Master Thesis

A Wireless Sensor Network for Machine Dynamics Performance Monitoring

Alvaro Torres Di Zeo

Master Thesis

A Wireless Sensor Network for Machine Dynamics Performance Monitoring

by

Alvaro Torres Di Zeo

in partial fulfillment of the requirements for the degree of

Master of Science
in Embedded Systems

at the Delft University of Technology,
to be defended publicly on Monday August 26, 2019 at 10:00 AM.

Acknowledgements

I want to express my gratitude to my advisor dr. Stoyan Nihtianov for his attentive guidance and support during this project. To my teammates from the research group Reza and Swarna who were always keen to discuss any issues in the project and provide advice. To the people at the analog electronics testlab at ASML who made me feel welcome in the company and whom I could always consult whenever I had hardware-related questions. To Marc van Duijnhoven, Martin Buysse, and Ruud Klaver from the System Dynamics group at ASML for proposing the topic and providing feedback throughout the project. To Michiel Dorrepaal and Gergely Berces at VDL ETG for allowing us to use the Wafer Handler test rig and for their help during the experiments. To my friends from TU Delft who made my time in the Netherlands a happy experience. Finally, I want to thank my parents, my brother, and my partner for their unconditional support and love, this thesis is dedicated to them.

Alvaro Torres Di Zeo
Delft, August 2019

Contents

List of Tables

List of Figures

Abbreviations and Acronyms

API	application programming interface. 23
BLE	Bluetooth Low Energy. 18, 21, 22, 24, 26, 27, 33, 38, 40, 47
COTS	commercial off-the-shelf. 1, 14, 15, 18–20, 46, 47
CPE	Command and Packet Engine. 23
CSV	comma-separated values. 41
EMI	electromagnetic interference. 26, 47
MCS	Modulation and Coding Scheme. vii, x, 17, 18, 38, 39, 41, 43–46
MCU	microcontroller unit. 13, 14, 18–20, 22, 23, 27, 28, 30–32, 34, 45, 47
PCB	printed circuit board. 15, 20, 21
PER	Packet Error Rate. vii, x, 17, 41–44, 46
RAT	radio timer. 23, 24, 28
RTC	real-time clock. 22–24, 27, 28, 44–46
RTOS	real-time operating system. 22
TDMA	time-division multiple access. 25
WH	Wafer Handler. x, 1, 3–11, 13, 15, 26, 27, 37, 38, 41, 45–47
WSN	Wireless Sensor Network. vii, 1, 4, 7–9, 13, 17

1 Introduction

The availability of photolithography machines is key in the semiconductor industry, as downtime generally incurs in immense economic loss. Time to market is also very important for suppliers of photolithography systems, such as ASML. Photolithography machines include numerous measurement systems that are frequently used for qualification and troubleshooting, most of which are required during normal operation. Due to increased costs and complexity, it would not be practical to include dedicated sensors for every machine performance parameter relevant for diagnosis, therefore many parameters regarded as non-critical are not monitored. However, in some cases the information about the behaviour of these parameters can be very valuable to find the root cause of a failure or defect promptly. In these cases, a Wireless Sensor Network (WSN) would allow for a temporary installation of a measurement system to monitor such parameters.

This report presents the design of a WSN for monitoring the dynamics performance of the Wafer Handler (WH), which is one of the major subsystems of a photolithography machine. The scope of the project includes:

- The selection of a radio technology on which the network is based on.
- The hardware design of a wireless sensor node for measuring acceleration, based on commercial off-the-shelf (COTS) components.
- The identification and firmware implementation of key principles in which the communication protocol should be based on according to the requirements of the application.
- The estimation of the power consumption and lifetime of the network.
- The design and execution of experiments to assess the reliability of the proposed solution.

Among the requirements of the system, the size of the sensor nodes, the network synchronization accuracy, and the maximum power dissipation stand out as the main challenges. As the available space inside the WH is very scarce, the sensor nodes must have a compact form factor to fit in the locations where they are meant to be installed. The reliability of the system is greatly determined by its capacity to remain synchronized with relatively high accuracy during a measurement. This can be difficult to achieve in a harsh environment such as the inside of a machine, where interference and signal fading are expected to recurrently cause the loss of packets used for network synchronization. Packet re-transmission should not be abused to alleviate the problem, as the sensor nodes are required to operate in vacuum and overheating can become a problem.

This report describes how the aforementioned challenges were addressed. It is organized in the following way:

- Chapter 2 starts with a description of the problem, after which the requirements of the system are presented. This chapter ends with an overview of the related work.

- Chapter 3 describes the proposed solution, starting with an overview of the architecture of the network and the nodes that comprise it. The hardware design is then described, including the criteria for the selection of the different components. A description of the embedded software architecture and implementation is also provided. This section is followed by the description of the proposed communication protocol, which includes network synchronization. The last section provides an estimation of the power consumption and lifetime of the network.

- Chapter 4 is the last one, it describes the test setup and the results of the experiments carried on to evaluate the link quality inside the machine and the accuracy of the synchronization protocol. Finally, conclusions and future work are provided.

2 Problem Analysis and Related Work

2.1. Overview

The availability of photolithography machines is of critical importance in a semiconductor fabrication plant as machine down-time can result in immense economic loss. Accumulated losses are normally in the order of millions of Euros per failure [2], implying that great effort must be put in reducing machine downtime. Effective and prompt fault diagnosis plays a fundamental role to this end. Because of the extraordinary complexity of modern photolithography machines fault diagnosis is an arduous task. Improvements in current methods and tools to ease the diagnosis of such machines are thus valuable contributions. Moreover, time to market is imperative in the photolithography industry [2], and machine qualification accounts for a significant part of the development cycle. A more efficient fault diagnosis could potentially reduce the qualification phase and therefore the time to market.

ASML's photolithography machines are organized into several subsystems, the Wafer Handler (WH) being one of them. The WH is the entry and exit point of the machine, delivering wafers to be exposed to the next module and unloading them after exposure. The WH is divided into an atmospheric side and a vacuum side, connected through *load-locks* that allow transferring wafers from one side to the other. A simplified diagram of the WH is presented in figure 2.1. Wafers are loaded to the *pre-alignment unit* (PA) where they will be transferred by the *load robot* (LR) to the *input load-locks* (LL-I) making them available to the *in-vacuum robot* (IVR), which in turn delivers the wafers to the next subsystem in the machine. After exposure, the *in-vacuum robot* places the wafers in the *output load-lock* (LL-O) to be handed over to the *discharge unit* (DU) by the *unload robot* (UR). A more detailed description of the WH can be found in [1].

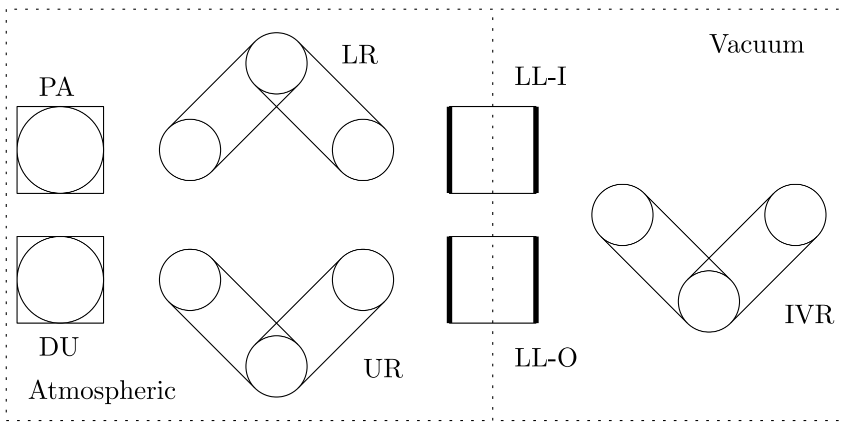


Figure 2.1: Wafer-handler, adapted from [1].

When a failure occurs at the WH its dynamics performance must be monitored in order to diagnose the problem. This entails synchronous acquisition of acceleration signals at dispersed locations inside the WH around the time of an event, such as a machine stop or when an acceleration threshold is exceeded. Performance monitoring can last from a few hours, when bringing up a machine after a failure, to several days in the case of qualification tests. The devices employed for signal acquisition are not required during normal operation hence having dedicated built-in sensors would be excessive. A *standalone* system is rather preferred, avoiding to further increase machine costs and complexity. It must be considered that the devices comprising such system must have a reduced volume and mass, given that the available space in the WH is scarce and that excessive weights would affect its dynamics. Furthermore, any device operating in vacuum should dissipate very little power in order to avoid overheating.

Possible approaches for a standalone measurement system include: *wired sensor networks*, *data loggers*, and WSNs. Among these, wired sensor networks can provide the highest performance in terms of lifetime, synchronization accuracy, and availability of the measurements. *Online monitoring* and precise synchronization are generally straightforward to achieve in these networks. However, the space inside the WH is very packed and cable routing can be difficult and time consuming, potentially defeating the purpose of reducing machine downtime. Cabling can also be restrictive when measurements must take place on rotating or moving parts, and worse, cable stiffness can be a source of error when measuring acceleration.

Battery-powered data loggers and *radio-based* wireless sensors remove the need for cabling. Nevertheless, battery lifetime becomes a concern when measurements must be carried out for long periods (days or weeks). Data loggers generally consume less power and can operate for longer periods than radio-based wireless sensors. However, data loggers do not allow for online monitoring as the acquisition phase must finalize before having access to the samples. Synchronous sampling is also limited as data loggers cannot be re-synchronized once installed. Because of these reasons data loggers are not well-suited for dynamics performance monitoring.

WSNs seem to be a better alternative as synchronization and online monitoring become possible. However, there is a trade-off between battery lifetime, and synchronization accuracy and availability of the measurements. Periodic communication is needed to re-synchronize the nodes in the network as local clocks will drift with respect to real time due to component tolerance. As mentioned before, not all samples must be transferred but only the ones acquired around the time of an event. Since the power consumption of a wireless sensor depends greatly on the radio utilization, battery lifetime will be determined significantly by the re-synchronization period and the frequency of the events. Link quality will also affect battery lifetime as power consumption will increase due to poor quality links that require frequent re-transmissions or higher transmission power. Link quality is generally expected to be low inside a machine as metallic enclosures, strong electromagnetic fields, and obstacles between transmitter and receiver are abundant in this environment. All of these conditions aggravate common issues in wireless communications such as signal reflection, fading, and interference. On the other hand, as the volume of the WH amounts to a few cubic meters the distance between any pair of radios in the network will be relatively short (a few meters). This would compensate to some extent for the adverse environment inside the WH, e.g. it might be possible to achieve reliable communication using relatively low transmission power.

Contrary to other alternatives, WSNs offer an opportunity to realize a measurement system for dynamics performance monitoring at the WH. Nevertheless, the feasibility of such system is not immediately clear and must be investigated. The main problem consisting of how to achieve accurate synchronization and prompt sample retrieval while providing a battery lifetime of several days, considering adverse conditions for wireless communications and stringent limitations on the volume and mass of the sensor nodes. In order to offer a solution to the problem at hand, this work proposes a design for such WSN and provides an implementation as a proof-of-concept.

2.2. Scope

This project focuses mainly on the problems related to the design of a WSN requiring low-power network synchronization and event-based data transfer inside the WH of ASML's photolithography machines. Hardware design for this WSN includes only the sensor node. **This work does not consider:**

- Vacuum compatibility, with the exception of power dissipation.
- The performance of the sensors, from an instrumentation point of view.

- Characterization of the electromagnetic spectrum inside the WH.
- The interface between the sink node and the machine.
- Hardware design for the sink node.
- Network security.

2.3. Functional and Performance Requirements

This section presents the functional and performance requirements for a standalone measurement system for dynamics performance monitoring at the WH. The requirements have been divided into three categories: measurement, data retrieval, and physical requirements.

2.3.1. Measurement

Diagnosing a failure at the WH requires the measurement of acceleration signals at least at 10 different locations inside of either the atmospheric or the vacuum side, as normally these are tested separately. In order to use the acceleration data for dynamics performance analysis, the sample rate should be the same at all locations and must be at least 1 kHz. Additionally, acquisition at all locations should be synchronous with a maximum difference between the sampling times at any two locations of $20/360$ of a sampling cycle, which at a sampling rate of 1 kHz gives $55.\overline{55}$ microseconds. Figure 2.2 shows the sampling times at two different locations, the sampling time offset is denoted by δ. Performance monitoring normally takes a few hours when bringing up the machine after a failure, this is not the case during qualification where tests can last a few days. Because of this, the system must be continuously available for at least 48 hours. Continuous measurement however is not required and the system should allow the user to start or stop the acquisition at any time. A summary of the functional requirements related to the signal acquisition is presented in table 2.1, the performance requirements are included in table 2.3.

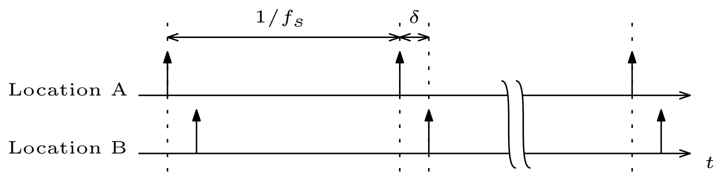


Figure 2.2: Delta between the sample times at two different measurement locations.

Table 2.1: Functional requirements related to signal acquisition.

Requirement ID	Description	Priority
FR-01	The system samples acceleration signals at multiple locations in the WH. All locations belonging either to the atmospheric or to the vacuum side of the WH.	MUST
FR-02	The system samples the acceleration signals synchronously and at the same rate at all measurement locations.	MUST
FR-03	The system starts or stops the acquisition whenever this is signaled by the user.	MUST

2.3.2. Data Retrieval

Two different events can trigger the retrieval of the acceleration data whenever acquisition is enabled, these are: a *machine stop* and exceeding an *acceleration threshold* at any of the measurement locations. When a machine stop occurs, this is notified to the measurement system by the user and the samples collected briefly **before** the time of the event are retrieved. Conversely, when an acceleration threshold occurs the measurement system reports it to the user, samples acquired briefly **after** the event are then retrieved. These scenarios are illustrated in figure 2.3. In both cases, the samples collected at all locations during the last or next 30 seconds (depending on the event) must be transferred to the user. At a sampling rate of 1 kHz, 30 000 samples are collected in 30 seconds per measurement location, which gives a total of 300 000 samples considering all locations. There are no hard constraints on the total time to retrieve all samples, however this time should be as short as possible given that the test cannot be resumed until data transfer has finished. A total data retrieval time of a few minutes is tolerated. The functional requirements related to data retrieval are summarized in table 2.2, the performance requirements are included in table 2.3.

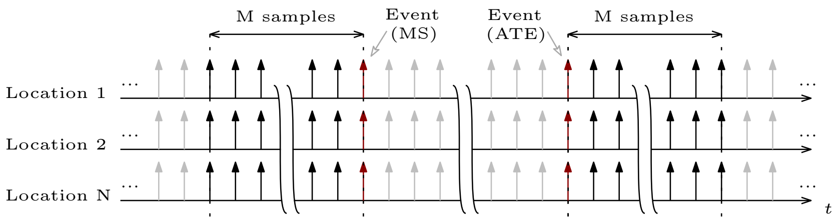


Figure 2.3: Samples to be retrieved when a machine stop (MS) occurs or an acceleration threshold is exceeded (ATE); event time (red), required samples (red and black), don't care (grey).

Table 2.2: Functional requirements related to data retrieval.

Requirement ID	Description	Priority
FR-04	During acquisition and whenever a machine stop occurs, the system retrieves the samples of all measurement locations acquired shortly before the time of the event.	MUST
FR-05	During acquisition and whenever an acceleration threshold is exceeded at any location, the system retrieves the samples of all measurement locations acquired shortly after the time of the event.	MUST

Table 2.3: Performance requirements related to measurement and data retrieval.

Parameter	Description	Min.	Max.	Unit
N	Number of measurement locations	10	-	-
f_s	Sampling frequency	1	-	kHz
δ	Clock offset between any pair of measurement locations	-	55.55	μs
L	System lifetime	48	-	hours
M	Number of samples retrieved per event from each measurement location	$30 \cdot f_s$	-	-

2.3.3. Physical Requirements

Overheating of electronic components can easily occur in the vacuum side of the WH as radiation is the only way to dissipate heat for a standalone system, the power consumption at each measurement location should therefore be limited. As a rule of thumb, any electronic device inside the vacuum side of the WH without a dedicated cooling system should not consume more than 10 mW on average to prevent overheating. Additionally, the sensing devices at each location should be compact and lightweight to fit in the available space and to prevent affecting the dynamics of the WH. It is required that the dimensions and mass of the sensing devices do not exceed 2x2x2 cm and 30 g respectively. The physical requirements are listed in table 2.4.

Table 2.4: Physical requirements.

Parameter	Description	Min.	Max.	Unit
P	Average power consumption at each measurement location	-	10	mW
m	Mass at each measurement location	-	30	g
V	Volume at each measurement location	-	2x2x2	cm^3

2.4. Related Work

Experimental modal analysis is widely employed to determine the inherent dynamic properties of mechanical systems. Traditional methods consist of applying known excitation forces with varying oscillation frequency to a mechanical structure and measure the displacement, velocity or acceleration at multiple locations to determine the frequency response of the system [3]. By doing this it is possible to find the natural frequencies of a system and their associated mode shapes, which are specific deformation patterns that occur when the oscillation frequency of the excitation forces correspond to one of the natural frequencies of the structure. More recent methods are able to accurately estimate the modal properties of a system based only on its response under operating conditions, removing the need to generate and measure external excitation forces [4]. Knowing the modal properties of a mechanical system is often extremely useful during the design and validation processes as it allows to clearly identify weakness and improvement areas. Data collected during operation combined with the modal properties of the system can allow to diagnose and solve structural dynamic problems. Diagnosing a fault at the WH involves the analysis of the response of the system in the frequency domain to determine its modal properties and in the time domain to detect acceleration spikes in the robots that move around the wafers.

WSNs have been used for experimental modal analysis mainly in *structural health monitoring* and *machine condition monitoring* applications. Compared to typical WSNs applications, experimental modal analysis applications have more demanding requirements on network synchronization accuracy and sampling rate. Typical sampling rates range from hundreds to a few thousand samples per second, and the required synchronization accuracy lays in the order of tens to hundreds of microseconds. Event-based data collection is often preferred due to the relatively high volumes of data, as a result of the high sampling rates, that normally do not allow for raw sensor data streaming. Long delays in data collection ranging from minutes to hours are generally acceptable provided that there is no data loss. In-sensor fault location and diagnosis is challenging because of the relatively high computational complexity of modal analysis algorithms which aggregate and process time-correlated samples from multiple sensors. These requirements coincide with the ones for fault diagnosis at the WH presented in section 2.3.

In the case of structural health monitoring, network scalability is particularly important as a large number of nodes is required to cover massive civil structures such as bridges, buildings, and stadiums. Multi-hop routing is also often required in order to relay data packets from nodes outside of the range of the sink node, and also to account for broken communication links due to RF signal propagation along structures based on materials such as concrete and steel. Existing structural health monitoring systems consist of up to 70 nodes with a number of hops ranging from 1 to 47 [5]. While the number of nodes and the area to be covered is normally lower in machine condition monitoring applications, multi-hop communication can be still required mainly when link quality is deficient due to harsh environmental conditions such as strong electromagnetic interference and multipath fading [6]. The environment inside the WH can be even worse as the network is completely enclosed by metallic materials and the space is very packed. However, the distance between the sensor nodes and the sink is very short (a few meters at most) and single-hop communication might be possible. This is generally desired as communication overhead, and in turn power consumption, would be reduced. Another important distinction is that for structural health or machine condition monitoring sensor node size is not as critical as in fault diagnosis for the WH. Conversely, network lifetime has a higher priority; energy harvesting hardware and batteries with relatively high capacity can be generally used without worrying about their volume and weight.

In the remainder of this section related work on network synchronization, reliable data collection, and wireless accelerometer miniaturization will be discussed.

2.4.1. Wireless Sensor Networks for Experimental Modal Analysis

A comprehensive survey on WSNs for structural health monitoring, including a review of experimental work with real structures, was presented in [5]. The specifications and performance of WSNs presented in these experimental studies are included in table 2.5, civil structures that were monitored include: bridges [7–9], a football stadium [10], buildings [11, 12], and the base of wind turbines [13]. The sensor nodes used in all of these networks are mainly intended for measuring acceleration signals.

In most cases the number of nodes installed was equal or higher than required for fault diagnosis at the WH, networks installed in bridges had the highest node and hop count due to their long linear structure. This is particularly notable in the network presented by Kim et al. [7], where there are 64 nodes and 44 hops, however this together with a sampling rate of 1 kHz results in a very high delay in data collection (9 hours in total). The networks proposed by Cerotti et al. [12] and Jang et al. [8] have shorter delays but it should be noted that the hop count and the sampling rates are much lower, theses delays (8 and 30 minutes respectively) could be still too high when testing at the WH, especially when machine downtime is critical. With the exception of [7], the sensor nodes have lower sampling rates than required for diagnosing a fault in the WH. The reported network lifetime was relatively high, ranging from 2 to 3 months, mainly due to event-based measurement and data collection, and the use of high-capacity batteries and energy harvesters. The latter are not feasible at the WH because of their size.

Network synchronization was not implemented by Chintalapudi et al. in [11], instead the sink computes the sampling times based on timestamps provided by the sensor nodes and estimations on the communication latency. Kim et al. [7] demonstrated the feasibility of achieving network-wide synchronization with microsecond accuracy in a very dense network with many hops. A higher synchronization accuracy was achieved by Yu et al. [9] using GPS radios, which is only possible when the sensors are installed outdoors and comes at the cost of higher power consumption and increased volume of the sensor nodes. In the other works synchronization accuracy was either not reported or it was lower than required as specified in section 2.3.

While the authors of these experimental studies concluded that in general the deployed networks were appropriate for condition assessment of civil structures, most of the proposed designs are not directly applicable for acceleration measurement at the WH mainly due to insufficient sampling rates and synchronization accuracy. The only exception is the network proposed by Kim et al. [7] that has a sampling rate of 1 kHz and a synchronization accuracy of 10 microseconds. Nevertheless, the network and protocol design could be simplified considering that much fewer measurement locations are needed in the WH. In a single-hop network, traffic could be greatly reduced as packet routing is no longer required, giving the opportunity to extend the network lifetime by reducing the power consumed in the wireless communication.

Table 2.5: Performance parameters of WSNs for structural health monitoring used in real scenarios.

Study	No. of Nodes	No. of Hops	Sampling Rate [Hz]	Sync. Accuracy [μs]	Retrieval Delay [minutes]	Network Lifetime [days]
Chintalapudi2006 [11]	10	4	200	No sync.	N/A	N/A
Kim2007 [7]	64	44	1000	10	540	70
Ceriotti2009 [12]	16	6	200	732	8	90
Jang2010 [8]	70	1	50	N/A	30	60
Swartz2010 [13]	4	1	500	N/A	N/A	N/A
Yu2012 [9]	8	1	120	0.04	N/A	N/A
Phanish2015 [10]	10	2	100	300	N/A	N/A

Machine condition monitoring generally involves the measurement of multiple quantities with the purpose of determining if industrial machinery requires maintenance, such quantities include acoustic noise, current consumption, and vibration. WSNs for machine condition monitoring have been studied in less extent than structural health monitoring. Most works focus on health assessment of electric motors where the main benefit of wireless communications is removing the restrictions of cabling when performing measurements in rotating parts. In many cases, network synchronization is not necessary as generally it is possible to determine the health of an electric motor based only on its natural frequencies. These, conversely to mode shapes, can be determined without requiring time-correlated acceleration samples from multiple locations in the machine, in some cases even samples from a single sensor node can suffice. Due to the much lower node count and the smaller area that must be covered by the sensor networks, in comparison with structural health monitoring applications, single-hop communication in machine condition monitoring is possible in most cases. Required sampling rates are higher as common damage conditions in electric motors produce vibrations with frequency components in the order of kilohertz that must be detected for an effective diagnosis [14]. The specifications of WSNs used in experimental studies on motor vibration monitoring are included in table 2.6, only networks with more than one sensor node have been considered.

Table 2.6: Performance parameters of WSNs for machine condition monitoring.

Study	No. of Nodes	No. of Hops	Sampling Rate [Hz]	Sync. Accuracy [μs]	Retrieval Delay [minutes]	Network Lifetime [days]
Hou2012 [15]	2	1	3100	No sync.	N/A	N/A
Huang2015 [16]	3	1	20000	0.19	N/A	0.375
Zhang2016 [17]	4	1	1000	N/A	N/A	N/A

Hou et al. [15] implemented a network where asynchronous raw data is processed locally at the sensor node for feature extraction, the result from all sensor nodes is combined at the sink using a neural network classifier to determine the condition of a motor. In the network proposed by Zhang et al. [17] raw data from four nodes is collected by the sink, however network synchronization, communication delay and network lifetime were not discussed. Huang et al. [16] proposed a network of three nodes supporting a sampling rate of 20 kHz and sub-microsecond synchronization accuracy that was validated by determining the natural frequencies and mode shapes of a steel frame. Sustaining this sampling rate required a high capacity memory (2 GB) with a high-speed bus (SDIO) to store the relatively large volumes of data produced before transmission to the sink node, data collection delay was not discussed. Network lifetime was reported to be 9 hours, considerably lower than in the case of structural health monitoring applications. While WSNs for machine condition monitoring support sampling rates high enough for diagnosing a failure at the WH, the reduced number of nodes, and in some cases the limited lifetime or the lack of synchronization impede applying directly these designs for a measurement system for the WH.

2.4.2. Wireless Accelerometers for Experimental Modal Analysis

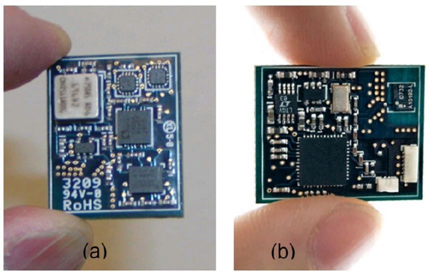
The volume and mass of sensor nodes used in WSNs for experimental modal analysis applications are usually not a concern. A higher priority is given to network lifetime, which can be extended by using batteries with relatively high capacity and energy harvesting hardware, however this results in relatively bulky and heavy sensor nodes. Only a few works have focused on the miniaturization of wireless accelerometers [18, 19]. The specifications of wireless sensor nodes used in studies published after 2010 have been included in table 2.7, the battery package has been provided when available to give an idea of the size and mass of a node as this information is usually incomplete or not explicitly reported.

MEMS-based wireless accelerometers used for structural health monitoring from 2006 to 2016 have been compiled and reviewed in [20], from which [21–25] have been included in table 2.7. This survey discusses the suitability of each hardware platform mainly with respect to sensor performance, power consumption, and network scalability. As noted before, most wireless sensor nodes for structural health monitoring do not provide the required sampling rate for fault diagnosis at the WH. Moreover, multiple AA/AAA batteries, and even lead-acid batteries[22, 25], which have a relatively high volume and mass are common in these applications to achieve network lifetimes in the order of weeks or months. More recently the Xnode platform has been proposed for structural health monitoring (an overview can be found in [26, 27]), this platform supports a much higher sampling frequency than typical structural health monitoring. However, due to its size and volume it could not be used in the WH. The same is true for the sensor node proposed by Huang et al. [16] for machine condition monitoring which has a relatively high sampling frequency but its size is excessive for the WH.

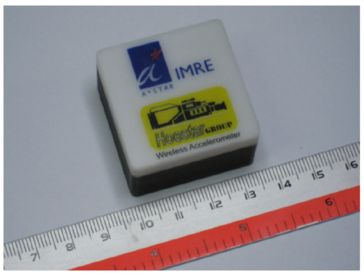
Table 2.7: Performance parameters of wireless accelerometer used in academic studies.

Study	Platform Name	Sampling Rate [Hz]	Dimensions [mm]	Mass [g]	Battery Package	Radio
Jo2011 [21]	SHM-H	280	> 48x36	N/A	3 x AAA	802.15.4
Chae2012 [22]	u-Node	60	N/A	N/A	Pb batt.	802.15.4
McGinnis2012 [18]	-	1 k	> 19 x 24	18	Coin cell	ShockBurst
Hu2013 [23]	S-Mote	500	N/A	N/A	2 x AA	802.15.4
Sabato2014 [24]	ALE	3 k	N/A	N/A	N/A	Analog FM
Huang2015 [16]	WSNG2	20 k	> 50 x 50	N/A	N/A	802.15.4
Kohler2015 [25]	ShakeNet	500	N/A	N/A	Pb batt.	802.15.4
Shen2016 [19]	-	80 k	36 x 36 x 21	67.5	Coin cell	N/A
Zhu2018 [26]	Xnode	16 kHz	150 x 70 x 50	750	N/A	802.15.4

McGinnis et al. [18] and Shen et al. [19] have made efforts towards the miniaturization of wireless accelerometers. In [18] a wireless *inertial measurement unit* (IMU) was used to obtain the angular velocity of a rigid body during free flight. The sensor node allowed for a sampling rate of 1 kHz and the samples were sent to a laptop using a ShockBurst radio, a proprietary transceiver from Nordic Semiconductor, which supports a maximum data rate of 2 Mbps. The board containing all components occupies an area of 19x24 mm^2, and together with a coin cell battery and the housing weights 18 grams. However, under uninterrupted use the battery lifetime was only 4 hours. In [19] a miniature wireless accelerometer for vibration measurement for machine condition monitoring was presented. The study focused mainly on the design of the sensor and the evaluation of its performance, including the effect of the device package on the dynamic response of the sensor and on the rejection of electromagnetic interference. The package including all components has a volume of 36x36x21 mm^3 and its mass is 67.5 grams. No details were provided about the performance of the wireless communication. Pictures of the miniature accelerometers are shown in figure 2.4.



(a) McGinnis et al. [18]



(b) Shen et al. [19]

Figure 2.4: Miniature wireless accelerometers.

Several commercial wireless accelerometers for machine condition and structural health monitoring are also available, their specifications are presented in table 2.8. With the exception of Monnit's and Resensys' sensor nodes the commercial wireless accelerometers offer higher sampling rates with respect to the requirements presented in section 2.3. Benair's and Valmet's sensor node are based on IEEE 802.11 (WiFi) radios which allow for data rates up to 72 Mbps, much higher than data rates of typical wireless sensor nodes. However, this comes with much higher power consumption, namely 1 W in the case of Valmet's sensor node. Monnit's sensor node can communicate in the 433 and 900 MHz frequency bands (the wireless technology used is not specified), and the other wireless accelerometers are based on the 802.15.4 standard and operate in the 2.4 GHz band. Most of the sensor nodes allow for network synchronization but the maximum timing errors are greater than a millisecond, the only exception is Microstrain's sensor nodes. The IEPE-Link-LX and G-Link-200 wireless accelerometers from Microstrain have a maximum synchronization error of 32 and 50 microseconds respectively.

Table 2.8: Performance parameters of commercial off-the-shelf wireless accelerometers.

Part num.	Manufacturer	Sampling Rate [Hz]	Dimensions [mm]	Mass [g]	Battery Capacity [mAh]	Radio
AX-3D	Benair	2 k	35 x 59 x 65	220	780	802.11
G-Link-200	MicroStrain	4 k	46 x 44 x 43	122	3600	802.15.4
IEPE-Link-LXRS	MicroStrain	104 k	94 x 79 x 21	114	650	802.15.4
MNS-9-IN-AC-GM	Monnit	800	94 x 58 x 35	133	1500	433, 900 MHz
SenSpot	Resensys	400	50 x 50 x 34	120	N/A	802.15.4
WVS-100	Valmet	3.2 k	112 x 45 x 42	350	3100	802.11

All the commercial wireless accelerometers presented exceed the limits on mass and volume specified for a dynamics measurement system for the WH. Microstrain's G-Link-200 has the most compact form factor with a volume of 46x44x43 mm^3 and a total mass of 122 grams. It is also important to note that the gateways (sink nodes) used along with these sensor nodes are bulky and heavy, and have much higher power consumption than the maximum allowed at the vacuum side of the wafer handler. The only exception is the WSDA-200-USB gateway from Microstrain which has a compact form factor (58.2x20.3x10.8 mm).

3 Design Approach and Realization

3.1. Overview

This chapter presents the approach followed for the design of a WSN for dynamics performance monitoring at the WH. This entails the hardware design for the sensor nodes, and the design of the embedded software for both the sink and the sensor nodes.

Figure 3.1 shows the proposed system architecture; the network follows a star topology, and is comprised of ten sensor nodes and a sink node, the former can communicate with the user through the *machine interface*. As mentioned in chapter 2, the sink node and all sensor nodes will be located either in the vacuum or the atmospheric side of the WH and will be separated by a relatively short distance (a few meters). Given this proximity, it is expected that all sensor nodes will be within the range of the sink and therefore single-hop communication is possible. The operation of the network can be controlled by the user through the sink node, which exchanges control messages with the user, and gathers the acceleration data from the sensor nodes to pass it to the user whenever an event occurs. As stated in section 2.2, the hardware design of the sink node and its interface with the machine are beyond the scope of this work.

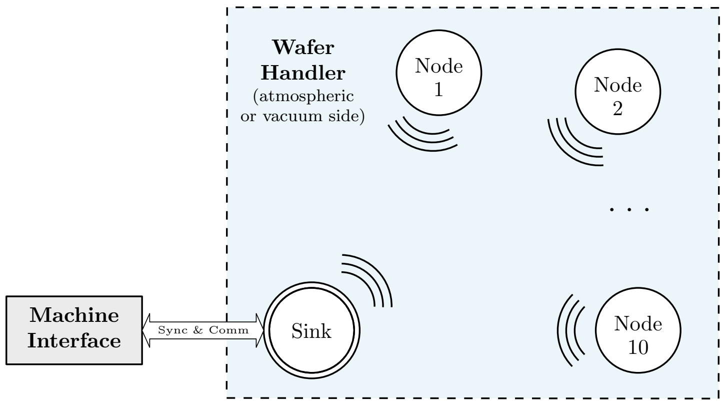


Figure 3.1: Proposed system architecture; the network follows a star topology

In general, wireless sensor nodes follow the hardware architecture shown in figure 3.2. Almost every sensor node used in the WSNs presented in section 2.4 adheres to this generic architecture. Wireless sensor nodes are composed mainly of a microcontroller unit (MCU), a radio (RF wireless transceiver), and one or multiple memory units, sensors, and battery cells. Memory units can be omitted in some applications, however in experimental modal analysis they are usually required in order to buffer the relatively high volumes of data produced, which could not be streamed to the sink nor stored in the limited memory available in the MCU. In early designs the MCU and the radio usually belonged to different integrated circuits that communicated through a standard serial bus, more recently it is common to find MCUs which include an RF transceiver. Section 3.2 discusses the hardware design of the sensor node, including the selection of COTS parts for each component, a comparison of wireless technologies, and a description of the overall design.

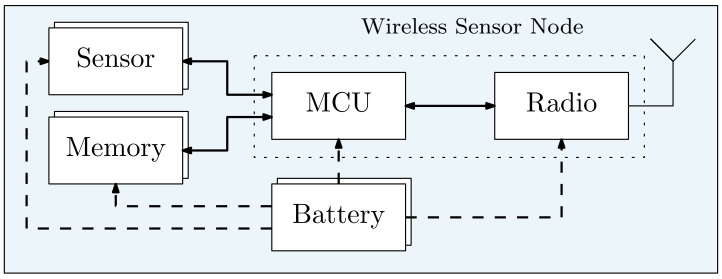


Figure 3.2: Generic wireless sensor node hardware architecture.

Based on the hardware architecture of the sensor node and the requirements presented in section 2.3 a general hardware and software stack can be defined for the MCU, this stack is shown in figure 3.3. At the bottom of the stack there are the peripherals of the MCU (e.g. timers, serial interfaces, radio) which, in accordance with the hardware architecture, are controlled by the *sensor*, *memory* and *radio drivers*. These device drivers aim to abstract the specifics of the devices and offer a simple interface that can be used by upper layers; examples of functions of such interfaces could be `Radio_Turn_On()`, `Memory_Store_Byte_Array()`, `Sensor_Get_Sample()`, etc. In the layer on top of the device drivers are the *communication protocol* and *measurement* blocks. The communication protocol implements link-level functionality such as clock synchronization, packet formatting, and packet retransmission. The measurement block transfers the data from the sensor to the memory and keeps track of the ordering and timing of the samples. Finally, the *application* layer integrates the measurement and the communication functionality to implement the behavior concerning the event-based measurement and data collection described in section 2.3. In figure 3.3 the measurement, memory driver, and sensor driver blocks have been greyed out as they have not been considered in detail in this work, the application and communication protocol blocks have a grey tilling pattern as only the critical aspects regarding the wireless communication have been considered as it will be explained in section 3.3.

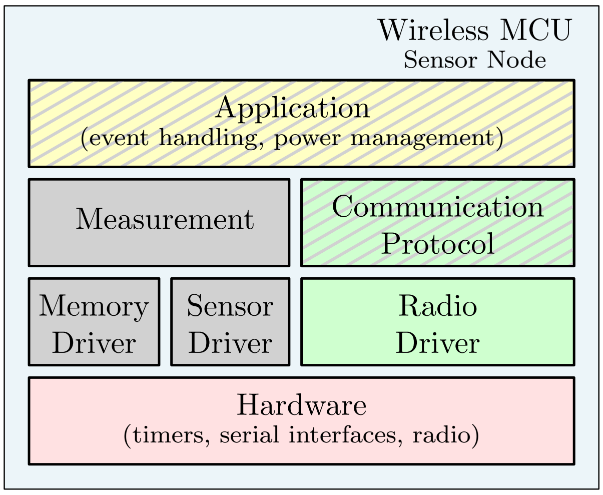


Figure 3.3: Sensor node hardware and software stack.

Similarly, a general hardware and software stack can be defined for the MCU in the sink node, this is shown in figure 3.4. In this case, the communication protocol, radio driver and application blocks remain and have the same purpose, but the measurement block and drivers have been replaced by the *machine interface* block and its associated device drivers, which are also greyed out as they will not be discussed in this work. In section 3.3 the software architecture for both the sensor and sink nodes is presented in detail, and the design and implementation of the communication protocol are explained in section 3.4. Based on the selected hardware and the communication protocol used, power consumption and network lifetime are estimated in section 3.5.

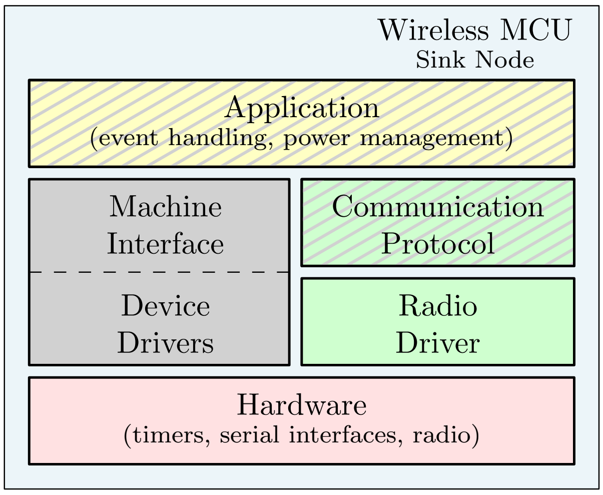


Figure 3.4: Sink node hardware and software stack.

3.2. Sensor Node Hardware Design

The design of the sensor node comprises the selection of COTS components for the sensor, memory, microcontroller, radio, and battery, and the design of a printed circuit board (PCB). Recalling the physical requirements from section 2.3, the design goals include having an average power dissipation of less than 10 mW and a compact form-factor of less than 20x20x20 mm^3 with a maximum mass of 30 grams.

This section does not discuss the selection of the accelerometer as the measurement performance of the sensor nodes is beyond the scope of this project. However, the specifications of the accelerometer are required to dimension other components of the sensor node, e.g. the capacity of the memory based on the sample size, or the capacity of the battery based on the current consumption. *The stakeholders of this project have provided a part number for the accelerometer, which has been partially qualified for monitoring the dynamics performance of the WH.*

3.2.1. Accelerometer

The accelerometer that was proposed for this application is the ADXL355 from Analog Devices. This 3-axis MEMS accelerometer is a *smart transducer* that provides a digital interface for communication with a host device, it also includes an analog-to-digital (ADC) converter, a digital filter, a voltage regulator, a temperature sensor, and a FIFO with 32 positions to buffer the samples. The key performance parameters of the ADXL355 are presented in table 3.1. The maximum sampling frequency supported is 4 kHz, however the device will operate at the minimum sampling frequency required, which is 1 kHz, to limit the volume of data produced and the power consumption. The ADC of the accelerometer has three channels (one per axis) and a resolution of 20 bits, which results in 60 bits of data produced every sampling period. As stated in section 2.3, whenever an event occurs acceleration data from the last or next 30 seconds with respect to the event time must be retrieved. Considering a sampling frequency of 1 kHz and a sample size of 60 bits, the total amount of data that must be transferred per event by each sensor node is 1.8 Mbits.

The digital interface of the sensor includes a communication bus that supports the SPI and I2C standards, and three other signals which function can be configured. The digital interface of the ADXL355 when operating in *SPI mode* is shown in figure 3.5. Signals *INT1*, *INT2*, and *DRDY* can be used to trigger interrupts in the host processor, such as buffer overrun, buffer full, and data (sample) ready. Additionally, the sensor can be configured to use *INT2* and *DRDY* for clock synchronization with the host processor, there are two possible options for clock synchronization: interpolation and full synchronization. In the synchronization by interpolation an external clock with frequency f_s (1 kHz) is provided by the host to the *DRDY* pin. In this operation mode the acquisition is still driven by the internal clock of the sensor, however the resulting samples are the output of an interpolation filter

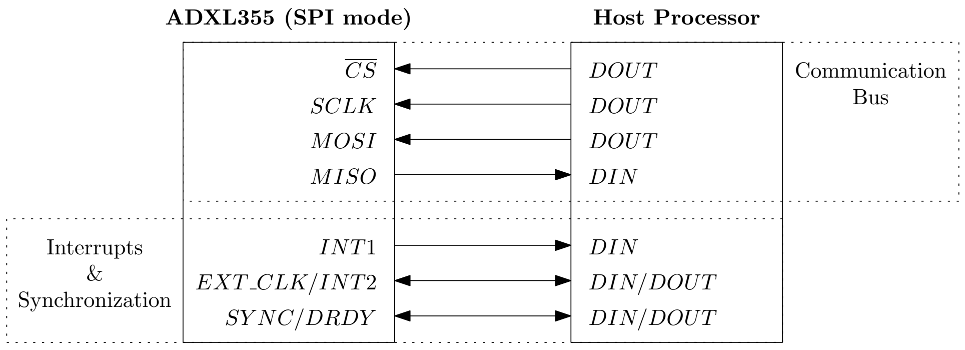


Figure 3.5: Digital interface of the ADXL355.

which estimates the acceleration values at the rising edges of the external clock. The interpolation filter has a time resolution of 64 times the sampling frequency. This approach is simple but introduces errors to the measurement including increased group delay, increased attenuation at the band edge, and distortion related to the mismatch between the internal and external clocks. On the other hand, full synchronization can be achieved by providing an external clock with a frequency of $4{\cdot}64{\cdot}f_s$ (256 kHz) to the *DRDY* pin that will be used as the master clock of the sensor. Additionally, the phase of the sampling cycle can be adjusted with the rising edge of a pulse connected to the *INT2* pin, so that the devices become fully synchronized.

Table 3.1: Key performance parameters of the ADXL355 accelerometer.

Parameter	Description	Min.	Nom.	Max.	Unit
f_s	Sampling frequency	3.9	-	4000	Hz
M	ADC resolution	-	20	-	$bits$
f_{SCLK_SPI}	Clock frequency of serial bus in SPI mode	0.1	-	10	MHz
f_{SCLK_I2C}	Clock frequency of serial bus in I2C mode	0.1	-	3.4	MHz
U_{supply}	Supply voltage	2.25	-	3.6	V
I_{meas}	Current consumption in measurement mode	-	200	-	μA
$I_{standby}$	Current consumption in standby mode	-	21	-	μA
t_{wakeup}	Standby to measurement mode transition time	-	-	10	ms
V	Volume, LCC package	-	6 x 6 x 2.1	-	mm^3
m	Mass	-	0.26	-	g

The ADXL355 has an internal voltage regulator which input ranges from 2.25 to 3.6 V. The manufacturer recommends to connect a pair of decoupling capacitors to each of the 4 different supply pins in the chip, which gives a total of 8 capacitors. The current consumption when the measurement is enabled is 200 μA while in standby mode it is 21 μA, the maximum time required to switch between the standby and measurement modes is 10 ms. Both the volume and mass of the sensor account for around 1 % of the total volume and mass budget.

The ADXL354 is a simpler version of the ADXL355 that includes the same sensing element but no analog-to-digital conversion, the frequency response and the signal-to-noise ratio (SNR) of the ADXL354 have been qualified for this application by the stakeholders.

3.2.2. Memory Selection

In the previous section it was determined that 1.8 Mbits of acceleration data must be transferred following an event such as a machine stop or exceeding an acceleration threshold. At least the same amount of data must be stored in memory in the case of a machine stop as the samples obtained before the time of the event must be transferred, this gives the minimum required capacity of the memory. However, considering that 60 kbits are produced every second some margin on the capacity is required to account for delays in the reception of the event notification (e.g. due to packet loss) and avoid sample loss.

Memory with parallel interfaces can provide much shorter access times compare to memory with serial interfaces, which in turn can lead to lower power consumption as the host processor can go back to sleep earlier if memory operations take less time. Package size is however larger for memory with a parallel interface as a higher pin count is required, because of the very limited space available only memory units with serial interfaces were considered. The SPI and I2C standards are widely used in serial memory, between them SPI is preferred for this application as it allows for higher communication speeds.

Table 3.2: Key performance parameters of SPI memory units based on different technologies.

Part Number	Type	f_{spi} [MHz]	Capacity $[Mbit]$	Endurance [cycles]	V_{supply} $[V]$	I_{active} $[mA]$	$I_{standby}$ $[\mu A]$	Volume $[mm^3]$
SST25VF080B	Flash	66	8	10^4	2.7 - 3.6	30	20	6 x 5 x 0.8
23LC1024	SRAM	20	1	N/A	2.5 - 5.5	10	20	6 x 5 x 1.75
CY15B104Q	FRAM	40	4	10^{14}	2 - 3.6	3	8	6 x 5 x 0.75

Table 3.2 presents the performance parameters of SPI memory units based on different technologies. Flash memory has the highest capacity but also the highest power consumption and requires longer active times as sectors must be erased before writing. Reliability is another concern as this type of memory has a relatively low endurance and only 10000 write cycles are guaranteed. Static RAM (SRAM) has no limitation on the number of write operations, it does not require erase cycles and consumes three times less current than the Flash memory, its capacity is however not sufficient for storing the amount of data required. Ferroelectric RAM (FRAM) has an even lower current consumption and it also does not require erase cycles, endurance is much higher than in the case of Flash memory allowing for 10 trillion write cycles and guaranteeing a minimum lifetime of 10 years. Its capacity is two times the required amount giving ample margin to account for delays in the reception of an event and thus avoid overwriting data that should be transferred to the sink. Because of these advantages, the CY15B104Q serial FRAM from Cypress Semiconductor was selected for the implementation of the sensor node.

3.2.3. Radio Technology Selection

The criteria for selecting a wireless standard on which the network would be based on was: the peak current consumption, the maximum data rate, and the support for multiple MCSs. In general, small batteries (e.g. coin cells) can only handle relatively low peak currents, the sensor nodes should therefore be based on a wireless standard that offers low current consumption as the radio usually contributes the most to the peak current. The data rate and the quality of the link determines greatly the total time required to retrieve the samples from all sensor nodes, faster modulation schemes require less time to transfer a packet but are more susceptible to environmental conditions, and vice versa. A wireless standard supporting multiple MCSs can potentially allow for overcoming adverse environmental conditions, e.g. if the Packet Error Rate (PER) exceeds a threshold the nodes can switch to a more robust (and slower) MCS to improve the link quality and the overall throughput.

The IEEE 802.15.4 standard is the de facto standard for WSNs, almost every network for experimental modal analysis included in section 2.4 is based on it. Different options for the physical layer are specified in this standard, most of which are based on the phase-shift keying (PSK) modulation and adopt spread spectrum techniques to reduce interference, such as direct sequence spread spectrum (DSSS) and chirp spread spectrum (CSS). However, commercial transceivers based on the IEEE 802.15.4 standard generally support only one of these MCSs. Moreover, the maximum data rate (250 kbps) is low for this application taking into account the relatively high volumes of data that must be transferred. For these reasons other alternatives were considered, from which WiFi and Bluetooth Low Energy stand out due to their performance and relevance in today's technology, emerging technologies for WSNs such as LoRa and Sigfox have been discarded due to their very low data rates. Table 3.3 includes key performance parameters for IEEE 802.15.4, WiFi, and BLE 5.

Table 3.3: Key performance parameters of most prominent wireless standards.

Standard	Max. Data Rate	Current Consumption	Supported MCS [1,2]
IEEE 802.15.4	250 kbps	~10 mA	1 (PSK)
WiFi	650 Mbps	>100 mA	8+ (PSK, QAM)
Bluetooth LE	2 Mbps	~10 mA	4 (FSK)

[1] Number of different modulation and coding schemes (MCSs) typically supported together in the same IC.
[2] Between parentheses, the fundamental modulation schemes supported.

WiFi (IEEE 802.11n/ac) supports much higher data rates than most unlicensed wireless standards, reaching several hundreds of megabit per second. Any radio based on this standard can operate under at least eight different MCSs, based either on PSK or quadrature amplitude modulation (QAM). Its high performance in terms of communication speed and versatility at the physical layer comes at the cost of a high current consumption with typical values in the order of hundreds of milliamperes. WiFi COTS components usually fall into one of three categories: full-stack (TCP/IP) WiFi modules, SoftMAC chipsets, and FullMAC chipsets.

Sensor nodes based on WiFi usually incorporate a *full-stack (TCP/IP) WiFi module* that communicates with a host microcontroller through a low-speed interface such as UART, I2C, or SPI. Their current consumption (several tens of milliamperes) is lower than other types of WiFi components but still higher than IEEE 802.15.4 radios. These WiFi modules are attractive in applications where devices based on resource-constrained MCUs must connect to the Internet as they provide a simple interface to the host processor, hiding the complexity of the TCP/IP protocol stack. Access to the link or physical layers is however desired in this application as in general the accuracy of a synchronization algorithm depends greatly on precise timestamping at the lower layers.

The so-called *SoftMAC* WiFi chipsets include up to the real-time link layer functionality (e.g. channel access) leaving the implementation of non-time-critical functionality to the host processor. These chipsets provide the highest data rates compared to other types of WiFi components, they also have the highest current consumption reaching several hundreds of milliamperes. SoftMAC chipsets are widely used in laptop and desktop computers, and operate over high-speed interfaces (e.g. PCI express) which are generally not available as a peripheral in MCUs. In contrast, *FullMAC* chipsets implement all the functionality of the link layer. Data rates are in the order of tens of megabit per second with typical current consumption above 100 mA. FullMAC devices are commonly found in mobile devices such as smartphones and communicate with the host processor over an SDIO interface, which is available in some low-power MCUs. However, device drivers are only available for general-purpose operating systems such as GNU/Linux, which cannot run on small MCUs like the ones found in typical wireless sensor nodes. In some cases the source code of such drivers is available but porting them (fully or partially) to a small MCU requires a major effort as no documentation such as a *programmer's manual* is provided by the manufacturers.

Version 5 of the Bluetooth Low Energy (BLE) standard offers four different MCSs all based on Gaussian frequency-shift keying (GFSK) modulation, possible data rates are 1 Mbps and 2 Mbps. When operating at 1 Mbps *channel coding* can optionally be used to map a bit to either two or eight symbols to improve the sensitivity of the radio. Compared to IEEE 802.15.4, BLE 5 has a lower protocol overhead at the link layer; the ratio of header size to payload size is lower in the case of BLE 5, with a payload size of nearly double than IEEE 802.15.4. Radio current consumption is comparable to IEEE 802.15.4, making possible to operate using a small battery. Due to its support for multiple MCSs and its low current consumption the BLE 5 was selected as the radio technology for the network.

3.2.4. Wireless MCU Selection

Table 3.4 includes key parameters of wireless microcontrollers from different manufacturers supporting the BLE 5 standard. All of them are based on either the ARM Cortex-M3 or Cortex-M4 microprocessors; apart from including support for DSP instructions and (optionally) for floating-point arithmetic the Cortex-M4 is identical to the Cortex-M3. As sensor data will not be processed at the sensor node but only transferred to the sink node no computationally-intensive operations are needed, hence the Cortex-M4 does not represent a significant advantage compared to the Cortex-M3. Some of the wireless

MCUs in table 3.4 include a co-processor that controls the radio hardware and implements part of the protocol's link layer, e.g. channel access, packet formatting and error detection. These radio co-processors are based on the ARM Cortex-M0 core which consumes less power than other processors in the Cortex-M family. This dual-core architecture can potentially result in lower power consumption, compared to single-core architectures, as it allows the main CPU to sleep while a radio operation is being executed.

Table 3.4: Key parameters of BLE 5 wireless microcontrollers from different vendors.

Parameter	EFR32BG13	PSoC 63	nRF52810	QN908x	RSL10	**CC2640R2F**
Manufacturer	Silicon Labs	Cypress	Nordic	NXP	On Semi	TI
Main CPU	Cortex-M4	Cortex-M4	Cortex-M4	Cortex-M4	Cortex-M3	Cortex-M3
Coprocessor	No	Cortex-M0	No	No	LPDSP32	Cortex-M0
I_{active} [μA/MHz]	69	46	34.4	33.7	N/A	61
I_{sleep} [μA]	1.3	7	1.5	2.5	0.3	1.1
P_{TX} [dBm]^1	19	4	4	2	6	5
S [dBm]^2	-94.8	-95	-95	-95	-94	-97
I_{TX} [mA]^3	8.5	5.7	5.8	3.5	4.6	6.1
I_{RX} [mA]^2	9.5	6.7	6.1	3.5	3	6.1
COTS Modules^4	Yes	No	Yes	No	No	Yes

^{1} Maximum transmission power
^{2} Sensitivity and current consumption during reception at 1 Mbps
^{3} Current consumption during transmission at 0 dBm
^{4} Availability of commercial off-the-shelf RF modules

COTS RF modules include all or most of the external components required by a wireless MCU to operate, such as the impedance matching circuitry and the antenna, reducing the effort in the design of a wireless sensor node. The availability of transceiver modules based on the wireless MCUs included in table 3.4 was part of the selection criteria; the PSoC 63, QN908x, and RSL10 were discarded due to the lack of availability of COTS modules at the time of the selection. Among the remaining devices the CC2640R2F was preferred because of its dual-core architecture in which the radio coprocessor offers a relatively simple interface yet allowing for low-level control over the execution of radio operations. Several modules based on the CC2640R2F are available in the market, from which the SaBLE-x-R2 was chosen as it had the most compact form factor, its key performance parameters are listed in table 3.5.

3.2.5. Battery Selection

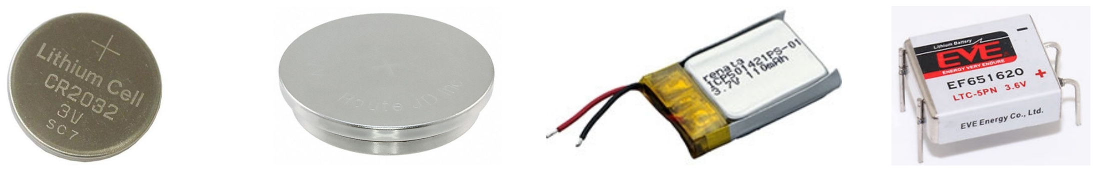
As mentioned in section 2.3 the minimum system lifetime is 48 hours and the average power dissipation is limited to 10 mW, this is thus the power budget for the sensor node. The less of this power budget is used the lower will be the required battery capacity, which greatly determines the battery size. On the other hand, the frequency of the events that trigger the data collection is limited by the maximum power consumption, to allow for the maximum possible number of events it was decided to use the totality of the power budget.

Considering a nominal battery voltage of 3.6 V and an average power consumption of 10 mW the maximum average current would be 2.77 mA, which gives a minimum battery capacity of 133.33 mAh for a lifetime of 48 hours. Table 3.6 includes the performance parameters of batteries that exceed this capacity and have a relatively small size, other important aspects that have been considered are the maximum current that can be handled by the battery and the initial battery voltage. The form factor of these batteries is shown in figure 3.6.

The CR2032 is a widely used non-rechargeable coin cell battery, it is the most compact and lightweight among the presented alternatives and the only one that meets the required dimensions (see section 2.3), however it can only handle relatively low currents. A rechargeable coin cell battery that can handle much higher current is the RJD2450, but its initial voltage exceeds the maximum operating voltage of the other components in the sensor node. Both the sensor and the wireless MCU

Table 3.5: Key performance parameters of the SaBLE-x-R2 (CC2640R2F) BLE 5.0 module

Parameter	Description	Min.	Nom.	Max.	Unit
U_{supply}	Supply voltage	1.8	-	3.8	V
f_{clk}	Clock frequency	-	-	48	MHz
P_{TX}	Transmission power	-21	-	5	dBm
I_{sleep}	Current consumption during deep sleep	-	1.1	3	μA
I_{CPU}	Current consumption main CPU	-	2.9	-	mA
I_{SPI}	Current consumption SPI peripheral	-	0.113	-	mA
I_{RFC}	Current consumption RF core	-	0.237	-	mA
I_{TX1}	Current consumption during TX at 0 dBm	-	6.1	-	mA
I_{TX2}	Current consumption during TX at 5 dBm	-	9.1	-	mA
I_{RX}	Current consumption during RX at 1 Mbps	-	6.1	-	mA
V	Volume	-	17.9 x 11.6 x 2.3	-	mm^3
m	Mass	-	0.75	-	g



(a) CR2032 (b) RJD2450 (c) ICP582930P (d) **LTC-5PN**

Figure 3.6: Form factor of considered batteries (not to scale)

include an internal voltage regulator so that these devices can be supplied directly by a battery. Using the RJD2450 would require a voltage regulator just to bring down the battery voltage, this comes with additional current consumption and higher board area, the same is true for the ICP582930PR rechargeable battery. The selected battery for the sensor node is the LTC-5PN which offers the highest capacity compared to the other options and can handle a current greater than the sum of the current consumption of the devices in the sensor node. An additional advantage of this battery is the high stability of its supply voltage which only drops significantly when the battery is almost depleted.

Table 3.6: Key performance parameters of considered batteries.

Part num.	Type	Capacity [mAh]	V_{init} [V]	V_{nom} [V]	I_{pulse} [mA]	I_{cont} [mA]	Volume [mm^3]	Mass [g]
CR2032	LiMnO2	210	3.4	3.0	10	0.2	Ø20 x 3.2	3.2
RJD2450	Li-Ion	190	4.2	3.7	400	40	Ø24.5 x 5.4	6.5
ICP582930PR	LiPo	430	4.2	3.7	900	450	32 x 29.5x 6.2	9.1
LTC-5PN	Li-SOCl2	550	3.6	3.6	20	10	24 x 17 x 7	8

3.2.6. Board Design

Figure 3.7 shows the final architecture of the sensor node board, the wireless MCU communicates with the accelerometer and the memory through an SPI bus. Connectors for programming the microcontroller and accessing the SPI bus and GPIO signals were also included in the board for convenience when testing and debugging the firmware.

As the selected battery is generally used as a backup battery throughout the lifetime of a system it is meant to be soldered directly to a PCB, for this reason there are no COTS holders available that allow

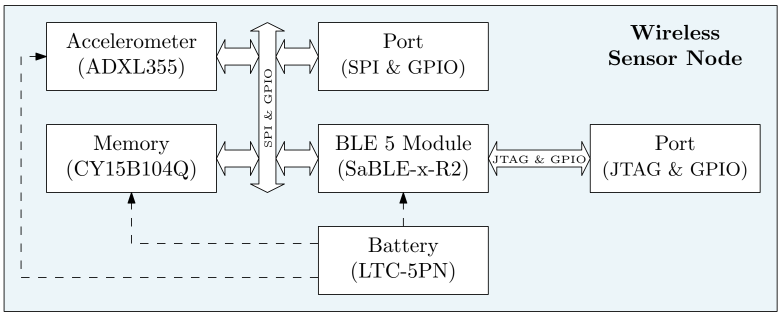


Figure 3.7: Block diagram of sensor node.

to easily replace the battery after depleted. As in this application the battery will be often replaced it is inconvenient to desolder the battery from the PCB every time it is depleted, thus it was decided to design a second PCB that acts as the battery holder and connects to the sensor board through a standard 2.54 mm connector. Mounting holes were also included in both boards to allow for a stable mechanical connection between them and the housing. To allow for installing the accelerometer close to the target surface the sensor node PCB, the battery, and the battery holder PCB have been stacked up as shown in figure 3.8. The schematics of the sensor node and the battery holder are included in appendix A.

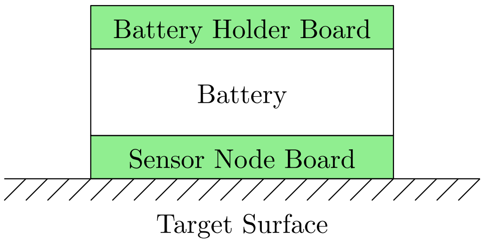


Figure 3.8: Stack comprised of the sensor node board, the battery and the battery holder board, and its orientation with respect to the target surface.

The board dimensions were determined by the area of the battery plus the required area for the mounting holes. The battery holder board and the sensor node board have dimensions of 27 x 24 mm, the PCB layout of these boards is shown in figure 3.9. The yellow areas in the sensor node board represent sections where connectors and other components that are not required for the sensor node's operation were placed. If a smaller battery is used in the future and the size of the battery holder board is reduced, these yellow areas can be freed and the BLE module's orientation and position can be modified to reduce the total board area.

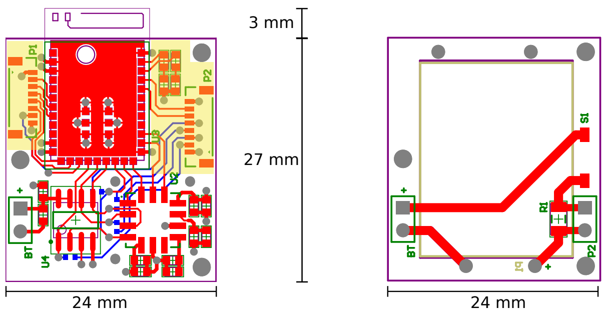


Figure 3.9: PCB layout of sensor node board (left) and the battery holder board (right); yellow area can be freed to reduce the total area of the sensor node board.

3.3. Firmware Design

An implementation of the BLE 5 communication stack for the CC2640R2F is provided by Texas Instruments, a real-time operating system (RTOS) is also provided on top of which the communication stack must run. Using this protocol stack can potentially reduce the effort in the development of the firmware for the sensor node and the sink, microsecond-level network synchronization is however not supported neither by the BLE 5 standard nor the communication stack implementation. A solution might be modifying the protocol stack to add precise network synchronization, but this is not possible as the source code was not made publicly available by the manufacturer. Because of this limitation, it was decided to implement a simple communication protocol to asses the feasibility of low power and reliable wireless synchronization inside the WH based on BLE 5 radios. The software for the sensor node and the sink was developed from the ground up and does not rely on an RTOS, and only makes use of Texas Instruments *DriverLib* library for register access.

Figures 3.10 and 3.11 show respectively the software architecture that was defined for the sensor nodes and the sink, the software was divided into multiple functional blocks that will be described in the remainder of this section. In the diagrams, the greyed-out blocks were not implemented and blocks with a tilling pattern were partially implemented to evaluate critical performance parameters that determine the fulfillment of the application's requirements, other blocks were entirely developed.

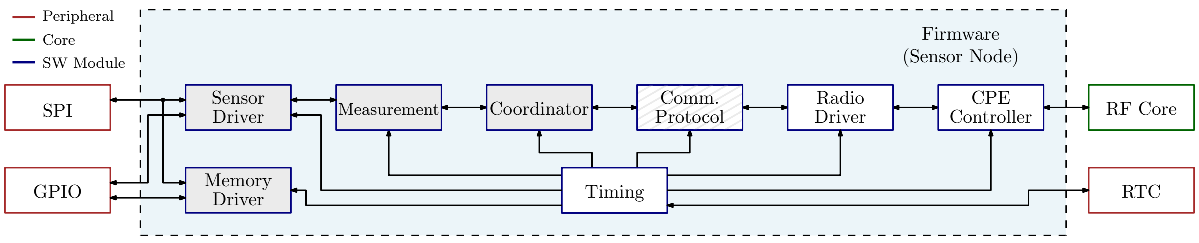


Figure 3.10: Sensor node firmware components.

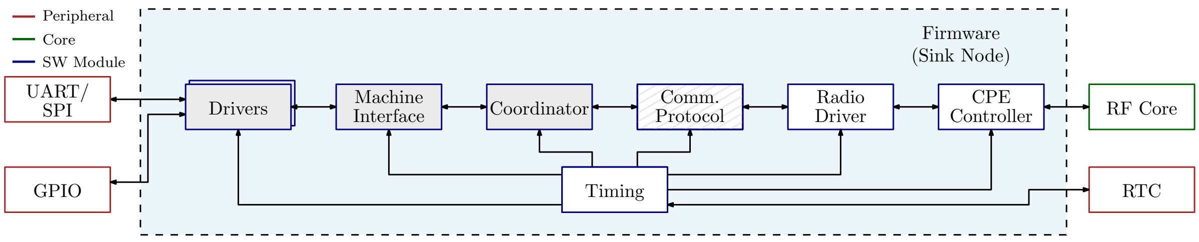


Figure 3.11: Sink node firmware components

3.3.1. Timing Module

The *Timing Module* provides a set of software timers based on the real-time clock (RTC) to allow other modules to program time events such as timeouts and periods. The number of timers is static as it is determined at compilation time. The RTC is driven by a 32768 Hz crystal oscillator and the Timing Module updates the software timers every 33 RTC cycles or 1.007 ms. These general-purpose timers are used for time events that require a coarse accuracy, e.g. a timeout in the CPE Controller to detect if the RF core became unresponsive.

Additionally, the Timing Module allows to synchronize other clock domains with the RTC by blocking the program execution until the start of the next RTC cycle. The RTC is also used to wake up the MCU from the sleep state, referred to as the *standby mode* in the manufacturer's documentation, as it is the only timer available in this low power mode. The Timing Module allows reading the current value of the RTC to schedule the time in which the MCU must wake up. The rate of the RTC can be adjusted to compensate for the tolerances in the crystal oscillator and reduce the clock offset accumulated over a sleep period, this is described in detail in section 3.4. This module is identical for both the sensor node and the sink.

3.3.2. Command and Packet Engine (CPE) Controller

This is another module that is identical for the sensor nodes and the sink. The *Command and Packet Engine (CPE) Controller* is named after the hardware interface that allows the communication between the main processor and the RF core, this interface is also called the *Radio Doorbell* in the documentation of the manufacturer. Figure 3.12 shows the main components in the wireless MCU involved in the communication between the RF core and the main CPU; which can both access the system RAM and the CPE. An application programming interface (API) defines a set of instructions, and its parameters, that can be executed by the RF core. The CPE consists of a set of registers that allow to submit an instruction to the RF core and to check the status of its execution. The execution of an instruction is triggered by writing to the CPE either its *opcode* directly, for simple instructions with a few or no parameters, or a pointer to a data structure stored in system RAM containing the opcode and the instruction's parameters. The CPE Controller software module provides a simple interface to other modules for the execution of RF core instructions.

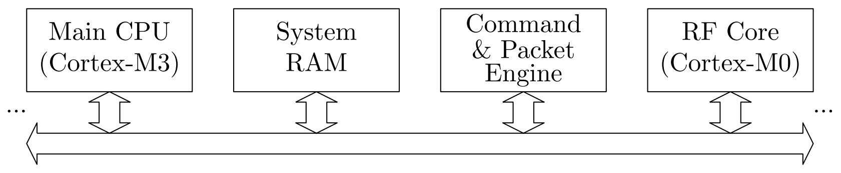


Figure 3.12: Simplified architecture of the CC2640R2F, including the main components involved in radio operations

3.3.3. Radio Driver

The *Radio Driver* module simplifies the operation of the radio hardware by offering to other modules a set of radio operation primitives, such as `Set_Tx_Power` and `Receive_Packet`. It makes use of the *CPE Controller* to send instructions to the RF core, table 3.7 lists the radio driver operations and the associated RF core instructions. The RF core supports several tens of instructions, in this implementation only six of them have been used.

Table 3.7: Radio driver operations and the associated RF core instructions

Radio Operation	RF Core Instructions	Description
Turn radio on	CMD_BLE5_RADIO_SETUP CMD_FS CMD_SYNC_START_RAT	RF core initialization Frequency synthesizer initialization Start radio timer (RAT) and synchronize with RTC
Turn radio off	CMD_SYNC_STOP_RAT	Stops RAT, stores value used for synchronization with RTC
Set data rate	CMD_BLE5_RADIO_SETUP CMD_BLE5_ADV_AUX CMD_BLE5_SCANNER	Modifies the parameters of these instructions
Set TX power	CMD_BLE5_RADIO_SETUP CMD_BLE5_ADV_AUX CMD_BLE5_SCANNER	Modifies the parameters of these instructions
Set freq. channel	CMD_BLE5_RADIO_SETUP CMD_BLE5_ADV_AUX CMD_BLE5_SCANNER	Modifies the parameters of these instructions
Transmit packet	CMD_BLE5_ADV_AUX	Transmits a BLE 5 AUX_ADV_IND packet
Receive packet	CMD_BLE5_SCANNER	Listens to the channel waiting for an AUX_ADV_IND packet

The RF core includes a timer called the radio timer (RAT) that can be used to schedule the execution of radio operations and timestamp the arrival of packets, this timer is driven at 4 MHz which gives a resolution of 250 ns. The RAT cannot run while the MCU is sleeping, however its counter can be updated with respect to the RTC after wake up so that it appears as being always enabled from the programmer's perspective. This synchronization between the RAT and the RTC is done by issuing the `CMD_SYNC_STOP_RAT` instruction before going to sleep and the `CMD_SYNC_START_RAT` instruction during the initialization of the radio after wake up. During radio initialization the `CMD_BLE5_RADIO_SETUP` and the `CMD_FS` instructions must also be executed, the former starts up the radio and the later initializes the frequency synthesizer.

To transmit a packet the Radio Driver executes a `CMD_BLE5_ADV_AUX` instruction, which handles the transmission of a BLE *advertising packet*. Reception is accomplished by starting a `CMD_BLE5_SCANNER` instruction which listens to the channel until a packet is received or until a timeout occurs. The parameters of transmission and reception operations are presented in tables 3.8 and 3.9 respectively. The start of a transmission or reception operation can be scheduled to an absolute time of the RAT, this is indicated by the `delayed_start` flag in the operation's parameters. The parameter data structure passed to the Radio Driver for receiving a packet also contains the `timestamp` and `rssi_dBm` fields, the former corresponds to the estimation of the start of the transmission of the packet and the later to the *received signal strength indication* (RSSI).

The data rate, transmission power, and frequency channel are configured by modifying the associated parameters in the data structures of the `CMD_BLE5_RADIO_SETUP`, `CMD_BLE5_ADV_AUX`, and `CMD_BLE5_SCANNER` instructions. The Radio Driver is also identical for the sensor nodes and the sink.

Table 3.8: Parameters of a packet transmission operation.

Parameter	Type	Description
delayed_start	bool	If 0 start immediately, else start at *start_time*
start_time	uint32_t	Start time of operation, RAT absolute time
payload_len	uint8_t	Length of packet payload, 0 to 254
payload_p	uint8_t*	Pointer to payload buffer

Table 3.9: Parameters of a packet reception operation.

Parameter	Type	Description
delayed_start	bool	If 0 start immediately, else start at *start_time*
start_time	uint32_t	Start time of operation, RAT absolute time
timeout	uint32_t	Reception timeout, RAT absolute time
dest_buf_len	uint8_t	Capacity of payload buffer
dest_buf	uint8_t*	Payload buffer
payload_len	uint8_t	Length of packet payload, 0 to 254
timestamp	uint32_t	Reception timestamp
rssi_dBm	int8_t	RSSI [dBm]
error	uint8_t	Error code, if any ocurred

3.3.4. Communication Protocol Module and Coordinator

The Communication Protocol module was partially developed only to demonstrate the critical functionality of the wireless communication between the sensor nodes and the sink. A set of independent tests was implemented to evaluate the accuracy of the network synchronization, the maximum achievable throughput, and the link quality inside the machine for different radio configurations. The implementation of these tests is different for the sensor nodes and the sink as specific behavior is required depending on the role, the tests are described in chapter 4. In a complete implementation this module must handle all the aspects of the communication between the sensor nodes and the sink.

The *Coordinator* module purpose is to integrate the functionality of the Communication Protocol module and the *Measurement module* to implement the behavior of the sink and sensor nodes specified by the functional requirements of the application. For example, if a sensor node receives a request from the sink to start a measurement, the Communication module passes this message to the Coordinator module which in turn enables the memory and sensor through the Measurement module. Additionally, it should handle the power management of the sink and sensor nodes, going to sleep whenever all other software modules are idle. As this work only considers aspects of the firmware related the wireless communication, only the functionality of the Coordinator related to power management has been implemented in the Communication module as part of the tests mentioned before.

3.3.5. Other Modules

Other modules defined in the firmware architecture are beyond the scope of this work. In the case of the sink node the remaining software modules handle the communication with the machine for clock synchronization and for reporting the occurrence of an event and transferring the measurements associated with such event. For the sensor nodes, the *Measurement* module coordinates the acquisition of the acceleration data and its storage in the external memory, this is done through the *Sensor Driver* and the *Memory Driver* modules.

3.4. Communication Protocol

Wireless communication between the sink and the sensor nodes should allow for clock synchronization, event notification, and data retrieval. While there are no hard requirements on the delay in the retrieval of the acceleration samples, clock synchronization and timely event notification are critical for the reliability of the system. As stated in section 2.3, acceleration data should be sampled synchronously at all sensor nodes with a maximum error of 55 μs, meaning that the difference in the sampling time at any two nodes in the network should not be greater than this value. If this requirement is not met then the obtained data has no value as the error in the results of modal analysis would be unacceptable, therefore the communication protocol should ensure with high confidence that the network will remain synchronized during the measurement. Timely event notification is also critical as excessive delay in the arrival of an event would result in data loss, either because a node has not received the message indicating the start of a measurement when a data retrieval event occurs or because the excessive delay in the arrival of a data retrieval event causes the overwriting of samples stored in the external memory.

This section describes the proposed communication protocol for network synchronization and event notification. The protocol does not define how to perform the data retrieval, however it was designed to allow coexistence with a file transfer protocol that could be added in future work.

3.4.1. Overview

Figure 3.13 presents the different states of the system, which were derived from the functional requirements included in section 2.3, these are:

- *initialization*: the system performs a set of procedures required to start its normal operation,
- *idle*: after initialization or whenever a measurement or data collection is finished or canceled, the system waits for the user's indication of the start of a measurement,
- *measurement*: after the user has indicated the start of a measurement the system synchronously samples the acceleration at all measurement locations, and
- *data collection*: either a machine stop has occurred or an acceleration threshold was exceeded, the samples associated with such event have been stored in memory, and the sink collects this data from all sensor nodes.

Figure 3.14 presents the general idea of the communication protocol used during the initialization, idle, and measurement states of the system. Due to the requirement for network-wide synchronization, time-division multiple access (TDMA) comes as a natural choice for the channel access mechanism used by the wireless network. In all states the sink transmits periodically a *beacon* packet that is used for network control and clock synchronization. A *time frame* is defined as the time interval between two consecutive beacons, each time frame is divided into eleven *time slots*. At compile time a sensor node is given a unique id from 1 to 10 which represents the time slot that is allocated to it, time slot 0 is used by the sink to transmit the beacon. Sensor nodes are only allowed to transmit a packet to the sink at

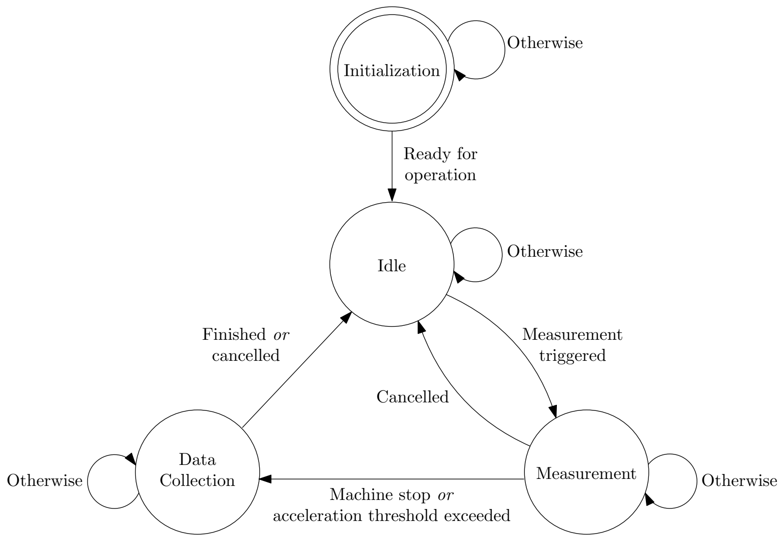


Figure 3.13: State machine of the measurement system.

the start of their corresponding time slot and only if it has been indicated in the beacon packet; with the exception of the measurement state in which they are allowed to transmit a packet if an acceleration event occurred regardless of the indication in the beacon. The sink can specify in the beacon if all sensor nodes, none of them, or one in particular should transmit a packet in the current time frame. In this way the sink can verify the presence of the sensor nodes at any time, e.g. at initialization to confirm if all sensor nodes are ready for operation.

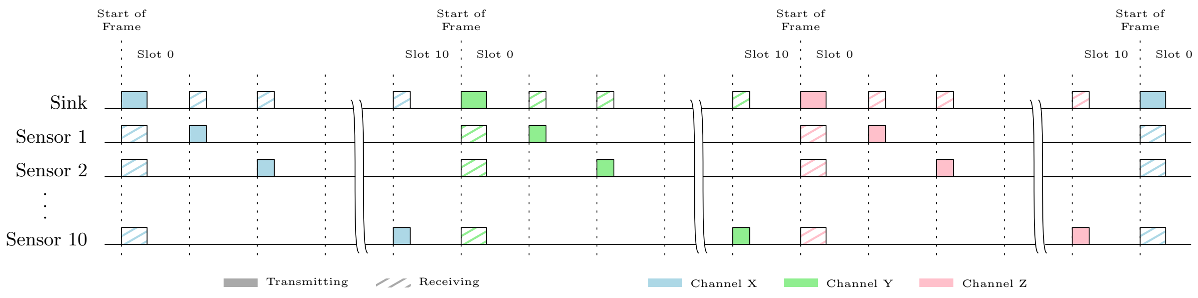


Figure 3.14: Beacons for network control and clock synchronization are transmitted at the start of every frame, sensor nodes reply to the sink at the start of their respective time slot if indicated in the beacon.

As the network will be located inside the WH (i.e. a metallic enclosure) it is reasonable to assume that there will be low or no interference from external wireless networks, for the same reason multipath fading is expected to be a major limiting factor on the link quality. According to Watteyne et al. [28] multipath fading can be mitigated by either shifting the channel frequency or the length of the wireless link, experimental results show that, at 2.4 GHz and for short-range links, a shift in length of 5.5 cm or a shift in frequency of 15 MHz allows to transition from a deep fading region to a region with average signal strength. Frequency hopping has been adopted to improve the reliability of the communication in multiple wireless standards, such as BLE and IEEE 802.15.4e. In addition to multipath fading, electromagnetic interference (EMI) is expected in such environment due to the presence of electric actuators and power supplies. The energy of this interference has not been measured inside the machine at the 2.4 GHz band, in this work it was assumed that EMI at this frequency band is either negligible or sporadic. To account for multipath fading and EMI, a different frequency channel can be used at each time frame according to a predefined sequence.

In the remainder of this section the clock synchronization protocol will be described along with the specifics of the communication at each of the states of the system. The section closes with the details about the implementation and a comparison with the BLE protocol.

3.4.2. Network Synchronization

As mentioned before, network synchronization is crucial for the reliability of the system as an excessive offset in the sampling times would produce an unacceptable error in modal analysis. Sensor nodes should therefore remain synchronized within specification limits despite sporadic communication loss, which is expected in a harsh environment such as the WH. Because all devices in the network operate in vacuum they are equally constrained on the power dissipation, the synchronization protocol should use low power and distribute the load evenly among the nodes in the network. As the sink is responsible for collecting the samples from all sensor nodes after an event occurs, radio utilization is much higher for the sink compared to a sensor node. The network synchronization protocol should therefore aim for fairness with regards to power consumption so that it does not contribute significantly to this asymmetry in the radio utilization.

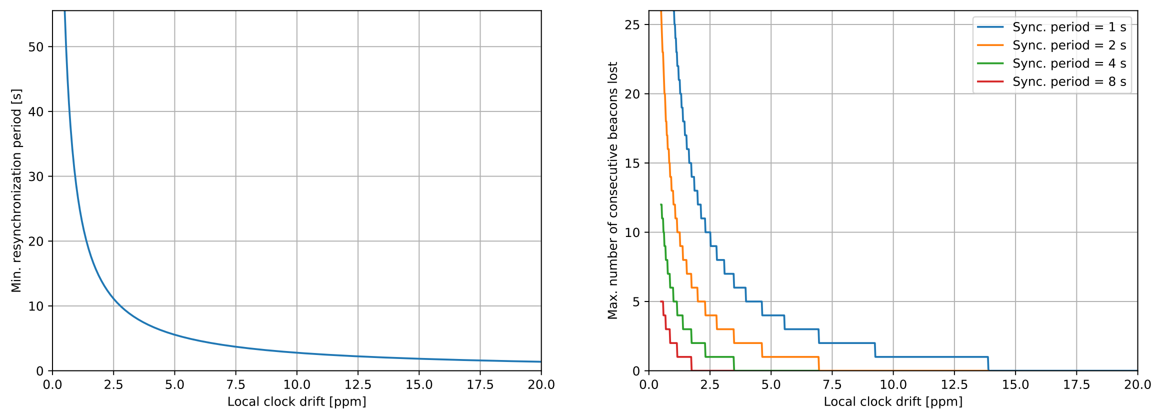
The BLE module (SaBLE-x-R2) in the sensor nodes includes two oscillators, a 48 MHz crystal driving the MCU clock and a 32768 Hz crystal driving the RTC. Both of these crystal oscillators have a tolerance of 20 ppm, which means that in the worst case the local clocks of any two sensor nodes in the network would drift away from each other at a rate of 40 $\mu s/s$, i.e. when a sensor node has tolerance of +20 ppm and the other one has a tolerance of −20 ppm. If this drift is not compensated, then the local clocks of all sensor nodes should be synchronized periodically with a maximum period of 1.39 seconds (equation 3.1) to guarantee that they will remain synchronized as specified.

$$T_{sync} = \frac{Accuracy}{2 * Tolerance} = \frac{55.55\ \mu s}{2 * 20\ ppm} = 1.389\ seconds \tag{3.1}$$

In the wireless network, the sink acts as the master clock as it is directly connected to the machine and has access to a precise time source. To synchronize all nodes at once the sink transmits periodically a beacon packet. By doing so every sensor node can estimate its local clock offset, relative to the clock of the sink, as the difference between the reception timestamp of the beacon and the expected reception time (equation 3.2). The same principle is used by other protocols such as IEEE 802.15.4 and BLE for synchronizing two or more devices with a single packet.

$$Clock\ offset = Reception\ timestamp - Expected\ arrival\ time \tag{3.2}$$

A sensor node will remain synchronized unless beacons are lost consecutively for a time interval longer than the maximum re-synchronization period. A straightforward approach to improve the reliability of the system is to increase the rate at which the beacons are transmitted to reduce the likelihood of missing too many packets in a row. This potential solution however would increase the power consumption and would be effective only if the interference causing the communication errors does not persist for longer than the maximum re-synchronization period. As this period is only ∼1.4 seconds it is not unrealistic to consider that sporadic interference (e.g. from an electric machine) could prevent the reception of a beacon for longer than this short time interval.



(a) Minimum resynchronization period (b) Maximum consecutive beacon loss

Figure 3.15: Resynchronization period and consecutive beacon loss depending on worst-case local clock drift, considering an accuracy of 55 μs.

A more robust approach is to increase the maximum re-synchronization period, allowing for lower power consumption and higher reliability. As it can be inferred from equation 3.2, the maximum re-synchronization period can be increased by reducing the drift (tolerance) of the sensor nodes' local clock. The drift can be estimated as the clock offset divided by the time elapsed since the last received beacon (equation 3.3), assuming a low variance in the drift during this interval. Based on this measurement the drift of the local clock can be compensated either in software or, if supported, in hardware. Figure 3.15 shows the minimum resynchronization period and the maximum consecutive beacon loss depending on the worst-case local clock drift, considering an accuracy of 55 μs. In figure 3.15b it can be observed that for a drift of 20 ppm no beacon loss is tolerated, while if the worst-case drift is reduced to 2 ppm more than ten consecutive beacons can be lost if the synchronization period is one second.

$$Clock\ drift = \frac{Clock\ offset}{Time\ since\ last\ beacon} \tag{3.3}$$

In IEEE 802.15.4e networks, clock offset and drift estimation are also based on the expected and actual arrival times of control packets. Elsts et al. [29] showed that time synchronization errors in implementations of the IEEE 802.15.4e standard are *"caused primarily by imprecise clock drift estimations, rather than by real unpredictable drift variance"*. According to Elsts et al., the error in the reception timestamp due to quantization is the main source of error in the estimation of the drift. Timing errors in packet transmission do not contribute significantly as in general this operation can be scheduled precisely on the edge of a clock. Elsts et al. showed that the overall measurement error can be reduced by averaging multiple samples, and noted that this error is inversely proportional to the re-synchronization period.

As mentioned in section 3.3.2, the CC2640R2F includes a timer that can be used for packet reception timestamping and packet transmission scheduling. The radio timer (RAT) has a resolution of 250 nanoseconds and thus can be used for precise clock synchronization. However, this timer cannot run in the lowest power mode of the MCU, and the RTC should be used instead during this state to wake up the MCU. During radio startup the radio timer (RAT) can be synchronized with the RTC so that it appears as if it were continuously operating, this feature is used in Elsts et al. implementation [29]. The drift of the RTC can be estimated based on the reception timestamp of the beacon as explained before, and it can be compensated in hardware by modifying the rate at which the counter of the RTC is increased. The CC2640R2F allows to compensate for the drift of the RTC with a resolution of 0.119 ppm.

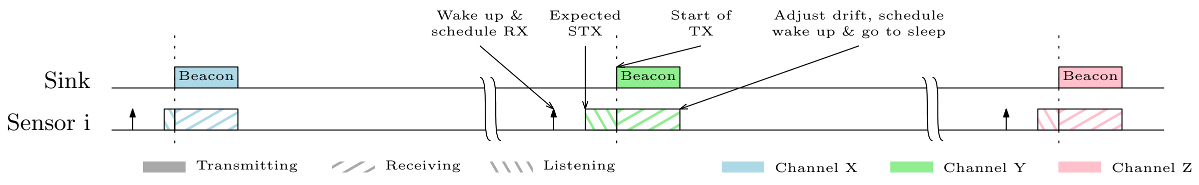


Figure 3.16: Clock synchronization and duty-cycled operation.

Figure 3.16 shows the different actions performed for clock synchronization in duty-cycled operation. A sensor node wakes up before the start of a time frame and schedules the start of the reception. After the beacon is received the sensor node estimates its local clock drift and adjusts the increment rate of the RTC accordingly. The sensor node can go back to sleep after having scheduled the wake-up time to the start of its respective time slot if a reply was requested, or to the start of the next time frame if otherwise.

3.4.3. Initialization and Idle States

During the initialization state, the sink will periodically transmit a beacon as in normal operation but in addition it will transmit auxiliary beacons within the time frame, sweeping across the predefined frequency channel sequence in a circular fashion. These auxiliary packets contain the relative time to the start of the next time frame, i.e. the start of the transmission of the next beacon, and the frequency channel that will be used. After all sensor nodes have replied in their respective time slots the sink stops sending the auxiliary beacons and goes to the idle state. The purpose of the auxiliary beacons is to reduce the time spent by the sensor nodes in the initialization state, as in this state the sensor nodes listen to the channel for long periods as explained below.

A sensor node will be in the initialization state after a power-on or after it has lost synchronization with the sink. In this state the sensor node will listen to a single channel for the duration of a time frame plus the reception time of a beacon. If a beacon is not received the sensor node will sleep for the duration of a time frame, after which the process repeats using a different channel from the predefined sequence. After receiving a beacon for the first time, the sensor node schedules the next wake-up time for the start of the following time frame, when the drift will be compensated for the first time. At this point the node is in the idle state and will reply to the sink if this is indicated in the beacon packet. To reduce power consumption the sink node should be operating before the sensor nodes are powered on, as otherwise these would be listening worthlessly for relatively long periods.

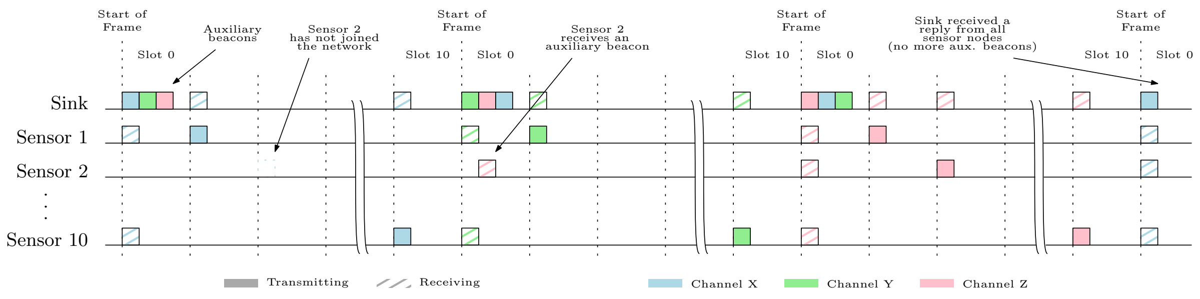


Figure 3.17: Auxiliary beacons are transmitted as sensor 2 has not joined the network, after this node replies the sink stops transmitting the auxiliary beacons as all sensor nodes have now joined the network.

Figure 3.17 shows the transmission of auxiliary packets as not all sensor nodes have replied to the sink, in this case sensor node 2 has not been yet powered up in the first time frame and thus its reply is missing in this frame. By the start of the next time frame, sensor node 2 is ready and as it is initially listening to the ”pink” channel it receives one of the auxiliary beacons. In the third time frame sensor node 2 receives the beacon and replies to it, as the sink has now verified the presence of all sensor nodes it stops transmitting auxiliary packets.

In the idle state the system is waiting for the user to start a new measurement. After the user requests the sink to start the measurement, the new state will be indicated in the coming beacons which will also request the sensor node to reply in their corresponding time slot to verify that all sensor nodes are now in the measurement state. In the idle state the sink node also assesses the liveliness of the sensor nodes by periodically requesting them to reply to a beacon, in this way the system can detect if communication was lost with a sensor node and report it to the user.

3.4.4. Measurement and Data Collection States

After having confirmed the presence of all sensor nodes the sink notifies the user that the system is ready to start the operation, at this point the system is in the idle state and the user can then start a measurement. When this occurs the beacons indicate the new state and the sink will request replies from all sensor nodes until it has confirmed that all of them are in the measurement state. In this state the sink will always listen to the sensor nodes in their respective time slots as these could report at any time an acceleration threshold event. The sink can also receive a machine stop event from the user at any moment.

When the sink receives an event the sensor nodes are notified through the beacon packet, which will contain the elapsed time since the event occurred. The sink will include this field in the beacon packet until the replies from all sensor nodes are received. Using the elapsed time the sensor nodes can find in memory the samples associated with the event. In the case of a machine stop event the sensor nodes will stop the acquisition immediately after receiving a beacon indicating the event, if the event corresponds to an acceleration threshold the nodes will continue sampling until 30 seconds have elapsed since the event occurred. Figures 3.18 and 3.19 show respectively an example scenario of an acceleration threshold event and a machine stop event.

In figure 3.18 sensor 2 transmits a packet during its respective time slot reporting an acceleration threshold event (indicated with a dot), this packet is received by the sink which in turn includes the event in the next beacon. As requested by the sink the sensor nodes reply to the beacon to acknowledge

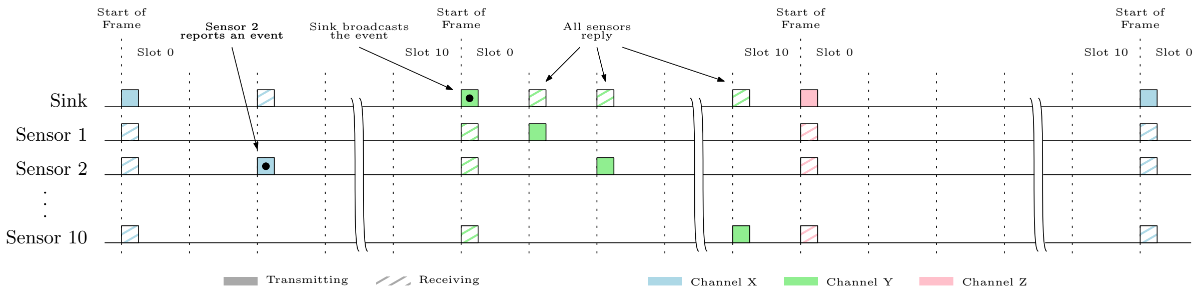


Figure 3.18: Sensor 2 reports to the sink an acceleration threshold event, the sink forwards this event to all other sensor nodes in the next beacon.

the reception of the event. In the next time frame the sink does not include the event in the beacon as the reply from all sensor nodes has already been received. Because the event was an acceleration threshold the sensor nodes will continue sampling until 30 seconds have elapsed since the occurrence of the event, after which data collection will take place.

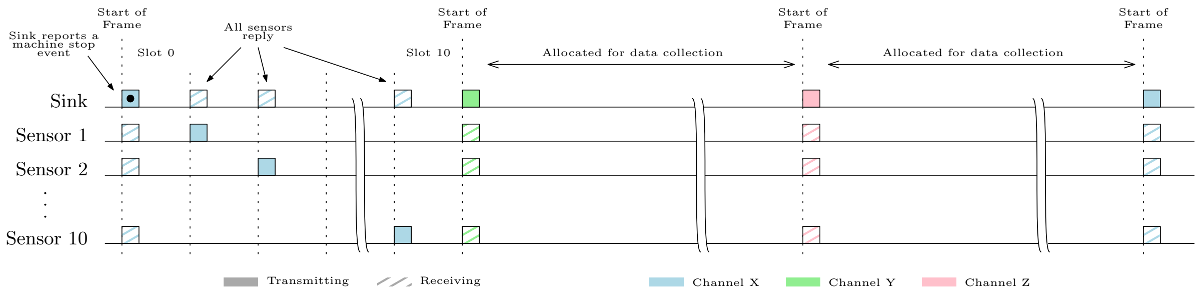


Figure 3.19: The sink notifies to the sensor nodes of a machine stop event in the beacon, as the reply from all sensor nodes is received during the same time frame data collection can start in the next one.

In figure 3.19 the sink node notifies the sensor nodes of a machine stop event. In this case, all sensor nodes receive the beacon and reply to it during their respective time slots. The sink receives the replies from all sensor nodes and does not include the event in the next beacon. As this was a machine stop event all samples associated with the event are already in memory, and hence data collection can start after all sensor nodes have acknowledged the reception of the event.

Data collection can start when all samples associated with an event are present in memory. During data collection the beacons are still transmitted periodically, these packets are used to indicate which sensor node should remain active during the rest of the time frame for transferring the acceleration data to the sink. In this work, the protocol used for data transfer was not defined and was left for future work.

3.5. Power Consumption Estimation

This section examines the contribution of individual components (i.e. sensor, memory, radio, MCU) to the power consumption in the measurement and data collection states for the sink and the sensor nodes, taking into account the influence of the system's operation parameters such as the synchronization period, the data rate, and the transmission power on the power consumption. Additionally, this section gives an estimate of the upper bound on the rate of data collection events to not exceed the maximum power dissipation. For estimating the power consumption several assumptions have been made for considering a worst-case scenario and simplifying the analysis. These assumptions are:

- During the measurement state the sensor nodes always reply to the sink. This is not necessary as if there are no events to report by the sensor nodes, the transmission of the reply can be skipped to save power at the sensor nodes and the sink (lower idle listening).
- The payload size for the beacons and replies was assumed to be maximum (254 bytes), in practice a few bytes suffice for control information.

- To calculate the maximum rate of data collection events, it was assumed that during the measurement state the most robust (and also the most power consuming) radio configuration is used, i.e. the highest transmissions power and the lowest data rate with the lowest channel code rate. In practice, a less reliable radio configuration could also provide the required reliability consuming less power. It was also assumed that the network synchronizes every second, depending on the link quality and the accuracy of the drift compensation a lower synchronization rate could be used to lower power consumption.
- An overhead of 50% on the wireless communication was considered for estimating the power consumption of data collection, i.e. data collection takes 50% longer. This overhead is meant to account for communication errors and protocol overhead, the actual value will depend on the link quality and how efficient the data transfer protocol is.
- It is assumed that the MCU is always active whenever an operation is being executed by the sensor, the memory or the radio. In practice, the MCU can enter a lower power mode while waiting for an event from one of these devices, e.g. the radio indicates the end of a transmission.
- It is assumed that the SPI bus is driven at a frequency of 2 MHz. A higher frequency can be used to reduce the access times of the memory and sensor, and reduce the power consumption of the MCU.

3.5.1. Power Consumption in the Measurement State

As mentioned in section 3.4, in this state the sink transmits periodically a beacon packet for synchronizing all sensor nodes at once, the sensor nodes reply in their corresponding time slot if it is indicated in the beacon packet. For simplifying the analysis and give a worst-case estimation it is assumed that the sensor nodes always reply to the beacon. In this state the sensor nodes are constantly sampling the acceleration and storing the data in memory. As these components are only present in the sensor nodes, only the radio and the MCU contribute to the power consumption of the sink. In this section the power consumption of the sink and the sensor nodes during the measurement state is estimated, this gives a baseline value to which the power consumption of asynchronous data collection events adds up.

Power consumption of the sensor nodes

The accelerometer is always active during the measurement state, the power consumption of this component is about 0.5 mW (equation 3.4). To estimate the power consumption of the memory it is assumed that the samples are read from the sensor and stored in memory one by one. A single sample consists of 9 bytes (3 bytes per channel), and a write access requires 4 bytes to indicate the opcode and the address. The power consumption of the memory depends on the access times, which are determined by the clock frequency of the SPI bus. Considering a sampling frequency of 1 kHz and a SPI clock frequency of 2 MHz, the power consumption of the memory is 67.12 μW.

$$\boxed{P_{sen} = V * I_{sen\_act} = 3.3V * 150\mu A = 495\mu W} \tag{3.4}$$

$$D_{mem} = \frac{f_s * (N_{header} + N_{sample})}{f_{SPI}} = \frac{1kHz * (32bits + 3 * 24bits)}{2MHz} = 0.052 \tag{3.5}$$

$$P_{mem} = V * [D_{mem} * I_{mem\_act} + (1 - D_{mem}) * I_{mem\_sb}] \tag{3.6}$$

$$\boxed{P_{mem} = 3.3V * (0.052 * 300\mu A + 0.948 * 5\mu A) = 67.12\mu W} \tag{3.7}$$

Within a synchronization period a sensor node receives a beacon and transmits a reply, the rest of the time the radio remains disabled. The radio takes about 1 millisecond to boot after being enabled. The packet reception and transmission times depend on the data rate used. While the startup current and the reception current are constant, the current consumed during packet transmission depends on the transmission power. In summary, in the measurement state the power consumption of the radio depends on the data rate, the transmission power, and the synchronization period (equations 3.8 and 3.9).

$$D_{rad\_boot} = \frac{2 * t_{boot}}{T_{sync}} = \frac{2ms}{T_{sync}}, \quad D_{rad\_tx} = \frac{t_{pkt\_tx}}{T_{sync}}, \quad D_{rad\_rx} = \frac{t_{pkt\_rx}}{T_{sync}} \tag{3.8}$$

$$\boxed{P_{rad} = V * (D_{rad\_boot} * I_{rfc} + D_{rad\_tx} * I_{tx} + D_{rad\_rx} * I_{rx})} \tag{3.9}$$

As mentioned before, it is assumed that the MCU remains active whenever the sensor, memory, or radio are active (equations 3.10). Therefore the utilization of the MCU also depends on the data rate, which gives the packet transmission and reception times, and the synchronization period. The ratio of time that the MCU spends reading the sensor is calculated in equation 3.11, in assuming that the frequency of the SPI bus is 2 MHz as in the case of the memory.

$$D_{rad} = D_{rad\_boot} + D_{rad\_tx} + D_{rad\_rx} \tag{3.10}$$

$$D_{sen} = \frac{f_s * (N_{header} + N_{sample})}{f_{SPI}} = \frac{1kHz * (8bits + 3 * 24bits)}{2MHz} = 0.04 \tag{3.11}$$

$$D_{mcu} = D_{sen} + D_{mem} + D_{rad} = 0.092 + D_{rad} \tag{3.12}$$

$$P_{mcu} = V * [D_{mcu} * I_{mcu\_act} + (1 - D_{mcu}) * I_{mcu\_sb}] \tag{3.13}$$

$$\boxed{P_{mcu} = 3V * [D_{mcu} * 3.27mA + (1 - D_{mcu}) * 3\mu A]} \tag{3.14}$$

The power consumption of a sensor node is the sum of the power consumption of the accelerometer, memory, radio, and MCU:

$$\boxed{P_{node} = P_{sen} + P_{mem} + P_{rad} + P_{mcu}} \tag{3.15}$$

Power consumption of the sink

On the other hand, only the radio and the MCU contribute to the power consumption of the sink (equation 3.17). During the measurement state, the sink transmits one beacon and receives ten replies, one per each sensor node; therefore the reception time of the sink is ten times higher than in the case of a sensor nodes, and the radio has to be powered up eleven times in total within a synchronization period (equation 3.16). Figure 3.20 shows the power consumption of the sink and the sensor nodes during the measurement state for the different data rates and for a transmission power of 0 dBm and 5 dBm, the synchronization period is 1 second. Figure 3.21 shows the power consumption of the sink and the sensor nodes during the measurement state depending on the data rate and synchronization period, the transmission power is 5 dBm.

$$D_{rad\_boot} = \frac{11 * t_{boot}}{T_{sync}} = \frac{2ms}{T_{sync}}, \quad D_{rad\_tx} = \frac{t_{pkt\_tx}}{T_{sync}}, \quad D_{rad\_rx} = \frac{10 * t_{pkt\_rx}}{T_{sync}} \tag{3.16}$$

$$\boxed{P_{sink} = P_{rad} + P_{mcu}} \tag{3.17}$$

3.5.2. Power Consumption in the Data Collection State

Whenever a machine stop or an acceleration threshold event occurs, the sink node retrieves from all sensor nodes the samples recorded 30 seconds before or after the event depending on its type. These events are asynchronous, however to simplify the analysis it is assumed that these events occur at a given rate. In this section the contribution that would have a single data collection event every hour to the average power consumption is estimated for the sink and the sensor nodes. Using this estimation, the maximum number of events per hour that can occur without exceeding the maximum power dissipation can be found based on the calculations of the previous section, which gives the base power consumption for the sink and the sensor nodes.

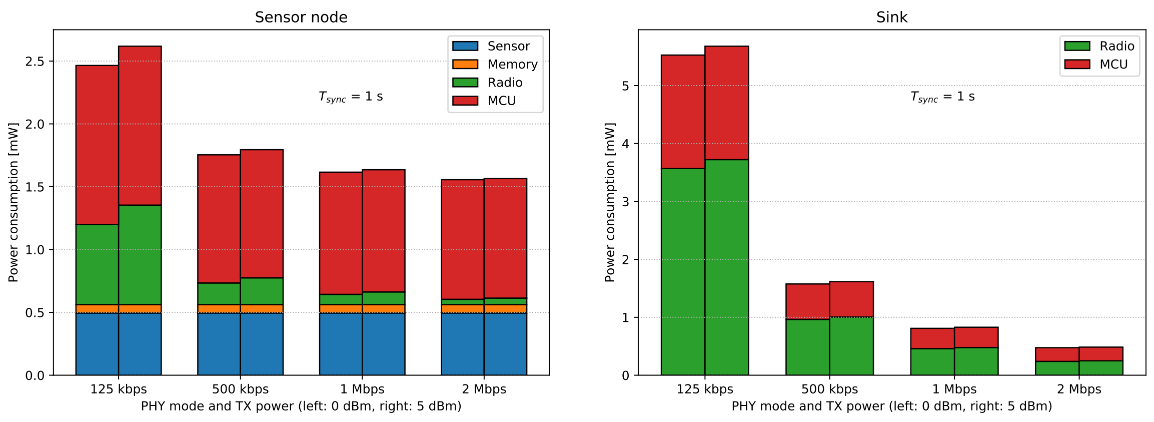


Figure 3.20: Power consumption of the sensor node and the sink in the *measurement state* for different data rates, and transmission power, considering a synchronization period of 1 second.

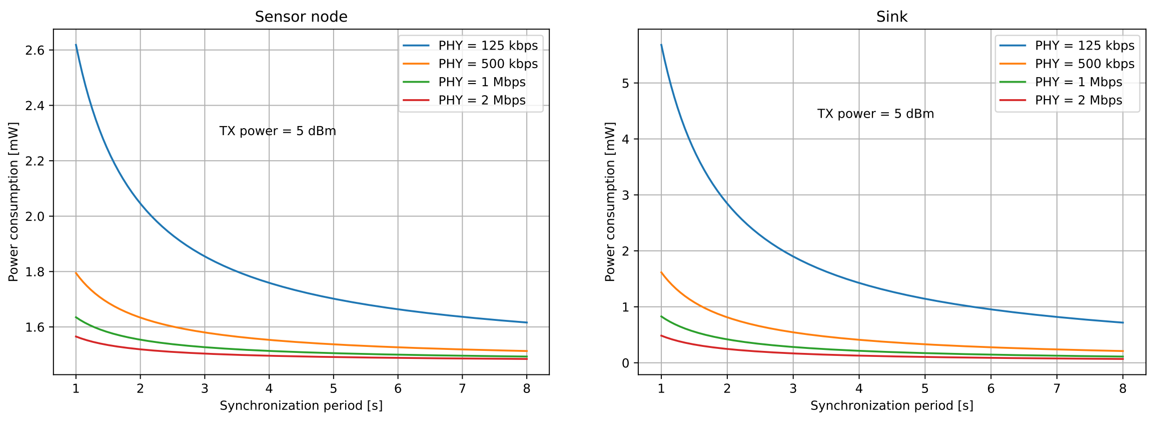


Figure 3.21: Power consumption of the sensor node and the sink in the *measurement state* depending on the synchronization period and for different data rates, considering a transmission power of 5 dBm.

Power consumption of the sensor nodes

The contribution of the accelerometer is already included in the power consumption estimation of the measurement state, as it was considered that this component is always on. A total of 30 000 samples must be read from memory when an event occurs, which results in 16.53 μW by the memory (equation 3.19).

$$D_{mem} = \frac{f_s * N_{sample} * t_{meas}}{f_{SPI} * 3600s} = \frac{1kHz * (3 * 24bits) * 30s}{2MHz * 3600s} = 37.5 \times 10^{-6} \tag{3.18}$$

$$\boxed{P_{mem} = 3.3V * (7.5 \times 10^{-6} * 300\mu A + 0.99 * 5\mu A) = 16.53\mu W} \tag{3.19}$$

The number of packets that must be transferred from a sensor node to the sink is equal to the total data size over the maximum payload size of the BLE packets. To account for the overhead of the data transfer protocol and the errors in the wireless communication, the factor $k_{overhead}$ is included in the calculation of the number of packets transferred (equation 3.23). This factor expresses the amount of additional packets that must be transmitted due to the protocol overhead and communication errors. To simplify the analysis, only the power consumption resulting from packet transmission is considered (equation 3.20); it is reasonable to assume that the number of acknowledgement packets received from the sink will be much lower than the number of data packets transmitted by the sensor node. In this case, the power consumed by the radio depends on the data rate and the transmission power.

$$N_{packets} = \left\lceil \frac{f_s * N_{sample} * t_{meas}}{N_{max\_payload}} * (1 + k_{overhead}) \right\rceil \tag{3.20}$$

$$N_{packets} = \left\lceil \frac{1kHz * (3 * 24bits) * 30s}{254 * 8bits} * (1 + k_{overhead}) \right\rceil = \lceil 1062.99 * (1 + k_{overhead}) \rceil \tag{3.21}$$

$$D_{rad\_tx} = \frac{N_{packets} * t_{pkt\_tx}}{3600s} \tag{3.22}$$

$$\boxed{P_{rad} = V * (D_{rad\_tx} * I_{tx})} \tag{3.23}$$

The MCU is active whenever the memory or the radio are active (equation 3.24), and thus its duty cycle depends on the radio data rate. The power consumed by a sensor node per event per hour will be equal to the sum of the power consumption of the memory, the radio, and the MCU (equation 3.26).

$$D_{mcu} = D_{mem} + D_{rad} = 37.5 \times 10^{-6} + D_{rad} \tag{3.24}$$

$$\boxed{P_{mcu} = 3V * [D_{mcu} * 3.27mA + (1 - D_{mcu}) * 3\mu A]} \tag{3.25}$$

$$\boxed{P_{node} = P_{mem} + P_{rad} + P_{mcu}} \tag{3.26}$$

Power consumption of the sink

The power consumed by the sink is the sum of the power consumption of the radion and the MCU. In a similar way as done for the sensor nodes, only the power consumption resulting from the reception of the data packets is considered when estimating the power consumption of the radio (equations 3.27 and 3.28); it is assumed that during data collection the number of transmitted packets is negligible compared to the number of received packets. As the sink has to collect the acceleration data from all sensor nodes, radio utilization is about ten times greater than in the case of a sensor node. The power consumption of the radio and the MCU depends on the data rate, as it determines the total time required to transfer the data. The MCU is assumed to be active whenever the radio is active. Figures 3.22 and 3.23 present the contribution of a single event per hour to the overall power consumption of the sink and the sensor nodes. In figure 3.22 the power consumption is plotted for different data rates and transmission power (in the case of the sensor nodes), considering an overhead of 50%. In figure 3.23 the power consumption is plotted for different data rates and overhead levels, considering a transmission power of 5 dBm in de case of the sensor nodes.

$$D_{rad\_rx} = \frac{10 * N_{packets} * t_{pkt\_rx}}{3600s} \tag{3.27}$$

$$\boxed{P_{rad} = V * (D_{rad\_rx} * I_{rx})} \tag{3.28}$$

$$D_{mcu} = D_{rad} \tag{3.29}$$

$$\boxed{P_{sink} = P_{rad} + P_{mcu}} \tag{3.30}$$

Having estimated the increase in the average power consumption resulting from an event occurring once per hour, the maximum event rate (equation 3.31) can be estimated considering the maximum allowed power dissipation and the base power consumption set by the measurement state. The bound on the rate of events was calculated using the estimated power consumption of the sink, as it is much higher in the data collection state than in the case of a sensor node. The maximum number of events per hour for different data rates used during data collection is presented in figure 3.24, an overhead of 50% and the worst-case power consumption during the measurement state were considered.

$$\boxed{N_{max\_events} = \left\lfloor \frac{P_{max} - P_{meas}}{P_{data\_collect}} \right\rfloor = \left\lfloor \frac{10mW - P_{meas}}{P_{data\_collect}} \right\rfloor} \tag{3.31}$$

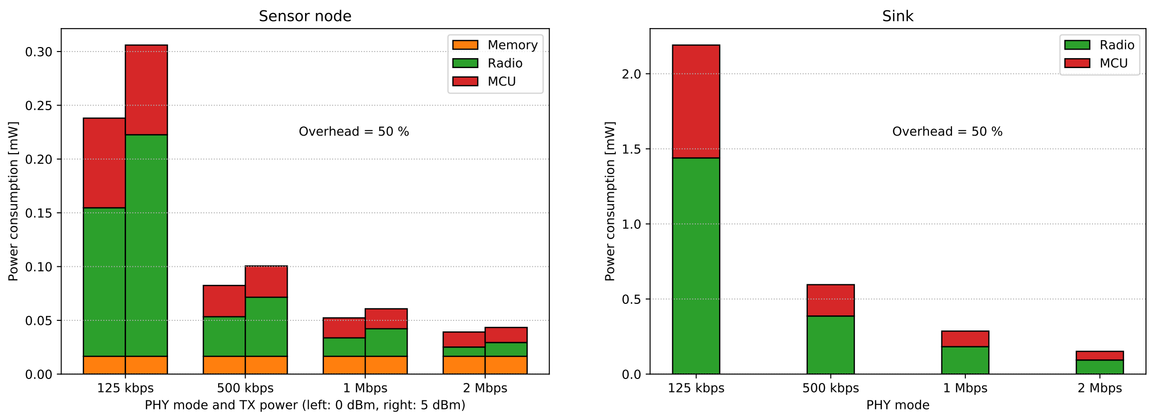


Figure 3.22: Power consumption of the sensor node and the sink in the *data collection state* for different data rates, and transmission power, considering a communication overhead of 50%.

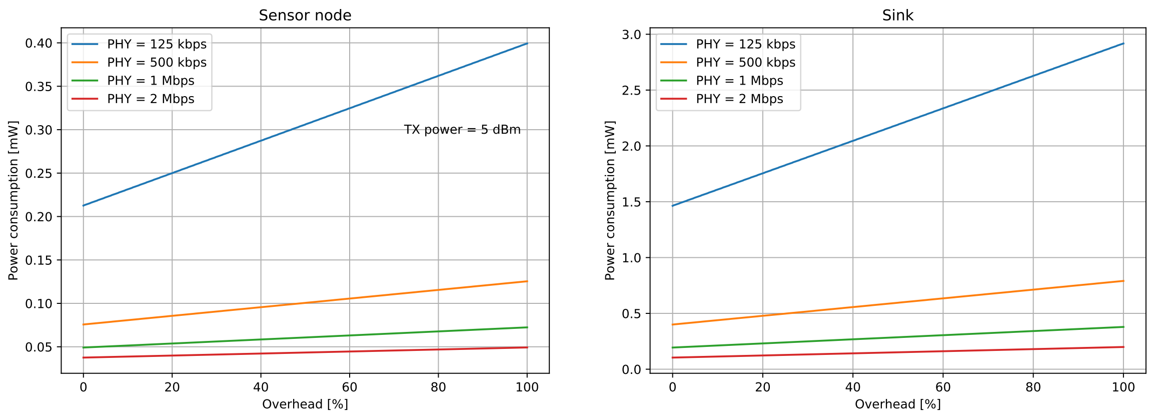


Figure 3.23: Power consumption of the sensor node and the sink in the *data collection state* depending on the communication overhead and for different data rates, considering a transmission power of 5 dBm.

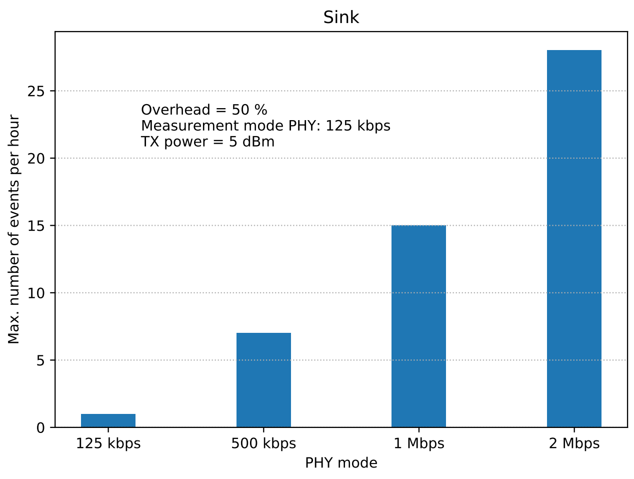
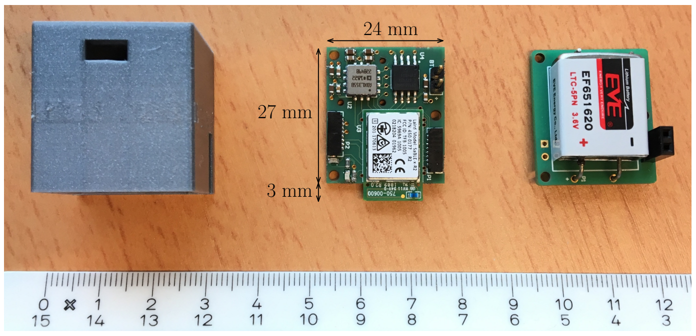


Figure 3.24: Maximum number of data collection events per hour for different data rates used in the data collection state, considering that a data rate of 125 kbps and a transmission power of 5 dBm are used in the measurement state.

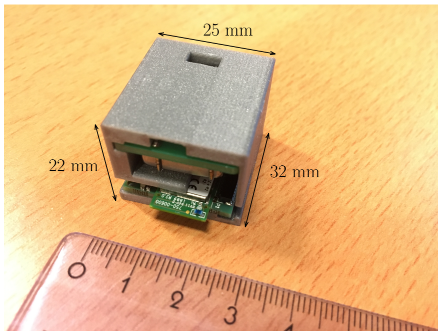
4 Evaluation of the Proposed Solution

4.1. Physical Characteristics of the Sensor Node Prototype

As described in section 3.2.6, the sensor node consists of two boards: the battery-holder board and the main board, the battery is soldered to the former and the latter includes all other components. The battery-holder board and the main board are connected through a standard 2.54 mm two-position header, and are inserted in a custom-designed plastic housing. Figure 4.1 shows a fully assembled sensor node and the individual pieces that comprise it, the sensor node prototype has a volume of 32 x 25 x 22 mm^3 and a total mass of 22 grams.



(a) From left to right: housing, sensor board, and battery



(b) Sensor node

Figure 4.1: The sensor node has a volume of 32 x 25 x 22 mm^3 and a mass of 22 grams.

4.2. Link Quality Inside the Wafer Handler

4.2.1. Test Setup

To evaluate the reliability of the wireless communication, a network including three sensor nodes and the sink was installed inside a Wafer Handler (WH) test rig, figure 4.2 shows a simplified diagram of the setup. The test rig consists mainly of an atmospheric WH which has two large apertures indicated by the dotted lines in the diagram. The aperture on the top of the diagram gives access to a container that holds wafers which can be reached by the load robot (LR) and taken inside the WH. The other aperture on the right side of the diagram faces a platform on a rail that moves between two locations. This system emulates the wafer going to the next stage of the machine and coming back to the WH after processed; when one of the robots places a wafer on the platform, the platform moves to the opposite location where the other robot can reach the wafer. The test rig allows to continuously run a predefined sequence where the load and unload robots (LR, UL) move around wafers as if it were operating on a lithography machine.

Figure 4.2 also shows the location of the sensor nodes and the sink inside the atmospheric WH:

- Sensor node 1 was located on a side of the pre-alignment unit (PA), which moves on the vertical axis with a range of a few centimeters.
- Sensor node 2 was located on a side of the base of the load robot (LR), which has a radius of several centimeters and rotates around the vertical axis.
- Sensor node 3 was located on a side of the second segment of the unload robot (UL), next to the joint. Thus, node 3 moves with two degrees of freedom.
- The sink was located on an inner side of the WH, which can be easily accessed by an operator, and was connected to a laptop located outside of the WH which receives and stores the output of the test. The *CC2640R2F LaunchPad*, an evaluation board from Texas Instruments, was used as the sink. A picture of this board is shown in figure 4.3.

The locations of the sensor nodes were chosen because of their relevance for monitoring the dynamic performance of the WH, and because each location provides a different mobility pattern and degree, which might have an effect on the reliability of the wireless communication.

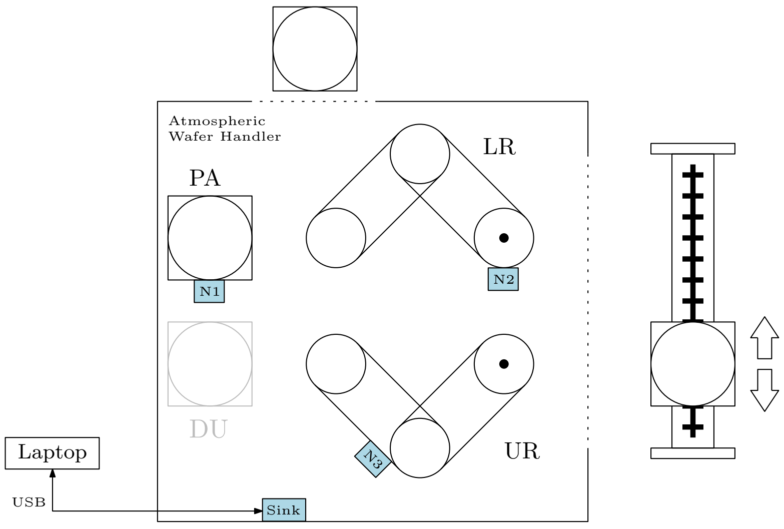


Figure 4.2: Diagram of the atmospheric Wafer Handler (WH) test rig showing the location of the sensor nodes (N1, N2, N3) and the sink. The dotted lines represent apertures in the WH. Wafers enter the WH from the aperture on the top of the diagram. The aperture on the side faces a platform on a rail that moves a wafer between two locations where it can be reached either by the load robot (LR) or the unload robot (UR).

For this experiment, the protocol for exchanging control packets between the sink and the sensor nodes was implemented as described in section 3.4. Time is divided into time frames, which in turn are divided into time slots that are allocated among the devices in the network. The first slot is allocated to the sink which by the start of the time frame transmits a beacon packet containing control information, the arrival time of the beacon is used by the sensor nodes for time synchronization. Successively, the sensor nodes transmit a reply packet to the sink at the start of their respective time slots, the reply packet contains control and status information (e.g. the acknowledgement of the reception of the beacon packet). In addition to control packets, in this experiment the sink exchanges test or probe packets with each sensor node. After transmitting a control packet (i.e. a beacon or a reply), the sink or a sensor node transmits consecutively a fixed number packets equally spaced in time. During the first time slot of a time frame the sink transmits a beacon followed by the test packets while the sensor nodes listen to the channel. Similarly, a sensor node transmits a reply packet followed by the test packets during its corresponding time slot, while the sink listens to the channel.

In order to measure the link quality for different combinations of data rates, transmission power, and frequency channels, a different radio configuration is used during each time frame for transferring the test packets. The network iterates over all BLE MCS (125 kbps, 500 kbps, 1 Mbps), and over four different transmission power levels (-21 dBm, -9 dBm, 0 dBm, 5 dBm) and three different BLE frequency channels (37 at 2402 MHz, 18 at 2442 MHz, 36 at 2480 MHz). The frequency channels were selected so that they were at least 15 MHz apart, as according to Watteyne et al. [28] a frequency shift greater than this magnitude can mitigate multipath fading at the 2.4 GHz band. The sequence of radio configurations followed by the network is described by algorithm 1.

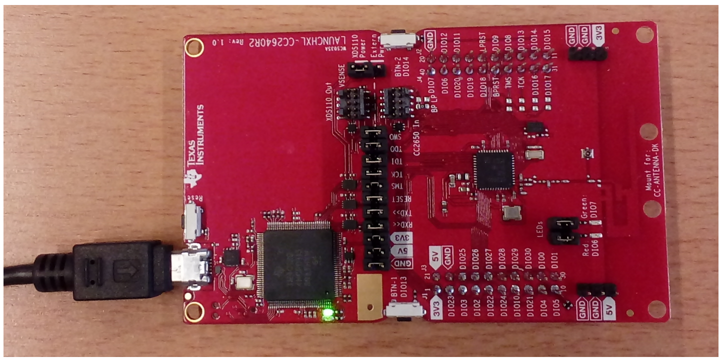


Figure 4.3: The *CC2640R2F LaunchPad* evaluation board from Texas Instruments, used as the sink in the experiments.

Algorithm 1 Radio configuration sequence

```
 1: loop
 2:   for $i$ in $[125\ kbps, 500\ kbps, 1\ Mbps, 2\ Mbps]$ do
 3:     $phy \leftarrow i$
 4:     for $j$ in $[-21\ dBm, -9dBm, 0dBm, 5dBm]$ do
 5:       $power \leftarrow j$
 6:       for $k$ in $[37, 18, 36]$ do
 7:         $channel \leftarrow k$
 8:       end for
 9:     end for
10:   end for
11: end loop
```

Conversely, the radio configuration used for exchanging control packets (i.e. beacons and reply packets) is static so that a sensor node can quickly recover if it ever loses synchronization with the sink node; in this way, there is no need for the sensor node to scan the medium iterating over different radio configurations in search for a beacon to join the network. Control packets are transferred at 125 kbps over channel 37 (2402 MHz) with a transmission power of 0 dBm. These radio parameters were expected to provide a robust communication link as the MCS has the lowest channel code rate (highest redundancy), the frequency channel is outside of WiFi's non-overlapping channels to avoid interference from external networks, and the transmission power is relatively high.

An overview of the behavior of the network during the experiment is presented in figure 4.4. It should be noted that the frequency channel used for transferring the test packets is different at every time frame, while channel 37 is always used for exchanging control packets. In this experiment, the duration of the time frames was set to 1 second, as a time frame is divided into equal intervals among all devices in the network each time slot had a duration of 250 ms (1 second / 4). The time slots were divided further into 11 intervals of equal duration, in the first one a control packet is transferred while at each of the remaining intervals a single test packet is transferred. Whenever a sensor node becomes idle (no data to transmit, receive, or process) it goes to sleep waiting for the start of the next interval to save power. The number of intervals (or subslots) was chosen to fit the maximum number of test packets within a time slot considering the worst-case packet transmission and reception times for the test packets, which corresponds to 17.04 ms in case of the lowest MCS (125 kbps). With a subslot duration of 22.72 ms (250 ms / 11), there is a slack of about 5 ms in the worst case for other tasks apart from receiving or transmitting a packet, such as starting up the radio hardware after waking up[1] and constructing the payload of a packet.

[1]The RF core in the CC2640R2F takes up to 1 ms to boot.

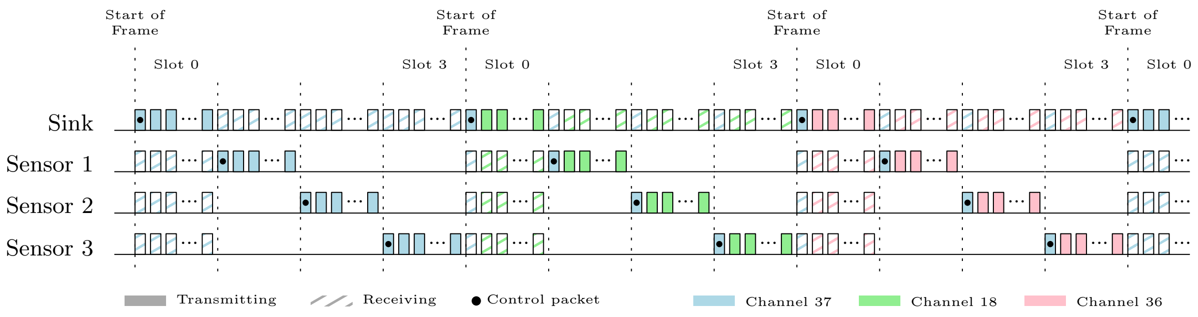


Figure 4.4: Every node transmits ten packets during its corresponding time slot. The beacons include the radio configuration used during the current time frame. The sensor nodes report their observations using the beacon reply packet.

The different fields included in the payload of the beacon, reply, and test packets are presented in table 4.1. The payload of the beacons includes the radio configuration parameters used during the current time frame. According to these parameters, the sensor nodes set the radio configuration for receiving or transmitting test packets during a given time frame. The sequence of radio configuration parameters presented in algorithm 1 is known by the sensor nodes (at compile time), so that whenever a beacon is lost the sensor nodes can determine the current radio configuration based on the last received beacon. Only after three consecutive attempts to receive a beacon have failed the sensor node assumes that synchronization with the sink is lost, and then listens to the channel for the duration of a time frame to re-synchronize with the next beacon.

Table 4.1: Payload fields of beacon, reply, and test packets.

Packet type	Payload fields	Field size [bytes]	Description
Beacon	Node ID	1	Sensor node ID (sink: 0)
	Timestamp	4	RTC timestamp
	PHY	1	0: 125 kbps, 1: 500 kbps, 2: 1 Mbps, 3: 2 Mbps
	TX pow	1	0: -21 dBm, 1: -9 dBm, 2: 0 dBm, 3: 5 dBm
	Channel	1	0: channel 37, 1: channel 18, 2: channel
Reply	Node ID	1	Sensor node ID (1-3)
	Beacon status	1	ACK: beacon received, NAK: beacon lost
	Consec. errors	1	Max. consecutive test packet error count
	Total errors	1	Total test packet error count
	RSSI	1	Average RSSI of received test packets
	Batt. voltage	1	Battery voltage
Test	Sequence	254	Pseudo-random sequence (Fibonacci LFSR)

The payload of the reply packets contains mainly the status of the reception of the beacon and the test packets sent by the sink during the current time frame. The *beacon status* field indicates if the beacon was successfully received. The *consecutive errors* field is a counter of the maximum number of consecutive (test) packets lost, whereas the *total errors* field is a counter of the total number of packets lost. The average RSSI of the test packets is also included in the payload of the reply packets. As the RSSI can only be obtained if packets are received, this field can be set to zero by a sensor node to indicate that all test packets were lost and the RSSI is not available.

The payload of the test packets consists of a sequence of pseudo-random numbers generated by a software implementation of the Fibonacci linear-feedback shift register (LFSR), using as a seed a true random number obtained from the true random number generator (TRNG) in the CC2640R2F. A pseudo-random sequence of 254 bytes is generated to be included in the payload of each test packet, this length corresponds to the maximum payload size of a BLE packet. Contrary to the beacon and reply packets, which were meant to be short to reduce the probability of communication errors, the test packets were made as long as possible so that the result of the experiment would correspond to the worst-case scenario. Table 4.2 presents the duration of the beacon, reply, and test packets, it should be noted that the duration of a test packet at 125 kbps is about 13 times greater than the duration of the control packets.

Table 4.2: Packet transmission and reception times for the different packet types and MCS. The beacon and reply packets are only transferred at 125 kbps.

Packet type	Packet transmission/reception time			
	125 kbps	500 kbps	1 Mbps	2 Mbps
Beacon	1.29 ms	N/A	N/A	N/A
Reply	1.16 ms	N/A	N/A	N/A
Test	17.04 ms	4.54 ms	2.12 ms	1.06 ms

The results of a communication round between the sink and a sensor node, which refers to the exchange of control and test packets in both directions of the link within a single time frame, are formatted as comma-separated values (CSV) and sent to the laptop by the sink over the USB connection to be stored in a file. Table 4.3 enumerates the fields included in every line of the CSV output file, these are:

- A timestamp relative to the start of the test.
- The radio configuration parameters used for transferring the test packets.
- The status of the reception of the beacon. If the reply packet was received successfully, this field indicates the status of the reception of the beacon by the sensor node (ACK or NAK). Otherwise, this field indicates if the reply packet was lost due to a CRC error or a reception timeout.
- The maximum number of consecutive test packets lost for the downlink and uplink.
- The total number of test packets lost for the downlink and uplink.
- The average RSSI of the test packets that were successfully received for the downlink and uplink.

The results for the downlink are obtained by the sink from the reply packet sent by a sensor node. If this packet is lost then this information is not available, and the fields corresponding to the downlink will have a special value: -1 for the error counts and 0 for the average RSSI. For both the downlink and the uplink, the RSSI will not be available if all test packets are lost, as the RSSI is only valid for received packets. This scenario is also flagged using the special value for the RSSI.

4.2.2. Results

The WH test rig was programmed to continuously execute a predefined sequence while the network was running the link quality test. The results of each communication round were recorded for more than 19 hours, and a total of 69821 samples per node were obtained. Appendix B contains plots of the time series of the maximum consecutive error count, the total error count, and the average RSSI. Plots of the histogram of the maximum consecutive error count and the average RSSI are also included in appendix B. These results are presented for each sensor node, for the downlink and the uplink, and for each combination of MCS and transmission power.

This section includes only the most relevant results to determine the feasibility of the proposed solution, these are:

- The Packet Error Rate (PER) of the beacons, and the distribution of the maximum consecutive packet loss for the downlink. These are crucial for the reliability of the system, as the network will remain synchronized as long as the communication errors do not persist over relatively long periods of time.

Table 4.3: Columns in CSV output file of the link quality test.

Column	Range	Description
0. Timestamp	$[0, 2^{32})$	RTC timestamp relative to the start of the test
1. PHY	$[0, 3]$	0: 125 kbps, 1: 500 kbps, 2: 1 Mbps, 3: 2 Mbps
2. TX pow	$[0, 3]$	0: -21 dBm, 1: -9 dBm, 2: 0 dBm, 3: 5 dBm
3. Channel	$[0, 2]$	0: channel 37, 1: channel 18, 2: channel
4. Node ID	$[1, 3]$	Sensor node ID
5. Reply STA	$\{0, 1, 2, 4\}$	0: NAK, 1: ACK, 2: CRC error, 4: timeout error
6. DL consec. errors	$[-1, 10]$	Count of max. consecutive errors (downlink). -1: unknown, reply packet lost
7. UL consec. errors	$[0, 10]$	Count of max. consecutive errors (uplink)
8. DL total errors	$[-1, 10]$	Count of total errors (downlink). -1: unknown, reply packet lost
9. UL total errors	$[0, 10]$	Count of total errors (uplink)
10. UL RSSI	$[-100, 0]$	Uplink average RSSI. 0: unknown, all test packets lost
11. DL RSSI	$[-100, 0]$	Downlink average RSSI. 0: unknown, reply packet lost or all test packets lost
12. Battery voltage	-	Node's battery voltage. 0: unknown, packet lost

- The uplink PER, which allows to estimate more accurately the power consumption of the sink node during the data collection state, as it gives an indication of the number of packets that would require retransmission.

Plots of the time series of the status of the beacon reception for each sensor node are presented in figure 4.5. These plots show for each time frame whether the beacon was acknowledged or not by the sensor node, or if the reply packet was lost. The PER of the beacon and the reply packets is given in table 4.4. As there is missing information for the calculation of the beacon PER because of the loss of several reply packets, the best-case and worst-case values are included for the beacon PER. For the best-case it was assumed that the beacons were successfully received whenever the reply packet was lost, for the worst-case it was assumed that the beacons were also lost. Node 1 had the highest beacon PER with about 445 packets per million in the worst-case, while node 3 had the lowest with nearly 60 packets per million.

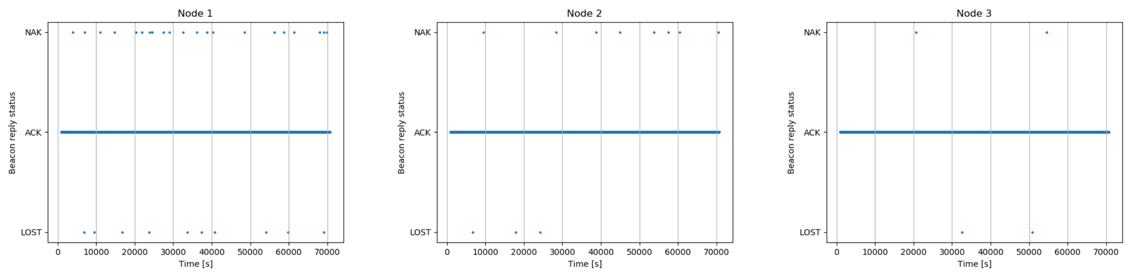


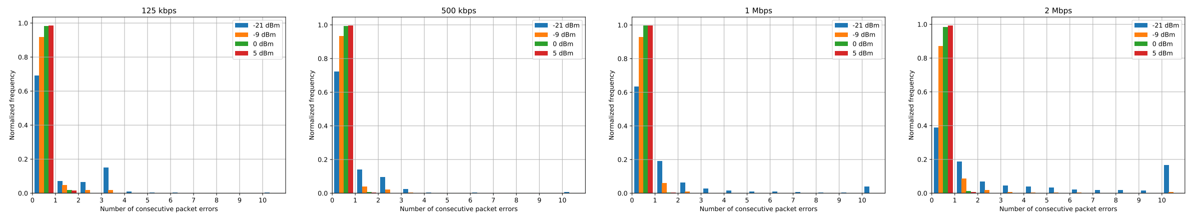
Figure 4.5: Status of the beacon reply packets sent by each sensor node to the sink. ACK: the beacon was received successfully; NAK: the beacon was lost; LOST: the reply packet was lost, it is unknown if the beacon was successfully received.

Table 4.4: Beacon and reply Packet Error Rate (PER) for each sensor node.

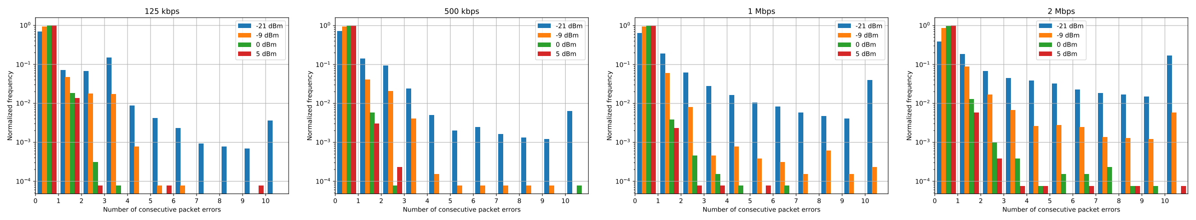
	Total number of beacon packets	ACK	NAK	Reply packets lost	Best-case beacon PER	Worst-case beacon PER	Reply PER
Node 1	69821	69790	21	10	$300.76\text{x}10^{-6}$	$443.99\text{x}10^{-6}$	$143.22\text{x}10^{-6}$
Node 2	69821	69810	8	3	$114.57\text{x}10^{-6}$	$157.54\text{x}10^{-6}$	$42.96\text{x}10^{-6}$
Node 3	69821	69817	2	2	$28.64\text{x}10^{-6}$	$57.28\text{x}10^{-6}$	$28.64\text{x}10^{-6}$

Figure 4.6 shows the distribution of the consecutive number of test packets lost during a communication round, for the downlink, and for every MCS and transmission power. The samples from all sensor nodes were considered together in the calculation of the histograms, this gives a total of $210 \cdot 10^3$ samples, and more than $13 \cdot 10^3$ samples for each combination of MCS and transmission power. The histograms are plotted using both a linear and a logarithmic scale, to respectively show the proportion and the range of the values. The plots can be more easily read if a single color (transmission power) is considered at a time, by doing so each of the normalized histograms can be clearly seen in plots with a linear scale.

Consecutive errors are observed in at least 30% and 10% of the samples for all MCS when using a transmission power of -21 dBm and -9 dBm, respectively. This rate decreases to less than 2% when using a transmission power of 0 dBm or 5 dBm. In general, for all MCS the distribution of the number of consecutive errors concentrates towards zero as the transmission power is increased. However, samples with a number of consecutive errors greater than five can be found even when the transmission power is 0 dBm or 5 dBm.



(a) Linear scale



(b) Logarithmic scale

Figure 4.6: Normalized histograms of the overall downlink consecutive packet loss observations for all sensor nodes depending on the Modulation and Coding Scheme (MCS) and the transmission power. In total, ten test packets are sent during each communication round.

Figure 4.7 shows the uplink PER for each MCS and transmission power, indicating the contribution of each sensor node. Table 4.5 contains the value of the overall uplink PER for each scenario. There is no uncertainty in the measurement of the uplink PER as the observations are made directly by the sink, and therefore the loss of reply packets does not affect the measurement as in the case of the downlink. It can be observed that the uplink PER ranges from 5% to 22% for the lowest transmission power, while for a transmission power of 0 dBm or 5 dB% the PER is lower than 1% for all MCS. Additionally, for the lowest transmission power the distribution of the PER among the sensor nodes is more balanced than in the case of a higher transmission power. Contrary to expectations, the 125 kbps MCS did not provide the lowest PER, with exception of the case where a data rate of 2 Mbps and transmission power of -21 dBm are used, all other MCS showed a better performance.

Table 4.5: Overall uplink Packet Error Rate (PER).

	125 kbps	500 kbps	1 Mbps	2 Mbps
-21 dBm	15.53 %	5.05 %	7.93 %	21.94 %
-9 dBm	4.46 %	2.03 %	1.33 %	2.65 %
0 dBm	0.60 %	0.30 %	0.17 %	0.30 %
5 dBm	0.22 %	0.06 %	0.04 %	0.09 %

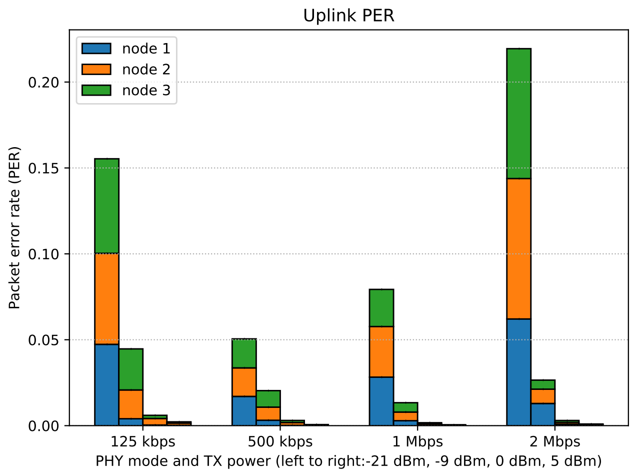


Figure 4.7: Uplink Packet Error Rate (PER) depending on the Modulation and Coding Scheme (MCS) and the transmission power.

4.3. Worst-Case Synchronization Error

4.3.1. Test setup

The objective of the experiment was to measure the worst-case clock offset between any two nodes in a network running the synchronization protocol described in section 3.4. A diagram and pictures of the test setup are shown in figures 4.8 and 4.9. The network consisted of the sink and four different sensor nodes, a CC2640R2F LaunchPad evaluation board was again used as the sink.

Each device in the network generates a pulse around the time of the start of a time frame. In the case of the sink, the rising edge of the pulse corresponds to the start of the transmission of the beacon, which coincides exactly with the start of the time frame and serves as a reference. On the other hand, the rising edge of the pulse generated by a sensor node indicates the predicted or expected time of the start of the transmission of the beacon based on its local clock. The output of the sink was connected to the external trigger input of an oscilloscope, while the output of each sensor node was connected to one of the input channels of the oscilloscope. In this way, the offset between the local clocks of the sensor nodes can be observed. A digital oscilloscope with a bandwidth of 350 MHz and a sampling rate of 4 Gsps was used.

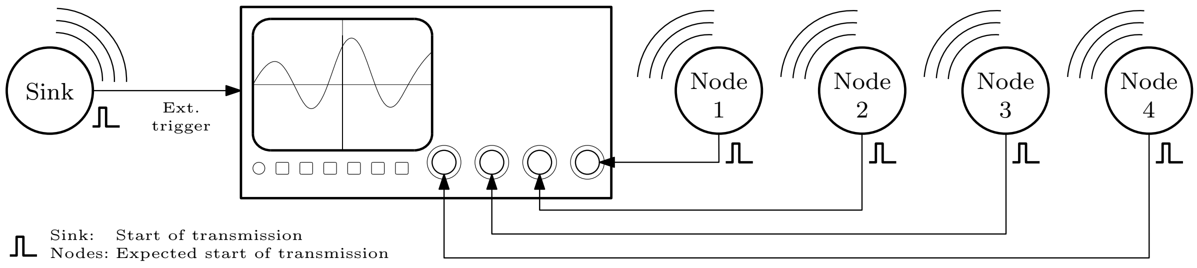
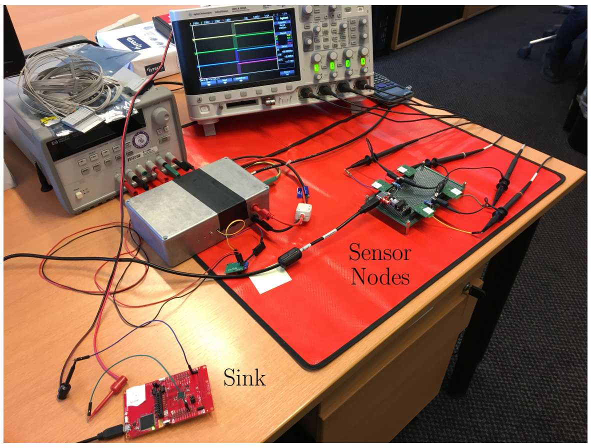

Sink
Ext. trigger
Node 1
Node 2
Node 3
Node 4
Sink: Start of transmission
Nodes: Expected start of transmission

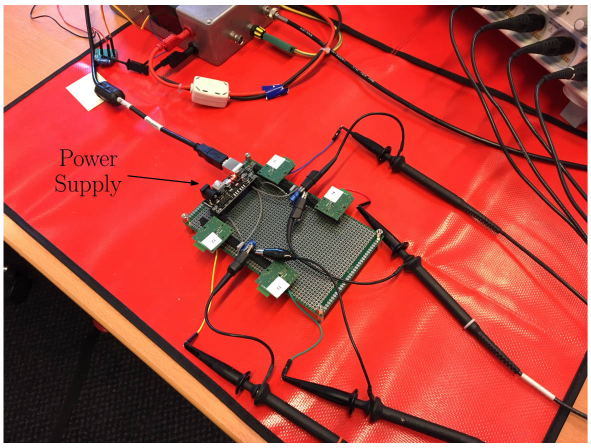
Figure 4.8: Test setup for measuring the worst-case clock offset between four sensor nodes and the sink. The sink generates a pulse at the start of the transmission of every beacon, while the sensor nodes generate at the expected start of the transmission.

The diagram in figure 4.10 shows the events that occur around the start of the time frames, which duration was set to 1 second for this experiment and corresponds to the re-synchronization period. The red arrows indicate the rising edge of the output pulse, which in the case of a sensor node signals the expected start of the transmission of the beacon. The upward and downward grey arrows represent respectively the wake-up and sleep time of the sensor node, for clarity these have been omitted for the sink.

About 2 milliseconds before the expected start of a frame, a sensor node is awakened by an interrupt from the real-time clock (RTC). After having initialized the radio hardware, which includes the synchronization of the radio timer with the RTC, the main CPU sends a command to the RF core to schedule the start of the reception. Another command is sent to the RF core to set a compare value for



(a) The sink is connected to the external trigger input of the oscilloscope



(b) The sensor nodes are connected to a power supply board

Figure 4.9: Pictures of the network synchronization test setup.

the radio timer, the compare value corresponds to the expected start of the transmission. The radio timer generates an event when its counter matches the programmed compare value, this event was routed to an output pin of the MCU and corresponds to the red arrows in figure 4.10. After the beacon is received, the drift of the RTC is compensated as described in section 3.4, and the start of the next frame is calculated based on the timestamp of the beacon. At this point the MCU goes back to sleep after having scheduled the wake-up time by setting the RTC interrupt. If a beacon is lost, the wake-up time and the start of the next time frame are calculated from the timestamp of the last received beacon. In the diagram, the sensor node starts listening to the channel earlier than the expected start of the beacon transmission, this is because there is a guard time of $50\ \mu s$ to account for the synchronization error.

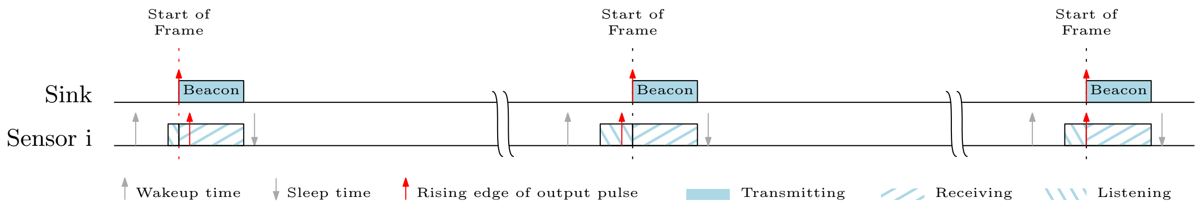


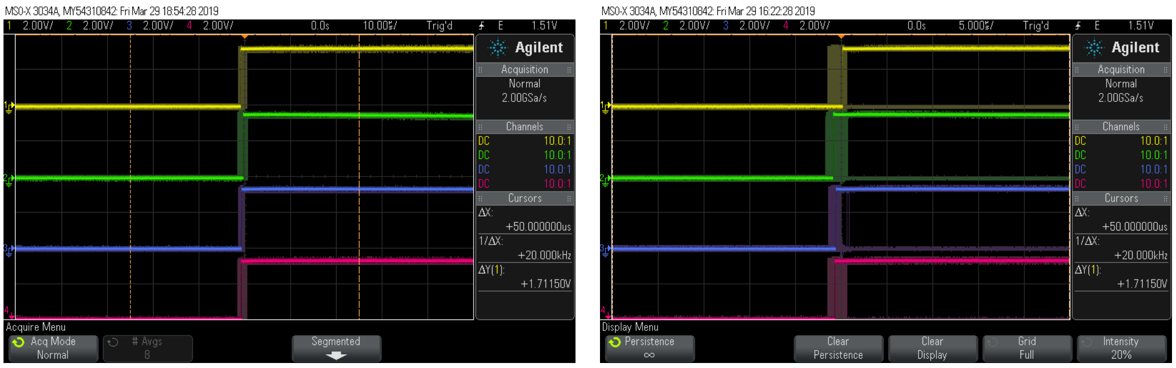
Figure 4.10: Before the start of a time frame the sensor node wakes up and schedules the transition on the output signal, after the beacon is received the offset and drift are adjusted and the sensor node goes back to sleep.

The beacons were transferred over channel 17 (2440 MHz) with a transmission power of 0 dBm, using the 125 kbps MCS. These radio configuration parameters where selected as they are expected to provide a robust link, as it would be required for the communication inside the WH.

4.3.2. Results

To measure the worst-case offset between the local clocks of the sensor nodes, the network ran for about 48 hours with the persistent display mode of the oscilloscope enabled. Figure 4.11 presents captures of the screen of the oscilloscope showing the superposition of the rising edges of the output pulse from each sensor node, for the total duration of the experiment.

It can be observed in figure 4.11b that the maximum clock offset between any pair of nodes was lower than $5\ \mu s$, and that for all sensor nodes the boundaries of the offset were appreciably symmetric with respect to the reference signal, i.e. the output pulse from the sink. As describe in section 3.4, the sensor nodes include a crystal oscillator with a tolerance of $\pm 20\ ppm$. Considering this, the results of the experiment indicate that the effective drift of the local clocks was reduced to less than $\pm 2.5\ ppm$ in the worst-case. In figure 4.11b a larger time scale and markers were used to show the distance between the worst-case clock offset and the required maximum synchronization error ($\approx 50\ \mu s$).



(a) Markers indicate the maximum synchronization error ($\approx 50\ \mu s$)

(b) The synchronization error is less than $5\ \mu s$ for all sensor nodes

Figure 4.11: Oscilloscope screen captures showing the maximum clock offset between four different sensor nodes and the sink.

4.4. Conclusions

The size of the sensor nodes does not meet the requirements presented in section 2.3, which set a maximum of 20 mm for each dimension. The depth, width, and height of the sensor nodes exceed this value by 60%, 20%, and 10%, respectively. The height could be easily improved in a redesign of the housing, and the surface of the main board can be reduced by removing unnecessary components that were included for convenience while testing and debugging. However, due to the dimensions of the battery, the surface of the battery-holder board cannot be reduced enough to meet the requirements. Smaller batteries cannot provide the required capacity and/or peak currents, therefore the proposed design comes close to the smallest form factor that can be achieved with COTS components.

The measurements of the link quality inside the atmospheric WH indicate the feasibility of the proposed solution with regards to the requirements on the reliability and lifetime of the system. The reliability of the proposed system depends on the capacity of the network to remain synchronized whenever a measurement is being performed. The synchronization protocol tolerates errors in the communication, however these cannot persist over a relatively long time interval as explained in section 3.4.

The measurements of the downlink PER show that multiple consecutive errors are unlikely to occur when a relatively high transmission power is used (0 dBm or 5 dBm) in combination with any of the MCS. However, samples in which all packets were lost were observed even in the case when the highest transmission power was used. It should be noted that the conditions of the test correspond to a pessimistic worst case, in which the payload size is maximum and all packets are transferred within a relatively short time interval (the duration of a time slot is $250\ ms$, which gives a packet transmission period of $\approx 23\ ms$). In the actual implementation of the protocol, the payload of the beacons would be an order of magnitude smaller than the payload of the test packets used in the experiment, and the re-synchronization period would be greater than 1 second. Considering that the occurrence of multiple consecutive errors was sporadic even under worst-case conditions, it is reasonable to expect that the network will be reliable under normal operating conditions.

On the other hand, the lifetime of the system is greatly determined by the power consumed during the data collection state. The power consumption during this state was estimated in section 3.5 under the assumption of an overhead on the communication of 50%, which consists of the protocol overhead and the packet retransmission rate. The overall uplink PER measured was however lower than 1% for all MCS when a relatively high transmission power (0 dBm or 5 dBm) was used. As presented in 3.5, the most important factor to reduce the power consumption during the data collection state is the communication speed rather than the transmission power. The results obtained show that it is feasible to use the 2 Mbps MCS during the data collection state to minimize the total duration of the data transfer and in turn reduce the power consumption.

The results of the synchronization accuracy test showed that the drift of the local clocks was reduced to less than $\pm 2.5\ ppm$, which is eight times lower than the tolerance of the crystals driving the RTC of the sensor nodes. This capacity to compensate for the drift of the RTC, provided by the synchronization protocol, is key to the reliability of the system as it gives the network resiliency against consecutive communication errors.

4.5. Future Work

The general ideas proposed in section 3.4 for the communication protocol are based on well-known and widely-used principles for channel access and wireless synchronization. A concrete specification of the communication protocol is still missing, e.g. the specification of the channel hoping strategy, the format of the packets, and the protocol for bulk data transfer in the data collection state. As the ideas presented for the communication protocol are not incompatible with the BLE standard, these could be implemented on top of the BLE communication stack as an alternative to designing a fully custom protocol. The initialization, standby, and measurement states could be implemented using the broadcaster-observer roles defines in the BLE standard, while the data collection state could be realized using the central-peripheral roles. However, implementing the synchronization protocol using a COTS BLE stack requires it to allow for precise control over the timing of the communication and access to low-level timestamps. It should be investigated then if there are COTS BLE stacks that meet these requirements.

In this work only the aspects related to the wireless communication were considered for the design of the sensor nodes. The firmware associated with the measurement functionality is yet to be developed in order to fully validate the design. It is particularly important to measure the worst-case error in the synchronization between the MCU and the accelerometer, as it determines, together with the network synchronization error, the overall time accuracy of the system. Additionally, different strategies for driving the operation of the memory and the sensor can be evaluated with the objective to minimize CPU utilization. As the contribution of the CPU to the overall power consumption was estimated to be significant, the implementation of the measurement functionality should strive for a power-efficient use of the CPU.

The results obtained for the link quality inside the WH provide evidence of the reliability of the system. However, there are significant differences between the test rig used in the experiment and a fully operational lithography machine, e.g. the network would be fully enclosed in metallic structures, would operate in a vacuum environment, and would possibly be exposed to stronger EMI. Therefore, to fully asses the reliability of the system the experiments should be repeated with the network installed inside an actual machine.

A

Schematics: Sensor Node and Battery Holder

Figure A.1: Sensor node schematic.

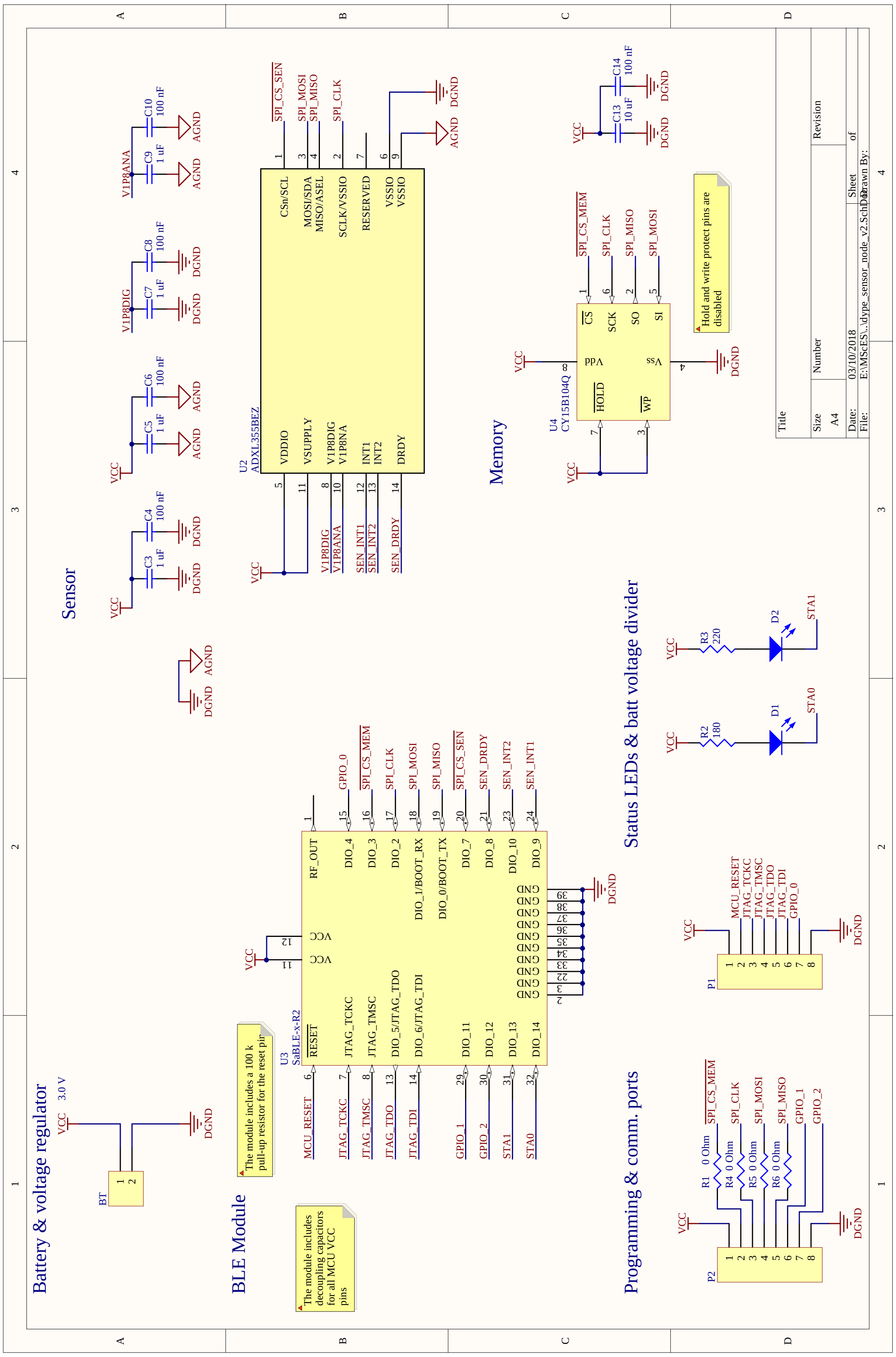
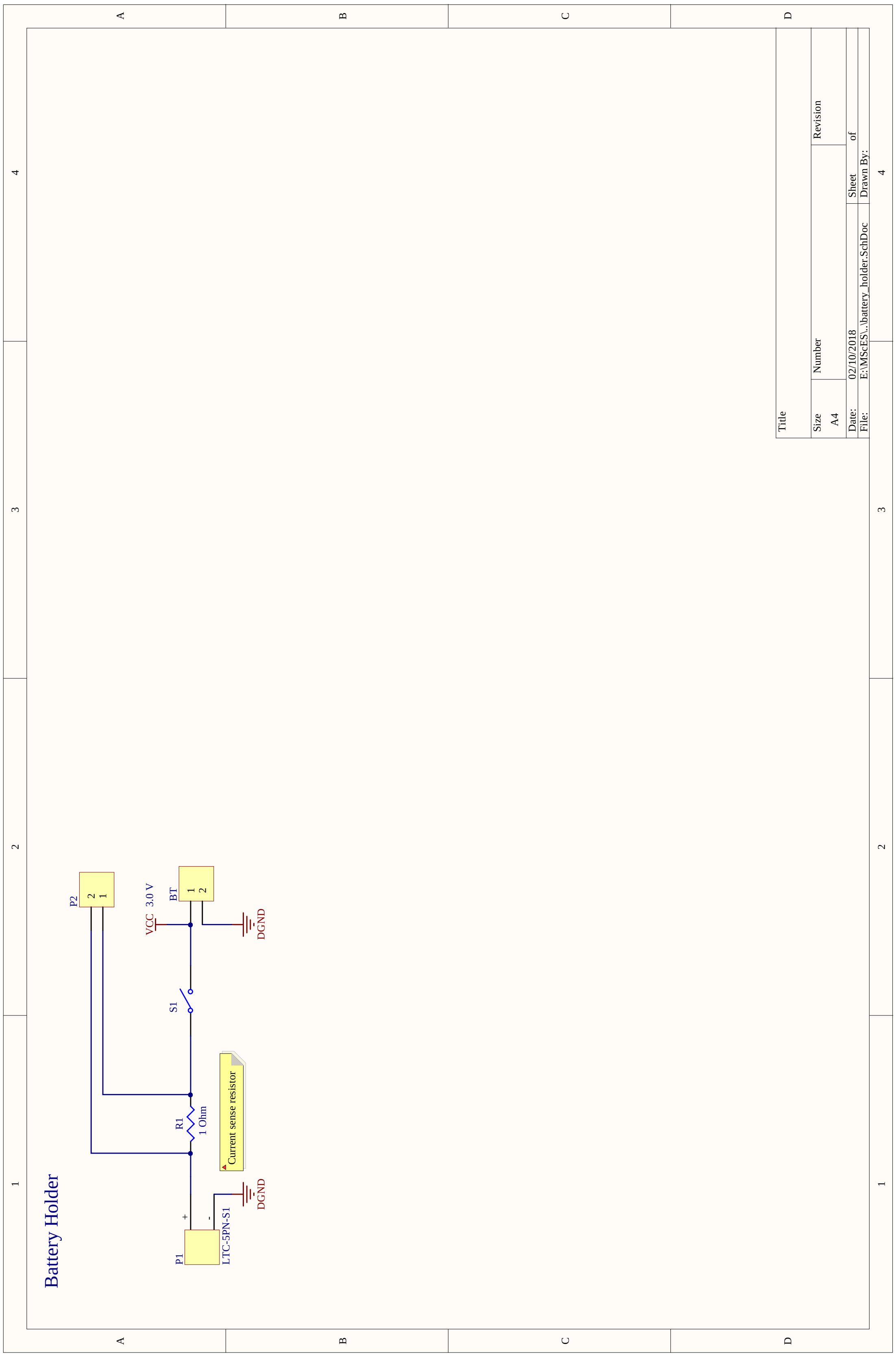


Figure A.2: Battery holder schematic.

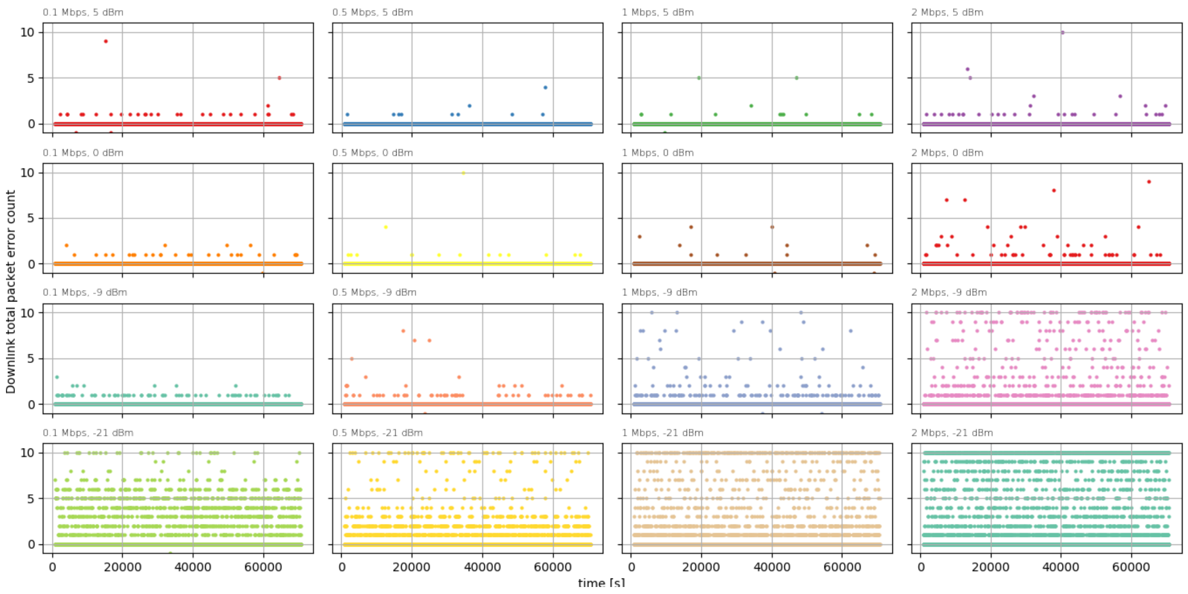


B

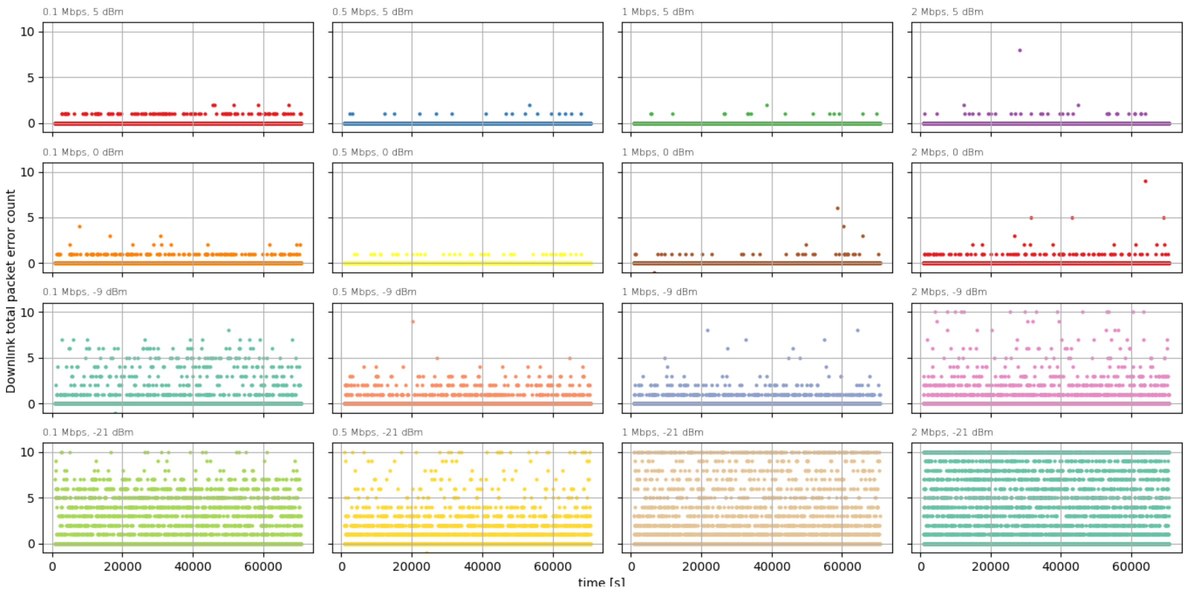
Results of the Link Quality Test

Figure B.1: Downlink total packet error count.

(a) Node 1



(b) Node 2



(c) Node 3

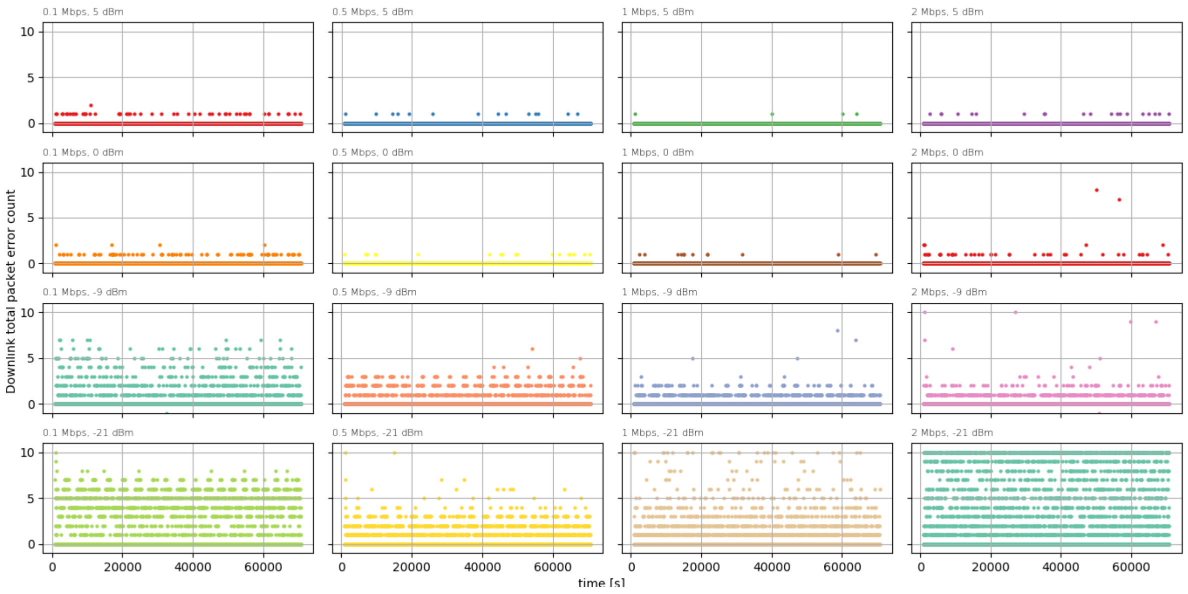
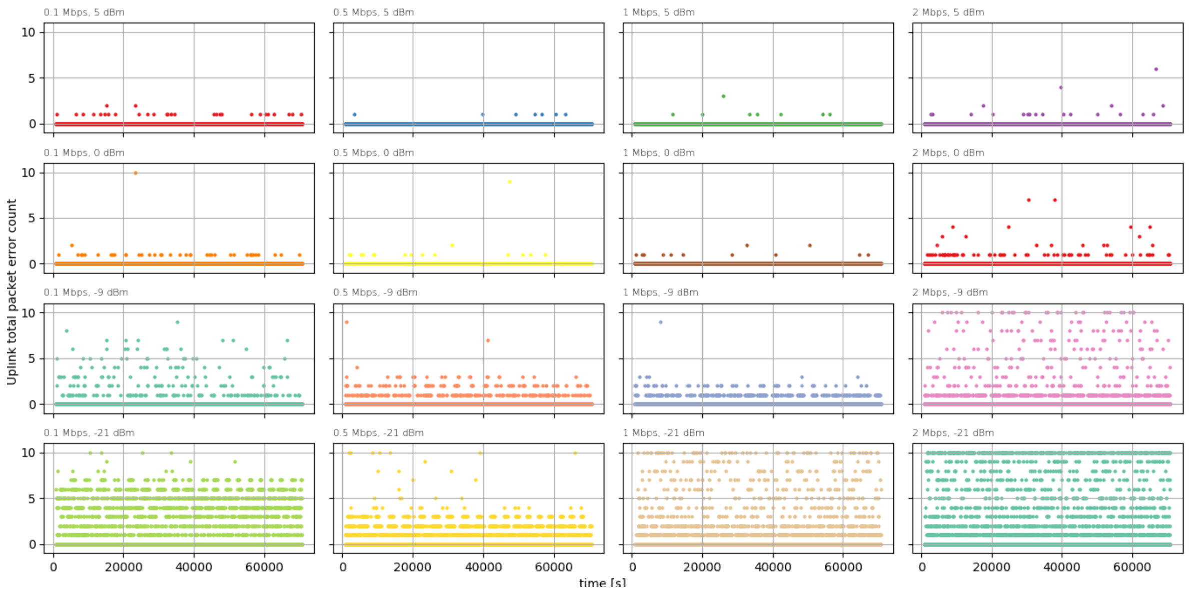
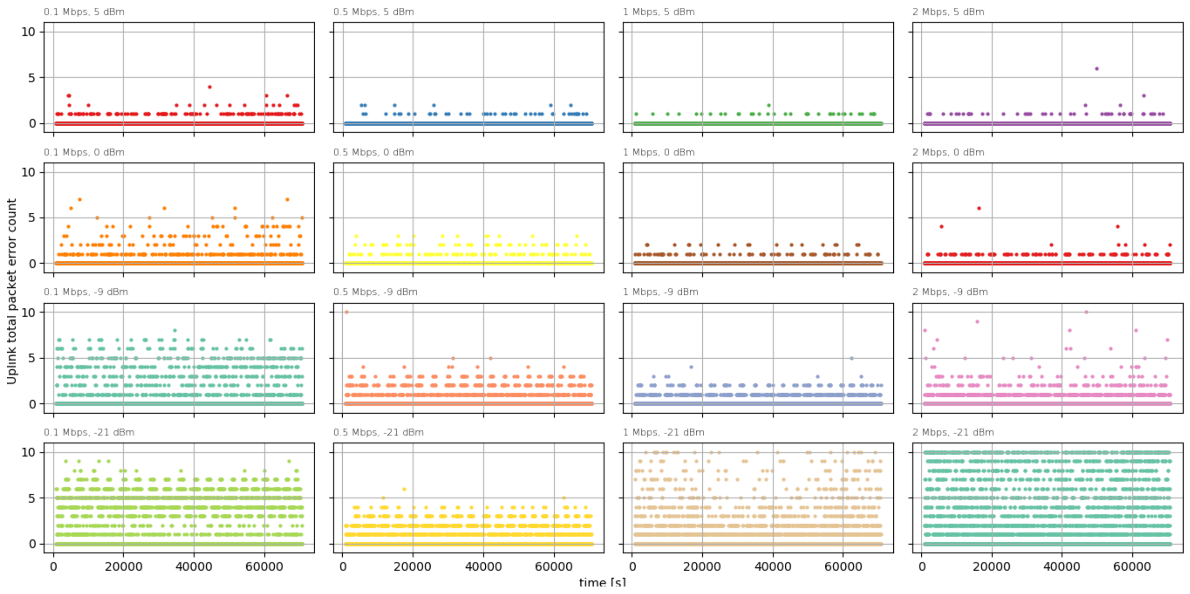


Figure B.2: Uplink total packet error count.

(a) Node 1



(b) Node 2



(c) Node 3

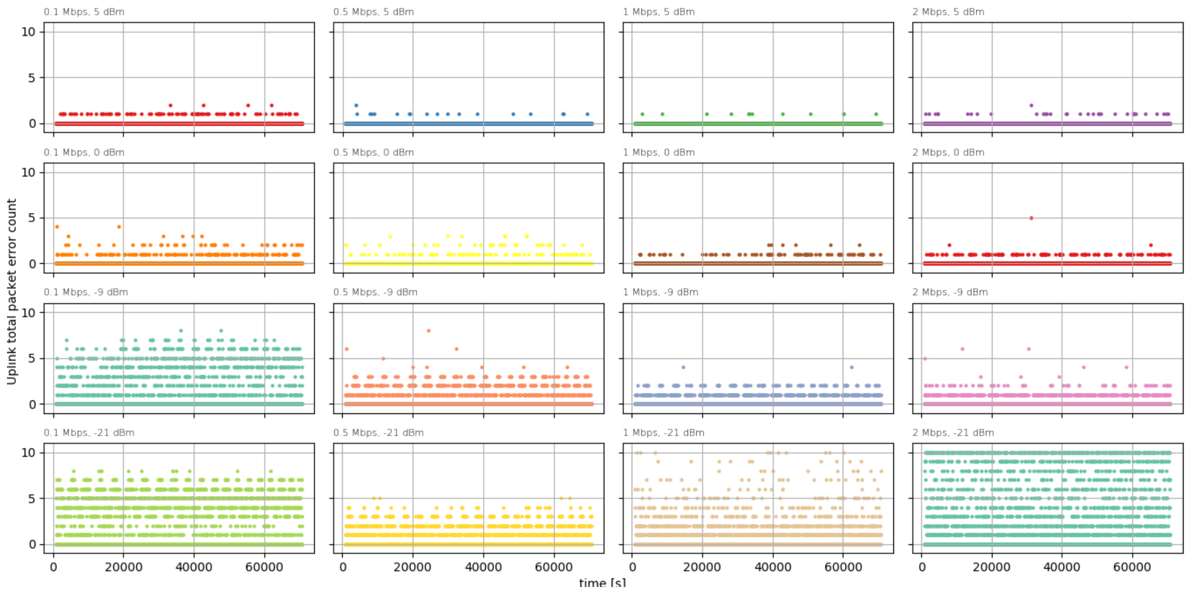
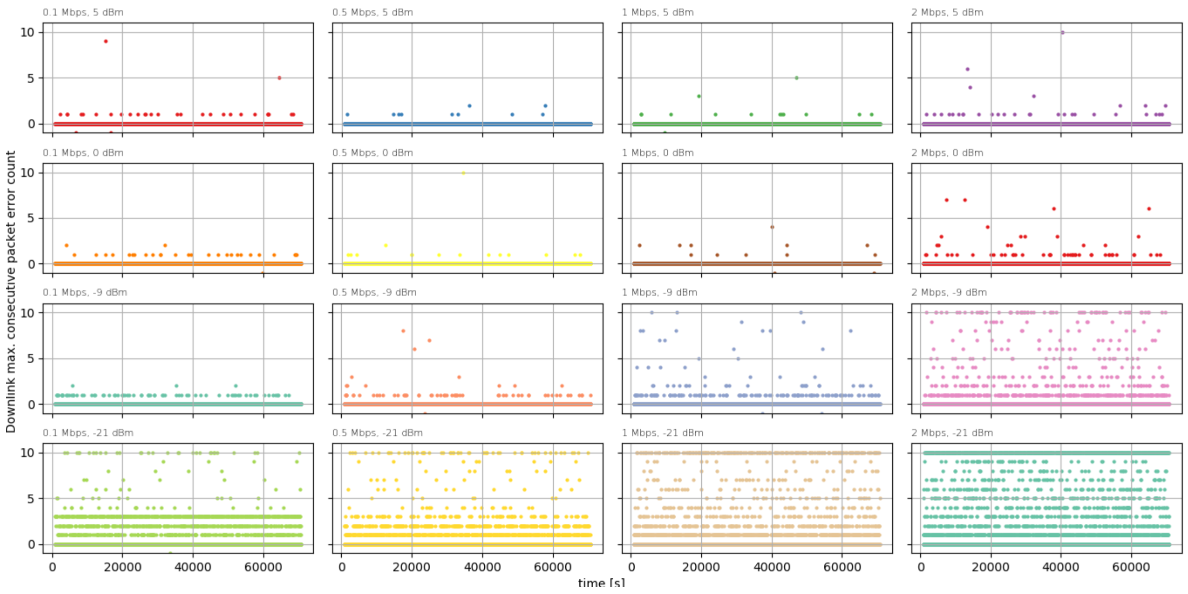
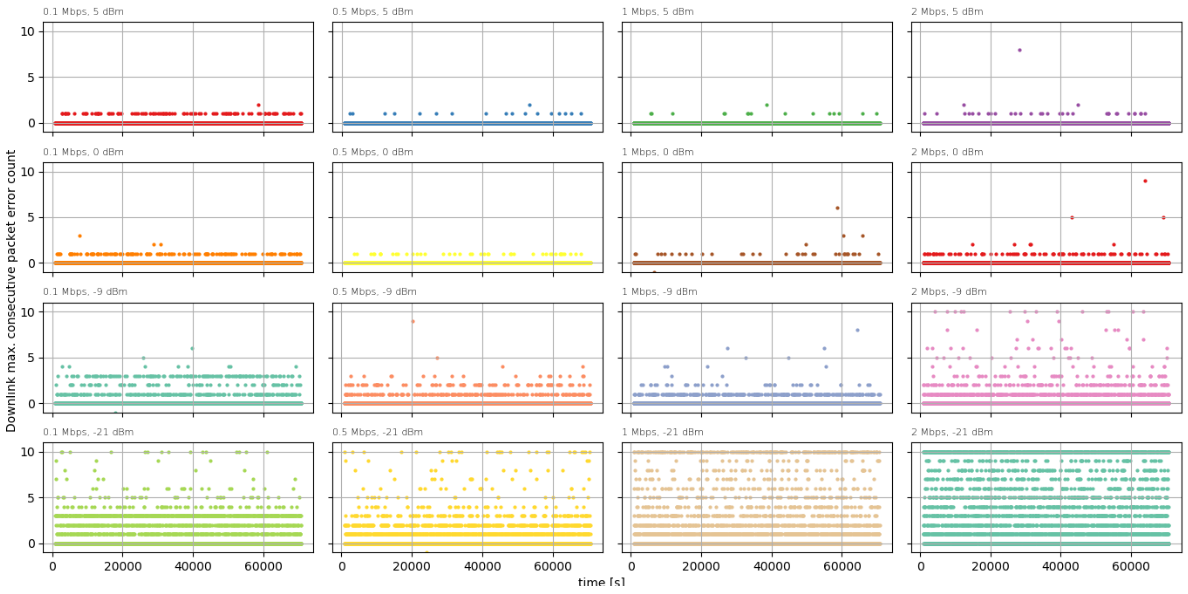


Figure B.3: Downlink maximum consecutive packet error count.

(a) Node 1



(b) Node 2



(c) Node 3

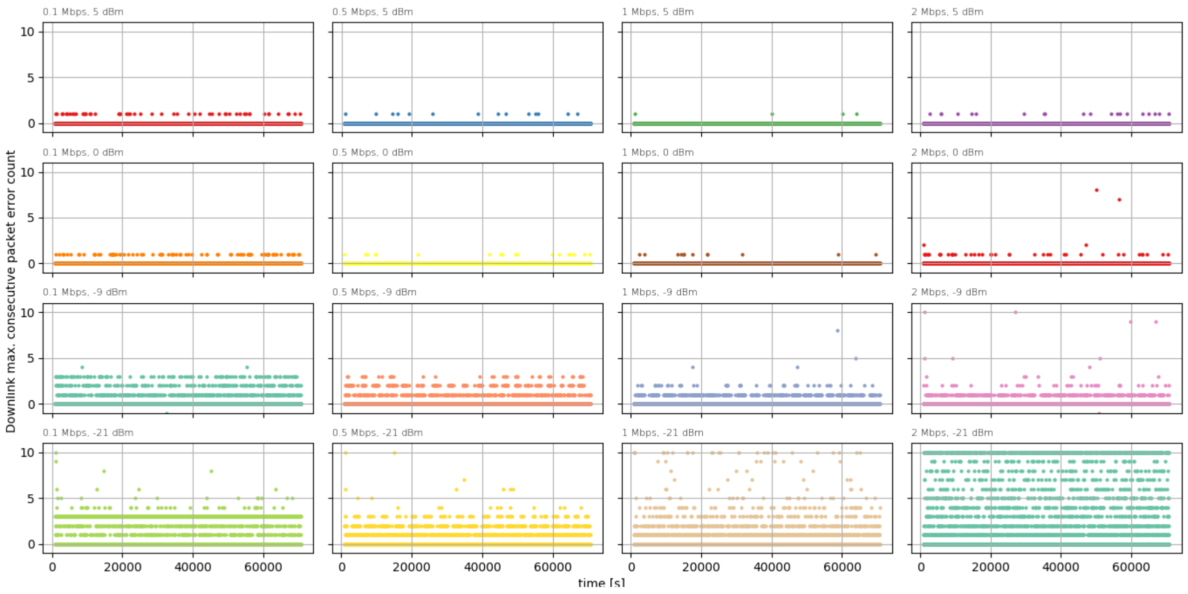
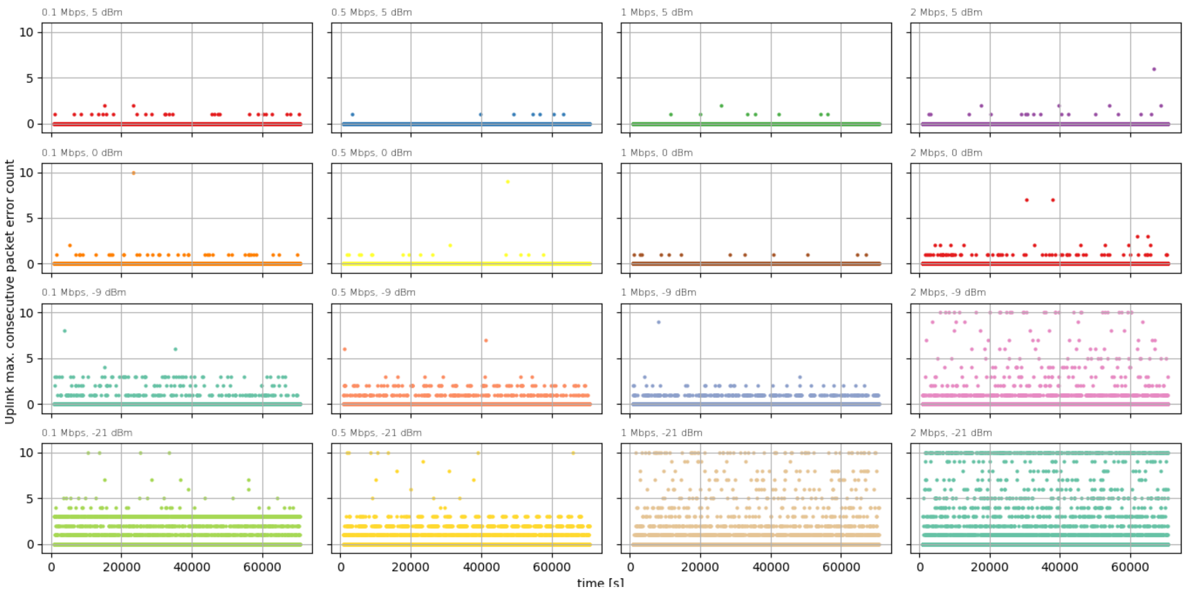
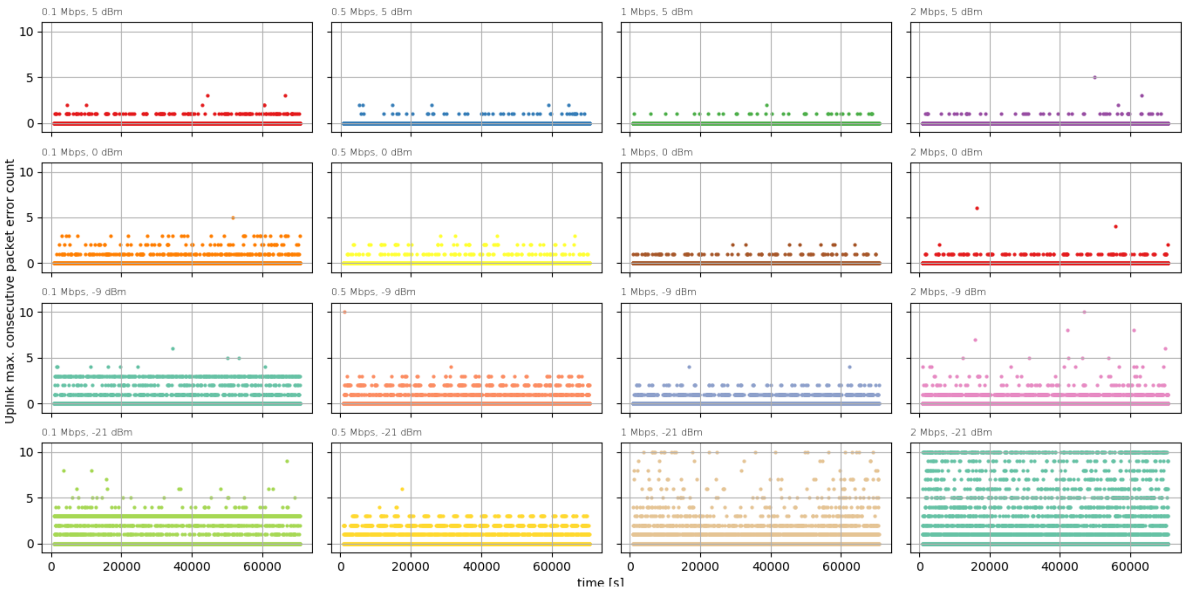


Figure B.4: Uplink maximum consecutive packet error count.

(a) Node 1



(b) Node 2



(c) Node 3

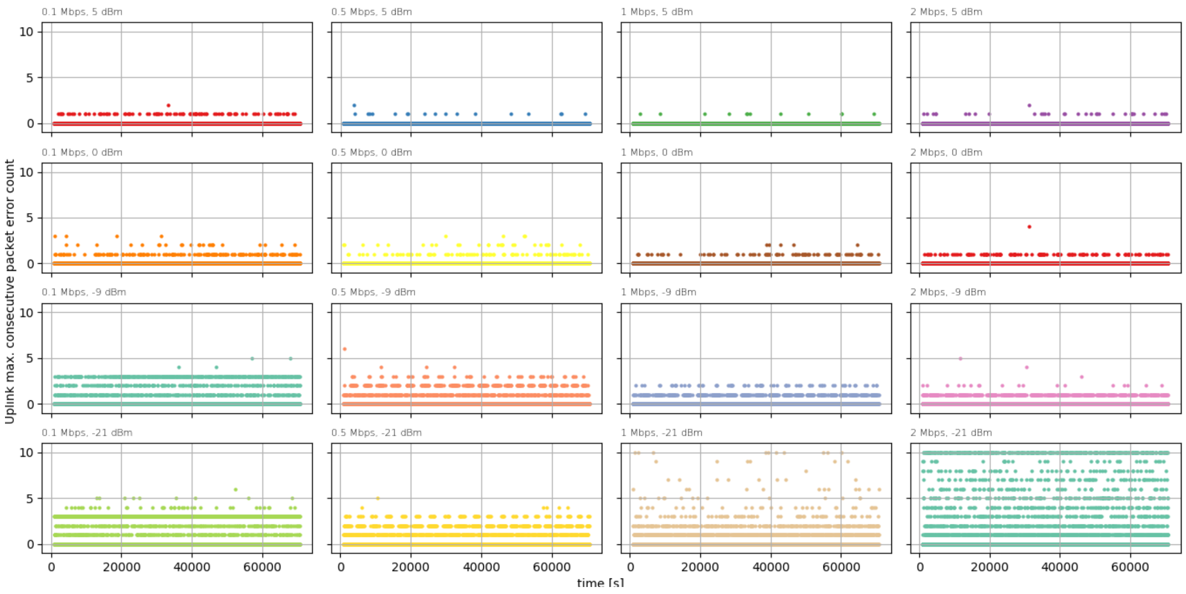
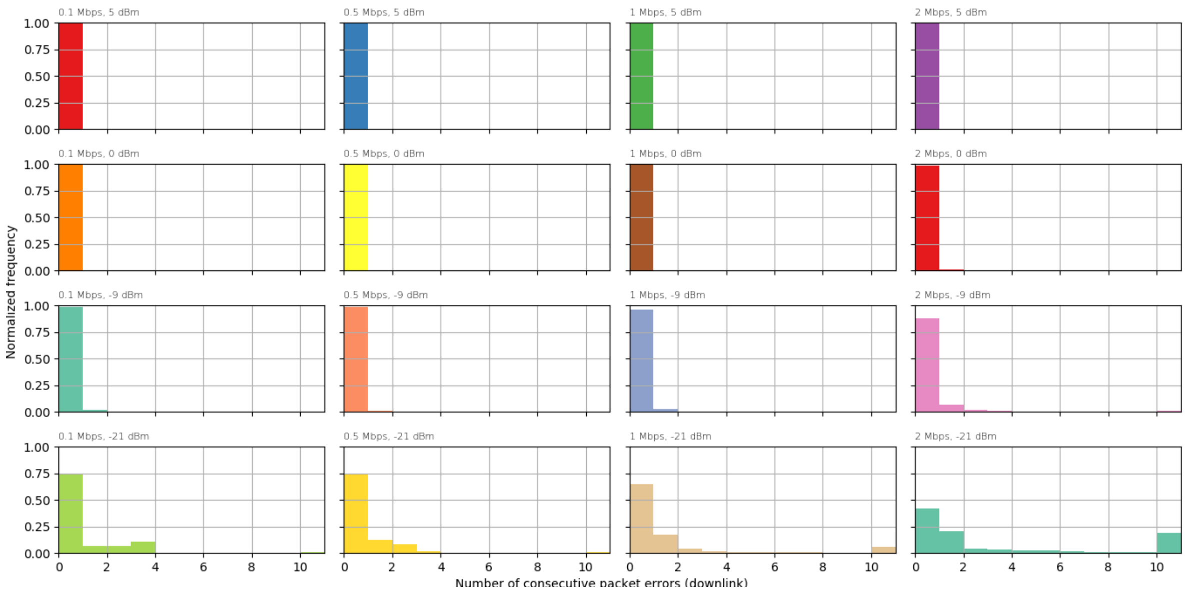
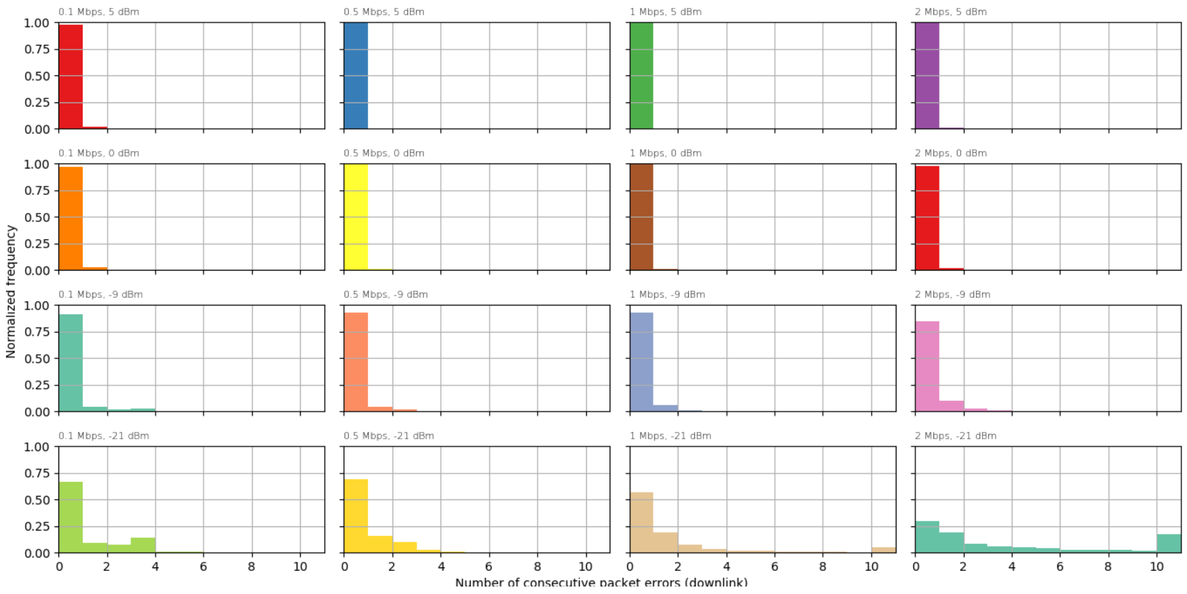


Figure B.5: Distribution of the downlink consecutive packet error count.

(a) Node 1



(b) Node 2



(c) Node 3

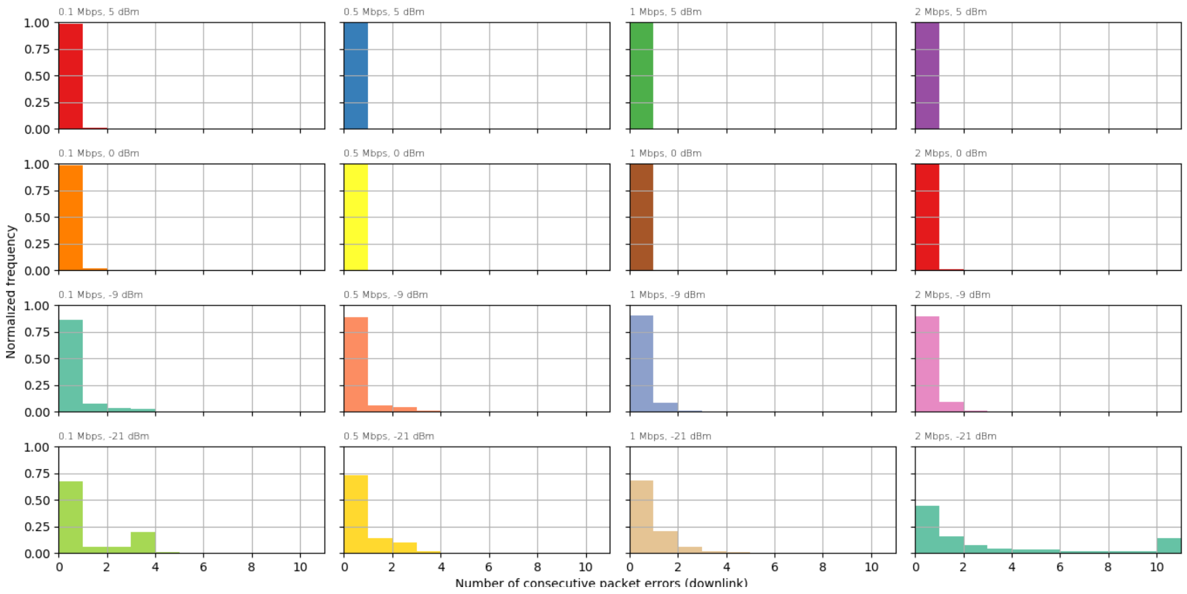
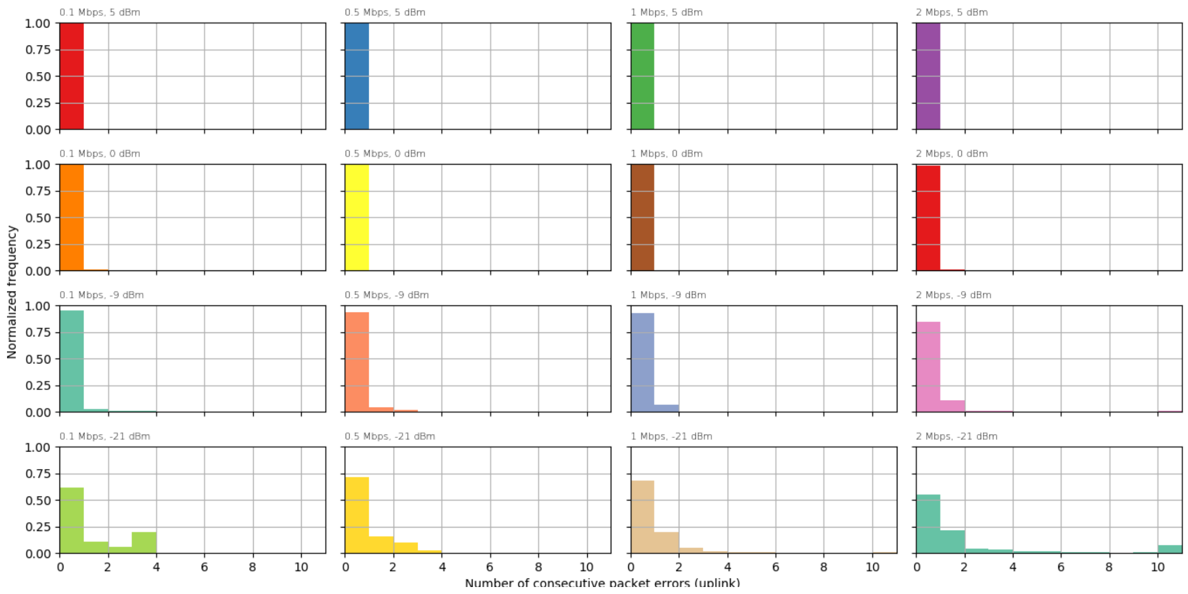
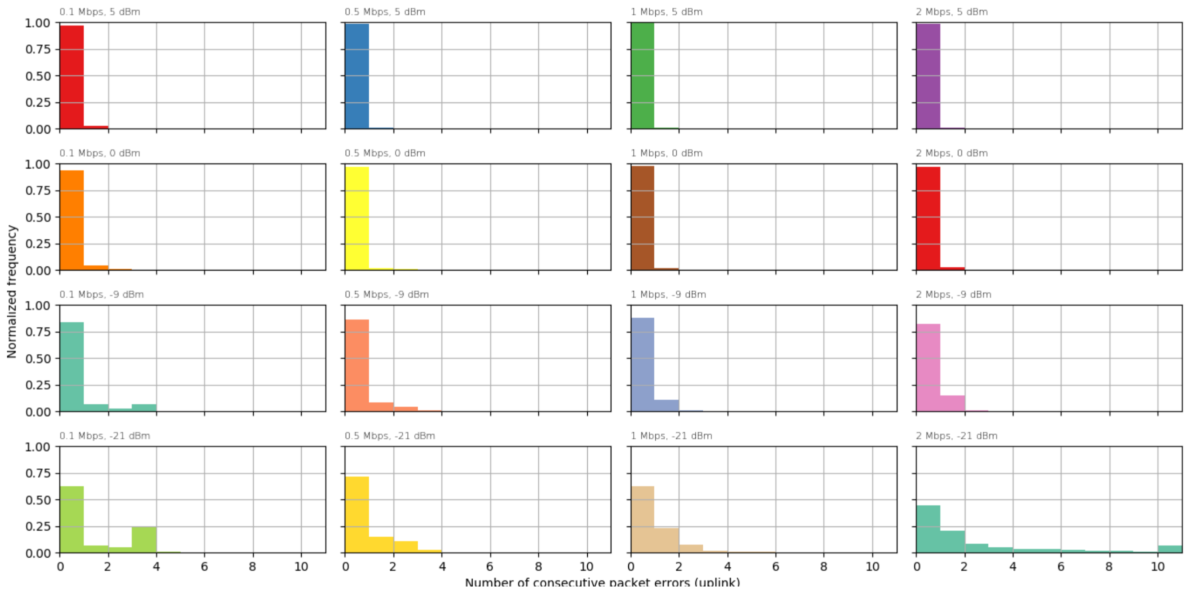


Figure B.6: Distribution of the uplink consecutive packet error count.

(a) Node 1



(b) Node 2



(c) Node 3

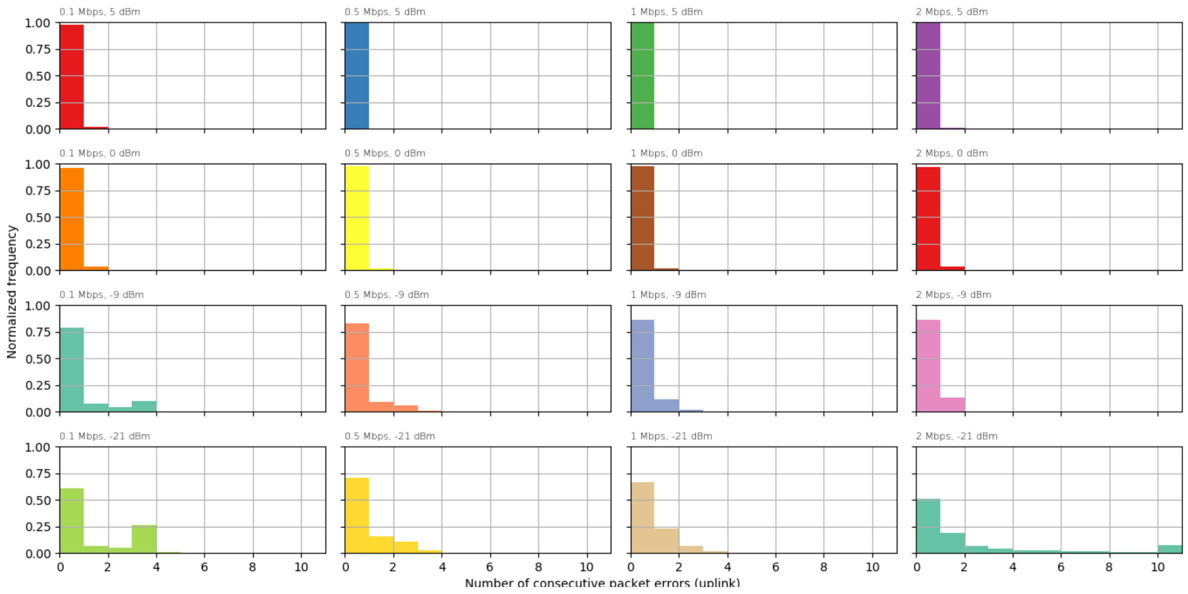
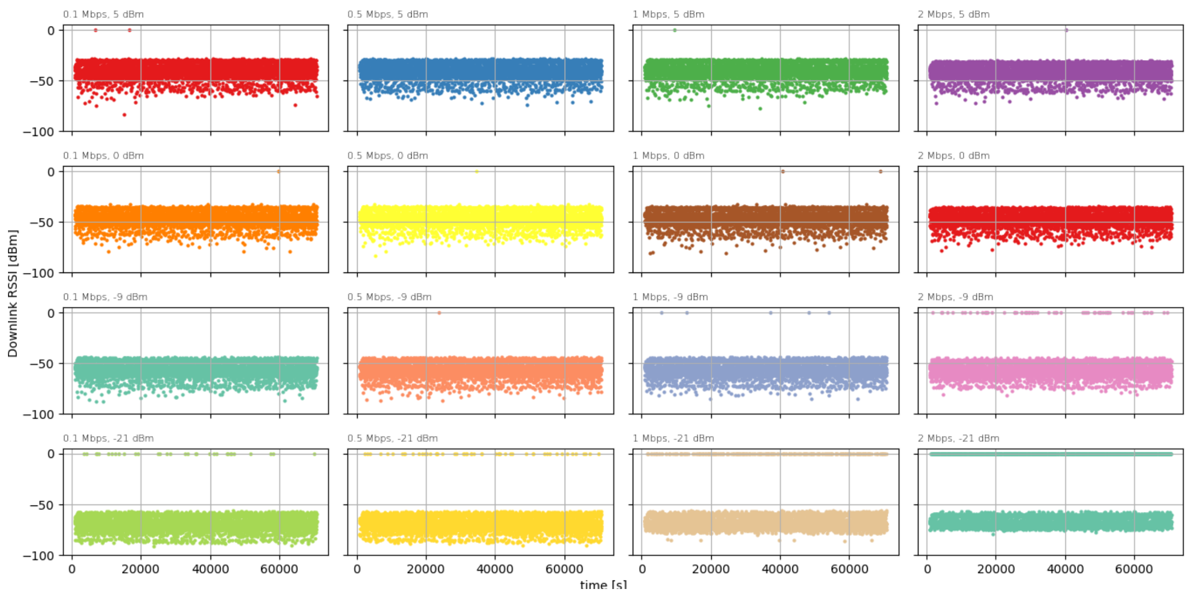
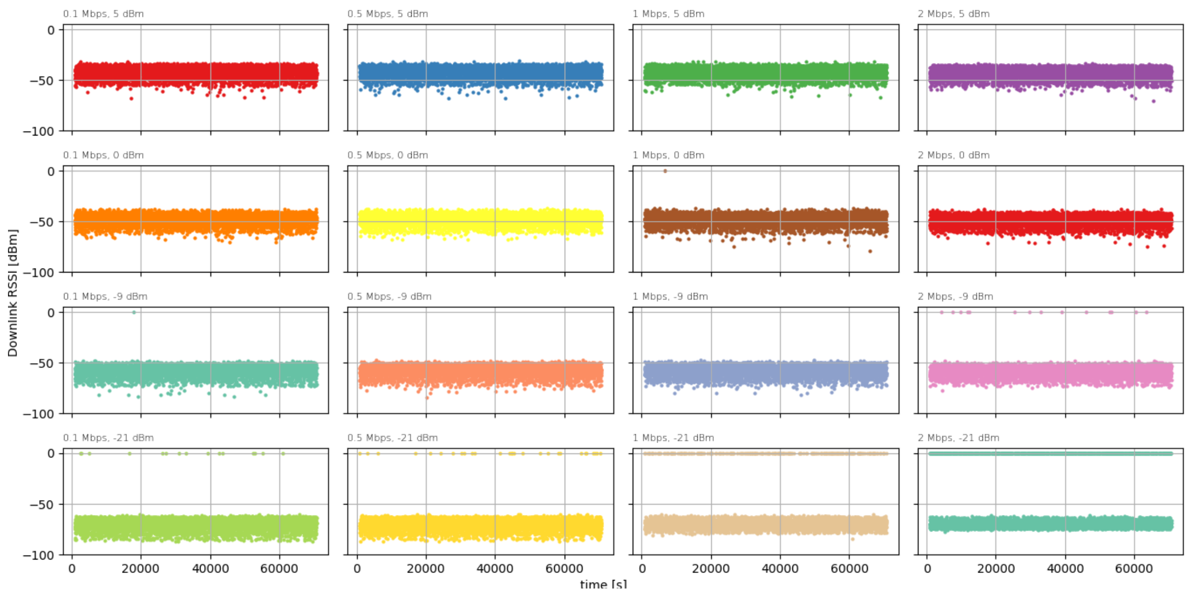


Figure B.7: Downlink RSSI.

(a) Node 1



(b) Node 2



(c) Node 3

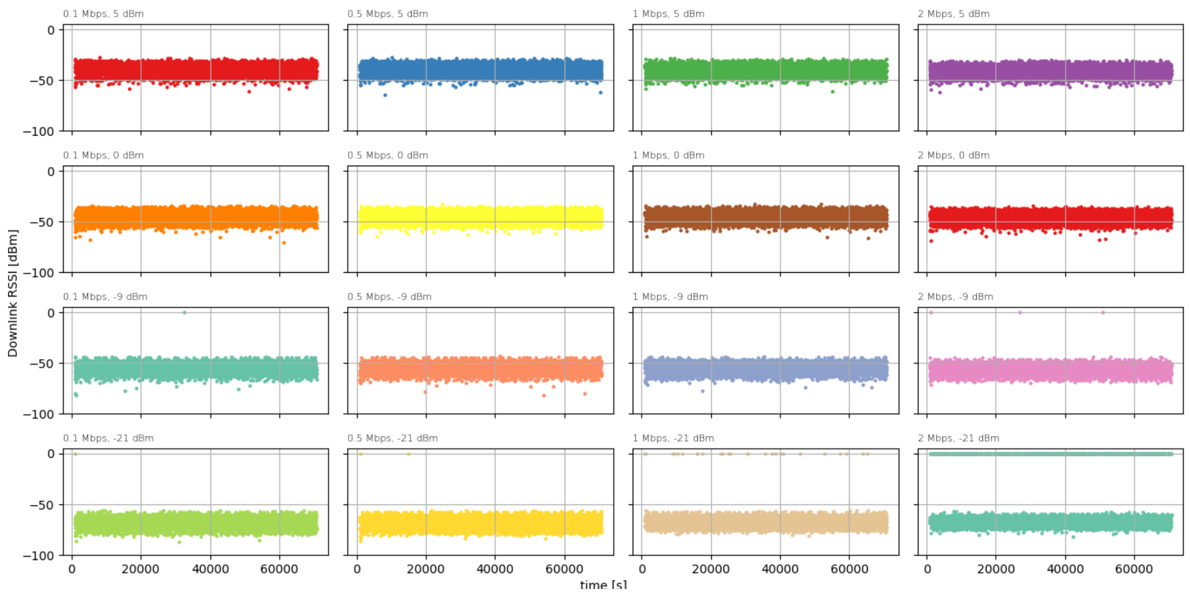
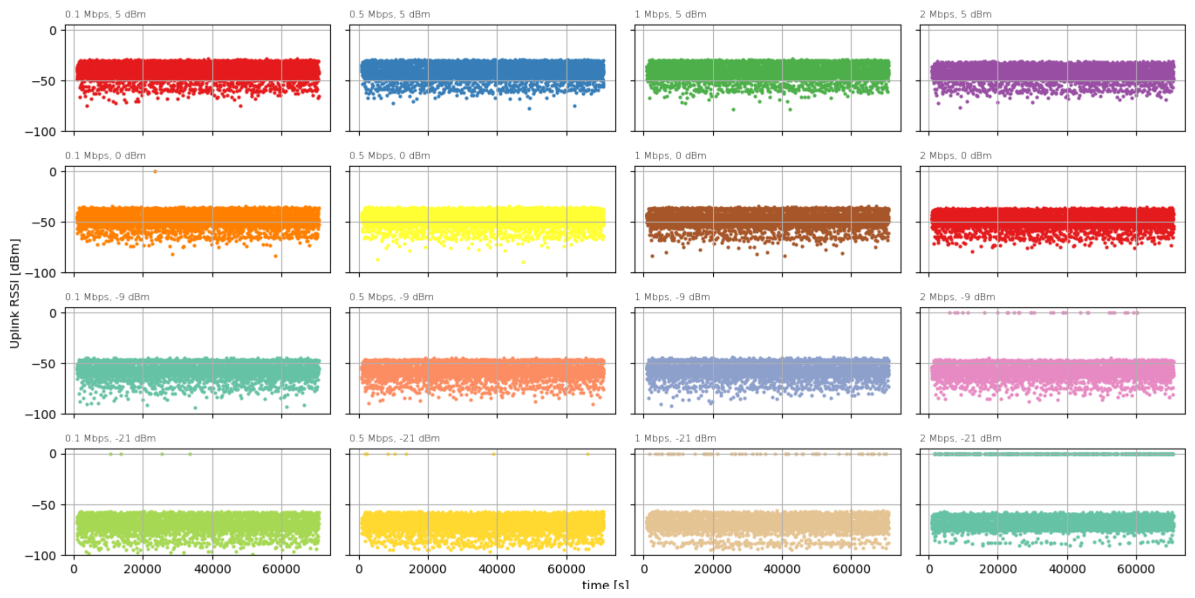
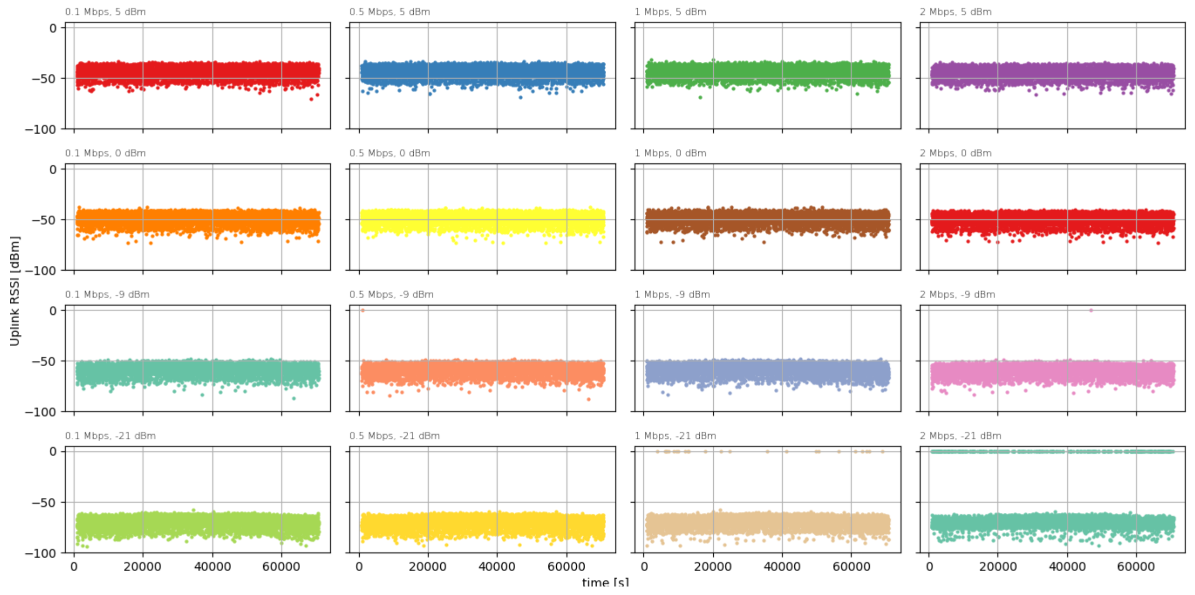


Figure B.8: Uplink RSSI.

(a) Node 1



(b) Node 2



(c) Node 3

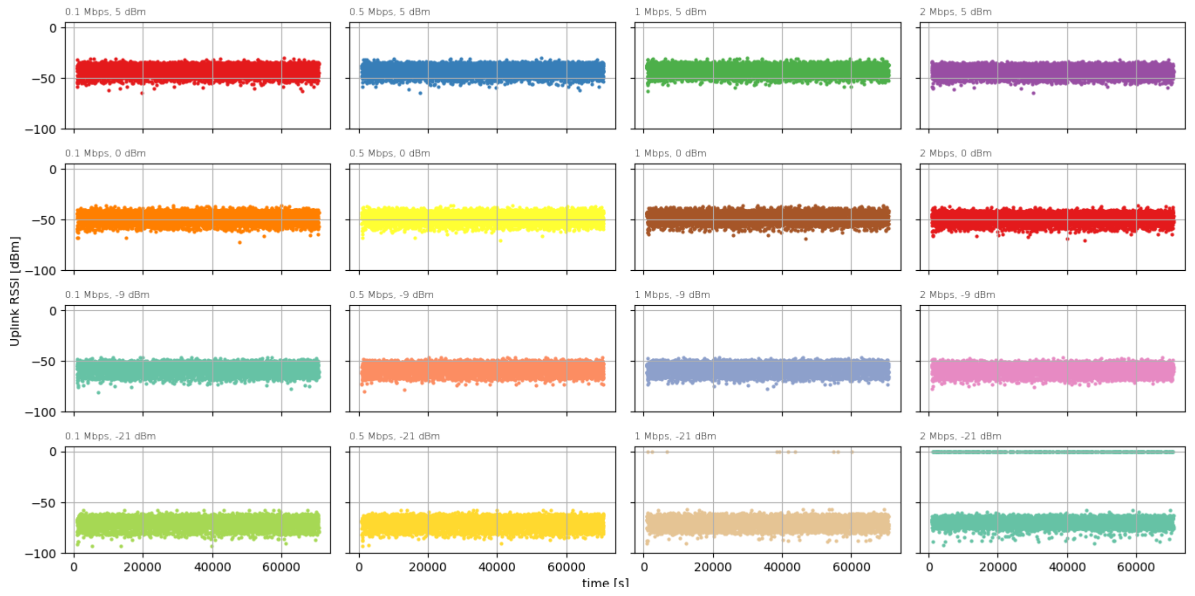
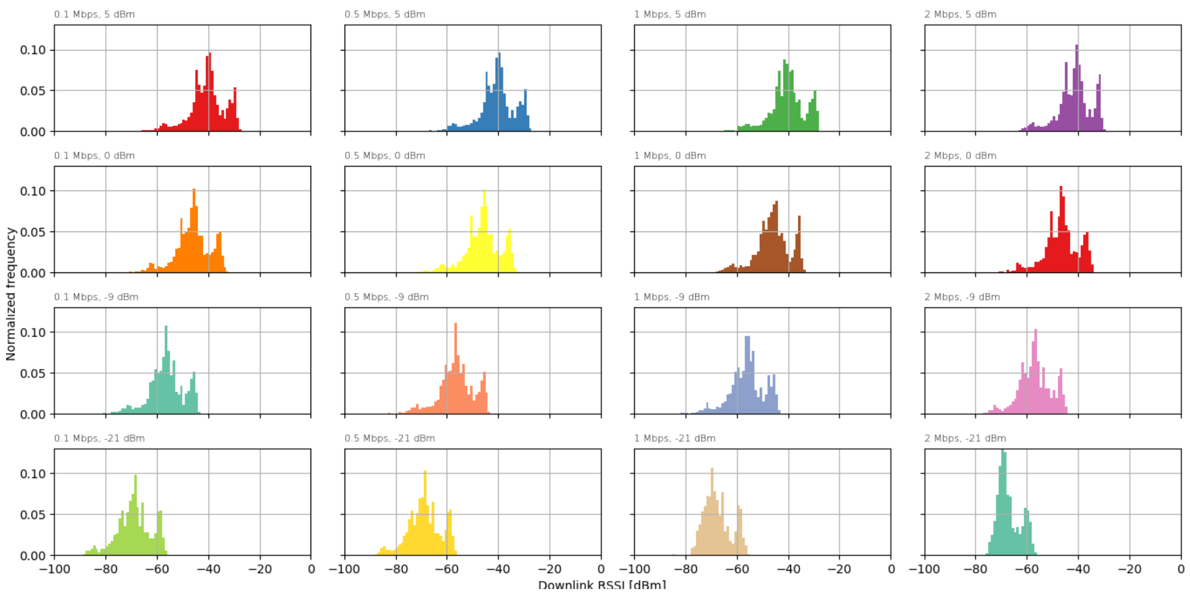
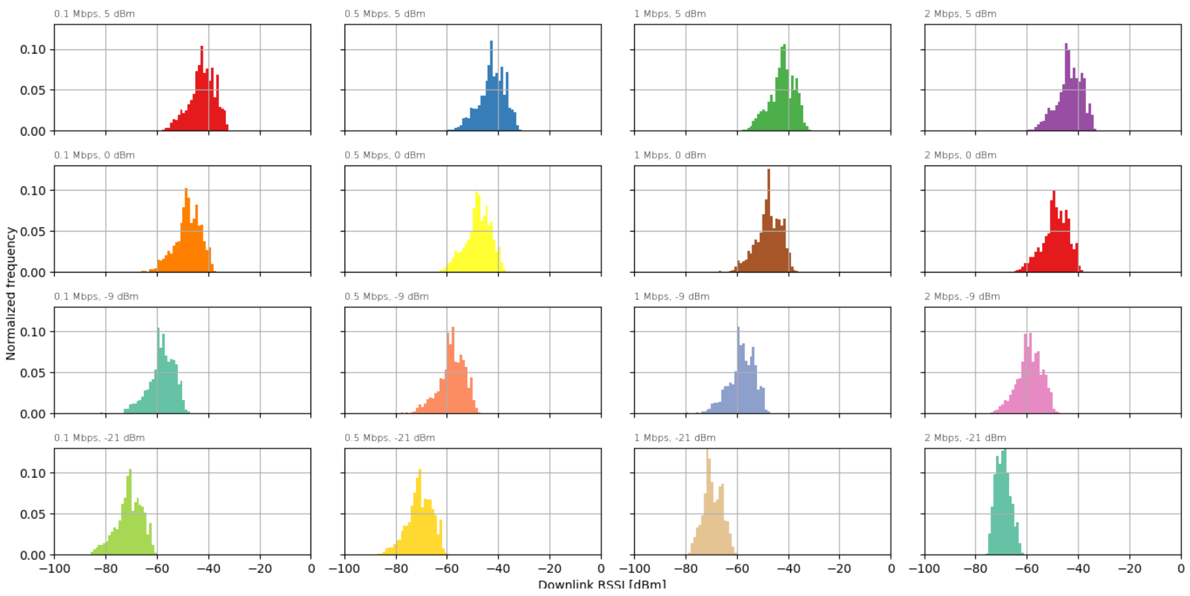


Figure B.9: Distribution of the downlink RSSI.

(a) Node 1



(b) Node 2



(c) Node 3

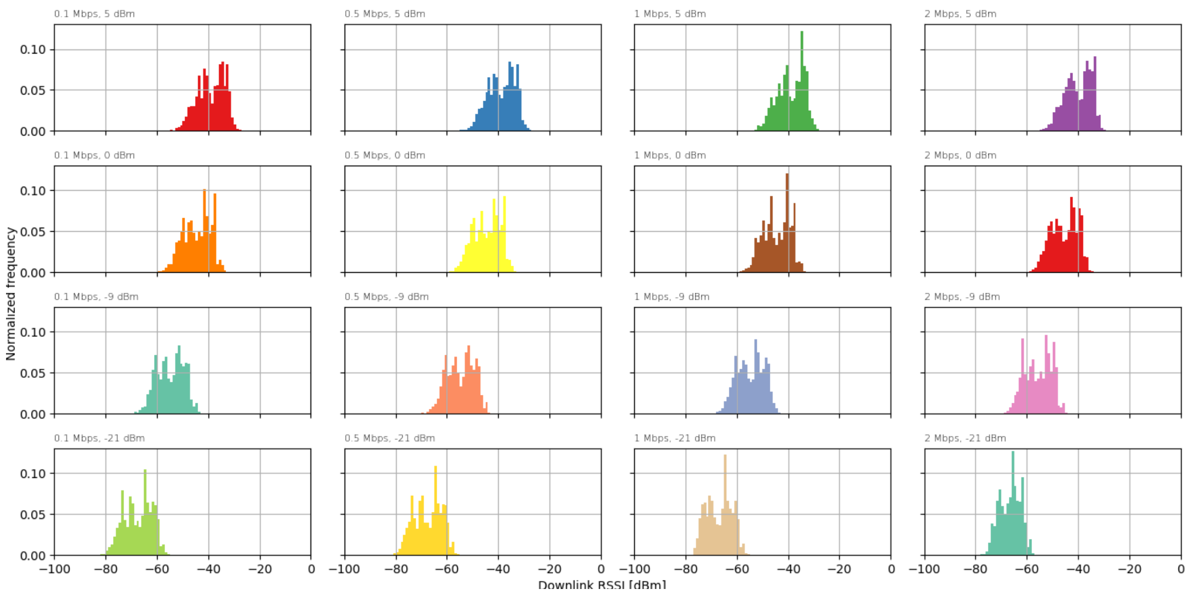
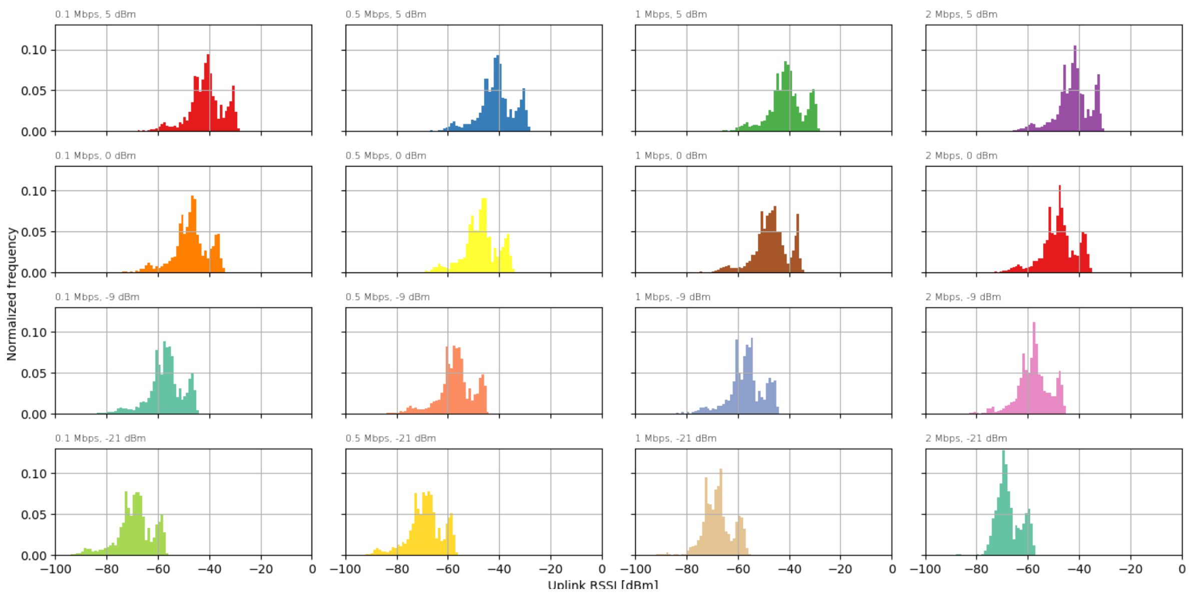
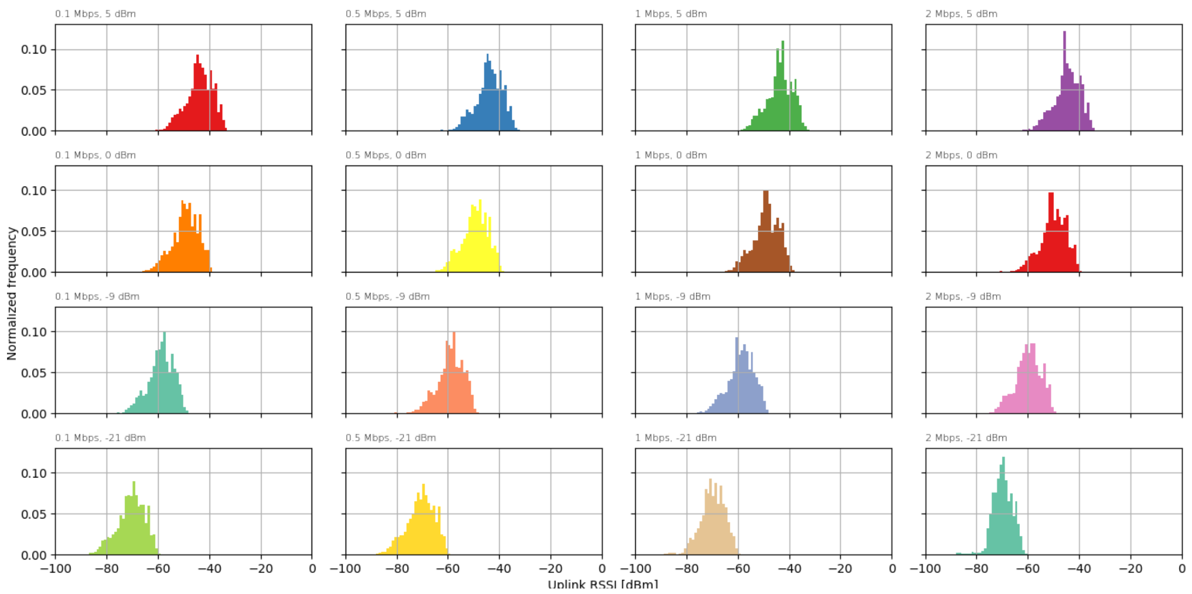


Figure B.10: Distribution of the uplink RSSI.

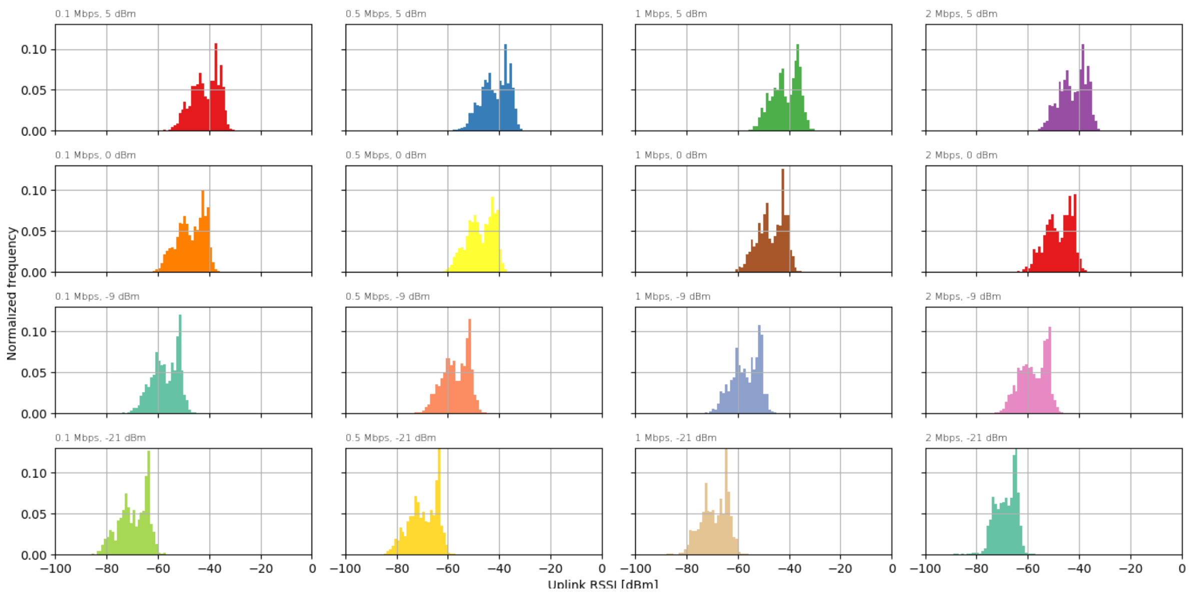
(a) Node 1



(b) Node 2



(c) Node 3



Bibliography

[1] U. Uyumaz and Technische Universiteit Eindhoven (TUE). Stan Ackermans Instituut. Software Technology (ST), *Wafer flow simulator visualizer*, PDEng rapport (Technische Universiteit Eindhoven, 2013).

[2] W. Stein, *Spare parts planning at ASML*, eSCF operations practices: insights from science (Technische Universiteit Eindhoven, 2010).

[3] P. Avitabile, *Experimental modal analysis–a simple non-mathematical overview,* Sound & Vibration (2001).

[4] M. Batel, *Operational modal analysis-another way of doing modal testing,* Sound and Vibration **36**, 22 (2002).

[5] A. B. Noel, A. Abdaoui, T. Elfouly, M. H. Ahmed, A. Badawy, and M. S. Shehata, *Structural health monitoring using wireless sensor networks: A comprehensive survey,* IEEE Communications Surveys Tutorials **19**, 1403 (2017).

[6] I. Ullah, N. Arbab, and W. Gul, *State of the art vibration analysis of electrical rotating machines,* Journal of Electrical Engineering **5**, 84 (2017).

[7] S. Kim, S. Pakzad, D. Culler, J. Demmel, G. Fenves, S. Glaser, and M. Turon, *Health monitoring of civil infrastructures using wireless sensor networks,* in *Proceedings of the 6th international conference on Information processing in sensor networks* (ACM, 2007) pp. 254–263.

[8] S. Jang, H. Jo, S. Cho, K. Mechitov, J. A. Rice, S.-H. Sim, H.-J. Jung, C. B. Yun, B. F. Spencer Jr, and G. Agha, *Structural health monitoring of a cable-stayed bridge using smart sensor technology: deployment and evaluation,* Smart Struct. Syst. (2010).

[9] Y. Yu, J. Wang, X. Mao, H. Liu, and L. Zhou, *Design of a wireless multi-radio-frequency channels inspection system for bridges,* International Journal of Distributed Sensor Networks **8**, 743673 (2012).

[10] D. Phanish, P. Garver, G. Matalkah, T. Landes, F. Shen, J. Dumond, R. Abler, D. Zhu, X. Dong, Y. Wang, *et al.*, *A wireless sensor network for monitoring the structural health of a football stadium,* in *2015 IEEE 2nd World Forum on Internet of Things (WF-IoT)* (IEEE, 2015) pp. 471–477.

[11] K. Chintalapudi, T. Fu, J. Paek, N. Kothari, S. Rangwala, J. Caffrey, R. Govindan, E. Johnson, and S. Masri, *Monitoring civil structures with a wireless sensor network,* IEEE Internet Computing **10**, 26 (2006).

[12] M. Ceriotti, L. Mottola, G. P. Picco, A. L. Murphy, S. Guna, M. Corra, M. Pozzi, D. Zonta, and P. Zanon, *Monitoring heritage buildings with wireless sensor networks: The torre aquila deployment,* in *Proceedings of the 2009 International Conference on Information Processing in Sensor Networks* (IEEE Computer Society, 2009) pp. 277–288.

[13] R. A. Swartz, J. P. Lynch, S. Zerbst, B. Sweetman, and R. Rolfes, *Structural monitoring of wind turbines using wireless sensor networks,* Smart structures and systems **6**, 183 (2010).

[14] A. Vogl, D. T. Wang, P. Storås, T. Bakke, M. M. Taklo, A. Thomson, and L. Balgård, *Design, process and characterisation of a high-performance vibration sensor for wireless condition monitoring,* Sensors and actuators a: physical **153**, 155 (2009).

[15] L. Hou and N. W. Bergmann, *Novel industrial wireless sensor networks for machine condition monitoring and fault diagnosis,* IEEE Transactions on Instrumentation and Measurement **61**, 2787 (2012).

[16] Q. Huang, B. Tang, and L. Deng, *Development of high synchronous acquisition accuracy wireless sensor network for machine vibration monitoring,* Measurement **66**, 35 (2015).

[17] K. Zhang and X. Yan, *A wireless sensor network for online identification of rotor blade crack,* Smart Materials and Structures **25**, 067001 (2016).

[18] R. McGinnis, N. Perkins, and K. King, *Reconstructing free-flight angular velocity from a miniaturized wireless accelerometer,* Journal of Applied Mechanics **79**, 041013 (2012).

[19] Z. Shen, C. Y. Tan, K. Yao, L. Zhang, and Y. F. Chen, *A miniaturized wireless accelerometer with micromachined piezoelectric sensing element,* Sensors and Actuators A: Physical **241**, 113 (2016).

[20] A. Sabato, C. Niezrecki, and G. Fortino, *Wireless MEMS-based accelerometer sensor boards for structural vibration monitoring: A review,* IEEE Sensors Journal **17**, 226 (2017).

[21] H. Jo, S.-H. Sim, T. Nagayama, and B. Spencer Jr, *Development and application of high-sensitivity wireless smart sensors for decentralized stochastic modal identification,* Journal of Engineering Mechanics **138**, 683 (2011).

[22] M. Chae, H. Yoo, J. Kim, and M. Cho, *Development of a wireless sensor network system for suspension bridge health monitoring,* Automation in Construction **21**, 237 (2012).

[23] X. Hu, B. Wang, and H. Ji, *A wireless sensor network-based structural health monitoring system for highway bridges,* Computer-Aided Civil and Infrastructure Engineering **28**, 193 (2013).

[24] A. Sabato and M. Q. Feng, *Feasibility of frequency-modulated wireless transmission for a multi-purpose mems-based accelerometer,* Sensors **14**, 16563 (2014).

[25] M. D. Kohler, S. Hao, N. Mishra, R. Govinda, and R. Nigbor, *Shakenet: A portable wireless sensor network for instrumenting large civil structures,* (2015).

[26] L. Zhu, Y. Fu, R. Chow, B. F. Spencer, J. W. Park, and K. Mechitov, *Development of a high-sensitivity wireless accelerometer for structural health monitoring,* Sensors **18**, 262 (2018).

[27] B. Spencer Jr, J.-W. Park, K. Mechitov, H. Jo, and G. Agha, *Next generation wireless smart sensors toward sustainable civil infrastructure,* Procedia engineering **171**, 5 (2017).

[28] T. Watteyne, S. Lanzisera, A. Mehta, and K. S. Pister, *Mitigating multipath fading through channel hopping in wireless sensor networks,* in *2010 IEEE International Conference on Communications* (IEEE, 2010) pp. 1–5.

[29] A. Elsts, S. Duquennoy, X. Fafoutis, G. Oikonomou, R. Piechocki, and I. Craddock, *Microsecond-accuracy time synchronization using the ieee 802.15. 4 tsch protocol,* in *2016 IEEE 41st Conference on Local Computer Networks Workshops (LCN Workshops)* (IEEE, 2016) pp. 156–164.