
# Flapping-Wing Aerial Aquatic Vehicles(FAAV) Software & Electronics

Laidlaw Research Report

---

Cristian Fortuna

6 October, 2023

# Acknowledgments

I wish to express my heartfelt gratitude to my supervisor, Raphael Zufferey, for affording me the opportunity to engage in this thrilling project during my summer internship. His guidance and expertise have been invaluable to my growth and development.

I am also deeply thankful to EPFL and the Laidlaw Foundation for their generous support throughout my journey at the lab this summer. I had the privilege of participating in numerous enriching workshops that not only enhanced my soft skills but also develop my understanding of ethical leadership.

# Flapping-Wing Aerial Aquatic Vehicles(FAAV) Software & Electronics

**Abstract**

The Laboratory of Intelligent Systems at EPFL is presently engaged in the development of Bio-Inspired Flapping-Wing Aerial-Aquatic Vehicles (FAAV), drawing inspiration from nature to create a robot capable of seamless transitions between flying and swimming. This innovative robot, inspired by the morphology of bird wings, is designed to collect data for oceanography and climate research. Despite notable progress in the realms of flapping-wing and aerial aquatic robotics, the integration of these two concepts remains largely unexplored, presenting several significant challenges. One such challenge involves striking the right balance in the robot's design to ensure efficient performance in both swimming and flying modes without overburdening it. Adapting wing motion to different environments poses another intricate problem to address. Moreover, achieving autonomy in the robot's operations is paramount, ensuring its ability to autonomously manage transitions between air and water without heavy reliance on user input. Bio-inspiration offers valuable insights and opens new avenues for advancements in robotics. Emulating the combined underwater swimming and flying capabilities, as seen in creatures like puffins, holds great potential to yield invaluable knowledge for further progress in this field.

# 1 Introduction

## 1.1 Context

A shoreline is within 100 km for 66% of the world's population and within 60 km for 40% of the world's population [1]. As a result, one can observe a significant negative impact on the world's seas, coasts, and marine ecosystems. In order to preserve the well-being of marine ecosystems, humanity needs to understand the significance of surveying, assessing, and recording the changes occurring in the ocean and coastal areas. Besides protecting the local habitats, monitoring water bodies has been crucial in climate studies due to their role in serving as greenhouse gas sinks, absorbing about 31% [3] of the CO2 emissions as well. However, present monitoring solutions are cumbersome, expensive, laborious, and frequently occur in hazardous areas for people, such as near icebergs, oil spills and contaminated lakes. Therefore, there is an urgent need to develop better solutions.

## 1.2 State of the art

Aerial-aquatic robots, which has only a few working prototypes, is expected to have a big impact, according to oceanographers [4]. Most aerial-aquatic robots struggle to transition from water to air smoothly or are unable to function underwater at all, which limits their utility

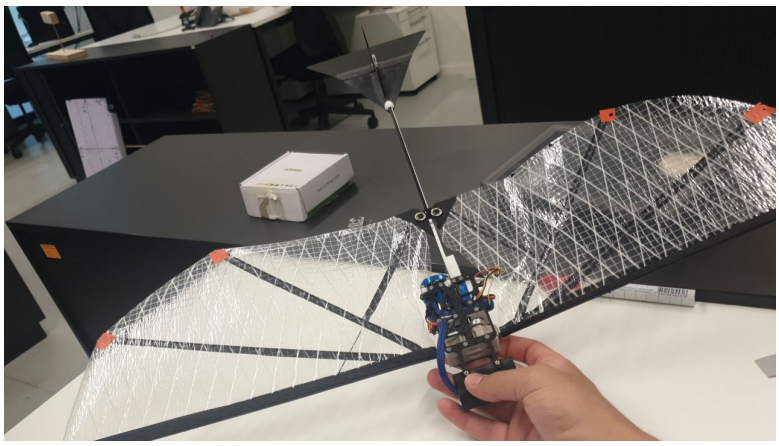


**Figure 1:** FAAV prototype, credits to my supervisor Raphael Zufferey

in aquatic operations. These robots normally have very limited flying times and distances, and they are unable to communicate underwater. Furthermore, their operation has only been proved in controlled settings with calm water conditions, making it unsuitable for fieldwork in actual field situations. These robots are not safe for close human collaboration because they use high-pressure systems, combustion, or propellers. Flapping-wing robots offer a viable option to increase mobility in both water and the air, which is essential for reliable data collecting missions.

## 1.3 Goals

My main contributions to the FAAV project will be the creation of software and electronics that will enable the robot to move on its own in both the air and the water.

My duties include developing a wireless graphical user interface that will allow people to alter the robots' behaviour, such as their wing frequency for instance. The user interface will also be in charge of collecting information from the numerous sensors that are mounted on the FAAV. I put in place procedures to keep the link between the FAAV and the master computer stable in order to guarantee seamless communication. In the event of any interruptions, the robot will quickly try to reconnect to the master.

Regarding electronics, my task will be to connect the microcontroller on the robot to all its peripherals: such as sensors and motors, making sure that the circuit is waterproof and quite robust. This setup will work thanks to a printed circuit board that I have designed specifically for this project.

# 2 Methodology

## 2.1 Electronics

### 2.1.1 Circuit

Given the robot's ability to swim, the priority was ensuring waterproofing integrity. Additionally, the imperative of minimizing the robot's bulkiness presented another notable concern. Hence, deliberations were held to consolidate all interconnections among system components onto a printed circuit board (PCB) encased within a water-resistant enclosure. The PCB design underwent two iterations, initially opting to abstain from directly embedding the microcontroller on the board. The PCB dimensions are notably compact, measuring approximately 18mm x 21mm. Consequently, unconventional design decisions were made, such as placing vias on the soldering pads to accommodate all the various components, including 0805 surface-mount resistors and capacitors. Milling the sides of PCB and MCU was also necessary to make the component as small as possible.

Considering the type of robot being developed, two primary objectives take precedence: minimizing the robot's weight and keeping its power consumption to a minimum. To achieve favorable outcomes in these aspects, it is essential for the components to be compact and cost-effective. The circuit comprises the following components:

- The XIAO BLE nRF52840 Sense microcontroller (MCU) is the optimal choice for this implementation. It boasts a small size, measuring approximately 18 mm x 21 mm, closely resembling the PCB's dimensions. With 6 analog pins equipped with a 10-bit resolution ADC (analog-to-digital converter), it proves ideal for sensor-centric applications. Furthermore, the MCU is equipped with a generous 2MB onboard flash memory, enabling extended data recording capabilities. The XIAO BLE Sense also incorporates a Bluetooth 5.0 NFC module with an onboard antenna, allowing for relatively long distance communication (around 200m).
- A basic voltage divider functioning as a voltage sensor, constructed with two resistors (approximately 330k$\Omega$ and 33k$\Omega$).
- The high-precision ACS723LLCTR-20AU-T serves as a low-offset, linear Hall sensor, capable of measuring currents up to 20A. Current flow through the copper conduction path generates a magnetic field, detected by the integrated Hall IC, and transformed into a proportional voltage. Its compact footprint, considerable sensitivity, and low internal resistance (approximately 0.65 m$\Omega$, rendering it well-suited for low-power applications) position it as an ideal solution for this robot.
- Three servos, motors designed for precise angular displacement control, are responsible for managing the robot's two wings and tail. Additionally, an Electronic Speed Controller (ESC) has been integrated to facilitate dynamic servo management. While ESCs are commonly associated with regulating the speed of electric motors (such as those found in drones or RC vehicles), they can be adapted for servo control in position-based applications.
- A pair of golden pins, two electrodes separated by a 5 mm gap, serve as water sensors, coupled with a 1M$\Omega$ resistor.

- A SK6812 RGBW LED, known for its low power consumption, has been included for debugging purposes.
- A battery power supply.

### 2.1.2 Manufacturing

Hand soldering becomes notably challenging when working with 5 mm components on an exceedingly small board. Consequently, we opted for a common technique in soldering components onto a PCB, namely hot plate soldering. This method entails the use of a heated plate, or hot plate, to melt solder and establish electrical connections between the components and the PCB.

To initiate the process, we bought a stencil for the PCB from the manufacturers. Subsequently, solder paste was applied to the stencil, which was securely taped to the PCB. Each component was then meticulously positioned on its designated pads. As the next step, the hot plate was gradually heated to a specific reflow temperature, typically hovering around the solder alloy's melting point, which is approximately 200°C in this case.

During this phase, the components were precisely aligned and held in place by the surface tension driven forces. Upon cooling, the solder connections solidified, effectively anchoring the components securely in their designated spots.

Furthermore, cable connections to the servos were soldered onto the pads as per the specific requirements of each experiment.

## 2.2 Software

### 2.2.1 General Approach

The primary purpose of the software is to establish and maintain a connection between the central PC and the robot throughout various tests, with the overarching goal of enhancing the overall testing experience. The Central User Interface (UI) has been developed using Python, while the MCU is programmed in C++, thanks to Platform IO, a VS Code extension.

One of the key priorities for this project has been modularity. While modularity has always been a crucial aspect of software development, it takes on particular significance in this project. In the course of different lab projects, it's common to find oneself re-implementing solutions that others have tackled in the past. Therefore, the creation of versatile and independent modules serves as a time-saving strategy for future lab projects.

This approach is especially valuable considering the widespread use of the XIAO BLE nRF52840 MCU. The modules developed during this project can easily find applications in other projects, eliminating the need to revisit poorly documented and obscure libraries.

### 2.2.2 BLE Communication Protocol

At its essence, Bluetooth is a short-range connectivity technology that employs radio waves for communication. The most familiar standard is Bluetooth Classic, which the majority of us are accustomed to, considering it is ubiquitous in consumer electronics like wireless headphones.

Introduced in 2010 as a part of the Bluetooth 4.0 specification, Bluetooth Low Energy (BLE) is specifically designed for ultra-low power applications [2]. Its optimization for such applications caters to a market of battery-powered devices that require wireless networking capabilities. This characteristic makes it particularly well-suited for use in a wireless robot.

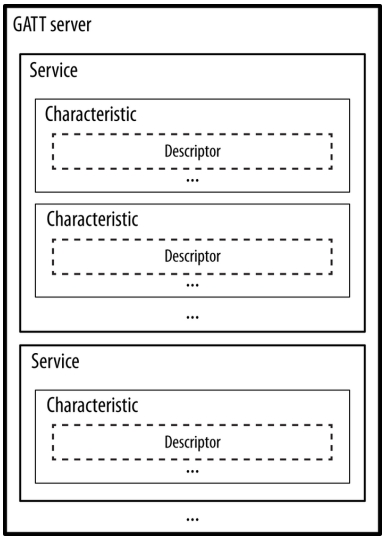



**Figure 2:** GATT hierarchy. Source: https://www.oreilly.com

The Generic Attribute Profile (GATT) defines the process for exchanging all profile and user data across a BLE connection. It has a predefined hierarchy. See figure 2. It aims to organize attributes in a reusable and practical manner, enabling access and retrieval of information between the client and server to adhere to a concise set of rules. These rules collectively form the framework employed by all GATT-based profiles.

### 2.2.3 MCU Programming

The robot operates in three primary modes: Remote Control, Configuration, and Sequence. The Remote Control (RC) mode is primarily used to accept direct movement commands from the central PC, without performing any additional actions to avoid introducing latency. The role of the Configuration mode is to enable the user to configure various parameters that govern the robot's behavior, while also allowing for straightforward sensor testing by collecting real-time data. The Sequence mode is intended for recording data onto the flash memory while the robot adheres to a predefined series of hard-coded actions. For instance, owing to water absorption and signal scattering, the robot may experience disconnection while submerged in water. Consequently, it proves beneficial to have a mode in which the robot follows a specific set of commands during this period, all the while gathering data on factors like power consumption. To support the above functionalities, three modules have been developed: Parameter, Sensor, and QSPI flash modules.

The parameter module serves to facilitate the initialization and transfer of parameters over BLE. Users can modify the MCU code according to their specific robot experiment requirements. Nevertheless, coding these changes for each experiment can be both troublesome and error-prone. Therefore, the central software must adapt to these alterations and generate a user interface based on the robot's code. Consequently, there is a need to develop a structured and relatively straightforward approach to parameter management. Currently, three types of parameters are in use: an integer, a bool and a list. Considering the structure of the GATT profile and the necessity to update them individually, parameters will possess their own characteristics within the parameter service. Considering that latency is not a big issue during configuration, this approach is quite viable. An illustration of this library's utilization is demonstrated in listing 1.

**Listing 1:** Parameter Module Usage

```
// Used for Platform IO on VS Code
#include <Arduino.h>
// BLE Arduino Library
#include <ArduinoBLE.h>
#include "Parameter.hpp"

#define DEVICE_NAME "Test Robot"

// Initializes all parameters
Parameter frequency("Frequency(Hz)", 60, ONE_VAL, "range(20,100)");
Parameter ledBlink("Blink Led?", 0, BOOL);
Parameter ledColour("Blink Colour", "RED", LIST, "RED#BLUE#GREEN");

void setup() {
    ...
    BLE.setLocalName(DEVICE_NAME);

    all_param_setup();
    add_paramService();
    advertise_paramService();

    BLE.advertise();
    ...
}
void loop(){
    ...
    switch (currentMode){
    case RC_MODE:
        ...
        break;
    case PARAM_MODE:
        // Checks all params for update
        all_param_update();
        break;
    case SEQUENCE:
        ...
        break;
    }
    delay(10);
}
```

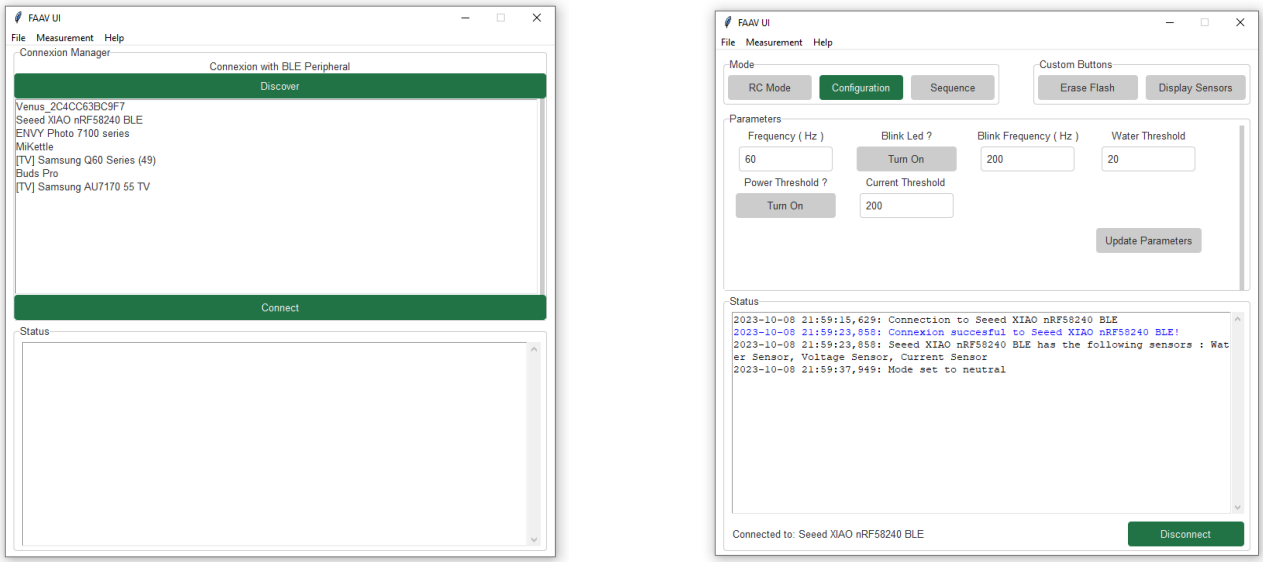
Another crucial module for the robot's operation is the sensor module, which bears strong similarities to the parameter one. However, due to the necessity of recording data while addressing potential latency issues, the approach to data transmission and storage differs significantly. Firstly, the readings from all sensors at a given time are concatenated and transmitted via a single characteristic. Secondly, to safeguard against data loss in case of power failure, the data is continuously written to flash memory. Analogous to the implementation of the parameter module, the sensor module follows a straightforward pattern, as depicted in listing 1.

The task of writing data to the Flash memory is managed by the QSPI Flash module [5]. The Quad Serial Peripheral Interface (QSPI) protocol serves as a standard interface for linking

MCUs with external Flash memory. QSPI operates as a high-speed, full-duplex, synchronous serial communication protocol, enabling fast data transfer. To prevent any potential data loss, a data counter stored in the Flash memory is continually incremented with each write operation.

### 2.2.4 Central User Interface

The user interface is coded mostly using the tkinter library with a custom theme. It consists of two primary modes: discover and connection modes. In discover mode, the primary task is to establish a connection with the chosen device from a list of peripherals currently being advertised. This involves the operation of two threads: one dedicated to managing the user interface, and the other responsible for handling the BLE connection. Additionally, a logger module is in place to handle various events occurring within different coroutines and update the status log through the use of an event queue. After establishing a connection with a device, the




**(a)** **(b)**

**Figure 3:** FAAV User Interface. (a) User is looking for devices to connect to. (b) Central is connected to the robot in configuration mode.

UI will dynamically generate widgets based on the list of parameters configured on the MCU. Simultaneously, it will initiate the creation of files to correspond with each sensor present on the robot. Given the numerous sensors involved, the FAAV UI heavily relies on the asyncio library to manage inputs and outputs from the robot in an asynchronous manner.

In the RC mode, the software directly transmits data from the Taranis remote to the robot. This mode is intentionally designed to be highly restrictive to minimize any potential latency during direct remote control operations. While the second mode is mainly for testing out different configurations of the robot, the third one primarily concentrates on data collection while executing a predefined sequence.

# 3 Results and Conclusion

## 3.1 Challenges

The most demanding aspect of this project entailed acquiring a diverse range of skills within a limited timeframe. It encompassed various competencies that were not extensively covered in my undergraduate studies, such as PCB design and concurrency and parallelism in programming. Additionally, a significant challenge involved navigating through numerous obscure and relatively new libraries, for which there was limited to no code readily available online. Many forum posts contained either outdated or unrelated code, making it challenging to find relevant resources for my specific application.

## 3.2 General Conclusions

Throughout this laboratory internship, I made significant contributions to the advancement of enhanced and dependable electronics for FAAVs. My adoption of PCBs instead of manually soldering components onto boards played a pivotal role in establishing robust connectivity and a more waterproof system. Furthermore, the software I developed during my internship holds the potential to be of great assistance in experiments related to FAAVs or other types of robots. In conclusion, my contributions will prove invaluable to the development of FAAVs, thereby facilitating climate research and oceanography.

# References


[1] "Are Arctic Sea Ice Melts Causing Sea Levels to Rise?" In: *Scientific American* (2008). URL: https://www.scientificamerican.com/article/arctic-ice-melts-cause-rising-sea/.

[2] "Bluetooth® Wireless Technology". In: (2023). URL: https://www.bluetooth.com/learn-about-bluetooth/tech-overview/.

[3] Nicolas Gruber et al. "The oceanic sink for anthropogenic CO2 from 1994 to 2007". In: *Science* 363.6432 (2019), pp. 1193–1199.

[4] P Stevenson. "Report on air launched autonomous underwater vehicles". In: (2011).

[5] "The QSPI Flash Usage on Seeed Studio XIAO nRF52840 Sense". In: (2023). URL: https://wiki.seeedstudio.com/xiao-ble-qspi-flash-usage/.