Co-funded by the European Union

6GSNS

SUPERIOT

# D3.4 Energy model and optimization algorithms

| | |
|---|---|
| Project number | **101096021** |
| Project name | **Truly Sustainable Printed Electronics-based IoT Combining Optical and Radio Wireless Technologies** |
| Project acronym | **SUPERIOT** |
| Call | **HORIZON-JU-SNS-2022** |
| Deliverable No | **D3.4** |
| Deliverable Name | **Energy model and optimization algorithms** |
| Status | **Final** |
| Dissemination level | **Public** |
| Due date of deliverable | **2025-11-30 (M35)** |
| Actual submission date | **2025-11-28 (M35)** |
| Resubmission date | **2026-05-22** |
| Work package | **WP3 "Network Architecture"** |
| Lead beneficiary | **U of Bristol** |
| Authors | Senhui Qiu (editor), Junaid Bocus, Kerstin Eder, Robert Piechocki, U of Bristol<br>Hazem Sallouha, KU Leuven<br>Alaa Abdellatif, INESC TEC<br>Mohammad Khalili, Konstantin Mikhaylov, UOULU<br>Marcin Drzewiecki, MPICOSYS |
| Reviewers: | Marcin Drzewiecki, MPICOSYS<br>Guido Dolmans, IMEC-NL<br>Khodr Hammoud, KU Leuven |

**The SUPERIOT project has received funding from the Smart Networks and Services Joint Undertaking (SNS JU) under the European Union's Horizon Europe research and innovation programme under Grant Agreement No 101096021, including top-up funding by UK Research and Innovation (UKRI) under the UK government's Horizon Europe funding guarantee.**

# Executive Summary

The Truly Sustainable Printed Electronics-based IoT Combining Optical and Radio Wireless Technologies (SUPERIOT) project aims to develop next-generation Reconfigurable Internet of Things (RIoT) systems based on printed electronics, focusing on energy efficiency, adaptability, and large-scale integration. Deliverable D3.4 – Network-Level Energy Optimization in RIoT Environments marks a key milestone within Work Package 3 (WP3), consolidating methodologies, tools, and validated results for achieving intelligent, energy-aware, and scalable IoT networks within the SUPERIOT ecosystem.

This deliverable presents a comprehensive multi-layer framework that combines empirical energy modelling, Artificial Intelligence (AI)-driven prediction, and cross-layer optimization. It includes accurate energy modelling, AI/Machine Learning (ML)-based prediction tools, and optimization of hybrid communication using Bluetooth Low Energy (BLE), Visible Light Communication (VLC), and Radio Frequency (RF)/optical technologies. It also incorporates a Digital Twin (DT) environment in ns-3 and ultra-low-power hardware and firmware design, enabling adaptive and sustainable operation across IoT environments.

Empirical analyses characterized the energy consumption of BLE and VLC communication modes, gateways, and infrastructure devices such as Raspberry Pi and routers. The results revealed trade-offs between throughput and power usage, defining optimal configurations. Advanced AI/ML models, including Gradient Boosting and Temporal Fusion Transformers, achieved up to 97.8% prediction accuracy and were integrated into a graphical tool that visualizes and predicts energy outcomes. The intelligent optimization framework reduced total network energy consumption by up to 77% compared to baseline configurations.

To enhance adaptability across heterogeneous communication technologies, adaptive algorithms were developed for modality selection, handover, and power allocation. A multi-attribute decision-making (MADM) framework managed handover decisions by balancing energy, Quality-of-Service (QoS), and network load, while a multi-objective Q-learning approach optimized throughput and energy efficiency. In parallel, a Graph Neural Network (GNN)-based architecture efficiently addressed large-scale resource allocation problems, achieving near-optimal results with low computational overhead and high robustness to imperfect network data.

A DT of the RIoT network was developed in ns-3 to support traffic- and usage-aware simulation and cross-layer optimization. It integrates BLE and VLC modules, detailed energy models, a polling-based Medium Access Protocol (MAC) mechanism, and Message Queuing Telemetry Transport (MQTT) control interfaces, enabling accurate simulation of multimodal, energy-constrained IoT scenarios. Monitoring residual energy, Signal-to-Noise Ratio (SNR), and packet loss allows scenario-specific profiling, while adaptive scheduling lets nodes sleep or switch interfaces proactively. This provides a flexible testbed that minimizes the need for physical prototyping and accelerates validation of energy-aware strategies.

At the hardware and firmware level, significant progress was achieved in reducing node energy consumption. Integration of a 2.13-inch E-ink display (250×122 pixels) yielded over a fivefold reduction in energy use, while firmware optimizations further reduced power consumption more than sixfold and shortened display refresh times by over tenfold. Additional firmware improvements for SUPERIOT demonstrators achieved up to 80% total energy savings, ensuring that node-level performance aligns with network-wide strategies for sustainable and long-lived RIoT deployments.

# Table of contents

# Editions

| Version | Date | Modified by | Modification |
|---|---|---|---|
| 0.1 | 2025-05-07 | Senhui Qiu, Junaid Bocus, U of Bristol | Initial draft |
| 0.2 | 2025-07-02 | Senhui Qiu, Junaid Bocus, U of Bristol | Preliminary input from all contributors |
| 0.3 | 2025-08-13 | Senhui Qiu, Junaid Bocus, Kerstin Eder, Robert Piechocki, U of Bristol<br>Hazem Sallouha, KU Leuven<br>Alaa Abdellatif, INESC TEC<br>Mohammad Khalili, UOULU<br>Marcin Drzewiecki, MPICOSYS | Revised draft following feedback |
| 0.4 | 2025-10-31 | Senhui Qiu, Junaid Bocus, Kerstin Eder, Robert Piechocki, U of Bristol<br>Hazem Sallouha, KU Leuven<br>Alaa Abdellatif, INESC TEC<br>Mohammad Khalili, UOULU<br>Marcin Drzewiecki, MPICOSYS | Ready for internal review |
| 0.5 | 2025-11-14 | Senhui Qiu, Junaid Bocus, Kerstin Eder, Robert Piechocki, U of Bristol<br>Hazem Sallouha, KU Leuven<br>Alaa Abdellatif, INESC TEC<br>Mohammad Khalili, UOULU<br>Marcin Drzewiecki, MPICOSYS | Ready for final submission |
| 1.0 | 2025-11-28 | Senhui Qiu, Junaid Bocus, Kerstin Eder, Robert Piechocki, U of Bristol<br>Hazem Sallouha, KU Leuven<br>Alaa Abdellatif, INESC TEC<br>Mohammad Khalili, Konstantin Mikhaylov, UOULU<br>Marcin Drzewiecki, MPICOSYS | Final version submitted to the EC |
| 2.0 | 2026-05-22 | Junaid Bocus, and Kerstin Eder, U of Bristol<br>Hazem Sallouha, KU Leuven | Modifications addressing the request for revision in sections 3.12.3.1, 3.12.3.2, 3.12.4, 3.13 |

| | | | |
|---|---|---|---|
| | | Mohammad Khalili, Konstantin Mikhaylov, UOULU | (whole section), 3.14, 4.6.2.1, 4.7, 4.7.1. |

# 1 Acronyms

| | |
|---|---|
| AMQP | Advanced Message Queuing Protocol |
| AI | Artificial Intelligence |
| AoI | Age of Information |
| AP | Access Point |
| API | Application Programming Interface |
| ATT | Attribute Protocol |
| BBB | Beagle Bone Black |
| BER | Bit Error Rate |
| BLE | Bluetooth Low Energy |
| CI | Connection Interval |
| CoAP | Constrained Application Protocol |
| CPE | Customer Premise Equipment |
| CPU | Central Processing Unit |
| CV | Cross-Validation |
| DC | Direct Current |
| DGET | Dual-Graph Embedding with Transformer |
| DL | Downlink |
| DLE | Data Length Extension |
| DNN | Deep Neural Network |
| DSL | Digital Subscriber Line |
| DT | Digital Twin |
| EM | Energy Model |
| EPD | E-Paper Display |
| ERR | Energy-to-Rate Ratio |
| ETNO | Energy-aware Threshold-based Node Optimization |
| EUNO | Energy-aware Utility-based Node Optimization |
| EWMA | Exponentially Weighted Moving Average |

| | |
|---|---|
| FoV | Field of View |
| GATT | Generic Attribute Profile |
| GFSK | Gaussian Frequency Shift Keying |
| GNN | Graph Neural Network |
| GPIO | General-Purpose Input/Output |
| GUI | Graphical User Interface |
| GW | Gateway |
| HTTP | Hypertext Transfer Protocol |
| IC | Integrated Circuit |
| IGW | IoT Gateway |
| IoT | Internet-of-Things |
| IPC | Inter-Process Communication |
| IR | Infrared |
| ISR | Interrupt Service Routine |
| JSON | JavaScript Object Notation |
| LED | Light-Emitting Diode |
| LUT | Look-Up Table |
| LoS | Line-of-Sight |
| MAC | Medium Access Protocol |
| MAE | Mean Absolute Error |
| MADM | Multi-Attribute Decision-Making |
| MAPE | Mean Absolute Percentage Error |
| MCDM | Multi-Criteria Decision-Making |
| MCS | Modulation and Coding Scheme |
| MDP | Markov Decision Process |
| MILP | Mixed-Integer Linear Programming |
| MINLP | Mixed-Integer Non-Linear Programming |
| ML | Machine Learning |

| | |
|---|---|
| MQTT | Message Queuing Telemetry Transport |
| MTU | Maximum Transmission Unit |
| MOO | Multi-Objective Optimization |
| NEC | Nippon Electric Company |
| NBVLC | Narrowband Visible Light Communication |
| OOK | On-Off Keying |
| OQPSK | Offset Quadrature Phase Shift Keying |
| OS | Operating System |
| OWC | Optical Wireless Communication |
| PL | Prediction Length |
| PLR | Packet Loss Ratio |
| PHY | Physical Layer |
| PON | Passive Optical Network |
| PPK II | Power Profiler Kit II |
| PWM | Pulse Width Modulation |
| QoS | Quality-of-Service |
| $R^2$ | Coefficient of Determination |
| RAM | Random Access Memory |
| RF | Radio Frequency |
| RIoT | Reconfigurable Internet-of-Things |
| RIS | Reconfigurable Intelligent Surfaces |
| RMSE | Root Mean Squared Error |
| RPi | Raspberry Pi |
| RSSI | Received Signal Strength Indicator |
| RX | Receive |
| RW | Random Waypoint |
| SNR | Signal-to-Noise Ratio |

| | |
|---|---|
| SUPERIOT | Truly Sustainable Printed Electronics-based IoT Combining Optical and Radio Wireless Technologies |
| TCP | Transmission Control Protocol |
| TDMA | Time Division Multiple Access |
| TFT | Thin Film Transistor |
| TFTM | Temporal Fusion Transformer Model |
| TL | Training Length |
| TV | Television |
| TX | Transmit |
| UART | Universal Asynchronous Receiver/Transmitter |
| UDP | User Datagram Protocol |
| UL | Uplink |
| USB | Universal Serial Bus |
| UUID | Universally Unique Identifier |
| VLC | Visible Light Communication |
| VPPM | Variable Pulse Position Modulation |
| WLAN | Wireless Local Area Network |

# 2 Introduction

## 2.1 Motivation

The proliferation of Internet-of-Things (IoT) devices has led to an exponential increase in energy demand, raising critical concerns about sustainability, especially in energy-constrained environments. Efficient energy management is essential to support reliable and scalable IoT networks while meeting application-specific QoS requirements. The integration of advanced AI/ML techniques offers a promising avenue to achieve these goals by enabling intelligent decision-making, proactive optimization, and adaptive resource allocation. Task 3.4 addresses these challenges by focusing on network-wide energy modeling, communication modality optimization, and traffic-aware cross-layer enhancements, contributing to the realization of sustainable RIoT systems.

## 2.2 Background and Objectives

The RIoT framework seeks to revolutionize IoT network performance by combining Reconfigurable Intelligent Surfaces (RIS), multi-modal communication, and advanced energy management. Task 3.4 builds upon prior foundational work to develop innovative methodologies for energy optimization. This deliverable focuses on the optimization of RIoT network resources, namely energy, spectrum, and target application QoS. The main goals of this task are threefold, corresponding to the three subtasks listed below:

- **Energy Modeling and Prediction**: Create novel energy consumption models based on hardware-level measurements and validate these models through empirical data (develop energy models of network elements and network as a whole). Model, analyze and predict energy consumption at the network level, including some high-level network simulations using these energy models and simplified RIoT node abstractions; executed in the context of T3.4.1.
- **Communication Modality Optimization**: Develop mechanisms to optimize communication media and access technology configurations in real-time using AI/ML techniques. Configure and optimize the access technology and communication modality, including the integration and interaction with the RIS, executed in T3.4.2.
- **Traffic-Aware Optimization**: Implement proactive cross-layer optimization strategies that predict traffic and usage patterns, allowing dynamic adjustment of network configurations to optimize energy efficiency without compromising QoS. Proactive cross-layer optimization based on traffic and usage pattern prediction, executed in T3.4.3.

By achieving these objectives, Task 3.4 will provide scalable, energy-efficient solutions tailored to the operational demands of RIoT systems.

## 2.3 Scope of the Deliverable

Deliverable D3.4 encompasses the outcomes of Task 3.4, presenting comprehensive energy models, optimization algorithms, and detailed experimental evaluations. The scope includes:

- **Network-Level Energy Models**: Analytical models developed and validated through direct hardware measurements, providing accurate energy usage predictions for RIoT networks.
- **Optimization Algorithms**: Algorithms for real-time selection of communication media (e.g., RF, optical) and access technology configurations, leveraging AI/ML for enhanced decision-making (AI/ML-based energy management of the network and optimizing use of resources - primarily energy, while remaining mindful of spectrum and target application QoS).
- **Cross-Layer Mechanisms**: Proactive optimization strategies based on application traffic and usage pattern predictions, enabling dynamic reconfiguration of RIoT nodes and network parameters.

- **Integration and Validation**: High-level network simulations and empirical testing to ensure that the proposed solutions are effective, scalable, and adaptable across diverse RIoT scenarios.

This deliverable serves as a foundational resource for implementing energy-efficient RIoT systems, advancing the SUPERIOT project's overarching goals.

## 2.4 Structure of the Document

This document is organized to present a coherent framework for energy-efficient design and optimization of the RIoT system. Chapter 3 focuses on energy consumption prediction and optimization at the network level, including data collection from the RIoT hardware platform and the development of AI/ML-based models for energy estimation and optimization. Chapter 4 builds upon this by addressing communication modality and access technology configuration optimization, proposing adaptive mechanisms for real-time selection of RF or optical communication modalities and transmission parameters to balance energy efficiency, latency, and scalability. Chapter 5 extends the optimization framework across network layers by incorporating traffic and usage pattern prediction to enable proactive, cross-layer adjustments that improve performance without compromising QoS. Finally, Chapter 6 concentrates on system-level energy optimization for RIoT nodes through hardware and firmware enhancements that ensure low-power operation and consistency with network-wide energy-saving goals. Together, these chapters provide an integrated approach to achieving intelligent, sustainable, and energy-aware operation across all layers of the RIoT ecosystem.

Sections 3.12.3.2., 3.13 (with all subsections) and 4.7 were introduced during revision for detailing the added value of ML approach, provide formalized optimization problem formulation, and justification of MQTT protocol selection, respectively. Section 3.12.4 and the beginning of section 4.1 were extended to include the discussion of the relevant stakeholders and use of the developed solutions, and include more information and relevant references to publications.

# 3 Energy Modelling, Analysis, and Prediction at the Network Level

## 3.1 Energy Characterization

This task focuses on studying and measuring patterns of energy consumption and power dissipation at the network level, with the goal of developing novel models and estimation techniques. These models support both the analysis and optimization of energy and power usage during system development and runtime operations.

To ensure accuracy, energy and power measurements are collected directly from the target hardware platforms. The task involves investigating state-of-the-art modelling approaches to identify the most effective and accurate techniques for capturing network-level behavior. These models are then validated against empirical hardware measurements to ensure their reliability and applicability in real-world scenarios. The resulting models and techniques play a key role in enabling energy-aware optimization and informed decision-making at the network level.

## 3.2 Throughput and Power Considerations in BLE Communication

The performance of the BLE device, in terms of data throughput and power consumption, is influenced by various connection parameters. Among these parameters, the ones that significantly impact data throughput include the connection interval (CI) and the Attribute Protocol (ATT) Maximum Transmission Unit (ATT MTU). Additionally, adjusting the radio power level can optimize power consumption based on signal strength, distance, and line of sight (LoS) between devices. When the central device (e.g. mini lamp gateway (GW) or Beagle Bone Black (BBB) Access Point (AP)) identifies the peripheral (e.g. SUPERIOT node) with a specific name, it initiates a connection request to the peripheral. Upon establishing the connection, the Generic Attribute Profile (GATT) client and server exchange parameters such as ATT MTU, Data Length Extension (DLE), and Physical Layer (PHY)rate. These parameters are described below:

- The ATT MTU specifies the maximum amount of data that a device can transmit or receive per GATT operation. The default Maximum Transmission Unit (MTU) is 23 bytes. Increasing this value allows for longer ATT payloads, thereby enhancing ATT throughput.
- The default data length for a radio packet is 27 bytes. Enabling DLE allows for the use of larger radio packets, facilitating the transmission of more data in a single packet, and boosting throughput.
- The CI defines the frequency at which devices in a connection (like a central device and a peripheral device) exchange data packets. With a shorter CI, devices exchange data more frequently, allowing for faster data transmission. Shorter intervals allow for more packets to be sent at a given time, thus increasing throughput.
- BLE PHY rate determines the data rate and range of communication. Coded PHY enables long-range communication with lower throughput, making it suitable for power-sensitive, long-distance applications.

### 3.2.1 BLE Throughput Measurements on Si-based RIoT Node

In this experiment, we include an additional functionality in the node's original firmware that allows us to get the throughput when the node is transmitting dummy data to a central device (e.g. mini lamp GW or BBB AP) positioned at 5 m from the node. We analyzed the throughput achieved by the node under different configuration parameters, including BLE CI, MTU, and PHY rate, and we also measured the corresponding average current consumption across the node. The default BLE transmit (TX) power level of 0 dBm is considered. As shown in Figure 1, following a complete transmission of the payload by the node, the next throughput test occurs after 5 seconds.

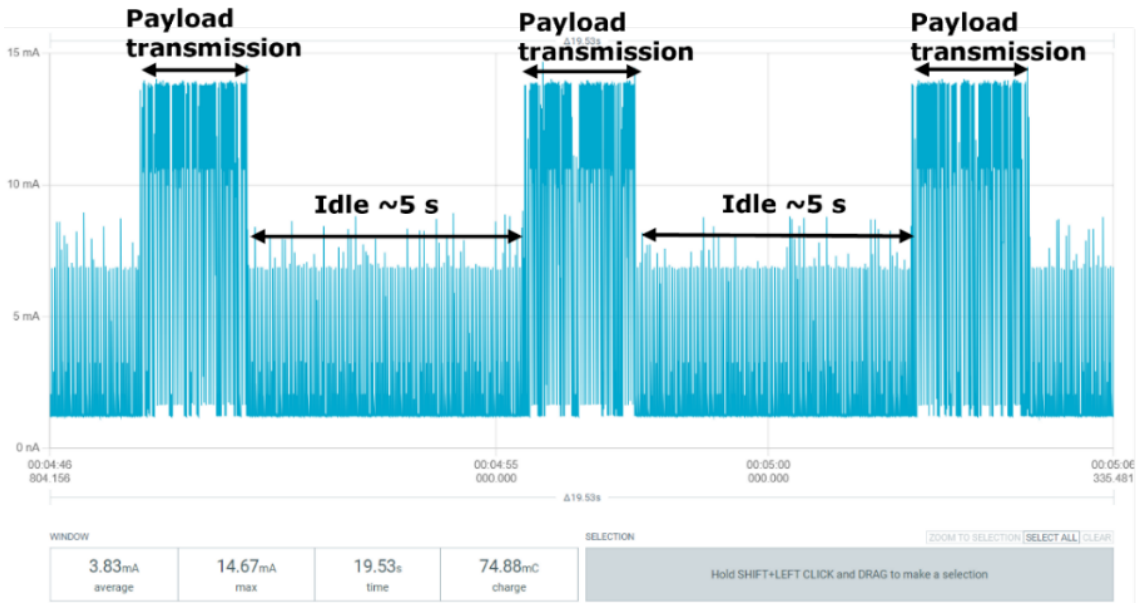



Figure 1. Current profile of node during payload transmissions of 244 kB (CI=45 ms, MTU=247, PHY rate=2 Mbps).

#### *3.2.1.1 Different PHY Rate*

Table 1 correctly reflects that the BLE PHY rate of 2 Mbps offers improved performance over the 1 Mbps rate, with shorter transmission durations and higher throughput. This highlights the benefit of higher PHY rates in BLE communication for more efficient and faster data transfer. The 2 Mbps PHY rate is clearly more energy-efficient and faster than the 1 Mbps rate for the same payload. It completes the transfer in less time and consumes less energy per bit, making it the preferable option in scenarios where both throughput and energy efficiency are important.

Table 1. Throughput and average current consumption for different BLE PHY rates.

| **MTU = 247 bytes, CI = 45 ms, Payload = 244 kB, Node's operating voltage = 3.3 V, BLE TX power = 0 dBm** | | | | | |
|---|---|---|---|---|---|
| **PHY rate (Mbps)** | **Average Throughput (kB/s)** | **Average duration of payload transfer (s)** | **Average current consumption during payload transfer (mA)** | **Average current consumption during idle (mA)** | **Energy per bit (nJ/bit)** |
| 2 | 119.73 | 2.06 | 9.10 | 1.36 | 32 |
| 1 | 70.4 | 3.47 | 9.42 | 1.36 | 56 |

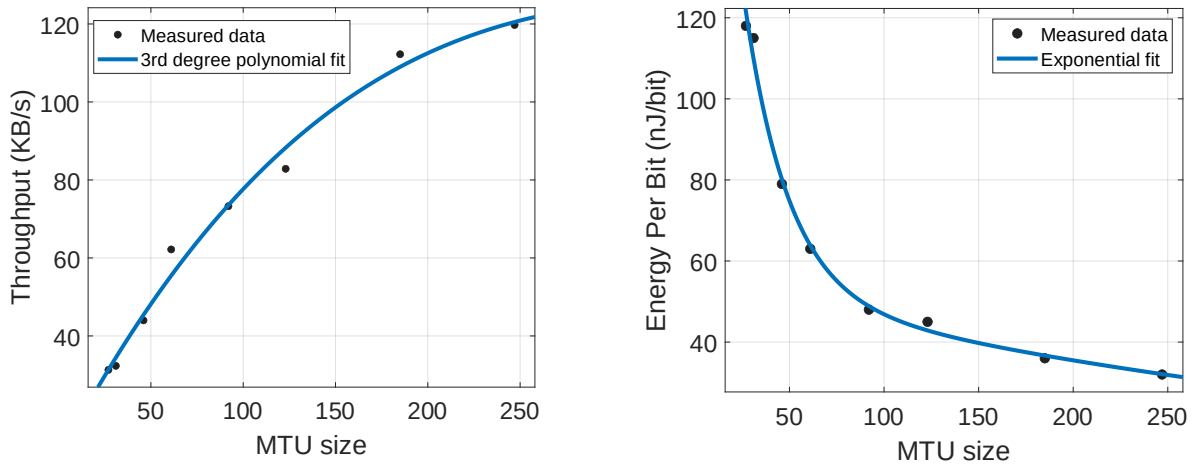
#### *3.2.1.2 Different MTU*

It can be observed from Table 2 and Figure 2(a) that increasing the MTU size leads to a higher throughput, as larger packets can carry more data, reducing the number of packets required and overhead associated with packet headers. This efficiency gain also translates to lower energy consumption, as shown in Figure 2(b). The best trade-off between throughput and energy efficiency is achieved with the largest MTU size, making it the preferred configuration for high-efficiency BLE communication, especially when transmitting large amounts of data. We note from Table 2 that during payload transmission, the current consumed is significantly higher (~9 mA)

than the current during the idle BLE connection (~1.36 mA), considering a default TX power level of 0 dBm, and BLE connection interval of 45 ms.

Table 2. Throughput and average current consumption measurements for different MTU.

| **PHY = 2 Mbps, CI = 45 ms, Payload = 244 kB, Node's operating voltage = 3.3 V, BLE TX power = 0 dBm** | | | | | |
|---|---|---|---|---|---|
| **MTU (bytes)** | **Throughput (kB/s)** | **Average duration of payload transfer (s)** | **Average current consumption during payload transfer (mA)** | **Average current consumption during idle (mA)** | **Energy per bit (nJ/bit)** |
| 247 | 119.73 | 2.06 | 9.10 | 1.36 | 32 |
| 185 | 112.24 | 2.18 | 9.57 | 1.36 | 36 |
| 123 | 82.86 | 2.96 | 9.00 | 1.36 | 45 |
| 92 | 73.27 | 3.34 | 8.41 | 1.36 | 48 |
| 61 | 62.18 | 3.94 | 9.36 | 1.36 | 63 |
| 46 | 43.99 | 5.57 | 8.34 | 1.36 | 79 |
| 31 | 32.33 | 7.60 | 8.89 | 1.36 | 115 |
| 27 | 31.30 | 7.83 | 8.87 | 1.36 | 118 |




(a) (b)

Figure 2. Impact of MTU size on node's throughput and energy per bit (PHY = 2 Mbps, CI = 45 ms, Payload = 244 kB, BLE TX power = 0 dBm, operating voltage = 3.3 V).

Based on the data in Table 2, we can fit 3rd degree polynomial and exponential decay models to get an estimate of the throughput (in kB/s) and energy per bit (nJ/bit), respectively, considering a PHY rate of 2 Mbps, BLE CI of 45 ms and BLE TX power level of 0 dBm:

$$\text{Throughput}\left(\frac{\text{kB}}{\text{s}}\right) = 1.983 \times 10^{-6}x^3 - 0.002301x^2 + 0.9002x + 8.659 \text{ for } 27 \leq x \leq 247 \quad (3.1)$$

$$E_{\text{b}}(\text{nJ/bit}) = 225.03\text{e}^{-0.0431x} + 54.19\text{e}^{-0.0021x} \text{ for } 23 \leq x \leq 247 \quad (3.2)$$

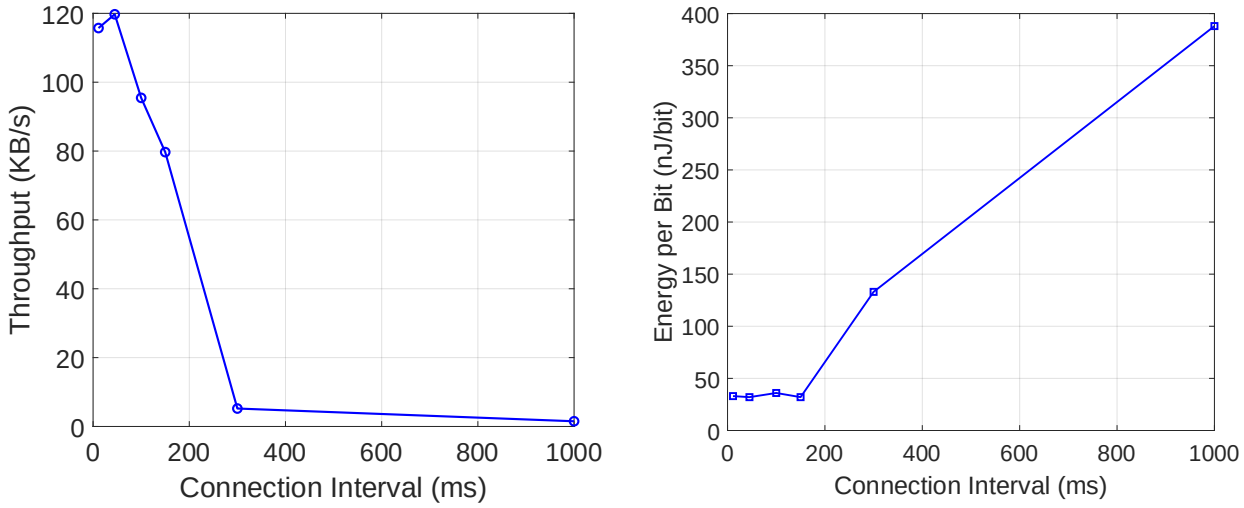
where $x$ is the MTU size. The throughput model achieves a Root Mean Squared Error (RMSE), Mean Absolute Percentage Error (MAPE), and coefficient of determination ($R^2$) value of 3.39 kB/s, 0.9886 and 3.87%, respectively. The energy per bit model achieves an RMSE, MAPE, and $R^2$ value of 2.39 nJ/bit, 0.9943, and 2.48%, respectively.

#### *3.2.1.3 Different Connection Intervals*

Considering a fixed MTU of 247 and PHY rate of 2 Mbps, the observations from Table 3 and Figure 3 reveal key trade-offs in the relationship between CI, throughput, energy consumption, and efficiency. Throughput is highest, reaching approximately 120 kB/s, for shorter CIs (11.25 ms and 45 ms) but decreases drastically at longer intervals, especially beyond 150 ms (see Figure 3(a)). Energy efficiency is also better at shorter intervals, with energy per bit as low as 32–36 nJ/bit for intervals up to 150 ms. However, at longer intervals, energy per bit increases significantly, peaking at 388 nJ/bit for a 1000 ms interval (see Figure 3(b)). The duration of payload transfer shows a sharp increase with longer CI, rising from ~2 seconds at 11.25 ms to over 160 seconds at 1000 ms. Average current consumption during payload transfer decreases significantly with longer CI, while idle current decreases only slightly. Thus, shorter CIs optimize throughput and energy efficiency. For a balance between energy efficiency and throughput, CI between 45–150 ms may offer the optimal configuration, depending on application requirements.

Table 3. Throughput and average current consumption of node for different BLE CIs.

| **PHY = 2 Mbps, MTU = 247, Payload = 244 kB, Node's operating voltage = 3.3 V, BLE TX power = 0 dBm** | | | | | |
|---|---|---|---|---|---|
| **Connection interval (ms)** | **Throughput (kB/s)** | **Average duration of payload transfer (s)** | **Average current consumption during payload transfer (mA)** | **Average current consumption during idle (mA)** | **Energy per bit (nJ/bit)** |
| 11.25 | 115.73 | 2.11 | 9.18 | 1.64 | 33 |
| 45 | 119.73 | 2.06 | 9.10 | 1.36 | 32 |
| 100 | 95.45 | 2.61 | 8.01 | 1.30 | 36 |
| 150 | 79.69 | 3.13 | 6.04 | 1.29 | 32 |
| 300 | 5.21 | 46.99 | 1.67 | 1.27 | 133 |
| 1000 | 1.51 | 161.43 | 1.42 | 1.26 | 388 |




(a) (b)

Figure 3. Impact of BLE CI on throughput and energy per bit (PHY = 2 Mbps, MTU = 247, Payload = 244 kB, BLE TX power = 0 dBm).

#### *3.2.1.4 Impact of BLE TX Power Level During Data Payload Transfer*

From Table 4, we observe that for shorter CIs (11.25 ms and 45 ms), throughput is significantly higher, ranging from 107.56–126.18 kB/s, but decreases drastically to ~5–6 kB/s for longer intervals like 300 ms (as mentioned in Section 3.2.1.3). Among the reasons causing this are (i) the limited size of the data buffers, and (ii) the retransmission policy of BLE, causing communication to continue only in the next CI in case of packet losses. BLE TX power does not have a direct impact on throughput; instead, it influences throughput indirectly by affecting signal strength, range, and packet delivery success rate. Higher TX power improves signal reliability in challenging environments, reducing packet loss and retransmissions, which can indirectly enhance throughput. For instance, at a CI of 45 ms, throughput decreases slightly from 126.18 kB/s (8 dBm) to 107.56 kB/s (-8 dBm), likely due to reduced signal reliability.

Energy efficiency varies significantly with CI and TX power levels. Energy per bit is lowest for shorter CI, reaching 27–32 nJ/bit at 11.25 ms and 45 ms with optimal TX power levels (-8 dBm to 0 dBm). A longer CI (300 ms) results in higher energy per bit (133–165 nJ/bit), highlighting reduced efficiency. The duration of payload transfer also increases sharply with longer intervals, from ~2 seconds at 11.25 ms and 45 ms to over 38–46 seconds at 300 ms, resulting in slower data transmission. Current consumption during payload transfer is higher for shorter CI and higher TX power levels, such as 22 mA at 8 dBm for a connection interval of 45 ms, while lower TX power levels reduce current consumption to ~6.97 mA at -8 dBm for the same CI, as shown in Figure 4. For BLE CI of 11.25 ms and 45 ms, the relationship between data payload TX current and BLE TX power level is very similar, and they can be modelled using a $4^{th}$ degree polynomial fit as in Figure 4. For a higher CI, we observe from Table 4 that the increase in data payload TX current is very subtle with increasing BLE TX power levels. Idle current varies only slightly across different BLE CI and BLE TX power levels, ranging from 1.27 mA to 1.76 mA.

Therefore, we can conclude that shorter BLE CI (e.g. 11.25 ms and 45 ms) maximizes throughput and minimizes energy per bit but requires higher current during transmission, especially at higher TX power levels (such as +4 dBm and +8 dBm). The longer BLE CI reduces current during payload transmission but is energy inefficient (in terms of energy per bit) and unsuitable for high-throughput applications. Lower TX power levels (0 dBm to -8 dBm) generally improve energy efficiency without severely impacting throughput (for small BLE CI such as 11.25 ms and 45 ms), provided the link quality remains stable (no packet retransmissions required during payload transmission).

Table 4. Impact of BLE TX power level during payload transmission.

| PHY = 2 Mbps, MTU = 247, Payload = 244 kB, Node's operating voltage = 3.3 V | | | | | | |
|---|---|---|---|---|---|---|
| **Connection interval (ms)** | **BLE TX power level (dBm)** | **Throughput (kB/s)** | **Average duration of payload transfer (s)** | **Average current consumption during payload transfer (mA)** | **Average current consumption during idle (mA)** | **Energy per bit (nJ/bit)** |
| 45 | 8 | 126.18 | 1.94 | 22.00 | 1.38 | 72 |
| | 4 | 121.82 | 2.01 | 13.10 | 1.36 | 45 |
| | 0 | 119.73 | 2.06 | 9.10 | 1.36 | 32 |
| | -4 | 113.71 | 2.15 | 7.61 | 1.36 | 28 |
| | -8 | 107.56 | 2.28 | 6.97 | 1.36 | 27 |
| 11.25 | 8 | 109.63 | 2.238 | 20.05 | 1.76 | 76 |
| | 4 | 106.91 | 2.28 | 13.06 | 1.69 | 51 |
| | 0 | 115.73 | 2.11 | 9.18 | 1.64 | 33 |
| 300 | 8 | 6.308 | 38.83 | 2.52 | 1.28 | 165 |
| | 4 | 5.85 | 41.71 | 1.93 | 1.28 | 137 |
| | 0 | 5.21 | 46.99 | 1.67 | 1.27 | 133 |

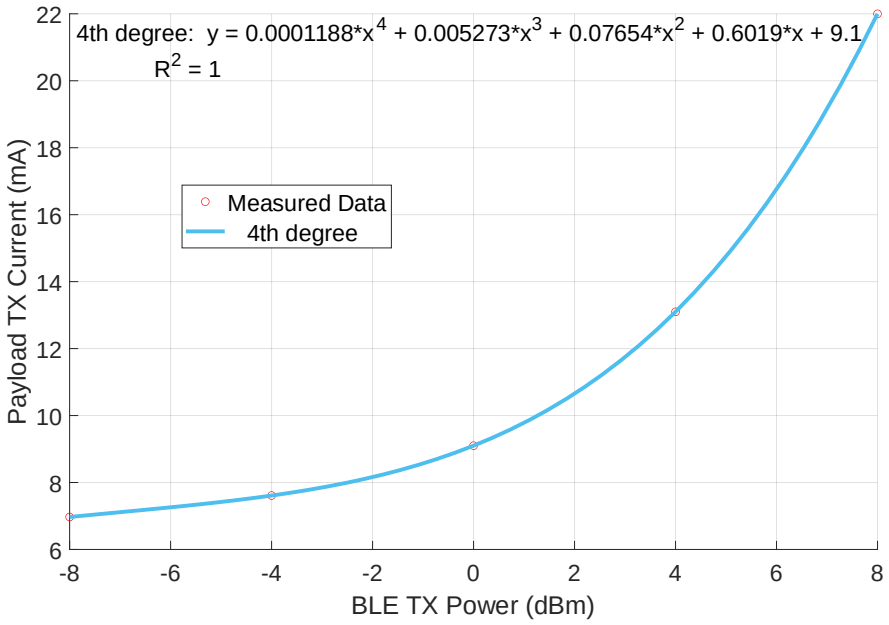



Figure 4. Relationship between data payload TX current (mA) and BLE TX power level (dBm) considering a BLE CI of 45 ms.

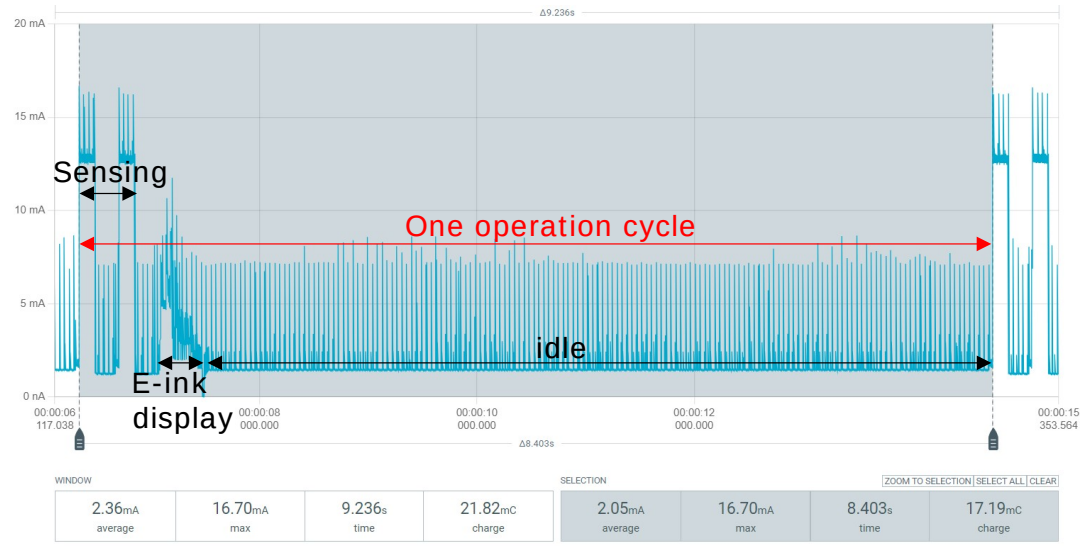
## 3.3 Transmission of Data and Reception of Commands via BLE

In this experiment, we transmit BME sensor readings and dummy location data via BLE from the node (peripheral) to a central device (e.g. mini lamp GW or BBB AP), using the Nordic Universal Asynchronous Receiver-Transmitter (UART) service. The sensor readings and dummy location data are sent as a string of characters every 5 seconds as per the format below:

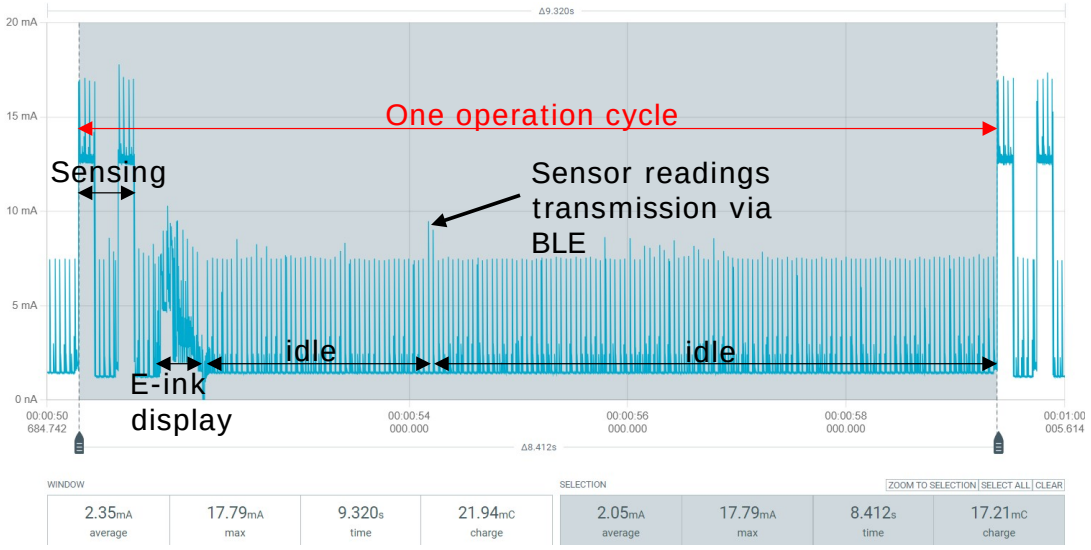
```
{
   "SID": "01",                    (source ID, i.e. node ID)
   "DID": "F1",                    (destination ID, i.e. gateway/access point)
   "T": 23.45,                     (temperature)
   "H": 56.78,                     (humidity)
   "P": 1013.25,                   (pressure)
   "G": 99.61,                     (gas)
   "XYZ": {                        (node location coordinates)
      "X": 10.25,
      "Y": 10.25,
      "Z": 10.25
   }
}
```

The data payload (e.g. JSON string shown above with all structural characters) for transmission from the node to the mini lamp GW or BBB AP is 116 bytes, which is considerably smaller than the 244 kB dummy payload analyzed in earlier sections. For clarity, the term “mini lamp GW” denotes the standalone lamp device that integrates BLE and VLC capabilities but operates without the BBB platform (see Section 3.8). Conversely, “BBB AP” refers to the mini lamp GW when it is connected to a BBB platform, thereby supporting remote data and command forwarding between the MQTT server and the nodes (see Section 3.9).

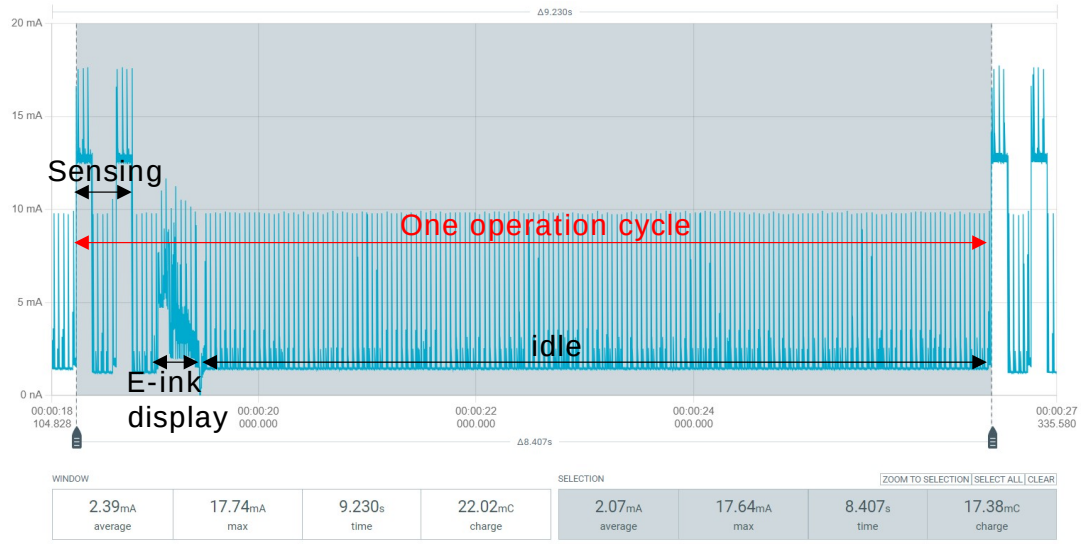
We assume a PHY rate of 2 Mbps, an MTU size of 247, and a BLE CI of 45 ms—parameters that have demonstrated optimal throughput and energy efficiency. Figure 5(a) and Figure 5(c) illustrate the node’s operation cycle, comprising sensing, E-ink display, and idle states, for BLE TX power levels of 0 dBm and 8 dBm, respectively. The results reveal only a negligible increase in average current consumption, from 2.05 mA to 2.07 mA, when the TX power level is raised from 0 dBm to 8 dBm. Figure 5(b) and Figure 5(d) present the node’s operation cycle with the transmission of data via BLE included, for the same TX power levels. Interestingly, the addition of data transmission has no measurable effect on the average current consumption during the operation cycle for both power levels. This is because the small payload size of 116 bytes can be transmitted within a single BLE connection event, leading to an insignificant impact on overall energy consumption. Thus, considering the connection parameters above, the energy required to transmit data during a BLE connection event is ~105 µJ and ~61 µJ for BLE TX power levels of 8 dBm and 0 dBm, respectively. On the other hand, as observed in section 3.2.1.4, a much larger data payload would have significantly influenced energy consumption during the operation cycle, especially at higher TX power levels such as 8 dBm.


Sensing
One operation cycle
idle
E-ink display
WINDOW
SELECTION
ZOOM TO SELECTION
SELECT ALL
CLEAR
2.36mA average
16.70mA max
9.236s time
21.82mC charge
2.05mA average
16.70mA max
8.403s time
17.19mC charge

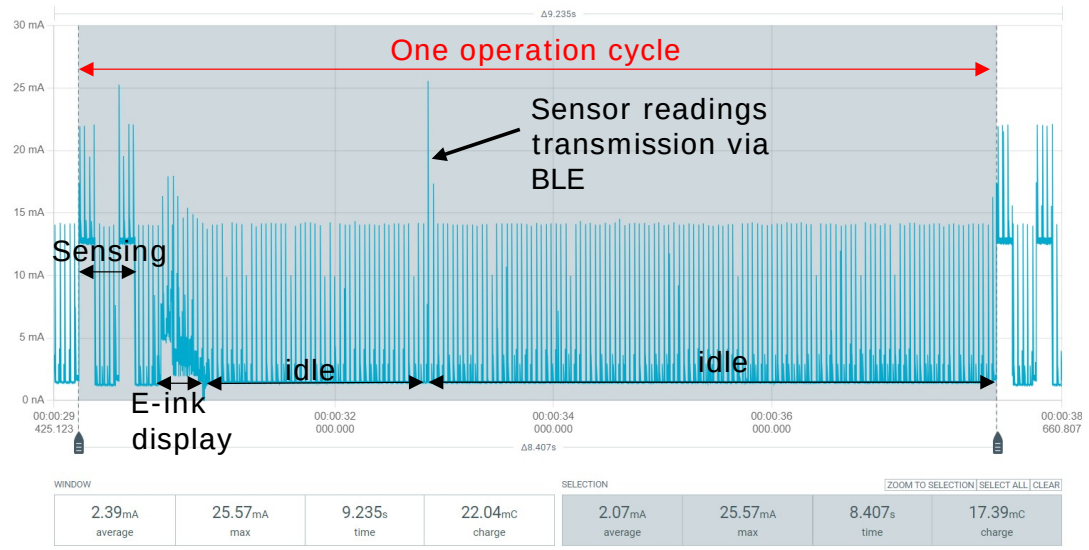
(a)


One operation cycle
Sensing
Sensor readings transmission via BLE
idle
idle
E-ink display
WINDOW
SELECTION
ZOOM TO SELECTION
SELECT ALL
CLEAR
2.35mA average
17.79mA max
9.320s time
21.94mC charge
2.05mA average
17.79mA max
8.412s time
17.21mC charge

(b)


Sensing
One operation cycle
idle
E-ink display
WINDOW
SELECTION
ZOOM TO SELECTION
SELECT ALL
CLEAR
2.39mA average
17.74mA max
9.230s time
22.02mC charge
2.07mA average
17.64mA max
8.407s time
17.38mC charge

(c)




(d)

Figure 5. Current profiles during the node's operation cycle (PHY rate = 2 Mbps, MTU = 247, BLE CI = 45 ms): (a) without data transmission (TX power level = 0 dBm), (b) with data transmission (TX power level = 0 dBm), (c) without data transmission (TX power level = 8 dBm), (d) with data transmission (TX power level = 8 dBm).

In terms of data reception, we expect the node to receive commands from the mini lamp GW or BBB AP via BLE or VLC to perform specific tasks. Similar to the data transmission, we observe from Figure 6 that receiving a command via BLE from the mini lamp GW or BBB AP happens over only one BLE connection event and would cause a negligible increase in overall current consumption. The energy consumed during the reception of the command via BLE can vary between 32 µJ to 37 µJ, depending on the BLE received (RX) power level.

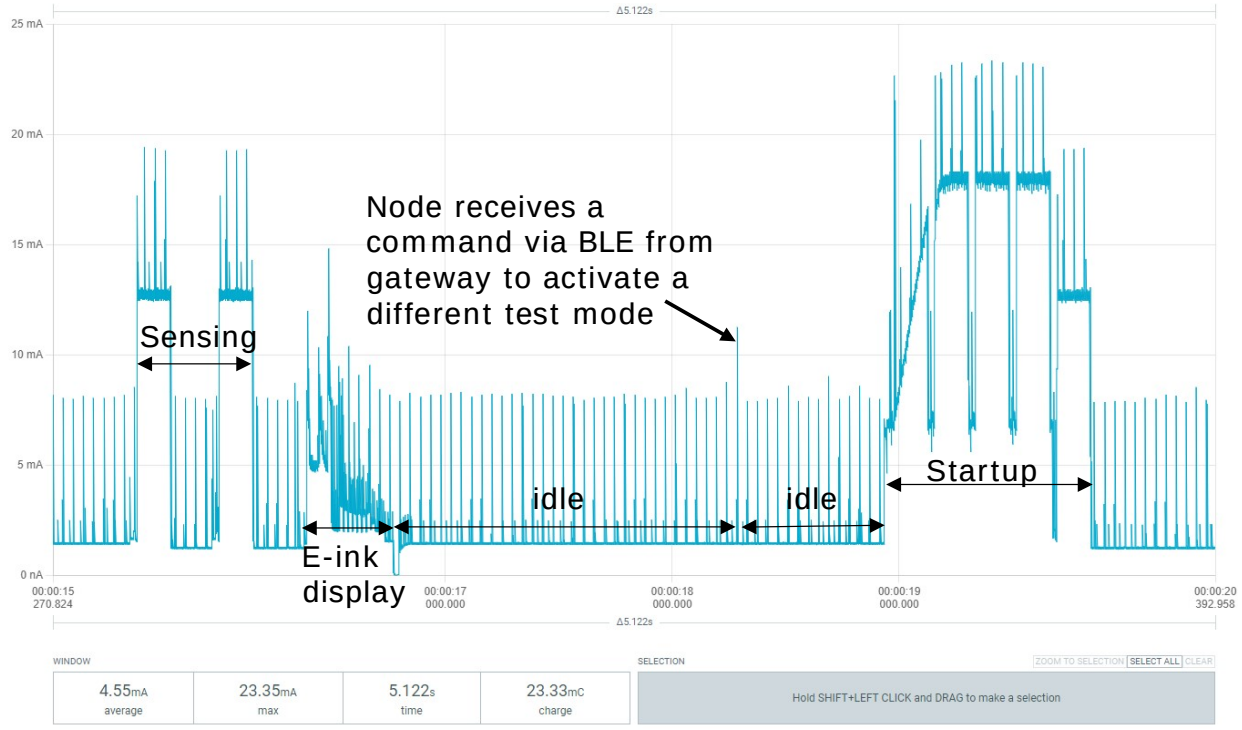



Figure 6. Illustration of the current profile of a node receiving a command via BLE from mini-lamp GW (TX power level = 0 dBm, PHY rate = 2 Mbps, MTU = 247, CI = 45 ms).

## 3.4 Energy Comparison Between BLE Advertising and Connected Modes

One interesting question is whether the mini lamp GW or BBB AP should disconnect from the node after sending a command (e.g., to activate a specific node operation or request data from the node) or remain in BLE-connected mode with the node throughout (considering the case where the node does not turn its BLE functionality off and go into a deep sleep mode).

Figure 7 illustrates three scenarios where the RIoT node performs sensing, E-ink display updates, and data transmission tasks, with BLE functionality enabled:

- Scenario 1 (Figure 7(a)): After the mini lamp GW disconnects, the node begins BLE advertising at an interval of 152.5 ms (assuming fast BLE advertising at 20 ms is disabled) while continuing its operations. In this scenario, the mini lamp GW does not receive any data.
- Scenario 2 (Figure 7(b)): The mini lamp GW remains connected to the node with a BLE CI of 152.5 ms, allowing data to be also received on the mini lamp GW.
- Scenario 3 (Figure 7(c)): Similar to Scenario 2, but the BLE CI is set to 45 ms (default value).

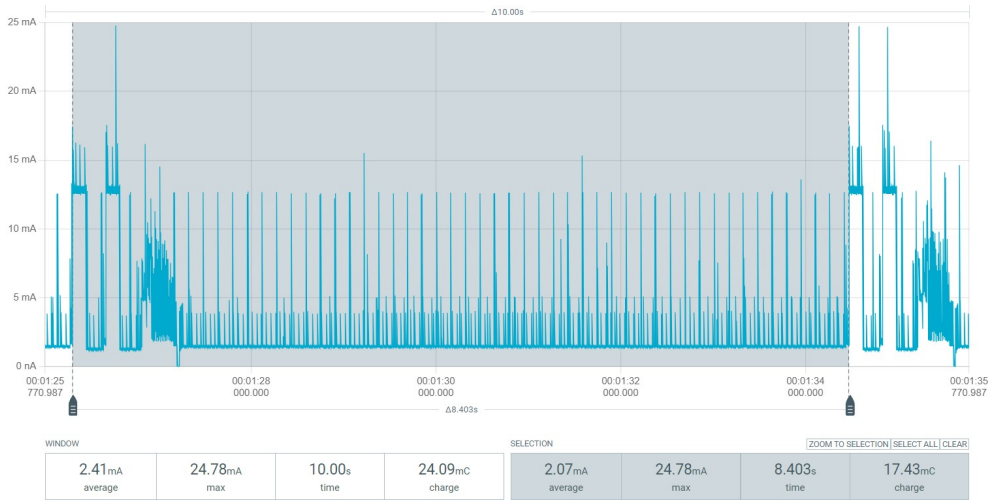
In all three scenarios, the BLE TX power level is set to 0 dBm. The current consumption results show:

- In Scenario 1, where the node advertises at a 152.5 ms interval, the overall current consumption during the operation cycle is 2.07 mA.
- In Scenario 2, with a BLE CI of 152.5 ms, the current consumption decreases to 1.99 mA.
- In Scenario 3, with a BLE CI of 45 ms, the current consumption is 2.04 mA.

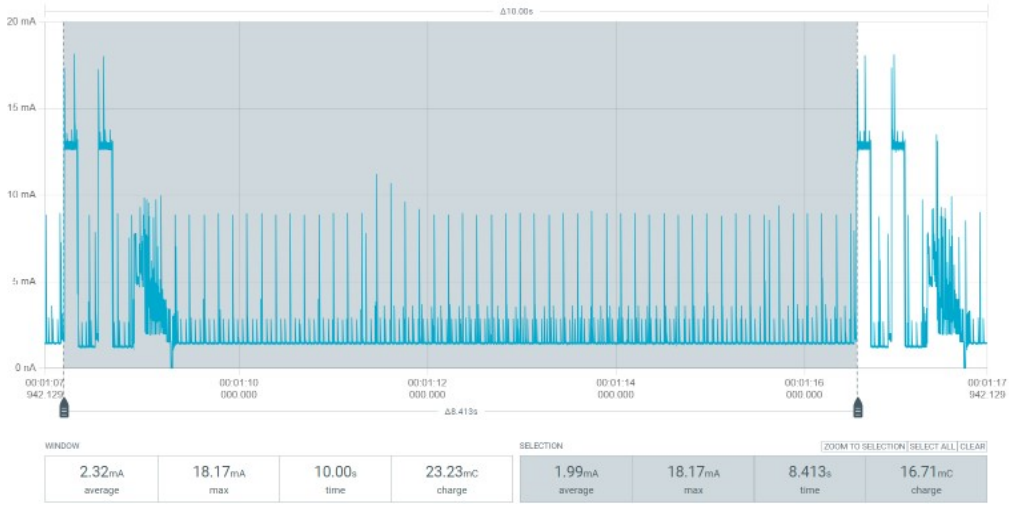
Interestingly, even when the BLE CI is reduced to 45 ms, the current consumption remains lower than the advertising scenario at 152.5 ms. When the BLE TX power level is increased to 8 dBm, the current consumption trends are amplified:

- Scenario 1 (Advertising, 152.5 ms): 2.25 mA
- Scenario 2 (Connected, 152.5 ms): 2.00 mA
- Scenario 3 (Connected, 45 ms): 2.07 mA

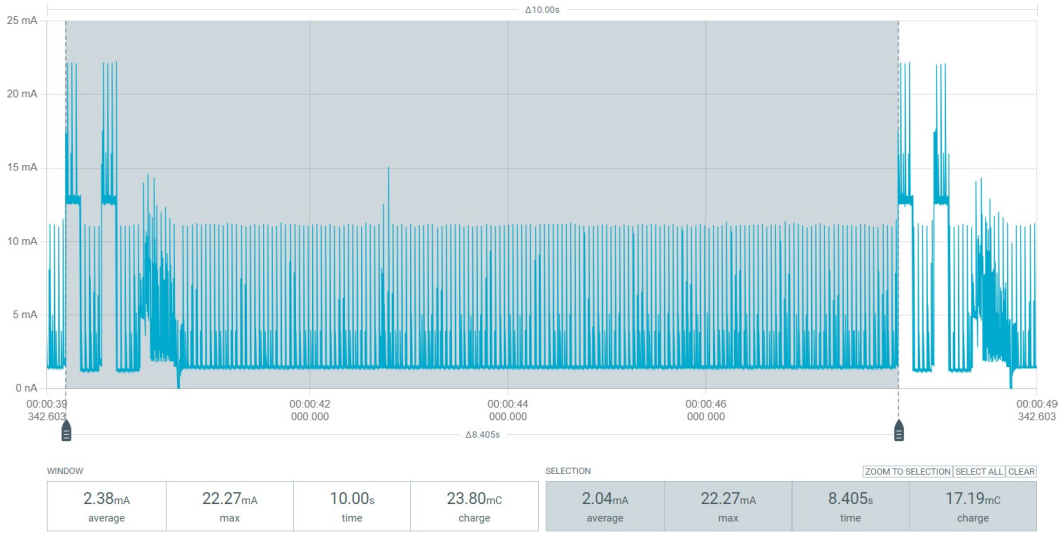
These results indicate that BLE advertising consumes more energy compared to maintaining a BLE connection, even with frequent communication intervals. The higher energy cost of advertising is due to its continuous broadcasting of advertisement packets on three channels (37, 38, and 39). Each advertising event involves transmitting the same packet multiple times across these channels to ensure visibility and reliability, allowing the radio to remain active for longer durations. While the energy differences may appear minor, it is important to consider the additional energy overhead incurred by the mini lamp GW when it disconnects from the node. Upon disconnection, the mini lamp GW enters BLE scanning mode to search for advertising nodes, and this consumes more energy than remaining in BLE-connected mode, as will be discussed later.




(a)




(b)




(c)

Figure 7. Current profiles of node during different operations (BLE TX power = 0 dBm): (a) Node is disconnected from mini lamp GW and advertising at an interval of 152.5 ms, (b) Node remains connected to mini lamp GW and BLE CI is 152.5 ms, (c) Node remains connected to mini lamp GW and BLE CI is 45 ms.

## 3.5 Transmission of Data and Reception of Commands via VLC

In the original firmware, the actual useful data/command per transmission is only 8-bits (as per command/data field in Figure 8) and 75% (remaining 24 bits) of the packet is for redundancy/error checking. For test purposes, only a dummy character of 8-bit (e.g. 0x20) was transmitted and received via VLC. The packet structure in Figure 8 is not ideal for sending and receiving data payloads such as sensor readings and location data (coordinates). Thus, we modified the initially implemented NEC protocol (which consists of 8-bit segments: address, complement of address, data/command, complement of data/command).

Let's say we want to transmit sensor readings (as per BME sensor ranges), and location coordinates (X, Y, Z) as follows:

- Temperature in °C: -40.0 to 85.0°C (2 bytes needed)
- Pressure in hPa: 300.0 to 1100.0 hPa (2 bytes needed)
- Humidity in %: 0.0 to 100.0% (2 bytes needed)
- Gas resistance in kΩ: 0.0 to 200.0 kΩ (2 bytes needed)
- Location coordinates range (X, Y, Z): -10.00 m to 10.00 m (2 bytes each, thus 6 bytes needed)

At least 14 bytes will be required to transmit the sensor readings (with one decimal point precision) and location coordinates (with two decimal points precision). Using the original unoptimized packet structure in Figure 8 to transmit and receive this data payload would lead to inefficient transmission times and significantly reduced energy efficiency since the actual data (e.g. sensor/location) would take much more than 8 bits. Thus, many packets such as in Figure 8 need to be exchanged between the node and mini lamp GW for transmitting the actual data/commands (also considering that the packets need to be interleaved with a small delay to allow proper decoding). This will undeniably consume more energy, and the data transmission will be slower, as compared to BLE.


Figure 8. Initial NEC packet structure for transmitting an 8-bit data/command.

### 3.5.1 Optimized VLC Frame Structure

We define a new frame structure for efficient transmission and reception of commands and data. The full 32 bits of the NEC protocol are utilized to transmit the frame in a few manageable chunks. As illustrated in Figure 9, the frame includes start and end markers, allowing the decoder to easily identify the beginning and end of the frame when it receives the chunks. The frame also includes source and destination address fields, enabling the node or mini lamp GW to verify whether the frame is intended for it; if not, it can simply ignore the frame. The command field allows the mini lamp GW to send specific instructions to the node, such as requesting sensor readings, location coordinates, or both. As for the node, the command field indicates what data it is sending to mini lamp GW. Other fields in the frame include the payload size, which aids in decoding the frame, and a checksum for ensuring the integrity of the frame. The actual payload can be up to 16 bytes in size, allowing for additional data such as altitude readings from the BME environmental sensor. All other fields in the frame are 1 byte in size.

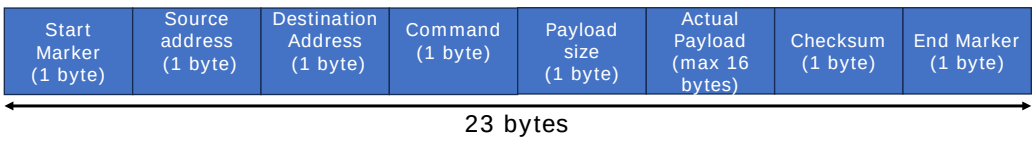



Figure 9. Frame structure for transmitting and receiving data/commands between node and mini lamp GW or BBB AP.

Since the frame structure is 23 bytes in size, and each chunk of the frame is 32 bits, the maximum number of chunks that will be transmitted and received is 6 (after ceiling operation), as shown in Table 5. A minimum inter chunk delay of 100 ms is configured to ensure that the decoder has enough time to receive the chunks and decode them appropriately.

Table 5. Layout of frame chunks for transmitting VLC data.

| **Frame chunk** | **Content** | **Description** |
|---|---|---|
| Chunk 0 | [startMarker][sourceAddr][destAddr][command] | Frame Header |
| Chunk 1 | [payloadSize][payload[0:2]] | Payload Size & Start of Payload |
| Chunk 2 | [payload [3:6]] | Payload Data |
| Chunk 3 | [payload [7:10]] | Payload Data |
| Chunk 4 | [payload [11:14]] | Payload Data |
| Chunk 5 | [payload[15]][checksum][endMarker][padding] | End of Payload & Frame Footer |

The last chunk may not be completely filled with valid data, especially since the total frame size (23 bytes) does not perfectly fit into 32-bit chunks. Padding is involved in the last chunk to ensure that the data aligns correctly within the 32-bit chunks. The implemented protocol takes care to clear any unused bits in the last chunk using a mask to avoid sending irrelevant data. The decoder decodes the received chunks of data through a series of steps that involve processing the timing data, validating the received frame, and extracting the relevant information based on the command type. If the start marker (e.g. 0xAA) is detected, it indicates that a new frame is being received. If the decoder is in the receiving state, it stores the processed chunk in a buffer array. Once all chunks have been received, the decoder validates the complete frame. This involves checking the start and end markers, destination address to ensure it matches the receiver's address, payload size against the maximum allowed size, command type to ensure it is valid, and checksum to verify data integrity. If the frame is valid, the decoder processes the frame based on the command type. For example:

For **CMD_SENSOR** command, it extracts and validates the sensor data.

For **CMD_LOCATION** command, it extracts and validates the location data.

For **CMD_COMBINED** command, it extracts both sensor and location data.

For request commands (**CMD_REQ_SENSOR**, **CMD_REQ_LOC**, **CMD_REQ_BOTH),** it prepares a response frame with the requested data.

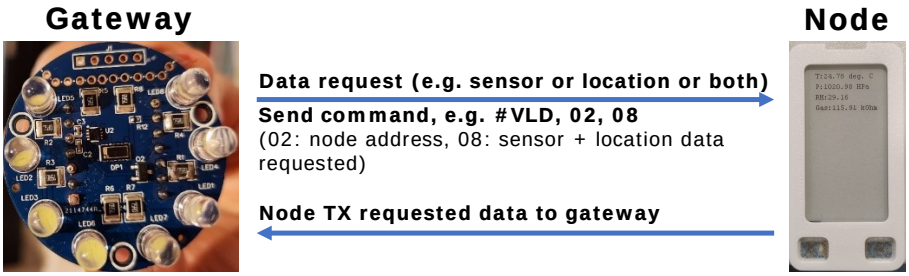



Figure 10. Communication between mini lamp GW and node via VLC.

The GW (mini lamp standalone) or AP (mini lamp integrated in BBB platform) can send commands to the nodes via BLE or VLC. As shown in Figure 10, we implemented functions for the mini lamp GW to send commands and receive data from the node via VLC, using the frame

structure defined in Figure 9. The mini lamp GW will also use the same frame structure as the node. However, it will not transmit any payload data but only a command to request specific data from node, either sensor-only, location-only, or both. In the setup shown in Figure 10, the mini lamp GW and node constantly listen to and decode any VLC frames being transmitted and received across the network. This will have certain energy implications as we shall see in the following section.

### 3.5.2 Energy Implications of Concurrent VLC and BLE Operation

Depending on the demonstration scenario, the node might need to have dual mode of communication (BLE and VLC) active at the same time in order to receive commands from the mini lamp GW. This means that the node constantly needs to listen to incoming BLE and VLC frames (in downlink). For the node to listen and decode the received VLC frames, the *nbvlc_on()* function has to be used in the *void_loop()* all the time, which will increase the current consumption, as we have investigated in deliverable D2.4. For example, let's consider the following cases:

1) **Case 1 – Node listens to commands from mini lamp GW transmitted via both BLE and VLC**

For this test, we disabled the fast BLE advertising mode (interval = 20 ms) to prevent the node from initiating fast BLE advertising for a 30-second timeout period (the default) each time the mini lamp GW disconnects. Fast BLE advertising consumes more energy compared to slow mode. Since the node needs to listen for commands from both communication modes, the *nbvlc_on()* function must be called within the *void_loop()* function of the firmware. Without this, the node will not be able to detect or decode any VLC frames. The default BLE TX power level is 0 dBm. The node operates in slow BLE advertising mode (advertising interval = 152.5 ms) and remains idle. When the mini lamp GW sends a BLE command requesting the node to perform sensing and E-ink display update tasks, the mini lamp GW disconnects from the node after a delay of approximately 5 seconds. Upon receiving the command, the node switches back to slow BLE advertising mode (since fast BLE advertising is disabled) while performing the requested tasks. After some time, the mini lamp GW sends another command via VLC, requesting the node to transmit sensor and location data. The current consumption profile for this test case is shown in Figure 11.

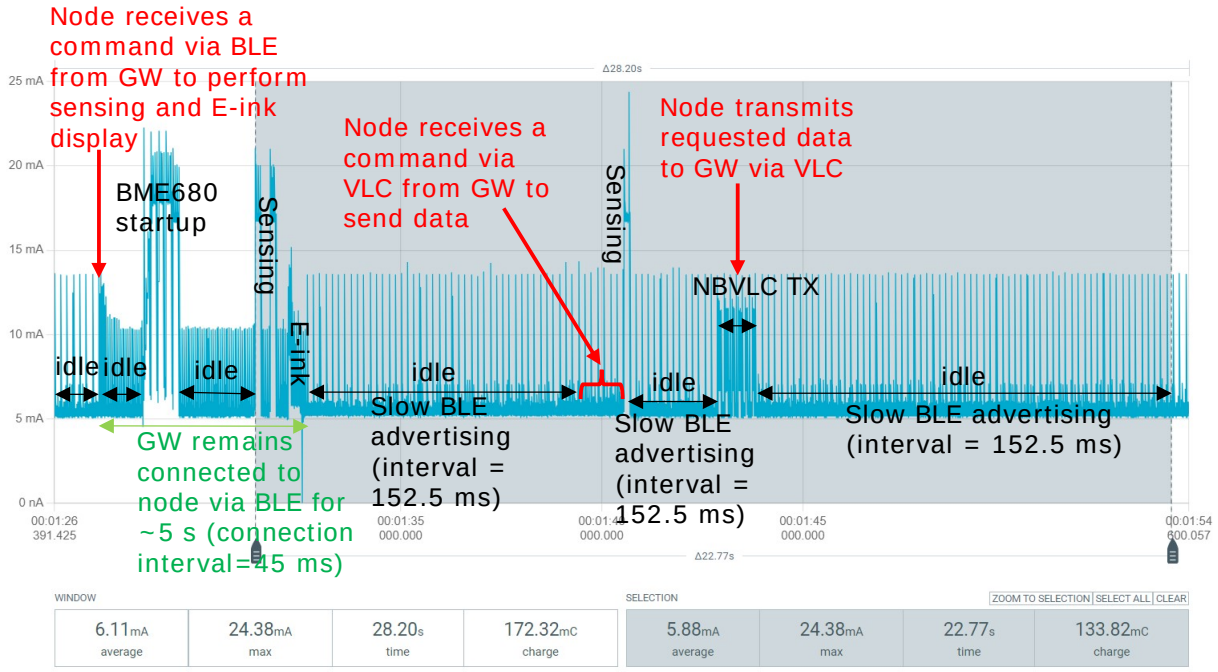



Figure 11. Current consumption profile for Case 1: Node listens to commands from mini lamp GW via both BLE and VLC and then performs its operations accordingly.

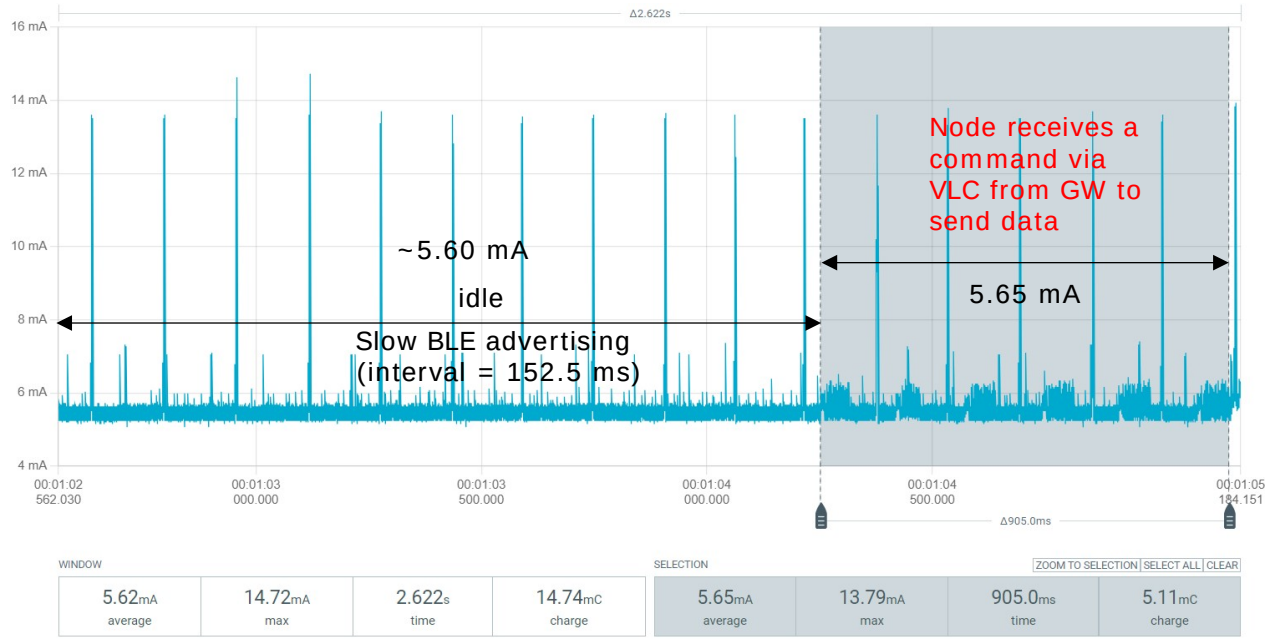



Figure 12. Close-up look at current consumption profile for Case 1 when node receives a command via VLC from mini lamp GW, while node is in slow BLE advertising mode (interval = 152.5 ms).

**2) Case 2 – Node operates with periodic duty cycle and both BLE and VLC enabled, without listening to commands from mini lamp GW**

In this scenario, the node does not actively listen for commands from the mini lamp GW via either BLE or VLC. However, it still performs the same operations as in Case 1, including sensing, E-ink display update, and transmitting data via VLC to the mini lamp GW, all on a periodic duty cycle (see Figure 13). For a fair comparison of energy consumption, we configure the operation intervals to match those in Case 1 exactly. Since the node is not required to constantly listen to commands, we can apply the *nbvlc_off()* function in the firmware code. However, since VLC transmission (TX) still occurs in the uplink, we need to call the *nbvlc_on()* function just before sending a VLC frame chunk and immediately follow it with *nbvlc_off()*. This ensures that the node does not switch to a high-power mode during the short delays (idle periods) between the chunks, which are crucial for effective frame decoding.

If we compare the energy consumed during the operation duty cycle in Case 2 (Figure 13) with the same operations at the same intervals in Case 1 (see grey shaded area in Figure 11), we observed that in Case 1 the energy consumed over the period of 22.77 seconds is 0.442 J (considering node's operating voltage of 3.3 V), while in Case 2 the energy consumed is only 0.143 J (approx. 68% less energy consumption). In both Case 1 and Case 2, BLE and VLC are used; however, data transmission occurs in the uplink via VLC (instead of BLE).

A closer examination of the power consumption in Case 1 (refer to Figure 14) reveals the inefficiencies of VLC-based communication when BLE is already active on the node. When the node receives a command from the GW via VLC and subsequently transmits the requested data using VLC, the energy consumption is 16.9 mJ for reception and 21.2 mJ for transmission (assuming a 3.3V operating voltage). This is considerably higher than the energy required for BLE-based communication, as demonstrated in Section 3.3. From Figure 11, we observe that when the node receives the VLC command over a period of ~905 ms, the current consumption is slightly increased from 5.60 mA (idle, slow BLE advertising mode) to around 5.65 mA.

In Case 2, where the node follows a periodic duty cycle without constantly listening to commands from the GW, the VLC-based uplink transmission consumes an energy of 14.3 mJ (as seen in Figure 15). However, even in this scenario, BLE remains the better option for data transmission. From an energy optimization perspective, if BLE is already enabled, it is preferable to use BLE for both transmitting and receiving data/commands (since the timer functions of the VLC protocol

are energy-intensive when constantly running in the firmware). Thus, disabling VLC functionality (especially in the downlink) when it is not needed can lead to significant energy savings, as demonstrated in Case 2. In environments where BLE communication is feasible, VLC can be entirely repurposed—rather than being used for data/command exchange; the mini lamp GW can instead serve as a source for energy harvesting. This approach further enhances the energy efficiency of the system while maintaining the benefits of dual-modality functionality.

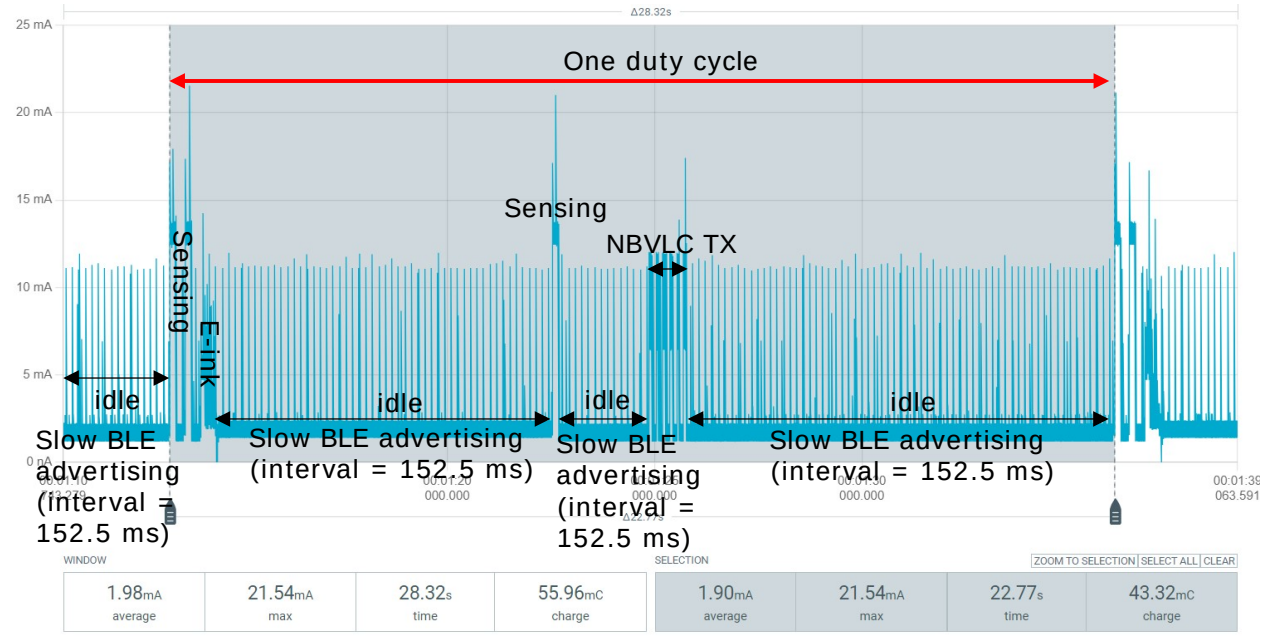



Figure 13. Current consumption profile for Case 2: Node does not listen to commands from mini lamp GW and performs its operations in a periodic duty cycle.

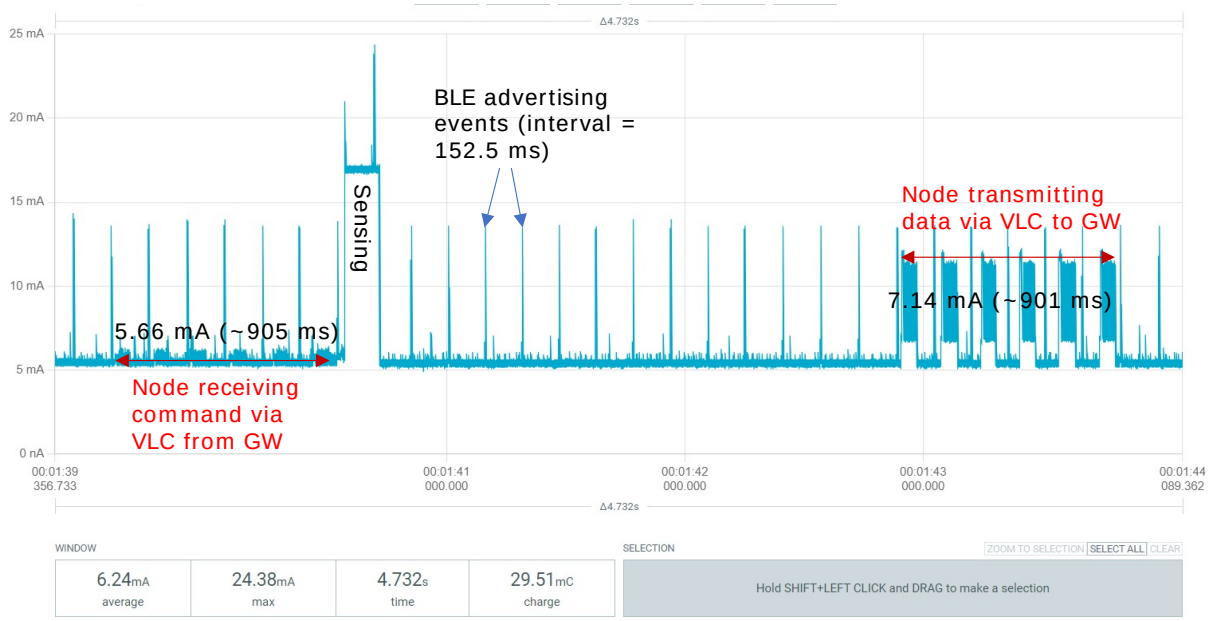



Figure 14. Close-up look of the node's current profile in Case 1 during the VLC reception and transmission of command and data, respectively, between node and mini lamp GW (node is BLE advertising with an interval of 152.5 ms and BLE TX power level of 0 dBm).

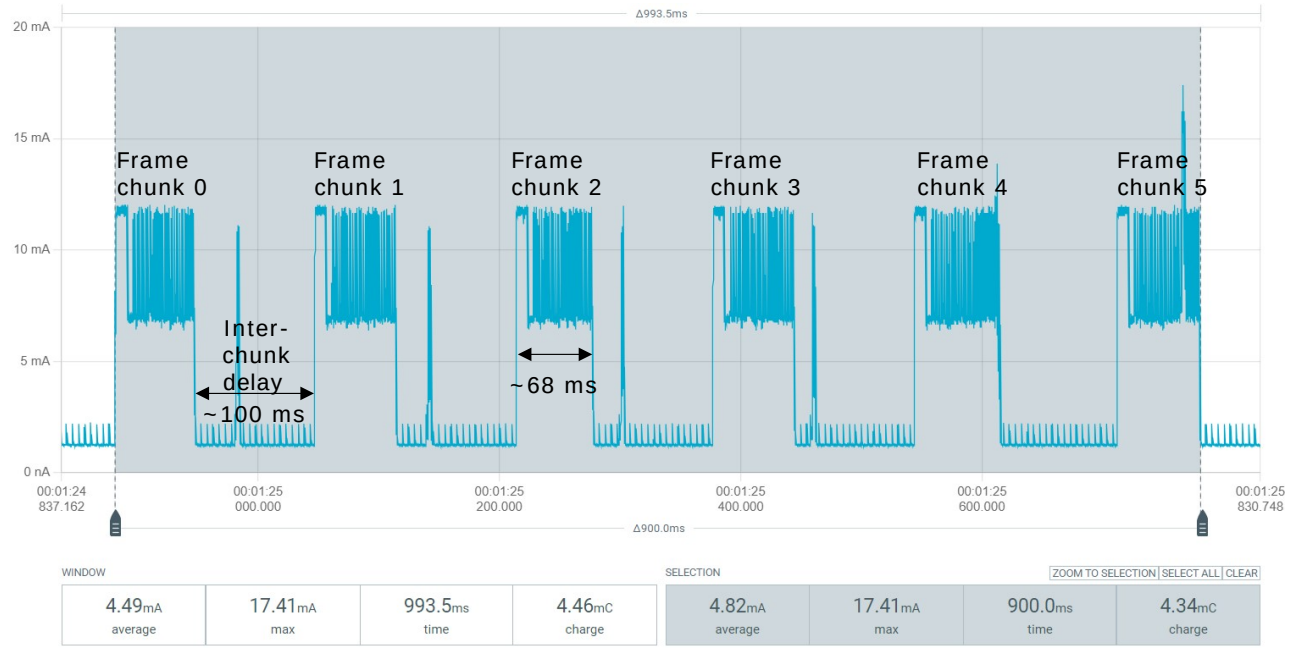



Figure 15. Close-up look of the node's current profile in Case 2 during the VLC transmission of data (sensor + location) to mini lamp GW (node is BLE advertising with an interval of 152.5 ms and BLE TX power level of 0 dBm).

### 3.5.3 Energy Consumption During VLC TX in Very Low Power Mode (BLE OFF, VLC Uplink ON)

Figure 16 illustrates the current profile of the very low power firmware with BLE disabled and VLC uplink TX functionality enabled. For this test, measuring point U9 remains short circuited while U6 is cut (more details in deliverable D2.4). In this test, we assume that the node does not receive any command from the mini lamp GW. Basically, the node periodically wakes up from deep sleep and performs sensing, E-ink display update, and transmission of the sensor data through VLC (as per frame structure defined in Section 3.5). In this case, the data is transmitted as a broadcast (e.g. address: 0xFF), and any mini lamp GW within range can decode the received VLC data and know which node has transmitted the data. In Figure 16(a), the energy consumed during the VLC TX is ~16.1 mJ. In this case, the VLC RX is assumed to remain active (U9 shorted) although the node will not receive any commands from the mini lamp GW. Note that the deep sleep current can be further optimized in this scenario by cutting U9 (VLC RX) as well, and the deep sleep current will drop to around 202 µA, as shown in Figure 16(b). In practice, we will not physically cut the VLC RX module. The deactivation of the VLC RX component can be achieved by soldering in a 0 Ohm resistor to allow powering of the circuitry by switching the relevant nRF52833 General Purpose Input/Output (GPIO) pin to HIGH during normal operation, and to shut down the circuit by switching the pin to LOW during sleep. If both BLE and VLC modalities were deactivated and the node performs only sensing and E-ink display

update operations, we can achieve a deep sleep current around 5 µA when both U6 and U9 are cut, as reported in deliverable D2.4.

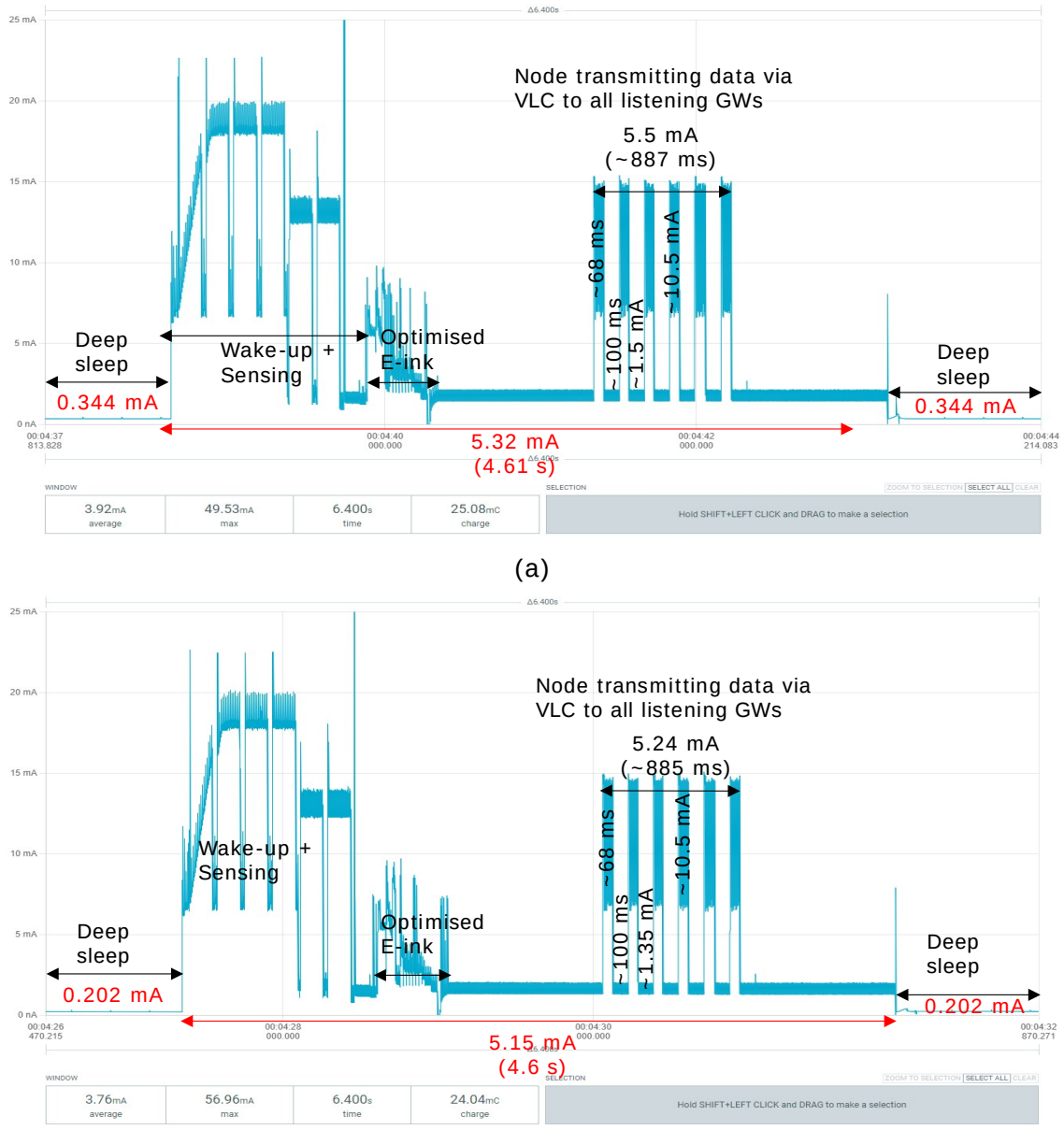



Figure 16. Current of very low power firmware with VLC uplink functionality enabled (BLE OFF): (a) U6 cut, U9 shorted, (b) both U6 and U9 cut.

### 3.5.4 VLC Throughput and Energy Analysis

In Figure 16, we observed that the VLC data transmission time for each chunk of the frame (32-bit NEC packets) takes around 68 ms. If we consider the initial NEC packet in Figure 8 where only 8 bits of useful data or command is transmitted through the NEC protocol (out of the 32 bits), then the effective throughput would be around 8/0.068 = 118 bps. Since the actual data (sensor readings and location coordinates) will take much more than 8 bits as per Section 3.5, say a maximum payload of 16 bytes if we also consider an additional 2 bytes for altitude sensor readings, and we consider a delay of 100 ms between each NEC packet (to allow enough time for the decoder to successfully decode the NEC packets), we can compute the effective throughput as follows (assuming we transmit all sensor readings and location coordinates):

1. Total useful data payload, $D$ = 16 bytes
2. Number of NEC packets needed, $N = \left(\frac{D \times 8}{B}\right) = \left(\frac{16 \times 8}{8}\right) = 16$ (excluding any frame synchronization markers needed to detect and decode VLC frames).

where $B$ is the number of useful bits per NEC packet. Note that out of the 32 bits in the initial NEC packet in Figure 8, only 8 bits are used for the actual command/data transmission.

3. Total time to transmit data payload (excluding delays between packets), $T_{\text{tx}} = N \times t_{\text{packet}} = 16 \times 68 \text{ ms} = 1.088 \text{ seconds}$.

where $t_{\text{packet}}$ is the transmission time of one NEC packet.

4. Total delay between NEC packets, $T_{\text{delay}} = (N - 1) \times t_{\text{delay}} = (16 - 1) \times 100 \text{ ms} = 1.5 \text{ seconds}$.

where $t_{\text{delay}}$ is the delay between NEC packet transmissions.

5. Total duration to transmit the whole data payload, $T_{\text{total}} = T_{\text{tx}} + T_{\text{delay}} = 2.588 \text{ seconds}$.
6. Effective throughput = $\left(\frac{D \times 8}{T_{\text{total}}}\right) = \left(\frac{16 \times 8}{2.588}\right) = 49.5 \text{ bps}$.
7. Average current consumption over the entire transmission period,

$$I_{\text{avg}} = \frac{(I_{\text{tx}} \times T_{\text{tx}}) + (I_{\text{delay}} \times T_{\text{delay}})}{T_{\text{total}}}$$

where $I_{\text{tx}}$ is the current consumed during the transmission of one NEC packet, $I_{\text{delay}}$ is the current consumed during the delay (idle) period between NEC packet transmissions.

Considering the very low power firmware where BLE is OFF and VLC uplink is activated, and assuming U6 and U9 are both cut as in Figure 16(b), the energy during VLC transmission is computed as:

8. VLC transmission energy = $I_{\text{avg}} \times V_{\text{oper}} \times T_{\text{tx}} = \frac{(10.5 \text{ mA} \times 1.088 \text{ s}) + (1.35 \text{ mA} \times 1.5 \text{ s})}{2.588 \text{ s}} \times 3.3 \text{ V} \times 2.588 \text{ s} = 44.4 \text{ mJ}$

On the other hand, if we consider the optimized VLC frame structure (23 bytes total) in Figure 9, the actual data payload of 16 bytes will be transmitted in only 6 chunks of 32 bits each (considering the additional overhead such as addresses, payload size, checksum, start and end markers). As illustrated in Figure 16(b), the total VLC transmission time will be around 0.885 seconds. Thus, the effective throughput and VLC transmission energy will be around 145 bps and 16 mJ, respectively. Thus, it is evident that the VLC frame structure (23 bytes total) defined in Figure 9 is much more efficient than the initial NEC protocol in terms of both effective throughput and VLC transmission energy.

## 3.6 Overview of Node's Energy Analysis Under Different Operating Modes

In this section, we estimate the energy consumption of the node under different scenarios using the implemented Energy Consumption Prediction App developed in Task 2.4.1 (refer to node's optimization details, energy models, and tool description in deliverable D2.4, [1] and [2]). We made some updates to the app to incorporate VLC TX through multiple frame chunks because the data (sensor readings and/or location coordinates) that needs to be transmitted to the mini lamp GW or AP will not fit within a single NEC packet of 32 bits, as detailed in Section 3.5.

### 1) Scenario 1: BLE and VLC activated, and node listens to commands from mini lamp GW to perform tasks (full hybrid communication configuration)

This scenario represents the worst case in terms of power consumption. Here, the node performs all its core operations, including BLE, VLC, sensing, and E-ink display updates and receives commands from the mini lamp GW via both BLE and VLC. However, in this scenario, we assume that the node does not go into a deep sleep mode at any point in time. As analyzed in Section 3.5.2, if the node must continuously listen for incoming VLC frames from the GW, it must keep

the *nbvlc_on()* timer function active at all times. This significantly increases power consumption. The operational sequence for this scenario is as follows:

1. The node powers on and initiates fast BLE advertising with a default interval of 20 ms. After 10 seconds, a mini lamp GW establishes a BLE connection with the node, at a BLE CI of 45 ms.
2. The node remains in an idle BLE-connected state for another 5 seconds, maintaining a BLE TX power of 0 dBm.

   As discussed in Section 3.4, keeping the node in BLE-connected mode is more energy-efficient than frequently disconnecting and reconnecting. Since the mini lamp GW can manage multiple concurrent connections with peripheral devices (nodes), it is also more efficient for the mini lamp GW to remain in BLE-connected mode rather than switching to BLE scanning mode, which consumes more power (as we shall see later). If the mini lamp GW disconnects, the node will reinitiate fast BLE advertising (20 ms interval) before switching to slow advertising (152.5 ms interval), leading to higher energy consumption. This default behavior can however be modified in the firmware to reduce energy usage.
3. The mini lamp GW sends a BLE command (by writing to the appropriate node's BLE characteristic value) to initialize the BME environmental sensor (also referred to as startup state). The initialization process takes approximately 916 ms.
4. After a 10-second idle period in BLE-connected mode following startup, we assume that the mini lamp GW starts sending commands via VLC at regular intervals (e.g., every 1 minute or 1 hour). Basically, for each command transmitted by mini lamp GW, the node will perform the following operations: Sensing (516 ms), idle (1 second), optimized E-ink display update (435 ms), idle (1 second), uplink VLC transmission of sensor and/or location data (908 ms). Then the node will be idle (BLE-connected only mode) until the next command is received from mini lamp GW after either 1 minute or 1 hour.

A visual representation of these states and their corresponding instantaneous current consumption is provided in Figure 17(a) for Scenario 1. The initial states include BLE advertising, BLE-connected idle mode, sensor startup, and post-startup idle period.

**2) Scenario 2: Periodic duty-cycled operation with BLE uplink/downlink and VLC uplink (partial hybrid communication configuration)**

This scenario closely resembles Scenario 1, with the key difference being that the node operates independently, following a periodic duty cycle for tasks such as sensing, E-ink display updates, and VLC data transmission to the mini lamp GW in the uplink, without listening for commands from the mini lamp GW (i.e. no downlink communication). Alternatively, the node can also receive commands from the mini lamp GW (in downlink) via BLE only. Therefore, if the node does not need to listen for VLC frames from the mini lamp GW, the VLC functions can remain disabled and be activated only just before transmitting each VLC frame chunk (as described in Section 3.5.2, see Case 2). The time intervals and BLE parameters remain consistent with those in Scenario 1. Note that, as previously mentioned, if BLE is available, it would remain the preferred method for data transmission over VLC. For experimental analysis and evaluation, we included this scenario to highlight the significant impact on energy consumption when VLC functions are utilized, when BLE functionality is also enabled—particularly when continuously listening for commands via VLC, as seen in Scenario 1. Note that in this scenario, when the node transmits the data via VLC, it will be sent as a broadcast, and any mini lamp GW listening to incoming VLC frames will be able to decode the data and identify which node has sent it. As for Scenario 1, the VLC data transmission from the node will be targeted to the intended recipient (i.e. mini lamp GW or BBB AP) using its address field, while other mini lamp GWs will see that the data is not intended for them and will ignore the frame. A visual representation of the states and their corresponding instantaneous current consumption is shown in Figure 17(b) for Scenario

2. As in the previous scenario, the initial states when the node first powers on include BLE advertising, BLE-connected idle mode, BME sensor startup, and the post-startup idle period.

**3) Scenario 3: BLE on and VLC off, and node performs tasks in a periodic duty cycle**

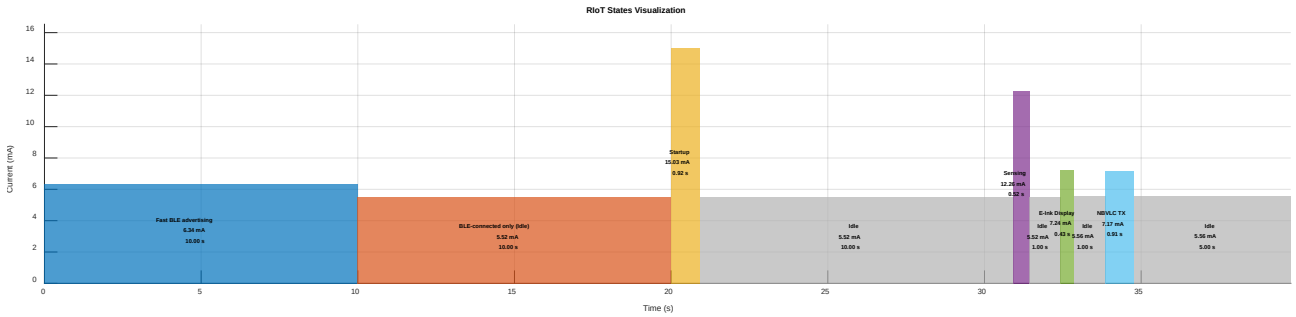
In this scenario, the VLC functionality is disabled and data transmission is handled via BLE only. The other operations remain similar to Scenarios 1 and 2. The time intervals and BLE parameters also remain consistent with those in the previous scenarios. Since only BLE functionality is enabled, it makes no significant difference whether the node operates on a periodic duty cycle or receives commands from the mini lamp GW or BBB AP at regular intervals via BLE. It is expected that there will be negligible energy consumption differences between the two cases, as BLE transmission and reception of commands/data between the node and mini lamp GW incur insignificant energy consumption, as demonstrated in Section 3.2. As in the previous scenarios, the initial states upon powering on include BLE advertising, BLE-connected idle mode, BME sensor startup, and the post-startup idle period. A visual representation of the states and their corresponding instantaneous current consumption is shown in Figure 17(c) for Scenario 3.

**4) Scenario 4: BLE off and VLC uplink on, and node performs tasks in a periodic duty cycle, including deep sleep state**

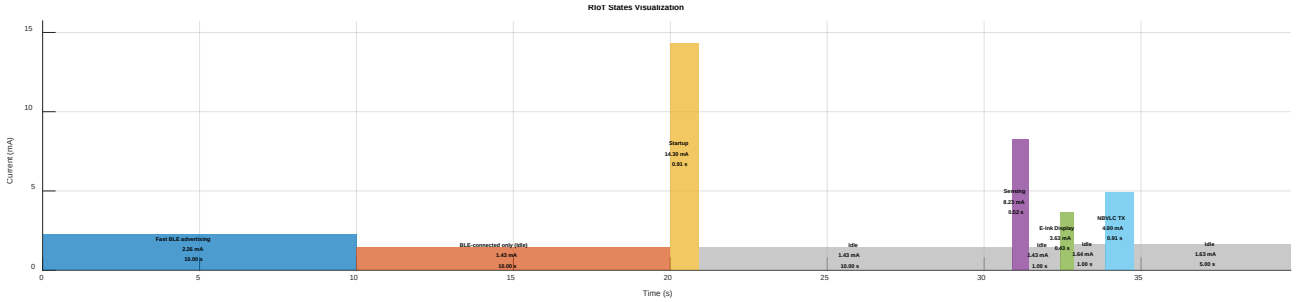
In this scenario, the node performs its basic operations such as sensing and optimized E-ink displaying, while BLE is deactivated and only VLC is used to transmit data in the uplink to a mini lamp GW in a periodic duty cycle. After carrying out its main operations (active states), the node will go into a deep sleep state. Since the node does not have to constantly listen to mini lamp GW's commands via VLC, the VLC RX will be unused. Thus, we consider the hardware optimization configuration where points U6 and U9 are both cut on the Si-based node. Recall, measuring point U9 is dedicated to measuring current to the VLC RX module only, while U6 is not used for current measurement but can be cut to reduce the deep sleep current while the E-ink display is powered off. The operation duty cycle consists of the following states: Startup (911 ms)→ Idle for 30 ms → Sensing (149 ms)→ Idle for 250 ms → Optimized E-ink displaying (0.450 s) → Idle for 250 ms→ VLC TX (908 ms)→ Deep sleep (1 min or 1 hour). Note that compared to the previous scenarios, the startup (or wake-up) state will be part of the operation cycle and will be repeated over the node's operating time. An illustration of the various states in Scenario 4 and their instantaneous current consumption is shown in Figure 17(d).

**5) Scenario 5: Both BLE and VLC off, and node performs tasks in a periodic duty cycle, including deep sleep state**

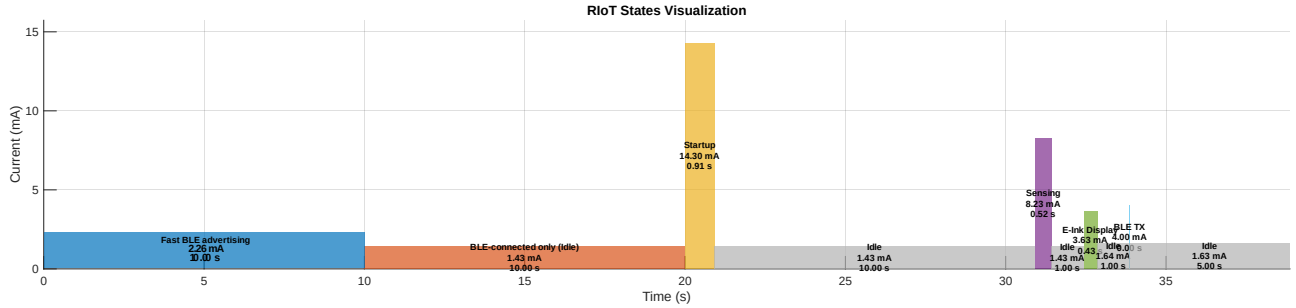
Finally, in Scenario 5, both communication modules are deactivated, i.e. BLE off and VLC off. This means that there will be no transmission of data to a mini lamp GW or AP, and the node performs only sensing and actuation tasks like E-ink display update. Thus, the operation duty cycle consists of the following states: Startup (909 ms)→ Idle for 30 ms → Sensing (149 ms)→ Idle for 250 ms → Optimized E-ink displaying (0.544 s) → Deep sleep (1 min or 1 hour). To optimize the current consumption, we consider the hardware optimization configuration, where cut points U6 and U9 are both cut on the Si-based node. Note that in Scenario 4 and 5, the wake-up IC module may be configured to wake up the node at regular intervals. An illustration of the various states in Scenario 4 and their instantaneous current consumption is shown in Figure 17(e).



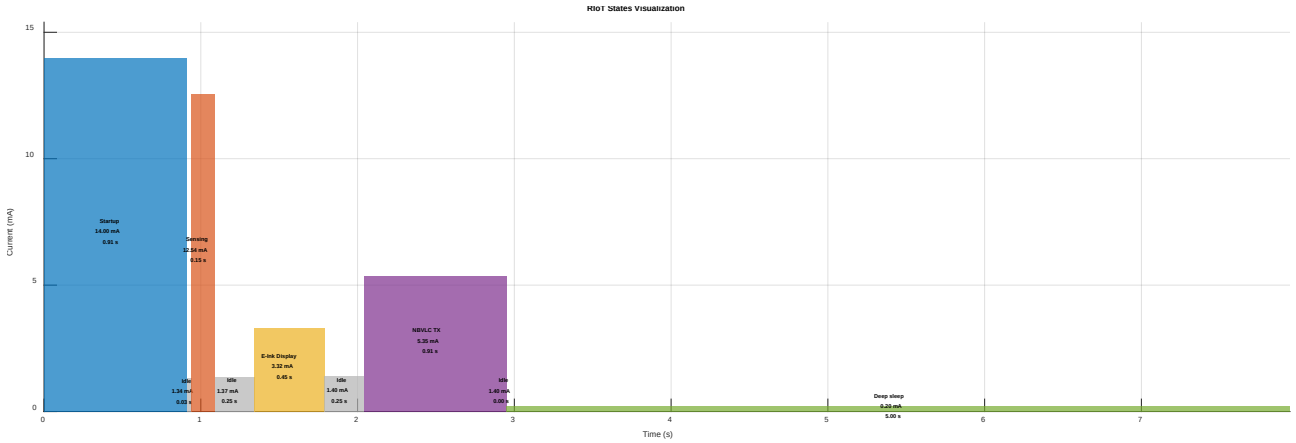

(a)



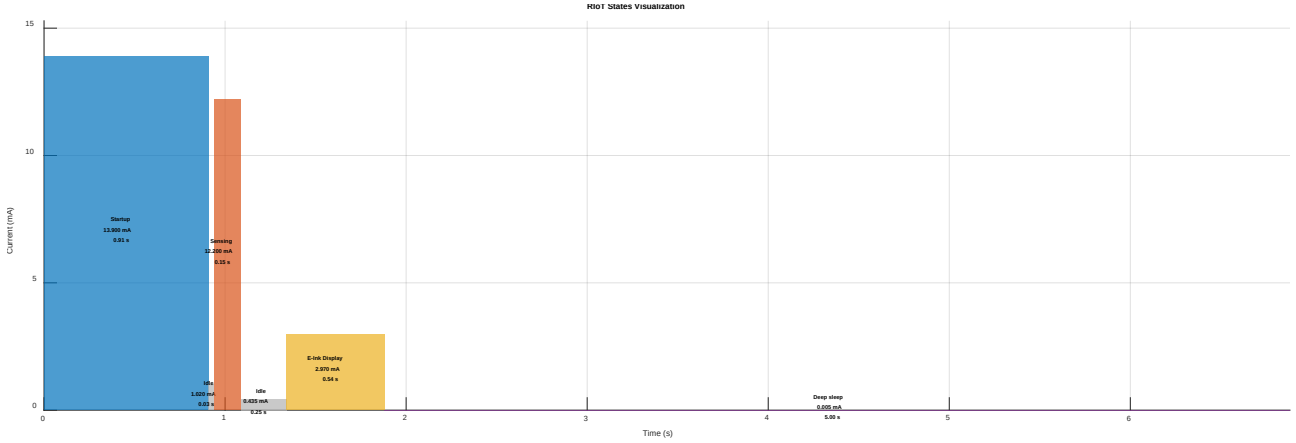

(b)




(c)




(d)




(e)

Figure 17. RIoT states visualization from energy consumption prediction App: (a) Scenario 1, (b) Scenario 2, (c) Scenario 3, and (d) Scenario 4 (for better illustration of states, the operation cycle was set to repeat after every 5 s in these illustrations).

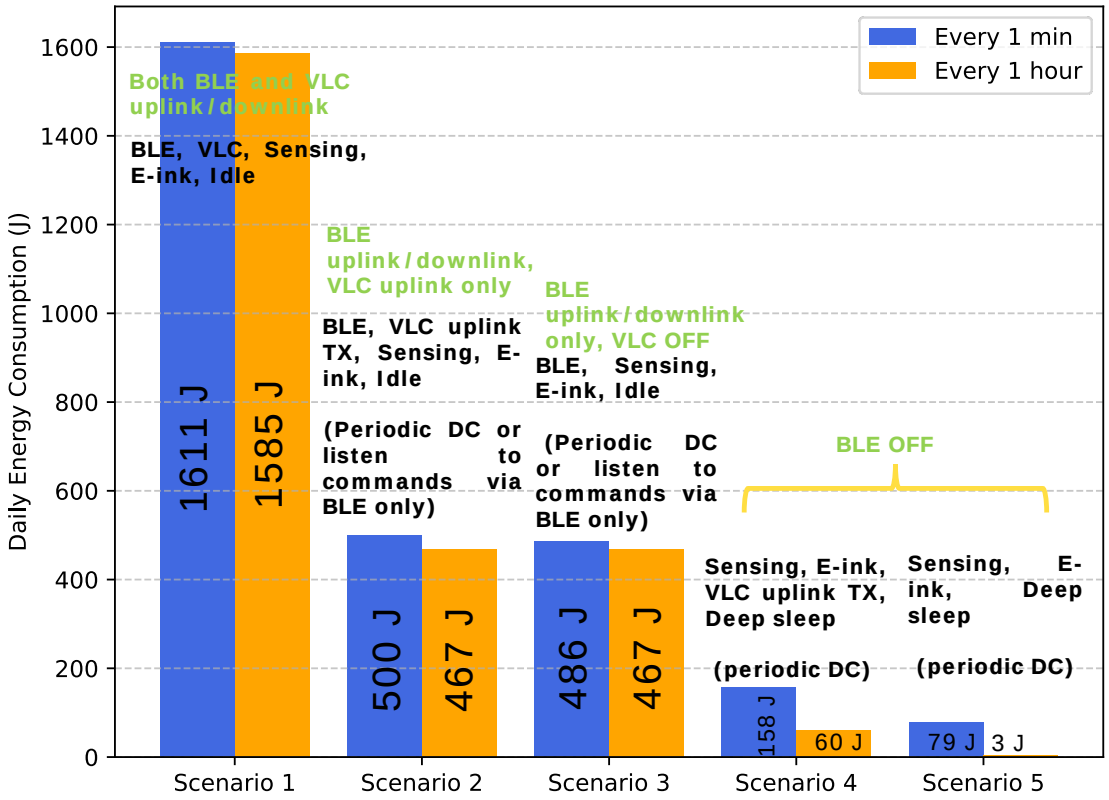



Figure 18. Summary of predicted energy consumption of Si-based nodes over a 24-hour period under various scenarios.

Figure 18 illustrates the expected energy consumption of the node across five different scenarios. It is evident that varying operational scenarios lead to different levels of energy usage. In Scenario 1, where all functionalities of the node are enabled and it continuously listens for commands from the mini lamp GW using both BLE and VLC, the highest energy consumption is observed. Specifically, considering a 24-hour period, the node consumes approximately 1611 J when receiving commands at 1-minute intervals and 1585 J when the interval is 1 hour. This elevated energy usage is primarily due to the timer functions responsible for Pulse Width Modulation (PWM) and encoding VLC data. These functions significantly increase the CPU workload, thereby raising energy consumption. In practical applications, if BLE is available and the node is expected to receive commands from the mini lamp GW and transmit data in the uplink, it would be more energy-efficient to disable the VLC functionality and rely solely on BLE for communication. This is because BLE offers higher throughput and lower energy consumption, as detailed in Section 3.2. Furthermore, transmitting and receiving commands or real data (sensor readings and location coordinates) via VLC can take almost 1 second (based on the frame structure defined in Section 3.5), leading to increased energy usage, as shown in Section 3.5.2.

In Scenario 2, the node performs the same operations as in Scenario 1 but does so using a periodic duty cycle, without waiting for commands from the mini lamp GW. Data transmission from the node to the mini lamp GW (uplink) is still carried out via VLC. However, since the node is not continuously listening for commands—particularly through VLC downlink—the timer function *nbvlc_off()* can be utilized. In this scenario, *nbvlc_on()* is only activated just before transmitting each chunk of the frame (*nbvlc_off()* is used for inter-chunk delays), allowing the node to conserve energy. As shown in Figure 18, this approach leads to a significant reduction in energy consumption, with values around 500 J for a 1-minute duty cycle and 467 J for a 1-hour duty cycle over a 24-hour period. This represents approximately a 70% reduction in energy usage compared to Scenario 1, despite performing the same operations.

In Scenario 3, the node operates similarly to Scenario 2, with the key difference being that data transmission from the node to the mini lamp GW (uplink) is performed using BLE instead of VLC. As a result, the VLC functionality remains disabled throughout the node's entire operating period. As shown in Figure 18, this leads to energy consumption of approximately 486 J for a 1-minute duty cycle and 467 J for a 1-hour duty cycle over a 24-hour period. This demonstrates that using BLE for uplink communication further optimizes energy efficiency compared to VLC.

In Scenario 4, the BLE functionality is disabled, and the node carries out the same operations as before—sensing, E-ink display update, and transmitting data to the mini lamp GW via VLCusing a periodic duty cycle. Unlike the previous scenarios, where the node remained in an idle state (i.e., maintaining a BLE-connected mode with the mini lamp GW), the node in Scenario 4 enters a deep sleep state after completing its active tasks for either 1 minute or 1 hour. During data transmission via VLC, the node broadcasts the data, allowing any listening mini lamp GW or AP to decode the frame and identify the source node. By disabling BLE, relying solely on VLC for communication, and allowing the node to enter deep sleep mode, Figure 18 shows a substantial reduction in energy consumption. Specifically, the node consumes approximately 158 J for a 1-minute duty cycle and 59 J for a 1-hour duty cycle over a 24-hour period.

In Scenario 5, all communication functionalities are disabled. The node only performs sensing and E-ink display update before entering a deep sleep state within a periodic duty cycle. As shown in Figure 18, this approach achieves the lowest energy consumption, with values around 78 J for a 1-minute duty cycle and just 3 J for a 1-hour duty cycle over a 24-hour period. This scenario highlights how disabling communication modules and leveraging deep sleep states can dramatically reduce energy usage, making them ideal for applications where minimal power consumption is crucial.

## 3.7 AI-based Node-Level Energy Prediction Model

Building on the measurement-driven modelling approach described in deliverable D2.4, we have generated a time-series dataset using our updated node-level energy consumption prediction application (Figure 19) to enable the implementation of an AI-based energy model. Originally developed in T2.4.1—with a validation accuracy exceeding 97%—this application has been enhanced to better support demonstration planning from an energy perspective.

The latest version, available in Tab 4 of the app, introduces the following key capabilities:

1. **Custom IoT node operation definition** – Users can define RIoT node behavior for any scenario, specifying operational states such as BLE TX/RX, BLE advertising and connected modes, VLC TX/RX, sensing, unoptimized and optimized E-ink display update, wake-up (or BME startup/initialization), idle, sleep, and more. Scenarios can be configured as periodic duty cycles or as randomly triggered events. Configuration parameters are easily adjustable, allowing quick analysis of their impact on energy consumption.
2. **Energy harvesting simulation** – Supports radio frequency (RF) and solar harvesting simulations with user-defined parameters, including supercapacitor charge/discharge dynamics, all linked to the defined node operations.
3. **Save/Load results** – Enables saving of prediction results (e.g., stepwise progress) and reloading them later, so analyses can be paused and resumed without loss of progress.
4. **Realistic RIoT dataset generation** – Produces realistic energy datasets (see sample data in Figure 20 and Figure 21) for the node based on the configured operations, with optional inclusion of simulated harvesting dynamics. These datasets can be used to train AI-based models for on-node energy prediction.

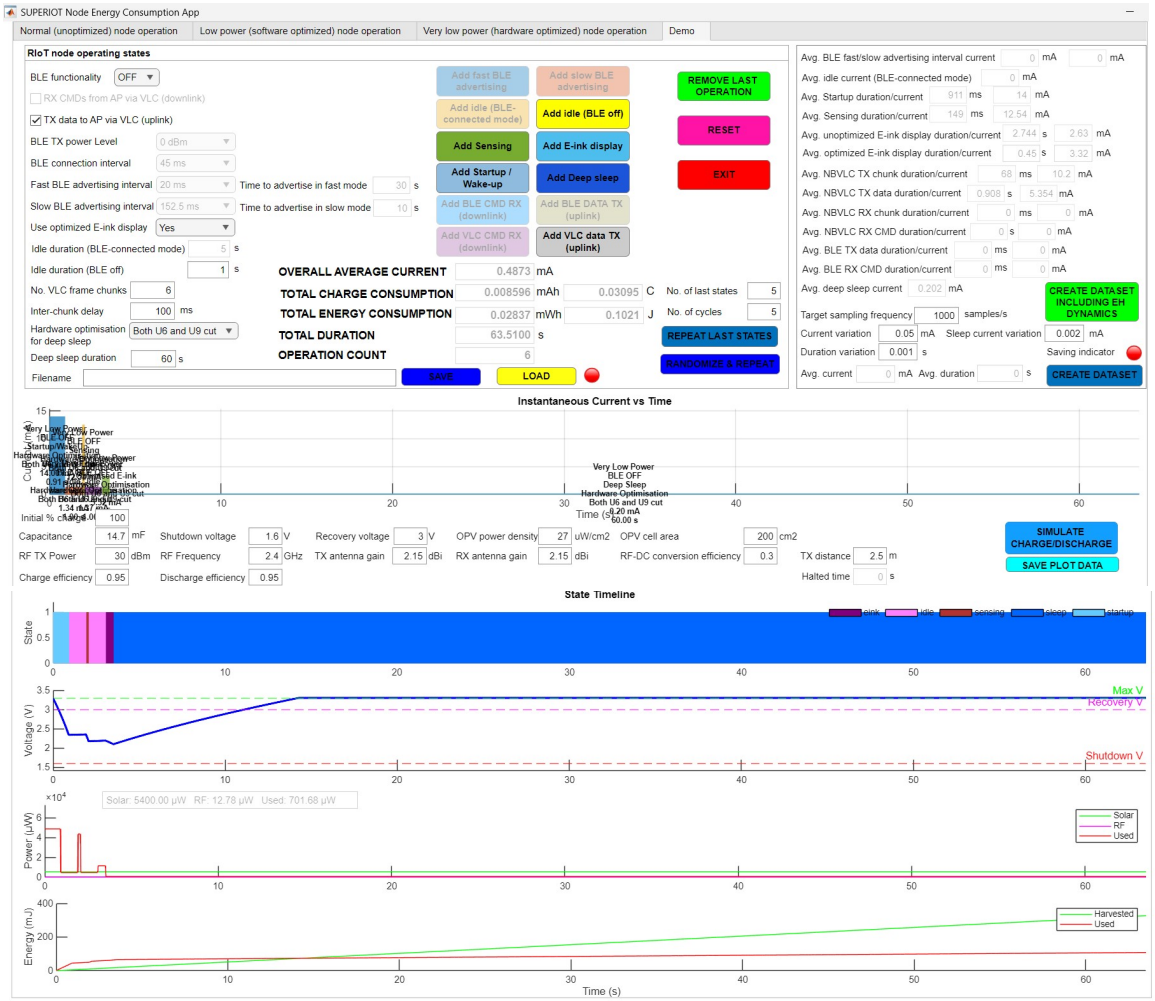



Figure 19. Updated node-level energy consumption prediction App.

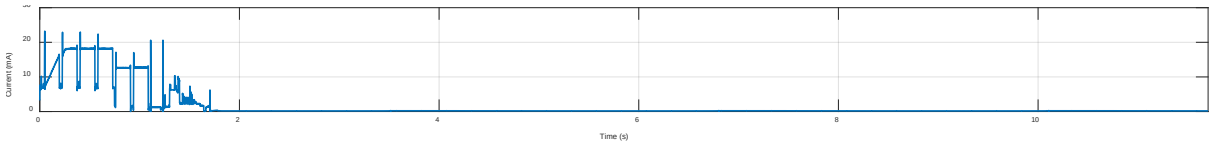
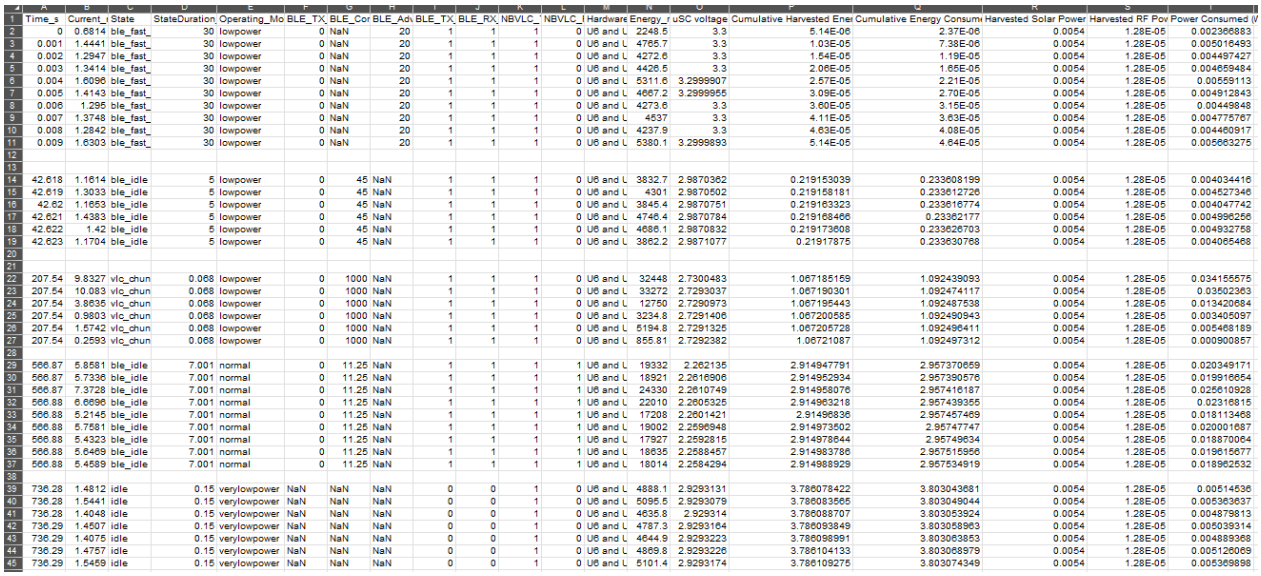



Figure 20. Example of a realistic current profile generated from node-level energy consumption prediction App.



| | A | B | C | D | E | F | G | H | I | J | K | L | M | N | O | P | Q | R | S | T |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | Time_s | Current_ | State | StateDuration | Operating_Mo | BLE_TX | BLE_Cor | BLE_Adv | BLE_TX | BLE_RX | NBVLC_ | NBVLC_I | Hardware | Energy_r | uSC voltage | Cumulative Harvested Ener | Cumulative Energy Consum | Harvested Solar Power | Harvested RF Pow | Power Consumed (W |
| 2 | 0 | 0.6814 | ble_fast_ | 30 | lowpower | 0 | NaN | 20 | 1 | 1 | 1 | 0 | U6 and L | 2248.5 | 3.3 | 5.14E-06 | 2.37E-06 | 0.0054 | 1.28E-05 | 0.002366883 |
| 3 | 0.001 | 1.4441 | ble_fast_ | 30 | lowpower | 0 | NaN | 20 | 1 | 1 | 1 | 0 | U6 and L | 4765.7 | 3.3 | 1.03E-05 | 7.38E-06 | 0.0054 | 1.28E-05 | 0.005016493 |
| 4 | 0.002 | 1.2947 | ble_fast_ | 30 | lowpower | 0 | NaN | 20 | 1 | 1 | 1 | 0 | U6 and L | 4272.6 | 3.3 | 1.54E-05 | 1.19E-05 | 0.0054 | 1.28E-05 | 0.004497427 |
| 5 | 0.003 | 1.3414 | ble_fast_ | 30 | lowpower | 0 | NaN | 20 | 1 | 1 | 1 | 0 | U6 and L | 4426.5 | 3.3 | 2.06E-05 | 1.65E-05 | 0.0054 | 1.28E-05 | 0.004659484 |
| 6 | 0.004 | 1.6096 | ble_fast_ | 30 | lowpower | 0 | NaN | 20 | 1 | 1 | 1 | 0 | U6 and L | 5311.6 | 3.2999907 | 2.57E-05 | 2.21E-05 | 0.0054 | 1.28E-05 | 0.00559113 |
| 7 | 0.005 | 1.4143 | ble_fast_ | 30 | lowpower | 0 | NaN | 20 | 1 | 1 | 1 | 0 | U6 and L | 4667.2 | 3.2999955 | 3.09E-05 | 2.70E-05 | 0.0054 | 1.28E-05 | 0.004912843 |
| 8 | 0.006 | 1.295 | ble_fast_ | 30 | lowpower | 0 | NaN | 20 | 1 | 1 | 1 | 0 | U6 and L | 4273.6 | 3.3 | 3.60E-05 | 3.15E-05 | 0.0054 | 1.28E-05 | 0.00449848 |
| 9 | 0.007 | 1.3748 | ble_fast_ | 30 | lowpower | 0 | NaN | 20 | 1 | 1 | 1 | 0 | U6 and L | 4537 | 3.3 | 4.11E-05 | 3.63E-05 | 0.0054 | 1.28E-05 | 0.004775767 |
| 10 | 0.008 | 1.2842 | ble_fast_ | 30 | lowpower | 0 | NaN | 20 | 1 | 1 | 1 | 0 | U6 and L | 4237.9 | 3.3 | 4.63E-05 | 4.08E-05 | 0.0054 | 1.28E-05 | 0.004460917 |
| 11 | 0.009 | 1.6303 | ble_fast_ | 30 | lowpower | 0 | NaN | 20 | 1 | 1 | 1 | 0 | U6 and L | 5380.1 | 3.2999893 | 5.14E-05 | 4.64E-05 | 0.0054 | 1.28E-05 | 0.005663275 |
| 12 | | | | | | | | | | | | | | | | | | | | |
| 13 | | | | | | | | | | | | | | | | | | | | |
| 14 | 42.618 | 1.1614 | ble_idle | 5 | lowpower | 0 | 45 | NaN | 1 | 1 | 1 | 0 | U6 and L | 3832.7 | 2.9870362 | 0.219153039 | 0.233608199 | 0.0054 | 1.28E-05 | 0.004034416 |
| 15 | 42.619 | 1.3033 | ble_idle | 5 | lowpower | 0 | 45 | NaN | 1 | 1 | 1 | 0 | U6 and L | 4301 | 2.9870502 | 0.219158181 | 0.233612726 | 0.0054 | 1.28E-05 | 0.004527346 |
| 16 | 42.62 | 1.1663 | ble_idle | 5 | lowpower | 0 | 45 | NaN | 1 | 1 | 1 | 0 | U6 and L | 3845.4 | 2.9870751 | 0.219163323 | 0.233616774 | 0.0054 | 1.28E-05 | 0.004047742 |
| 17 | 42.621 | 1.4383 | ble_idle | 5 | lowpower | 0 | 45 | NaN | 1 | 1 | 1 | 0 | U6 and L | 4746.4 | 2.9870784 | 0.219168466 | 0.23362177 | 0.0054 | 1.28E-05 | 0.004996256 |
| 18 | 42.622 | 1.42 | ble_idle | 5 | lowpower | 0 | 45 | NaN | 1 | 1 | 1 | 0 | U6 and L | 4686.1 | 2.9870832 | 0.219173608 | 0.233626703 | 0.0054 | 1.28E-05 | 0.004932758 |
| 19 | 42.623 | 1.1704 | ble_idle | 5 | lowpower | 0 | 45 | NaN | 1 | 1 | 1 | 0 | U6 and L | 3862.2 | 2.9871077 | 0.21917875 | 0.233630768 | 0.0054 | 1.28E-05 | 0.004065468 |
| 20 | | | | | | | | | | | | | | | | | | | | |
| 21 | | | | | | | | | | | | | | | | | | | | |
| 22 | 207.54 | 9.8327 | vlc_chun | 0.068 | lowpower | 0 | 1000 | NaN | 1 | 1 | 1 | 0 | U6 and L | 32448 | 2.7300493 | 1.067185159 | 1.092439093 | 0.0054 | 1.28E-05 | 0.034155575 |
| 23 | 207.54 | 10.083 | vlc_chun | 0.068 | lowpower | 0 | 1000 | NaN | 1 | 1 | 1 | 0 | U6 and L | 33272 | 2.7293037 | 1.067190301 | 1.092474117 | 0.0054 | 1.28E-05 | 0.03502363 |
| 24 | 207.54 | 3.8635 | vlc_chun | 0.068 | lowpower | 0 | 1000 | NaN | 1 | 1 | 1 | 0 | U6 and L | 12750 | 2.7290973 | 1.067195443 | 1.092487538 | 0.0054 | 1.28E-05 | 0.013420684 |
| 25 | 207.54 | 0.9803 | vlc_chun | 0.068 | lowpower | 0 | 1000 | NaN | 1 | 1 | 1 | 0 | U6 and L | 3234.8 | 2.7291406 | 1.067200585 | 1.092490943 | 0.0054 | 1.28E-05 | 0.003405097 |
| 26 | 207.54 | 1.5742 | vlc_chun | 0.068 | lowpower | 0 | 1000 | NaN | 1 | 1 | 1 | 0 | U6 and L | 5194.8 | 2.7291325 | 1.067205728 | 1.092496411 | 0.0054 | 1.28E-05 | 0.005468189 |
| 27 | 207.54 | 0.2593 | vlc_chun | 0.068 | lowpower | 0 | 1000 | NaN | 1 | 1 | 1 | 0 | U6 and L | 855.81 | 2.7292382 | 1.06721087 | 1.092497312 | 0.0054 | 1.28E-05 | 0.000900857 |
| 28 | | | | | | | | | | | | | | | | | | | | |
| 29 | 566.87 | 5.8581 | ble_idle | 7.001 | normal | 0 | 11.25 | NaN | 1 | 1 | 1 | 1 | U6 and L | 19332 | 2.262135 | 2.914947791 | 2.957370659 | 0.0054 | 1.28E-05 | 0.020349171 |
| 30 | 566.87 | 5.7336 | ble_idle | 7.001 | normal | 0 | 11.25 | NaN | 1 | 1 | 1 | 1 | U6 and L | 18921 | 2.2616906 | 2.914952934 | 2.957390576 | 0.0054 | 1.28E-05 | 0.019916654 |
| 31 | 566.87 | 7.3728 | ble_idle | 7.001 | normal | 0 | 11.25 | NaN | 1 | 1 | 1 | 1 | U6 and L | 24330 | 2.2610749 | 2.914958076 | 2.957416187 | 0.0054 | 1.28E-05 | 0.025610928 |
| 32 | 566.88 | 6.6696 | ble_idle | 7.001 | normal | 0 | 11.25 | NaN | 1 | 1 | 1 | 1 | U6 and L | 22010 | 2.2605325 | 2.914963218 | 2.957439355 | 0.0054 | 1.28E-05 | 0.023168815 |
| 33 | 566.88 | 5.2145 | ble_idle | 7.001 | normal | 0 | 11.25 | NaN | 1 | 1 | 1 | 1 | U6 and L | 17208 | 2.2601421 | 2.91498836 | 2.957457469 | 0.0054 | 1.28E-05 | 0.018113468 |
| 34 | 566.88 | 5.7581 | ble_idle | 7.001 | normal | 0 | 11.25 | NaN | 1 | 1 | 1 | 1 | U6 and L | 19002 | 2.2596948 | 2.914973502 | 2.95747747 | 0.0054 | 1.28E-05 | 0.020001687 |
| 35 | 566.88 | 5.4323 | ble_idle | 7.001 | normal | 0 | 11.25 | NaN | 1 | 1 | 1 | 1 | U6 and L | 17927 | 2.2592815 | 2.914978644 | 2.95749634 | 0.0054 | 1.28E-05 | 0.018870064 |
| 36 | 566.88 | 5.6469 | ble_idle | 7.001 | normal | 0 | 11.25 | NaN | 1 | 1 | 1 | 1 | U6 and L | 18635 | 2.2588457 | 2.914983786 | 2.957515956 | 0.0054 | 1.28E-05 | 0.019615677 |
| 37 | 566.88 | 5.4589 | ble_idle | 7.001 | normal | 0 | 11.25 | NaN | 1 | 1 | 1 | 1 | U6 and L | 18014 | 2.2584294 | 2.914988929 | 2.957534919 | 0.0054 | 1.28E-05 | 0.018962532 |
| 38 | | | | | | | | | | | | | | | | | | | | |
| 39 | 736.28 | 1.4812 | idle | 0.15 | verylowpower | NaN | NaN | NaN | 0 | 0 | 1 | 0 | U6 and L | 4898.1 | 2.9293131 | 3.786078422 | 3.803043681 | 0.0054 | 1.28E-05 | 0.00514636 |
| 40 | 736.28 | 1.5441 | idle | 0.15 | verylowpower | NaN | NaN | NaN | 0 | 0 | 1 | 0 | U6 and L | 5095.5 | 2.9293079 | 3.786083665 | 3.803049044 | 0.0054 | 1.28E-05 | 0.005363637 |
| 41 | 736.28 | 1.4048 | idle | 0.15 | verylowpower | NaN | NaN | NaN | 0 | 0 | 1 | 0 | U6 and L | 4635.8 | 2.929314 | 3.786088707 | 3.803053924 | 0.0054 | 1.28E-05 | 0.004879813 |
| 42 | 736.29 | 1.4507 | idle | 0.15 | verylowpower | NaN | NaN | NaN | 0 | 0 | 1 | 0 | U6 and L | 4787.3 | 2.9293164 | 3.786093849 | 3.803058963 | 0.0054 | 1.28E-05 | 0.005039314 |
| 43 | 736.29 | 1.4075 | idle | 0.15 | verylowpower | NaN | NaN | NaN | 0 | 0 | 1 | 0 | U6 and L | 4644.9 | 2.9293223 | 3.786098991 | 3.803063853 | 0.0054 | 1.28E-05 | 0.004889368 |
| 44 | 736.29 | 1.4757 | idle | 0.15 | verylowpower | NaN | NaN | NaN | 0 | 0 | 1 | 0 | U6 and L | 4869.8 | 2.9293226 | 3.786104133 | 3.803068979 | 0.0054 | 1.28E-05 | 0.005126069 |
| 45 | 736.29 | 1.5459 | idle | 0.15 | verylowpower | NaN | NaN | NaN | 0 | 0 | 1 | 0 | U6 and L | 5101.4 | 2.9293174 | 3.786109275 | 3.803074349 | 0.0054 | 1.28E-05 | 0.005369898 |

Figure 21. Sample dataset generated from node-level energy consumption prediction App.

We use the App in Figure 19 to generate a time-series dataset for the node over 7 hours, operating across a wide range of functional states, modes, and configurations (in a randomized fashion). The dataset provides high-resolution temporal information with a sampling rate of 1000 samples per second (i.e., millisecond resolution), where each entry represents a discrete snapshot of the node's behavior at a given moment. Each record includes the instantaneous current consumption (in milliamperes), the operational state of the RIoT node—such as wake-up, sleep, BLE advertising, BLE-connected, E-ink display updates, VLC TX/RX, BLE TX/RX, sensing, and idle—along with the duration of each state. Additional parameters include operating mode (e.g., normal, low-power or very low-power), BLE-related settings such as transmit power, advertisement interval, and CI (when applicable), binary flags indicating the activity status of the BLE and VLC TX/RX functionalities, the active hardware optimization configuration, and the computed energy consumed at each timestep.

Collectively, these features capture the complex interplay between energy consumption and system functionality under realistic workloads and configuration variations. This dataset is specifically designed to facilitate the training and evaluation of deep learning models for energy prediction and to support the development of intelligent, context-aware power management strategies in embedded and IoT systems.

To forecast the instantaneous current consumption (the target variable), we utilize the *TimeSeriesDataSet* module from the PyTorch Forecasting library and train a Temporal Fusion Transformer Model (TFTM) on the collected data, treated as a multivariate time series. The dataset is partitioned into three subsets: 70% for training, 15% for validation, and 15% for testing. The time-varying known categorical variables include the operational state of the node and the hardware optimization configuration. Time-varying known real-valued features comprise the timestamp (in milliseconds), BLE-related parameters such as transmit power, advertising interval, and connection interval, along with their corresponding binary flags to indicate applicability (e.g., in very-low-power mode, BLE is disabled, and these fields are inactive). Additional real-valued features include the state duration and binary flags representing the activity status of VLC transmission and reception. The only time-varying unknown real — and the prediction target — is the instantaneous current consumption in milliamperes. This setup enables the TFTM to learn temporal dependencies and contextual relationships across multiple dynamic features for accurate current prediction under varying system conditions. The sequence length is set at 50 empirically, translating to 50 ms. The TFTM is trained for 50 epochs using a learning rate of 0.03, a hidden size of 16, an attention head size of 4, a hidden continuous size of 8, and a dropout rate of 0.1. The model is optimized using the Quantile loss function.

We evaluate the performance of the TFTM in predicting instantaneous current consumption across multiple forecasting horizons, specifically 1 ms, 10 ms, and 50 ms. The results are

compared against the default Baseline model from the PyTorch Forecasting library, which uses the last observed target value as prediction. Sample traces of the power consumption predicted by the TFTM are shown in Figure 22, considering a prediction length of 1 ms. It can be observed from the traces that the predictions are accurate across various states, modes of operations, and configurations. The comparison results between the TFTM and the Baseline model are also shown in Table 6. The TFTM consistently outperforms the Baseline model in forecasting current consumption (in mA) across all evaluated prediction lengths. Notably, the TFTM achieves significantly lower MAPE, with values of 3.85%, 11.9%, and 18.9% for prediction lengths (PL) of 1, 10, and 50 respectively, compared to the Baseline's 15.5%, 20.5%, and 27.4%. This represents a fourfold improvement in short-term predictions and a substantial gain even over longer horizons. Similarly, the RMSE is markedly reduced with TFTM, indicating superior pointwise accuracy. At PL=50, the RMSE drops from 2.02 (Baseline) to 1.47 with the TFTM. Moreover, the coefficient of determination ($R^2$) is consistently higher for the TFTM, reaching 0.986 at PL=1 versus 0.902 for the Baseline, and maintaining a notable advantage at PL=50 (0.815 vs. 0.654). These results highlight TFTM's enhanced capability to model temporal dependencies and variability in current consumption, making it a more effective and reliable choice for energy forecasting in resource-constrained IoT systems.

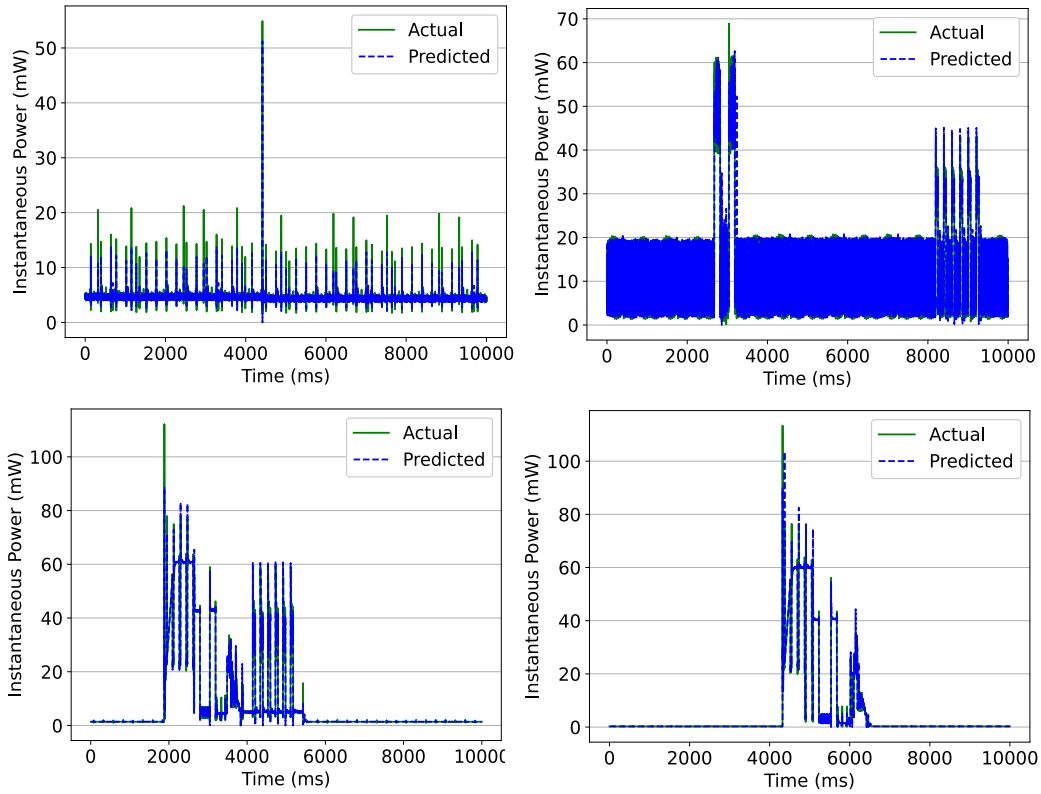


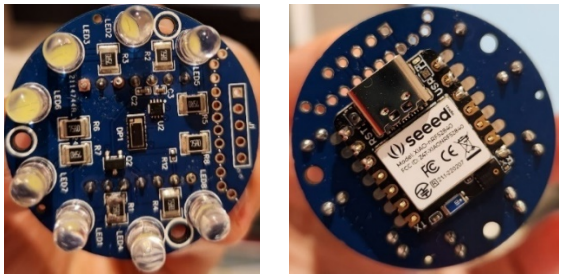

Figure 22. Comparison of measured power consumption traces (green) from the test set and one-step-ahead predictions (blue) generated by the TFTM. The model uses a 50-step historical window as input to forecast future current consumption.

Table 6. Performance comparison of models at different prediction lengths (PL) with a fixed training length (TL=50).

| Model | TL=50, PL=1 | | | TL=50, PL=10 | | | TL=50, PL=50 | | |
|---|---|---|---|---|---|---|---|---|---|
| | MAPE (%) | RMSE | $R^2$ | MAPE (%) | RMSE | $R^2$ | MAPE (%) | RMSE | $R^2$ |
| TFT | 3.85 | 0.4 | 0.986 | 11.9 | 1.06 | 0.905 | 18.9 | 1.47 | 0.815 |
| Baseline | 15.5 | 1.07 | 0.902 | 20.5 | 1.82 | 0.72 | 27.4 | 2.02 | 0.654 |

## 3.8 Mini-Lamp Gateway Energy Consumption and Parameter Optimizations

The mini-lamp GW (standalone without BBB platform), initially designed in the SUPERIOT project (see Figure 23), communicates with the SUPERIOT node, facilitating the exchange of data and commands via both VLC and BLE. In the following sections, we performed measurements on the mini lamp GW which runs a hybrid firmware that supports both VLC and BLE communication methods.



(a) (b)

Figure 23. Mini-lamp GW providing BLE and VLC functionalities: (a) Front (b) Back.

### 3.8.1 Impact of Different VLC PWM Duty Cycles on Mini-Lamp Gateway's Current Consumption (Lamp LEDs ON, BLE OFF)

In this experiment, we investigate the average current consumption of the mini-lamp GW when it is configured for VLC only (while BLE functionality remains off). The current measurement setup is shown in Figure 24. The programmable DC power supply was used to power the mini lamp GW with a 5 V supply voltage. Note that a SUPERIOT node was positioned at a distance of 5 m from the mini lamp GW with clear LoS.

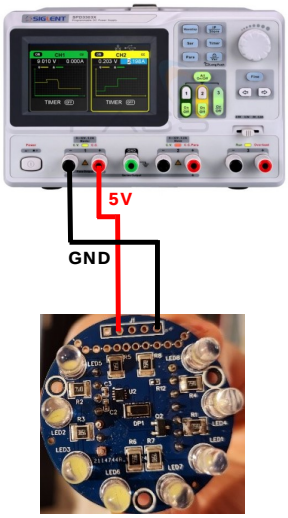



Figure 24. Current measurement setup for mini-lamp GW.

Figure 25 shows the average current measured when powering the mini lamp GW with 5 V, for different VLC PWM duty cycle values (which vary the brightness of the 8 LEDs and thus the overall current consumption). When the brightness level was set to either 0% or 100%, the VLC LEDs were off, and no data was received on the node. The average current consumption of the mini-lamp GW when the LEDs are off is around ~12 mA. When the brightness level was set to 98% and 99%, the maximum average current value of ~238 mA was recorded on the mini-lamp GW. It is to be noted that the node did not capture any data when the VLC PWM duty cycle was set to 99%. Therefore, at VLC PWM duty cycle values of 0%, 99%, and 100%, effective modulation for VLC data transmission does not occur. As for the remaining PWM duty cycle

values (1% to 98%), the LEDs were on, and the node could receive the data transmitted by the mini-lamp GW.

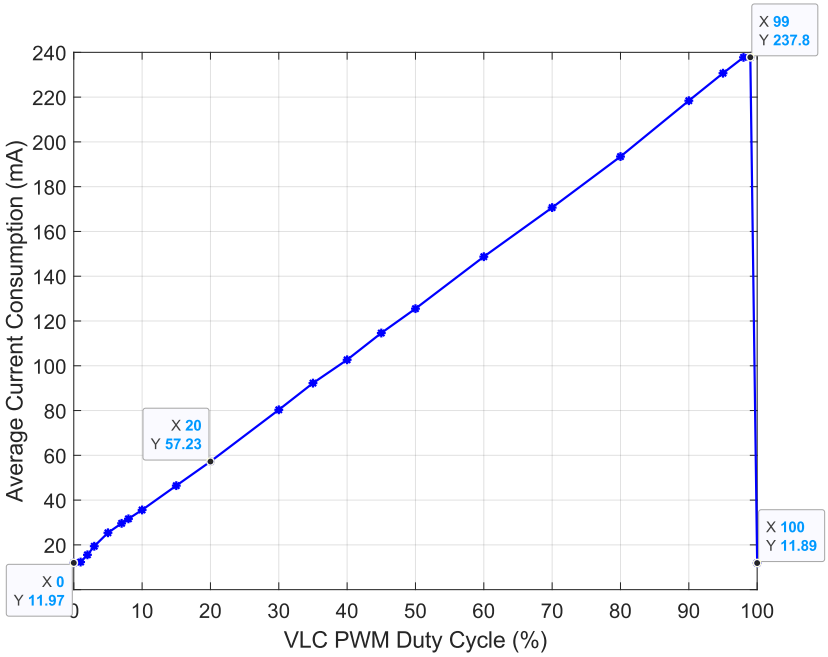



Figure 25. Average current consumption of mini-lamp GW for different VLC PWM duty cycles values (BLE off).

From Figure 25, we can derive a 3rd order polynomial (cubic) equation for the average current consumption of the mini lamp GW as a function of the VLC PWM duty cycle as follows:

$$I_{\text{minilamp\_vlc\_idle\_only}}\,(\text{mA}) = 2.23 \times 10^{-5}x^3 - 2.8 \times 10^{-3}x^2 + 2.36x + 11.97 \qquad (3.3)$$

where $x$ represents the VLC PWM duty cycle, $0 \leq x \leq 98\%$. This model achieves a RMSE, $R^2$ value and MAPE of 0.83 mA, 0.9999, and 2.19%, respectively.

### 3.8.2 Node's Current Consumption Under Varying VLC PWM Duty Cycles of Mini-Lamp GW

In another experiment, we vary the PWM duty cycle of the VLC transmission, as previously carried out in Section 3.8.1, and measure the average current consumption across the U9 (VLC RX module) measuring point on the node. The latter is connected via BLE to the mini lamp GW and only the VLC functionality is activated (no sensing or E-ink display update) on the node. Note that VLC data is also being transmitted and received by the node's own VLC TX and RX, in addition to the VLC data received from the mini-lamp GW. The node's VLC transceiver transmits and receives data every ~5 seconds, while the mini-lamp GW transmits data to the node every ~1 second. Figure 26 shows the current profiles of the U9 measuring point on the node when the mini-lamp GW is transmitting VLC data with different PWM duty cycles (i.e. different LED brightness levels). Note that the node was positioned at approximately 3 m from the mini-lamp GW for all the VLC PWM duty cycle tests. In Figure 26(a) and (b), the small current peaks correspond to the VLC data reception from the mini-lamp GW (every ~1 s), while the bigger current peaks correspond to the VLC data reception from the node's own VLC RX (every ~5 s). Figure 26(a) and (b) refer to the case when the mini-lamp GW was transmitting VLC data with a PWM duty cycle of 10% and 20%, respectively. Since the node's VLC TX and RX are very close to each other, the current peaks are very high. When the mini-lamp GW transmits VLC data with a higher PWM duty cycle of 40% (higher LED brightness), the peaks get bigger, as illustrated in Figure 26(c). However, the average current consumption recorded across U9 on the node remained essentially very similar when the mini-lamp GW transmits with different VLC PWM duty cycle values (even as high as 98%), with a value around ~334 uA (idle current between VLC

transmissions is around 330 μA). Likewise, the average current consumption measured across the whole node remained essentially unchanged across different VLC PWM duty cycle values.

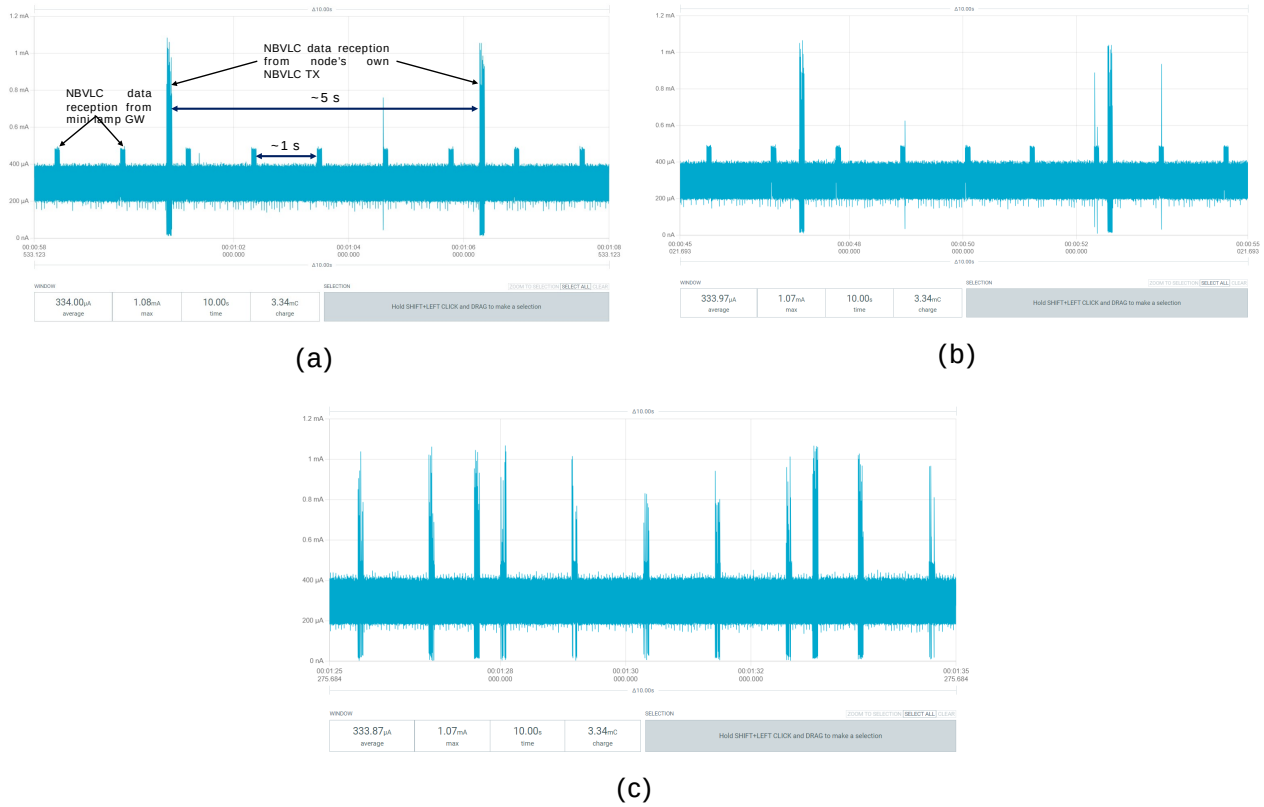



Figure 26. Current profile across U9 measuring point on node during VLC data reception when mini-lamp GW transmits VLC data with a PWM duty cycle of (a) 10%, (b) 20%, and (c) 40%.

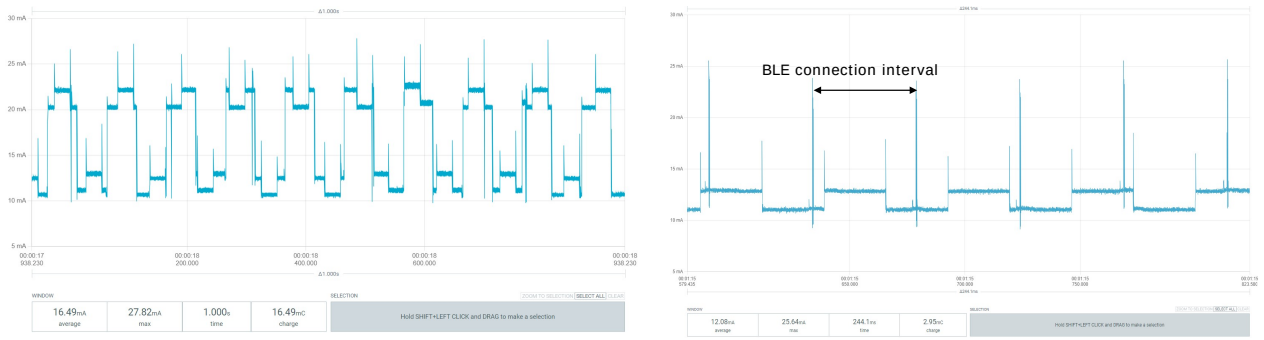
### 3.8.3 Current Consumption of Mini-Lamp GW During BLE Scanning and Connection (Lamp LEDs OFF, BLE ON)

This experiment examines the average current consumption of the mini-lamp GW when the VLC lamp LEDs are off and only BLE functionality is active. First, we analyze how the BLE scanning duty cycle affects the average current consumption of the mini lamp GW. An example of the current profile during BLE scanning is shown in Figure 27.(a) (lamp LEDs off). The results in Figure 28.(a) show that as the BLE scanning duty cycle increases, the average current consumption of the mini lamp GW increases linearly, with a 75% increase observed from a duty cycle value of 2.5% to 100% (i.e. continuous scanning). Based on the results in Figure 28.(a), we can derive a linear model for predicting the overall average current consumption of the mini lamp GW when it is in BLE scanning mode (VLC off) as follows:

$$I_{\text{minilamp\_ble\_scan\_only}}\ (mA) = 9.30 \times Duty\ Cycle + 11.80 \qquad (3.4)$$

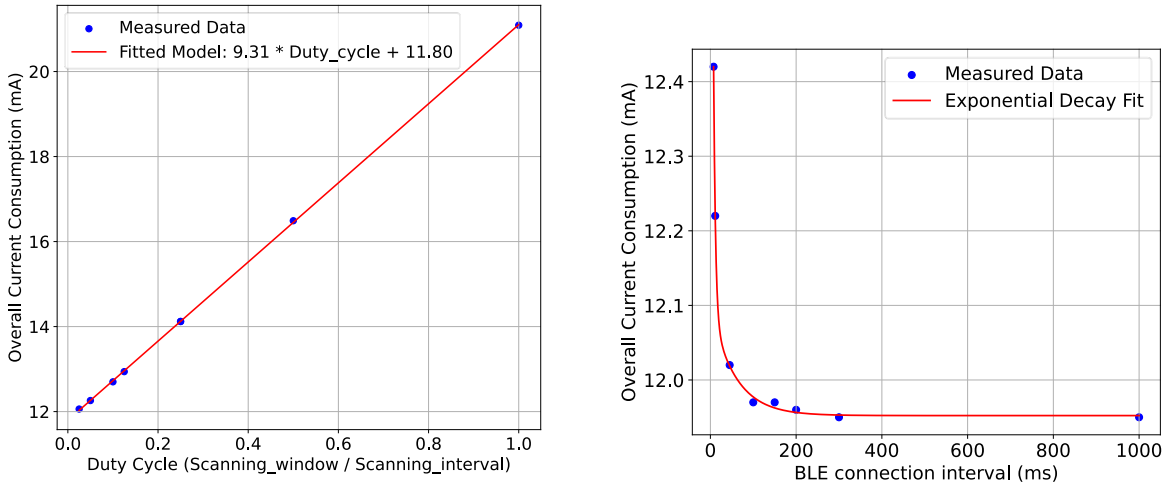
where $Duty\ Cycle = \frac{\text{BLE scanning window (ms)}}{\text{BLE scanning interval (ms)}}$.

The linear model in (3.4) closely fits the measured data in Figure 28.(a). It achieves an RMSE, $R^2$ value, and MAPE of 0.0224 mA, 1, and 0.128%, respectively. When only BLE is active on the mini-lamp GW (LEDs off), the current consumption is significantly lower. Therefore, if VLC is not required, turning off the LEDs on the mini-lamp GW will improve energy efficiency across the network.




(a) (b)

Figure 27. Current profile of mini-lamp GW during: (a) BLE scanning (100,50) ms, (b) BLE-connected mode with a CI of 45 ms (lamp LEDs off in both cases).




(a) (b)

Figure 28. Current consumption analysis of mini lamp GW during: (a) different BLE scanning duty cycle values, (b) different BLE connection intervals.

Next, we analyze the average current consumption when the mini-lamp GW is connected via BLE to a node. An example of the current profile when the mini lamp GW is in BLE-connected mode with a BLE CI of 45 ms is shown in Figure 27.(b). From Figure 28.(b), it can be observed that as the BLE CI increases from 11.25 ms to 1000 ms, the average current consumption of the mini lamp GW decreases slightly from 12.22 mA to 11.95 mA (considering no BLE data transfer), which can be almost constant. Nonetheless, we can derive an exponential decay model for estimating the average current consumption of the mini lamp GW when it is in BLE connected-only mode (lamp off) as follows:

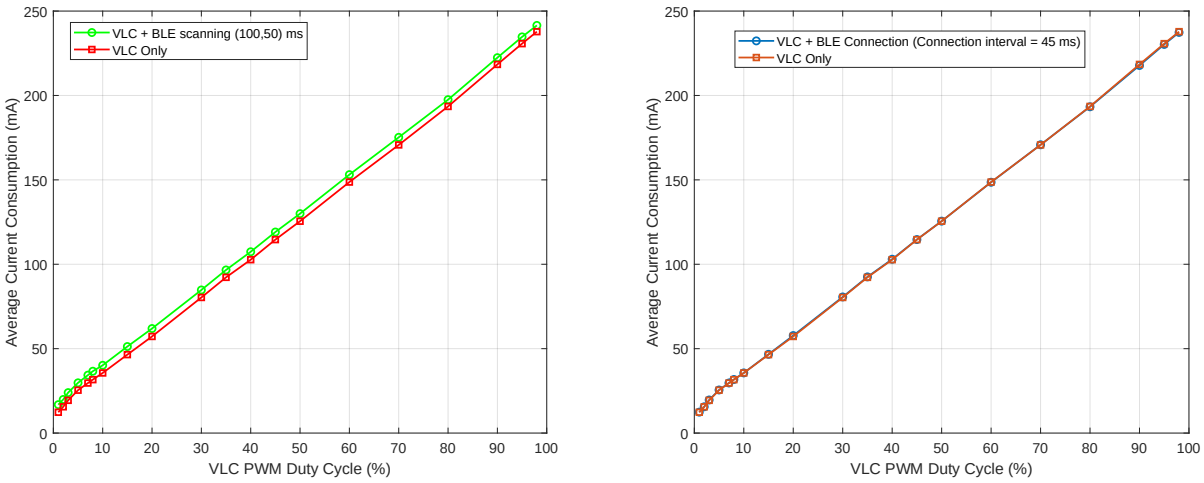
$$I_{\text{minilamp\_ble\_conn\_idle\_only}}(mA) = 1.798e^{-0.2226z} + 0.14757e^{-0.017775z} + 11.95 \quad (3.5)$$

where $z$ is the BLE CI (ms) in the range 7.5 ms ≤ $z$ ≤ 4000 ms. The model in (3.5) achieves an RMSE, $R^2$ value and MAPE of 0.0047 mA, 0.9991, and 0.0304%, respectively.

### 3.8.4 Impact of Different VLC PWM Duty Cycles on Average Current Consumption of Mini-Lamp Gateway (Lamp LEDs ON and BLE ON)

In this experiment, the VLC PWM duty cycle varied between 1% and 98%, while BLE scanning is also active, and the average current consumption of the mini-lamp GW is measured. BLE

scanning is initiated with a scan window of 50 ms, a scan interval of 100 ms, and a TX power level of 0 dBm.




(a) (b)

Figure 29. Average current consumption of mini-lamp GW for different VLC PWM duty cycles values: (a) VLC + BLE scanning mode (100,50) ms, and (b) VLC + BLE connected mode (CI=45 ms).

Figure 29(a) compares the average current consumption of the mini-lamp GW under two scenarios: (1) VLC only and (2) simultaneous VLC and BLE scanning. When both VLC and BLE scanning are active, we notice an almost parallel upward shift in current consumption, with an average offset of approximately 4.43 mA, compared to the average current consumption for VLC only. Thus, equation (3.3) can be modified as

$$I_{\text{minilamp\_vlc\_idle\_and\_ble\_scan}}\,(\text{mA}) = 2.23 \times 10^{-5}x^3 - 2.8 \times 10^{-3}x^2 + 2.36x + 11.97 + C \quad (3.6)$$

where $x$ represents the VLC PWM duty cycle, $0 \leq x \leq 98\%$, and $C$ represents the average offset.

Considering the fact that the average current consumption increases linearly with the BLE scanning duty cycle as per Figure 28.(a), using equation (3.4), we can rewrite equation (3.6) for different BLE scanning duty cycles as follows:

$$I_{\text{minilamp\_vlc\_idle\_and\_ble\_scan}}\,(\text{mA}) = 2.23 \times 10^{-5}x^3 - 2.8 \times 10^{-3}x^2 + 2.36x + 11.80 + (9.30 \times Duty\ Cycle) \quad (3.7)$$

where $Duty\ Cycle = \frac{\text{BLE scanning window (ms)}}{\text{BLE scanning interval (ms)}}$, and $x$ represents the VLC PWM duty cycle, $0 \leq x \leq 98\%$. This model achieves an RMSE, $R^2$ value, and MAPE of 0.89 mA, 0.9999, and 1.923%, respectively.

On the other hand, as shown in Figure 29(b), when the VLC lamp is ON (but idle with no VLC data TX and RX) and the mini lamp GW is connected to a node via BLE with a CI of 45 ms (idle with no data transfer), the difference in current consumption is negligible when compared to the average current consumption for VLC mode only. Hence, it can be deduced from Figure 29(a) and Figure 28.(a) that BLE scanning consumes more energy than the BLE-connected idle mode. With higher BLE scanning duty cycle values, we expect that the difference in current consumption observed in Figure 29(a) will be even higher, in line with Figure 28.(a). Thus, if we apply the model in equation (3.3) for VLC + BLE-connected mode (no data transfer), we obtain an RMSE, $R^2$ value and MAPE of 0.82 mA, 0.9999, and 2.25%, respectively.

These findings suggest the potential for implementing a dynamic optimization mechanism within the mini-lamp GW. Such a system could automatically adjust parameters like the VLC PWM duty

cycle based on whether the GW is operating with VLC alone or in combination with BLE scanning or connected mode. By selecting the most energy-efficient configuration for each scenario, the mini-lamp GW could achieve significant energy savings while maintaining reliable communication.

### 3.8.5 Current Consumption of Mini-Lamp GW During BLE TX and RX (Lamp LEDs OFF)

When the mini-lamp GW either sends a command to the node by writing to its BLE characteristics or receive sensor/location data from the node via BLE UART, there is no noticeable change in current consumption due to the very small size of the commands and data. This remains consistent across different BLE power levels (0 dBm, 4 dBm, and 8 dBm), similar to the behavior observed for the node in Section 3.3.

## 3.9 Beagle-Bone Black (BBB) Access Point (AP) Energy Consumption Analysis, Modeling and Prediction

The main AP developed as part of the SUPERIOT project consists of a BBB platform with the custom-engineered mini-lamp GW (see Figure 23) integrated on a BBB Cape, as shown in Figure 30. The BBB AP shown in Figure 30 communicates using BLE and VLC protocols with the SUPERIOT nodes. The BBB platform is a low-power consumption, single board microcomputer featuring 512 MB RAM, an AM335x 1 GHz ARM Cortex-A8 processor and provides connectivity via mini-HDMI, Ethernet, and USB 2.0. The BBB APs in the network will connect to master node (developed on a Raspberry Pi platform) via an Ethernet switch. The master node is expected to perform network and communication management, as well as AP registrations, and it consists of two MQTT brokers, master-to-cloud and AP-to-master. A router will also be connected to an Ethernet switch so that the information can be relayed to a cloud broker via the internet. The recommended input supply voltage is 5V/2A (10 W).

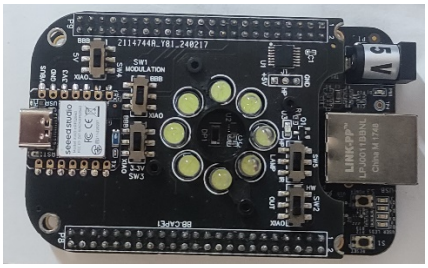


Figure 30. SUPERIOT BBB Access Point (BBB platform + mini-lamp GW mounted on a Cape).

### 3.9.1 Average Current Consumption of BBB AP (Lamp LEDs OFF and BLE OFF)

Table 7 and Figure 31 show the average current consumption of the BBB AP under various configurations. For test IDs 1 to 5, the VLC lamp and BLE functionality are both OFF. During the initial boot sequence (test ID 1), the BBB AP consumes approximately 405 mA (2.03 W) on average. In this configuration, a USB connection links the BBB's USB 2.0 port to the Type-C interface of the Seeed Xiao nRF52840 module on the lamp CAPE, while an Ethernet cable connects the BBB AP to a network switch. The boot process takes around 72 seconds to complete, after which the BBB platform enters the idle state. The idle current obtained in test ID 3 when both USB and Ethernet cables are connected on the AP is around 255 mA (1.275 W). We also record the current consumption during the idle state when only the Ethernet cable is connected (test ID 4), and also when no USB and no Ethernet cables are connected (test ID 2). It can be deduced from Table 7 that the connection of a USB cable between the BBB platform and the USB Type-C on the Seeed Xiao NRF52840 increases the current consumption by approximately 14 mA, while the connection of an Ethernet cable between the BBB platform and an Ethernet switch increases the current consumption by approximately 71 mA (no data transfer via Ethernet). Comparing test ID2 with test ID7 and test ID4 with test ID8, we observe that the lamp CAPE increases overall current consumption of the BBB platform by approximately 12 to 15 mA during idle. We also perform data transfer using iPerf3 software via Ethernet between the BBB AP and

a laptop, both of which are connected to an Ethernet switch (test ID 5). A bidirectional (uplink and downlink) network bandwidth test is run between the client (e.g. BBB AP) and server (e.g. laptop). The test is configured to run indefinitely. At each time interval in the test, around 11.2 Mbytes of payload are transmitted with a bandwidth of around 94.2 Mbps. With continuous data transfer, the average current draw increases significantly to around 388 mA (1.94 W). This means that with data transfer via Ethernet, an additional current consumption of around 133 mA occurs, when compared to the idle current consumption (with both USB and Ethernet connected, i.e. test ID 3). Maximum power consumption of the BBB AP under load is around 3 W (test ID 6).

Table 7. Average current consumption of BBB AP under different configurations.

| **Test ID** | **Configuration** | **Average current/power consumption of BBB AP (mA / W)** |
|---|---|---|
| 1 | During Boot – USB + Ethernet connected– ~1:12 min (lamp OFF and BLE OFF) – lamp CAPE connected | 405 / 2.025 |
| 2 | Idle Current – NO USB and NO Ethernet (lamp OFF and BLE OFF) – lamp CAPE connected | 170 / 0.85 |
| 3 | Idle Current - USB + Ethernet connected (lamp OFF and BLE OFF) – lamp CAPE connected | 255 / 1.275 |
| 4 | Idle Current – Ethernet connected only, No USB (lamp OFF and BLE OFF) – lamp CAPE connected | 241 / 1.205 |
| 5 | Continuous data transfer through Ethernet + USB ON (lamp OFF and BLE OFF) – lamp CAPE connected | 388 / 1.94 |
| 6 | Continuous data transfer through Ethernet + USB ON (lamp 98% and BLE OFF) – lamp CAPE connected | 590 / 2.95 |
| 7 | Idle Current – No USB and No Ethernet - lamp CAPE NOT connected | 155 / 0.775 |
| 8 | Idle Current – Ethernet connected only, No USB - lamp CAPE NOT connected) | 229 / 1.145 |

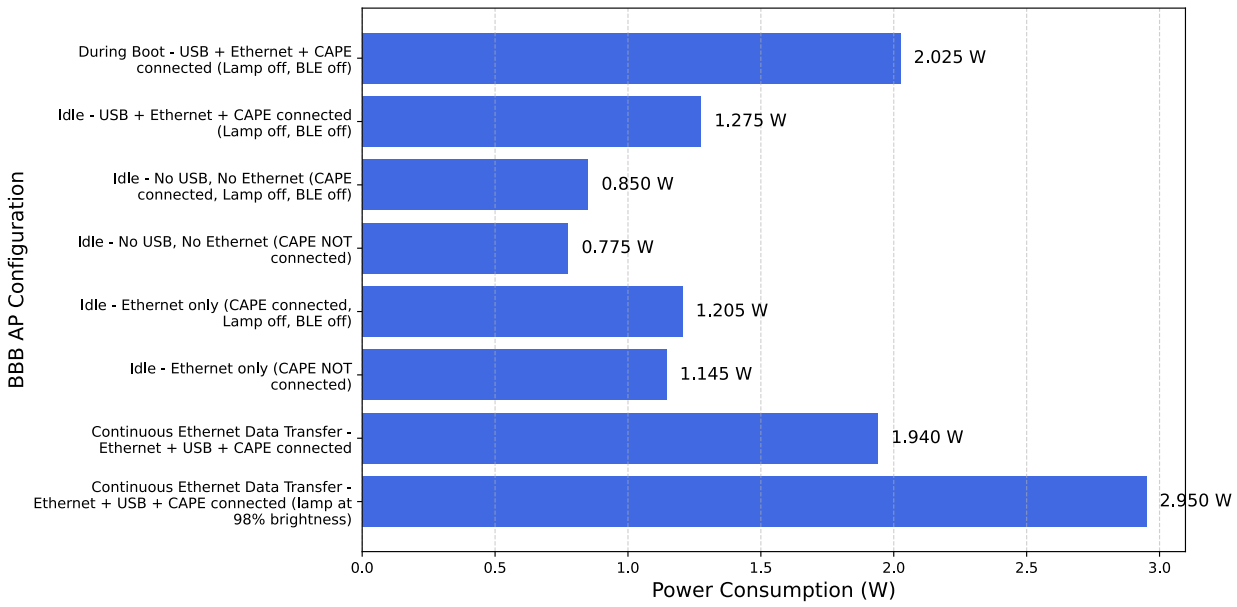



Figure 31. Power consumption of BBB AP under different configurations.

### 3.9.2 Impact of Different VLC PWM Duty Cycles on Average Current Consumption of BBB AP

Similar to the mini-lamp GW results, we vary the VLC PWM duty cycle which controls the brightness of the lamp LEDs and analyze its effect on the overall average current consumption on the BBB AP. Table 8 shows the average current consumption of the BBB AP under 3 test cases:

1) Only VLC enabled
2) VLC and BLE scanning (100 ms scanning interval, 50 ms scanning window)
3) VLC and BLE-connection with a node (connection interval = 45 ms).

Note that for all three test cases there was no data transfer between the BBB AP and the node (i.e. idle mode). The BLE TX power level is 0 dBm, and the USB and Ethernet ports were both connected on the BBB AP (no data transfer via Ethernet as well).

Similar to the mini-lamp GW results, regarding the mini-lamp GW, we observe the same trend for the BBB AP. Namely, the average current consumption of the BBB AP increases linearly with the VLC PWM duty cycle, as shown in Figure 32. When both VLC and BLE scanning are active, we notice an almost parallel upward shift in current consumption, with an average offset of approximately 5.31 mA, when compared to the average current consumption for VLC only (this offset was around 4.43 mA for the mini lamp GW). Similar to the mini-lamp GW results, when VLC is active and the BBB AP is connected to a node via BLE with a CI of 45 ms (idle with no data transfer), the difference in current consumption is negligible when compared to the average current consumption for VLC mode only. From Figure 32, we observe that the BBB AP consumes a higher current, with an average difference (excess) of approximately +235 mA (+1.175 W) for all three test cases when compared to the mini lamp GW's current consumption.

Based on the measurements in Table 8 and Figure 32, we can derive a 3rd order polynomial (cubic) equation for estimating the average current consumption of the BBB AP as a function of the VLC PWM duty cycle as follows:

$$I_{\text{BBB\_vlc\_idle\_only}}\,(\text{mA}) = 3 \times 10^{-5}x^3 - 5.582 \times 10^{-3}x^2 + 2.319x + 255.6543 \quad (3.8)$$

where $x$ represents the VLC PWM duty cycle (%), in the range $0 \leq x \leq 98\%$. The model in (3.8) achieves an RMSE, $R^2$ value, and MAPE of 1.55 mA, 0.9994, and 0.344%, respectively.

In Table 8 and Figure 32, we observe that the average current consumption of the BBB AP under **VLC idle + BLE-connected mode** is very similar to that of the VLC case only, i.e.

$$I_{\text{BBB\_vlc\_idle\_and\_ble\_conn\_idle}}\,(\text{mA}) \approx I_{\text{BBB\_vlc\_idle\_only}}\,(\text{mA}) \quad (3.9)$$

If we apply the model in (3.8) to the **VLC + BLE-connected mode**, the RMSE, $R^2$ value and MAPE are 1.62 mA, 0.9994, and 0.326%, respectively. When both VLC and BLE scanning are active, we notice an almost parallel upward shift in current consumption (average slope in Figure 32 is ~2.3 mA/%), with an average offset of approximately 5.38 mA, when compared to the average current consumption for VLC only. Thus, equation (3.8) can be modified as

$$I_{\text{BBB\_vlc\_idle\_and\_ble\_scan}}\,(\text{mA}) = 3 \times 10^{-5}x^3 - 5.582 \times 10^{-3}x^2 + 2.319x + 255.6543 + C \quad (3.10)$$

where $C$ represents the average offset.

Since the BBB AP exhibits a similar slope (~2.3 mA/%) to the mini-lamp gateway when BLE scanning is active (see Figure 32), we can approximate the BLE-scanning-only current equation for the BBB AP by adopting the mini-lamp GW's slope during BLE scanning only (9.30 mA/%) from Figure 28.(a) and adjusting the intercept to account for the BBB AP's higher baseline consumption. The average excess current consumption of the BBB AP relative to the mini-lamp GW for **VLC only** (baseline) is:

$$I_{\text{excess,VLC only}}\,(\text{mA}) = 254.54 - 11.97 = 242.57\ \text{mA},$$

(values extracted from Table 8 and Figure 25 considering a **VLC PWM duty cycle of 0%**).

and for **VLC + BLE scanning** at a 50% BLE duty cycle:

$$I_{\text{excess,VLC+BLE scan}}\,(\text{mA}) = 258.91 - 16.49 = 242.42\ \text{mA},$$

(values extracted from Table 8 and Figure 28.(a) for **VLC PWM duty cycle of 0%**).

Since the excess is nearly constant (~**242.5 mA**), we assume that BBB AP and mini lamp GW have the same slope, but BBB AP has a higher base consumption (due to BBB board – see Table 7). Thus, for the BBB AP, the average current consumption during **BLE scanning only (lamp off)** can be approximated as follows:

$$I_{\text{BBB\_ble\_scan\_only}}\,(mA) = 9.30 \times Duty\ Cycle + (11.80 + 242.5) = 9.30 \times Duty\ Cycle + 254.30 \qquad (3.11)$$

where $Duty\ Cycle = \frac{\text{BLE scanning window (ms)}}{\text{BLE scanning interval (ms)}}$.

This means that for the same BLE scanning conditions, the BBB AP has a similar increase per duty cycle as the mini lamp GW, but its base current draw is ~254.30 mA instead of 11.8 mA due to hardware differences. To incorporate the BLE scanning duty cycle, equation (3.10) can be re-written as:

$$I_{\text{BBB\_vlc\_idle\_and\_ble\_scan}}\,(\text{mA}) = 3 \times 10^{-5}x^3 - 5.582 \times 10^{-3}x^2 + 2.319x + 254.30 + 9.30 \times Duty\ Cycle \qquad (3.12)$$

where $Duty\ Cycle = \frac{\text{BLE scanning window (ms)}}{\text{BLE scanning interval (ms)}}$, and $x$ represents the VLC PWM duty cycle (%), in the range $0 \leq x \leq 98\%$. When this model is evaluated against the real measured data, we obtain an RMSE, $R^2$ value, and MAPE of 3.1 mA, 0.9977, and 0.614%, respectively.

Table 8. Average current consumption of BBB AP during VLC only and during VLC and BLE (scanning and connection) modes (both Ethernet and USB connected).

| | **Average current consumption (mA)** | | |
|---|---|---|---|
| **VLC PWM Duty Cycle (%)** | **VLC only (idle – No VLC TX & RX)** ($I_{\text{BBB\_vlc\_idle\_only}}$) | **VLC (idle) + BLE scanning (100,50) ms** ($I_{\text{BBB\_vlc\_idle\_and\_ble\_scan}}$) | **VLC (idle) + BLE-connected mode (connection interval = 45 ms) – No data transfer via BLE** ($I_{\text{BBB\_vlc\_idle\_and\_ble\_conn\_idle}}$) |
| 0 (lamp off) | 254.54 | 258.91 | 254.92 |
| 10 | 279.42 | 284.98 | 277.78 |
| 20 | 301.01 | 307.07 | 299.95 |
| 30 | 321.16 | 325.1 | 320.52 |
| 40 | 340.87 | 344.55 | 339.51 |
| 50 | 361.83 | 366.74 | 361.01 |
| 60 | 378.32 | 383.85 | 379.33 |
| 70 | 400.48 | 407.05 | 400.95 |
| 80 | 423.66 | 430.36 | 423.75 |
| 90 | 442.8 | 449.62 | 443.97 |
| 98 | 455.86 | 460.85 | 455.73 |

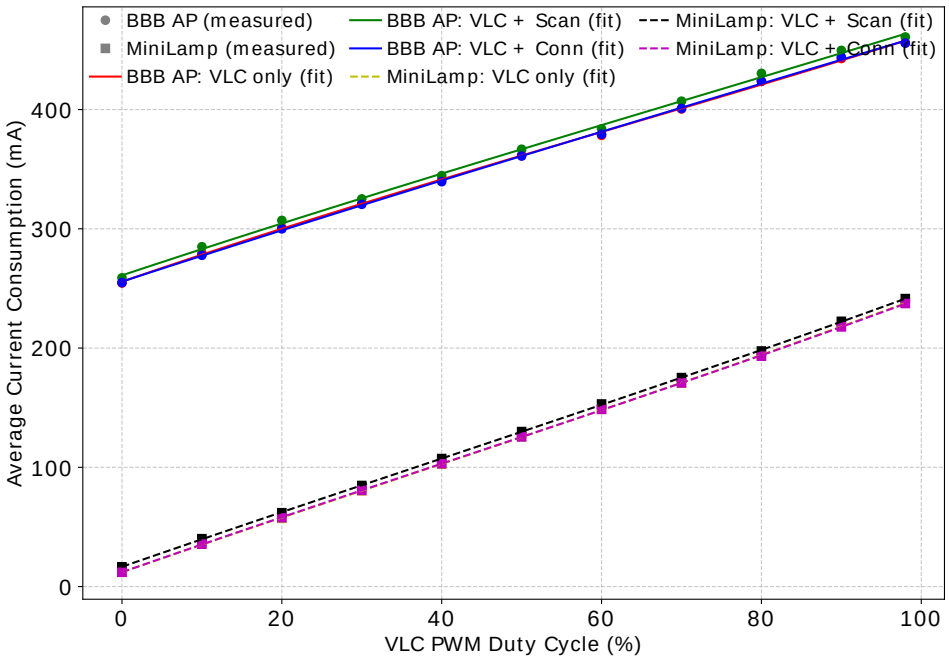



Figure 32. Comparison of the average current consumption between BBB AP and mini lamp GW under three test cases.

To evaluate the VLC transmission behavior, a frame composed of six 32-bit chunks (each lasting approximately 68 ms) was transmitted using the BBB AP, with BLE functionality disabled. The experiment was conducted across VLC PWM duty cycles ranging from 10% to 90%. The relationship between the VLC idle current and the average VLC transmission current (computed over the six frame chunks) is expressed as follows:

$$I_{\text{BBB\_vlc\_TX}}\,(\text{mA}) = 1.003 \times I_{\text{BBB\_vlc\_idle\_only}} + 0.4656 \tag{3.13}$$

Equation (3.13) suggests that the VLC TX current increases linearly with the idle current. The slope is close to 1, meaning VLC TX current is only slightly higher than idle current, indicating low additional power consumption during VLC transmission. The $R^2$ value is very close to 1, meaning the linear model is an excellent fit for the data. This confirms that VLC transmission does not introduce significant fluctuations in power consumption beyond idle consumption. The small difference between idle and VLC TX current suggests that the transmission process is energy efficient.

We also derive the current (in mA) relationship between **VLC idle + BLE scanning** and **VLC transmission + BLE scanning**, based on measurements obtained across VLC PWM duty cycles ranging from 10% to 90%, as follows:

$$I_{\text{BBB\_vlc\_TX\_with\_ble\_scan}}\,(\text{mA}) = 0.9995 \times I_{\text{BBB\_vlc\_idle\_and\_ble\_scan}} + 3.113 \tag{3.14}$$

The slope in the equation (3.14) is close to 1, meaning VLC transmission only marginally increases the power draw. The computed $R^2$ value is close to 1, confirming that a linear model is a good fit for the data. This means the relationship between idle current and transmission current is highly predictable. As the VLC PWM duty cycle increases, the overall power consumption (both idle and TX) increases proportionally. This is expected since a higher duty cycle means more energy is transmitted through the VLC channel. The relatively small difference between idle and VLC TX currents suggests that BLE scanning is the dominant contributor to power consumption in this scenario.

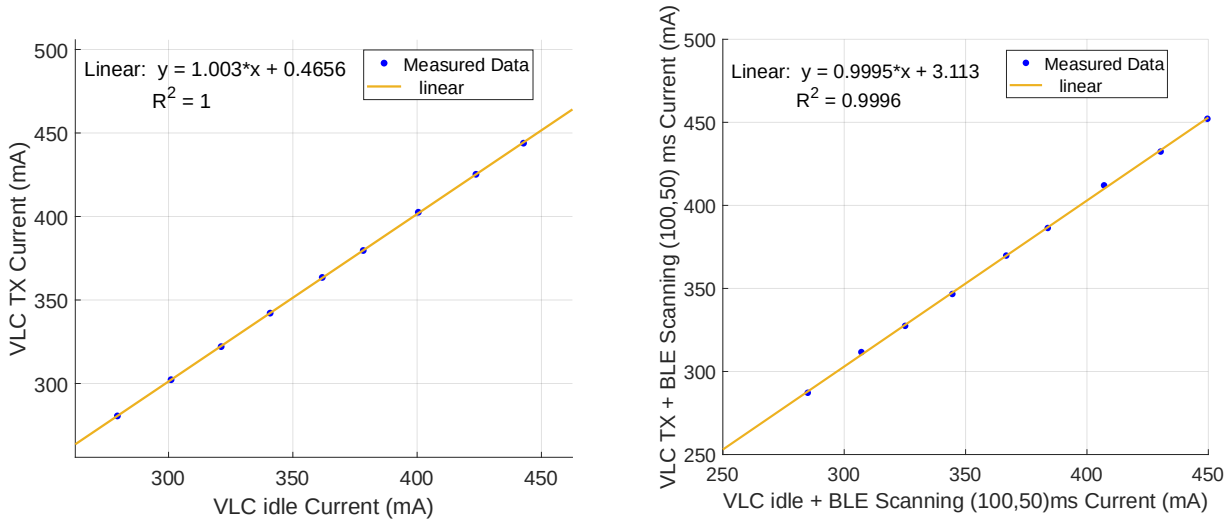



Figure 33. Relationship between. (a) VLC idle current versus VLC TX current (BLE off), and (b) VLC idle + BLE scanning current versus VLC TX + BLE scanning current (current consumption recorded on BBB AP with both USB and Ethernet connected).

As for the **VLC RX with and without BLE scanning**, the average current consumption remained essentially unchanged, thus we can approximate:

$$I_{\text{BBB\_vlc\_RX}}\,(\text{mA}) \approx I_{\text{BBB\_vlc\_idle\_only}} \tag{3.15}$$

$$I_{\text{BBB\_vlc\_RX\_with\_ble\_scan}}\,(\text{mA}) \approx I_{\text{BBB\_vlc\_idle\_and\_ble\_scan}} \tag{3.16}$$

### 3.9.3 Impact of BLE TX and RX on BBB AP's Energy Consumption

In this experiment, the BBB AP sends commands to the node by writing its BLE characteristics, and the node responds by sending the requested sensor and location data to the AP using BLE UART. As we have seen in Section 3.3, receiving commands or transmitting data at the node level has no measurable impact on energy consumption due to the small size of the data payload/command and much higher throughput in BLE when compared to VLC. In the same way, we analyzed the impact of receiving data and transmitting commands at the AP level. In practice, in Figure 30 a USB cable will be connected between the Seeed Xiao NRF52840 and the BBB platform. Thus, if we open a serial terminal on the Debian Linux operating system on the BBB AP, we will be able to see the information received from or transmitted to the node, as illustrated in Figure 34.

```
Data received!
"SID": "01","DID": "F1","T": 20.98,"H": 53.84,"P": 1022.19,"G": 105.07,"XYZ": }
```

Figure 34. Example of sensor readings and dummy location data transmitted from node to BBB AP (obtained from serial terminal output on Debian Linux system running on BBB AP).

In this experiment we vary the VLC PWM duty cycle, i.e. LED brightness, but no data is transmitted or received via VLC (idle). At the same time, the BBB AP is connected via BLE to the node using a BLE CI of 45 ms. The default TX power of the BBB AP is 4 dBm. The transmission of command via BLE causes a peak in the current profile with a duration of around 4.5 ms, while the reception of data from the node causes a peak in the current profile with a duration of around 2.5 ms. In Figure 35, we measure the average current consumption of the peaks during data reception (Figure 35(a)) and command transmission (Figure 35(b)) via BLE on the BBB AP and we did this for different PWM duty cycle ranging from 0% to 98% (VLC idle). Figure 35 suggests a strong linear correlation in both cases. In Figure 35(a), the additional current consumption during BLE sensor/location data reception (over a duration of ~4.5 ms) is on average around

103 mA over the range of VLC PWM duty cycles. In Figure 35(b), the additional current consumption during BLE command transmission (over a duration of ~2.5 ms) is on average around 80 mA over the range of VLC PWM duty cycles. The durations for BLE transmission and reception are very brief and won't have a measurable impact on the overall average current consumption. Nevertheless, we model the current (in mA) relationship between the **VLC idle + BLE connected (idle)** and **VLC idle + BLE connected (data reception)** states as follows:

$$I_{\text{BBB\_vlc\_idle\_and\_ble\_RX}}(\text{mA}) = 0.9915 \times I_{\text{BBB\_vlc\_idle\_and\_ble\_conn\_idle}} + 106.1 \quad (3.17)$$

In the same way, we model the current (in mA) relationship between the **VLC idle + BLE connected (idle)** and **VLC idle + BLE connected (command transmission)** states as follows:

$$I_{\text{BBB\_vlc\_idle\_and\_ble\_TX}}(\text{mA}) = 1.015 \times I_{\text{BBB\_vlc\_idle\_and\_ble\_conn\_idle}} + 74.25 \quad (3.18)$$

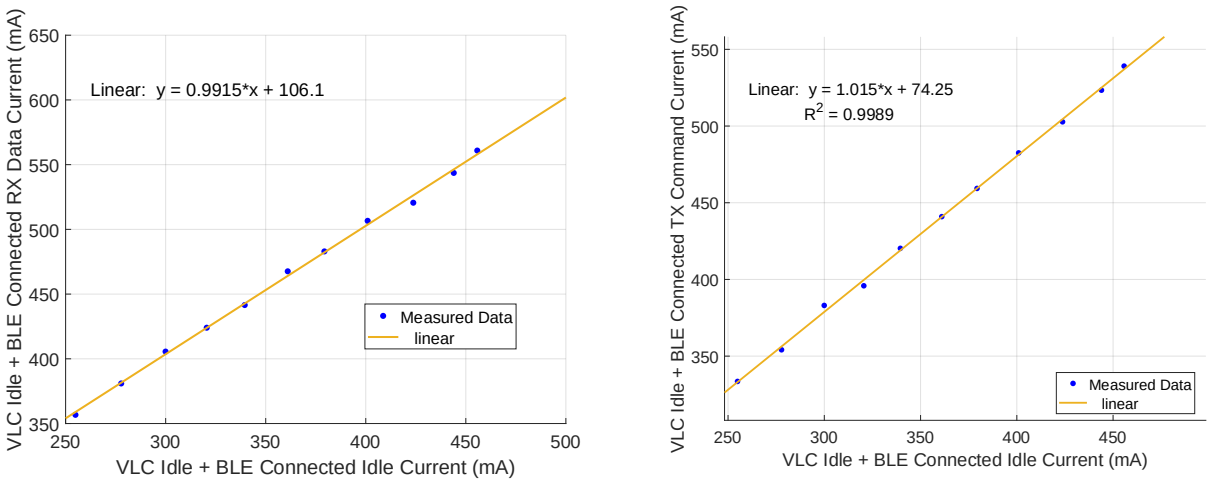



Figure 35. Impact of BLE TX and BLE RX on BBB AP's current consumption as analyzed under different VLC PWM duty cycles (idle, no VLC TX and VLC RX).

Similar to the node-level energy prediction application, we also developed an energy consumption prediction app for the BBB AP as shown in Figure 36. We define a scenario as per Table 9 and validate the App energy prediction output with new measurements on the BBB AP under different configurations and parameters, as shown in Figure 37. The validation results are given in Table 10. The comparison between the predicted and measured values shows a high level of accuracy, with all the metrics exhibiting results in the 97.5% to 97.8% range. This is a strong indication that the prediction model is highly effective in estimating the current consumption, charge, and energy of the BBB AP, closely matching the measured values.

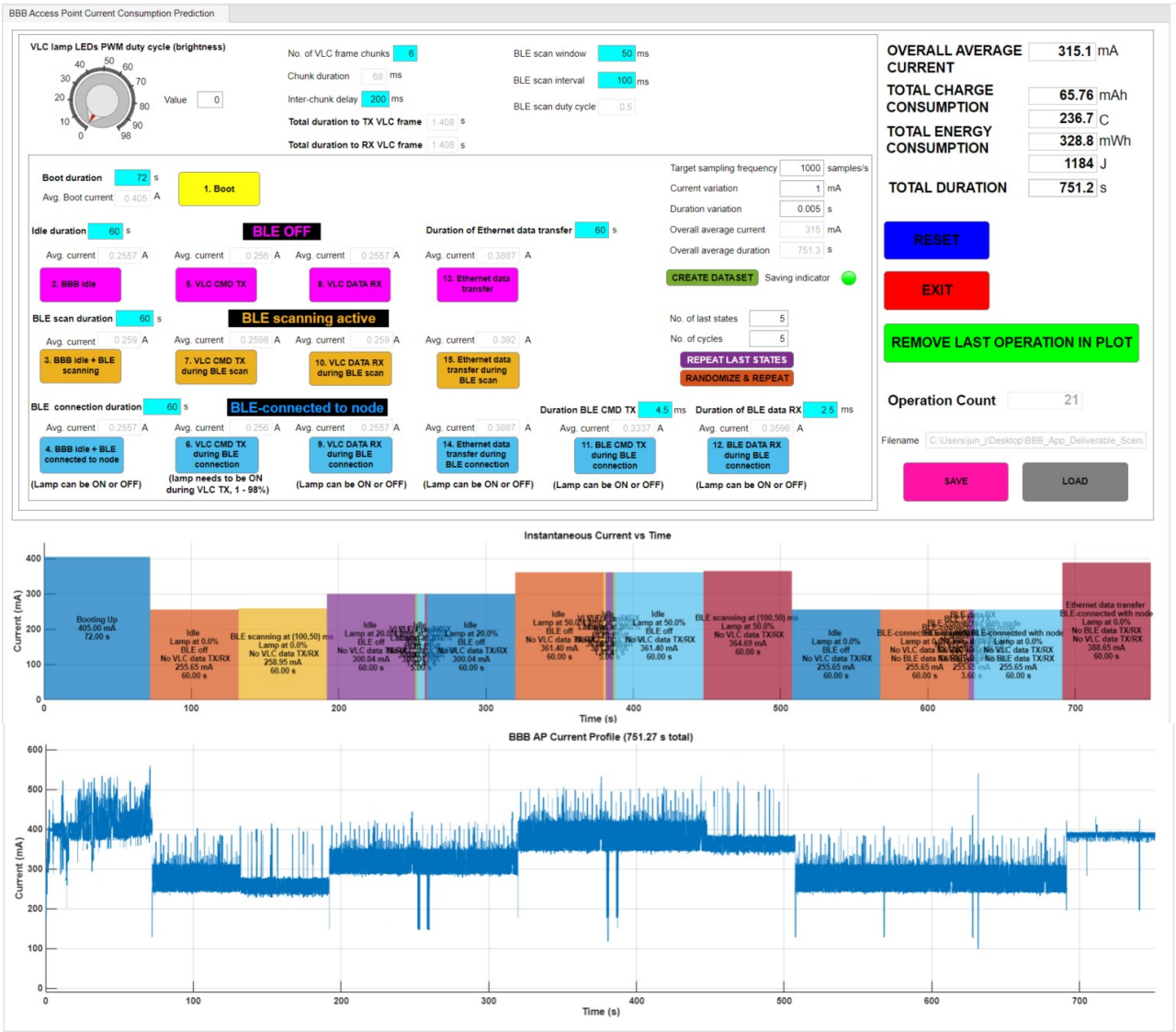



Figure 36. BBB AP energy consumption prediction App.

Table 9. Sequence of events on BBB AP for a given operating scenario.

| | **Operation** | **Duration (seconds)** | **Predicted Average current (mA)** |
|---|---|---|---|
| 1 | Boot (USB + Ethernet connected) - lamp off | 72 | 405 |
| 2 | Idle (VLC lamp off and BLE off) | 60 | 255.65 |
| 3 | VLC lamp off + BLE scanning (100,50) ms | 60 | 258.95 |
| 4 | VLC lamp ON and idle (20% duty cycle) – No VLC TX/RX, BLE off | 60 | 300.04 |
| 5 | VLC frame CMD TX to node requesting data (6 chunks of ~68 ms each, with inter-chunk delay of 200 ms) - (lamp at 20% duty cycle, BLE off) | 1.408 | 300.66 |
| 6 | VLC lamp ON and idle (20% duty cycle) – waiting for VLC data from node (BLE off) | 5 | 300.04 |

| | | | |
|---|---|---|---|
| 7 | VLC frame data RX from node with requested data (6 chunks of ~68 ms each, with inter-chunk delay of 200 ms) - BLE off - lamp at 20% duty cycle | 1.408 | 300.04 |
| 8 | VLC lamp ON and idle (20% duty cycle) – No VLC TX/RX, BLE off | 60 | 300.04 |
| 9 | VLC lamp ON and idle (50% duty cycle) – No VLC TX/RX, BLE off | 60 | 361.40 |
| 10 | VLC frame CMD TX to node requesting data (6 chunks of ~68 ms each, with inter-chunk delay of 200 ms) - (lamp at 50% duty cycle, BLE off) | 1.408 | 362.10 |
| 11 | VLC lamp ON and idle (50% duty cycle) – waiting for VLC data from node (BLE off) | 5 | 361.40 |
| 12 | VLC frame data RX from node with requested data (6 chunks of ~68 ms each, with inter-chunk delay of 200 ms) - BLE off - lamp at 50% duty cycle | 1.408 | 361.40 |
| 13 | VLC lamp ON and idle (50% duty cycle) – No VLC TX/RX, BLE off | 60 | 361.40 |
| 14 | VLC lamp ON (50% duty cycle) + BLE scanning (100,50) ms | 60 | 364.69 |
| 15 | Idle (VLC lamp off and BLE off) | 60 | 255.65 |
| 16 | Connect to node via BLE at CI of 45 ms but no BLE data TX/RX (VLC lamp off) | 60 | 255.65 |
| 17 | Send a BLE CMD to node to request data (lamp off) | 0.0045 | 333.74 |
| 18 | BLE connected mode with node at CI of 45 ms – lamp off – waiting for sensor data from node via BLE | 3.6 | 255.65 |
| 19 | RX data from node via BLE (lamp off) | 0.0025 | 359.60 |
| 20 | BLE connection maintained with node but no TX/RX of commands/data – lamp off | 60 | 255.65 |
| 21 | Ethernet dummy data transmission (lamp off) – while still connected via BLE to node | 60 | 388.65 |
| | **Total duration of operations (s)** | **751.2** | |
| | **Overall average current consumption (mA)** | | **315.1** |
| | **Overall energy consumption (mWh)**<br>**Total Charge Consumption (C)**<br>**Overall Energy Consumption (Joules)** | **328.8**<br>**236.7**<br>**1184** | |

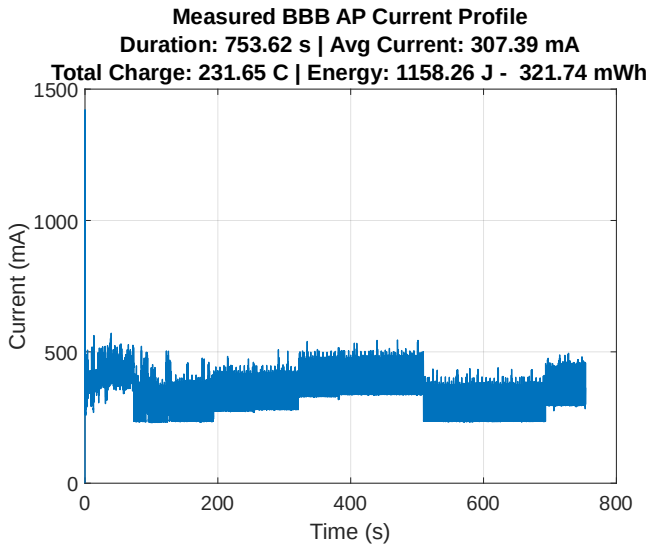



Figure 37. Measured BBB AP current profile based on operating scenario defined in Table 9.

Table 10. BBB AP validation results.

| Metric | Predicted | Measured | Accuracy | MAPE |
|---|---|---|---|---|
| Average Current (mA) | 315.1 | 307.39 | 97.49% | 2.51% |
| Total Charge (Coulombs) | 236.7 | 231.65 | 97.82% | 2.18% |
| Energy (Joules) | 1184 | 1158.26 | 97.78% | 2.22% |
| Energy (mWh) | 328.8 | 321.74 | 97.81% | 2.19% |

## 3.10 Representative Energy Models for Other Networking Elements

In addition to the RIoT nodes, mini-lamp GW and BBB AP, additional networking components are required to facilitate end-to-end information flow from source to sink. In the network architecture, the BBB APs connect to an Ethernet switch, which also links to a master node responsible for network and communication management, as well as AP registration. The master node hosts two MQTT brokers: one for master-to-cloud communication and another for AP-to-master communication. Additionally, an internet router is connected to the Ethernet switch, enabling data transmission to a cloud broker over the internet. This setup allows for remote management of network entities.

### 3.10.1 Raspberry Pi based Master Node

The master node used in the IoT network will be implemented on a Raspberry Pi (RPi). It is important to note that each RPi model exhibits different power consumption characteristics, as shown in Figure 38 [3].

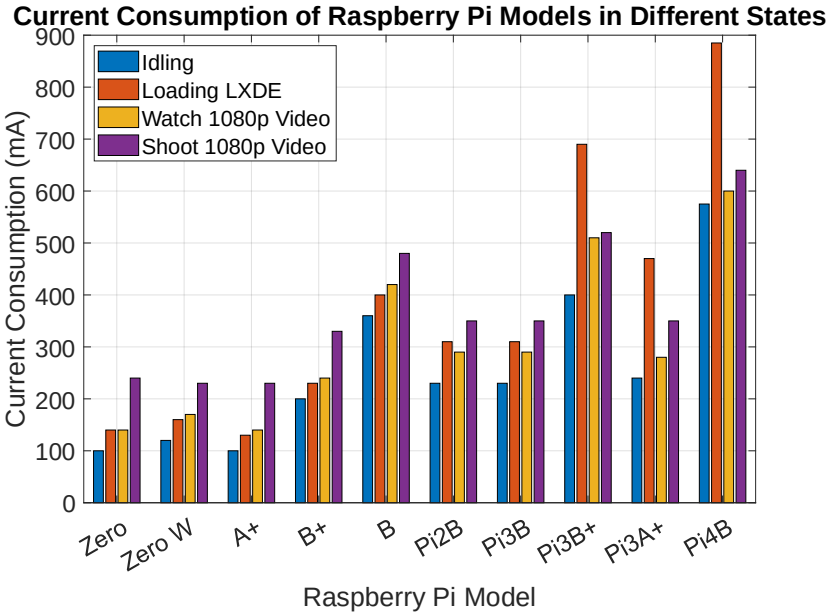



Figure 38. Current consumption of RPi models [3].

Newer Raspberry Pi models offer improved performance at the cost of higher power consumption. The Pi4B typically requires between 2.5 W and 5 W, depending on workload and peripherals, making it less suitable for battery-powered IoT applications. For low-power IoT projects, the Raspberry Pi Zero or Zero W is a more energy-efficient choice, consuming under 1 W in typical scenarios. In [4] it is reported that the RPi5 can draw around 4-5 W, but this can spike up to 12 W under full load. The PCIe port, faster USB ports, improved CPU performance, and the potential for power-hungry HATs (add-on boards) all contribute to the increased power demands. A power consumption comparison between RPi4 and RPi5 is shown in Figure 39 [4].

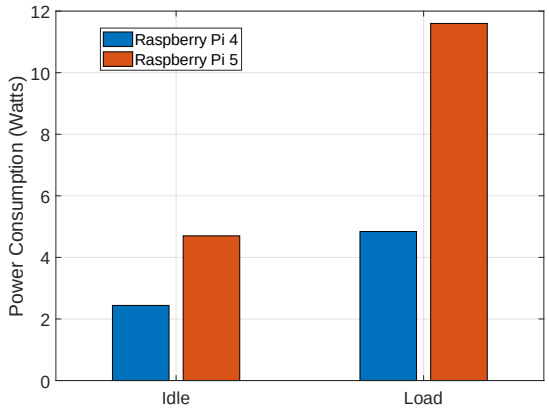



Figure 39. Power consumption of Raspberry Pi 4 & 5 [4].

Not all Raspberry Pi models are equally efficient. If intensive processing is not required, using a Pi Zero or Pi3A+ instead of a Pi4B can significantly reduce energy consumption. Reducing the clock speed of the Raspberry Pi lowers energy consumption [5]. Underclocking the CPU (e.g., from 1.5 GHz to 1 GHz) can yield significant power savings while maintaining adequate performance for many applications. Additionally, reducing core voltage further minimizes power draw, though care must be taken to avoid system instability. Certain built-in peripherals such as HDMI, USB, and LED indicators consume power even when unused. Disabling these can save energy. Though individual savings are small, they add over extended periods. The Raspberry Pi lacks built-in deep sleep capabilities like microcontrollers, but power consumption can still be reduced through scheduling. Furthermore, IoT applications often use external peripherals such

as sensors, network interfaces and displays, which should be carefully selected for efficiency. Optimizing power consumption in Raspberry Pi-based IoT applications requires a combination of hardware and software strategies. Selecting the appropriate model, underclocking, using sleep modes, disabling unused peripherals, and managing external power draw can extend battery life and improve efficiency [5].

#### *3.10.1.1 Modelling Power Consumption of Raspberry Pi*

In [6], a power consumption model was developed for the Raspberry Pi Model B, which serves as an alternative to conventional home gateways. Their study examines the impact of key hardware components on overall energy consumption. By measuring the device's various power states, they derived a model that estimates power usage based solely on CPU and network utilization. Four metrics are collected from the OS in terms of CPU cycles: *user*, *nice*, *system*, and *idle*. The CPU monitor reads these cycles from the */proc/stat file* folder, and the cycles are used to calculate the system utilization *u[t]* at time *t* as follows:

$$u[t] = \frac{c_{\text{busy}}[t] - c_{\text{busy}}[t-1]}{c_{\text{total}}[t] - c_{\text{total}}[t-1]} \tag{3.19}$$

where $c_{\text{total}}[t]$ is the sum of busy cycles $c_{\text{busy}}[t]$ and the number of idle cycles, $c_{\text{idle}}[t]$. Hence, the utilization at time *t* is calculated based on the difference of utilization cycles during the last measurement period. $c_{\text{busy}}[t]$ is the total number of busy cycles up to time *t* and is defined by:

$$c_{\text{busy}}[t] = c_{\text{user}}[t] + c_{\text{nice}}[t] + c_{\text{system}}[t] \tag{3.20}$$

where $c_{\text{user}}[t]$ are the user generated CPU cycles, $c_{\text{nice}}[t]$ and $c_{\text{system}}[t]$ denote the number of cycles created by low priority processes, and the system, respectively.

Similar to the CPU monitor, the authors of [6] wrote a network monitor to keep track of the network utilization. The */proc* filesystem is read at */proc/net/dev*, returning the processed packets on kernel level. This is advantageous, as it contains the raw number of bytes sent and received via the interface. Similar to the */proc/stat* file, the counters are incremental. The current bandwidth (in B/s) is calculated by:

$$r[t] = \frac{B[t] - \mathrm{B}[t-1]}{\Delta T} \tag{3.21}$$

where $\Delta T$ is the time interval between *t−1* and *t*, and $B[t]$ is the absolute amount of data transmitted or received on the interface.

The power model based on CPU utilization can be expressed as:

$$P_{\text{Pi,CPU}}(u) = 1.5778 + 0.181u \text{ Watts} \tag{3.22}$$

where $u$ is the CPU utilization in the range 0 to 1, as per equation (3.19).

Table 11 presents approximations of the Raspberry Pi model B power consumption under various utilization levels. The first column lists the symbols used in the text to reference each function, while the second column indicates the order of approximation. The third column provides the RMSE, representing the expected mean error when using the model. The final column contains a formula describing the relationship between utilization and power consumption. The table is organized into distinct measurement categories. The first group details the idle power consumption of the Raspberry Pi in different power states, where $P_{\text{Eth,idle}}$ and $P_{\text{WiFi,idle}}$ represent power draw in idle conditions. The second group illustrates power consumption as a function of CPU utilization, denoted by *u*. The remaining groups focus on power consumption related to data transfers across both network interfaces. Each model in these groups includes an initial correction term to align with previously determined constants, followed by terms that depend on either the transferred data rate (*r* in Mbps) or CPU utilization (*u*). This

results in an additive model, enabling the estimation of total power consumption on a per-component basis. The model is expressed as:

$$P_{\text{Pi}} = P_{\text{idle}} + P_{\text{CPU}}(u) + \sum_{\text{if}} (P_{\text{if,idle}} + P_{\text{if,up}}(r) + P_{\text{if,dn}}(r)) \quad (3.23)$$

where $P_{\text{idle}}$, $P_{\text{if,idle}}$ and the approximations $P_{\text{CPU}}(u)$ and $P_{\text{if,dn}}(r)$ are defined in Table 11. Here, "*if*" denotes either the Wi-Fi or Ethernet interface. The RMSE values for most built-in components are relatively low (<18 mW), indicating high model accuracy [6].

Table 11. Power models of Raspberry Pi model B. *u* is the CPU utilization in the range 0 to 1, and *r* the traffic rate in Mbps [6].

| Function | Order | RMSE | Model |
|---|---|---|---|
| $P_{\text{idle}}$ | 0 | 15 mW | 1.5578 Watts |
| $P_{\text{Eth,idle}}$ | 0 | 9 mW | 0.294 Watts |
| $P_{\text{WiFi,idle}}$ | 0 | 8 mW | 0.942 Watts |
| $P_{\text{CPU}}(u)$ | 1 | 18 mW | $0.181u$ Watts |
| $P_{\text{Eth,dn}}(r)$ | 4 | 14 mW | $-0.008 + 4.792e^{-3}r - 0.164e^{-3}r^2 + 2.509e^{-6}r^3 - 12.498e^{-9}r^4$ |
| $P_{\text{Eth,up}}(r)$ | 2 | 8 mW | $-0.002 + 5.542e^{-3}r - 45.85e^{-6}r^2$ |
| $P_{\text{WiFi,dn}}(r)$ | 2 | 26 mW | $0.01 + 11.003e^{-3}r - 71.988e^{-6}r^2$ |
| $P_{\text{WiFi,up}}(r)$ | 2 | 71 mW | $0.02 + 24.387e^{-3}r - 112.8e^{-6}r^2$ |

### 3.10.2 Typical Energy Consumption of Network Infrastructure Devices (Switches and Routers)

A network element's power consumption can be categorized into two states: idle and active. Energy efficiency is often measured using the energy consumption per bit of transmitted data, expressed in joules per bit (J/b). This metric is widely used to assess the efficiency of individual devices or entire networks. As illustrated in Figure 40, the power consumption of various network devices, such as ADSL modems, routers and switches, typically follows a characteristic linear relationship with load [7][8]:

$$P(R)_{\text{unshared}} = P_{\text{idle}} + \left(\frac{P_{\text{max}} - P_{\text{idle}}}{R_{\text{max}}}\right) R = P_{\text{idle}} + E_{\text{inc}} R \quad (3.24)$$

where $P(R)_{\text{unshared}}$ represents the power consumption of a network element with a maximum capacity $R_{\text{max}}$ (bits/sec) and actual traffic load $R$ (bit/sec). $P_{\text{max}}$ is the maximum power consumption. $P_{\text{idle}}$ represents the no-load power consumption (when $R = 0$). $E_{\text{inc}}$ is the incremental power component for a bit rate $R$ (incremental energy per bit), which is obtained from the slope in Figure 40.

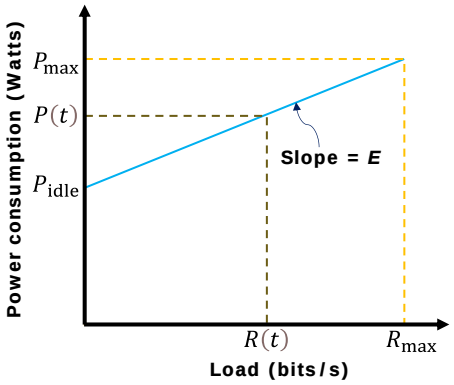



Figure 40. Power consumption profile of a generic network element.

In addition to the traffic-dependent power consumption $E_{\text{inc}}R$, the "capacity-based" modelling approach distributes a device's idle power consumption across its traffic streams using the expression $P_{\text{idle}}/UR_{\max}$, where $U$=average utilization [8]. Consequently, the average energy per bit, $\lambda$, of a network element is given by:

$$\lambda = \frac{P_{\text{idle}}}{UR_{\max}} + E_{\text{inc}} \tag{3.25}$$

Since multiple services can simultaneously access a network element, the additional energy consumption of a given network element $n$, attributable to a specific service $E_{\text{n}}$ that generates, transmits, and/or receives a total traffic volume of $A_{\text{bit}}$ (in bits), is expressed as:

$$E_{\text{n}} = \left(\frac{P_{\text{idle}}}{UR_{\max}} + E_{\text{inc}}\right) A_{\text{bit}} = \lambda A_{\text{bit}} \tag{3.26}$$

The network switch in the IoT network is generally used to link the IoT gateways and all IoT devices, for example, to a public network (e.g. internet) or server which is used to collect data from the nodes and display the information on a dashboard.

### *3.10.2.1 Modelling Energy Consumption of Unshared Network Devices*

An unshared network device refers to network equipment that is dedicated to a single or few users (e.g. Customer Premise Equipment (CPE) modem). This includes an access modem for xDSL services, a Passive Optical Network (PON) customer's optical network terminal, and a personal Wi-Fi router for home use (excluding shared or corporate Wi-Fi networks) [9]. Using a power-based model for single-user equipment [10], the power consumption of an unshared network device is often expressed as in equation (3.24). The difference between $P_{\text{idle}}$ and $P_{\max}$ is usually quite small; $P_{\text{idle}}$ can be as high as 90% of $P_{\max}$ in high-capacity equipment [7]. In this case, the product $E_{\text{inc}}R$ is insignificant for small values of $R$ (<1 Mbps), and thus typical power consumption values from datasheets can be accepted as a representative power value for the overall data throughput range. If $P_{\text{idle}} \geq 0.90P_{\max}$, then

$$E_{\text{inc}} = \frac{P_{\max} - P_{\text{idle}}}{R_{\max}} \approx 0 \tag{3.27}$$

Under this approximation, the energy consumption of an unshared network device for a period of time $T$ is often approximated as:

$$E_{\text{unshared}} = \int_0^T P(R)_{\text{unshared}}\, dt \approx P_{\text{idle}} \times T \tag{3.28}$$

### *3.10.2.2 Modelling Energy Consumption of Shared Network Devices*

A shared network device is a high-capacity network element that simultaneously supports numerous users, applications, and services, often ranging from hundreds to tens of thousands. This means that network administrators need to monitor and maintain the network equipment utilization $\rho$ (a percentage of $R_{\max}$), such that $0 < \rho < 1$. Consider an IoT Gateway (IGW) with a data rate $R_{\text{IoT}}$ such that:

$$R_{\text{IoT}} = \sum_{i} R_{\text{dev}_i}, \qquad \text{for } i = 1,2,\dots,N \tag{3.29}$$

where $R_{\text{dev}_i}$ is the data rate of an IoT device *i*, and *N* is the total number of IoT devices connected to the IGW. The background traffic generated ($R_{\text{bgd}}$) by all other services can be expressed as [8]:

$$R_{\text{bgd}} = \sum_{s} R_{\text{svc}_s}, \qquad \text{for } i = 1,2,\dots,N_s \tag{3.30}$$

where $R_{\text{svc}_s}$ is the data rate of a unique service *s*, and $N_s$ is the total number of services accessing the shared network device. Hence, the aggregated total traffic, $R_{\text{total}}$, is given by:

$$R_{\text{total}} = R_{\text{bgd}} + R_{\text{IoT}} \tag{3.31}$$

The background traffic level ($R_{\text{bgd}}$) of a shared network device at any time of the day is modelled as $R_{\text{bgd}} = \rho R_{\max}$. For a shared access network element carrying traffic in addition to its background traffic, its total power consumption can be expressed as:

$$P(R)_{\text{shared}} = P_{\text{idle}} + \frac{P_{\max} - P_{\text{idle}}}{R_{\max}} \times (R_{\text{bgd}} + R_{\text{IoT}}) \tag{3.32}$$

where $R_{\text{IoT}}$ represents a segment of total traffic through a shared network access device that is generated by IoT devices at the edge of the network. We use $E_{\text{bit}}$ as a convenient metric that allows us to add the energy used by individual data streams as they pass through several network elements. The overall network element's energy-per-bit can hence be expressed as:

$$E_{\text{bit}} = \frac{P(R)_{\text{shared}}}{R_{\text{bgd}} + R_{\text{IoT}}} = \frac{P_{\text{idle}}}{R_{\text{bgd}} + R_{\text{IoT}}} + \frac{P_{\max} - P_{\text{idle}}}{R_{\max}} \tag{3.33}$$

The average power consumption of the shared network device with average data rate $R_{\text{IoT}}$, at a specific time of day with background traffic, $R_{\text{bgd}_l}$, is given by:

$$P_l(R_{\text{IoT}}) = \frac{P_{\text{idle}} \times R_{\text{IoT}}}{\mu_l \langle R_{\text{ave}} \rangle + R_{\text{IoT}}} + \left(\frac{P_{\max} - P_{\text{idle}}}{R_{\max}}\right) R_{\text{IoT}} \tag{3.34}$$

where $R_{\text{bgd}_l} = \mu_l \langle R_{\text{ave}} \rangle$ is the background traffic in interval of traffic level *l*, $\mu_l$ is the load level, and $R_{\text{ave}}$ is the average daily data rate.

## 3.11 RIoT Network-Level Energy Measurement Setup

Accurate measurement of energy consumption is widely recognized as a cornerstone for designing sustainable IoT networks, particularly as device deployments scale and application requirements become increasingly diverse. Existing approaches generally fall into two categories: simulations or isolated node-level measurements that primarily capture micro-level factors, such as device hardware configurations and local operating conditions. While these methods provide useful insights, they often overlook broader macro-level influences, including

network topology, the number of active nodes, communication protocols, and aggregate payload sizes, which can significantly affect overall energy demand.

Conversely, many network-level measurement frameworks focus largely on macro-level parameters and tend to exclude detailed node-level characteristics. This separation between micro- and macro-level perspectives limits the ability to gain a comprehensive and precise understanding of energy consumption—an understanding that is especially critical for energy-sensitive applications such as remote environmental monitoring, large-scale smart city systems, and real-time healthcare networks.

Furthermore, current measurement processes often depend on manual data collection and ad-hoc instrumentation. This reliance introduces inconsistencies, reduces scalability, and makes it challenging to reflect the dynamic and unpredictable nature of real-world IoT deployments.

To address these limitations, we propose an automated, network-level energy measurement framework that systematically captures both micro-level (node-specific) and macro-level (network-wide) factors under realistic operating conditions. By producing time-synchronized datasets, this framework supports the development of AI/ML models for accurate prediction and optimization of energy consumption across resource-constrained IoT networks, ultimately enabling more sustainable and adaptive system designs.

### 3.11.1 RIoT Network Energy Measurement and Analysis

In this work, we designed and deployed a comprehensive measurement setup to accurately capture the real-world energy consumption of the RIoT network under diverse operating conditions. Recognizing that energy demand in IoT networks depends on multi-level factors, spanning node configurations, and communication modalities. Our setup was tailored to systematically analyze these interdependencies.

Key network-level factors considered include the number of active nodes and mini lamps, duration of each device state: idle, advertising, operation, deep sleep, and Payload size during operation (in bytes). Complementary node-level factors such as BLE/VLC configuration parameters Transmit power (-4 dBm, 0 dBm, and +4 dBm), advertising interval (fast:20-100 ms, slow:100-2000 ms), PHY rate (1 Mbps, 2 Mbps, 500 kbps/125 kbps), MTU (23-247 bytes), connection interval (7.5-4000 ms), and communication protocols were integrated into the measurement framework. Together, these allow for a detailed investigation of how isolated and combined parameter variations influence total network energy consumption. Figure 41 illustrates the complex interplay between node-level and network-level factors shaping overall energy consumption. This analytical foundation ensures that our measurement data captures realistic usage scenarios and informs accurate modeling and optimization.

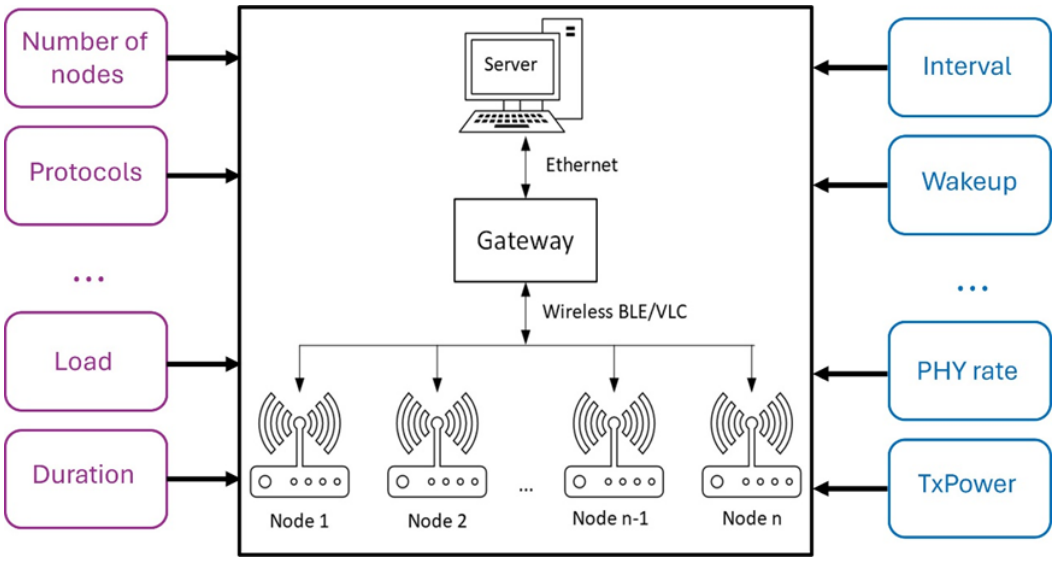



Figure 41. Network-level factors and node-level factors exert a multi-faceted influence on total energy consumption.

### 3.11.2 Energy Consumption Collection System

To address the challenge of acquiring precise, time-synchronized energy data from distributed IoT nodes and gateways, we developed an automated Energy Consumption Collection and Analysis System. The system architecture (Figure 42) comprises three key components:

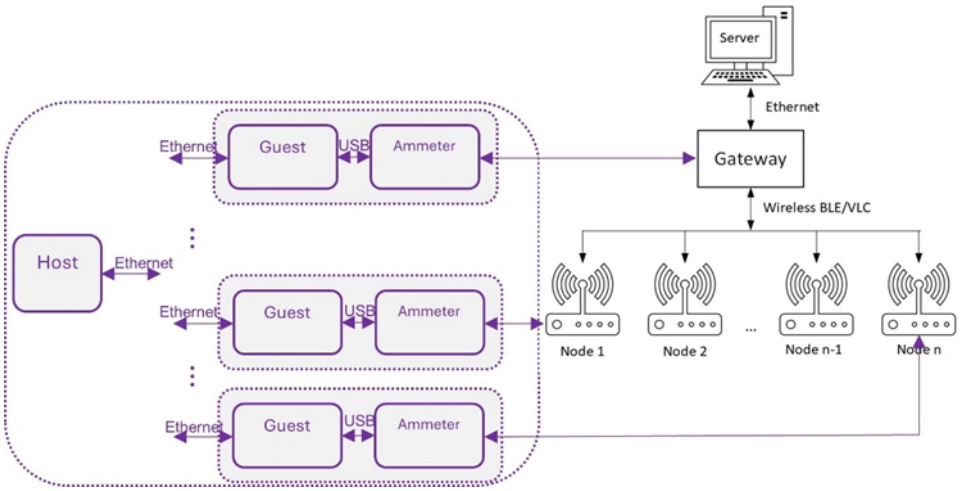
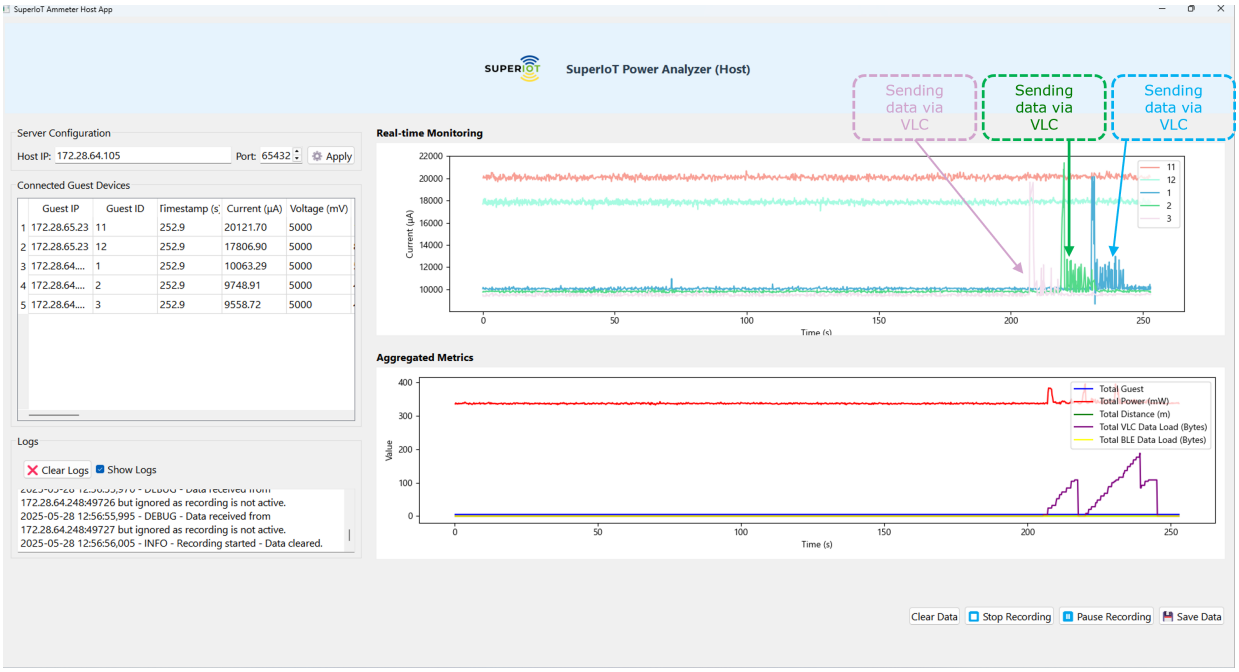


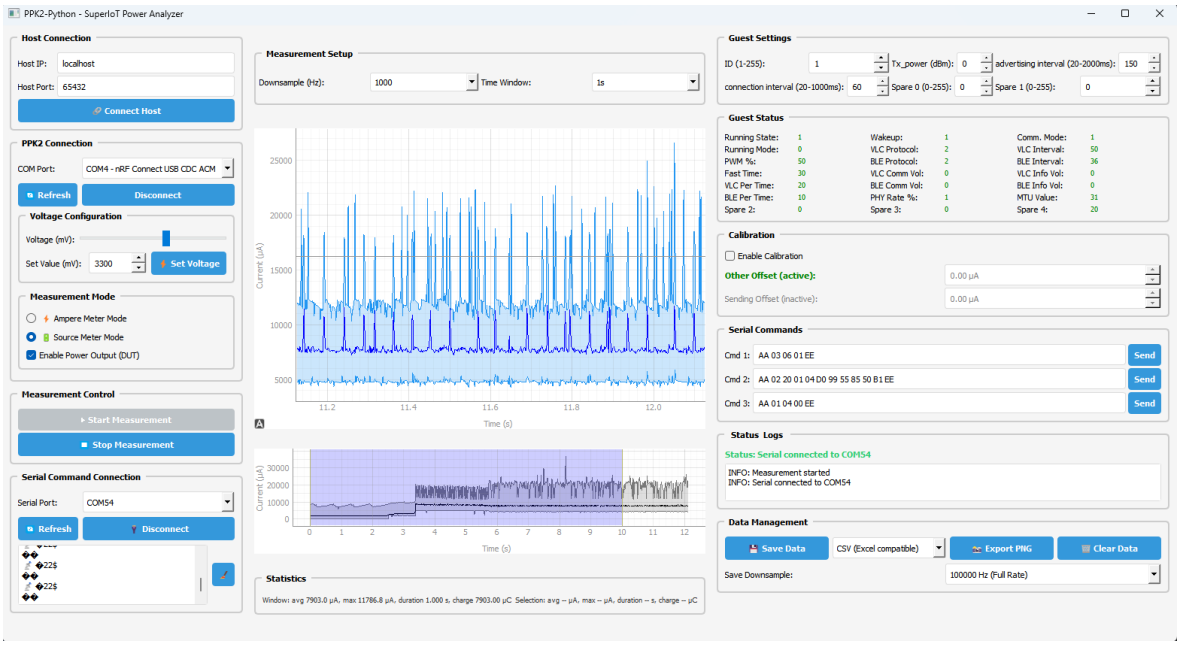

Figure 42. An automated Energy Consumption Collection and Analysis System.

- Host PC and Host Application: Centrally aggregates and manages data, coordinating measurement sessions, and storing synchronized datasets.
- Guest PCs and Guest Applications: Deployed for each IoT node and gateway, these applications collect local current readings and relevant node-level factors (e.g., BLE/VLC configuration parameters) and transmit them to the Host Application.
- Power measurement devices such as Power Profiler Kit II (PPK II) boards: Integrated with each node and gateway, these ammeters provide fine-grained current measurements, ensuring precise energy profiling.

The Host and Guest Applications feature user-friendly interfaces (Figure 43), enabling real-time monitoring and system configuration. This modular and automated setup supports scalable data acquisition, crucial for characterizing network energy consumption across multiple deployment scenarios.




(a) Host Application




(a) Guest Application

Figure 43. The user interfaces for the Host and Guest Applications.

Figure 44 illustrates the actual IoT network setup, which consists of three sensor nodes and two mini lamps functioning as controllable actuators, along with three computers configured for data collection and aggregation. Specifically, one computer runs a guest application to collect data from the sensor nodes, a second computer runs another guest application to gather data from the mini lamps, and a third computer operates a host application that aggregates and manages data received from both guest systems. These components are interconnected according to the architecture shown in Figure 41. This design supports synchronized, automated measurement and analysis of energy consumption across both sensing and actuation layers under realistic operating conditions, enabling a comprehensive evaluation of micro-level and macro-level factors that influence the overall energy performance of the IoT network.

Figure 44. Actual experimental IoT network energy measurement setup.

### 3.11.3 Different Communication Protocols

IoT network deployments necessitate a variety of communication protocols to accommodate diverse use case requirements, such as high security, reliability, or lower transmission frequencies. In this work, we propose two distinct communication protocols for VLC, namely a Byte level protocol and a Bit level protocol. Figure 45 illustrates the defined format of the Byte level protocol. This protocol employs a header composed of 5 bytes, encompassing fields for: Start, Sender Address, Receiver Address, Command, and Data Length. Following the header is a data payload of 'n' bytes, the length of which is determined by the Data Length field. Finally, a 2-byte Trailer concludes the frame. Therefore, the total frame size for the Byte level protocol is 7 + n bytes.

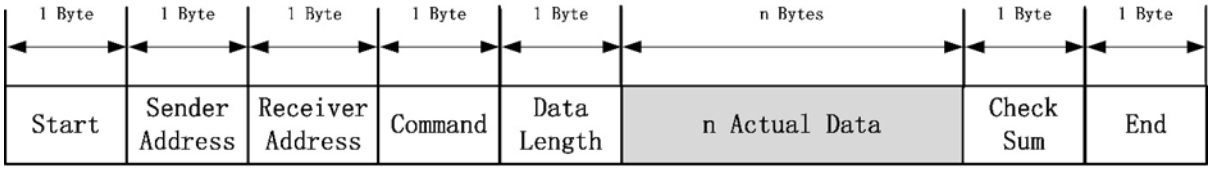



Figure 45. The structure of the Byte level protocol.

The structure of the Bit level protocol is detailed in Figure 46. While sharing a similar frame format with the Byte level protocol, the Bit level protocol employs a more compact header. Specifically, it utilizes a 20-bit header (comprising 2 bytes and 4 bits), in contrast to the Byte level protocol's 5-byte header. The Trailer size remains consistent at 1 byte for the Bit level protocol. Consequently, the total frame size for the Bit level protocol is reduced to 3.5 bytes + n bytes, which is demonstrably smaller than that of the Byte level protocol.

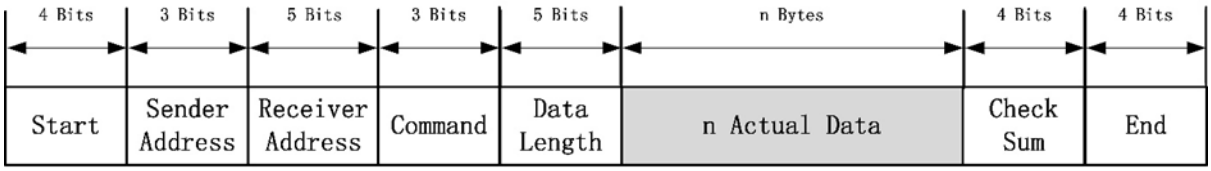



Figure 46. The structure of the Bit level protocol.

To verify the functionality of the Byte level protocol, a serial port (simulating a BBB) was used to transmit a command to the lamp. Subsequently, the lamp autonomously received a response frame from the designated IoT node, utilizing the Byte level communication protocol. Detailed experimental results from this interaction are presented in Figure 47.

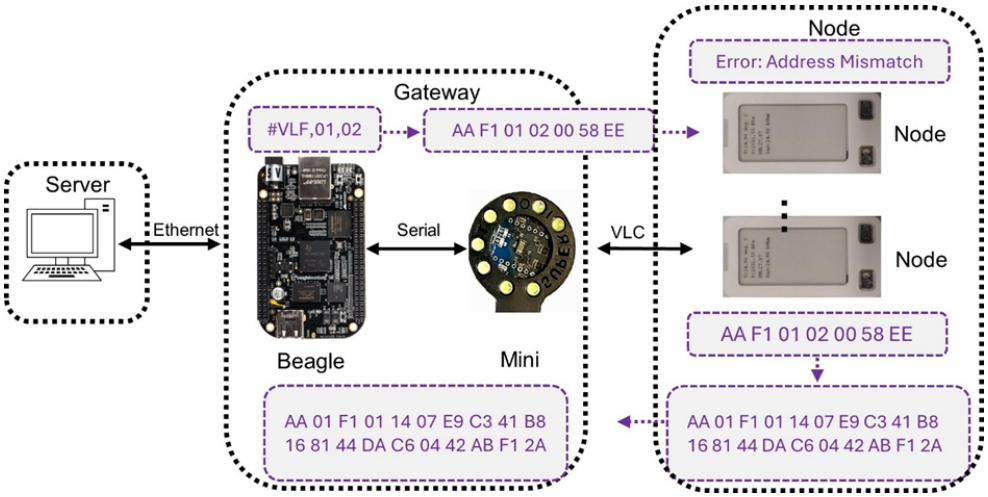



Figure 47. Illustration of sensor data collection via the Byte level protocol.

### 3.11.4 Energy Consumption Dataset Preprocessing

Based on the network-level energy measurement setup, raw data were collected from distributed nodes across 27 distinct IoT network configurations (see in Figure 48). Each configuration was evaluated using three independent nodes, resulting in a total of 81 experimental trials. The final dataset contains 1,026 measurement samples, each characterized by seven input features: operational state (State), VLC payload size (VLC_Payload), BLE payload size (BLE_Payload), transmission power (TxPower: -4, 0, +4 dBm), advertising interval (AdvInterval: 40, 100, 250 ms), connection interval (ConnInterval: 45, 250, 1000 ms), and BLE Disconnected/Connected (Comm_Mode: 0/1=disconnected, 2=connected). Each node operates in four distinct states: State 0 (Idle), State 1 (advertising), State 2 (operation), and State 3 (deep sleep). Before model training and analysis, the collected data underwent systematic preprocessing:

1) **Noise filtering:** Removing transient spikes and measurement artifacts to improve data quality.
2) **Multi-node synchronization**: Measurements from all 3 nodes across 27 configurations were synchronized to a common time base to ensure dataset consistency.
3) **Transient filtering:** The first 4.5 seconds of each measurement session were removed to eliminate initialization artifacts and transient spikes caused by hardware warm-up.
4) **Chronological ordering:** Data were sorted chronologically within each node to preserve temporal dependencies.
5) **State mapping:** Raw sensor states were mapped to the standardized 4-state system, with communication mode tracked separately for State 2 (operation).
6) **Communication chunk identification:** Within State 2, distinct VLC and BLE transmission periods were identified and labeled for fine-grained energy analysis.

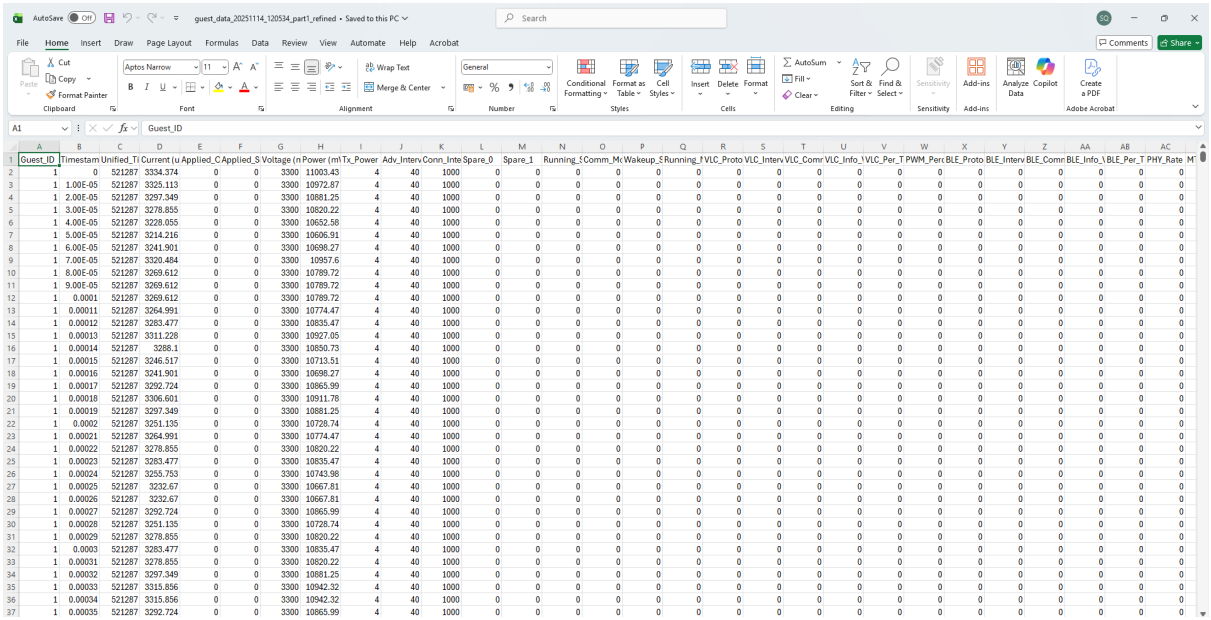


Figure 48. The dataset collected from the actual IoT network.

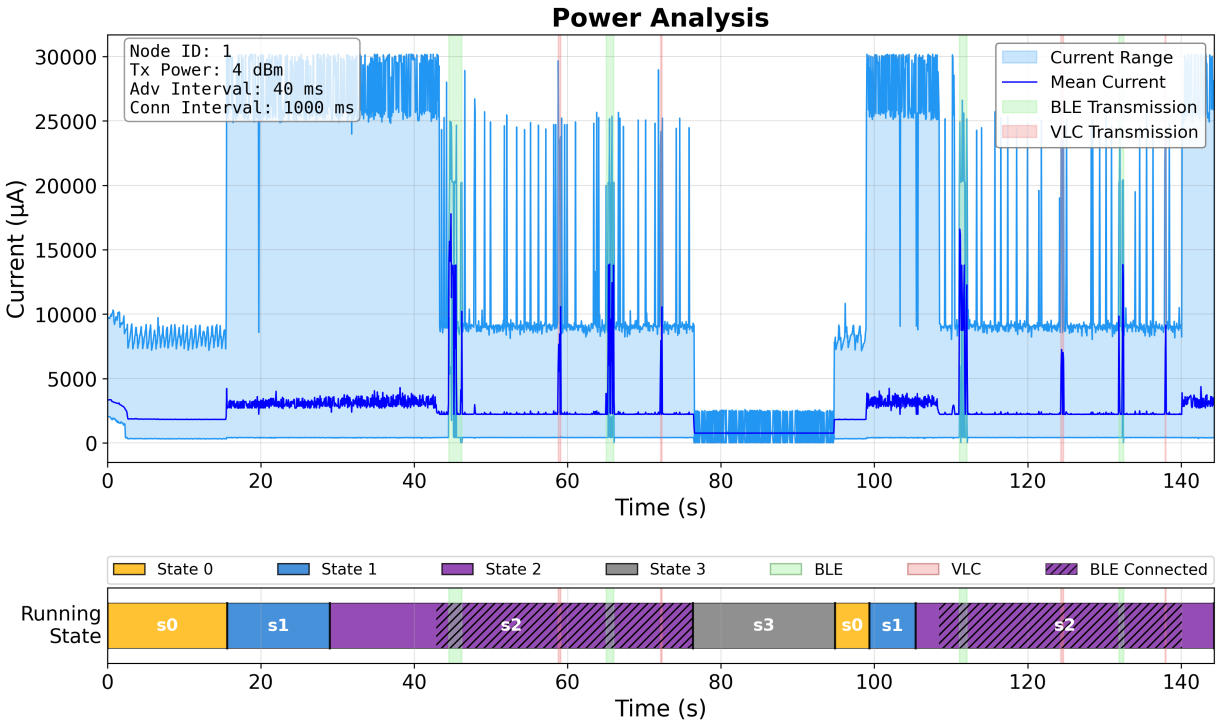



Figure 49. The collected data was systematically preprocessed and visualized.

Figure 49 presents the collected data after systematic preprocessing and visualization. The time-series data are color-coded to represent different system states. It should be noted that the VLC (light red) and BLE duration (light green) can be clearly identified during data transmission using the proposed protocols. This demonstrates that the proposed IoT energy consumption collection system is capable of capturing high-resolution samples at a rate of 100,000 Hz.

To rigorously evaluate model generalization capabilities and robustness, we implemented three complementary Cross-Validation (CV) strategies rather than a single train-test split, enabling comprehensive assessment of: (1) generalization to unseen network configurations, (2) robustness to device-to-device hardware variability, and (3) upper-bound model capacity under ideal conditions (Section 3.12.2).

## 3.12 RIoT Network Energy Consumption Prediction based on AI/ML Models

### 3.12.1 Mechanism of Prediction AI/ML Models

Leveraging the preprocessed dataset, we trained six ML models to predict RIoT network current consumption (µA) under varying configurations. The input feature space comprises 7 features: (1) operational state (0-3), (2) VLC payload size (bytes), (3) BLE payload size (bytes), (4) transmission power in dBm, (5) advertising interval in ms, (6) connection interval in ms, and (7) BLE disconnected/connected. This extended feature set enables the models to capture complex interactions between network parameters and energy consumption patterns. Six model architectures were evaluated for their ability to capture both linear and non-linear relationships:

1) Neural Network (Multi-Layer Perceptron): A compact architecture with two hidden layers of 16 and 8 neurons, respectively, using ReLU activation functions. To prevent overfitting on the relatively small per-fold datasets in CV, we employed strong L2 regularization ($\alpha$=0.5), adaptive learning rate starting at 0.0005, and Adam optimizer with up to 5000 iterations. Early stopping was disabled to ensure consistent training across all CV folds.
2) Random Forest: An ensemble of 100 decision trees with bootstrap aggregation, maximum tree depth of 5, minimum samples per split of 2, and minimum samples per leaf of 1, using all 7 features.
3) Gradient Boosting: Sequential boosting with 100 stages, learning rate of 0.1, and maximum tree depth of 3 to balance model complexity and generalization.
4) Extra Trees: An ensemble of 100 extremely randomized trees with maximum depth of 5, providing additional variance reduction compared to Random Forest.
5) Linear Regression: Ordinary Least Squares regression with standardized input features (zero mean, unit variance), serving as a baseline for linear relationships.
6) Ridge Regression: L2-regularized linear regression with regularization parameter $\alpha$=1.0, applied to standardized features.

For models sensitive to feature scales (Neural Network, Linear Regression, Ridge Regression), input features were standardized using StandardScaler. Model performance was evaluated using three complementary metrics: $R^2$ (coefficient of determination), MAE (Mean Absolute Error in µA), and RMSE (Root Mean Square Error in µA).

### 3.12.2 Cross-Validation Strategies

To comprehensively evaluate model robustness and generalization capabilities, we implemented three distinct CV methods, each testing different aspects of model performance:

1) Method 1: Configuration-Level CV

This method evaluates the model's ability to generalize novel IoT network configurations not encountered during training. Using 5-fold CV on the 27 configurations, each fold withholds approximately 5-6 complete configurations (all 3 nodes, all measurement samples) as the test set, while training on the remaining ~22 configurations. This simulates the realistic scenario where network designers need to predict energy consumption for new parameter settings (e.g., untested TxPower or AdvInterval combinations) without physical measurements. The configurations are randomly shuffled and split into 5 folds using K-Fold CV on configuration IDs. Each test fold contains all data (all nodes, all states, all samples) from the held-out configurations, ensuring complete independence between training and test sets at the configuration level. This is the most challenging validation method, as it tests true parameter-space generalization, a critical capability for practical network optimization tools where engineers must estimate energy consumption for configurations that have never been physically tested.

2) Method 2: Node-Level CV

This method assesses model robustness to device-to-device variability, a common challenge in IoT deployments where manufacturing tolerances, component aging, and environmental factors cause measurable differences between nominally identical devices. For each of the 27 configurations, we perform Leave-One-Node-Out CV, training on 2 nodes, and testing on the 3rd node. This yields 27 configurations × 3 nodes = 81 independent train-test splits. For each configuration *c* and each node *n*, the model trains data from nodes {1,2,3}\{*n*} within configuration *c*, then predicts energy consumption for the held-out node *n* in the same configuration. Performance metrics are averaged across 81 folds. This validates that the model learns configuration-specific patterns that generalize across hardware units rather than memorizing node-specific artifacts (e.g., calibration offsets, sensor drift). High performance in this validation indicates the model is robust for heterogeneous multi-device network deployments where nodes may have individual hardware characteristics.

3) Method 3: Random Split CV

This method provides a baseline for model capacity by performing standard 5-fold CV with random shuffling of all 1,026 samples, regardless of configuration or node identity. This allows the model to train data from the same configurations (and even the same nodes) that appear in the test set, representing the least challenging generalization scenario. All data rows are randomly shuffled and split into 5 folds using K-Fold CV, with 80% training and 20% testing per fold. No constraints are applied to keep configurations or nodes together. This establishes the performance upper bound under ideal conditions, revealing the model's inherent capacity to learn input-output mappings when train-test similarity is maximized (i.e., when independent and identically distributed assumptions hold). The gap between Method 3 and Methods 1-2 quantifies the cost of true generalization to unseen configurations or nodes.

### 3.12.3 Results and Discussion

#### *3.12.3.1 Results and Analysis of ML Models*

Table 12, 13 and 14 summarize the performance of all six ML models using config-level, node-level, and random split CV methods, respectively. Each cell reports the mean metric ± standard deviation across all folds. $R^2$ scores are presented as percentages for clarity. The results indicate that, at the Configuration-Level, Gradient Boosting achieves the best performance ($R^2$ = 92.10%); at the Node-Level, Extra Trees performs best ($R^2$ = 87.50%); and under the Random Split strategy, Gradient Boosting again ranks highest ($R^2$ = 91.48%). Figure 50, Figure 51, and Figure 52 present the $R^2$ performance comparison of ML models. These results should be interpreted in the context of the modelling scope of Section 3.12, namely network-level prediction across multiple devices and configurations, rather than single-device prediction alone, with particular emphasis on generalization to previously unseen network settings.

Table 12. Performance comparison of ML models for predicting RIoT network current consumption using config-level CV strategies.

| Model | $R^2$ (%) | MAE (µA) | RMSE (µA) |
|---|---|---|---|
| **Gradient Boosting** | **92.10 ± 2.21** | **299.6± 26.4** | **603.0** |
| Random Forest | 91.64 ± 2.09 | 322.1± 25.9 | 621.5 |
| Extra Trees | 91.40 ± 2.23 | 347.1 ± 20.4 | 630.1 |
| Neural Network | 76.68 ± 7.78 | 605.7 ± 106.7 | 983.5 |
| Linear Regression | 77.37 ± 6.63 | 636.5 ± 69.3 | 1020.0 |

| | | | |
|---|---|---|---|
| Ridge Regression | 77.37 ± 6.61 | 636.1 ± 69.2 | 1020.1 |

Table 13. Performance comparison of ML models for predicting RIoT network current consumption using node-level CV strategies.

| Model | $R^2$ (%) | MAE (µA) | RMSE (µA) |
|---|---|---|---|
| Gradient Boosting | 84.19 ± 20.46 | 347.1 ± 199.4 | 750.0 |
| Random Forest | 84.87 ± 19.07 | 351.4 ± 194.0 | 739.1 |
| **Extra Trees** | **87.50 ± 13.10** | **357.8 ± 168.3** | **691.6** |
| Neural Network | 69.01 ± 19.70 | 753.3 ± 232.2 | 1159.0 |
| Linear Regression | 75.49 ± 22.89 | 660.9 ± 220.6 | 982.8 |
| Ridge Regression | 76.14 ± 20.91 | 646.2 ± 217.1 | 979.9 |

Table 14. Performance comparison of ML models for predicting RIoT network current consumption using random split CV strategies.

| Model | $R^2$ (%) | MAE (µA) | RMSE (µA) |
|---|---|---|---|
| **Gradient Boosting** | **91.48 ± 0.68** | **296.9 ± 26.4** | **630.5** |
| Random Forest | 90.89 ± 0.70 | 325.7 ± 36.1 | 652.2 |
| Extra Trees | 91.04 ± 0.57 | 343.6 ± 24.1 | 646.3 |
| Neural Network | 75.69 ± 6.46 | 655.3 ± 70.8 | 1058.4 |
| Linear Regression | 76.47 ± 6.61 | 641.6 ± 77.2 | 1040.4 |
| Ridge Regression | 76.47 ± 6.60 | 641.3 ± 77.1 | 1040.4 |

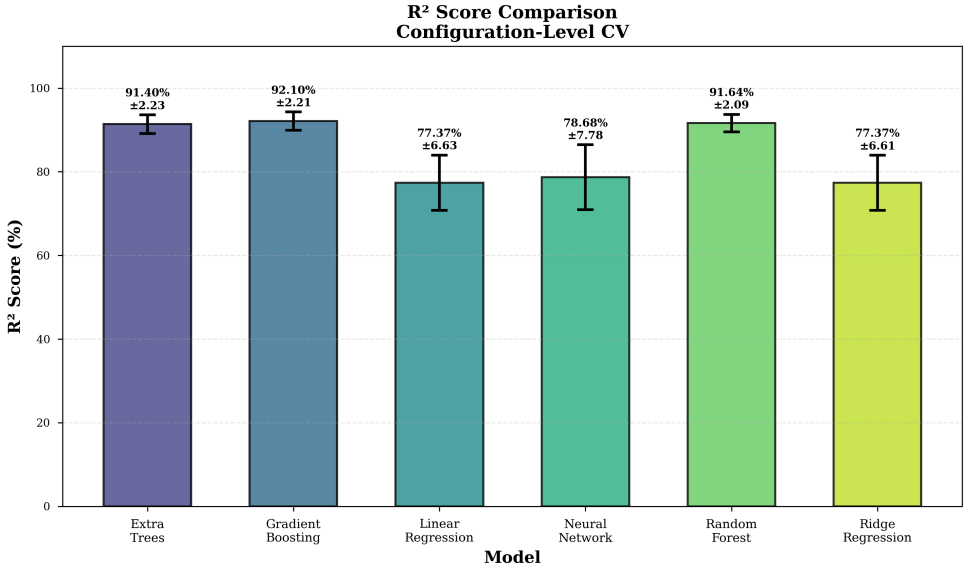


Figure 50. $R^2$ Performance comparison of ML models using configuration-level CV.

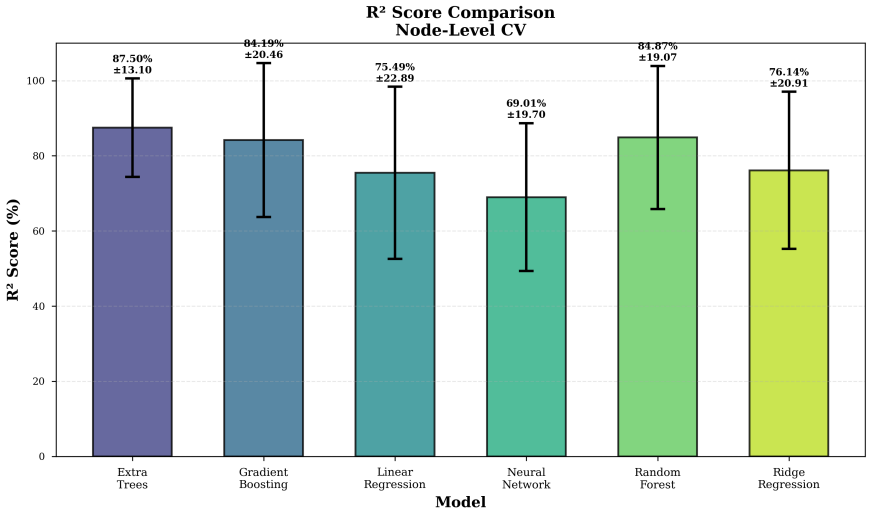



Figure 51. $R^2$ Performance comparison of ML models using node-level CV.

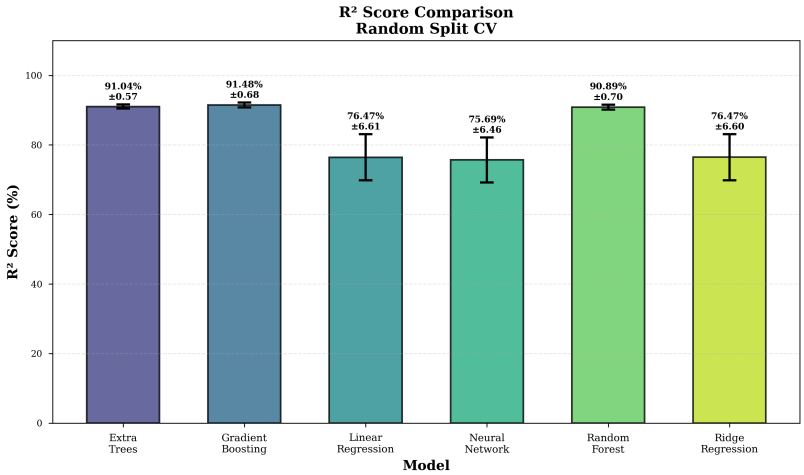



Figure 52. $R^2$ Performance comparison of ML models using random split CV.

The comprehensive three-method CV also reveals several critical findings:

1) Non-linear Models Dominate Across All Scenarios

Ensemble methods (Gradient Boosting, Random Forest, Extra Trees) consistently outperform linear models across all three validation strategies, achieving $R^2$ scores above 91% in Configuration-Level CV (the most challenging scenario) and above 84% in Node-Level CV. This confirms the strongly non-linear nature of the relationship between network parameters (TxPower, advertising/connection intervals, payloads, communication mode) and current consumption. Linear models plateau at ~76% $R^2$, indicating that simple additive combinations of the 7 input features cannot adequately capture energy dynamics.

2) Configuration-Level CV: Successful Extrapolation to Unseen Parameter Combinations

The minimal performance gap between Configuration-Level CV (92.10% for Gradient Boosting) and Random Split CV (91.48%) demonstrates that ensemble models successfully extrapolate to completely unseen network configurations. The average MAE of ~299 µA represents only ~5-10% error relative to typical operational currents (ranging from ~1000-6000 µA across states), confirming practical utility for predicting energy consumption of untested parameter combinations without physical prototyping.

3) Node-Level CV: Robustness to Hardware Variability with Higher Uncertainty

Node-Level CV shows larger performance variance (e.g., Extra Trees: 87.50% ± 13.10%) compared to other methods, reflecting real-world device-to-device heterogeneity. The higher standard deviations (e.g., MAE=357.8 ± 168.3 µA) indicate that some node-configuration pairs exhibit significant hardware-specific behavior that is difficult to predict from other nodes. Nevertheless, $R^2$ scores above 84% for tree-based ensembles demonstrate that models capture the dominant configuration-level trends while tolerating hardware variations, making them suitable for multi-device IoT deployments.

4) Neural Network Underperformance: Small-Data Regime Challenges

The Neural Network model underperforms across all methods (69.01-78.68% $R^2$), particularly in Node-Level CV where training sets are smallest (~25-30 samples per fold for some configurations). Despite strong regularization (α=0.5, small learning rate, compact architecture), the model struggles with the small-sample, high-variability regime characteristic of Leave-One-Node-Out splits. This highlights that ensemble methods are more robust for limited IoT datasets where collecting exhaustive training data is expensive.

5) Model Selection for Deployment

Based on comprehensive evaluation, Gradient Boosting emerged as the best overall model:

- Highest Configuration-Level CV (92.10%): Best for predicting unseen network designs
- Highest Full-Dataset Performance (93.70%, MAE=260.2 µA): Best absolute accuracy
- Competitive Node-Level CV (84.19%): Adequate for multi-device deployments
- Low computational cost: Faster inference than Neural Networks

#### *3.12.3.2 Relationship to Analytical Prediction Tools*

The ML-based prediction framework presented in this section should be interpreted as a complement to, rather than a replacement for, the analytical prediction tools developed earlier in this deliverable. In particular, the previously developed node-level Energy Consumption Prediction App (see Figure 19) and the mini-lamp GW and BBB-based AP-level prediction models (see Section 3.8 and Section 3.9) remain the baseline methods for estimating the energy consumption of a single RIoT node and a single AP/GW, respectively, under user-defined operating scenarios and configuration settings. Their main strengths are physical interpretability, state-wise traceability to measured current profiles, and high single-device accuracy.

By contrast, the purpose of the ML framework in Section 3.12 is different. Rather than predicting one node or one AP at a time, it is designed to predict network-level energy consumption across multiple devices and across a broader configuration space. This framework is trained using a synchronized real-world dataset collected from the actual RIoT network, comprising 27 distinct configurations, 81 experimental trials, and 1,026 samples, and captures the joint influence of both node-level and network-level variables, including operational state, VLC payload, BLE payload, transmit power, advertising interval, connection interval, and communication mode.

Accordingly, a direct one-to-one comparison between the analytical tools and the present ML model is not entirely appropriate, since the two approaches serve different modelling scopes. The analytical tools are best suited for accurate and transparent evaluation of user-defined single-device scenarios, where the operating states, durations, and configuration parameters of one node or one AP are specified explicitly. The ML model, on the other hand, is best suited for rapid prediction across previously unmeasured network configurations, where manually enumerating all interactions among multiple parameters and devices would be cumbersome. In this sense, the ML framework adds value by extending the earlier single-device modelling work to the network level and by enabling scalable scenario analysis and optimization over a large multi-dimensional design space. The complementary roles of the two approaches are summarized in Table 15.

This added value is supported by the validation results. In particular, the configuration-level cross-validation explicitly evaluates generalization to unseen network configurations, while the measured-versus-predicted visualizations in Figure 54–Figure 56 show that the model

reproduces the main current-consumption trends observed in the measured dataset. Gradient Boosting achieved the best overall performance, with 92.10% $R^2$ at the configuration level and 91.48% $R^2$ under random-split validation, confirming that the proposed ML framework is sufficiently accurate for network-level design exploration and scenario screening

Table 15: Complementary roles of the analytical and ML-based energy prediction approaches.

| **Aspect** | **Analytical node / AP tools** | **ML-based network-level framework** |
|---|---|---|
| **Scope** | One node or one AP per evaluation, with user-defined scenario/configuration | Multiple nodes and APs at the network level |
| **Basis** | Analytical / measurement-informed device models | Data-driven model trained on synchronized measured network dataset |
| **Main inputs** | User-defined sequence of states/functions, durations, and protocol/configuration parameters | State, payloads, TX power, advertising interval, connection interval, communication mode |
| **Main strength** | High physical interpretability, state-wise traceability, and single-device accuracy | Fast prediction across large nonlinear multi-device configuration spaces |
| **Best use case** | Detailed evaluation of a custom scenario for one node or one AP | Network design exploration, unseen-configuration prediction, scenario testing, and optimization |
| **Limitation** | Does not directly model synchronized multi-device/network-level interactions in one evaluation | Lower physical interpretability than analytical equations |

### *3.12.3.3 Visualization and Analysis of IoT Networks Energy Consumption*

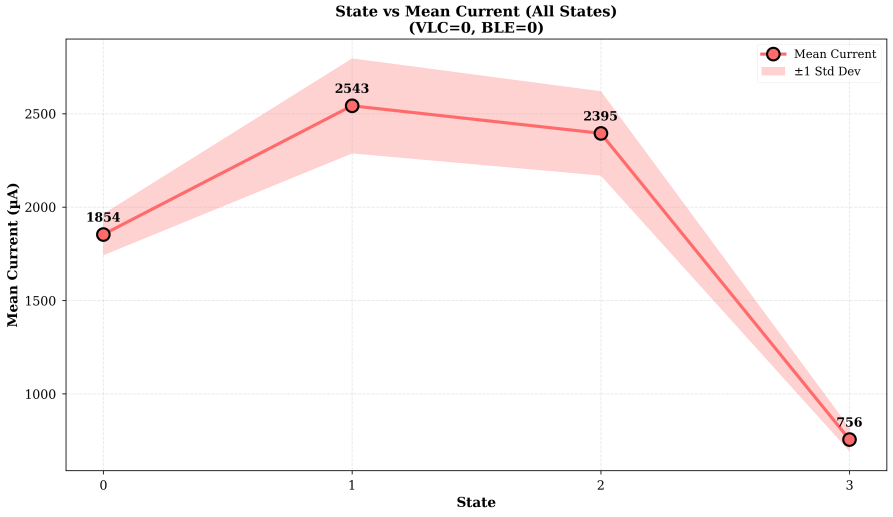
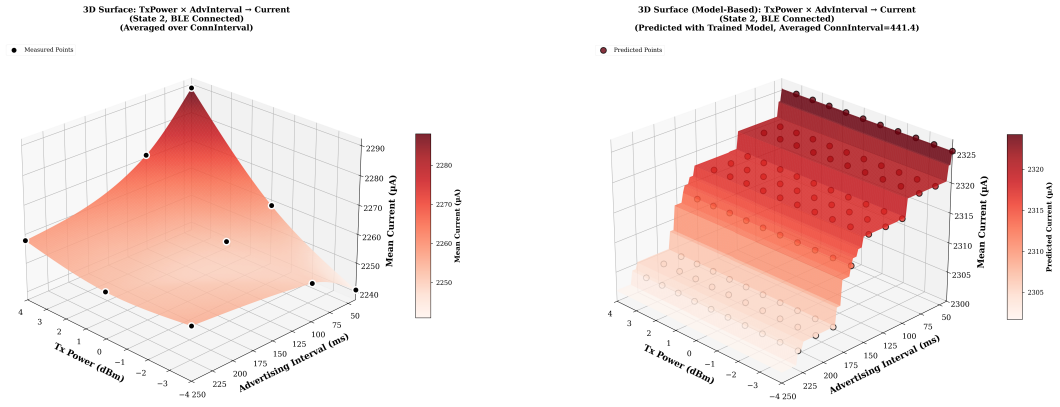


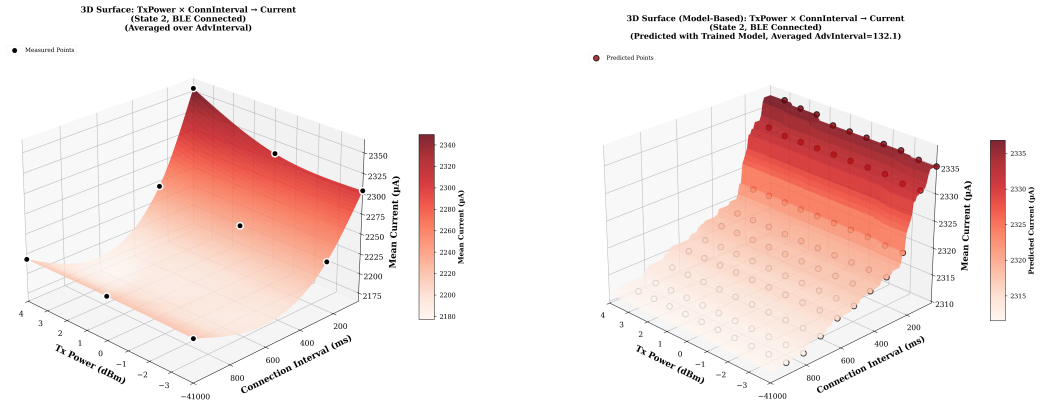

Figure 53. Relationship between operational state and mean current consumption of all IoT nodes.

Figure 53 shows the relationship between operational states and the mean current consumption of all IoT nodes. At State 0, all nodes are idle and perform no operations, consuming the second-lowest current (1854 µA). When the nodes begin fast advertising and transition to State 1, the current consumption increases sharply to the highest level (2543 µA). As the nodes switch to slow advertising or establish BLE connections, the mean current decreases by approximately 5.8%, reaching the second-highest level (2395 µA). Finally, when the nodes enter the deep sleep state, they exhibit the lowest current consumption (756 µA).



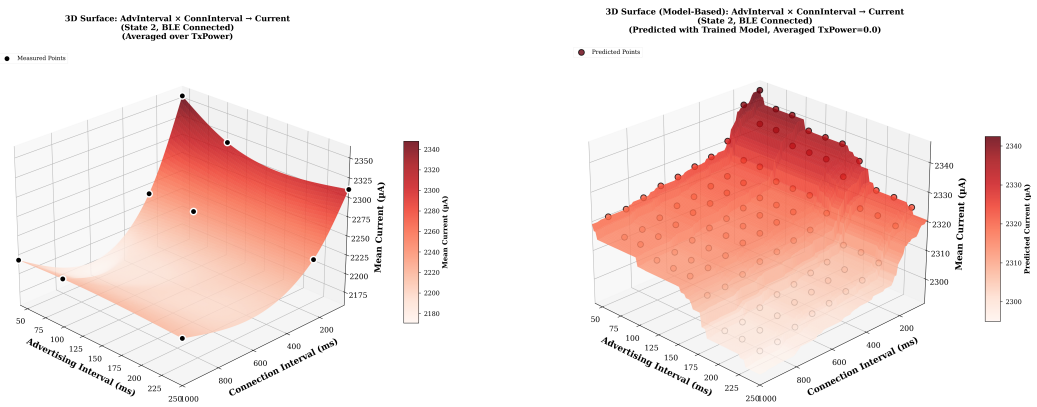

(a) Rendering based on measured data (b) Rendering based on predicted data using the trained model

Figure 54. Relationship between Tx power and advertising interval.




(a) Rendering based on measured data (b) Rendering based on predicted data using the trained model

Figure 55. Relationship between Tx power and connection interval.




(a) Rendering based on measured data (b) Rendering based on predicted data using the trained model

Figure 56. Relationship between advertising intervals and connection intervals.

Figure 54, Figure 55 and Figure 56 illustrate the relationships between transmission power and advertising interval, transmission power and connection interval, and advertising interval and connection interval, respectively, when the nodes are in State 2 and connected via BLE. In each figure, the left subfigure presents the current rendered from measured data extracted from the dataset, while the right subfigure shows the current rendered from predicted data generated using the trained model. It can be observed that the measured and predicted plots exhibit highly consistent current variation trends. Specifically, when the transmission power increases and the advertising or connection interval decreases, the current consumption becomes higher, as shown in Figure 54 and Figure 55. Similarly, when both the advertising and connection intervals are shorter, the current consumption also increases, as illustrated in Figure 56. These consistent patterns demonstrate that the trained model can accurately reproduce the current consumption behavior observed in the measured data, confirming its capability to predict unseen operating conditions with high reliability.

Together, Table 12–Table 14 and Figure 54–Figure 56 indicate that the trained model not only achieves high aggregate prediction accuracy when compared to real measurements, but also reproduces the main physical trends observed in the measured network dataset, including behavior under previously unseen operating conditions.

For deployment in the energy optimization tool, all six trained models are made available, with Gradient Boosting as the default recommendation. Network designers can select alternative models (e.g., Extra Trees for better node-level robustness) based on specific deployment scenarios. These findings confirm that advanced ML models can accurately predict network-level energy consumption across diverse, previously unmeasured configurations, reducing reliance on time-consuming physical prototyping by ~95% (only 5% configurations need physical measurement to train the model, then 95% of the parameter space can be predicted). This enables rapid what-if analysis and systematic optimization of large-scale RIoT networks.

### 3.12.4 Practical Use for RIoT Stakeholders

The primary intended users of the developed predictor tool are network designers, deployment planners, system integrators, and project engineers who need to assess candidate RIoT network configurations before physical deployment or demonstration. For these stakeholders, the key benefit of the tool is that it enables rapid comparison of alternative network settings, such as the number of nodes and APs, device state durations, payload sizes, and communication parameters, without requiring repeated manual measurements or full analytical recalculation for each case. In this context, “rapid scenario testing” refers to the ability to perform comparative analysis across multiple candidate deployment configurations and to immediately observe the corresponding predicted total energy, total power, and node current. This makes the tool particularly useful for early-stage design screening, deployment planning, and preparation of energy-aware operating scenarios.

A related use case has already been demonstrated within the SUPERIOT project through the node-level prediction tool, which was used by MPICOSYS, the Demonstrator 1 leader, to evaluate the energy balance of the planned demonstration scenario (see D4.1). In that case, the tool helped the designers understand the available energy budget. That information was used to adapt the demonstration to the available energy budget without the need to build and repeatedly measure. Thus, the tool provided engineers with resource consumption information that resulted in the correct sizing of key storage and harvesting components, including the micro-supercapacitor capacitance and the number of photovoltaic cells, to ensure stable operation of the energy-autonomous batteryless node. This illustrates the broader practical value of such prediction tools for deployment-feasibility assessment as well as for demonstration planning.

To make the trained models accessible and user-friendly, we developed an energy consumption predictor tool, shown in Figure 57. The tool allows users to input custom parameters, including the total number of nodes and mini-lamps, device state durations (idle, advertising, operation, and deep sleep), and payload size, in order to predict and visualize the expected energy consumption of the IoT network. The outputs include total energy (mWh), total power (mW), and node current (mA) across different network states.

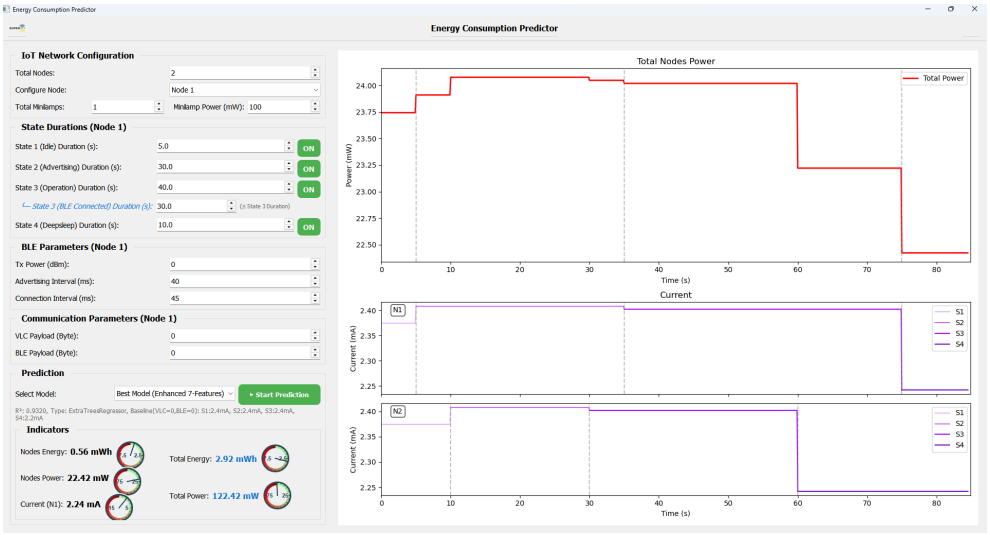



Figure 57. The interface of energy consumption predictor.

## 3.13 ML-Driven RIoT Energy Consumption Optimization Framework

Building upon the energy prediction model developed in Section 3.12, we formulate the RIoT configuration-selection task as a constrained optimization problem over communication and operating parameters. The objective is to identify feasible node configurations that minimize mission energy consumption and, when required, maximize useful communication work under a given energy budget. Unlike manual tuning, the proposed framework systematically explores the admissible configuration space using the trained ML-based current predictor, while preserving the underlying physical and protocol constraints of the node.

### 3.13.1 Formal Problem Statement

Let the mission scenario be characterized by fixed phase durations,

$$\mathbf{T} = \{T_1, T_2, T_3, T_4\},$$

where $T_1$, $T_2$, $T_3$, and $T_4$ denote the durations of initialization, advertising, operation, and sleep, respectively. Let $T_{\text{conn}}$ denote the BLE-connected time available within the operational phase, with

$$0 \leq T_{\text{conn}} \leq T_3. \tag{3.35}$$

Let the decision vector be

$$\mathbf{x} = [P_{\text{tx}} \quad I_{\text{adv}} \quad I_{\text{conn}} \quad B_{\text{BLE}} \quad B_{\text{VLC}} \quad m]^{\text{T}}, \tag{3.36}$$

where $P_{\text{tx}}$ is the BLE transmit power, $I_{\text{adv}}$ is the BLE advertising interval, $I_{\text{conn}}$ is the BLE connection interval, $B_{\text{BLE}}$ is the BLE payload size, $B_{\text{VLC}}$ is the VLC payload size, and $m \in$ {disconnected,connected} is the BLE communication mode.

The admissible domain is

$$\mathbf{x} \in \mathcal{X} = \mathcal{P} \times \mathcal{A} \times \mathcal{C} \times \mathcal{B}_{\text{BLE}} \times \mathcal{B}_{\text{VLC}} \times \mathcal{M}, \tag{3.37}$$

where $\mathcal{P}$, $\mathcal{A}$, $\mathcal{C}$, $\mathcal{B}_{\text{BLE}}$, and $\mathcal{B}_{\text{VLC}}$ are either discrete sets supported by measurements or bounded intervals explored by the optimization algorithm, and $\mathcal{M} =$ {disconnected,connected}.

The trained ML model from Section 3.12 provides an estimate of the average current for each operating sub-state,

$$\hat{I}_k\big(\boldsymbol{\phi}_k(\mathbf{x})\big), \quad k \in \mathcal{K}, \tag{3.38}$$

where $\mathcal{K}$ is the set of modeled sub-operations and $\boldsymbol{\phi}_k(\mathbf{x})$ is the corresponding feature vector. In the present framework, $\mathcal{K}$ may include initialization, BLE advertising, disconnected idle, connected idle, BLE payload transmission, VLC payload transmission, and sleep.

The total mission energy of a single RIoT node is then written as

$$E_{\text{tot}}(\mathbf{x}) = V_{\text{out}} \sum_{k \in \mathcal{K}} \hat{I}_k\left(\boldsymbol{\phi}_k(\mathbf{x})\right) \tau_k(\mathbf{x}), \tag{3.39}$$

where $V_{\text{out}}$ is the node operating voltage and $\tau_k(\mathbf{x})$ is the duration assigned to sub-operation $k$. Note, that equation (3.39) implies a stabilized supply voltage (corresponds to the SUPERIOT RIoT node design); by introducing time-dependent voltage variation the problem can be extended to the scenarios without voltage stabilization.

This formulation expresses the optimization directly in terms of admissible operating durations and payload-dependent communication times, while allowing the ML predictor to estimate the corresponding current consumption.

### 3.13.2 Duration Relations and Protocol Constraints

The communication-related durations are determined from the payload sizes and the experimentally obtained transmission-time coefficients. Let $\kappa_{\text{BLE}}$ and $\kappa_{\text{VLC}}$ denote the average BLE and VLC transmission times per byte of application payload, respectively. Then,

$$\tau_{\text{BLE,tx}} = \kappa_{\text{BLE}}\, B_{\text{BLE}}, \qquad \tau_{\text{VLC,tx}} = \kappa_{\text{VLC}}\, B_{\text{VLC}}. \tag{3.40}$$

The first protocol constraint arises from BLE connectivity. Since BLE payload transfer is only possible during BLE-connected operation,

$$B_{\text{BLE}} = 0 \quad \text{if } m = \text{disconnected}, \tag{3.41}$$

and, when $m =$ connected,

$$\tau_{\text{BLE,tx}} \le T_{\text{conn}}. \tag{3.42}$$

For VLC, payload transmission is not necessarily tied to BLE connectivity. Instead, it must fit within the total VLC-available mission time. Denoting the aggregate optical transmission time allowed by the scenario by $T_{\text{VLC}}^{\max}$, the corresponding constraint is

$$\tau_{\text{VLC,tx}} \le T_{\text{VLC}}^{\max}, \tag{3.43}$$

where $T_{\text{VLC}}^{\max}$ may include any combination of disconnected, advertising, or connected intervals in which optical transmission is physically supported.

The decision variables are also bounded by the experimentally supported ranges:

$$\begin{aligned} &P_{\text{tx}}^{\min} \le P_{\text{tx}} \le P_{\text{tx}}^{\max}, \\ &I_{\text{adv}}^{\min} \le I_{\text{adv}} \le I_{\text{adv}}^{\max}, \\ &I_{\text{conn}}^{\min} \le I_{\text{conn}} \le I_{\text{conn}}^{\max}, \\ &0 \le B_{\text{BLE}} \le B_{\text{BLE}}^{\max}, \qquad 0 \le B_{\text{VLC}} \le B_{\text{VLC}}^{\max}. \end{aligned} \tag{3.44}$$

These bounds ensure that the optimizer remains within the region supported by measurements and, therefore, within the validity range of the trained predictor.

### 3.13.3 Mission-Level Energy Budget

The optimization App incorporates a mission-level energy budget constraint derived from the usable initial supercapacitor energy and the average harvested energy over the mission duration. This provides a practical first-order feasibility check for energy-harvesting RIoT operation.

For a supercapacitor of capacitance $C_{\mathrm{sc}}$, with starting voltage $V_{\mathrm{start}}$ and minimum allowable voltage $V_{\mathrm{min}}$, the usable stored energy is

$$E_{\mathrm{cap}} = \frac{1}{2} C_{\mathrm{sc}} (V_{\mathrm{start}}^2 - V_{\mathrm{min}}^2). \tag{3.45}$$

If optical harvesting is used, and the OPV power density is $\rho_{\mathrm{OPV}}$ over area $A_{\mathrm{OPV}}$ with harvesting efficiency $\eta_h$, then the total mission time is

$$T_{\mathrm{mis}} = T_1 + T_2 + T_3 + T_4, \tag{3.46}$$

and the harvested energy is

$$E_{\mathrm{harv}} = \eta_h \, \rho_{\mathrm{OPV}} \, A_{\mathrm{OPV}} \, T_{\mathrm{mis}}. \tag{3.47}$$

Accordingly, the total available mission energy is

$$E_{\mathrm{bud}} = E_{\mathrm{cap}} + E_{\mathrm{harv}}. \tag{3.48}$$

When budget mode is enforced as a hard feasibility condition, the mission-energy constraint is

$$E_{\mathrm{tot}}(\mathbf{x}) \leq E_{\mathrm{bud}}. \tag{3.49}$$

### 3.13.4 Objective Functions

Three optimization objectives are considered.

#### (a) Minimum-energy formulation

The first objective is pure energy minimization:

$$\mathbf{x}^\star = \arg \min_{\mathbf{x} \in \mathcal{X}} E_{\mathrm{tot}}(\mathbf{x}), \tag{3.50}$$

subject to the feasibility constraints in (3.41)–(3.49), when enabled. This formulation is appropriate when the application requires the most energy-efficient admissible configuration for a fixed scenario.

#### (b) Maximum useful work under energy budget

In some applications, the goal is not only to reduce energy but also to maximize useful communication work. A normalized work metric may be defined as

$$W(\mathbf{x}) = w_{\mathrm{BLE}} \frac{B_{\mathrm{BLE}}}{B_{\mathrm{BLE}}^{\max}} + w_{\mathrm{VLC}} \frac{B_{\mathrm{VLC}}}{B_{\mathrm{VLC}}^{\max}} + w_c \, \mathbf{1}_{\{m=\mathrm{connected}\}}, \tag{3.51}$$

where $w_{\mathrm{BLE}}, w_{\mathrm{VLC}}, w_c$ are non-negative weights and $\mathbf{1}_{\{.\}}$ is the indicator function.

The corresponding optimization problem is

$$\mathbf{x}^\star = \arg \max_{\mathbf{x} \in \mathcal{X}} W(\mathbf{x}), \tag{3.52}$$

subject to

$$E_{\text{tot}}(\mathbf{x}) \leq E_{\text{bud}}, \tag{3.53}$$

together with the feasibility constraints in (3.41)–(3.44). This formulation is relevant when the system must deliver as much communication utility as possible within a fixed energy envelope.

**(c) Balanced Scalarized Formulation**

To jointly favor high work and low energy, a balanced scalarized score can be used. Let

$$\bar{W}(\mathbf{x}) = \frac{W(\mathbf{x})}{W_{\max}}, \tag{3.54}$$

and define the normalized energy term as

$$\bar{E}(\mathbf{x}) = \begin{cases} \dfrac{E_{\text{tot}}(\mathbf{x})}{E_{\text{bud}}}, & \text{if budget mode is enabled,} \\ \dfrac{E_{\text{tot}}(\mathbf{x})}{E_{\text{ref}}}, & \text{otherwise,} \end{cases} \tag{3.55}$$

where $W_{\max}$ is the maximum achievable work score under the selected weights and $E_{\text{ref}}$ is a reference energy derived from the admissible search space when no explicit budget is enforced. The balanced objective is then

$$J(\mathbf{x}) = \lambda_W \bar{W}(\mathbf{x}) - \lambda_E \bar{E}(\mathbf{x}), \tag{3.56}$$

with $\lambda_W > 0$ and $\lambda_E > 0$. The optimization problem becomes

$$\mathbf{x}^\star = \arg\max_{\mathbf{x}\in\mathcal{X}} J(\mathbf{x}). \tag{3.57}$$

When the energy budget is enabled, infeasible points are penalized by

$$J(\mathbf{x}) = -M - \lambda_V (E_{\text{tot}}(\mathbf{x}) - E_{\text{bud}}), \text{ if } E_{\text{tot}}(\mathbf{x}) > E_{\text{bud}}, \tag{3.58}$$

where $M$ is a large positive constant and $\lambda_V > 0$ is a violation-penalty coefficient. This balanced formulation is used as the default mode because it avoids trivial low-energy solutions that perform little useful work, while still discouraging unnecessarily energy-expensive configurations.

### 3.13.5 Search Methods

Let the optimization criterion be written generically as

$$F(\mathbf{x}) = \begin{cases} E_{\text{tot}}(\mathbf{x}), & \text{for energy minimization,} \\ -W(\mathbf{x}), & \text{for work maximization,} \\ -J(\mathbf{x}), & \text{for balanced-score maximization,} \end{cases} \tag{3.59}$$

so that the optimization problem can be expressed uniformly as

$$\mathbf{x}^\star = \arg\min_{\mathbf{x}\in\mathcal{X}_{\text{feas}}} F(\mathbf{x}), \tag{3.60}$$

where $\mathcal{X}_{\text{feas}} \subseteq \mathcal{X}$ denotes the feasible set satisfying all operational and energy constraints.

**(a) Grid search**

In grid search, each decision variable is discretized into a finite set of candidate values,

$$\mathcal{P}_d, \mathcal{A}_d, \mathcal{C}_d, \mathcal{B}_{\text{BLE},d}, \mathcal{B}_{\text{VLC},d}, \mathcal{M}, \tag{3.61}$$

which defines the discrete search space

$$\mathcal{X}_{\text{grid}} = \mathcal{P}_d \times \mathcal{A}_d \times \mathcal{C}_d \times \mathcal{B}_{\text{BLE},d} \times \mathcal{B}_{\text{VLC},d} \times \mathcal{M}. \tag{3.62}$$

The optimal grid-search solution is therefore

$$\mathbf{x}^{\star}_{\text{grid}} = \arg \min_{\mathbf{x} \in \mathcal{X}_{\text{grid}} \cap \mathcal{X}_{\text{feas}}} F(\mathbf{x}). \tag{3.63}$$

If the number of candidate values for each variable is $N_P, N_A, N_C, N_B^{\text{BLE}}, N_B^{\text{VLC}}, N_M$, the total number of evaluated configurations is

$$N_{\text{grid}} = N_P N_A N_C N_B^{\text{BLE}} N_B^{\text{VLC}} N_M. \tag{3.64}$$

This method is exhaustive over the chosen discretization, but its computational cost grows rapidly with the number of parameters and sampling levels.

**(b) Random search**

In random search, candidate configurations are sampled randomly from the admissible domain. Denoting the $i$-th sampled candidate by $\mathbf{x}^{(i)}$, with

$$\mathbf{x}^{(i)} \sim \mathcal{U}(\mathcal{X}), i = 1,2, \dots, N_{\text{rand}}, \tag{3.65}$$

the best random-search solution after $N_{\text{rand}}$ valid samples is

$$\mathbf{x}^{\star}_{\text{rand}} = \arg \min_{\mathbf{x}^{(i)} \in \mathcal{X}_{\text{feas}}} F(\mathbf{x}^{(i)}). \tag{3.66}$$

Equivalently, if $\mathcal{S}_{\text{rand}}$ denotes the set of feasible sampled points,

$$\mathcal{S}_{\text{rand}} = \left\{ \mathbf{x}^{(i)} \in \mathcal{X}_{\text{feas}} \right\}_{i=1}^{N_{\text{rand}}}, \tag{3.67}$$

then

$$\mathbf{x}^{\star}_{\text{rand}} = \arg \min_{\mathbf{x} \in \mathcal{S}_{\text{rand}}} F(\mathbf{x}). \tag{3.68}$$

Random search does not guarantee exhaustive coverage, but it can be much more efficient than grid search when the dimensionality of the decision space is high.

**(c) Differential Evolution**

Differential Evolution is a population-based evolutionary optimization method. Let the population at iteration $g$ be

$$\mathcal{P}^{(g)} = \left\{ \mathbf{x}_1^{(g)}, \mathbf{x}_2^{(g)}, \dots, \mathbf{x}_{N_p}^{(g)} \right\}, \tag{3.69}$$

where $N_p$ is the population size. For each target vector $\mathbf{x}_i^{(g)}$, a mutant vector is generated as

$$\mathbf{v}_i^{(g)} = \mathbf{x}_{r_1}^{(g)} + \beta \left( \mathbf{x}_{r_2}^{(g)} - \mathbf{x}_{r_3}^{(g)} \right), \tag{3.70}$$

where $r_1, r_2, r_3$ are distinct indices chosen randomly from the population and $\beta \in (0,2]$ is the mutation factor. A trial vector $\mathbf{u}_i^{(g)}$ is then produced by crossover between the target and mutant vectors:

$$u_{i,j}^{(g)} = \begin{cases} v_{i,j}^{(g)}, & \text{if } r_{i,j} \le C_r \text{ or } j = j_{\text{rand}}, \\ x_{i,j}^{(g)}, & \text{otherwise}, \end{cases} \tag{3.71}$$

where $C_r \in [0,1]$ is the crossover probability, $r_{i,j} \sim \mathcal{U}(0,1)$, and $j_{\text{rand}}$ ensures that at least one component is inherited from the mutant vector. The selection step is

$$\mathbf{x}_i^{(g+1)} = \begin{cases} \mathbf{u}_i^{(g)}, & \text{if } \mathbf{u}_i^{(g)} \in \mathcal{X}_{\text{feas}} \text{ and } F(\mathbf{u}_i^{(g)}) \le F(\mathbf{x}_i^{(g)}), \\ \mathbf{x}_i^{(g)}, & \text{otherwise.} \end{cases} \tag{3.72}$$

After $G$ generations, the Differential Evolution solution is

$$\mathbf{x}^{\star}_{\text{Diff\_Evol}} = \arg \min_{\mathbf{x} \in \mathcal{P}^{(G)}} F(\mathbf{x}). \tag{3.73}$$

**(d) Practical interpretation**

All search methods solve the same constrained optimization problem in (3.60). Their difference lies only in how the candidate solutions are explored:

$$\mathbf{x}^{\star}_{\text{method}} = \arg \min_{\mathbf{x} \in \mathcal{S}_{\text{method}} \cap \mathcal{X}_{\text{feas}}} F(\mathbf{x}), \tag{3.74}$$

where $\mathcal{S}_{\text{method}}$ is the subset of configurations visited by the selected search algorithm. In this work, grid search is used when the admissible set is sufficiently small, while random search and Differential Evolution are preferred for larger spaces where exhaustive enumeration becomes impractical.

### 3.13.6 Evaluation Setup

To facilitate practical deployment of the optimization framework, we developed a GUI application that integrates the trained ML models with the implemented search algorithms. As shown in Figure 58, the application allows users to define the mission scenario, specify parameter bounds and state durations, configure the optimization objective and energy-budget settings, and automatically identify the highest-ranked feasible configurations together with their predicted energy consumption and state-wise breakdown. Furthermore, to ensure transparency and reproducibility, the implementation of the proposed optimization frameworks and associated tools is made available through our GitHub repository [11]. The repository contains the source code underpinning the energy prediction tools and optimization workflows described in this deliverable. Readers are referred to [11] for further details and ongoing updates.

The optimization framework was evaluated using the best model (Gradient Boosting) together with Grid Search (Exhaustive). The state durations were set to $T_1 = 2$ s, $T_2 = 5$ s, $T_3 = 15$ s, and $T_4 = 20$ s, with the State 3 BLE connected duration also fixed to $15$ s. The parameter bounds used in the optimization were: $P_{\text{tx}} \in [0,8]$ dBm, $I_{\text{adv}} \in [20,250]$ ms, $I_{\text{conn}} \in [11.25,250]$ ms, $B_{\text{VLC}} \in [23,115]$ bytes, $B_{\text{BLE}} \in [116,580]$ bytes, $m \in \{1,2\}$, with grid resolutions of 3, 5, 5, 5, 5, and 3 values, respectively. This corresponds to $3 \times 5 \times 5 \times 5 \times 5 \times 2 = 3750$ raw candidate configurations, not all of which are valid according to the feasibility rules. In fact, the optimizer reported 1875 valid configurations, which is consistent with the feasibility rules. The optimization formulation used in the experiment was Balanced Score, with BLE weight $= 1.0$, VLC weight $= 1.0$, energy penalty weight $= 0.35$, and connected bonus $= 0.05$. The energy budget was enabled using $C = 38.4$ mF, $V_{\text{start}} = 3.3$ V, $V_{\text{min}} = 1.6$ V, harvesting power of $2245\,\mu$W, and $\eta_h = 1.0$. The resulting available mission energy was $254.23$ mJ, comprising $159.94$ mJ from the supercapacitor and $94.29$ mJ from harvesting.

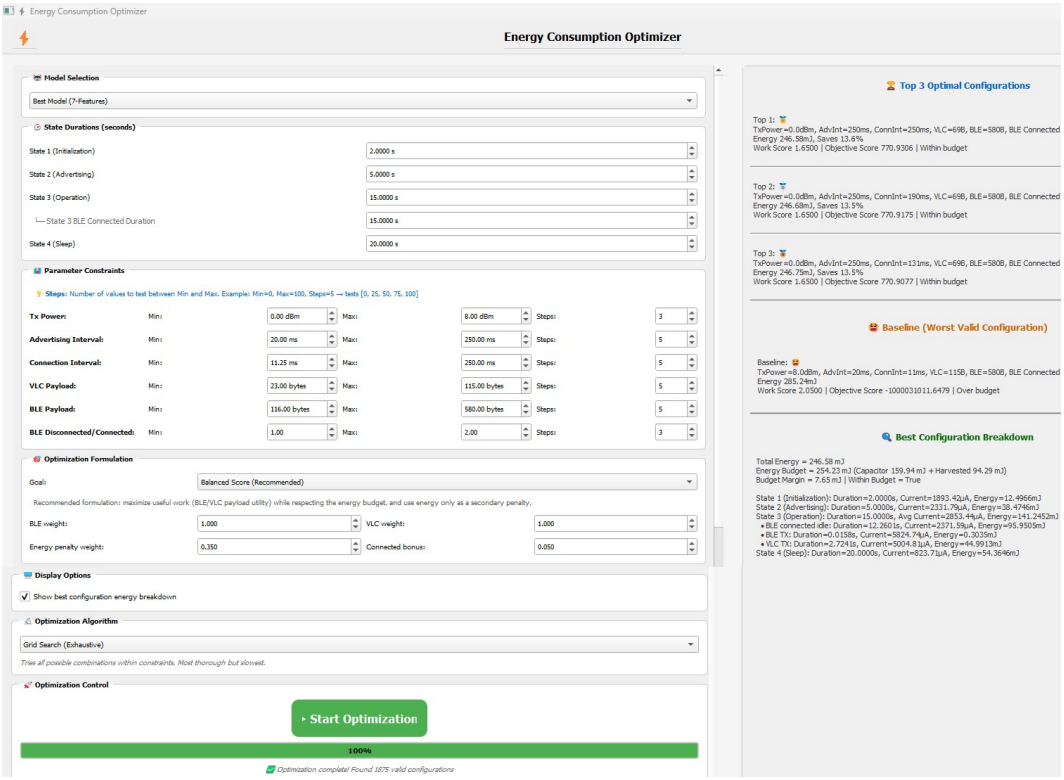



Figure 58. The interface of energy consumption optimizer.

### 3.13.7 Optimization Results

Table 16 summarizes the top three configurations alongside the worst valid configuration identified by the optimizer.

Table 16: Comparison of top and worst valid configurations.

| Rank | $P_{tx}$ (dBm) | $I_{adv}$ (ms) | $I_{conn}$ (ms) | $B_{VLC}$ (Bytes) | $B_{BLE}$ (Bytes) | Energy (mJ) | Work Score | Objective Score | Energy Feasibility |
|---|---|---|---|---|---|---|---|---|---|
| **Top 1** | 0 | 250 | 250 | 69 | 580 | 246.58 | 1.65 | 770.9306 | Within budget |
| **Top 2** | 0 | 250 | 190 | 69 | 580 | 246.68 | 1.65 | 770.9175 | Within budget |
| **Top 3** | 0 | 250 | 131 | 69 | 580 | 246.75 | 1.65 | 770.9077 | Within budget |
| **Worst** | 8 | 20 | 11 | 115 | 580 | 285.24 | 2.05 | $-1\times10^9$ | Over budget |

The top-ranked configuration returned by the optimizer has a predicted total mission energy of 246.58 mJ, useful-work score of 1.65, and balanced objective score of 770.9306. Since $246.58\ \text{mJ} < 254.23\ \text{mJ}$, the solution is feasible, with a remaining energy budget margin of 7.65 mJ. Equivalently, it uses approximately 97.0% of the available energy budget. The results in Table 16 that the optimum solution is not an isolated point, but rather a small cluster of very similar feasible solutions. The worst valid configuration has a predicted energy of 285.24 mJ, with useful-work score 2.05 and objective score of $-1\times10^9$. Since this exceeded the available budget by 31.01 mJ, it was correctly marked as over budget and strongly penalized. Relative to this worst valid case, the best feasible configuration reduces energy by approximately 13.6%. Although the worst configuration achieved a higher useful-work score, it was not energy-feasible under the available mission budget. This demonstrates that the optimizer does not simply maximize useful work;

instead, it prioritizes energy-feasible solutions, selecting the highest-performing configurations that satisfies the mission constraint.

The total energy consumption of the optimal configuration (246.58 mJ) can be decomposed into four operational states, as shown in Table 17. The results show that the operation state (including VLC and BLE payload transmissions) dominates total energy consumption, accounting for more than half of the total (57.3%), followed by sleep (22.0%) and advertising (15.6%). Initialization contributes only a small fraction (5.1%).

Table 17: State-wise energy breakdown.

| State | Description | Energy (mJ) | Contribution (%) |
|---|---|---|---|
| **1** | Initialization | 12.50 | 5.1 |
| **2** | Advertising | 38.47 | 15.6 |
| **3** | Operation | 141.25 | 57.3 |
| **4** | Sleep | 54.37 | 22.0 |

Overall, the results confirm that the proposed optimization framework can systematically identify practical trade-offs between useful work and available mission energy in a transparent and interpretable manner.

## 3.14 Chapter Conclusion

This chapter presented a comprehensive study on the analysis, modelling, and prediction of energy consumption at the network level. We examined the impact of key communication parameters on throughput and energy usage for both BLE and VLC systems. For VLC communication, we proposed an optimized frame structure designed to improve efficiency in command reception and data transmission.

Through energy profiling across diverse operating scenarios—including simultaneous BLE activity—we highlighted the trade-offs between throughput and current consumption. These findings enabled the identification of operating regimes where VLC can function as a low-power yet effective complementary communication channel. Our results showed that a careful combination of optimized BLE parameters, refined VLC frame structures, and targeted gateway/AP energy profiling can significantly enhance both throughput and overall energy efficiency at the network level.

Detailed analysis of the mini-lamp gateway and BBB AP further demonstrated how parameters such as VLC PWM duty cycles and BLE activity shape overall energy consumption. Varying duty cycles revealed clear trade-offs among illumination, communication, and power use, while BLE's scanning, transmitting, and receiving phases introduced measurable variations in current draw.

At the node level, we employed our Energy Consumption Prediction App—built on highly-accurate energy consumption models—to evaluate the impact of different operating configurations. This enabled us to identify conditions under which nodes achieve the optimal balance between energy consumption and harvested energy. Using the Energy Consumption Prediction Tool, we generated time-series datasets across diverse node states and operating modes. These datasets were used to train a TFTM, which consistently outperformed the baseline predictor. The results confirm that AI-driven prediction models are powerful tools for anticipating and optimizing energy usage in dynamic multi-modal communication environments.

In parallel, we developed predictive models for the BBB AP, capable of estimating consumption with 97.5–97.8% accuracy across scenarios with varying PWM duty cycles and BLE activity. These models were integrated into an intuitive GUI that allows users to explore different configurations and operational modes, observing their direct impact on energy consumption.

We also extended our modelling to include representative networking infrastructure components such as Raspberry Pi devices, network switches, and routers. This provided a unified perspective on infrastructure-level energy footprints.

At the synchronized multi-device level, this work further introduced an ML-based network energy prediction framework trained on measured RIoT data collected across multiple configurations and trials. The resulting models captured the joint influence of communication state, payload, BLE parameters, and communication mode, with the best-performing Gradient Boosting model achieving strong predictive accuracy. In this respect, the node-level and AP-level tools remain the preferred approach for transparent and physically interpretable single-device analysis, whereas the ML framework extends prediction capability to broader multi-parameter and previously unmeasured operating scenarios.

Building on this prediction capability, Section 3.13 introduced an ML-driven optimization framework for mission-based configuration selection. Rather than relying on manual parameter tuning, the developed optimizer combines the trained ML predictor with constrained search algorithms to identify feasible communication settings under user-defined state durations, protocol constraints, and an optional mission-level energy budget derived from supercapacitor storage and OPV harvesting. The framework supports multiple objective formulations, including minimum-energy optimization, maximum useful work under an energy budget, and a balanced utility–energy objective. A GUI-based application was developed to make this optimization practical, allowing users to define mission scenarios, parameter bounds, budget settings, and objective weights, and then automatically obtain the highest-ranked feasible configurations together with their predicted energy consumption and state-wise breakdown. The reported results showed that the optimizer can identify configurations that remain close to the available mission energy budget while still preserving useful work, thereby providing a practical and interpretable tool for scenario screening, parameter tuning, and energy-aware deployment planning.

Overall, this chapter establishes a complete end-to-end framework, from empirical measurement to AI-driven prediction and optimization, enabling scalable, energy-aware IoT network design. The integration of machine learning–based prediction with intelligent optimization represents a significant step toward adaptive, sustainable, and self-optimizing IoT infrastructures.

# 4 Communication Modality and Access Technology Configuration Optimization

## 4.1 Overview of Hybrid RF and Optical Wireless Networks and Communication Modalities in RIoT

In traditional Wireless Local Area Networks (WLANs), each node connects to an AP based on factors such as signal strength, interference, and its load demand. Hybrid RF and OWC require more complex AP selection as they need to consider the capabilities and requirements of RF and OWC simultaneously. This modality selection depends on the availability and characteristics of APs in each technology that generalizes the traditional AP assignment problem. The modality selection must account for the availability, capabilities, and characteristics of APs within each modality. Consequently, modality selection becomes a natural extension—and generalization—of the traditional AP assignment problem, requiring simultaneous evaluation of both technologies to determine the optimal connectivity strategy.

Moreover, these hybrid networks support dual-mode devices that a node can connect to both RF and optical APs simultaneously. While this enhances flexibility and performance, it significantly complicates the modality of selection in hybrid RF/OWC networks compared to conventional RF-only systems.

## 4.2 Challenges in Modality Selection

In hybrid RF/OWC networks, modality selection extends beyond conventional AP association, requiring a joint evaluation of various performance, contextual, and environmental factors. These include energy efficiency, achievable data rates, operational and deployment costs, communication security, and application-specific QoS requirements. Additionally, situational factors such as device mobility, user location, and channel conditions must also be taken into account.

In addition, hybrid RF/OWC networks often feature extensive overlapping coverage areas where multiple APs—across both RF and optical domains—are concurrently accessible. These APs typically possess heterogeneous characteristics in terms of range, capacity, reliability, and environmental sensitivity. Additionally, the asymmetric nature of uplink and downlink communication in such hybrid environments introduces another layer of complexity. In other words, APs may exhibit differing characteristics for uplink and downlink transmission—such as bandwidth availability, power constraints, and latency—necessitating separate evaluations for each communication direction. As a result, AP and modality selection in hybrid RF/OWC networks evolves into a complex, multi-dimensional decision problem that demands adaptive and context-aware strategies to ensure optimal and reliable connectivity.

Therefore, the modality selection process must involve a multi-faceted and adaptive decision-making framework that accounts for variations across access modalities (optical vs. radio), link directions (uplink vs. downlink), and network conditions (static vs. dynamic).

This environment inherently involves uncertainty, variability, and non-linear relationships among decision factors. Consequently, a robust decision-making framework must be able to:

- Navigate ambiguity and incomplete information,
- Adaptively weigh multiple criteria,
- Manage trade-offs among competing objectives, and
- Maintain resilience in the presence of fluctuating network dynamics.

To address these multifactorial demands, multi-criteria decision-making (MCDM) techniques, particularly multi-objective optimization (MOO) and MADM methods, have been widely advocated. These approaches offer a structured means to balance competing objectives and

identify optimal or near-optimal configurations. However, encapsulating all relevant performance metrics and constraints into a single MOO formulation dramatically increases computational complexity, particularly for large-scale deployments with many nodes and APs.

## 4.3 Challenges of Handover

The communication modality of each node must adapt dynamically to changing network conditions, user requirements, service-level agreements, and device capabilities. This dynamic adaptation gives rise to the so-called handover problem, which involves switching the communication modality or the associated access point to maintain seamless connectivity and service quality.

In hybrid RF/OWC networks, maintaining uninterrupted connectivity and meeting required QoS often requires that nodes seamlessly perform handovers—not only between access points of the same modality (e.g., RF-to-RF) but also across modalities (e.g., RF-to-OWC and vice versa). This process must account for a dynamic set of factors, including AP capacity, application-specific QoS requirements (e.g., latency or data rate), energy efficiency targets, device orientation (which critically affects optical links), and external environmental conditions such as obstacles or ambient light interference.

Handover management in hybrid RF/OWC networks is significantly more complex than in conventional single-mode wireless systems due to several key challenges:

1. **Heterogeneity of technologies:** RF and OWC have distinct PHY characteristics, medium access mechanisms, and propagation behaviors, requiring technology-aware decision processes.
2. **Small optical coverage areas**: Optical APs typically provide much smaller coverage than RF APs. Even modest node mobility can trigger frequent transitions, increasing the frequency and precision needed for handover decisions.
3. **Overlapping coverage zones**: The simultaneous availability of multiple APs across both modalities expands the handover decision space, complicating the selection of the optimal AP or modality.
4. **Fragility of optical links**: Optical channels are highly sensitive to device alignment, ambient lighting, and reflective surfaces, necessitating a responsive and context-aware handover mechanism.
5. **Asymmetric uplink/downlink scenarios**: Many hybrid deployments decouple uplink and downlink transmissions, potentially requiring different APs or modalities for each direction. For example, an uplink may need to switch from an optical AP to an RF AP due to misalignment, while the downlink remains optical.
6. **Preemptive and precise execution**: To avoid service disruption and QoS degradation, handovers must be executed both accurately and preemptively, accounting for dynamic network conditions and application requirements.

Consequently, handover management in hybrid RF/OWC networks must be both intelligent and adaptive. It should be capable of anticipating contextual changes, processing multi-criteria inputs in real time, and making robust decisions under uncertainty. Moreover, it must be resource-aware, balancing trade-offs between performance, energy consumption, handover frequency, and signaling overhead.

## 4.4 MADM-based Handover

To effectively address the handover challenge in hybrid RF/OWC networks, it is essential to consider multiple performance metrics. MADM techniques are therefore well suited for this context. Moreover, the solution must be both decentralized and computationally efficient. In response to these needs, we propose a novel decentralized MADM framework that overcomes the limitations of existing centralized and single-metric handover strategies in hybrid RF/OWC networks.

Unlike traditional approaches, the proposed scheme evaluates handover decisions based on predicted network conditions over the next $m$ time steps, treating each forecasted state as a separate decision attribute. This forward-looking strategy enables a more holistic evaluation by incorporating both current and anticipated network dynamics.

The decision-making process is guided by three core attributes: handover cost, the ratio of data rate demand to achievable rate, and the ratio of utilized to available capacity. The handover cost itself is further divided into two components: a fixed cost—accounting for baseline energy consumption and inherent network delays—and a variable cost, which discourages unnecessary handovers while still allowing for QoS improvements when justified. To rank candidate APs, the VIKOR method is employed, enabling dynamic evaluation across all $3 \times m$ attributes.

The decentralized nature of the scheme allows any node $n_{\mathrm{i}}$ to make autonomous decisions based on locally available information. At each time step $s_{\mathrm{t}}$, the node identifies the set of accessible APs, denoted $\mathbb{A}_i^{(t)}$, and collects attribute data for each AP over the upcoming m time steps (i.e., $\{s_{\mathrm{t+1}}, \ldots, s_{\mathrm{t+m+1}}\}$). This information can be obtained through piggybacked transmissions or periodic BEACON broadcasts. The VIKOR algorithm is then applied to rank the APs, guiding the node in deciding whether to maintain its current connection or initiate a handover to a more optimal AP.

### 4.4.1 Handover Cost in Hybrid RF/OWC Networks

Handover cost refers to the impact and resources required when a node transitions from one AP to another. In particular, handovers can negatively affect network efficiency by increasing latency, energy consumption, data loss, and resource wastage. Despite these costs, some handovers are necessary and may improve QoS.

We define handover cost with two components: a fixed portion, $\theta_{(i,l)}$, and a variable portion, $\vartheta_{(i,l)}$, both of which quantify the cost for transitioning node $n_i$ to AP $a_l$. Each node can independently evaluate these costs based on its parameters, allowing it to compute the total handover cost autonomously, without requiring communication with APs or other nodes.

- The fixed portion, $\theta_{(i,l)}$, represents a constant cost incurred with every handover associated with factors like the baseline energy required to re-establish connection or inherent delays in network protocols.
- The variable portion discourages unnecessary or frequent handovers, while giving the opportunity for handover in spite of costs to get better QoS. As the impact of the variable portion decreases over time, it imposes a higher initial cost for handovers that occur shortly after a previous one.

The variable portion is particularly important in dynamic environments where nodes may frequently move between coverage areas, as it helps balance the need for maintaining QoS with the cost of transitioning between APs.

The handover cost varies at each of the next $m$ time steps, and at time step $\tau \in [1, m]$, the total handover cost is calculated by:

$$\Theta_{(i,l)}^{(t)} = \theta_{(i,l)} + \frac{\vartheta_{(i,l)}}{t - \varepsilon_i^{(t)} + \tau}, \tag{4.1}$$

where $\varepsilon_i^{(t)}$ denotes the time step that node $n_i$ performed its last handover and established a connection with its current AP.

### 4.4.2 Node Data Rate Demand vs. Achievable Rate

To determine whether a candidate AP can fulfill the data rate demand of a node, we define the ratio of the node's data rate demand to the achievable data rate as follows:

$$\chi_{(i,l)}^{(t)} = \frac{B_i^{(t)}}{B_{(i,l)}^{(t)}}, \tag{4.2}$$

where $B_i^{(t)}$ represents the data rate demand of node $n_i$, and $B_{(i,l)}^{(t)}$ denotes the achievable data rate for node $n_i$ from the candidate AP $a_l$.

The achievable data rate can be derived using the Shannon-Hartley theorem:

$$B_{(i,l)}^{(t)} = \frac{B_l^{max}}{|N_l|}\log_2(1 + \Gamma''^{(t)}_{(i,l)}) \tag{4.3}$$

where $|N_l|$ denotes the number of nodes connected to AP $a_l$, assuming that the AP's total capacity is equally distributed among all connected nodes. In addition, $\Gamma''^{(t)}_{(i,l)}$ represents the received SNR by node $n_i$ from the candidate AP $a_l$.

### 4.4.3 Load Balancing

We incorporate the ratio of the utilized capacity of candidate APs to their maximum capacity into the decision-making process for load balancing across the network. This load-balancing metric is defined as:

$$\rho_l^{(t)} = \frac{\delta_l^{(t)}}{C_l^{max}} \tag{4.4}$$

where $\rho_l^{(t)}$ is the load ratio of candidate AP $a_l$ at time step $s_t$, $\delta_l^{(t)}$ denotes the utilized capacity of AP $a_l$ at time step $s_t$, and $C_l^{max}$ represents the maximum capacity of the AP, which is determined based on the AP's available bandwidth and the prevailing SNR conditions.

### 4.4.4 Algorithm Description

To select the most appropriate AP from the available options, this study applies to the VIKOR method, addressing conflicting criteria, and achieving a compromise solution. The VIKOR ranks alternatives based on their closeness to an ideal solution, determined through a weighted normalized decision matrix and compromise measures.

The proposed scheme considers $3m$ attributes $C_1, \ldots, C_{3m}$. In particular, a candidate AP with lower values for handover cost ($C_1$ through $C_m$), the ratio of data rate demand to achievable data rate ($C_{m+1}$ through $C_{2m}$), and the capacity utilization ratio ($C_{2m+1}$ through $C_{3m}$) is preferable. The steps for applying VIKOR in the handover decision process are as follows:

**Initialization**

The node $n_i$ at each time step $s_t$ collects the values of the considered attributes from all accessible APs and then forms the decision matrix $DM_1$ represented as:

$$DM_1 = \begin{array}{c} \begin{matrix} C_1 & C_2 & \cdots & C_{3m} \end{matrix} \\ \begin{bmatrix} x_{1,1} & x_{1,2} & \cdots & x_{1,3m} \\ x_{2,1} & x_{2,2} & \cdots & x_{2,3m} \\ \vdots & \vdots & x_{l,j} & \vdots \\ x_{|\mathbb{A}_i^{(t)}|,1} & x_{|\mathbb{A}_i^{(t)}|,2} & \cdots & x_{|\mathbb{A}_i^{(t)}|,3m} \end{bmatrix} \end{array} \tag{4.5}$$

where $x_{l,j}$ represents the suitability of candidate AP $a_l \in \mathbb{A}_i^{(t)}$ with respect to attribute $C_j$. We exclude APs that cannot provide sufficient resources from being considered candidates. Therefore, any row with a value greater than 1 in any column from $C_{m+1}$ through $C_{2m}$ will be deleted. This ensures that an AP will only be selected if its achievable data rate meets or exceeds

the node's data rate demand and $\chi_{(i,l)}^{(t)}$ remains lower than 1 for all candidate APs over the next $m$ time steps.

**Normalization**

To ensure all criteria are on a uniform scale and prevent any single criterion from dominating due to differing magnitudes, the decision matrix is normalized. The normalized matrix, referred to as $DM_2$, is calculated as:

$$r_{l,j} = \frac{x_{l,j}}{\max_l x_{l,j} - \min_l x_{l,j}} \tag{4.6}$$

**Weighting the Normalized Decision Matrix**

The significance of each criterion is incorporated by applying weights to the normalized matrix. The weighted normalized decision matrix is denoted as $DM_3 = [v_{l,j}]$, where $v_{l,j} = w_l \times r_{l,j}$.

**Determining the Best and Worst Values**

The best and worst values for each criterion are identified. Since all attributes have negative impact, $f_j^+ = \min_l v_{l,j}$ and $f_j^- = \max_l v_{l,j}$.

**Computing Utility and Regret Measures**

The utility measure $S_l$ and the regret measure $R_l$ for each alternative $a_l$:

$$S_l = \sum_{j=1}^{3m} w_j . \left(\frac{f_j^+ - v_{l,j}}{f_j^+ - f_j^-}\right) \tag{4.7}$$

$$R_l = \max_j \left( w_j . \left(\frac{f_j^+ - v_{l,j}}{f_j^+ - f_j^-}\right)\right) \tag{4.8}$$

**Computing the VIKOR Index**

The VIKOR index $Q_l$ is computed to rank the alternatives:

$$Q_l = v\left(\frac{S_l - S^-}{S^+ - S^-}\right) + (1-v)\left(\frac{R_l - R^-}{R^+ - R^-}\right) \tag{4.9}$$

where $S^+ = \min_l S_l$, $S^- = \max_l S_l$, $R^+ = \min_l R_l$, $R^- = \max_l R_l$, and $v$ is a weight for the strategy of the majority of criteria (usually set to 0.5).

**Ranking the Alternatives**

The candidate AP $a_l$ with the lowest $Q_l$ is considered the best choice.

### 4.4.5 Performance Evaluation of MADM-based handover

We considered two distinct deployment configurations for the optical APs: 4 optical APs arranged in a $2 \times 2$ grid, and 9 optical APs arranged in a $3 \times 3$ grid. or the radio APs, either 2 or 4 are positioned at the corners of the deployment area. In the case of 2 radio APs, they are placed at diagonally opposite corners. Nodes are randomly deployed within the environment and move according to the Random Waypoint (RW) model, with a speed of 0.1 m/s. To ensure the robustness and accuracy of results, we performed Monte Carlo simulations with 100 runs for each configuration and presented the averages across these runs. We define two different values for the fixed handover cost, $\theta_{(i,l)}$. If the node switches between different AP types (e.g., from OWC to RF or vice versa), the cost is set to 200. If the node connects to an AP of the same type,

the cost is 100. Similarly, the variable handover cost, $\vartheta_{(i,l)}$, follows the same logic but with higher values: 400 for vertical handovers (between different AP types) and 200 for horizontal handovers (within the same AP type).

Figure 59 shows the average execution time per node, obtained by dividing the total execution time by the number of nodes. The results indicate a clear trend: execution time increases as both the forecast horizon, $m$, and the total number of APs, $N_O + N_R$, grow. This is consistent with the computational complexity of the scheme, which follows the same pattern as the VIKOR method, where a matrix of size $\left|\mathbb{A}_i^{(t)}\right| 3m$ is processed at each decision step. The overall computational complexity can thus be expressed as $O((N_O + N_R)m)$, where $N_O$ and $N_R$ represent the number of optical and radio APs, respectively. As the forecast horizon $m$ increases, a broader set of future states is considered during the decision-making process, and the execution time extends. This results in more computational resources being required to evaluate handover decisions over a larger time window. In our experiments, we assumed equal attribute weights for decision criteria, i.e., $w_j = \frac{1}{3m}$. Therefore, increasing $m$ also leads to lower weight for the attributes. Simulation results demonstrate that if $m > 10$, the performance of the proposed scheme decreases, particularly in terms of handover ratio, due to overemphasis on distant future states. Therefore, in subsequent experiments, we fix the forecast horizon at $m$=10 to ensure the best trade-off between execution time and system performance.

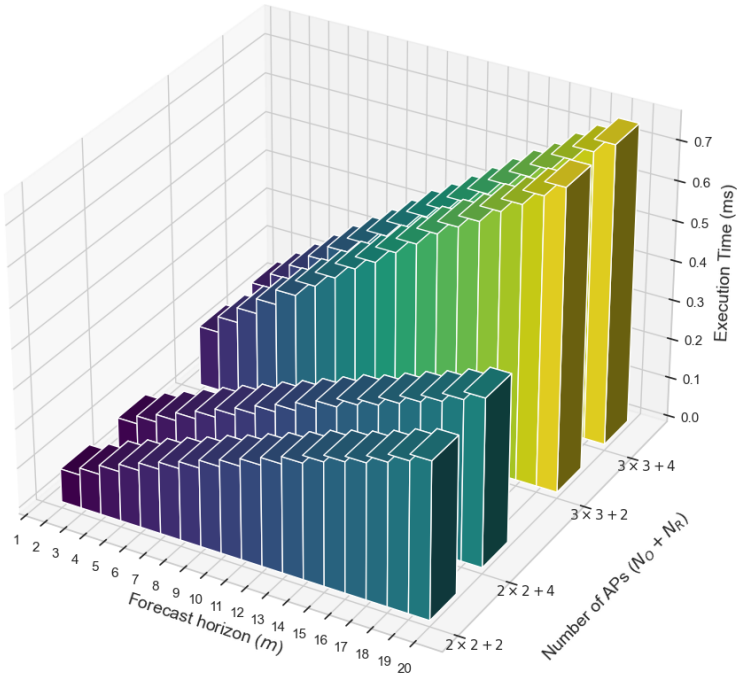
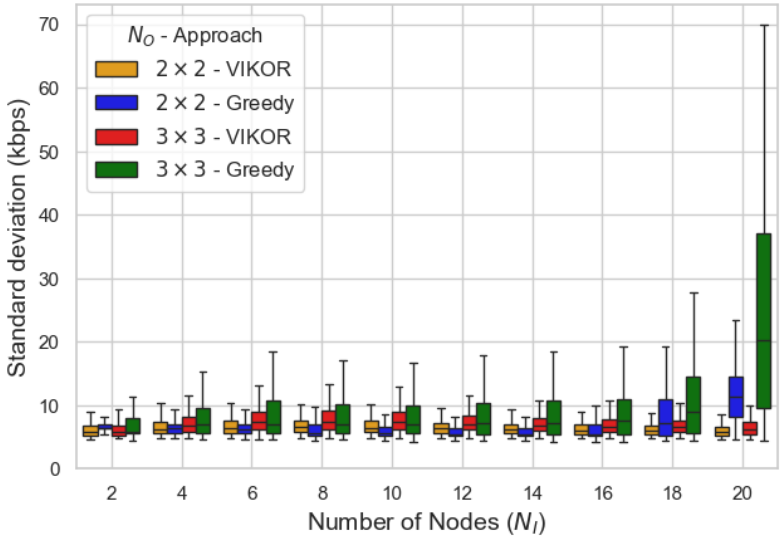
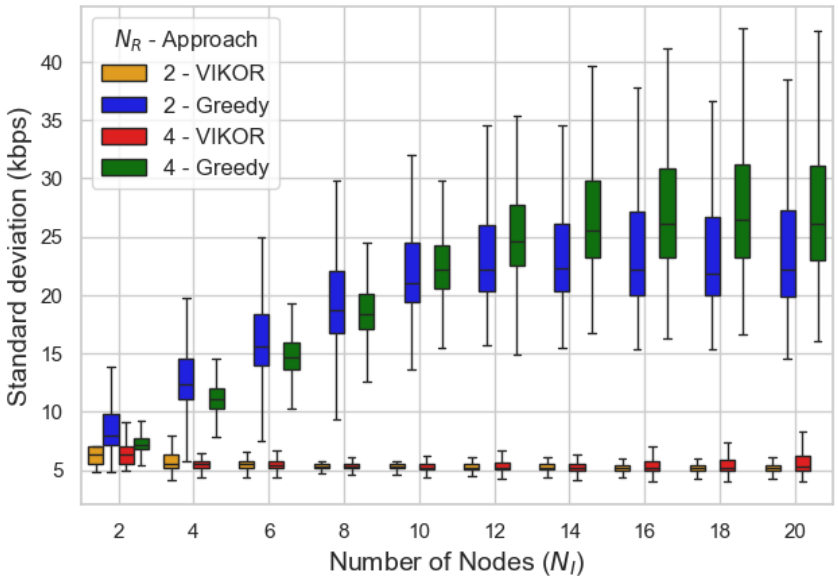



Figure 59. Average execution time per node as a function of m and the total number of APs.

Figure 60 presents a boxplot comparing the standard deviation of remaining capacity across APs. The results show that the proposed approach achieves smaller interquartile ranges, indicating better load distribution compared to the greedy algorithm. As the number of APs increases, variability in remaining capacity also rises, highlighting the challenge of maintaining consistent load distribution with network expansion. Furthermore, increased node count exacerbates remaining capacity variability across APs. Both algorithms show higher variability with more nodes, but the impact is more pronounced in the greedy algorithm. Comparing Figure 60 (a) and (b), we observe that with fewer nodes, optical APs exhibit lower standard deviation in remaining capacity compared to radio APs. Initially, radio APs are more heavily utilized, leading to greater variability. As node count increases, the higher data rate demand shifts more load to optical APs, which, despite having lower variability in low-load scenarios, show increased variability as the network scales.




a) Optical APs




b) Radio APs

Figure 60. Standard deviation of remaining capacity across APs.

Figure 61 illustrates the handover ratio, comparing the number of handovers in the proposed approach to the greedy algorithm. Here, handovers are counted as new connections to APs, tracked separately for optical and radio APs. The results show that the proposed approach leads to more handovers, a trend that becomes more pronounced as the number of nodes increases. This is due to its forward-looking strategy, which anticipates future network conditions and may lead to unnecessary handovers or load balancing decisions. In contrast, the greedy algorithm initiates handovers only when a node's demand cannot be met, resulting in fewer handovers. We can potentially reduce handovers in the proposed approach by adjusting the weights assigned to the attributes, particularly increasing the weight for handover costs. However, this adjustment may impact other metrics and load balancing. Future research will focus on optimizing these weights to improve the proposed algorithm's performance across various metrics, including minimizing unnecessary handovers. Additionally, while the greedy algorithm's reactive handovers may cause temporary connectivity loss, the proposed approach's proactive handovers maintain continuous connection through the handover process, offering a smoother transition and avoiding connectivity issues associated with the greedy algorithm.

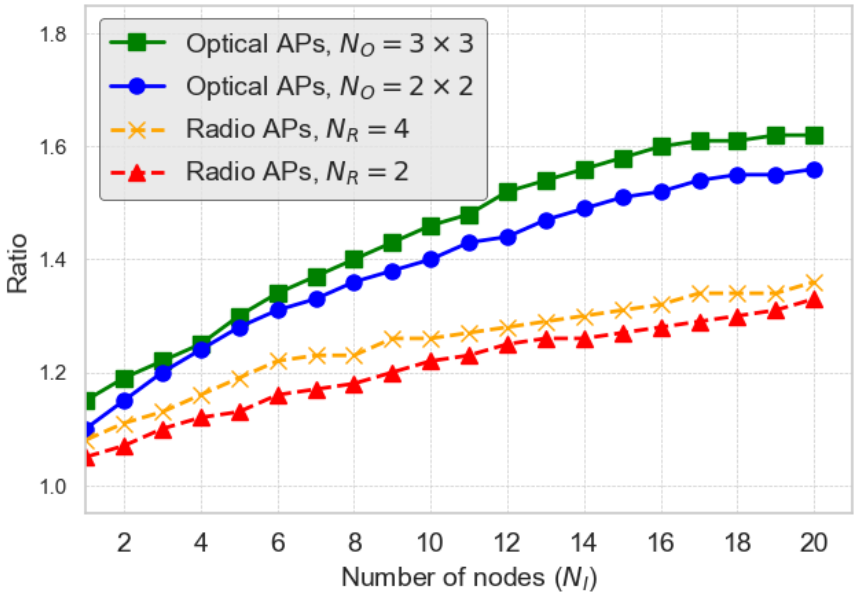



Figure 61. Handover ratio in different APs for varying numbers of nodes.

## 4.5 Joint Transmission Power Allocation and Modality Selection

In a hybrid RF/OWC networks, efficient power management is essential for achieving both high data throughput and energy efficiency. However, the two communication modalities exhibit distinct propagation characteristics and differ in their transmission power requirements and energy consumption patterns. To this end, we formulate three objective functions: one for minimizing transmission power in OWC, another for minimizing transmission power in RF communication, and a third for maximizing the overall achievable data rate. To solve this multi-objective optimization problem, we employ a multi-criteria reinforcement learning framework that dynamically determines the optimal transmission power for each modality and selects the best AP association within the hybrid OWC/RF network.

### 4.5.1 Optimization Objectives

#### *4.5.1.1 Transmission Power*

The total transmission power of all nodes at time step $t$ for OWC and RF communications is given by the following two formulas:

$$z^1(t) = \sum_{i \in I} \left[ p_i^n(t) \sum_{n \in N^O} b_i^n(t) \sum_{c \in C^O} d_i^{(n,c)}(t) \right] \tag{4.10}$$

$$z^2(t) = \sum_{i \in I} \left[ p'^n_i(t) \sum_{n \in N^R} b_i^n(t) \sum_{c \in C^R} d_i^{(n,c)}(t) \right], \tag{4.11}$$

where the binary variable $b_i^n(t)$ represents the association between node $i$ and AP $n$ at time step $t$, and $d_i^{(n,c)}(t)$ denotes the channel allocation for APs. Specifically, $b_i^n(t) = 1$ indicates that node $i$ is connected to the $n$th AP at time $t$; otherwise, $b_i^n(t) = 0$. The variables $p_i^n(t)$and $p_i'^n(t)$represent the transmission power levels of node $i$ for optical and radio communication, respectively. The sets $N^O$ and $N^R$ denote the optical and radio APs, while $C^O$ and $C^R$ represent the numbers of available channels in the optical and radio domains, respectively. All APs are assumed to operate over a set of orthogonal channels.

#### *4.5.1.2 Data Rate Throughput*

Operating in a dual-mode configuration enables nodes to communicate simultaneously with both optical and radio APs, thereby enhancing communication efficiency and reliability. The total achievable data rate of all nodes at time step $t$ across both optical and radio communication channels is expressed as

$$z^3(t) = \sum_{i \in I} \sum_{n \in N^R} \sum_{c \in C^R} R_i^{(n,c)}(t), \tag{4.12}$$

where $R_i^{(n,c)}(t)$ denotes the achievable data rate of node $i$ through AP $n$ over channel $c$ at time step $t$.

### 4.5.2 Multi-Criteria Reinforcement Learning

However, addressing this multi-objective optimization problem poses significant challenges due to the inherently conflicting nature of the objectives and complex network dynamics. To address these challenges, we adopt a Q-learning-based approach. Conventional Q-learning is designed for single-objective optimization, where rewards are typically aggregated into a single scalar value through weighted summation. In contrast, we propose a multi-objective Q-learning approach that maintains distinct Q-values for each state-action-reward tuple. By preserving individual objectives without aggregation, our approach enables more flexible and adaptive decision-making, allowing the learning process to adjust to varying network conditions, application requirements, and trade-offs among competing objectives.

We model the problem as a Markov Decision Process (MDP), represented by the tuple $< \mathcal{S}, \mathcal{A}, \boldsymbol{r}, \Pi, F >$. Here, $\mathcal{S}$ refers to the state space, and $\mathcal{A}$ denotes the set of available actions. The reward function $\boldsymbol{r}$ is defined as a vector, considering a separate reward for each objective of the multi-objective optimization problem. The policy set $\Pi$ governs the mapping of states to actions, whereas $F$ characterizes the state transition probabilities, capturing the system's stochastic dynamics.

The set of states $S$ encompasses all possible states across all nodes and time steps. The state space is represented as $\mathcal{S} = \cup_{t \in T} \mathcal{S}(t)$, defined as $\mathcal{S}(t) \triangleq \{s_1(t), s_2(t), \dots s_I(t)\}$, where $s_i(t)$ denote the state for node $i$ at time step $t$, defined as $< x_i(t), y_i(t), dx_i(t), dy_i(t) >$. Here, $x_i(t)$ and $y_i(t)$ correspond to the node's spatial coordinates at time $t$ on the floor of the environment, while $dx_i(t)$ and $dy_i(t)$ capture its movement dynamics relative to the preceding time step. These movement components are computed as $dx_i(t) = x_i(t) - x_i(t-1)$ and $dy_i(t) = y_i(t) - y_i(t-1)$, reflecting the node's displacement along the respective axes.

The action space for the entire network, denoted as $A$, is defined as the union of the individual action spaces $\mathcal{A}(t)$ across all time steps $t \in T$, i.e., $\mathcal{A} = \cup_{t \in T} \mathcal{A}(t)$. At each time step $t$, the joint action set $\mathcal{A}(t)$ comprises the actions of all nodes, formally expressed as $\mathcal{A}(t) \triangleq \{a_1(t), a_2(t), \dots a_I(t)\}$, where $a_i(t)$ denote the action for node $i$. Each action $a_i(t)$ is defined as $< \hat{n}, \hat{p}, \widehat{n'}, \widehat{p'} >$, where $\hat{n} \in [0, N^O]$ and $\widehat{n'} \in [0, N^R]$ represent the selected OWC and RF APs for node $i$, respectively. If $\hat{n} = 0$, the node is not connected to any optical AP, and if $\widehat{n'} = 0$, it has no RF AP connection. When both values are zero, the node remains disconnected. The corresponding transmission power levels for OWC and RF are given by $\hat{p} \in [1, P^O]$ and $\widehat{p'} \in [1, P^R]$, respectively. Given this definition, the action space remains consistent across all time steps, implying that $A = A(t)$ for all $t$.

In the Q-learning algorithm, the system's state at time $t$, denoted as $\mathcal{S}(t)$, transitions to a new state upon executing an action. Unlike conventional Q-learning, which is designed for single-objective optimization, the proposed approach extends the framework to accommodate multiple objectives. Thus, each action yields a set of rewards rather than a single scalar reward. To align with optimization objectives, we define three distinct reward functions, $r^1(t)$, $r^2(t)$ and $r^3(t)$, each corresponding to a specific objective. Since $z^1(t)$ and $z^2(t)$ represent cost functions to be minimized, their respective rewards are defined as $r^1(t) = -z^1(t)$ and $r^2(t) = -z^2(t)$. Conversely,

as $z^3(t)$ represents the achievable data rate, which is to be maximized, its reward function is set as $r^3(t) = -z^3(t)$.

To mitigate the action space's dimensionality, the defined action space does not account for the specific channel assigned to each AP. Instead, we introduce a reward function that tracks the number of active connections to each AP. If the number of connections exceeds an AP's maximum channel capacity (i.e., more than $C^R$ for radio APs and $C^O$ for optical APs), a penalty is imposed. The reward function $r^4(t)$ reflects the negative sum of the number of APs operating beyond their capacity.

The goal is to learn a deterministic optimal policy $\pi^*$, which maps each state to one action such that the value function of a state $\mathcal{S}(t)$, i.e., its expected return received in time step $t$ onward, is maximized. The value function $V^{\pi,l}(\mathcal{S}(t))$ represents the expected cumulative discounted reward $l$for following a policy $\pi$ from state $\mathcal{S}(t)$ for a specific reward function $r^l(t)$:

$$V^{\pi,l}\big(\mathcal{S}(t)\big) = \mathbb{E}\left[\sum_{\tau=t}^{T} \gamma^{\tau-t}\, r^l(\tau) | \mathcal{S}(t), \pi\right], \tag{4.13}$$

where $\mathbb{E}$ denotes the expected value, and $\gamma \in [0,1]$ is the discount factor.

The state–action value function, commonly referred to as the Q-function $Q^{\pi,l}(\mathcal{S}(t), \mathcal{A})$, quantifies the expected cumulative discounted reward $l$ when the system is in state $\mathcal{S}(t)$, executes action $\mathcal{A}$, and follows policy $\pi$. This function defines a mapping between state–action pairs and their corresponding expected returns for reward $l$, and is given by

$$Q^{\pi,l}(\mathcal{S}(t), \mathcal{A}) = \mathbb{E}\left[\sum_{\tau=t}^{T} \gamma^{\tau-t}\, r^l(\tau) | \mathcal{S}(t), \mathcal{A}, \pi\right], \tag{4.14}$$

In single-objective problems, for any two policies $\pi$ and $\pi'$, the value functions $V^{\pi}(\mathcal{S}(t))$ and $V^{\pi'}(\mathcal{S}(t))$ exhibit a total order, meaning that one value function is always greater than, equal to, or less than the other. Thus, determining the optimal policy $\pi^*$ reduces to maximizing the expected cumulative discounted reward using an appropriate method. Therefore, the value function $V^{\pi}(\mathcal{S}(t))$ is given by

$$V^{\pi}\big(\mathcal{S}(t)\big) = \max_{a'} Q^{\pi}(\mathcal{S}(t), a'), \tag{4.15}$$

and the Q-value function is updated as

$$Q^{\pi}(\mathcal{S}(t), \mathcal{A}) = (1-\alpha) Q^{\pi}(\mathcal{S}(t), \mathcal{A}) + \alpha \left[r^l(t) + \gamma \max_{a'} Q^{\pi}(\mathcal{S}(t+1), a')\right], \tag{4.16}$$

where $\alpha$ is the learning rate.

However, when multiple value functions corresponding to different reward functions are defined, a total order may not necessarily exist. In particular, it is possible for one policy $\pi$ to yield a higher value than another policy $\pi'$ for a given objective $l_1$, i.e., $V^{\pi,l_1} > V^{\pi',l_1}$, while simultaneously yielding a lower value for another objective $l_2$, i.e., $V^{\pi,l_2} < V^{\pi',l_2}$.

In this study, the scalar Q-value for each state–action pair is extended to a vector-valued Q-value $\widehat{\mathbf{Q}}$, which maintains a separate $\hat{Q}$-value for each state–action–reward tuple. The expected reward for each state, action, and reward is stored, retrieved, and updated independently. The vector-valued Q-function is defined as

$$\widehat{\mathbf{Q}}^{\pi}(\mathcal{S}(t), \mathcal{A}) = \big(\hat{Q}^{\pi,1}, \hat{Q}^{\pi,2}, \hat{Q}^{\pi,3}, \hat{Q}^{\pi,4}\big). \tag{4.17}$$

The Q-table is thus extended to store separate value estimates for each reward function, requiring an adaptation of the single-objective Q-learning update rule to accommodate the multi-

objective setting. When selecting an action, instead of using $\max_{a'} Q^{\pi}(\mathcal{S}(t+1), a')$, the *ActionSelection* function from Algorithm 1 is employed to evaluate the four Q-values.

In the *ActionSelection* function, a random number is drawn from a uniform distribution $U$ to implement the $\epsilon$-greedy approach, balancing exploration, and exploitation. If the value is lower than $\epsilon$, a random action is selected from the action space; otherwise, the model exploits by choosing the best action. For exploitation, Q-values of all actions are first normalized by dividing each by its respective maximum value and then weighted according to predefined coefficients. The best and worst actions for each objective are identified based on the maximum and minimum weighted Q-values. The Euclidean distance between each action's weighted Q-value and those of the best and worst actions is then computed. The action with the highest score (closest to the best action and farthest from the worst) is chosen, ensuring an effective trade-off between objectives.

**Algorithm 1** Multi-Objective Q-Learning Action Selection

**Require:** Set of actions $\mathcal{A}$, Q-values $\hat{\mathbf{Q}}$,

1: **function** ACTIONSELECTION($\mathcal{S}(t)$, $\mathbf{w}$)
2: Sample rand $\sim U(0,1)$
3: **if** rand $< \epsilon$ **then**
4: **Exploration:** Choose a random action from $\mathcal{A}$
5: **else**
6: **for** each action $a \in \mathcal{A}$ **do**
7: **for** $l = 1$ to $4$ **do**
8: $\omega[a][l] \leftarrow \frac{\hat{\mathbf{Q}}[\mathcal{S}(t)][a][l]}{\max_{l'} \hat{\mathbf{Q}}[\mathcal{S}(t)][a][l']} \times \mathbf{w}[l]$
9: **end for**
10: **end for**
11: **for** $l = 1$ to $4$ **do**
12: $a_{\text{best}}[l] \leftarrow \arg\max_{a \in \mathcal{A}} \omega[a][l]$
13: $a_{\text{worst}}[l] \leftarrow \arg\min_{a \in \mathcal{A}} \omega[a][l]$
14: **end for**
15: **for** each action $a \in \mathcal{A}$ **do**
16: $d_{\text{best}}[a] \leftarrow \sum_{l=1}^{4} (\omega[a][l] - \omega[l][a_{\text{best}}])^2$
17: $d_{\text{worst}}[a] \leftarrow \sum_{l=1}^{4} (\omega[a][l] - \omega[l][a_{\text{worst}}])^2$
18: $\text{score}[a] \leftarrow \frac{d_{\text{worst}}[a]}{d_{\text{best}}[a] + d_{\text{worst}}[a]}$
19: **end for**
20: Choose the highest-scoring action
21: **end if**
22: **return** action
23: **end function**

Figure 62. Multi-objective action selection algorithm.

We evaluate the proposed multi-objective Q-learning framework, assessing its effectiveness in balancing the trade-off between energy consumption (for transmission power allocation in both optical and radio communications) and the overall achievable data rate. To assess the performance of the proposed algorithm, we consider a network deployment consisting of nine optical APs arranged in a $3 \times 3$ grid, along with four RF APs positioned at the four corners of the deployment area. The network operates within a three-dimensional environment measuring $10 \times 10 \times 3$ m$^3$. Nodes are randomly distributed across the floor and exhibit mobility according to the Random Waypoint Model, with an average speed of 1 m/s.

In the first experiment, we evaluated the convergence rate of the proposed multi-objective Q-learning algorithm compared to standard Q-learning. Both algorithms share the same state and action definitions. The reward function for standard Q-learning is defined as $r(t) = \sum_{l=1}^{4} \mathbf{w}[l]\, r^l(t)$, where $\mathbf{w}$ represents the weight coefficients. In multi-objective Q-learning, these weights are incorporated within the *ActionSelection* function (line 8). Figure 63 presents the normalized absolute Q-value variations, $\Delta Q_{\text{norm}}$, across training episodes, highlighting Q-value fluctuations at each step. Normalization enables a fair comparison of convergence behavior. While standard

Q-learning maintains a single Q-value per state–action pair, producing a single convergence curve, multi-objective Q-learning tracks four Q-values corresponding to distinct objectives. The results show that multi-objective Q-learning not only converges more rapidly but also provides deeper insight by allowing independent analysis of each objective's convergence pattern. Figure 63 compares two network configurations: Case 1 (Figure 63 (a)), where the network consists of three nodes ($I = 3$) with three available channels per communication type ($C^{\mathrm{O}} = C^{\mathrm{R}} = 3$), and Case 2 (Figure 63 (b)), where the network also has three nodes ($I = 3$) but only one available channel per communication type ($C^{\mathrm{O}} = C^{\mathrm{R}} = 1$). A key observation is that the reward function $r^4$(overload penalty), denoted by $\hat{Q}^4$, converges significantly faster in Case 1 than in Case 2. This behavior is attributed to the sufficient number of channels in Case 1, which reduces contention and simplifies optimization, allowing $\Delta Q_{\mathrm{norm}}$ to stabilize from the first episode. Conversely, in Case 2, where nodes outnumber channels, higher contention initially slows the convergence of $\hat{Q}^4$, which stabilizes after approximately 30 episodes. The reward function $r^1$, represented by $\hat{Q}^1$, exhibits the slowest convergence in multi-objective Q-learning, indicating that optimizing transmission power for optical communication is more complex than for other objectives. This slower convergence is due to the larger number of optical APs ($N^{\mathrm{O}}$) compared to radio APs, which increases the search space for optimal power allocation.

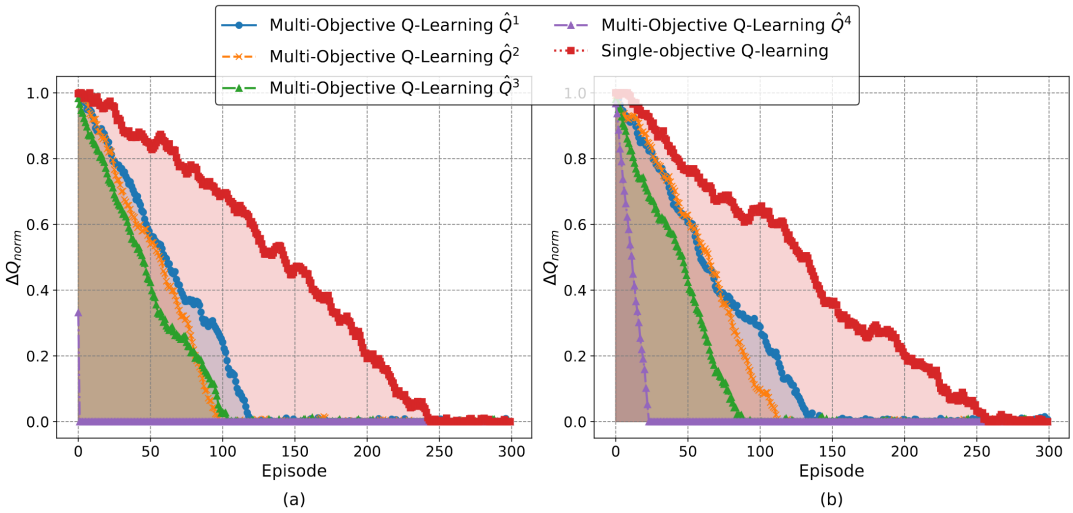



Figure 63. Normalized Q-value deltas ($\Delta Q_{\mathrm{norm}}$) during training for different network configurations: (a) $I = 3$, $C^R = C^O = 3$; (b) $I = 3$, $C^R = C^O = 1$.

To evaluate energy efficiency in a dynamic environment, we employ the Energy-to-Rate Ratio (ERR), defined as the total transmission power divided by the achieved data rate. Specifically, $\mathrm{ERR}^{\mathrm{O}}$ and $\mathrm{ERR}^{\mathrm{R}}$ represent the ERR for OWC and RF, respectively. These metrics are computed by summing the transmission power over time for each communication type and dividing it by the corresponding data rate. In this experiment, fixed weight coefficients are assigned to data rate maximization and AP overload penalties, with $\mathbf{w}[3] = 0.25$ and $\mathbf{w}[4] = 0.25$. Meanwhile, the weights for transmission power in OWC ($\mathbf{w}[1]$) and RF ($\mathbf{w}[2]$) are varied while ensuring $\mathbf{w}[1] + \mathbf{w}[2] = 0.5$. This configuration maintains equal emphasis on data rate and AP overload penalties (each contributing 25%) while adjusting the trade-off between OWC and RF power consumption. Figure 64 illustrates the impact of increasing $\mathbf{w}[1]$ from 0.1 to 0.4 on $\mathrm{ERR}^{\mathrm{O}}$ and $\mathrm{ERR}^{\mathrm{R}}$. Since OWC transmission power is used for both illumination and data transmission, its scale is significantly higher than that of RF transmission power. Consequently, the y-axis scales of $\mathrm{ERR}^{\mathrm{O}}$ and $\mathrm{ERR}^{\mathrm{R}}$ differ to reflect this disparity. Increasing $\mathbf{w}[1]$ emphasizes minimizing energy consumption in OWC, leading to a reduction in $\mathrm{ERR}^{\mathrm{O}}$. This decrease is more pronounced up to $\mathbf{w}[1] = 0.25$, after which the improvement slows due to reduced data rates from lower OWC power. As $\mathbf{w}[1]$ increases, $\mathbf{w}[2]$ decreases, lowering the priority on RF energy efficiency and causing $\mathrm{ERR}^{\mathrm{R}}$ to rise. However, beyond $\mathbf{w}[1] = 0.25$, the limited RF bandwidth means that increased energy use does not translate into significantly higher data rates, further increasing $\mathrm{ERR}^{\mathrm{R}}$. The ERR for single-

objective Q-learning is notably higher than that of multi-objective Q-learning, highlighting the limitations of single-objective methods in balancing energy consumption and data rate. Single-objective approaches often result in either excessive power usage or reduced throughput due to conflicting goals. In contrast, the proposed multi-objective Q-learning maintains separate Q-tables for each objective, enabling dynamic rebalancing without retraining and better adaptation to real-world network conditions.

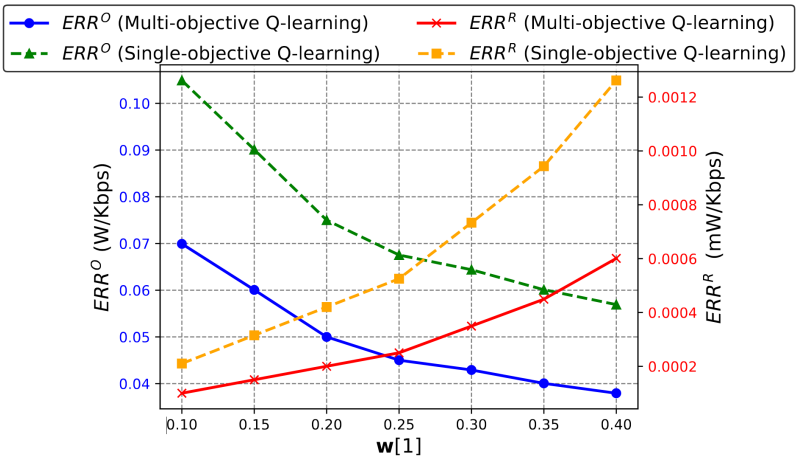



Figure 64. ERR for OWC and RF with varying importance weights on transmission power for each technology.

While the proposed multi-objective Q-learning approach offers a superior convergence rate and enhanced energy efficiency, it requires greater memory allocation for storing Q-values. Notably, the complexity of the *ActionSelection* function is $O(3 \times |\mathcal{A}|)$; however, due to the faster convergence of multi-objective Q-learning, its impact on computational complexity remains minimal. Moreover, since standard Q-learning is already impractical for large-scale scenarios, the additional complexity has little effect on its applicability. Thus, the benefits of improved convergence and energy efficiency make this approach valuable in diverse networking environments, despite the increased memory requirements.

## 4.6 Network Implementation

### 4.6.1 Network topology of RIoT nodes and APs/GWs, Raspberry PI (broker + master node), router/switch

The architecture of the network is depicted in Figure 65.

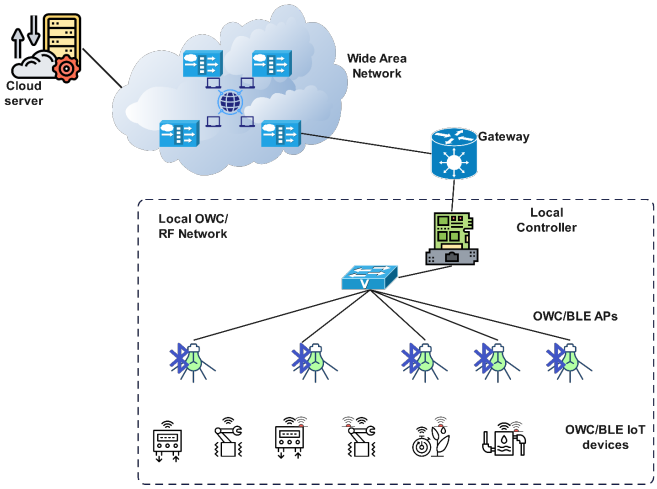



Figure 65. Main Architecture of practical implementation.

The main elements of this network are as follows:

**Cloud Broker:**

The cloud broker serves as the primary interface between users and the SUPERIOT network, hosting an MQTT broker.

**Master Node:**

The master node acts as the central control unit in the SUPERIOT network, responsible for managing and coordinating various network functions. Figure 66 illustrates the communication flow and the master node's central position within the network's architecture.

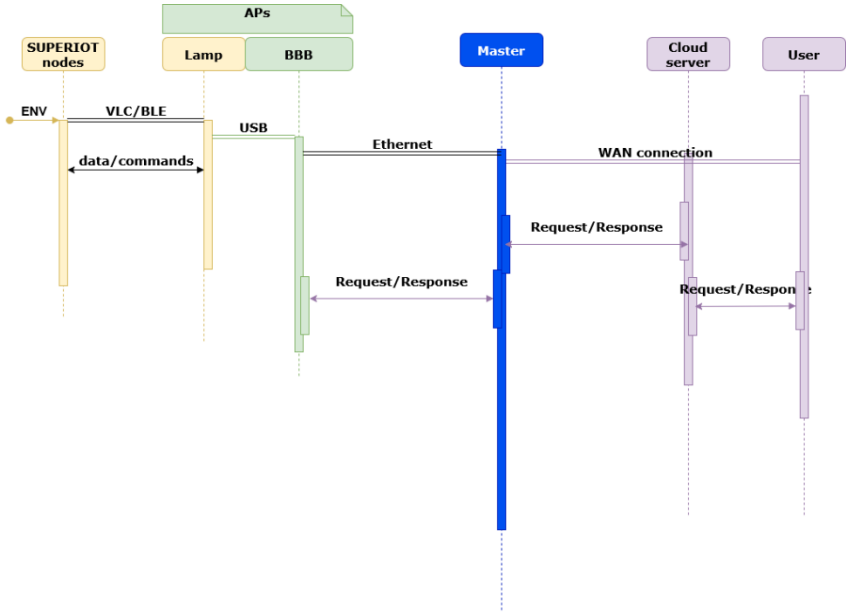



Figure 66. The Master node's role as the central communication hub in the SUPERIOT operational procedure.

The key responsibilities of master node in the network include:

- **Network Management**: Maintains detailed information about SUPERIOT nodes, lamps, and APs, including their addresses and potential connections. It defines the network topology, determining which SUPERIOT node connects to which lamp or AP.
- **APs Registration**: Manages the registration process for APs.
- **Data Integration and Forwarding**: Processes data received from APs and forwards it to the cloud broker for further handling.
- **Communication Management**: Operates its own MQTT broker to manage message exchanges between APs and the master node.

Master needs the addresses of two MQTT brokers:

- **Master-to-Cloud Broker**: Manages communication with cloud broker.
- **AP-to-Master Broker**: Handles internetwork messages between the master node and APs.

**APs:**

Each AP is a BBB and a Lamp, serve as intermediaries between SUPERIOT nodes and the network. Lamps communicate using BLE and VLC protocols, providing flexible connectivity with SUPERIOT nodes.

Key features include:

- **MQTT Brokers**: Each AP has two MQTT broker addresses:
  - **AP-to-Lamp Broker**: Manages local communication with lamps.
  - **AP-to-Master Broker**: Handles internetwork messages with the master node.

As shown in Figure 67, each AP requires an inter-process communication (IPC) mechanism to facilitate seamless communication between two essential components: accesspoint.py and node_translator.py. To implement this IPC mechanism, we adopt the MQTT protocol.

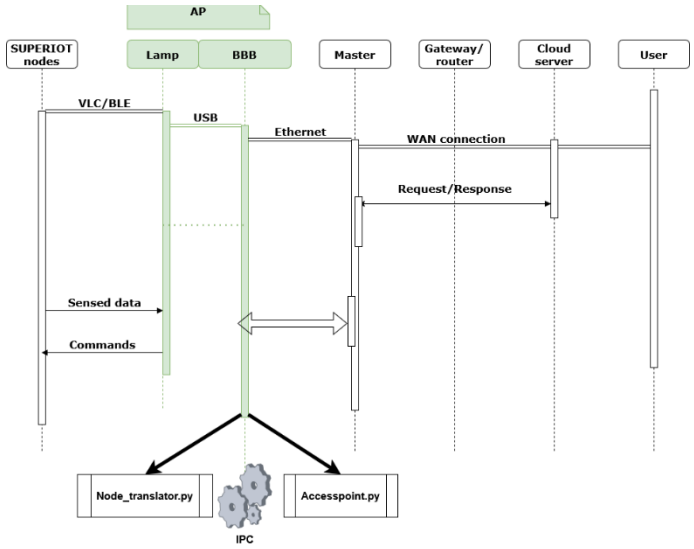



Figure 67. Sequence diagram illustrating the internal software architecture of an AP.

**Overall architecture:**

The overall architecture of the SUPERIOT network is illustrated in Figure 68.

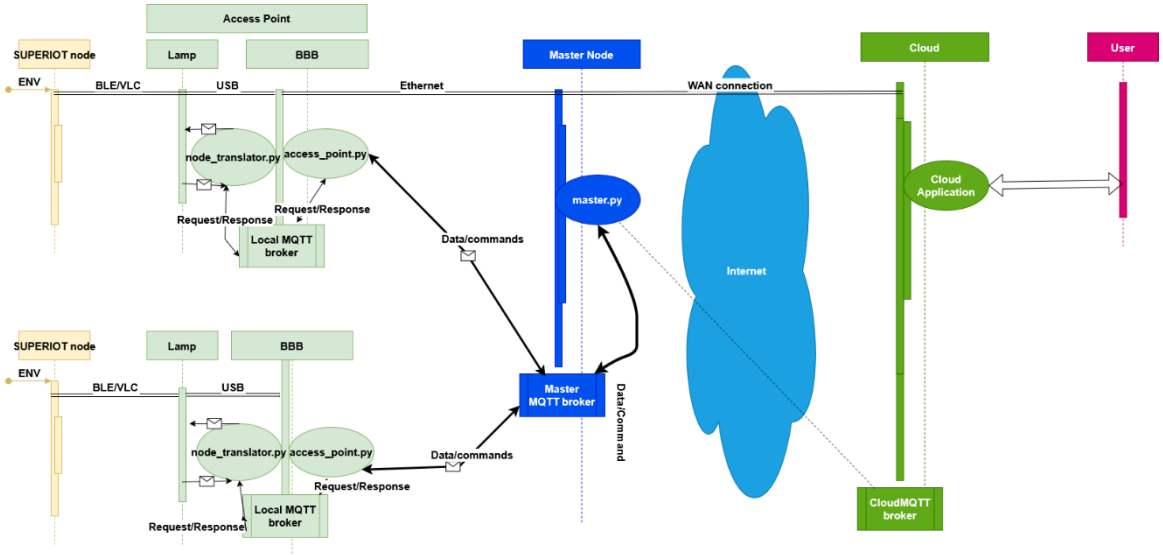



Figure 68. System structure.

### 4.6.2 MQTT Architecture and Implementation

#### *4.6.2.1 Protocol Selection and Design Rationale*

Several communication protocols were considered during the design phase of the SUPERIOT network architecture, including the Constrained Application Protocol (CoAP), Advanced Message Queuing Protocol (AMQP), Hypertext Transfer Protocol (HTTP), and Message Queuing Telemetry Transport (MQTT). CoAP is attractive for constrained IoT environments because it typically involves low message size and protocol overhead. It uses the User Datagram Protocol (UDP) as its transport layer, which avoids the connection setup overhead associated with Transmission Control Protocol (TCP)-based protocols. However, UDP does not provide connection-oriented delivery guarantees by default, and reliability must be handled through CoAP-level mechanisms such as confirmable messages and retransmissions. In addition, CoAP follows a client-server request-response model, which is less aligned with the one-to-many and many-to-one data dissemination pattern required in the SUPERIOT architecture.

AMQP was also considered because it provides richer messaging features, reliability mechanisms, and strong support for secure enterprise communication. However, these capabilities introduce higher protocol complexity and larger message overhead, making AMQP more suitable for enterprise-grade business messaging than lightweight sensor telemetry. HTTP was ruled out because of its verbose text-based structure and relatively high per-message overhead, which makes it poorly suited for constrained IoT nodes and frequent small-payload communication.

MQTT was selected because it provides a practical balance between lightweight operation, scalability, reliability, and implementation simplicity. MQTT can be implemented with minimal resources and is therefore suitable for small embedded devices and microcontroller-based systems. Its control messages and headers are compact, which helps optimise bandwidth usage in constrained networks. By decoupling senders and receivers through a publish-subscribe architecture, MQTT enables efficient asynchronous communication between the cloud server, master node, access points, gateways, applications, and IoT nodes. Furthermore, MQTT supports three configurable Quality of Service levels, allowing message delivery guarantees to be tuned according to the reliability requirements of each communication link.

Beyond the protocol-level comparison, the specific topology of the SUPERIOT system further justified the selection of MQTT over UDP-based alternatives such as CoAP. The architecture comprises three main communication segments: the persistent link between the cloud server and the master node running on a Raspberry Pi, the fixed link between the master node and the access point running on a BeagleBone Black, and the link between the access point and the IoT end nodes. The first two segments involve stable, long-lived connections that are established once and then maintained continuously. In this topology, the TCP connection setup overhead associated with MQTT is incurred only during connection establishment and does not recur for every message transmission. Therefore, the persistent connection model of MQTT becomes an advantage rather than a limitation.

For the final segment between the access point and the IoT nodes, MQTT was retained to preserve protocol uniformity across the complete communication stack. This avoids the added complexity of protocol translation at the access point layer and ensures consistent topic naming, message handling, and QoS semantics throughout the system. In addition, the SUPERIOT deployment relies on a self-hosted cloud server rather than a managed cloud platform. MQTT is therefore well suited to this architecture because mature open-source broker implementations, such as Mosquitto, are widely available and straightforward to deploy, configure, and maintain without dependency on vendor-specific infrastructure.

The SUPERIOT implementation further extends a standard MQTT deployment by using a hierarchical broker arrangement, with a local broker on the master node operating alongside the cloud broker. This design improves robustness by allowing local communication between the master node, access points, and local applications to continue even if the connection to the cloud broker is temporarily unavailable. MQTT broker features such as message buffering, persistent sessions, and broker-to-broker bridging support this design by enabling selected topic flows to be mapped between local and cloud brokers. Once cloud connectivity is restored, relevant data

can be synchronised with the cloud broker. This hierarchical architecture reduces dependence on a single central communication point, although it also introduces additional requirements for topic-bridging configuration, synchronisation logic, and master-node processing.

It is acknowledged that MQTT is not universally optimal for all IoT scenarios. UDP-based protocols such as CoAP may be preferable in highly constrained, intermittent, or very frequent short-message communication scenarios. However, in the context of the SUPERIOT system, which involves a moderate number of devices, structured topic-based communication, persistent links, and a centralised self-hosted server architecture, MQTT provides a suitable and pragmatic balance between efficiency, reliability, scalability, and implementation simplicity. The combination of lightweight design, configurable reliability, persistent connection efficiency, protocol stack uniformity, and robust open-source tooling makes MQTT a well-justified choice for the communication requirements of the SUPERIOT network.

### *4.6.2.2 MQTT Architecture, Topic Structure and Message Flows*

MQTT is the lightweight protocol designed for low-bandwidth or high-latency environment operating devices. Besides lightweight and low bandwidth it is also scalable, low power and low-cost solution that handles operating over unstable connection. MQTT is based on a publish/ subscribe pattern, where clients (SUPERIOT gateways or applications) can subscribe to receive information and / or publish information on selected topics. MQTT protocol allows group data in so called 'topics'. Topic is similar to a path as known on the computer directories (folders) that allows information to be filtered. The general view of the MQTT architecture idea is presented in the Figure 69. The MQTT architecture with respect to the SUPERIOT system structure is detailed in Figure 70.

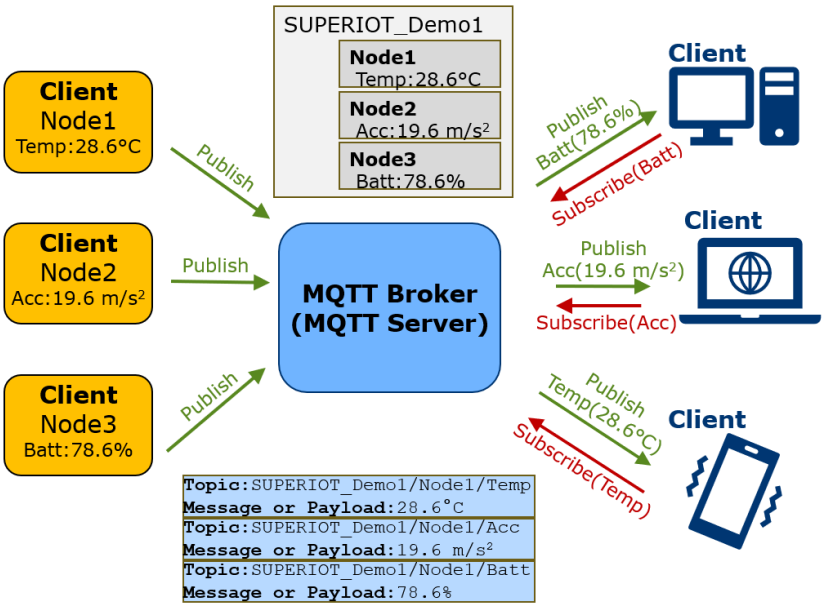



Figure 69. The general view of the MQTT architecture idea.

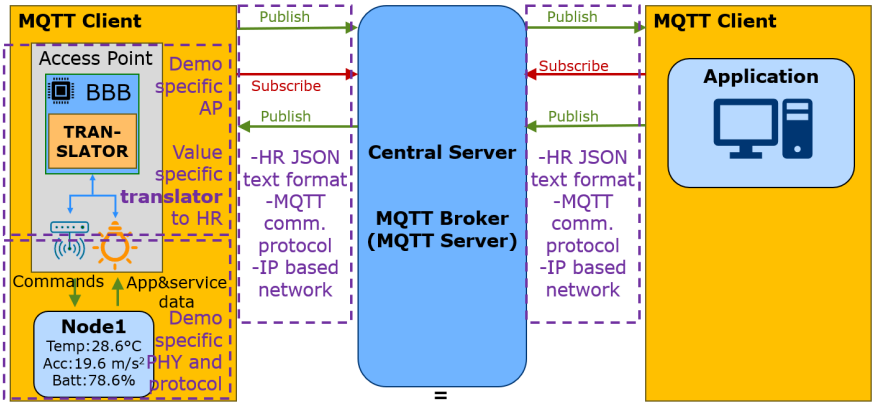



Figure 70. The MQTT architecture with respect to the SUPERIOT system structure.

Topics are constructed like "/demo1/gateway_1/node_123", wherein "/" is used to separate topic levels. You can also use wildcards to subscribe to multiple topics, like:

"+" (plus sign) - example: "/demo1/+/node_123" - here the "+" character replaces any string; therefore you will receive information send to topics like:

"/demo1/gateway_1/node_123"

"/demo1/gateway_2/node_123",

"/demo1/anyotherstring/node_123"

etc.

"#" (hash sign) - example: "/demo1/#" - replaces all remaining levels of the topic, thus client will receive data from topics like:

"/demo1/gateway_1/node_123"

"/demo1/gateway_3/node_222"

"/demo1/gateway_1/node_555"

### *4.6.2.3 MQTT Server and Broker Deployment*

The MQTT server (broker) is deployed at the following address:

Server URL: blacked out for cybersecurity

Port: 1883 (default MQTT port)

The broker is configured to accept authenticated client connections using the following credentials:

Username: blacked out for cybersecurity

Password: blacked out for cybersecurity

#### *4.6.2.4 Data Formats and JSON Schema Development*

**MQTT Broker Configurations**

Data conveyed by the MQTT protocol is treated as binary, thus it can be in any form. As agreed at the project meetings, we are transmitting data in JSON format with as much descriptive information as possible.

All network-layer communication is structured using JSON messages. Below is the general format of a network-layer JSON message used in SUPERIOT project:

```
message = {
   "protocol_id": "SUPERIOT_NET",
   "protocol_ver": PROTOCOL_VERSION,
   "message_header": {
      "message_id": --- ,
      "message_timestamp": --- ,
      "message_destination_address": --- ,
      "message_destination_subelement": --- ,
      "message_source_address": ---- ,
      "message_source_subelement": 1,
      "message_acknowledgement_requested": ---
   },
   "message_payload": {
      ...
   }
}
```

In this format, each message consists of two primary components:

1. **Message Header:** Contains metadata about the message, including sender and receiver details.
2. **Message Payload:** Contains the actual data being transmitted, such as commands, responses, or status updates.

The **message header** includes information about the transmitter and receiver, as well as a **message source** and **message destination** sub-elements. These sub-elements facilitate addressing at different levels within the network.

For example, when a command originates from the **cloud server**, the message will have:

- "message_source_address" set to the cloud server's identifier.
- "message_source_subelement" left empty.
- "message_destination_address" set to "master_node" (indicating the next hop).
- "message_destination_subelement" set to the MAC address of the Access Point (AP) that should receive the command.

**Message Payload Structure**

The **message payload** varies depending on the purpose of the message. In the case of **commands sent from the cloud server to the master node**, the following format is used:

```
message_payload = {
   "message_type": "command",
   "command_payload": {
      "command_category": ---,
      "exact_command": --- ,
      "SUPERIOT_NODE": "All_MAC_addr",
      "response_flag": False
   }
}
```

Some fields (e.g., "SUPERIOT_NODE") are included as placeholders for potential future use, even though they may not be implemented yet. If needed, this format can be modified based on project requirements.

#### *4.6.2.5 Cloud-to-Network Command Flow*

**How to Send a Command from the Cloud Server to the Network**

**1. Connecting to the Cloud MQTT Broker**

Any user can connect to the cloud server's MQTT broker to send commands to the network. The client must first subscribe to the appropriate topic:

```
CLOUD_TOPIC_COMMANDS = "cloud_broker/commands"
cloud_mqtt_client.subscribe(CLOUD_TOPIC_COMMANDS)
```

**2. Sending a Command Message**

Once connected, the user can send a command using the following message format:

```
message = {
   "protocol_id": "SUPERIOT_NET",
   "protocol_ver": PROTOCOL_VERSION,
   "message_header": {
      "message_id": --- ,
      "message_timestamp":  --- ,
      "message_destination_address":  --- ,
      "message_destination_subelement": --- ,
      "message_source_address":  ---- ,
      "message_source_subelement": 1,
      "message_acknowledgement_requested": ---
```

```
    },
    "message_type": "command",
    "command_payload": {
        "command_category": "cloud_to_master_command",
        "exact_command": "#LON,51", #or other commands
        "SUPERIOT_NODE": "All_MAC_addr",
        "response_flag": False
    }
}
```


**3. Message Handling at the Network layer**

- The message reaches the **master node** via the Cloud MQTT broker.
- The function **handle_cloud_command_message(payload)** in master.py processes the received message and determines the appropriate AP to forward it to.
- Master makes some changes in message format and then send this command to the destination AP(s).
- AP calls send the command to the node_translator (interface between network and MAC layer). And node_translator calls command_handler function in get_temperature_from_mac.py.

## 4.7 Advanced Techniques for Hybrid RF-Optical Communication

This section investigates the resource allocation challenges in a hybrid dual-band IoT network that utilizes RF and OWC technologies, following the example shown in Figure 71. We first formulate the optimization problem as a Mixed-Integer Non-Linear Programming (MINLP) with two-fold objectives: 1) maximize the successful data transmission across the network with respect to energy consumption, and 2) minimize the overall age-of-information (AoI) in the network. As direct minimization of the time-average AoI is known to be intractable in general multisource, multi-destination networks [27] due to the coupling of scheduling decisions over time and across links, we adopt a tractable surrogate that minimizes the age of each update at its delivery instant. It is worth noting that, as stated in D3.2, AoI as a metric primarily captures the freshness of data transmission, while throughput remains strongly dependent on the underlying communication technology. For BLE, throughput is significantly influenced by PHY configuration: under a 2 Mbps PHY, it can reach approximately 120 kB/s. In contrast, optical can support up to 145 bps due to protocol and modulation constraints. This highlights that, beyond AoI considerations, the achievable throughput varies substantially between BLE and optical links depending on their respective configurations and protocol designs.

We consider a hybrid IoT network consisting of APs and nodes that can communicate through RF and OWC links. We assume that nodes communicate with APs using either OWC or RF, whereas only RF is used for inter-node communication. RF transmissions are assumed to happen in pre-assigned time slots using Time Division Multiple Access (TDMA), ensuring no two devices transmit simultaneously over the same channel. Moreover, we suppose that OWC uses tightly focused beams of visible or infrared light, which are inherently directional. Therefore, each device pair can establish either a LoS OWC/RF link or rely on a quasi-LoS RF link, minimizing inter-device interference. The system model with an RF-OWC hybrid configuration is illustrated in Figure 71 with different IoT nodes transmitting multiple types of application data, such as sensor readings, status updates, and alarm messages.

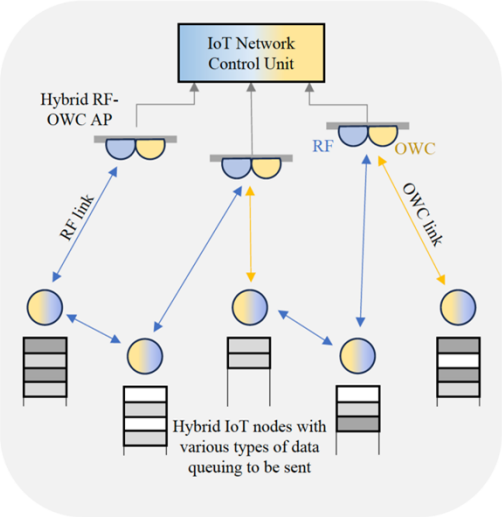



Figure 71. A visualization of the RF-OWC model showing a hybrid RF-OWC network exchanging different types of applications' data.

Our framework facilitates dynamic and context-aware band selection, where IoT devices autonomously choose the most suitable communication technology, either RF or OWC, based on real-time network and environmental conditions. In particular, we adopted a system formulation for a hybrid RF-OWC network and formulate a constrained MINLP optimization problem, approximated as a Mixed-Integer Linear Programming (MILP), that seeks to maximize the overall throughput of the network while taking into account the energy consumption and the delivery-based AoI. The detailed formulation of this optimization problem can be found in Section IV in [28].

### 4.7.1 Graph Neural Network-Based Optimization

The hybrid RF-OWC resource allocation problem is categorized as NP-hard [29]. Due to the intractability of solving such NP-hard problems efficiently at scale and the impractical assumption of full system knowledge by deterministic optimizers, we propose an alternative data-driven Dual-Graph Embedding with Transformer (DGET) architecture utilizing GNNs and Transformers [12]. We first model the IoT network as a time-series graph, where each snapshot represents the network's state at a given time. The input snapshots consist of known network states, including initial device's energy levels, transmission schedules, and available communication links over coherence time. The recorded snapshots are obtained from solving the scheduling problem using optimization techniques, and they contain ground truth values for the network changes over several time steps, such as evolving device's energy levels and the selected communication links at each step. The input snapshots are first processed by a transductive GNN, which learns fine-grained embeddings that capture both local and global structural patterns of the IoT network. Transductive GNNs [13] enforce structural priors from observed networks and achieve local consistent embeddings, reflecting the connectivity and state information present in the input graphs. However, they alone lack the flexibility to adapt to new or unseen graph instances, especially in dynamic environments like IoT networks where the topology and conditions evolve over time. Therefore, we propose to leverage an inductive GNN [14] component which could further refine and enrich the transductive embeddings by incorporating decision-level semantics from the recorded graph, such as actual scheduling decisions, selected communication links, and resulting energy consumption during each time step. The inductive GNN is trained as a domain adaptation mechanism, where a consistency loss is used to align its reconstructed embedding output with embeddings derived from the recorded graphs. This loss guides the DGET model to transfer decision-level knowledge from the training domain to the inference domain as a first task. Furthermore, the DGET model is trained using a multi-task learning framework, where the second task involves link classification. For the classification task,

a transformer-based encoder head is applied to the reconstructed embeddings to predict the communication link type for each device. This multi-task learning approach allows the model to simultaneously optimize two coupled tasks; aligning the embeddings reconstruction to the ground truth recorded graph and also improving the accuracy of link prediction.

#### *4.7.1.1 Graph Modeling and Parameter Tuning*

As an alternative to solving the computationally expensive resource allocation problem in RF-OWC IoT networks, the RF-OWC hybrid IoT network can be modelled as a temporal directed graph $\mathcal{G}_k(\mathcal{V},\mathcal{E})$, $k \in \mathcal{T}$, where the vertices represent the total IoT devices (i.e., $|\mathcal{V}| = |\mathcal{N}|$, including IoT nodes and APs) and the edges denote the hybrid communication links at any time step $k \in \mathcal{T}$. An example is shown in Figure 72. Since the graph is directed and self-loops are disallowed (i.e., a device cannot communicate with itself), the total number of potential edges is $|\mathcal{E}| = |\mathcal{N}|(|\mathcal{N}| - 1)$. To capture the time-varying nature of the network, we define two types of time-series graph snapshots: $\mathcal{G}_k^I(\mathcal{V},\mathcal{E})$ and $\mathcal{G}_k^R(\mathcal{V},\mathcal{E})$. The first graph, $\mathcal{G}_k^I(\mathcal{V},\mathcal{E})$ which will be referred to as the input graph, captures the state of the environment at time k and includes prior knowledge of the network, including allowable communication technologies between devices, initial energy levels, and the planned communication schedule. In contrast, the second graph, $\mathcal{G}_k^R(\mathcal{V},\mathcal{E})$, which will be referred to as the recorded graph, represents the resulting state of the system after communication activities have taken place. This graph reflects communication updates such as exchanged messages, link selections, and energy depletion resulting from actual communication activities.

For the input graph, each device $i \in \mathcal{V}$ is associated with a feature vector $\mathbf{v}_i^I$, which encodes its local state information (e.g., available packet for transmission, initial energy level, energy per bit consumption, etc). In the recorded graph, the devices' features are referred to as $\mathbf{v}_i^R$, which contain $\mathbf{v}_i^I$ but also the parameters evolution over time to reflect communication decisions made at each step (e.g., energy reduction due to data transmission). Similarly, the input graph includes a link feature vector from device i to device j, denoted as $\mathbf{e}_{ij}^R$, as depicted in Figure 72. The latter includes the prior knowledge to link specific information, such as the queue of messages pending between the devices and the types of packets scheduled for transmission. The edges feature vector eI ij contains the possible communication links $\mathcal{E}_t = \{0,1,2,3\}$, which encodes the allowed communication technologies between two devices: 0 means no communication is possible, 1 allows only RF, 2 allows only OWC, and 3 allows either RF or OWC. This information is prior to knowledge of both the input and the recorded graphs. In contrast, the recorded graph includes a link feature vector from device $i$ to device $j$, denoted as $\mathbf{e}_{ij}^R$, which not only includes the possible communication links $\mathcal{E}_t$ but also the ground truth link status $\mathcal{E}_c = \{0,1,2\}$. The latter indicates the chosen communication links at a given time with 0 denoting no communication occurred, 1 denoting RF communication was selected, and 2 denoting communication was via OWC. In this context, we note $\mathcal{E}_{t_{ij}}$ as the permissible communication technologies from device $i$ to device $j$, while $\mathcal{E}_{c_{ij}}$ indicates the actual communication technology selected for transmission from device $i$ to device $j$. Both the recorded device feature $\mathbf{v}_i^R$, and the link's status $\mathbf{e}_{ij}^R$ are obtained as a result of solving the resource allocation problem at hand and recording the network parameter evolution. By definition, the selected communication status $c \in \mathcal{E}_c$ must satisfy $c \leq f$ for any edge, with $f \in \mathcal{E}_t$, ensuring the chosen technology complies with the available link constraints. This modeling allows us to shift the resource allocation problem in the IoT network to predicting the value of $c \in \mathcal{E}_c$ for each pair $i,i'$ in the graph $\mathcal{G}_k^I(\mathcal{V},\mathcal{E})$ at any given time step k ∈ T. We reformulate the task as a classification problem acting on the input graph. Specifically, we aim to predict the appropriate communication status for each edge in the input graph $\mathcal{G}_k^I(\mathcal{V},\mathcal{E})$ at each time step $k$ by establishing a learning process from the information in the recorded graph.

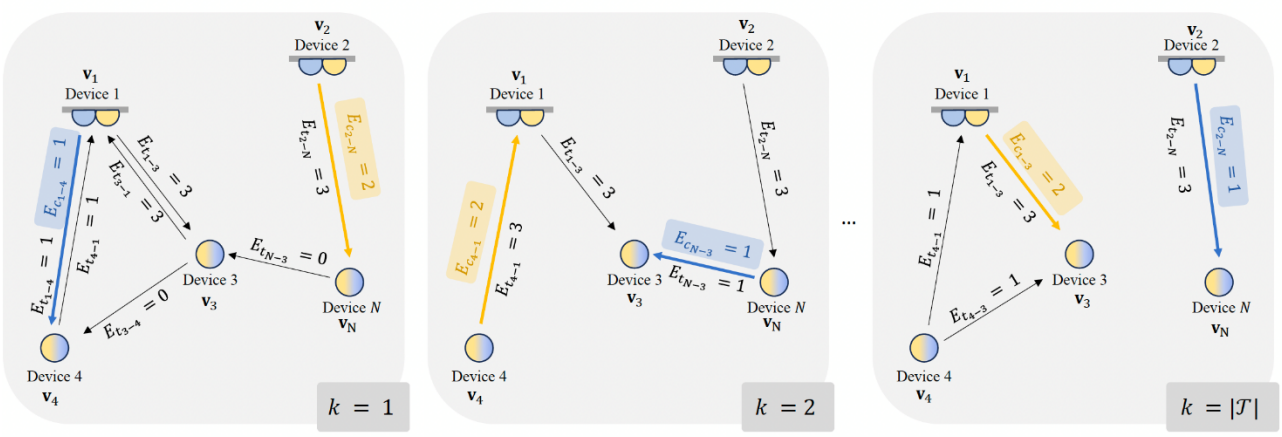



Figure 72. An example of a graph modeling for a network with $N$ devices over a period of time $\mathcal{T}$. $\mathcal{E}_{t_{i-j}}$ represents the allowed communication links from device $i$ to device $j$. $\mathcal{E}_{c_{i-j}}$, on the other hand, represents the chosen communication technology for that link.

Table 18. M-AoI and P-AoI for both RF (MINLP), RF-OWC (MINLP), and RF-OWC (DGET) configurations computed for the overall system, data type 1, and data type 2.

| | RF (MINLP) | RF-OWC (MINLP) | RF-OWC (DGET) |
|---|---|---|---|
| All Data | M-AoI: 7.3<br>P-AoI: 8.9 | M-AoI: 6.1<br>P-AoI: 7.8 | M-AoI: 6.4<br>P-AoI: 8 |
| Data Type 1 | M-AoI: 7.5<br>P-AoI: 8.9 | M-AoI: 6<br>P-AoI: 7.7 | M-AoI: 6.2<br>P-AoI: 7.9 |
| Data Type 2 | M-AoI: 6.9<br>P-AoI: 8.9 | M-AoI: 6.3<br>P-AoI: 8.1 | M-AoI: 6.6<br>P-AoI: 8.3 |

#### *4.7.1.2 Scalability and Minimizing Age of Information (AoI)*

In Table 18, we evaluate the performance of three communication configurations using 9 RIoT nodes and 2 access points. The results compare mean age of information (M-AoI) and peak age of information (P-AoI) across: (1) an RF-only system optimized via MINLP, (2) a hybrid RF-OWC system also optimized via MINLP, and (3) a hybrid RF-OWC system where scheduling is handled by the DGET (GNN) model. The optimized RF-OWC configuration yields the best performance, with an M-AoI of 6.1 and a P-AoI of 7.8, showing clear improvements over the RF-only baseline (M-AoI of 7.3, P-AoI of 8.9). These gains are consistent across both data categories (i.e., Data Type 1 and Data Type 2). The DGET model exhibits slightly higher AoI values (e.g., M-AoI of 6.4 vs. 6.1 for all data), but the differences are minor. This indicates that, despite not being globally optimal, the learned DGET policy can closely approximate the performance of the optimization-based solution in terms of AoI.

In Figure 73, we evaluate the running time to highlight the inference computational efficiency of the DGET compared to the optimization solver. We maintain the ratio of 1 APs per 3 IoT nodes, and increase the overall number of IoT devices, respectively. The RF-OWC system, which incorporates both RF and optical wireless communication capabilities, shows the highest computational cost due to the added complexity of managing multi-band interfaces and link selection processes. In contrast, the traditional RF system presents a more moderate increase in execution time, reflecting its simpler communication model. The DGET framework consistently outperforms both baseline methods in terms of execution efficiency. Despite incorporating advanced graph-based reasoning and sequence modeling, it maintains the lowest running time across all network sizes, demonstrating a polynomial trend that is estimated $O(|\mathcal{N}|^2\,\varphi)$ with $\varphi$ is the aggregated hidden dimensionality across model components.

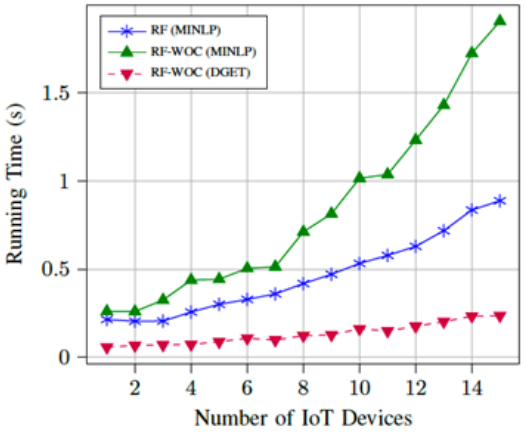



Figure 73. Running time (s) of the RF, RF-OWC, and DGET model vs. number of IoT devices in the network.

#### *4.7.1.3 Model Resilience*

In Figure 74, we evaluate DGET's robustness against missing or outdated link information for RF-OWC network. We replace edge weights in the input graph $\mathcal{G}_k^I(\mathcal{V},\mathcal{E})$ with deprecated values from previous snapshots to simulate outdated channel information of the allowed communication technology. The snapshots used for replacement are non-subsequent snapshots, i.e., not directly adjacent in time, preventing the carryover of stale information across consecutive updates. We evaluate three scenarios in which edge weight replacement occurs in 1, 2, or 3 randomly selected non-subsequent snapshots among $|\mathcal{T}| = 10$, with the overall replacement rate uniformly varied between 10% and 50%. Across all settings, DGET consistently outperforms the MINLP baseline. For example, when 20% of all edges are outdated across 2 snapshots, DGET maintains nearly 90% accuracy, whereas MINLP experiences a significantly sharper decline in performance. These results demonstrate DGET's ability to maintain acceptable link allocation under imperfect channel knowledge while leveraging historical snapshot data.

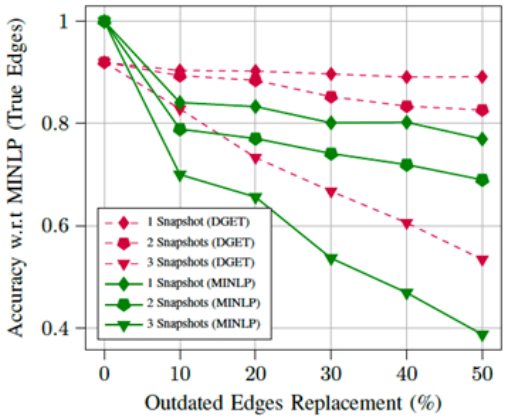



Figure 74. Impact of edge staleness on DGET prediction (link class) accuracy when using outdated allowed technologies for RF-OWC network with $|\mathcal{T}| = 10$.

## 4.8 Chapter Conclusion

This chapter provided a comprehensive analysis of communication modality and access technology optimization in hybrid RF/OWC networks. It outlines the fundamental challenges of modality selection and handover management, emphasizing the complexity introduced by heterogeneous technologies, overlapping coverage, and asymmetric link characteristics. To address these challenges, the chapter presented a series of advanced, progressively sophisticated solutions. First, a decentralized, forward-looking MADM framework using the

VIKOR method was proposed for intelligent handover management. This approach dynamically balances handover costs, QoS requirements, and network load, achieving superior load distribution compared to greedy algorithms. Second, the problem of joint transmission power allocation and modality selection was tackled using a novel multi-objective Q-learning framework. This reinforcement learning approach effectively manages the inherent trade-offs between minimizing power consumption and maximizing data throughput, demonstrating faster convergence and enhanced energy efficiency over traditional single-objective methods. Furthermore, an advanced GNN-based architecture (DGET) was introduced to solve the NP-hard resource allocation problem on a large scale. This data-driven model proved highly efficient, achieving performance by minimizing the AoI that closely approximates computationally expensive optimization methods but with significantly lower overhead. It also exhibited strong resilience to imperfect or outdated network information. Finally, the chapter detailed the practical network implementation of these concepts, outlining a robust architecture featuring a master node, APs, and a cloud broker interconnected via the lightweight MQTT protocol.

# 5 Cross-Layer Optimization using Traffic and Usage Pattern Prediction

## 5.1 Overview and State-of-the-Art of ns-3 Network Simulator

### 5.1.1 State-of-the-Art in IoT Node Simulation Models

Developing a DT of the RIoT node in the ns-3 simulation environment requires accurate modeling of its primary subsystems, including: RF via BLE communication, OWC via VLC, and the energy harvesting and consumption behavior that enables autonomous operation. This section surveys the current state-of-the-art in each of these three modeling domains, with an emphasis on existing work in ns-3.

### 5.1.2 BLE Simulation Models

BLE is a cornerstone of low-power short-range communication in IoT environments. It is highly optimized for energy efficiency and supports key configuration parameters that allow developers to tune the trade-off between latency, throughput, and power consumption[15].

Despite its prominence in real-world IoT systems, BLE lacks native support in ns-3. The most significant contribution in this area comes from a community-developed module introduced by Stijn Geysen in 2018 [16]. This implementation includes a basic BLE stack inspired by Bluetooth 4.2 and supports physical and MAC layer abstraction, including configuration of fundamental parameters such as Connection Interval and Slave Latency, which are critical for energy efficiency tuning [17].

However, this module has the following limitations:

- It does not support BLE's higher protocol layers such as the ATT and GATT.
- It lacks modulation modeling, particularly the Gaussian Frequency Shift Keying (GFSK) used by BLE PHYs.
- Documentation is sparse, making integration and debugging challenging [16].

### 5.1.3 VLC Simulation Models

OWC, especially through VLC, is an emerging alternative to radio-frequency communication, offering high bandwidth, directional transmission, and natural interference avoidance [18]. Its applicability to IoT has gained traction in recent years, particularly for indoor environments where LED lighting infrastructure can be reused for communication [19].

Simulation support for VLC in ns-3, however, remains sparse. A notable attempt to implement VLC simulation in ns-3 was made in 2016, where researchers developed a hybrid Wi-Fi/VLC module [20]. The module includes mathematical models for bit error rate (BER) and SNR computation and simulates a basic VLC link using On-Off Keying (OOK) and Variable Pulse Position Modulation (VPPM).

Nonetheless, this module has the following architectural limitations:

- It inherits from the Point-to-Point NetDevice, a structure not suited for wireless or multi-hop scenarios.
- It lacks separation of PHY and MAC layers, violating ns-3's modular design principles [21].
- It does not implement a PHY state machine, hindering integration with the energy model [22].

### 5.1.4 Energy Harvesting and Consumption Models

Energy autonomy is a defining characteristic of the RIoT node. By using printed electronics and ambient energy harvesting (e.g., solar), the node operates without batteries or external power

sources, storing energy in supercapacitors [23]. Therefore, energy modeling is a core requirement of the DT.

The developed ns-3 energy framework [22][24] provides the foundation for simulating power dynamics in wireless devices. It consists of:

- Energy Sources: Models for batteries or supercapacitors that track remaining energy and supply voltage.
- Device Energy Models: Modules linked to communication devices that define energy consumption behavior based on device states (e.g., transmit, receive, idle).
- Energy Harvesters: Simulate ambient energy sources like solar panels and their charging behavior.

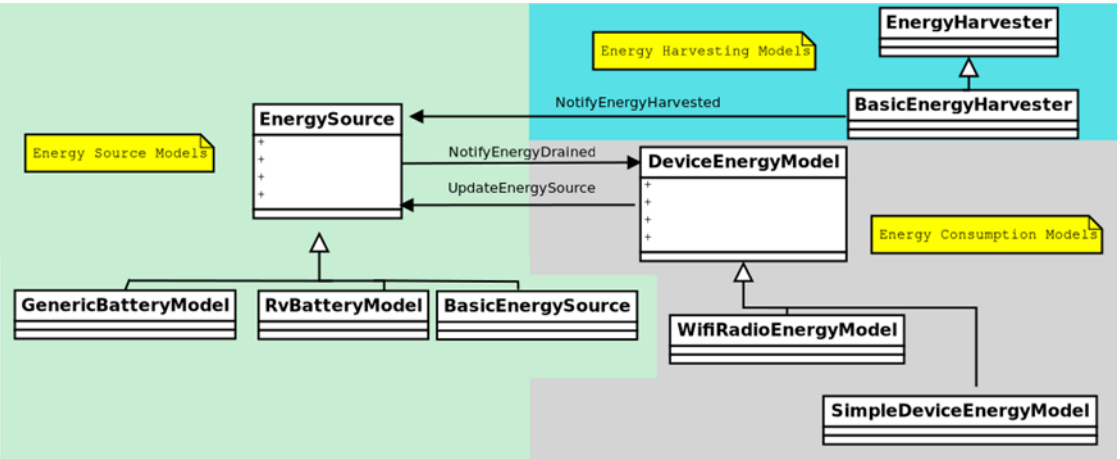



Figure 75. Overview of the energy framework developed in ns-3.

While the energy source and harvester models were adopted with minimal modifications, custom Device Energy Models were developed specifically for the BLE and OWC modules, as well as for peripheral components such as screen update and localization functions. These tailored models enable precise energy profiling and support proactive cross-layer optimization of energy consumption across diverse components.

## 5.2 Design and Implementation of the ns-3 RIoT Node - Architecture and Abstraction of the RIoT Node in ns-3

The DT architecture of the RIoT node in ns-3 was designed to serve as a high-fidelity simulation tool for testing and optimizing sustainable IoT networks. This section presents the digital abstraction of the RIoT node and its implementation in ns-3, supporting use cases such as network planning, multimodal handover, and energy-aware operation.

### 5.2.1 High-Level Digital Abstraction of the RIoT Node

The RIoT node architecture was developed by translating the physical node's structure into ns-3 components. The core is the `ns3::Node` class, which contains modules for communication interfaces, mobility, and energy behavior.

Our modular architecture is detailed in Figure 76, in which:

- Communication interfaces (BLE and OWC) are each implemented as `NetDevice` subclasses.
- A custom `OrientationAwareMobilityModel` provides realistic modeling of directional optical links.
- Energy-related behavior is modeled using `DeviceEnergyModel` classes attached to a central `BasicEnergySource`.

- A placeholder module allows the integration of optimization algorithms.

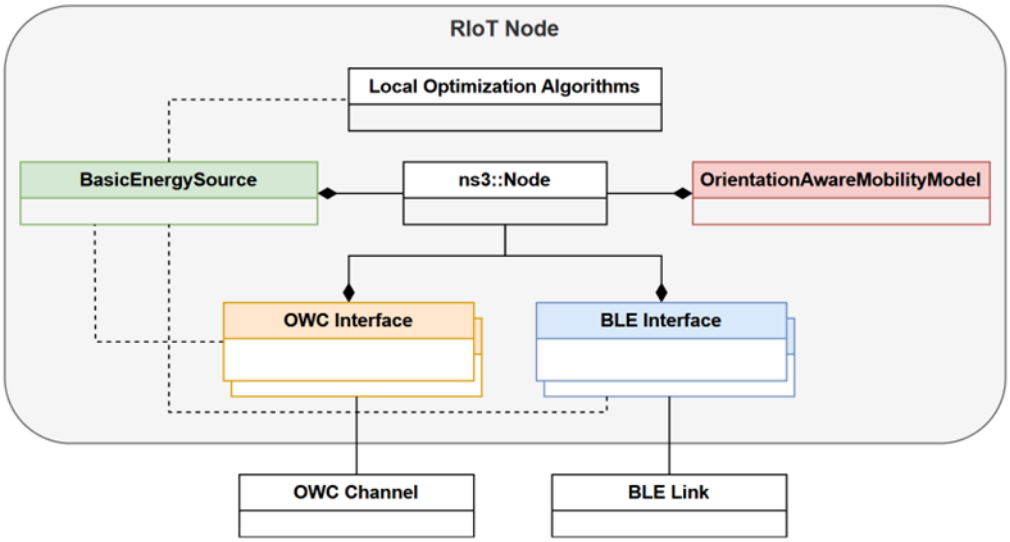



Figure 76. The developed architecture of the RIoT node in ns-3.

### 5.2.2 Hybrid RF/Light Network Support

The RIoT node supports hybrid communication using BLE and VLC. Both interfaces can operate simultaneously, enabling full-duplex uplink and downlink communication. BLE provides bidirectional radio communication, while the OWC interface operates with visible light for downlink and infrared for uplink.

The BLE module was ported from a community-developed implementation and updated for ns-3.40. It includes a stable MAC/PHY structure and supports traffic generation at the MAC level. Unnecessary protocol layers such as device discovery were excluded to maintain simplicity and focus on periodic data exchange.

The OWC module was developed from the ground up to overcome architectural limitations identified in prior implementations. Key features include:

- Structured PHY and MAC layers.
- Directional optical channel modeling based on physical light propagation.
- Full integration with ns-3's energy framework.

### 5.2.3 BLE and VLC Model Alignment with Physical RIoT Nodes

The BLE and OWC models in ns-3 were developed to align closely with real RIoT hardware:

- The BLE model supports GFSK modulation and incorporates state-specific energy consumption for transmission, reception, and idle states.
- The OWC model allows configuration of optical properties such as Tx power, semi-angle, and photodetector field-of-view, while also simulating BER and SNR using realistic physical models.

Each protocol stack was modularized according to ns-3 design practices, ensuring clean separation of layers and enabling future extensions.

### 5.2.4 Energy Modeling

Energy awareness is central to the RIoT node simulation. The energy model includes three core components:

1. Energy Buffer (Source): A `BasicEnergySource` is used to simulate energy storage in a capacitor or battery, constrained by a defined maximum capacity in joules.

2. Energy Harvesting: Energy input is modeled through an `EnergyHarvester`, simulating photovoltaic charging with customizable input power profiles.
3. Energy Consumption: Custom `DeviceEnergyModel` classes for BLE and OWC define instantaneous power usage for each device state. These can be configured using either:
    a. A constant current model with static draw values.
    b. A linear model where current draw varies with Tx power, baud rate, and packet size.

This modeling framework enables dynamic tracking of node energy use and supports real-world behaviors such as energy starvation and charging cycles.

### 5.2.5 Integration with Real RIoT Data, MQTT and Optimization Loops

The developed architecture supports future integration with real-world node telemetry and optimization feedback loops:

- The `LocalOptimizationAlgorithms` module allows implementation of adaptive control strategies based on sensed energy or channel conditions.
- The mobility model can be synchronized with external data sources, such as trajectory datasets.

This closed-loop simulation capability supports online algorithm testing, where ns-3 simulations and physical node deployments interact in real time.

The current framework supports structured MQTT interaction integrated with ns-3's real-time simulation mode. It can both consume and generate JSON-formatted MQTT messages, formatted according to Section 4.6.2.4, allowing bidirectional communication between simulated nodes and an external MQTT broker. Incoming messages are parsed using the `nlohmann::json` library to extract protocol metadata, message headers, and payload fields such as sensor type and value.

Leveraging ns-3's `RealtimeSimulatorImpl`, the simulation runs in real time, making it suitable for integration with the MQTT server. This framework provides a foundation for advanced features such as remote configuration and status monitoring.

The simulation subscribes to the MQTT topics related to node and AP positioning and reflects those changes inside the simulation. The simulated nodes can also report their presence via MQTT.

### 5.2.6 Calibration and Realism

The RIoT node model is designed to be calibrated using empirical data:

- Energy model parameters could be accurately derived from real-world measurements of BLE and OWC hardware, utilizing energy and power data collected directly from the target platforms, as detailed in Chapter 3.
- BER/SNR models can be validated against controlled lab measurements of RF and optical links.
- End-to-end communication cycles and energy consumption patterns can be matched to hardware behavior.

This ensures the simulation behaves as a faithful DT, ready to support high-level tasks such as T3.4.2 and T3.4.3, including network testing, planning, and optimization of multimodal communication strategies.

### 5.2.7 BLE Communication Module

The BLE communication module implemented in ns-3 models key aspects of BLE operation in RIoT nodes. The goal was to represent real-world BLE behavior while maintaining modularity, energy traceability, and compatibility with other subsystems.

Two PHY modes were implemented: 1 Mbit/s and 2 Mbit/s. These influence both bandwidth and energy consumption. The 2 Mbit/s mode provides faster transmission but requires a higher SNR, reducing the effective range.

**Effective Range per PHY Rate:**

Although empirical range measurement was not automated, the simulator uses propagation models (e.g., Friis) that, combined with the PHY rate selection, reflect shorter effective ranges at 2 Mbit/s due to the reduced link margin in Figure 77.

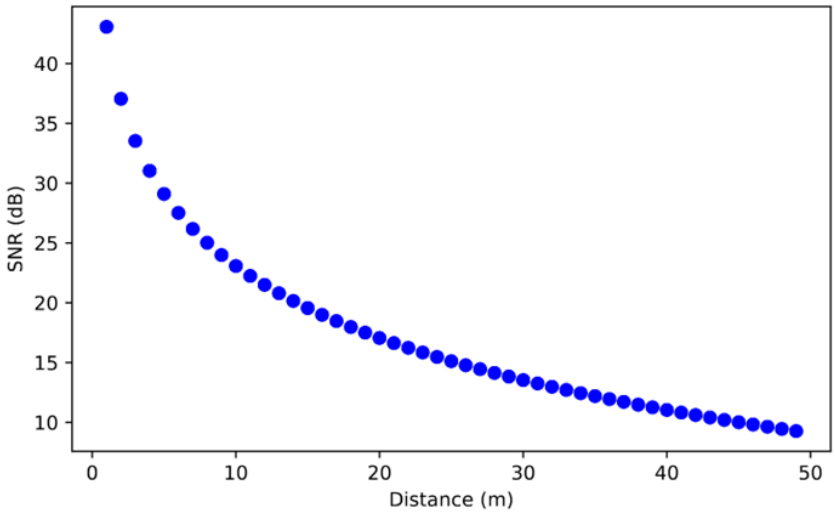



Figure 77. SNR versus distance using the Friis Propagation Loss Model

**Independent PHY Rate Selection:**

Each BLE node can be configured to use a different PHY mode, enabling simulation of mixed-speed networks. This flexibility allows modeling of adaptive rate strategies or heterogeneous hardware.

**Modulation and Error Models (GFSK, Offset Quadrature Phase Shift Keying (OQPSK)):**

GFSK, the modulation scheme used in BLE, was not previously available in ns-3 and has been newly implemented as part of this work. To validate its accuracy, the resulting BER performance was compared with theoretical (MATLAB-generated) reference curves, confirming close alignment between the implemented model and expected behavior. The comparison between the theoretical and implemented BER results is shown in Figure 78.

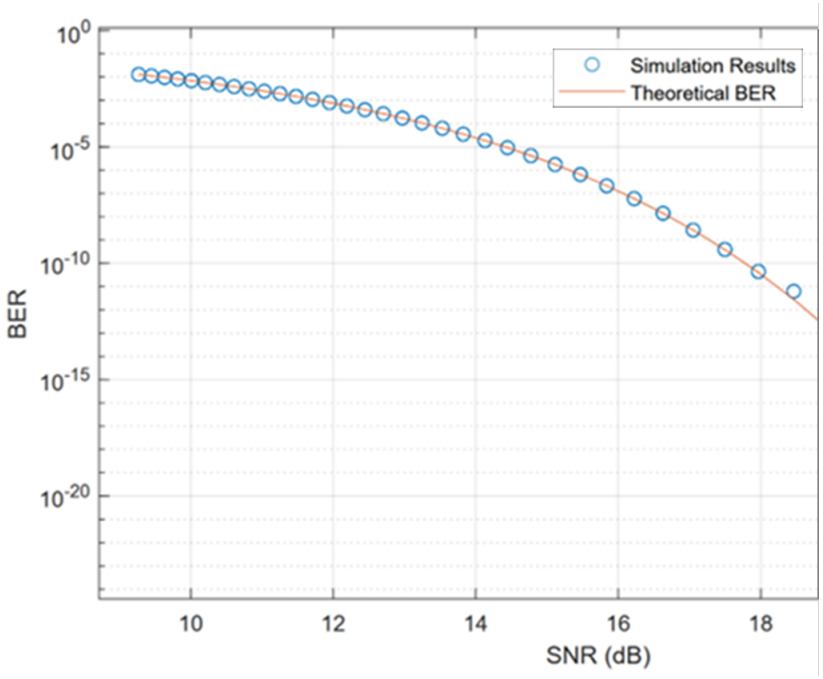



Figure 78. Simulation results from BER versus SNR using the implemented GFSK compared to MATLAB results.

**Extended Advertisements:**

To simplify implementation and reduce simulation overhead, BLE advertisement and discovery processes were omitted. All connections are established statically, which is sufficient for star-topology and low-traffic IoT scenarios.

### 5.2.8 VLC Communication Module

The developed VLC module in ns-3 simulates bi-directional optical wireless communication by modeling separate spectral bands and distinct physical-layer characteristics for transmission and reception.

**Downlink and Uplink Design (TDMA, IR Spectrum):**

The system is designed for full-duplex communication (as shown in Figure 79). Downlink is handled via visible light (e.g., white LEDs), while uplink transmissions use infrared (IR). A basic TDMA scheme, used by VLC, manages medium access by assigning time slots for transmission and reception, preventing overlap and collisions.

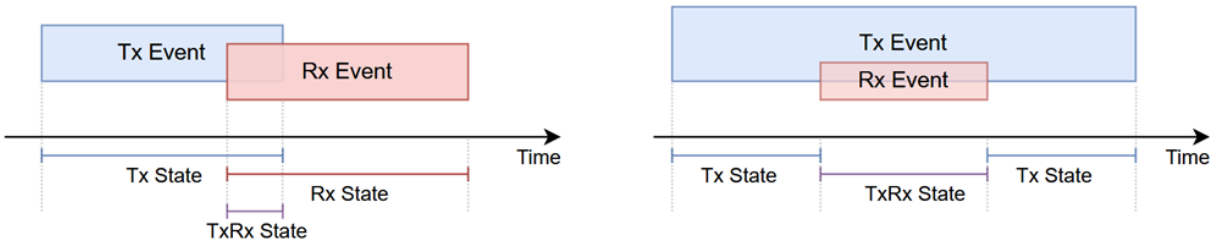



Figure 79. Special cases of the Full-duplex operation in the OWC module.

**Collision Avoidance and Shared Medium Enhancements:**

Due to the directional nature of VLC, spatial separation serves as an inherent collision mitigation strategy. Further enhancements include:

- Use of different spectral bands for uplink/downlink.

- Orientation-aware mobility to simulate changing FoV and LOS.
- TDMA-based scheduling to avoid temporal collisions.

To further support this design, Figure 80 presents a study of the SNR as a function of distance and incidence angle. The results demonstrate how optical alignment and positioning significantly influence link quality.

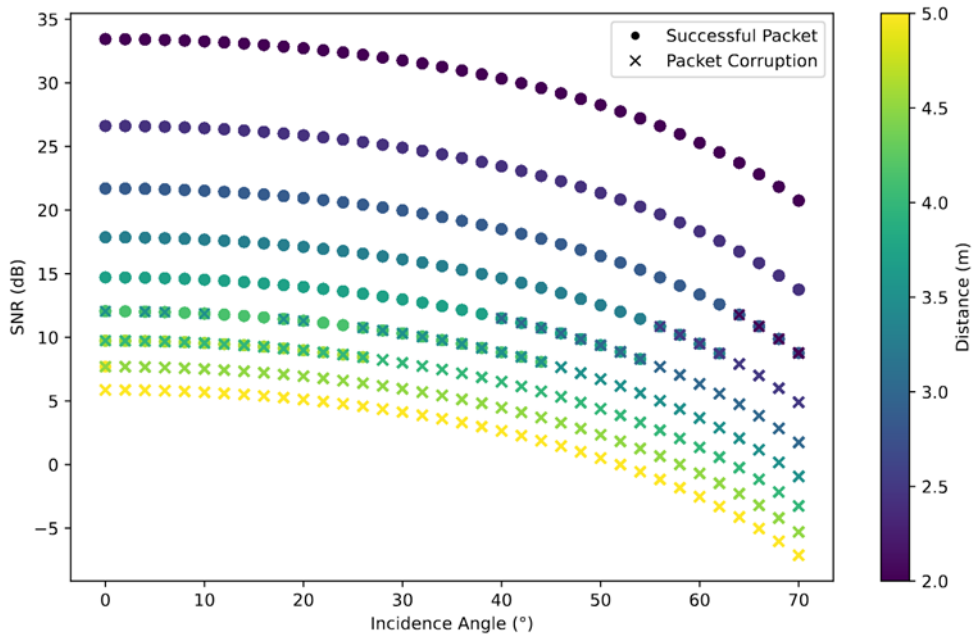



Figure 80. Relation between incidence angle, SNR, and distance, with markers indicating packet reception success for the wavelength of 1300 nm

Figure 80 illustrates the variation of SNR with incidence angle and distance in a VLC link. The results show that SNR—and thus packet reliability—decreases with both increased distance and misalignment, highlighting the sensitivity of optical channels to geometric orientation. This behavior supports the collision avoidance strategy, as the directional nature of VLC inherently limits interference between spatially separated links. Combined with spectrum separation, orientation-aware mobility modeling, and TDMA scheduling, these findings reinforce the design of robust shared-medium management for VLC networks.

The PHY model also incorporates parameters like LED semi-angle, photodetector field-of-view (FoV), and radiation pattern using Lambertian emission. The full details and implementation can be found in [25].

### 5.2.9 Energy Models

To realistically simulate RIoT node behavior, three categories of energy models were developed in ns-3:

1. **Energy Consumption Models:**

Custom `DeviceEnergyModel` classes were created for BLE and VLC interfaces. These models track current draw per PHY state (TX, RX, idle). The architecture supports:

- Constant current mode for simple profiling
- Linear models where energy consumption varies with transmission power, baud rate, and activity

For processing units and accessories (e.g., sensors, E-Ink displays), placeholder models estimate energy draw for computation, localization, and display refresh cycles.

2. **Energy Harvesting Mechanisms:**

The node supports energy harvesting via light (solar photovoltaic). A harvester model periodically injects energy into the `BasicEnergySource`, simulating ambient charging based on configurable power curves.

3. **Energy Storage (Buffer):**

The energy buffer simulates a supercapacitor or battery with a configurable maximum capacity in joules. It tracks depletion based on device activity and charges from harvesters.

OWC exhibits a steep linear increase in energy consumption, exceeding 0.45 J before stabilizing, whereas the BLE maintains consistently low energy usage, remaining below 0.05 J throughout. The harvested energy increases very slowly and remains significantly below the consumption of either interface.

To improve fidelity, real energy data from RIoT hardware measurements was used as a lookup table, providing accurate current values for each state. The values were derived from experimental measurements detailed in Chapter 3 of this deliverable and in Chapter 3 of Deliverable D2.4.

### 5.2.10 Integration with BLE and VLC Modules

**State Machine Implementation:**

Each communication interface is modeled with a finite state machine (FSM) that controls its current state (e.g., transmit, receive, idle, off). Transitions between states trigger energy consumption updates and propagate events to the mobility and application layers. The state machine diagram is presented below in Figure 81. It includes six main states: OFF, SLEEP, IDLE, TX (transmitting), RX (receiving), and TX_RX (simultaneous transmit and receive). Transitions between these states are triggered by events such as TransmitStart, ReceiveStart, TransmitEnd, ReceiveEnd, SLEEP_SIGNAL, WAKE_SIGNAL, and battery-related signals (BatteryLow, BatteryCharged).

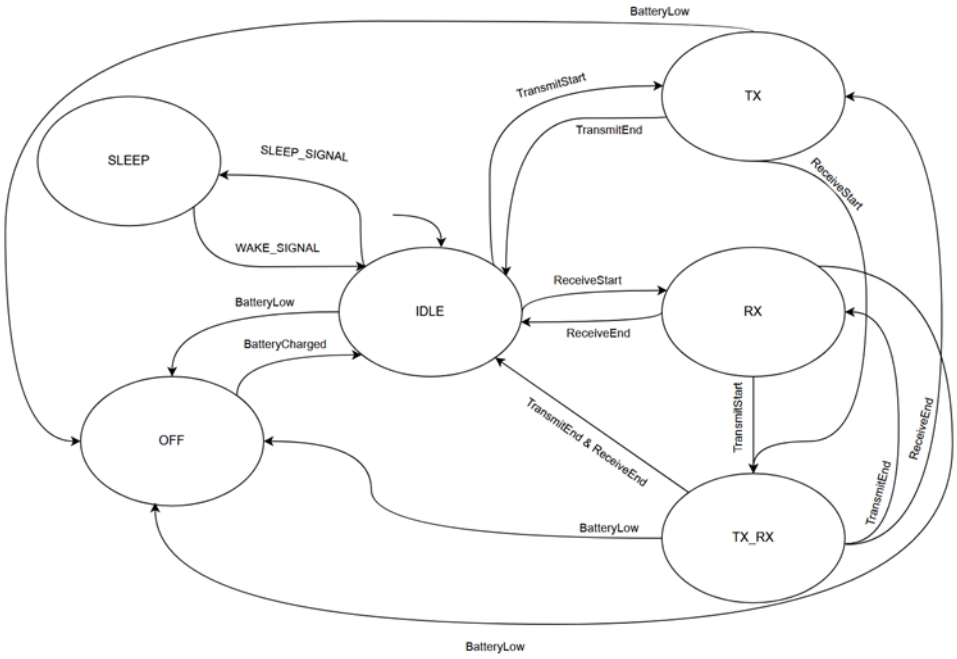



Figure 81. OWC PHY state machine

The state machine for BLE is presented in Figure 82, with the main states being IDLE, OFF TX_BUSY and RX_BUSY, controlled by a very similar set of triggers and events to the OWC state machine.

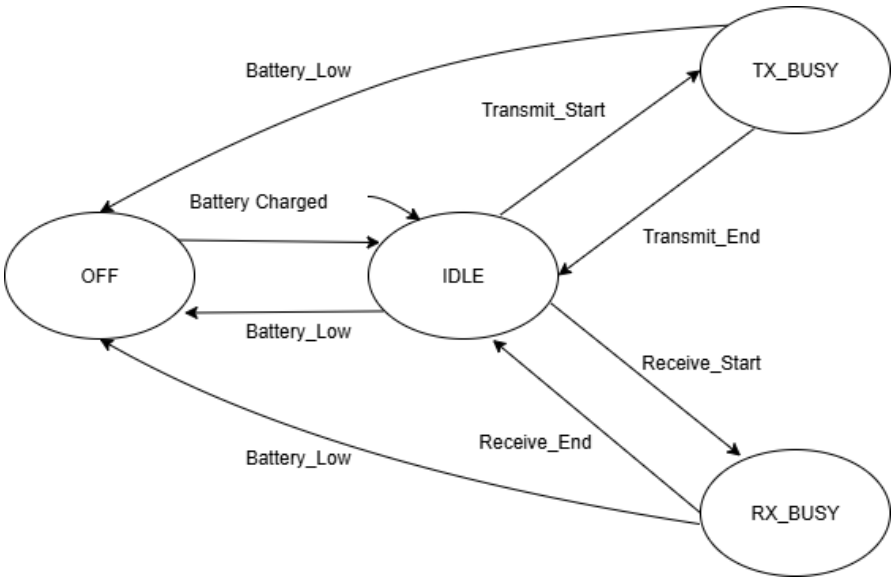



Figure 82. BLE PHY state machine

### 5.2.11 Polling-based Medium Access Control

To efficiently manage communications with energy-constrained RIoT nodes, we implement a polling-based medium access control protocol that jointly optimizes the operation of the OWC and BLE modules while adaptively supporting RIS-assisted connectivity when required. This approach enables centralized coordination of nodes' activities, ensuring both energy efficiency and scalability in a hybrid BLE/VLC communication environment. The polling-based design minimizes unnecessary power consumption by precisely scheduling node activity while dynamically allowing for controlling the RIS configuration to support the active transmitting node. In this context, the network controller can periodically poll each node, triggering its transition from a low-power sleep mode to an active state for data exchange. When not polled, nodes remain in an ultra-low-power or dormant state, thereby conserving energy and extending their operational lifetime.

To further enhance connectivity, especially for nodes situated at the boundaries of coverage or within signal-degraded zones, RIS can be used, and their control can be supported within our simulation environment. These surfaces can dynamically reflect or redirect polling signals and data transmissions, effectively mitigating signal loss and improving link reliability in obstructed or non-line-of-sight (NLoS) environments. Thus, by combining RIS-assisted coverage extension with polling-based scheduling, the system can ensure reliable, low-latency communication while maintaining a minimal energy footprint across the network.

## 5.3 Network Simulations and Optimization

### 5.3.1 Monitoring Traffic, Usage Patterns, and Network Conditions

A set of core variables has been defined to monitor the performance, energy status, and operational behavior of RIoT nodes, serving as the foundation for learning traffic and usage patterns. These variables act as inputs to the cross-layer optimization logic, enabling each node to anticipate changing network conditions and adapt its configuration proactively to optimize energy use while maintaining QoS.

The monitored variables include:

- SNR: Used to assess channel quality and influence decisions related to modulation, interface selection, and transmission parameters.

- Node Location: Enables directional optimization (e.g., VLC alignment) and supports mobility-aware resource allocation.
- Packet Loss Ratio (PLR): Indicates the reliability of communication and informs fallback strategies or interface switching logic.
- Remaining Energy in Buffer: Expressed in voltage or energy (Joules), this value is key to determining whether a node can remain active, and at what duty cycle.
- Transmission Power (TX Power): Acts both as a measurable status and as a tunable parameter for managing energy and connectivity trade-offs.
- Modulation and Coding Scheme (MCS): Helps optimize data rate and energy usage by adjusting physical layer characteristics.
- Active Interface(s): Identifies whether BLE, VLC, or both are active, supporting multimodal access management.
- Peripheral Activity (e.g., screen, localization): Used to account for non-communication energy loads, which are factored into energy balancing models.

These parameters are selected for their influence on node longevity, link quality, and application-level responsiveness.

#### *5.3.1.1 Identification and Ranking of Variables*

Each monitored variable contributes differently to performance and energy optimization, carrying specific weight based on its impact on system behavior. High-priority variables, such as remaining energy, SNR, and PLR, are directly linked to node survivability and network reliability. These are monitored more frequently and drive time-critical decisions such as interface switching, adaptive duty cycling, or power control. Medium- and low-priority variables, including node location or peripheral activity, influence longer-term or scenario-specific decisions. For example, location data can be used for spatially aware channel reuse or RIS configuration, while screen activity may influence power budgeting in nodes with limited energy stores.

The prioritization of these variables aligns with three main objectives:

- QoS assurance: maintaining reliable communication links.
- Energy efficiency: extending operational lifetime.
- Operational flexibility: enabling rapid adaptation to dynamic network conditions.

The developed optimization algorithm leverages these weighted inputs to trigger context-aware actions, such as peripheral management, adaptive duty cycling, or selecting the most suitable communication modality. This enables each RIoT node to respond intelligently to variations in traffic, usage patterns, and environmental conditions.

#### *5.3.1.2 Status Information Gathering*

Status information is gathered at varying frequencies, depending on the criticality of the variable and the constraints of the node. While some variables are updated on a per-packet basis (e.g., SNR, PLR), others follow configurable sampling periods or are event-triggered when thresholds are crossed, or changes are detected.

#### *5.3.1.3 Traffic and Usage Pattern Profiling*

Building on the monitored variables, historical observations, and environmental dynamics, our DT can perform proactive reconfiguration actions, such as adjusting standby durations, transmission power, modulation schemes, or access technologies (e.g., switching between BLE and VLC). The monitoring frequency is adaptive, depending on the node's role, available energy, and environmental variability; some parameters are sampled periodically, while others are event-driven or aggregated over defined time windows.

To enable context-aware optimization, the developed DT continuously monitors each RIoT node's traffic behavior and builds lightweight statistical profiles capturing variations in data rate and activity over time. Each node maintains a profile that records its transmission history, including instantaneous, average, and variable rates. The *ProfileTraffic* function periodically samples transmitted packets and classifies traffic into Low, Medium, or High levels based on recent rate statistics, allowing the system to recognize changes in activity and associate them with operational contexts, such as idle, sensing, or high-transmission states. The *AdaptDataRate* function then adjusts each node's transmission rate according to its traffic level, smoothing transitions to prevent abrupt changes and constraining rates between 20 kb/s and 100 kb/s. Nodes in sleep mode are assigned a zero rate, while adaptation intervals dynamically vary from one to five seconds based on traffic variability. Finally, the *UpdateDataRates* function applies periodic network-wide updates every 50 seconds, cycling between predefined rates to maintain synchronization and ensure that node-level adaptations stay within overall system limits.

Figure 83 illustrates the operation of the traffic profiling mechanism. When a node enters the sleep state, its transmission rate drops to zero. Upon waking, its target data rate is set according to the system's scheduled configuration. Once transmission begins, the adaptation algorithm monitors the changes in data rate and classifies them into traffic levels, low, medium, or high, based on deviations from the recent historical average. During the first wake-up (around 50 s), the target rate is 100 kb/s, and the node's transmission gradually converges to this value. When the node returns to sleep mode, the rate drops back to zero. During the second wake-up (around 125 s), the target rate is lower (20 kb/s), and the algorithm reaches this target more rapidly, reflecting the reduced throughput demand. This profiling and adaptation mechanism provides a lightweight yet effective method for learning node-specific traffic dynamics within the ns-3-based DT. It enables reproducible evaluation of traffic-aware adaptation strategies and supports predictive control within the Energy-aware Utility-based Node Optimization (EUNO) optimization framework, allowing nodes to anticipate workload variations and proactively adjust communication parameters for improved energy efficiency and performance balance.

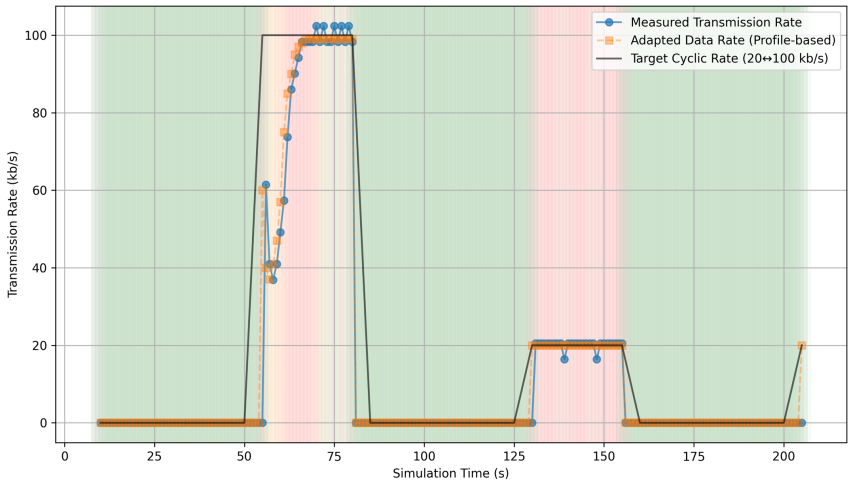



Figure 83. Transmission rate adaptation of an RIoT node over time, illustrating the transition between sleep, high, and low traffic levels.

### *5.3.1.4 Prediction Model for Network Conditions*

In addition to traffic profiling, our DT integrates an Exponentially Weighted Moving Average (EWMA)-based predictor to estimate short-term variations in network quality. This model analyzes the time-series behavior of the received SNR to compute the probability $p_m(t)$, which quantifies the likelihood of node movement or channel variations at time $t$. The resulting probability serves as a mechanism to anticipate network condition degradation with reasonable certainty, allowing the system to proactively allocate resources before a complete link failure occurs. It enables the system to predict potential connectivity loss or performance degradation and to reconfigure communication modalities or localization activities accordingly. By embedding this predictor in the ns-3 DT, network condition forecasting can be tested and refined in a closed-

loop virtual environment, minimizing real-world risks. This integration enhances the robustness of the RIoT system by enabling proactive energy and connectivity management while maintaining low computational complexity. Further details on the predictor design and its integration within the proposed optimization framework can be found in [26].

Figure 84(a) illustrates an example of the raw SNR time-series variations together with the EWMA-based SNR used as a baseline for detecting channel variations. The instantaneous SNR fluctuates due to mobility effects, while the EWMA filter effectively suppresses short-term fading and preserves the long-term channel trend. This demonstrates the predictor's ability to maintain a stable reference level, which is essential for identifying meaningful mobility, induced deviations. Figure 84(b) shows the resulting probability $p_m(t)$, obtained by mapping the absolute deviation between the EWMA baseline and the instantaneous SNR onto a sigmoid function. The behavior of $p_m(t)$ closely follows the rate of SNR deterioration: during periods where the SNR decreases gradually, the deviation remains small and the predicted mobility probability stays low. However, when the channel quality drops sharply, representing a significant change in node position or obstruction, the deviation increases and $p_m(t)$ rapidly rises toward one. This behavior confirms that the predictor is sensitive to large variations in channel quality while remaining robust to short-term noise.

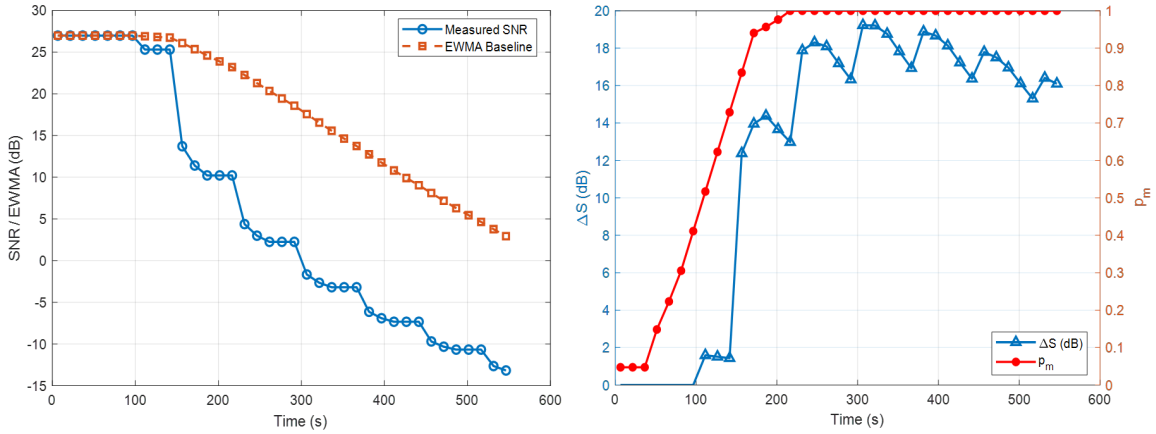



Figure 84. (a) Instantaneous and EWMA-based SNR over time. (b) Corresponding channel variations probability $p_m$ derived from SNR deviation ΔS.

### *5.3.1.5 Actions for Energy Optimization and QoS Assurance*

Based on the collected traffic patterns and environmental conditions, several configuration actions can be executed to maintain or improve network performance and energy sustainability. These actions include:

- Wake-up and Sleep Scheduling: Nodes can enter low-power sleep states or be woken up based on energy levels, traffic demand, or environmental triggers.
- Transmit Power Adjustment: Lowering or increasing TX power to balance energy usage and connectivity strength based on link conditions.
- MCS Selection: Switching to more robust or efficient PHY settings depending on channel quality and data rate needs.
- Interface Selection and Activation: Dynamically enabling or disabling BLE or VLC interfaces to optimize energy or latency per application flow.
- Peripheral Management: Controlling the activation of non-communication components such as screens or localization units to reduce energy draw.
- Update Period Adjustment: Tuning the reporting or sensing intervals to match energy budgets or QoS constraints.

These actions form the core of the simulation-driven optimization loop. By simulating these changes within the ns-3 DT, their effects on performance and energy consumption can be evaluated before deployment in real RIoT nodes.

### 5.3.2 Proactive Cross-Layer Optimization Technique

For sustainable and energy-efficient operation of RIoT nodes, we propose the EUNO algorithm. This algorithm adopts a heuristic decision-making framework that enables each node to autonomously select the most appropriate operational action by evaluating a unified utility function. This function quantifies the expected benefit of every potential action based on the node's current state, residual energy, traffic pattern, and network conditions in relation to the action's proposed settings, which is composed of modality, which can be OWC or BLE, and operational mode, which can be Performance, Conservation and Sleep. Moreover, to evaluate this developed optimization algorithm, a configurable simulation environment was implemented using ns-3. This simulation is composed of multiple nodes arranged according to a star topology, with a central node (gateway) and multiple RIoT nodes. The RIoT nodes operate with limited energy reserves and communicate exclusively with the gateway.

To define different traffic and usage patterns, each node runs a custom application specifically developed for the purposes of this evaluation. This application transmits continuously packets between a pair of nodes at a fixed data rate using OWC or BLE, the e-ink screen and the localization module (responsible for the localization features), adjusting their usage (choosing modality, disabling interaction, etc.) to better control the traffic between the connected nodes. All the optimization actions are performed on top of this application. Three distinct operational modes have been also defined for the RIoT nodes: Performance Mode, Conservation Mode, and Sleep Mode. In Performance Mode, the node operates under normal conditions without restrictions on energy consumption or communication capabilities, ensuring maximum performance and responsiveness. In Conservation Mode, the node maintains reception functionality but limits its transmission rate to preserve energy. It also temporarily disables non-essential components such as screen updates and localization features. This mode is activated when the node's remaining energy is low but not yet critical, allowing continued operation while preventing premature battery depletion. In Sleep Mode, the node suspends all communication and sensing activities until a higher operational mode is reactivated. This mode is employed when the node's energy reaches a critical threshold or when communication is unnecessary for a defined period, thereby minimizing power usage and extending overall network lifetime. Accordingly, in performance mode, the node uses the modality which exhibits the highest SNR, which allows for a higher data rate and high energy levels. In the conservation mode, the node chooses the most energy efficient modality, overlooking transmission rates, and focusing on saving energy.

Beyond the components already mentioned, the nodes are also equipped with an energy harvester that retrieves energy at a fixed rate, allowing nodes to regain energy during sleep periods. The values of the SNR, packet loss, and such are simulated by considering the nodes' positions, distance, and orientation, through the ns-3 framework.

The proposed EUNO algorithm compared against a baseline approach, namely the Energy-aware Threshold-based Node Optimization (ETNO) algorithm. This algorithm takes its actions based on the current energy level, changing state and modality after the energy level reaches predefined thresholds. Our implementation uses two thresholds: the *sleep threshold*, which characterizes the energy level under which the node goes automatically to sleep, and the *conservation threshold*, under which a node in the performance mode enters conservation mode and above which any sleeping node resumes its functionality.

The EUNO algorithm takes a more advanced approach. It defines a unified utility function, according to the current properties of the node and the application at the time the node optimization is run. This utility takes into consideration the requirements of the application over the next optimization period, such as consumed energy, needed transmission rate, screen update, and localization.

The utility function is formulated as a weighted combination of multiple sub-utilities, each representing different aspects of the node's behavior.

- Modality Utility: This utility represents the relationship between the selected communication modality and its corresponding transmission rate. It is inherently energy-dependent, favoring the modality with the highest SNR when operating in performance mode with sufficient energy reserves, and switching to the most energy-efficient modality when the node enters conservation mode.
- Screen Utility: Assigns higher utility to actions that activate the screen when increased interaction is required by the application and deactivate it otherwise to conserve energy.
- Localization Utility: This utility assigns a higher value to actions that enable localization when required, and to those that disable it when localization updates are unnecessary.
- Energy Cost: This utility penalizes high-power-consuming actions by assigning them lower utility values, thereby preventing the node from selecting actions that could deplete its energy within the next optimization period.

Afterwards, the final utility value is given as a weighted sum of the previous utilities.

$$U(a) = p_{Mod} ModUtility + p_{Screen} ScreenUtility + p_{Loc} LocUtility + p_{EnergyCost}(t) EnergyCost$$

This algorithm is also subject to key constraints, such as not allowing any non-sleep action when the remaining energy level falls below a critical threshold. The main steps of the EUNO algorithm are summarized in Figure 85. At each decision step, the node updates its local state, predicts network dynamics using an EWMA-based estimator, computes utilities for modality, screen, localization, and energy, and selects the optimal action maximizing the total utility. This adaptive process allows each node to autonomously balance energy sustainability and communication performance under varying network conditions. Detailed definitions of all parameters are provided in [26].

**Algorithm 1** EUNO: Energy-aware Utility-based Node Optimization

**Require:** Action set $\mathcal{A} = \{(\mathcal{S}, \mathcal{M})\}$ where $\mathcal{S} \in \{P, C, S\}$ (Performance, Conservation, Sleep) and $\mathcal{M} \in \{\text{OWC}, \text{BLE}\}$; switching penalty $p_{ch}$; critical energy threshold $f_c$; residual energy fraction $f_r$; EWMA parameters $(\lambda, k, C)$; thresholds $(\theta_s, \theta_l)$.

**Initialization:** Initialize system and predictor.
**for** each decision step $t_n$ (duration $\Delta$) **do**
  Collect all monitored variables (update SNR, residual energy, and traffic metrics).
  Update mobility predictor and predict link degradation or node mobility using EWMA.
  Determine if screen or localization should be active.
  Compute dynamic energy weight:

$$p_E(t_n) \leftarrow 1 - \frac{f_r - f_c}{1 - f_c}$$

  ▷ Increase energy weight as residual energy decreases.
  **if** $f_r < f_c$ **then**
    Enter Sleep mode under critical energy level.
    **continue**
  **end if**
  **for** each feasible action **do**
    Compute Modality Utility:

$$U_M \leftarrow f_r(p_p x_p + p_t x_t) + (1 - f_r)(p_c x_c + p_e x_e) - p_{ch} x_{ch}$$

    ▷ Trade off throughput vs. energy; penalize frequent switching.
    Compute Screen Utility:

$$U_S \leftarrow \begin{cases} \alpha, & (\mathcal{S}{=}P \wedge \mathcal{R}(s)) \text{ or } (\mathcal{S}{=}C \wedge \neg\mathcal{R}(s)) \\ 0, & \text{otherwise} \end{cases}$$

    ▷ Reward correct activation of screen updates.
    Compute Localization Utility:

$$U_L \leftarrow \begin{cases} \beta, & (\mathcal{S}{=}P \wedge \mathcal{R}(l)) \text{ or } (\mathcal{S}{=}C \wedge \neg\mathcal{R}(l)) \\ 0, & \text{otherwise} \end{cases}$$

    ▷ Trigger localization only when mobility is detected.
    Compute Energy Utility:

$$U_E \leftarrow 1 - \frac{E_a(t_n)}{E_{\max}}$$

    ▷ Reward energy-efficient actions.
    Compute Overall Utility $U(a)$.
  **end for**
  Select action with maximum total utility.

$$a^\star \leftarrow \arg\max_{a \in \mathcal{A}} U(a)$$

  Set new operating state and modality, update $a_{\text{prev}} \leftarrow a^\star$.
**end for**
**Output:** Sequence of optimal actions $\{a^\star\}$ across time.

Figure 85. Energy-aware Utility-based Node Optimization (EUNO) Algorithm.

### 5.3.3 Experimental validation under varying conditions

To evaluate the effectiveness of the proposed algorithm, we implemented it in our simulation environment configured according to the conditions described in the previous section. Most model parameters were calibrated using real measurements presented in Chapter 3. This section compares the developed optimization algorithm against the threshold-based algorithm and examines the impact of disabling nodes between transmission periods on overall system performance.

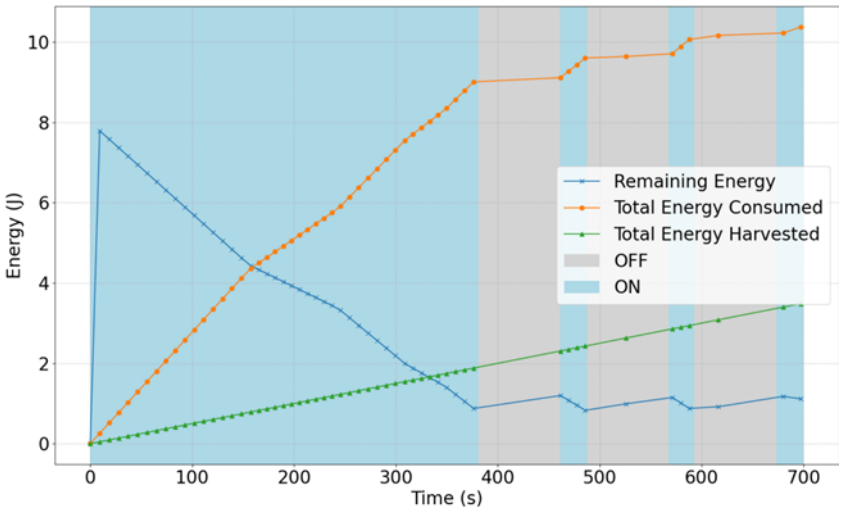



Figure 86. Energy consumption, harvesting, and remaining energy over time for a node with an energy harvester. Blue background indicates active state; gray background indicates recharging state.

First, Figure 86 illustrates an example of the evolution of energy consumption, harvested energy, and remaining energy over time for a RIoT node. In this example, the threshold-based algorithm is applied, considering OWC as the sole transmission modality. The aim of this figure is to demonstrate the importance of adaptive mode transitions. Specifically, switching the RIoT node from Performance Mode to Sleep Mode allows for preventing complete energy depletion while enabling effective energy harvesting during low-activity periods.

In the following, we present a comparative evaluation of the threshold-based and EUNO algorithms under two operational scenarios: (a) without inter-transmission sleep periods and (b) with inter-transmission sleep periods. Each run carried the simulation for 1000 seconds, plus 5 seconds for the application startup. The node topology includes three end nodes, each with a battery supporting 8 joules of energy, with transmission periods of 25 seconds assigned using a round-robin approach. Each application transmits packets of 512 bytes, at a target rate of 300 kb/s, cutting down to 60 kb/s in conservation mode. Finally, node-specific measurements, like the remaining node energy and data transmitted, were measured for the first end node created in the simulation.

For the threshold-based algorithm, the slow threshold was defined at 40% of the total energy capacity and the sleep threshold at 20%. The results with and without inter-transmission sleep periods are presented in Figure 87 and Figure 88, respectively.

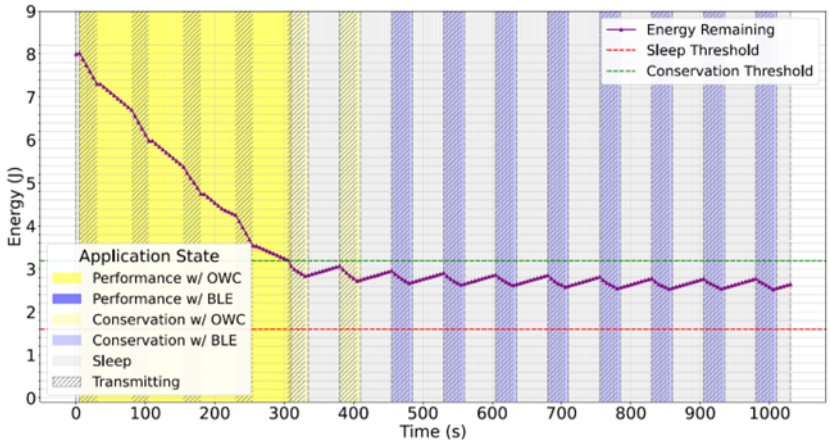



Figure 87. Evolution of the remaining energy over time for an RIoT node, under the threshold-based optimization algorithm, without inter-transmission sleep periods

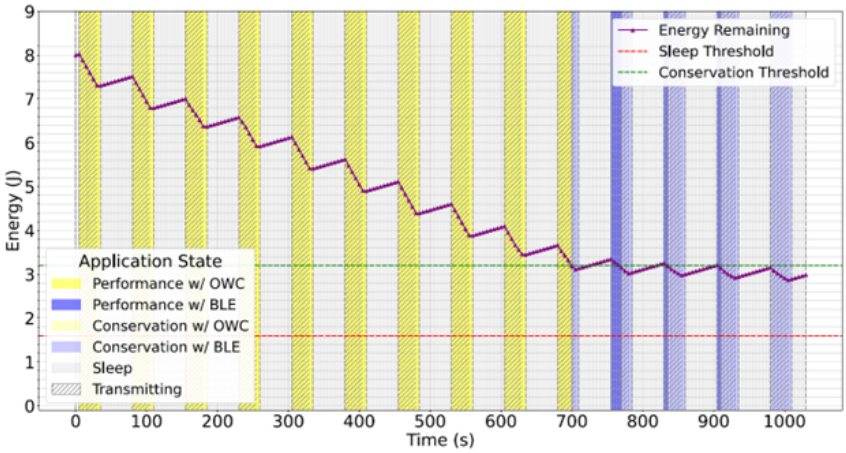



Figure 88. Evolution of the remaining energy over time for an RIoT node, under the threshold-based optimization algorithm, with inter-transmission sleep periods.

As it can be observed, in both cases, the node uses primarily OWC for transmitting data in the beginning of the simulation (which, in this case, is the modality with the best SNR), later switching to BLE once the energy hits below the slow threshold, also switching from performance to conservation mode. Nonetheless, since the node in the first situation consumes much more energy initially between transmission phases, it ends up reaching the sleep threshold much faster. In the first experiment, the node was able to transfer a total of 5.78 MB of data (average data rate of 132.1 kb/s), while in the second experiment it reached 10.72 MB (average data rate of 245.1 kb/s).

In the case of the EUNO algorithm, the sleep threshold was also set to 20%. The results with and without inter-transmission sleep periods are presented in Figure 89 and Figure 90, respectively.

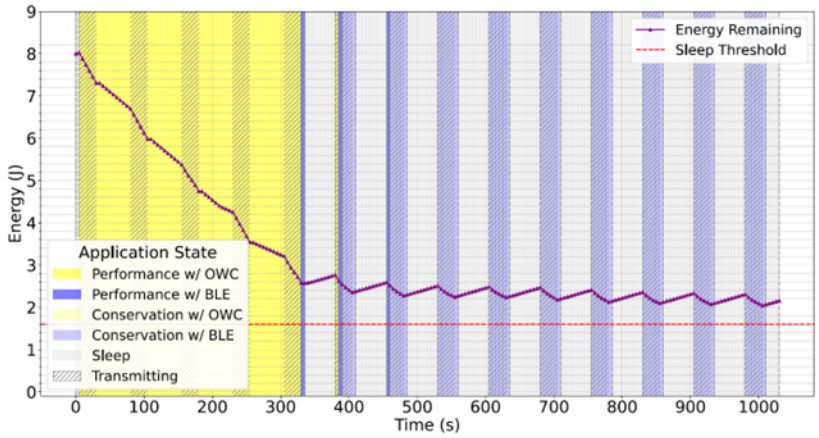



Figure 89. Evolution of the remaining energy over time for a RIoT node, under the EUNO algorithm, without inter-transmission sleep periods.

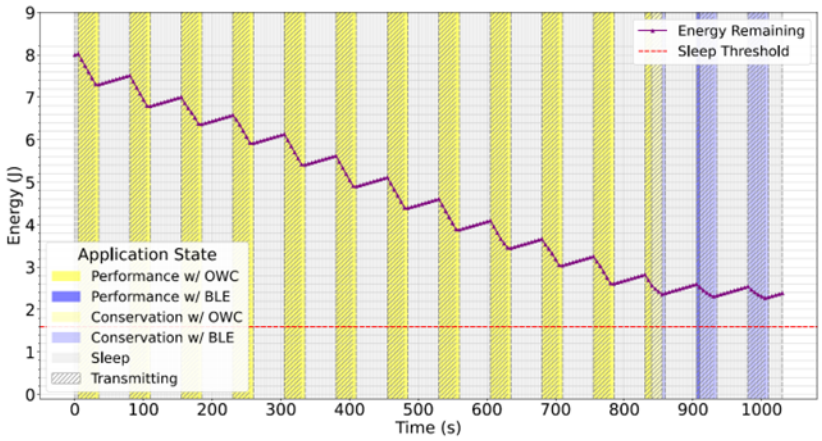



Figure 90. Evolution of the remaining energy over time for a RIoT node, under the EUNO algorithm, with inter-transmission sleep periods.

Using the EUNO algorithm, the node could switch from OWC to BLE before entering conservation mode. This proactive adaptation enables the nodes to achieve higher transmission rates before reaching the low-power threshold. Consequently, the node operating without inter-transmission sleep periods transmitted a total of 6.68 MB of data, with an average data rate of 152.6 kb/s, while the node with inter-transmission sleep periods achieved 11.32 MB, with an average data rate of 258.9 kb/s.

Finally, the aforementioned observations are further supported by the results in Figure 91, which presents the average transmission data rate achieved under different target application data rates. In this comparison, the EUNO algorithm is evaluated against ETNO and ETNO-OWC algorithms. The latter applies the threshold-based optimization algorithm (i.e., ETNO strategy), but constrained to OWC only. At low target data rates, all algorithms perform similarly, as intermittent sleep periods allow nodes to recover sufficient energy. However, as the target rate increases, energy stress begins to impact node operation. In these conditions, EUNO clearly outperforms ETNO and ETNO-OWC by maintaining a higher effective data rate while preserving energy consumption. This improvement stems from EUNO's ability to dynamically adapt both operational mode and communication modality based on residual energy and environmental conditions, achieving a more efficient trade-off between performance and energy sustainability.

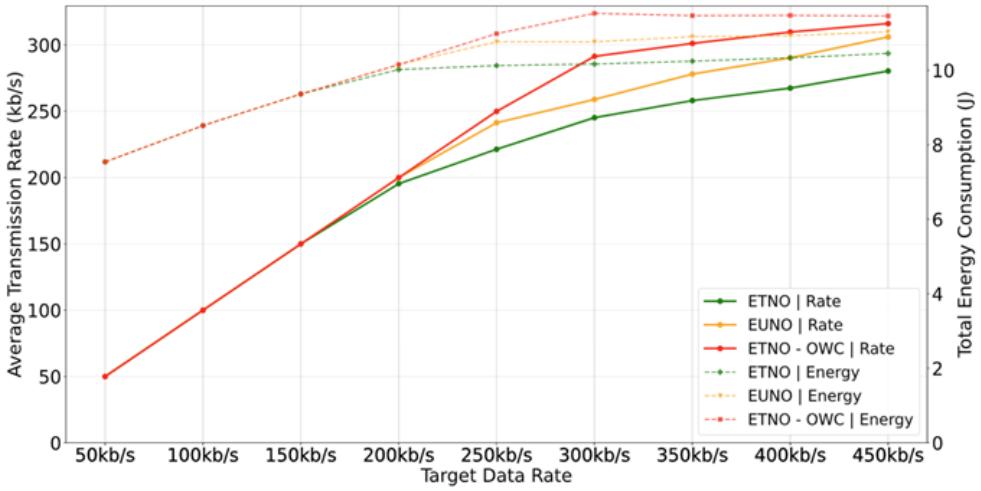



Figure 91. Average node transmission data rate as a function of the target application data rate for ETNO, ETNO-OWC, and EUNO algorithms.

## 5.4 Chapter Conclusion

This chapter presented the key components of the ns-3 RIoT DT developed to support traffic- and usage-aware simulation and cross-layer optimization strategies. This includes the development of BLE and VLC communication modules, detailed energy models, a polling-based MAC mechanism, and control interfaces via MQTT. The integration of these components enables the simulation of multimodal, energy-constrained IoT scenarios under dynamic traffic patterns and varying environmental conditions. Monitoring variables such as residual energy, SNR, and packet loss have been implemented as simulation outputs, allowing scenario-specific performance profiling.

As a result of these implementations, the simulation environment supports accurate modeling of hybrid communication behavior and energy consumption under diverse usage patterns. The BLE module supports configurable PHY rates, and the VLC module captures directionality and spectral separation for full-duplex optical links. Energy models reflect realistic per-component behavior using empirical data from physical RIoT nodes. Polling-based MAC control allows simulation of scheduling strategies that adapt to usage and traffic profiles, enabling nodes to enter sleep modes or switch communication modality proactively.

# 6 Optimization of System for Low Power Operation of IoT Nodes

The central component of the SUPERIOT system, in terms of power efficiency, is the SUPERIOT node. This node must operate autonomously and with minimal energy consumption, as it is not connected to a wired power source and has only limited access to harvested energy. A typical use case for such a node is a sustainable smart tag, particularly in advanced logistics within medical ICT scenarios. Within the node, a significant share of the harvested and stored energy is consumed by the actuator, which in this case is an e-paper display. Consequently, much of the research on energy models and optimization algorithms was dedicated to minimizing the power required for e-paper actuation. Reducing this consumption represented one of the key challenges in optimizing IoT nodes for low-power operation.

Efforts to lower the actuator's energy usage were carried out on two levels: hardware design and firmware development. On the hardware side, both theoretical and experimental studies were conducted to identify the most energy-efficient display technology. As a result, the initially considered 2.36" four-color (Red/Yellow/Black/White) e-paper display was replaced by a more energy-efficient 2.13" monochrome model, which was ultimately selected. On the firmware side, further optimization was achieved through research into optimal e-paper driving techniques, again combining theoretical and experimental approaches.

In addition to the display optimizations, substantial improvements were also made to the Demonstrator firmware, leading to a significant reduction in overall energy consumption. This further enhanced the energy efficiency and autonomy of the SUPERIOT node.

Through these combined hardware and software optimizations, an optimized, energy-efficient SUPERIOT node was developed. The process and outcomes of this work are presented in detail in the following subsections.

## 6.1 E-paper Optimization Techniques for Low Power Operation of IoT Nodes

The optimization of e-paper displays within the SUPERIOT node was carried out at both the hardware and software levels.

- **Hardware optimization** focused on selecting a display with sufficiently low power consumption by carefully evaluating the film type and ink type.
- **Software optimization** concentrated on designing optimal waveforms for e-paper driving.

An e-paper waveform is a sequence of electrical signals that control the pixels of the display, enabling them to switch states and render an image. By tailoring these waveforms, it was possible to maintain the required display quality while significantly reducing the energy consumed during display actuation.

### 6.1.1 Hardware Optimization

The primary goal of the hardware optimization was to identify the most suitable e-paper display technology for the SUPERIOT node, balancing energy efficiency with content readability. Given the strict energy constraints of the system, the focus was on minimizing power consumption while still ensuring legible visual output.

Two categories of displays were evaluated, based on the film and ink types available in the laboratory:

- A **monochrome** display with Aurora film type and monochrome ink:
  - *Model*: E2437PS0C1 4.37″ E-ink display (Pervasive Displays).
- A **color** display with Spectra film type and color ink:
  - *Model*: 4.2″ e-Paper (B) (Waveshare).

For fair comparison, the results of the evaluation were normalized to a unit area, allowing direct assessment of film and ink types in terms of energy consumption.

An overview of the E2437PS0C1 display (hereafter referred to as the Aurora monochrome type) is presented below:

| | |
|---|---|
| Outline (mm) (H x V x T) | 115.86 x 46.6 x 1.0 |
| Active Area (mm) | 104.16 x 38.192 |
| Resolution (pixel) | 480 x 176 |
| Pixel Pitch | 0.217 x 0.217 |
| Pixel Density | 117dpi |
| E ink film type | Aurora |

An overview of the 4.2″ e-Paper (B) display (hereafter referred to as the Spectra red type) is presented below:

| | |
|---|---|
| Outline (mm) (H x V x T) | 91.0 x 77.0 x 1.18 |
| Active Area (mm) | 84.8 x 63.6 |
| Resolution (pixel) | 400 x 300 |
| Pixel Pitch | 0.212 x 0.212 |
| Pixel Density | 120dpi |
| E ink film type | Spectra |

To evaluate energy efficiency, a study was conducted on two display types: the Aurora monochrome type and the Spectra red type. The goal was to determine energy consumption and refresh time required to display content using different E Ink film and ink combinations.

#### *6.1.1.1 Energy Consumption and Refresh Time*

Measurements were performed in global update mode with default waveforms. The Aurora monochrome display was refreshed using a black-and-white checkerboard. The Spectra red display was refreshed using three checkerboards: black-and-white, red-and-white, red-and-black.

#### *6.1.1.2 Measurement Software Configuration*

The measurement software communicated with the display's internal driver via the SPI interface. The sequence of operations was as follows:

1. Set the default display configuration.
2. Send the image data.
3. Trigger the display refresh command.

The software was written in C, compiled with STM32CubeIDE, and uploaded to the microcontroller via ST-LINK connected to a USB port. This setup enables uploading both monochrome and color images (black, red, white) and refreshing the displays in global update mode using default waveforms.

#### *6.1.1.3 Measurement Hardware Configuration*

E-Paper Display (EPD) Extension Kit Generation (EXT3) has been used as an extension of the NUCLEO-WL55JC1 board to connect the aurora monochrome display. A 4.2inch e-Paper Module 400x300 Rev2.1 V2 has been used as an extension of the STM32F0DISCOVERY board to connect the spectra red display.

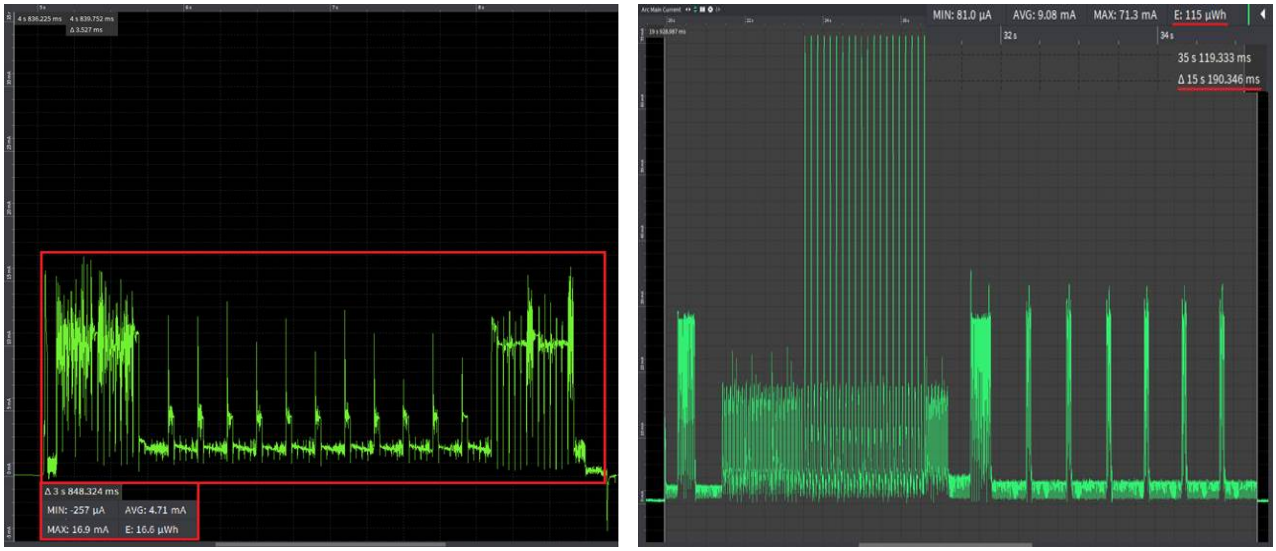
#### *6.1.1.4 Measurement Process and Worst-Case Energy Consumption of EPD*

The energy consumption was measured using the Otii Arc device. Measurements were performed with the Aurora monochrome display powered at 3.3 V and the Spectra red display powered at 3.0 V. The measurements were performed using checkerboard content, representing the worst case for energy consumption.

To evaluate worst-case consumption, a checkerboard pattern was used as the test image. The checkerboard was constructed pixel-by-pixel, ensuring maximum pixel transitions. This approach provided:

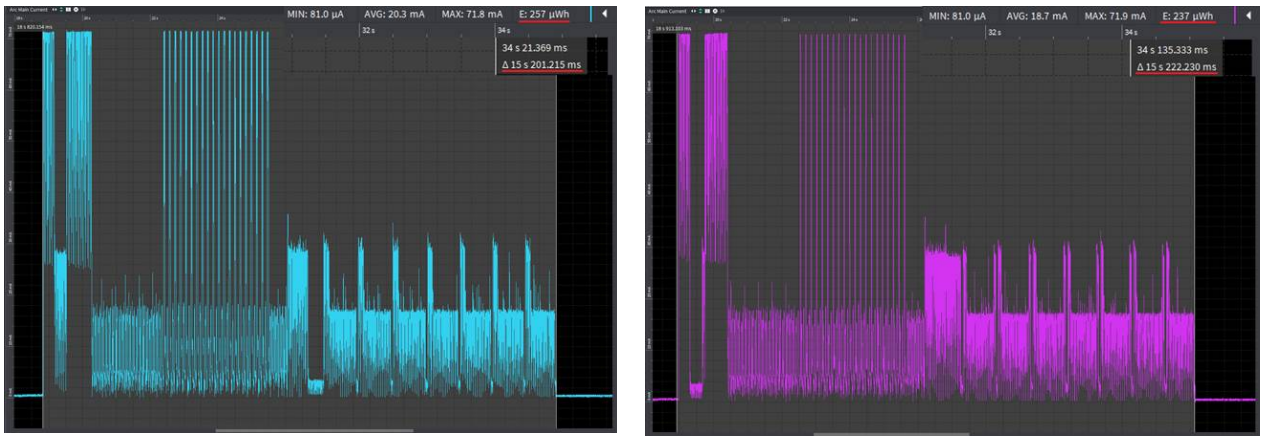
- a realistic upper bound for energy requirements,
- a valid basis for comparing displays, since both had nearly identical pixel densities (117 dpi vs. 120 dpi).

The energy consumption for the investigated displays is shown in Figure 92 and Figure 93.




(a) Aurora monochrome display (b) Spectra red display

Figure 92. Energy consumption of the Aurora monochrome display (16.6 µWh, 3.85 s) and the Spectra red display (115.0 µWh, 15.2 s) during refreshing with a black-and-white checkerboard.




(a) (b)

Figure 93. Energy consumption of the Spectra red display during refresh in 15.2 s (a) 257.0 µWh with a red-and-white checkerboard, and (b) 237.0 µWh with a red-and-black checkerboard.

#### *6.1.1.5 Measurement Results and Conclusion*

Under the worst-case scenario (checkerboard content), the measured energy consumption and refresh times were as follows:

**Aurora monochrome display (black-and-white checkerboard):**

- 16.6 µWh in 3.85 s
- Equivalent: 0.417 µWh in 97 ms per cm²

**Spectra red display:**

- Black-and-white checkerboard: 115.0 µWh in 15.2 s → 2.132 µWh in 282 ms per cm²
- Red-and-white checkerboard: 257.0 µWh in 15.2 s → 4.765 µWh in 282 ms per cm²
- Red-and-black checkerboard: 237.0 µWh in 15.2 s → 4.394 µWh in 282 ms per cm²

**Key comparisons:**

- Spectra red vs. Aurora monochrome (black-and-white):
  - 5.1× higher energy consumption
  - 2.9× longer refresh time
- Spectra red (red-and-white vs. black-and-white):
  - 2.25× higher energy consumption
  - Refresh time unchanged
- Spectra red (red-and-black vs. black-and-white):
  - 2.06× higher energy consumption
  - Refresh time unchanged
- Spectra red (red-and-white vs. red-and-black):
  - 1.08× higher energy consumption
  - Refresh time unchanged

The Aurora monochrome display demonstrated far superior energy efficiency and shorter refresh times. It was therefore selected as the optimal low-power solution, replacing the previously considered 2.36″ Spectra-based e-paper display (Red/Yellow/Black/White).

### 6.1.2 Embedded Software (Firmware) Optimization

The objective of the embedded software (firmware) optimization was to design an energy-efficient waveform for driving the selected 2.13″ monochrome e-paper display. Given the system's strict energy constraints, the waveform needed to minimize power consumption while maintaining legible display output.

The study evaluated two driving methods for the e-paper display: the default full-update waveform provided by the manufacturer, and a low-power optimized waveform refined to minimize energy consumption. The first method, the default full-update waveform (hereafter referred to as "lut_full_update"), is defined in the display's Look-Up Table (LUT) as follows:

```
unsigned char lut_full_update[]= {
  0x80, 0x4A, 0x40, 0x0,  0x0,  0x0,  0x0,  0x0,  0x0,  0x0,  0x0,  0x0,
  0x40, 0x4A, 0x80, 0x0,  0x0,  0x0,  0x0,  0x0,  0x0,  0x0,  0x0,  0x0,
  0x80, 0x4A, 0x40, 0x0,  0x0,  0x0,  0x0,  0x0,  0x0,  0x0,  0x0,  0x0,
  0x40, 0x4A, 0x80, 0x0,  0x0,  0x0,  0x0,  0x0,  0x0,  0x0,  0x0,  0x0,
  0x0,  0x0,  0x0,  0x0,  0x0,  0x0,  0x0,  0x0,  0x0,  0x0,  0x0,  0x0,
  0xF,  0x0,  0x0,  0x0,  0x0,  0x0,  0x0,
  0xF,  0x0,  0x0,  0xF,  0x0,  0x0,  0x2,
  0xF,  0x0,  0x0,  0x0,  0x0,  0x0,  0x0,
  0x1,  0x0,  0x0,  0x0,  0x0,  0x0,  0x0,
  0x0,  0x0,  0x0,  0x0,  0x0,  0x0,  0x0,
  0x0,  0x0,  0x0,  0x0,  0x0,  0x0,  0x0,
  0x0,  0x0,  0x0,  0x0,  0x0,  0x0,  0x0,
  0x0,  0x0,  0x0,  0x0,  0x0,  0x0,  0x0,
  0x0,  0x0,  0x0,  0x0,  0x0,  0x0,  0x0,
  0x0,  0x0,  0x0,  0x0,  0x0,  0x0,  0x0,
  0x0,  0x0,  0x0,  0x0,  0x0,  0x0,  0x0,
  0x22, 0x22, 0x22, 0x22, 0x22, 0x22, 0x0,  0x0,  0x0,
  0x22, 0x17, 0x41, 0x0,  0x32, 0x36
};
```

The second method, refined by MPICOSYS for low-power optimization (hereafter referred to as "lut_full_low_power"), is defined in the display's Look-Up Table (LUT) as follows:

```
unsigned char lut_full_low_power[] =

{

0x90, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,

0x60, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,

0x90, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,

0x60, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,

0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,

0x00, 0x06, 0x00, 0x00, 0x00, 0x00, 0x00,

0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,

0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,

0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,

0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,

0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,

0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
```

```
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x24, 0x42, 0x22, 0x22, 0x23, 0x32, 0x00, 0x00, 0x00,
0x22, 0x17, 0x41, 0xAE, 0x32, 0x38,
};
```

#### *6.1.2.1 Energy Consumption and Refresh Time*

The global update mode using default waveforms was analyzed in terms of energy consumption and refresh time. The final 2.13″ monochrome display was refreshed with identical content using two different driving methods: first with the waveform defined in “lut_full_update”, and second with the low-power optimized waveform defined in “lut_full_low_power”.

#### *6.1.2.2 Measurement Software Configuration*

In the “core_fw2_for_KUL_demo.ino” firmware provided by UOULU, the “epd.Init()” function is used to initialize the display. Within this function, the waveform LUT is configured. The “epd.Init()” function is defined in the “epd2in13_V3.cpp” file, where either “Lut(lut_full_update)” or “Lut(lut_full_low_power)” can be selected. This mechanism allows for the application of two different driving methods. Energy consumption measurements were conducted using Power Profiler v4.1.1 software.

#### *6.1.2.3 Measurement Hardware Configuration*

The study was conducted on the SUPERIOT Node Core v2.0 JH-UOULU-2024. Measurements were performed using the Nordic Semiconductor PPKII. To isolate the current drawn by the e-ink driver, the following PCB connections were disconnected:

- U2: 3.3 V to e-ink driver and NBVLC RX/TX
- U7: 3.3 V to NBVLC TX
- U9: 3.3 V to NBVLC RX

After these modifications, VIN and VOUT of the PPKII were connected to U2, and the ammeter mode was configured in the PPKII software to measure the e-ink driver current exclusively.

#### *6.1.2.4 Measurement Process*

Measurements were performed in “test1” mode, activated by writing to the appropriate BLE characteristics value on the node, and monitored through the serial port terminal as follows:

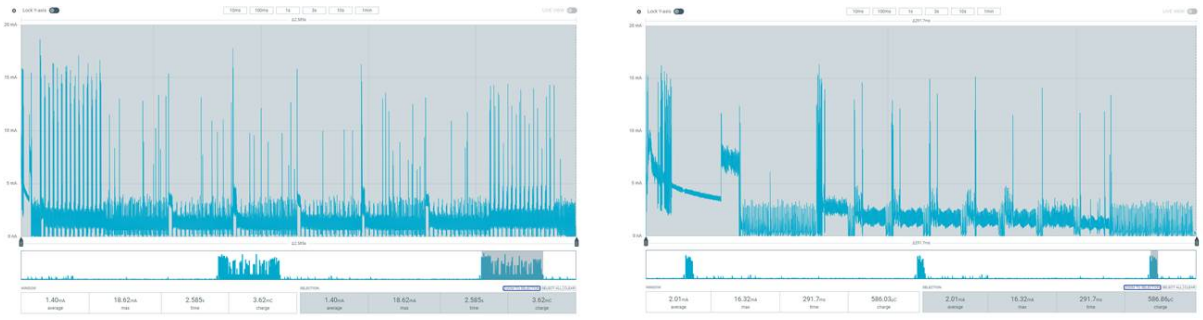
```
test1: Checking AS3933 functionality
RC-oscillator OK
Everything set up.
Temperature = 26.35 *C
Pressure = 1005.18 hPa
Humidity = 56.34 %
Gas = 70.56 KOhms
```

```
Approx. Altitude = 67.32 m

epd FULL

e-Paper clear and sleep
```

The energy consumption for the investigated waveform driving methods is shown in Figure 94.



(a) (b)

Figure 94. Comparison of measured energy consumption for the two e-paper driving waveforms: (a) Default “lut_full_update” — measured charge: 3.62 mC at 3 V (U2), energy consumed: 10.86 mJ; (b) Refined “lut_full_low_power” — measured charge: 0.587 mC at 3 V (U2), energy consumed: 1.761 mJ.

#### *6.1.2.5 Measurement Results and Conclusion*

Using the optimized low-power waveform (“*lut_full_low_power”*), the energy consumption was reduced from **10.86 mJ** to **1.76 mJ**. This is equivalent to a **sixfold energy reduction** compared to the default “*lut_full_update”* waveform.

## 6.2 Optimization of Demonstrator Firmware

Reducing the node’s energy consumption without compromising the required functionality is a key priority. Our analysis shows that the VLC receiving (downlink) functionality significantly increases the MCU workload. The PWM timer functions used for VLC modulation run continuously, preventing the node from entering low-power modes. By disabling VLC downlink functions in the firmware, the gateway can no longer send commands to the node via VLC. However, the VLC uplink remains operational, allowing the node to periodically transmit data to the AP/GW over the optical link. For example, the AP/GW could send a data request via BLE, and the node would then return the requested sensor data via BLE, VLC, or both. The uplink communication over light is also essential for room-level localization in Demo 1.

We tested the developed demo firmware for the node (“*core_fw_SSL.ino”* in the “*SUPERIOT-node-demo-1-2-node-core-fw”* repository on GitLab). Measurements show higher-than-expected current consumption. For instance, during BLE advertising at a 152.5 ms interval and

a TX power level of 4 dBm, the average current consumption is approximately **9.14 mA** (see Figure 95). This is with both BLE and VLC uplink and downlink functions enabled.

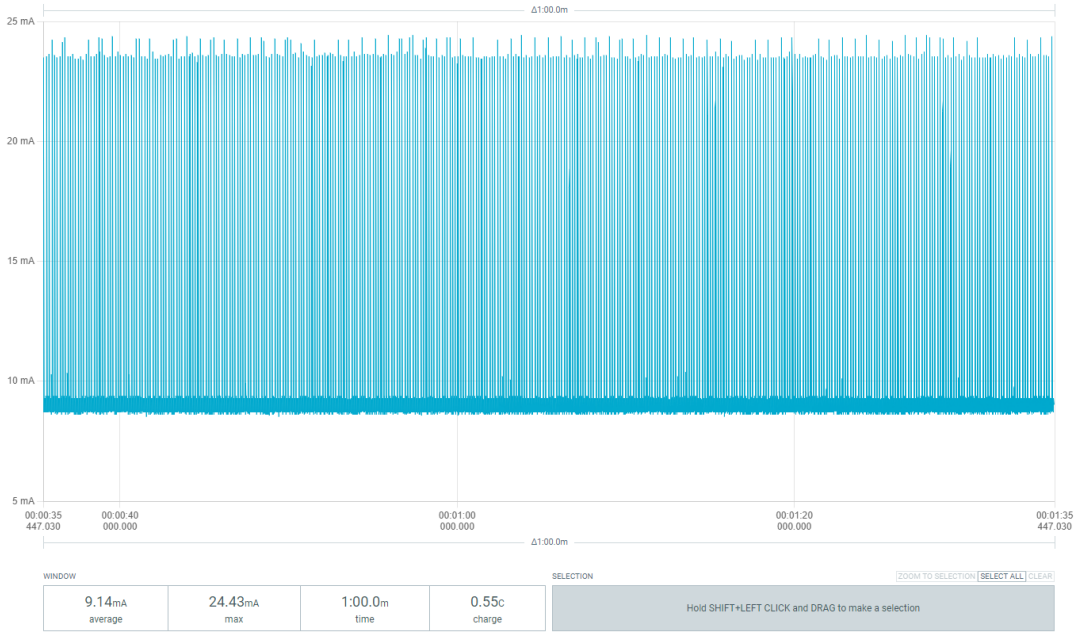



Figure 95. Average current consumption of node during BLE advertising at an interval of 152.5 ms and BLE TX power level of 4 dBm.

### 6.2.1 Optimization Proposal 1 (Both VLC and BLE Uplink and Downlink Enabled)

If the demo requires the node to always listen for mini-lamp GW/AP commands (via VLC downlink or BLE downlink), we can still reduce energy consumption significantly through firmware optimization. We made tweaks to the firmware code in terms of how the node handles VLC downlink (DL)/uplink (UL). The steps are described below.

**STEP 1**: The following function ("nbvlc_on()") is added to the "nbvlc.cpp" file:

```
void nbvlc_on() {
  ITimer0.setInterval(TIMER0_INTERVAL_MS / 10, TimerHandler0);
}
```

**STEP 2**: In the "nbvlc.cpp" file, the "nbvlc_setup()" function is modified by adding the line "nbvlc_on();" as follows:

```
void nbvlc_setup()
{
// Interval in microsecs
  if (ITimer0.attachInterruptInterval(TIMER0_INTERVAL_MS / 10, TimerHandler0))
  {
    Serial.print(F("Starting ITimer0 OK, millis() = "));
    Serial.println(millis());
  }
  else
    Serial.println(F("Can't set ITimer0. Select another freq. or timer"));

  pinMode(pinToUse, OUTPUT);
  pinMode(IN1, INPUT_PULLUP);
  digitalWrite(pinToUse,LOW);

  attachInterrupt(digitalPinToInterrupt(IN1), in1_handler, CHANGE);
  PWM_Instance = new nRF52_PWM(pinToUse, frequency, dutyCycle);

  pinMode(22, OUTPUT); // Power to VLC RX
  digitalWrite(22, HIGH);
  nbvlc_on();
}
```

**STEP 3**: In the “nbvlc.h” header file, the “nbvlc_on();” line is added as follows:

```
void nbvlc_setup();
void updateDC(uint16_t level);
void send_NEC(int8_t add,int8_t c);
int nbvlc_rx_run();
int nbvlc_send(unsigned char add, unsigned
//void send_NEC(int8_t c);
//int nbvlc_rx_run();
//int nbvlc_send(unsigned char c);
unsigned char nbvlc_get_data();
unsigned char nbvlc_get_address();
void nbvlc_lamp_off();
void nbvlc_lamp_on(float p);
void nbvlc_lamp_p(float p);

void nbvlc_on();

#define TIMER_INTERRUPT_DEBUG         0
#define _TIMERINTERRUPT_LOGLEVEL_     3

#define USING_TIMER          false   //true
#define TIMER0_INTERVAL_MS   500   //1000

#define pinToUse   16 //TX PIN
#define IN1        15 //RX PIN
```

**STEP 4**: In the “nbvlc.cpp” file, the “updateDC()” function is modified as follows:

**from:**

```
void updateDC(uint16_t level)
{
  PWM_Instance->setPWM(pinToUse,(level?frequency:frequency*4),lamp_pwm);
}
```

**to:**

```
void updateDC(uint16_t level) {
  uint32_t _level;
  // Mapping data to any other frequency from original data 0-100 to actual 16-bit Dutycycle
  PWM_Instance->setPWM_manual(pinToUse, ((uint32_t)level * MAX_16BIT) / 100);
}
```

Following steps 1 to 4, the average current consumption decreases to ~6.77 mA (by ~26% from 9.14 mA) as shown in Figure 96.

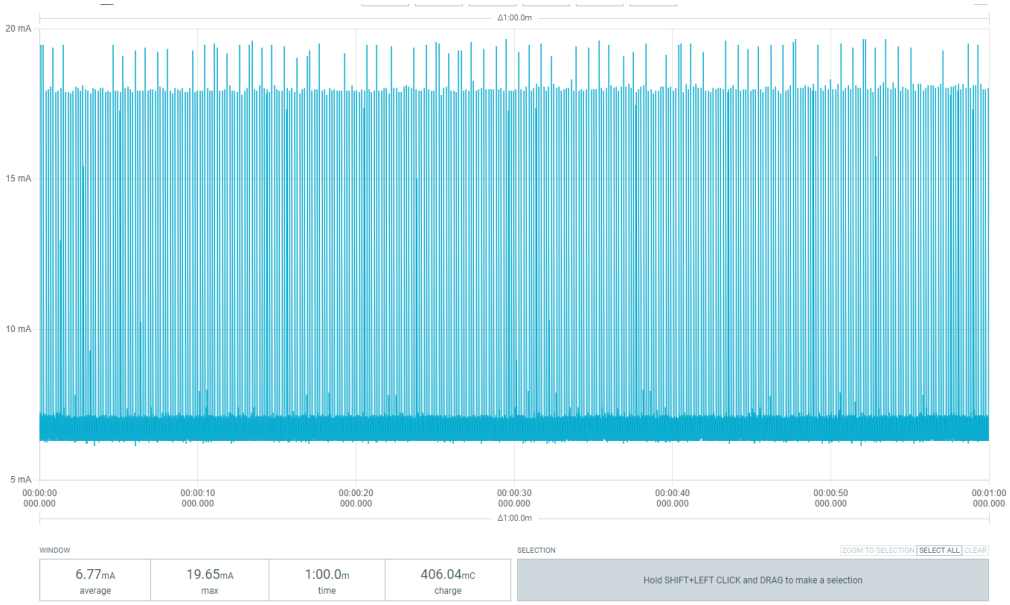



Figure 96. Average current consumption of node during BLE advertising at an interval of 152.5 ms and BLE TX power level of 4 dBm, following optimizations steps 1 to 4.

**STEP 5**: In the "void_loop()" function in the "core_fw_SSL.ino" file, the initial "delay(1);" line is replaced by "delay(100);" as per below:

```
void loop()
{
  //Serial.println(VLC_MAC_ADDR);
  if( nbvlc_rx_run() )
  {
    vlcActions(nbvlc_get_data());
  }
  if ( Bluefruit.connected() ) {if (char_m==6) {Serial.println("Screen changed to environmental conditions");char_m=0; displayEnvCond(); bme.performReading(); startAdv(bme.temperature, bme.humidity);}
                                if (char_m==5) {Serial.println("Screen changed to 1 with BLE");char_m=0; epd.Init(FULL); epd.Display(IMAGE_DATA);}
                                if (char_m==2) {Serial.println("Screen changed to 1 with BLE");char_m=0; epd.Init(FULL); epd.Display(PARACETAMOL_ENG);}//epd_message2();}
                                if (char_m==3) {Serial.println("Screen changed to 2 with BLE");char_m=0; epd.Init(FULL); epd.Display(PARACETAMOL_GER);}//epd_message();}
                                if (char_m==4) {Serial.println("Screen changed to 3 with BLE");char_m=0;epd.Init(FULL);  epd.Display(PARACETAMOL_DUT);}



                                }//Serial.print(char_m);{/* if (VLC_MAC_ADDR==00) {VLC_MAC_ADDR=char_m;Serial.print("VLC Address: ");Serial.println(VLC_MAC_ADDR);}*/};
  //buffer_BLE="";

  //delay(1);
  delay(100);
  //Serial.println(buffer_BLE);
}
```

Applying steps 1 to 5 lowers the average current to **~5.56 mA as shown in** Figure 97, representing a **39% reduction** from the baseline 9.14 mA. These firmware-level optimizations do not affect the intended node's functionality.

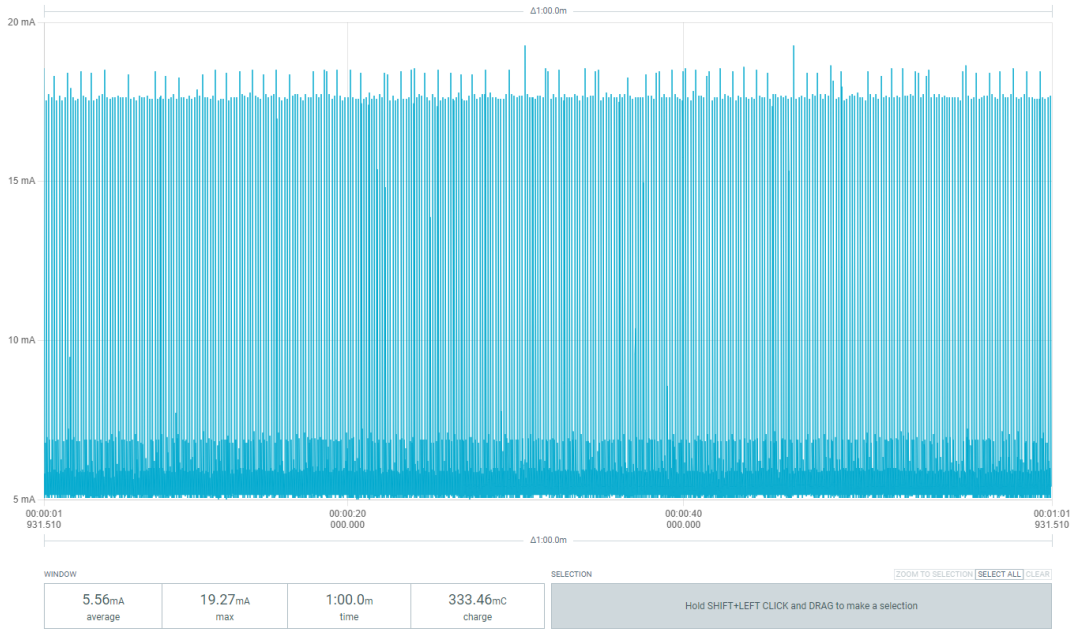



Figure 97. Average current consumption of node during BLE advertising at an interval of 152.5 ms and BLE TX power level of 4 dBm, following optimizations steps 1 to 5.

### 6.2.2 Optimization Proposal 2 (BLE Uplink/Downlink Active, VLC Downlink Disabled, VLC Uplink Active)

In addition to the modifications proposed in Section 6.2.1, further tweaks can be made to the node's firmware to decrease energy consumption significantly. More specifically, the functions responsible for VLC downlink are disabled, while VLC uplink remains fully functional. In this configuration, the node no longer receives commands via VLC from the gateway but continues to support BLE uplink and downlink, and VLC uplink. The optimization steps (continuing from Section 6.2.1) are described below:

**STEP 6**: In "void_setup()" function in "core_fw_SSL.ino", the line "sendVLCSensors(bme.temperature, bme.humidity);" is commented, and the line "nbvlc_off();" is added as shown below:

```
void setup()
{ //randomSeed(analogRead(A0));
  VLC_MAC_ADDR=  (MAC_ADDRESS_LOW & 0xFF) ;

  //Serial.begin(115200);

  pinMode(5, OUTPUT); // EPD POWER ON
  digitalWrite(5,HIGH);
  pinMode(16, OUTPUT);
  digitalWrite(16,LOW);

  bme_setup();
  as_setup();
  ble_setup();
  nbvlc_setup();

  bme.performReading();
  startAdv(bme.temperature, bme.humidity);

  //sendVLCSensors(bme.temperature, bme.humidity); //*********************
  nbvlc_off(); //*****************

  //save_TRH(bme.temperature, bme.humidity);
  if(save_TRH(bme.temperature, bme.humidity))
  {
    pinMode(5, OUTPUT); // EPD POWER ON
    digitalWrite(5,HIGH);
    delay(100);
    epd_message();
  }
}
```

**STEP 7**: In “void_loop()” function, we add some code lines to transmit VLC data in the uplink:

```
void loop()
{
  //*****************  commented*****************
  //Serial.println(VLC_MAC_ADDR);
  // if( nbvlc_rx_run() )
  // {
  //   vlcActions(nbvlc_get_data());
  // }
  //*****************
  if ( Bluefruit.connected() ) {if (char_m==6) {Serial.println("Screen changed to environmental conditions");char_m=0; displayEnvCond(); bme.performReading(); startAdv(bme.temperature, bme.humidity);}
                                if (char_m==5) {Serial.println("Screen changed to 1 with BLE");char_m=0; epd.Init(FULL); epd.Display(IMAGE_DATA);}
                                if (char_m==2) {Serial.println("Screen changed to 1 with BLE");char_m=0; epd.Init(FULL); epd.Display(PARACETAMOL_ENG);}//epd_message2();}
                                if (char_m==3) {Serial.println("Screen changed to 2 with BLE");char_m=0; epd.Init(FULL); epd.Display(PARACETAMOL_GER);}//epd_message();}
                                if (char_m==4) {Serial.println("Screen changed to 3 with BLE");char_m=0;epd.Init(FULL);  epd.Display(PARACETAMOL_DUT);}


                                //*****************  added *****BLE DL command, VLC UL data**********************
                                if (char_m==1) {Serial.println("VLC Uplink Sensor Data TX");
                                char_m=0;
                                bme.performReading();
                                sendVLCSensors(bme.temperature, bme.humidity);
                                }
                                //*********************************************************************
                                }//Serial.print(char_m);{/* if (VLC_MAC_ADDR==00) {VLC_MAC_ADDR=char_m;Serial.print("VLC Address: ");Serial.println(VLC_MAC_ADDR);}*/};
  //buffer_BLE="";

//*****************  added **********************
  //  New: send VLC data every 30s
  if (millis() - lastVLCSend >= vlcInterval) {
    bme.performReading();  // update sensor readings
    sendVLCSensors(bme.temperature, bme.humidity);
    lastVLCSend = millis();  // reset timer
  }
//************************************************************
  //delay(1);
  delay(100);  //   changed
  //Serial.println(buffer_BLE);
}
```

As implemented in “void_loop()”, the node automatically transmits sensor data every 30 seconds via VLC to any listening gateways within line-of-sight. Alternatively, the gateway can send a BLE downlink command to the node, as and when required, which triggers the node to send sensor data in the uplink via VLC.

**STEP 8**: Finally, the “void sendVLCSensors(float T, float RH)” function is modified with the addition of the “nbvlc_on();” and “nbvlc_off();” functions:

```
void sendVLCSensors(float T, float RH) {
  int16_t T_int = T / 0.005;
  uint16_t RH_int = RH / 0.0025;

  Serial.print("Node "); Serial.print(VLC_MAC_ADDR); Serial.print(" -> Bytes (HEX): ");
  Serial.print(T_int & 0x00FF, HEX); Serial.print(",");
  Serial.print(T_int >> 8, HEX); Serial.print(",");
  Serial.print(RH_int & 0x00FF, HEX); Serial.print(",");
  Serial.print(RH_int >> 8, HEX);

  Serial.print(" | Values -> Temp: ");
  Serial.print(T_int * 0.005); Serial.print(" °C, Hum: ");
  Serial.print(RH_int * 0.0025); Serial.println(" %");
nbvlc_on();
  // Send temperature low byte
  while (!nbvlc_send(VLC_MAC_ADDR, T_int & 0x00FF)) delay(5);
  // Send temperature high byte
  while (!nbvlc_send(VLC_MAC_ADDR, T_int >> 8)) delay(5);
  // Send humidity low byte
  while (!nbvlc_send(VLC_MAC_ADDR, RH_int & 0x00FF)) delay(5);
  // Send humidity high byte
  while (!nbvlc_send(VLC_MAC_ADDR, RH_int >> 8)) delay(5);

  delay(100);
  nbvlc_off();
}
```

After applying optimization steps 1 to 8, the resulting node’s current profile is shown in Figure 98.

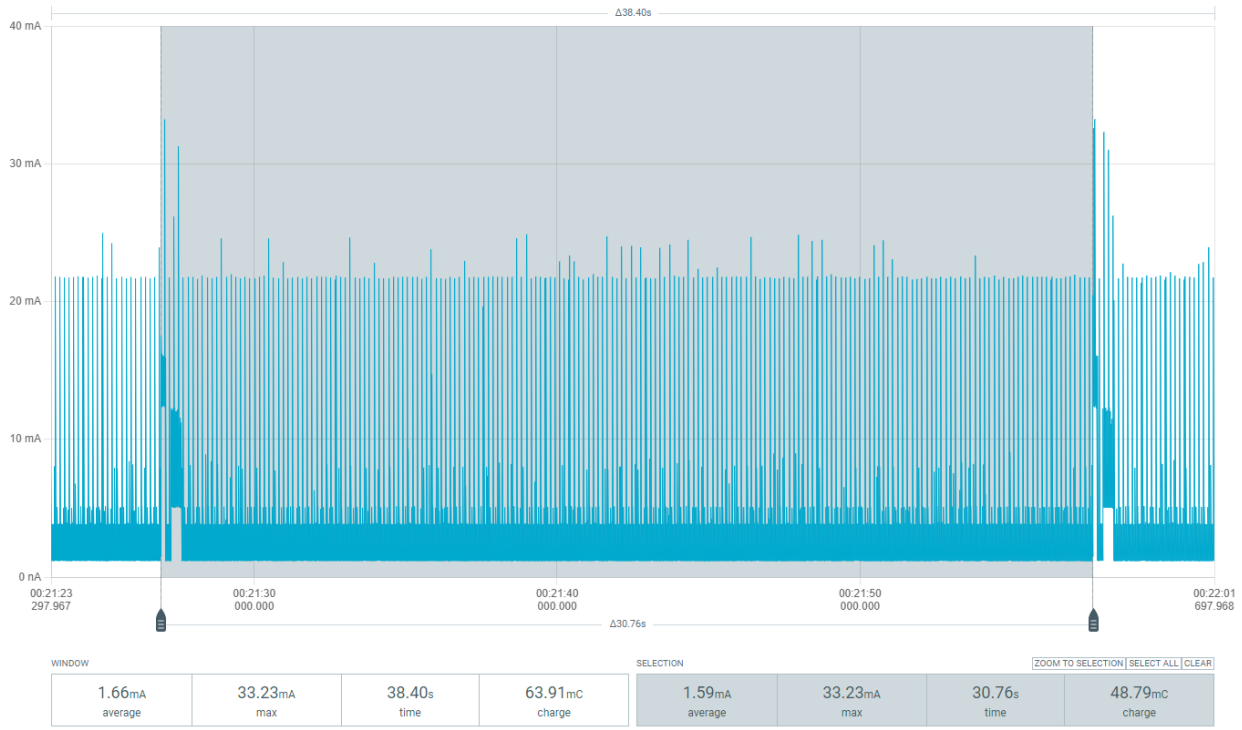



Figure 98. Current profile when node is in BLE advertising mode (BLE TX power = 4 dBm, BLE advertising interval = 152.5 ms), and the node is performing environmental sensing + VLC TX in uplink to the mini-lamp GW every 30 seconds. In this case, the downlink VLC function is disabled, that is, the node does not listen to commands via VLC from mini-lamp GW (only via BLE).

Compared to Figure 97, disabling the VLC downlink functions reduces the average current from 5.56 mA to 1.59 mA, a **71%** improvement. Importantly, the node still receives commands via BLE. Overall, starting from the initial unoptimized firmware (with full BLE and VLC uplink/downlink), applying the optimization steps outlined in Sections 6.2.1 and 6.2.2 results in an **~83% reduction in energy consumption**.

Furthermore, we experimentally validated the three design cases using a commercial 0.47 F supercapacitor (EDC474Z5R5C), which was initially charged to 4.92 V. The supercapacitor was charged from an initial voltage of 0.02 V to 4.92 V using a 5 V supply through a 150 Ω resistor. The corresponding charging profile is presented in Figure 99.

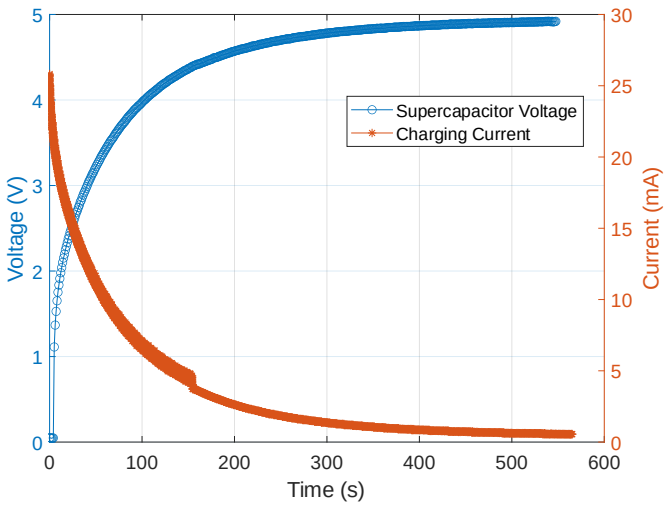



Figure 99. 0.47 F supercapacitor charging profile using a 5V power supply through a 150Ω resistor.

The supercapacitor was then left to discharge under the three operating conditions:

- **Case 1 (Initial, unoptimized):** BLE uplink/downlink + VLC uplink/downlink
- **Case 2 (Optimized):** BLE uplink/downlink + VLC uplink/downlink
- **Case 3 (Optimized):** BLE uplink/downlink + VLC uplink only

(as illustrated previously in Figure 95, Figure 97, and Figure 98, respectively).

In all experiments, the node was configured with a BLE transmit power of 4 dBm and a BLE advertising interval of 152.5 ms. The measured discharge curves are shown in Figure 100.

From the results, the supercapacitor voltage falls to the shutdown threshold of 2.2 V at approximately:

- **83 s** in Case 1 (baseline unoptimized),
- **157 s** in Case 2 (first optimized configuration), and
- **581 s** in Case 3 (further optimized with VLC uplink only).

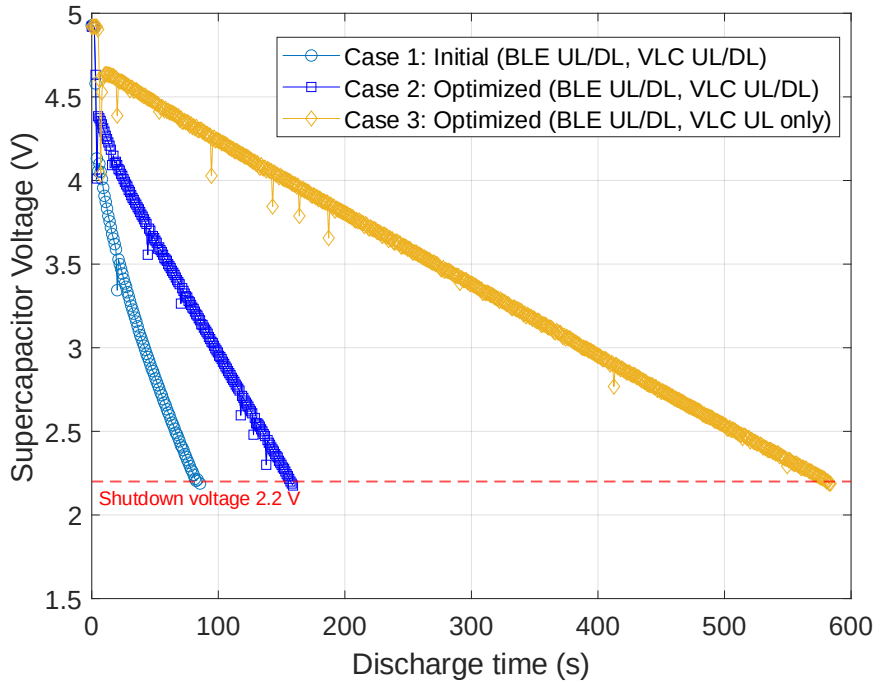



Figure 100. Discharge profiles of a 0.47 F commercial supercapacitor (initial voltage 4.92 V) under three operating conditions: Case 1 (unoptimized, BLE UL/DL + VLC UL/DL), Case 2 (optimized, BLE UL/DL + VLC UL/DL), and Case 3 (optimized, BLE UL/DL + VLC UL only).

**Insights**

- The optimized Case 2 nearly **doubles the operational lifetime** compared to the unoptimized Case 1, demonstrating the benefits of improved system-level power management.
- The further refinement in Case 3 yields a dramatic improvement, extending the runtime by more than **7× relative to Case 1**.
- This highlights the significant impact of communication modality selection (BLE + VLC UL only vs. BLE + VLC UL/DL) on energy efficiency.

We further evaluated the predicted supercapacitor energy depletion using our Energy Consumption Prediction App for Case 3 and compared it against the measured discharge profile shown in Figure 100. In this analysis, we assumed no energy harvesting sources, and a 0.47 F supercapacitor initially charged to 4.92 V. Within the Energy Consumption Prediction App, the node was configured to perform BLE advertising with a transmit power of 4 dBm and an advertising interval of 152.5 ms, over an operating window of 600 seconds. A shutdown voltage of 2.2 V and a discharge efficiency of 99% were specified. The predicted discharge curve, shown in Figure 101, indicates that the supercapacitor voltage drops to the shutdown threshold of 2.2 V at approximately 580 seconds. This closely matches the experimentally observed profile under the same operating conditions, thereby validating the accuracy of the prediction framework.

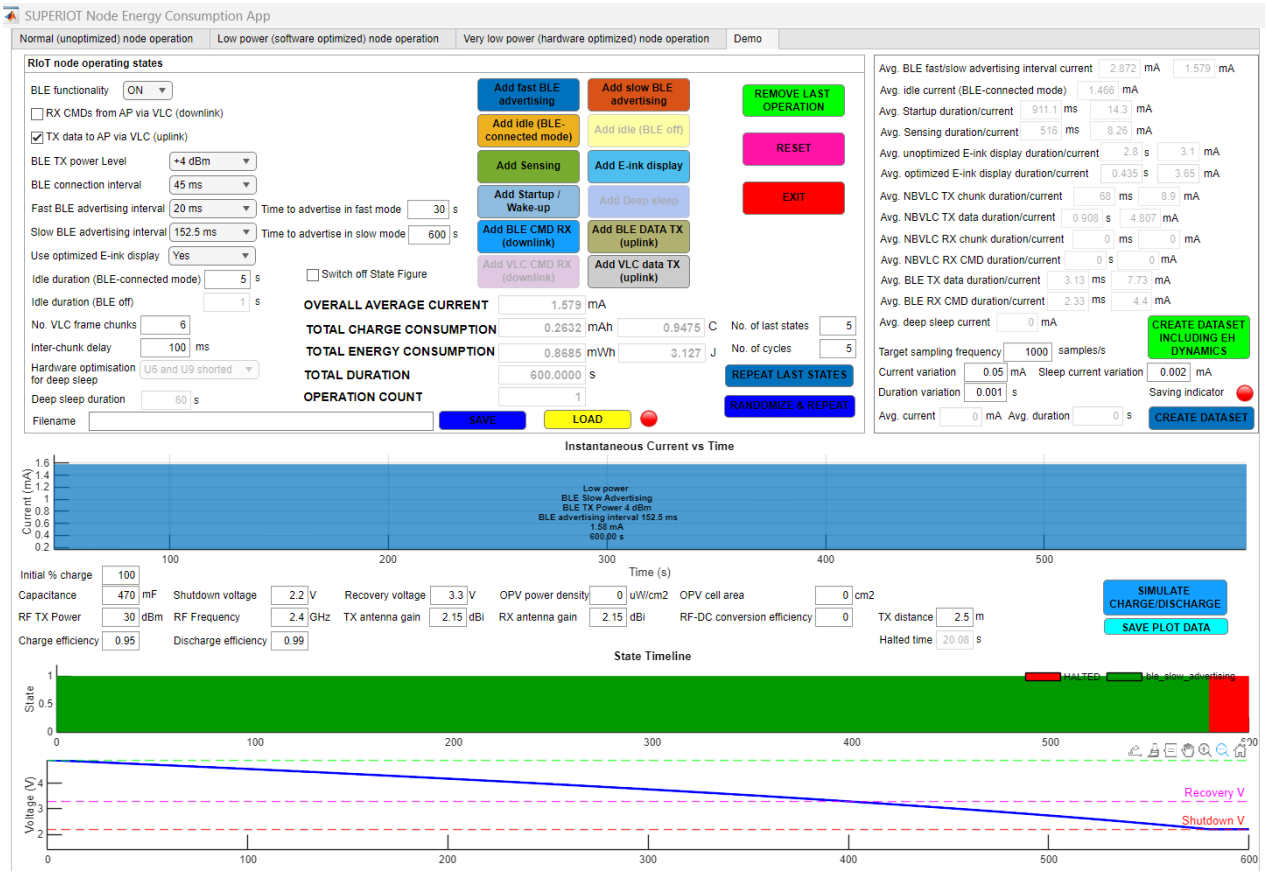



Figure 101. Energy Consumption Prediction App output for Case 3 when node is in BLE advertising mode (BLE TX power = 4 dBm, BLE advertising interval = 152.5 ms). In this case, the node does not listen to commands via VLC from mini-lamp GW but can transmit VLC data in uplink (BLE UL and DL remain active).

## 6.3 Chapter Conclusion

Through optimization of the IoT node hardware within the SUPERIOT system for low-power operation, we were able to integrate an energy-efficient 2.13-inch E-ink display (250×122 pixels), achieving more than a fivefold reduction in energy consumption. Complementary optimization of the IoT node embedded software enabled the adoption of an improved e-paper driving method, which further reduced energy consumption by over six times. In addition, the optimized firmware significantly shortened display's refresh times—by more than a factor of ten.

Finally, for the node firmware developed for the Demonstrators on GitLab, we proposed firmware-level optimizations, with potential energy savings exceeding 80%.

Looking ahead, the next steps will involve updating the demonstrator's use-case application scenario to account for the physical constraints of energy harvesting and storage, as well as the

energy demands of tasks delegated to nodes within the SUPERIOT system. This work will leverage the SUPERIOT Node Energy Consumption Prediction Tool provided by the U of Bristol with detailed experimental validation on the demonstrator.

# 7 Conclusions

## 7.1 Summary of Findings

Task 3.4 has comprehensively addressed energy efficiency in the reconfigurable IoT system through an integrated framework for energy modelling, prediction, and optimization across multiple system layers. At the network level, detailed empirical measurements were used to derive accurate energy-consumption models for the dual-communication SUPERIOT platform, covering nodes, gateways, access points, and supporting infrastructure components. These measurement-informed models then enabled the development of AI/ML-based prediction tools capable of estimating energy usage with high accuracy. Building on this foundation, an ML-based optimization framework was further developed to support automated configuration selection under practical operating constraints, balancing available energy budgets against communication and mission requirements. Collectively, these contributions provide a scalable and practical basis for energy-aware network design, supporting more efficient, adaptive, and sustainable IoT deployments.

Building upon this foundation, adaptive optimization mechanisms were introduced for hybrid RF/optical communication networks, addressing the complexities of modality selection, handover, and power allocation. Through the use of multi-objective reinforcement learning and GNN-based methods, the framework achieved efficient, scalable, and resilient network operation while maintaining QoS and minimizing power consumption.

A DT environment, implemented in ns-3, further extended the framework by supporting traffic- and usage-aware simulations that enable proactive, cross-layer energy optimization. The integration of BLE, VLC, and energy models within the simulator allowed for realistic experimentation under diverse traffic and environmental conditions, validating the feasibility of intelligent, adaptive network control.

At the system level, substantial progress was achieved in hardware and firmware optimization of IoT nodes, including the integration of low-power e-paper displays and energy-efficient embedded software. These enhancements delivered over a fivefold reduction in energy consumption and a tenfold improvement in display refresh times, contributing to the overall energy-saving goals of the SUPERIOT system.

Collectively, these achievements demonstrate a complete end-to-end framework, from empirical measurement and AI-driven prediction to cross-layer and system-level optimization, that advances the state of energy-efficient IoT network design. The outcomes of Task 3.4 provide a strong technological foundation for developing sustainable, self-optimizing RIoT systems aligned with the broader objectives of the SUPERIOT project.

## 7.2 Future Work and Recommendations

Future work will focus on the integration and experimental validation of the developed models and optimization mechanisms within real-world RIoT demonstrators under WP4. Emphasis will be placed on enhancing the adaptability of traffic prediction and modality selection algorithms, refining RIS coordination strategies, and extending AI/ML models to account for real-time energy harvesting and storage dynamics. Additionally, continued efforts will target the co-optimization of hardware and communication layers to further reduce energy overhead while maintaining application-specific QoS. These advancements will support the SUPERIOT vision of achieving scalable, sustainable, and intelligent IoT deployments through the convergence of reconfigurable networking, printed electronics, and AI-driven energy management.

# 8 Bibliography


[1] M. J. Bocus, J. Hakkinen, H. Fontes, M. Drzewiecki, S. Qiu, K. Eder, and R. Piechocki, “An energy-aware RIoT system: Analysis, modeling and prediction in the SUPERIOT framework”, 2025, https://arxiv.org/abs/2501.10093 arXiv: 2501.10093 [cs.ET].

[2] M. J. Bocus, S. Qiu, R. Piechocki, and K. Eder, “Towards Automated and Predictive Network-Level Energy Profiling in Reconfigurable IoT Systems”, 2025, 10.48550/arXiv.2510.09842.

[3] RasPi.TV, “How much power does the Pi4B use? Power Measurements,” Jun. 2019. [Online]. Available: https://raspi.tv/2019/how-much-power-does-the-pi4b-use-power-measurements. [Accessed: Mar. 13, 2025].

[4] Bret, “How to Power the Raspberry Pi 5: A complete Guide,” Jan. 2025. [Online]. Available: https://bret.dk/how-to-power-the-raspberry-pi-5-a-complete-guide/#Raspberry-Pi-5-Power-Requirements. [Accessed: Mar. 13, 2025].

[5] C. Gittins, “Considering the energy consumption of a Raspberry Pi,” Sep. 2024. [Online]. Available: https://www.iotinsider.com/news/considering-the-energy-consumption-of-a-raspberry-pi/#:~:text=Understanding%20Raspberry%20Pi%20power%20performance&text=The%20Raspberry%20Pi%204%2C%20for,can%20significantly%20affect%20battery%20life. [Accessed: Mar. 13, 2025].

[6] F. Kaup, P. Gottschling and D. Hausheer, "PowerPi: Measuring and modeling the power consumption of the Raspberry Pi," *39th Annual IEEE Conference on Local Computer Networks*, Edmonton, AB, Canada, 2014, pp. 236-243, doi: 10.1109/LCN.2014.6925777.

[7] P. Mahadevan, P. Sharma, S. Banerjee, and P. Ranganathan, "A power benchmarking framework for network devices," in NETWORKING 2009, ed: Springer, 2009, pp. 795-808.

[8] C. Gray, “Energy consumption of internet of things applications and services,” 2018. [Online]. Available: https://api.semanticscholar.org/CorpusID:199002380

[9] C. Gray, R. Ayre, K. Hinton and R. S. Tucker, "Power consumption of IoT access network technologies," *2015 IEEE International Conference on Communication Workshop (ICCW)*, London, UK, 2015, pp. 2818-2823, doi: 10.1109/ICCW.2015.7247606.

[10] A. Vishwanath, F. Jalali, R. Ayre, T. Alpcan, K. Hinton and R. S. Tucker, "Energy consumption of interactive cloud-based document processing applications," *2013 IEEE International Conference on Communications (ICC)*, Budapest, Hungary, 2013, pp. 4212-4216, doi: 10.1109/ICC.2013.6655224.

[11] M. Bocus, “SUPERIOT_UoB,” GitHub repository. [Online]. Available: https://github.com/mb13530/SUPERIOT_UoB. [Accessed: Apr. 15, 2026].

[12] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, “Attention is all you need,” 2023. [Online]. Available: https://arxiv.org/abs/1706.03762.

[13] L. Anghinoni, Y.-t. Zhu, D. Ji, and L. Zhao, “Transgnn: A transductive graph neural network with graph dynamic embedding,” in 2023 International Joint Conference on Neural Networks (IJCNN), 2023, pp. 1–8.

[14] W.-D. Li, K. Hu, C. Larsen, Y. Wu, S. Alford, C. Woo, S. M. Dunn, H. Tang, M. Naim, D. Nguyen, W.-L. Zheng, Z. Tavares, Y. Pu, and K. Ellis, “Combining induction and transduction for abstract reasoning,” 2024. [Online]. Available: https://arxiv.org/abs/2411.02272.

[15] Bluetooth SIG. The Bluetooth Low Energy Primer, 2023. URL: https://www.bluetooth.com/bluetooth-resources/the-bluetooth-low-energy-primer/.

[16] Stijn Geysen. Design and implementation of a bluetooth low energy ns-3 module for simulation of large bluetooth low energy mesh networks. Master’s thesis, KULeuven, 2018.

URL:https://kuleuven.limo.libis.be/discovery/fulldisplay?docid=alma9992367317601488&context=L&vid=32KUL_KUL:KULeuven&search_scope=All_Content&tab=all_content_tab&lang=en .

[17] Carles Gomez, Joaquim Oller, and Josep Paradells. Overview and evaluation of bluetooth low energy: An emerging low-power wireless technology. Sensors, 12:11734–11753, 8 2012. doi:10.3390/s120911734.

[18] Tilahun Zerihun Gutema, Harald Haas, and Wasiu O. Popoola. Wdm based 10.8 gbps visible light communication with probabilistic shaping. Journal of Lightwave Technology, 40:5062–5069, 8 2022. doi:10.1109/JLT.2022.3175575

[19] Stephen S. Oyewobi, Karim Djouani, and Anish Matthew Kurien. Visible light communications for internet of things: Prospects and approaches, challenges, solutions and future directions. Technologies, 10, 2 2022. doi:10.3390/technologies10010028.

[20] Adel Aldalbahi, Michael Rahaim, Abdallah Khreishah, Moussa Ayyash, Ryan Ackerman, James Basuino, Walter Berreta, and Thomas D.C. Little. Extending ns3 to simulate visible light communication at network-level. In 2016 23rd International Conference on Telecommunications (ICT), pages 1–6, 2016. doi:10.1109/ICT.2016.7500485

[21] ns-3 Documentation – Wi-Fi Module, 2024. Accessed: June, 2024. URL: https://www.nsnam.org/docs/models/html/wifi.html.

[22] The ns-3 Project. ns-3 Documentation – Energy Framework, 2019. Accessed: December - 2023. URL: https://www.nsnam.org/docs/models/html/energy.html

[23] Martina Capuzzo, Carmen Delgado, Jeroen Famaey, and Andrea Zanella. An ns-3 implementation of a battery-less node for energy-harvesting internet of things. In Proceedings of the 2021 Workshop on Ns-3, WNS3 '21, pages 57–64, New York, NY, USA, 2021.Association for Computing Machinery. URL: https://doi.org/10.1145/3460797.3460805, doi:10.1145/3460797.3460805.

[24] The ns-3 Project. ns-3 Documentation – Energy Model API Architecture and API, 2018. Accessed: December, 2023. URL: https://www.nsnam.org/wiki/Energy_model.

[25] T. Ribeiro, S. Silva, J. P. Loureiro, E. N. Almeida, N. T. Almeida and H. Fontes, "Short-Range Energy-Aware Optical Wireless Communications Module for Ns-3," 2025 Joint European Conference on Networks and Communications & 6G Summit (EuCNC/6G Summit), Poznan, Poland, 2025, pp. 846-851, doi: 10.1109/EuCNC/6GSummit63408.2025.11037159.

[26] A. A. Abdellatif, S. Silva, E. Baltazar, B. Oliveira, S. Qiu, M. J. Bocus, K. Eder, R. J. Piechocki, N. T. Almeida, and H. Fontes, "RIoT Digital Twin: Modeling, Deployment, and Optimization of Reconfigurable IoT System with Optical–Radio Wireless Integration," arXiv:2511.09303, 2025. [Online]. Available: http://arxiv.org/abs/2511.09303.

[27] A. Zakeri, M. Moltafet, M. Leinonen, and M. Codreanu, "Minimizing the AoI in resource-constrained multi-source relaying systems: Dynamic and learning-based scheduling," IEEE Trans. Wireless Commun., vol. 23, no. 1, pp. 450–466, Jan. 2024.

[28] A. Hamrouni, S. Pollin and H. Sallouha, "Resource Allocation in Hybrid Radio-Optical IoT Networks Using GNN With Multi-Task Learning," in IEEE Transactions on Machine Learning in Communications and Networking, vol. 4, pp. 542-561, 2026.

[29] Pia, A.D., Dey, S.S. & Molinaro, M. Mixed-integer quadratic programming is in NP. Math. Program. 162, 225–240, 2017.

# 9 List of figures

# 10 List of tables

# 11 List of contributors

| Contributors | Short name | Country |
|---|---|---|
| OULUN YLIOPISTO | UOULU | Finland |
| INESC TEC - INSTITUTO DE ENGENHARIADE SISTEMAS E COMPUTADORES, TECNOLOGIA E CIENCIA | INESC TEC | Portugal |
| MPICOSYS - EMBEDDED PICO SYSTEMS SPZOO | MPICOSYS | Poland |
| KATHOLIEKE UNIVERSITEIT LEUVEN | KU Leuven | Belgium |
| UNIVERSITY OF BRISTOL | U of Bristol | United Kingdom |