time on Air for an 11-bytes payload
SF10	980	8 km	371 ms
SF9	1760	6 km	185 ms
SF8	3125	4 km	103 ms
SF7	5470	2 km	61 ms

Table 3.1: Lora Spreading factors (adapted from [28])

For this reason, the LoRa Alliance has set some rules for the use of LoRa [27]:

- The maximum power allowed for transmission is 14dBm;
- The duty-cycle (active time in a certain period) of the devices varies between 0.1 and 1%.

3.2.3.2 Bluetooth

One of the most famous wireless technologies is Bluetooth, which is intended for short-range communication (<20m) and its specifications are overseen by the Bluetooth Special Interest Group (SIG) [24].

Over time, the protocol has evolved and expanded its scope of application. Now the technology has defined Bluetooth Low Energy (BLE) for applications with power consumption constraints. BLE is a point-to-point protocol. Therefore, radios cannot communicate beyond their range, typically 10 meters [3].

After that, another evolution of the standard was made, Bluetooth Mesh. It extends the simple point-to-point BLE with additional routing and networking standards to create mesh networks so the network can have a broader range.

Bluetooth V5 also recently presented another mode called Bluetooth Long Range Mode that can achieve ranges over 1 Km but is not yet supported by all BLE chipsets [29].

Nonetheless, BLE is not suited for supporting real-time traffic since message delays cannot be precisely bounded [30].

3.2.3.3 IEEE 802.15.4

IEEE 802.15.4 [3] [31] is a standard that defines the specifications of the PHY and MAC layers. It was designed for low data rate monitoring and control applications that require very low power consumption.

The standard is aligned to specific unlicensed frequency bands available in different geographic regions. In Europe, a single 868.3 MHz channel or 16 channels in the 2.4 GHz frequency band are used.

Some well-known standards, such as ZigBee, WirelessHART, and ISA100.11a, use IEEE 802.15.4 as the PHY and MAC layer along with specific upper layer definitions and characterizations, as shown in Figure 3.9. A new version of the standard, IEEE 802.15.4 enhanced, is also being used more often for low-power applications.

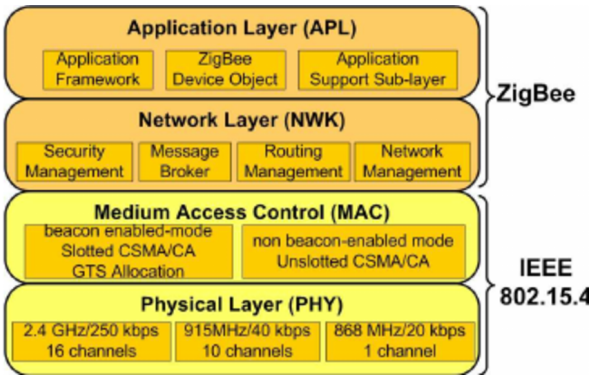

Figure 3.9: ZigBee on top of IEEE 802.15.4 [32]

The IEEE 802.15.4 implements duty-cycling in the MAC protocol to conserve energy by periodically turning off the node's transceivers to maintain the quality required and maximize network lifetime. It is clear that IEEE 802.15.4 offers attractive features for general IoT applications.

However, some recent studies point out some limitations for industrial applications that require high reliability, low latency, and energy efficiency, especially for multi-hop typologies for which no delay guarantee can be provided [33]. As analyzed in [34], a random selection of the binary exponent (in conjunction with CSMA/CA Carrier-Sense Multiple-Access with Collision Avoidance) causes nodes to sleep for extended periods, which degrades throughput. All protocols and amendments presented below attempt to take advantage of the standard while working around the pointed limitations, each one in its own way.

ZigBee . Zigbee is a short-range(< 100 m), low-power standard that uses a PHY and (MAC) of IEEE 802.15.4. It is commonly deployed in mesh topologies to extend coverage by relaying sensor data over multiple sensor nodes. Because of this, Zigbee provides higher data rates when compared to LPWANs, but at the cost of power efficiency configuration [32].

Because it uses upper layers, unlike IEEE 802.15.4, ZigBee adds enhancements that include authentication with valid nodes, encryption, and a data routing and forwarding capability that enables mesh networking.

ZigBee is highly suitable for monitoring and alerting systems, where energy savings are given priority, has said before.

WirelessHART and ISA 100. If, on the one hand, Zigbee is better suited for projects where energy saving is a priority, on the other hand, most of the other communication protocols on top of IEEE 802.15.4 were developed explicitly for factory automation applications. WirelessHART, for example, was designed to support closed-loop supervisory and regulatory applications because of its efficient routing capabilities and a high potential for communication between

multiple field devices to ensure multi-channel frequency hopping. ISA 100 is another example of this application area where safety and system management functionalities are administrated to enable low power and low data rate communications. Like WirelessHART, ISA physical layer is based on the IEEE 802.15.4 standard, but some limitations of the standard are identified in terms of access to the medium, which prevents it from meeting the requirements for applications in industrial scenarios. WirelessHART, for example, was designed to support closed-loop supervisory and regulatory applications because of its efficient routing capabilities and high potential communication between multiple field devices to ensure multi-channel frequency hopping.

WirelessHART is more suitable for the automation process because it can be extended to control the process rather than monitor it.

The implementation of these protocols prompted the introduction of the IEEE 802.15.4e amendment, which incorporates the fundamental ideas of these technologies [35].

IEEE 802.15.4e . Around 2011, the IEEE 802.15 working group proposed the IEEE802.15.4 amendment to enhance and expand the previous version's functionalities to align with industry requirements. Five new protocols, known as MAC behaviors or modes, were introduced, each with particular characteristics and targeting a specific application. Some of these new modes promise real-time, low power, low data rate, and robust communication features. More details will be given in the following sections [25].

3.2.3.4 IEEE 802.11ax (Wi-Fi6)

Unlike IEEE 802.15.4 devices, IEEE 802.11 devices operate at high transmit power, high data rate, and long range. To improve some less appreciated aspects of the previous 802.11ac standard (Wi-Fi5), a new Wi-Fi was developed: Wi-Fi6 or 802.11ax. It is recognized as a next-generation WLAN that uses many new technologies such as Orthogonal Frequency Division Multiple Access (OFDMA) based Multiuser (MU) Transmission , Spatial Reuse (SR), and Target Wake Time (TWT) to improve performance [36].

3.2.3.5 Summary

It is crucial to keep in mind that this project aims to implement a reliable real-time system with low power consumption, so all the options need to be evaluated at this level. In terms of range, reliability, and networking complexity, LoRa-based wireless communications have several advantages over competing technologies. Because of its modulation, LoRa is better suited for multi-level scenarios and avoids situations where repeaters and multi-hop topologies are required, as with other technologies, such as BLE or Zigbee, to implement a high-performance network [37]. For most IoT applications and low power and high coverage scenarios, LoRa seems to be the best option. However, the duty cycle limitation of 1% for device communication makes this option unsuitable for the currently planned system. This duty cycle restriction means a device can only exchange information for 36 seconds in an hour. The fact that we are dealing with a fire alarm system that requires constant checking of the

network is incompatible with this limitation because devices must send repeated notifications. For this reason, Lora is not a viable option.

Among the Bluetooth options, BLE is the closest to the specifications intended for the desired system, as it significantly reduces the power consumption of the communication stack. However, BLE is intended for short-distance communication and is unsuitable for supporting real-time traffic since the delay of messages cannot be precisely bound.

As shown above IEEE, the 802.15.4 standard is widely used for wireless applications, especially in conjunction with upper layers protocols such as ZigBee and WirelessHART. Despite all the adjacent advantages of these protocols, the new version of the physical and MAC layers defined by the IEEE 802.15.4e standard brings promising features for real-time applications that require robustness and low power consumption. So far, there is no evidence that these MAC functions can be interleaved with the specifications in the upper layers such as Zigbee, wirelessHart or ISA 100 to take advantage of both. For this reason, the functions of the extended IEEE 802.15.4 are closer to the desired ones, so they would fit better into this system.

The promises of the new Wi-Fi6 in terms of efficiency and power consumption ratio are very thriving. However, it is not appropriate for the desired target because the devices would still need a power grid for this implementation.

Standard/ Specification		Coverage	*Throughput*	Power Consumption	Cost	Applications
IEEE 802.11	Wi-fi6	30-70 m	600-9607 Mbits/s	High	>50 €/end device	Fixed power supply
Bluetooth	Bluetooth V5	1 km	125-500 kbits/s	Medium	30-60 €/end device	Long range low data rate applications
	BLE	10-100 m	1 Mbit/s	Low	<30€/end device	Short distances High data rate
IEEE 802.15.4	IEEE 802.15.4e	10-100 m	1 Mbit/s	Low	<30€/end device	Real time applications
	ZigBee	100 -300 m	20-250 kbits/s	Low Medium	>30-60 €/end device	Short distances Mesh Networks
LoRa	Plain LoRa LoRaWan	5 km in urban 20 km in rural	0.3 -5.5 kbit/s duty cycle < 1%	Very low	>1000€/base station 3-5 €/end device	Simple connections Low data rate

Figure 3.10: Some Wireless Technologies Comparison

In summary, LoRa would be the most viable choice. However, the duty cycle limitation prevents the use of this technology for this type of application. BLE is not suitable for real-time applications. ZigBee and other upper layer protocols would prevent using the MAC promised features by IEEE 802.15.4e. Wi-fi6 offers desirable features for implementation where power consumption is not a problem.

The technology to be studied in this context is IEEE 802.15.4e, as the new features promise to meet the system's requirements. Thus, the goal is to explore the standard and take advantage of the modes that enable the development of a robust, low-power, low-data-rate, and error-prone network. These modes are explored in the following sections of this paper.

Relatively to the network topology, as the project's main objective is to develop a proof of concept with a prototype, the star topology is the most suitable as it is a more straightforward implementation than the others and can meet the main requirements.

CHAPTER 4

IEEE 802.15.4: An overview

After doing a survey on the existing wireless technologies, it was found that IEEE 802.15.4e has features that respond to the system requirements, being the most suitable option. In order to understand how this technology appeared, how it operates, and what features it has, this chapter provides an overview of it. Going deeper into the standard, it will be explained each feature of the MAC layer, giving emphasis on the most promising one, Time Slotted Channel Hopping. Finally a comparison between these features is done to select the most appropriate for the system

4.1 IEEE 802.15.4- overview

The history of IEEE 802.15.4 standard begins in 2003 when it was first published for WPAN (Wireless Personal Area Networks) [38]. It defines only the physical and data transmission layers and can operate in different frequency bands using simple and effective media access control, similar to the MAC layers of WLAN IEEE 802.11. In both standards, CSMA/CA (Carrier Sense Multiple Access with Collision Avoidance) is used as the channel access method, but 802.15.4 does not use request to send/clear to send (RTS/CTS) frames. This standard was designed to allow the establishment of a low data rate wireless connection between devices that have limited processing and battery level capabilities.

4.1.1 Principle of working

As shown in Figure4.1, IEEE 802.15.4 operates in three different frequency bands and has a different number of available channels for each band. The frequency bands are:

- 868 MHz band for Europe (with 1 channel)
- 915 MHz band for America (with 10 channels)
- 2.4 GHz world wide ISM band (with 16 channels)

Table 4.1 summaries the main differences between the 3 frequency bands where the variation in the data rate stands out.

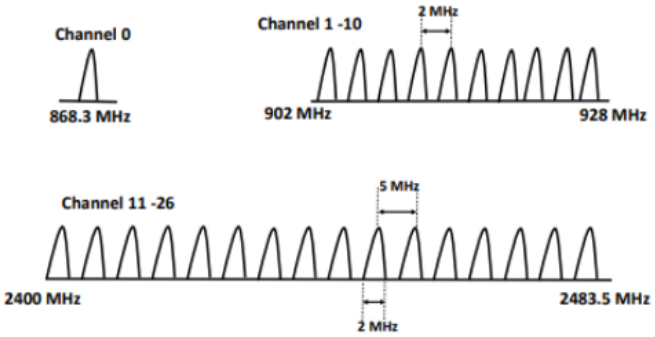

Figure 4.1: Operational frequency bands for 802.15.4 and respective number of channels [3]

Options For Frequency Assignments			
Geographical regions	Europe	Americas	Worldwide
Frequency assigment	868 to 868.6 MHz	902 to 928 MHz	2.4 to 2.4835 GHz
Number of channels	1	10	16
Channel bandwidth	600 kHz	2 MHz	5 MHz
Symbol rate	20 ksymbols/s	40 ksymbols/s	62.5 ksymbols/s
Data rate	20 kbits/s	40 kbits/s	250 kbits/s
Modulation	BPSK	BPSK	Q-QPSK

Table 4.1: IEEE 802.15.4 supported frequencies and characteristics [39]

4.1.2 Network concepts

Normally, IEEE 802.15.4 networks follow a star or peer-to-peer topology, and the devices using this standard can be divided into Fully Functional Devices (FFD) or Reduced Functional Devices (RFD) [40] [3].

4.1.2.1 Fully Functional Devices (FFD):

The Personal Area Networks (PAN) Coordinator is a device that acts as the main controller for all other devices in a network, and it is usually in a centered position as can be seen in Figure 4.2. These devices can perform all functions such as routing, association, and building a network from scratch. They allow other FFDs and RFDs to associate with them to extend the network.

4.1.2.2 Reduced Function Devices (RFD):

These devices are intended for simple applications, as they are not capable of performing routing and association functions themselves. They are typically synchronized with an FFD to transmit data and are typically the end node of an IEEE 802.15.4 network (for example a light switch or a passive infrared sensor).

In this version of the standard, the MAC sublayer is designed to work either on a beacon enabled or a non-beacon enabled mode [42] [3]:

- Beacon enabled:

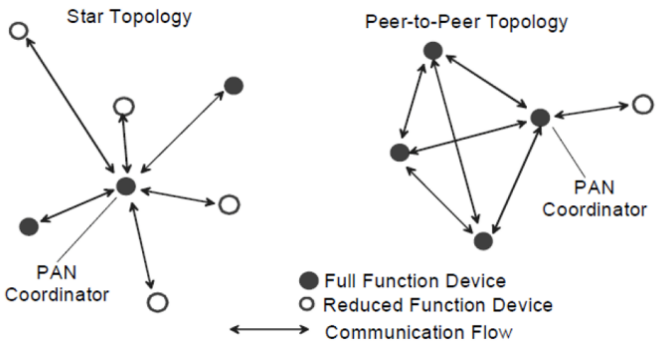

Figure 4.2: Topologies of IEEE 802.15.4 networks [41]

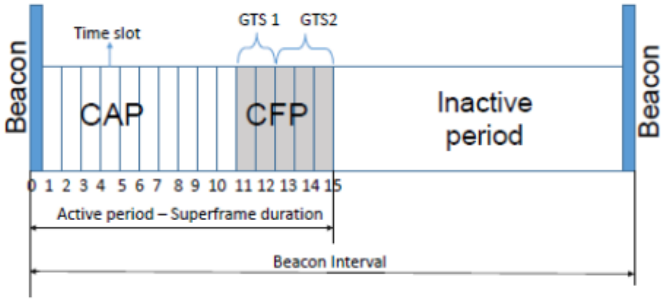

Figure 4.3: Superframe structure of IEEE 802.15.4 [3]

All the network nodes exchange information and wait for a periodic beacon sent by the coordinator in order to synchronize all the network that is supported by a structure denoted as the superframe (Figure 4.3). This superframe consists of an active period: that is divided into the Contention Access Period (CAP) and the Contention Free Period (CFP); and an inactive period. During the CAP, the nodes in the network dispute the access to the channel using slotted CSMA/CA . The CFP is equipped with a maximum of 7 Guaranteed Time Slots (GTS), which are used by the nodes that require guaranteed bandwidth. During the inactive period nodes go into a low power state to save energy.

- Non-beacon enabled:

  The coordinator does not broadcast beacons and there are no superframes. Nodes are always active (energy conservation is delegated to the layers above the MAC protocol) and an unslotted CSMA/CA mechanism is used for channel access.

4.1.3 Frame Structure of IEEE 802.15.4

The IEEE 802.15.4 standard defines a physical layer that was created to address demanding low-power applications involving unlicensed bands, including the 2.4 GHz and sub GHz Industrial bands. For this reason, some requirements in terms of frame size and data rate with restricted transmission power were imposed to achieve lower collision probability, lower packet error rate, and an acceptable range. As a result, the PHY layer can support a maximum PHY service data unit (PSDU) of 127 bytes with a transmission time of 32μs per byte [43].

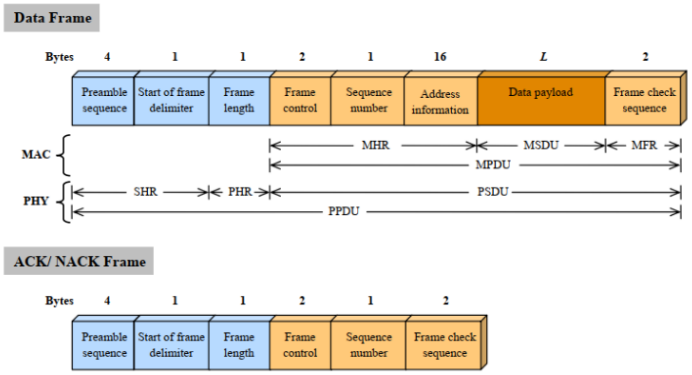

Figure 4.4: Frame structure of 802.15.4 [44]

This PSDU is illustrated in Figure4.4 as well as the remainder of a detailed IEEE 802.15.4 data and ACK (Acknowledgement) frame. The ACK frame has a similar structure to the Data frame with the exception of the address and data payload fields. The two header fields, SHR and PHR, in a PHY protocol data unit (PPDU), stand for synchronization header and PHY header, respectively. PHY service data unit (PSDU) refers to the MAC header (MHR), MAC service data unit (MSDU), and MAC footer (MFR). The first one, MHR, has bytes for specifications and identification of the frame and also for identification of its source and destination. The MSDU contains the payload and, at the end, MFR is the part of the frame that contains the necessary information for error detection.

With this in mind, the payload L, which is the actual application data to be transmitted, can range from 0 bytes to 127 - (2 + 1 + 16 + 2) = 106 bytes [44]. Overall, each frame has 27 slots reserved for specific data frame parameters and 11 bytes for ACK frame. So, admitting that a transmission is composed by a transmission of a data frame and the respective ACK frame, the final equation to keep in mind for the Length of a transmission frame is:

Lt = L+27+11 (bytes); If L = 5 bytes: Lt= 43 bytes

So, theoretically, the time needed to transmit a 5 bytes payload with IEEE 802.15.4 is $Tt = 43 * 32\mu s = 1,38ms$.

4.1.4 Evolution of IEEE 802.15.4- IEEE 802.15.4e

The first revision of the standard took place in 2006, during which the respective market applicability was expanded, improvements were made, and ambiguities in the standard were removed [3]. Some of the improvements are listed below [45] [41] :

1. Improvements in synchronization through broadcast messages in Personal Area Networks (PAN) in beacon-enabled mode.
2. Support for beacon scheduling
3. Improvements at the security level in the MAC layer

Besides all the features presented above, this version of IEEE 802.15.4 MAC continued to suffer from several limitations, such as unbounded latency and low reliability, which make the standard unsuitable for applications with strict QoS requirements. The fact that transmissions are made over a single shared channel results in higher latency and losses due to possible

collisions. In addition, the fact that the standard operates in the same 2.4GHz ISM band as IEEE 802.11 WLAN systems, Bluetooth and microwave ovens makes it more susceptible to interference. Therefore, IEEE 802.15.4 MAC was until then only suitable for applications with flexible latency and throughput requirements [40].

The current version of the standard was approved in 2012 with IEEE 802.15.4e MAC Enhancement Standard, which surpasses the previous standard by introducing two different categories of MAC enhancements (MAC behaviors) to support different QoS requirements of different applications and also introducing other general functional improvements [46].

The main changes introduced in IEEE 802.15.4e and adopted in 2012 can be summarized in the following points [3] [46]:

- Different MAC behaviors, each one with specific features and functions to support a particular application: For example, the Deterministic and Synchronous Multi-channel Extension (DSME) mode provides more flexibility for applications that require high availability and efficiency, or the Low Latency Mechanism Deterministic Network (LLDN), which is very useful for star topologies with low latency requirements. However, the Time-Slotted Channel Hopping (TSCH) mechanism undoubtedly stands out from the others in meeting the emerging needs of industrial applications with low energy consumption devices. The other two MAC behaviors are non-real-time, known as RFID Blink and ACMA [47]. These modes are explained in more detail below.
- Multi-channel access: in some MAC behaviors such as DSME and TSCH, nodes have the ability to access the channel, either through channel-hopping mechanisms with a predefined order or through channel-matching mechanisms where the PAN -coordinator has the ability to assign different channels for data transmission, depending on the channel quality at the time.
- Mechanisms for low latency and low power consumption: the devices can operate in a very low duty cycle, while for the layers above the MAC layer, the node always remains active. It also provides deterministic latency, which is an important requirement for time-critical applications.
- MAC performance metrics: feedback to upper layers on network performance.
- Fast Association: the device requests association from the PAN Coordinator. If resources are available, the device is assigned a short address by the PAN coordinator.

4.1.5 Deterministic and Synchronous Multichannel Extension

The DSME mode [41] [47] [3] [46] [40] has been designed for applications that require high availability, efficiency, scalability and robustness, supporting both industrial and commercial applications. To meet these requirements, DSME is equipped with various mechanisms and features such as Multi-Superframe Structure.

As mentioned earlier, IEEE 802.15.4 provides Guaranteed Time Slots (GTS), but they only include up to seven slots, which means they cannot support large networks. This medium access mechanism increases the number of GTS by leveraging multi-channel operation and combining multiple superframes into a multi-superframe structure (Figure 4.5). This not only

reduces the overall delay, but also increases channel diversity and enables a larger number of transmissions, which increases overall reliability and scalability. The number of superframes in a multi-superframe is determined by the PAN coordinator based on the number of data packets to be transmitted within the time interval and is obtained from formula 4.1.

$$numberOfsuperframes = 2(MO - SO) \tag{4.1}$$

Where MO is the MultiSuperframe Order, which describes the multi-superframe length, and the Superframe Order (SO), which is the length of the active part of the superframe [48].

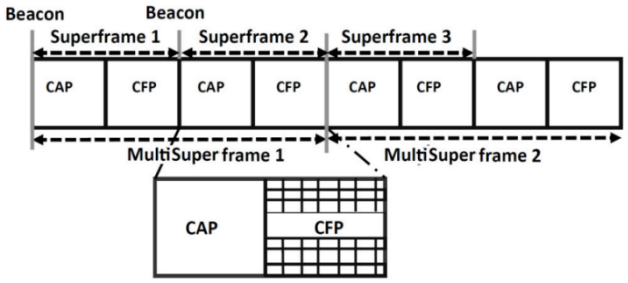

Figure 4.5: Multi-superframe structure of DSME [47]

DSME implements a channel hopping methodology with a predefined set of channels that is decided by the upper layers (hopping sequence) and that is followed by all the devices in the network. It also has other features to save energy like CAP reduction and to reduce latency like Group Acknowledgement (GACK). More about this features on [47].

4.1.6 Low Latency Deterministic Network

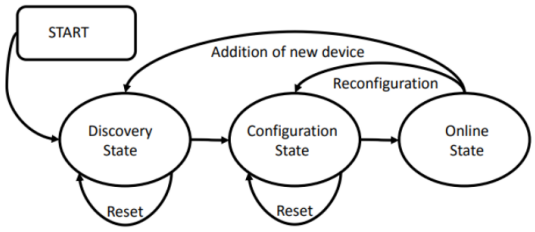

Figure 4.6: Configuration states of LLDN [47]

LLDN mode[47] [41] [46] was developed for industrial and commercial applications that require low and deterministic latency, e.g. robots, cranes, etc. To ensure these requirements, this access mechanism only allows the formation of star topologies (i.e. single hop) and limits the maximum number of transmissions to the coordinator (uplink) and other slots for bidirectional traffic. It also includes a GACK feature to reduce bandwidth overhead. LLDN networks go through 3 phases, as shown in Figure 4.6: Discovery, Configuration, and Online. In the first two phases, nodes build the network, and those nodes that have been lightly used for a while are resynchronized. Also, the necessary configurations are deployed so that the remaining devices can transition to the online state, where the nodes can already send information to the coordinator.

4.1.7 RFID Blink and AMCA

- Radio Frequency Identification Blink (RFID Blink) mode [45]: designed for application domains to identify or track objects or people. In particular, it allows a device to transmit its ID, its address, and optionally additional data in the payload to other devices.
- The Asynchronous Multi-Channel Adaptation (AMCA) mode [45]: is used in large and geographically distributed networks, such as smart utility, infrastructure monitoring, and process control networks. This mode depends on asynchronous multichannel adaptation and can only be used in non-beacon PANs.

4.1.8 Time Slotted Channel Hopping

The TSCH mode [45] is mainly targeted for industrial environments with time-critical requirements, where energy consumption must be minimized and diversity and robustness to interference must be high. In TSCH, nodes are synchronized, allowing timely transmission scheduling, and the frequency hopping mechanism reduces the negative effects of interference between different communication technologies. This mode is discussed in more detail below, as it represents the MAC behavior used in this work.

The TSCH method [41] [45] [3] changes the initial approach of the standard, which makes it different from the rest of the mechanisms proposed in the 2012 amendment.

4.1.8.1 Slotframes and timeslots

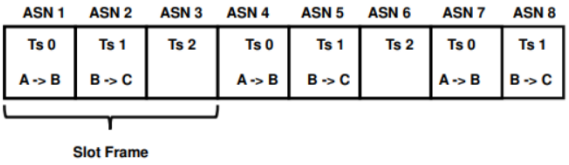

Figure 4.7: TSCH slotframe structure [47]

A TSCH network follows a restrictive synchronization in the sense that the superframe concept of the standard is replaced by the "slotframe" concept. The main difference is that the network nodes are expected to have a common notion of time, so that the slotframe automatically repeats n time slots in cyclic periods. Each slotframe is a collection of time slots. Figure 4.7 illustrates the TSCH slotframe structure. Each time slot can accommodate a transmission between devices with an eventual acknowledgement and is associated with an Absolute Slot Number (ASN), which is the absolute numberof slots that have elapsed since the start of the network [47].

The slotframe size is defined by the number of time slots in the slotframe. The assignment of timeslots to devices within the slotframe may initially be communicated via a beacon, but typically they are configured on the coordinator when the device joins the network [45]. Communication in each timeslot can be either contention-based (i.e., using CSMA-CA) or contention-free using the (GTS) mechanism. TSCH can help accommodate a dense network under strict timing constraints by using fixed-length timeslots in compliance with the standard

and multichannel access [47]. The duration of the timeslots is not defined by the TSCH standard, but literature points to 10 ms, which is enough to send an IEEE 802.15.4 frame and receive its cknowledgement. Figure 4.8 represents the typical structure of the TX and RX timeslots of TSCH, for transmission and reception respectively.

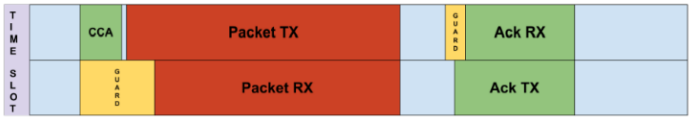

Figure 4.8: A typical timeslot of the TSCH protocol [49]

Thus, in TSCH, the communication between two nodes is defined by the number of the time slot in which the communication takes place and the respective channel offset. The fact that TSCH has high reliability is largely due to the channel hopping mechanism. By using multiple channels, eventual retransmission of a packet will occur on a different channel (frequency) than the first transmission attempt. This means that successive packets between two devices are sent at different frequencies, resulting in a higher success rate for retransmissions.

4.1.8.2 Radio Channel Selection

The frequency of the communication is determined by the channel hopping mechanism and is given by:

$$f = F(\text{ASN} + ch_{offset}) mod C \tag{4.2}$$

where the function F can be defined as a lookup table, and C defines the number of channels used for the current network, corresponding to the channel sequence length. Thus, this mode computes which channel to use locally without requiring any other information from the network, in a pseudo random manner. This way, if a transmission fails, the next attempt to transmit will occur at a different frequency, mitigating the effects of multipath fading or interference [50].

4.1.8.3 Types of Timeslots

In TSCH, slots can be separated into two categories, dedicated and shared slots. Dedicated slots are meant for a direct and deterministic link between two nodes, and transmissions are more reliable because there is no competition between nodes to occupy the channel. Shared slots are for link connections that are intentionally allocated to more than one device (multiple transmitters and/or receivers). Here it is expected that collisions and unpredictable behavior can occur, but shared slots are still suitable for applications with requirements of high throughput and low energy consumption [51]. As seen before, a slot can be TX only or RX only, but it can also be both. It is also possible and usual to have dedicated slots for beacons.

4.1.8.4 TSCH scheduling

Given these multiple possibilities for operations in time and frequency due to multi-channel and hopping, it is important that access to these cells is made in a planned way to preserve

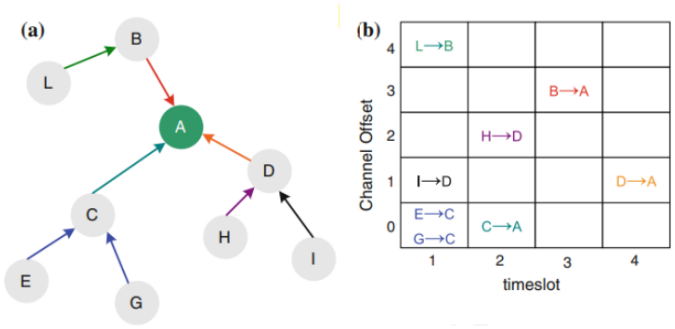

Figure 4.9: A sensor network with a tree-topology (a) with a possible link schedule for data-collection(b) [46]

the deterministic nature of the TSCH mechanism. In order to determine which operation each node in each cell performs (sleep, transmit, or receive), a proper management system known as scheduler is required. In [46], the authors give an insightful example of a possible link schedule of a simple network with a tree topology that helps understanding what a slot scheduler is, as shown in Figure 4.9.

In the example, it can be seen that eight transmissions can be accommodated in four time slots due to the multi-channel approach used by TSCH. It is also noted that all assignments are dedicated slots except one ([1,0])([timeslot, Ch_Offset]). All the devices in the network need to have a notion of this schedule. This way, each node knows when it is time for it to transmit or listen, and if neither is the case, it can go into a sleep mode and save energy.

The scheduling methodology is not defined in the standard and leaves it to the implementation, which is why many studies and proposals are made specifically for the scheduling mechanism for TSCH [46].

Thanks to these features and mechanisms, this method is the one that generates the most interest in the community and leads to various studies and improvements. TSCH is able to meet the requirements of multi-hop WSN networks with high reliability and deterministic behavior. For this reason, it is particularly useful in industrial scenarios where applications require a certain level of QoS and where there is a high level of interference from other systems or obstacles.

4.1.8.5 Timing of a timeslot in TSCH

The former timing calculations for the IEEE 802.15.4 frames have in consideration the size and content of the transmission packets and the respective acknowledgements. However, there are more things to consider when analyzing transmission and reception times. As can be seen in Figure 4.10, with TSCH, the RX/TX and ACK packets are handled within a timeslot, typically 10 ms long, that has some delays and preparations in order to certify that the medium and the radios are ready to send/receive this information. For example, after a "TsCCAOffset" Delay, a clear channel assessment mechanism will determine whether the medium is idle or not and only after that the transmission will be initiated. This CCA is optional [50]. The time between the beginning of the timeslot and the "TsTxOffset" is the time necessary for the radio to be enabled to transmit, and this offset is the exact time for

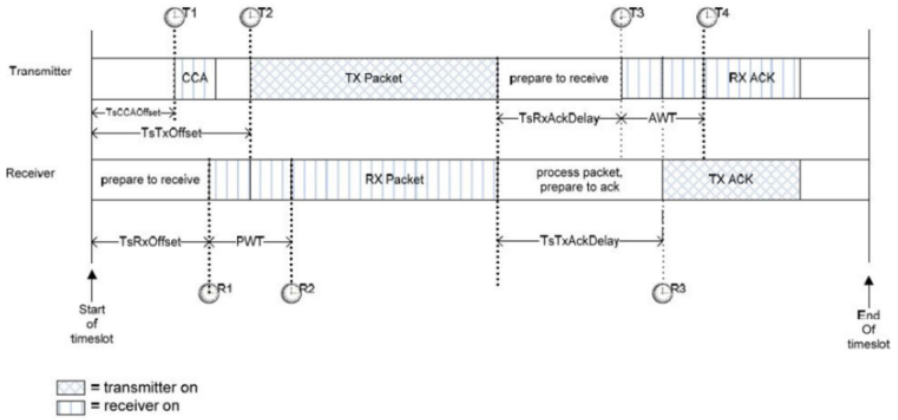

Figure 4.10: Timing of a timeslot [50]

the transmission of the packet. After the transmission, the radio needs to be prepared for the reception of the respective acknowledgement, and so the TSRxAckDelay needs to be respected. The reception of this feedback occurs following an Acknowledge Wait Time. A similar approach happens when receiving a packet. There is a TsRsoffset for radio preparation followed by a Packet Waiting time.

Since the transmission of the data and ACK packets takes 1,38ms for the 5 bytes analyzed earlier it remains 10-1,38 = 8,62 ms for these delays and preparations as well as some possible security operations.

4.2 Summary

A general comparison between the main MAC behaviors on IEEE 802.15.4e standard is shown in Table 4.2 . It is important to point out that the main requirements of the project of this dissertation were low power consumption and high reliability, and the main goal is to build

MAC behaviors and variants	Scalability	Reliability	Power Efficiency	Multichannel Access	Available tools Simulations: S Hardware: H Non Available: NA	Applications Intended
RFID	-	medium	high	no	S,H	Tracking and identification
AMCA	yes	medium	medium	yes	NA	Non time sensitive applications with high scalability requirement
DSME	yes	high	depends on the number of nodes	yes	S	Time critical applications with high scalability non critical applications
TSCH	yes	very high	higher than DSME	yes	S,H	Time critical applications with energy efficiency requirements
LLDN	no	very high	higher than DSME	yes	S	Industrial Automation-fixed number of nodes

Table 4.2: Comparison of the MAC behaviors (adapted from [42])

a real-time wireless fire alarm system, so determinism and reliability are key requirements. Two modes mentioned so far can be excluded from consideration because they are not intended for real-time applications, namely RFID Blink and AMCA. LLDN is also a non-ideal mode for this application due to its limitations in network structure and the use of a single channel for communication, which makes it more vulnerable to interference and reduces overall network scalability, Moreover, this MAC behavior is intended for simple systems with a fixed number of nodes, which is another limiting factor. The other two remaining modes are very similar in their advantages and disadvantages, even though they have different methods and functions: DSME and TSCH. Both modes are suitable for time-critical applications and offer good scalability. However, the first mode has a rigid structure (DSME multi-superframes) since the number of superframes in this structure grows with a power of 2. This fact poses a problem when the number of nodes in the network is small, since it leads to an excess of GTS, i.e more slots are used than those actually needed, which directly affects the bandwidth efficiency and leads to long transmission delays. This fact is confirmed in [49] when the authors evaluate the reliability, scalability and delay of DSME and TSCH. For this reason, it is said that TSCH has better delay performance than DSME when the number of nodes is less than 30 [49]. On the other hand, for larger networks, DSME achieves better results compared to TSCH due to the smaller size of its slots [49]. Unlike DSME, TSCH has a different network supporting infrastructure that can be fully set as a contention-free period, which means that the energy efficiency of TSCH is theoretically higher than that of DSME. Until the day of the decision on the technology to be used, no hardware platforms that support and implement DSME have been found, only studies on simulations and surveys on open questions of this mode. Considering all the above points, TSCH proved to be the more suitable MAC behavior, because of all the features added to IEEE 802.15.4, namely the channel hopping mechanism that provides robustness to mitigate possible interference while allowing the use of low power devices. Another important aspect is that this mode is very popular in the community, so more people are working with it to test and improve it, and thus more information is available.

CHAPTER 5

Development Platform

Having the technology chosen, the next step is to select the development platform suitable for the proposed system. This chapter presents some possible hardware solutions and also some software options typically used for this kind of systems. After that, an overview of the chosen operating system will be done in order to understand its basics features.

5.1 Assessed development platforms

Supporting TSCH on a platform requires some specific features in the hardware, communication stack and radio drivers, so the main goal was to identify commercial products that support this protocol [52]. On the hardware side, it was important to find low-power boards that supported IEEE 802.15.4 whose suppliers could be trusted. At the same time, solutions were sought that made the protocol stacks and radio drives available natively, without forgetting the quality of the documentation.

5.1.1 Hardware Platforms

There are a few platforms that have the radio features required for TSCH and thus are possible hardware solutions for this project:

- From Zolteria: Z1, Zoul CC2538, Zoul CC1200
- From Texas: CC2650 LaunchPad, CC2652R1LaunchXL, CC1310 LaunchPad, CC2538DK
- From Nordic: nRF5340dk, nRF52840dk/dongle
- Others: OpenMote-CC2538

However, to implement the TSCH MAC behavior specifications, radio drivers and software are required.

5.1.2 Software Platforms

The platforms presented above can not implement TSCH alone. This section presents three popular software solutions that came up to use IEEE 802.15.4 and possibly implement TSCH: Simple Link TI, Nordic nRF 802.15.4 Radio Driver, Contiki SO and Contiki NG.

SimpleLink TI. The TI SimpleLink MCU Platform is a TI's microcontroller platform [53] for wired and wireless devices, which provides a single software development environment for all SimpleLink devices.

This Software Development Kit (SDK) provides a comprehensive software package for the development of Sub-1 GHz and 2.4 GHz applications including support for Bluetooth Low Energy, Mesh, Zigbee, Thread, 802.15.4-based, proprietary, and multiprotocol solutions on the SimpleLink CC13xx and CC26xx wireless MCUs and others wireless ARM -based MCUs.

Texas Instrument (TI)is famous for its extensive documentation aggregated with its devices. At the moment of this research, there were no references regarding the support of TSCH MAC behaviour. After contacting Texas Instrument Online Support, the Software Development Team has confirmed that "the TI 15.4-Stack does not include support for the TSCH channel hopping scheme".

Nordic nRF 802.15.4 Radio Driver . Nordic is another well known company because of its robust documentation[54]..

Regarding their software solutions, Nordic has a set of drivers for the hardware platforms presented in Section 5.1.1, the nRF IEEE 802.15.4 radio driver. Its architecture is independent of the operating system and the IEEE 802.15.4 MAC layer. This allows the use of the drivers in any stack that is based on IEEE 802.15.4 and implements protocols such as Thread, Zigbee, or RF4CE [55]. Nevertheless, it was confirmed that some of the MAC behaviors of IEEE 802.15.4e, like LLDN(Low Latency Deterministic Networks) were not supported in their drivers and, consequently, their devices. As for TSCH, the developers confirmed that it was possible to implement it, but it would have to be done from scratch [56].

Contiki OS and Contiki-NG . Contiki [57] is an open source operating system that runs on tiny low-power microcontrollers. It allows the development of applications that make efficient use of the hardware while providing standardized low-power wireless communication. Contiki is widely used in numerous commercial and non-commercial systems such as city sound monitoring, street lights, alarm systems, remote house monitoring, and so on.

Another important aspect of Contiki is its flexibility with various platforms including a useful application for simulation that allows the emulation of real hardware (Cooja). It also supports popular platforms like Zoletia (used in [3]) and SimpleLink from TI allowing the use of trustworthy devices from TI [58].

Note that there are other software platforms that propose the implementation of TSCH mode [59]. However, most of them do not have concrete evidence that it was deployed with success in real devices, as is the case of TinyOs [60]. OpenWSN [61] was also a promising project focused on an open source implementation of TSCH. It was expected to run on 11 hardware platforms, from MSP430-based motes to state-of-the-art Cortex-M multi-radio boards [62]. However, the firmware seemed to be unavailable at the moment of this research.

5.2 General comparison of development platforms

Table 5.1 shows the main characteristics of the hardware platforms available to ship to Portugal, with the respective supported SDK or OS presented above. As stated before, the main problem with the Nordic solution was that it did not have ready to use radio drivers and stack. Following this solution would require developing the TSCH stack from scratch, which would be beyond the scope of this dissertation. Because of this and the fact that the cost of the hardware is higher than the other solution, this one was excluded.

Development kit	Manufacture	Peripherals	Price	SDK/OS	Documentation	TSCH Support	Frequency operation
nRF5 Series DK	Nordic nRF	UART,USB	44,59€[63]	nRF Radio Driver	Robust and organized [54]	Supported but it had to be self developed	2.4 GHz
Launchboad XL CC13xx and CC26xx	Texas Instrument	GPIO, I2C, SCPI, UART, USB	35,10 €[64]	Simple Link TI ->TI-RTOS	Robust with many examples[53]	Not included	CC1310- SUB-1GHZ CC2650-2.4 GHz
				Contiki NG	Github repository with many contibutions [65]	Yes	CC1310- SUB-1GHZ CC2650-2.4 GHz

Table 5.1: Development boards comparison

As can be seen in Table 5.1, SimpleLink solution presents a lack of support for IEEE 802.15.4e MAC behaviors so, the option of using this SDK was also excluded. Nonetheless, it served to point in the right solution, the TI board with Contiki Operating System (OS).

Contiki has support for other well-known boards like Zolteria, however TI is a recognized company and their boards for this type of implementation where affordable. Within the TI board options the Launchpad XL stood out and the next step was to choose which SoC was necessary according to the desired operating frequency. For Sub- 1GHz support, CC1310 would be indicated and CC2650 was meant for the 2.4GHZ band.

As said in section 4.1.1 of this report, IEEE 802.15.4 has different numbers of channels available depending on the frequency of operation. In Europe, working on Sub-1GHz would mean work on 868 MHz band with only 1 channel available which would go against the important feature of TSCH, the channel hopping mechanism.

For this reason the CC2650 Launchpad was chosen.

However, it should be noted that there are multiple advantages to using Sub-GHz bands, as they promise longer communication range, and less interference with widely used technologies such as BLE, Wi-Fi and others [66].

5.3 Contiki Operating System

The application requirements impose the use of resource constrained devices, with limited RAM, Flash and processing power capacity, which means that an accurate consideration of the embedded software is required. In particular, it was considered essential to find flexible solutions, that allowed further developments of WSN and IoT, while coping with the hardware resource limitations.

In 2006, the Contiki Operating System (OS) was open-sourced, but its development started around 2003, making it an early desired solution for WSNs. It is a cross-platform operating system designed for severely constrained wireless embedded devices and later adapted for more powerful devices [67]. Some of the main strengths that made Contiki a major step in the evolution of modern IoT operating systems and for WSNs are listed below [67]:

- It was designed for resource-constrained wireless sensor devices with program memory in the order of 100 kB and less than 10 kB of volatile memory.
- Unlike other "competing" software like TinyOS [60], Contiki uses the standard C language
- It has an event-based kernel which allowed a fast response time to external events with energy efficiency
- Contiki stacks offer support for standard and well-known protocols, such as IPv6, Routing Protocol for Low-Power and Lossy Networks (RPL), 6LoWPAN, and CoAP, while supporting PHY technologies, such as IEEE 802.15.4, Wi-Fi or BLE.

5.3.1 Contiki OS vs Contiki-NG

Contiki OS was widely used as open source software and was always open to user suggestions and contributions. After a while, some limitations began to appear, such as some supported platforms becoming obsolete with the evolution of embedded devices. With the replacement of 8-bit and 16-bit microcontrollers by 32-bit ARM Cortex-M based devices, witch allowed new features, the operating system would had to be adapted [67]. Beyond that, the wireless sensor network research field evolved, and interoperability and support for new standards became more and more important. Adapting the code to overcome these limitations would mean considerable complexity. In 2017, Contiki-NG appeared as a huge revision and optimization, with new configurations and a major cleanup of the code base. This update eventually entailed the removal of obsolete protocols and standards, minimizing the final binary size [68].

5.3.2 Contiki-NG architecture

Conceptually speaking, the Contiki source code base can be divided into 2 parts [68]:

1. Hardware independent part, that hosts all the implementation for the “hardware-blind” components of the OS such as the kernel, data structure libraries and networking protocols.
2. Hardware-specific parts consist of drivers for hardware components, including timers, peripherals, and radio interfaces. This, along with several hardware abstraction layers, allows the use of well-defined APIs (Application Programming Interfaces) specific to some hardware without having to develop new interfaces.

The currently available version of Contiki has various practical examples for many development boards including the one chosen for the project, the CC2650 Launchpad. This will be much helpfull in the development of the system since it introduces an abstraction from the hardware specifications [69].

5.3.3 Process and events

Contiki-NG relies on the event-driven model of the original Contiki OS with a cooperative multitasking style. However, the original support for preemptive multithreading has been

removed in this latest version. Therefore, the Contiki-NG scheduler will never force a context switch from one running process to another. However, the execution process can get interrupted unexpectedly by hardware interrupt handlers [70]. In a standard execution scenario, each process will leave its idle state when an event occurs and use a desired resource until a chunk of the work is done. Then, the process will suspend its own execution until another event is received. Such events can be an incoming network packet, a timer expiration, or a serial line message arriving. In Contiki-NG, processes are built on top of a lightweight threading library called Protothreads [71]. Protothreads are stackless threads that remove the overhead of allocating multiple stacks, making them ideal for memory-constrained systems. Rather than using complex state machines or multithreading, prototheads implement a sequential flow of control [72].

5.3.4 Networking support

As said before, Contiki-NG was developed to enhance the support for networking standards on resource-constrained embedded devices.

Figure 5.1 illustrates an overview of the network stack of Contiki-NG whose protocols are explained below.

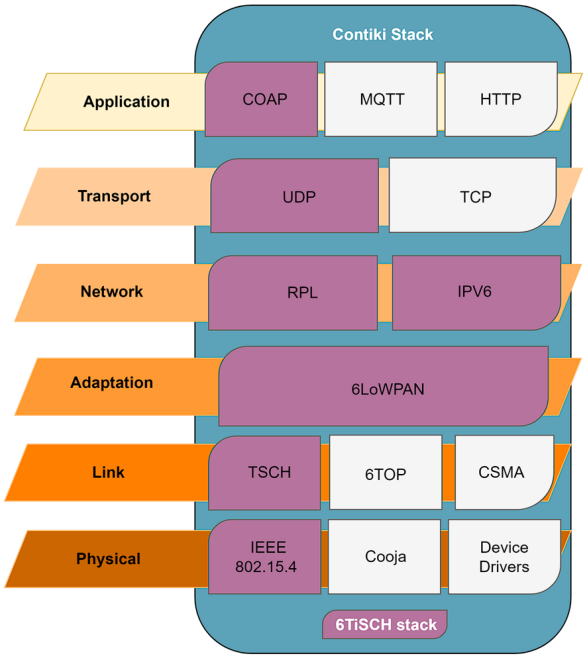

Figure 5.1: Contiki Network and 6TiSCH stack (adapted from [73])

5.3.4.1 Application Layer

Contiki-NG supports application-layer protocols and profiles such as CoAP(IETF Constrained Application Protocol) and MQTT(Message Queuing Telemetry Transport). CoAP is a specialized internet application protocol similar to Rest-like applications but easier to implement and with lower bandwidth costs for building web applications. It is transported

over UDP with a block transfer feature instead of TCP, as is the case with HTTP, because CoAP provides support for acknowledged messages [73].

Contiki-NG also features a client implementation of MQTT , allowing subscription and pushing to a public MQTT broker. More information about this topic is out of the scope of this dissertation.

5.3.4.2 Transport Layer

Contiki supports the well-known protocols, UDP (User Datagram Protocol) and TCP(Transmission Control Protocol). UDP is a protocol to transfer data between two network devices with no need for handshaking dialogues and no guarantee of delivery or ordering, which can be a disadvantage if it is not performed on other layers. This fact also means an avoidance of the overhead of such processing in the protocol stack. On the other hand, TCP has a three-way handshake feature and error detection, which adds reliability but increases latency and overhead [74]. Despite all this, there are more examples and implementations with UDP since it has less overhead when compared to TCP, and other layers can deal with acknowledgement, making it typically preferred for real-time deterministic low-latency operations.

5.3.4.3 Network Layer

IPv6: With the evolution of technology, more and more devices are expected to be connected to the internet, so IPv4 would no longer be able to accommodate them all. IPv6 addresses the problem by increasing the IP address size from 32 bits to 128 bits, among other advantages[73]. In a network, IPv6 provides identification for each device.

RPL: RPL is a relatively recent routing protocol, specifically usefull for low-power and lossy networks (LLNs) and their limitations on energy-consumption and processing capabilities. In RPL, networks are represented in Directed Acyclic Graphs (DAG) with the possibility of dividing them into more than one DAG in the same network for different root nodes, usually called Destination-Oriented DAG (DODAG)[25]. A DAG is a tree-like structure with a single root node, which has no parents and usually represents a border router. This graph organizes each link in such a way that closed loops can not occur when defining routes between nodes. All the nodes are ranked according to their distance from the root, using neighbors with smaller ranks to forward their packets to the root, which means it is especially useful in multi-hop networks [73].

5.3.4.4 Adaptation Layer

6LoWPAN (IPv6 over Low Power Wireless Personal Area Network) appears as an adaptation layer working on top of IEEE 802.15.4 MAC so that IPv6 packets can be carried over multiple IEEE 802.15.4 packets, dividing IPv6 frames into several smaller segments. In other words, its main mechanisms consist of fragmentation and reassembly of IPv6 packets that have a maximum transmission unit of 1280 bytes, unlike IEEE 802.15.4 packets that have a maximum of 127 bytes. In order to reassemble packets correctly after fragmentation, additional header information is created as presented in Figure 5.3 [75]. It can also compress

the IPv6 header from 40 bytes to a maximum of 2 bytes depending on the communication scenario [75].

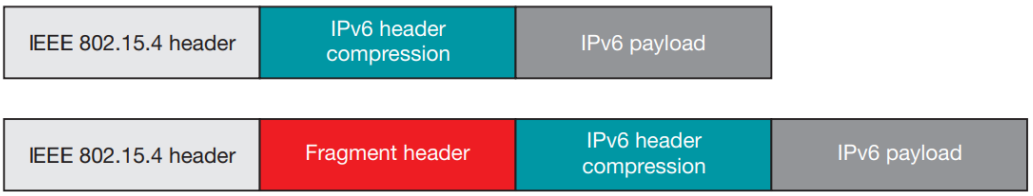

Figure 5.2: Additional header data on 6LoWPAN fragmentation [75]

5.3.4.5 Link Layer

6Top stands for 6TiSCH Operation Sublayer and is a logical link layer that provides abstraction of IP links over the TSCH MAC layer. 6Top objective is to facilitate the dynamic cell negotiation that might be needed between two direct neighbors in the network. For example, if a node A communicating to a neighbor node B determines that the network traffic is lower than the capacity of the TSCH cells scheduled for that communication, a 6P transaction is triggered from node A to delete one or more cells in the TSCH schedule of both nodes [70]. This means that 6Top has some Scheduling Functions (ScF) that decide when to add/delete a cell to/on a neighbor. In Contiki, these ScF can be personalized and 6Top can be deactivated [76]. To sum up, this layer adds the possibility of distributed schedule management in a 6TiSCH network.

5.3.4.6 MAC layer

Contiki-NG implements IEEE 802.15.4e that mainly defines two MAC layer modes, i.e., unslotted CSMA-CA and Time Slotted Channel Hopping (TSCH). TSCH provides better robustness against interference than single-channel MAC protocols such as CSMA-CA MAC of IEEE 802.15.4-2011.

5.3.4.7 PHY Layer

Contiki-NG has a radio driver layer that handles low-level radio operation for specific hardware. It is also important to highlight the Contiki open-sourced simulator for IEEE802.15.4 networks, Cooja, that can simulate the real behavior of a sensor network based on the Contiki communication stack [51].

5.3.5 Contiki 6TiSCH stack

6Tisch is an acronym for IPv6 over the TSCH mode of IEEE 802.15.4e that integrates IPv6 into industrial standards because it defines a custom protocol stack suitable for low-power lossy industrial wireless networks. Figure 5.1 presents the protocols already defined by IEEE 802.15 at various layers of the protocol stack, underlining those that are also part of the 6TiSCH stack [73].

5.3.5.1 Network formation

TSCH networks have a central node known as coordinator that typically is also the RPL root node. It is responsible for periodically transmitting EBs to the surrounding nodes and keeping them synchronized. Other nodes that want to join the TSCH network begin the joining process by scanning the medium for beacons. Once a beacon is received, the network information is passed to that node, i.e., the Absolute Slot Number (ASN), join priority, TSCH schedule, and hopping sequence, and the node associates itself with the TSCH mode and becomes part of the network [51].

5.3.5.2 TSCH scheduling in Contiki

As said in section 4.1.8.4, the slot scheduling policy is not defined in the standard, which is why many studies and proposals are made specifically for the scheduling mechanisms for TSCH.

Most of the developed schedulers focus on applications with non-static networks, i.e., there is an interest in easy adaptations to possible changes in the network (example: introduction of a node or a change in priority).

In general, scheduling can be classified into autonomous, distributed, and centralized schedulers.

The Contiki-NG operating system has a distributed schedule with TSCH minimal by default and the possibility to include the autonomous scheduler Orchestra as well. In the default case, the scheduling is stored in the node, which means that there are no processes for cell allocation. Thus, all types of traffic are sent over a single scheduled cell for all nodes, a shared cell where collisions have a high probability. In Orchestra, each node computes its own TSCH schedule and updates it when there are changes in the network through EBs. For this reason, Orchestra is usefull for dynamic networks.

Contiki NG also allows the developer to define their own custom schedule and has an example file that shows how to manually add cells in the TSCH schedule [77]. This way, it is possible to implement a specific schedule for the developed system. This degree of freedom is great because it will allow the customization of the system. For example, by implementing this custom schedule, it will be possible to allocate 255 devices, increasing the size of a slotframe so that each one has a cell to communicate. Note that this increase has other consequences that will be analyzed in the testing section, such as the increase of worst case end-to-end latency.

5.3.5.3 Time Synchronization

All the theory behind TSCH requires full synchronization between all the nodes in the network. Initially, the synchronization is achieved by each node through the exchange of beacons with the beacon source (coordinator). However, this single time synchronization is not enough since the time source in each device (usually crystal oscillators) suffers from time drift [51]. For this reason, TSCH in Contiki synchronizes each node with the coordinator whenever a packet is received from the coordinator or in an acknowledgment. In the first

case, each node calculates the time difference between the arrival of a packet and its expected arrival time inside that timeslot. In the second case, the coordinator calculates the drift and sends it in the acknowledgement. After that, the node corrects its time drift [51].

5.3.5.4 Contiki 6TiSCH stack software description

In Contiki-NG, the functions of each protocol are exposed as a set of services, and the higher layers can use the services of the lower layers in the stack [73].

In order to implement the desired system, it was necessary to understand the processes and functions that the operating system provides for this stack. Figure 5.3 and its description below intend to show some of the existing functionalities in Contiki for the implementation and manipulation of TSCH for a better understanding of the implemented solution (Chapter 6).

When a device is connected and the application has support for TSCH (see configurations in Section 6.2.5), the program will do all the needed initializations and configurations for TSCH MAC behaviour with the function tsch_init(). After that, if the device in question is a coordinator, the process "tsch_process" declares it as such and starts its function with tsch_start_coordinator() to establish the TSCH network. However, if it is an end node, this process evokes a protothread to start scanning the medium until association.

When there is an association, both devices will be officially in a TSCH network and the "tsch_process" will remain yield as long as this continues evoking several handlers to implement the TSCH functionalities. For example, both nodes will periodically send TSCH EBs (Enhanced Beacons). The process "tsch_pending_events_process" handles the TX or RX input callbacks. The developed code will focus mostly on these callbacks. Inside this process there is an auto-reschedule function called tsch_schedule_slot_operation() that will manage most of the TSCH behaviour. This function sets a global time with the ASN and "jumps" between slots to search for active links (activated by the schedule) and manage what the link requires (receive, transmit or sleep). Here it is calculated the time to the next wake up of the node, for example. It is also done, if needed, the drift compensation.

These processes, which maintain the TSCH network functional, will continue until a there is a reset of the board or it is disconnected from the power source.

In the event that the node disassociates from the network, for example when an end node is to far from the coordinator, the connection is lost and all the process will stop and the medium will be scanned to try association again.

Contiki has a special set of buffers with different sizes used in the respective layers through which a packet can traverse from the link layer to the application layer[73]. So, when TSCH is activated and a node has a message to transmit, the information will be copied from buffer to buffer going upper in the layers. When it has all the information needed, it is delivered to the respective destination in the appropriate slot.

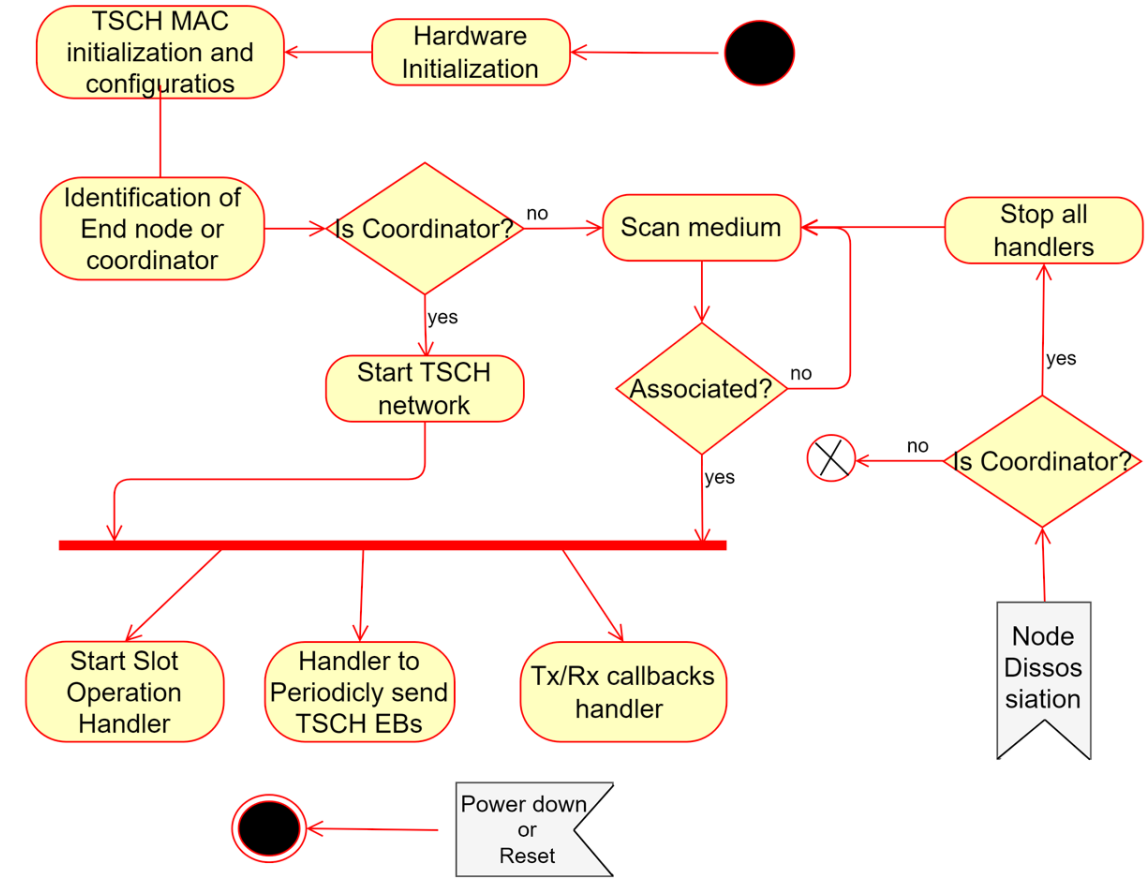

Figure 5.3: Activity diagram of Contiki 6TiSCH relevant processes

CHAPTER 6

System Development

This chapter focuses on presenting the steps made towards the development of the project with the platform and technology chosen. Initially, it will be given an overview of the architecture and models, having in consideration the initial requirements. Then the steps that were taken in order to implement the system as required will be presented. This chapter summarizes the practical work done, that is, what tools were used, what had to be programmed and modified, and how, serving as a tutorial.

6.1 System Architecture and characterization

This section presents the system architecture of the system developed keeping in mind the requirements presented in section 1.2. After that,it is introduced the functional characterization of the system, including the envisioned memory model and schedule, as well as the expected communication flow between the nodes in the network for the fire alarm system.

6.1.1 System architecture

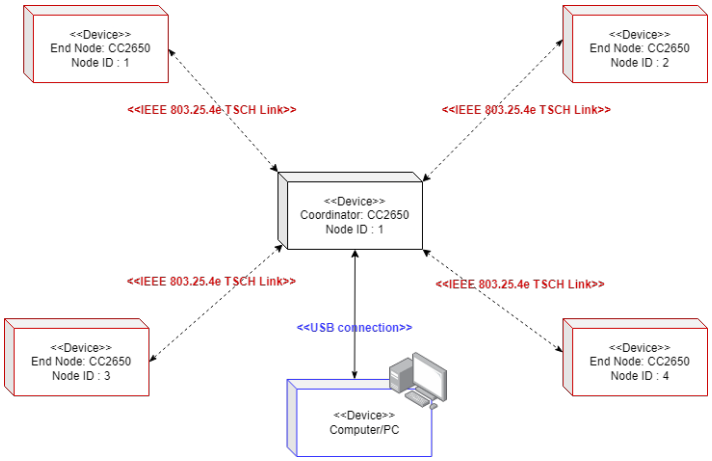

Figure 6.1: System architecture (adapted from [73])

For a proof-of-concept, the network topology implemented was a star topology where all the end-nodes have a point-to-point connection with the coordinator/gateway device, as shown in Figure 6.1. The central device is the TSCH coordinator that will broadcast periodic beacons and act as a time source for all the end-nodes in the network. It will also work as a bridge to all the network-relevant information that will be displayed in the user interface present on the PC. The end nodes will be connected to sensors and periodically send their status to the coordinator. When an alarm or a malfunction is detected, the central device will receive this information and act on it. The PC is connected to the coordinator via USB and will be used to display the network information in a user-friendly way, reporting any issues or alarms.

Figure 6.1 presents the setup used for the proof of concept. Ideally, each end node would be connected to a sensor (heat, CO2, smoke, etc.). However, for the proof of concept, it was decided to use a button to simulate an alarm event when it is pressed. This button is identified in Figure 6.2 (BTN-1) as well as all the other relevant ones.

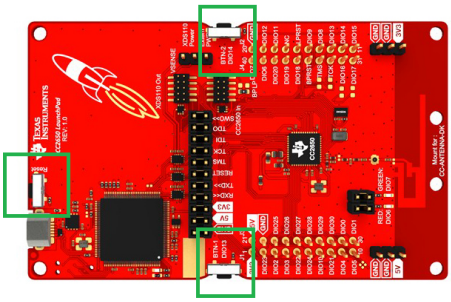

Figure 6.2: CC2650 Lauchpad buttons

6.1.2 Memory model

Figure 6.3 represents the memory model for the devices in the network. The memory model contains the registers manipulated by the devices where each register has information that can be accessed periodically or asynchronously, either due to a request from another device (usually the coordinator) or due to a trigger (alarm). Once accessed, this information will be sent to the intended destination at the right time (slot).

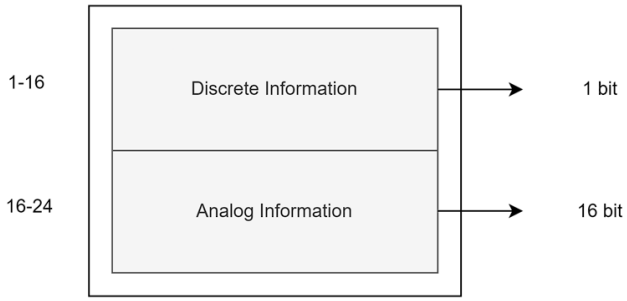

Figure 6.3: Memory model of the system

The arrangement of the memory in a device can be divided into areas, where each one is dedicated to different types of information to be stored. For example, the "Discrete Information" area will have 16 1-bit registers for status information such as whether an alarm or heart bit occurred. For more complex information, the second memory area, “Analog Information”, will contain 8 registers of 16 bits each for reading/writing analog information, such as information from a carbon monoxide/dioxide sensor or any other analog data.

6.1.3 Network communication model

The process of network formation and the subsequent expected exchange of messages and alerts between the devices is represented in Figure 6.4, which goes as follows: One device is set to be the network coordinator and starts broadcasting Enhanced Beacon (EB) messages to try to find TSCH associations. Likewise, the end nodes will start scanning the medium, switching from channel to channel until at some point the coordinator’s beacon is received on the channel that the end node is scanning triggering the network association process. After that, a TSCH network is established and the exchange of information can begin.

In the pretext of wanting to develop a proof of concept to study the limits of the technology, a status message is to be sent periodically every 1 second with the current status information of the respective end-node registers. When an alarm event is detected, the status of "Reg1" will change immediately, but this alteration will only be reported to the coordinator after the established period is up.

After that, the coordinator will communicate this alarm to the control station when possible, and will be responsible for resetting the alarm triggered in the sending this information back to the respective end-node.

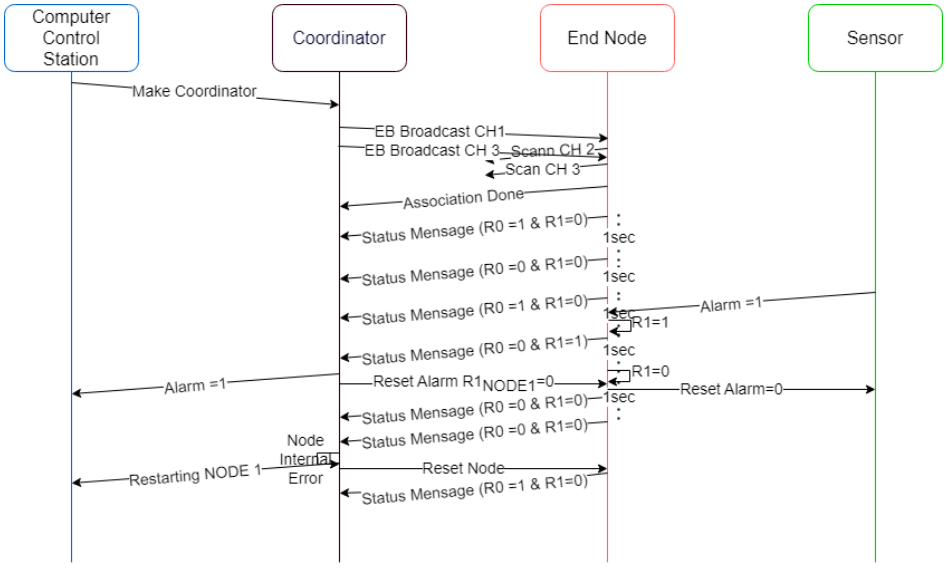

Figure 6.4: Sequence diagram of the communication model

6.1.4 TSCH Schedule

As mentioned earlier, there is a need to specify the scheduling of each slot since TSCH does not do this by default, although it supports it. Each node in the network needs to follow

a schedule in order to understand when it is time for it to sleep, listen, or transmit some information.

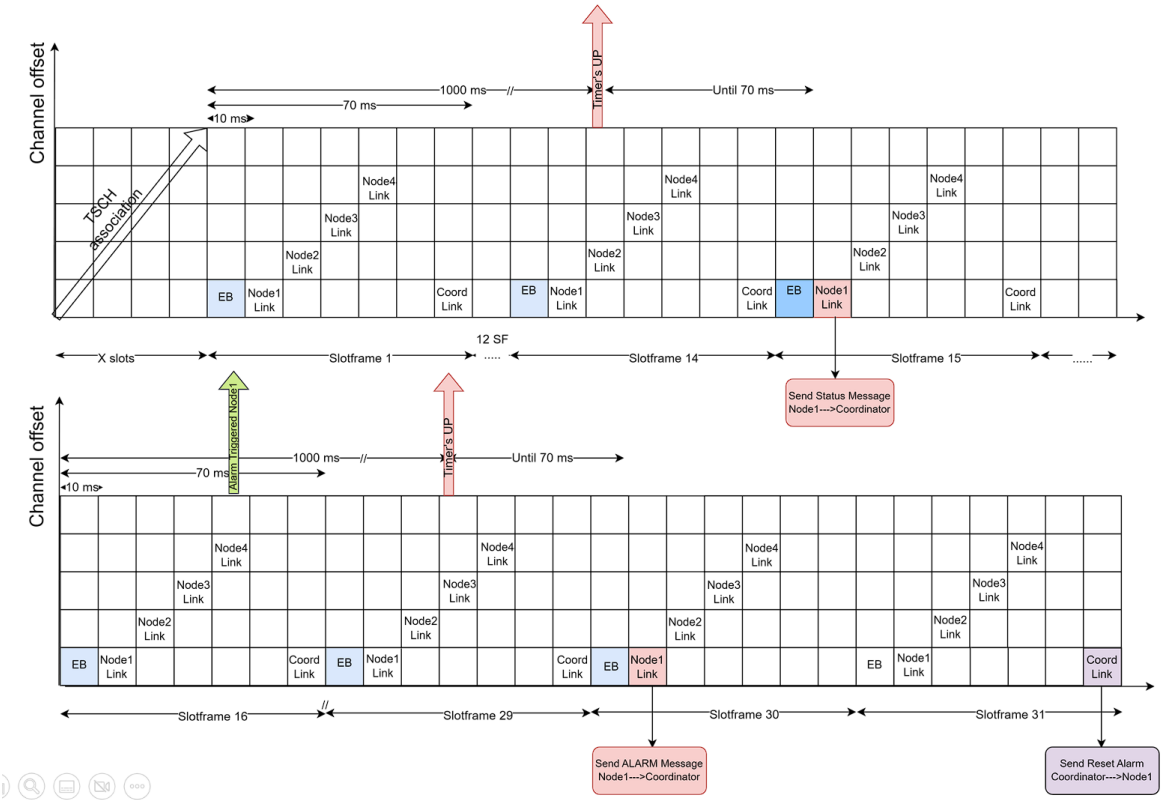

Figure 6.5: Theoretical TSCH scheduling

Figure 6.5 schematizes the schedule for the network shown in Figure 6.1 in order to understand the theory behind the intended system. It is a matrix of cells, each one indexed by a timeslot and channel offset. It also presents some possible worst case scenarios of the network whose consequences are described bellow.

The association between the coordinator and the respective nodes will take as many slots as needed and go through all the channels available for that purpose. As soon as the association is done, the communication between the devices begins in the respective slot (previously scheduled).

In this example, it is considered that the slotframe consists of seven timeslots with 10 ms each and five channel offsets available. The fact that the network is a star topology makes the coordinator the center of every communication, so shared links in the same timeslot are not a possibility because the coordinator can only listen to one node at a time. For this reason, the links are spread across the channel offsets available, each one in a different cell from the previous one. When implemented, the channel offsets used in the representation will not correspond to reality; they are only for illustration purposes. In reality, TSCH computes the communication frequency (channel) for an [ASN,Choffset] link between two nodes by following the function 4.2, where F is a lookup table that contains the sequence of the available physical channels and C is the length of the sequence. So, the horizontal lines in the scheme are not

representative of the frequency of the communication but of the channel offset that, with the ASN, will lead to the calculation of that frequency.

With that said, the first slot of each slotframe will be dedicated to the exchange of enhanced beacons, followed by links between the coordinator and the nodes. In these dedicated slots, each node will be able to share its periodic information at the proposed time (Section 6.1.3), which in the case of this example is every second. This periodicity is achieved with the help of a timer in each node that, when expired, triggers the necessity to send the status message. In Figure 6.1, it is signaled by the “Timer’s Up" arrow referring to node 1, which means that at this point, node 1 needs to send the periodic status information to the coordinator. Since the slot intended for communication between the node and the coordinator has already passed, this information can only be sent in the next slotframe, which can cause a worst-case scenario delay of up to 7*10 ms = 70 ms, the length of the slotframe multiplied by the 10 ms timeslot.

In a scenario when an alarm is triggered by an external event ("Alarm Triggered Node 1” arrow in Figure 6.1), that is, the sensor detects a possible fire, the node has to send this information to the coordinator and to do that, it was defined that it needs to wait for the timer to expire, meaning that in the next periodic message the status will be an alarm situation. For that reason, the coordinator will receive this information in slotframe number 30. The subsequent response message from the coordinator to the device that triggered the alarm, in this case node 1, will be sent in the next slotframe in a specific slot that the coordinator has to communicate specific information to a given node. In this “downlink” message, the coordinator will reset the alarm that was triggered by writing in the right register of the end-node.

The schedule scheme presented above could not be met with any of the available schedules in Contiki-NG presented in section 5.3.5.2. For this reason, a new and customized scheduling policy was implemented to fulfill the required system specifications. The steps of these implementations are presented in section 6.3.1.

6.2 Development tools

This section presents the main tools used and practical work done to implement what was introduced in the previous section.

6.2.1 Documentation and code

The Contiki documentation was the main source of information for developing the code for the nodes and coordinator. Contiki has several APIs for each supported platform, and some of the relevant drivers for the CC2650 launchpad were available under launchpad/cc2650. The port came with several practical examples demonstrating how to read sensors and how to use board peripherals, such as “base demo” and “cc26x0-web-demo”. Besides this, there are more examples of how to implement and use TSCH that were helpful not only for this part but also for the testing part.

In the folder Contiki-NG/examples, there are examples on how to use UDP, MQTT, and other applications, as well as some basic examples. An example is the “Hello-Word” folder,

which is simply to familiarize with the development platform. This program simply prints an "hello word" message in the terminal in a native manner (no board needed), and it was then possible to do the same by programming a real device.

To do this, it was necessary to clone the Contiki-NG repository from GitHub to manipulate the available code [65]. After that, GitHub was the version control tool used for the project.

6.2.2 Contiki Build System

To implement the software presented above, some steps were followed after understanding how the Contiki build system work.

The Contiki build system has multiple makefiles that make compiling the code for different platforms with different parameters easier than with other operating systems [78].The base makefile is located in the root directory of Contiki-NG/Makefile.include and, depending on the platform, other make files can be merged before compilation [79].

6.2.3 Serial Interface

Contiki has a tool to identify the IoT devices connected to the computer, called Motelist [80]. By running the command python motelist.py, the output is a table with the port and identification of the current connected devices. This way, it was possible to see the identification of the device port, so then it could be targeted for flashing and to open a serial port connection. The command "make" was constantly used to invoke the serial-dump-linux application and open the serial port on the target platform [79]. When more than one node needed to be connected, Putty was the tool used as the terminal emulator on Windows to see the output of these boards. The serial port configuration was set to {115200, 8, N,1}

6.2.4 Programing the CC2650 Launchpad

In order to program the CC2650 launchpad, it is necessary to erase the current firmware using a Texas Instruments program called UniFlash [81]. After that, programming the device through its ROM bootloader is easier. It is also possible to flash the code via JTAG using Uniflash. However, the bootloader approach was used, and the steps are described below.

Before flashing the code to the device, it is necessary to compile the system firmware. Some toolchains are required for that: Since the CC2650 device contains a 32-bit ARM Cortex-M3 processor, an ARM compiler[82] was used: arm-none-eabi-gcc from the ARM GNU toolchain 9.2.1 version [83].

To enter the bootloader mode on a TI Launchpad, it is necessary to follow the next steps:

1. connect the board to a host computer via USB cable
2. press and hold the left user button
3. press and release the reset button
4. release the left user button.

When entered into the bootloader mode, it is possible to program the device. So, to run a program on an IoT device such as the CC2650 launchpad, a target needs to be made through the following command that will flash the code to the targeted board when it is ready: *make*

TARGET=cc26x0-cc13x0 BOARD=launchpad/cc2650 PORT=<-device-usb-port> <program-name>.upload

Typing *make TARGET=cc26x0-cc13x0 BOARD=launchpad/cc2650 PORT=<-device-usb-port> login*, for example, opens up a serial connection with targeted device connected to the development machine (usually at /dev/ttyACM0).

6.2.5 Configurations

In order to implement the developed system some configurations were needed. Table 6.1 shows the main configurations that are importante for the proper functioning of the framework.

Parameter	Configuration	File	Folder	Meaning
TSCH_CONF_ASSOCIATION_POLL_FREQUENCY	1000	*tsch-conf.h and cc13xx-cc26xx-def.h*	os/net/mac/tsch and arch/cpu/cc26x0-cc13x0	The radio polling frequency (in Hz) during association process
TSCH_SCHEDULE_DEFAULT_LENGTH	7 and 10	*tsch_conf.h*	os/net/mac/tsch	Size of the slotframe (used 7 and 10 for tests)
CLOCK_CONF_SECOND	128	*arm-def.h*	arch/cpu/arm	Number of software clock ticks per second
MAKE_MAC	MAKE_MAC_TSCH	makefile	Project folder	Add TSCH support
TSCH_LOG_CONF_PER_SLOT	1	*project-conf.h*	Project folder	Enable TSCH logs per slot for debugging purposes

Table 6.1: Configuration parameters for developed framework

Some of the parameters were not altered but are important for the system characterization, such as the CLOCK_CONF_SECOND.

6.3 Developed code

In this section, a brief presentation of the code developed for the devices is made. Here, it is important to have a notion of the Contiki OS methods and process presented in Section 5.3.5.4.

With intensive understanding of the available firmware it was possible to use and manipulate functions and variables of Contiki-NG in order to develop code for the end-nodes and the coordinator, and also develop a custom TSCH schedule.

6.3.1 Custom Schedule

In order to implement the system with the specifications needed, a custom schedule was programmed as suggested in the Contiki-NG documentation [77]. In each device, cells were manually created by adding links between the nodes with the function tsch_schedule_add_link() as can be seen in the Code 1 in the end-node perspective. The specifications needed were, the slot offset, the channel offset, the option of the link (RX, TX, or Shared), and if it was broadcast or unicast (specifying the respective address).

This schedule is specified in all of the system's devices. The coordinator has all the specifications about all the nodes in its schedule, and the nodes have the links to send and receive information to and from the coordinator. For example, when the coordinator has a link for reception (RX), one node has a transmission (TX) scheduled in that same slotframe and

Code 1 TSCH shedule implementation in end-node

```
static void
initialize_tsch_schedule(void)
{
struct tsch_slotframe *sf = tsch_schedule_add_slotframe(0,
TSCH_SCHEDULE_DEFAULT_LENGTH);
uint16_t slot_offset;
uint16_t channel_offset;
// A "catch-all" cell at (0, 0)
slot_offset = 0;
channel_offset = 0;
tsch_schedule_add_link(sf,
LINK_OPTION_RX | LINK_OPTION_SHARED,
LINK_TYPE_ADVERTISING, &tsch_broadcast_address,
slot_offset, channel_offset, 1);
//slot for the end-node to communicate
slot_offset = 4;
channel_offset =3;
tsch_schedule_add_link(sf, LINK_OPTION_TX, LINK_TYPE_NORMAL, &tsch_broadcast_address,
slot_offset, channel_offset, 1);
...}
```

channel offset. As a result, it generated a schedule similar to Figure 6.5, with the coordinator having the inverse link option as the nodes to create the expected behavior.

One other important aspect was the fact that the cell for the exchange of enhanced beacons needed a specific type of link called LINK_TYPE_ADVERTISING and, typically, this cell is the first one of the slotframe, [0,0].

6.3.2 Coordinator and end-nodes code

Both codes were based on the Contiki examples in the examples/rpl-udp/ [84] folder to establish simple UDP connections and exchange information between a server and a client. This code was adapted to implement TSCH according to the tutorial: switching to TSCH of the documentation [85], and to send information through UDP API's after the needed connection initialization, as can be seen in the code below.

The code is similar and for both devices since it is used the same functions. However, the organization and objective of each one is different.

Code 2 Main thread for the coordinator

```
PROCESS_THREAD(udp_server_process, ev, data)
{
  PROCESS_BEGIN();

  /* Initialize DAG root */
  NETSTACK_ROUTING.root_start();

  /* Initialize UDP connection */
  simple_udp_register(&udp_conn, UDP_SERVER_PORT, NULL,
                      UDP_CLIENT_PORT, udp_rx_callback);

  PROCESS_END();
}
```

6.3.2.1 *Coordinator Code*

For the coordinator, the main process thread only presents what can be seen in the code above. As can be seen in Figure 6.6, after this initialization, the node will be at the yield "tsch_process" as referred to in section 5.3.5.4, until a packet is received and the callback registered in the UDP initialization is evoked.

In this callback, analysis of the message received will be done to see if an alarm was triggered. If so, the coordinator needs to notify the control panel and reset the alarm by sending a reset message to the node that communicated the alarm signal. If not, it means that it was a periodic message. If there is nothing important to communicate, an usual acknowledgement will be sent back.

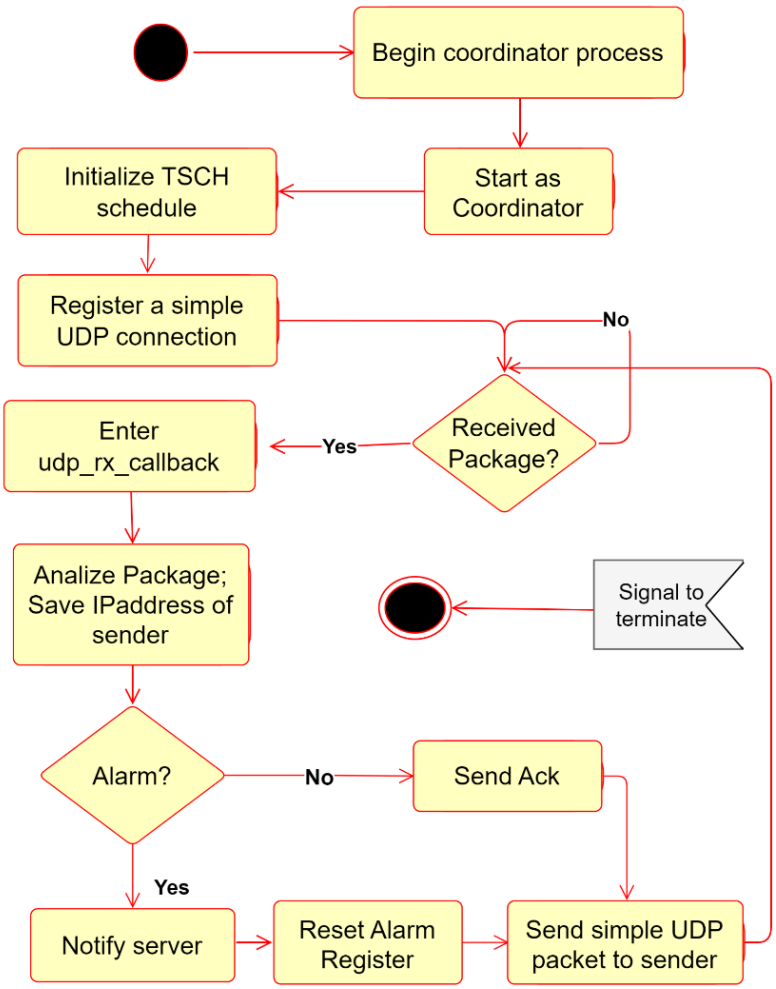

Figure 6.6: Activity diagram of coordinator code

6.3.2.2 *End-nodes code*

For the end-nodes, the main process thread is more complex, as can be seen in figure 6.7. Here, the timer needs to be set to start counting 1 second for the transmission of the periodic message. When the timer is up, it triggers an event, and then it is again reset to count the next second. As soon as the package is prepared and the tsch_shedule_slot_operation()

Code 3 Set timer Code for the end-node

```
if(etimer_expired(&periodic_timer )) {
  etimer_reset(&periodic_timer);  //reset of the timer to be periodic
  //get root ip_adress if reachable
  if(NETSTACK_ROUTING.node_is_reachable() &&  NETSTACK_ROUTING.get_root_ipaddr(&dest_ipaddr))
  {
        simple_udp_sendto(&udp_conn, disc_buff,
        sizeof(disc_buff), &dest_ipaddr);
  } else {
    LOG_INFO("Not reachable yet\n");
  }
}
```

(Section 5.3.5.4) finds a link to transmit, the UDP API for transmission is evoked, as can be seen in the code 3.

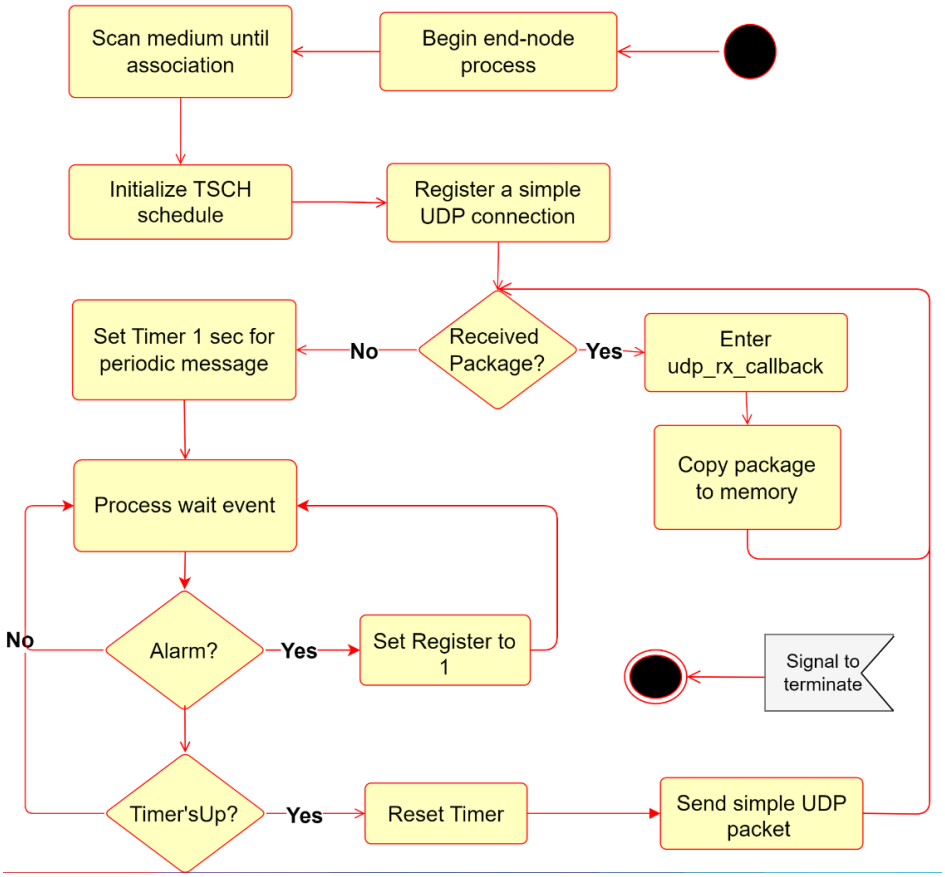

Figure 6.7: Activity diagram of end-nodes code

On the other hand, another event needs to be watched: the fire alarm event. When the board button is pressed, it will trigger an event on the thread and the respective alarm register will be set to one. From now on, the information contained in the variable "disc_buff" will already have the alarm information. Therefore, when the timer expires the periodic message that is subsequently sent already conveys the alarm signal.

Whenever there is an incoming package, process "tsch_pending_events_process" is polled from interrupt and handles the TX or RX input callbacks (Section 5.3.5.4). In

Code 4 Callback function in the end-node code

```
udp_rx_callback(struct simple_udp_connection *c,
        const uip_ipaddr_t *sender_addr,
        uint16_t sender_port,
        const uip_ipaddr_t *receiver_addr,
        uint16_t receiver_port,
        const uint8_t *data,
        uint16_t datalen)
{
  //first bit tells if is received or send
  LOG_INFO("Received response '%u' from ",*data);
  LOG_INFO_6ADDR(sender_addr);
  //Copy and parse payload to memory array
  memcpy(&disc_buff, data, sizeof(uint8_t));
  }
```

the udp_rx_callback function, presented in the code 4, the payload will be copied to the memory buffer to update the registers. For example, the alarm register is reset this way. If the coordinator sends a reset message, the register will have the value zero in this incoming message and it will be copied to the end-node memory.

Each code explained above was programmed in the respective device, in order to implement a proof of concept of a fire alarm system with wireless communication. The next part will focus on testing and validate it. To do these tests, minor changes were made to the above code.

CHAPTER 7

Tests and validation

A fire alarm system is a system that, in the event of a failure, can cause much destruction or even deaths. Therefore, such systems are strictly tested by developers, putting them through extreme conditions. BOSCH for example, has a fire laboratory prepared to perform test fires according to EN54 and UL268 standards [86]. This section focuses on verifying the wireless communication of the developed system by testing its basic functionalities and performance.

7.1 Important design choices

To test the developed system, there were some important aspects to take into consideration:

- The environment in which the tests were carried out was kept as consistent as possible between tests. The tests were made in a room with few people, so the possible interferences in the same frequency band as the system (2.4 GHz) were few but existent. There were 3 computers and 2 cell phones with Wi-Fi access.
- The transmit power is set in the ieee-mode.c file as the maximum possible value for the CC2650 launchpad (5 dBm) which is an acceptable power for low power systems of this type.
- The test platform consists of two application binaries that are flashed on the respective devices (coordinator and end nodes).
- The implemented custom schedule always presents the first slot dedicated for enhanced beacons advertisement.
- As can be seen the configurations shown in table 6.1, throughout the test the length of the slotframe was changed from 7 to 10 to see the impact in the end to end latency. With a slotframe size of 10, the system retains all its functionality as expected, ideally improving only the performance of the system. Thus, the results of the initial verifications, that were performed with a slotframe size of 7 are not invalidated because of this.
- The size of the payloads transmitted between the devices in validation was always less than 4 bytes and during the performance tests was always less than 8 bytes.

- The distance between the devices in the initial tests always remained short (< 1 meter). Only when the goal was to test the range of the system, the nodes were moved away from the coordinator. The normal setup is shown in Figure 7.1.

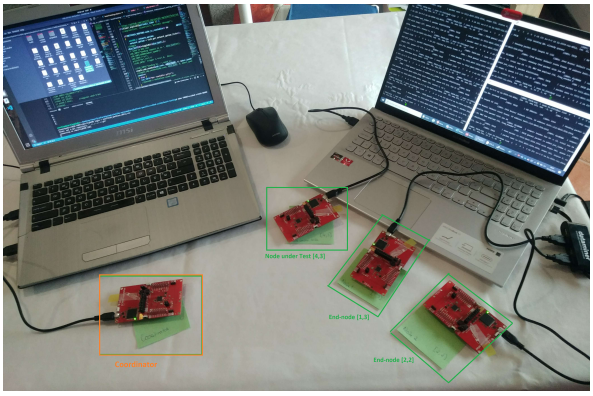

Figure 7.1: Tests setup

7.2 Basic functionalities tests

Since the beginning of the development of the system, each development step was tested in order to evaluate its correctness, prior to move to the next one. The implementation of the system had its core work done over a previously unknown operating system, therefore the initial tests aimed to get familiar with Contiki and the board functionalities. One example of this was already mentioned in the previous section, the hello-world test. These and other basic tests were done, but the description and presentation of all of them is not possible due to the lack of space. For this reason, only the more relevant tests will be presented in this section.

7.2.1 Test 1- TSCH communication in the respective slot

In order to validate if the devices were communicating through TSCH and in the respective timeslot, the following test was conducted:

- Configure the schedule in the coordinator to advertise EBs in the cell [0,0], to listen in the slot that the node was transmitting [4,3], and to have a slot for transmissions from the coordinator to the end-nodes [5,0].
- Configure schedule in one end-node to have a "catch-all" cell at [0, 0], transmit in the slot [4,3], and listen in the slot [5,0].
- Configure each node to send a periodic message every 1 second.

This test was performed individually for each device, with different transmission cells for each one. Then, all the nodes were connected to see if each one could transmit in the respective cell as expected.

7.2.2 Test 2- Alarm trigger test

After the basic communication validation, the response of an external trigger (alarm) was put to test:

- Configure the end nodes to write in a register if one specific button on the board was pressed. This button is simulating the digital output of a sensor (for example a smoke or heat sensor).
- If the button was pressed, the alarm information is passed to the memory so that it can be send in the next periodic message.

7.2.3 Test 3- Coverage test

The objective of this test was mainly to see, in a nondeterministic way, how far the nodes were still able to connect and exchange information.

- The end-nodes were moved away from the coordinator by a distance of 5 meters with no walls interfering but with people passing by and using the test space.
- The end-nodes were moved away from the coordinator by a distance of 7 meters with a wall in between.

7.3 Performance tests

This part of the tests validates the timing and performance of the communication between the devices in the system.

7.3.1 Test 4 - End-to-end latency

The end-to-end latency is the time taken for a packet to travel from one node to another node. One main feature of TSCH is that all nodes are synchronized to a global time counter maintained by the coordinator. This fact can be exploited to measure properties such as latency.

Contiki has a macro that defines a second, measured in system clock time, CLOCK_SECOND = 128 ticks. So, each "real" second is 128 ticks (system clock time). Besides this, Contiki has several API's that measure the time the system is up, in ticks, that were helpful in these measurements. One of them is tsch_get_network_uptime_ticks(). This test was based on the Contiki example *6tisch/timesync-demo* [87].

- One end-node was programmed to periodically send its detected network uptime to the coordinator.
- When the coordinator receives these packets, it prints the time difference and the uptime of its local network. This time difference is equal to the end-to-end latency, which accounts for the time it takes to prepare a packet, wait for an appropriate time slot in the TSCH schedule, and for the time spent transmitting data over the air (negligible). The packet also contains the retransmission time if it is not received on the first try.
- These values were collected from the terminal and passed to Matlab to have a better understanding of the results.

After the validation with one node, the test was repeated with the other nodes connected to the network. A similar test was also conducted but with the end-nodes far away from the coordinator (approx. five meters).

7.3.2 Test 5 - Interarrival time test

This test has the objective of checking the amount of time that elapses after the reception of a packet until the next packet arrives. This calculation is done at the level of the coordinator when it receives a package sent by the specific node under test. It is expected to be similar to the period of the periodic messages sent by the end node under test to the coordinator.

- When a packet is received by the coordinator, the current network uptime is saved in memory and it is then subtracted from the network uptime value when the next reception occurs.

This test was also conducted with multiple nodes connected to the network. It was also tested when the end-devices were far from the coordinator (approx. five meters).

7.3.3 Test 6 - Packet Loss test

This test aimed to see the reliability of the implemented solution by evaluating the packet loss rate. Ideally, this type of test would benefit from having multiple sources of possible interference to evaluate the capacity of the system in these conditions. However, those conditions were not possible, so the test conducted was the following:

- In the end-node code, it was implemented a counter that was incremented each time a packet was sent. This information was the sent to the coordinator in each tested packet.
- In the coordinator, the number of received packages from the node in evaluation were also verified through a counter. The information was then compared and, if the number of packages sent from the end-node were the same as the number of receive in the coordinator, a zero would be printed. One the other hand, if this number would be different it would count as 1 packet loss.

The same test was conducted with other nodes connected to the coordinator.

7.4 Results and analysis

In this section will be presented the results as well as the analysis of the tests referred to in sections 7.2 and 7.3. The results presented are for the case when only one node is connected to the coordinator. However, as said before, the tests were also conducted with multiple nodes in the network, and the difference in the results was not significant. Therefore, the conclusions drawn from these tests can also be applied to the other tests, but their results will not be presented to avoid repetition.

7.4.1 Results Test 1- TSCH communication in the respective slot

As soon as the end-node is associated with the coordinator, the status of the TSCH network in the perspective end-node is presented in Figure 7.2. Here, the coordinator is identified by its IP address as the time source.

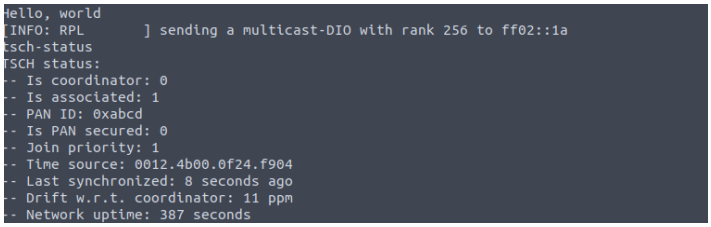

Figure 7.2: TSCH network status in a end-node perspective

When the schedule is programmed into both devices, the way the packet exchange information is displayed on the terminal is as shown in Figure 7.3.

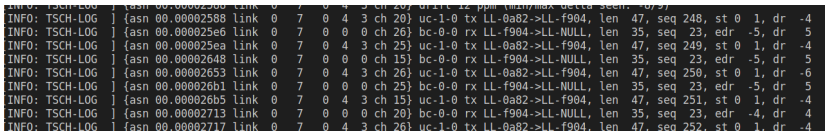

Figure 7.3: TSCH logs in a end-node perspective

The ASN (Absolute Slot Number) comes first. Next, some details regarding the TSCH link time offset, slotframe handle, slotframe length, time offset, and channel offset by order. The physical channel is last. Broadcast/unicast is referred to as bc/uc. The following bit shows if this was an EB or a data packet. The following integer represents the security level (0 means no security). Then, RX/tx indicates if it is a reception (RX) or a transmission (tx). Then, the source and destination addresses for this packet appear in a condensed format. The packet's size and MAC sequence number are then given. Last but not least, for RX packets dr denotes the drift correction applied during reception, measured in rtimer ticks, and edr is the opposite. On the other hand, for tx packets, st indicates the status of the transmission. The first integer is the MAC status, and 0 indicates success while 2 indicates no-ack. And the dr in Tx packages shows the drift correction applied by the node after receiving an ACK from its time source.

Figure 7.3 is from the perspective of the end node under test, and the time offset and channel offset have the values of 4 and 3, as it is supposed. This proves that the communication is happening at the expected time slot. Here, it can also be seen that the catch-all cell is used for transmission EBs as imposed by TSCH. Another relevant information shown in Figure 7.3 is that each exchange of information is made in a different channel, meaning different frequencies, as expected from TSCH.

This test was one of the first with the full TSCH communication,and with the respective schedule. For this reason, the test had a duration of approximately 1 hour to certificate the veracity of the communications. Then more devices were schedule to communicate with the coordinator to see if the behaviours would be maintained, and it did.

7.4.2 Results Test 2- Alarm test

Figure 7.4 shows that when the bottom is pressed (Key Left pressed), this information is passed to the memory by setting the respective register to 1, when it is printed "press event".

After that, the information from the alarm is put in the TSCH queue and sent when the timer of 1 second is up, in the periodic message.

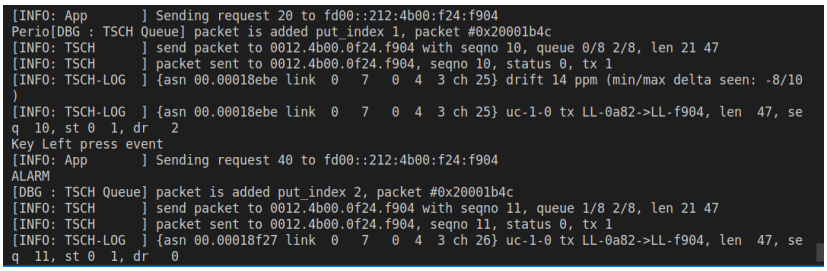

Figure 7.4: TSCH logs in the event of an alarm

This test was repeated multiple times by pressing the button in random occasions and watching if the information was received by the coordinator.

7.4.3 Results Test 3- Coverage test

This test was made with the end-nodes already associated with the coordinator. Having all nodes "online", the end-nodes were slowly moved away from the coordinator until the message displayed in the terminal of each node presented the log shown in Figure 7.5.

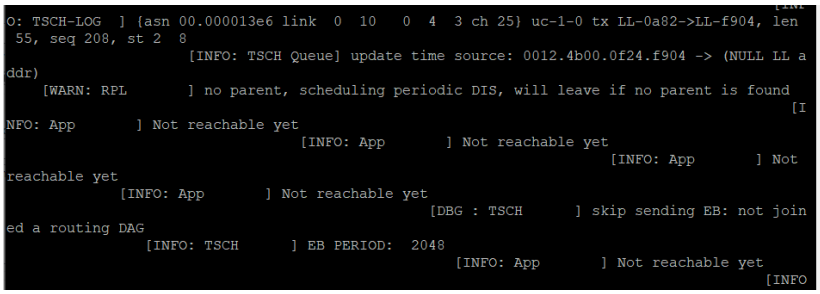

Figure 7.5: TSCH logs when connection is lost from the end-node perspective

The experiment showed that within a distance of up to five meters, communication existed without any problems. Then the nodes were moved to another room with walls in between which caused the network nodes to disconnect and were not able to reach the coordinator again.

This test gave an idea of what the coverage capability was for the communications between the devices in the implemented system. Until five meters apart, these nodes can communicate without problems. After that, the connection is not guaranteed, especially if there are walls in between. Given the conditions in which these tests were conducted, the values are not precise.

It was already expected that the coverage would not be very large because the technology is LR-WPAN. However, for the type of fire alarm systems intended, this value is far below what is desirable.

7.4.4 Results Test4- End-to-end latency test

As can be seen in the previous tests, the slotframe length was 7, so the end-to-end latency was initially tested in these conditions.

The results in these conditions are presented in Figure 7.6. They were taken in the coordinator device that is set to calculate and print the latency each time it receives a message. So, it can be said that every second a sample was taken for approximately 6 hours.

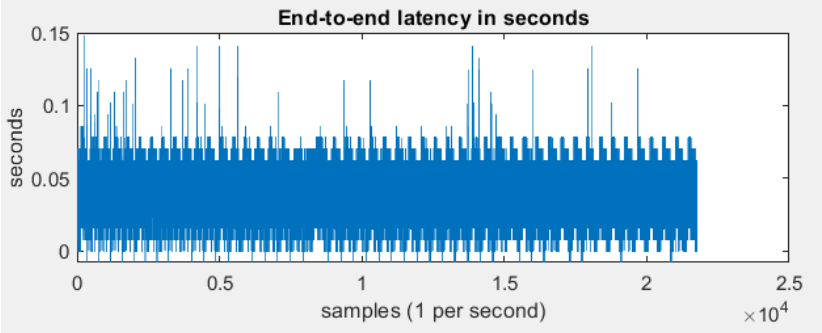

Figure 7.6: End-to-end latency with a slotframe length of 7

With a slotframe size of 7 and time slots of 10 ms, each slotframe will have a duration of 70 ms. As a result, if a message has to be sent every 1 second, it will not always happen in the correct slot, as is the case of figure 6.5 in section 6.1.4. Here, when the timer is armed, it starts counting the seconds and will end after 1000ms. If the slotframe size is 70 ms, when the timer expires, it will not always land on the timeslot of the node that armed it and wants to send the information as fast as possible when the timer expires. If the timer expires moments after the transmit slot, there will be an additional delay of, in the worst case, 70 ms until a transmit slot appears again in the next slotframe. This happens because there is a mismatch due to a lack of synchronization between the sending of the message and the start of the respective timeslot. From this, it was realized that it was possible to optimize the implemented solution so that the transmission happens as soon as possible, reducing and ideally eliminating the waiting time for the appropriate time slot in the TSCH schedule.

A simple way to try to solve this problem is to change the size of the slotframe to a multiple of 1 second (the timer for periodic information delivery). In this case, the ideal was 10 to avoid increasing the size of the slotframe too much. In this manner, an alarm set in a specific slot and slotframe can be responded to in the same slot 10 slotframes later (10*100 ms=1000 ms).

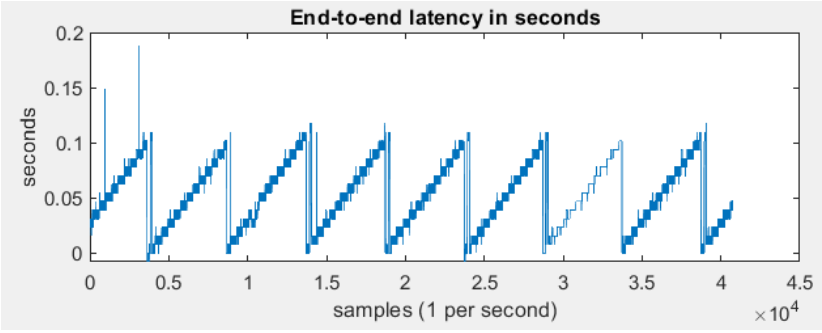

Figure 7.7: End-to-end latency with a slotframe length of 10

This test was left running for 12 hours to validate the behavior of the system over an

extended time. The result, presented in Figure 7.7, now provides a more consistent and predictable response, which is important in real-time systems. This behavior confirms what was previously stated, that there is a sync between the time of message transmission and the start of the appropriate timeslot. However, this result presents an unexpected periodic degradation of the latency. One possible reason can be other types of temporal influences that are being accounted for in the end-to-end latency and will eventually deviate the moment of transmission from the expected slot. Packet preparation values and even the time spent transmitting data over the air may accumulate.

At some point, the end-to-end latency reaches the worst case, which is 100 ms due to slotframe expansion, and goes back to the expected timeslot for transmission (approx. 0 end-to-end latency).

Either way, the average end-to-end latency is very acceptable for the system in general since it is in the range of 0.05 s and never goes higher than 0.2 s.

However the system implemented is small compared to what is expected for fire alarm system (respond to at least 240 devices in 5 seconds and other requirements). TSCH is a time slotted solution, witch means that these values of end-to-end latency will increase if the number of devices in the network also increase. More reflections on this will be made bellow.

7.4.5 Results Test5- Interarrival time test

The interarrival time for the case of a slotframe length of 7 is presented in Figure 7.8. These values are to be calculated in the coordinator while taking into account the current and prior information to the receipt of a package from a specific node under test.

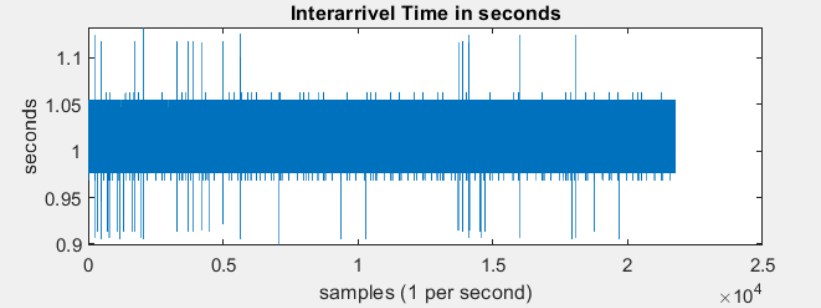

Figure 7.8: Interarrival time with a slotframe length of 7

As previously stated, it was anticipated that the time of reception by the coordinator (of transmissions from the node under test) would be similar to the time it takes to send the messages from the node under test. In these conditions, it is possible to see that the value is approaching 1 second with some margin of error, causing the Interarrivel Time in seconds to be 1,01.

Assuming the slotframe size is 10, the margin of error is reduced and the value is centered on 1 second, as shown in Figure 7.9. The existence of periodic peaks corresponds to the point at which end-to-end latency reaches its worst case and returns to a value close to zero.

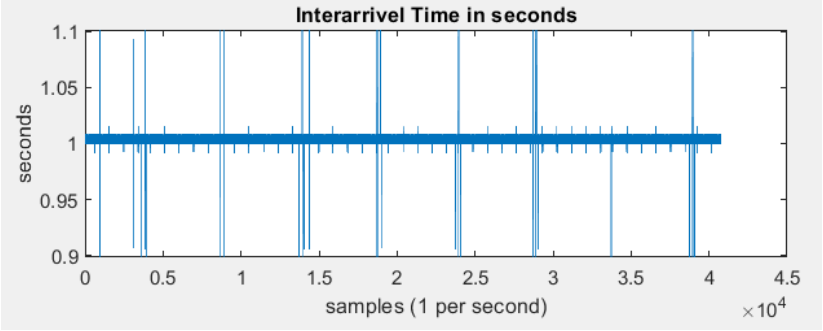

Figure 7.9: Interarrival time with a slotframe length of 10

7.4.6 Results Test 6- Packet Loss test

The test conducted in this part showed that no package was lost in the course of this realization. Even when adding more nodes to the network and putting them far away from the coordinator (approx. five meters) the results were the same. The average packet loss of the developed system is zero.

7.4.7 Final result analysis

The project's goal from the start was to investigate wireless communications that could be used to implement fire alarm systems. The tests carried out bring this exploration and understanding with the implementation of TSCH in a small laboratory testing system. From here, it was possible to see that the obtained results were very positive, and were within the scope of what was expected from the technology.

The results confirmed that nodes effectively communicate in their respective slots allocated in the schedule, propagating information efficiently to the destination.

Although it was not possible to test the system under extreme conditions of interference, it is possible to understand the robustness of TSCH, mainly due to the frequency hopping feature. This was witnessed during the tests performed since there was no packet loss and it was proven that the packet exchange always happened on different channels, i.e., different frequencies.

With end-to-end latency tests it was possible to conclude that the synchronization of sending messages with the respective slots that are intended for transmission has a significant impact on latency. This is because, just by increasing the number of slotframes in order to achieve this sincronization the latency had different but more consistent behavior.

These tests also confirmed that the larger the slotframe size, the higher the latency value will be in its worst case.

In the proof of concept, with a slotframe length of 10, it was possible to allocate 8 end-nodes, because one slot was dedicated to the exchange of EBs and another slot was defined for coordinator specific transmissions. This means that with TSCH, the way the schedule was made, it is possible to poll 8 devices in 100 ms.

Although these results allowed us to realize that the system was working well, it was desirable that some more tests be conducted to fully understand the implementation behavior.

It is necessary to clarify the response obtained in the latency tests that were not the expected ones. More tests would allow to find out the reason for the periodic lag and investigate whether it was due to some asynchronism of the operating system clocks, for example.

As mentioned before, it would also be ideal to do more exhaustive tests on the system, subjecting it to scenarios with more interference to see how it responds.

If the future goal is to scale up the system there are some considerations that the tests developed in this project brought to attention:

To support the 240 devices referred by the norms for fire alarm systems, the slotframe size must be larger than this value. This is because, since there is only one central coordinator with only one radio, the communication with the end-nodes has to be done one at a time. This way it is not possible to take advantage of the shared slots feature of the TSCH. Therefore, the schedule implemented reflected this and forced the slotframe size to have the minimum value equal to the number of devices on the network.

With a slotframe size of 250 (245+5 slots for other operations) the sweep of all devices is done in 250*10 ms, or 2.5 seconds. This value is already quite distant from the one studied in the development of 100 ms. The impact is such that the periodic message established for testing purposes is not a reality anymore, since in 1 second it was not possible to scan all the detectors.

This is a limitation of the implementation that can be overcome if it is possible for the coordinator to have more than one radio or to have multiple coordinators cooperating in order to take advantage of the TSCH feature and listen to more than one end-node in each slot.

The analysis of the results of the implemented system tests allowed to open more doors for further study of this technology for fire alarm systems through its understanding and evaluation of its limitations.

CHAPTER 8

Conclusions and Future Work

A final overview and reflection of the work done in this project is done in this section. Then, some bullet points on possible future work are presented.

8.1 Conclusions

The new trend in the fire alarm industry focuses on taking advantage of the advancement of wireless communications for real-time applications. Taking into account the benefits coming from wireless solutions, more and more companies are looking to upgrade the abundant wired systems in the market.

This work started by studying the existing types of wireless communications and which ones could be used in a system with strict time and energy requirements. The next step was to proceed to the implementation of a proof of concept system to assess the basic functionality.

After the choice of IEEE 802.15.4e with TSCH MAC as the technology, it was necessary to look for development platforms that would support it as well as its functionalities.

Much of the time invested in this project was spent on learning and adapting to the chosen development platform. The complexity of the operating system used (Contiki- NG), and the way the TSCH is aggregated there, brought an added difficulty to the development of the project. Besides this, there was an initial difficulty in integrating the firmware into Texas Instrument's CC2650 Launchpad.

With these difficulties overcome, the proof-of-concept system was developed with some requirements in mind:

- Alarm Transmission Time: the maximum time allowed between the detection of a fire and signalization at the control panel is 10 s. Since Siemens' SWING system declared that its wireless communication happened in 5 s, this became the time objective to be reached in the transmissions.
- Monitoring: missing or malfunctioning devices have to be reported to the user at the control panel within at most 300 s after the error occurs.

Subsequently, tests were carried out on the system to see if the chosen technology was actually a good option for fire alarm systems with the above requirements.

The results have been positive for the simple 4 device system that was implemented for proof of concept. The latency and interarray time values were within the expected values, which indicates that the basic mechanism associated with scheduling and data transmission are correctly implemented. Nevertheless, some tests on the system's scalability were still missing. That is, in the future, it will be necessary to implement this framework in a full scale fire alarm system, which, according to the standards, can have up to 240 devices. This step shall require further development, both at the scheduling and topology levels.

The tests also revealed that the range allowed by the technology was disappointing, even though a short range was already expected due to the LR-WPAN technology. However, this can be enhanced as can be seen in the future work.

The development of this project allowed not only to recognize that IEEE 802.15.4 with TSCH is a good option for wireless communications for next generation fire alarm systems, but also to understand what kind of adaptations are required for an application with a large number of devices.

8.2 Future Work

In the scope of this project were carried out the first steps towards the study of wireless communication technologies for fire alarm systems, resulting in a possible implementation of a full alarm system. There are some obvious future work that can be made to enhance the developed system:

- Continue studying the potential of TSCH with more intensive tests to the system.
- Support fire related sensors: In this project, it was used a button to simulate the signal of a discrete sensor. Fire alarm system can have a huge variety of analog and discrete sensors such as: smoke, heat, CO2, etc.
- Add a user-friendly interface to monitor the system status.
- Implement mesh network to increase coverage: Contiki brings Orchestra schedule that proves to be efficient in TSCH mesh and dynamic networks[88].
- Scale the system to support 240 devices
- Use more than on radio in the coordinator or have multiple coordinators cooperating
- Enhance the security of the communications. Contiki has the option of AES128 encryption [89] and link layer security options that can be explored [90].

The developed system served as a proof of concept for a possible future development of a complete, robust and reliable fire alarm system using the platforms and technologies presented here.

References

[1] M. N. Hassan Reza, C. Agamudai Nambi Malarvizhi, S. Jayashree, and M. Mohiuddin, "Industry 4.0–technological revolution and sustainable firm performance," in *2021 Emerging Trends in Industry 4.0 (ETI 4.0)*, 2021, pp. 1–6. DOI: 10.1109/ETI4.051663.2021.9619363.

[2] S. Ben Yaala and R. Bouallegue, "On mac layer protocols towards internet of things: From ieee802.15.4 to ieee802.15.4e," in *2016 24th International Conference on Software, Telecommunications and Computer Networks (SoftCOM)*, 2016, pp. 1–5. DOI: 10.1109/SOFTCOM.2016.7772165.

[3] J. V. Pérez, "Diseño y evaluación de mecanismos de optimización en redes de sensores inalámbricas industriales," Ph.D. dissertation, Universitat Politècnica de València, 2021.

[4] M. Majid, S. Habib, A. R. Javed, *et al.*, "Applications of wireless sensor networks and internet of things frameworks in the industry revolution 4.0: A systematic literature review," *Sensors*, vol. 22, no. 6, 2022, ISSN: 1424-8220. DOI: 10.3390/s22062087. [Online]. Available: https://www.mdpi.com/1424-8220/22/6/2087.

[5] *A brief history of fire alarm systems - eps security*, Accessed: 2022-9-3. [Online]. Available: https://www.epssecurity.com/news/eps-news/a-brief-history-of-fire-alarm-systems/.

[6] *Our place in history | nkm fire protection*, Accessed: 2022-9-3. [Online]. Available: http://www.nkmfireprotection.co.uk/about-us/our-history.

[7] S. Bryant and I. Preston, "Focus on trends in fires and fire-related fatalities," 2017. [Online]. Available: http://webarchive.nationalarchives.gov.uk/20120919132719/http:/www.communities.gov.uk/publications/fire/researchbulletinno9.

[8] *Bs en 54: Fire detection and alarm systems*, Accessed: 2022-1-17. [Online]. Available: https://www.fia.uk.com/resources/british-standards/bs-en-54-series-fire-detection-alarm-systems.html.

[9] M. Blagojević, R. Jevtić, and D. Ristić, "Comparative analysis of rules in five leading standards for smoke detectors siting in the presence of a ceiling irregularity," *Transactions of the VSB-Technical university of Ostrava Safety Engineering Series*, vol. XII, 2 2017. DOI: 10.1515/tvsbses-2017-0011.

[10] "Wireless fire alarms en54 part 25 compliant - fire systems ltd," Accessed: 2021-12-27. [Online]. Available: https://firesystemsltd.co.uk/wireless-fire-alarm-systems/en54-part-25.

[11] S. K. S. Hafner, *Viewer • smart information delivery (sid)*, Jul. 2017. [Online]. Available: https://sid.siemens.com/v/u/A6V11218603.

[12] "Fire alarm systems," 1992, Accessed: 2022-10-3. [Online]. Available: www.boschsecurity.com.

[13] *Wi-fyre wireless fire detection eurofyre*, Accessed: 2022-10-5. [Online]. Available: https://eurofyre.co.uk/systems/wireless-fire-detection/wi-fyre-wireless-fire-detection/.

[14] *Wi-fyre xenos wireless fire detection eurofyre*, Accessed: 2022-10-5. [Online]. Available: https://eurofyre.co.uk/systems/wireless-fire-detection/wi-fyre-xenos-wireless-fire-detection/.

[15] A. El-Hoiydi and J.-D. Decotignie, "Wisemac: An ultra low power mac protocol for multi-hop wireless sensor networks," in *Algorithmic Aspects of Wireless Sensor Networks*, S. E. Nikoletseas and J. D. P. Rolim, Eds., Berlin, Heidelberg: Springer Berlin Heidelberg, 2004, pp. 18–31, ISBN: 978-3-540-27820-7.

[16] ©. S. Switzerland, "Planning tool maximum protection with easy-to-plan technology," 2012, Accessed: 2021-12-8. [Online]. Available: www.siemens.com/swing.

[17] *Hard wired vs wireless fire alarm systems — news*, Jul. 2018. [Online]. Available: https://www.vermontlifesafety.com/new-blog/2018/7/13/cviaexysnby0km8ef416503g4du2rr.

[18] *Addressable or conventional fire alarm system? | protec fire and security group ltd*, Jul. 2021. [Online]. Available: https://www.protec.co.uk/latest-news/addressable-or-conventional-fire-alarm-system/.

[19] *What's the best technology? | fire monitoring*, Jun. 2020. [Online]. Available: https://www.midwestalarmservices.com/our-company/blog/fire-monitoring-whats-best-technology.

[20] *Communications methods for fire alarm | national training center*. [Online]. Available: https://nationaltrainingcenter.com/communications-methods-for-fire-alarm/.

[21] *Nfpa 72®: National fire alarm and signaling code®*. [Online]. Available: https://www.nfpa.org/codes-and-standards/all-codes-and-standards/list-of-codes-and-standards/detail?code=72.

[22] "Local securitynetwork we know what it's all about," [Online]. Available: https://resources-boschsecurity-cdn.azureedge.net/public/documents/Detectors_Commercial_Brochure_enUS_9007201918053771.pdf.

[23] *Wireless concepts*, https://www.emerson.com/documents/automation/training-wireless-topologies-en-41144.pdf.

[24] *Specifications | bluetooth technology website*. [Online]. Available: https://www.bluetooth.com/specifications/.

[25] J. H. Kurunathan, "Improving qos for ieee 802.15.4e dsme networks phdthesis," Accessed: 2021-11-22.

[26] *Lora documentation*, https://lora.readthedocs.io/en/latest/.

[27] A. Nikoukar, S. Raza, A. Poole, M. Güneş, and B. Dezfouli, "Low-power wireless for the internet of things: Standards and applications," *IEEE Access*, vol. 6, pp. 67 893–67 926, 2018. DOI: 10.1109/ACCESS.2018.2879189.

[28] *What are lora and lorawan*, https://lora-developers.semtech.com/documentation/tech-papers-and-guides/lora-and-lorawan/, Accessed: 2021-11-3.

[29] *Bluetooth long range mode: How to achieve ranges over 1 km*, https://novelbits.io/bluetooth-long-range-how-to-achieve-results-over-1km/, Accessed: 2022-10-3.

[30] G. Patti, L. Leonardi, and L. Lo Bello, "A bluetooth low energy real-time protocol for industrial wireless mesh networks," in *IECON 2016 - 42nd Annual Conference of the IEEE Industrial Electronics Society*, 2016, pp. 4627–4632. DOI: 10.1109/IECON.2016.7793093.

[31] J. Adams, "An introduction to ieee std 802.15.4," in *2006 IEEE Aerospace Conference*, 2006. DOI: 10.1109/AERO.2006.1655947.

[32] A. Cunha, A. Koubaa, R. Severino, and M. Alves, "Open-zb: An open-source implementation of the ieee 802.15.4/zigbee protocol stack on tinyos," Oct. 2007, pp. 1–12. DOI: 10.1109/MOBHOC.2007.4428602.

[33] B. R. Ines Khoufi Pascale Minet, *Beacon advertising in an ieee 802.15.4e tsch network for space launch vehicles. eucass 2017 - 7th european conference for aeronautics and aerospacesciences*, Milano, Italy. ffhal-01636659ff, 2017.

[34] S. Raza, "Medium access control protocols for reliable communication in low-power industrial applications," Ph.D. dissertation, Otto-von-Guericke-Universität Magdeburg, 2019.

[35] P. A. M. Devan, F. A. Hussin, R. Ibrahim, K. Bingi, and F. A. Khanday, "A survey on the application of wirelesshart for industrial process monitoring and control," *Sensors*, vol. 21, no. 15, 2021. DOI: 10.3390/s21154951. [Online]. Available: https://www.mdpi.com/1424-8220/21/15/4951.

[36] S. Brahmi, M. Yazid, and M. Omar, "Multiuser access via ofdma technology in high density ieee 802.11ax wlans: A survey," in *2020 Second International Conference on Embedded Distributed Systems (EDiS)*, 2020. DOI: 10.1109/EDiS49545.2020.9296440.

[37] L. P. Fraile, S. Tsampas, G. Mylonas, and D. Amaxilatis, "A comparative study of lora and ieee 802.15.4-based iot deployments inside school buildings," *IEEE Access*, vol. 8, pp. 160 957–160 981, 2020. DOI: 10.1109/ACCESS.2020.3020685.

[38] "Ieee sa - ieee 802.15.4-2003," Accessed: 2021-12-3, 2003. [Online]. Available: https://standards.ieee.org/ieee/802.15.4/3388/.

[39] "What's the difference between ieee 802.15.4 and zigbee wireless? | electronic design," Accessed: 2021-11-3. [Online]. Available: https://www.electronicdesign.com/technologies/wireless/article/21796046/whats-the-difference-between-ieee-802154-and-zigbee-wireless.

[40] F. Kauer, M. Köstler, and V. Turau, "Reliable wireless multi-hop networks with decentralized slot management: An analysis of ieee 802.15.4 dsme," Jun. 2018. [Online]. Available: http://arxiv.org/abs/1806.10521.

[41] N. Salman, I. Rasool, and A. H. Kemp, "Overview of the ieee 802.15.4 standards family for low rate wireless personal area networks," *Proceedings of the 2010 7th International Symposium on Wireless Communication Systems, ISWCS'10*, pp. 701–705, 2010. DOI: 10.1109/ISWCS.2010.5624516.

[42] E. Toscano and L. L. Bello, "Comparative assessments of ieee 802.15.4/zigbee and 6lowpan for low-power industrial wsns in realistic scenarios," *IEEE International Workshop on Factory Communication Systems - Proceedings, WFCS*, pp. 115–124, 2012. DOI: 10.1109/WFCS.2012.6242553.

[43] A. Elsts, J. Pope, X. Fafoutis, R. Piechocki, and G. Oikonomou, "Instant: A tsch schedule for data collection from mobile nodes," [Online]. Available: https://tinyurl.com/ybkyh89v..

[44] J. V. Pérez, "Diseño y evaluación de mecanismos de optimización en redes de sensores inalámbricas industriales," p. 1, 2021. [Online]. Available: https://dialnet.unirioja.es/servlet/tesis?codigo=305902&info=resumen&idioma=SPA%20https://dialnet.unirioja.es/servlet/tesis?codigo=305902.

[45] S. Biswas and A. Chandra, "Df versus af: Energy consumption comparison for ieee 802.15.4 networks," *2014 6th International Conference on Communication Systems and Networks, COMSNETS 2014*, 2014. DOI: 10.1109/COMSNETS.2014.6734904.

[46] D. D. Guglielmo, G. Anastasi, and A. Seghetti, "From ieee 802.15.4 to ieee 802.15.4e: A step towards the internet of things," *Advances in Intelligent Systems and Computing*, vol. 260, pp. 135–152, 2014, ISSN: 21945357. DOI: 10.1007/978-3-319-03992-3_10.

[47] H. Kurunathan, R. Severino, A. Koubâa, E. Tovar, and A. Koubaa, "Ieee 802.15.4e in a nutshell: Survey and performance evaluation," [Online]. Available: www.cister.isep.ipp.pthttp://www.cister.isep.ipp.pt.

[48] G. Alderisi, G. Patti, O. Mirabella, and L. L. Bello, "Simulative assessments of the ieee 802.15.4e dsme and tsch in realistic process automation scenarios," *Proceeding - 2015 IEEE International Conference on Industrial Informatics, INDIN 2015*, pp. 948–955, Sep. 2015. DOI: 10.1109/INDIN.2015.7281863.

[49] S. Duquennoy, A. Elsts, B. A. Nahas, and G. Oikonomo, "Tsch and 6tisch for contiki: Challenges, design and evaluation," *Proceedings - 2017 13th International Conference on Distributed Computing in Sensor Systems, DCOSS 2017*, vol. 2018-January, pp. 11–18, Jan. 2018. DOI: 10.1109/DCOSS.2017.29.

[50] X. Vilajosana, S. Member, T. Watteyne, *et al.*, "Ietf 6tisch: A tutorial,"

[51] F. E. Elahi, "Concurrent internet of things protocol stacks," [Online]. Available: http://www.teknat.uu.se/student.

[52] "Radio features required for tsch," [Online]. Available: https://github.com/contiki-ng/contiki-ng/blob/develop/doc/programming/TSCH-and-6TiSCH.md#porting-tsch-to-a-new-platform.

[53] "Texas instruments simplelink™ microcontrollers (mcus)," Accessed: 2021-12-9. [Online]. Available: https://pt.mouser.com/new/texas-instruments/ti-simplelink-solutions/.

[54] "Nrf 802.15.4 radio driver — nrfxlib 2.1.99 documentation," Accessed: 2021-12-3. [Online]. Available: https://developer.nordicsemi.com/nRF_Connect_SDK/doc/latest/nrfxlib/nrf_802154/README.html.

[55] "Nodric semiconductor- nrf ieee 802.15.4 radio driver," Accessed: 2021-12-3. [Online]. Available: https://infocenter.nordicsemi.com/index.jsp?topic=%5C%2Fstruct_drivers%5C%2Fstruct%5C%2Fradio_driver_latest.html.

[56] "Ieee 802.15.4 device with support for tsch -nordic q&a - nordic devzone - nordic devzone," Accessed: 2021-12-3. [Online]. Available: https://devzone.nordicsemi.com/f/nordic-q-a/82725/ieee-802-15-4-device-with-support-for-tsch.

[57] "Github - contiki-os/contiki: The official git repository for contiki, the open source os for the internet of things," Accessed: 2022-10-3. [Online]. Available: https://github.com/contiki-os/contiki.

[58] "Platform simplelink · contiki-ng/contiki-ng wiki · github," Accessed: 2021-12-3. [Online]. Available: https://github.com/contiki-ng/contiki-ng/wiki/Platform-simplelink.

[59] F. Javed, M. K. Afzal, M. Sharif, and B.-S. Kim, "Internet of things (iot) operating systems support, networking technologies, applications, and challenges: A comparative review," *IEEE Communications Surveys & Tutorials*, vol. 20, no. 3, pp. 2062–2100, 2018. DOI: 10.1109/COMST.2018.2817685.

[60] "Github - tinyos/tinyos-main: Main development repository for tinyos (an os for embedded, wireless devices).," Accessed: 2021-12-10. [Online]. Available: https://github.com/tinyos/tinyos-main.

[61] "Openwsn - confluence," Accessed: 2021-12-15. [Online]. Available: https://openwsn.atlassian.net/wiki/spaces/OW/overview.

[62] X. Vilajosana, S. Member, T. Watteyne, *et al.*, "Ietf 6tisch: A tutorial,"

[63] "Mouser: Nrf52840-dk," Accessed: 2022-1-3. [Online]. Available: https://mou.sr/3DmhdMC.

[64] "Mouser: Launchxl-cc2650," Accessed: 2022-1-3. [Online]. Available: https://mou.sr/3faY4pf.

[65] "Toolchain installation on linux - contiki-ng/contiki-ng wiki," Accessed: 2021-4-1. [Online]. Available: https://github.com/contiki-ng/contiki-ng/wiki/Toolchain-installation-on-Linux.

[66] M. Brachmann, S. Duquennoy, N. Tsiftes, and T. Voigt, "Core view metadata, citation and similar papers at core.ac.uk provided by universidad carlos iii de madrid e-archivo ieee 802.15.4 tsch in sub-ghz: Design considerations and multi-band support," [Online]. Available: https://www.ieeelcn.org/index.html.

[67] G. Oikonomou, S. Duquennoy, A. Elsts, J. Eriksson, Y. Tanaka, and N. Tsiftes, "The contiki-ng open source operating system for next generation iot devices," *SoftwareX*, vol. 18, Jun. 2022, ISSN: 23527110. DOI: 10.1016/J.SOFTX.2022.101089.

[68] M. Silva, D. Cerdeira, S. Pinto, and T. Gomes, "Operating systems for internet of things low-end devices: Analysis and benchmarking," *IEEE Internet of Things Journal*, vol. 6, no. 6, pp. 10 375–10 383, 2019. DOI: 10.1109/JIOT.2019.2939008.

[69] "Contiki-ng github examples/platform-specific," Accessed: 2022-2-15. [Online]. Available: https://github.com/contiki-ng/contiki-ng/tree/develop/examples/platform-specific.

[70] Q. Wang, X. Vilajosana, and T. Watteyne, "6tisch operation sublayer (6top) protocol (6p)," Q. Wang, Ed., Nov. 2018, ISSN: 2070-1721. DOI: 10.17487/RFC8480. [Online]. Available: https://www.rfc-editor.org/info/rfc8480.

[71] A. Dunkels, O. Schmidt, T. Voigt, and M. Ali, "Protothreads: Simplifying event-driven programming of memory-constrained embedded systems," *SenSys'06: Proceedings of the Fourth International Conference on Embedded Networked Sensor Systems*, pp. 29–42, 2006. DOI: 10.1145/1182807.1182811.

[72] "Contiki-ng: Protothreads," [Online]. Available: https://contiki-ng.readthedocs.io/en/master/_api/group__pt.html#details.

[73] D. Sudhakar, "Effectiveness of time-slotted channel hopping in wireless in-vehicle networks," 2015.

[74] J. F. Kurose, K. W. Ross, B. Columbus, *et al.*, *COMPUTER NETWORKING A Top-Down Approach SIXTH EDITION*. 2013, pp. 493–499, ISBN: 9780132856201.

[75] J. Olsson, "6lowpan demystified by texas instruments," 2014.

[76] "Documentation: 6tisch 6top sub layer · contiki-ng/contiki-ng wiki · github," Accessed: 2022-3-3. [Online]. Available: https://github.com/contiki-ng/contiki-ng/wiki/Documentation:-6TiSCH-6top-sub-layer.

[77] "6tisch custom-schedule," Accessed: 2022-4-3. [Online]. Available: https://github.com/contiki-ng/contiki-ng/blob/dad2f5917829e84f81a0433397cf7a48e7c7002e/examples/6tisch/custom-schedule/README.md.

[78] "Contiki build system · contiki-os/contiki wiki," Accessed: 2022-4-1. [Online]. Available: https://github.com/contiki-os/contiki/wiki/Contiki-Build-System.

[79] "Contiki build system - contiki," Accessed: 2022-3-6. [Online]. Available: https://anrg.usc.edu/contiki/index.php/Contiki%5C_build%5C_system.

[80] "A mote detection script for contiki-ng," Accessed: 2021-12-29. [Online]. Available: https://github.com/contiki-ng/motelist.

[81] "Uniflash software programming tool - ti.com," Accessed: 2022-3-22. [Online]. Available: https://www.ti.com/tool/UNIFLASH.

[82] "Downloads - gnu arm embedded toolchain downloads – arm developer," Accessed: 2022-3-22. [Online]. Available: https://developer.arm.com/downloads/-/gnu-rm.

[83] "Arch linux - arm-none-eabi-gcc 12.2.0-1 (x86_64)," Accessed: 2022-3-19. [Online]. Available: https://archlinux.org/packages/community/x86%5C_64/arm-none-eabi-gcc/.

[84] "Github: Contiki-ng code," Accessed: 2021-9-22. [Online]. Available: https://github.com/contiki-ng/contiki-ng.

[85] "Github: Contiki-ng tutorial: Switching to tsch," Accessed: 2022-10-22. [Online]. Available: https://github.com/contiki-ng/contiki-ng/wiki/Tutorial:-Switching-to-TSCH.

[86] "Solutions: Fire alarm systems: Fire laboratory bosch," Accessed: 2022-10-3. [Online]. Available: https://www.boschsecurity.com/xc/en/solutions/fire-alarm-systems/.

[87] "6tisch/timesync-demo from contiki ng," Accessed: 2022-10-1. [Online]. Available: https://github.com/contiki-ng/contiki-ng/tree/develop/examples/6tisch/timesync-demo.

[88] S. Duquennoy, B. Al Nahas, O. Landsiedel, and T. Watteyne, "Orchestra: Robust mesh networks through autonomously scheduled tsch," Nov. 2015. DOI: 10.1145/2809695.2809714.

[89] "Add aes-128 driver for cc26x0/cc13x0," Accessed: 2022-10-22. [Online]. Available: https://github.com/contiki-ng/contiki-ng/blob/df0a157b2335ff9c3bfd40052edf66b8a1cb2c2b/arch/cpu/cc26x0-cc13x0/dev/cc26xx-aes.h.

[90] "Tsch-and-6tisch link layer security," Accessed: 2022-10-22. [Online]. Available: https://docs.contiki-ng.org/en/develop/doc/tutorials/TSCH-and-6TiSCH.html.