Master *Radiation and its Effects on MicroElectronics and Photonics Technologies* (RADMEP)

Feasibility Evaluation and Performance Analysis of Wireless Mesh Networks in particle accelerator under Radiation

Presented by

Luca Yago Wolinsky Mancini

and defended at

University Jean Monnet

September 9th, 2025

Academic Supervisor: Prof. Dr. Sylvain Girard, University Jean Monnet

Host Supervisors: Alessandro Zimmaro, CERN

Dr. Salvatore Danzeca, CERN

Jury Committee:

Master Thesis Report

Feasibility Evaluation and Performance Analysis of Wireless Mesh Networks in particle accelerator under Radiation

Author:
Luca Yago Wolinsky Mancini

Academic Supervisor:
Prof. Dr. Sylvain Girard, University Jean Monnet
Host Supervisors:
Alessandro Zimmaro, CERN
Dr. Salvatore Danzeca, CERN

September 2025

Acknowledgements

I would like to express my gratitude to the RADMEP team for giving me the opportunity to be part of this great experience.

I am also grateful to the four universities involved in the program: University of Jyväskylä, KU Leuven, Université Jean Monnet, and University of Montpellier, as well as to their professors, for their valuable guidance, knowledge, and support.

My special thanks go to CERN, in particular to my supervisors Alessandro Zimmaro and Salvatore Danzeca, for their guidance and help during my internship, which made the development of this thesis possible.

I would also like to thank all my RADMEP friends, with whom I began this journey two years ago and with whom I am now about to complete it.

Finally, I am deeply grateful to my family for their unconditional love and support.

Disclaimer

I did not use generative AI assistance tools during the research/writing process of my thesis, except for mere language assistance. The text, code, and images in this thesis are my own (unless otherwise specified). Generative AI has only been used in accordance with the RADMEP guidelines and appropriate references have been added. I have reviewed and edited the content as needed and I take full responsibility for the content of the thesis.

Abstract

In the industrial and academic sectors, wireless mesh networks are gaining increasing interest due to the advantages they offer over traditional wireless networks, such as star or point-to-point technologies. Wireless mesh networks are characterized by a decentralized topology, in which each node can communicate directly with the other nodes within the network. Additionally, wireless mesh networks feature self-configuration and self-healing capabilities. This approach provides enhanced reliability, resilience, and scalability, overcoming limitations of non-mesh networks, such as single points of failure, limited coverage, and flexibility.

These features make wireless mesh networks suitable for dynamic and complex environments, especially relevant for applications in particle accelerator facilities, like the Future Circular Collider, or the space environment, where long distances and their radiation nature are significant challenges. Although mesh networks have found various uses today, their performance under radiation exposure has not yet been thoroughly evaluated. In this work, the dependability of a wireless mesh network was studied within the CHARM facility, exploring and analyzing a network configuration based on a wireless ad hoc network to ensure optimal performance. Beyond network validation, this study highlighted how CHARM provides an ideal environment for testing various network setups, thanks to its large and uniform radiation area, unlike other facilities where testing multiple nodes would be more complicated.

Contents

List of Abbreviations

ACK	Acknowledgement.
AODV	Ad Hoc On-Demand Distance Vector.
BAN	Body Area Network.
BW	Bandwidth.
CERN	Conseil Européen pour la Recherche Nucléaire.
CHARM	CERN Highly-Accelerated Mixed Field.
COTS	Commercial Off-The-Shelf.
CR	Code Rate.
CSS	Chirp Spread Spectrum.
DARPA	Defense Advanced Research Projects Agency.
DD	Displacement Damage.
DSDV	Destination Sequence Distance Vector.
DSR	Dynamic Source Routing.
DTN	Delay-Tolerant Network.
ETX	Expected Transmission Count.
FANET	Flying Ad hoc Network.
FCC	Future Circular Collider.
FSK	Frequency Shift Keying.
GloMo	Global Mobile Information Systems.
IEL	Ionizing Energy Loss.
IETF	Internet Engineering Task Force.
IoT	Internet of Things.
ITS	Intelligent Transportation Systems.
LAN	Local Area Network.
LET	Linear Energy Transfer.
LHC	Large Hadron Collider.

LoRa	Long Range.
MAC	Medium Access Control.
MAN	Metropolitan Area Network.
MANET	Mobile Ad hoc Network.
NIEL	Non-Ionizing Energy Loss.
NLOS	Non-Line-of-Sight.
NRL	Normalized Routing Load.
OLSR	Optimized Link State Routing.
PDR	Packet Delivery Ratio.
PKA	Primary knock-on atom.
PLR	Packet Loss Ratio.
PRnet	Packet Radio Network.
PS	Proton Synchrotron.
RDREQ	Route Discovery REQuest.
RDRES	Route Discovery RESponse.
RSSI	Received Signal Strength Indicator.
RTT	Round-Trip-Time.
SEB	Single Event Burnout.
SEE	Single Event Effect.
SEFI	Single Event Functional Interrupt.
SEGR	Single Event Gate Rupture.
SEL	Single Event Latch-up.
SET	Single Event Transient.
SEU	Single Event Upset.
SF	Spreading factor.
SNR	Signal-to-Noise Ratio.
SURAN	Survivable Adaptive Radio Networks.
TID	Total Ionizing Dose.
TORA	Temporally-Ordered Routing Algorithm.
TP	Throughput.

UAV	Unmanned Aerial Vehicles.
VANET	Vehicular Ad hoc Network.
WAN	Wide Area Network.
WANET	Wireless Ad hoc Network.
WMN	Wireless Mesh Network.
WRP	Wireless Routing Protocol.
ZHLS	Zone-Based Hierarchical Link State.
ZRP	Zone Routing Protocol.

List of Figures

List of Tables

1 Introduction

1.1 Motivation

Particle accelerators and the space environment are examples of radiation-harsh environments that demand reliable, resilient, robust, and scalable communication systems. In this context, wired infrastructures and traditional wireless topologies show considerable limitations in terms of deployment complexity and scalability. Wireless Mesh Networks (WMNs), mainly based on Wireless Ad Hoc Networks (WANETs), appear as a potential alternative. The decentralized approach of WANETs, in addition to its multihop communication capabilities and self-organizing, self-healing nature, can enhance not only reliability and robustness but also enable its deployment within highly dynamic scenarios.

However, the implementation of WMNs in radiation environments remains unexplored. The devices forming a wireless mesh network are usually based on electronic devices, and therefore, as is widely studied, ionizing radiation can degrade and disrupt electronic components, especially in commercial off-the-shelf (COTS) devices. The effects of radiation on the devices affect the mesh network performance, in terms of topology, routing, packet delivery ratio, and latency, among other parameters.

This work is embedded within the AstroMesh project at CERN, which aims to develop and test radiation-tolerant wireless platforms for applications in the Future Circular Collider (FCC) and inter-satellite communication. This thesis addresses the impact of radiation effects on electronic devices in the context of wireless communication systems.

1.2 Objective

This thesis focuses on the development and validation of a WMN architecture able to operate in a radiation environment using COTS devices. The main goal is to demonstrate the feasibility of deploying a WMN in harsh environments such as particle accelerators and space applications.

The general objective can be subdivided into the following specific objectives:

- **O1.** Demonstrate a proof of concept for a wireless mesh network architecture based on existing libraries, using COTS microcontrollers and radio transceivers, and evaluate the performance and fault tolerance of it under radiation exposure.
- **O2.** Develop, evaluate, and validate an experimental test setup for wireless mesh networks under radiation and define metrics for assessing the impact of radiation on network-level performance.
- **O3.** Evaluate the impact of mitigation schemes at component and network-level to improve the reliability of the network.

1.3 Thesis structure

The document of this thesis is divided into five sections. In Section 1, a brief introduction is provided, enhancing the motivation of the work and objectives. The principles of wireless mesh networks, focusing on WANETs, as well as the fundamentals of radiation effects on electronics, are explained in Section 2. Section 3 describes the experimental methodology, including the evaluation platform specifications, test design, tested mesh network implementation, and data analysis workflow. The results of the tested mesh network, followed by a comprehensive interpretation and discussion, are exposed in Section 4. Finally, Section 5 provides the conclusions and future work of this thesis.

2 Literature Review

Wireless Mesh Networks (WMNs) have emerged as a robust and scalable communication architecture, characterized by decentralization, self-organization, self-healing, dynamic and adaptive routing, and multihopping capabilities. WMN features make them a suitable technology to be deployed not only on surface and aerospace environments, but also in harsh environments such as outer space, nuclear plants, or particle accelerators. However, when electronic devices, acting as nodes of the network, are deployed in high ionizing radiation environments, the robustness of WMNs is even more challenged than in traditional surface use cases. This chapter provides a review of the state of the art in WMN and radiation effects on electronics. The first part of the chapter presents an overview of the principles, use cases, features and routing methods of WMN, which is followed by a section describing foundational knowledge on particle interaction with matter, and radiation induced degradation mechanisms on electronics, such as Total Ionizing Dose (TID), Single Event Effect (SEE) and Displacement Damage (DD). The radiation environments of interest in this work, such as the space radiation and CERN's accelerator environment, are presented at the end of this chapter.

2.1 Wireless Mesh Network

Wireless communication technologies have experienced an exponential growth in recent decades, driven by their flexibility, scalability, and extensive applicability. The next generation of wireless networks is enhancing the performance of the networks to provide better services, allowing a great adaptability to a wide range of applications. Due to the increased interest in wireless technologies, research on WMNs has increased. WMNs offer advantages such as their ability to dynamically self-organize and self-configure, relatively low setup, installation, and maintenance costs, and increased coverage and reliability [1] [2] [3].

Wireless network communication relies on wireless channels for communication between devices. Therefore, the spectrum range and radio frequency used for the communication are key parameters. Signals at higher frequencies suffer more attenuation with distance compared to lower frequencies, but can support higher data rates or throughput.

The architecture of WMN can be represented by layers using the Open Systems Interconnection (OSI) model, as shown in Figure 1 [4] [5]. Each layer addresses specific needs and challenges of such networks. The physical layer is responsible for the transmission and reception of information through modulation and coding protocols. The data link layer provides error control within the network. The medium access control (MAC) layer regulates the access to the shared channel, aiming to minimize collisions. The network

layer is responsible for finding the optimal routing within the network. The transport layer handles delays and lost packets. And the application layer deals not only with disconnections and reconnections but varying delays and lost packets. This work will focus mainly on the network layer, which is crucial for the communication and routing within nodes in a mesh network.

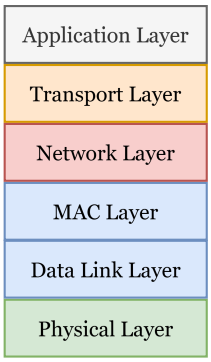


Figure 1. OSI layered model adapted for WMNs.

Wireless networks can be categorized into two classes, as depicted in Figure 2: infrastructure - based wireless networks and infrastructure-less wireless networks [2] [6]. The infrastructure-based wireless network relies on access point devices. This topology connects the wireless devices to a gateway to access an existing wired network. Typical examples of this infrastructure-based wireless network can be found in offices, hospitals, or homes, in which the end nodes have access to the Internet via an access point. In contrast, an infrastructure-less wireless network, also known as a Wireless Ad hoc Network (WANET), consists of a dynamic network topology in which devices can connect with each other using single or multihop wireless links without the need for infrastructure such as access points or wired networks. Therefore, WANETs are naturally a wireless mesh topology. When the devices are mobile, the network is referred to as a Mobile Ad hoc Network (MANET).

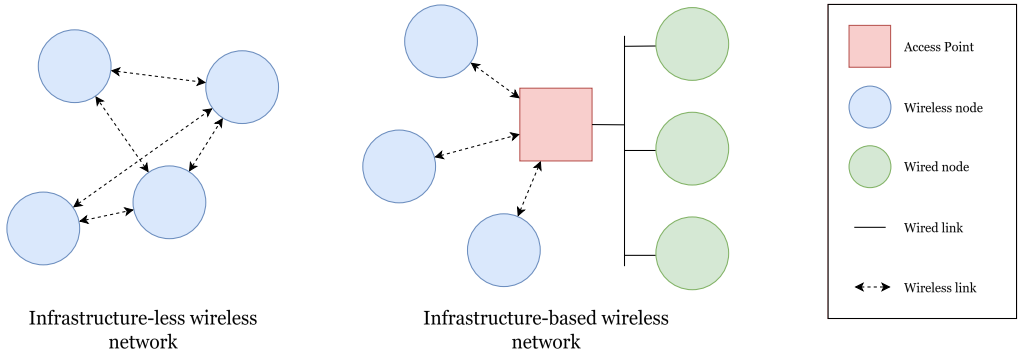


Figure 2. Comparison between an infrastructure - based wireless network and an infrastructure - less wireless network.

In the literature, the term Wireless Mesh Network can refer to a network architecture consisting mainly of two parts: mesh backbone and mesh clients [1]. The mesh backbone is formed by stationary mesh routers interconnected with single or multihop wireless links. Moreover, some of these nodes act as an access point by having a wired connection with a wired network. And the mesh clients comprise a network of the end nodes, which can communicate with any of the nodes within this subnetwork by single or multihop links. WMNs can be hybrid, combining a backbone and client mesh network, but also the backbone or client networks can work independently. Figure 3 shows an example of a hybrid WMN. For this work, we are going to focus mainly on independent client networks, which can be considered as a WANET or MANET.

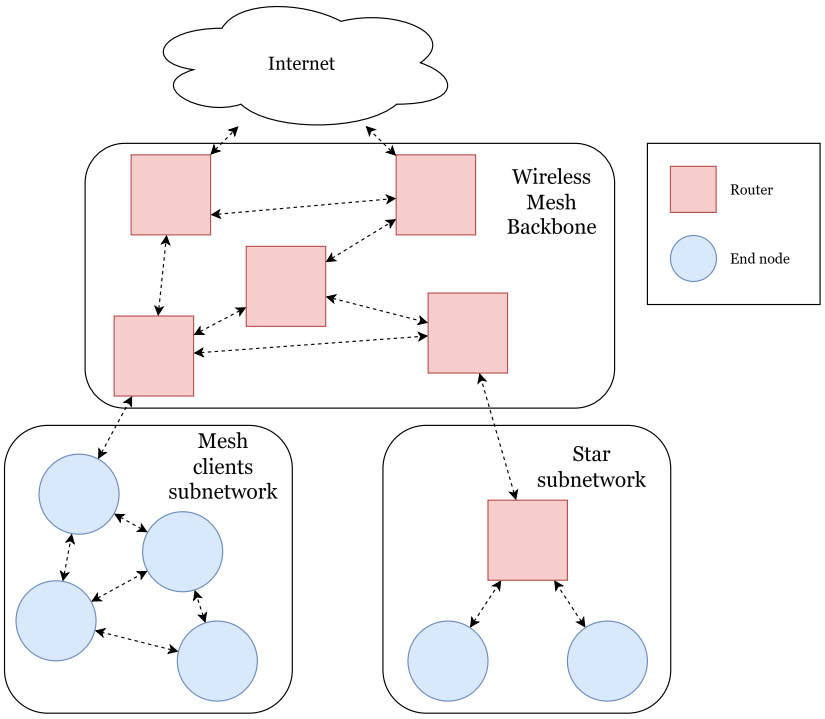


Figure 3. Hybrid WMN topology formed by a wireless mesh backbone, mesh clients subnetwork, and star-topology subnetwork.

Within the extensive category of WANET topologies, other specialized architectures exist [7]. For example, a Vehicular Ad Hoc Network (VANET) consists of nodes embedded in vehicles, resulting in higher mobility of nodes. Although VANETs and MANETs share many characteristics, they may differ at the physical, MAC, and application layers to fulfill the specific requirements of vehicular communication [8]. A Flying Ad hoc Network (FANET) is a similar approach to a VANET, but specifically for drones or unmanned aerial vehicles (UAVs), where the degree of mobility is very high and the nodes operate in a three-dimensional space, instead of the typical two-dimensional space of MANETs [8]. Another related concept is Delay-Tolerant Networks (DTNs), which is a

type of opportunistic routing that assumes data transmission is not delay-sensitive, and consequently, a store-carry-forward approach can be implemented [7]. In this context, the nodes are capable of storing data until an opportunity to forward the packet to the correct destination is found. Moreover, a higher mobility of nodes can be beneficial for a DTN, as it enhances the number of encounters between nodes.

Mesh topologies are an alternative to the most traditional centralized, star or tree-based, or point-to-multipoint topologies, where some of the devices are intrinsically more important than others as they provide single links to some subsets of devices in the network, which makes it a hierarchically organized topology [3]. Figure 4 shows a schematic comparison between a mesh network topology and a star topology. Although the hierarchical-centralized topologies are well established and highly standardized, in the case of mesh network topologies, no common standards have been developed yet for this emerging technology [3]. For instance, in [3] a survey on different wireless communication technologies (such as IEEE 802.11-based, Bluetooth, IEEE 802.15.4-based, and LoRa) supporting mesh network capabilities is provided.

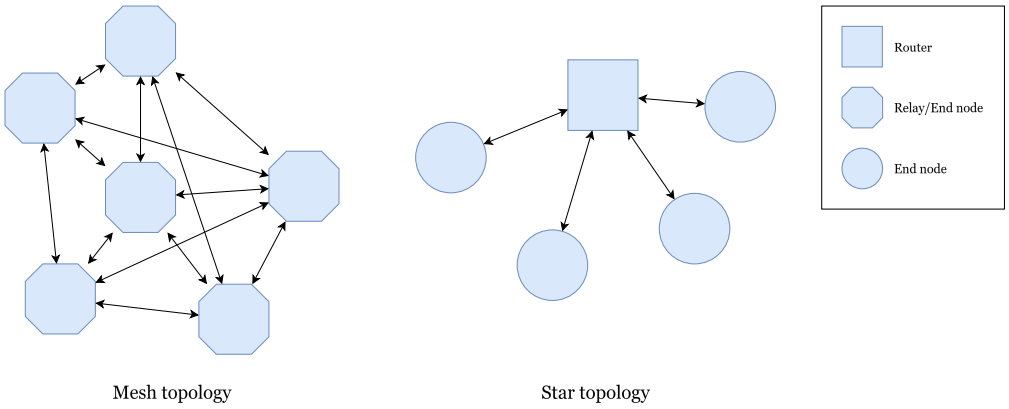


Figure 4. Comparison between a mesh network topology and a star network topology. In a mesh network, every node in the network can act as an end node, transmitting or receiving packets, and as a relay node, forwarding packets. In a star topology, each end node is connected to a central node, which is responsible for managing the flow within the network.

Mesh topologies consist of nodes supported by electronic devices that are directly connected to every node in the network, allowing communication between nodes via several paths or routes, thanks to the cooperation within nodes. Therefore, each node in the network coordinates with each other to find the most efficient route between a source and destination node. For this, each node not only participates as a host but as a router, allowing the relaying of packets [3].

The feature of relaying packets provides the network the capability to establish multihop links, which can increase the coverage, as a source node can communicate with a destination node that is not in its visibility range [3].

2.1.1 Modulation schemes for WMNs

In literature, the study of low power wireless technologies for WMNs has gained interest. In this context, frequency-based modulation techniques are often used due to their increased robustness against noise and interference, allowing for low power and long range links. Frequency modulation also allows for frequency hopping and multi-channel schemes. The most common modulation technologies implemented for WMNs are FSK (Frequency Shift Keying) and LoRa modulation [9].

FSK is a digital modulation scheme in which a variation in frequency of the carrier signal represents different symbols. Meaning that the signal can be demodulated at the receiver by detecting the frequency shift. The key parameters of FSK are the carrier frequency, frequency deviation, bit rate, modulation index, bandwidth, and the detection scheme, which can be coherent or non-coherent. An example of a FSK signal transmitted over time is shown in Figure 5.

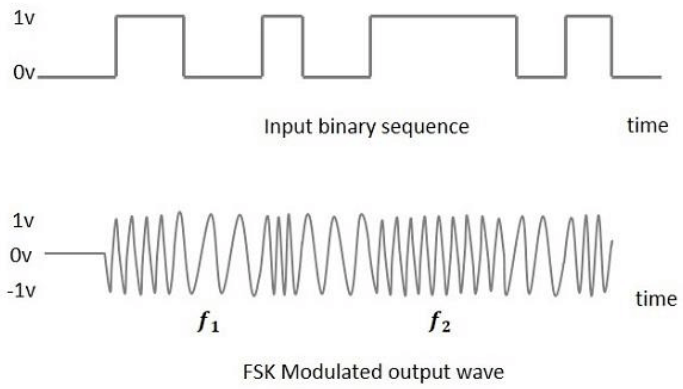


Figure 5. FSK modulated waveform. Figure from [10].

LoRa systems have been extensively used in WANETs and MANETs, as they feature energy efficiency as a result of their low-power nature. LoRa is a long-range wireless communication system based on Chirp Spread Spectrum (CSS) radio modulation technique [11]. LoRa CSS modulation technique consists of encoding the information with a linear variation of frequency over time, known as frequency chirps. The main parameters of LoRa are Bandwidth (BW), Spreading Factor (SF), and Code Rate (CR). Moreover, LoRa features cyclic error coding, allowing for forward error correction. An example of a LoRa signal transmitted over time is shown in Figure 6.

LoRa features higher performance in long-range and low-power applications, compared to traditional modulation schemes such as FSK. However, FSK schemes show higher data rates.

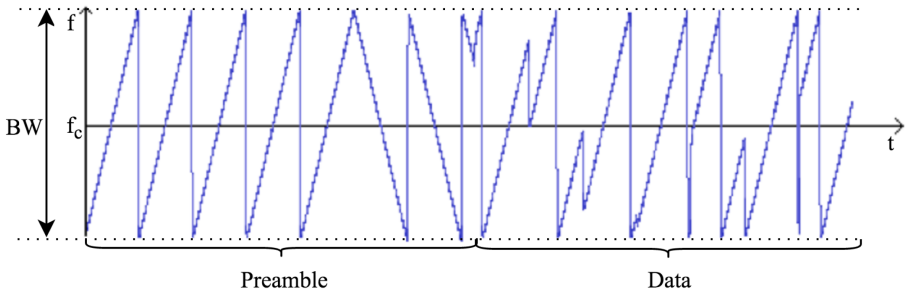


Figure 6. Frequency variation over time of an example LoRa transmitted signal. Where f_c is the central frequency of the wireless channel. Figure from [11].

2.1.2 MANET

MANETs consist of a dynamic and non-hierarchical topology where the nodes, which are able to move in the space, act as relay or routers forwarding packets within the network, allowing a multihop ad hoc communication [6] [12]. The main feature of a MANET is its infrastructureless nature, where no special centralized node or structure is needed to support the wireless communication.

The goal of MANETs is to provide a robust and efficient operation by implementing routing functionalities within the mobile nodes of the network. In the case of MANETs, all the nodes in the network are allowed to move and act as an end node as well as a router, allowing for a dynamic network topology. Each node autonomously implements the discovery of new neighbors and route maintenance functionalities in order to face topology changes [6].

MANETs are characterized by their self-forming, self-configuring, and self-healing capabilities. Each node is able to discover new nodes as well as create, maintain, and modify routes, automatically without any centralized infrastructure. Moreover, the nature of the mobile nodes, with the displacement, departure, and arrivals of the devices, modifies the network topology in an unpredictable way. MANET nodes are commonly powered by a battery, limiting their energy consumption. Moreover, the bandwidth is also constrained, as the nodes operate in a shared wireless channel, competing with each other to access the medium [6].

2.1.3 History

The first MANETs were developed in the 1970s [13], driven by the Department of Defense of the United States, with the aim of providing a fast deployable communication system for military applications. In this scope, in 1973, the Defense Advanced Research Projects Agency (DARPA) developed Packet Radio Network (PRnet), a system supporting multihop wireless link capabilities between devices. In 1983, the latter was implemented as part of the Survivable Adaptive Radio Networks (SURAN) project, which aimed for small, low-cost, and low-power devices, with efficient protocols to enhance the

scalability of the ad hoc network. In 1994, DARPA presented further improvement by launching the Global Mobile Information Systems (GloMo) program, whose aim was to develop new wireless ad hoc networking technologies compatible with Internet technology. In the 1990s, due to the exponential growth of personal wireless communication devices, the IEEE 802.11 subcommittee adopted the term ”ad hoc networks”, and both the industry and the research community showed a big interest in MANETs, which allowed to start exploring the feasibility of this type of network in other areas and fields of application. And in 1997, the Internet Engineering Task Force (IETF) established the Mobile Ad Hoc Network Working Group to standardize the MANET routing protocol functionalities.

2.1.4 MANET classifications

MANETs can be classified in terms of communication style, topology, node configuration, and network size [6].

Classification according to the communication Ad hoc network communication can be either single-hop or multihop (Figure 7).

Single-hop communication is the simplest type of link between nodes. This link is established within nodes that are within their coverage area. An entire network can be fully single-hop if all the nodes remain within the range of each other. This means the nodes can also be mobile as long as they are able to communicate directly.

Multihop communication is commonly studied in literature as it allows Non-Line-of-Sight (NLOS) communication between nodes, enhancing the coverage area. In this class, the communication link between two nodes is not direct, and therefore, the traffic between two end nodes has to be forwarded by other intermediate nodes. In this case, the intrinsic nature of the mobile nodes results in a dynamic topology with continuous modifications. To handle the dynamic nature of the network, routing protocols have to be adaptive to the network conditions.

Classification according to the topology Ad hoc networks can be classified according to the network topology. Three main classes of networks are identified: flat, hierarchical, and aggregate networks.

Flat networks are characterized by equal nodes, in which every node in the network has the same responsibilities and functions, therefore, control messages, for instance, for route creation, are transmitted by each node in the network. This approach is suitable for highly dynamic networks, although the performance still degrades with an increase in the number of nodes.

Hierarchical networks consist of independent clusters, with a master node and some slave nodes. These clusters are linked to each other. For each cluster, the master node

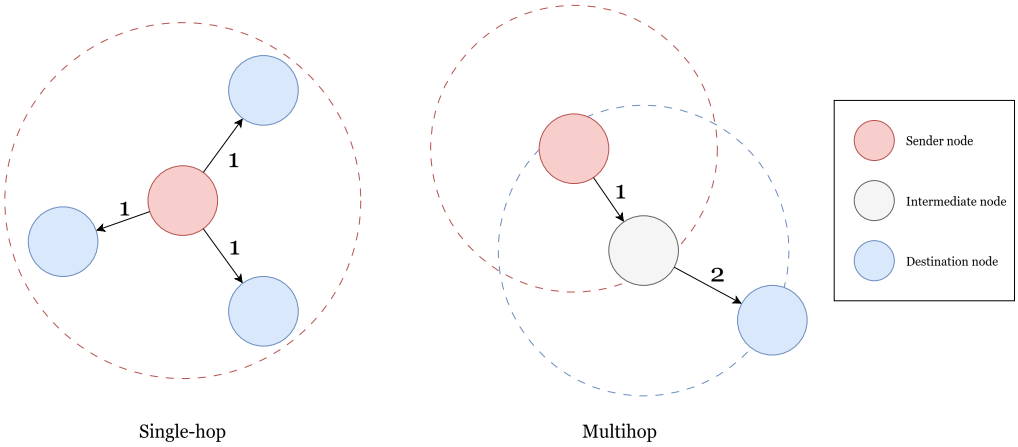


Figure 7. Schematic of the comparison between single-hop and multihop communication.

administrates the cluster and is responsible for transmitting the data to the other clusters. While the slave nodes communicate within their cluster nodes and with other cluster nodes with the help of the master node. This type of network is more suitable for low mobility nodes, allowing better scalability.

In aggregate ad hoc networks, the network is divided into a set of zones, each identified by an identification (ID) number.

Classification according to the node configuration Ad hoc networks can be classified in terms of the hardware configuration of the nodes: homogeneous networks and heterogeneous networks.

In a homogeneous network, all the nodes have the same hardware characteristics, such as processor, memory, and peripheral devices. While in a heterogeneous network, the nodes possess different hardware configurations, meaning that not all nodes may support the same functionalities.

Classification according to the coverage area Ad hoc networks can be classified depending on their coverage area: body area network (BAN), personal area network (PAN), local area network (LAN), metropolitan area network (MAN), and wide area network (WAN).

2.1.5 Routing Protocols

Wireless ad hoc networks feature automatic network connectivity with self-organized, self-configured, and self-healing capabilities. This can only be achieved with routing protocols that handle the dynamic topology of the network and meet the requirements of a specific application or case scenario, for instance, in terms of power consumption, bandwidth, and data rates. Some routing protocols are standardized by the IETF.

Ad hoc routing protocols can be categorized into three main types: table-driven or proactive, on-demand-driven or reactive, and hybrid.

Table-driven routing protocols, also known as proactive routing protocols, ensure that each node in the network has a continuously updated routing table that is able to represent a part, or the complete network topology [3]. To achieve having up-to-date routing information, the nodes within the network periodically exchange the network topology information. These types of protocols allow to have a route already available when a node needs to send a packet to a destination node. Proactive routing protocols are generally used in static or low mobility node scenarios [6], with a low number of total nodes. However, the constant routing information transmissions not only increase the overhead but may also be undesired or ineffective if the topology changes are infrequent. Some examples of table-driven routing protocols are: Optimized Link State Routing (OLSR), Destination Sequence Distance Vector (DSDV), and Wireless Routing Protocol (WRP).

On-demand routing protocols, also known as reactive routing protocols, are protocols characterized by creating routes only when needed [3]. This means that a new route is only found when a node wants to transmit a packet and the current route is not available, and the new route is maintained until needed. Reactive protocols generally consist of a route discovery mechanism based on flooding or broadcasting techniques and a route maintenance procedure to detect broken links. Usually, each node stores the information of the next-hop to reach a destination node. The main advantage of on-demand routing protocols is the reduced communication overhead, which leads to lower power consumption of the nodes and lower traffic over the scarce wireless channel, compared to proactive protocols [6]. Therefore, reactive protocols can perform better in higher mobility and a greater number of nodes scenarios (considering limited resources). Some examples of on-demand routing protocols are: Ad Hoc On-Demand Distance Vector (AODV), Dynamic Source Routing (DSR), and Temporally-Ordered Routing Algorithm (TORA).

Hybrid routing protocols aim to combine some of the features of reactive and proactive protocols. Some typical hybrid routing protocols are: Zone Routing Protocol (ZRP) and Zone-Based Hierarchical Link State (ZHLS).

Routing protocols use routing metrics in order to evaluate and select the best path through the network. The routing metric generally assigns a value to each route, reflecting the cost of transmitting a packet through it, based on a metric parameter or optimization objective [2]. Some of the most used optimization objectives routing metrics try to achieve are: minimize delay, maximize the probability of data delivery, maximize throughput (either path throughput or network throughput), reduce energy consumption, and distribution of the traffic load [2]. There are mainly five categories of routing

metrics: topology-based, signal strength-based, active probing-based, mobility-aware, and energy-aware.

Topology-based routing metrics are characterized by their simplicity, as they consider parameters such as the number of neighbors of the nodes or the number of hops needed to reach a destination node. A typical example of route selection is considering the shorter path in terms of the number of hops. Although this type of routing metric is simple, it might not be optimal as it does not reflect the actual condition of the paths, for example, leading to higher packet loss probabilities in lower hop routes compared to higher hop routes.

Signal strength-based metrics use the signal strength as an indicator of the link quality. This type of metric assumes that higher signal strength, or at least above a threshold, correlates with higher probabilities of successful packet reception.

Active probing-based metrics essentially use probe packets to estimate the successful packet delivery probabilities. However, some challenges arise from the use and treatment of the probe packets in the network [2]. One of the most common metrics in this category is the Expected Transmission Count (ETX).

Energy-aware metrics aim to minimize the power consumption of the nodes in the network, and mobility-aware metrics aim to select routes with higher expected lifetime in order to reduce the changes in topology and routing overhead.

2.1.6 Performance Metrics

To evaluate the performance of an ad hoc network, it is necessary to define and calculate relevant metrics to quantify key parameters of the network. The most common performance evaluation metrics for WMN are related to packet delivery probabilities, communication delay, throughput, overhead and control packets, and energy consumption [14] [15] [16].

- **Packet delivery statistics**: the Packet Delivery Ratio (PDR) is generally used to compare the number of successful packet deliveries with the total packets sent, calculated as in Equation 1.

$$PDR = \frac{\text{Packets}_{\text{ successfully transmitted}}}{\text{Packets}_{\text{ total transmitted}}} \tag{1}$$

  Another similar metric is the Packet Loss Ratio (PLR), which is the complement of the PDR, as $PLR = 1 - PDR$.

- **Delay Metrics**: the time to transmit a packet from a source node to a destination node is typically calculated, known as end-to-end delay. This metric can be related to the bits, bytes, or packets sent, among others. The delay in these networks can be affected by several parameters, such as the route discovery, processing delay, and

other factors. Alternatively, the Round-Trip-Time (RTT) is commonly measured in reliable communications working with acknowledgment (ACK) packets, which corresponds to the time it takes the packet to go from source to node, plus the time it spends the acknowledgment from destination back to source.

- **Throughput (TP) and Network Load**: it is the rate of successfully delivered packets in an interval of time. It can be calculated as in Equation 2. Moreover, the network load can be measured to evaluate the total number of packets carried by the entire network over time.

$$TP = \frac{\text{sent packets}}{\text{time interval}} \tag{2}$$

- **Routing Overhead**: the routing overhead is a widely used metric; this metric shows the amount of routing and control packets sent, which can be compared with the amount of data packets transmitted to obtain a Normalized Routing Load (NRL). The NRL provides information on the amount of overhead packets, including route discovery and route maintenance processes, required for correct network behavior.

- **Link Quality Metrics**: The communication links between nodes in the network can be assessed with metrics such as the Signal-to-Noise Ratio (SNR), which compares the strength of the desired signal with the background noise, and the Received Signal Strength Indicator (RSSI), which indicates the power of the received signal.

2.1.7 Applications

MANETs' features provide them with great adaptability to different environments, situations, and applications. As ad hoc networking can be applied without any type of infrastructure, its flexibility enables automatic network communication between devices. MANETs can be developed in a wide range of applications, ranging from large-scale to small networks, or from mobile and highly dynamic to static networks [6].

Military applications have been the primary use of MANETs since their origins. Communications network and information sharing are key factors for successful tactical operations. Usually, military missions occur where communications infrastructure is scarce or nonexistent [17]. Therefore, MANETs can provide connectivity not only within a group of soldiers but also with vehicles, headquarters, or bases, or deployed units, allowing fast and reliable communication.

Disaster recovery communications [1] [6] is another common scenario where MANETs are used. In this case, a fast and effective way of communication is required to treat victims in a disaster area. Therefore, a MANET can be developed with the rescue team's

devices, such as laptops, phones, or PDAs, to have a fast deployable network.

The Internet of Things (IoT) has enhanced the study and exploration of ad hoc networking as it shows potential to integrate MANETs into different scenarios such as smart cities, smart health, smart agriculture, or smart industries applications, among others [3]. Each specific application has its own requirements for which the ad hoc network should be adapted.

Another potential application area of wireless ad hoc networks is found in VANETs [18], where Intelligent Transportation Systems (ITS) can be developed. ITS collects applications such as urban awareness, driver assistance, traffic management, and accident avoidance.

Several industrial companies have developed mesh network systems tailored to specific application domains. Some projects can be highlighted, such as NeoMesh from NeoCortec, which targets mainly sensor-type applications, and MeshMerize, which focuses on mesh networks for industry, supporting mobile robots, drones, tunnels, and construction connectivity. Moreover, some open source projects can be found, such as Meshtastic, which uses LoRa radios as a long-range off-grid communication platform, mainly for areas without communications infrastructure.

Moreover, some specific projects can be highlighted. For instance, NASA has tested the Starling CubeSat Warm Project [19], which is a MANET for cubesats relying on a proactive routing protocol known as Better Approach to Mobile Ad hoc Networking (BATMAN), demonstrating successful discovery and establishment of the network, allowing up to three multihop communication links. In [20], a wireless sensor network of 150 devices was deployed on Great Duck Island, United States, for habitat monitoring applications. In [21], a wireless sensor network was deployed in an area of $480000\ m^2$ using 19 LoRa devices, demonstrating a 88.5% of PDR with a mesh topology, compared with the PDR of 58.7% using a star topology. Other mesh network projects using LoRa have been published for different applications, such as animal tracking [22], swarm of drones [23], or IoT for low-energy consumption [24], among others.

2.1.8 Challenges in wireless ad hoc networks

Wireless ad hoc networks feature some interesting advantages compared to more traditional networks, as explained in this chapter. However, this type of network also shows some constraints or challenges that should be addressed. In terms of the devices forming the WANET, generally they have limited battery power, a small CPU and memory, short bandwidth, and low physical protection [25]. Some of the main challenges in ad hoc networks that can be highlighted are scalability, multicasting, quality of service, energy efficiency, and security [2].

Scalability is one of the main challenges in ad hoc networks, and can be defined as the ability of the network to maintain an acceptable level of service while introducing a

high number of nodes in the network [4]. In terms of scalability, mesh networks perform better than infrastructure-based networks, but it is still limited. When more nodes are deployed in the network, more traffic needs to be handled over the scarce shared channel, and consequently, there will be more interference, causing a degradation in performance and throughput. Some possible solutions to overcome scalability issues are explained in [2], including adding few hierarchical levels in the network architecture instead of maintaining a flat network [25] [26], using multiple radio channels instead of a single frequency, using directional antennas instead of omnidirectional antennas and choosing the proper routing protocol for the specific needs of the scenario, highlighting the possibility to implement hybrid routing protocols, where a proactive style is used for nearby or often used nodes, and a reactive behavior is adopted for further nodes. It is important to take into account that these features need to be supported by the overall system, including the physical layer, MAC design, and network layer, among others.

The energy is limited, as generally each device acting as a node in the network is battery powered [4]. This supports the idea of energy-efficient devices, mainly achieved by reducing the power used by the radio transceiver, for instance, by sending the transceiver to sleep mode when unused. Another approach can be to implement energy-efficient routing algorithms, for example, by prioritizing routes with shorter hops instead of longer hops to reduce the required power.

Security in ad hoc networks is a main challenge because, due to the intrinsic nature of WANETs, neither the nodes nor their information is secured from threats. The shared medium and lack of central coordination arise four key security problems in wireless ad hoc networks, such as the authentication of devices, the secure key establishment among the authenticated nodes, the secure routing of packets, and the secure storage of data [25]. These security vulnerabilities make WANETs susceptible to digital or cyber attacks [6] [4]. Some of the most common attacks over ad hoc networks are: eavesdropping, denial of service attack, routing attack, black hole attack, wormhole attack, man in the middle attack, or jamming [27] [28].

In addition to the intrinsic challenges of wireless ad hoc networks, the deployment of it under harsh radiation environments will introduce further degradation to the electronic devices as well as more unpredictability to the network topology. This is an additional challenge that needs to be addressed with more robust designs in network protocols and hardware.

Wireless ad hoc mesh networks feature great adaptability to several case scenarios, making them a great technology to be explored and studied under harsh environments such as radiation environments. WANETs demonstrate a great potential to be used for inter-satellite communication as well as in tunnels under radiation environments such as the Large Hadron Collider (LHC) or the Future Circular Collider (FCC) at CERN.

WANET's performance has not been widely evaluated under radiation exposure. Therefore, in the next sections, a brief explanation of radiation environments and radiation effects on electronics is provided to better understand the context and feasibility of deploying WANETs in these scenarios.

2.2 Radiation Effects on Electronics

WMNs offer great potential for deployment in harsh environments, such as in space or particle accelerator facilities. As discussed in Section 2.1, WMNs consist of nodes supported by electronic devices that can communicate in a decentralized topology. These electronics can suffer from radiation-induced effects, resulting in transient or permanent damage. This subsection provides an overview of the foundational physics of particle interaction with matter, focusing on the radiation effects on electronic devices. Moreover, this subsection introduces the main characteristics of the radiation environment in space and at CERN's accelerator facilities.

2.2.1 Particle interactions with matter

When a particle interacts with matter, ionizing radiation effects may occur. Ionizing radiation is the result of particles interacting with a target material, which detach electrons from the target atoms, producing electron-hole pairs. The different particle-matter interaction depends on the type of incident particles. The incident particles can be grouped in photons, neutrons, and charged particles, and their interactions are described in Figure 8 [29] [30] [31].

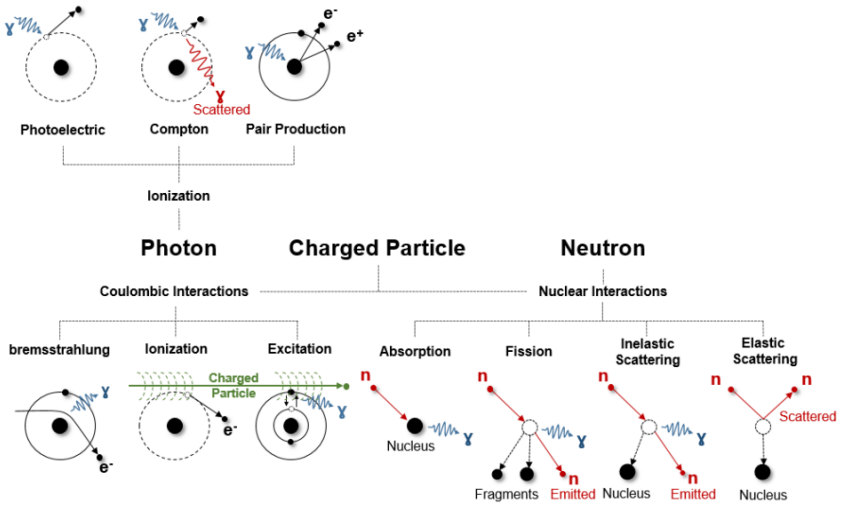


Figure 8. Particle interaction mechanisms. Figure obtained from [31].

Photons, due to their absence of electrical charge, cannot produce elastic collisions. However, three types of ionizing interactions can occur with matter:

- **Photoelectric effect**: it is the process in which the photon interacts with an electron with a similar binding energy to that of the incident particle, and consequently, it releases the electron.
- **Compton effect**: it is the process in which the photon, with a higher energy than the binding energy of the electron, collides with it, partially transferring its energy and releasing the electron. The remaining energy of the photon results in a scattering and wavelength change of the photon.
- **Pair production**: it is the effect in which the photon traverses close to the nucleus of the atom with a high enough energy to create an electron-positron pair.

As shown in Figure 9, there are different dominant regions for each form of interaction of photons with matter, depending on the photon energy and atomic number (Z) of the target material atoms.

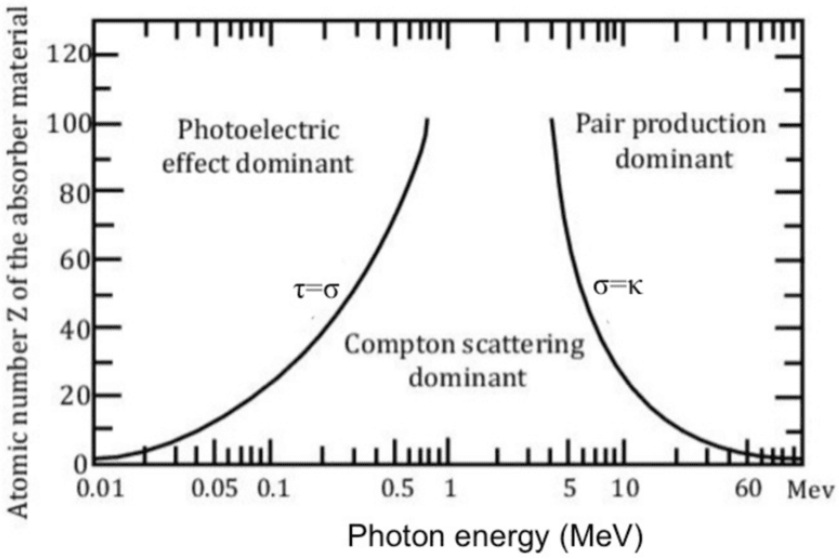


Figure 9. Dominant regions for photon interaction with matter mechanisms: photoelectric effect, compton effect, and pair production. Figure obtained from [29].

Neutrons can only interact with matter through nuclear interactions, as they do not have charge. There are four main types of nuclear interactions:

- **Absorption**: the incident neutron hits the nucleus, increasing its energy and taking it to a higher state. When it comes back to its original state, a photon is emitted.
- **Elastic scattering**: the incident particle hits the nucleus, transferring some energy. Both the incident particle and nucleus get scattered.
- **Inelastic scattering**: the incident particle that hits the nucleus gets absorbed. Then the nucleus gets scattered and emits a lower energy neutron, leaving it in

an excited state. When the nucleus comes back to its original state, the energy is released by photons.

- **Fission**: the nucleus absorbs the incident particle, and gets split into smaller fragments, known as fission fragments. Then, neutrons and photons are released.

Charged particles, such as pions ($\pi^{\pm}$), protons ($p^{\pm}$), kaons ($K^{\pm}$), muons ($\mu^{\pm}$), taus ($\tau^{\pm}$), and electrons ($e^{\pm}$), can interact with matter by Coulombic interaction and nuclear interaction. Nevertheless, Coulombic interactions are the predominant mechanism for charged particles, and their types are:

- **Bremsstrahlung effect**: the charged particle is deflected by the electric field of the nucleus, losing some of its energy. The energy lost by the charged particle is then emitted as a photon. This effect is mainly caused by light charged particles such as electrons.
- **Ionization**: it is the process in which a charged particle travels close to an orbital electron, with higher energy than the binding energy of the electron, and the electron gets ejected. This effect is mainly caused by heavily charged particles.
- **Excitation**: it is the process in which a charged particle travels close to an orbital electron, with lower energy than the binding energy of the electron. Therefore, the electron absorbs the energy of the charged particle and gets excited, moving to a higher energy state. When the electron comes back to its original energy state, it emits a photon. This effect is mainly caused by heavily charged particles.

These effects can be caused when the primary particle interacts with matter, known as direct ionization, but also can be caused by secondary particles produced by this first interaction, known as indirect ionization.

The particle-matter interactions described previously can be produced by a primary particle or through secondary particles generated by prior interactions. Typically, primary particle interactions are more energetic than secondary interactions.

Every particle hitting matter suffers a loss of energy. The energy loss can be defined as Ionizng Energy Loss (IEL), due to ionizing interactions, or Non-Ionizing Energy Loss (NIEL), caused by non-ionizing interactions. As a result, the stopping power (S) (Equation 3) is defined as a function of the IEL and NIEL, normalized to the mass material density (ρ). Where the units of the IEL and NIEL are expressed in $MeV \cdot cm^{-1}$, the ρ in $kg \cdot cm^{-3}$, and consequently the stopping power is expressed in $MeV \cdot cm^2 \cdot kg^{-1}$.

$$S = \frac{1}{\rho} \cdot \left(\left.\frac{\partial E}{\partial x}\right|_{\text{IEL}} + \left.\frac{\partial E}{\partial x}\right|_{\text{NIEL}} \right) \tag{3}$$

An important aspect to consider is that the rate of energy loss of an impinging particle is not constant, in fact, while the particle losses energy, its speed is also decreased. The

speed of the particle is closely related to the probability of interaction between the particle and matter, as slower particles have a higher chance of interaction. This phenomenon results in the Bragg Peak.

2.2.2 Radiation Effects on Electronics

In the context of radiation effects on electronic components, the IEL and NIEL are associated with effects such as Total Ionizing Dose (TID), Single Event Effects (SEE), and Displacement Damage (DD). Generally, IEL contributes to TID and SEE, while NIEL is responsible for DD.

2.2.3 Total Ionizing Dose

TID quantifies the cumulative damage induced by ionizing radiation, therefore associated with IEL effect [29]. The unit for TID is typically the Gray (Gy). One Gy corresponds to one joule absorbed by one kilogram. Alternatively, the TID is measured in rad units, especially by the space community, where one Gy is equal to 100 rad.

To understand the TID, we must define the Linear Energy Transfer (LET), which is a parameter quantifying the energy (E) deposited in a material per unit distance traveled (x), therefore, it is commonly expressed in $MeV \cdot cm^2 \cdot mg^{-1}$ units. The LET is defined as shown in Equation 4 [29].

$$LET = \frac{\partial E}{\partial x} \tag{4}$$

The TID deposited by a single charged particle can be calculated as a function of its LET and fluence (Φ), expressed in cm^2, as defined in Equation 5 [31] [29], where the factor K_{Gy} is the unit conversion parameter, with a value of $1.6 \cdot 10^{-7}\ Gy \cdot g \cdot MeV^{-1}$.

$$TID = K_{Gy} \cdot \frac{LET(E)}{\rho} \cdot \Phi(E) \tag{5}$$

Therefore, the total TID deposited by charged particles on a material in a mixed-field environment, such as the space radiation environment or CERN's LHC and CHARM facility, is computed as the sum of the contributions of each particle (p), as expressed in Equation 6.

$$TID = K_{Gy} \cdot \sum_{p=e^{\pm},\pi^{\pm},p^{\pm},K^{\pm},\mu^{\pm}} \int_0^{E_{max}} \frac{LET(E,p)}{\rho} \cdot \Phi(E,p)\, dE \tag{6}$$

Alternatively, photon interactions with matter are defined by their mass energy transfer coefficient, $\frac{\mu_{en}}{\rho}$, since photons are typically scattered or absorbed in a single process, and consequently cannot be described by LET [31]. The mass energy transfer coefficient represents the energy lost by the particle per unit length in the material and it is defined in Equation 7, where $\frac{\mu_{en}}{\rho}$ is expressed in $cm^{-2} \cdot g^{-1}$, $\frac{\mu}{\rho}$ is the mass interaction coefficient and is expressed in $cm^{-2} \cdot g^{-1}$, and the $(1 - R(E))$ factor represents the fraction of energy

transferred during the photon interaction [29].

$$\frac{\mu_{en}}{\rho} = \frac{\mu}{\rho} \cdot (1 - R(E)) \tag{7}$$

The TID deposited by a single photon is defined as a function of the mass energy transfer coefficient of the photon, its energy, fluence, and the K_{Gy} unit conversion factor. The TID deposited by a single photon is defined in Equation 8, and the TID deposited by the contribution of several photons in a radiative environment is computed as in Equation 9 [31] [29].

$$TID = K_{Gy} \cdot \frac{\mu_{en}(E)}{\rho} \cdot E \cdot \Phi(E) \tag{8}$$

$$TID = K_{Gy} \cdot \int_0^{E_{max}} \frac{\mu_{en}(E)}{\rho} \cdot E \cdot \Phi(E)\, dE \tag{9}$$

TID induces damage to semiconductor materials used in microelectronics. In MOS devices, ionizing radiation affects the oxide region due to charge trapping in the SiO_2 layer and at the Si/SiO_2 interface. An incident particle can generate e-h pairs along its path (the energy to create an e-h pair, as well as the generation rate for SiO_2 and Si/SiO_2, are shown in Table 2). Some of the generated e-h pairs recombine, depending on the LET of the particle, while others are split by the electric field. Electrons, due to their higher mobility, diffuse away faster compared to holes, which may become trapped by intrinsic defects or energy wells in the SiO_2 and Si/SiO_2 regions, degrading the device performance. Traps can be classified as oxide traps, border traps, and interface traps, depending on their position within the MOS structure. Figure 10 depicts the band diagram of a MOSFET with a positive gate bias and grounded bulk, and the mechanisms produced by ionization.

Table 2. Ionization energy and pair generation rate for Si and SiO_2.

Material	**Ionization Energy** [eV]	**Pair Generation Rate** [e-h pairs $\cdot$ Gy^{-1} $\cdot$ cm^{-3}]
Silicon (Si)	3.6	$4 \cdot 10^{15}$
Silicon Dioxide (SiO_2)	18	$8.2 \cdot 10^{14}$

2.2.4 Single Event Effect

A Single Event Effect (SEE) consists of ionization caused by a single particle at a sensitive region of the device that can result in either destructive or non-destructive damage [29]. A SEE is a stochastic event, and the type of effect caused on the electronic component depends on different factors such as the semiconductor technology, function, and topology.

Non-destructive SEEs, also known as soft errors, are events that do not result in the

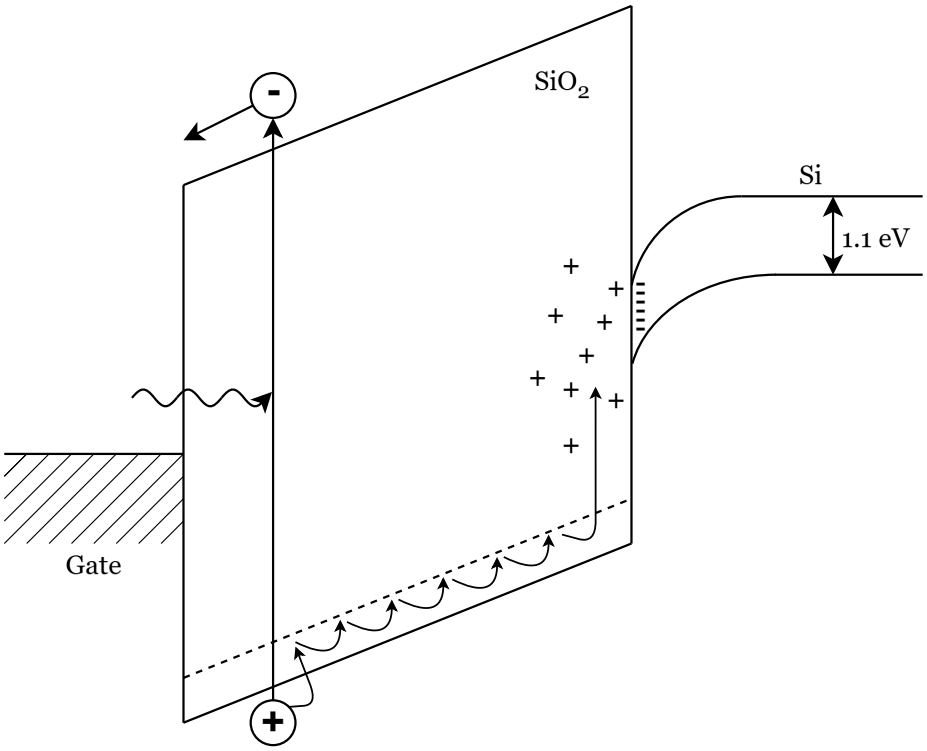


Figure 10. Band diagram of a MOSFET and the mechanism occurring due to ionization.

destruction of the electronic component [29]. The physical process that can lead to non-destructive events is depicted in Figure 11. When a charged particle traverses through the device, it can produce electron-hole pairs. Then the charges drift due to the electric field and depletion region, creating a current transient. Finally, diffusion of charges occurs, which reduces the current generated.

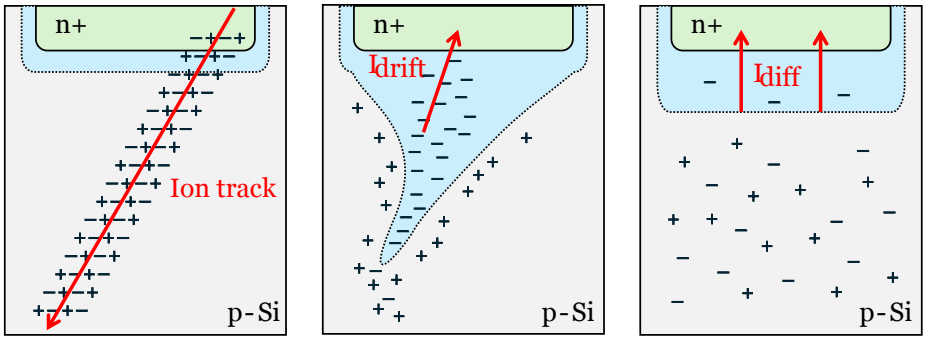


Figure 11. Charge generation and collection.

The main types of non-destructive SEE are listed below [29]:

- **Single Event Transient (SET)**: voltage or current spike producing a glitch in the circuit that can be propagated through logic, and cause a functional error in device operation.

- **Single Event Upset (SEU)**: the strike of a particle induces a transient signal causing data to change states in elements such as flip-flops, or latches.
- **Single Event Functional Interrupt (SEFI)**: the strike of a particle induces temporary non-functionalities, caused by affecting registers for control configuration. Recovering from a SEFI typically requires a reset or power cycle.

Hard errors can cause a complete failure of the device. The main destructive SEEs are listed below [29]:

- **Single Event Latch-up (SEL)**: the strike of a particle triggers a parasitic thyristor (PNPN structure), which leads to a high current flow, which can overheat the device, resulting in the permanent loss of device functionality.
- **Single Event Burnout (SEB)**: the strike of a particle triggers a regenerative avalanche breakdown effect, inducing a high current state in the device that can permanently damage the device.
- **Single Event Gate Rupture (SEGR)**: the strike of a particle deposits charges at the silicon-oxide interface, resulting in a strong electric field that can produce a short-circuit between gate and substrate.

The probability of having a SEE in a device is defined by the cross-section (σ) as expressed in Equation 10. Where N is the number of SEEs observed during an experiment, and Φ is the fluence (cm^{-2} units).

$$\sigma = \frac{N}{\Phi} \tag{10}$$

An alternative magnitude to quantify the SEE is the High Energy Hadron fluence (Φ_{HEH}), as well as the Thermal Neutron fluence (Φ_{Th}). The (Φ_{HEH}) is defined as the fluence of hadrons above 20 MeV. Hadrons below such energies have a negligible probability of producing SEEs in electronics. Moreover, a HEH equivalent fluence ($\Phi_{HEH_{eq}}$) is defined as the (Φ_{HEH}) plus the intermediate neutron fluence, corresponding to the neutron contributions between 0.2 and 20 MeV.

2.2.5 Displacement Damage

Displacement Damage (DD) is a radiation effect occurring on electronics in which, through Coulombic or nuclear interactions, an atom is displaced from its original lattice [29] [31]. An impinging particle, known as the primary knock-on atom (PKA), such as neutrons or charged particles, knocks an atom, creating a defect in the lattice, such as a vacancy, an interstitial, a divacancy, or a Frenkel Pair (Figure 12).

A vacancy refers to the absence of an atom from its normal lattice position, while an interstitial indicates a displaced atom at a non-lattice position. Two adjacent vacancies

are called divacancies, and a pair of an interstitial atom and a vacancy is known as a Frenkel Pair [29].

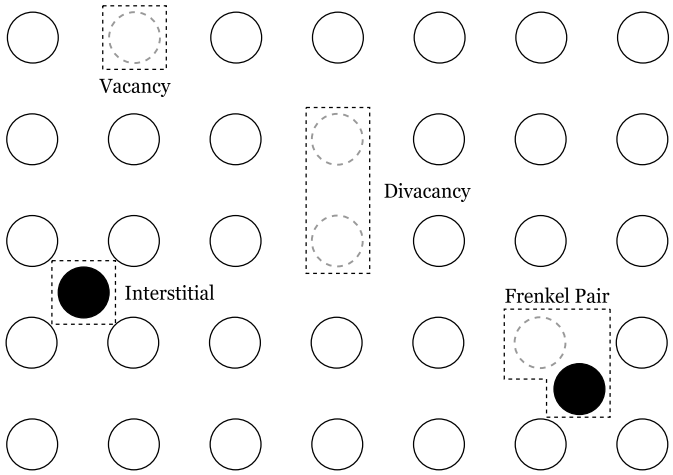


Figure 12. Defects created by displacement damage.

The number of defects generated is proportional to the energy of the PKA and incident particle NIEL [29]. Low-energy PKAs, below 2 keV, can be created with charged particles at low-energy and electrons through Coulombic scattering, and through nuclear scattering with neutrons. Low-energy PKAs (below 2 keV) can produce Frenkel (21 eV is the minimum energy to create a Frenkel pair in silicon) pairs or small defect clusters. Intermediate-energy PKAs (between 2 and 12 keV) are generated with 10-20 MeV heavy charged particles, and can create regions with high density defects (single defect clusters). And high-energy PKAs (above 12 keV) are created with > 20 MeV heavy charged particles, and can generate multiple and large cascades of defects.

The Displacement Damage Dose (DDD) produced by a single particle is defined in Equation 11. However, DD effects are commonly expressed as a function of an equivalent fluence of 1 MeV neutron. Therefore, the Displacement Damage Equivalent Fluence (DDEF) is calculated by multiplying the fluence by a parameter κ, which is the ratio of the particle NIEL and the 1 MeV neutron NIEL, as defined in Equation 12.

$$DDD = \frac{NIEL(E)}{\rho} \cdot \Phi(E) \tag{11}$$

$$DDEF = \Phi_{1\,MeV\,neq.} = \kappa_{n_{1\,MeV}} \cdot \Phi(E) = \frac{NIEL(E)}{NIEL_{n_{1\,MeV}}} \cdot \Phi(E) \tag{12}$$

Similarly to TID, the DDEF in a mixed field environment can be calculated by adding the contributions of all the different particles, as in Equation 13.

$$DDEF = K_{Gy} \cdot \sum_{p=e^{\pm},\pi^{\pm},p^{\pm},K^{\pm},\mu^{\pm}} \int_0^{E_{max}} \kappa(E,p) \cdot \Phi(E,p)\, dE \tag{13}$$

2.2.6 Space Environment

WMNs are a technology with potential for space applications, and therefore, devices forming the network can be deployed in satellites or space stations. For such applications, it is important to understand the space radiation environment, which consists of a complex mixed field of various particles at different energies. Space radiation originates from both extragalactic sources and sources within our solar system, such as from the solar activity. Table 3 shows the main sources of space radiation. In the context of electronics, there are three main types of space radiation of concern [30] [32]:

- **Trapped radiation**: refers to particles (electrons, protons, hadrons, and ions) trapped by Earth's magnetic field in the Van Allen belts. The Van Allen belts consist of two belts: the inner and outer belts, mainly containing electrons (up to 10 MeV) and protons (up to 500 MeV). The inner belt extends from approximately, 1.1 times the Earth's radius (R_E) to $4 \cdot R_E$, while the outer belt ranges from $4 \cdot R_E$ to $6 \cdot R_E$. The flux levels depend on the solar cycles and the solar activity.
- **Cosmic radiation**: consists of galactic cosmic rays (GCRs), which originate from outside the Solar system and, from Earth's perspective, arrive isotropically. The particles (mainly protons, alphas, electrons, and heavy ions) are highly energetic, reaching energies of several GeV. GCRs fluxes are inversely proportional to solar activity, due to the fact that at maximum solar activity, the magnetic field is stronger and deflects the GCRs.
- **Solar radiation**: the Sun releases protons and electrons continuously, producing solar winds of relatively low energies. Moreover, higher energy particles are emitted during solar flares or Coronal Mass Ejections (CMEs). The intensity of these phenomena is not constant over time, but is subject to the solar cycles, which follow an approximately 11-year periodic cycle of solar activity.

Table 3. Main sources of space radiation, the type of particles for each source, and their energy ranges [30].

Radiation source	Particle	Energy range
Radiation belts	Electrons	eV - 10 MeV
	Protons	keV - 500 MeV
	Ions	keV - 500 MeV
Solar flares	Protons	keV - 500 MeV
	Ions	1 to 10 MeV/n
Galactic Cosmic Rays	Protons and Ions	Max. flux at 300 MeV/n

2.2.7 CERN Environment

CERN, founded in 1954 and located on the Franco-Swiss border, is the largest particle physics laboratory in the world [31]. CERN's main objective is to provide particle accel-

erators and infrastructure for high-energy physics research, but also contributes to the development of state-of-the-art technologies in fields such as engineering, mechanics, or computer science, among others.

CERN's accelerator structure consists of seven main accelerators, which represent sequential stages in the accelerator chain to gradually accelerate the beam to high energies, as shown in Figure 13. The first circular accelerator is the Proton Synchrotron Booster (PSB), which is fed by 160 MeV protons from the linear accelerator LINAC 4 and provides 2 GeV to the Proton Synchrotron (PS). PS has a length of 628 meters and accelerates the particles up to 24 GeV. The PS feeds the Super Proton Synchrotron (SPS), as well as the Antiproton Decelerator complex and fixed target experiments such as CHARM, IRRAD, and nTOF. The SPS accelerates particles up to 450 GeV and feeds the Large Hadron Collider (LHC), originally being the Large Electron Positron (LEP) collider. SPS also sends particles to fixed target experiments such as HiRadMat, AWAKE, and CENF.

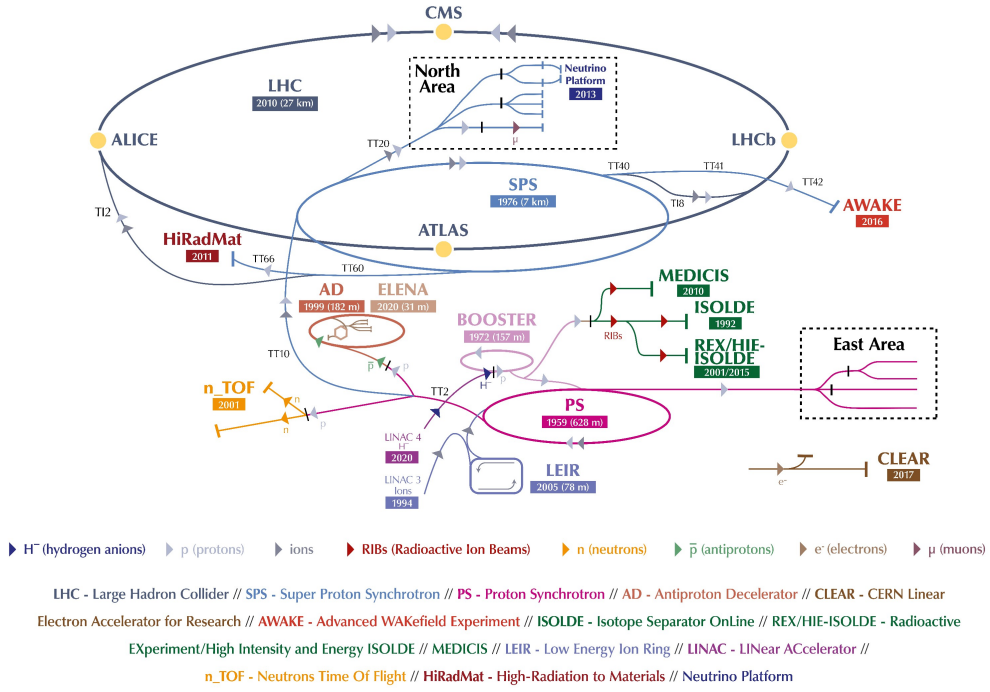


Figure 13. CERN's accelerator structure [33].

The LHC is a 27-kilometer circular particle accelerator containing superconducting magnets and acceleration structures to guide and accelerate two beams of particles, that are moving in opposing directions in ultrahigh vacuum, to collide at 13.6 TeV at the four detector sites (ATLAS, CMS, LHCb and ALICE) [29] [31]. The superconducting electromagnets are cooled down with liquid helium to $-271.3\,^{\circ}\mathrm{C}$ to operate in a superconducting state, allowing efficient electrical conduction without resistance or energy loss.

The luminosity is the parameter used to quantify the number of interactions per unit

surface and time. The capability to generate collisions relevant to high energy physics is determined by the integral luminosity, typically expressed in inverse femtobarn (fb^{-1}) units [29] [31]. As a reference, a fb^{-1} is approximately equal to 10^{14} proton-proton interactions. Therefore, the performance of the accelerator is directly linked to its luminosity, and can be improved by increasing its availability or by increasing its luminosity. Accordingly, the LHC is being upgraded to the High-Luminosity LHC (HL-LHC) with the aim of increasing the LHC performance by a factor of 10. Figure 14 shows the integrated luminosity for each year of the LHC operation, where the performance in 2024 exceeded the previous years. The luminosity increase results in higher radiation levels at which electronic components are exposed. This is the reason why the impact of radiation needs to be studied and limited, being the main mission of the Radiation to Electronics (R2E) project.

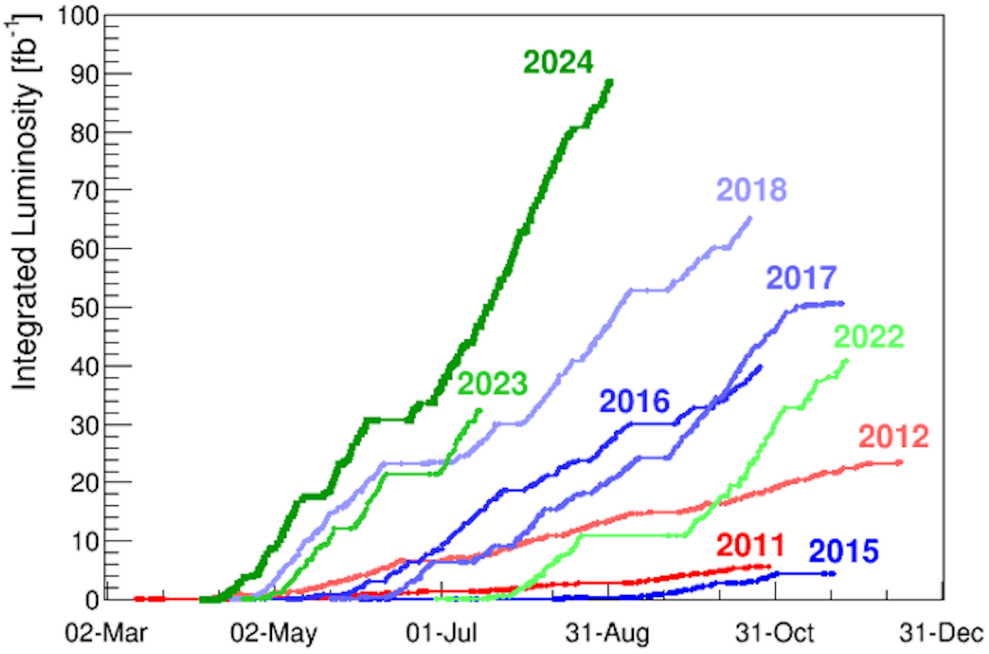


Figure 14. Integrated luminosity as a function of time for each year of LHC operation [34].

There are three main radiation sources at the LHC [29] [31]. One source of radiation is the inelastic collisions at the interaction points of the LHC, where a proton-proton collision produces a particle debris and a shower of secondary particles, including protons, pions, kaons, electrons, photons and muons. This radiation source is proportional to the luminosity of the accelerator. Another radiation source is the beam interactions with the LHC elements, in which the beam's halo particles interaction with the collimators produces a particle shower. This type of interaction depends on the beam intensity. The last main radiation source is due to the beam interaction with residual gas in the

vacuum pipes.

The LHC operation confirmed the predicted Standard Model (SM), as a result of the Higgs Boson discovery in 2012. The SM is able to describe all phenomena possible with current collider experiments, however, there are still open questions regarding fundamental phenomena that remain unexplained, such as the evidence of dark matter, the abundance of matter over antimatter, and the non-zero neutrino masses. Therefore, physics beyond the Standard Model (BSM) is required to further enhance the current knowledge [35]. In this context, the Future Circular Collider (FCC) appears as a potential research infrastructure capable of providing a higher performance particle collider, allowing for the extension of the current research.

The FCC is still under a feasibility study phase, however, it is planned to be an underground circular tunnel with a circumference of 90.7 kilometers, with eight surface sites and four experiments. The FCC will consist of two stages, the first stage, the FCC-ee, an electron-positron collider expected to start in the late 2040s, and the second stage, the FCC-hh, a proton-proton collider, expected to operate for 25 years starting in the 2070s. The goal of the FCC-hh is to provide proton-proton collisions in the order of 85 TeV with an integrated luminosity of about $20\,ab^{-1}$ [36]. Moreover, proton-ion and ion-ion collisions are planned, and electron-proton and electron-ion collisions could be possible.

Despite the limitation to deploy electronic devices and develop communication networks due to the radiation environment, the normal operation as well as maintenance and repair activities at CERN's accelerators require to be supported by high speed data links and advanced communication networks to provide connectivity within the tunnels [37].

Wireless networks are a potential solution due to their benefits, such as flexibility, accessibility, and reduced cost compared to infrastructured and wired solutions. However, wireless networks at CERN's accelerators need to be power efficient, responsive, reliable, scalable, and mobile as well as robust against interference and ionizing radiation [38]. Besides the intrinsic requirements of wireless networks in such environments, the wireless emission at CERN is regulated by *L'agence nationale des fréquences* (ANFR) in France, and the Federal Office of Communication (OFCOM) in Switzerland, in order not to cause any interference to external RF services.

Wireless connectivity in the industrial sector is typically provided by Wi-Fi connectivity, but this technology is not suitable for CERN's environment due to scalability and connectivity constraints. Firstly, to provide full coverage, a large number of devices need to be deployed due to power and distance limitations. Secondly, with the current network configuration at CERN, roaming between Wi-Fi access points at CERN is required, degrading the performance of real-time services. Moreover, the devices will be affected by ionizing radiation. Nowadays, a wireless network for CERN's accelerator environment

based on Wi-Fi is impractical, although solutions to address its limitations exist.

CERN's current wireless solution for the accelerators relies on a leaky feeder antenna system, a technology commonly used for underground environments such as tunnels and mines. This system consists of periodically slotted coaxial cables which act as distributed antennas allowing the emission and reception of radio signals [37] [38]. However, due to signal loss, these radiating cables are typically used below 1 GHz, limiting the maximum data rate.

At CERN's accelerator and experimental areas, over 60 kilometers of radiating cable are deployed, which are connected to more than 50 injection points (located in non-radiation areas). The leaky feeder antenna system deployed at CERN is common to different radio services, supporting different frequency bands from 400 to 900 MHz [37] [38]. For instance, it is used for TETRA and TETRAPOL services operating in the 400 MHz band for CERN's fire brigade and public safety agencies, and for 5G, LTE, and UMTS in the 700 MHz, 800 MHz, and 900 MHz bands, respectively.

In [38], a Low Power Wide Area Network (LPWAN) solution is introduced and successfully tested for CERN's accelerator environment for radiation monitoring of electronics, using LoRaWAN, operating at the ISM band (868 MHz in Europe). It is an architecture based on wireless sensor end-nodes inside the harsh areas and LoRa gateways installed in shielded caverns connected to the radiating cables [39]. This solution, featuring a LoRaWAN network, allows redundancy as the transmitted data by the nodes can be received by multiple gateways. Then the gateways forward the data received to the network server. Despite this solution fits with the radiating cable requirements (it operates at 868 MHz) and high coverage is provided, possible collisions and bottlenecks are potential constraints. These limitations can be addressed with proper modulation methods and enhanced reliability mechanisms.

The existing wireless solution implemented at the LHC provides the basis for the development of a wireless communication system for the FCC. However, due to the FCC's larger scale and more intense radiation environment, the current approach is not directly scalable. Providing full coverage to the new FCC accelerator by using the current infrastructure would require the deployment of considerably more radiating cable length and related devices, resulting in increased complexity and cost. To address these constraints, an optimized and robust solution is required, and it should already be considered at the infrastructure level from the early design stages. A fully wireless architecture is a potential solution due to its higher flexibility and reduced overall costs.

In this context, a custom WANET topology emerges as a promising alternative to be studied for implementation at the FCC and other accelerator facilities, not only due to its flexibility, adaptability, and scalability but also because of its lower deployment and maintenance costs.

3 Experimental Methodology

Mesh networks, as discussed in the previous section, are an emerging technology with potential applications in radiation environments, such as space and accelerator facilities. The experimental campaign was conducted to evaluate the behavior of a WMN under radiation exposure, focusing on system-level performance, robustness, and resilience. The objective is to validate the experimental methodology, demonstrate a proof of concept, and assess the feasibility of deploying mesh networks for applications in harsh environments.

In this context, a WANET topology was deployed and tested at the CHARM facility at CERN, which provides a representative mixed-field radiation environment. The AstroMesh is being designed for this purpose, integrating LoRa transceivers and a radiation tolerant design. However, as an initial evaluation phase and risk-mitigated strategy, this first evaluation campaign has been conducted using a functionally representative platform based on Feather M0 boards. This approach enables the validation of mesh network functionalities and resilience mechanisms, without risking the AstroMesh hardware.

This section presents the CHARM facility, where tests were conducted, followed by a description of the AstroMesh platform as the system to explore mesh networks at CERN. The hardware and software implemented in the experimental testbed, including the system platform and libraries, are explained. Finally, the experimental setup is described, as well as the monitoring, control, and methodology used for data collection and analysis.

3.1 CHARM: CERN Highly-Accelerated Mixed Field

The CERN Highly-Accelerated Mixed Field (CHARM) is an irradiation facility designed to test electronic systems in representative radiation environments, such as accelerators, ground and atmospheric conditions, and space applications. The facility has approximately $70\,m^3$ of space available for radiation tests [40].

The CHARM facility is located in the Proton Synchrotron (PS) East Area Hall at CERN. PS supplies CHARM with a mono-energetic $24\,GeV/c$ proton beam. The beam is pulsed, meaning that it is structured in spills with 350 ms length, with an intensity of around $5 \cdot 10^{11}$ protons per pulse.

The interaction of the proton beam and a cylindrical target generates the mixed field. The target can be of 3 types: solid copper, solid aluminum, and aluminum with holes. When the proton beam hits the target, it results in a secondary mixed radiation field, which is the one used to test the electronic components inside the facility. Moreover, the facility has patch panels connected to the control room, to power and control the

electronic devices [40].

Due to the nature of the mixed field, it may be quite complex to differentiate between fluxes of single particles. Therefore, quantities such as the High Energy Hadron (HEH) are used for defining the flux intensity. The HEH is the flux of all hadrons, such as protons, neutrons, or pions, with an energy above $20MeV$.

The radiation field can be adjusted by the different configurations of the facility and the test position inside the room [40]. The radiation intensity at specific positions inside the facility can be adjusted by four movable shielding structures, two of which are made of concrete and the other two of iron. This allows the modulation of the particle spectra to the required radiation field. The radiation field expands all over the radiation area, being homogeneous in an area of at least one square meter. Positions closer to the beam are dominated by charged hadrons with energies in the GeV range. And although electronics could be tested at any location inside the facility, there are some standard test positions. These test positions are shown in Figure 15, and are the next: grid position (G0), lateral rack positions (R1 to R9), and longitudinal positions (R9 to R13).

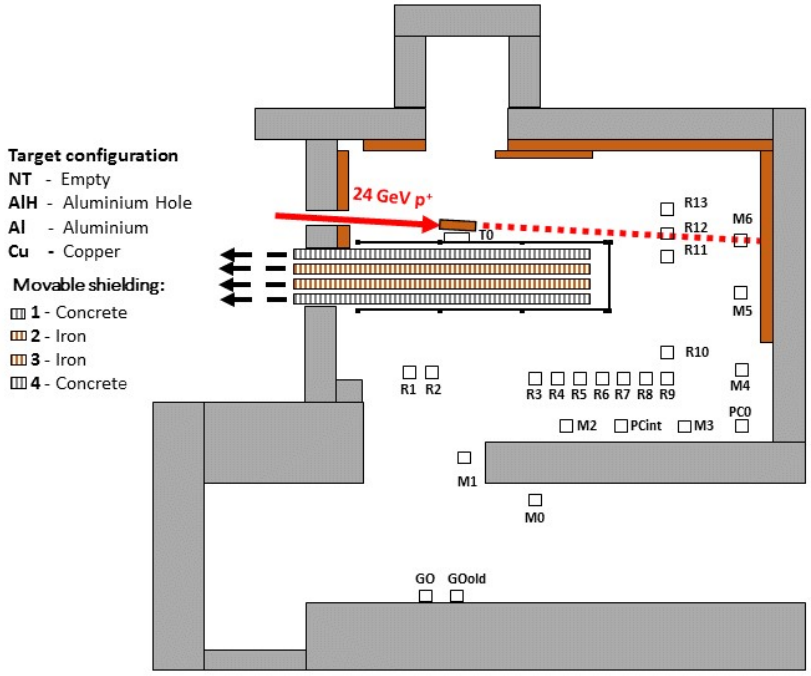


Figure 15. Schematic of the irradiation room at CHARM. Test positions are numbered, and the target position and shielding are indicated.

Therefore, CHARM provides a representative radiation environment that allows the testing of mesh networks. The next sections describe the systems designed to operate under radiation environments and the implementation of the test campaign.

3.2 AstroMesh Platform

AstroMesh is a platform designed and developed at CERN to evaluate the feasibility of mesh networks operating in radiation-prone environments. It consists of an intersatellite communication and radiation monitoring system for CubeSats, developed within the framework of the CERN Venture Connect Knowledge Transfer program. AstroMesh design is based on the previous radiation monitoring payload SpaceRadMon-NG developed by CERN [41].

The primary goal of AstroMesh is to demonstrate a robust, fault-tolerant wireless communication network using the LoRa protocol in harsh environments such as space or accelerator facilities. This platform consists of a main board ($96 \cdot 92 \cdot 16.56\ mm^3$) that integrates a Field Programmable Gate Array (FPGA), a Microcontroller Unit (MCU), the LoRa transceivers, and power and interface units. Moreover, a mezzanine board ($53 \cdot 77 \cdot 3$ mm^3) with additional radio transceivers can be attached to extend its communication capabilities. In this context, the MCU, based on an ARM Cortex M0+ core, interfaces the LoRa transceivers and communicates with the FPGA through Serial Peripheral Interface (SPI). The FPGA, based on Microchip ProASIC, communicates externally using the I2C protocol.

The main board integrates two LoRa transceivers operating at 868 MHz and 2.4 GHz, respectively, designed to enable mesh network communication. The mezzanine board extends the communication capabilities with two additional ultra-wideband (UWB) radio transceivers, designed to operate in the frequency ranges of 150 to 960 MHz and from 2.1 to 6.5 GHz.

The board operates from a 5 V supply and supports low-power modes, consuming approximately 3 mA when both the MCU and FPGA are disabled. Power consumption increases to 137 mA during LoRa transmission with the FPGA enabled. The platform is designed to have an expected lifetime of 500 Gy and operate in a temperature range between -20 and $+80$ °C.

Although AstroMesh is the target system for radiation investigation of mesh network performance, it was not directly used in the initial experimental campaign described in this work. Instead, a risk-mitigation strategy was adopted using a functionally representative platform that allows for the validation of the experimental methodology and mesh network functionalities before exposing the more complex and expensive AstroMesh hardware to radiation.

3.3 Evaluation Platform

3.3.1 Hardware overview

Although the AstroMesh platform is designed to explore the mesh network performance in harsh radiation environments, this work presents the first evaluation phase

of the project. To mitigate the risk of exposing the AstroMesh hardware to radiation during this initial testing, a functionally representative platform was employed instead. The mesh network tested in this first assessment stage was composed of nodes based on Adafruit Feather M0 boards.

For this testing campaign, the nodes consisted of an Adafruit Feather M0 device with the RFM69 transceiver, which integrates an ATSAMD21G18 ARM Cortex M0 MCU. The MCU features an ARM Cortex M0 core running at up to 48 MHz, with 256 KB of flash memory, 32 KB of SRAM, and operating at 3.3 V [42].

The HopeRF RFM69 is a low-power RF module capable of operating in a wide range of frequencies, including 433, 868, and 915 MHz license-free ISM (Industry, Scientific and Medical) frequency bands. It supports FSK and GFSK modulation schemes, with bit rates up to 300 kbps [43].

The Adafruit Feather M0 with RFM69 is compatible with the Arduino Integrated Development Environment (IDE). Additionally, Adafruit provides open-source software libraries that integrate with the IDE. The use of the Arduino platform enhanced a rapid and simplified firmware deployment across multiple nodes. For future testing, it is recommended to transition to a fully C++ environment, which provides higher control over the operations and functionality.

Although this configuration lacks the full radio capability of the AstroMesh platform, it provides a valid and cost-effective testbed for evaluating and assessing mesh network functionalities as well as the behavior of the MCU under radiation exposure.

3.3.2 Software Overview

The test consisted of evaluating and studying the performance of a WANET. Therefore, the WANET functionalities of the nodes were obtained by using the RadioHead packet radio library. RadioHead is a packet radio library for microprocessors, which provides a complete object-oriented library for communication of packetized messages for a wide range of data radios and embedded microprocessors [44]. This library consists mainly of two sets of classes, the drivers and the managers. The drivers provide low-level access to the packet radios, while the managers provide a high-level access for sending and receiving tools for different requirements. Therefore, the program will have an instance of a driver, which provides access to the data radio, and an instance of a manager, which uses this driver for the communication. Among the drivers in the library, the driver RH_RF69 is of interest in this test, as it works with the Hope-RF RF69B-based radio modules, such as the RFM69 transceiver, supporting FSK and GFSK modulations. Among the managers provided by the library, it is important to highlight the following instances: RHDatagram for addressed unreliable messages, RHReliableDatagram for addressed, reliable, retransmitted, acknowledged variable length messages, RHRouter for multihop delivery with pre-programmed routing, and RHMesh for multihop delivery

with automatic route discovery and route maintenance.

The basic RHDatagram functionalities can be extended with RHReliableDatagram, which adds reliability through the implementation of acknowledgments and retransmissions [44]. The RHRouter subclass extends the RHReliableDatagram with routing capabilities through the network with reliably addressed messages between each intermediate hop in a link between a source and a destination node. For the routing functionalities of RHRouter, each node has to have programmed the first hop to an intermediate node to communicate with a destination node, meaning that each node in the network will have a routing table with the information of each destination node and the first hop to reach each of them. Finally, the RHMesh subclass extends the multihop routing capabilities with automatic route discovery and route maintenance. This enhances its performance, especially in dynamic networks, where the topology can be modified. Therefore, a node will automatically discover a route for a destination node, and it will maintain it until it becomes unavailable.

The packet structure for the mesh network, as shown in Figure 16, consists of an RHDatagram header, followed by a RoutedMessage header, which contains a MeshMessage header and the application payload [44]. The RHDatagram header consists of 4 bytes, and it is used for basic addressing and message identification. It consists of four fields of 1 byte each: TO (destination node address), FROM (source node address), ID (message ID used for tracking), and FLAGS (flags to determine specific conditions such as ACK or retries). The RoutedMessage header consists of 5 bytes and is used for routing purposes, providing end-to-end delivery information. The five fields of this header are: DEST (final destination node address), SOURCE (source node address), HOPS (number of hops traversed), ID (message ID), and FLAGS (flags to determine specific conditions). The MeshMessage header consists of 1 byte indicating the mesh message type: application message, route discovery request, route discovery reply, and route failure notification. For the application message type, a variable-length payload is added. For the route discovery request and route discovery response messages, three fields are added: length of the destination node address (1 byte), address of the destination node (1 byte), and route from the source to the destination node (variable length depending on the size of the route). For the route failure notification type of message, 1 byte is added indicating which node of the route failed.

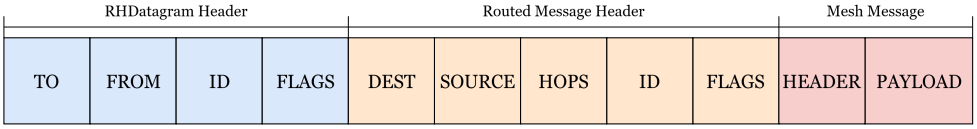


Figure 16. Packet structure.

3.3.3 Mesh Network Topology design

The mesh network implemented in this evaluation phase follows a WANET architecture. The network consists of a set of static nodes that communicate with each other in a flat and homogeneous topology. This means that every node has the same hardware capabilities and operates under the same communication logic. Due to the non-hierarchical and non-centralized structure, the network features self-organization and self-healing capabilities.

The tested network is designed with a reactive routing protocol, enabling dynamic routing discovery and packet forwarding. These features allow the network to adapt to changes in connectivity caused by radiation-induced faults.

For the tests conducted in CHARM, the payload of the application packets will also consist of some fields that allow for the computation of relevant metrics of the network. The payload structure is shown in Figure 17. The payload length depends on the number of nodes conforming to the network, which is previously programmed for these tests. The first field (1 byte), PKTCNT, is a counter of the number of attempts to communicate with each node in the network. This counter is not incremented for each attempt of a packet sent, but per iteration of the full network. For each node in the network, three 1-byte fields are transmitted: HOP – the first hop required to reach the destination node; PKT – the total number of packet transmission attempts to the destination node; and ACK – the count of packets successfully delivered, as confirmed by acknowledgments from the destination node. Therefore, in a network with three nodes, the payload sent is 10 bytes, while in a network with four nodes, the payload is 13 bytes long. Therefore, a complete application message packet for a 3-node network is 20 bytes.

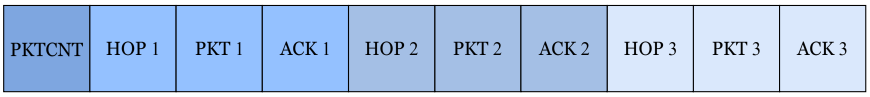


Figure 17. Payload structure.

Figure 18 shows the flow chart of the program that manages the communication, route discovery, and route maintenance of each node in the network. After the initialization of the node, it will enter an infinite loop in which, at each iteration, it will try to send a packet to a node in the network, and it will listen for incoming packets addressed to itself. Firstly, it will check if a route already exists for the destination node; if no route exists, it will start the route discovery process. If the route is created, it will transmit the packet and wait for an ACK reception confirming the successful delivery from the destination node. If, after a certain amount of time, the ACK is not received, the node will send the same packet, and this is done up to a specific number of retrial attempts. If no ACK is received after the retry attempts, the route to that destination is removed. On the next communication iteration with that destination node, a new route will be estab-

lished through the route discovery process if the node is functional or in range. Secondly, the node will start listening for a certain amount of time, determined by the established timeout, for upcoming messages addressed to it. If, after the timeout, no packet has been received, the node will start the next iteration. If a packet has been received, the information is extracted, and an ACK will be sent to the source node, or the packet will be forwarded if needed.

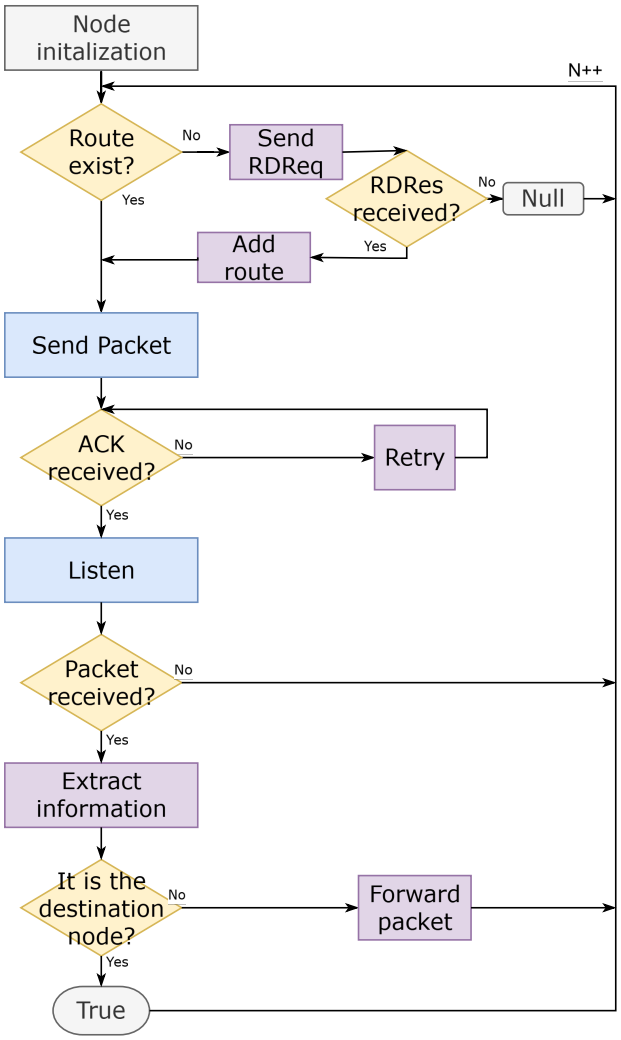


Figure 18. Flow chart of the operating nodes.

The route discovery process consists of discovering a route to a destination node by sending a Route Discovery REQuest (RDREQ) and waiting for the Route Discovery RESponse (RDRES). The RDREQ is broadcasted by the source node. When an RDRES is received, the node checks if it is the destination node. If it is the destination node, it replies with a unicast RDRES. If the receiver node is not the destination node, it will rebroadcast the request, adding itself to the route of nodes visited so far. To avoid loops within the network, if a node receives an RDRES that includes itself already in the route,

it will ignore it. When a node receives the RDREQ, it can deduce routes to the originator node from the route list created, allowing it to also create a route back from the final destination node to the originator source node. Therefore, the RDRES and RDREQ packets ensure that the source and destination establish a route, as well as create a route between the intermediate nodes and the source and destination of the former link. It is important to take into account that in the case of multi-path routes, the first reply from the destination will be the one used.

It is important to note that this library handles reliable multihop communication by implementing hop-to-hop acknowledgments, but not end-to-end acknowledgments. This means that, for instance, a node knows that it has been successfully delivered to the next hop, but not if it was delivered to the destination node. Therefore, when a packet is lost in an intermediate node, this node transmits a unicast Route Failure Message to the original sender node, following back the same route. And when a node receives this Route Failure Message, it deletes the route to the destination node, meaning that it will need to discover a new route on the next attempt of communication with the destination node.

3.4 Experimental Setup Implementation

The experimental setup, shown in Figure 19, allows for performing a network characterization under radiation conditions. The Feather M0 boards, as node devices, are placed inside the irradiation room in CHARM. Depending on the test performed, the number of nodes and gateways can vary. Including multiple gateways in a setup can serve as a mitigation technique at network-level where redundancy is implemented. Multiple gateways allow for failover capabilities, meaning that if a single gateway suffers a failure, the other gateway is still active to receive the network packets. All the testing nodes are powered with power supplies (PSUs) located in the control room. A coaxial cable connected to the 5V and GND pins of the Feather M0 is connected to the patch panel, which is connected to the control panel with cables of approximately 30 meters long. From the control panel, a cable is connected to the PSU's corresponding channel. Moreover, for the gateway nodes, an additional serial communication is needed. Therefore, a TTL232rg connected to the UART RX, TX, and GND pins on the board provides USB connectivity with the patch panel, which is connected to the control panel and the PC. Furthermore, the PSU is monitored with the test PC, as both of them are connected to a router via Ethernet.

3.4.1 Data Collection and Monitoring Control

The experiment is monitored with a Python script on the PC (see Appendix A), which is responsible for acquiring the data and power cycling the devices in case of failure. The script's main functionalities are to acquire and store the current data of each channel

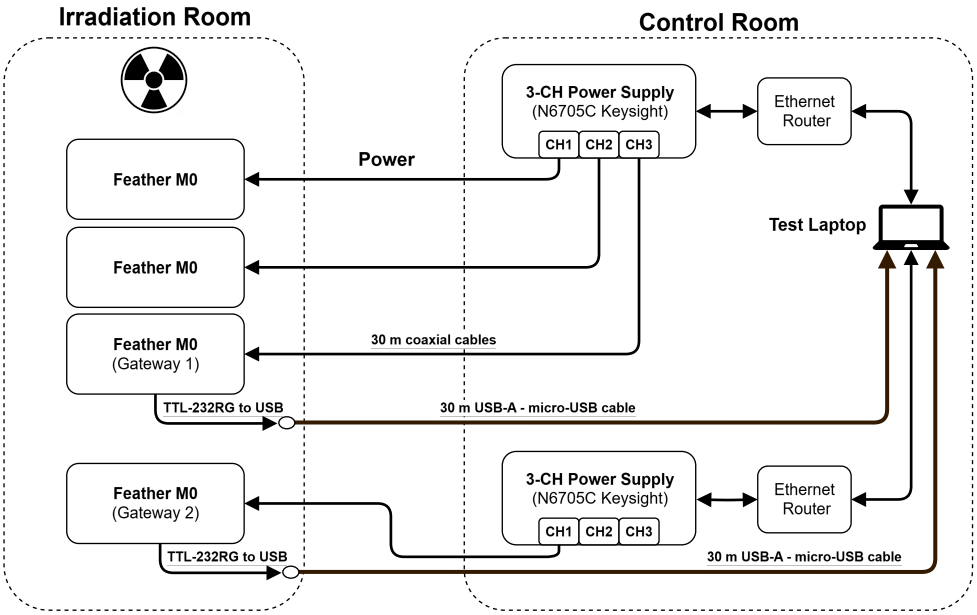


Figure 19. Experimental setup.

of the PSU used to power the nodes and the network data obtained by the serial communication of the gateways in the network, as well as to monitor that each node in the network is communicating with the others. For this monitoring task, each transmitted or received packet by the gateway is sent via serial communication to the PC, where it is parsed. The next hop fields are analyzed to check if the nodes are still responding and communicating within the network. A timer is initialized every time a node is seen in the packets, and after a specific node becomes inactive for a certain amount of time, it is power cycled.

3.4.2 Data Analysis

The Python script used for monitoring the nodes provides four text files as an output: a file with the power cycling events, a file with the current monitoring from the PSU, and two files with the network data obtained by each of the gateway nodes. The data analysis is done with MATLAB (see Appendix B).

On the one hand, regarding the network performance analysis, firstly, the two files containing the network information are imported and preprocessed. The preprocessing consists of limiting the data to the time window of interest, and an error correction code is implemented to avoid corrupted data. Secondly, the fluence data is obtained from CHARM's commissioning web interface, and for this experiment, the HEH fluence is imported. As the information obtained from the network is faster than the fluence samples, the latter is interpolated to match the network information timestamps. Once the fluence data is added to the network information, a MATLAB struct is created containing a table for each node in the network. Each table is created by filtering the preprocessed

data by the nodes that are transmitting a packet. Moreover, as the gateway provides more information about transmitted packets, the gateway node is divided again into tables filtered by each destination node. Therefore, the struct contains six tables. Then, the counters of 1 byte, contained at each table, need to be unwrapped to have a continuous count of packets. Additionally, a check on the counter values is done, and potential bit errors during the transmission are detected. Finally, metrics such as the PDR, RTT, RSSI, packet ratios, and throughput are calculated, and graphs are generated.

On the other hand, the text file containing the power cycling events for each node and its respective timestamp is imported. As the timestamp information is related to a fluence, the power cycling events are plotted as a function of the fluence. Moreover, from the number of power cycling events, which is related to a SEE, the cross-section is calculated as in formula 10.

4 Results and Discussion

This section presents the results obtained from the radiation testing of the WMN. The tested mesh network consisted of a static WANET architecture, characterized by a flat and homogeneous topology, and a reactive routing protocol. The WANET is formed by four nodes (Node 1, Node 2, Node 3, and Node 4) based on Adafruit Feather M0 hardware and configured to communicate with each other, all located at G0 position inside the CHARM facility. The nodes are configured to operate with FSK, at 868 MHz with a frequency deviation of 5 kHz and a bit rate of 2 kbps, featuring non-coherent detection. Although every node in this network operates under the same logic, Node 1 and Node 4 are also designated to be gateways, Gateway 1 (GW1) and Gateway 2 (GW2), respectively. The gateway nodes are directly connected to the control PC in the control room, where they log real-time network activity and facilitate data acquisition. This setup enables network-level mitigation strategies, providing redundancy in packet acquisition from other nodes across the network.

4.1 Network Performance Evaluation without Radiation

A dry-run test of the WMN without exposing it to radiation conditions was conducted to check the measurement setup and procedure, and to be used as a reference. This network consisted of a total of three nodes, with one gateway, and was active for approximately 72 hours. Figure 20 shows a high PDR of the gateway node of the network. The other nodes in the network showed similar performance in terms of PDR, as seen in Figure 21. The drop of the PDR at around hour 9 of the test is due to lost packets, possibly caused by interferences or collisions. Moreover, no power cycling events occurred during the run, meaning that the network is stable.

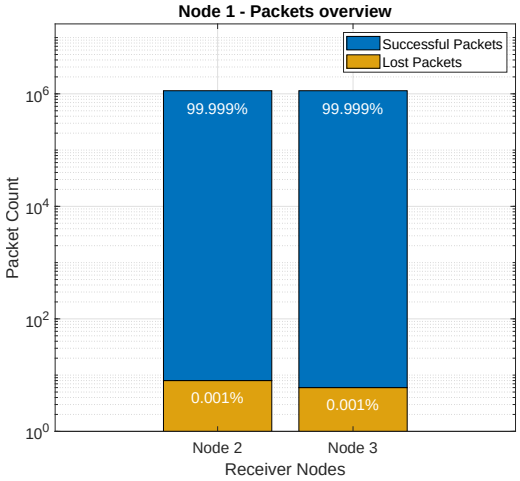
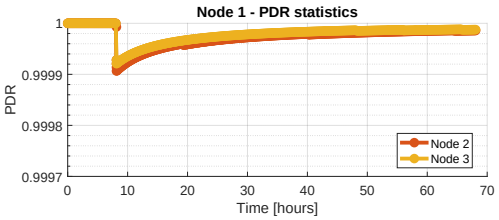
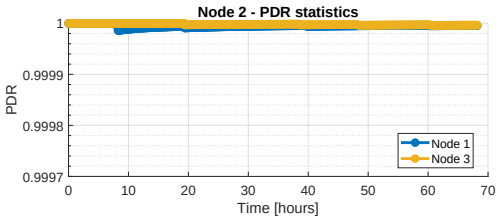


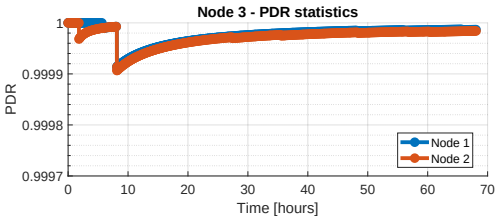
Figure 20. Packet delivery rate for Node 1 under no-radiation conditions



(a) Node 1 overview.



(b) Node 2 overview.



(c) Node 3 overview.

Figure 21. Overview of the PDR over time for the three nodes in the network.

4.2 WMN Performance

The tested wireless network operated for approximately 238 hours under irradiation at position Go inside CHARM following the setup described in Section 3.4, reaching a total fluence of $9.59 \cdot 10^{10}$ HEH/cm^2 (corresponding to a total TID of 55.84 Gy), as shown in Figure 22.

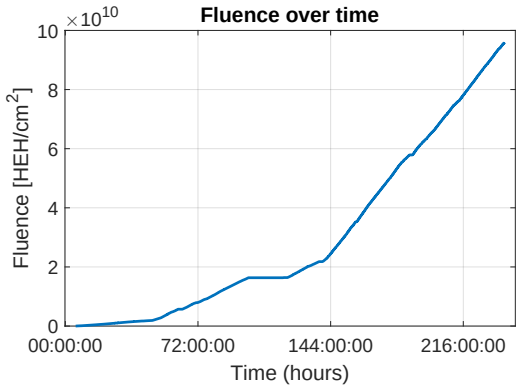
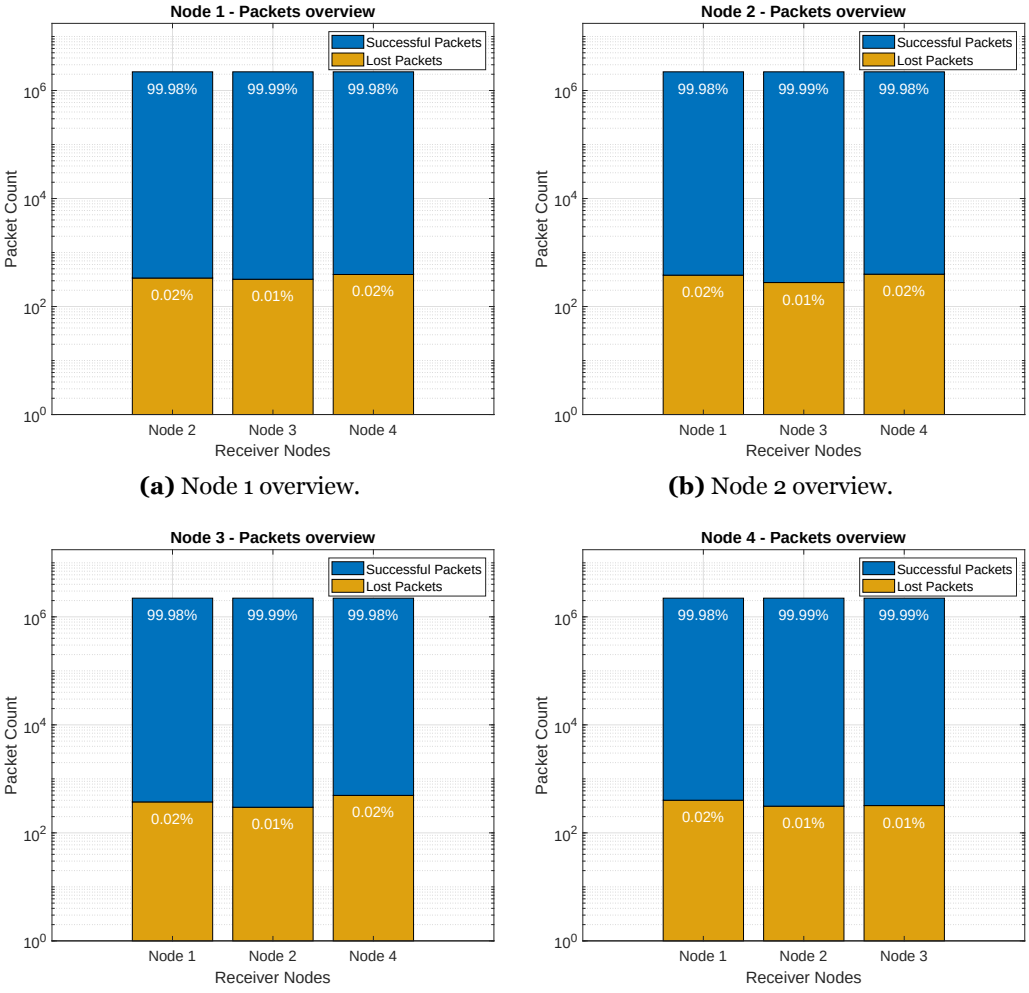


Figure 22. Fluence over time during the experiment.

Network performance is primarily evaluated by the PDR information. Figure 23 provides a summary of the final PDR values for each communication link across the network. Figures 23a, 23b, 23c and 23d present a detailed breakdown of each node's performance as a sender, indicating the ratio of successfully delivered (in blue) and lost packets (in yellow) for each receiver node. The results are represented by stacked bar charts, where the vertical axis, in logarithmic scale, indicates the total number of transmitted packets. Each bar corresponds to a unidirectional link between a sender and a receiver node.

However, it is important to note that the successful transmission of a packet is only confirmed when the corresponding ACK is received, involving a bidirectional communication at the protocol level.

The performance in terms of PDR of all four nodes in the network is similar, showing a robust communication reliability despite radiation exposure. Each node transmitted approximately $2 \cdot 10^6$ packets to each of the other three nodes in the network during the experiment. The PDR is consistently high for all links with values higher than 99%, while the amount of lost packets remains relatively low compared to the total amount of packets transmitted, as shown with the logarithmic scale of the chart. The main reason for lost packets is due to the radiation exposure of the electronic devices acting as nodes, as confirmed by the performance shown by the network evaluated without radiation. However, some losses are also due to the intrinsic nature of wireless communication.



(a) Node 1 overview.

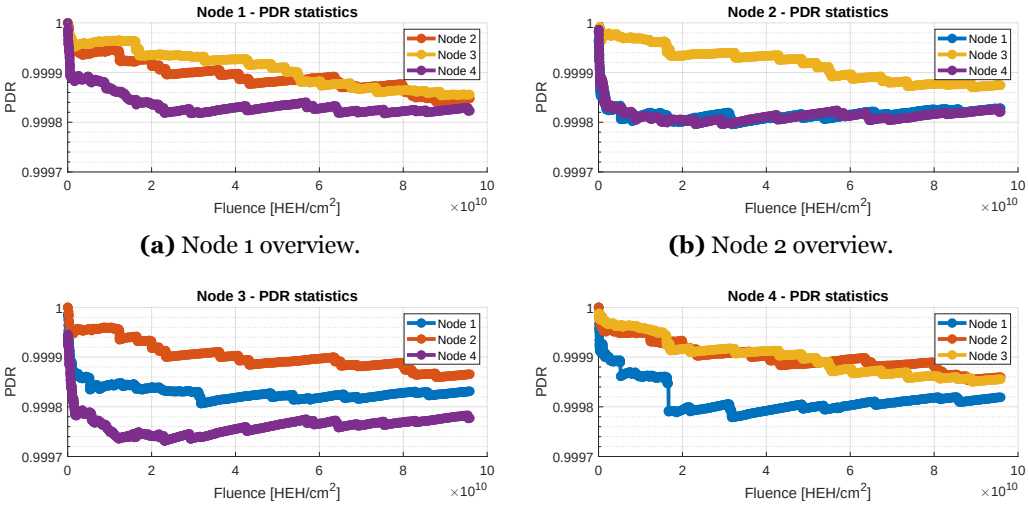
(b) Node 2 overview.

(c) Node 3 overview.

(d) Node 4 overview.

Figure 23. Overview of packet transmissions and receptions for the four nodes in the network when tested in CHARM.

The continuous monitoring of the transmitted and received packets within each node of the network allowed for the calculation of the PDR as a function of the fluence per each communication link in the network, as shown in Figure 24. In general, the PDR is above 99.99 %. The drops recorded in this parameter are due to lost packets, mainly as a consequence of failures induced by radiation to the electronic devices operating as nodes of the network. It is important to consider that the PDR is accumulating information on transmitted and lost packets since the beginning of the experiment: the impact of a single lost packet is higher at early moments, when the total amount of transmitted packets is lower.



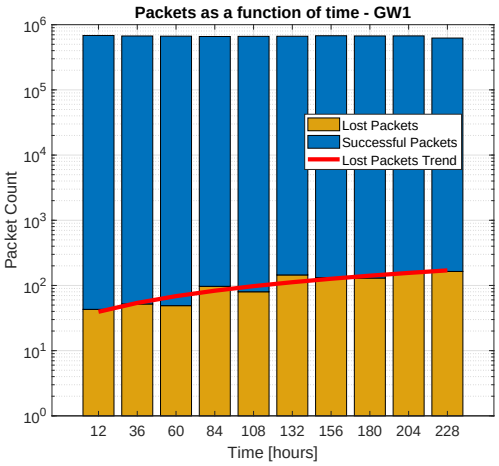
(a) Node 1 overview. **(b)** Node 2 overview.

(c) Node 3 overview. **(d)** Node 4 overview.

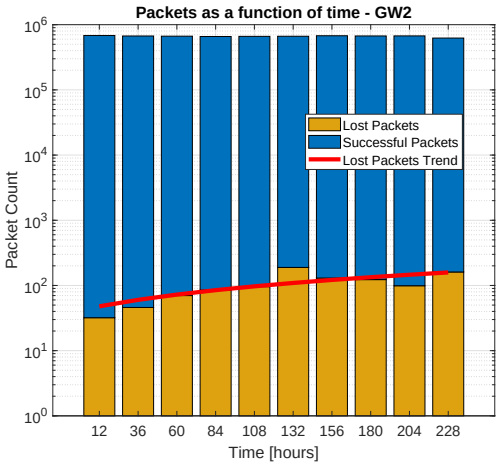
Figure 24. Overview of the PDRs for the four nodes in the tested network when tested in CHARM.

The performance of the network slightly degraded with time in terms of lost packets. Figure 25 shows the number of total transmitted and lost packets of both gateways in the network, in bins of 24 hours. Figure 25a shows that the amount of lost packets increases by 0.6058 packets per hour for GW1, increasing 14.5 lost packets per day. Similarly, Figure 25b shows that the amount of lost packets increases by 0.5114 packets per hour for GW2, increasing 12.3 lost packets per day.

The performance of the network is also described by parameters such as the throughput, packet ratio, RTT, and RSSI. Table 4 shows the mean RTT and mean RSSI of both gateways (as it is not possible to measure them for nodes N2 and N3), as well as the throughput and packet ratio for every node in the network. These parameters are measured and calculated for three intervals of time: a) the first 24 hours of the experiment (start), b) the last 24 hours of the experiment (end), and c) for the entire duration of the experiment (overall). The mean RTT, around 35 ms, for both gateways is slightly improved in the end compared to the beginning of the test. This is assumed to be because



(a) Total transmitted (blue) and lost packets (yellow) of GW1.



(b) Total transmitted (blue) and lost packets (yellow) of GW2.

Figure 25. Total transmitted and lost packets to every node in the network (in logarithmic scale), over time, represented in 24-hour bins. A trend line of the increase in lost packets over time is plotted in red.

when the network is forming at the start, all the routes and links need to be found, causing more delays in the network. In respect of the RSSI, this was around -70 dB during the run. In the case of GW2, the RSSI slightly improved during the run.

Table 4. Evaluation metric parameters.

Metric		GW 1	Node 2	Node 3	GW 2
Mean RTT [ms]	start	40	-	-	35
	overall	33			34
	end	32			34
Mean RSSI [dB]	start	-70	-	-	-73
	overall	-70			-68
	end	-70			-61
Throughput [pkt/s]	start	7.908	7.908	7.908	7.908
	overall	7.748	7.748	7.748	7.748
	end	7.733	7.731	7.732	7.733
Packet Ratio[pkt/s]	start	7.909	7.909	7.909	7.909
	overall	7.749	7.749	7.749	7.749
	end	7.735	7.732	7.734	7.734
Overall Network Throughput [pkt/s]	start	31.632			
	overall	30.992			
	end	30.929			
Overall Network Load [pkt/s]	start	31.636			
	overall	30.997			
	end	30.937			
Current Consumption [mA]	start	38.4	29.5	29.5	39.8
	overall	38.2	31.7	29.5	39.3
	end	37.6	35.0	29.2	39.6

The packet ratio, which represents the amount of total packets transmitted over time,

as well as the throughput, which represents the amount of successfully transmitted packets, was similar for every node in the network, being around 7.8 packets per second. Consequently, the overall network load, taking into account the packet ratio of each node, and the overall throughput are around 31 packets per second. It is important to mention that there is a slight degradation of these parameters over time, probably caused by the lost packets and power cycling events due to radiation.

4.3 Radiation Induced Effects on the WMN

The radiation environment affected the electronic devices, mainly through non - destructive SEEs (SET, SEU, and SEFI) that caused temporary failures and non-functionality, requiring a power cycle of the nodes. Figure 26 shows all the power cycling events recorded for each node in the network as a function of the fluence. As SEEs are stochastic events, it is unlikely that two or more nodes fail simultaneously. The measured cross-sections for gateways 1 and 2 (nodes 1 and 4, respectively), shown in Table 5, can be used to estimate the probability of simultaneous failure of both gateways. This probability is on the order of $9.1 \cdot 10^{-21}$, and therefore the availability of the network is approximately 100%.

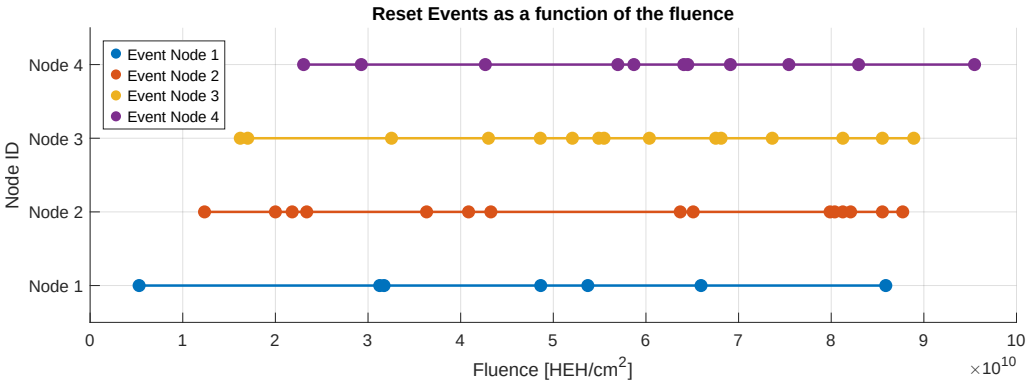


Figure 26. Power cycling events done for each node in the network as a function of the fluence.

Table 5 shows the number of power cycling events as well as the cross-section values for each node in the network and the network as a system. As the cross-section values depend on the hardware, these values are only a reference for this specific case. For instance, with other platforms, such as AstroMesh, the cross-section is expected to be higher due to the increased complexity of the MCU and the resources utilized.

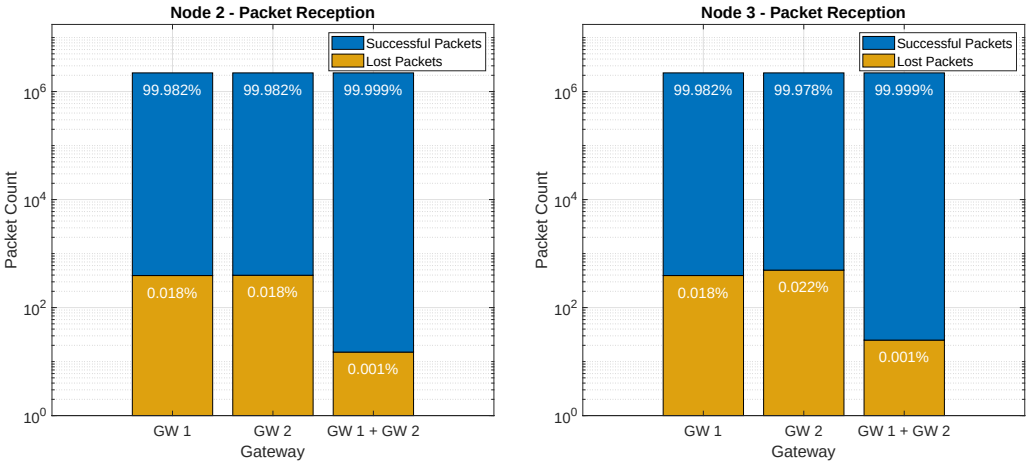
The tested WMN exhibited some multihop links at specific times; however, this behavior was unexpected and unnecessary given the small number of nodes in the network and their proximity. A low-level firmware analysis of this behavior is recommended, as it is likely due to a flaw in the library's handling procedure, which should be revised to enhance reliability.

Table 5. Power cycling events and cross-section values for each node in the network, and the network overall.

Node ID	Power Cycling Events	σ-Cross-section [cm^2 per device]		
		Lower	Average	Upper
1	7	$2.42 \cdot 10^{-11}$	$7.30 \cdot 10^{-11}$	$1.53 \cdot 10^{-10}$
2	15	$7.31 \cdot 10^{-11}$	$1.56 \cdot 10^{-10}$	$2.68 \cdot 10^{-10}$
3	15	$7.31 \cdot 10^{-11}$	$1.56 \cdot 10^{-10}$	$2.68 \cdot 10^{-10}$
4	11	$4.77 \cdot 10^{-11}$	$1.15 \cdot 10^{-10}$	$2.11 \cdot 10^{-10}$
Network	48	$7.52 \cdot 10^{-11}$	$1.25 \cdot 10^{-10}$	$1.81 \cdot 10^{-10}$

4.3.1 Multiple Gateways Mitigation Approach

The approach of having multiple gateways provides redundancy and failover capabilities, resulting in more robustness under radiation exposure. This means that when one of the gateways fails, the other gateway can still acquire the data from the end nodes. The increased robustness of this system is shown in Figure 27, in which the number of packets transmitted by each of the end nodes and received by each gateway is represented. While the gateways individually failed to receive around 0.02 % of the total transmitted packets of each node (left bar corresponds to packets acquired by GW1 (Node 1), and the middle bar corresponds to the packets acquired by GW2 (Node 4)), when the packets acquired are seen as a combination of both gateways, the amount of single packet lost is reduced to 0.001 % in both cases.



(a) Packet reception statistic for Node 2. **(b)** Packet reception statistic for Node 3.

Figure 27. Packet reception statistics for both end nodes: Node 2 and Node 3.

Therefore, having two gateways increases the robustness of the network in terms of acquiring data. This scheme resulted in a reduced packet loss by a factor of 10, reaching a similar performance to the non-irradiated network in terms of PDR. However, most of the packets lost by both gateways occurred when a gateway was affected by a SEE, and the communication with the other gateway also failed. This is because, with this

configuration, when a node of the network fails, all the other nodes are still attempting to communicate with it using all the mechanisms of retrials and timeouts, making the network more susceptible to communication issues. This can be solved by a more robust protocol.

4.3.2 Radiation Induced Failures on the Network

As explained previously, radiation can affect the performance and behavior of electronic devices, and this is the case for the tested wireless mesh network deployed in CHARM. Radiation mainly affected the electronic devices acting as nodes of the network by non-destructive SEE such as SETs, SEUs, and SEFIs. In this work, some radiation-induced failures are identified at the system level of the tested devices. Some Software Mitigation Schemes for MCU-based designs discussed in [45] were applied. The three main type of failures that were identified during the test campaign are listed below:

- **Transmission stop**: due to radiation, the transceiver stops transmitting, although the MCU, which is not aware of the issue, continues its normal operation. This results in a node that seems to be working, but it is not able to communicate with the other nodes in the network. Therefore, the control script supervises the actual nodes participating in the network. To mitigate this failure, as explained in [45], if a node is not able to communicate after a certain amount of time, it is power-cycled.

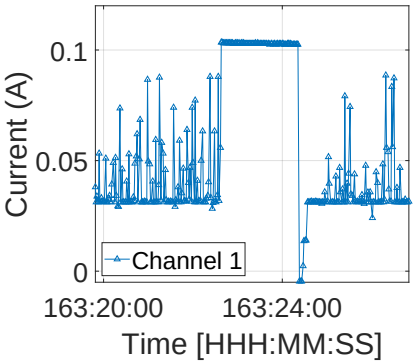
- **Blocked device**: due to radiation, the device is stuck at some point in its operation and becomes unresponsive [45]. This can be due to bit flips in sensitive parts that cause the device to be blocked, and therefore is not able to communicate with the other nodes in the network. In this case, the node is power cycled after a certain time during which the node is not able to communicate with the other nodes in the network. To mitigate these errors, an external or software watchdog is needed, which triggers a reboot or power cycle when this event occurs. Figure 28 and 29 show some examples of the current monitoring information, previous to the power cycle of a node for the tested network.

- **Bit Flips**: radiation can cause bit flips in both register configurations and device configurations. When a bit state changes in the SRAM or register configuration, it is considered a **Soft Bit Flip**, which can affect the transmitted information, counter registers, packet headers, packet payloads, control logic, and internal node registers such as its ID or routing table. The impact of this type of failure depends on the bit affected.

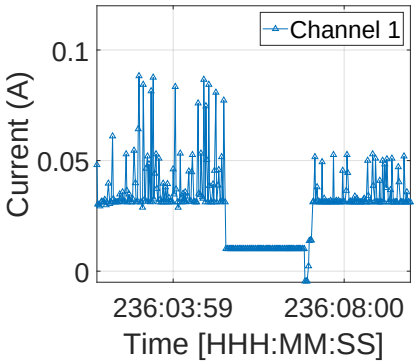
  A bit flip on a counter register can be identified and treated in the data analysis stage, while a bit flip in the node ID directly affects the communication, as if a node transmits with a wrong ID, the receiver nodes are not aware of it.

To mitigate soft bit flips, techniques such as Hamming error correction, storing critical parameters in external flash, and applying cyclic redundancy checks (CRC) can be implemented. If the MCU's SRAM is equipped with ECC, Software Triple Modular Redundancy (TMR) can be considered, ensuring that variables are placed in non-adjacent memory locations to reduce the risk of simultaneous impact from multi-cell upsets.

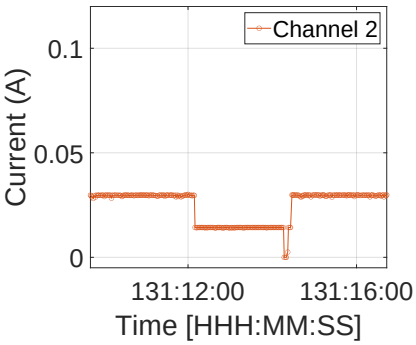
Hard Bit Flips refer to more sensitive bit flips affecting memory cells or configuration registers in the non-volatile storage. These failures require powering the device to restore its normal operation.



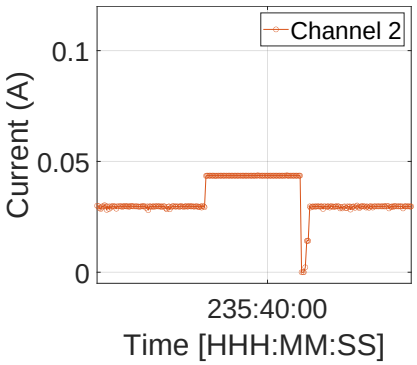
(a) Node 1 power cycle case.



(b) Node 1 power cycle case.

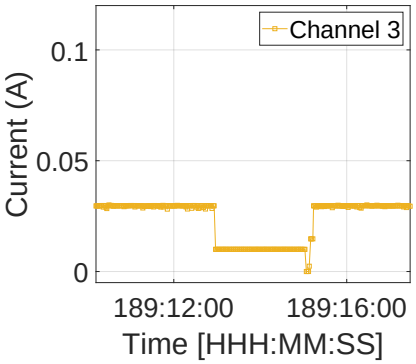


(c) Node 2 power cycle case.

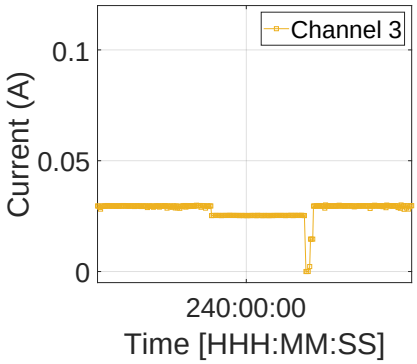


(d) Node 2 power cycle case.

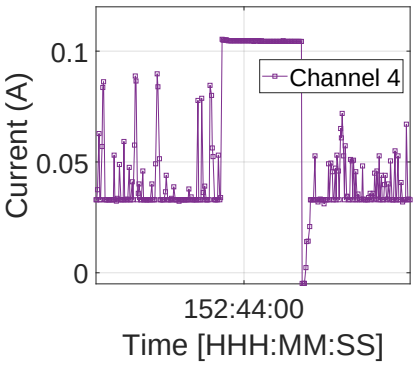
Figure 28. Current plots for some of the power cycling cases for nodes 1 and 2.



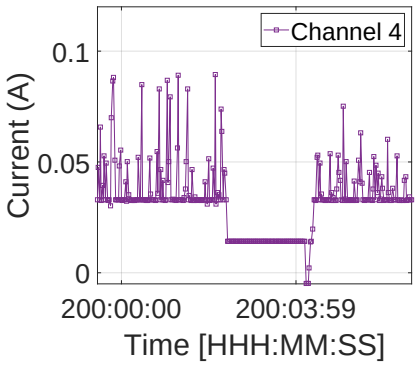
(a) Node 3 power cycle case.



(b) Node 3 power cycle case.



(c) Node 4 power cycle case.



(d) Node 4 power cycle case.

Figure 29. Current plots for some of the power cycling cases for nodes 3 and 4.

5 Conclusion

5.1 Conclusion

This work presents a first step towards the integration of reliable and robust wireless mesh networks in radiation environments. This thesis has presented the design, implementation, and experimental validation of a mesh network architecture based on COTS components, tested at the CHARM radiation facility.

The resulting proof of concept validated the feasibility of WMN for applications in harsh environments such as particle accelerators and space environments. The tested mesh network, composed of decentralized nodes based on COTS components and implemented using the open-source RadioHead library, has exhibited a robust and resilient performance during irradiation. The performance evaluation under radiation demonstrated a high PDR (greater than 99.9 %) throughout the entire test campaign. However, radiation-induced fault events in the nodes, mainly identified as SEE, require power cycles to restore the normal operation of the nodes. The main source of lost packets during the experiment is the radiation effects on the devices. Moreover, an increase in the packet loss over time, around 13 packets per day, was observed, suggesting a small degradation in the network performance caused by radiation effects. These results also suggest the development of mitigation strategies such as node redundancy, error correction, and active monitoring.

In addition to the overall good performance of the network, the redundancy strategy of multiple gateways allows for failover capabilities, which enhances the reliability and robustness under radiation exposure. This improvement in resilience provided by the dual gateway scheme shows the importance of node redundancy. Even in small-scale networks, using two gateways enhances the availability and connectivity of the network. In terms of the end nodes lost packets, the performance of the test showed that using the multiple gateway mitigation scheme under radiation conditions resulted in a decrease of lost packets by a factor of 10, showing a similar performance to the network tested outside the radiation environment.

The experimental methodology for testing the wireless mesh networks in radiation environments developed in this work provides a starting point for standardizing the evaluation of wireless networks under radiation.

5.2 Future work

Although still in early stages, the tested network architecture demonstrated strong potential for deployment in harsh environments. Therefore, future work remains necessary, focusing on testing different configurations and topologies, scaling the network, implementing a custom ad hoc protocol for application-specific requirements, and im-

proving fault recovery and error correction mechanisms.

Moreover, the project has to be moved to a C++ environment, instead of the Arduino environment, to have complete control of the network operation, being able to modify and add functions tailored for specific applications. In this context, it would be interesting to develop a standard to fulfill specific environment requirements, such as for space and the FCC. It is important to consider that both scenarios have different needs, and therefore, individual protocols are required.

References

[1] Ian F Akyildiz, Xudong Wang, and Weilin Wang. “Wireless mesh networks: a survey”. In: *Computer networks* 47.4 (2005), pp. 445–487.

[2] Sudip Misra, Subhas Chandra Misra, and Isaac Woungang. *Guide to wireless mesh networks*. Vol. 9. 9. Springer, 2009.

[3] Antonio Cilfone et al. “Wireless mesh networking: An IoT-oriented perspective survey on relevant technologies”. In: *Future internet* 11.4 (2019), p. 99.

[4] Ram Ramanathan and Jason Redi. “A brief overview of ad hoc networks: challenges and directions”. In: *IEEE communications Magazine* 40.5 (2002), pp. 20–22.

[5] Zhengyang Cao. “Optimization Design of Multi-UAV Communication Network Based on Reinforcement Learning”. In: *Wireless Communications and Mobile Computing* 2022.1 (2022), p. 7726338.

[6] Jonathan Loo, Jaime Lloret Mauri, and Jesús Hamilton Ortiz. “Mobile ad hoc networks: current status and future trends”. In: (2011).

[7] DG Reina et al. “A survey on multihop ad hoc networks for disaster response scenarios”. In: *International Journal of Distributed Sensor Networks* 11.10 (2015), p. 647037.

[8] Ashish Srivastava and Jay Prakash. “Future FANET with application and enabling techniques: Anatomization and sustainability issues”. In: *Computer Science Review* 39 (2021), p. 100359. ISSN: 1574-0137. DOI: https://doi.org/10.1016/j.cosrev.2020.100359. URL: https://www.sciencedirect.com/science/article/pii/S1574013720304597.

[9] Jesus Sanchez-Gomez, Ramon Sanchez-Iborra, and Antonio Skarmeta. “Transmission Technologies Comparison for IoT Communications in Smart-Cities”. In: *GLOBECOM 2017 - 2017 IEEE Global Communications Conference*. 2017, pp. 1–6. DOI: 10.1109/GLOCOM.2017.8254530.

[10] Md Mahmud and Sohag Sarker. “EFFECT ON PATH LOSS MODEL ON PERFORMANCE OF 5G COMPATIBLE MIMO WINDOW-OFDM SYSTEM”. PhD thesis. Dec. 2019.

[11] Aloÿs Augustin et al. “A Study of LoRa: Long Range & Low Power Networks for the Internet of Things”. In: *Sensors* 16.9 (2016). ISSN: 1424-8220. DOI: 10.3390/s16091466. URL: https://www.mdpi.com/1424-8220/16/9/1466.

[12] Raffaele Bruno, Marco Conti, and Enrico Gregori. “Mesh networks: commodity multihop ad hoc networks”. In: *IEEE communications magazine* 43.3 (2005), pp. 123–131.

[13] Mohit Kumar and Rashmi Mishra. “An overview of MANET: History, challenges and applications”. In: *Indian Journal of Computer Science and Engineering (IJCSE)* 3.1 (2012), pp. 121–125.

[14] Ahmed M Eltahlawy et al. “A survey on parameters affecting MANET performance”. In: *Electronics* 12.9 (2023), p. 1956.

[15] Tri Kuntoro Priyambodo, Danur Wijayanto, and Made Santo Gitakarma. “Performance optimization of MANET networks through routing protocol analysis”. In: *Computers* 10.1 (2020), p. 2.

[16] Fahim Maan and Nauman Mazhar. “MANET routing protocols vs mobility models: A performance evaluation”. In: *2011 Third international conference on ubiquitous and future networks (ICUFN)*. IEEE. 2011, pp. 179–184.

[17] Jack L Burbank et al. “Key challenges of military tactical networking and the elusive promise of MANET technology”. In: *IEEE Communications Magazine* 44.11 (2006), pp. 39–45.

[18] Felipe Cunha et al. “Data communication in VANETs: Protocols, applications and challenges”. In: *Ad hoc networks* 44 (2016), pp. 90–103.

[19] Scott Miller et al. “Smallsat 2024-Starling Cubesat Swarm Technology Demonstration Flight Results”. In: *Proceedings of the 38th Annual Small Satellite Conference*. SSC24-I-06. Small Satellite Conference. 2024.

[20] Robert Szewczyk et al. “An analysis of a large scale habitat monitoring application”. In: *Proceedings of the 2nd international conference on Embedded networked sensor systems*. 2004, pp. 214–226.

[21] Huang-Chen Lee and Kai-Hsiang Ke. “Monitoring of large-area IoT sensors using a LoRa wireless mesh network system: Design and evaluation”. In: *IEEE Transactions on Instrumentation and Measurement* 67.9 (2018), pp. 2177–2187.

[22] Jithu G Panicker, Mohamed Azman, and Rajiv Kashyap. “A LoRa wireless mesh network for wide-area animal tracking”. In: *2019 IEEE International Conference on Electrical, Computer and Communication Technologies (ICECCT)*. IEEE. 2019, pp. 1–5.

[23] Qazawat Zirak, Dmitriy Shashev, and Stanislav Shidlovskiy. “Swarm of drones using LoRa flying ad-hoc network”. In: *2021 International Conference on Information Technology (ICIT)*. IEEE. 2021, pp. 400–405.

[24] Misbahuddin Misbahuddin et al. “EAM-LoRaNet: Energy aware multi-hop LoRa network for Internet of Things”. In: *KINETIK: Game Technology, Information System, Computer Network, Computing, Electronics, and Control* (2022), pp. 81–90.

[25] Bing Wu et al. “A survey of attacks and countermeasures in mobile ad hoc networks”. In: *Wireless network security* (2007), pp. 103–135.

[26] Fabrizio Granelli et al. “Mobile Lightweight Wireless Systems”. In: (2009).

[27] Adnan Nadeem and Michael P Howarth. “A survey of MANET intrusion detection & prevention approaches for network layer attacks”. In: *IEEE communications surveys & tutorials* 15.4 (2013), pp. 2027–2045.

[28] Priyanka Goyal, Vinti Parmar, Rahul Rishi, et al. “Manet: vulnerabilities, challenges, attacks, application”. In: *IJCEM International Journal of Computational Engineering & Management* 11.2011 (2011), pp. 32–37.

[29] Alessandro Zimmaro. “Radiation effects and System Level Testing: Application to Wireless Communications”. PhD thesis. Université de Montpellier, 2024.

[30] Jasper Dijks. “Precise LEO Space Radiation Monitoring using the Space RadMon-NG Payload”. PhD thesis. Delft Tech. U.

[31] Rudy Ferraro. “Development of Test Methods for the Qualification of Electronic Components and Systems Adapted to High-Energy Accelerator Radiation Environments”. Theses. Université Montpellier, Dec. 2019. URL: https://theses.hal.science/tel-02879667.

[32] Andrea Coronetti. “Relevance and guidelines of radiation effect testing beyond the standards for electronic devices and systems used in space and at accelerators”. PhD thesis. University of Jyvaskyla (FI), 2021.

[33] Ewa Lopienska. “The CERN accelerator complex, layout in 2022”. In: (2022). General Photo. URL: https://cds.cern.ch/record/2800984.

[34] CERN. *Accelerator Report: LHC Run 3 achieves record-breaking integrated luminosity*. accessed Aug. 15, 2025. Sept. 2024. URL: https://home.cern/news/news/accelerators/accelerator-report-lhc-run-3-achieves-record-breaking-integrated-luminosity.

[35] M Benedikt et al. *Future Circular Collider Feasibility Study Report: Volume 1, Physics, Experiments, Detectors*. Tech. rep. Geneva: CERN, 2025. DOI: 10.17181/CERN.9DKX.TDH9. arXiv: 2505.00272. URL: https://cds.cern.ch/record/2928193.

[36] M Benedikt et al. *Future Circular Collider Feasibility Study Report: Volume 2, Accelerators, Technical Infrastructure and Safety*. Tech. rep. Geneva: CERN, 2025. DOI: 10.17181/CERN.EBAY.7W4X. arXiv: 2505.00274. URL: https://cds.cern.ch/record/2928793.

[37] S Agosta, R Sierra, and F Chapron. "High-Speed Mobile Communications in Hostile Environments". In: *Journal of Physics: Conference Series* 664.5 (Dec. 2015), p. 052001. DOI: 10.1088/1742-6596/664/5/052001. URL: https://dx.doi.org/10.1088/1742-6596/664/5/052001.

[38] Salvatore Danzeca et al. "Wireless IoT in Particle Accelerators: A Proof of Concept with the IoT Radiation Monitor at CERN". In: *JACoW IPAC* 2022 (2022), pp. 772–775. DOI: 10.18429/JACoW-IPAC2022-TUOXGD2. URL: https://cds.cern.ch/record/2845851.

[39] Rodrigo Sierra and Hubert Odziemczyk. "Readying CERN for connected device era". In: *EPJ Web Conf.* 245 (2020), p. 07015. DOI: 10.1051/epjconf/202024507015. URL: https://cds.cern.ch/record/2752529.

[40] J Mekki et al. "CHARM: A mixed field facility at CERN for radiation tests in ground, atmospheric, space and accelerator representative environments". In: *IEEE Transactions on Nuclear Science* 63.4 (2016), pp. 2106–2114.

[41] Jasper Dijks et al. "Design, Validation, and Characterization of CERN's SpaceRadMon-NG CubeSat Payload for Space Radiation Monitoring Missions". In: *IEEE Transactions on Aerospace and Electronic Systems* (2025), pp. 1–10. DOI: 10.1109/TAES.2025.3595559.

[42] Microchip Technology Inc. *ATSAMD21G18A Datasheet – SMART ARM-Based Microcontroller*. Rev. K. Mar. 2025. URL: https://ww1.microchip.com/downloads/aemDocuments/documents/MCU32/ProductDocuments/DataSheets/SAM-D21-DA1-Family-Data-Sheet-DS40001882.pdf.

[43] HopeRF Electronics. *Low Power Integrated UHF Transceiver with On-Chip +20dBm PA*. 2006. URL: https://www.hoperf.com/ic/rf_transceiver/RF69W.html#apDocument (visited on 07/20/2025).

[44] Mike McCauley. *RadioHead Packet Radio library for embedded microprocessors*. https://www.airspayce.com/mikem/arduino/RadioHead/. [Online; accessed 20-July-2025]. 2025.

[45] A. Zimmaro et al. "Using Software Mitigation Schemes to improve the availability of IoT applications in harsh radiation environment". In: *Journal of Instrumentation* 19.02 (Feb. 2024), p. C02059. DOI: 10.1088/1748-0221/19/02/C02059. URL: https://dx.doi.org/10.1088/1748-0221/19/02/C02059.

A Appendix: Python monitoring code

The WMN test conducted at CHARM was monitored with a Python script to make logs of the network packets, power cycling events, and current consumption.

```
import time
import serial
from IPython.display import clear_output
import time
import threading
from datetime import datetime
import pyvisa
import os
from Instruments.keysight_n6705b import KeysightN6705B

UART_PORT1 = "COM17"     # Node 1 UART port
UART_PORT2 = "COM18"     # Node 2 UART port
baudrate = 9600              # baudrate
INACTIVITY_TIMEOUT = 120        # Seconds for inactivity timeout (all nodes)
INACTIVITY_TIMEOUT_NODE_1 = 100 # Seconds for inactivity timeout (Gateways only)
OFF_TIMEOUT = 20        # Seconds for inactivity timeout (all nodes)
NODES = {1: 1, 2: 2, 3: 3, 4: 4}     # Mapping node ID with PSU channel

# PSU Channels
CHANNEL_1 = 1
CHANNEL_2 = 2
CHANNEL_3 = 3
CHANNEL_4 = 1 # this will be in the other PSU (select good channel)

# Variables to monitor the node activity (last time seen)
node_last_seen = {} # Contains the last seen time for each node
node_last_ack1 = {}  # Contains the last ACK time for node 1 (Gateway 1)
node_last_ack4 = {}  # Contains the last ACK time for node 4 (Gateway 2)
node_last_ack = {}  # Contains the last ACK time for each node

dir = os.path.dirname(os.path.abspath(__file__))
log_g1 = os.path.join(dir, "gateway1_data.txt")
log_g2 = os.path.join(dir, "gateway2_data.txt")
log_current = os.path.join(dir, "CURRENT_data.txt")
reset_log = os.path.join(dir, "RESETS_data.txt")

# PSU initialization
rm = pyvisa.ResourceManager('@py')
psu_1 = KeysightN6705B("TCPIP::192.168.0.160::INSTR")  # Replace with the
    correct IP address
psu_2 = KeysightN6705B("TCPIP::192.168.0.161::INSTR")  # Replace with the
    correct IP address
print("Connected to PSU")
```

```
42
43 # Set voltage to 5V and current limit to 0.3A for all channels
44 psu_1.set_voltages([CHANNEL_1], 5)
45 psu_1.set_voltages([CHANNEL_2, CHANNEL_3], 5)
46 psu_1.set_currents([CHANNEL_1, CHANNEL_2, CHANNEL_3], 0.3)
47 psu_1.set_outputs([CHANNEL_1, CHANNEL_2, CHANNEL_3], True)
48 psu_2.set_voltages([CHANNEL_4], 5)
49 psu_2.set_currents([CHANNEL_4], 0.3)
50 psu_2.set_outputs([CHANNEL_4], True)
51 time.sleep(10)
52
53 def restart_node(node_id): # power cycles (0V - 5V) the PSU channel for a single
       node, and log the events
54     if node_id == 4:
55         channel = CHANNEL_4
56         log_reset_event(node_id, reset_log)
57         psu_2.set_voltages([channel], 0)
58         time.sleep(4)
59         psu_2.set_voltages([channel], 5)
60
61     elif node_id == 1:
62         channel = NODES[node_id]
63         log_reset_event(node_id, reset_log)
64         psu_1.set_voltages([channel], 0)
65         time.sleep(4)
66         psu_1.set_voltages([channel], 5)
67
68     elif node_id in NODES:
69         channel = NODES[node_id]
70         log_reset_event(node_id, reset_log)
71         psu_1.set_voltages([channel], 0)
72         time.sleep(4)
73         psu_1.set_voltages([channel], 5)
74     else:
75         return
76
77
78 # Read UART lines: the function parses the incoming line from UART and extracts
       node IDs and ACK values
79 def parser(line):
80     line = line.strip()
81     tokens = line.split(",")
82     nodes = set()
83     ack_values = {}
84     if len(tokens) < 12:  # Check the line has enough tokens
85         return nodes, ack_values
86
87     sender_str = tokens[0]
```

```
88      sender = int(sender_str)
89      receiver_str = tokens[1]
90      receiver = int(receiver_str)
91      nodes.add(sender)
92
93      if sender == 1 or sender == 4:
94          for i in range(5, 15, 3):  # Extract each ACK and HOP values
95              ack_value = int(tokens[i]) #ACK values
96              hop = int(tokens[i - 2])  #HOP values
97              if hop != 0 and hop != 255:     # if hop is different to 0 or 255,
    add hop value to nodes
98                  nodes.add(hop) # if valid hop, add to nodes
99                  ack_values[hop] = ack_value
100
101     else: #similar function but for the nodes that are not gateways
102         for i in range(5, 15, 3):
103             hop = int(tokens[i - 2])
104             if hop != 0 and hop != 255:
105                 nodes.add(hop)
106
107     return nodes, ack_values
108
109 # Save the current values in the txt file
110 def log_currents_txt(p1, p2, filename):
111     #Get current from both PSUs
112     currents1 = p1.get_currents([CHANNEL_1, CHANNEL_2, CHANNEL_3])
113     current_values1 = []
114     for x in currents1.split(","):
115         current_values1.append(x.strip())
116     currents2 = p2.get_currents([CHANNEL_4])
117     current_values2 = []
118     for x in currents2.split(","):
119         current_values2.append(x.strip())
120     timestamp = datetime.now()
121     timestamp = timestamp.strftime("%Y-%m-%d %H:%M:%S")
122
123     total_currents = current_values1 + current_values2
124
125     #Write data to the txt file
126     with open(filename, "a") as file:
127         file.write(f"{timestamp},Currents,{', '.join(total_currents)}\n")
128
129 # Store the reset events of all the nodes in the network in a file
130 def log_reset_event(node_id, filename):
131     now = datetime.now()
132     timestamp = now
133     timestamp = timestamp.strftime("%Y-%m-%d %H:%M:%S")
134     with open(filename, "a") as file:
```

```
        file.write(f"{timestamp} _ Node {node_id} reset\n")
```

```
# define different threads to operate in parallel

#thread for saving the current values
def log_psu_readings_txt():
    while True:
        log_currents_txt(psu_1, psu_2, log_current)
        time.sleep(0.8)

# thread for monitroing UART port and try connecting to them
def monitor_uart(port, baudrate, log_filename):
    ser = None
    while True:
        clear_output(wait=True)
        try:
            if ser is None:
                ser = serial.Serial(port, baudrate, timeout=1)
                time.sleep(0.1)
            if ser is not ser.is_open:
                ser = serial.Serial(port, baudrate, timeout=1)
                time.sleep(0.1)
            line = ser.readline()
            now = time.time()
            with open(log_filename, "a") as logfile:
                now = datetime.now()
                aa = f"{line},{now}\n"
                logfile.write(aa)
                logfile.flush()

            nodes_seen = parser(line)
            for node in nodes_seen:
                node_last_seen[node] = now  # Update seen time

            if len(nodes_seen) != 1: #update node 1 and 4 (the gateways)
                for node in nodes_seen:
                    if node == 1:
                        node_last_ack1[node] = now
                    if node == 4:
                        node_last_ack4[node] = now

        except serial.SerialException as e:
            if ser is not None:
```

```
                ser.close()
            ser = None
            time.sleep(5)


#thread for monitoring the nodes (power cycle if needed)
def watch_nodes():
    global node_last_seen , node_last_ack, node_last_ack1, node_last_ack4

    if not node_last_seen:
        now = time.time()
        for node in NODES.keys(): # Initialize timers for all nodes in the first iteration, so the timers start working
            node_last_seen[node] = now

    while True:
        now = time.time()
        for node, last_seen in list(node_last_seen.items()):
            if node == 1 or node == 4:
                timerplus = INACTIVITY_TIMEOUT - INACTIVITY_TIMEOUT_NODE_1 # reduction of timer for NODE 1 --> restart timer for NODE 2 and NODE 3 ( avoid unnecessary restarts)
            else:
                timerplus = 0

            if now - last_seen > INACTIVITY_TIMEOUT - timerplus:
                restart_node(node)
                node_last_seen[node] = now  # Reset timer after restart

                if node == 1 or node == 4: # If GW 1 or GW 2 is restarted, reset timers for all nodes
                    node_last_seen[1] = now
                    node_last_seen[2] = now
                    node_last_seen[3] = now
                    node_last_seen[4] = now
                    now = time.time()

        for node in list(node_last_ack.items()):
            if node in node_last_seen and now - node_last_seen[node] > INACTIVITY_TIMEOUT - timerplus:
                restart_node(node)
                node_last_seen[node] = now  # Reset timer after restart

        if now - node_last_ack1 > INACTIVITY_TIMEOUT: # Restart Node 1 if in the prints of N1, it is not able to communicate
            restart_node(1) #power cycle Node 1 (G1)
        if now - node_last_ack4 > INACTIVITY_TIMEOUT:
            restart_node(4) #power cycle Node 4 (G2)
```

```
225
226         time.sleep(5)
227
228
229 # Main
230 if __name__ == "__main__":
231     # Start the threads: monitoring UART, log PSU and log nodes power cycles
    events
232     threading.Thread(target=monitor_uart, args=(UART_PORT1, baudrate, log_g1),
    daemon=True).start()
233     threading.Thread(target=monitor_uart, args=(UART_PORT2, baudrate, log_g2),
    daemon=True).start()
234     threading.Thread(target=watch_nodes, args=(), daemon=True).start()
235     threading.Thread(target=log_psu_readings_txt, daemon=True).start()
236
237     try:
238         while True:
239             time.sleep(1)
240     except KeyboardInterrupt: #when ESC stop the program and turn off PSU
    channels
241         psu_1.set_outputs([CHANNEL_1, CHANNEL_2, CHANNEL_3], False)
242         psu_2.set_outputs([CHANNEL_4], False)
```

B Appendix: MATLAB Code

The data analysis of the WMN was done using MATLAB. This appendix presents the code used to import, process, and plot the WMN data.

After importing as a *.mat* file the *.txt* file with the network information of a gateway and the *.csv* file with the radiation data from CHARM, the data is prepared to be analyzed. For this example, *DATA_gateway1.mat* and *rad_DATA_W22.mat* are the imported files of the gateway and radiation data respectively.

```
%Load data: network data and radiation data
load('DATA_gateway1.mat')
HEHdata1=load('rad_DATA_W22.mat');
HEHdata1 = HEHdata1.rad_DATA;

featherM0DATAgateway1 = featherM0DATAgateway1week1;

%First I will remove lines with wrong or corrupted timestamp values (VarNamw19
    is the column containing rhe timestamp)
isnat_idx = find(isnat(featherM0DATAgateway1.VarName19)==1);
featherM0DATAgateway1(isnat_idx,:) = [];

%Then I interpolate the radiation (HEH fluence) data to match the gateway
    timestamps values
time_rad = HEHdata1.Time;
heh_rad = HEHdata1.HEH;
[time_rad, idx] = unique(time_rad);
heh_rad = heh_rad(idx);
time = featherM0DATAgateway1.VarName19;
time.TimeZone = 'Europe/Zurich';

% Add the interpolated HEH column after the timedate column -> VarName20
featherM0DATAgateway1.VarName20 = interp1(time_rad, heh_rad, time);
featherM0DATAgateway1.VarName20(find(isnat(featherM0DATAgateway1.VarName20))) =
    0; %if a time is not a time make it 0
```

When the data acquired from the gateway is imported and merged with the radiation data, a struct that contains a table for each node participating as a sender is created.

```
% Identify all the different sender node IDs participating in the network:
    sender values are in VarName1 column
senders = unique(featherM0DATAgateway1.VarName1);

% Initialize a struct to store tables for each sender
senderStruct = struct();

% Loop to create tables for each sender and store it in the struct
for i = 1:length(senders)
    senderID = senders(i);
```

```
    if isnan(senderID) || senderID == 36
        continue  % Skip the corrupted sender values
    end
    senderName = sprintf("Sender_%d", senderID);  % Name of the sender ID table
    senderStruct.(senderName) = featherM0DATAgateway1(featherM0DATAgateway1.
    VarName1 == senderID, :);  % Filter row
    senderStruct.(senderName) = senderStruct.(senderName)(senderStruct.(
    senderName).VarName19 > '2025-05-28 12:00:00' & ...
                                senderStruct.(senderName).VarName19 ...
    < '2025-06-03 08:00:00', :);  % Filter row by timestamps
end

% Save the created struct in a .mat file
save('senderStruct1.mat', 'senderStruct', '-v7.3');
```

Once the struct is created, the information is checked for errors and cleaned if needed.

```
%Load structure
load("senderStruct1.mat", "senderStruct");
%Extract parameters of interest: Send1, ACK1, Send2, ACK2, Send3, ACK3, Send4,
    ACK4,
parameters = {'VarName5','VarName6','VarName8', 'VarName9', 'VarName11', '
    VarName12', 'VarName14', 'VarName15'};

% Look for errors or missing fields
count_nan = 0;
idx = find(any(ismissing(senderStruct.Sender_4), 2));
senderStruct.Sender_4(idx.', :) = [];
count_nan = count_nan + length(idx);
idx = find(any(ismissing(senderStruct.Sender_2), 2));
senderStruct.Sender_2(idx.', :) = [];
count_nan = count_nan + length(idx);
idx = find(any(ismissing(senderStruct.Sender_3), 2));
senderStruct.Sender_3(idx.', :) = [];
count_nan = count_nan + length(idx);
idx = find(any(ismissing(senderStruct.Sender_1), 2));
senderStruct.Sender_1(idx.', :) = [];
count_nan = count_nan + length(idx);

count = 0;

% Get field names of the struct
senderFields = fieldnames(senderStruct);

% Loop over each field in the struct
for f = 1:length(senderFields)
    proc_table = senderStruct.(senderFields{f}); %Table of the sender node

    % Check for potential bit flips by checking a fast decrease in values
```

```
    allIdx = [];
    for i = 1:length(parameters)
        param = proc_table.(parameters{i});
        idx = find((diff(param) < 0) & (diff(param) ~= -255)) + 1;
        allIdx = [allIdx; idx]; % Store indices of corrupted data
    end
    uniqueIdx = unique(allIdx);

    count = count + length(uniqueIdx);
    proc_table(uniqueIdx, :) = []; % Remove flagged rows from table

    % Check for potential bit flips: fast increase (diff > 10)
    idx_flips = find( ...
        diff(proc_table.VarName5) > 10 | diff(proc_table.VarName6) > 10 | ...
        diff(proc_table.VarName8) > 10 | diff(proc_table.VarName9) > 10 | ...
        diff(proc_table.VarName11) > 10 | diff(proc_table.VarName12) > 10 | ...
        diff(proc_table.VarName14) > 10 | diff(proc_table.VarName15) > 10 );

    proc_table(idx_flips.', :) = []; % Remove flagged rows from table
    count = count + length(idx_flips);
    senderStruct.(fieldName) = proc_table; %Update table inside struct
end
```

```
% Load processed struct
load("senderStruct1_corrected.mat", "senderStruct");

% Define column names
parameter_name = ["Sender", "Receiver", "PacketCount", ...
                "Hop1", "Send1", "ACK1", ...
                "Hop2", "Send2", "ACK2", ...
                "Hop3", "Send3", "ACK3", ...
                "Hop4", "Send4", "ACK4", ...
                "Time", "Extra1", "Extra2", ...
                "timestamp", "HEH"];

senderNames = fieldnames(senderStruct); % Obtain name of tables in th estruct

for i = 1:length(senderNames)
    name = senderNames{i}; % Iterate for every sender in the struct
    senderStruct.(name).Properties.VariableNames = parameter_name; % Change name
     of variables
end
```

Functions to unwrap the counters are defined.

```
% Function to unwrap the counter
function [unwrappedPkt, unwrappedAck] = unwrap_resets(pkt, ack)
MAX_COUNTER = 255; % The counter is 1 byte --> 255 maximum value
MIN_GLITCH_DIFF = -1; % To tolerate a minimum single backward count
```

```
MIN_WRAP_JUMP = 255; % parameter to control the unwrap minimum jump
MAX_REASONABLE_DELTA = 240; % Max possible value of delta in a unwrap
MAX_GAP_JUMP = 18; % Maximum pkt-ack difference

% Initialize arrays
n = length(pkt);
unwrappedPkt = zeros(size(pkt));
unwrappedAck = zeros(size(ack));
unwrappedPkt(1) = 0;
unwrappedAck(1) = 0;

% Loop over the whole table
for i = 2:n
    prev_pkt = pkt(i-1); % Store previous pkt value
    curr_pkt = pkt(i); % Store current pkt value
    prev_ack = ack(i-1); % Store previous ack value
    curr_ack = ack(i); % Store current pkt value

    % If the current pkt and ack is 0, make delta equal to 1
    if curr_pkt == 0 && curr_ack== 0 && prev_pkt ~= 0 && prev_ack~= 0
        delta_pkt = 1;
        delta_ack = 1;

    else
        % If not compute deltas
        delta_pkt = compute_delta(prev_pkt, curr_pkt);
        delta_ack = compute_delta(prev_ack, curr_ack);
    end

    % Store temporary pkt and ack
    tmpPkt = unwrappedPkt(i-1) + delta_pkt;
    tmpAck = unwrappedAck(i-1) + delta_ack;

    % Check previous and current pkt-ack gap
    prevGap = unwrappedPkt(i-1) - unwrappedAck(i-1);
    newGap = tmpPkt - tmpAck;

    % Check different potential pkt-ack combinations, and choose the best
    if abs(newGap - prevGap) > 20 % if gap is higher to 20, is not a 'normal'
    case
        % CASE 1: pkt unwraps from 255 to 0, but ACK continues normally
        delta_pkt = 255 - prev_pkt + curr_pkt + 1;
        tmpPkt = unwrappedPkt(i-1) + delta_pkt;

        prevGap = unwrappedPkt(i-1) - unwrappedAck(i-1);
        newGap = tmpPkt - tmpAck;

        if abs(newGap - prevGap) > 10 %check gap difference is not small
```

```
            % CASE 2: both pkt and ack unwraps from 255 to 0
            delta_ack = 255 - prev_ack + curr_ack + 1;
            tmpAck = unwrappedAck(i-1) + delta_ack;
            newGap = tmpPkt - tmpAck;

            if abs(newGap - prevGap) > MAX_GAP_JUMP %check gap difference is
    high
                % compute deltas
                delta_pkt = compute_delta(prev_pkt, curr_pkt);
                delta_ack = compute_delta(prev_ack, curr_ack);
                tmpPkt = unwrappedPkt(i-1) + delta_pkt;
                tmpAck = unwrappedAck(i-1) + delta_ack;

                % Check gaps
                prevGap = unwrappedPkt(i-1) - unwrappedAck(i-1);
                newGap = tmpPkt - tmpAck;

                if abs(newGap - prevGap) > 60
                    % Last option, add 1 to ack and pkt --> Check result
                    tmpAck = unwrappedAck(i-1) + curr_ack;
                    tmpPkt = unwrappedPkt(i-1) + delta_pkt;
                    newGap = tmpPkt - tmpAck;
                elseif abs(newGap - prevGap) > 240
                        tmpAck = unwrappedAck(i-1) + 1;
                        tmpPkt = unwrappedPkt(i-1) + 1;

                end
                end
            end
        end
    end

    % Store
    unwrappedPkt(i) = tmpPkt;
    unwrappedAck(i) = tmpAck;
end
end

% Function to compute the delta (increment from previous to actual value)
function delta = compute_delta(prev, curr)
if isnan(curr) || isnan(prev)
    delta = 0;
    return;
end

if (curr == 0) && (prev == 255)
    delta = 1;
else
```

```
    if curr >= prev
        delta = curr - prev;
    else
        diff = curr - prev;
        if diff >= -2 %MIN_GLITCH_DIFF
            delta = 0;  % small glitch
        else
            delta = curr+1;  % restart
        end
    end
end
end
```

The counters are unwrapped using the unwrap function.

```
% Process data by unwrapping the counters in the tables within the struct
senderTables = struct(); %create struct to store the data

for i = 1:length(senderNames) %loop over each sender table
    sender_X = senderNames{i};
    T = senderStruct.(sender_X);

    %Inside each sender table, loop for columns (send and ack)
    % Unwrap counters
    for j = 1:4 %4 nodes in the network
        sendCol = ['Send' num2str(j)];
        ackCol  = ['ACK'  num2str(j)];
        timeCol  = 'timestamp';
        [T.(sendCol), T.(ackCol)]= unwrap_resets(T.(sendCol), T.(ackCol));
    end

    % Compute PDR from the sent and ack values
    for j = 1:4 %4 nodes in the network
        % for each sender and ack respective column, calculate the PDR per
    destination node (4 nodes)
        sendCol = ['Send' num2str(j)];
        ackCol  = ['ACK'  num2str(j)];
        pdrCol  = ['PDR'  num2str(j)];
        T.(pdrCol) = T.(ackCol) ./ max(T.(sendCol), 1);
    end

    % Loop to convert hop values of 255 (no available route) or bigger than 4 to
     0
    for j = 1:4
        hop = ['Hop' num2str(j)];
        T.(hop) > 4 = 0;
    end
    senderTables.(sender_X) = T;  % Save the updated  unwrapped table
end
```

Once the data is processed, some statistics and parameters can be computed and plotted. The following code shows the plots of the PDR for each node in the network.

```
numHops = 4; %hops for loop over the tables, four nodes in the network

start_timestamp = '2025-05-28 12:00:00';
stop_timestamp  = '2025-06-09 01:59:56';
start_dt = datetime(start_timestamp, 'InputFormat', 'yyyy-MM-dd HH:mm:ss');
stop_dt  = datetime(stop_timestamp, 'InputFormat', 'yyyy-MM-dd HH:mm:ss');
xLimits = [start_dt stop_dt];

%% PDR 1
T = senderTables.Sender_1; % Extract sender 1
HEHdata = T.HEH; %Extract HEH data

colors = [ %Set colors for graph
    %0, 0.4470, 0.7410;   % blue
    0.8500, 0.3250, 0.0980; % orange
    0.9290, 0.6940, 0.1250; % yellow
    0.4940, 0.1840, 0.5560; % purple
];
colorIdx = 1; %color index

% set figure parameters
figure;
set(gcf, 'Units', 'centimeters', 'Position', [0, 0, 20, 8]);

hold on;
pdrLegend = {};
for j = 1:numHops %for each node in the network
    if j == 1
        continue; % Skip if sender is same as j index
    end
    pdrCol = ['PDR' num2str(j)];
    pdrNode = ['Node ' num2str(j)];
    plot(HEHdata, T.(pdrCol), 'o-', 'LineWidth', 4, 'Color', colors(mod(colorIdx
    -1, size(colors, 1)) + 1, :));
    pdrLegend{end+1} = pdrNode;
    colorIdx = colorIdx + 1;
end

% plot and set labels
title('Node 1 - PDR statistics');
xlabel('Fluence [HEH/cm^2]', 'FontSize', 14);
ylabel('PDR', 'FontSize', 14);
ylim([0.9997 1.000001]);
set(gca, 'YScale', 'log');
ax=gca;
ax.FontSize=14;
```

```
legend(pdrLegend, 'Location', 'northeast');
grid on;

% export the graph to pdf
exportgraphics(gcf, 'pdr_n1.pdf', 'ContentType', 'vector', 'BackgroundColor', '
    none');


%% PDR 2


T = senderTables.Sender_2; % Extract sender 2
HEHdata = T.HEH; %Extract HEH data

colors = [ %Set colors for graph
    0, 0.4470, 0.7410;   % blue
    %0.8500, 0.3250, 0.0980; % orange
    0.9290, 0.6940, 0.1250; % yellow
    0.4940, 0.1840, 0.5560; % purple
];
colorIdx = 1; %color index

% set figure parameters
figure;
set(gcf, 'Units', 'centimeters', 'Position', [0, 0, 20, 8]);

hold on;
pdrLegend = {};
for j = 1:numHops %for each node in the network
    if j == 2
        continue; % Skip if sender is same as j index
    end
    pdrCol = ['PDR' num2str(j)];
    pdrNode = ['Node ' num2str(j)];
    plot(HEHdata, T.(pdrCol), 'o-', 'LineWidth', 4, 'Color', colors(mod(colorIdx
    -1, size(colors, 1)) + 1, :));
    pdrLegend{end+1} = pdrNode;
    colorIdx = colorIdx + 1;
end

% plot and set labels
title('Node 2 - PDR statistics');
xlabel('Fluence [HEH/cm^2]', 'FontSize', 14);
ylabel('PDR', 'FontSize', 14);
ylim([0.9997 1.000001]);
set(gca, 'YScale', 'log');
ax=gca;
ax.FontSize=14;
```

```
legend(pdrLegend, 'Location', 'northeast');
grid on;

% export the graph
exportgraphics(gcf, 'pdr_n2.pdf', 'ContentType', 'vector', 'BackgroundColor', '
    none');

%% PDR 3


T = senderTables.Sender_3;% Extract sender 3
HEHdata = T.HEH; %Extract HEH data

colors = [ %Set colors for graph
    0, 0.4470, 0.7410;   % blue
    0.8500, 0.3250, 0.0980; % orange
    %0.9290, 0.6940, 0.1250; % yellow
    0.4940, 0.1840, 0.5560; % purple
];
colorIdx = 1; %color index

% set figure parameters
figure;
set(gcf, 'Units', 'centimeters', 'Position', [0, 0, 20, 8]);

hold on;
pdrLegend = {};
for j = 1:numHops %for each node in the network
    if j == 3
        continue; % Skip if sender is same as j index
    end
    pdrCol = ['PDR' num2str(j)];
    pdrNode = ['Node ' num2str(j)];
    plot(HEHdata, T.(pdrCol), 'o-', 'LineWidth', 4, 'Color', colors(mod(colorIdx
    -1, size(colors, 1)) + 1, :));
    pdrLegend{end+1} = pdrNode;
    colorIdx = colorIdx + 1;
end

% plot and set labels
title('Node 3 - PDR statistics');
xlabel('Fluence [HEH/cm^2]', 'FontSize', 14);
ylabel('PDR', 'FontSize', 14);
ylim([0.9997 1.000001]);
set(gca, 'YScale', 'log');
ax=gca;
ax.FontSize=14;
legend(pdrLegend, 'Location', 'northeast');
```

```
grid on;

% export the graph
exportgraphics(gcf, 'pdr_n3.pdf', 'ContentType', 'vector', 'BackgroundColor', '
    none');

%% PDR 4


T = senderTables.Sender_4;% Extract sender 4
HEHdata = T.HEH; %Extract HEH data

colors = [ %Set colors for graph
    0, 0.4470, 0.7410;   % blue
    0.8500, 0.3250, 0.0980; % orange
    0.9290, 0.6940, 0.1250; % yellow
    %0.4940, 0.1840, 0.5560; % purple
];
colorIdx = 1; %color index

% set figure parameters
figure;
set(gcf, 'Units', 'centimeters', 'Position', [0, 0, 20, 8]);

hold on;
pdrLegend = {};
for j = 1:numHops %for each node in the network
    if j == 4
        continue; % Skip if sender is same as j index
    end
    pdrCol = ['PDR' num2str(j)];
    pdrNode = ['Node ' num2str(j)];
    plot(HEHdata, T.(pdrCol), 'o-', 'LineWidth', 4, 'Color', colors(mod(colorIdx
    -1, size(colors, 1)) + 1, :));
    pdrLegend{end+1} = pdrNode;
    colorIdx = colorIdx + 1;
end

% plot and set labels
title('Node 4 - PDR statistics');
xlabel('Fluence [HEH/cm^2]', 'FontSize', 14);
ylabel('PDR', 'FontSize', 14);
ylim([0.9997 1.000001]);
set(gca, 'YScale', 'log');
ax=gca;
ax.FontSize=14;
legend(pdrLegend, 'Location', 'northeast');
grid on;
```

```
184
185 % export the graph
186 exportgraphics(gcf, 'pdr_n4.pdf', 'ContentType', 'vector', 'BackgroundColor', '
    none');
```

For the bar graph showing the ratio of total successfully sent and lost packets, the following code was used.

```
1  % initialize variables
2  barData = [];
3
4  % loop over each sender table and each destination node to obtain the total
5  % sent and ack packets and store it in barData variable for plotting
6  for i = 1:length(senderNames)
7      senderName = senderNames{i};
8      T = senderTables.(senderName);
9      final_value = []; %final values of send and ack counters
10     for j = 1:4
11         sendCol = ['Send' num2str(j)];
12         ackCol  = ['ACK'  num2str(j)];
13         % previous values, send, ack, lost packets
14         lastValues = [lastValues, T.(sendCol)(end), T.(ackCol)(end), T.(sendCol)
       (end)-T.(ackCol)(end)];
15     end
16     barData = [barData; lastValues];
17 end
18
19 % Data for three nodes network, therefore, a sender node has two links
20 packets1 = [barData(1,6), barData(1,5);   % Link 1
21         barData(1,9), barData(1,8)];  % Link 2
22
23 % Labels
24 label = {'Node 2', 'Node 3'};
25 sections = {'Lost', 'ACK'};
26
27 figure;
28 b = bar(packets1, 'stacked');
29 set(gca, 'XTickLabel', label);
30 title('Gateway 1 - Packets overview');
31 xlabel('Receiver Nodes');
32 ylabel('Packet Count');
33 ylim([1 17500000]);
34 legend(sections);
35 grid on;
36 set(gca, 'YScale', 'log');
37
38 bar_percentage = 100*packets1./sum(packets1,2); %calculate %
39
40 % put the colors
```

```
set(b, 'FaceColor', 'Flat')
b(1).CData = [0.87, 0.63, 0.06] ;
b(2).CData = [0, 0.4470, 0.7410];
```

Moreover, the power cycling events are plotted by extracting the reset events from the generated text file.

```
% read power cycling events
filename = 'featherM0_RESETS_full.txt';
fid = fopen(filename, 'r');
rawLines = textscan(fid, '%s', 'Delimiter', '\n');
fclose(fid);
rawLines = rawLines{1}; %obtain table from the struct created

% read HEH data from tables
load('full_table_1.mat')
senderTables = full_table;
radiationTable.Timestamp = senderTables.Sender_1.timestamp;
radiationTable.HEH = senderTables.Sender_1.HEH;

% initialize variables
eventTimes = datetime.empty;
nodeIDs = [];

% extract event node inforamtion from txt
for i = 1:length(rawLines)
    line = rawLines{i};
    splitLine = split(line, '|');

    %extract datetime value
    timedate = strtrim(splitLine{1});
    eventTime = datetime(timedate, 'InputFormat', 'yyyy-MM-dd HH:mm:ss');

    % extract node ID
    node_ID = strtrim(splitLine{2});
    nodeNum = sscanf(node_ID, 'Node %d');

    % store both values in respective variables
    eventTimes(end+1,1) = eventTime; %datetime array
    nodeIDs(end+1,1) = nodeNum; %node ID arrays
end

%% Plot
figure;
set(gcf, 'Units', 'centimeters', 'Position', [0, 0, 25, 8]);
hold on;
grid on;
nodeList = unique(nodeIDs); %array with nodes in the network
% (considering that all nodes have suffered a power cycling event)
```

```
43
44 % loop over each node, to plot events per node
45 for i = 1:length(nodeList)
46     node = nodeList(i); %select node
47     idx = find(nodeIDs == node); %find events for selected node
48
49     HEH_events = eventHEH(idx); %extract
50     IDs = node * ones(size(x));
51
52     % points for each event
53     scatter(HEH_events, IDs, 60, 'filled', 'MarkerFaceColor', colors(i,:));%plot
        events
54     %lines for each node
55     plot(HEH_events, IDs, '-', 'Color', colors(i,:), 'LineWidth', 1.5); %to plot
        line
56
57 end
58
59 xlabel('Fluence [HEH/cm^2]')
60 ylabel('Node ID')
61 title('Reset Events as a function of the fluence')
62 legend(labels, 'Location', 'northwest');
63
64 %export graph
65 exportgraphics(gcf, 'reset_events.pdf', 'ContentType', 'vector',...
66     'BackgroundColor', 'none');
```