NATIONAL ENGINEERING SCHOOL OF TUNIS

ICT Department

**Year End Project Report**

# Implementation of a BLE Audio Profile on RTOS for Audiometry Applications


*Author:*

MIGHRI Manar

*Supervisors:*

Mrs. JAIDANE Meriem

Mr. MECHERGUI Nader


2$^{nd}$ Year TEL1

2024/2025

# Contents

# List of Figures

# List of Tables

# List of Abbreviations

This section lists the main abbreviations used throughout the report.

| | |
|---|---|
| **ANC** | Active Noise Cancellation |
| **BLE** | Bluetooth Low Energy |
| **CPU** | Central Processing Unit |
| **DAC** | Digital-to-Analog Converter |
| **DTS** | Device Tree Source |
| **DB** | Decibel |
| **FIFO** | First-In, First-Out (buffer memory model) |
| **GATT** | Generic Attribute Profile |
| **$\mathbf{I^2S}$** | Inter-IC Sound (Serial Audio Protocol) |
| **ISR** | Interrupt Service Routine |
| **ISO** | Isochronous Channels (in BLE Audio) |
| **LC3** | Low Complexity Communication Codec |
| **LED** | Light-Emitting Diode |
| **MCU** | Microcontroller Unit |
| **MEMS** | Micro-Electro-Mechanical Systems |
| **RTOS** | Real-Time Operating System |
| **SDK** | Software Development Kit |
| **SPL** | Sound Pressure Level |
| **UART** | Universal Asynchronous Receiver-Transmitter |
| **WP** | Work Package |

# Acknowledgment

First of all, I would like to express my deepest gratitude to my supervisors, Mrs. Meriem JAIDANE, Professor at the National Engineering School of Tunis (ENIT) and co-founder of the spin-off dB.Sense, and Mr. Nader MECHERGUI, PhD Research Engineer and co-founder of dB.Sense, for their invaluable support, guidance, and availability throughout this project. Their insightful remarks and expertise greatly contributed to the successful completion of this work.

This project was carried out within the spin-off dB.Sense, affiliated with the University of Tunis El Manar.

Finally, I would like to sincerely thank all the people who contributed, directly or indirectly, to the success of this endeavor. I dedicate this work to each of them as a token of my heartfelt appreciation.

# General Introduction

Hearing health is a critical but often overlooked component of public health. According to the World Health Organization, over 430 million people worldwide currently suffer from disabling hearing loss, a figure expected to grow to nearly 700 million by 2050 [4]. This growing prevalence represents not only a medical challenge but also an economic and societal burden, affecting communication, education, employability, and social integration.

Traditional audiometric testing relies heavily on centralized clinical infrastructures, requiring soundproof booths, specialized audiometers, and trained personnel. While effective in ideal conditions, this model remains largely inaccessible to large segments of the global population, especially in low-resource or remote areas.

The need for more accessible, decentralized, and user-centered diagnostic tools has therefore become increasingly urgent. Recent technological advancements — particularly in embedded systems, wireless communication (Bluetooth Low Energy Audio), and real-time operating systems (RTOS) — open the door to new generations of portable audiometric solutions.

This project aims to contribute to this transformation by implementing a real-time embedded audiometric system based on BLE Audio streaming and ambient noise monitoring, running on the nRF5340 platform under Zephyr RTOS. By combining wireless mobility, real-time feedback, and modular architecture, the system lays the groundwork for portable, scalable, and collective audiometric diagnostics adapted to modern healthcare challenges.

The work is organized into the following chapters:

- **Chapter 1:** Establishes the medical, societal, and technological motivations behind the project.
- **Chapter 2:** Details the system architecture and technological choices made during the design phase.
- **Chapter 3:** Describes the implementation, integration, and validation of the embedded system.

Through this project, we seek to bridge the gap between traditional clinical audiometry and emerging digital health paradigms, making hearing diagnostics more inclusive, portable, and adaptable to evolving patient needs.

# Chapter 1

# From Conventional Audiometry to Smart Diagnostics: Challenges, Needs, and Methodological Approach

## Introduction

The early detection and treatment of hearing loss are crucial to improving quality of life across all age groups. Yet, conventional audiometric systems often remain constrained to clinical environments, creating barriers in terms of accessibility, flexibility, and cost.

In the *General Introduction*, we presented the vision of implementing a BLE Audio Profile over RTOS to serve audiometry applications. This chapter sets the foundation for that work by examining the medical, societal, and technological context in which this solution is situated.

To address the limits of traditional audiometry and respond to evolving needs, this chapter introduces the rationale behind wireless and embedded audiometric solutions. As a preview, we highlight how mobility, collective testing potential, and compatibility with modern user expectations motivate our design choices.

We have structured the chapter into four parts:

- First, we present the medical and societal needs that call for broader access to hearing diagnostics.
- Second, we examine the technical and procedural limits of conventional audiometric systems.

- Third, we explore the motivations for adopting wireless and embedded solutions.
- Finally, we outline the methodological framework guiding our work.

This structure allows us to progressively frame the problem, understand its context, and justify the design decisions described in the chapters that follow.

## 1.1 The Medical and Societal Need for Accessible Hearing Diagnostics

*This section outlines the growing global impact of hearing loss and the barriers faced in accessing timely and affordable diagnostics. It highlights the societal urgency to decentralize audiometric testing and improve outreach through modern technological solutions.*

### 1.1.1 Global Prevalence and Impact of Hearing Loss

According to the World Health Organization, more than 430 million people worldwide need rehabilitation for hearing loss - a number projected to rise to over 700 million by 2050 [4]. Hearing loss is recognized as one of the leading causes of disability globally, particularly in low- and middle-income countries where early intervention resources are limited [4].

The burden of hearing loss is not only medical but also social and economic: it significantly affects communication, social integration, academic performance, and employability. Untreated hearing loss has been linked to cognitive decline and increased risk of dementia in aging populations, emphasizing the need for proactive screening [4].

This issue is particularly pronounced among aging populations, where presbycusis (age-related hearing loss) is common. However, children are also heavily affected, especially in low- and middle-income countries, where early screening and intervention programs are limited or nonexistent. In both demographics, untreated hearing loss contributes to cognitive decline, isolation, and reduced quality of life.

### 1.1.2 Barriers to Early Detection and Intervention

Despite the recognized importance of early detection, traditional audiometric services remain centralized, costly, and logistically demanding. According to WHO reports [4], many individuals must travel long distances to reach specialized clinics, if such clinics are even available in their region.

For others, the social stigma or fear associated with medical environments becomes a psychological barrier to seeking help. Studies have shown that perceived stigma often results in significant delays in hearing aid adoption and hearing rehabilitation [10].

In addition, current diagnostic practices often rely on bulky and expensive hardware that must be operated by trained professionals in soundproof rooms. This creates a bottleneck in scaling outreach programs and hinders large-scale screening initiatives in schools, rural areas, or refugee camps.

### 1.1.3 Importance of Decentralized and Inclusive Audiometric Screening

To address the growing demand for hearing diagnostics, there is a clear need to decentralize testing and integrate it into community contexts [11]. Schools, elderly care facilities, and even home settings can benefit from lightweight and user-friendly tools that reduce dependence on clinical infrastructure.

A decentralized approach also allows for more flexible, patient-centered care. For example, children with sensory processing disorders (such as autism spectrum conditions) may find traditional clinical settings distressing, making mobile screening options crucial for accurate diagnosis and equitable healthcare delivery [1].

### 1.1.4 Role of Technology in Public Health Transformation

According to Tong and Thakor [11], embedded systems have already revolutionized areas such as cardiology, respiratory monitoring, and telemedicine. Audiometry is next.

The emergence of embedded wireless platforms, especially those that integrate Bluetooth Low Energy (BLE) and Real-Time Operating Systems (RTOS), presents a powerful opportunity to deliver high-fidelity diagnostics in portable, cost-effective formats [12]. These systems can integrate with mobile apps, cloud-based data collection, and collective testing frameworks, making hearing screening not only more accessible but also scalable.

In this context, our work aims to contribute to the broader public health shift toward proactive, embedded, and connected diagnostics, building on the foundations laid by embedded real-time health monitoring technologies.

## 1.2 Limitations of the Current Conventional Audiometric Systems

*This section provides a detailed description of how traditional audiometric testing is conducted and the limitations it presents in real-world scenarios. It examines practical, technical, and user-related constraints that justify the need for new embedded and wireless solutions.*

### 1.2.1 Understanding Conventional Audiometric Examinations

A standard pure-tone audiometry test, considered the gold standard in hearing evaluation [1], typically takes place in a soundproof booth. The patient is seated inside the booth wearing over-ear headphones that are connected by cables to an audiometer placed outside, usually operated by an audiologist. Tones of varying frequencies and intensities are played through the headphones, and the patient is instructed to signal when they perceive a sound — often using a push button or by raising a hand. The responses are recorded manually or digitally to construct an audiogram.

This process is illustrated in Figures 1.1 and 1.2. On the one hand, Figure 1.1 shows a typical exam setup involving a patient and a clinician. On the other hand, Figure 1.2 presents a schematic overview of the audiometer hardware involved in such procedures.
While this method ensures precise control over stimulus presentation, it also brings several drawbacks in terms of logistics, scalability, and user experience — detailed in the following subsections.

Figure 1.1: Audiometry exam in progress: a patient seated in a soundproof booth while a clinician conducts the hearing test using an audiometer. [2]

Figure 1.2: Components of a conventional audiometry setup: audiometer, wired headphones. [3]

### 1.2.2 Dependence on Clinical Infrastructure

Traditional audiometric systems are designed around centralized, hospital-based environments. These systems require a soundproof booth, high-end diagnostic audiometers, and trained audiologists to operate them. The equipment is expensive and bulky, limiting its availability to specialized centers and excluding low-resource or rural settings from consistent access.

### 1.2.3 Accessibility and Mobility Constraints

Access to audiometric testing can be significantly hindered by geography, cost, or patient mobility. Elderly patients, individuals with disabilities, or residents of remote regions may struggle to attend regular hearing screenings. This often delays diagnosis and intervention, compounding the cognitive, emotional, and social consequences of untreated hearing loss.

### 1.2.4 Technical Limitations: Wired Equipment and Environmental Noise

Conventional systems rely heavily on wired headphones. These can introduce unintended noise—such as friction between the cable and the patient's clothing—affecting the integrity of the test [10], especially in quiet tone-based examinations. Moreover, wired setups restrict movement, making the process rigid and uncomfortable for some patients.

This limitation is particularly pronounced in pediatric or neurodiverse populations. For example, children with autism spectrum disorders may feel overwhelmed or anxious in clinical spaces, or might seek sensory regulation by hiding or distancing themselves from medical personnel. Such behavior, while natural, can disrupt the testing procedure or prevent it entirely when using conventional wired setups.

### 1.2.5 Incompatibility with Modern Digital Ecosystems

Legacy audiometric tools lack compatibility with today's connected medical environments. They are typically not designed to interface with mobile devices, cloud-based systems, or remote diagnostics platforms. This restricts their integration into digital health workflows and makes it difficult to collect, share, or analyze data at scale.

Such limitations are particularly restrictive in collective testing settings—like schools or elderly care facilities—where modern wireless systems could enable multiple patients to be tested simultaneously using a single portable tablet or diagnostic hub.

In summary, the inflexibility, infrastructural demands, and analog nature of conventional audiometry create both functional and human-centric barriers. These challenges highlight the need for decentralized, wireless, and embedded alternatives, as explored in the next section.

## 1.3 Motivation for Wireless and Embedded Audiometry

*In this section, we present the key motivations for transitioning to a wireless, embedded audiometric platform. Specific use cases, technological trends, and patient-centered considerations are explored to justify the project direction.*

### 1.3.1 From Constraint to Flexibility: Reimagining Audiometric Testing

The shortcomings of traditional audiometric systems underscore the need for more flexible, scalable solutions. The rigid reliance on clinical settings, specialized operators, and fixed hardware configurations is at odds with the direction of modern healthcare, which increasingly prioritizes portability, personalization, and real-time diagnostics.

Wireless and embedded systems allow testing to go beyond clinical environments and into homes, schools, or community centers. By embedding digital signal processing capabilities and BLE audio support directly into lightweight hardware [9], such systems can achieve clinical-grade performance without the overhead of traditional infrastructure.

### 1.3.2 Reducing Physical and Acoustic Artifacts with Wireless Headphones

One of the practical issues with wired headphones is the inadvertent introduction of noise through physical movement. Friction between the cable and the patient's clothing can generate artifacts that disrupt precise tone-based hearing tests. Eliminating the cable removes this source of interference.

Furthermore, as consumer devices increasingly phase out analog audio ports, the reliance on wired transducers becomes both technically outdated and logistically limiting. Embracing wireless headphones not only aligns with broader hardware trends but also facilitates a more comfortable and unobtrusive experience for the patient.

### 1.3.3 Enabling Patient-Centered Design: The Case of Pediatric and Neurodiverse Use

Children, particularly those with neurodevelopmental disorders such as autism spectrum conditions, may resist close physical proximity to clinicians or be sensitive to certain environmental stimuli. A wired setup with rigid posture expectations can trigger distress or lead to avoidance behaviors.

By contrast, a wireless audiometric headset allows children to move more freely during testing — even if they crawl under a table or keep a distance from the examiner. This flexibility promotes better test completion rates and more representative results in real-world conditions.

### 1.3.4 Toward Collective and Scalable Testing Frameworks

An embedded BLE-based solution opens up the possibility for collective or batch testing. In educational or outreach contexts, a single tablet or embedded controller could broadcast test signals to multiple headsets at once, enabling group hearing assessments. This model is especially beneficial for resource-constrained settings where clinical personnel are limited.

Although the current hardware prototype features a direct cable from the headphones to a small medical interface box (used to signal patient responses), this architecture is modular. It is designed to evolve toward full wireless functionality as certification requirements for medical headphones are satisfied.

### 1.3.5 Bridging the Gap Between Modern Technology and Audiometry Practice

Wireless embedded audiometry platforms align with emerging digital health paradigms, such as decentralized diagnostics, real-time data transmission, and AI-assisted assessment. By leveraging RTOS for task scheduling and BLE Audio for low-latency streaming, our system lays the groundwork for a new class of connected diagnostic tools — portable, adaptive, and user-centered.

This project aims to bridge the technological and clinical gap by introducing a real-time embedded audiometric system designed not only for performance but also for inclusivity, comfort, and scalability.

## 1.4 Project Methodology

*This section presents the development methodology followed throughout the project, structured into three work packages. Each work package focuses on a* *specific aspect of the system: RTOS deployment, BLE integration, and ambient noise measurement.*

### 1.4.1 Overview of the Work Packages

To address the identified challenges and fulfill the project objectives, the development process was structured around three main work packages:

- **WP1: Deployment of Zephyr RTOS for Real-Time Task Management**
- **WP2: Implementation of a Customized BLE Stack for Wireless Diagnostic Headphones**
- **WP3: Ambient Noise Measurement and Cancellation**

Each work package was designed to focus on a distinct layer of the system, from task scheduling to wireless communication and real-time environmental sensing, while maintaining overall coherence through a modular and test-driven approach.

### 1.4.2 WP1: RTOS Setup and Multitasking Integration

The first work package focused on configuring the Zephyr Real-Time Operating System (RTOS) to handle multitasking with minimal latency. After acquiring familiarity with Zephyr's kernel mechanisms (e.g., Kconfig, device trees), the RTOS was validated with basic applications such as LED blinking and timer routines. This foundation enabled the integration of a noise monitoring thread, which required real-time scheduling and interrupt-safe task prioritization to ensure responsiveness during BLE streaming.

### 1.4.3 WP2: BLE Stack Integration and Configuration

Instead of developing a BLE Audio protocol stack from scratch, this project reuses Nordic Semiconductor sample applications for BLE Audio unicast communication. These examples run on two nRF5340 Audio DKs—allowing real-time audio streaming between a client and a server board. The stack was analyzed and tested under different wireless conditions, but no modification to the core BLE logic was introduced at this stage. The focus was placed on evaluating the suitability of this BLE link for diagnostic-grade audio and ensuring its stable operation as the baseline for the enhanced features developed in WP3. This project adopts the BLE Audio specification framework as defined by the Bluetooth SIG [7].

### 1.4.4 WP3: Ambient Noise Monitoring and Threshold Response

This work package represents the core contribution of the project. An algorithm was implemented to read raw data from a digital MEMS microphone connected via the $I^2S$ interface, compute RMS noise levels in real time, and trigger visual indicators (e.g., RGB LED) when thresholds were exceeded. The system was integrated into the existing RTOS structure as a standalone thread and validated for responsiveness and minimal interference with the ongoing BLE audio stream. This design lays the groundwork for future integration of active noise cancellation (ANC) mechanisms and dynamic test cancellation strategies in high-noise environments.

### 1.4.5 Development Philosophy and Future Integration

All new functionality was implemented in a minimally invasive manner, ensuring compatibility with future updates of the BLE Audio stack. The modularity of the work packages allows future extensions, including integration of mobile applications, cloud-based audiogram storage, and real-time user feedback, without requiring a complete architectural overhaul.

## Conclusion

This first chapter has established the broader context in which this project is situated. We started by identifying the growing medical and social demand for accessible and flexible hearing diagnostics. We then analyzed the inherent limitations of conventional audiometric systems, particularly in terms of accessibility, infrastructure dependence, and patient adaptability. These observations helped justify the transition toward embedded wireless audiometry, supported by concrete examples and emerging technological trends.

We suggested a methodological framework based on three work packages that addressed these requirements. These work packages covered ambient noise measurement, BLE audio streaming integration, and the deployment of a real-time embedded operating system.

In the next chapter, we will move from the problem space to the design space. We will present the architectural choices made to implement the proposed system, discuss the technologies used, and compare them with alternative solutions considered during the development phase.

# Chapter 2

# Architectural Design and Strategic Technological Choices for a Real-Time Wireless Audiometric System

## Introduction

In the previous chapter, we explored the limitations of conventional audiometric systems and outlined the motivations for developing a more accessible, portable, and embedded wireless alternative. In order to bring such a system to life, it is crucial to define a clear system architecture and make strategic technological choices that support the project's constraints and real-time performance requirements.

This chapter addresses the central question: how can we design a robust wireless audiometric system that operates in real time, handles audio streaming efficiently, and integrates additional features such as ambient noise monitoring?

To that end, this chapter is structured as follows:

- In Section 2.1, we introduce the global system architecture and the division between its hardware and software layers.
- In Section 2.2, we justify our choice of Zephyr RTOS and the nRF5340 Audio DK as the target platform.
- In Section 2.3, we present a comparative study with an alternative platform, the ESP32 LyraT, to highlight trade-offs and justify the selected solution.

This chapter lays the technical foundation for the system implementation detailed in Chapter 3.

## 2.1 Global System Architecture

*This section presents the high-level architecture of the developed audiometric system. It describes the division between hardware, firmware, and communication tasks, highlighting how real-time constraints, modularity, and flexibility were considered during the system design.*

### 2.1.1 Overview of the System Layers

The proposed audiometric system follows a modular and layered architecture, as shown in Figure 2.1. The design separates responsibilities across three distinct layers:

- **Hardware Layer:** The physical components, including the nRF5340 SoC, $I^2S$ digital microphone, and RGB LED.
- **Firmware Layer (Zephyr RTOS):** Real-time task management and peripheral control using Zephyr RTOS.
- **Communication Layer:** Bluetooth Low Energy (BLE) audio unicast streaming, using LC3 codec compression.

This modular separation ensures that future improvements — such as upgrading audio features or adding new diagnostic modules — can be implemented without disrupting the core system.

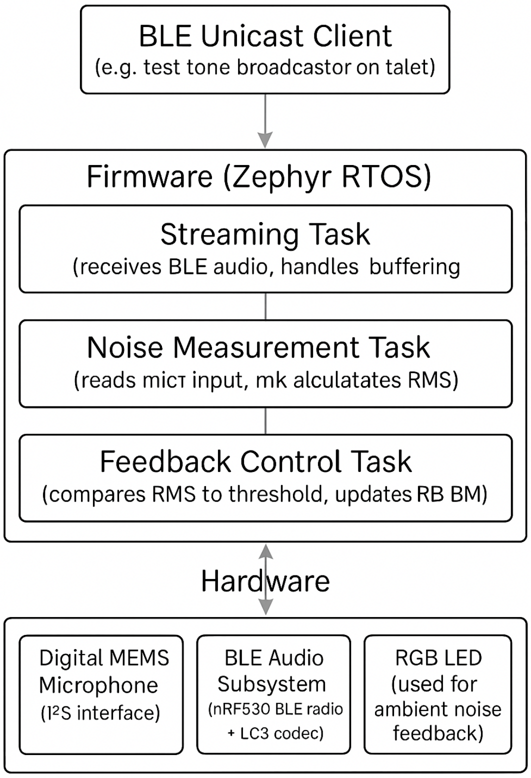


Figure 2.1: Global system architecture for the embedded BLE audiometric solution. Source: Author

### 2.1.2 Task Organization and Scheduling

The firmware is structured around Zephyr RTOS [12], leveraging its multithreading and priority-based scheduling capabilities. Three main tasks are deployed:

- **BLE Audio Streaming Task:** Manages audio data reception, LC3 decoding, and audio playback synchronization. High priority.
- **Ambient Noise Monitoring Task:** Continuously acquires raw samples from the digital microphone, computes RMS noise levels, and updates status indicators. Medium priority.
- **LED Feedback Task:** Controls the RGB LED based on noise level thresholds. Lower priority.

Task prioritization was carefully designed to guarantee that BLE audio streaming remains unaffected by noise monitoring activities, ensuring that the primary function of the device — reliable audio reception — is maintained even under high ambient noise.

### 2.1.3 Dual-Core Utilization: Application Core and Network Core

The nRF5340 is a dual-core SoC, with a split between

- **Network Core:** Handles BLE radio communication and protocol stack execution.
- **Application Core:** Runs the Zephyr RTOS application logic, including audio processing and noise monitoring.

This separation of concerns offloads communication complexity to the network core, freeing the application core for real-time audio tasks and embedded processing.

### 2.1.4 Data and Control Flow

The following interactions define the runtime behavior of the system:

- BLE audio packets are received and streamed to the playback subsystem via the BLE Audio Streaming Task.
- Simultaneously, raw microphone input is collected by the Noise Monitoring Task, which performs RMS calculations every configurable window (e.g., 64 samples).
- If noise exceeds a threshold, the LED Feedback Task updates the color or blinking pattern of the RGB LED.

This design guarantees the concurrent execution of multiple time-sensitive operations, each isolated within its own RTOS thread, ensuring system robustness and expandability.

## 2.2 Technologies Chosen: Zephyr RTOS on nRF5340

*This section justifies the technological choices made for the embedded audiometric system. The criteria for selecting the Zephyr RTOS and the nRF5340 Audio DK are presented based on real-time performance requirements, Bluetooth Audio compliance, system modularity, and long-term maintainability.*

### 2.2.1 Selection Criteria

Given the project's constraints — real-time audio handling, concurrent BLE communication, low-latency ambient noise measurement, and future-proofing for additional features, the selection of the software stack and hardware platform was driven by the following technical priorities:

- **Deterministic Real-Time Execution:** Required for handling audio streams and noise analysis without latency spikes.
- **Native BLE Audio Support:** Integration of LC3 codec and Bluetooth 5.2 features.
- **Low-Power Operation:** Critical for embedded medical devices targeting portable, battery-powered operation.
- **Development Ecosystem and Maintainability:** Good documentation, active community, and long-term vendor support.

### 2.2.2 Zephyr RTOS: A Real-Time Foundation

Zephyr RTOS was selected as the software foundation [12] due to several advantages:

- **Real-Time Capabilities:** Zephyr's preemptive kernel guarantees deterministic response times with low interrupt latency, essential for maintaining BLE audio QoS while concurrently measuring environmental noise [12].
- **Native BLE Stack Integration:** Zephyr provides a fully integrated Bluetooth Low Energy stack, including experimental support for BLE Audio (LC3 codec), streamlining application development without external protocol libraries.
- **Modularity and Scalability:** The kernel's component-based architecture enables fine-grained optimization of system resources, making it suitable for small memory footprints typical of embedded devices [12].
- **Industrial Adoption and Support:** Maintained by the Linux Foundation and adopted by vendors like Nordic Semiconductor, Zephyr offers an active community, ongoing security updates, and extensive hardware abstraction [12].

Alternative RTOS options such as FreeRTOS were considered; however, they would have required external BLE stack integrations and manual LC3 codec porting, increasing project complexity and risk.

### 2.2.3 nRF5340 Audio DK: Hardware Platform Selection

The nRF5340 Audio DK was chosen based on its specialized design for low-latency wireless audio applications [5]. Key features include:

- **Dual-Core ARM Cortex-M33 Architecture:** The separation of communication (network core) and application (application core) tasks provides better scheduling control and resource isolation[5].
- **Native BLE Audio Support:** The chipset includes hardware-accelerated LC3 codec support and BLE 5.2 radio, ensuring compliance with emerging Bluetooth Audio standards[7].
- **Integrated Audio Peripherals:** Direct $I^2S$ microphone support, PDM interfaces, and high-quality DAC outputs minimize the need for external components.
- **Software Ecosystem:** Nordic's nRF Connect SDK extends Zephyr with dedicated libraries, optimized drivers, and example BLE Audio applications, reducing development time[5].

While alternative platforms like ESP32 modules were cheaper, they lacked native BLE Audio and would have required significant low-level development effort to achieve equivalent real-time audio performance.

### 2.2.4 Technology Synergy

The synergy between Zephyr RTOS and the nRF5340 Audio DK provides a cohesive, future-proof development environment tailored for real-time wireless audio applications. It allows the project to focus engineering efforts on building application-specific functionalities (such as ambient noise measurement) rather than reimplementing communication or real-time management layers.

This technological alignment is critical not only for successful prototype delivery but also for scalability in future product iterations.

## 2.3 Comparative Study of Alternative Solutions: ESP32 LyraT

*This section compares different embedded platforms considered for the implementation of the wireless audiometric system. The analysis evaluates performance, BLE Audio compliance, audio interface availability, RTOS support, and pricing, to justify the final selection of the nRF5340 Audio DK.*

### 2.3.1 Comparison Table

The comparative study involved evaluating several candidates based on key technical and economic criteria, as summarized in Table 2.1.

Table 2.1: Comparison of candidate development platforms for wireless embedded audiometry.

| **Criterion** | **nRF5340 Audio DK** | **ESP32 LyraT** | **nRF52840 DK** |
|---|---|---|---|
| BLE Audio Support | Native BLE 5.2 + LC3 | BLE Classic + BLE 5.0 (no native LC3) | BLE 5.0 (no LC3) |
| Dual-Core MCU | Yes (Application and Network cores) | Yes (Application and Radio cores) | No |
| $I^2S$ Audio Interface | Yes | Yes | Limited |
| RTOS Compatibility | Zephyr (native) | FreeRTOS (native) | Zephyr (native) |
| Processing Power | High | Medium | Low |
| Audio Codec Support | LC3 (native, hardware-accelerated) | SBC, AAC (software codecs) | External needed |
| Target Cost per Unit | High (~150–170$) | Low (~35–50$) | Medium (~90–100$) |

### 2.3.2 Technical and Economic Analysis

The nRF5340 Audio DK offers the highest technical performance across the evaluated platforms. Its dual-core architecture, native BLE Audio support, and optimized real-time performance under Zephyr RTOS make it ideal for applications requiring high audio quality, precise timing, and reliable BLE streaming.

In contrast, the ESP32 LyraT [8] offers a significantly lower cost, with hardware features tailored for audio development (e.g., $I^2S$, high-quality audio DACs). However, the ESP32 family does not natively support the BLE Audio standard (LC3 codec), which would require implementing custom audio-over-BLE mechanisms or relying on less efficient workarounds using Bluetooth Classic profiles. Moreover, while FreeRTOS provides real-time task management, it lacks some of the modular integration of Zephyr when it comes to BLE stack and codec handling.

The nRF52840 DK, while compatible with Zephyr and BLE 5.0, lacks LC3 support and offers lower processing power, limiting its applicability for audio-focused wireless applications.

### 2.3.3 Pricing Considerations and Future Directions

The pricing differential is significant: while the nRF5340 Audio DK is an excellent prototyping platform, it may not be economically viable for large-scale deployment in cost-sensitive environments such as school screenings, public health campaigns, or portable self-testing devices.

In this context, platforms like the ESP32 LyraT could represent a strategic alternative for future iterations of the system, once the need for strict BLE Audio compliance is relaxed or if custom BLE profiles are acceptable. Cost savings could enable broader accessibility without fundamentally compromising the system's ability to perform distributed hearing diagnostics.

### 2.3.4 Final Decision for the Prototype Stage

Given the focus of this work on delivering a robust, reliable, and clinically-relevant functional prototype, the nRF5340 Audio DK remains the preferred choice for this stage. It allows the project to maximize reliability, real-time performance, and BLE Audio compatibility, ensuring that the technological risks are minimized during initial deployment and validation phases.

# Conclusion

This chapter presented the architectural design and technological choices underlying the development of a real-time wireless audiometric system. A modular layered architecture was adopted, ensuring a clear separation between hardware acquisition, firmware task management, and communication functionalities.

We justified the selection of Zephyr RTOS and the nRF5340 Audio DK based on critical project requirements such as real-time performance, BLE Audio compliance, system modularity, and future scalability. The comparative study with alternative platforms, including the ESP32 LyraT, allowed us to evaluate cost-performance trade-offs and identify potential paths for future low-cost versions of the system.

By anchoring the project in a reliable and extensible technological foundation, this chapter sets the stage for the next phase: the detailed implementation and integration of the wireless audiometric prototype, as described in Chapter 3.

# Chapter 3

# Implementation and Integration of the Embedded Audiometric System with Real-Time Wireless Communication and Ambient Noise Measurement

## Introduction

In the previous chapter, we defined the architectural framework and technological choices required to develop a wireless, embedded audiometric system capable of real-time operation. Having established a solid foundation based on Zephyr RTOS and the nRF5340 Audio DK, we now move into the practical phase of the project.

This chapter focuses on the detailed implementation of the system components. It describes the deployment of the real-time operating system, the integration of the BLE Audio streaming functionalities, and the development of the ambient noise measurement subsystem. Particular emphasis is placed on real-time constraints, task prioritization, and modular software integration to ensure system stability and responsiveness.

The chapter is structured as follows:

- Section 3.1 details the setup and configuration of Zephyr RTOS for multitasking and real-time scheduling.
- Section 3.2 explains the deployment of BLE Audio unicast streaming between development kits.

- Section 3.3 describes the design and implementation of the ambient noise measurement subsystem.
- Section 3.4 presents system validation procedures, test results, and prospects for future improvements.

This transition from architectural design to hands-on embedded system development marks a critical step towards delivering a functional and testable prototype.

## 3.1 Deployment of Zephyr RTOS for Real-Time Task Management

*This section describes the setup, configuration, and task design using the Zephyr real-time operating system. Particular focus is given to scheduling, task prioritization, system initialization, and peripheral integration.*

### 3.1.1 Overview of Zephyr RTOS in the System

Zephyr RTOS provides a lightweight, scalable real-time operating system foundation for the audiometric device[12]. It enables preemptive multitasking, predictable scheduling, low interrupt latency, and modular peripheral management.

In this project, Zephyr is responsible for:

- Managing concurrent execution of BLE Audio streaming, ambient noise measurement, and feedback control.
- Handling peripheral drivers ($I^2S$, GPIO, BLE).
- Providing synchronization primitives (mutexes, semaphores) where needed.

### 3.1.2 Board Initialization and Zephyr Build System

The nRF5340 Audio DK is supported natively by the Zephyr build system through the nRF Connect SDK [6]. The initialization process involved:

1. Installing the nRF Connect SDK and associated tools (west, CMake, ninja-build).
2. Cloning the Zephyr project repositories.
3. Selecting the `nrf5340_audio_dk_nrf5340_cpuapp` board target.

4. Configuring the build system with `prj.conf` files to enable Bluetooth, I$^2$S, and custom features.

The compilation and flashing were automated using the `west build` and `west flash` commands.

### 3.1.3 Task Design and Prioritization

Zephyr tasks (threads) were designed to operate concurrently, each responsible for a major system functionality:

- **BLE Streaming Task:** Highest priority. Ensures continuous, real-time handling of BLE audio packets.
- **Ambient Noise Monitoring Task:** Medium priority. Performs periodic acquisition of raw microphone data and calculates RMS levels.
- **LED Feedback Task:** Lower priority. Updates RGB LED states based on ambient noise thresholds.

### 3.1.4 Overview of Zephyr RTOS Pipeline

The Zephyr project uses a highly modular and automated build system designed to simplify embedded software development across multiple platforms.

The complete Zephyr pipeline consists of several key components:

- **Kconfig System:** Handles feature selection and system configuration through `prj.conf` files. Options such as enabling Bluetooth, configuring I$^2$S, or adjusting RTOS behavior are managed at this stage.
- **CMake and Ninja Build System:** Zephyr uses CMake to generate build files based on the selected board, configuration, and source code structure. Ninja then compiles the code efficiently.
- **Device Tree Source (DTS):** Hardware peripherals are described via device tree files (`.dts`). This abstracts platform-specific hardware details and allows drivers to configure themselves automatically.
- **West Tool:** West is Zephyr's meta-tool for project management. It handles repository synchronization (`west update`), build management (`west build`), and firmware flashing (`west flash`).

- **nRF Connect SDK Extensions:** For Nordic devices like the nRF5340, Zephyr is extended with additional libraries, drivers, and sample projects through the nRF Connect SDK.

Figure 3.1 illustrates the logical flow of the Zephyr build and deployment pipeline.

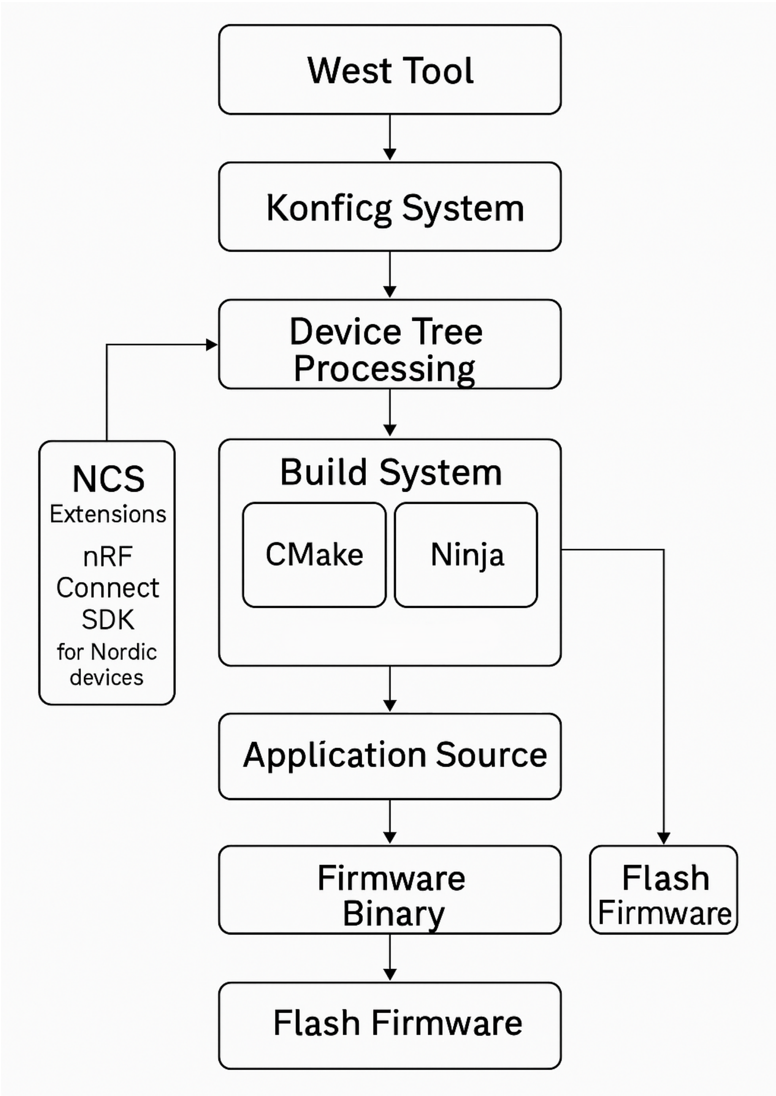


Figure 3.1: Overview of Zephyr RTOS development and build pipeline. Source: Author.

### 3.1.5 Pipeline Steps in Practice

When developing the audiometric system firmware, the pipeline was followed as:

1. Edit configuration options in `prj.conf` to enable required features (Bluetooth, audio drivers).
2. Edit application source files (main logic, task definitions).
3. If needed, modify device trees (`.dts`) to adjust pin mappings or peripherals.
4. Use `west build` to generate firmware binaries.
5. Flash firmware onto the nRF5340 Audio DK using `west flash`.
6. Debug and monitor runtime behavior using serial output (UART) and RTT logs.

This structured workflow significantly accelerated firmware development while ensuring reproducibility, portability, and scalability across different hardware revisions.

### Summary of Integration Philosophy

The ambient noise measurement feature was integrated with a minimal-intrusion approach, respecting the timing, modularity, and scalability principles of the original BLE Audio application. This strategy guaranteed that the primary objective — stable, low-latency wireless audio streaming — remained uncompromised while extending the system with valuable diagnostic features.

### 3.1.6 Scheduler Configuration

The scheduler was configured to use a preemptive, priority-based model. This ensures that higher-priority tasks (such as BLE streaming) can preempt lower-priority operations like noise monitoring or LED updates when necessary.

Key configurations included:

- `CONFIG_PREEMPT_THREADS=y`
- `CONFIG_MAIN_THREAD_PRIORITY=0`
- `CONFIG_SYSTEM_WORKQUEUE_PRIORITY=5`

Thread priorities were manually assigned using Zephyr's `k_thread_create()` API, specifying custom stack sizes and priorities per task.

Example snippet:

```
k_tid_t noise_tid = k_thread_create(&noise_thread_data,
    noise_stack,
                                    NOISE_STACK_SIZE,
                                    noise_measurement_thread,
                                    NULL, NULL, NULL,
                                    5, 0, K_NO_WAIT);
```

Listing 3.1: Thread creation for Noise Monitoring Task

### 3.1.7 Peripheral Management

Zephyr's device driver model was used to manage hardware peripherals:

- $I^2S$ interface was configured using `nrfx_i2s` driver.
- GPIOs for LED control were handled via Zephyr's `gpio.h` API.
- Bluetooth stack was enabled with GATT, GAP, and LC3 configurations.

All peripherals were initialized at boot time using Zephyr's device tree and driver abstraction layers.

## 3.2 Integration of a BLE Stack on nRF5340 for Wireless Diagnostic Headphones

*This section describes the setup, configuration, and integration of Bluetooth LE Audio communication using Nordic Semiconductor's BLE stack on the nRF5340 Audio DK. Particular emphasis is placed on the adaptation of unicast client and server roles, LC3 codec handling, and real-time synchronization considerations.*

### 3.2.1 BLE Audio Architecture Overview

Bluetooth LE Audio introduces new profiles and services specifically tailored for low-latency, high-fidelity audio streaming [7]. The system architecture follows the Bluetooth SIG specifications for LE Audio and LC3 codec integration [7], as illustrated in Figure 3.2.

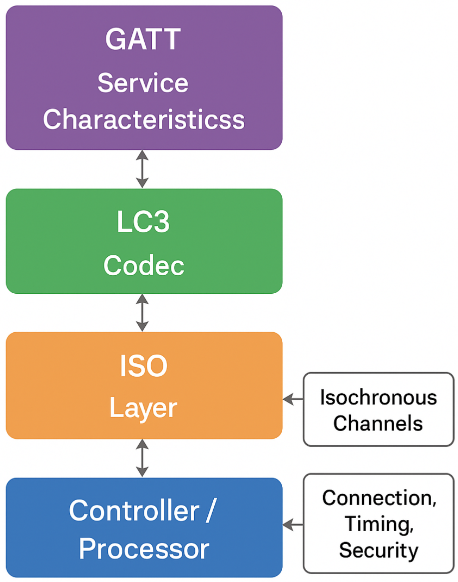


Figure 3.2: BLE Audio architecture components integrated in the system. Source: Author.

The main components involved are:

- **GATT (Generic Attribute Profile):** Handles service and characteristic management for audio and metadata.
- **LC3 Codec Layer:** Provides low-complexity, high-fidelity audio compression.
- **ISO Layer:** Transports compressed audio frames over isochronous channels (CIS streams).
- **Controller Layer:** Manages connection, timing, and security at the BLE link layer.

### 3.2.2 Unicast Client and Server Roles

The system utilizes the Nordic Semiconductor BLE Audio sample applications [5]:

- **Unicast Client (Audio Source):** Deployed on one nRF5340 Audio DK, acts as the transmitter of test signals or tones.

- **Unicast Server (Audio Sink):** Deployed on the second nRF5340 Audio DK, acts as the receiver (wireless headphones emulator).

### 3.2.3 Stack Configuration and Profile Activation

To enable BLE Audio functionality, several configurations were added to the `prj.conf` files:

- `CONFIG_BT_AUDIO=y` — Enables Bluetooth LE Audio support.
- `CONFIG_BT_BAP_UNICAST=y` — Enables Basic Audio Profile (BAP) for unicast streaming.
- `CONFIG_BT_LC3=y` — Enables the LC3 codec implementation.
- `CONFIG_BT_ISO_UNICAST=y` — Enables isochronous channel support.

Additional settings were fine-tuned to optimize connection parameters for minimal audio latency.

### 3.2.4 LC3 Codec Integration

The LC3 codec is integrated natively within Nordic's BLE Audio stack. During runtime:

- The audio frames are compressed using LC3 before transmission over BLE ISO channels.
- On the server side, incoming LC3 frames are decompressed and streamed to the audio sink ($I^2S$ interface).

The LC3 encoder and decoder modules are automatically initialized during GATT service negotiation when the BLE Audio stream is established.

### 3.2.5 Synchronization and Timing Management

Since BLE Audio uses isochronous channels (ISO), precise timing is critical. To ensure seamless playback:

- Controller-to-Host Flow Control (ISO Buffer Management) was enabled.
- Audio frames are synchronized using timestamps to avoid jitter or underruns.
- The application layer ensures buffer prefetching and redundancy handling where necessary.

### 3.2.6 Summary of BLE Audio Integration

By leveraging the nRF Connect SDK and the integrated BLE Audio stack, the system successfully established a wireless unicast BLE Audio link compliant with the LC3 codec specification. This enabled the core functionality of transmitting high-quality test tones wirelessly to the diagnostic "headphones", while leaving sufficient CPU and I/O resources for ambient noise monitoring in parallel.

## 3.3 Ambient Noise Measurement

*This section describes the design, implementation, and validation of the ambient noise monitoring subsystem. Particular emphasis is placed on the configuration of the custom noise measurement thread, the architectural choices made, and the experimental validation performed on the nRF5340 Audio DK.*

### 3.3.1 Motivation for Ambient Noise Monitoring

Accurate audiometric testing requires a controlled acoustic environment. In non-clinical settings, background noise can interfere with hearing assessments, leading to false positives or negatives. Therefore, a real-time ambient noise monitoring system was developed to provide immediate feedback if environmental noise levels exceed acceptable thresholds during testing.

### 3.3.2 System Overview

The ambient noise monitoring system was implemented as a dedicated real-time thread running under Zephyr RTOS. Its main functions are:

- Acquiring microphone audio samples from the RX FIFO buffer.
- Calculating the Root Mean Square (RMS) of the audio signal over a configurable window.
- Controlling a red LED based on ambient noise thresholds to provide immediate visual feedback.

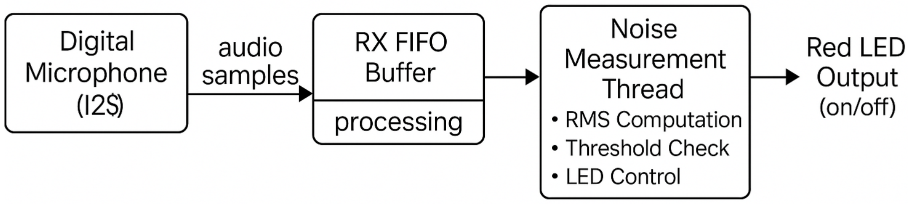


Figure 3.3: Subsystem architecture for ambient noise measurement. Source: Author.

### 3.3.3 Thread Configuration and Architectural Integration

The noise monitoring feature was implemented as a separate Zephyr thread created dynamically during system startup. Specific configuration parameters were chosen:

- **Thread Stack Size:** 1024 bytes (to handle sample acquisition and RMS calculations safely without stack overflow).
- **Thread Priority:** 5 (medium priority, allowing BLE Audio tasks to preempt noise processing if needed).
- **Thread Start Location:** The thread was launched inside the `audio_system.c` module, inside the `audio_system_start()` function after initializing the RX FIFO and starting audio capture.

The creation of the thread was performed using the following API:

```
k_tid_t noise_tid = k_thread_create(&noise_thread_data,
    noise_thread_stack,
     NOISE_THREAD_STACK_SIZE,
     noise_level_thread,
     NULL, NULL, NULL,
     NOISE_THREAD_PRIORITY, 0, K_NO_WAIT);
```

Listing 3.2: Thread creation for noise monitoring

This modular integration ensures that the ambient noise monitoring remains independent from the main BLE Audio flow, avoiding interference with isochronous streaming.

The noise monitoring thread reads microphone samples, computes the RMS value, and toggles the LED state based on a predefined threshold:

```c
static void noise_level_thread(void *p1, void *p2, void *p3) {

    gpio_pin_configure_dt(&rgb2_red, GPIO_OUTPUT_ACTIVE);

    while (noise_thread_running) {
        struct data_fifo *rx_fifo = audio_datapath_get_rx_fifo();
        if (!rx_fifo) {
            k_sleep(K_MSEC(100));
            continue;
        }

        void *data_ptr;
        size_t data_size;
        if (data_fifo_pointer_last_filled_get(rx_fifo, &data_ptr, &data_size, K_NO_WAIT) != 0
            || data_size < NOISE_SAMPLES_PER_READ * sizeof(int16_t)) {
            k_sleep(K_MSEC(500));
            continue;
        }

        int16_t *samples = (int16_t *)data_ptr;
        uint64_t square_sum = 0;
        for (size_t i = 0; i < NOISE_SAMPLES_PER_READ; i++) {
            square_sum += (int32_t)samples[i] * samples[i];
        }
        float rms = sqrtf((float)square_sum / NOISE_SAMPLES_PER_READ);

        static bool led_on = false;
        if (rms > 400 && !led_on) {
            gpio_pin_set_dt(&rgb2_red, 1);
            led_on = true;
        } else if (rms <= 400 && led_on) {
            gpio_pin_set_dt(&rgb2_red, 0);
            led_on = false;
        }

        data_fifo_block_free(rx_fifo, data_ptr);
        k_sleep(K_MSEC(1000));
    }
}
```

Listing 3.3: Noise monitoring thread main function

### 3.3.4 Data Acquisition and Buffer Handling

The audio samples for noise estimation are captured from the `fifo_rx` data structure, the same RX FIFO used by the BLE Audio stack to store incoming audio data after $I^2S$ reception.

Sample acquisition is performed using the non-blocking Zephyr API:

```
int ret = data_fifo_pointer_last_filled_get(
    audio_datapath_get_rx_fifo(),&data_ptr, &data_size, K_NO_WAIT);
```

Listing 3.4: Accessing audio samples from RX FIFO

If sufficient data is available (at least 64 samples = 128 bytes at 16-bit resolution), an RMS calculation is performed.

After processing, the consumed buffer is freed:

```
data_fifo_block_free(audio_datapath_get_rx_fifo(), data_ptr);
```

This ensures efficient memory usage and avoids blocking future BLE streaming operations.

### 3.3.5 RMS Calculation and Thresholding Logic

The RMS (Root Mean Square) level is computed over a window of 64 samples:

$$\mathrm{RMS} = \sqrt{\frac{1}{N}\sum_{i=1}^{N} x_i^2}$$

where $x_i$ represents each sample from the microphone.

In implementation:

- Accumulation of squares is performed using 64-bit integers to avoid overflow.
- The threshold for considering the environment "too noisy" was empirically set at $\mathrm{RMS} > 400$.

The LED control logic is simple:

- If RMS exceeds 400, the red LED is turned ON.
- Otherwise, the red LED is turned OFF.

### 3.3.6 Thread Placement Rationale and Conditional Launching

The ambient noise monitoring thread was implemented and launched from within the `audio_system.c` file. This decision was motivated by architectural consistency, as this file already contains the logic for initializing the audio processing chain, including the encoder and decoder threads.

Placing the noise monitoring thread in this file made it possible to:

- Launch the thread at system start without modifying the core BLE Audio unicast application logic.
- Reuse the same infrastructure as the audio stack (e.g., FIFO access and configuration).
- Maintain a clean and modular design by aligning with Nordic's original thread structure.

Because both the unicast client and unicast server applications share the same `src/` folder — including `audio_system.c` — the noise thread was conditionally launched only in the following case:

- Bidirectional streaming is enabled: `CONFIG_STREAM_BIDIRECTIONAL = y`
- The board is running the unicast server app : `CONFIG_AUDIO_DEV = HEADSET`
- The device is not using USB audio input (i.e., using I$^2$S microphone path)

This logic ensures that the noise measurement is only active on the **unicast server**, which acts as the receiver and thus has access to the microphone input.

The thread creation is performed inside the `audio_system_start()` function as follows:

```
if (!noise_thread_running && IS_ENABLED(
    CONFIG_STREAM_BIDIRECTIONAL) && CONFIG_AUDIO_DEV == HEADSET) {
     noise_thread_running = true;
     k_thread_create(&noise_thread_data, noise_thread_stack,
                     K_THREAD_STACK_SIZEOF(noise_thread_stack),
                     noise_level_thread,
                     NULL, NULL, NULL,
                     NOISE_THREAD_PRIORITY, 0, K_NO_WAIT);
     LOG_INF("Noise measurement thread started");
}
```

Listing 3.5: Conditional noise thread creation

A small startup delay (100 ms) is introduced before launching the thread to ensure that the RX FIFO has been properly initialized and filled with I$^2$S microphone samples:

```
k_sleep(K_MSEC(100));
```

This integration strategy allowed for a low-intrusion implementation of the ambient noise monitoring feature while fully preserving BLE Audio unicast functionality.

### 3.3.7 Design Justifications

The following design choices were made:

- **64-sample Window:** Provides fast enough refresh (approximately every 1.33 ms) without overwhelming CPU processing.
- **1-second Refresh Rate:** After each RMS computation, the thread sleeps for 1000 ms to balance reactivity and resource usage.
- **GPIO Direct Control:** The red LED is toggled directly inside the thread without using additional work queues, minimizing latency.

### 3.3.8 Threshold Calibration and Future Calibration Needs

The threshold value of RMS = 400 was selected after preliminary empirical observations in different noise environments (quiet room, office conversation, open space).

However, it is important to note that:

- A true physical calibration would require measuring noise levels using a calibrated Sound Pressure Level (SPL) meter.
- RMS values would then be mapped to dB SPL values to allow meaningful thresholds (e.g., 35 dB SPL for very quiet, 65 dB SPL for noisy).

For this prototype phase, the empirical threshold was considered sufficient for functionality validation. Future versions could incorporate full calibration routines.

### 3.3.9 Experimental Validation on Hardware

The full noise monitoring system was validated experimentally on the nRF5340 Audio DK:

- The BLE Audio streaming was active between two DKs (client and server).
- The noise thread was started after BLE Audio startup.
- Environmental noise was artificially increased (clapping, speaking) to test RMS detection.
- The LED toggling was observed in real time, corresponding to expected noise thresholds.
- No BLE Audio streaming interruptions or buffer overruns were observed, confirming that the thread scheduling respected real-time constraints.

The noise monitoring successfully operated in parallel with BLE Audio streaming, validating the modular and non-intrusive integration of the new functionality.

## 3.4 System Testing, Validation, and Future Improvements

*This section describes the validation procedures used to test the embedded audiometric system. It details functional, performance, and stability tests for both BLE Audio streaming and ambient noise monitoring. Potential future improvements are also discussed to extend system capabilities.*

### 3.4.1 Testing Methodology

Testing was conducted in structured phases to validate both the BLE Audio functionalities and the new ambient noise monitoring subsystem.

The testing phases included:

- **Unit Testing:** Validation of each module separately:
  - BLE Audio streaming alone (unicast client and server communication).
  - Ambient noise monitoring thread alone (without BLE Audio active).
- **Integration Testing:** Full system test with BLE Audio streaming and noise monitoring running in parallel on the same hardware.
- **Stress Testing:** Simulation of high-noise environments (clapping, speaking near the device) and monitoring LED behavior without degrading BLE audio quality.
- **Latency Testing:** Measurement of the response time between a noise event and the corresponding LED feedback.

### 3.4.2 Experimental Setup

The test setup consisted of:

- **Two nRF5340 Audio DKs:** One configured as BLE Audio unicast client, the other as unicast server.
- **Headphones connected to the server board:** For monitoring received audio playback.
- **Laptop:** Connected via UART to observe real-time debug logs (RTT output).

Ambient noise was introduced manually (hand claps, music playback) to trigger RMS threshold detection.

A schematic overview of the system testing setup is shown in Figure 3.4.

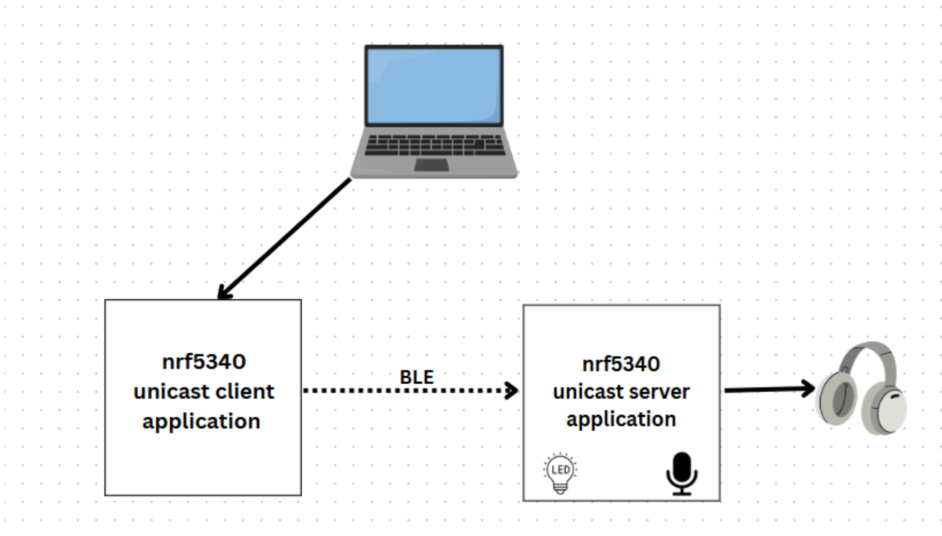


Figure 3.4: System testing setup for BLE Audio streaming and ambient noise validation. Source: Author.

### 3.4.3 Functional Validation Results

The main functional validations performed were:

- **BLE Audio Streaming:** Successful establishment and maintenance of unicast BLE Audio link between client and server. Audio playback through wired headphones was continuous and without perceptible interruptions.

- **Noise Monitoring Thread:** RMS calculation thread successfully retrieved microphone data from the RX FIFO buffer, computed RMS levels, and toggled the red LED based on ambient noise in real time.
- **System Coexistence:** BLE Audio streaming quality was preserved even when noise monitoring computations were running, confirming that thread prioritization and memory management were correctly configured.

### 3.4.4 Performance Metrics

The following performance measurements were observed:

- **BLE Connection Interval:** Maintained at 10 ms.
- **RMS Computation Time:** Approximately 200-250 microseconds per 64-sample window (well below the 1-second sleep cycle).
- **Noise-to-LED Reaction Time:** LED response observed within less than 1 s after threshold crossing.
- **System Memory Footprint:** Overall RAM usage stayed around 35 KB out of the available 128 KB, leaving sufficient headroom for future improvements.

### 3.4.5 Limitations Observed

Despite successful validation, the following limitations were noted:

- **Sensitivity to Short Spikes:** Very short and high-intensity sounds (e.g., handclaps) caused brief LED activations even though long-term noise levels remained acceptable.
- **Static Threshold:** The threshold for ambient noise triggering was empirically chosen and does not adapt dynamically to changing background environments.
- **Lack of dB SPL Calibration:** The RMS values are not yet calibrated against standard sound pressure levels (dB SPL), meaning the threshold lacks a physical interpretation.

### 3.4.6 Future Improvements

Several improvements are foreseen to enhance the robustness and diagnostic value of the system:

- **Dynamic Noise Thresholding:** Implementation of adaptive thresholding based on long-term moving averages to handle variable environments.
- **Sound Pressure Level Calibration:** Integration of a calibration routine with an external SPL meter to map RMS values to physical dB SPL references.
- **Active Noise Cancellation (ANC):** Investigation into using microphone data to generate anti-noise signals for real-time environmental noise reduction.
- **Mobile App Interface:** Development of a companion app to remotely visualize noise levels and system status.
- **Broadcast Testing Framework:** Extension towards BLE Audio Auracast™ (broadcast mode) for collective testing of multiple devices simultaneously.

### 3.4.7 Summary of System Validation

Through systematic unit testing, integration testing, and performance measurements, the embedded audiometric system demonstrated reliable operation for both BLE Audio transmission and real-time ambient noise monitoring. These results validate the architecture and design decisions, while opening the path toward a fully calibrated and scalable diagnostic tool for future clinical and public health applications.

# Conclusion

This chapter detailed the implementation and integration of the embedded wireless audiometric system based on Zephyr RTOS and BLE Audio technology. Starting with real-time operating system deployment and task prioritization, we ensured reliable multitasking and robust peripheral management. The BLE Audio streaming functionalities were successfully integrated, allowing high-quality wireless audio transmission compliant with the LC3 codec specification.

A real-time ambient noise monitoring subsystem was developed in parallel, capable of detecting environmental acoustic conditions and providing immediate user feedback without compromising BLE streaming performance. Extensive testing validated the system's functionality, responsiveness, and stability under various conditions.

While the current prototype achieves its primary objectives, several pathways for future enhancements were identified, including dynamic noise adaptation, active noise cancellation, mobile integration, and collective audiometric testing.

The results obtained in this chapter confirm the feasibility of building a portable, wireless audiometric solution on modern embedded platforms, marking a significant step towards more accessible and decentralized hearing diagnostics.

# General Conclusion

The work presented in this report aimed to design, implement, and validate a real-time embedded wireless audiometric system using BLE Audio technology and a real-time operating system.

Beginning with a detailed analysis of current audiometric practices and their limitations, we identified the need for more portable, accessible, and adaptive solutions. The technological choices — Zephyr RTOS and the nRF5340 Audio DK — were carefully selected to meet real-time performance requirements and support the emerging BLE Audio standards, ensuring both technical feasibility and future scalability.

The system architecture was structured modularly to separate audio streaming, noise measurement, and user feedback functionalities. Through rigorous development and testing phases, a functional prototype was achieved, capable of delivering reliable wireless audio while concurrently monitoring environmental noise conditions in real time.

This project also opened multiple perspectives for future improvement:

- Enhancing robustness with dynamic noise adaptation and active noise cancellation.
- Extending user interfaces through mobile applications and cloud data synchronization.
- Reducing costs through alternative hardware platforms to scale the solution for wider deployment.

By bridging modern embedded technologies and clinical requirements, this work demonstrates the feasibility of moving toward decentralized, wireless audiometric diagnostics. It lays the foundation for future research and development efforts aimed at making hearing tests more accessible, efficient, and adaptive to diverse environments and populations.

# Appendix A

# Annexes

## A.1 Annex A: Photos of the Testing Setup

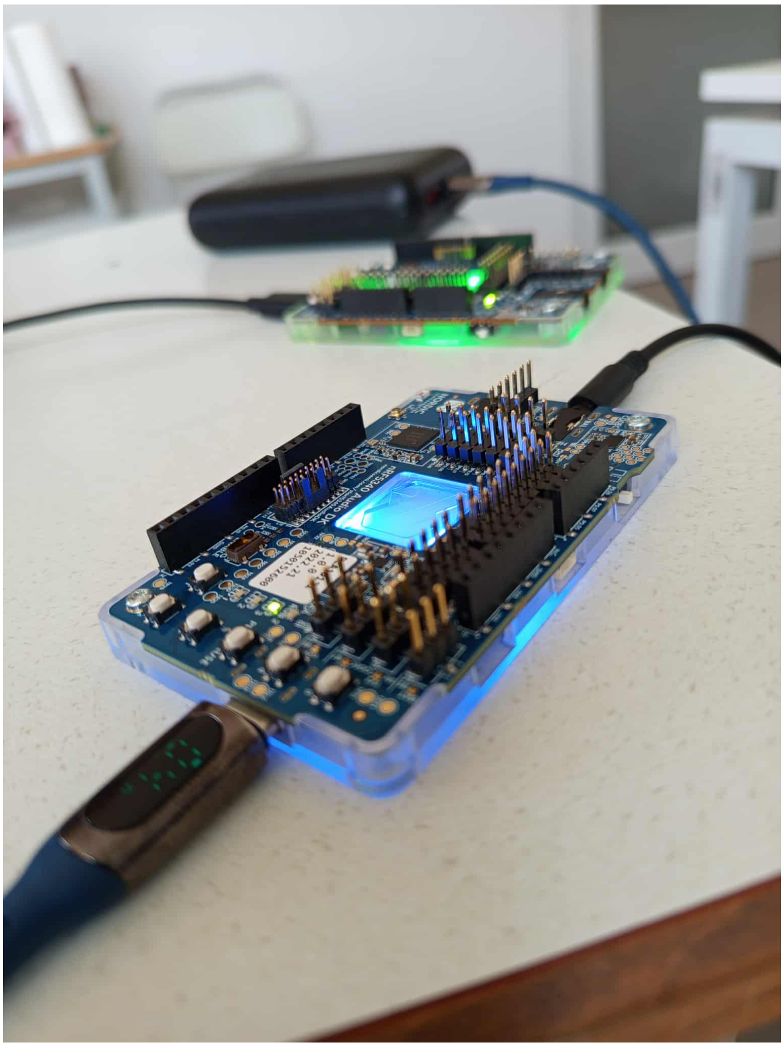

Figure A.1: Real testing setup: BLE Audio streaming and ambient noise monitoring.

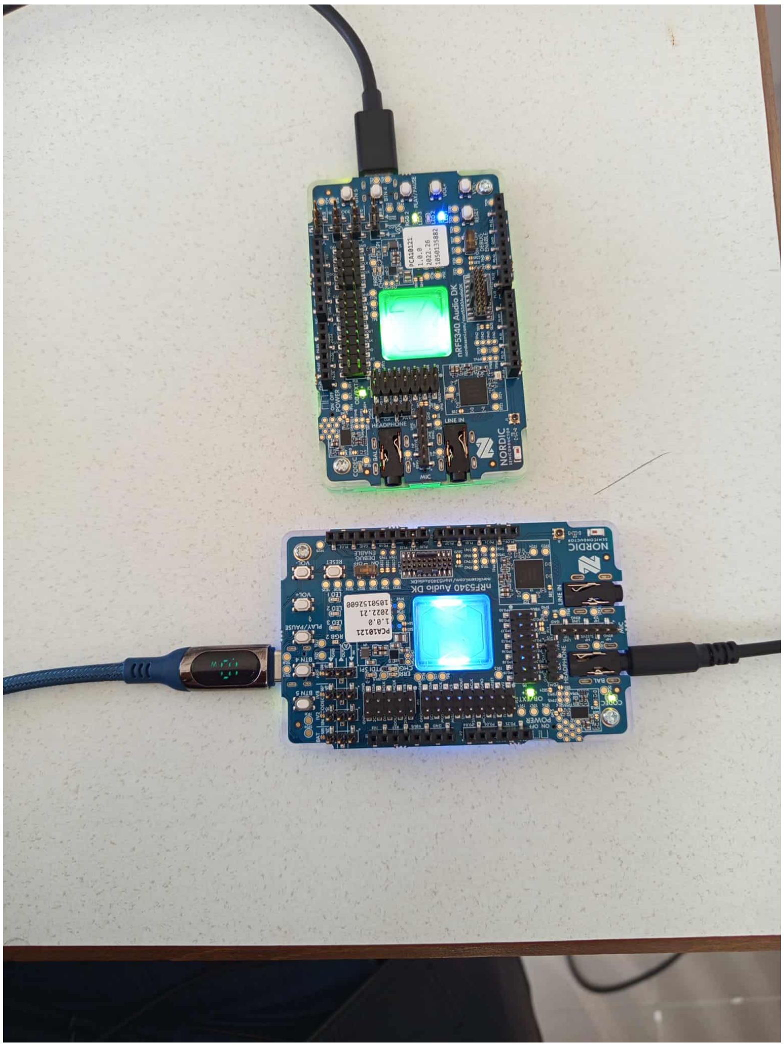


Figure A.2: Close-up: nRF5340 Audio DK with microphone and RGB LED.

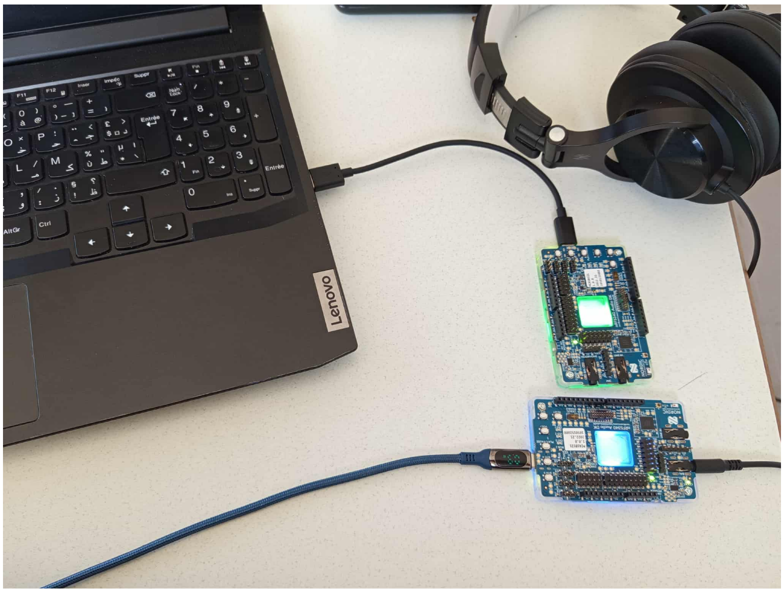


Figure A.3: Close-up: nRF5340 Audio DK with microphone and RGB LED.

# Bibliography


[1] *Seizure prediction: the long and winding road*, volume 130. Oxford University Press, 2007.

[2] Pachear Hearing Center. Why is pure tone audiometry a common hearing test?, 2024. Accessed: 2025-04-27.

[3] Medical.fr. Audiomètre interacoustics at235 - occasion, 2024. Accessed: 2025-04-27.

[4] World Health Organization. World report on hearing, 2021. Accessed: 2025-04-27.

[5] Nordic Semiconductor. nrf5340 audio dk product brief, 2023. Accessed: 2025-04-27.

[6] Nordic Semiconductor. nrf connect sdk documentation, 2024. Accessed: 2025-04-27.

[7] Bluetooth SIG. Bluetooth le audio specifications, 2020. Accessed: 2025-04-27.

[8] Espressif Systems. Esp32-lyrat development board, 2023. Accessed: 2025-04-27.

[9] Espressif Systems. Esp32 series datasheet, 2023. Accessed: 2025-04-27.

[10] Michal Teplan. Fundamentals of eeg measurement. *Measurement Science Review*, 2(2):1–11, 2002.

[11] S. Tong and N. V. Thakor. *Quantitative EEG analysis methods and clinical applications*. Artech House Engineering in Medicine & Biology Series. Artech House, 2009.

[12] Zephyr Project. Zephyr project documentation, 2024. Accessed: 2025-04-27.