Design and Implementation of a Smart Floor with Topology Reconstruction

Technical Documentation

Blaž Jerman

Mentor: doc. dr. Aleksandar Tošić

Contents

1 Problem Description

Smart floors are modular floor systems that detect force, movement, and the position of people or objects on the surface. A smart floor is composed of several floor modules, where each module contains multiple force sensors. The collected sensor data can be used to determine whether a person or object is standing on the floor, where the force is applied, and how movement changes over time.

Such systems can be used in many practical applications. Examples include fall detection for elderly people in smart homes or care facilities, gait analysis and health monitoring, sports training, interactive gyms, smart environments, and interactive games.

The existing implementation of the smart floor system was already functional, but it had several important limitations. The main problem was that data collection was too slow and unreliable. Since the previous implementation used Wi-Fi communication, data from different nodes did not always arrive at the same time. A larger number of wireless connections could also cause communication collisions and increase delay. Another limitation was that the positions of the nodes had to be entered manually, which made the setup process slower and more prone to errors. Power distribution also became more difficult when the number of modules increased.

The goal of this project was to design and implement a new smart floor architecture that improves the speed, reliability, and usability of the system. The new system replaces Wi-Fi communication with wired communication and introduces automatic topology reconstruction. This allows the system to determine the physical arrangement of the floor modules without requiring the user to manually enter their positions.

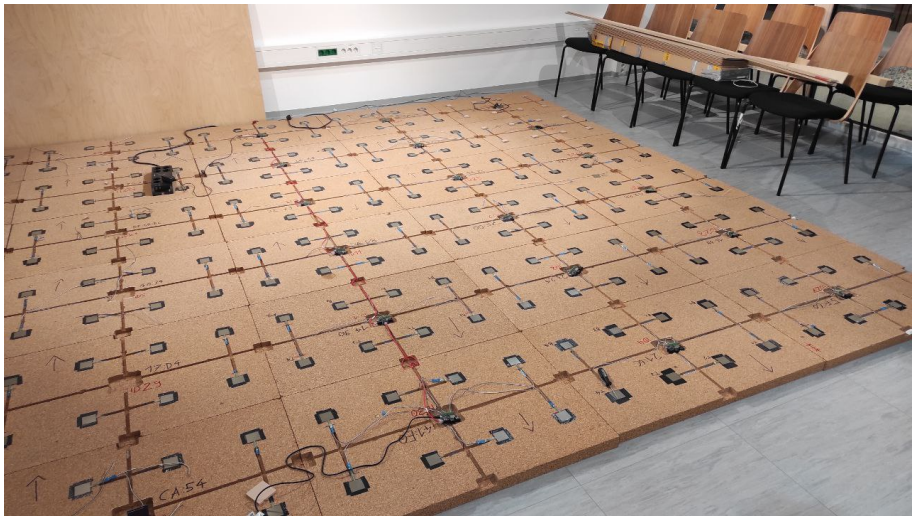

Figure 1: Example of the existing smart floor system.

2 System Analysis and Design

The system is based on a modular architecture. Each floor module is called a node. A node represents one physical part of the floor and contains force sensors, an ESP32 microcontroller, communication connectors, and power regulation components. In the existing system, one node has a size of 65 cm $\times$ 95 cm and contains eight force sensors.

The original system used Wi-Fi communication to transfer sensor data from the nodes to a central computer. Although this approach allowed wireless communication, it introduced several problems when the number of nodes increased. Wireless communication can suffer from collisions, unstable latency, and synchronization problems between nodes. For this reason, the new design uses wired communication between nodes.

The proposed architecture keeps the modular structure of the previous smart floor but changes the way communication and setup are performed. Nodes are connected with standard cables, which makes communication more reliable and simplifies power distribution. The system is designed so that additional nodes can be connected to expand the floor.

A central feature of the new system is topology reconstruction. Topology reconstruction means that the system can automatically detect how the nodes are connected and reconstruct the layout of the floor. This removes the need for manual position entry. As a result, the floor can be rearranged or expanded more easily, and the graphical interface can reconstruct the floor layout based on the detected topology.

The system consists of the following main components:

- smart floor nodes with force sensors,
- ESP32 microcontrollers,
- wired communication between nodes,
- a central computer for receiving and displaying data,
- a graphical interface for testing and visualization,
- a topology reconstruction mechanism for determining the node layout.

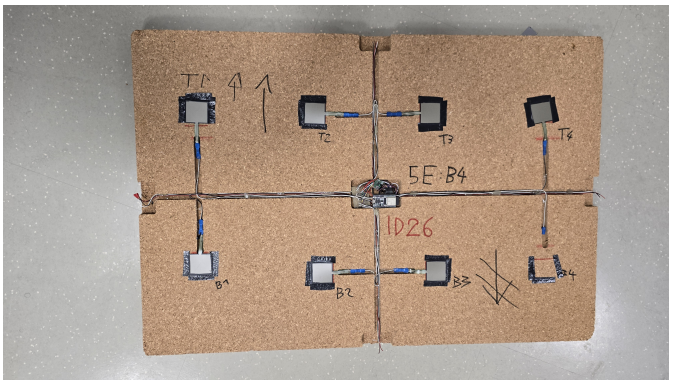

Figure 2: Smart floor node with force sensors.

3 Requirements and Functional Specification

The main objective of the project was to create a new smart floor system that improves the limitations of the previous implementation. Based on this objective, the following requirements were defined:

- The floor must be built from multiple modular nodes.
- Each node must be able to read values from multiple force sensors.
- Communication between nodes must be implemented using wired communication instead of Wi-Fi.
- Sensor data transfer must be faster and more reliable than in the previous Wi-Fi-based system.
- The system must support automatic topology reconstruction.
- The user should not need to manually enter the physical position of each node.
- A graphical interface must be available for testing, debugging, and visualization.
- The system should support future expansion with additional nodes.
- The hardware must provide stable power distribution for the connected components.

3.1 System Model

The system can be modeled as a tree of connected smart floor nodes, as shown in Figure 3. The main node is connected to the data logger, while the remaining nodes are connected as child nodes.

Communication between nodes uses two directions. The trigger signal is sent from the main node toward the child nodes, while sensor data is transmitted back from the child

nodes toward the main node. This allows the system to collect data from all connected nodes and reconstruct the topology based on the node connections.

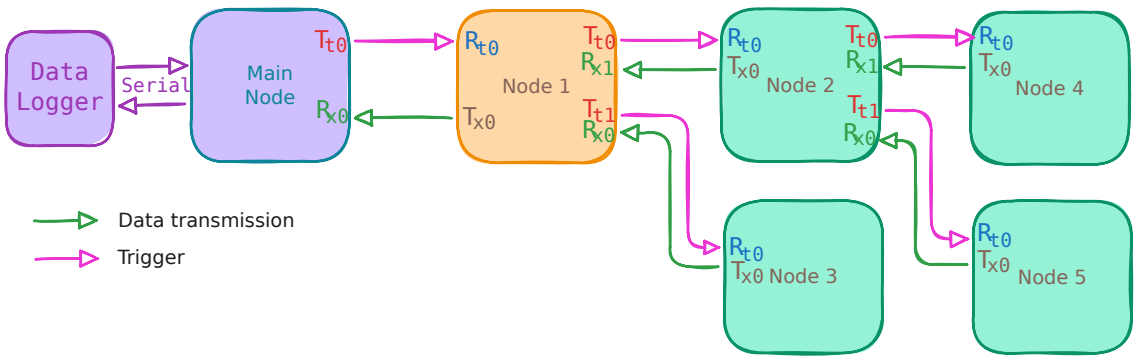

Figure 3: System model with trigger and data transmission between connected nodes.

4 Technologies

4.1 ESP32

The ESP32 microcontroller was used as the main controller for each smart floor node. It was selected because it provides enough performance for sensor reading, communication, and control tasks. It also supports many communication interfaces, which makes it suitable for developing a custom wired communication protocol.

In this project, the ESP32 is responsible for reading force sensor values, communicating with other nodes, and participating in the topology reconstruction process.

4.2 ESP-IDF and C

The firmware for the ESP32 was implemented using ESP-IDF. ESP-IDF is the official development framework for ESP32 microcontrollers and provides access to low-level hardware functionality, communication interfaces, timing, and debugging tools.

The communication protocol was implemented in the C programming language. This allowed direct control over the behavior of the microcontroller and made it possible to implement the protocol close to the hardware level.

4.3 Force Sensors

Each smart floor node contains multiple force sensors. The sensors are used to detect pressure on different points of the module. By reading the values from these sensors, the system can determine whether force is applied to the floor and approximately where the force is located.

4.4 Java and Swing

A graphical user interface was implemented in Java using the Swing library. The interface is used to display sensor values, test communication, debug the system, and visualize the floor layout. During development, the graphical interface was important because it made it easier to observe the behavior of the nodes and verify whether data was being received correctly.

4.5 KiCad

KiCad was used for designing the custom printed circuit board. After the first prototype was tested, a custom PCB was designed and manufactured. The final PCB includes the ESP32 microcontroller, connectors for force sensors, connectors for wired communication, power regulation components, a USB-to-UART converter, and a header for an LED strip planned for future work.

4.6 GitHub Projects

GitHub Projects was used for planning and managing the development process. The work was divided into smaller tasks, such as literature review, ESP-IDF environment setup, prototype development, protocol implementation, graphical interface development, circuit design, production, testing, and final report preparation.

5 Implementation

5.1 Prototype

The first part of the implementation was the development of a prototype. The prototype was built using an ESP32 development board, force sensors connected to the node, and a basic wired communication setup. The purpose of the prototype was to test the main concept before designing the final hardware.

The prototype allowed early testing of sensor readings, communication behavior, and protocol logic. It was also useful for debugging problems before producing the custom PCB. This step reduced the risk of errors in the final hardware design.

Figure 4: First prototype with ESP32 and connected components.

5.2 Communication Protocol

The communication protocol was developed earlier as part of the broader smart floor system and was later also used in the related Research Seminar project, supervised by the same mentor, doc. dr. Aleksandar Tošić. Since this project continues work on the same system, the existing protocol was reused and adapted where necessary.

The protocol was implemented on the ESP32 using ESP-IDF and the C programming language. Its main purpose is to transfer sensor data between nodes and to provide the communication basis for topology reconstruction. The protocol was designed for wired communication, which makes it more reliable than the previous Wi-Fi-based approach.

In the previous system, multiple nodes communicated wirelessly, which could cause collisions and inconsistent data arrival times. With wired communication, the system can achieve more stable data transfer and better control over the communication process. This is especially important when the number of nodes increases, because the system must collect data from many sensors while maintaining reliable timing.

The communication protocol also prepares the system for topology reconstruction. By exchanging information between neighboring nodes, the system can determine which nodes are connected and use this information to reconstruct the floor layout automatically.

5.3 Topology Reconstruction

Topology reconstruction is one of the most important features of the new smart floor system. It allows the system to automatically determine how the nodes are connected. Instead of requiring the user to manually enter the position of each module, the system detects neighboring nodes and reconstructs the floor layout based on the physical connections.

This feature makes the system easier to use and reduces the chance of configuration errors. It also makes the smart floor more flexible, because modules can be rearranged or expanded without manually updating the layout every time.

5.4 Graphical Interface

The graphical interface was implemented in Java Swing. It displays sensor values and helps with debugging the system. It is also used for floor layout visualization, which is important for testing topology reconstruction.

During development, the interface was used to check whether the data from the nodes was received correctly and whether the reconstructed layout matched the physical arrangement of the floor. The graphical interface therefore served both as a testing tool and as a visualization tool.

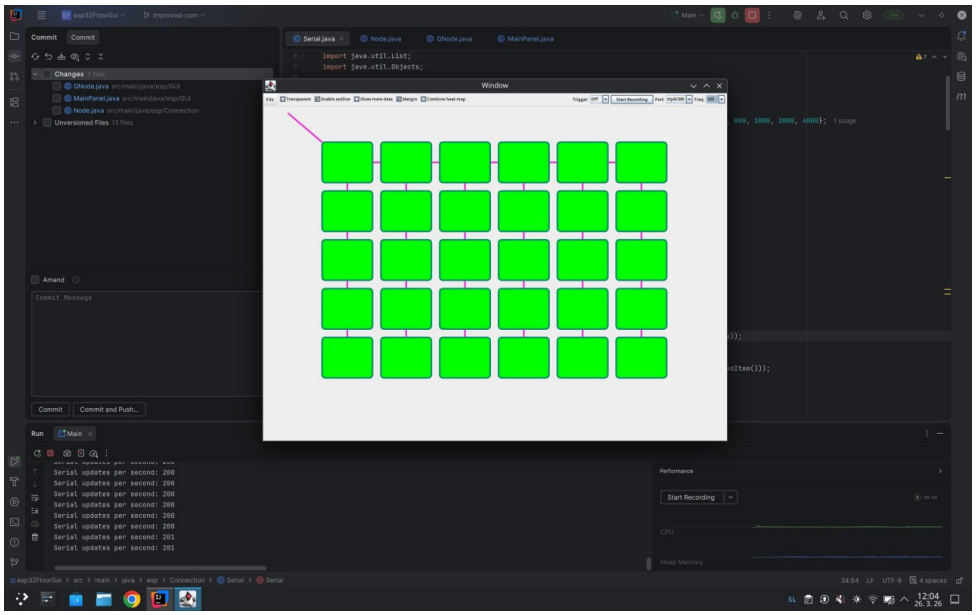

Figure 5: Graphical interface for testing and visualization.

5.5 Final Hardware

After the prototype was tested, a custom circuit was designed in KiCad and manufactured. The final hardware includes all required components for a smart floor node in a cleaner and more reliable form than the first prototype.

The final PCB includes:

- ESP32 microcontroller,
- connectors for force sensors,
- connectors for wired communication,
- power regulation components,
- USB-to-UART converter,
- a header for an LED strip, which is planned for future work.

The final PCB makes the system easier to install and test. It also provides a better foundation for future extensions, because the hardware is more organized and better suited for repeated use than the prototype.

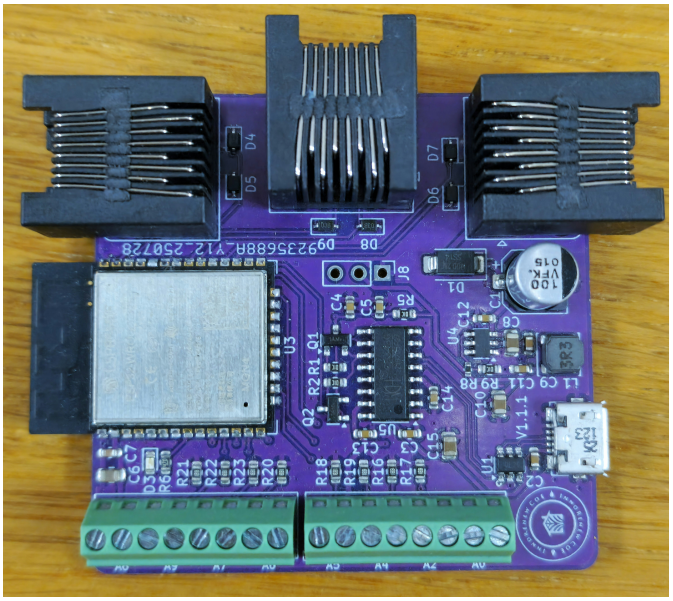

Figure 6: Final PCB designed for the smart floor node.

5.6 Testing

Testing was performed on both the prototype and the final hardware. The main testing areas were sensor readings, communication reliability, protocol behavior, power stability, and the comparison between the prototype and the final circuit.

The graphical interface was used during testing to display sensor data and visualize the floor layout. The final system was tested as a complete product, including the custom PCB, sensors, communication protocol, and graphical interface.

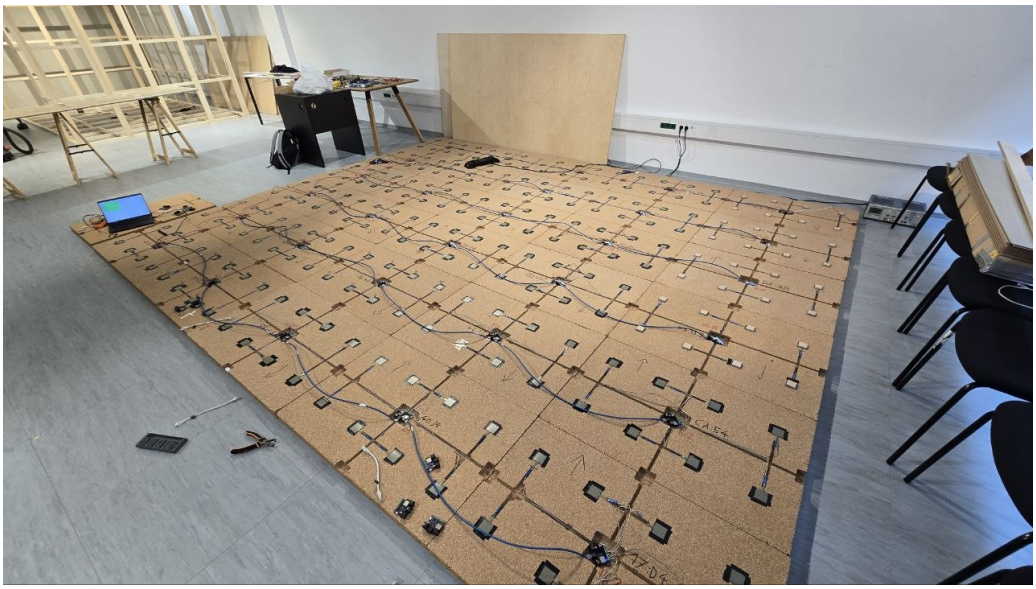

Figure 7: Final smart floor system.

6 Improvements and Future Work

The system works as a prototype and was also tested with the final PCB. The main parts of the project were completed: ESP32 firmware, wired communication, graphical interface, custom hardware, and final testing.

Future work should focus on two main improvements. The first is the integration of LED strips, since the final PCB already includes a header for them. LEDs could be used for visual feedback, debugging, or interactive use of the smart floor.

The second improvement is the use of differential pairs for communication between nodes. This could improve signal reliability, especially when using longer cables or larger floor configurations.