



BRNO UNIVERSITY OF TECHNOLOGY

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

FACULTY OF INFORMATION TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ

DEPARTMENT OF COMPUTER SYSTEMS

ÚSTAV POČÍTAČOVÝCH SYSTÉMŮ

**EMBEDDED SYSTEM FOR MONITORING SPORTS
PERFORMANCES IN BOATING SPORTS WITH AIM
OF IMPROVING THEM**

VESTAVĚNÝ SYSTÉM PRO SLEDOVÁNÍ SPORTOVNÍCH VÝKONŮ V LODNÍM SPORTU S CÍLEM
JEJICH ZDOKONALOVÁNÍ

BACHELOR'S THESIS

BAKALÁŘSKÁ PRÁCE

AUTHOR

AUTOR PRÁCE

LUKÁŠ NEUPAUER

SUPERVISOR

VEDOUCÍ PRÁCE

Ing. JOSEF STRNADEL, Ph.D.

BRNO 2025

Bachelor's Thesis Assignment



161644

Institut: Department of Computer Systems (DCSY)
Student: **Neupauer Lukáš**
Programme: Information Technology
Title: **Embedded System for Tracking Sports Performances in Boating Sport with Aim of Improving Them**
Category: Embedded Systems
Academic year: 2024/25

Assignment:

1. Familiarize yourself with the area of boat sports; focus on sports based on moving a boat using a paddle/oar. Choose a class of boat sports you will characterize closely.
2. Conduct a survey on measurement and embedded systems.
3. Summarize requirements placed on an embedded system able to monitor key quantities related to the selected class of boat sports and to store and use the monitored data for improving sports performance and summarize use cases of such a system. Based on a research/analysis, choose elements, propose an architecture and mechanism of operation of such a system, and propose a basic mobile application to communicate with a user of the system.
4. Implement the proposed system taking into account the ease of its intended purpose and its use cases. Implement the proposed mobile application.
5. Check your solution (i.e., the created system and mobile application) in real conditions, interpret and evaluate the facts found - especially with regard to the fulfillment of the requirements in the use cases from the item 3.
6. Identify strengths and weaknesses of your solution, suggest possible modifications/enhancements as well as possible directions of continuation in the solution.

Literature:

- According to the supervisor's recommendation.

Requirements for the semestral defence:

- Completion of the items 1 to 3 of the assignment.
- Presentation of a functional prototype capable of measuring and storing at least one of key quantities.
- Presentation and interpretation of initial measurement results.
- Presentation of the mobile application design.

Detailed formal requirements can be found at <https://www.fit.vut.cz/study/theses/>

Supervisor: **Strnadel Josef, Ing., Ph.D.**
Head of Department: Sekanina Lukáš, prof. Ing., Ph.D.
Beginning of work: 1.11.2024
Submission deadline: 14.5.2025
Approval date: 31.10.2024

Abstract

This thesis describes the development process of an embedded device and an Android application for performance analysis of the C1 canoe discipline. The work summarizes component selection for embedded device and data processing during the analysis. The final system consists of the embedded device inside the paddle wirelessly communicating with the Android application, which provides a detailed analysis of the paddler's performance. This analysis consists of boat speed, cadence, stroke phase durations, and paddle tilt during a stroke. The purpose of this work is to explore new ways of performance analysis in water sports.

Abstrakt

Táto práca sa zaoberá vývojom vstavaného systému a Androidovej aplikácie, ktorá analyzuje výkon vo vodnom športe kategórie C1. V práci je zhrnutý proces výberu komponentov pre vstavaný systém a spracovanie dát počas analýzy. Výsledný systém pozostáva zo vstavaného systému vo vnútri pádla, ktorý bezdrôtovo komunikuje s androidovou aplikáciou, ktorá analyzuje výkon pádlera. Táto analýza pozostáva z rýchlosti lode, kadencii, trvania fáz záberu a náklonu pádla počas záberu. Zmyslom tejto práce je preskúmať nové možnosti analýzy výkonu vo vodných športoch.

Keywords

water sport, paddle, analysis, performance, Arduino, Android, Bluetooth, accelerometer, gyroscope, embedded system

Klíčová slova

vodný šport, pádlo, analýza, výkon, Arduino, Android, Bluetooth, akcelerometer, gyroskop, vstavaný systém

Reference

NEUPAUER, Lukáš. *Embedded System for Monitoring Sports Performances in Boating Sports with Aim of Improving Them*. Brno, 2025. Bachelor's thesis. Brno University of Technology, Faculty of Information Technology. Supervisor Ing. Josef Strnadel, Ph.D.

Embedded System for Monitoring Sports Performances in Boating Sports with Aim of Improving Them

Declaration

I hereby declare that this Bachelor's thesis was prepared as an original work by the author under the supervision of Ing. Josef Strnadel Ph.D. I have listed all the literary sources, publications and other sources, which were used during the preparation of this thesis.

.....
Lukáš Neupauer
May 14, 2025

Acknowledgements

I would like to express my gratitude for the guidance to Ing. Josef Strnadel Ph.D. I would also like to thank Mgr. Svatava Nováková Ph.D for her professional insights into canoeing and Mr. Tomáš Řezníček for providing an opportunity for on-water testing.

Contents

| | | |
|----------|--|-----------|
| 1 | Introduction | 3 |
| 2 | Boating sports | 4 |
| 2.1 | Paddling sports | 4 |
| 2.2 | Paddle | 5 |
| 2.3 | Performance analysis in paddle sports | 6 |
| 3 | Embedded systems | 7 |
| 3.1 | Microcontroller unit | 7 |
| 3.2 | Peripheral Interfaces | 7 |
| 3.3 | Communication protocols | 8 |
| 4 | Problem Analysis and Choice of Implementation Means | 9 |
| 4.1 | System requirements | 9 |
| 4.2 | Hardware selection | 10 |
| 4.3 | Prototype | 16 |
| 4.4 | Firmware | 17 |
| 5 | Android application | 24 |
| 5.1 | Captured data | 24 |
| 5.2 | Database | 25 |
| 5.3 | User interface | 26 |
| 5.4 | Activity analysis | 28 |
| 5.5 | Improving performance | 31 |
| 6 | Testing | 32 |
| 7 | Conclusion | 33 |
| | Bibliography | 34 |

List of Figures

| | | |
|-----|--|----|
| 2.1 | Double-blade paddle | 5 |
| 2.2 | Single-blade paddle | 5 |
| 2.3 | Path of paddle in C1 category | 6 |
| 4.1 | Efficiency of MPM3610 step-down converter[12] | 13 |
| 4.2 | Discharge voltage of Li-ion batteries [7] | 14 |
| 4.3 | Wheatstone diagram circuit | 15 |
| 4.4 | Prototype wiring diagram | 16 |
| 4.5 | Time synchronization of Arduino | 21 |
| 4.6 | Flow of activity packets | 22 |
| 4.7 | Completed prototype | 23 |
| 5.1 | Database diagram | 26 |
| 5.2 | Interface design of main and Bluetooth screen | 27 |
| 5.3 | Interface design of stored activities and detailed activity screen | 28 |
| 5.4 | Arduino IMU axis | 29 |
| 5.5 | Stroke phases based on Z accelerometer axis | 30 |

Chapter 1

Introduction

Boating sports incorporate many variants of activities. It ranges from the art of sailing to the adrenaline of powerboat racing. People have been improving and building new types of vessels for centuries. In the past decades, we have mastered the techniques and equipment to control the ship and use its full potential. Nowadays, boating has become widely popular as a recreational activity as well as a competitive sport.

This thesis focuses on analyzing the paddling technique of athletes in the flatwater C1 (canoe single) discipline. Understanding and quantifying the performance of different paddling techniques can provide valuable feedback to athletes and coaches, which aims to improve their performance.

This work approaches this by embedding sensors and microcontrollers directly into the paddle shaft. This system measures data from an accelerometer and gyroscope and wirelessly transmits the collected data to a mobile device for further processing and visualization. The mobile application contributes to this system through its sensors and combines them into the analysis.

The main goal of this work is to develop a functional prototype of the paddle-embedded system and a custom application that processes, stores, and visualizes the performance. In the research for this work, no similar system featuring an inbuilt device in the paddle with wireless communication with a mobile application has been found. This makes the proposed solution unique in performance diagnostics in paddling sports.

Chapter 2

Boating sports

Boating sports represent a wide range of recreation and competitive activities with many disciplines that captivate enthusiasts all around the world. From the serene leisure of sailing through the harsh whitewater riding to the adrenaline-fuelled intensity of powerboat racing. Each discipline requires a different set of skills and techniques that are mastered by athletes who compete professionally. These techniques and skills are fundamental in paddling sports.

2.1 Paddling sports

Paddling sports combine human strength and precision, which presents a unique challenge in the paddling experience. Paddling was developed in ancient times as a simple way to transport goods and to make traveling faster. Through time, it developed into modern competitive sports and experiential activities. This evolution from practical transportation to a refined sport and recreational pursuit made it more appealing to many people who seek these sports as a free time activity. Nowadays the most popular paddling sports are rafting, canoeing, kayaking, dragon boating, and paddleboarding.

Rafting

Rafting is a team sport that involves a crew that typically consists of between 5 and 8 members. One member who sits on the back steers the boat, and all the others paddle synchronously to propel the boat. The crew sails on inflatable rafts using single-blade paddles. Rafts are used mainly in whitewater rivers because of their size, it is hard to flip them, and it is not complicated for the crew to leave the raft in case of emergency.

Canoeing

Canoeing involves two crew members who can either sit or knee in a canoe. The canoe has an open top and for propelling a single-blade paddle is used. Steering is done by the crew member in the back of the canoe. The canoe can be used on both whitewater and flatwater.

Kayaking

Kayaking, like canoeing, is done in whitewater and flatwater. The difference is that in kayaking, we use double-blade paddles and the athlete sits in a kayak. It is usually an individual sport except for double-seated kayaks.

Olympic categories

In the Olympics, two boating sports use the paddle, canoe flatwater, and canoe slalom. In this sport, there are two types of discipline. Canoes, where the athlete is strapped into the boat in a kneeling position and uses a single-blade paddle (C), and kayaks, with the athlete seated and propelling the boat with a double-bladed paddle (K). These disciplines have their sub-disciplines, individuals (C1, K1), pairs (C2, K2) and groups of four (C4, K4). Canoe slalom takes place in whitewater where athletes have to pass through upstream and downstream gates as quickly as possible. On the other hand, canoe flatwater involves sprinting along a straight path. [2]

2.2 Paddle

In paddling sports, we have two different types of paddles. Double-blade paddles are used in kayaking, where the paddle is held by a shaft with both hands. Blades can be offset up to 90 degrees, and the kayaker is paddling on both sides of the kayak. That means the paddle is rotated by the kayaker during the propelling phase so the blade can face the water with the flat side. The rotation of the paddle depends on the twist of the blades.[8]

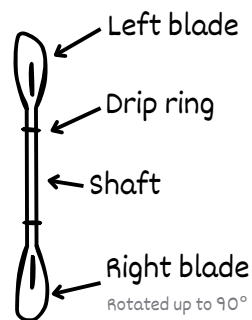


Figure 2.1: Double-blade paddle

Single-blade paddles are used only on one side of the boat and are held with one hand on the shaft and the other on the T-grip. When there is only one person on the boat, the straight path on the boat is accomplished by rotating the paddle during the stroke to balance the turning forces.

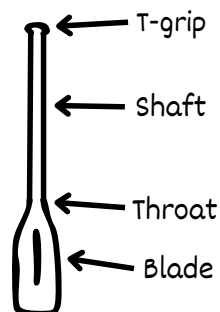


Figure 2.2: Single-blade paddle

2.3 Performance analysis in paddle sports

Performance analysis in paddle sports has evolved significantly in recent years, primarily due to the integration and accessibility of modern sensor technologies. Traditional coaching methods, which heavily rely on visual assessment and subjective judgment, are now being accompanied by modern technologies. This shift allows for a more detailed understanding of an athlete's technique performance.

In paddle sports, performance can be analyzed from multiple perspectives, the most relevant are kinematics and dynamics.

- Kinematics focuses on describing the motion of the paddle and the boat without considering the force involved. This involves metrics such as stroke cadence, durations, paddle angle, and rotation.
- Dynamics focuses on the forces and accelerations causing the motion. These metrics include boat acceleration, speed gain, and energy transformation into motion.

Thanks to the integration of the accelerometers and gyroscopes in the paddle and the boat, it is possible to analyze various kinematic and dynamic aspects of the athlete's performance on the water. From a kinematic perspective, it is possible to analyze the rotation of the paddle and its path during stroke, stroke cadence, or paddle blade angle. From a dynamic point of view, the acceleration of the boat, speed, or force on the paddle can be analyzed.[\[11\]](#)

This thesis focuses on the C1 Olympic category, where only one paddler is present in the boat, which uses a single-blade paddle. The key aspects for analyzing the performance are the angle of the blade relative to the ground, stroke phase duration, cadence over the course of the activity, and the speed of the boat have been chosen. The visualization of the paddle path in the C1 category is visualized in image 2.3. The paddle during the propelling phase also rotates for a straight trajectory of the boat.

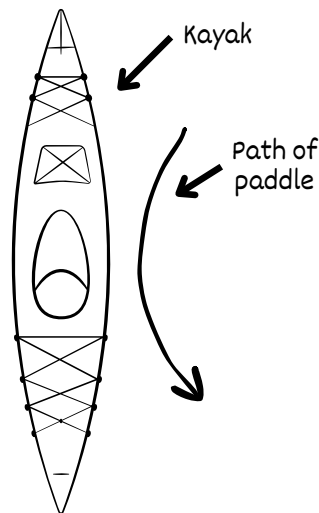


Figure 2.3: Path of paddle in C1 category

Chapter 3

Embedded systems

Embedded systems are dedicated devices designed to perform one of a few specific functions. Unlike general computers, embedded systems are typically optimized for efficiency and reliability. They are often constrained in terms of processing power, memory, and energy consumption, just sufficient enough for their dedicated purpose to minimize hardware cost and maximize performance for the intended application. Modern embedded systems commonly consist of microcontroller units and various input/output peripherals.

3.1 Microcontroller unit

The microcontroller unit is a compact integrated circuit containing a processor, memory, and input/output interfaces. They typically include features such as timers, analog-digital converters, digital input/output pins, and communication modules. Timers are usually based on an internal oscillator, which tends to drift, and is not ideal for precise timing, or an external crystal oscillator, which provides higher accuracy for time-sensitive tasks.

3.2 Peripheral Interfaces

The analog-digital converter is a component that converts an analog signal, such as voltage from the sensor, into discrete digital values that can be processed by the microcontroller. The analog signal is a continuous voltage that must be in the range from zero up to the reference voltage of the microcontroller. If the signal exceeds the reference voltage, it may cause damage to the microcontroller. The signal, when read, is mapped to the nearest digital value based on the resolution. For example, the input voltage is 3 V on the microcontroller with a reference voltage of 5 V with 10-bit resolution would result in the digital value 614.

Digital pins are used to read or write digital signals. A digital signal has only two states: HIGH (reference voltage) or LOW (0 V). When configured as input, digital pins often require a pull-up or pull-down resistor to ensure a stable and defined logic level. The pull-up resistor connects the pin to a high voltage, defaulting the input to the HIGH state. This prevents the pin from picking up electrical noise that can cause random values to be read. When an external component connects the pin to the ground, it overrides the HIGH state and brings the input to LOW. In contrast, a pull-down resistor connects the pin to the ground, keeping the input at a default LOW state until an external signal drives it HIGH.

When configured as output, digital pins can set their signal to either HIGH or LOW, allowing them to control external devices. Some digital pins also support Pulse-Width

Modulation (PWM). This technique rapidly switches the pin between HIGH and LOW states at a fixed frequency. By adjusting the ratio of HIGH and LOW states, the PWM can simulate analog output levels.

3.3 Communication protocols

Embedded systems often need to exchange data with other peripheral devices. To achieve this, several communication protocols are commonly used. Among the most frequently used protocols are UART, I²C, and SPI.

- UART - Universal Asynchronous Receiver/Transmitter is a communication protocol that uses two lines, TX (transmit) and RX (receive). Data are transmitted serially bit by bit without the need for a separate clock line. Both devices must agree on the transmitting baud rate (bits per second) to ensure synchronized data transfer.
- I²C - Inter-Integrated Circuit is a synchronous communication protocol based on the master-slave principle using two lines SDA (data) and SCL (clock). It allows multiple slave devices to share the same network, where each device is identified with a unique address. The master device initiates the data transfer and generates the clock signal.
- SPI - Serial Peripheral Interface is a synchronous communication protocol also featuring the master-slave. This protocol uses four lines, namely MOSI (Master Out Slave In), MISO (Master In Slave Out), SCK (Serial Clock), and CS (Chip Select). This allows the data transfer both ways simultaneously between the master and one slave. The CS line is used for targeting the slave device, making it possible to communicate with multiple peripherals. This protocol provides the highest speed among the three mentioned.

Chapter 4

Problem Analysis and Choice of Implementation Means

This system consists of two different devices. The embedded device is inside the paddle, and the mobile phone is placed in the boat. These devices collect data during the performance and provide an analysis of the finished activity.

4.1 System requirements

These devices must have a wireless communication method. The paddle-side device must be capable of capturing and storing paddle movements and transmitting the measured data to the mobile device for further analysis. It must operate independently without any external wires connected and be compact enough to fit entirely within the paddle structure. The user's mobile phone must be capable of receiving and storing measured data from the paddle as well as capturing the speed of the vessel.

Paddle embedded system

A major limitation of the paddle-embedded system is its internal structure, which provides very limited space for device placement. The paddle used for this thesis has an internal diameter of its shaft 26 millimeters. This constraint significantly limits the range of viable component choices. As a result, all components must fit within this small cross-sectional area.

For capturing the paddle movement, the device must include an accelerometer and gyroscope with sufficient accuracy and sampling frequency to reliably detect and record rapid changes in paddle orientation and motion during paddle strokes. Based on the study comparing beginner and intermediate paddlers, the upper limit of stroke cadence for an athlete is estimated to be around 65 strokes per minute. To accurately estimate the paddle motion during a stroke, we need around 50 samples per stroke. Given the stroke duration of approximately 0.92 seconds at a cadence of 65 strokes per minute, the minimum sampling rate for both the gyroscope and accelerometer must be at least 54 Hz to ensure sufficient data resolution for detecting paddle motion.[16]

To ensure sufficient memory for storing the measured data from the paddle minimal storage capacity is required. Based on the estimate that no activity will last longer than one day and assuming one measurement occupies 4 bytes of memory, the total storage requirement can be calculated accordingly. At the given sampling frequency of 54 Hz, one

measurement occupies 4 bytes of memory, one sample consists of 6 measurements (three-axis accelerometer and three-axis gyroscope) plus a timestamp stored on 4 bytes to properly represent values exceeding 86,400,000 milliseconds in a day. Therefore, the final minimum memory required is calculated as:

$$7 \times 4 \times 54 \times 86,400 = 130,636,800 \text{ bytes} \approx 125 \text{ MB} \quad (4.1)$$

The device must maintain its internal clock with sufficient accuracy to ensure correct timestamping of measured data and to preserve data integrity over the full course of activity. The time drift must be minimized for the purpose of the time-dependent analysis.

The battery supply must provide sufficient capacity to ensure at least 24 hours of continuous operation, including the time required to transfer all the measured data, depending on the device’s expected power consumption. It must be rechargeable and compatible with common chargers that are widely available for home use.

Lastly, the device must support bidirectional wireless communication with the mobile phone to enable both data transmission from the paddle and control commands from the mobile application.

Mobile phone

Most modern smartphones already meet all the necessary criteria for this use case. For this thesis, the selected platform for the mobile application is Android, due to its more user-friendly development process and wider popularity among the general population.

4.2 Hardware selection

The initial step in hardware selection involves selecting a viable microcontroller that complies with specified limitations and requirements.

Microcontroller

Based on the system requirements, multiple viable options for a paddle-embedded system were selected. Summarization of this option is presented in Table 4.1.

| Board | Processor | Wireless | Flash/RAM (KB) | Acc. & Gyro |
|-----------------------------|------------------------------|------------|----------------|----------------|
| Arduino Nano 33 BLE | nRF52840 @64MHz | BLE | 1,024 / 256 | Yes (LSM9DS1) |
| Arduino Nano RP2040 Connect | RP2040 dual-core @133MHz | WiFi + BLE | 16,384 / 264 | Yes (LSM6DSOX) |
| Seeed Studio XIAO ESP32C3 | ESP32-C3 @160MHz | WiFi + BLE | 4,096 / 400 | No |
| Adafruit QT Py RP2040 | RP2040 dual-core @133MHz | None | 8,192 / 264 | No |
| Teensy 4.0 | ARM Cortex-M7 @600MHz | None | 1,984 / 1,024 | No |
| ESP32-WROOM-32 DevKitC | ESP32 @240MHz | WiFi + BLE | 4,480 / 520 | No |
| Waveshare ESP32-S3 Pico | ESP32-S3R2 dual-core @240MHz | WiFi + BLE | 16,384 / 2,560 | No |

Table 4.1: Comparison of viable microcontroller boards

For the purposes of this thesis, the boards featuring a wireless communication module along with an accelerometer and a gyroscope were selected since none of the viable boards met the criteria for the minimum storage capacity, and an external memory module is necessary.

Among the two remaining boards, 2 different communication methods with two different chips remain. Table 4.2 summarizes the power consumption of the modules used in these boards.

| Board | Wireless Module | Technology | Power Consumption |
|---------------------|------------------|------------|---------------------------------|
| Nano RP2040 Connect | u-blox NINA-W102 | WiFi + BLE | WiFi 95–190 mA BLE 95–130 mA |
| Nano 33 BLE | u-blox NINA-B306 | BLE only | BLE 1–20 mA |

Table 4.2: Power consumption of wireless modules used in the selected boards.[20][19]

Based on power consumption, considering the limited power supply, the Arduino Nano 33 BLE is the most suitable choice. Given its integrated Bluetooth Low Energy (BLE) module and significantly lower energy requirements, BLE will be used as the method of wireless communication between devices.

Arduino nano 33 BLE

The selected microcontroller board, Arduino nano 33 BLE, satisfies the requirements for measuring the paddle movement thanks to an integrated LSM9DS1 internal measurement unit (IMU) equipped with an accelerometer and gyroscope capable of operating at frequencies up to 952 Hz. According to the datasheet, the accelerometer can operate at 10, 50, 119, 238, 476, and 952 Hz, while the gyroscope supports 10, 59.5, 119, 238, 476 and 952 Hz. Among these options, the 119 Hz is the lowest frequency that simultaneously satisfies the requirements for both sensors. Although the gyroscope can operate at 59.5 Hz the corresponding option for the accelerometer in this range is only 50 Hz leading to frequency mismatch. Therefore, to ensure simple synchronization, both sensors will operate at 119 Hz. Higher frequencies are avoided, as they would introduce additional noise, increase storage occupancy, and prolong wireless transmitting duration. This sampling frequency updates the previous requirement for minimal storage capacity accordingly.[17]

$$7 \times 4 \times 119 \times 86,400 = 287,884,800 \text{ bytes} \approx 275 \text{ MB} \quad (4.2)$$

The board is equipped with a Serial Peripheral Interface (SPI) that supports external memory connection at 32 MHz. This frequency can theoretically export data to external memory at 4 MB/s and data captured by IMU at 119 Hz, plus timestamps per second, is only around 3 kB/s. Even with consideration of other tasks, the system needs to accomplish the transfer rate is more than sufficient for storing sample data. Arduino Nano 33 BLE also has an Inter-Integrated Circuit (I²C) capable of communicating with peripherals at speeds up to 400 kHz. This communication is already using the LSM9DS1 module, and although I²C is capable of communicating with multiple peripherals, the SPI is more common among large memory extensions, and is currently unoccupied by the device. Thus, the SPI is the more viable interface for the memory extension module and does not provide strain on I²C currently used by IMU.[6][15]

Memory extension

The SPI allows the connection of various types of external memory. Each type has different performance characteristics. Table 4.3 provides a summary of commonly used SPI-based memory technologies for embedded systems.

| Memory Type | Write Speed | Typical Capacity |
|-------------|------------------------------|------------------|
| SPI EEPROM | kilobytes/second | kilobytes |
| SPI Flash | hundreds of kilobytes/second | low megabytes |
| SPI FRAM | megabytes/second | kilobytes |
| SPI SD Card | low megabytes/second | gigabytes |

Table 4.3: Comparison of SPI-compatible memory types by write speed and capacity.

The memory requirements are fulfilled only by the SD card, which also provides sufficient write speed. Before selecting the SD card module, it is important to consider the microcontroller’s operating voltage. Arduino Nano 33 BLE operates only at 3.3 V and does not support 5 V logic. Therefore, the selected SD card module must be compatible with 3.3 V operating voltage.

For this purpose, the LaskaKit microSD Card module was chosen. It supports 3.3 V logic and also complies with dimension requirements with its width of 21 mm. This module is compatible with microSD card capacity up to 32 GB which is more than sufficient for the intended data stored. The specific card chosen for this module is Sandisk Ultra microSDHC 32 GB capable of data transfer at 98 MB/s. [10][21]

Power source

Since having a wire connected to the paddle would be a major inconvenience for the athlete, the device needs to be able to power itself and be easily rechargeable. The device must be equipped with its own source of power. According to the datasheet of the processor used in Arduino Nano 33 BLE, the current consumption during active CPU is 6.3 mA. For transmitting the data, the board uses 6.4 mA, and for receiving data 6.26 mA. Unfortunately, the producer of the SD card module and SD card itself does not provide current consumption values in their datasheets. Therefore, the power consumption for the current consumption for the SD card module is estimated at 5 mA, and for the SD card, based on data provided by different manufacturers, the estimation for current consumption is up to 200 mA. [18][15]

Based on this data, the maximum current consumption at 3.3 V can be calculated as:

$$I_{\max} = 6.3 \text{ mA} + 6.4 \text{ mA} + 6.26 \text{ mA} + 5 \text{ mA} + 200 \text{ mA} \approx 224 \text{ mA} \quad (4.3)$$

The Arduino Nano 33 BLE features a VIN pin for connecting an external power source, which according to the datasheet must be in the range from 4.5 V to 21 V. However, to maximize energy efficiency at current levels around 200 mA, the input voltage should aim towards the lower end of this range. This is because the integrated step-down converter MPM3610 becomes less efficient at higher input voltages, as shown in the efficiency graph. [12][6]

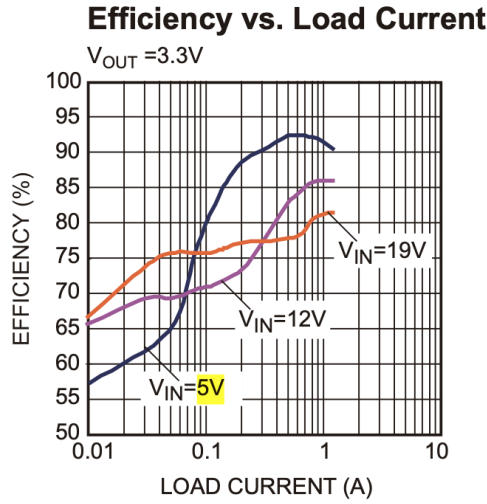


Figure 4.1: Efficiency of MPM3610 step-down converter[12]

Due to the tight internal space limitation of 26 mm diameter, accounting for the reserve clearance for battery mounting and wiring, and keeping the number of battery count low for simple charging, the options were narrowed down to the list of batteries in table 4.4.

| Battery Type | Form Factor | Dimensions (mm) | Voltage | Capacity |
|--------------|-------------|-----------------|---------|---------------|
| 18650 Li-ion | Cylindrical | Ø18 × 65 | 3.7 V | 2600–3500 mAh |
| 14500 Li-ion | Cylindrical | Ø14 × 50 | 3.7 V | 800–1000 mAh |
| LiPo 602030 | Flat pouch | 20 × 6 × 30 | 3.7 V | ~350 mAh |
| LiPo 702025 | Flat pouch | 20 × 7 × 25 | 3.7 V | ~400 mAh |
| LiPo 402530 | Flat pouch | 25 × 4 × 30 | 3.7 V | ~400–450 mAh |

Table 4.4: Battery options.

Since all viable battery options operate at a nominal voltage of 3.7 V, two cells connected in series are necessary to meet the input voltage requirements of the MPM3610 step-down converter. This configuration brings the nominal voltage of the power source to 7.4 V. Based on this knowledge, the efficiency of the power source can be estimated around 85% based on graph 4.1.

As a result, the batteries' joint capacity requirement can be estimated accordingly. The first step is to convert Arduino's current consumption into power:

$$P_{out} = 3.3 \text{ V} \times 0.224 \text{ A} = 0.7392 \text{ W} \quad (4.4)$$

Input power needed from the batteries, accounting for an assumed efficiency of 85% is then:

$$P_{in} = \frac{P_{out}}{\eta} = \frac{0.7392}{0.85} = 0.8696 \text{ W} \quad (4.5)$$

Finally, the battery's current draw at their nominal voltage of 7.4 V is:

$$I_{\text{battery}} = \frac{P_{\text{in}}}{V_{\text{battery}}} = \frac{0.8696}{7.4} = 0.118 \text{ A} = 118 \text{ mA} \quad (4.6)$$

Based on this calculation, in order to comply with the battery life requirement of 24 hours of continuous activity, the minimum required battery capacity is:

$$118 \text{ mA} \times 24 \text{ h} = 2832 \text{ mAh} \quad (4.7)$$

This minimum capacity requirement makes the 18650 Li-ion battery a favorable choice, as it offers the highest capacity among the viable options. It meets the energy demand using the least amount of batteries and eliminates the need for parallel connection of cells to increase capacity.

For this purpose, two WESTINGHOUSE 18650 Li-ion 2600 mAh batteries were chosen, resulting in a total capacity of 5200 mAh. While the manufacturer specifies a cut-off voltage of 2.75 V and full charge voltage of 4.2 V a buffer for preventing battery fast degradation of 0.5 V was chosen for extended battery longevity. This sets the joint voltage range from the power source from 6.5 V to 8.4 V.

This trade-off slightly reduces the usable capacity of batteries down to 90% of the nominal rating, as illustrated in the graph below.[7][13]

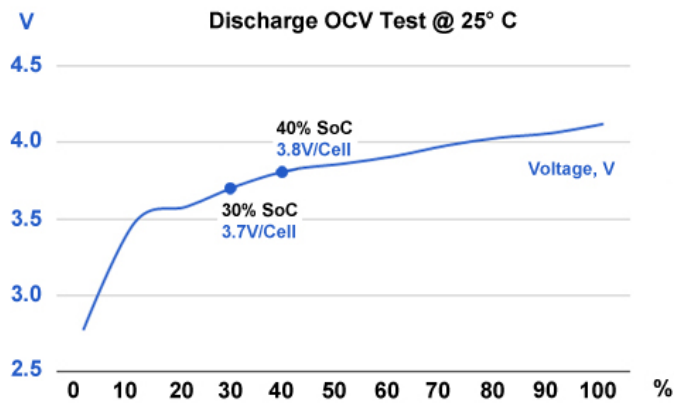


Figure 4.2: Discharge voltage of Li-ion batteries [7]

As a result, the available capacity of the batteries is down to 4680 mAh, which results in the estimated battery life of the device as approximately 40 hours of continuous usage.

Charging

When charging lithium-ion batteries, the protection circuit is strongly advised. Such a circuit ensures cell balancing and prevents overcharging, both of which are essential for maintaining battery safety and maximizing the lifespan of the batteries. For this purpose, the HW-391 battery management system (BSM) was selected. This BSM handles charging and balancing cells during charging and protects batteries from overcharge and over-discharge. It is specifically designed for 2-cell Li-ion battery packs, making it well-suited for this application.[1]

To ensure compatibility with a common phone charger, which typically operates at 5 V a step-up booster MT3608 is connected to the input BSM terminal and set to boost 5 V

up to 8.6 V required by the BSM. MT3608 is capable of boosting input voltage from 2 V to 24 V up to 28 V at the output with high efficiency.[5]

With the charging circuit in place, the theoretical maximum charging power is determined by the most restrictive component:

$$\begin{aligned} P_{\text{BMS}} &= 8.6 \text{ V} \times 10 \text{ A} = 86 \text{ W}, \\ P_{\text{cell}} &= 4.2 \text{ V} \times 2.6 \text{ A} \approx 10.92 \text{ W} \Rightarrow 2 \times P_{\text{cell}} \approx 22 \text{ W}, \\ P_{\text{boost}} &= 5 \text{ V} \times 4 \text{ A} \times 0.8 \approx 16 \text{ W}. \end{aligned} \quad (4.8)$$

Since P_{boost} is the smallest of the three, the MT3608 boost converter defines the maximum practical charging power in this setup. To prevent overheating of the voltage booster 10Ω resistor is placed between the booster and BSM. The estimated charging time, ignoring the small power consumption of the charging system, can be calculated using the equation:

$$\begin{aligned} I_{\text{lim}} &= \frac{V_{\text{boost}} - V_{\text{pack}}}{R} = \frac{8.6 \text{ V} - 7.4 \text{ V}}{10 \Omega} = 0.12 \text{ A} \\ t &= \frac{5.2 \text{ Ah} \times 7.4 \text{ V}}{(V_{\text{pack}} \times I_{\text{lim}}) \times 0.85} = \frac{38.48 \text{ Wh}}{(7.4 \text{ V} \times 0.12 \text{ A}) \times 0.85} \approx 51 \text{ h}. \end{aligned} \quad (4.9)$$

Finally, a waterproof USB-C port will be connected to the input terminals at MT3608 and placed in the T-grip of the paddle, ensuring minimal water exposure and inconvenience for the user.

Strain gauge

In the initial stage of development, measuring the mechanical strain of the paddle was also considered as part of performance analysis. The concept was to place a strain gauge that measures the bending of the paddle during strokes in the Wheatstone bridge. The strain gauges change their electrical resistance when bent even slightly, and by running a voltage across this bridge, the force put on the paddle can be deduced. The analog signal from the bridge would then be amplified and connected to an analog digital converter, which would be connected to the microcontroller's digital pin for data processing as illustrated in diagram 4.3. [3]

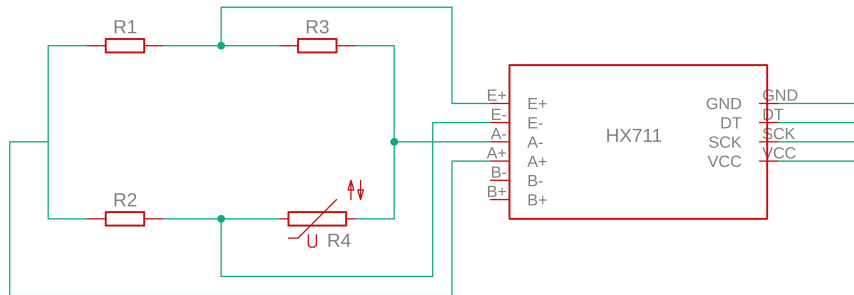


Figure 4.3: Wheatstone diagram circuit

This concept was later abandoned due to the significant difficulty of properly mounting the strain gauge inside the paddle shaft. The strain gauge's need for proper measurement

would result in a permanent attachment to the inner wall of the paddle shaft and be fixed to mimic the shaft’s material flexing. This would also limit the position of the lower hand to certain areas and restrain the athletes during performance.

4.3 Prototype

After the detailed process of component selection, the prototype was designed and wired as shown in diagram 4.4.

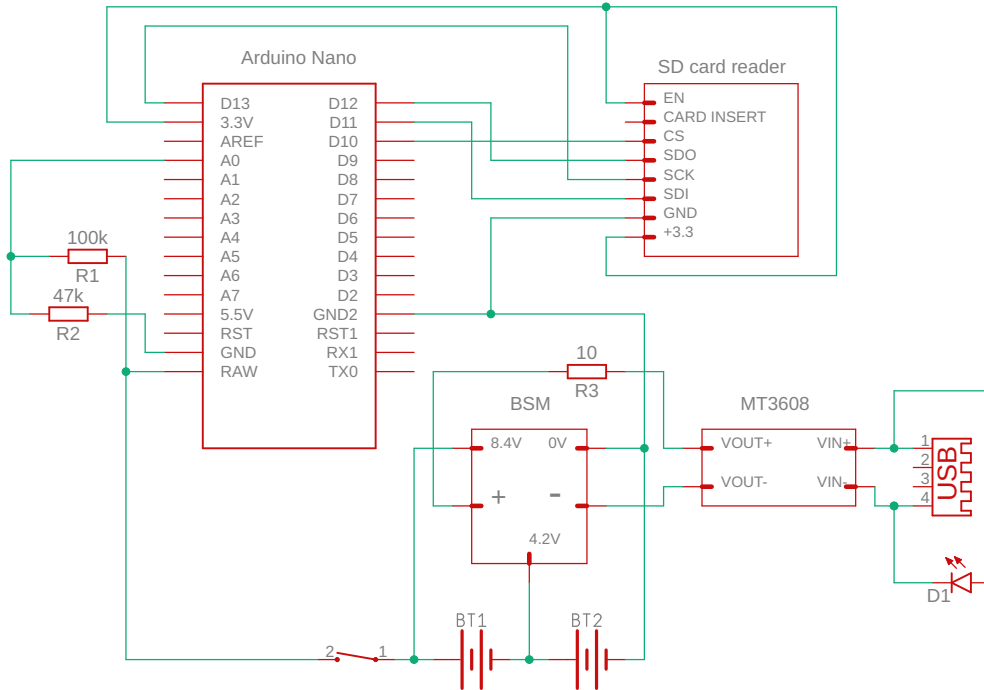


Figure 4.4: Prototype wiring diagram

This prototype features all the hardware components previously chosen, with the addition of a power switch, an LED diode, and a voltage divider made of 2 resistors. The switch is intended to be installed in the T-grip of the paddle, allowing turning the device on and off without interfering with the charging circuit. The LED diode is connected to the USB power lines to indicate the charging process. Finally, the voltage divider enables the microcontroller to read the battery’s voltage and estimate the current stage of charge.

The switch selected to be mounted in the paddle is the FLM12-FW-1, which meets all electrical requirements and has an integrated LED diode. This eliminates the need to add additional components to the limited space in the T-grip of the paddle. The switch features a self-locking mechanism that prevents accidental switching off during the activity. The fixed attachment is secured by the threaded body of the switch and tightened with a nut. The switch is IP65-rated, providing protection against dust and low-pressure water, making it suitable for this application.[22]

The voltage divider is present for safely reading the voltage due to the voltage limit that the Arduino can handle. The maximum voltage the batteries can provide is 8.4 V and Arduino can safely read signals on analog pins up to 3.3 V. To scale to voltage down to a readable level, the voltage divider consists of a 100 kΩ resistor, and a 47 kΩ resistor is used. This configuration reduces the input voltage according to the standard voltage divider formula:

$$V_A = V_{\text{BAT}} \cdot \frac{R_2}{R_1 + R_2} = 8.4 \text{ V} \cdot \frac{47 \text{ k}\Omega}{100 \text{ k}\Omega + 47 \text{ k}\Omega} \approx 2.69 \text{ V} \quad (4.10)$$

According to this equation, now the maximum voltage on the Arduino analog pin is 2.69 V, well below the safe limit of the Arduino’s analog pins. Using Ohm’s law, the current consumption of this divider can be calculated accordingly:

$$I = \frac{V}{R_1 + R_2} = \frac{8.4 \text{ V}}{100\,000 \Omega + 47\,000 \Omega} \approx 57 \mu\text{A} \quad (4.11)$$

This current consumption, at the maximum voltage the batteries can provide, is negligible compared to the overall power consumption of the circuit and is not accounted for in battery life expectancy.

4.4 Firmware

The firmware was developed using the Arduino IDE, designed specifically for programming microcontrollers. It provides a user-friendly interface for writing, compiling, and uploading code to the Arduino Nano 33 BLE. This environment supports also serial communication with the microcontroller which is great for debugging and testing the software and hardware capabilities.

Accelerometer and gyroscope

For collecting the data from the accelerometer and gyroscope, the `Arduino_LSM9DS1.h` library was used, which is designed specifically for the board used. The default setting for the library is listed in table 4.5.

| Sensor | Measurement Range | Resolution | Sampling Rate |
|---------------|-------------------|------------|---------------|
| Accelerometer | ±4 g | 0.122 mg | 119 Hz |
| Gyroscope | ±2000 dps | 70 mdps | 119 Hz |

Table 4.5: Default settings of the `Arduino_LSM9DS1` library.

These parameters are set internally in the library by following setting following registers without any function to change any of these parameters.

```
writeRegister(LSM9DS1_ADDRESS, LSM9DS1_CTRL_REG1_G, 0x78);
writeRegister(LSM9DS1_ADDRESS, LSM9DS1_CTRL_REG6_XL, 0x70);
```

According to the datasheet, the gyroscope is already at its maximal range of ±2000 dps. However, the accelerometer sensitivity can be set up to ±16 g. Based on the expectation that the paddle may experience forces beyond 4 g during intense strokes, this sensitivity is set to ±8 g. To set this accelerometer on ±8 g sensitivity, the `FS_XL` bits in the

CTRL_REG6_XL register are set from 10 to 11. This results in the final CTRL_REG6_XL register value of 0x78.[17]

```
writeRegister(LSM9DS1_ADDRESS, LSM9DS1_CTRL_REG1_G, 0x78);
writeRegister(LSM9DS1_ADDRESS, LSM9DS1_CTRL_REG6_XL, 0x78);
```

The new setting also affects the resolution of the accelerometer. The summary of the new setting is displayed in table 4.6.

| Sensor | Measurement Range | Resolution | Sampling Rate |
|---------------|-------------------|------------|---------------|
| Accelerometer | ± 8 g | 0.224 mg | 119 Hz |
| Gyroscope | ± 2000 dps | 70 mdps | 119 Hz |

Table 4.6: Final settings of the Arduino_LSM9DS1 library.

Calibration

To improve measurement accuracy, the sensors were calibrated through a two-step process. First, each axis of the accelerometer and gyroscope was offset-corrected by placing the Arduino in a stationary position where the values were expected to be zero, and by averaging the values, the offset was calculated. Next, the slope was calculated on offset-corrected values. For the accelerometer, this involves positioning the device where the values are expected to read ± 1 g, determined by the gravitation force, and comparing it to the actual values. For the gyroscope, a known rotation over a defined angle and duration was performed, and the resulting angular velocity was compared to the expected value. The final corrected measurement is then calculated accordingly:

$$\text{corrected_value} = \text{slope} \times (\text{value} - \text{offset}) \quad (4.12)$$

This correction was then incorporated into the library, providing calibrated accelerometer and gyroscope values to be computed automatically during every read operation.

SD card

For communication with the SD card module, the SDfat library was used. Since the data on the SD card is organized in 512-byte sectors, the program should aim to write data close to these sector boundaries to achieve optimal write performance. The data from the IMU is temporarily stored in a structure consisting of seven 4-byte values (a timestamp and six sensor measurements). The resulting size of one measurement sample is 28 bytes. The processor used in Arduino Nano 33 BLE aligns memory on 4-byte boundaries, so no additional padding is present. This results in writing the samples in a 504-byte block consisting of 18 samples. [15][18]

When testing the writing and reading speed of the SD card module connected to Arduino Nano 33 BLE using the SDfat library, the speed for reading was around 300 kB/s and for writing 200 kB/s. Based on this test, writing data to the SD card should not interfere with sampling from IMU since samples take approximately every 10 ms and writing data to the SD card should be finished in a few milliseconds.

Batteries charge

To monitor the state of charge of the batteries, the raw signal from the analog pin connected to a voltage divider is read. This value is a 10-bit integer ranging from 0 up to 1023. The actual voltage on the pin is calculated where the reference voltage of the microcontroller is 3.3 V:

$$V_{\text{pin}} = \left(\frac{\text{raw}}{1023.0} \right) \times V_{\text{REF}} \quad (4.13)$$

Then the voltage on the pin is scaled up to represent the voltage of the batteries:

$$V_{\text{batt}} = V_{\text{pin}} \times \left(\frac{R_2 + R_1}{R_1} \right) \quad (4.14)$$

Finally, the percentage of charge state is calculated based on the current battery's voltage and rounded:

$$\text{pct} = \left\lfloor \left(\frac{V - V_{\text{min}}}{V_{\text{max}} - V_{\text{min}}} \times 100 \right) + 0.5 \right\rfloor \quad (4.15)$$

Bluetooth

For communication with a mobile phone, the ArduinoBLE library was used. This library allows Bluetooth low energy (BLE) communication between Arduino and other BLE-compatible devices. BLE was introduced as part of the Bluetooth 4.0 standard and is optimized for periodic data exchange while preserving battery life. BLE uses client-server architecture where a central device (mobile phone) is connected to a peripheral (Arduino). Data is exchanged through services and characteristics, with communication supported via reading, writing, and notifications.

For communication, a custom BLE service is created. This service is identified by a Universally Unique Identifier (UUID) and contains two characteristics.

- Send characteristic is used by Arduino to transmit data to the mobile phone. It is configured as readable and notifiable, allowing the phone to receive updates without polling.
- Receive characteristic used by the mobile phone to send commands to the Arduino. It is configured as writable.

The packets consist of a header and payload. Each header has its unique code representing the type of packet. The list of different packet headers used is listed in table 4.7 and 4.8.

| Packet | Meaning |
|-----------------|--|
| T | Request to synchronize time. |
| C<timestamp> | Final corrected timestamp. |
| S | Start measurement and data logging. |
| E | End measurement, initiate data transmission. |
| B | Request current battery percentage. |
| ACK,<index> | Acknowledge individual data packet. |
| ACK,NUM,<count> | Acknowledge header packet indicating number of data samples. |
| ACKFIN,<count> | Acknowledge final packet confirming end of transmission. |
| ACK,0,<offset> | Acknowledge time offset synchronization packet. |

Table 4.7: Packets received by Arduino

| Packet | Meaning |
|-------------|--|
| R | Response to synchronization request. |
| B<percent> | Battery level in percent (0–100%). |
| NUM,<count> | Header indicating number of measurements to follow. |
| O<offset> | Time offset value . |
| M<data> | Binary measurement packet with a timestamp, accelerometer, and gyroscope data. |
| FIN,<count> | End of transmission confirmation with total number of samples sent. |

Table 4.8: Packets received by Android

Data compression

The maximum payload the BLE can transfer in one packet is 20 bytes. To minimize transfer time for measured data, the values for each sample were compressed and fit into one packet with all necessary information. The packet bits are used as represented in table 4.9.

| Byte | Field | Description |
|-------|--------------|----------------------------------|
| 0 | Header | Packet identifier, fixed to 'M' |
| 1–3 | Sample Index | 24-bit sample index |
| 4–7 | Timestamp | 32-bit timestamp in milliseconds |
| 8–9 | Gyro X | Scaled gyroscope X value |
| 10–11 | Gyro Y | Scaled gyroscope Y value |
| 12–13 | Gyro Z | Scaled gyroscope Z value |
| 14–15 | Accel X | Scaled accelerometer X value |
| 16–17 | Accel Y | Scaled accelerometer Y value |
| 18–19 | Accel Z | Scaled accelerometer Z value |

Table 4.9: Structure of a 20-byte sensor data packet

This packet composition allows for indexing up to $2^{24} = 16,774,216$ individual packets. The timestamp included in each packet represents the number of milliseconds since the activity began. It is stored as a 32-bit unsigned integer and can represent values up to

$2^{32} = 4,294,967,296$, corresponding to over 49 days. This packet composition results in a maximal duration of the activity of approximately 39 hours due to the packet indexing limit, which is well beyond any expected maximum duration.

The accelerometer data are scaled to a 16-bit integer in this packet. Since the forces can reach values in the range of -8 g to 8 g, the measured value is multiplied by 4095 using the full range of a 16-bit signed integer and then scaled back on the mobile phone. This scaling slightly reduces the accuracy of the measurement, but it is negligible in this use case.

Similarly, the gyroscope values ranging from -2000 dps up to 2000 dps are multiplied by 16.3835 to fully scale to the 16-bit integer.

Time synchronization

To synchronize data samples with measurements from a mobile phone, the time is synchronized between devices. This is done by sending to the Arduino a packet with the current timestamp and measuring the delay the transition introduces. The offset is then calculated on the mobile phone, and the corrected time is sent to Arduino, accounting for the transfer delay.

$$\theta = \frac{t_r - t_s}{2} \quad t_c = t_n + \theta \quad (4.16)$$

- t_s – Time when the synchronization request is sent from Android.
- t_r – Time when the response is received by Android.
- θ – Estimated one-way delay (round-trip time divided by 2).
- t_n - Current time when the corrected time is sent.
- t_c – Corrected time sent to Arduino.

This calculation assumes the duration to be equal for both directions. The flow of the packets during time synchronization is illustrated in diagram 4.5.

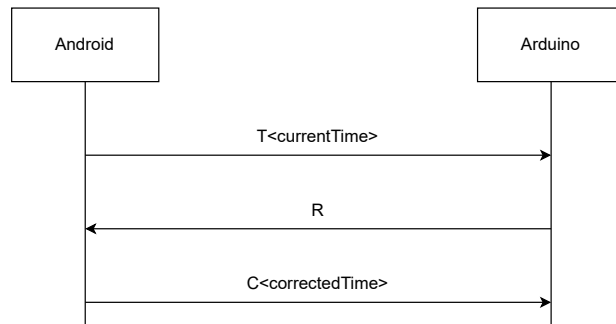


Figure 4.5: Time synchronization of Arduino

After the time synchronization process, the current time from Arduino’s internal clock is stored. The correct current time is then calculated when needed by adding the elapsed time since the last synchronization to the previously synchronized time.

The BLE library does not support indicate data operation and only notify for data operations without a request. During data transfer, some packets may be lost during transmission due to system overload. Indication provides the acknowledgment of each sent packet, but notification does not. For this reason, the acknowledgment system was implemented, ensuring each measurement is received by the phone. Each measurement packet includes an index of the sample that serves as the acknowledgment of the sent packet. When the packet is not acknowledged within a certain time span the packet is resent until the successful delivery is confirmed. The activity starts with sending an S packet and collecting measurements ends with an E packet. After Arduino receives the E packet, it sends a time offset packet. This offset is added to the timestamp of the measurement and reduces the number of bytes in the measurement packet. After that, the packet with the number of samples is sent, providing feedback for the phone on the progress of the transmission. The measurements are then sent, and Arduino expects acknowledgment before sending the next packet. The transmission ends with a FIN packet with a number of samples for control, and ACKFIN confirms the end of transmission. The packet flow is visualized in graph 4.6.

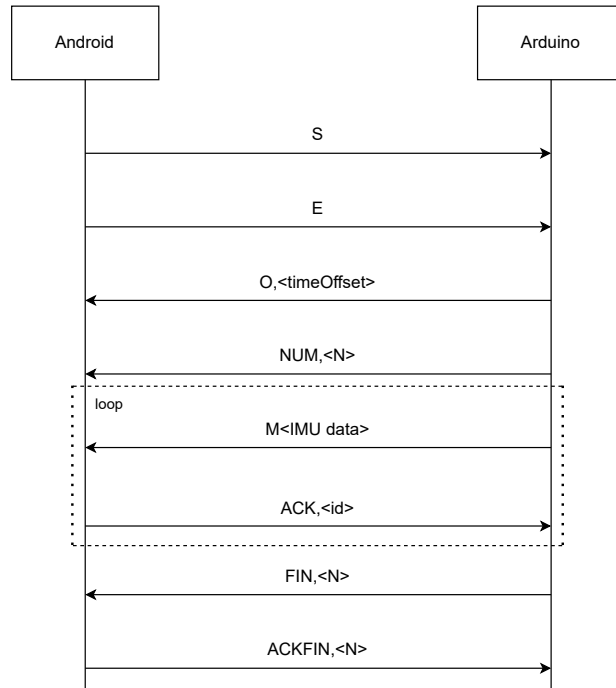


Figure 4.6: Flow of activity packets

Final prototype

The final prototype was made by placing and soldering the selected components on a universal printed circuit board (PCB). The PCB used is 160 mm long and 20 mm wide, fitting all components meant to be placed in the paddle shaft. The Arduino is placed at the bottom closest to the paddle blade for the precise measurements from the IMU, with the USB port facing down for any software updates. Right above the Arduino is the SD card

module placed close to the SPI pins, ensuring the shortest path of wires for SPI communication and reducing the resistance, which would slow down communication speed. The charging components are then placed at the top of the PCB. The batteries are placed in battery holders welded together and secured to the bottom part of the PCB with a slotted L-bracket and screwed to the PCB. To ensure the fixed placement of the PCB inside the paddle shaft, a steel wall anchor is mounted to the upper part of the PCB, preventing any movement of the device.

To provide access to the USB port and power switch, two precise openings were drilled into the T-grip of the paddle. These openings were carefully located to minimize interference with the added components during the activity. The completed physical prototype is shown in figure 4.7.



Figure 4.7: Completed prototype

Before placing the device inside the paddle shaft, all exposed pins and conductive parts of the circuit that may come in contact with the inside wall were covered with isolation material to prevent short circuits. This precaution was necessary because the shaft is made of aluminum, which could cause electrical shorts if it comes into contact with unprotected logic components.

Chapter 5

Android application

The mobile application was developed using Android Studio, the official integrated development environment for Android development. It supports development in Kotlin and Java and provides tools for code editing and debugging with real-time testing on physical devices. Kotlin was chosen for the app development for its features and strong integration with Android APIs.

The device this application was developed on is a Samsung Galaxy S10e with Android 12 installed, corresponding to API level 31 of the Android SDK. As a result, the application is designed for devices with SDK 31 or higher. Compatibility with devices with lower SDK is not supported due to changes in Bluetooth API introduced with SDK 31, which affect permission handling and BLE operations.

The purpose of the mobile application is to receive and store data from the Arduino, send commands to the Arduino, capture and store relevant data from its sensors, and provide post-activity analysis based on the collected information.

5.1 Captured data

The application captures and stores data from phones' internal sensors, namely the accelerometer, gyroscope, and GPS. The gyroscope and accelerometer data are timestamped and stored in a database. To estimate the speed of the boat, the application uses a Kalman filter, which fuses data from the GPS and linear acceleration sensor.

The Kalman filter is a recursive algorithm used to estimate the state of a system from noisy measurements. It fuses multiple sources of data to obtain a more accurate estimate than any single source alone. This approach allows for more accurate and stable speed estimation by balancing the influence of noise from GPS and accelerometer drift and cumulative error. The filter operations used to estimate speed:

$$\begin{aligned}\hat{x}_p &= x_{k-1} + a_k \times \Delta t \\ p_p &= p_{k-1} + q \\ k_k &= \frac{p_p}{p_p + r} \\ x_k &= \hat{x}_p + k_k(z_k - \hat{x}_p) \\ p_k &= (1 - k_k)p_p\end{aligned}\tag{5.1}$$

- x_k – Boat speed at time step k
- x_{k-1} – Boat speed at time step $k - 1$
- \hat{x}_p – Predicted speed from acceleration integration
- a_k – Measured acceleration at time step k
- Δt – Time interval between $k - 1$ and k
- z_k – Measured speed from GPS at time step k
- k_k – Kalman gain at time step k
- p_k – Updated error covariance
- p_p – Predicted error covariance
- p_{k-1} – Previous error covariance
- q – Process noise covariance (uncertainty in the model)
- r – Measurement noise covariance (GPS noise)

If the GPS signal is too weak at any time step, the corresponding data is ignored to maintain accuracy at any time.

5.2 Database

The mobile application uses the Room persistence library, which provides an abstraction layer over SQLite, to store all captured data from the activity. The Room has three major components: [4]

- Database class - An abstract class annotated with `@Database`, that holds the database and serves as the main access point. In this class, the Entities and Data Access Object are registered to the database. This class also defines the version of the database and the migration of the data from older versions.
- Data Access Objects (DAO) - DAO is an interface annotated with `@DAO`, providing methods of interaction with the database. These methods include `@Insert`, `@Update`, `@Delete`, and `@Query`. At compile time, the Room automatically generates an implementation of the defined DAO.
- Data Entities - Data Entries are data classes annotated with `@Entity` and define each table in the database. There are defined keys, columns, and data types.

The database schema consists of several tables storing the activity data. The central table Records stores metadata about each activity. This table is linked to all the other tables via a foreign key, separating individual activities by their ID. Table DataArduino stores data received from Arduino at the end of the activity. AccelDataAndroid and GyroDataAndroid store data collected during activity from the phone's sensors in real-time. And finally, the SpeedDataAndroid stores the speed of the boat during activity in km/h. The schema of the database is visualized in diagram 5.1.

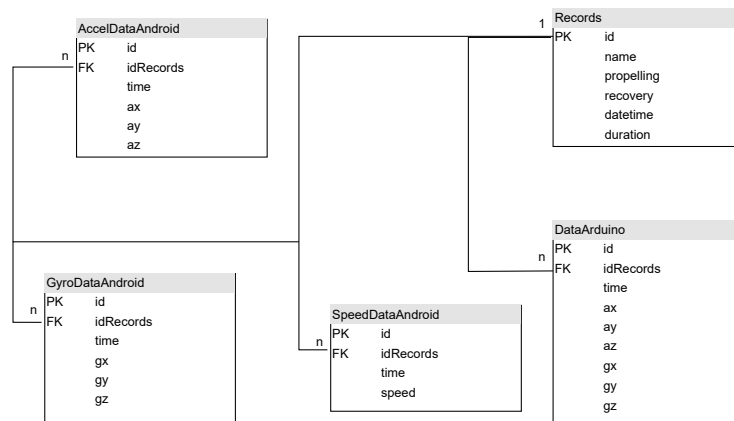


Figure 5.1: Database diagram

5.3 User interface

The user interface features a simple and clear design and ensures quick access to core features during activity. The application has four separate Activities (screens).

The main screen provides control and feedback on current activity and displays the status of the Arduino. The visual design is displayed in image 5.2. This screen features these components:

- Speed display - The current estimated speed in real time, even without starting the activity
- Start/End button - This button can start and end the activity. The button is disabled until the Arduino is connected and synchronized. When the Arduino is synchronized, the button displays the text Start. After the activity starts, the button displays the text End. When the activity ends, the button is disabled until the data transmission is finished.
- Timer - This timer shows the duration of the current session, and it resets when all data is transmitted.
- Progress bar - During the data transmission, the progress bar appears at the bottom with a percentage of the process and disappears when finished.
- Bluetooth icon - This icon indicates the Bluetooth connection with Arduino. The color of the icon is red when the Arduino is disconnected and turns green when the connection is established. This icon also serves as a button, and when clicked, it switches to the Bluetooth connection screen.
- Synchronization icon - This icon indicates the time synchronization with Arduino. If the icon is red, the time is not synchronized and turns green after synchronization. This icon also serves as a button for time synchronization if the automatic synchronization fails.
- Battery charge state - The battery icon shows the charge status of the Arduino with the exact percentage next to the icon. When the battery drops below 20 %, starting

new activity is disabled, and a message appears on the screen to notify the user to charge the battery.

- Menu - This menu features the main screen and the stored activities screen. The menu can be accessed by clicking on the menu icon or sliding the screen from the left corner.

The Bluetooth connection screen features a button for starting the scanning and a button for returning to the main activity for convenience. Found devices are then listed on the screen, and the user can click on the one to connect to. When scanning stops, the notification pops up as well as on a successful connection. The visual representation can be seen in image 5.2.

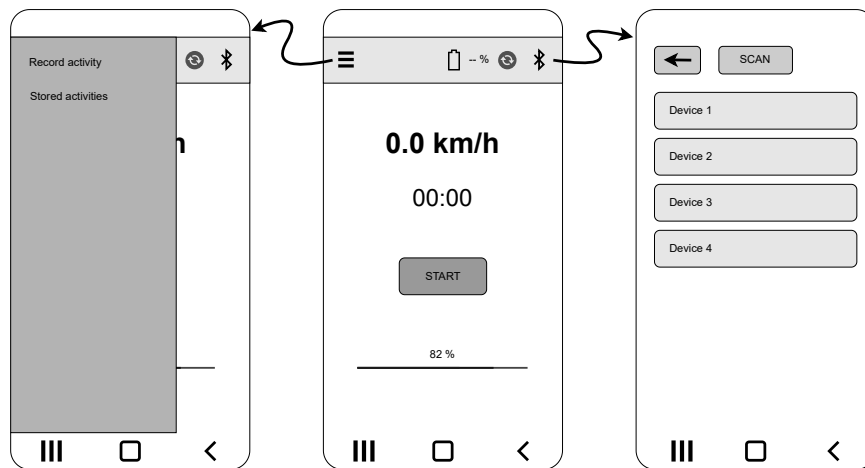


Figure 5.2: Interface design of main and Bluetooth screen

The screen with stored activities provides an overview of all activities sorted by time. The user can select an activity for detailed review. This screen features:

- Menu - Same menu as on the main screen.
- Time categories - The activities are sorted into three categories depending on the date they have been recorded. The first category is activities recorded today, the second category is activities recorded past seven days, excluding today, and the last category is activities older than seven days.
- List of activities - Here, the activities in the selected category are listed, and their name, date, and duration are displayed. When the activity is clicked, the screen switches to the detailed activity screen.
- Pencil icon - This icon serves as a button for changing the name of the activity. The user can name activities to describe the details of the activity for better navigation. When clicked, the pop-up window appears with text input and buttons Cancel and OK.
- Delete icon - This icon serves for deleting activities. When clicked, the app asks the user to confirm or cancel the deleting process of the activity.

- Page navigation - When there are more than 25 activities in the category, they are separated into pages, and buttons Prev and Next serve for switching between pages. The current page and the number of pages are displayed between these buttons.

The detailed activity screen provides a detailed analysis of the activity, where users can review their performance and see their strengths and room for improvement in the technique used during that activity. This screen consists of:

- Overview statistics - In this section, the app shows duration, average speed, cadence, average propelling and recovery phase duration, and number of strokes during activity.
- Graphs - There are four graphs showing cadence, speed, duration of the propelling phase, and duration of the recovery phase over the course of the activity. One graph shows an estimate of the average paddle blade angle according to the ground. Each graph has a line with a different color for better orientation.

The graphical visualization of stored activities and the detailed activity screen are displayed in image 5.3.

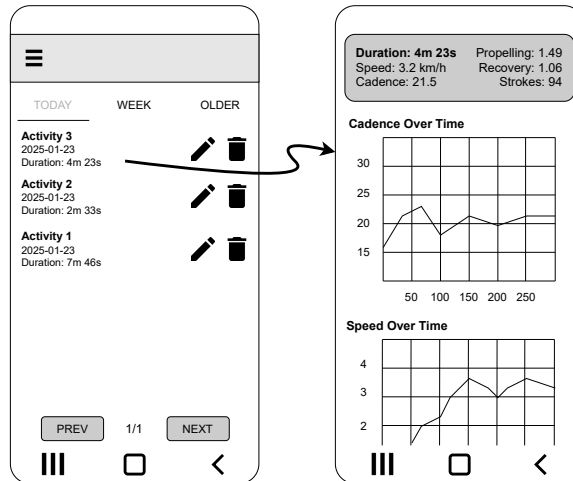


Figure 5.3: Interface design of stored activities and detailed activity screen

5.4 Activity analysis

The activity analysis provides a detailed review of cadence, stroke phase durations, speed, and blade angle. The first step in detailed analysis is to eliminate data from the start and end of the activity when the user is not engaged in paddling. The paddle at rest is usually placed in a horizontal position. Based on this assumption, the data from the time when the paddle is disengaged can be trimmed off and not included in the analysis. This is done by evaluating data from the Arduino X accelerometer axis and gravitational force. When the paddle is in the rest position, the value from this axis should be around 0g. When the value crosses the threshold of 0.5g, the paddle is considered to be engaged. At the end of the activity, when the value last crosses this threshold, it is also considered disengaged. On this trimmed data, the rest of the analysis is evaluated. The axis position of Arduino IMU when the paddle is in the engaged position is displayed in image 5.4.

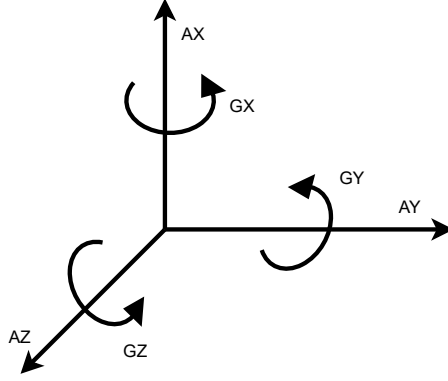


Figure 5.4: Arduino IMU axis

The cadence is calculated using a Fast Fourier Transform (FFT) on the accelerometer Z-axis. The FFT is an algorithm for computing a Discrete Fourier Transform (DFT), which transforms a signal from the time domain into the frequency domain. The DFT decomposes the signal into sinusoidal components and outputs a list of complex numbers representing strength and shift. From this output, the present waves are revealed. The DFT is defined as: [9]

$$X[k] = \sum_{n=0}^{N-1} x[n] e^{-j\frac{2\pi}{N}kn} \quad (5.2)$$

- $x[n]$ - Input signal
- N - Total number of samples
- k - Frequency index
- e - Euler's number
- j - Imaginary unit $\sqrt{-1}$

Once $X[k]$ is computed, each result corresponds to a physical frequency

$$f_k = \frac{k f_s}{N}, \quad f_s = \frac{1}{\Delta t} \quad (5.3)$$

where Δt is the sampling interval in seconds and f_s is the sampling rate in Hz. Then the frequency is converted to strokes per minute

$$C_k = 60 f_k \quad (5.4)$$

The range search is restricted from 5 to 80 strokes per minute. The strongest signal is picked from this range, and the cadence is then calculated in strokes per minute.

The next step in the analysis is to detect the propelling and recovery phases. This is achieved using data from the Z accelerometer axis, which captures forward and backward movement during strokes. A peak detection is applied to this signal. Due to the placement

of the Arduino in the paddle and the paddle tilt during the stroke, the acceleration signal on the z-axis typically increases during the propelling phase and decreases during the recovery phase. Previously calculated cadence is used to determine the search window as period * 1.2. Within the search window, the local minimum is found, representing the start of the propelling phase. The maximum value then represents the start of the recovery phase. The duration of each phase is then calculated by corresponding timestamps of the peaks, and the start time of the next stroke is set as the end of the previous stroke. This detection is visualized in graph 5.5 where the propelling phase is shown in green and the recovery phase in red.

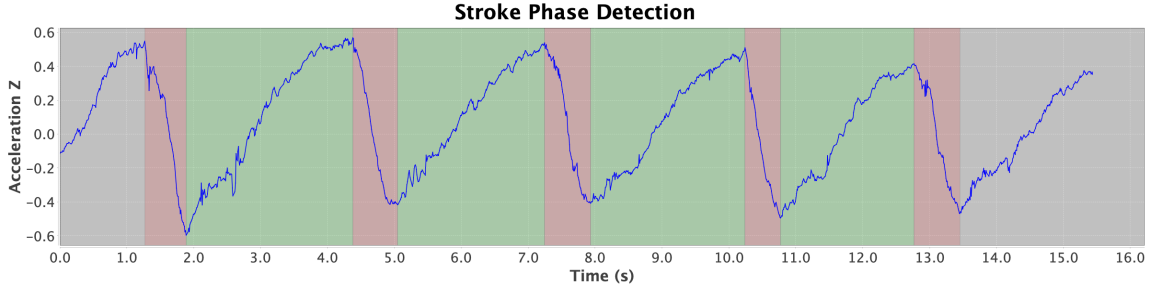


Figure 5.5: Stroke phases based on Z accelerometer axis

The cadence over the course of the activity is then derived from this phase detection and plotted into the graph.

The last step in performance analysis is to estimate the paddle's tilt during the propelling phase. For the most effective stroke, the paddle blade should be in a vertical position relative to the ground. This is very hard to achieve, and the paddle blade usually follows a curve instead of a straight line. The tilt is estimated using a complementary filter. This filter fuses the accelerometer X and Z axes and the gyroscope Y-axis. The initial tilt is calculated using the formula

$$\theta_0 = \arctan\left(\frac{-a_x}{a_z}\right) \quad (5.5)$$

The complementary filter is then applied [14]

$$\begin{aligned} \omega_g(t) &= g_y(t) \times \frac{\pi}{180} \\ \theta_a(t) &= \arctan\left(\frac{-a_x(t)}{a_z(t)}\right) \\ \theta_t &= \alpha (\theta_{t-1} + \omega_g \times \Delta t) + (1 - \alpha) \theta_a \end{aligned} \quad (5.6)$$

- θ_a - Angle estimated from the accelerometer using gravity
- ω_g - Angular velocity around Y-axis from the gyroscope
- α - Weight factor
- Δt - Time interval between samples
- θ_t - Estimated paddle tilt at time t in radians

The calculated tilt throughout the stroke is then converted from radians to degrees. Because every stroke has a different duration, every stroke tilt is resampled to 100 samples, and the average values are displayed in the graph.

5.5 Improving performance

From the conducted activity analysis, users are able to gain valuable insights into both the strengths and weaknesses of their paddling technique. The most forward aspect is the speed of the boat using different approaches. By comparing speed profiles of different paddling styles, users can identify which approach results in higher speeds. Generally, a higher sustainable speed implies a more efficient technique. The acceleration rate is also an important aspect of the technique, and users can experiment with which techniques are able to achieve a certain speed in the shortest time duration.

Another critical factor of technical proficiency is stroke consistency while maintaining the high, sustainable speed of the boat. An effective technique should maintain a stable cadence throughout speed maintenance with minimal fluctuations. Irregular strokes may suggest fatigue or inconsistent technique.

The tilt of the paddle plays a significant role in the effectiveness of transferring the force put into the paddle to the forward motion. The paddler should aim for the smallest curve of the paddle blade during the propulsion phase and try to maintain the paddle blade in a horizontal position relative to the ground. This angle transfers the most energy into propelling force and reduces the drag.

The propelling and recovery phases through the activity also show consistency and may reveal during which phase the irregularity occurs. Duration of phases can also reveal the unnecessarily long duration of the recovery phase compared to the propelling phase.

These metrics provide a detailed view of paddling performance, and based on them, the user can improve the paddling technique and follow the progress of improvement over time.

Chapter 6

Testing

One of the main concerns at the initial stage was the potential interference of the paddle aluminum shaft with the wireless signal, which would make wireless communication not a viable option. Testing has shown that the aluminum shaft does not have a significant impact on the stability or data transfer rate of Bluetooth connection. During the communication test, the acknowledgment of the measurement packet has proven necessary. Without the acknowledgment, the packet loss rate reached up to 30 %, which would have resulted in a significant loss of sensor data and made analysis practically impossible. The packet loss for other packets did not occur during the tests due to their smaller size and lower traffic.

The Bluetooth connection has proven to be stable without unwanted disconnections. The data transmission continues during the screen switching across all screens and even when the application runs in the phone's background.

While acknowledgment solved the data loss issue, it significantly prolonged the duration of data transfer. By acknowledging every measurement packet, the duration of data transfer exceeds the duration of the activity itself.

The resistor placed in the charging circuit, limiting the charging current, has proven to be necessary. Without this limitation, the booster reaches critical temperature in a few seconds and shuts down.

The prototype was successfully installed inside the paddle shaft without any issues and fixed in the position without sliding.

Finally, the system was tested in a real-world scenario on the water. All the metrics displayed by the application accurately represent the performance during the activity. The speed measurements were also compared to a third-party application, which showed similar results.

Chapter 7

Conclusion

The goal of this thesis was to design and implement a system consisting of an embedded system and a mobile application for analyzing the paddling performance. By embedding the device with sensors in the paddle and establishing communication with the phone, the system enables the paddlers to gain feedback on their performance.

The prototype successfully captured and transmitted acceleration and gyroscope data via Bluetooth to the phone, which provided a detailed analysis consisting of boat speed, cadence, stroke phase durations, and blade tilt.

The performance analysis accurately represents the technique and is able to detect changes in the technique aspects previously mentioned. This provides valuable insights into the paddler's technique and helps them to improve it.

The established requirements for the system were achieved, and the system serves as a reliable platform for performance analysis.

Despite these achievements, a few aspects still work differently than desired. The data transfer rate is relatively slow and has room for improvement. The charging speed is heavily compromised due to overheating and would benefit from a higher speed.

The Android application provides feedback for only one activity at a time and does not provide any statistics from multiple activities. Also, the conclusion of weaknesses is not suggested or pointed out, and it is up to the user to figure it out.

Further improvements could include optimizing data transfer and reducing the duration to a more reasonable range by using different technology, speeding up the charging by increasing the current or using different components, including suggestions in the activity analysis for improvements in technique, providing statistics from stored activities, providing real-time feedback during activity from the paddle, and implement other technique aspects in analysis like boat deflection during strokes, angle of paddle rotation, speed deflection between strokes or performance curves of the stroke.

In conclusion, this system provides a detailed paddling analysis in which the guarantor for canoeing at the Brno University of Technology showed interest in cooperation and further development.

- [14] NAZMI ÖZ ed, BUDAK, S., KURNAZ, E. and DURDU, A. *Orientation Determination in IMU Sensor with Complementary Filter* [online]. 2022. Available at: https://www.researchgate.net/publication/362248081_Orientation_Determination_in_IMU_Sensor_with_Complementary_Filter.
- [15] NORDIC SEMICONDUCTOR. *NRF52840 v1.1* [online]. 2024. Available at: https://docs.nordicsemi.com/bundle/nRF52840_PS_v1.1/resource/nRF52840_PS_v1.1.pdf.
- [16] ROTTENBACHER, C., MIMMI, G. and RAMPONI, A. *Experimental Analysis of Paddling Efficiency of Elite and Non-elite Athletes with Instrumented Canoe Sprint C1*. Pavia, IT, 2011. Research paper. University of Pavia, Department of Structural Mechanics,. Available at: <https://core.ac.uk/download/pdf/226415788.pdf>.
- [17] STMICROELECTRONICS. *LSM9DS1* [online]. 2015. Available at: <https://www.st.com/resource/en/datasheet/lsm9ds1.pdf>.
- [18] TRANSCEND INFORMATION. *TS256M 2GUSD* [online]. 2022. Available at: https://components101.com/sites/default/files/component_datasheet/Micro%20SD%20Card%20Datasheet.pdf.
- [19] U-BLOX. *NINA-W10 series* [online]. 2022. Available at: https://content.u-blox.com/sites/default/files/NINA-W10_DataSheet_UBX-17065507.pdf.
- [20] U-BLOX. *NINA-B3 series* [online]. 2023. Available at: https://content.u-blox.com/sites/default/files/NINA-B3_DataSheet_UBX-17052099.pdf.
- [21] WESTERN DIGITAL TECHNOLOGIES. *SanDisk Ultra® microSDXC™ & microSDHC™ UHS-I Cards* [online]. Available at: https://documents.westerndigital.com/content/dam/doc-library/en_us/assets/public/sandisk/product/memory-cards/ultra-uhs-i-microsd/data-sheet-ultra-uhs-i-microsd.pdf.
- [22] YUEQING YULIN ELECTRICAL. *FLM12-FW-1* [online]. Available at: <https://www.pushbuttonswitch.com/wp-content/uploads/2023/12/12mm-Push-Button-Switch-on-or-off.pdf>.