Christian Gun Wurzer, B.Sc. B.A.(Econ.)

Design and Implementation of an Ultra Low Power WSN: A Case Study for a Future IoT-enabling SoC Development

MASTER'S THESIS

to achieve the university degree of

Diplom-Ingenieur

Master's degree programme: Information and Computer Engineering

submitted to

Graz University of Technology

Supervisor

Ass.Prof. Dipl.-Ing. Dr.techn. Christian Steger

Institute for Technical Informatics

Advisors

Ass.Prof. Dipl.-Ing. Dr.techn. Christian Steger
Dipl.-Ing. Dr.techn. Rainer Matischek (Infineon Technologies Austria AG)

AFFIDAVIT

I declare that I have authored this thesis independently, that I have not used other than the declared sources/resources, and that I have explicitly indicated all material which has been quoted either literally or by content from the sources used. The text document uploaded to TUGRAZonline is identical to the present masters thesis.

Date	Signature

Abstract

The Internet of things (IoT) is a network of connected physical objects that can exchange data and communicate with each other. With an increasing number of objects being connected to the Internet, the transition to the IoT is taking place right now. Since applications of the IoT such as those in smart homes and connected vehicles improve the standard of living, they are becoming increasingly popular and opening new markets. This development will lead to fast growth in the number of such devices in the coming years. Furthermore, with each new generation of products, the enabling hardware has to be more efficient and powerful. The technical characteristics of IoT devices include low power consumption, wireless connectivity, and specific applications requiring only low computing power.

This thesis evaluates the requirements for the development of a next-generation IoT-enabling System on Chip (SoC). To this end, a functional prototype of an SoC is created with off-the-shelf hardware components including a power management circuit, a transceiver chip, and a microcontroller.
The main focus is on ultra-low power consumption of battery-powered devices. This is because maintenance and deployment costs are drastically reduced when a device can be operated with a single battery for its entire lifetime.
To gain valuable insights into the potential for development of such a low-power SoC, the prototype created in this work is used in the implementation of a Wireless Sensor Network (WSN). This WSN is designed for use in two applications: the structural health monitoring of a building with sensor nodes mounted within its walls and the detection of a human being falling where sensor nodes operate in the floor. In addition, a lightweight network protocol with low overhead is designed to be used for communication between the sensor nodes in order to keep the power consumption low.

The result of this thesis is a functional WSN consisting of multiple sensor nodes and a gateway node. The sensor nodes are built using the functional prototype of the SoC and feature ultra-low power consumption. Measurements performed during multiple test runs are also discussed in this work. The resulting operational life estimates of the sensor nodes are up to several decades when the nodes are operated with a CR2032 battery.

Kurzfassung

Das Internet of Things (IoT) beschreibt das Konzept, physische Gegenstände so miteinander zu verbinden, dass Kommunikation und Datenaustausch zwischen diesen Gegenständen möglich wird. Mit der zunehmenden Anzahl von Gegenständen, die mit dem Internet verbunden sind, vollzieht sich schrittweise der Übergang zum Internet of Things. Die stetige Verbesserung der Lebensqualität führt zu immer größer werdenden Nachfrage nach Anwendungen wie Smart Home Systemen oder vernetzten Fahrzeugen, wodurch neue Märkte entstehen. Diese Entwicklung wird in den nächsten Jahren die Zahl der verbundenen Geräte in die Höhe schnellen lassen. Wichtige Voraussetzung dafür ist jedoch, mit jeder neuen Produktgeneration eine immer effizientere und leistungsfähigere Hardware zu generieren. Die dabei zu berücksichtigenden technischen Anforderungen von IoT Geräten umfassen unter anderem einen geringen Stromverbrauch, drahtlose Kommunikation und ein sehr spezifisches Anwendungsfeld, welches nur geringe Rechenleistung erfordert.

Diese Arbeit evaluiert die relevanten Anforderungen, welche für die Entwicklung der nächsten Generation eines SoC für IoT Anwendungen notwendig sind. Zu diesem Zweck wird ein funktionsfähiger Prototyp eines solchen SoC mit handelsüblichen Hardwarekomponenten wie einem Schaltkreis für das Energiemanagement, einem Funkmodul und einem Mikrocontroller angefertigt. Das Hauptaugenmerk liegt dabei auf einem sehr niedrigen Stromverbrauch, um batteriebetriebene Geräte zu ermöglichen. Dies hat den Vorteil, dass Wartungs- und Installationskosten drastisch reduziert werden können, wenn die Geräte über die gesamte Lebensdauer mit einer einzigen Batterie versorgt werden können. Um wertvolle Erkenntnisse für die zukünftige Entwicklung eines SoC mit geringem Energieverbrauch zu gewinnen, wird der Prototyp für die Implementierung eines WSN verwendet. Das WSN wurde für zwei Anwendungen entwickelt. Zum einen für die Überwachung des strukturellen Zustands von Gebäuden, wofür Sensorknoten in die Wände montiert werden. Zum anderen, um den Sturz einer Person erkennen zu können, dafür werden die Sensorknoten im Boden verbaut. Um den geringen Energieverbrauch gewährleisten zu können, wurde für die Kommunikation zwischen den Sensorknoten ein einfaches Netzwerkprotokoll erstellt.

Das Ergebnis dieser Arbeit ist ein funktionelles WSN, bestehend aus mehreren Sensorknoten und einem Gateway. Die Sensorknoten bauen auf dem SoC Prototyp auf und zeichnen sich durch einen sehr geringen Stromverbrauch aus. Durch die Analyse von Messdaten in Testläufen kann auf eine Lebensdauer eines Sensorknoten, unter Verwendung einer CR2032 Knopfzelle, auf mehrere Jahrzehnte geschlossen werden.

Acknowledgements

This master thesis was carried out at the Institute of Technical Informatics at the Graz University of Technology. The practical part was executed at Infineon Technologies AG at the Development Center Graz.

I would like to thank my supervisor Ass.Prof. Dipl.-Ing. Dr.techn. Christian Steger and my advisor Dipl.-Ing. Dr.techn. Rainer Matischek as well as Dipl.-Ing. Christoph Glanzer for their continuous and professional support during the creation of this thesis. Their feedback has sustainable improved this thesis.
Furthermore I would like to thank all persons who have been on hand to give me advice and support during this work.

Graz, August 2018 Christian Wurzer

Contents

List of Figures

List of Tables

Chapter 1

Introduction

In 2018, the term IoT is ubiquitous. Some have tagged the IoT and resulting innovations as being even larger than the Industrial Revolution [1].
The underlying concept is to connect everyday devices over the Internet, which improves and increases collaboration and data exchange.
In the 1960s, when the Internet was first developed, it set off an upheaval. Data and knowledge could suddenly be exchanged easily and fast. Today, the Internet has penetrated the business world (e.g., online shopping) as well as private life (e.g., social media) and is part of everyday life.
The IoT is the next technological step toward ubiquitous computing, where all things are connected. Digital communication with all things, together with richer information about the environment, makes new applications possible. Similarly to the Internet, the IoT will serve as a breeding ground for technical progress and innovations. This will have a huge impact on society and affect daily life. Areas such as health monitoring, energy consumption, and traffic control will especially benefit from this development. Indeed, the number of IoT devices is rapidly growing. Nordrum [2] and IHS Markit [3] predict that the number of IoT devices will reach 30 billion by 2020 and 125 billion by 2030.
Among the many applications of IoT devices, this work focuses on the domain of home automation. This domain covers health, security, utilities, and appliances and aims to improve living by making it more comfortable, safer, and more efficient.

1.1 Motivation

Brush, Lee, Mahajan, et al. [4] reveal that the high cost of ownership constitutes a barrier to the wider adoption of home automation. Thus, to increase the acceptance of home automation devices, maintenance and deployment costs have to be reduced. This cost reduction would be a big step forward in the universal adoption of smart home devices.
Furthermore, a compact design allows for easier installation of such products. Compact devices can be pre-installed in prefabricated houses without requiring large changes in the production process. Also, further improvements in power consumption make the use of battery-powered devices possible. Additional power supply lines are therefore not needed anymore, which reduces planning costs prior to deployment. Moreover, if these devices work using the same battery their entire operating life, the maintenance costs are reduced

to a minimum. The deployment costs will therefore plummet, allowing for new fields of application. Smart devices could then be operated within walls or in the floor.

The development outlined above makes the use of home automation devices much more attractive. Given the prediction of 125 billion IoT devices existing by 2030 [3], huge potential exists for the advancement of integrated circuits.

1.2 Objectives

This work is part of a large project whose goal is to develop a new-generation hardware platform that positively influences the size and energy consumption of smart devices. This new hardware platform is a highly Integrated Circuit (IC) that includes a transceiver, a Central Processing Unit (CPU), a non-volatile memory, and a power management unit. The goal of this work is to create a prototype of the final IC, providing similar functionality. This prototype is then used to build a WSN to test the application of the IC. To speed up the development of the final products, the WSN created is used by project partners in two installations. Subsequently, tests are carried out to collect data and provide information about future use cases and possible improvements to the IC hardware platform.
Overall, this work comprises a case study that plays a crucial role in the course of the larger project.

1.3 Outline

In the course of this thesis, a WSN node is constructed that uses a prototype of the new IC. Chapter 2 introduces the basic theoretical principles of wireless networking and the construction of small monitoring devices. Some WSNs that have already been put into operation are analyzed. As the IC is not yet available, the prototype created is composed of off-the-shelf hardware components with similar functionality as the final IC. Chapter 3 identifies the problem with and requirements for the WSN in as much exact detail as possible. On the basis of this analysis, the important features are identified and a design concept formulated. After the design of the prototype is finalized, a sensor node that is able to measure the environment and communicate is designed and implemented. This implementation is described in detail in Chapter 4. Multiple nodes are then connected to create a WSN that is deployed at the operation site. The deployment and operation phases, on the one hand, reveal possible use cases for the project partners, and on the other hand, provide an indication of possible modifications in the new IC. Several tests are performed and the recorded data is analyzed in Chapter 5. The insights gained are then compiled for use in subsequent stages of the project and are presented in the final chapter.

Chapter 2

Related Work

2.1 Internet of Things

Today, in 2018, the term IoT is ubiquitous. The concept of the IoT is to connect everyday objects such that they can communicate with each other. In the early days of the Web, an increasing number of people got connected to the Internet to communicate with each other. In this sense, the IoT can be seen as the next technological step, where all objects get connected to the Internet. The basic idea behind connecting all objects is to enhance everyday life. The number of possible applications is huge, as is the impact on society. According to Nordrum [2] the number of IoT devices will reach 30 billion by 2020. Moreover, IHS Markit [3] estimates that by 2030 125 billion IoT devices will exist.

2.2 Low-Power Wide Area Networks (LPWANs)

A LPWAN is the key enabler of the widespread use of IoT devices. It refers to a class of network protocols and techniques for connecting low-energy devices such as battery-powered sensors to a network server. These protocols are designed to help the devices achieve a long range and low power consumption at low operating costs. By implementing these design criteria, it is possible to cover huge areas with a small number of base stations. Indeed, with such networks, discussed in the next two sections, it is possible to connect almost any device to the Internet.

2.2.1 Long Range Wide Area Networks

Long Range Wide Area Network (LoRaWAN) is a low-power wireless network protocol. It is specified by the LoRa Alliance [5] and accessible to everyone and uses the chirp spread spectrum (CSS) modulation scheme called *LoRa* developed by the Semtech Corporation. The network topology is a star topology (see section 2.4) in which IoT devices communicate with gateways that forward data packets to a network server. This server offers interfaces to IoT platforms and applications. To achieve high efficiency in data transfer and energy consumption, LoRaWAN uses a spread spectrum modulation. Interferences can thus be avoided as far as possible. Communication in LoRaWAN is encrypted with the Advanced Encryption Standard (AES). In Switzerland, Swisscom has successfully

implemented a radio network based on *LoRa* with almost nationwide coverage. Swisscoms Low Power Network (LPN) [5] uses the frequecny band from 863 to 870 MHz for SRD (SRD860) frequency band. The LoRaWAN gateways of the Swisscom LPN transmit data with a maximum transmission power of 500 mW.

In addition, SK Telecom, the largest South Korean telecommunications company, succeeded in introducing a nationwide LPWAN based on LoRa technology for IoT infrastructure at the beginning of July 2016.[6]

2.2.2 Sigfox

Sigfox [7] is a French telecommunications company and a world-leading IoT service provider. Besides LoRaWAN-based infrastructure, Sigfox uses its own technology and protocol, and provides a network of base stations connected to a cloud system. A Sigfox message takes about 6 seconds to send. This is because of the modulation and data rate used in the Sigfox network. As Sigfox uses the SRD860 band, the requirements specified in Section 2.6.1 must be met. Therefore, the Sigfox protocol allows six messages to be sent per hour.

2.3 Wireless Sensor Networks

Wireless Sensor Networks (WSNs) are networks of wirelessly connected nodes that can sense the environment. Typically, such networks consist of tens or thousands of nodes and only a few data sinks (base stations). Each node is capable of sensing the environment, processing data, and communicating wirelessly. Harsh operating conditions and extreme resource constraints are common among WSNs. A prime example is the WSN deployed in 2013 to monitor the Helheim Glacier in Greenland. [8]

> "Designing a new wireless sensor node is extremely challenging task and involves assessing a number of different parameters required by the target application, which includes range, antenna type, target technology, components, memory, storage, power, life time, security, computational capability, communication technology, power, size, programming interface and applications." [9]

Because of numerous constraints, the design of WSNs is all about trade-offs among processing power, power supply, size, sensors, communication technology, memory, and protocol.

Despite the diverse characteristics of WSNs, Vidyasagar Potdar, Atif Sharif, and Elizabeth Chang [9] was able to identify the following distinct features of a WSN:

- Unique network topology
- Diverse applications
- Unique traffic characteristics
- Severe resource constraints

As WSNs are often operated in harsh environments and have limited power, they communicate over networks called Low-Power and Lossy Network (LLN). This class of networks is characterized by poor link quality between resource-constrained participants. Some protocols and strategies to deal with the difficult characteristics of these networks are discussed in Section 2.5.

The following section provides examples of WSNs and the results drawn from their operation. More WSNs can be found in [10].

2.3.1 Habitat Monitoring on Island off Coast of Maine

In 2002, Szewczyk, Polastre, Mainwaring, et al. [11] used a WSN off the coast of Maine to monitor the behavior of a population of Leachs storm petrels living on the island. The island is 15 kilometers off the coast of Maine. A WSN was chosen in this context for habitat monitoring as the dense monitoring capabilities promised to provide more insightful results than those from conventional methods such as data loggers. Beginning in the summer of 2002, the system was deployed and operated for a duration of 4 months. The WSN consisted of 43 sensor nodes, each powered by two AA batteries (estimated capacity of 2200 mA h). The nodes were deployed in underground nesting burrows and designed to detect the presence of a petrel by measuring the infrared radiation and the surrounding temperature. In addition, the sensor nodes were capable of measuring humidity, barometric pressure, and light intensity. These parameters were also monitored to gain insights about the deployed location of the nodes. Also, as the nodes were placed underground, they were especially exposed to rain water. Therefore, the nodes were sealed with an acrylic plastic case. Only the sensors were exposed to the environment. The nodes woke up every 70 seconds and took measurements of the environment, then used their network stack and saved the data to their internal flash memory. After this, they went into a power-saving sleep mode for the next 70 seconds. The duty cycle was about 1.7 %. Also, the sensor nodes could only transmit rather than receive data. In addition to monitoring the habitat, Szewczyk, Polastre, Mainwaring, et al. [11] wanted to gain insights into the process of operating a WSN and the problems that could occur. Therefore, they analyzed their findings in detail. First, the data gathered to track the Leachs storm petrels could not be used. The sensors used to detect the presence of the birds either reported maximum or minimum values for the sensor type. Without any ground truth, the data could not be considered reliable. The authors concluded that the sensors might have malfunctioned as they reported only the extremes of their measuring scale. Second, the communication between the nodes and the gateway was unreliable. Some nodes in particular had problems communicating at all. Some other nodes caused massive interference and rendered the radio link unusable for the rest of the nodes. Third, the sealing of at least some nodes was insufficient. As a result, moisture caused these nodes to not operate while wet. This led to correlation between high humidity levels and failure of the sensor nodes. Szewczyk, Polastre, Mainwaring, et al. [11] were able to determine indicators to predict the failure of sensor nodes with high accuracy.

2.3.2 Monitoring Glacier Dynamics

Martin, O'Farrell, Aspey, et al. [8] deployed a sensor network on the Helheim Glacier in Greenland to monitor glacier dynamics throughout the summer of 2013. They wanted to measure the exact flow rates of the glacier by using the Global Positioning System (GPS) positions of the sensor nodes. To do this, they deployed 20 sensor nodes at the calving edge of the glacier. The system was operated for 50 days and provided near real-time position and velocity data. Each node sensed every seven seconds. To obtain accurate data, Martin, O'Farrell, Aspey, et al. [8] used dual frequency GPS, which provides location accuracy down to a few centimeters. They also used a standard Zigbee Waspmotes for communication (2.4 GHz band). Additionally, to avoid problems with transmission collisions, they used a round-robin scheme. This is because the glacier surface made collision avoidance hard as it caused the hidden node problem by shielding nodes from each other. The base station polled all sensor nodes associated with it every seven seconds. Furthermore, to minimize data loss, each sensor node operated two independent network modules sending data on different frequencies and communicating with different base stations. This type of radio diversity provides two different communication paths to each base station. The 20 nodes were further divided into groups of 5 and formed two subnets in each group. Each base station was responsible for managing two subnets of two groups of 5 sensor nodes. All nodes within a group were connected through a mesh network to allow data from the sensor nodes sent via other nodes in the network to reach the base station. The sensor nodes were powered by rechargeable lead-acid batteries that got charged by using solar panels. As weather conditions were good, there were no power failures during operation. The authors conclude that with the implemented radio diversity, the outage rate was below 1 percent. Without this diversity, the outage rate would have been much higher, as demonstrated by the data.

2.3.3 Monitoring Energy Consumption

Kappler and Riegel [12] goal was to save energy in typical office environments. They state that 20 percent of energy currently consumed could easily be saved without much effort. To do this, However, the actual energy consumption of devices has to be known. The WSN they installed achieved exactly this. Each sensor node was able to measure the energy consumption of a device connected to a power outlet. The sensor nodes were themselves powered by these power outlets. They communicated over the SRD860 band. Furthermore, with the regulations implemented by the Energy Regulatory Commision (ERC), the transmission power was limited to 25 mW (see Section 2.6.1). With such limited sending power, communication indoors was limited to about 35 m, because of which multi-hop routing protocols were required in order to cover large areas. While operating their simple WSN, they found, amid other useful information, that a packet loss of up to 10 percent is tolerable for such a power-measuring application as the rest of the data can easily be reconstructed from the rest of the data.

2.3.4 Monitoring rare and endangered species of plants

Another study deploying a WSN was conducted by Biagioni and Bridges [13]. The goal of this WSN was to gain insights into the growth of rare and endangered species of plants in

Hawaii. To achieve this, long-term time series of humidity, temperature, solar radiation, and wind were of relevance. The main reason for deploying a WSN was to minimize the impact on the habitat, which could place the rare species in further risk. The WSN made frequent visits unnecessary. One design criterion for the sensor nodes was a camouflaged appearance. As many people visited the deployment area, the nodes had to fit seamlessly into their surroundings. In total, 60 sensor nodes were deployed over an area of roughly 2 square kilometers. The sensor node density was high at the points of interest and low in surrounding areas. All sensors were equipped with temperature, humidity, wind, and light sensors, and some special nodes also had a high-resolution camera installed. The radios operated in the ITU band from 902 to 928 MHz (ISM900). The transmission range was tested and turned out be approximately 30 meters when placed 10 centimeters above ground, and 100 meters when placed 2 meters above ground. Accordingly, the sensor nodes placed on the ground were deployed at a distance of 30 meters. This way, each sensor had four to six neighbors within reach. The data sink was operated 2 kilometers away from the area of interest. To cover this distance, the sensor nodes were placed in trees (about 2 meters above ground) and about 50 meters away from the next node. Again, these nodes also had multiple neighbors so as to be redundant. The nodes were programmed to wake up every 10 minutes and perform networking and sensing tasks. In the sleep phase, the nodes were turned off and consumed almost no energy. Biagioni and Bridges [13] expected the nodes to achieve a lifetime of 6 weeks with four C-size batteries. For the data transmission, the routing protocol of Gradient Routing in Ad Hoc Networks (GRAd) (see [14]) was used.

2.4 Wireless Network Topologies

The topology of a network describes the relationship between nodes and is thus a key parameter that fundamentally influences the systems operation. The choice of topology has an impact on a number of features such as overall energy consumption, number of messages sent, link reliability, number of gateways necessary, and the maximum number of nodes.

Star Toplogy

The star topology has one main node that communicates with all the others. Figure 2.1 illustrates such a network. The main advantage of this topology is its simplicity. Minimal logic within network nodes is necessary to communicate with other nodes. For nodes to communicate with other nodes, the gateway has to work as a relay station. A downside of this topology is that if the link quality between a node and the gateway is poor, it becomes very expensive in terms of energy consumption. Even if better routes (over other nodes) exist to the gateway, these are not exploited. This could lead to early battery depletion in some nodes, while nodes with good link quality would last longer.

Tree Topology

An example of the tree topology is presented in Figure 2.2. It is a hierarchical topology. At the bottom of the hierarchical structure, the sensor nodes are located. The next layer

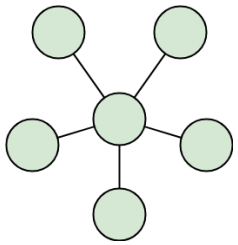


Figure 2.1: Star network topology.

contains intermediate nodes. Nodes below an intermediate node communicate only over the intermediate node. This structure can exist multiple times within the tree. The gateway is the root of this tree. An advantage of this topology is that it comprises only a few intermediate nodes that need to forward messages. Therefore, the network scales excellently with the number of nodes. However, nodes of different roles (intermediate and sensor nodes) have different tasks and need different resources. This increases the systems complexity.

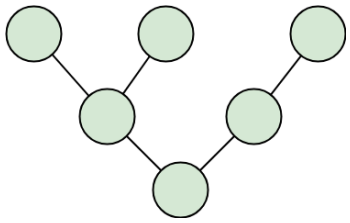


Figure 2.2: Tree network topology.

Mesh Topology

Another option is depicted in Figure 2.3. This topology is a combination of the previous two. Some nodes communicate directly with the gateway, whereas others communicate only with their neighbor nodes. Since the gateway is the data sink, all sensor data has to be sent to the gateway. As in the tree topology, some nodes function as relay stations. The main advantage of this topology is the spatially extended range of the network. Nodes that are too far away to have sufficient link quality can now use other nodes as relay stations to communicate with the gateway. However, energy consumption, especially of the nodes functioning as relay stations, increases and thus reduces system uptime. Another aspect worth considering is the need for a routing protocol if the mesh is built dynamically at runtime. Without the dynamic building of the mesh, the advantages diminish as structural conditions cannot be adapted to.

All three approaches have their advantages and disadvantages. The complexity of the star topology is low, while that of the mesh network is high. The tree topology has medium complexity. However, the star networks functionality, especially its robustness with regard to environmental factors, is low, while that of the mesh topology is high. The functionality of the tree structure, again, lies between the other two. As greater complex-

ity is costly in terms of the energy consumed by each node, the choice of topology depends on the required link quality, spatial sprawl, cost of transmission, and other factors.

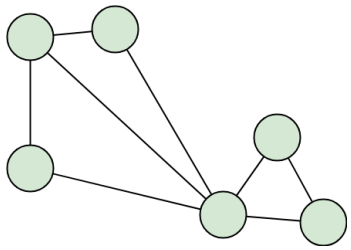


Figure 2.3: Mesh network topology.

2.5 Wireless Protocols

With the connection of multiple computers and the rapid spread of the Internet, a model that makes the development of new communication protocols easier has been created. This model, called the OSI Model, consists of seven layers. The interface between these layers is standardized, allowing for each layer to be developed separately. Figure 2.1 displays the structure of the OSI Model.

OSI Model		
Layer		Protocol data unit
Host Layers	7. Application	Data
	6. Presentation	
	5. Session	
	4. Transport	Segment, Datagram
Media Layers	3. Network	Packet
	2. Data link	Frame
	1. Physical	Symbol

Table 2.1: The OSI Model.

2.5.1 Time Division Multiple Access (TDMA) versus Carrier Sense Multiple Access with Collision Avoidance (CSMA/CA)

An important design feature of radio-based communication protocols is the access to the communication medium, in this case a frequency spectrum on which electromagnetic waves are transmitted. Nowadays, two main methods have prevailed and will be discussed in this section.

There is Carrier Sense Multiple Access with Collision Avoidance (CSMA/CA) which is widely used for non time-sensitive applications as it is very well scalable and does not need any additional communication overhead.

The main idea is that collisions are possible to happen and to deal with this kind of situation there are special procedures defined. On top of that everything is done to avoid collisions on the first hand.
Despite these properties Zhuo, Wang, Song, et al. [15] have analyzed CSMA/CA approach and their benefits for soft real-time applications.
An typical example of an CSMA/CA protocol behavior is to check weather the communication channel is clear or not. If the channel meets the condition of an clear channel the communication is initiated. If a collision was detected then some sort of resend has to occur or the message is dropped.

Time Division Multiple Access (TDMA) on the other hand is designed to not let collisions happen in the first place. One part of the communication has to take care of assigning all communication partners time slots where they are allowed to communicate. Therefore if implemented correct there can not occur any collisions with the communication partners on the same channel.
The downside of this approach is that it adds complexity compared to the CSMA/CA approach. On top of that there has to be some agreement between communication partners on correct time slots. Therefore it needs a master to taking care of the time slot assigning and handling of new participants entering the network.
A detailed comparison of the mentioned approaches are discussed in [15] and [16].

2.5.2 IEEE 802.15.4

Institute of Electrical and Electronics Engineers (IEEE) 802.15.4 is a standard for a communication protocol operating on the lowest two layers of the OSI model. Main goals were low power consumption to enable battery based operation of participants, such low data rates, secure and reliable communication over ISM bands. All these properties make IEEE 802.15.4 a good candidate for WSNs.

2.5.3 6LoWPAN - IPv6 over Low power Wireless Personal Area Network

IPv6 over Low power Wireless Personal Area Network (6LoWPAN) operates on the network and transport layer of the OSI model. The idea of 6LoWPAN is that networks can easily be integrated in already existing network infrastructure as it uses IPv6 packets. It is designed to operate on IEEE 802.15.4 based network (which operates on data-link and physical layer of the OSI model). It enables header compression and support for routing.

2.5.4 RPL - IPV6 Routing Protocol for Low-Power and Lossy Networks

A study conducted by the Internet Engineering Task Force (IETF) concluded that the performance of existing routing protocols such as Open Shortest Path First (OSPF), Intermediate System to Intermediate System Protocol (IS-IS) and Optimized Link State Routing (OLSR) on Low-Power and Lossy Networks (LLNs) perform terribly. The IETF came up with an routing protocol especially tailored for such networks, the Routing Protocol for Low power and Lossy Networks (RPL). The idea of RPL is to define a quasi standard in order to prevent a fragmentation of WSNs market. RPL was designed in close

cooperation of 6LoWPAN. It is an IPv6 routing protocol which features point to point, multi-point to point or point to multi-point connection and network sizes up to a few thousand routers. RPL is especially designed to deal with constraints as low processing power, limited memory high data loss, low data rates, limited energy (battery powered nodes). With all these features RPL seems to be the ideal solution for any WSN. The paper *A Critical Evaluation of the IPv6 Routing Protocol for Low Power and Lossy Networks (RPL)* [17] analyzed the routing protocol and concluded that RPL is not yet ready to become a industry wide standard. One of the major critics is that its complexity does not allow for a lightweight implementation.
The Contiki implementation of RPL which only implemented a subset of the functionality needs about 50kB of memory [17]. Typical configurations of ultra low power (ULP) sensor nodes have tens or hundreds of kB of memory. This means that for such nodes the protocol implementation needs a significant fraction of available memory thus limiting the functionality of these nodes. Detailed description of the RPL can be found in the RFC6550 [18].

2.6 Frequency Allocation

Simultaneous usage of the same radio frequency (RF) lead to interferences and unreliable communication. As the number of devices communicating wireless grow the RF spectrum usage increases and hence inefficiencies occur. To reduce this problem people have started to regulate the RF spectrum. But unlike countries RF are not bounded by borders. Over the time regions have come together and created regulators which manage spectrum usage. Such bodies are the International Telecommunication Union Radio Regulations (ITU), European Conference of Postal and Telecommunications Administrations (CEPT) and Inter-American Telecommunication Commission (CITEL). Despite efforts to standardize the regulations many different rules and or exceptions exist for countries or regions. Often rules for RF spectrum are covered by more than one regulator.

Regulators divide the whole RF spectrum (3kHz - 300 GHz) into smaller sections. Each band is then auctioned or allocated to specific companies or applications. Such frequency auctions are common in the mobile phone market, where providers pay lots of money to get exclusive usage rights for RF bands.
To fuel innovations there are some free of charge RF bands. These bands can be used by anyone such as amateur radio operators. To also keep interference low on these bands the regulators require to meet certain criteria for operating in these bands. A detailed description of the frequency band allocation for Germany can be found in [19]. As the WSN developed in this thesis operates in one of these free to use RF bands, the following sections discuss two of the most important regulators and the rules imposed these RF bands.

2.6.1 Industrial, scientific, and medical radio bands

The industrial scientific and medical radio bands (ISM bands) are radio bands reserved internationally by the ITU to be used for industrial, scientific and medical applications.

The usage of these radio frequency is license free and does not have to be registered. Table 2.2 shows the International Telecommunication Union Radio Regulations (RR) and ITU frequency allocations for different regions [20]. The ITU divides the world in three regions (ITU Region 1, 2 and 3). Not all frequency bands are available everywhere.
The ITU band from 433.05 to 434.79 MHz (ISM433) and ISM900 bands are widely used for temperature sensors remote controls or by amateur radio operators.

Frequency range			**Availability**
6.765	- 6.795	MHz	Subject to local acceptance
13.553	- 13.567	MHz	Worldwide
26.957	- 27.283	MHz	Worldwide
40.66	- 40.7	MHz	Worldwide
433.05	- 434.79	MHz	ITU Region 1 (Europe, Africa, former Soviet Union and the Middle East)
902	- 928	MHz	ITU Region 2 (South and North America including Greenland)
2.4	- 2.5	GHz	Worldwide
5.725	- 5.875	GHz	Worldwide
24	- 24.25	GHz	Worldwide
61	- 61.5	GHz	Subject to local acceptance
122	- 123	GHz	Subject to local acceptance
244	- 246	GHz	Subject to local acceptance

Table 2.2: ISM bands as defined by the RR (see Chapter 2 in [20]).

2.6.2 Short range device radio bands

Besides the ITU the Electronic Communications Committee (ECC) as part of European Conference of Postal and Telecommunications Administrations (CEPT) allocated frequency ranges to be used for Short Range Devices (SRD). The ECC created the category of SRD to be used in these free to use bands. Despite the efforts to standardize these frequency bands each partner of the CEPT can impose own rules. It is therefore advised to look up the specific regulations for each country, which can be found in [21].
Table 2.3 shows the allocated frequency ranges for usage of SRD. On top of the bands allocated for free use by the ITU the ECC defines four additional spectrum's SRD.
The ISM433 and the frequency band from 433.05 to 434.79 MHz for LPD (LPD433) bands are identical.

Frequency range			**Notes**
6765	- 6795	kHz	ISM
13.553	- 13.567	MHz	ISM
26.957	- 27.283	MHz	ISM
40.660	- 40.700	MHz	ISM
138.20	- 138.45	MHz	
433.050	- 434.790	MHz	ISM / LPD433
863	- 870	MHz	SRD860
2400.0	- 2483.5	MHz	ISM
5725	- 5875	MHz	ISM
24.00	- 24.25	GHz	ISM
61.0	- 61.5	GHz	ISM
122	- 123	GHz	ISM
244	- 246	GHz	ISM
3.1	- 4.8	GHz	
6	- 9	GHz	

Table 2.3: Frequency allocation for non-specific SRD by the ECC

The WSN designed in this thesis will operate in the SRD860 band. The specific regulations limitations and requirements are stated in table 2.4.
Section 4.2.3 discusses the transceiver configuration used for this project. The center frequency used for transmission is 868.025 635 MHz.

The WSN therefore has to satisfy the following regulations:

- maximum transmission power of 25 mW e.r.p.
- not user adjustable duty cycling
- guaranteed $< 1\%$ duty cylcing
  Typical Observation time is one hour. Maximum transmitter on time is 3.6 seconds therefore. For detailed explanation see the ERC Recommendation 70-03 page 79 [21].

Frequency Band		Power / Magnetic Field	Spectrum access and mitigation requirements	Modulation / maximum occupied bandwidt	Notes
h1.0	862-863 MHz	25 mW e.r.p.	0.1% duty cycle	350 kHz	
h1.1	863-870 MHz [3],[4]	25 mW e.r.p.	0.1% duty cycle or LBT [1],[5]	100 kHz for 47 or more channels [2]	FHSS. Parts of the frequency band are also identified in Annexes 2 and 3
h1.2	863-870 MHz [3],[4]	25 mW e.r.p.; Power density: -4.5 dBm/100 kHz	0.1% duty cycle or LBT+AFA	Not specified	DSSS and other wideband techniques other than FHSS. Parts of the frequency band are also identified in Annexes 2 and 3
h1.3	863-870 MHz [3],[4]	25 mW e.r.p.	0.1% duty cycle or LBT + AFA [1],[5]	100 kHz, for 1 or more channels modulation bandwidth 300 kHz [2]	Narrowband / wideband modulation. Parts of the frequency band are also identified in Annexes 2 and 3
h1.4	868-868.6 MHz [4]	25 mW e.r.p.	1% duty cycle or LBT +AFA [1]	Not specified for 1 or more channels [2]	Narrowband / wideband modulation. No channel spacing, however the whole stated frequency band may be used
h1.5	868.7-869.2 MHz [4]	25 mW e.r.p.	0.1% duty cycle or LBT+AFA [1]	Not specified for 1 or more channels [2]	Narrowband / wideband modulation. No channel spacing, however the whole stated frequency band may be used
h1.6	869.4-869.65 MHz	500 mW e.r.p.	10% duty cycle or LBT+AFA [1]	Not specified for 1 or more channels	Narrowband / wideband modulation. The whole stated frequency band may be used as 1 channel for high speed data transmission
h1.7	869.7-870 MHz [11]	5 mW e.r.p.; 25 mW e.r.p.	No requirement for 5 mW e.r.p.; 1% duty cycle or LBT+AFA [1] for 25 mW e.r.p.	Not specified for 1 or more channels	Narrowband / wideband modulation. No channel spacing, however the whole stated frequency band may be used

Table 2.4: Subdivision of the SRD860 band and usage conditions to be complied with.

[1]When either duty cycle, Listen Before Talk (LBT) or equivalent technique applies then it shall not be user dependent/adjustable and shall be guaranteed by appropriate technical means. For LBT devices without Adaptive Frequency Agility (AFA), or equivalent techniques, the duty cycle limit applies. For any type of frequency agile device the duty cycle limit applies to the total transmission unless LBT or equivalent technique is used.

[2]The preferred channel spacing is 100 kHz allowing for a subdivision into 50 kHz or 25 kHz.

[3]Sub-bands for alarms are excluded (see ERC/REC 70-03 Annex 7).

[4]Audio and video applications are allowed provided that a digital modulation method is used with a max. bandwidth of 300 kHz. Analogue and digital voice applications are allowed with a max. bandwidth = 25 kHz. In sub-band 863-865 MHz voice and audio conditions of Annexes 10 and 13 of ERC/REC 70-03 apply respectively.

[5]Duty cycle may be increased to 1% if the band is limited to 865-868 MHz.

[6]For wideband techniques, other than FHSS, operating with a bandwidth of 200 kHz to 3 MHz, the duty cycle can be increased to 1% if the band is limited to 865-868 MHz and power to =10 mW e.r.p.

[7]The power density can be increased to +6.2 dBm/100 kHz and -0.8 dBm/100 kHz, if the band of operation is limited to 865-868 MHz and 865-870 MHz respectively.

[11]Audio and video applications are excluded. Voice applications (analogue or digital) are allowed with a maximum bandwidth of 25 kHz, and with spectrum access technique such as LBT or equivalent and shall include a power output sensor controlling the transmitter to a maximum transmit period of 1 minute for each transmission.

2.7 Power Sources for Wireless Sensor Networks

Today, most WSNs are battery-powered as batteries offer an attractive and reliable way to power such devices. However, the size of batteries often makes them the limiting factor of sensing devices; moreover, replacing batteries is cost-intensive. Thus, as stated by Karl, Willig, and Wolisz [22], other sources of power should be explored to counter these problems.

2.7.1 Batteries

Smaller portable electronic devices are increasing the demand for smaller energy sources. However, advances in battery technology have not kept pace with innovations in microelectronics and wireless communications. Technological advancements have not yet been explored sufficiently for energy sources. To bridge this technological gap, the design focus for wireless sensing systems has been shifting toward energy efficiency. This is especially important as small form factors do not allow for the use of large capacity batteries [23]. An element in the design consideration is how the battery is operated as the lifetime does not depend only on the total power consumption but also on the discharge characteristics. Many wireless sensing systems are operated in two phases: active and standby. Both phases have different discharge characteristics. In standby, the current consumption is usually in the range of A, whereas in the active phase, the sensing devices need currents in the mA range. This results in pulsed discharge of the battery. As Yin Zhang [23] revealed, this kind of discharge could significantly improve battery life compared to a continuous discharge with the same power drawn from the battery but with a discharge current higher than the average discharge current for the duty-cycled application. Importantly, as the discharge currents in these active phases could significantly exceed the rated nominal current of the battery, they have to be tested beforehand to check whether they are suited for the application.

Moreover, the actual capacity of a battery depends on many factors such as discharge current, temperature, discharge time, recovery time, nominal capacity, self-discharge rate and age. Thus, the determination of the actual capacity is not trivial.

Capacity Considerations

Batteries have rated, or standard, capacities which do not have to be met in real operation. For example, the CR2032 coin cell from VARTA has a rated capacity of 230 mA h [24]. This capacity can be extracted when discarded with a standard load or current. Depending on the discarding current or load, the measured capacity varies (see [24]). Despite the discharge profile, several environmental parameters play a role in the actual capacity of a battery [25]. Therefore, the actual capacity can deviate widely from the rated capacity.
In particular, high currents drawn from the battery have a negative influence on the capacity as the chemical reactions in these situations render some of the available capacity useless. However, these effects are reversed if the battery is allowed to idle for some time. This is called the recovery effect, and it can significantly extend battery life [25]. Thermal effects also have a huge impact on the lifetime of batteries. At higher temperatures, the capacity increases and internal resistance decreases, but the self-discharge rate also

increases. The inverse is true for decreasing temperatures. This makes lifetime estimation even harder, especially for systems designed to operate for years. The self-discharge rate of lithium coin-cell batteries increases by under 1 to over 50 percent per year when the temperature increases from 25 to 80 degrees Celsius [26]. This means that even if no current is drawn at all, half the capacity is lost after 1 year of operation.
Tests conducted by Deng, Yang, Cai, et al. [26] reveal that higher transmission power levels do not only decrease battery life as expected but also decrease the available capacity of the battery. Lowering the transmission power could therefore increase the lifetime of the battery (and hence the system runtime).
The authors of this study also conclude that pulsed current consumption with long versus short recovery times has a huge impact on the actual battery capacity. Tests showed that the actual capacities deviated by a factor of 4. In addition, the use of a voltage regulator might help against transmission power degradation as described in [26]. However, voltage regulators have to be supplied with energy as well and could reduce the system runtime by up to 40 percent, as demonstrated in [11].

2.7.2 Energy Harvesting

For the autonomous operation of WSNs, energy harvesting is a possible solution to power sensing devices. Energy harvesting can be one of the few ways to ensure long-term operation when on-site maintenance is impossible. This method has been used in many applications before (see Section 2.3.2 and [8] for an example). Because the harvested energy is usually very low, complicated electronic circuits in combination with a form of energy storage must be used. Such energy storage devices could be small rechargeable batteries or super capacitors, and have to be chosen depending on the application.

RF Power Distribution

One way to harvest energy is to use electromagnetic radiation as a power source. This is widely used in radio-frequency identification (RFID) tags or smart cards. With this approach, power can be transmitted over the air by a power transmitting device. Because the radiated power decreases with the squared distance, high power transmission is impossible [22].

Photovoltaics

For devices that are used outdoors, photovoltaics is one way to guarantee autonomous operation. Solar panels can be used to charge secondary batteries when enough light is available. The use of photovoltaics is very common.

Temperature Gradients

Temperature gradients occur naturally almost everywhere and are therefore of interest as an energy source. Today, the most common method to generate electricity from temperature gradients is to use thermoelectric generators [22]. There are generators that produce up to 40 µW from a 5 °C temperature difference where the area is only half a square cen-

timeter [22]. However, one should take into account that these temperature gradients, which occur naturally, also disappear naturally over time.

Vibrations

Similar to temperature gradients, vibrations occur in many environments. Some of these are caused by windows, airplanes, building sites, bridges, and trains. These vibrations can be used to generate energy using the piezoelectric effect.

Chapter 3

Design and Concept

This chapter discusses the design and concept phase of the thesis. The use cases are discussed, after which the functional requirements are defined. The nonfunctional demands of all project partners are incorporated into the final set of requirements. These are then further analyzed in Section 3.2. After the system and network architecture are defined based on the requirements, the protocol design is explained. This chapter concludes with a short description of the hardware and software components of the project.

3.1 Use Cases

The WSN designed in this thesis is an early prototype of the WSN defined in the DeSSnet project proposal that specifies multiple scenarios and requirements to be fulfilled. The scenarios can roughly be divided into two groups: the *monitoring use case* and the *fall detection use case*. These use cases are mainly distinguished by the sensing interval, the type of parameters measured, and most important, the event that triggers the measurement.
The following sections describe the differences between and requirements of the use cases.

3.1.1 The Monitoring Use Case

As the name suggests, the system performs monitoring tasks in this use case. These tasks include but are not limited to monitoring the structural health of buildings by capturing data about temperature or humidity levels between or within walls, monitoring the living quality of a home by sensing volatile organic compound (VCO) temperature and humidity indoors, or saving energy through optimally heating or cooling a house by sensing the air temperature inside and outside the house.
All these tasks are characterized by a relatively low sensing interval some minutes to hours as the parameters of interest change slowly.
The system defined in the DeSSnet [27] research project proposal will primarily operate in this mode.

Smart Sensing

One of the main problems with battery-powered sensor nodes is the available energy. To reduce the energy spent on sending data to the gateway, one could modify the sensing and sending intervals. Depending on the environment, the state of a sensor not changing might not be of interest, and therefore, this information does not need to be sent to the gateway. An example could be windows that are monitored by a contact switch. That the window is still closed is low-value information and can be omitted. On the other hand, it might be of significant interest that the window has been opened. Another example is that of sensing the humidity of a wooden floor. The humidity is not expected to change rapidly; therefore, a large sensing interval can be chosen. In the case of a water leak, the humidity would increase above expected values and a shorter sensing interval might be of interest.

3.1.2 The Fall Detection Use Case

For highly important, time-sensitive conditions, the sensing interval of the normal case is not sufficient. For instance, in the case of an anti-theft system, a shattered window is a useful indicator of burglary. However, this information is only valuable if reported immediately.
Therefore, a mechanism must exist to sense and send data beyond the predefined intervals. An important consideration here is that more than one sensor node might detect the same event. Therefore, congestion within the network has to be managed.

3.2 Requirements

The goal of this thesis is to design and implement a WSN to monitor humidity, temperature, and other parameters. The data gathered by a WSN can be used to enhance living conditions. Applications include regulating temperature and humidity, saving energy, increasing the lifetime of objects by optimally scheduling service intervals, and monitoring the health status of residents.

3.2.1 Detailed Requirement Analysis

The following section presents an overview of the system requirements.

- **Sensing of the environment**
  To fulfill its monitoring task, the WSN must be able to periodically sense the environment. Relevant parameters include temperature, humidity, and barometric pressure.

- **Adaptive sensing and smart operation**
  In addition to the WSNs monitoring operation, events of high importance need increased sensing. For example, higher-than-normal humidity could be the first indicator of a water leak. A reduced sensing interval that allows for closely monitoring the situation and alerting the owner helps reduce the maintenance costs of the building.

- **Wireless communication**
  The processes of designing modern buildings are already complicated. The aim is to add as little overhead to existing processes as possible. Additional effort required to incorporate communication and power cables for sensing devices into building plans is not feasible. Wireless communication is therefore important because it can be easily installed and operated even after the building process is finished. This grants maximum flexibility and ensures easy installation and operation of the whole system.

- **Extended uptime**
  The WSN, which is powered by small batteries, will be placed and used in highly inaccessible places. Changing batteries might not be possible, or doing this may incur high costs. Therefore, the batterys lifetime is a major factor for the system to be fully operational. A longer battery lifetime reduces maintenance costs, thus increasing the benefits of the system.

- **Hardware components**
  The designed system represents an early stage of the DeSSnet project. Many components used in this version of the system will be optimized and adapted to meet requirements as the project moves to a subsequent stage. This entails using non-optimized components in the first stage in order to reveal problems and indicate the most promising optimizations. The system design therefore evolves during the production process of DeSSnet.

- **Form factor**
  The sensor nodes footprint should be as small as possible to increase deployability.

3.2.2 Limitations

As pointed out in Section 3.2, the designed system utilizes components that are not yet highly optimized. This leads to an overall system performance that is not the best in its field. Particularly, the uptime suffers in the early stage of the project. Despite the hardware limitations imposed, the system is designed to be as efficient and high-performing as possible.

3.3 System Architecture

The final system consists of multiple sensor nodes and one gateway node, and is illustrated in Figure 3.1. Each sensor node can be configured to monitor parameters such as humidity, temperature, pressure, barometric pressure, light intensity, vibration, or combinations of these. The time interval of measurement can also be configured. The gateway is connected by cable to a command server that is further connected to the cloud. The gateway receives all sensor data and manages all sensor nodes associated with it. The command server can be connected to multiple gateways that can again be coupled with multiple sensor nodes. This modular architecture is scalable and can easily be modified for many use cases. For example, there could be a gateway on every floor of a building. All sensor nodes are assigned according to the gateway, and all gateways are connected to the command server, which is connected to the Internet.

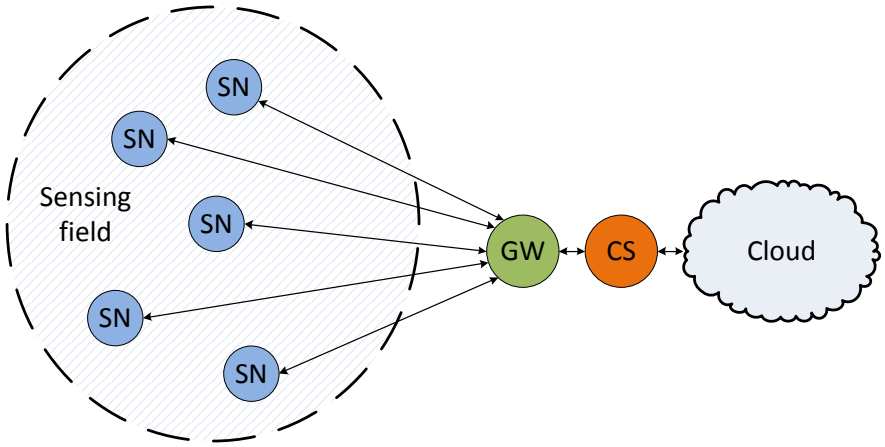


Figure 3.1: System overview. The command server (CS) communicates with the gateway (GW) and the gateway communicates with sensor nodes (SN). The command server (CS) can be connected to the the Cloud. For a fully functional system there has to be at least one command server one gateway and one sensor node.

The command server and connection to the cloud are not part of this thesis. The communication between the command server and the gateway is discussed in Section 3.4.3. The following sections explain the architecture of the sensor node and the gateway.

3.3.1 Sensor Node

Figure 3.2 illustrates the architecture of the sensor nodes. The nodes are battery-based and operate on a coin cell battery. A power management unit ensures long uptime of each node by periodically waking it up and turning it off afterward. The microcontroller has to be powerful enough to process measured data and communicate with the gateway but lightweight in energy consumption and size. For communication, the TDA5340 [28], a powerful transceiver that supports multiple frequencies and modulation schemes, is used. The system operates on the SRD860 band (see Section 2.6.1) and is connected to an appropriate Printed Circuit Board (PCB) antenna. This technique helps achieve a small footprint.
Moreover, the sensing is done via appropriate sensors that can easily be replaced. The following is a list of sensors, a detailed description of which can be found in Section 3.5.5.

- Barometric pressure sensor
  High sensitivity barometric pressure sensor for weather forecasts when used outdoors or to detect a malfunction of air conditioning when used indoors.

- Humidity sensor
  Humidity sensor capable of measuring the whole range of relative humidity and taking additional temperature measurements.

- Accelerometer
  Detection of the sensor node's orientation such as vibration measurements.
- Pyzo-Flex Foil
  Deformation detection based on the piezoelectric effect.

The PCB design is planned to be as small as possible, comprising only a few layers, which helps keep the cost low and makes a possible redesign easier.

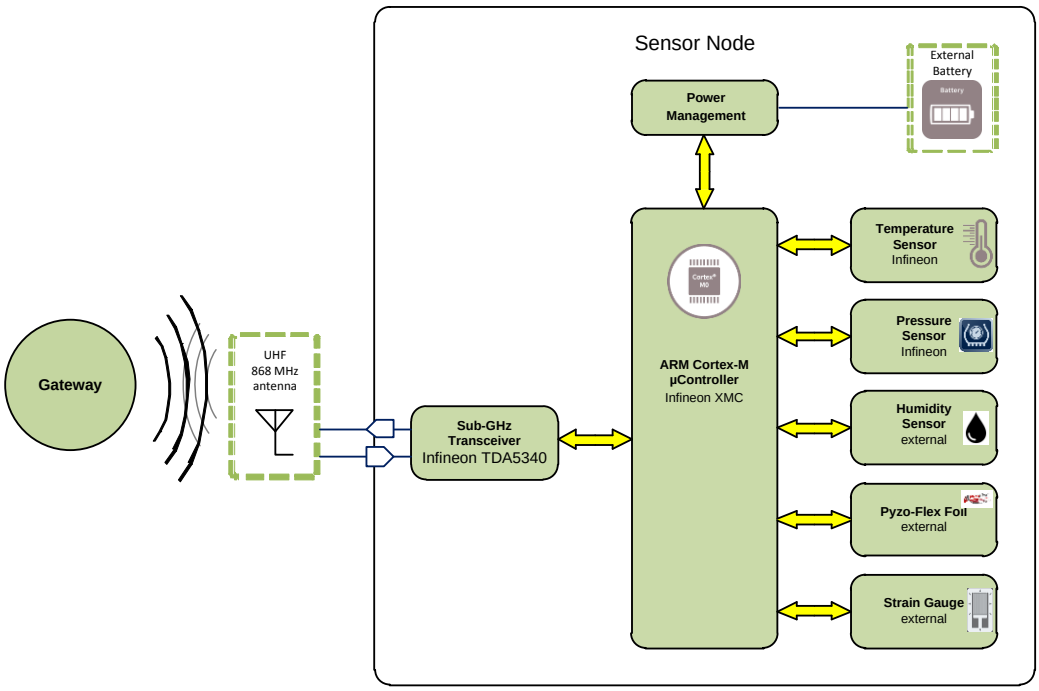


Figure 3.2: Sensor node architecture. The sensor node contains multiple sensors, a power management unit, a transceiver and a microcontroller. The antenna is used to communicate with the gateway.

3.3.2 Gateway

The gateway has to manage all incoming traffic from sensor nodes and handle protocol-specific tasks such as sending acknowledge messages. Furthermore, the gateway has to forward all the sensor data it received to the command server and process commands received from the command server. The commands sent back and forth between the sensor and the command server are listed in Section 3.4.3.

All sensor nodes within the network communicate directly with the gateway. Thus, all measured values are reported to the gateway and instructions are also received via the

gateway. The gateway communicates via Universal Asynchronous Receiver Transmitter (UART) with the command server. A common way to do this is by using a UART-to-USB converter, making the communication future-proof. Figure 3.3 shows the components of the gateway.

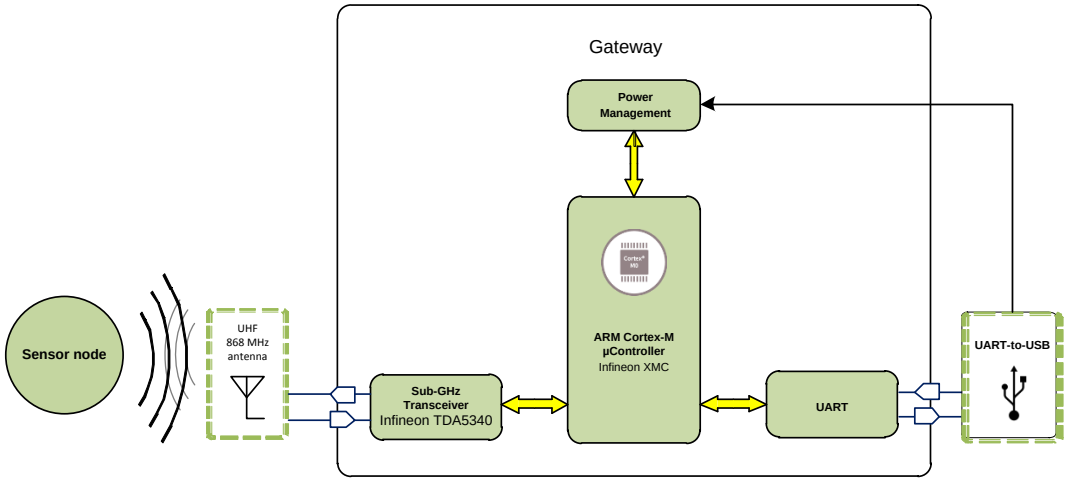


Figure 3.3: Gateway overview. The gateway is build of a transceiver, a microcontroller a power management unit and a UART interface. A UART-to-USB bridge can be used to power the gateway and serves as universal communication interface.

3.4 Network Protocol

This section describes the communication interfaces. It also discusses the messages sent within the system and those exchanged with the command server (see Figure 3.1). All sensor nodes such as the gateway are equipped with a wireless transceiver unit. It enables them to communicate with each other on a shared medium. However, similarly to Wi-Fi or Bluetooth, the communication channel is not guaranteed to be free of interference. Moreover, the link quality might vary between different participants. This type of communication is typical for WSNs and is referred to as an LLN. The communication protocol used has to be designed to tackle these difficulties and allow for the flawless operation of the whole system. Initial tests revealed (Figure 5.10) that the operation of the transceiver unit needed considerably more energy than the other components of the system. Thus, communication should be reduced to a minimum. Since the main feature of the system is the monitoring of the environment, reliable transmission is mandatory. In the fall detection use case, in particular, one has to guarantee successful transmission. The design criteria are therefore as follows:

- Minimum of communication
- Guaranteed transmission of high-priority data

- Reliable communication in general

The following two sections present the network architecture and the communication protocol design with respect to the problems discussed above and the limitations caused by the communication medium and hardware used.

3.4.1 Design Considerations

This section discusses the design of the wireless protocol. Operation of the radio consumes the most energy; thus, special attention must be paid to the design of the communication protocol.
With the main objective being to use as little energy as possible without loss of functionality, the following design decisions had to be made before the protocol could be designed.

- What does the network topology look like?
- Who starts communicating?
  There are two options. First, the base station can request data from the sensor nodes. Second, the sensor nodes can send data as soon as they are available.
- How should traffic and collisions be handled?
  Again, there are two options. The first option is to handle the collisions by resending lost data. The second is to avoid collisions in the first place by assigning dedicated time slots to the participants of the communication.

Network Topology

The topology of a network describes the relationship between nodes and is therefore a key parameter that fundamentally influences the systems operation. The choice of topology has an impact on a number of features such as overall energy consumption, number of messages sent, link reliability, number of gateways necessary, and the maximum number of nodes.
The stringent requirements as regards energy consumption limit the number of practicable network structures. The star and tree topologies comply with the requirements stated above. Another option combines the two topologies into a mesh network topology. An ad hoc decision about which topology works better is not possible, given the different characteristics of the network structures. In a later phase of the project, detailed analysis is conducted to determine the optimal choice. An adaptive topology that switches depending on the use case is possible. The following two sections elaborate upon the upsides and downsides of each choice.

Base Station Initiated versus Node Initiated Communication

Whether the sensor nodes themselves start the communication about available data or the base station polls the sensor nodes to retrieve information has a huge impact on the overall system.
In communication initiated by the base station, it decides when to retrieve data. Thus,

the time interval for measurements can be changed dynamically without great effort. Furthermore, any collisions during communication can be excluded. The main problem with this approach is that for it to function properly, the sensor nodes have to somehow listen into the communication channel to be able to respond to the base station. This means that the sensor nodes have to operate in a mode where the transceiver is fully functional as a receiver. This drains the battery fast. Even with sophisticated approaches where the sensor nodes do not have to listen all of the time, the problem of a rendezvous between two communication partners imposes significant time delays in communication, and energy consumption increases as well.
To increase the system uptime and reduce energy consumption at the sensor nodes, the nodes initiate the communication. However, bidirectional communication is then limited as the base station can only send data when the node is awake. In Section 3.4.3, these problems are discussed in greater detail, and a solution is found.

CSMA/CA versus TDMA

The main benefit of the TDMA is revealed in the fall detection use case. Because many nodes wake up at exactly the same time, they also start sending data at the same time. To reduce collisions, the TDMA media access approach could be chosen. However, as the system is used mainly in the monitoring use case, where a TDMA access approach does not add any gains in stability or reliability but adds complexity to the protocol implementation and introduces communication overhead, this approach has more impact on the energy consumption than its alternative, CSMA/CA.
Therefore, the CSMA/CA media access approach has been chosen for this project.

3.4.2 Communication between Sensor Node and Gateway

Figure 3.4 shows the communication between gateway and sensor node. The sensor nodes initiates the communication by sending data to the gateway. The gateway receives the data and replies with an acknowledge. If the gateway wants to send data to the sensor node it sets the data pending bit to one to notify the sensor node about further data transmission. From now on each message is acknowledged by the recipient and the communication stops when no further data has to be transmitted by either party.

A general problem with the acknowledge message in radio transmissions is that if the node does not receive the message but it has still arrived, then although the receiver has already noted this arrival, the sender sends the message again. The recipient then receives this message a second time. To ensure that these messages are recognized as duplicates, a sequence number is sent with each message. This allows the recipient to identify duplicate packets. This requires additional overhead when sending; however, it can also be detected on the receiving side how many messages have been lost. See also Section 3.4.3.

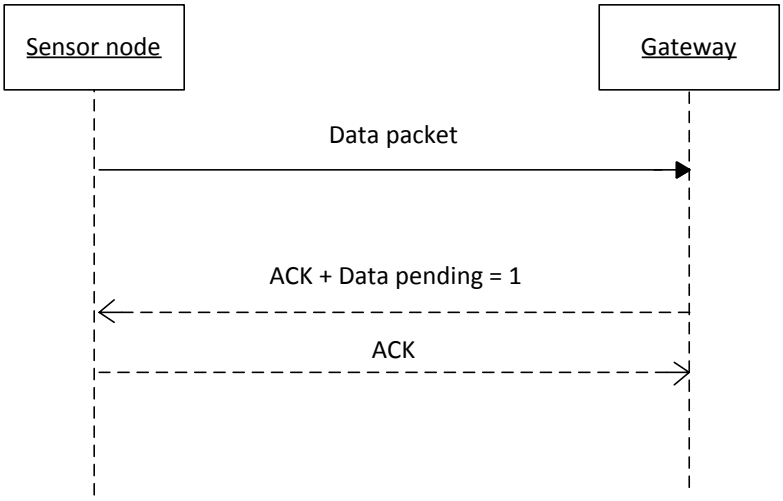


Figure 3.4: Communication flow between sensor node and gateway.

Messages

The communication between sensor nodes and the gateway involves the sending of messages back and forth. This section defines these messages and the data transmitted. The following messages are used:

- Sensor Data
  This message is sent by the sensor nodes and contains the sensor values. This is the most transmitted message type.
- Acknowledge
  This message type is sent in response to messages that request an acknowledgment. It includes a field - the data pending bit - that signals the recipient that the sender wants to transmit additional data.
- Status
  The status message contains information about the sensor nodes status. This message type provides the only possible way to monitor the health of the sensor nodes and the performance of the network.
- Beacon
  This message is a simple beacon to notify the gateway of the existence of the node sending the beacon and allow for further communication from the gateway.

3.4.3 Communication between Command Server and Gateway

This section discusses the interface between the gateway and the outer world, and is an critical part in the DeSSnet project [27]. In the final project, all data is gathered upstream and then analyzed and made accessible. All sensor data is forwarded upstream to the command server over the gateway.

Thus, one of the gateways tasks is to forward all received data from the sensor nodes to the command server. This task includes preprocessing of the received data packets from the sensor nodes to be compliant with the interface to the command server. To monitor the WSNs status, the command server can request status messages from sensor nodes or the gateway. This request has to be processed by the gateway and then be sent to the sensor nodes.
Figure 3.5 shows the common data flow between sensor node, gateway and command server.

Hardware Interface

The communication interface UART is used at a baud rate of 1,000,000 bits per second (1 megabaud). This method of communication has been chosen to reduce system load because the microcontroller offers out-of-the-box hardware support for UART communication. Reducing system load is crucial, given that the processing power of the gateway is limited. Unnecessary delays can drastically reduce overall system uptime since a busy gateway cannot handle incoming packets, leading to the retransmission of packets. When used with a USB-UART bridge, the gateway can be connected to any device on which the command server runs. This could be a Raspberry Pi [29] or any conventional PC.

Software Interface

The bidirectional communication is defined and discussed in this section. The command server was developed based on this definition. Both directions of communication are discussed in the following sections.

Gateway to Command Server

The gateway preprocesses all data received from the sensor nodes and then forwards these messages to the command server. This happens as soon as messages are received at the gateway. In addition, because the sensor nodes operate as autonomous units that wake up at predefined time intervals, messages from sensor nodes to the gateway can occur at any time.
The diagram in Figure 3.5 shows the timing and data flow of the messages sent from the sensor node to the command server.

Table 3.1 describes the communication from the gateway to the command server. Each message sent to the command server starts with a header followed by the payload referred to in the header. The following payload messages are defined and a detailed description can be found in Table 3.1, the data fields are described in Table 3.2.

- Sensor Data 1
  This message includes all measured data.

- Sensor Data 2
  As most applications do not need accelerometer readings of the sensor node, this is a version of the sensor data without the accelerometer readings.

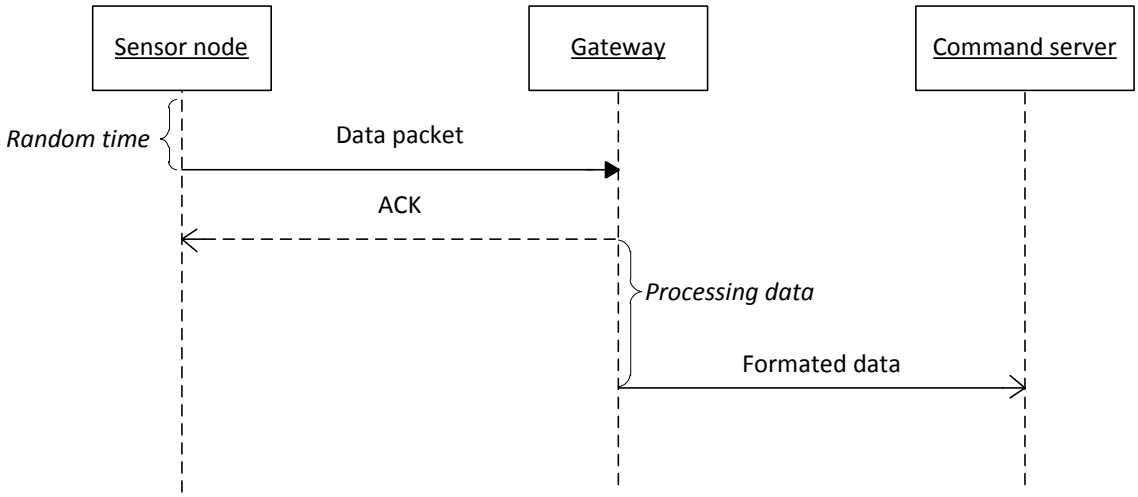


Figure 3.5: Communication from sensor node to the gateway and to the command server.

- Status
  This message includes all status information of the sensor node. As Szewczyk, Polastre, Mainwaring, et al. [11] stated, diagnostic and monitoring tools are essential for the system to run unattended. This message type, together with the implemented status logging of each sensor node, should serve as such a tool.

- OK
  This is a simple acknowledgment of commands being received by the command server. It returns information about whether the command can be processed or not

Packet Type	Format
Header	SENSOR_ID,SEQUENCE_NR,MSG_TYPE,
example	1441,190,1,
Sensor Data 1 (MSG_TYPE=1)	TEMP,HUM,PRES
example	23.4,0.3444,97739.33
Sensor Data 2 (MSG_TYPE=2)	TEMP,HUM,PRES,ACC_X,ACC_Y,ACC_Z
example	23.4,0.3444,97739.33,0.92,0.02,0.08
Status (MSG_TYPE=3)	WU,WDT,DP,NO_ACK,CBUSY,CRC
example	1455,1,344,531,40,2
OK (MSG_TYPE=4)	STATUS
example	1

Table 3.1: Communication interface from the gateway to the command server. Each packet consists of a header followed by one of the message types.

Field	Unit	Datatype	Description
TEMP	C	float16	Temperature reading
HUM	rel. %	float16	Relative humidity reading
PRES	hPa	float16	Barometric pressure reading
ACC_X(Y/Z)	$g = 9.81\frac{m}{s^2}$	float16	G-force (acceleration) in the x/y/z direction
SENSOR_ID	-	uint16	Sensor identification number
SEQUENCE_NR	-	uint8	Data packet sequence number
MSG_TYPE	-	uint8	Message type identifier
WU	-	uint16	Wakeup counter
WDT	-	uint16	Watchdog violation counter
DP	-	uint16	Dropped packet counter
NO_ACK	-	uint16	No acknowledgment received counter
CBUSY	-	uint16	Channel busy counter
CRC	-	uint16	cyclic redundancy check (CRC) error counter

Table 3.2: Detailed description of the fields within messages, including the data type.

Example

The command server receives the following message:

```
1441,190,1,23.4,0.3444,97739.33,0.92,0.02,0.08
```

This translates as a message sent by the sensor node with ID 1441. The message has a sequence number of 190, and the payload type is a SENSOR DATA packet. The node has measured the temperature to be 23.4 °C degrees Celsius, relative humidity to be 34.4 %, and air pressure to be 97 739.33 hPa. The current readings of the accelerometer are 0.92 g, 0.02 g, and 0.08 g in the X, Y, and Z axes.

The format most compatible with Comma-Separated Values (CSV) interpretations is chosen for the communication interface. This format can be found in Section 2 of [30]. Records are separated by *CRLF*; thus, each record is located on a separate line. Each field is separated by a comma, and there is no header.

Command Server to Gateway

Multiple WSNs are often operated in close proximity to each other. In such cases, the gateways of each WSN should receive only those packets that are sent from sensor nodes within their WSN. Therefore, a gateway has to be informed about all the sensor nodes that are managed by it. Moreover, the command server can request specific data from sensor nodes or from the gateway (status messages, in particular, are of interest). In this way, the overall network traffic can be kept low as data is only sent on request. Figure 3.6 illustrates one possible scenario of communication. This might not apply to another network topology. Furthermore, because the communication between sensor nodes and the gateway is initiated by the sensor nodes, the gateway has to buffer all commands from the command server for specific nodes until the relevant node initiates communication.

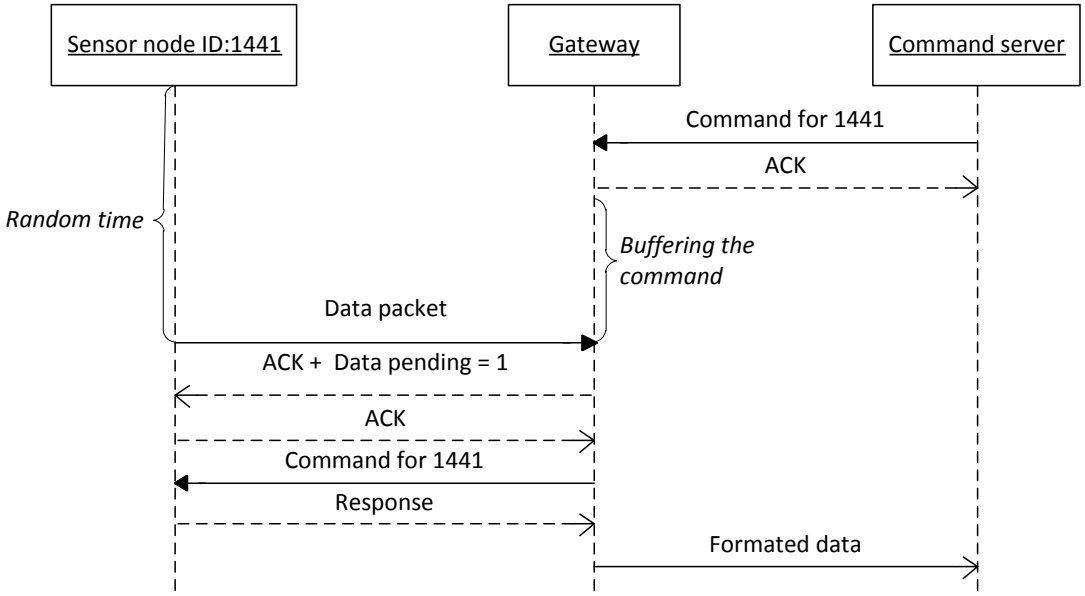


Figure 3.6: Commands sent from command server to sensor node.

The following commands are understood by the gateway:

- ADD_NODE
  The gateway is informed when a sensor node is added to the WSN. The gateway becomes ready to receive messages sent by this node also. This is necessary if multiple gateways are operated within radio range.
- STATUS_REQUEST
  This message is used to request a status message of the specified node.
- RESET_STATUS
  To request a reset of the status of the specified node, this message is used. It is especially useful for system monitoring and testing to restart the status logging period.
- CALIBRATION
  This message is used to calibrate the sensors of the sensor nodes. It transmits the correct readings for the SENSOR_DATA message. The sensor node uses this value for a two-point calibration. For a detailed description, see Table 3.3.

Packet Type	Format
ADD_NODE	MSG_TYPE,NODE_ID
example	1,1441
STATUS_REQUEST	MSG_TYPE,NODE_ID
example	2,1441
RESET_STATUS	MSG_TYPE,NODE_ID
example	3,1441
CALIBRATION	MSG_TYPE,NODE_ID,TEMP,HUM,PRES
example	4,1441,24.0,0.2,98000.0

Table 3.3: Messages sent from the command server to the gateway.

Example

The command server sends the following message via UART to notify the gateway to receive messages from the sensor node with ID 1441.

```
1,1441
```

This message resets the logging period of Sensor Node 1441. All counters of the status message from 1441 will now be reset.

```
3,1441
```

3.5 Hardware Modules

The sensor node and the gateway constitute a piece of hardware that utilizes multiple hardware components to fulfill its task. These components or modules are described in the following sections.

3.5.1 Microcontroller

The central component of the sensor node and the gateway is the microcontroller, which provides considerable functionality. For this project the XMC1100, a microcontroller for industrial applications [31], is used. It contains a 32-bit Cortex-M0 CPU , which has sufficient computing power for the tasks to be performed by the sensor nodes. One of the most important hardware peripherals for the sensor nodes and the gateway are the Universal Serial Interface Channel (USIC) modules. These modules are used not only to access the sensors and the transceiver but also to communicate with the command server and to output status messages for debugging. As the XMC1100 provides up to 64 kB of flash memory, it is perfectly suited to be the main computing unit for a sensor node. The memory can be used to buffer sensor readings to reduce the communication overhead and hence the power consumption.
Figure 3.7 displays the internal components of the XMC1100.

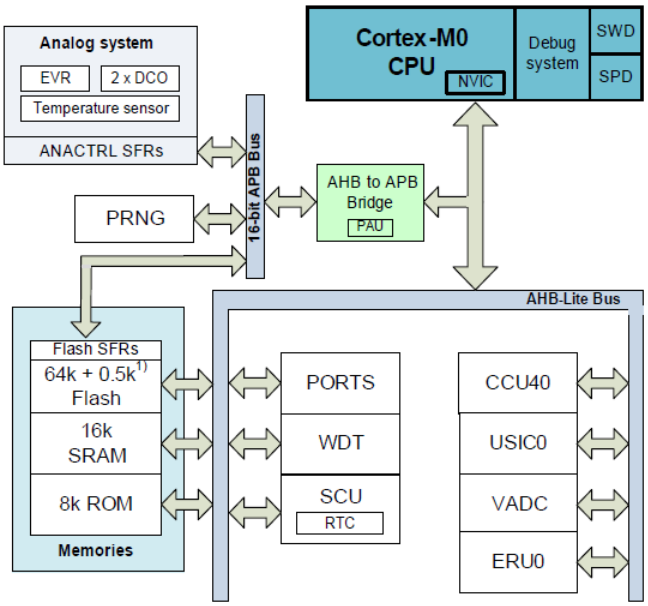


Figure 3.7: System block diagram of the XMC1100. Taken from [31]

3.5.2 Power Management

A key feature of the WSN is the average low-power consumption to enable long battery lifetime. To achieve good uptime results, an additional power management unit module is used. It consists of an ultra-low power timer and a few other components to ensure the proper functioning of the sensor node.

Timer

The TI TPL5110 [32], an ultra-low power timer, is used to power up and power down the whole sensor node in idle phases. This reduces power consumption to a minimum during idle times, because of which the current consumption in sleep mode can be reduced to 50 nA (see Figure 3.8). The configurable timer interval ranges from 100 ms to 2 hours and is set by a resistor connected to the timer chip. Because the interval is set by a hardware component, it cannot be changed while the system operates. The timer interval therefore must be wisely chosen.
The timer chip provides a pin to drive a metal-oxide-semiconductor field-effect transistor (MOSFET), which is used as a power switch. To disable the power switch, another pin is used to tell the timer chip to disable the power switch.

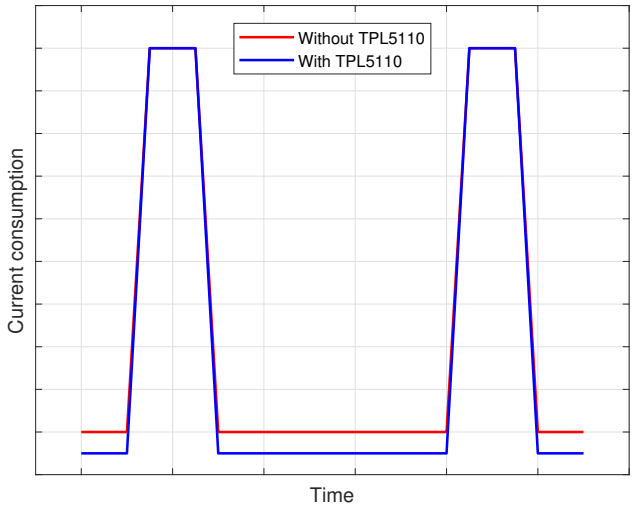


Figure 3.8: Explanatory current consumption with and without the timer chip. The timer chip reduces stand by current to a minimum. That enables battery powered WSNs with long operational lifetime. Example taken from [32]

Self-holding Circuit

A problem that can occur when the timer module provides the power supply for the microcontroller is that the timer module switches the voltage off and then on again at the beginning of each timer period. To bypass this behavior, another power switch is used in parallel to the one operated by the timer chip. This second power switch is turned on by the microcontroller. Once started up, it levers out the timer chip and its power switch. Figure 3.9 is a schematic of the keep-alive circuit.

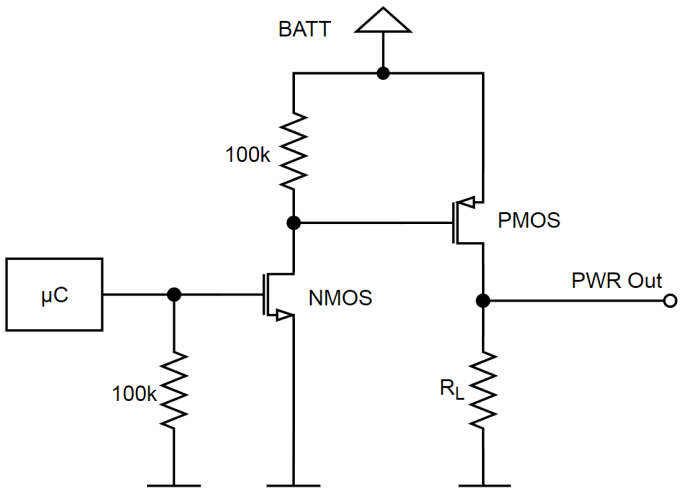


Figure 3.9: The keep-alive circuit used to bypass the timer chip.

3.5.3 Passive Wake-up

Permanent sensing of the environment is not possible with the timer chip. This means that the fall detection use case described in Section 3.1.2 cannot be fulfilled. To detect an impact or another special event, a separate module is needed to generate a wake-up signal for the timer chip. Moreover, this wake-up signal should be generated without an active power supply in order to avoid additional battery drainage.
The passive wake-up circuit is not part of this thesis. However, as this thesis is written in the context of the DeSSnet project, the option of using such a wake-up signal is provided.

3.5.4 Transceiver

A dedicated transceiver module is used in the WSN for communication. The SmartLEWIS - TDA5340 [28] fulfills the specifications with respect to the power consumption, frequency band, and communication interface. The microcontroller and the transceiver communicate using the Serial Peripheral Interface (SPI). The TDA5340 supports multiple operating modes and power amplification stages to finely tune and reduce its power consumption. This highly integrated and complex state machine simplifies network protocol implementation and reduces CPU time usage in the microcontroller. Moreover, the transceiver can scan multiple communication channels on its own and notify the host microcontroller about incoming messages.

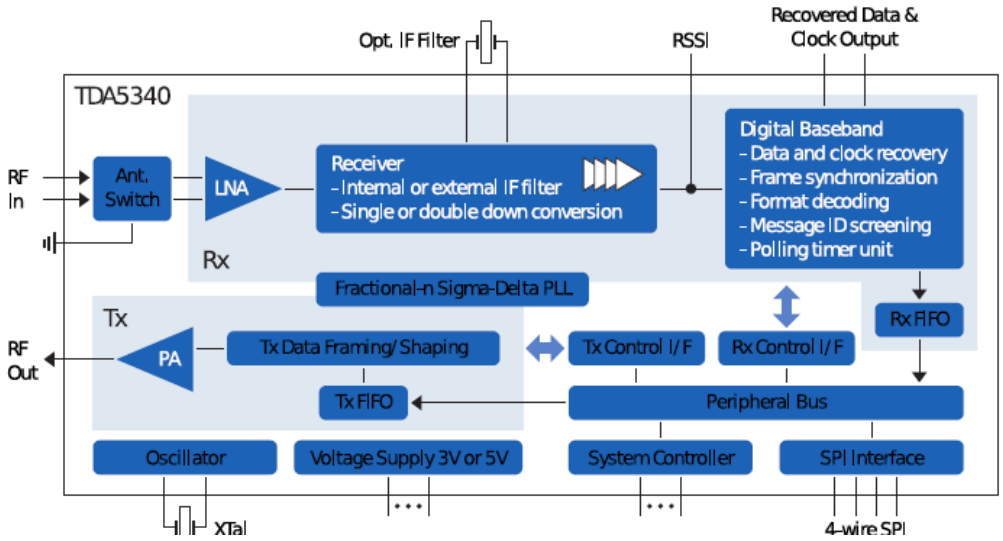


Figure 3.10: Functional block diagram of the used transceiver TDA5340 - SmartLEWIS. Taken from [33]

3.5.5 Sensors

Sensor nodes need appropriate sensors to gather information about the environment. For this thesis, the following sensors were carefully chosen as they cover a wide range of parameters of interest for many WSN applications. The Inter-Integrated Circuit (I2C) bus is used to address these sensors and retrieve their data. This design makes the easy adaptation of sensors possible, which means that the sensor node is flexible and can easily be used in various situations.

Barometric Pressure Sensor: DPS310

To sense ambient air pressure, the DPS310 from Infineon [34] is used. It is a digital barometric pressure sensor. Its key features are high accuracy and low power consumption. Because of the low power consumption, this sensor is ideal for the WSN. Its operation range is from 300 to 1200 hPa, resulting in a wide variety of possible applications. Furthermore, it can detect elevation changes down to 5 cm in the high-accuracy mode.
The DPS310 has an additional temperature sensor to eliminate measurement errors due to temperature changes. It can sense the following environmental parameters:

- Temperature
- Barometric pressure

Overall, this sensor covers most of the requirements of the WSN as regards barometric pressure.

Humidity and Temperature Sensor: HDC1080

To measure humidity, the HDC1080 from Texas Instruments [35] was chosen. This sensor was primarily chosen because of its low power consumption. Moreover, it offers high accuracy for temperature and humidity measurements. The HDC1080 features an operation range of 0 to 100 percent relative humidity, which is another reason for this sensor to be used. It can sense the following environmental parameters:

- Temperature
- Humidity

Accelerometer: LIS331

To detect orientation and vibrations, the LIS331 [36] from STMicroelectronics is used. This sensor also features low power consumption, which is why it was chosen for this project. It can sense the following parameters:

- Vibration
- Acceleration
- Orientation

3.6 Software Components

This section discusses the software modules used. Two pieces of software are used: the gateways firmware and the sensor nodes firmware. As both pieces of firmware run on the same hardware platform, some parts of the code can be shared between the two, such as the transceiver driver.

3.6.1 Design Considerations

As all hardware specifications for the sensor node and the gateway have already been determined, so has the network protocol. The question for the software design then is how to most efficiently use available resources taking into consideration the details of the planned protocol.
An examination of the illustrations of the program sequences of the sensor nodes and the gateway (Figure 3.11 and Figure 3.13) reveals that although the programs have to work in parallel, they are not particularly complex.

An operating system offers the possibility of writing concurrent programs without any problems. Because the operating system causes considerable overhead, both in the execution time and in the complexity of the entire firmware, no operating system is used in this project. The disadvantage of this approach is that the implementation effort is greater, and if necessary, workarounds have to be introduced to achieve pseudo concurrency of the program execution.
The following sections discuss the basic flow charts of the sensor node and gateway programs

3.6.2 Sensor Node

Figure 3.11 presents an overview of the sensor nodes program flow, which is relatively simple. After power-up, the sensor node takes sensor readings, processes them, and then sends this information to the gateway. If the data is sent, the sensor node puts itself to sleep. After the timer interval time expires, the sensor node wakes up again.

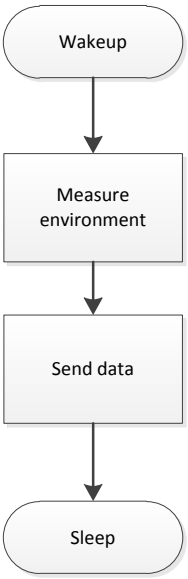


Figure 3.11: Overview of the sensor nodes program flow.

Figure 3.12 displays the basic sensor nodes program flow chart. A more detailed flow chart can be found in Section 4.2.1. After power-up, the sensor node is in the active state, where it performs the program depicted in Figure 3.11 and then transitions to the sleep state. After the timer interval expires or an external event occurs that triggers passive wake-up, the sensor node wakes up and enters the active state.
With regard to current consumption, there is a difference of at least a factor of 1,000 between the active and the sleep states. For detailed analysis of the current and power consumption, see Section 5.3.

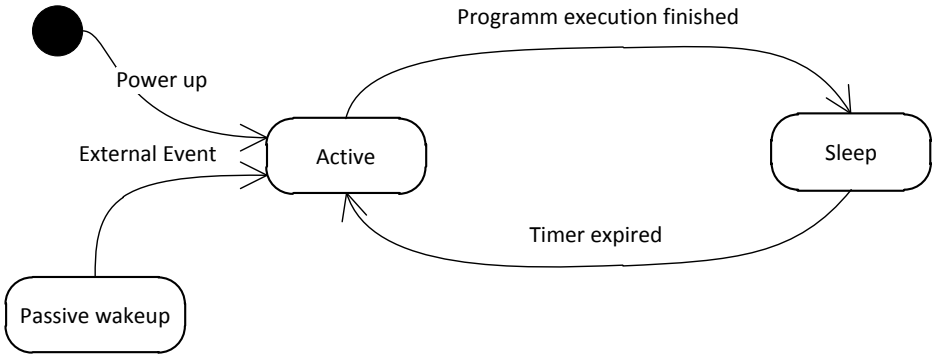


Figure 3.12: State chart of the sensor node.

3.6.3 Gateway

Figure 3.13 presents the basic program flow of the gateway firmware. As soon as the gateway is powered up, it waits for incoming transmission from sensor nodes. Immediately after data packets are received, the gateway processes the data and performs the protocol-specific data-flow logic (sending acknowledgment, additional commands, or both). After the communication with the sensor nodes has ended, the data is sent to the command server via UART.

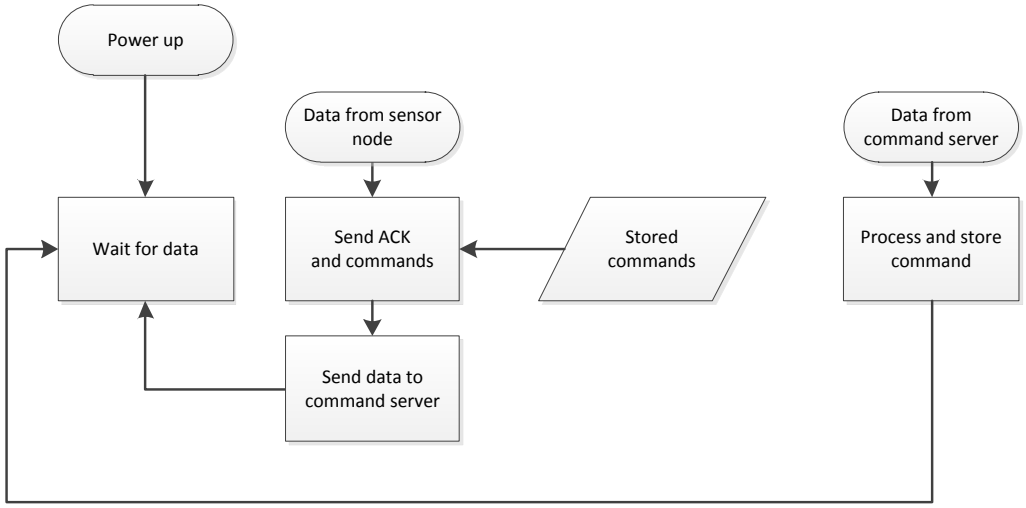


Figure 3.13: Overview of the gateways program flow. After power up the gateway is constantly waiting for data either form a sensor node or from the command server.

Chapter 4

Implementation

This chapter discusses the implementation of this project. The final WSN prototype underwent several prototyping and testing steps, which allowed for continuous improvements. Initial attempts were made using existing demonstration boards of the XMC and transceiver chips put together on breadboards. Later, the design was refined over several versions of the PCB to improve stability and add functionality.
The software was similarly developed at the same time. Device drivers were written first, followed by the implementation of more complex system behavior. Finally, the communication protocol was written. All these steps were tested comprehensively throughout development. The software-based tests were combined in a test suite used to test each newly assembled sensor node. The insights gained in each prototyping phase were used in subsequent ones. This procedure ensured high quality of all versions of the sensor node throughout the development process.

In this chapter, the development of the hardware is discussed, followed by the software development and protocol implementation.

4.1 Hardware

A large part of this project was the design of the WSN hardware. A fully functional WSN needs at least two parts, a sensor node and a gateway. As the gateway and the sensor node share many hardware components such as the transceiver and the microcontroller, a hardware platform was created that could function as either a sensor node or a gateway. The created platform could be assembled in different configurations, which together with the software used, affected the later role of the platform.
The decision to create a single hardware platform reduced development and testing efforts significantly. To further reduce testing efforts, special focus was placed on the testability of the platform. This included the use of additional hardware components that were not necessary for the functioning of the system, such as Light-Emitting Diode (LED) or test pins.
Additionally, at an early stage of the project, it became clear that a dedicated test platform would greatly speed up the development process. This additional testing platform was used extensively throughout the development cycle.

4.1.1 Additional Hardware Components

An initial test using a breadboard revealed that additional hardware components would make software development and debugging easier. Thus, the hardware design incorporated these components and also aimed to fulfill the requirements in order to make testing and verification as easy as possible.
The following components were included in the hardware design:

- Light-Emitting Diodes (LEDs)
  Two LEDs were used as status indicators on the gateway and the sensor nodes. The LEDs could be freely programmed to indicate whatever might be useful in further stages. In the actual version, they were used to indicate received and sent messages. A power LED was also placed in the PCB. This LED was especially useful to monitor the timer interval or the actual state of the sensor node (see Figure 3.12) as it was enabled when the node was in active mode.
  Moreover, as these LEDs drew about 1 mA of current each, they could easily be disabled with solder jumpers. With this method, no firmware had to be updated for power efficiency, which further reduced development and testing efforts.

- Button to manually wake the sensor node up
  At the time of development of this prototype of the sensor network, there was no functioning passive wake-up circuit available. Therefore, to simulate passive wake-up, an additional button was built in, using which, passive wake-up could be triggered. It turned out that this button was very useful for software testing in general since it allowed for the triggering of communication from the sensor nodes to the gateway. As communication was otherwise only initiated by the sensor nodes on a wakeup of the timer chip (see Section 3.4.1), this feature came in handy.

- Additional pull-up resistors
  The accelerometer [36] provided built-in pull-up resistors. To be able to omit this sensor, as it might not be used in all applications, pull-up resistors were soldered onto the PCB to still be able to use the other sensors.

- Test Pins and Connectors
  The first tests demonstrated that without appropriate test pins, testing the hardware and software was difficult. Therefore, to be prepared for future tests, almost all the wires–namely, I2C Clock and Data, SPI Data IN, Data Out, Clock and Chip Select, TDA Interrupt and Power On, Timer Power Off, and UART Transmit and Receive Lines such as the Serial Debug Lines and the Passive Wakeup–were equipped with test pins and pads.

- Additional trim potentiometer for variable timer intervals
  The timer chip generally uses a resistor to determine the timer interval. As tests might have required the timer interval to be changed, the hardware design used a trim potentiometer so that the timer interval could be changed manually. To save costs in the final application the trim potentiometer can be replaced by a resistor.

4.1.2 PCB Design

Size played an important role in the PCB design of the sensor node. The prototype created was intended to be built into the structure of buildings. A small footprint was therefore particularly important. The height of the prototype also had to be kept as low as possible. In addition, an attempt was made to select the simplest design with the lowest complexity in order to comply with the specifications.

This section discusses the PCB design completed in this project. This design was created in EAGLE [37].

4.1.3 Sensor Node PCB

For a special application of the sensor nodes in parquet floors, a maximum height of 5 millimeters must not be exceeded. To cover this case, it was decided to equip the PCB with surface-mounted device (SMD) components only on one side, since the component height was the limiting factor. For this purpose, the battery had to be mounted laterally next to the PCB. However, this reduced stability. Therefore, a battery clip was provided to fix the battery to the underside. This allowed the PCB to be assembled and manufactured in several versions.

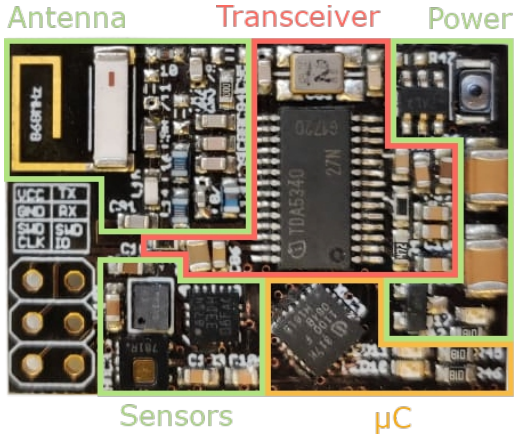
Figure 4.1 shows the final version of the sensor nodes PCB design. This was the second PCB design of the sensor node platform. The first version had some reliability problems (discussed in Section 5.6.1), because of which a redesign was necessary. The sensor node hardware platform consisted of five modules sectioned in five assembly groups: power supply and timer, transceiver, antenna, sensors, and microcontroller. The five hardware modules are described in more detail in the following sections. Figure 4.1 displays the assembled version with the different modules highlighted.

Antenna Module

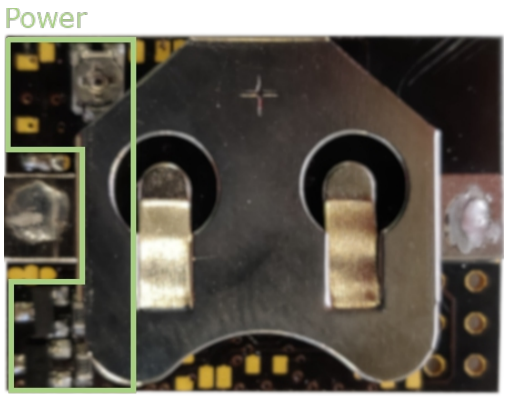
As the SRD860 frequency band was used for the WSN, an antenna operating in this frequency was necessary. To reduce the footprint as much as possible, a ceramic antenna element together with a small PCB antenna was used. As with all wireless devices, the signal range played a key role and should ideally have been as large as possible. With increased signal range, the transmission power could be reduced to achieve the same transmission quality or distance. To minimize the negative effects of signal reflections and power dissipation loss between the transceiver and the antenna, a matching network was used. By matching the input impedance of the antenna with the output impedance of the transceiver, the signal power transferred was maximized. This ensured that the transceiver and the antenna that were used worked well together. Moreover, the matching network was developed by an expert to guarantee the best results.
Figure 4.1 displays the antenna module on the upper left quarter of the PCB.

Transceiver Module

The transceiver was placed right next to the antenna. The feed lines from the transceiver to the matching network and the antenna were kept as short as possible to avoid the emer-



(a) Front: The five core modules on the sensor node are the antenna together with the matching circuit, the transceiver with crystal, the power circuit, the sensors, and the microcontroller.



(b) Back: On the back of the sensor node, the battery clip for CR2032 form factor batteries together with the rest of the power circuitry is mounted.

Figure 4.1: Sensor node in its stripped-down version. The pin connectors for use with the connector board are removed.

gence of additional parasitic resistance, capacities, and inductivities. The assembly group consisted of the transceiver chip (TDA5340 [28] from Infineon) and its wiring, including decoupling capacitors and a clock crystal.

Power Supply Module

The power module, which powered the sensor node, consisted of three submodules: the battery support, the keep-alive, and the timer circuit. By providing a periodic wake-up signal, the timer chip made it possible to power off the rest of the sensor node during inactive phases. This reduced the power consumption of the sensor node to a minimum. The main part of the timer circuitry was the timer chip (TexasInstrument's TPL5110 [32]) together with a MOSFET used as a power switch. The wake-up signal generated by the timer chip was used as the power-on signal for the MOSFET. To set the timer interval, a resistor or a trim potentiometer could be used, depending on the hardware configuration. The keep-alive circuitry was used to prevent the power MOSFET (switched on by the timer chip) from turning the power off during the sensor nodes operation, thus bypassing the timer chip and power MOSFET. The keep-alive circuit was operated by the microcontroller and used an NMOS and a PMOS.

The battery support circuit was used to limit the voltage drop when the timer chip turned on the power for the other modules because the decoupling capacities added up and led to peak currents when supplied with power. For this reason, two capacitors were inserted in parallel to the battery (e.g., on the input side of the power module). In Figure 4.2, C_{OUT} denotes the decoupling capacities and C_{IN} denotes the battery support capacities.
For a more detailed explanation about this problem, see Section 5.6.1. All the components for the power module were low-leakage, high-efficiency components to keep the

current leakage as low as possible and ensure a long battery lifetime.

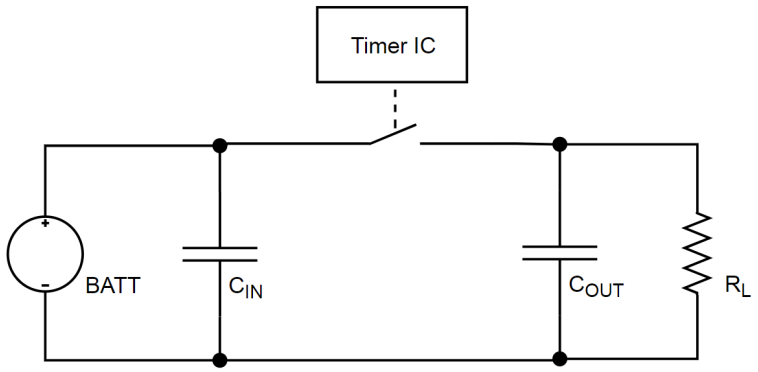


Figure 4.2: Schematic depiction of the distribution of capacitors on the input and output of the timer module.

Power Supply Issues with First Version

In the first version of the sensor nodes, the PCB behaved oddly in some situations. The nodes constantly powered on and off. At first, the error was thought to lie in the configuration or the wiring of the timer chip because the timer chip seemed to not be functioning correctly. However, even after every component of the timer module had been replaced, the error still persisted. Measurements conducted did not indicate any obvious errors. A voltage drop on the input side of the sensor node (i.e., the battery) was expected and was within 0.2 V. It was finally discovered that the two-layered PCB design provided an insufficient ground plane. The introduced parasitic resistances led to a position-dependent voltage drop. This voltage drop increased drastically farther away from the power switch. Since the first measurements were conducted directly at the battery, the voltage drop was about 0.2 V. When measured at the input of the timer chip, a voltage drop of about1.0 V was observed. Such a huge voltage drop caused the timer chip to reset. After the timer chip recovered, the timer cycle was started again, causing another power-on signal, leading to another voltage drop, and so on. This was the cause of the fast toggling.
The first idea was to fix the problem by placing additional decoupling capacitors on the battery side to reduce the voltage drop. However, it also turned out that the decoupling capacitors at the output of the power switch were draining a considerable amount of current when turned on. Therefore, the capacitor on the battery side had to be placed very close to the power switch in order to reduce the voltage drop.
While this fixed the on-off toggling problem, the actual problem of the insufficient ground connection along the sensor nodes PCB still persisted. This problem was fixed in the second version of the PCB, where care was taken to ensure that the ground plane was as continuous as possible. Despite the better ground connection, the support capacitors were still included in the second version to further improve stability. As a result, the voltage drop on power-on was limited to 0.1 V in the second version.

Microcontroller Module

The microcontroller was placed as centrally as possible on the PCB to be able to route all the wires without the need of a multilayer PCB, thus keeping production costs low. The microcontroller module also contained two status LEDs, which could be disabled by using solder jumpers.

Sensor Module

The sensor module contained the three sensors (Infineon's DPS310, Texas Instrument's HDC1080, and LIS331 from STMicroelectronics) and the decoupling capacitors required for each sensor. Special care was taken to arrange the sensors in such a way that interferences were kept to a minimum. The temperature sensor was thus placed as far as possible from all the other components to reduce the number of corrupted readings from self-heating of the other components.

Pinout

For the purposes of testing, most signal lines were also connected to pinouts. With these pinouts, the sensor nodes could be connected to the connector board (see Section 4.1.4). The connectors located on the top and bottom of Figure 4.1 could be detached once the development and testing phases were completed. Before deployment, the nodes could thus easily be tested and programmed by using these connectors. This allowed for comfortable development and debugging. Also, the pinout matched the connector board so that each sensor node could be mounted on top of a connector board, where all the equipment was set up. This sped up the validation process of the sensor nodes.

4.1.4 Connector Board PCB

The connector board was designed mainly for test purposes. It featured a pinout for all the relevant parts of the sensor node (such as the SPI and I2C communication pins, debugging interface, UART interface, power supply, timer manual-enable pin) and additional power supply with AA batteries. A dedicated connector for debugging and programming together with a connector for the sensor extender board was also included. The sensor node board was mountable onto the connector board. Once the test equipment was set up to measure and test certain characteristics, it was easy to test multiple nodes by plugging them to the connector board. This was also the case for debugging and programming the microcontroller. The unique pin layout prevented wrong mounting of the sensor nodes.
In addition, the batteries used with the connector board provided about 20 times as much energy as the battery used in the final application. This reduced the number of times batteries had to be changed while still making portable testing possible.
Furthermore, as the fall detection use case required a passive wake-up (see Section 3.5.3 and Section 3.1.2), which was under development and not available, the connector board provided a dedicated pin for simulating the fall (see Section 5.3.4).
The connector board also featured dedicated power consumption and current measurement pins. Figure 4.3 displays the top of the connector board. The footprint of the sensor node is highlighted on the left of the connector board.

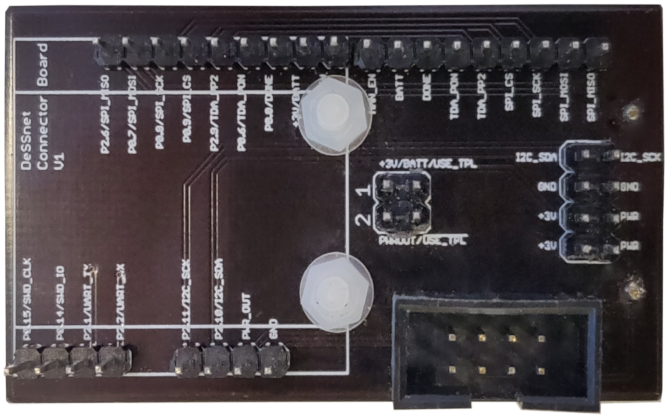


Figure 4.3: The connector and battery pack board. The sensor nodes are mountable (with pin connectors not removed) onto the connector board. On the left, the outline of the sensor nodes is printed on the PCB. It offers pinouts for most signal wires on the sensor nodes (top right), such as a dedicated connector for programming and debugging (bottom right) and a connector for the sensor extender (right).

4.1.5 Sensor Extender PCB

A separate set of sensors was mounted on the sensor extender. This is because a major drawback of WSNs is that once fitted, they are hard to replace or maintain. Since this project was meant to provide meaningful insights into the possible use cases of WSNs, it was thought useful and even necessary to replace the sensor node in a future stage of the project. With the sensor extender, the sensors could be deployed in their designated place with the sensor node still being replaceable (for a redesign or software update for example).
The connector board was designed such that the sensor extender could be used with a connector. When a sensor node was operated with the sensor extender, no sensors were allowed to be soldered onto the sensor node as this would have caused collisions on the I2C bus.
Multiple versions of this extender exist, each with different sensors mounted. Figure 4.4 displays a version with the HDC1080 humidity sensor and the DPS310 barometric pressure sensor.

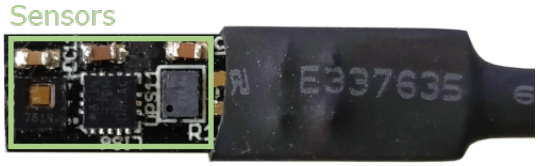


Figure 4.4: Sensor extender board. The sensor board is an additional board designed for testing purposes. It can be used together with the extender board and a sensor node without mounted sensors.

4.1.6 Gateway PCB

The gateway was based on the same hardware platform as the sensor node. For the gateway to be fully functional, it required only the transceiver modules and the microcontroller. The other modules could be omitted (sensor and power modules). To be able to communicate with the command server, the UART interface was designed such that it could be directly connected to the command server. The connector board offered a dedicated connector for the debugger, so that it was easy to use the gateway together with the debugger as a UART-to-USB bridge and as a power source.
On the left side of Figure 5.1 the typical gateway setup is illustrated. The node used as gateway is mounted on a connector board. The red board is the UART-to-USB bridge that also supplies the gateway with power.

4.2 Software

The entire software development was done in DAVE [38], which is an Eclipse-based Integrated Development Environment (IDE). It provides additional support for configuration of Infineon's XMC [31] microcontroller family by allowing for the mostly GUI-based setup of the microcontrollers dedicated hardware components, including pin configuration.
This section discusses the software implementation of the system components. State diagrams are used to describe the software. More complex parts are then split up into detailed state diagrams. For both the gateway and the sensor node, a super loop architecture was used. This implies that different programs we wanted to run on the nodes were not executed in parallel but in a loop. This was sufficient as the tasks each node had to perform were simple enough to not require the use of an operating system. Another consideration about the architecture was that an operation system introduces overhead, which we wanted to avoid, since power consumption had to be reduced to an absolute minimum. (For more details, see Section 3.6.)
The following two sections explain the sensor nodes and the gateway software by using flowcharts.

4.2.1 Sensor Node Flow Chart

This section describes the program flow of the sensor node. The flow chart in Figure 4.5 presents a rough overview of the program flow.
The whole program cycle begins with a wake-up. This wake-up, which powers up all the hardware modules of the sensor node, is triggered by the timer chip (Section 3.5.2), passive wake-up (Section 3.5.3), a press of the wake-up button (Section 3.5.2), or power-up of the sensor node.
After the microcontroller is successfully started, the configuration and status information is loaded from the internal flash of the XMC. This is necessary because in sleep mode, the microcontroller is powered off and loses all data stored in RAM. Some values, such as the node ID or the random number seed, are loaded in this phase.
After successful startup and loading of the configuration, the sensor node begins taking measurements. Each sensor is configured accordingly and then read.
All the gathered sensor data is then added to a data packet and sent to the gateway using

the transceiver module. The transceiver is configured such that after sending the data packet, it listens for incoming packets to receive the awaited acknowledge message. (This procedure is discussed in greater detail in Section 4.3.1.)
The received acknowledge message is processed to check whether the gateway wants to send any commands. If so, the transceiver continues listening; otherwise, the sleep state preparation routine is initiated. After reception and processing of the command as well, the shutdown routine is initiated. This shutdown routine saves the current system status and enables the sleep mode by signaling the timer chip to turn the power off.
Once the sensor node is in the Sleep state, it remains in this state until a wake-up is triggered.

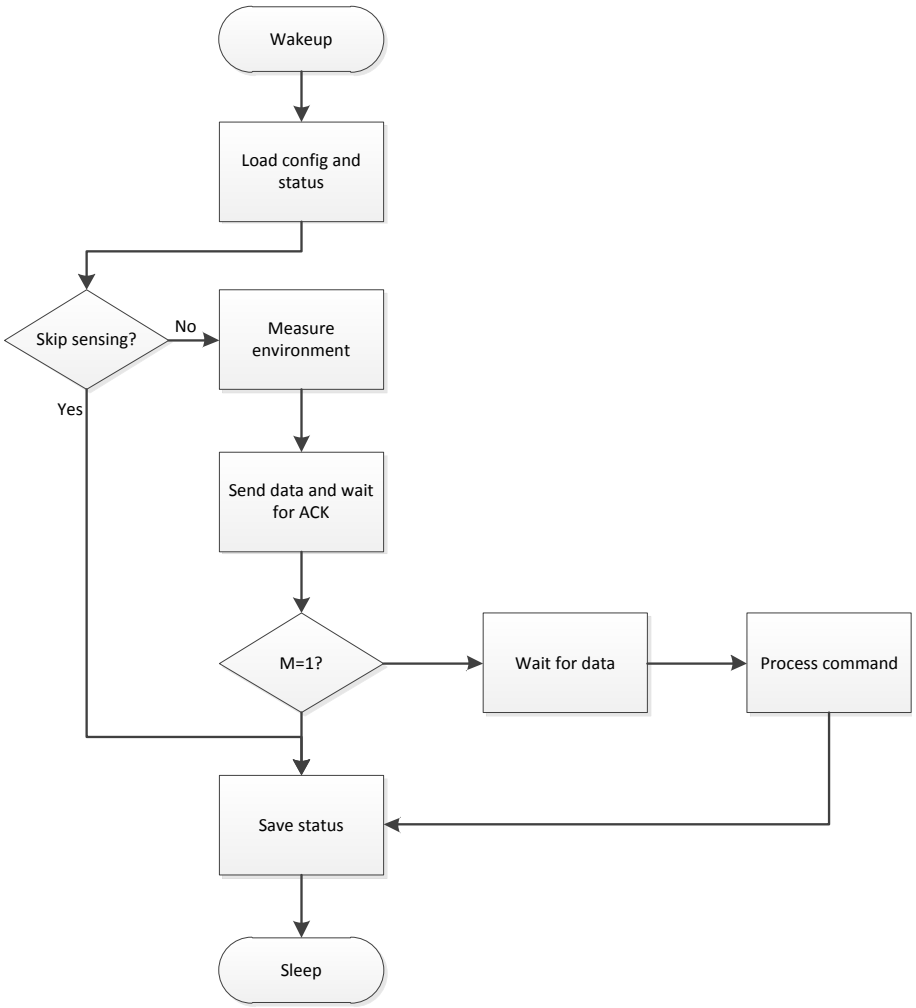


Figure 4.5: Program flow diagram of the sensor node. After wake-up, the sensor node takes measurements of its environment and transmits the data to the gateway. If the data is sent successfully, the sensor node goes back to sleep and waits for another timer period to wake up again.

4.2.2 Gateway Flow Chart

This section outlines the program flow of the gateway. When the gateway is connected to a power source, it starts executing the gateway software program.

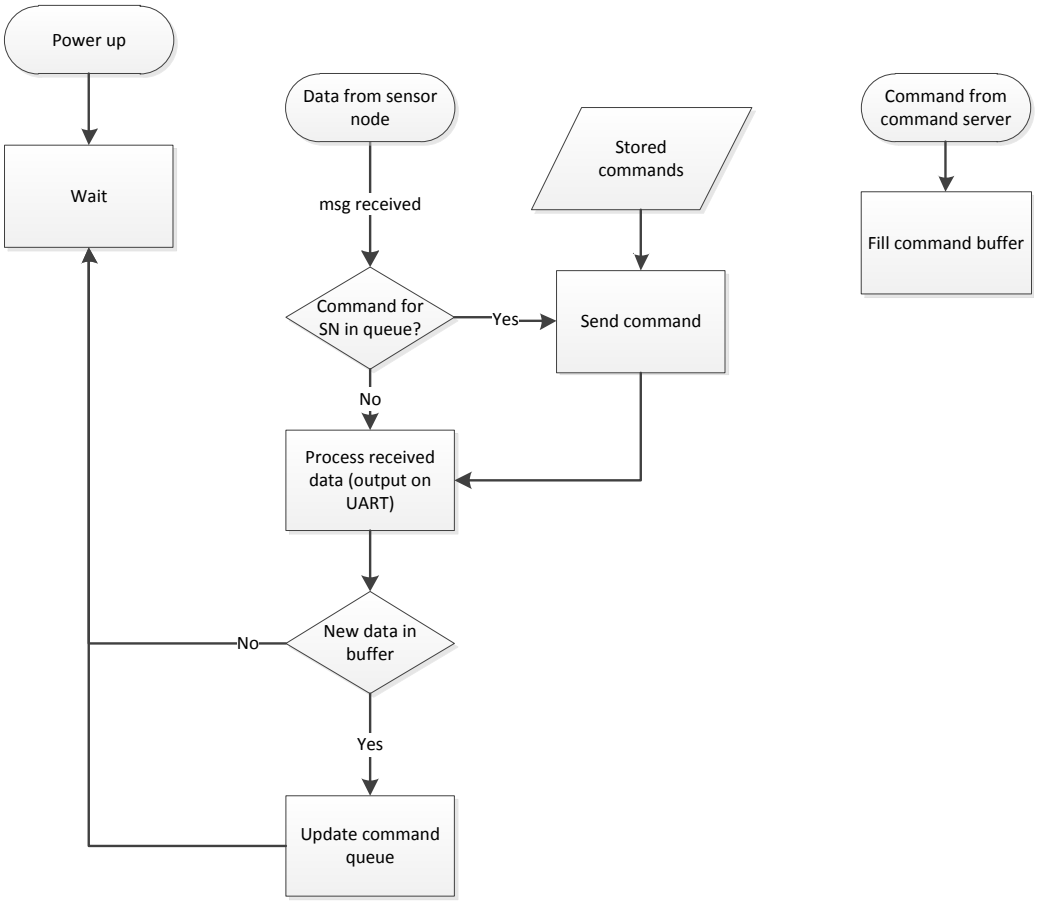


Figure 4.6: Program flow of the gateway. After power-up, the gateway listens for incoming data from sensor nodes and the command server. When a data packet from the sensor nodes is received, the gateway outputs the received data to the command server.

The transceiver is configured to listen into the channel and wait for a packet to arrive. If a packet addressed to the gateway is received, the gateway processes the incoming data. Before sending an acknowledge message, the gateway checks in its internal command buffer whether there is a command to send to the sensor node. If such a command is found, the sensor node is notified by the data pending (D) bit in the acknowledge message. This is depicted in Figure 3.6. However, if no command is found, the acknowledge message is sent back. Finally, the received data is sent to the command server via the UART interface. Before listening for another packet from the sensor nodes, the gateway processes all

buffered commands from the command server and maintains the command queue.

The communication from the command server is handled within an interrupt service routine as the commands can occur at any time. This routine is designed to take as little time as possible in order to cause the least delay in the execution of the main program. This is especially important as reception of the packets from the sensor nodes is time-critical. With this design, it is possible to receive commands from the command server and receive data packets from sensor nodes at the same time.

4.2.3 Configuration of Hardware Modules

The drivers for the sensors and the transceiver are part of the firmware. This section briefly discusses the configuration used and operation modes of the hardware components.

Humidity and Temperature Sensor: HDC1080

The HDC1080 is configured to read temperature and humidity at 14-bit resolution. Each measurement takes about 6.5 ms using 14-bit resolution. The resolution for the temperature readings can be reduced to 11-bit, and the resolution of the humidity readings can be reduced to 8-bit.
Because the temperature and humidity readings are important parameters to monitor, for this project, the maximum resolution was chosen.
Furthermore, the HDC1080 offers a heating element to avoid incorrect humidity measurements in excessively humid environments. As such environments are not planned to be monitored yet, the heating element was not used in this project.

Barometric Pressure and Temperature Sensor: DPS310

The DPS310 is capable of measuring both air pressure and temperature. The sensor is calibrated, which means that calibration coefficients are stored on the sensor. These calibration coefficients must be used for conversion of 24-bit measurements to high-accuracy pressure and temperature values. Thus, the coefficients are read first, then the measurements are performed, and finally, the pressure and temperature values are calculated.
However, a drawback of this sensor is that the measurement time is quite long. A high-precision pressure or temperature measurement takes about 105 ms. A huge part of the active time of the sensor nodes is thus due to the DPS310. (Figure 5.10 and Table 5.3 provide a breakdown of the sensor nodes active time.)
Also, the current configuration uses standard precision modes for the measurements to reduce the measurement time as much as possible without losing much information.

Accelerometer: LIS331

The accelerometer offers multiple power modes including several low-power modes. At startup, the accelerometer operates in power-down mode. Before carrying out measurements, the desired power mode must be set. The so-called normal mode offers multiple high sampling rates that have to be set after switching to this mode. In contrast, the low-power modes offer low sampling rates with reduced accuracy. As this project focused on

low power consumption, a low-power mode was selected such that a sample was generated every 2 seconds.
The accelerometer also offers dynamic sensitivity ranges from ±2 g to ±8 g. In this project, as the sensor node was designed to measure vibrations, the lowest sensitivity range was selected.
Once the measurements start being taken, the status register is polled to check for a valid measurement, after which the accelerometer readings for all three axes are read. Finally, the accelerometer is put back into power-down mode.

Transceiver: TDA5340

The transceiver offers multiple modulation schemes, such as shaped and non-shaped Amplitude Shift Keying (ASK), including On-Off Keying (OOK); Frequency Shift Keying (FSK); and Gaussian Frequency Shift Keying (GFSK).
Each modulation scheme has benefits and drawbacks. As FSK is more resilient to interferences, it was chosen as the modulation scheme for this project. To reduce the occupancy of the spectrum, the GFSK version was used. With a frequency of the oscillator of

$$f_{xosc} = 21.948\,717\,\text{MHz} \tag{4.1}$$

the parameters can be calculated as follows.
To transmit in the 868 MHz band, the TDA5340 must first be configured. To do this, the values INT and $FRAC$ must be calculated according to the following formula, which can be found in the data sheet [28]. The following equation determines the parameters for the sender.

$$\begin{aligned} f_{RF} &= f_{xosc}\left(INT + \frac{FRAC + .5}{2^{21} - .5}\right) \\ &= 21.948\,717\,\text{MHz}\left(39 + \frac{1149026.5}{2097151.5}\right) \\ &= 868.025\,635\,\text{MHz} \end{aligned} \tag{4.2}$$

The down conversion results in two images that occur. As the and the uppper one was used the following parameters have to be chosen for the receiver.

$$\begin{aligned} f_{RF} + f_{IF} &= f_{xosc}\left(INT + \frac{FRAC + .5}{2^{21} - .5}\right) \\ &= 21.948\,717\,\text{MHz}\left(39 + \frac{1175240.5}{2097151.5}\right) \\ &= 868.299\,990\,\text{MHz} \end{aligned} \tag{4.3}$$

The used configuration parameters for the radio frequency of the transceiver for sending and receiving can be found in table 4.1 and table 4.2.

Config:	RX	TX
INT	39	39
FRAC	1149026	1175240

Table 4.1: Configuration values for the TDA5340 transceiver.

The used frequency deviation for the FSK is calculated with the following formula. The chosen configuration values are listed in table 4.2.

$$\begin{aligned} \pm f_{deviation} &= FDEV \frac{f_{sys}}{2^{21}} \cdot 190 \cdot 2^{FDEVSCALE-6} \\ &= 12 \cdot \frac{21948717}{2097152} \cdot 190 \cdot 1 \\ &= 12 \cdot 10,4660 \cdot 190 \cdot 1 \\ &= 12 \cdot 1988,53313 \cdot 1 \\ &= 23\,862\,\text{Hz} \\ &= 23.862\,\text{kHz} \end{aligned} \tag{4.4}$$

Config:	TX
FDEVSCALE	6
FDEV	12

Table 4.2: Configuration values for the TDA5340 transceiver.

4.3 Protocol Implementation

In the design phase, the higher levels of the network protocol were defined and can be found in Section 3.4. This section focuses on a lower level of the protocol and its technical implementation.
The sending of messages is described in detail, followed by a short description of message reception. The data packets used for communication are then discussed at the end of this section.

4.3.1 Sending of Messages

This section describes the program flow of the send procedure for the sensor node and the gateway. Figure 4.7 presents the flow chart of the procedure.
Before sending, the sensor node listens into the channel to ensure that there is no communication taking place. If there is no activity on the channel, the transceiver is put in send mode and the data is sent. When the CCA fails, the sensor node assumes that communication is ongoing, waits for a random length of time, and then again tries to access the channel. To guarantee the reception of the data, each packet has to be acknowledged by the receiver. Therefore, after sending the data, the sender waits for the acknowledge message from the receiver and thereby puts the transceiver in reception mode. As an ac-

knowledge message can only appear within a short time frame, after sending the message, the sender waits for this time period. If no acknowledge message is received within this period, the data packet is assumed to be lost.
As a result, the sender starts over with the CCA and tries another time. This process is followed five times until the sensor node either drops the packet or stores it in the flash.

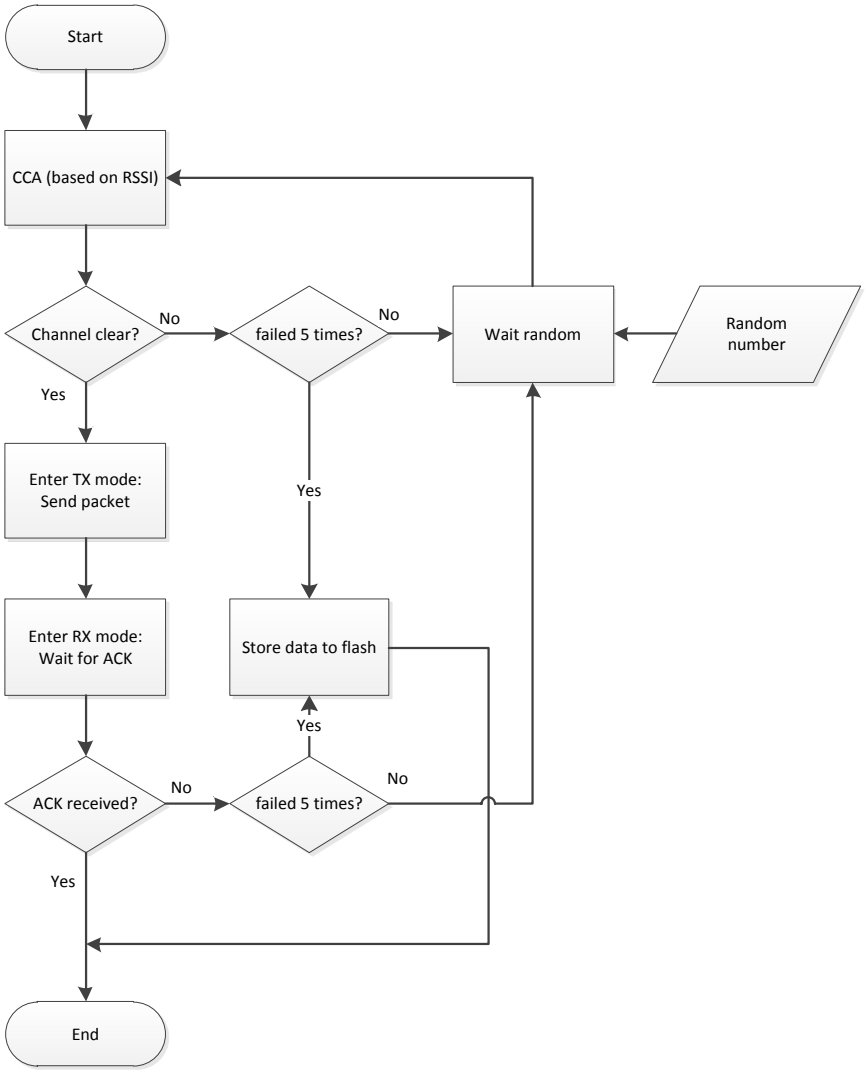


Figure 4.7: Transmission flow chart. After the CCA, the data packet is sent and the acknowledge message is received. To increase the robustness of the protocol, random waiting times and feedback loops are introduced.

Channel Clear Assessment (CCA) (based on Receive Signal Strength Indicator (RSSI))

For the CCA, the transceiver is put into reception mode to listen for any activity on the channel. The RSSI value is an indicator of the current activity. When in reception mode, the transceiver is set to automatically log the highest RSSI value since the previous read-out.
After switching to reception mode, the transceiver reads the RSSI value and resets it. It then waits for at least the inter-packet time (the longest time between two consecutive data packets) and reads the RSSI value again. This value indicates whether any communication went on during the listening period.
The channel is assumed to be clear when the RSSI stays below a certain threshold. As the RSSI value is unit-less and not a linear scaled value of the signal energy on the channel, it is not trivial to define thresholds that work in all situations. The RF pollution from nearby channels or devices operating in the same band might vary considerably among operating sites.

As a result, fixed values for the CCA, even if chosen carefully, are problematic and, in the worst-case scenario, could render the sensor node useless. To counter these problems, each sensor node keeps track of the noise-floor RSSI at its site and adapts the threshold accordingly. If the measured RSSI value is within a certain range of this noise floor, the channel is assumed to be clear.
To increase robustness and avoid wrong decisions based on temporary effects, the measurements of the noise floor are exponential smoothed. It is crucial that the initial average be set to the maximum possible RSSI value; otherwise, the channel will be considered busy until the correct value has been reached by smoothing.
The following equation shows how the internal value for the noise floor is calculated from actual measurements taken of the RSSI.

$$RSSI_{avg}[t] = RSSI_{avg}[t-1] \cdot 0.91 + RSSI[t] \cdot 0.09 \tag{4.5}$$

Wait for ACK

The time between the sending of a data packet and the reception of the acknowledge message should be as short as possible. This is because the longer the receiver of the sensor node has to be active and listen into the channel, the more energy is consumed. This imposes some restrictions on the reception routine of the receiving node as acknowledge messages have to be sent as fast as possible. To avoid such problems, the receiving and sending routines in this project were carefully written with hard timing constraints.

Binary Exponential Backoff

For the waiting routine, the binary exponential backoff algorithm (see [39] for more details) is used. Depending on the number of times the CCA fails, the number of possible waiting times increases exponentially. The internal random number generator of the XMC is used to define one of the possible waiting times, called slot times, at random. This is the time the sensor node waits until it again attempts to send the data packet.

The fall detection use case (see section 3.1.2), in which multiple nodes want to send simultaneously, might especially require this technique. With the use of a random number, it is very unlikely that all nodes that detected a busy channel would wait for the exact same time and cause a collision by sending their data simultaneously.

4.3.2 Message Reception

The counterpart of message sending is the reception of messages. Compared to the sending procedure, the receiving procedure is relatively simple. Once a message is received, it is checked that the data message is complete and does not contain any errors. This is done by a CRC checksum check.
If the message is complete and without errors, the recipient is matched with the receiver ID. If they match, then an acknowledge data packet is created and sent immediately. The only crucial part is that the acknowledge message has to be sent within 5 ms of packet reception as this is the time period the sender waits for the message.

4.3.3 Packet Outline

This section describes the packets used with the wireless protocol created in this project. All packets follow the basic structure presented in Figure 4.8.
The first three bytes are needed for the digital baseband receiver of the TDA5340 [28]. The process followed by this receiver is divided into four consecutive steps. The first step is the clock and data recovery, which needs a data stream for the internal filters setting and frequency adjustment. This is the runin sequence. After a clock signal is recovered, the next step is frame synchronization, for which the telegram start identifier (TSI) is used.

The TDA5340 allows the configuration of up to 16 bits for a TSI. All 16 bits are used to reduce false positive detection of data packets by increasing the length of the TSI. The runin and the TSI1 and TSI2 are chosen for optimal performance.
During the implementation, in some cases, the last 1 or 2 bits of data were not received correctly, or the sync was lost before the last bits were received. This caused the receiving node to drop the entire packet because of an invalid length or a wrong CRC. Therefore, to avoid loss of sync before the end of a message, additional padding is appended to every data packet. This padding consists of 1 byte of zeros.
runin, TS1, TS2, and padding are the same for each packet sent within the wireless protocol and do not change. Between the TSI and the padding, the actual data is transmitted.

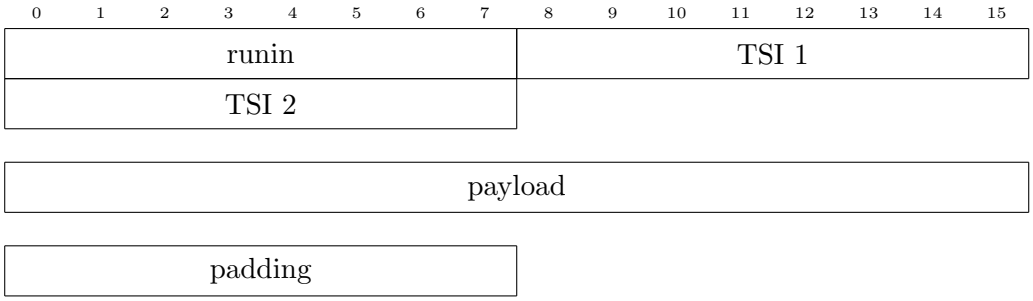


Figure 4.8: Outline of data packet sent between nodes. runin, tsi, and padding are required for the transceiver. The payload is the actual data transmitted by the network protocol.

The payload seen in Figure 4.8 can be further broken down into individual elements, presented in Figure 4.9. It starts with a header element, followed by the actual data message, and ends with the CRC checksum. Each message type has its own data segment. Also, the header and the CRC have the same structure for each type of message.

The header contains 1 byte of length information. This information is used by the receiver to determine the 2 bytes of CRC at the end of each data packet. After the packet with the CRC is validated, the received packet is further parsed. The next 8 bits are status bits and contain the following information.

- The A-bit
  It tells the receiver whether the reception of this message has to be acknowledged. This way the sender of a message can determine, depending on the type of message, if an acknowledge is necessary. The sender for example can set the A-bit for high important messages to guarantee a reception.

- The R-bit
  It is used to inform the receiver that it should reset its status information. This bit is currently used for the reset status command from the command server. In this way, fewer messages have to be sent and the energy consumption further decreases.

The next four bytes contain the receiver ID and the sender ID. At packet reception, the receiver drops all packets that are not addressed to it using the receiver ID.

Followed by one byte used to denote the message type and another byte contains the sequence number, which is increased every time a new packet is sent. Retransmission of the same packet (when no acknowledge message is received, for example) does not increase the sequence number. In this way, the receiver can gain information about the link quality and the number of dropped packets. The length of the data can vary depending on the sent message. In the following sections, the messages that can possibly be sent are described in detail. The CRC is calculated over the whole packet including the header and the data.

The following message types exist.

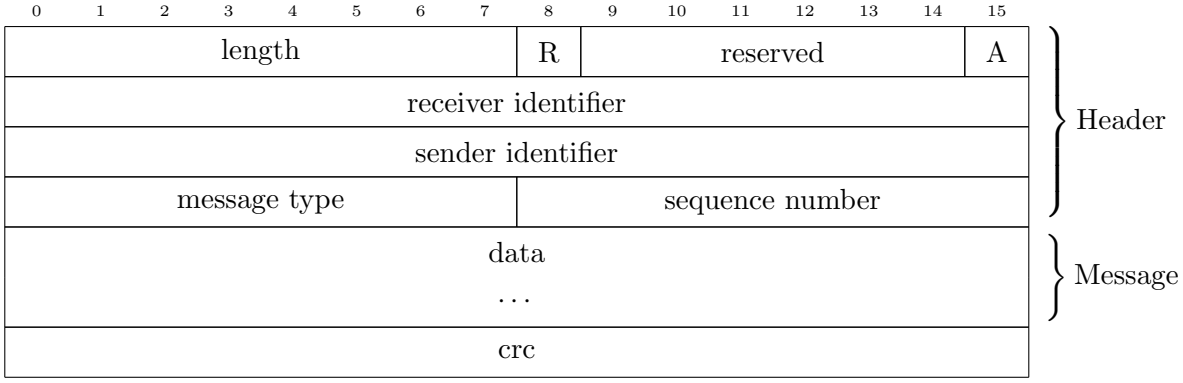


Figure 4.9: The data fields of the payload. The payload can be subdivided into the header message and CRC. Header and CRC fields are present in all packets. The data field varies among message types.

Sensor Data 1 and 2

These types of messages contain the sensor readings and constitute a majority of the messages. Two versions of these messages exist and are chosen depending on the application. The first version contains humidity, temperature, and barometric pressure values. The second version of the sensor data message additionally contains the acceleration measurements. Each sensor reading takes up 16 bits of data, resulting in 6 bytes of message length.

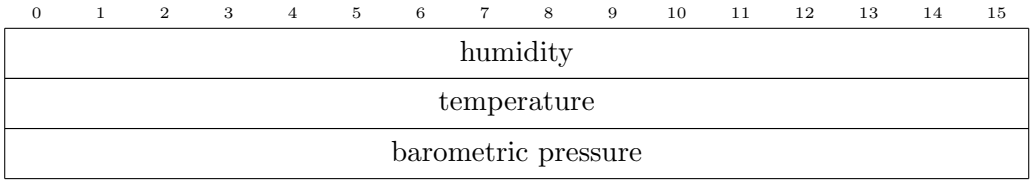


Figure 4.10: Structure of the message type Sensor Data 1.

Figures 4.10 and 4.11 show the data bytes sent for Sensor Data 1 and Sensor Data 2 data packets.

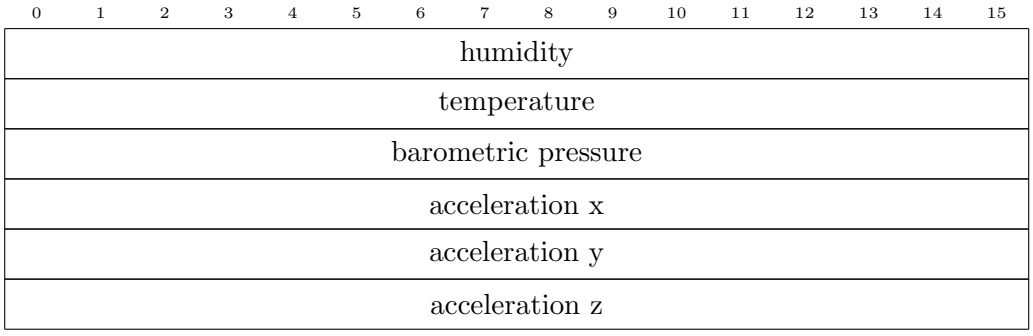


Figure 4.11: Structure of the message type sensor data 2.

Status

The status message is used to obtain information about the sensor nodes operation conditions. As the sensor node is operated, some important parameters are logged and stored in the XMC1100s flash memory.
The status message can be subdivided into the system status and the network status. The system status contains information about the wakeup-counter which is incremented by 1 every time the sensor nodes is powered up. The wake-up counter is 16-bit and can therefore count up to 65 535. If we assume a 10-year uptime for the system, the minimum timer interval must be 80 minutes; otherwise, a counter overflow will occur.
In addition, the internal watchdog timer is used to prevent the system from being stuck in certain waiting states. Every time this occurs, the watchdog violation counter is increased. The network status logs how many packets are sent, how many packets are dropped, how often no acknowledge message is received and therefore a packet has to be retransmitted, how many times the CCA fails, and how often received packets do not pass the CRC check.

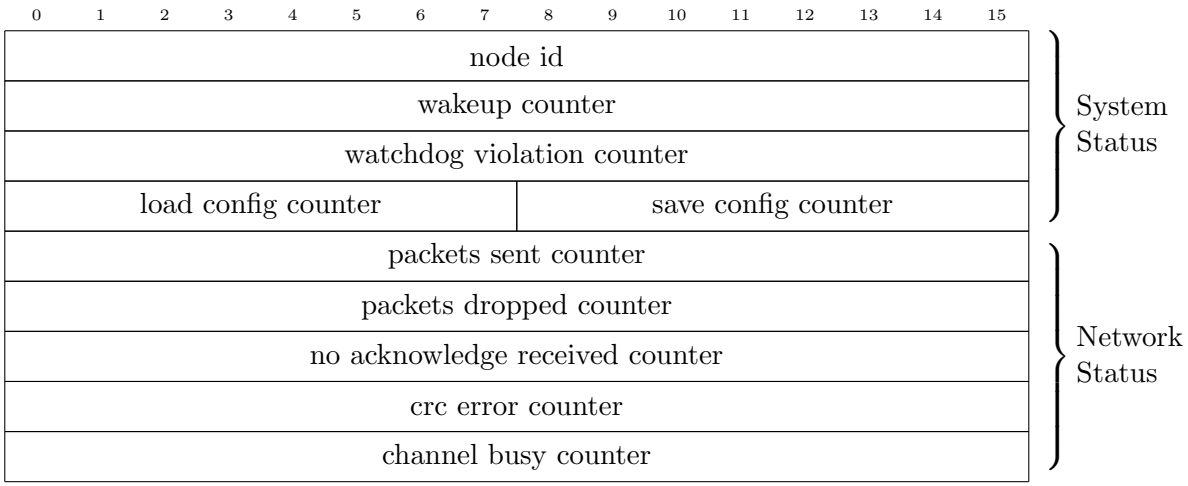


Figure 4.12: Structure of the status message. This message, which can be subdivided into the system status and the network status, logs important information about the operation of the sensor node.

Acknowledge

The acknowledge message contains only the sequence number of the message of which reception is being acknowledged by this message and a status bit data pending (D). This bit tells the receiver of the message that communications should be kept active and more data will be sent shortly.
To implement the message diagram shown in Figure 3.6, the data pending (D) bit is used with the ACK from the gateway to the sensor node. The sensor node now stays active

and listens into the channel until more packets are received.

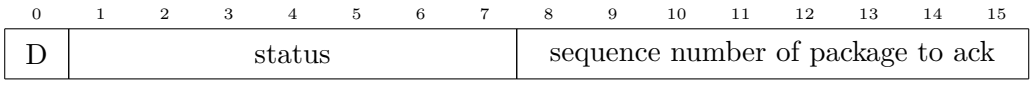


Figure 4.13: Structure of the acknowledge message type.

Beacon

This message does not contain any data and acts as a beacon to notify others nodes of the presence of the sensor node.

Chapter 5

Results

After the implementation, several tests were conducted to analyze the behavior of the network protocol in different scenarios and to estimate the power consumption. The results of these tests are presented in this chapter. The final hardware setup is briefly described, followed by an evaluation of the performance with regard to power consumption and communication. After this, a real-world test run that was performed is reviewed. The problems that occurred during the project and the test scenarios are then discussed. Importantly, in order to gain meaningful insights, most of the analysis is based on real measurements. In addition, a set of test scenarios was designed to mimic future applications. These scenarios are described in Section 5.7.

5.1 System Setup

A typical system setup contains one gateway node and multiple sensor nodes. The number of sensor nodes is limited by the protocol implementation as only 65 535 nodes can be addressed. The anticipated network size ranges from dozens to hundreds of nodes with up to 10 gateway nodes.
Figure 5.1 displays a small WSN with one gateway and three sensor nodes. All the possible variations of the sensor node are shown in this picture. On the left is the gateway node plugged into a connector board. The red board, which functions as a UART-to-USB bridge and power supply for the gateway node, is plugged into the command server. The next three nodes on the right are the sensor nodes. The first sensor node (from the left) is the one in the small-footprint version. It is operated by a CR2032 coin-cell battery, and all sensors are mounted on the sensor node. The next sensor node is plugged into the connector board and supplied by two AA batteries; all sensors are again mounted on the sensor node. The final version has external sensors and is operated using two AA batteries via the connector board. Figure 5.2 presents an example of a deployment site within the walls of a building. In the early stage of the project, only the sensors are deployed within the wall and connected to the sensor node via the connector board (the right sensor node in Figure 5.1). This sensor node is placed within the flush-mounted box, so that it can be replaced or reprogrammed easily while the sensors continue measuring at the correct location.

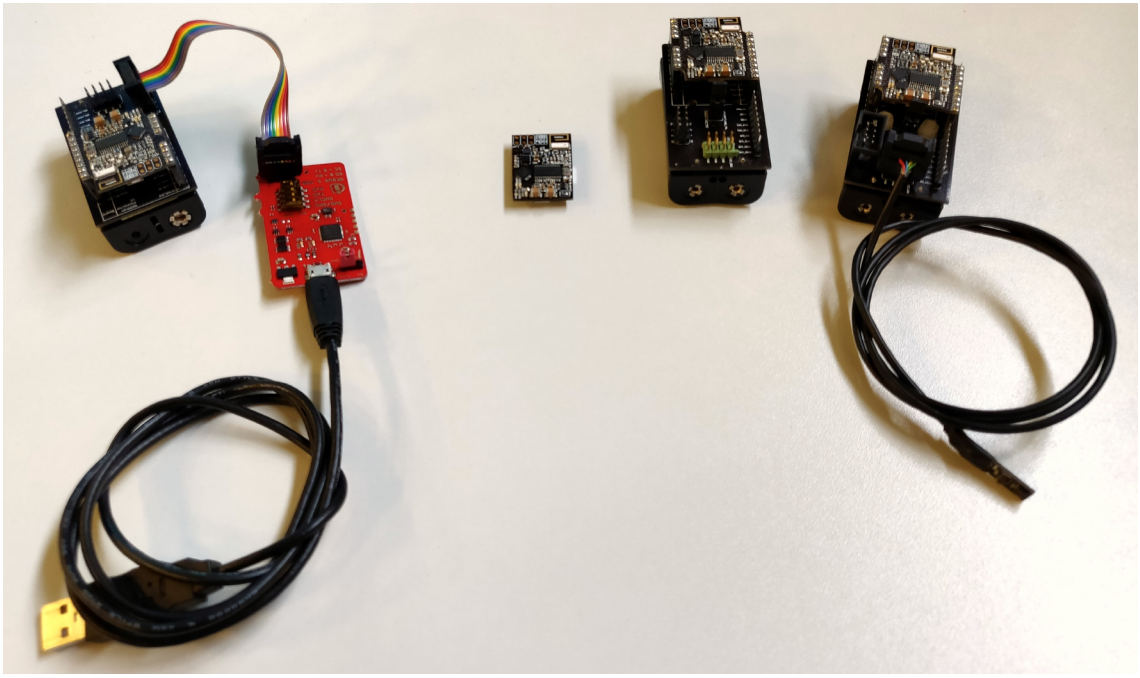
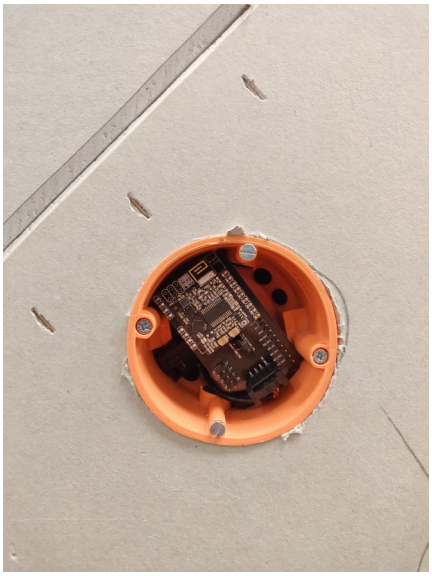


Figure 5.1: Typical system setup. From left to right: gateway node, sensor node without connector board, sensor node with connector board, and sensor node with connector board and external sensors. The gateway node is connected to a UART-to-USB bridge (red board)..

(a) Sensors are placed within the wall an exterior wall.



(b) Sensor node mounted on the connector board connected to the sensors in the wall.

Figure 5.2: Prototype of the on-site test of the sensor network. Sensors on the extender board are placed in the spots of interest and then connected to the sensor node in the flush-mounted box.

5.2 Evaluation of the Communication Performance

One of the main requirements of the sensor network designed in this project is that it must function wirelessly. The use of radio connections creates problems in communication. Because radio connections are based on a shared medium, there is no guarantee that one is the only participant wanting to communicate at a given moment. This section discusses the quality of communication with reference to lost messages and the overhead required for successful transmission. Because all these parameters are strongly dependent on the number of participants in the network, measurements were performed with a varying number of sensor nodes.
The following sections discuss the message loss followed by an analysis of the communication overhead induced by the lost messages.

5.2.1 Message Loss

The message loss in this work is measured as the percentage of messages lost with respect to the number of messages sent. To evaluate the message loss, the network was deployed at multiple operating sites (see Section 5.7) and operated for a certain length of time. Each node was monitored throughout the test. This allowed for the logging of messages lost over the operation period. The monitoring and the fall detection use cases were analyzed separately and are discussed in the following subsections.

Monitoring Case

The conducted tests showed that the network protocol can handle the monitoring use case very well. Even with a very large number of sensor nodes the message loss is minimal if any. Table 5.1 shows the results of the measurements. Even with over 90 sensor nodes (extrapolated) and a wakeup interval of one minute only 0.0079 % of all messages get lost. If the number of sensor nodes is reduced to about 80 no messages get lost at all.

Impact Case

In contrast, problems occurred in the fall detection use case. It has turned out that the protocol used is not suitable for this application. Figure 5.3 shows the packet loss over the number of sensor nodes waking up at exactly the same time (i.e., the fall detection use case). The system could handle up to four nodes waking up at the same time, after which messages were lost. Thus, the figure reveals a sharp increase in lost messages for more than four nodes. Only seven nodes were available for the test; hence, tests could only be conducted with these seven nodes. For larger network sizes, the results can only be estimated. Already, with seven nodes, it can be shown that the protocol is unsuitable for this case as 6 % of sent packets are lost.

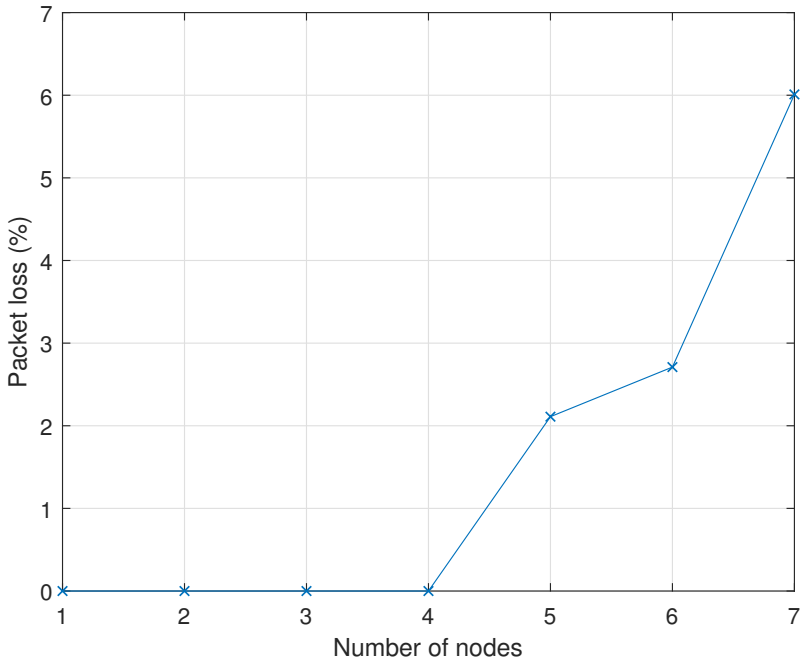


Figure 5.3: Packet loss versus network size for the fall detection use case. If more than four nodes wake up on a detected fall, messages are lost when the current protocol is implemented.

5.2.2 Communication Overhead

Transmission errors caused by network collisions lead to lost messages, as discussed in Section 5.2.1. To increase the robustness of the WSN, lost messages are resent. We use the retransmission rate as an indicator of network performance. This rate indicates how often a packet must be resent on average. If for 100 packets, 5 retransmits were necessary, the retransmission rate would be 0.05 (this could mean that 1 packet was retransmitted 5 times or that 5 packets were retransmitted once). On average, each packet is sent 1.05 times.
This makes the retransmission rate an important factor for calculating energy consumption, and thus, the operating time.

Monitoring Case

First, we examine the monitoring use case. In this scenario, multiple sensor nodes are operated at fixed wake-up intervals. Collisions occur when multiple nodes try to send data at the same time. The following analysis measures the retransmission rate over an increasing network size while the wake-up interval is fixed to 1 minute. The measurement was performed in a typical test environment, where the number of nodes operating and the wake-up interval were modified. In total, over a week's worth of test data were gathered for this analysis.
As it is difficult to set exact wake-up intervals, the actual intervals are mathematically adapted to represent the desired intervals. The following example illustrates the method

used. Each sensor node occupies the RF-channel once per wake-up interval. Channel occupancy is therefore dependent on the wake-up interval of the nodes and the number of nodes operated. The same RF-channel occupancy can be achieved by varying the network size (e.g., the number of nodes) and the wake-up interval.
For example, a sensor network with 10 nodes with a 1-minute wake-up interval causes the same RF-channel occupancy as a network of 5 nodes with a 30-second wake-up interval. In both cases, the RF-channel is used 10 times per minute. Therefore the following analysis is based on extrapolation of fewer nodes with higher duty cycle as representative for a realistic monitoring scenario.
Figure 5.4 shows the results of tests where the wake-up interval is fixed to 1 minute. With an increasing network size, the retransmission rate also increases. However, even with very large networks, the retransmission rate stays below 20 percent.

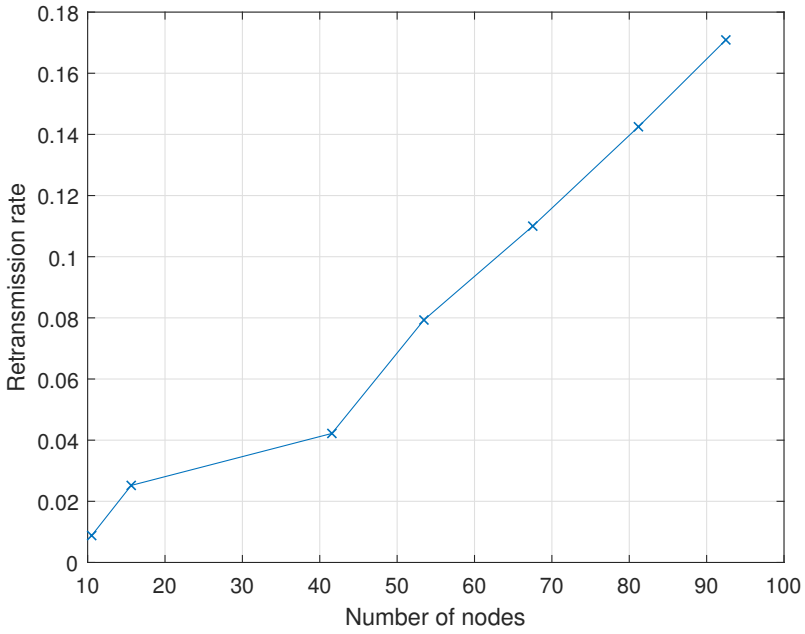


Figure 5.4: Monitoring case. Retransmission rate over network size with a wakup-interval fixed to 1 minute.

Impact Case

In the fall detection use case scenario, the number of sensor nodes cannot be extrapolated as in the monitoring case. Therefore, the analysis is limited to the number of nodes available.
For this test, the sensor nodes were connected such that with the press of a button, all nodes woke up simultaneously. This simulates the worst-case scenario of the fall detection use case.
For example, if someone fell on the floor, more than one sensor node would wake up and try to send data. However, it is highly unlikely that all the nodes would wake up at exactly the same time as detection for the wake-up originates in a physical impulse. These

impulses would not reach all the nodes at exactly the same time (e.g., a spreading shock wave as someone hits the floor).
Realistic tests could not be conducted as this feature had not been implemented yet. Thus, this test assumes that all nodes wake up at exactly the same time, which represents the worst possible case.
Figure 5.5 shows the retransmission rate in the fall detection use case. As can be seen, the retransmission rate is much higher than that in the monitoring case. This is because collisions are guaranteed to happen. With two nodes waking up at the same time, each packet is resent about three times. With seven nodes, each packet gets resent five times. Besides the retransmission rate, the loss rate also increases significantly with the number of nodes (see Figure 5.3).

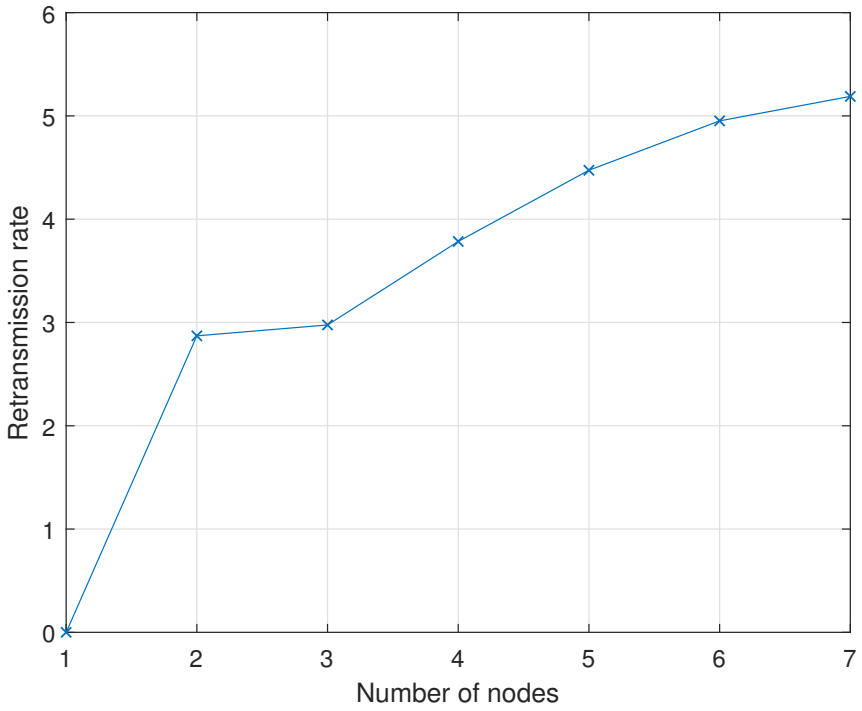


Figure 5.5: Impact case. Retransmission rate over the number of nodes simultaneously detecting the fall.

To summarize, the communication protocol works efficiently in the monitoring case, where collisions are rare. However, when collisions start to increases in the fall detection use case, the performance deteriorates rapidly. Figure 5.7 and Figure 5.6 present a side-by-side comparison of the retransmission rate and the packet loss in both the use cases. Table 5.1 and Table 5.2 display the corresponding values.

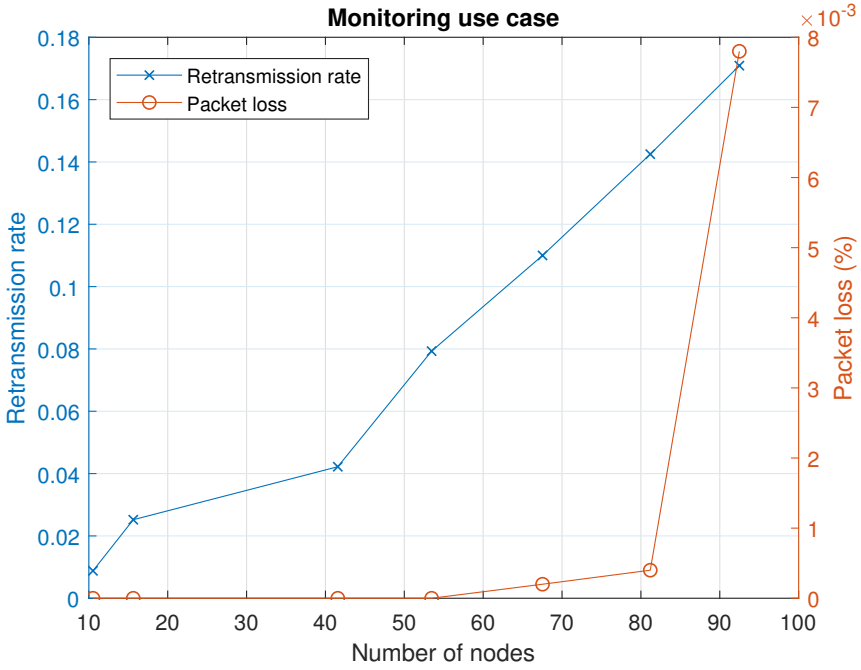


Figure 5.6: Comparison of packet loss and message overhead between different system sizes in the monitoring case.

Number of nodes	Retransmission rate	Message loss
92.5 @ 1 min	17.09 %	0.0078 %
81.2 @ 1 min	14.25 %	0.0004 %
67.5 @ 1 min	11.00 %	0.0002 %
53.5 @ 1 min	7.93 %	0.0000 %
41.6 @ 1 min	4.22 %	0.0000 %
15.6 @ 1 min	2.55 %	0.0000 %
10.4 @ 1 min	2.56 %	0.0000 %

Table 5.1: Retransmission rate and message loss in the monitoring use case. For comparison, the interval was fixed to 1 minute and the number of nodes was accordingly modified.

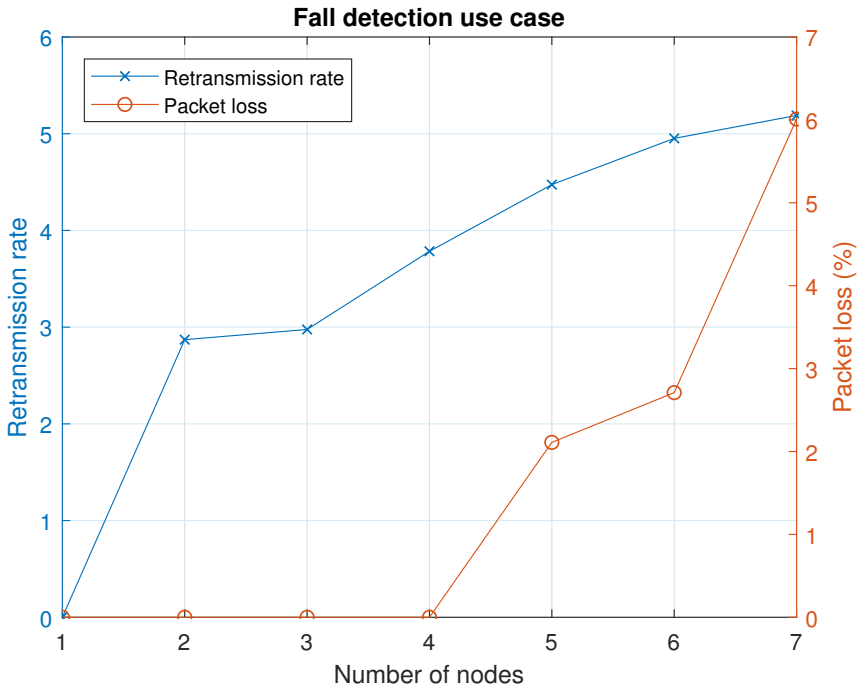


Figure 5.7: Comparison of packet loss and message overhead between different system sizes in the fall detection use case.

Number of nodes	Retransmission rate	Message loss
3	287.07 %	0.00 %
3	297.61 %	0.00 %
4	378.44 %	0.00 %
5	447.37 %	2.11 %
6	495.18 %	2.71 %
7	518.88 %	6.01 %

Table 5.2: Message loss and retransmission rate for the fall detection use case. Already with five nodes woken up simultaneously, packets are lost. The number of lost packets increases rapidly with a growing number of nodes woken up. At the same time the retransmission rate increases as well.

5.3 Evaluation of Power Consumption

One of the main characteristics of a battery-powered WSN is the operation time (the time for which the system is fully functional). As the gateway is powered by the command server, the uptime of the system depends solely on the operation time of the individual sensor nodes. This section analyzes the sensor nodes with respect to power consumption and the resulting operation time.
The analysis consists of two parts, where each operation cycle (monitoring and impact) is examined. The operation time estimates are based on the results of the two-part evaluation. This analysis provides a starting point for further development or improvement to increase the system uptime in the final system design.

5.3.1 Monitoring Cycle

In the monitoring cycle, the sensor node takes measurements of the environment at regular intervals and transmits the data to the gateway (for a detailed description, see Section 3.1.1). The current profile of a sensor node operating in the monitoring mode can be divided into two phases: active and sleeping. While in sleep mode, the sensor node draws a constant current. This current is about 90 nA at 3 V. For the estimates, it is assumed to be 100 nA as the leakage depends on the components used and can vary.

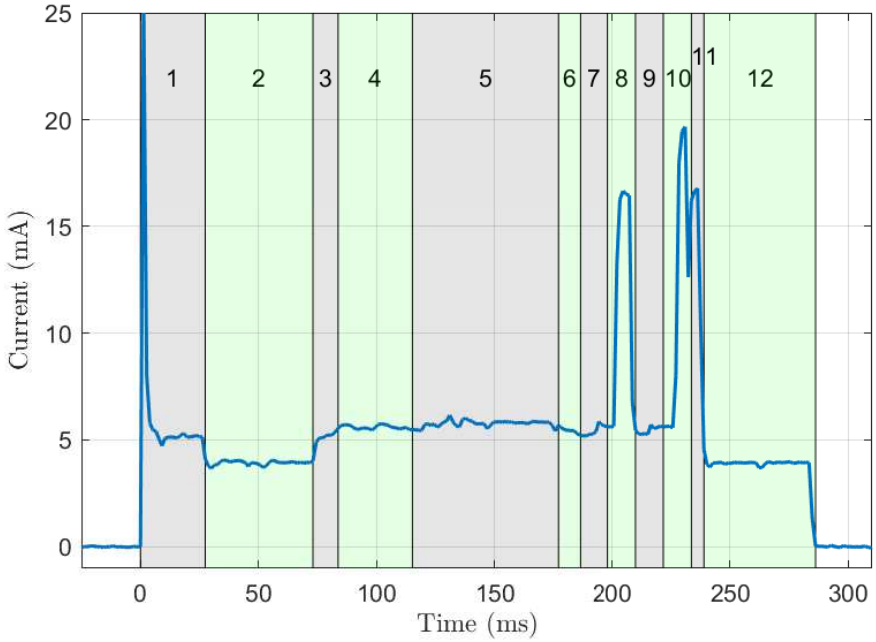


Figure 5.8: Current consumption of a sensor node operated at 3 V. At time zero, the node wakes up and performs the programmed actions. The following phases are highlighted: 1 - Startup; 2 - Load config; 3 - Init; 4 - HDC operation; 5 - DPS operation; 6 - LIS operation; 7 - Data processing; 8 - CCA execution; 9 - Switch to TX; 10 - Data transmission; 11 - Acknowledgment reception; 12 - Save status.

Figure 5.10 shows the current drawn from the battery of a sensor node operated in the active phase of the monitoring mode. The current profile is divided into sections. Each of these sections is marked in the figure and discussed below. In addition, Table 5.3 presents some important attributes of each phase.

Phase 1 - Startup

At time zero, the sensor node awakes and switches to active mode. This is marked by a jump in the power consumption. At the time of switching to active mode, the current consumption peaks at about 30 mA. Such high currents are needed to charge the decoupling capacitors at the output side of the power management unit. The resulting voltage drop can cause problems with a weak battery and lead to malfunctioning of the timer chip. To mitigate this effect, two compensatory 220 µF capacitors are used at the input side of the power management unit. For a detailed description of this problem, see Section 5.6. As soon as the microcontroller is powered up, its internal modules needed for operation are initialized and configured. The startup takes about 25 ms.

Phase 2 - Load Config and Status

In sleep mode, memory retention is turned off. All values stored in RAM are therefore lost and have to be restored on power-up. Phase 2 of Figure 5.10 represents this action. Values such as the node identifier, the seed for the random number generator, and configuration information are loaded from electrically erasable programmable read-only memory (EEPROM) in this phase. As all EEPROM activity utilizes the global bus system, the CPU is put into power-save mode while the EEPROM is accessed. This results in a lower current consumption, as can be seen in Figure 5.10. The process takes about 90 ms as the EEPROM is slow.

Phase 3 - Initialization

With the RAM restored, the initialization of the modules can be completed. One of the modules configured in this phase is the random number generator.

Phase 4 - Humidity and Temperature Measurement (HDC)

After the startup procedure is completed, the actual sensor node software is executed beginning with taking measurements of the environment. Figure 4.5 illustrates the program flow of the sensor node.
The first measurement taken is that of humidity and temperature by utilizing the HDC1080. In Phase 4 of Figure 5.8, the HDC1080 is configured to first take humidity and temperature readings, after which measurement is started. The microcontroller waits until the measurements have been carried out and buffers the values until the end of the measurement procedure.

Phase 5 - Air Pressure Measurement (DPS)

The next measurement is the barometric pressure measurement. The microcontroller configures the barometric pressure sensor (DPS310) to take highly accurate air pressure

measurements. At about 60 ms, these measurements take longer than the other measurements. After the measurement is taken, the calibration registers are read from the sensor. These are needed to calculate the barometric pressure and temperature. Then, raw values for the pressure and temperature are read. For calculation of the barometric pressure, the raw values of pressure and temperature, such as the calibration coefficients, are needed.

Phase 6 - Acceleration Measurement (LIS)

The final measurement records acceleration values. To do this, the accelerometer LIS331 is configured to operate in the highest sensitivity mode. Subsequently, measurements for each axis are performed and read. This is done relatively fast compared to the other sensors, which is why the accelerometer and its operation do not put a great strain on the energy budget.

Phase 7 - Data Processing

In this step, the gathered data is processed prior to being sent. A data message is then created (as shown in Figure 4.10 and Figure 4.11).

Phase 8 - Channel Clear Assessment (CCA)

The first step of the sending procedure is to perform a CCA. This is part of the CSMA/CA approach implemented.
To do this, the transceiver TDA5340 is configured to listen into the RF-channel for the maximum signal energy recorded in a 5ms listening window as that is the expected inter packet length. On the basis of the recorded maximum signal energy and the noise floor, the decision about whether the channel is clear or not is made. Furthermore, as listening consumes considerable energy, the transceiver is put to sleep directly after (as seen in Figure 5.10). About 12 mA is used in the receive mode (see [28]). After the transceiver is turned off, the current consumption drops to 5 mA.

Phase 9 - Switching to TX

When the channel is found to be clear, the sending starts, for which the transceiver is put into transmission mode. This takes about 10 ms because the internal clocks have to settle.

Phase 10 - Sending Data

After the transceiver is put into transmission mode, the data packet is written in the transceivers transmission buffer and sending is initiated. This can be seen in Phase 10 of Figure 5.10. At maximum output power, the transceiver uses about 15 mA for sending (see [28]). The high current consumption during the sending can also be seen in the figure.

Phase 11 - Waiting for ACK

Since the data packets have to be acknowledged, the transceiver is put in reception mode after sending. At the same time, a timer is started. Acknowledge messages can only occur shortly after a packet is sent, as discussed in Section 3.4. If 10 ms pass without an

acknowledge message, the data packet is assumed to be lost, and a retransmission attempt is made. Either after 10 ms or after an acknowledge message is received, the transceiver is put back to sleep. Figure 5.10 shows the current profile in the case of an acknowledge message being received.

Phase 12 - Saving Status

After successfully performing its task, the sensor node saves the current system status by writing it to the EEPROM and switches to the sleep state.

Table 5.3 shows the power consumption in the active state of the sensor node. For each phase, the energy consumed and the percentage of total energy consumed in the active state are listed. Grouped together, roughly a third of the energy consumed is used during system operation (startup, load and save status, and initialization), another third is used for communication (CCA, switching to TX mode, sending and receiving the acknowledge message), and the final third is used for operation of the sensors.

	Time (ms)	Energy Consumed	
		µA s	%
1 - startup	27.45	169.50	10.34 %
2 - load status	45.57	179.14	10.92 %
3 - init	10.65	54.42	3.32 %
4 - HDC	31.46	176.17	10.74 %
5 - DPS	61.87	356.86	21.76 %
6 - LIS	9.303	50.64	3.09 %
7 - data processing	11.87	64.67	3.94 %
8 - CCA	11.83	135.00	8.23 %
9 - switch to tx	11.76	64.23	3.92 %
10 - SEND	11.84	138.52	8.45 %
11 - ACK	5.31	71.80	4.38 %
12 - save status	47.21	178.96	10.91 %
$\sum$	286.12	1639.92	100 %

Table 5.3: Detailed description of the phases of the monitoring cycle depicted in Figure 5.10. The time needed and the energy consumed for each phase are presented. The energy consumption is roughly divided into three equal parts: system operation (1, 2, 3, 12); measuring (4, 5, 6, 7); and transmission (8, 9, 10, 11).

Figure 5.10 displays the ideal case when no retransmission of messages takes place. As this is not the only possible case, Figure 5.9 presents another extreme, a case where the gateway was not powered due to a power failure. This led to five retransmission attempts, of which none was successful.

5.3.2 Impact Cycle

In contrast to the monitoring case, the nodes in the fall detection use case stay in sleep mode until an external event occurs. This event can be generated by a fall detection system built into the floor, for example. In this case, it is possible that multiple sensor nodes wake up at the same time. These nodes then try to send their data simultaneously. Unfortunately, this behavior causes many collisions as the occupancy of the RF-channel is increased drastically. Ideally, packets should never be dropped by sensor nodes as this increases average power consumption per cycle, which decreases system uptime. This means that the power consumption in the fall detection use case is expected to be higher than that in the monitoring case, as collisions are more likely to occur in the fall detection use case, because of which retransmissions have to take place. An in-depth analysis of packet loss is presented in Section 5.2.
Figure 5.9 displays the current consumption of a sensor node that does not receive acknowledge messages for sent messages. This behavior is expected to happen regularly in the fall detection use case. Figure 5.9 shows five retransmit attempts by the sensor node, which lead to a dropped packet.

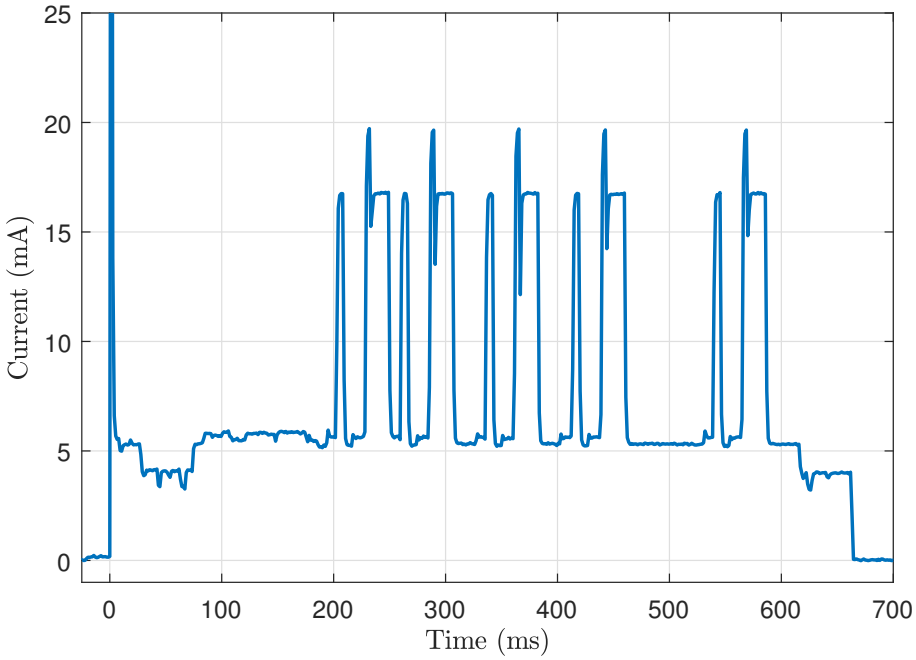


Figure 5.9: This figure shows the current profile of a sensor node when the gateway is off-line and no acknowledge messages are received. The node waits a random length of time between the CCA, SEND, and ACK phases. After the fifth unsuccessful attempt, the packet is dropped. The sequence of random waiting times in this case is 0 ms, 10 ms, 10 ms and 60 ms.

5.3.3 Mixed Cycle

The previous sections analyzed the two types of cycles monitoring and fall independently from each other. However, scenarios may exist where a combination of these two cycles is used. In such cases, the power consumption and operation life depend on all the parameters analyzed above.

5.3.4 System Uptime Estimation

The operational life of a battery-powered WSN is determined by the capacity of the battery and the energy consumption of the sensor node.
The capacity of the battery is fixed in this analysis as it is not possible to influence the capacity other than by choosing a different type of battery. For this reason, the operational life primarily depends on the wake-up interval, the number of impact events detected, and the number of retransmits necessary.
Moreover, since the operational life of the WSN exceeds the project duration, this section tries to estimate the system uptime in addition to measuring it.
The system runtime is estimated for different scenarios. These estimations are based on the measured power consumption (Figure 5.10, Figure 5.9, and Table 5.3) and the calculated average waiting times for the binary exponential backoff algorithm (Section 4.3.1).

With this information, the sensor nodes uptime can be estimated for the case with no RF interference. On examination, the transmission flow chart (Figure 4.7) indicates that a new transmission takes place if no confirmation is received within the acknowledgment timeout. This also happens when the channel is busy. To prevent the same two nodes from interrupting each other after a collision, each node waits a random length of time between repetitions.
This means that the time required to transmit a message depends on the activity in the channel and is therefore not static. If we want a realistic estimate of the uptime of the entire system, we must take this into account.

The maximum waiting time increases exponentially each successive time a transmission fails and is reset with a successful transmission. This maximum waiting time is split into time slots, each of 10 ms. One of these time slots is chosen at random. The maximum waiting time is capped at 256 time slots of 10 ms, resulting in a maximum possible time of 2.55 s between retries.
Waiting times are calculated as follows

$$\begin{aligned} failed\_counter &= (no\_ack\_cnt + cca\_failed\_cnt) \\ N_{slots} &= 2^{failed\_counter} \\ t_{wait} &= N_{slots} \cdot rand(N_{slots} - 1) \cdot t_{slot} \end{aligned} \tag{5.1}$$

where the $rand(argument)$ denotes the random function that generates a random number in the range $[0, argument]$. Therefore the maximum waiting time for each execution of the algorithm is calculated by $N_{slots} \cdot t_{slot}$. The minimum wait time is always zero. The average waiting time depends on the $rand$ function.

Furthermore, if we assume that the pseudo random number generator is reasonably good, all numbers within the range are equally likely to appear. This means the average of $rand(argument)$ is $argument/2$. On the basis of this assumption, all further calculations and evaluations are based on average waiting times.

In terms of waiting time the worst case scenario would be that the channel is busy four times and that four times no acknowledge message is received resulting in eight executions of the exponential back off algorithm. This results in a maximum waiting time of

$$\begin{aligned} dt_{maximum,8} &= t_{slot} \cdot \left(\sum_{i=1}^{8} 2^i - 1 \right) \\ &= t_{slot} \cdot (1 + 3 + 7 + 15 + 31 + 63 + 127 + 255) \\ &= 10\,\text{ms} \cdot 502 \\ &= 5.02\,\text{s} \end{aligned} \tag{5.2}$$

The total time the sensor node is active depends on the waiting times between sending attempts as well as the time needed for the sending and the CCA. Both, the time needed for sending and the CCA, are constant (listed in Table 5.3). This makes the calculation of power consumption based on waiting times straightforward.

Minimal and maximal waiting times have been derived in the previous paragraphs and lie between 0 and 5.02 s. Thus, to obtain the upper and lower bounds of the system uptime is a trivial task. Both extreme cases no congestion and therefore no waiting times, and the case with a maximally congested channel where each message has to be retransmitted eight times are very unlikely. Therefore, these absolute maximum and minimum values are not useful as they cover a huge range. Another reason that these values do not provide much information is the exponential nature of the algorithm. The waiting times scale exponentially with the number of consecutive retransmits. This means that it is possible to have different average waiting times for the same percentage of retransmits. For example, in both scenarios 10 data packets with eight consecutive retransmits out of 800 data packets, and 10 data packets with one retransmit out of 100 data packets the retransmission rate is 10 %, but the average waiting times differ.
Hence, a more sophisticated approach to calculate practical estimates is needed. Some tests were first conducted to determine the number of retransmits for each situation. A detailed analysis of retransmission within the network can be found in Section 5.2. All uptime estimates are based on these measured values so as to be as close to real world values as possible.
Furthermore, the energy consumption while waiting for the acknowledge message with the transceiver active and the energy consumed while waiting for the chosen time slot were evaluated (See Table 5.4). Together with the probability of retransmission and the distribution of the retransmits, an estimate for the system uptime can be derived.

Thus, from the energy consumed for reception and waiting (see Table 5.3), the operation time can be estimated.
The following section discusses the assumptions made and the scenarios on which the average waiting times are based.

	Time (ms)	Energy Consumed µA s
ACK not received	15	198.31
1 ms of waiting	1	5.32

Table 5.4: About 13 mA are needed for actively listening for an ACK. The ACK timeout is 15 ms. This results in about 200 µA s of consumed energy when no ACK is received. Between retransmits about 5 ms are needed for idle waiting. 1 ms of waiting in idle state therefore consumes about 5 µA s.

Theoretical Assumptions

In order to come up with useful estimates for the system runtime, we have to make a few assumptions.
A CR2032 coin cell from VARTA is used as the power source for the sensor nodes (see [24]). This coin cell has a rated capacity of 230 mA h and a self discharge rate of below 1 % per year. Furthermore, we assume that we can extract only 80 % of the energy stored in the battery as the voltage will drop below a critical value at some point. This means that after about 160 years of the battery remaining on the shelf without ever being used, it reaches this point of discharge due to the self discharge rate (assumed to be 1 percent per year in the worst case scenario).

However, the following analysis does not take the self discharge rate into account as other factors have a greater impact on the actual operational time. As demonstrated in [26], the battery's capacity is greatly influenced by the ambient temperature. It should therefore be noted that operation of the system at high temperatures could significantly decrease the system runtime. Also note that this theoretical assumption does not take into account other factors such as the chemical decomposition.

Figure 5.10 reveals the estimated operational life of a sensor node operated in the monitoring mode. A WSN consists of multiple sensor nodes and is fully functional until the first sensor node fails. However, depending on the application, a partly functional WSN could still provide useful information. As some nodes are expected to fail before others (for example, due to different environmental conditions), the operational life of the whole system might vary depending on the requirements of the application. This thesis focuses on the operational life of a single sensor node and leaves the determination of values for the whole system to system designers. All estimates are based on data measured and assumptions made. The consumed energy is calculated by using the wake-up interval and incorporating realistic packet retransmits and waiting times. The retransmission rate depends on the network size and the operation mode. Figure 5.10 shows the uptime estimates for different scenarios.

Tests conducted on a WSN with nodes operating in the fall detection mode demonstrated that the nodes had a drastically shorter lifetime than those operating in the monitoring mode. The reduction in life span is because of the protocol implementation discussed in Section 5.2.1.

A system of 25 sensor nodes and a sensing interval of 30 minutes, used to monitor the structural health of a building, could therefore be operated for about 20 years. In

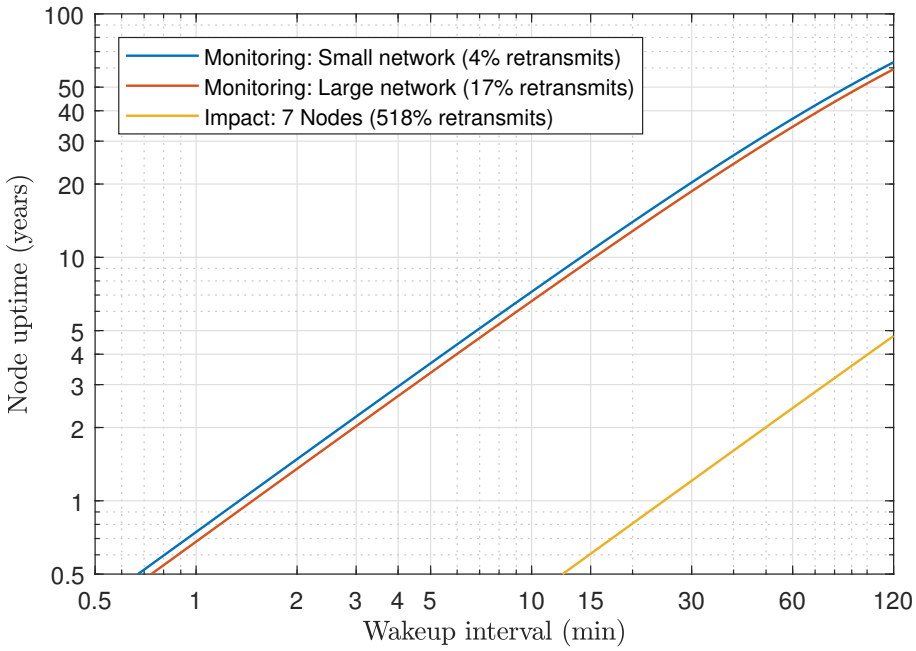


Figure 5.10: Estimated operational life of a sensor node operated in the monitoring mode. The sensor node wakes up once per wake-up interval and sends the gathered data to the gateway. If these intervals are increased, the sensor node uses less energy and the operational life increases. In contrast, if the network size is increased, collisions while communicating are more likely to occur. For each collision, additional energy is consumed because data packets have to be retransmitted.

this scenario the duty cycle (ratio of active to sleeping time) is about 0.0017 %, assuming average retransmission rate and waiting times. A maintenance free sensor network is thus feasible. Moreover, if placed within external wall insulation, the WSN could be replaced together with the insulation at regular building maintenance intervals.

5.4 Power Consumption versus Link Quality versus CCA

Every design aspect of the sensor nodes is a trade-off between functionality and lifetime. The decision to use CCA in the wireless protocol and in the transceivers transmission power involves one such trade-off. In the design and implementation phase of the thesis, it was decided that CCA should be used to reduce the transmission failure probability. As a result, the transmission power was set to the maximum. Following a detailed analysis of the energy consumption (see Section 5.3), these decisions are once again critically reviewed and discussed in this section.

Energy Overhead of the Channel Clear Assessment (CCA)

For the CCA the sensor node has to turn on the transceiver and switch to receiving mode for some time in order to listen for activity on the channel. When done, the transceiver has to be switched to transmit mode to transmit the data. As seen in Table 5.3, the CCA,

together with the switch from receiving to transmission mode, needs about 20 µA s, or 12 percent of the energy used in the monitoring cycle depicted in Figure 5.10. This is about half the energy consumed for communication.
The analysis of the energy consumed, presented in Table 5.4, is based on an ideal cycle where the gateway receives the message and replies with an acknowledge message so that no retransmission takes place. In contrast, in a non-ideal case (Figure 5.9), the power consumption for the CCA is higher because for each sending attempt, roughly 200 µA s are consumed.
However, Table 5.1 indicates that even on very large networks (about 100 nodes), the retransmission rate is quite low. Thus, the collision rate is also low.
Considering the low collision rates, it makes sense for small networks to switch off the CCA. This could improve the system uptime by up to 12 percent.
In fact, the tests revealed that the CCA does not decrease the collision rates as the channel occupancy consists of only very short communication bursts. If one of these bursts is detected with CCA, the communication is already over before the detecting node can even switch to the transmission mode. Thus, CCA does not provide any surplus but consumes power.

Adaptive Transmission Power

In the design phase of the protocol, an adaptive transmission power control was discussed to help save energy. However, on examining the energy consumption, it was found that the consumed energy varies very little between the maximum and minimum transmission power settings of the transceiver.
This is because the communication takes a very short amount of time. Thus, in the total power consumption, the difference would at maximum be 2 percent per cycle. Moreover, the adaptive power control would need a complex algorithm to detect malfunctions because of insufficient sending power, which in turn would again consume energy. Thus, the results suggest that it is best to not implement any sort of adaptive power control and use only the maximum transmission power setting.

5.5 Real World Test Run

A test run was performed to verify the functioning of the WSN designed in this thesis. The sensor network was deployed in a small room and operated from 23:00 to 08:30. Together with the sensor data, the status information of each sensor node was collected. The following figures present this data.
The setup was as follows. Node 61056 and Node 41428 were placed outside on the windowsill. Node 29281 and Node 19364 were placed on the inside of the window. Node 16603 was mounted on the ceiling, whereas Node 34128 was placed on the floor. In addition, Node 47993 was placed on a desk in the middle of the room. The setup is discussed in detail in Section 5.7

Figure 5.11 displays the temperature readings from each sensor node in the test. When the test was started, the window was opened for half an hour and then closed. All the sensor nodes placed next to the window showed a clear temperature decrease for this pe-

riod, followed by an increase after the window was closed.
The only node that was not affected by the open window was the node mounted on the ceiling. It showed a permanently high temperature of about 26 degrees Celsius.
Compared to the data from the nodes placed indoors, that gathered by the nodes outside is characterized by noisy readings. Also, the lowest temperature was reached at about 06:00 in the morning.

Moreover, readings from the nodes outside are very similar to those from the nodes indoors. This indicates that there was a temperature difference within the room. The placement of the nodes (height and distance to obstacles or walls) played a significant role in the temperatures measured.

Figure 5.12 shows the humidity readings from the recording period. Over time, the

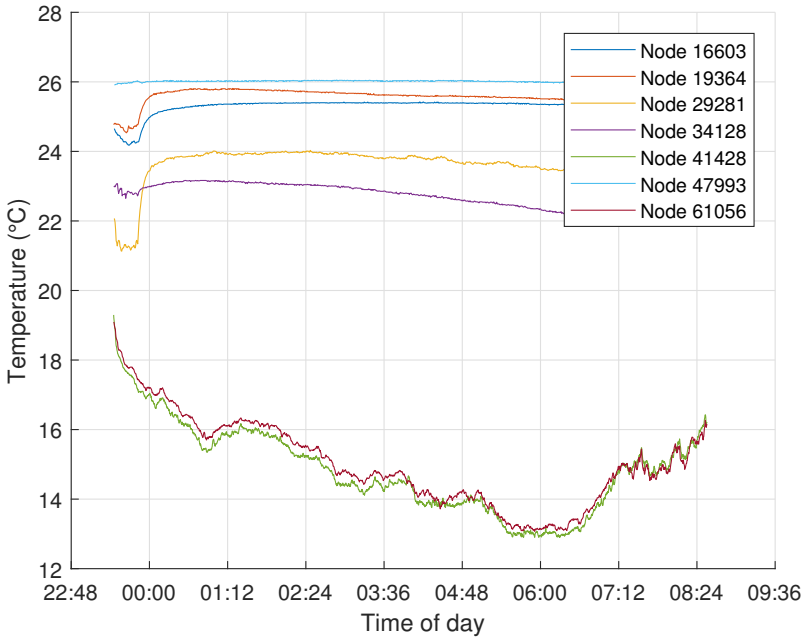


Figure 5.11: Temperature profile of seven sensor nodes in a test run performed over a 9h period. The sensor nodes were placed within a small room (and outside on the windowsill).

humidity increased for all the sensor nodes. As relative humidity is, among other things, dependent on temperature, one can see a negative correlation between temperature and humidity. As the temperature decreases, humidity increases.

Figure 5.13 displays the air pressure for all the sensor nodes during the recording period. Despite the offset among sensor nodes, all the readings are perfectly correlated. There seems to be no difference in air pressure on the outside and the inside of the room. The only noticeable difference is that the readings from the outside nodes are noisier.
The different offsets are due to the fact that the sensor nodes were not placed at the same height. As the DPS310 is very sensitive and can detect even small changes in altitude, the readings are expected to vary among different deployment sites according to their height.

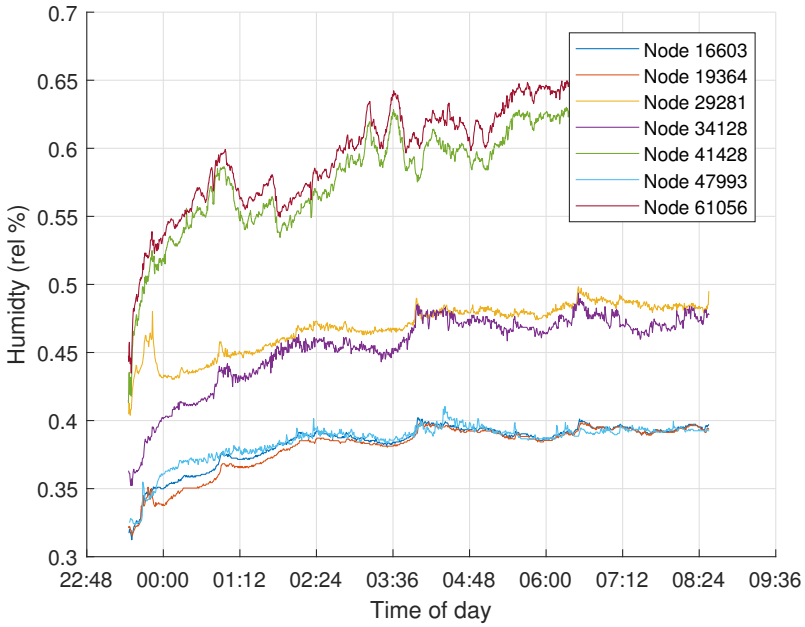


Figure 5.12: Humidity profile of seven sensor nodes in a test run performed over a 9h period. The sensor nodes were placed within a small room (and outside on the windowsill).

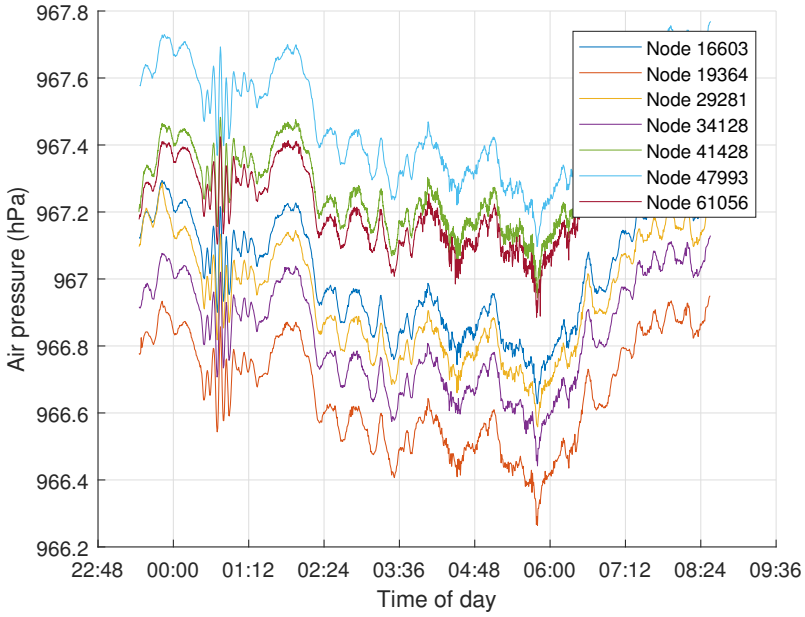


Figure 5.13: Pressure profile of seven sensor nodes in a test run performed over a 9h period. The sensor nodes were placed within a small room (and outside on the windowsill).

Additionally, status information, such as the number of dropped packets, was collected for each sensor node. To analyze the performance of the WSN, the status information of all the sensor nodes is combined and plotted in Figure 5.14. The upper plot shows the aggregated status information over the time of the test run. All the graphs indicate a steady increase without any irregularities. The WSN seems to have operated without any problems.

The network size equivalent to a 1 minute wake-up interval is 20 nodes (for this calculation, see Section 5.2). The communication overhead is 7.06 %. The distribution of the number of retransmits of packets is displayed in the lower part of Figure 5.14. It shows that most of the packets that were not acknowledged were retransmitted once. The distribution of retransmits was used to estimate the system uptime as it play a major role in the overall power consumption.

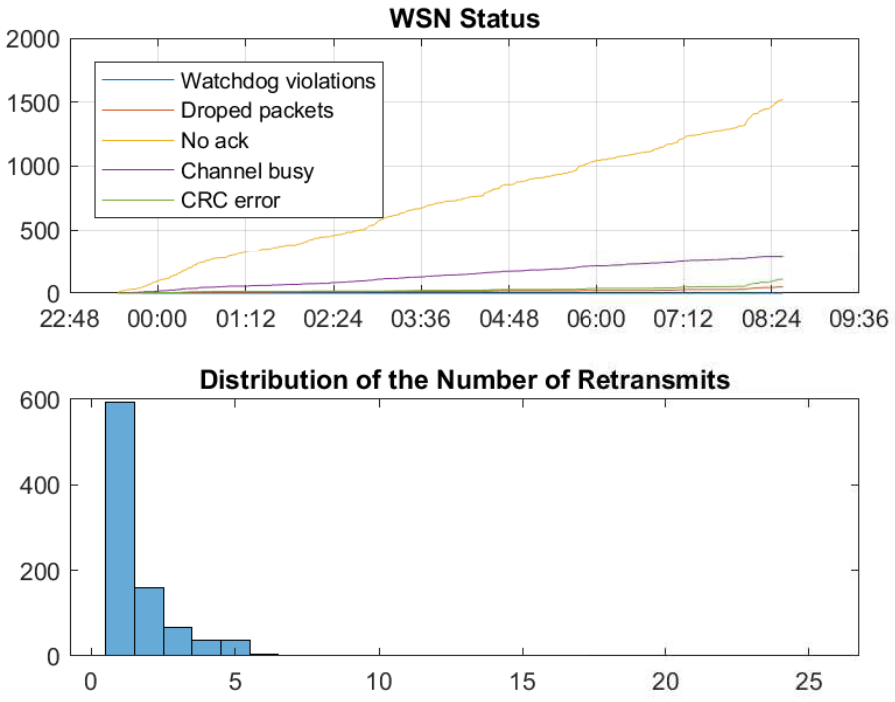


Figure 5.14: Status of the WSN during a test run performed over a 9h period. The sensor nodes were placed within a small room (and outside on the windowsill).

5.6 Challenges and unexpected Issues

During the implementation of the WSN, several problems occurred, which are described in this section. While some of these problems were solved during the development iterations, others still persist and provide a useful starting point for the next iteration of the project.

5.6.1 PCB Redesign

After the first breadboard-based prototype, a PCB was created. However, due to a simple naming error, some wires were not connected to the correct ports on the transceiver. Thus, communication was not possible, which made a redesign necessary.
Prior to the redesign, more tests were conducted in order to avoid further errors of this kind. The wrongly connected wires were cut on the PCB and connected again with bonding wires. This method was found sufficient to conduct further tests.

5.6.2 Power Supply CR2032

Another problem was the odd behavior exhibited by the nodes in some situations. The nodes were constantly powering on and off. At first, the error was thought to be within the configuration or wiring of the timer chip because the timer chip seemed to not be functioning correctly.
However, even after every component of the timer module had been replaced, the error still persisted. Measurements conducted did not reveal any errors and showed an expected voltage drop on the input side of the sensor node (i.e., the battery). The voltage drop caused by the charging current of the decoupling capacitors was about 0.2 V. Further measurements revealed that the limited ground plane due to the two-layered PCB design introduced large parasitic resistance, because of which the voltage drop along the wires of the sensor nodes increased drastically. The effect caused a drop of about 1 V at the timer chip. As a result, the timer chip was operating outside of its specification. This caused a constant restart of the timer cycle and led to the on-off toggling. As a quick fix for this problem, additional capacitors were placed on the battery side to reduce the voltage drop by reducing the internal resistance of the power supply (battery and capacitors). A satisfactory result was achieved by placing the new capacitors very close to the power switch in order to reduce the voltage drop.
This fixed the on-off toggling problem; however, the main problem persisted and was fixed in the second version of the PCB, where care was taken to ensure that the ground plane was as continuous as possible. Despite the better ground connection, the support capacitors were still included in the second version to further improve stability. As a result, the voltage drop on power-on was limited to 0.1 V in the second version.

5.6.3 CCA Threshold

To determine whether the channel is occupied, the transceiver listens into the channel for a short period of time. The maximum RSSI logged provides information about the energy level on the channel. Based on this value, the decision about whether the channel is free or not is made. An occupied channel is characterized by a high signal strength, and thus, a high maximum RSSI. The problem of how to define the threshold is a complex one.

For the correct decision to be made, the RSSI value of an unoccupied channel has to be known. Whenever the RSSI exceeds this value, we can assume that the channel is clear. The RSSI of an unoccupied channel might vary across different deployment sites. Many factors have an impact on the noise-floor RSSI, such as RF pollution, differences in production quality, and ambient temperature. All these factors can change over time; therefore, the actual noise-floor RSSI has to be modified accordingly. This is done by averaging the values of RSSI measured shortly after each data transmission. In this way, each sensor node keeps track of the noise floor at its operation site. To minimize false negatives for the CCA, the channel is assumed to be clear if the measured RSSI is within a band around the noise-floor RSSI.
The width of the band around the noise floor and the number of samples used for averaging are parameters chosen based on what worked best. The nature of the problem, however, makes predictive behavior for all cases almost impossible. Long-term test runs have to be performed to optimally define these parameters and adjust the algorithm for best performance.

5.7 Test Scenarios

This section provides a short overview of the tests carried out and the setups used. All results and data presented in this thesis are based on data gathered in one of the following test scenarios. The measurement series of both the fall detection and the monitoring cases have been evaluated.

Fall Detection Use Case

For the fall detection use case, all nodes are woken up simultaneously as would happen in the case of fall detection being activated. These test force collisions on data transmission therefore provide insights into whether and to what extent the protocol is capable of dealing with collisions in general.
In the test scenarios, the sensor nodes were placed next to each other and then connected in such a way that a simple press of a button would wake all of them up. They were woken up approximately every 10 seconds. This was done 100 times for each set of sensor nodes in a range of two to seven.

Monitoring Use Case

For the monitoring use case, the sensor nodes were placed next to each other and then operated at different timer intervals. For each setting, the data and status were logged. For this test, a limited number of sensor nodes were available. However, to still gain insights into real-world applications, results for more nodes were reconstructed in the following way. If we have two nodes sending data every minute, the communication channel will be busy twice per minute. The channel will also be busy twice per minute, on average, if we operate four nodes that send data every two minutes. This implies that we could interpret the results based on how many nodes we want or on the time interval we want to choose. Because we only had a few sensor nodes, we chose the time interval to be short in order to obtain results for a sufficient number of nodes mathematically.

For the real-world test run, five sensor nodes were installed in a small room with a large window. Two more sensor nodes were placed in front of the window on the windowsill. Four of the five sensor nodes in the room were placed at each corner of the room, and the fifth one was placed on the ceiling in the middle of the room.
The wake-up interval of the sensor nodes was set to be between 10 and 30 seconds. Therefore, the result is also representative of a wake-up time interval of 5 minutes with the operation of 70 nodes.
The sensor nodes were operated for a 10-hour period from 23:00 to 09:00.

Chapter 6

Conclusion and Future Work

This thesis is part of a project on dependable, secure, and time-aware sensor networks. As the project is in an early stage, the WSN developed in this work provides initial insights into how to proceed in subsequent stages.
In the course of this work, a prototype of a highly integrated IC was created. It was then used to build a WSN to provide valuable insights into further development of the IC.
Based on two use cases - a structural health monitoring case and a human fall detection case - requirements were defined. For the monitoring case, measurements have to be performed on a regular basis, while for the fall detection case, the measurements start with the detection of a fall. In both scenarios, the sensor nodes should be battery-powered and last as long as possible.
Nonfunctional requirements for the hardware are that the sensor nodes should reproduce the function of the final IC as closely as possible. To do this, off-the-shelf hardware components with functionality similar to that of the final IC were used to assemble the sensor nodes.
After multiple iterations of hardware and software design, the WSN was set up. It consists of multiple sensor nodes and one gateway node, which were developed simultaneously to the WSN. The gateway communicates directly with the sensor nodes and serves as an interface to a command server. The command server could be any computer-based system running software that receives data from the WSN and passes commands to the WSN. In this work, a computer running a UART terminal was used as the command server.
For communication between the gateway and the sensor nodes, a simple network protocol was designed. Although the communication requirements for the two use cases are considerably different, the protocol designed should be able to handle both scenarios well.

As energy consumption is a major concern with WSNs, the thesis mainly focused on this aspect. The goal of achieving low energy consumption and a long lifetime was considered from the beginning of the work. Whenever a design decision was made, it was aimed to lower energy consumption. A negative impact on other performance measures was considered acceptable in order to push the power consumption to its lowest limit.

An analysis of the power consumption (see Section 5.3) reveals that the lifetime of a sensor node in ideal conditions can be as long as 20 years on a single CR2032 cell. The energy consumed by the operating WSN can be roughly divided into three equal phases:

communication, system operation, and sensing. The energy consumed while running one phase is associated to only that phase. This, however, does not mean that the energy is consumed only by the hardware components used in this phase. Instead, it implies that the whole system needs the energy for operation in that specific phase (e.g., in the sensing phase, although the sensors are used to measure the environment, all the hardware components consume energy).

The energy consumed for communication greatly depends on the network protocol used, which in turn depends on the application. Hence, an extremely lightweight protocol was designed, and the scope for further improvements in power consumption for communication is assumed to be minimal.
More sophisticated applications might require a more complex network protocol, resulting in higher energy consumption. As the network protocol has to provide an application-specific functionality, it cannot be chosen freely. Moreover, because communication plays a large role in the systems energy consumption, the communication protocol should be carefully designed.

One option to further increase the lifetime is to reduce the energy consumed in system operation. During system operation, the most energy is consumed in accessing non-volatile memory (NVM). With the chosen design of the sensor nodes, NVM was necessary. However, the microcontroller lacked this feature; instead, a software-emulated EEPROM using flash memory was utilized. A built-in NVM that is energy-efficient and fast would further reduce energy consumption.

By using sensor fusion and a careful selection of sensors, the energy consumption can be kept low. Sensors with a short measuring time should be preferred to ones with longer measuring times.
In the case of the WSN developed in this work, by using sensor fusion and faster NVM, energy consumption could be cut down to 50 % to 60 % of the current values.

As regards the energy consumption of hardware components, the microcontroller consumes about 80 % of the total energy. The remaining energy is consumed by the transceiver, while the energy consumed by the sensors is negligible. Although the microcontroller is idle when the transceiver and the sensors are operating, it still consumes a large amount of energy. Hence, the microcontrollers energy consumption has a huge impact on the overall energy consumption. The most promising measure to reduce energy consumption is therefore to reduce the base current consumption of the microcontroller. The longer the sensor node stays in the active state, the more pronounced this effect will be.
If, for example, the energy consumption of the microcontroller used in this project could be reduced by 50 %, about two thirds of the total energy could be saved. The system runtime would thus be three times longer. The limiting factor in operational lifetime would then be the chemical lifetime of the battery used.

Section 5.2 analyzes in detail the network protocol used. As it turned out, despite careful planning, the designed protocol did not perform well in the fall detection use case. Simultaneous access to the channel by multiple sensor nodes cause network congestion, which

in turn leads to lost messages and increased power consumption. In the worst case, the power consumption increases about twentyfold, and even then messages get lost. Tests were only performed on small networks; nevertheless, the impact on larger networks is expected to increase exponentially.

However, none of these problems occurs in the monitoring case. Hence, the network protocol has to be redesigned for the fall detection use case in order for it to work well.
In order to achieve the goal of a hardware platform that enables the use of battery-powered, ultra-low-power IoT devices, the new hardware platform designed should include the following:

- A power management unit that enables power cycling of the device
  Ultra-low power consumption by such devices is only possible if the device is power-cycled. The alternative is that all hardware components used support ultra-low-power sleep states. This, however, requires careful selection of the components or additional design overhead to individually power off components that do not support such states.

- Low power consumption during the idle state of the IC
  The results reveal that in the tests performed, the microcontroller is active about 17 % of the time. The remaining 83 % of the time, it waits for certain external events to occur. Hence, the goal should be to reduce the energy consumed in the idle state.

- Dedicated controller units for the operation of the sensors
  The waiting in idle state can be avoided completely by the use of a dedicated sensor controller with reduced functionality and power consumption. This allows for the main CPU to be turned off completely.

- A transceiver unit with hardware support for ACK or retransmission
  Simple and commonly used networking procedures should be supported by the hardware. The CPU should be turned off during low-complexity tasks.

- Careful design of the communication protocol
  Communication will need the most energy thus the design of the communication protocol should be dedicated a good deal of work.

6.1 Future Work

The prototype presented of a WSN is a fully functional system that allows for the structural health monitoring of buildings. In fall detection, however, the WSN does not work as expected. Further tests have to be conducted to reveal to what extent it does work and which areas need improvement.
As the created WSN is a prototype, many aspects can be improved and modified to fit the desired application.

- Improve power consumption
  Energy consumption can further be reduced through sensor fusion. Measurements can be skipped, which decreases the active time of the sensor node. For example,

temperature is used in the measurement of pressure to compensate for temperature-related influences. Currently the temperature is measured twice.

- Improve network performance in the fall detection use case
  Section 5.2 reveals that the network protocol cannot handle multiple nodes waking up simultaneously very well. A way to improve this would be to switch from CSMA/CA to a TDMA strategy, such as in [40], for the fall detection use case. Alternatively, parameters such as the number of retransmits before a packet is dropped could be modified based on the number of nodes expected to wake up simultaneously. To do this, further real-world test runs are needed to analyze the wake-up behavior using a wake-up circuit. At present, there is no such circuit.

- Improve hardware design
  To enhance the overall stability of the system, a new PCB design could be used. Switch-on current such as the voltage drop could then be reduced.

- Extend bidirectional communication
  For productive operation of the system, some additional features will be beneficial. A bidirectional connection can already be established but is limited in function. Thus, it may be beneficial to add a command for enabling buffering of messages, adding post installation calibration, allowing firmware updates, and adjusting system parameters such as the number of retransmits.

- Add support for multiple gateways
  The current system does not support the operation of multiple gateways. To extend the range that the sensor network covers, additional gateways can be used such that each gateway manages a subset of the sensor nodes in the network

List of Abbreviations

6LoWPAN IPv6 over Low power Wireless Personal Area Network 10

AES Advanced Encryption Standard 3

ASK Amplitude Shift Keying 50

CCA Channel Clear Assessment VII

CEPT European Conference of Postal and Telecommunications Administrations 11

CITEL Inter-American Telecommunication Commission 11

CPU Central Processing Unit 2

CRC cyclic redundancy check 29

CSMA/CA Carrier Sense Multiple Access with Collision Avoidance 9

CSS chirp spread spectrum 3

CSV Comma-Separated Values 29

ECC Electronic Communications Committee 12

EEPROM electrically erasable programmable read-only memory 68

ERC Energy Regulatory Commision 6

LoRaWAN Long Range Wide Area Network 3

LPD433 frequency band from 433.05 to 434.79 MHz for LPD 12

LPWAN Low-Power Wide Area Network 3

MOSFET metal-oxide-semiconductor field-effect transistor 32

NVM non-volatile memory 84

OLSR Optimized Link State Routing 10

OOK On-Off Keying 50

OSPF Open Shortest Path First 10

PCB Printed Circuit Board 21

RFID radio-frequency identification 16

RF radio frequency 11

RPL Routing Protocol for Low power and Lossy Networks 10

RR International Telecommunication Union Radio Regulations 12

RSSI Receive Signal Strength Indicator 53

SMD surface-mounted device 41

SoC System on Chip II

SPI Serial Peripheral Interface 34

SRD860 frequecny band from 863 to 870 MHz for SRD 4

SRD Short Range Devices 12

TDMA Time Division Multiple Access 9

UART Universal Asynchronous Receiver Transmitter 23

ULP ultra low power 11

USB Universal Serial Bus

USIC Universal Serial Interface Channel 31

VCO volatile organic compound 18

WSN Wireless Sensor Network II

Bibliography

[1] Charles Saidu, Adamu Usman, and Peter Ogedebe. "Internet of Things: Impact on Economy". In: *British Journal of Mathematics & Computer Science* 7 (Jan. 2015), pp. 241–251.

[2] Amy Nordrum. *Popular Internet of Things Forecast of 50 Billion Devices by 2020 Is Outdated*. en. Aug. 2016. URL: https://spectrum.ieee.org/tech-talk/telecom/internet/popular-internet-of-things-forecast-of-50-billion-devices-by-2020-is-outdated (visited on Feb. 22, 2018).

[3] IHS Markit. *Number of Connected IoT Devices Will Surge to 125 Billion by 2030*. URL: http://news.ihsmarkit.com/press-release/number-connected-iot-devices-will-surge-125-billion-2030-ihs-markit-says (visited on Feb. 22, 2018).

[4] A.J. Bernheim Brush, Bongshin Lee, Ratul Mahajan, et al. "Home Automation in the Wild: Challenges and Opportunities". In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. CHI '11. New York, NY, USA: ACM, 2011, pp. 2115–2124. ISBN: 978-1-4503-0228-9. URL: http://doi.acm.org/10.1145/1978942.1979249 (visited on July 4, 2018).

[5] *LoRa Alliance - Homepage*. URL: https://lora-alliance.org/ (visited on May 11, 2018).

[6] *SK Telecom Leads in IoT Race, Completing Nationwide Network*. URL: http://pulsenews.co.kr/view.php?sc=30800018%5C&year=2016%5C&no=479523 (visited on May 11, 2018).

[7] *Sigfox - The Global Communications Service Provider for the Internet of Things (IoT)*. en. URL: https://www.sigfox.com/en (visited on May 11, 2018).

[8] I. Martin, T. O'Farrell, R. Aspey, et al. "A High-Resolution Sensor Network for Monitoring Glacier Dynamics". In: *IEEE Sensors Journal* 14.11 (Nov. 2014), pp. 3926–3931. ISSN: 1530-437X.

[9] Vidyasagar Potdar, Atif Sharif, and Elizabeth Chang. "Wireless Sensor Networks: A Survey". In: URL: https://espace.curtin.edu.au/bitstream/handle/20.500.11937/42847/132690_StreamGate.pdf?sequence=2%5C&isAllowed=y (visited on Feb. 8, 2018).

[10] Jeongyeup Paek, K. Chintalapudi, R. Govindan, et al. "A Wireless Sensor Network for Structural Health Monitoring: Performance and Experience". In: *The Second IEEE Workshop on Embedded Networked Sensors, 2005. EmNetS-II.* May 2005, pp. 1–10.

[11] Robert Szewczyk, Joseph Polastre, Alan Mainwaring, et al. "Lessons from a Sensor Network Expedition". en. In: *Wireless Sensor Networks*. Lecture Notes in Computer Science. Springer, Berlin, Heidelberg, Jan. 2004, pp. 307–322. ISBN: 978-3-540-20825-9 978-3-540-24606-0. URL: https://link.springer.com/chapter/10.1007/978-3-540-24606-0_21 (visited on May 7, 2018).

[12] Cornelia Kappler and Georg Riegel. "A Real-World, Simple Wireless Sensor Network for Monitoring Electrical Energy Consumption". en. In: *Wireless Sensor Networks*. Lecture Notes in Computer Science. Springer, Berlin, Heidelberg, Jan. 2004, pp. 339–352. ISBN: 978-3-540-20825-9 978-3-540-24606-0. URL: https://link.springer.com/chapter/10.1007/978-3-540-24606-0_23 (visited on May 9, 2018).

[13] Edoardo S. Biagioni and K.W. Bridges. "The Application of Remote Sensor Technology To Assist the Recovery of Rare and Endangered Species". en. In: *The International Journal of High Performance Computing Applications* 16.3 (Aug. 2002), pp. 315–324. ISSN: 1094-3420. URL: https://doi.org/10.1177/109434200201600310 01 (visited on May 11, 2018).

[14] Robert Poor. *Gradient Routing in Ad Hoc Networks*. 2000. URL: https://www.media.mit.edu/pia/Research/ESP/texts/poorieeepaper.pdf (visited on May 11, 2018).

[15] S. Zhuo, Z. Wang, Y. Q. Song, et al. "iQueue-MAC: A Traffic Adaptive Duty-Cycled MAC Protocol with Dynamic Slot Allocation". In: *2013 IEEE International Conference on Sensing, Communications and Networking (SECON)*. June 2013, pp. 95–103.

[16] Q. Wang, K. Jaffrès-Runser, Y. Xu, et al. "TDMA versus CSMA/CA for Wireless Multi-Hop Communications: A Comparison for Soft Real-Time Networking". In: *2016 IEEE World Conference on Factory Communication Systems (WFCS)*. May 2016, pp. 1–4.

[17] T. Clausen, U. Herberg, and M. Philipp. "A Critical Evaluation of the IPv6 Routing Protocol for Low Power and Lossy Networks (RPL)". In: *2011 IEEE 7th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob)*. Oct. 2011, pp. 365–372.

[18] Tim Winter. *RPL: IPv6 Routing Protocol for Low-Power and Lossy Networks*. en. RFC 6550. RFC Editor. URL: https://tools.ietf.org/html/rfc6550 (visited on Feb. 6, 2018).

[19] Bundesnetzagentur. *Frequenzplan*. URL: https://www.bundesnetzagentur.de/SharedDocs/Downloads/DE/Sachgebiete/Telekommunikation/Unternehmen_Institutionen/Frequenzen/Frequenzplan.pdf;jsessionid=75E3119F4A03881728E125560030CF54?__blob=publicationFile%5C&v=9 (visited on Jan. 22, 2018).

[20] International Telecommunication Union. *Radio Regulations - Articles, Edition of 2016*. URL: http://search.itu.int/history/HistoryDigitalCollectionDocLibrary/1.43.48.en.101.pdf (visited on Jan. 22, 2018).

[21] *ERC Recommendation 70-03 (Edition of October 13 2017)*. URL: http://www.erodocdb.dk/docs/doc98/official/pdf/rec7003e.pdf (visited on Jan. 30, 2018).

[22] Holger Karl, Andreas Willig, and Adam Wolisz, eds. *Wireless Sensor Networks: First European Workshop, EWSN 2004, Berlin, Germany, January 19-21, 2004, Proceedings*. Lecture notes in computer science 2920. New York: Springer-Verlag, 2004. ISBN: 978-3-540-20825-9.

[23] Yin Zhang. *Performance Characteristics of Lithium Coin Cells for Use in Wireless Sensing Systems*. June 2012. URL: https://scholarsarchive.byu.edu/cgi/viewcontent.cgi?article=4587%5C&context=etd (visited on May 7, 2018).

[24] Varta AG. *CR2032 Coin Cell Battery - Datasheet*. URL: http://products.varta-microbattery.com/applications/MB_DATA/DOCUMENTS/DATA_SHEETS/DS6032.pdf (visited on May 7, 2018).

[25] Chulsung Park and Kanishka Lahiri. "Battery Discharge Characteristics of Wireless Sensor Nodes: An Experimental Analysis". In: *In Proceedings of the IEEE Conf. on Sensor and Ad-Hoc Communications and Networks (SECON*. 2005.

[26] Zhongwei Deng, Lin Yang, Yishan Cai, et al. "Maximum Available Capacity and Energy Estimation Based on Support Vector Machine Regression for Lithium-Ion Battery". en. In: *Energy Procedia* 107 (Feb. 2017), pp. 68–75. ISSN: 18766102. URL: http://linkinghub.elsevier.com/retrieve/pii/S1876610216317209 (visited on May 7, 2018).

[27] *DeSSnet – Dependable, Secure and Time-Aware Sensor Networks*. URL: http://dessnet.at/de/ (visited on Aug. 3, 2018).

[28] Technologies AG Infineon. *TDA5340 - Data Sheet*. URL: https://www.infineon.com/dgdl/?fileId=db3a30433408410401340ea9a6a4300a (visited on Apr. 11, 2018).

[29] *Raspberry Pi - Teach, Learn, and Make with Raspberry Pi*. en-GB. URL: https://www.raspberrypi.org/ (visited on May 17, 2018).

[30] Yakov Shafranovich. *Common Format and MIME Type for Comma-Separated Values (CSV) Files*. en. URL: https://tools.ietf.org/html/rfc4180 (visited on Apr. 25, 2018).

[31] Technologies AG Infineon. *XMC1100 - Data Sheet*. URL: https://www.infineon.com/dgdl/?fileId=5546d46255dd933d0155e31763e577dc (visited on May 11, 2018).

[32] Texas Instruments Incorporated. *TPL5110 - Datasheet*. URL: http://www.ti.com/lit/ds/symlink/tpl5110.pdf (visited on Apr. 11, 2018).

[33] Technologies AG Infineon. *SmartLEWIS TRX - TDA5340 - Product Brief*. URL: https://www.infineon.com/dgdl/?fileId=db3a3043324cae8c01325d0b47b409d0 (visited on Apr. 11, 2018).

[34] Technologies AG Infineon. *DPS310 - Datasheet*. URL: https://www.infineon.com/dgdl/Infineon-DPS310-DS-v01_00-EN.pdf?fileId=5546d462576f34750157750826c42242.

[35] Texas Instruments Incorporated. HDC1080 - Datasheet. *Datasheet_HDC1080*. URL: http://www.ti.com/lit/ds/symlink/hdc1080.pdf (visited on Apr. 11, 2018).

[36] STMicroelectronics. *Accelerometer LIS331 - Datasheet*. URL: https://www.st.com/resource/en/datasheet/lis3dsh.pdf (visited on Apr. 11, 2018).

[37] Autodesk. *EAGLE*. URL: https://www.autodesk.de/products/eagle/overview (visited on May 14, 2018).

[38] Infineon Technologies AG. *Neue Version Der Entwicklungsplattform DAVE^{TM} Senkt Software-Entwicklungszeit Für XMC-Mikrocontroller von Infineon Beträchtlich - Infineon Technologies*. URL: https://www.infineon.com/cms/de/about-infineon/press/market-news/2015/INFATV201502-033.html (visited on May 14, 2018).

[39] Ibrahim Sayed Ahmad, Ali Kalakech, and Seifedine Kadry. "Minimizing Mobiles Communication Time Using Modified Binary Exponential Backoff Algorithm". In: *International journal of Computer Networks & Communications* 5.6 (Nov. 2013), pp. 85–102. ISSN: 09752293, 09749322. URL: http://www.airccse.org/journal/cnc/5613cnc05.pdf (visited on June 28, 2018).

[40] Rainer Matischek. *Real-Time Communication MAC Protocols for Wireless Sensor Networks: For Automotive and Industrial Applications*. Englisch. 1., Hamburg: Kovac, Dr. Verlag, Apr. 2012. ISBN: 978-3-8300-6349-0.