Technische Universität Braunschweig

Robust Hardware & Software Systems

Prof. Andrés Gómez, Dr. Juan Felipe Gutiérrez Gómez, Chen Jiang, Negar Halakou, Chien-Chia Huang

Low-Power Embedded Systems - WiSe 2025/26

Laboratory Final Projects

Handout Date : 17.12.2025
Due Date : 04.02.2026

1 Overview

This repository serves as a comprehensive resource for students and developers working on embedded systems projects. It includes detailed information about the available development platforms, sensors, and kits that can be leveraged to design and implement a wide range of final projects. Each platform and kit is described with its key features and supported software environments, ensuring compatibility and ease of use.

In addition to hardware resources, the repository provides a collection of innovative project ideas that demonstrate practical applications of embedded systems concepts. These ideas cover areas such as **wireless communication, robotics, sensor integration, user interface design, and real-time data processing**, offering inspiration and guidance for creating functional, real-world solutions.

The hardware resources provided are at your disposal to create innovative applications for your final project. You can explore the following areas: **Low Power Communication** (BLE, BLE-Mesh, IEEE 802.15.4, Thread/CoAP) **Machine Learning**(TensorFlow, custom frameworks), **Graphical User Interfaces**(LCD display).

2 Available Platforms

Board	Manufacturer	Key Features	Supported Software
Launchpad CC2652R1	Texas Instruments	BLE / IEEE 802.15.4	Baremetal, Zephyr RTOS, FreeRTOS
SensorTag LPSTK-CC1351R	Texas Instruments	BLE / IEEE 802.15.4 / Sensing	Baremetal, Zephyr RTOS, FreeRTOS
Launchpad CC1352P1	Texas Instruments	BLE / IEEE 802.15.4 / Sensing	Baremetal, Zephyr RTOS, FreeRTOS
NRF52840DK	Nordic Semiconductor	BLE / IEEE 802.15.4	Zephyr RTOS, nRF Connect SDK
Dongle NRF52840	Nordic Semiconductor	BLE / IEEE 802.15.4	Zephyr RTOS, nRF Connect SDK
STM32L433 Nucleo-64	STMicroelectronics	Ultra-low-power MCU	Zephyr RTOS
Nordic Thingy:53	Nordic Semiconductor	Bluetooth LE, Bluetooth Mesh, Thread, Zigbee, Matter, proprietary 2.4 GHz, NFC	Zephyr RTOS
STM32F746G Discovery	STMicroelectronics	Ethernet, display, audio in/out	Zephyr RTOS
Pololu 3pi+ 2040 Robot	Pololu	Motors, IR sensors, display	Manufacturer GitHub repository
Pololu Zumo 2040 Robot	Pololu	Motors, IR sensors, display	Manufacturer GitHub repository

Table 1: Development platforms accessible in the LPES laboratory.

Kits and Displays

Device	Manufacturer	Description	Support
Educational BoosterPack MK II	Texas Instruments	Display, sensors, joystick, microphone, buzzer	LaunchPad ecosystem
Adafruit TFT Shield 2.8"	Adafruit	2.8" TFT touch shield (320x240)	Zephyr drivers and Arduino support
M5Stack Core2 ESP32 IoT Kit	M5Stack	ESP32-D0WDQ6-V3 based IoT console with capacitive touch display	UIFlow, MicroPython, Arduino, .NET nanoFramework
Grove Vision AI Module	Seeed Studio	Thumb-sized AI camera with people-detection and custom ML models	Custom firmware support
Touch LCD BoosterPack	Texas Instruments	Touch-enabled color display BoosterPack	Code Composer Studio projects
Sharp Memory LCD	Texas Instruments	LS013B7DH03 ultra low-power TFT display	Code Composer Studio bare-metal library

Table 2: Display shields and system kits for rapid prototyping.

Sensors and Actuators

Device	Notes
Infrared Reflective Sensor	Simple digital indicator for proximity or line detection
Deek Robot Joystick	Analog joystick with X/Y axes
OV5647-62 FOV camera	5 MP camera module suited for embedded vision prototypes
OPT3001 ambient light sensor	I^2C digital light sensor with Zephyr support
VEML7700 ambient light sensor	High-resolution light sensing via I^2C
SHTC3 environmental sensor	Temperature and humidity sensing with Zephyr drivers
Grove Moisture Sensor	Analog soil-moisture transducer that can be read via ADC
Grove GPS Module	UART-based Air530 GPS receiver for geo-positioning
Sensor BoosterPack	Ambient light, temperature, geomagnetic, inertial, and environmental sensors
BOOSTXL-ULPSENSE	Demonstrates the ultra-low-power sensor controller of the CC13x2/CC26x2

Table 3: Sensors and actuators that can be combined with the supported platforms.

3 Project Ideas

The following concept descriptions outline representative final projects. Each subsection highlights the main objective, architectural choices, and suggested implementation details so that teams can adapt the ideas to their own learning goals.

3.1 Vision-Based Light-Following Robot

Implement a vision-guided robot that follows a flashlight in real time using a Pololu Zumo or Pololu 3pi+ platform paired with an ESP32-S3 XIAO Sense module featuring an OV2640 camera. The ESP32-S3 captures frames, detects the brightest region, and transmits motion commands to the robot controller, which performs low-level motor control.

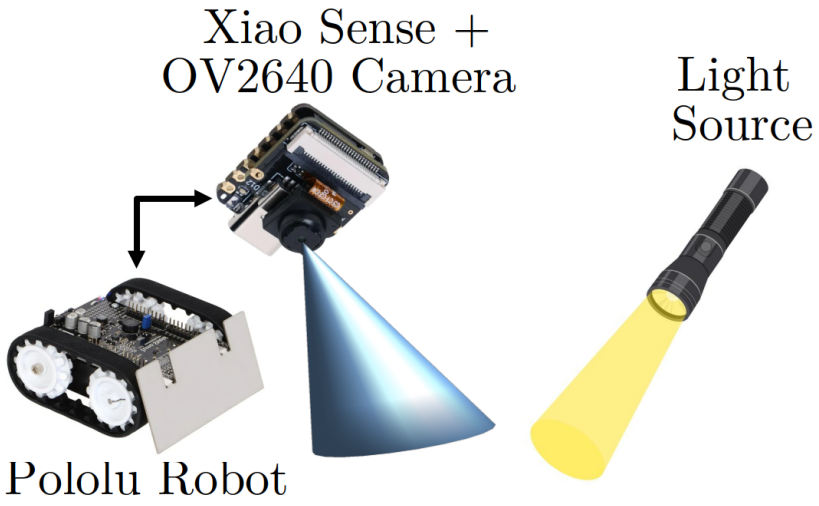

Figure 1: Vision-based light-following robot concept.

Implementation Details

- Integration of Zephyr RTOS on the ESP32-S3 XIAO Sense board for camera handling and task scheduling.
- Configuration and use of the OV2640 camera to capture image frames at a suitable resolution and frame rate.
- Implementation of a light detection and tracking algorithm, for example by identifying the brightest region in the image and computing its centroid.
- Mapping of the detected light position to motion commands (e.g., move forward, turn left, turn right, stop).
- Development of a communication interface between the ESP32-S3 and the robot controller.
- Implementation of bare-metal firmware on the Pololu Zumo or 3pi+ MCU to control motors and execute received motion commands.
- Basic closed-loop motor control to ensure stable and smooth robot motion.
- Consideration of low-power operation, including reduced camera frame rates and efficient task execution.

3.2 Pong Game over OpenThread and CoAP

Create a two-player Pong experience over an OpenThread network implemented with Zephyr RTOS. Two CoAP clients act as paddles that send position updates to a CoAP server. The server maintains game logic and renders the scene on the Adafruit 2.8" TFT Shield.

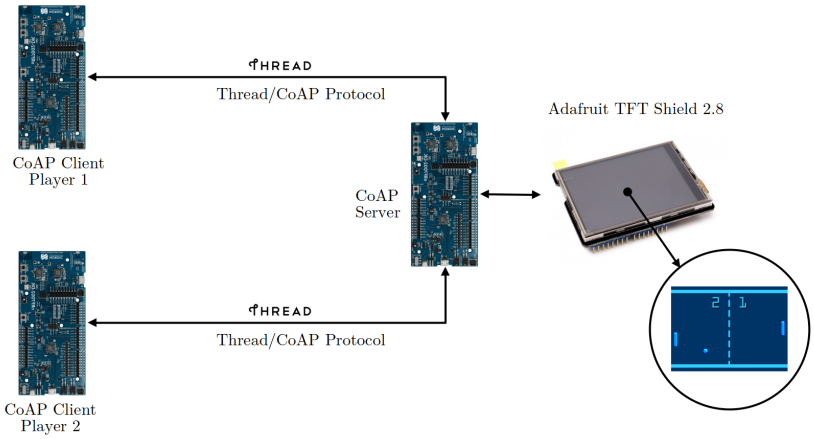

Figure 2: OpenThread and CoAP-based Pong architecture.

Implementation Details

- Implementation of a Thread/CoAP-based network using Zephyr RTOS and nRF Connect SDK.
- Development of two CoAP clients to serve as player remote controls, sending requests to move the paddles.
- Development of a CoAP server to handle client requests, process the Pong game logic, and maintain the game state.
- Implementation of Pong game mechanics, including paddle movement, ball physics, and collision detection.
- Integration of drivers to render the game interface on the Adafruit 2.8" TFT Shield as the visual display.

3.3 Theft Detection and Geo-Tracking with CC1352P1

Develop a theft detection system where a CC1352P1 LaunchPad reads position data from a GPS module, compares it against a safe-radius threshold, and transmits alerts over IEEE 802.15.4 to a monitoring node that plots the device on Google Maps.

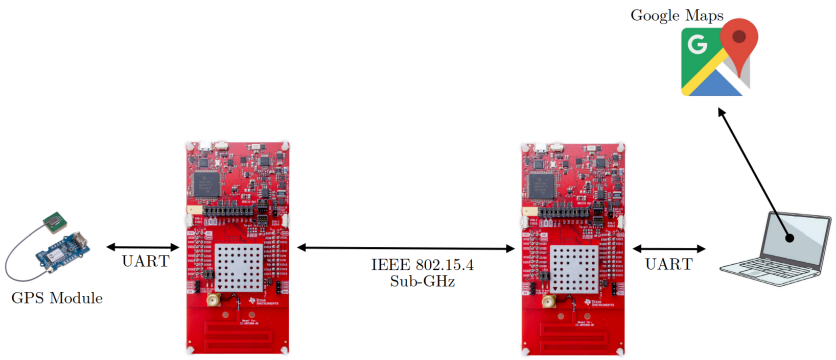

Figure 3: Geo-tracking pipeline using GPS and IEEE 802.15.4.

Implementation Details

- Integration of the Zephyr RTOS on the CC1352P1 LaunchPad to manage GPS module readings and IEEE 802.15.4 communication.
- Development of a GPS interface to read geo-position data at a specific sampling rate using UART or I2C communication.
- Event-based detection system:
  - Compare current GPS coordinates with a predefined reference point.
  - Trigger an alert if movement exceeds a specified distance.
- Implementation of IEEE 802.15.4 protocol for wireless transmission of alert packets containing GPS position to a receiver node.
- Receiver node development to decode the packet and extract GPS coordinates.
- Integration of Python and mapping libraries to display location on Google Maps.
- Optimization of power management on the CC1352P1 to ensure efficient operation.

3.4 Batteryless Sensor System with ePEAS AEM10990

Create a fully energy-harvesting sensor node that leverages the ePEAS AEM10330/AEM10990 power management ICs to collect energy from ambient light. A CC2652R1 development board samples environmental data and renders it on the Sharp 128x128 Memory LCD while remaining within the harvested power budget.

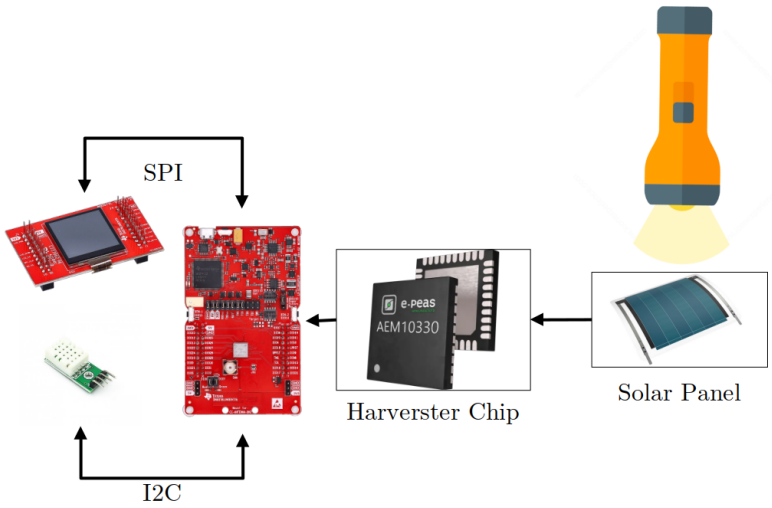

Figure 4: Batteryless sensing concept using ePEAS harvesters.

Implementation Details

- Use the ePEAS AEM10330 to harvest energy from an ambient light source (e.g., sunlight or indoor lighting) to power the entire system. It will regulate the harvested energy to charge a supercapacitor or battery and provide a stable voltage.
- Program the CC2652R1 using TI baremetal programming to efficiently manage power consumption while handling sensor reading and communication tasks.
- Integrate a suitable sensor (e.g., light sensor, temperature sensor, or environmental sensor) with the CC2652R1 to collect real-time data.
- Display the sensor readings using the Sharp 128x128 Memory LCD. The Sharp LCD display only consumes power during updates, ensuring that the system remains energy-efficient.
- The system will operate in ultra-low power mode, where the CC2652R1 and the sensor will sleep or enter low-power states when not actively reading data or updating the display.
- The Sharp 128x128 Memory LCD will show the sensor data (e.g., temperature, light intensity, environmental data) periodically or when significant changes are detected.

3.5 Smart Sensing Agriculture via BLE

Build a BLE-enabled sensing network where an NRF52840DK peripheral measures temperature, humidity, and ambient light before transmitting the data to a laptop acting as the BLE central. The laptop displays live readings and controls a smart light source based on the sensor-derived illumination level.

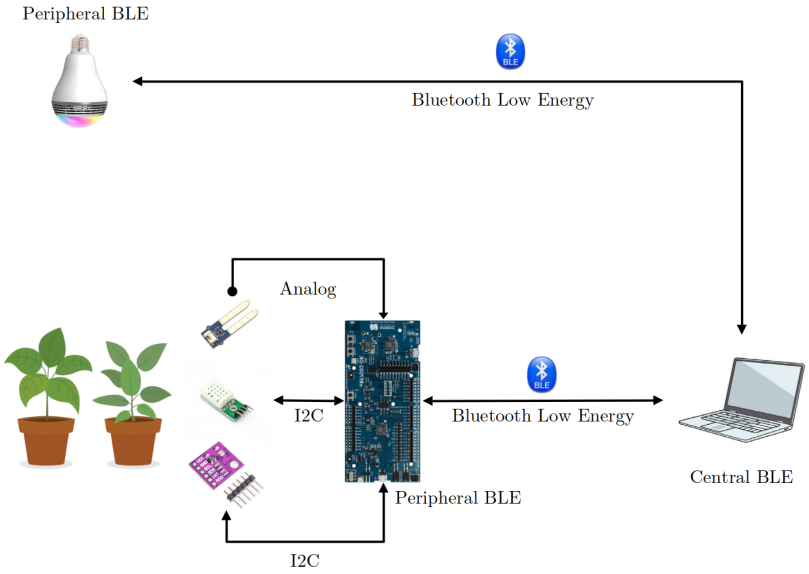

Figure 5: BLE-based smart agriculture monitoring system.

Implementation Details

- Development of Zephyr drivers to read analog values from the humidity sensor.
- Integration of Zephyr drivers to read digital values from the temperature and ambient light sensors using the I2C interface.
- Implementation of a BLE peripheral on the nRF52840DK to transmit temperature, humidity, and light intensity data.
- Development of a Python script acting as a BLE central node, subscribing to notifications and receiving sensor data.
- Creation of a graphical user interface (GUI) on the laptop to display real-time sensor values in a user-friendly format.
- Implementation of monitoring alerts for environmental variables (e.g., thresholds for temperature, humidity, and light intensity).
- Development of a Python script to control the brightness of a BLE-enabled smart light source based on ambient light sensor values.

3.6 Voice-Controlled Interface on STM32F746G Discovery

Program the STM32F746G Discovery board with Zephyr RTOS to recognize predefined voice commands captured through the on-board audio input. Provide immediate user feedback via the touchscreen interface and optional touch inputs to complement the speech pipeline.

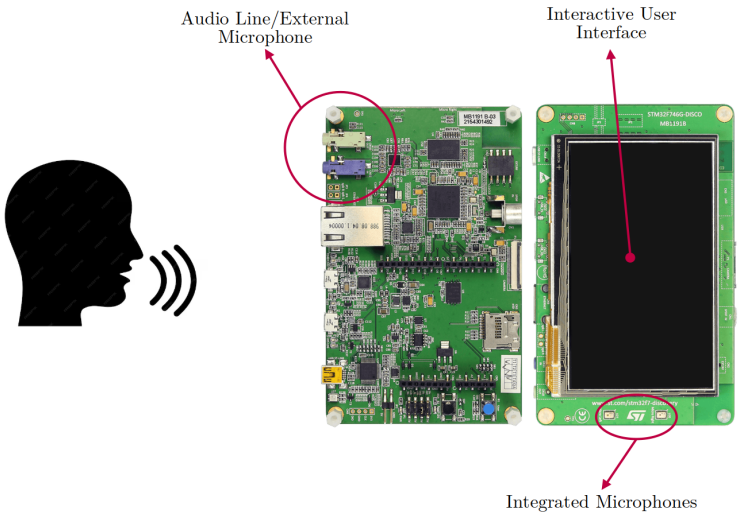

Figure 6: Voice-controlled interface running on the STM32F746G Discovery board.

Implementation Details

- Integration of the Zephyr RTOS on the STM32F746G Discovery board to manage audio processing and user interface tasks.
- Audio signal acquisition using the board's audio input line and Analog-to-Digital Converter (ADC) for signal capture.
- Voice command recognition using either:
  - Signal processing techniques (e.g., MFCCs) to match predefined voice commands.
  - Machine Learning via a pre-trained TensorFlow Lite model to identify voice commands.
- Design of a graphical user interface (GUI) on the onboard touchscreen display to provide visual feedback for recognized commands.
- Optimization for real-time voice command processing and resource-efficient operation on the STM32F746G board.
- Integration of touch input to allow seamless user interaction alongside voice command recognition.

3.7 UWB-Based Indoor Positioning Robot

The goal of this project is to design and implement an indoor localization system based on Ultra-Wideband (UWB) technology for a mobile robot platform. The system uses one mobile UWB tag mounted on the robot together with three to four fixed UWB anchors placed in the environment. By performing ranging measurements using Two-Way Ranging (TWR) or Time Difference of Arrival (TDoA), the robot's 2D/3D position can be estimated through trilateration. Real-time position data is then transmitted wirelessly to a PC for visualization and analysis.

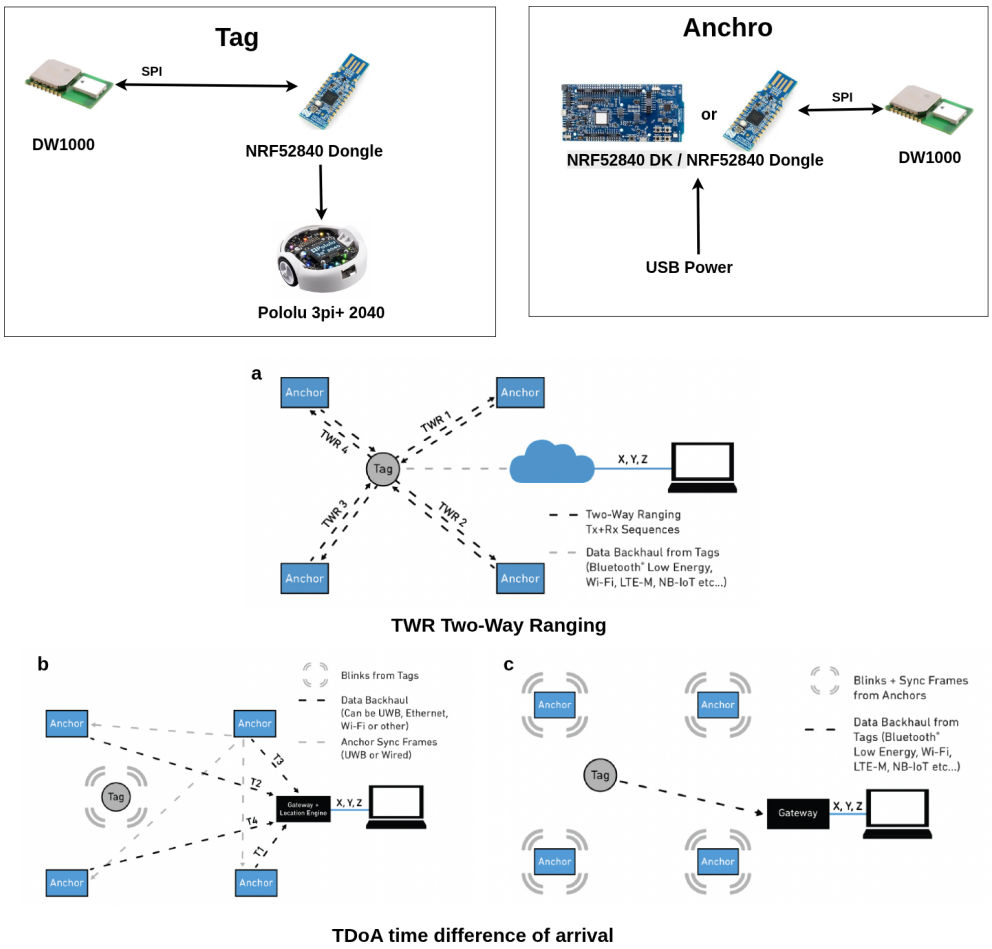

Figure 7: UWB-based indoor positioning architecture.

Implementation Details

- **Mobile Robot Platform:** The mobile robot is equipped with a DW1000 UWB module interfaced via SPI to an nRF52840 dongle, which handles wireless communication and data forwarding. The Pololu 3pi+ 2040 platform provides motor control through PWM signals and serves as the main processing unit.
- **Anchor Deployment:** Three to four fixed UWB anchors are deployed in the environment. Each anchor consists of a DW1000 transceiver connected to an nRF52840 DK or dongle and is powered via a stable USB supply to ensure continuous operation.
- **UWB Communication:** As the robot moves, the onboard UWB tag exchanges signals with the anchors to obtain ranging information. The collected measurement data are transmitted over IEEE 802.15.4 to a PC for monitoring.
- **UWB Ranging Methods:** Two ranging principles are supported:
  - *Two-Way Ranging (TWR):* Distance is estimated from the round-trip time of bidirectional message exchanges between the tag and individual anchors, without requiring anchor synchronization.
  - *Time Difference of Arrival (TDoA):* Positioning is based on timestamp differences of received blink messages, requiring synchronized anchors but significantly reducing communication load on the tag.
- **Position Estimation:** The robot's 2D or 3D position is computed using trilateration based on either absolute distance measurements from TWR or time-difference measurements from TDoA.

- **Visualization and Monitoring:** The estimated position is continuously transmitted to a PC, enabling real-time tracking, visualization, and debugging of the robot's motion.

3.8 BLE Mesh Machine Health Monitoring Dashboard

This project implements a BLE Mesh–based machine health monitoring system using multiple nRF52840DK sensor nodes and a dedicated nRF52840 Mesh Gateway, with the M5Stack Core2 Console serving as the user dashboard. Distributed sensor nodes collect vibration and temperature features and publish machine-health telemetry over the BLE Mesh network. The Mesh Gateway manages provisioning and network configuration and operates as a proxy to forward telemetry to the Core2 Console, which provides real-time visualization, device control, and data logging for analysis.

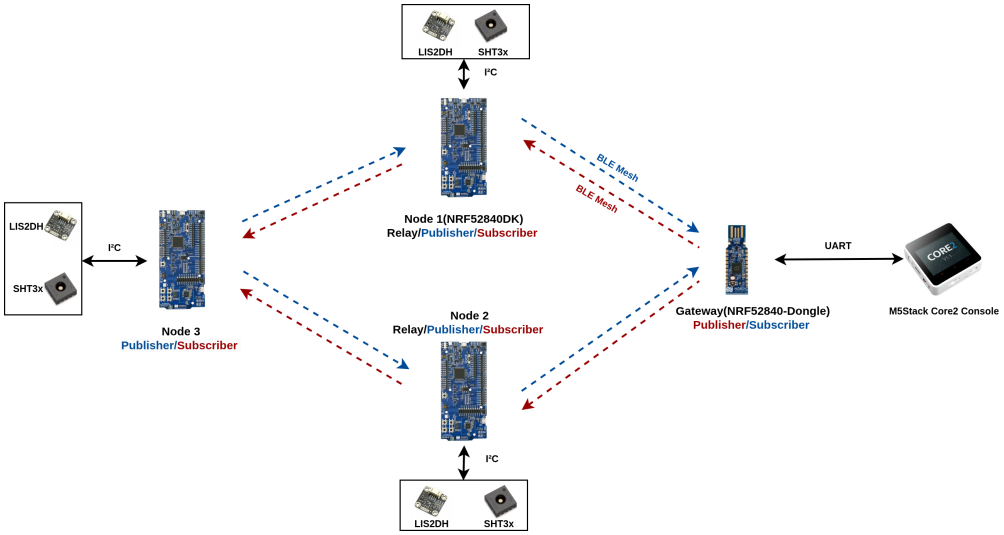

Figure 8: Machine-health dashboard driven by a BLE Mesh network.

Implementation Details

- **Sensor Node Deployment:** Each sensing node is built on the nRF52840DK platform and integrates vibration and temperature sensors. Local feature extraction is performed on-node, including vibration RMS, peak values, and temperature variation (ΔT).
- **Mesh Network Provisioning:** A nRF52840-Dongle Mesh Gateway manages the entire BLE Mesh network. It performs node provisioning, assigns unicast addresses, distributes network and application keys, and prepares each sensor node to participate in the mesh.
- **Telemetry Publication Configuration:** After provisioning, the gateway binds the appropriate AppKey to the Vendor Model and configures telemetry publication parameters, including the group publish address (e.g., `0xC001`) and the reporting interval (e.g., 2 s, 5 s, or 10 s).
- (Option)**Group Communication and Control:** Sensor nodes are configured to subscribe to a common group address, enabling group-based control commands. This mechanism supports unified configuration updates, such as threshold adjustment and sampling period control, across multiple nodes.
- **Mesh Gateway and Proxy Functionality:** The Mesh Gateway encapsulates all BLE Mesh networking logic and operates as a Mesh Proxy. It translates BLE Mesh messages into a structured UART-based application protocol, thereby isolating the BLE Mesh stack from higher-level components.
- **M5Stack Core2 Integration:** The M5Stack Core2 Console communicates with the Mesh Gateway via UART. It functions as a user dashboard by receiving telemetry data and alerts, visualizing system-wide machine health (Option:issuing configuration commands back to the sensor nodes).