Bachelor Thesis

Czech Technical University in Prague

F3

Faculty of Electrical Engineering
Department of Cybernetics

Design of a Robotic Water Sampler for an Unmanned Aerial Vehicle

Daniel Štanc

Supervisor: Ing. Pavel Stoudek
Field of study: Cybernetics and Robotics
January 2022

BACHELOR‘S THESIS ASSIGNMENT

I. Personal and study details

Student's name:	**Štanc Daniel**	Personal ID number:	**483474**
Faculty / Institute:	**Faculty of Electrical Engineering**		
Department / Institute:	**Department of Cybernetics**		
Study program:	**Cybernetics and Robotics**		

II. Bachelor’s thesis details

Bachelor’s thesis title in English:

Design of a Robotic Water Sampler for an Unmanned Aerial Vehicle

Bachelor’s thesis title in Czech:

Návrh robotického vzorkovače vody pro autonomní multirotorovou helikoptéru

Guidelines:

This thesis aims to design a robotic water sampler to be carried by an Unmanned Aerial Vehicle (UAV) for outdoor water sampling. The UAV will be equipped with sensors required for autonomous flight and a device for safe sample taking from a water source.
1) Design the mechanical solution.
2) Design the electronic control unit.
a) Select appropriate parts for constructing the water sampler.
b) Design a suitable electrical circuitry for the control.
c) Use a STM32 family microcontroller to control the device and communicate with the onboard computer.
3) Implement a program for the onboard computer of the UAV to control the microcontroller via USB. This program must be implemented using ROS for use with the MRS Group system.
4) Test the functionality of the device.

Bibliography / sources:

[1] Ore, John-Paul, et al. "Autonomous aerial water sampling." Journal of Field Robotics 32.8 (2015): 1095-1113.
[2] Koparan, Cengiz, et al. "Evaluation of a UAV-assisted autonomous water sampling." Water 10.5 (2018): 655.
[3] Koparan, Cengiz, et al. "Autonomous in situ measurements of noncontaminant water quality indicators and sample collection with a UAV." Water 11.3 (2019): 604.
[4] Anis Koubaa, Robot Operating System (ROS), The Complete Reference (Volume 3), 2018
[5] Carmine Noviello, Mastering, STM32, 2018

Name and workplace of bachelor’s thesis supervisor:

Ing. Pavel Stoudek, Multi-robot Systems, FEE

Name and workplace of second bachelor’s thesis supervisor or consultant:

Date of bachelor’s thesis assignment: **28.07.2021** Deadline for bachelor thesis submission: **04.01.2022**

Assignment valid until: **19.02.2023**

Ing. Pavel Stoudek	prof. Ing. Tomáš Svoboda, Ph.D.	prof. Mgr. Petr Páta, Ph.D.
Supervisor’s signature	Head of department’s signature	Dean’s signature

III. Assignment receipt

The student acknowledges that the bachelor’s thesis is an individual work. The student must produce his thesis without the assistance of others, with the exception of provided consultations. Within the bachelor’s thesis, the author must state the names of consultants and include a list of references.

.	
Date of assignment receipt	Student’s signature

Acknowledgements

First and foremost, I would like to thank my thesis supervisor, Ing. Pavel Stoudek, for his valuable advice and patience during my work on this thesis.

I would also like to acknowledge Ing. David Žaitlík for his helpful comments on the PCB design.

Last but not least, I would like to thank my family and partner for their continuous support during my studies.

Declaration

I declare that the presented work was developed independently and that I have listed all sources of information used within it in accordance with the methodical instructions for observing the ethical principles in the preparation of university theses.

Prohlašuji, že jsem předloženou práci vypracoval samostatně a že jsem uvedl veškeré použité informační zdroje v souladu s Metodickým pokynem o dodržování etických principů při přípravě vysokoškolských závěrečných prací.

In Prague, 4 January 2022

Signature: ....

Abstract

Water sampling is an essential procedure for water quality monitoring. Traditionally, the samples are handpicked by humans, making it an expensive and time-consuming process. This thesis proposes a solution to this problem by creating a water sampling manipulator for an Unmanned Aerial Vehicle.

In the beginning, the thesis provides an overview of currently used solutions utilizing various Unmanned Vehicles. Based on the review, a water sampler suitable for a Tarot T650 drone is developed, starting with the description of the mechanical design. Next, a printed circuit board is created to control the mechanism. With the necessary hardware complete, the control board is programmed, and a ROS node is created to integrate the manipulator with the MRS system. Finally, the device is thoroughly tested separately and on a drone.

Keywords: water sampler, manipulator, unmanned aerial vehicle

Supervisor: Ing. Pavel Stoudek

Abstrakt

Odebírání vzorků je důležitou součástí monitorování kvality vody. Tradičně jsou vzorky odebírány ručně, což činí tento proces nákladným a zdlouhavým. V této práci je popsán vývoj vzorkovače vody pro bezpilotní multirotorovou helikoptéru, který řeší tento problém.

Nejprve je představen současný stav využití různých bezpilotních vozidel pro vzorkování vody. Na základě tohoto přehledu je navrhnut vzorkovač vody pro Tarot T650, mechanickým designem počínaje. Poté je popsán vývoj desky plošných spojů sloužící ke řízení tohoto přístroje. Deska je následně naprogramována. Aby toto zařízení mohlo být použito s MRS systémem, je v rámci této práce vytvořen také potřebný software pro integraci s ROS. Na závěr je zařízení otestováno jak v laboratoři, tak při letu dronu.

Klíčová slova: vzorkovač vody, manipulátor, bezpilotní dron

Překlad názvu: Návrh robotického vzorkovače vody pro autonomní multirotorovou helikoptéru

Contents

Figures

Tables

Chapter 1
Introduction

Unmanned Aerial Vehicles (UAV), also known as drones, first appeared about 100 years ago. For the majority of the time, they were mostly used for military purposes. However, since the 2000s and especially in the last decade, drones have rapidly emerged in civilian sectors, like aerial photography, industrial inspection and security, and many more [1]. This development was made possible by extensive research.

One of the appearing research areas is the usage of drones for water sampling. This process allows us to observe and measure water quality over time. It is a crucial step for solving many problems people have to face, like the spreading of diseases, such as diarrhea, dysentery, hepatitis A and others. Poor drinking-water quality, sanitation, and hygiene cause up to 829 000 diarrhea deaths per year, out of which 297 000 are children under five years of age [2]. It is also vital for a swift response to environmental disasters, such as river water poisoning due to a leakage of dangerous substances, or oil spillage monitoring at seas.

Traditionally, surface water sampling is done by handpicking samples by humans, either from a shore or from a boat. This approach can be labor-intensive and expensive when dealing with large bodies of water, for example, lakes. To partially overcome these problems, some organizations involve volunteers in collecting the samples [3]. Nevertheless, many areas are inaccessible for humans or potentially harmful to their health, like flooded queries or rivers contaminated by microorganisms. The usage of drones for water sampling might solve all these problems.

1.1 Aim

The main goal of this thesis is to design a water sampling manipulator that will be mounted on a drone. Based on a review of tested solutions provided in Chapter 2, some additional aims were set:

- the drone should be able to collect more than one sample per flight to increase sampling speed

- the sampling mechanism should be able to clean itself to minimize cross-contamination
- the manipulator should be retractable for increased accessibility to smaller spaces

This manipulator will be mounted on the Tarot T650 Sport (Figure 1.1), which is one of the platforms used by MRS Group [12].

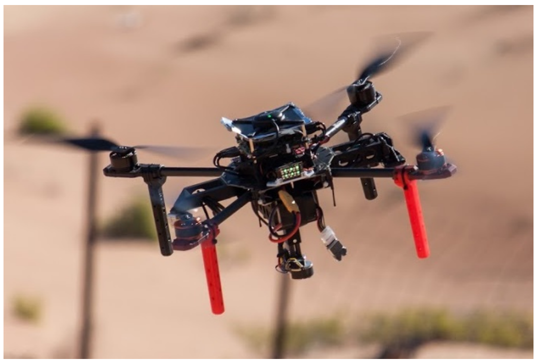


Figure 1.1: Tarot T650 Sport [13]

1.2 Outline

In Chapter 2, we will briefly discuss the currently used and researched water sampling methods, both with UAVs and other autonomous vehicles.

We then move on to our proposed solution, with the mechanical design being introduced first in Chapter 3. We will discuss the two parts composing the water sampler: the arm, mounted in front of the drone, and the sorter, mounted in the back. We will also mention the external components allowing the device to move (servomotors) and take water samples (a submersible water pump).

In Chapter 4, we will talk about the process of designing a printed circuit board (PCB), which will be responsible for the control of the water sampler. We will discuss the individual electronics components forming the PCB, especially the microcontroller unit (MCU) and the power management components.

With the hardware part behind us, we will take a look at the software in Chapter 5, starting with a brief overview of the Robot Operating System (ROS) and the water sampler node, a program used to send commands to the device from a central computer. We will then talk about the software running on the PCB’s MCU, controlling the water sampler manipulator.

In Chapter 6 we will take a look at how the water sampler functions. Before mounting the device on a drone, we will thoroughly test it in a lab, starting with the communication between the ROS node and the MCU, checking that it is capable of receiving commands and replying. We will then try out the servomotors and conduct a multiple sampling tests with the submersible water pump to asses for how long the pump should be turned on for correct sample volume to be taken. Finally, we will mount the device on the drone and check that it is functioning in flight.

In the last Chapter 7, we will recapitulate the thesis and discuss future work ideas and possible improvements.

Chapter 2
Related work

During the research of this topic, a few already tested water sampling mechanisms incorporating autonomous vehicles were found. These can be further divided into two categories:

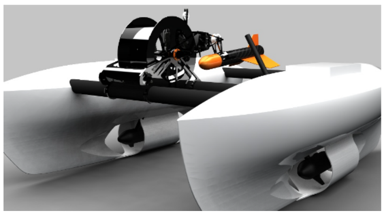
- water sampling using other autonomous vehicles
- water sampling using drones

2.1 Water sampling with using autonomous vehicles

Before drones were considered, autonomous surface or underwater vehicles (ASV or AUV) had already been used for water sampling.

An ASV introduced by Hitz et al. [4] was from the beginning designed for water quality monitoring, mainly for tracking the development of harmful cyanobacteria. It is capable of taking samples in variety of depth ranges over a long period of time. On the other hand, it weights around 120 kilograms, which makes it impractical for monitoring disconnected water bodies and impossible for one scientist to carry and set up.

EcoMapper is a commercial AUV developed by YSI [5]. It is a low cost, person-deployable solution for deep water monitoring [6], able to submerge in 100 m depth. However, it is capable of 4 knots (2.0 m/s) maximum speed, which could be time-consuming when used for sampling lakes. Also, with GPS being unavailable under water, precise navigation to a requested location could prove challenging [7].


(a) : Hitz et al. ASV [4]

(b) : EcoMapper [5]

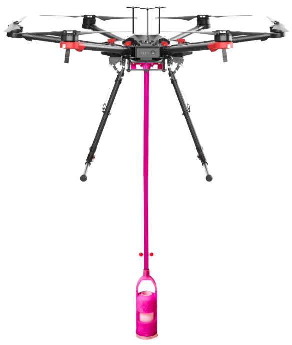
Figure 2.1: ASV and AUV water sampler examples

2.2 Water sampling using drones

In this section, a brief introduction to a few already tested solutions is going to be given.

Nixie [8], a commercial solution, is a water sampling attachment that can be mounted on an M600 drone made by DJI. It consists of a cage connected to the drone via a cable. A 500 ml bottle is put into the cage, and when a sample should be taken, the whole cage submerges in the water. However, it can only take one sample per flight, which could be tedious. Furthermore, it is non-retractable. This could be problematic in areas where a low-altitude flight is necessary.

Cengiz Koparan et al. design [9] uses a similar approach as the Nixie, but their improved version [10] can take up to three 120 ml samples per flight. Nevertheless, those samples can only be obtained from a single place.


(a) : Nixie [8]

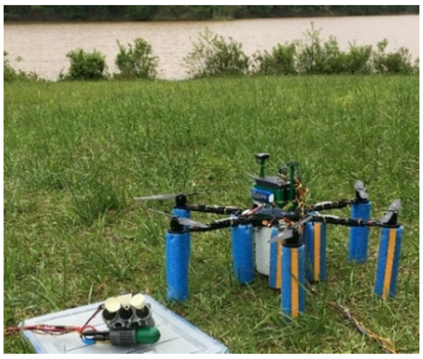
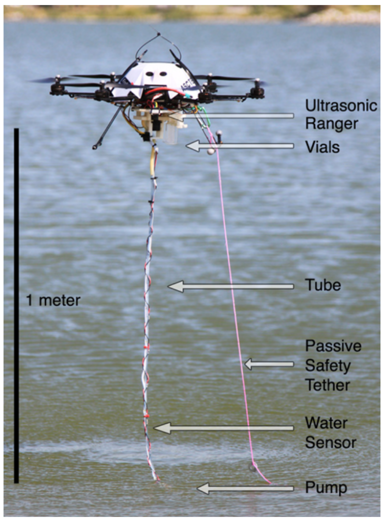
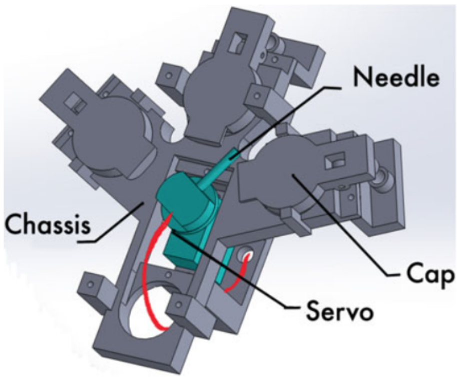

(b) : Koparan et al. [10]

Figure 2.2: Drone water sampler examples

A water sampling manipulator made by John-Paul Ore et al. [11] utilizes a different design. In the front, the system has a submersible pump connected through a tube to the back of the drone, where a water sorting mechanism with three 20 ml vials is attached. When a sample is requested, the pump is submerged, and the water is directed into one of the vials. This approach is beneficial because only one flight is necessary to obtain samples from three different locations. Furthermore, the manipulator has a self-cleaning procedure to avoid cross-contamination. Despite all of its advantages, the system is also non-retractable.



(a) : Sampler mounted on a drone



(b) : Sampler model

Figure 2.3: Ore et al. water sampler [11]

The last manipulator served as the main inspiration for our system.

Chapter 3

Mechanical design

The mechanical design of the water sampling mechanism was modeled using the Autodesk Inventor CAD software. The manipulator is composed of two main parts:

- an arm
- a sorter

Both parts will be mounted on a drone's battery holder, the arm to the front and the sorter to the back.

3.1 Arm design

The primary purpose for selecting an arm for holding the pump and taking water samples is to make the manipulator retractable, which increases safety during a flight and enables the drone to get into smaller spaces, like caves. It could potentially improve flight dynamics as well.

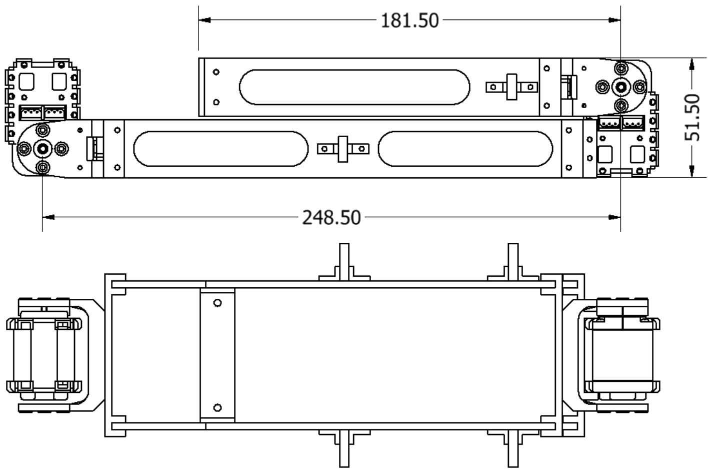


Figure 3.1: Arm design (all dimensions in mm)

The arm consists of two links and two rotary joints. Two links were used to make the arm longer, increasing the distance between the surface and the drone, which could possibly be damaged when coming into contact with water. The links were 3D printed with a Prusa i3 MK3S printer from a Polyethylene Terephthalate Glycol (PETG) filament. The joints are operated by the Dynamixel AX-12A servomotors.

3.2 Sorter design

The sorter, like the arm, was 3D printed. It consists of four cup holders arranged in a circle. Those are designed to hold typical 200 ml plastic cups, which are inexpensive and can be easily exchanged, permitting multiple samples to be taken. Each cup holder has a cut-out section, which permits the cup to be exchanged even when obstructed from the top. In the middle of the circle there is the third Dynamixel AX-12A servo. On this motor, a socket for the tube is mounted. It is slightly bent at the end to direct the tube into a cup. The servomotor then rotates the tube, directing the water pumped by the submersible pump into an appropriate plastic cup. There is also a cut-off section at the top of the circle. Its purpose is to allow the manipulator to flush its contents before taking a sample, reducing the possibility of cross-contamination.

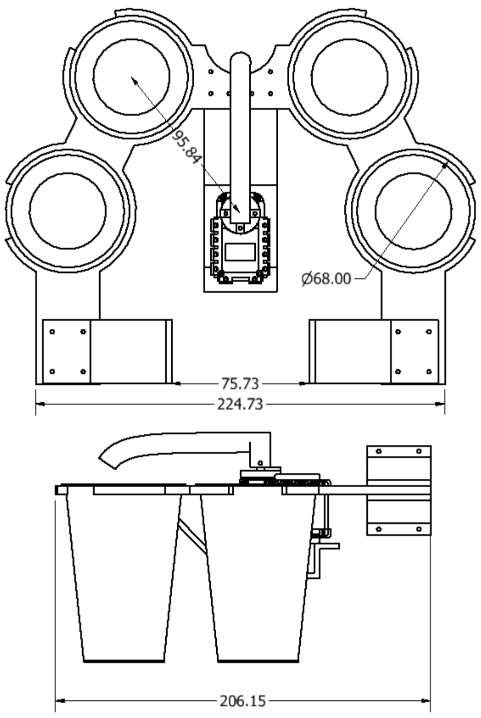


Figure 3.2: Sorter design (all dimensions in mm)

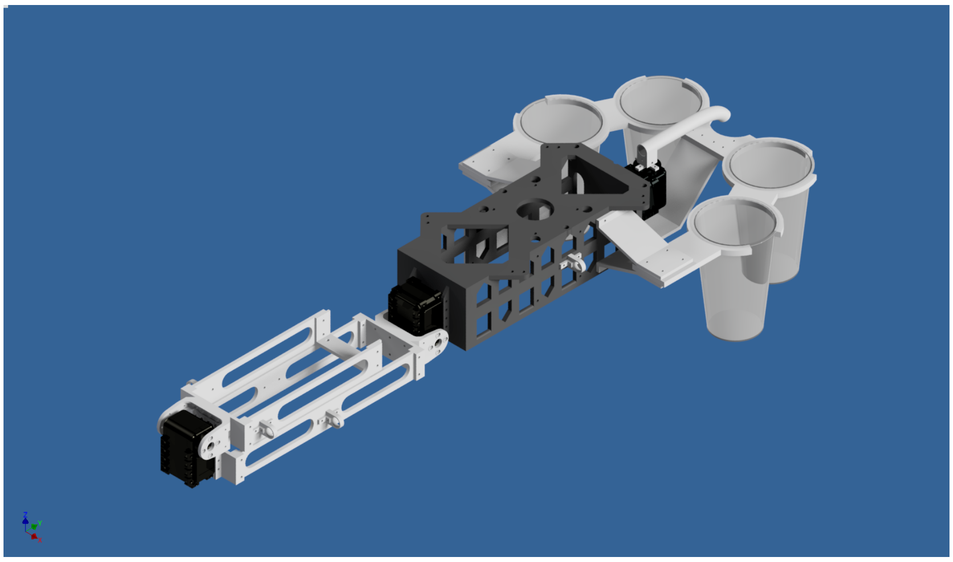


Figure 3.3: Water sampler render

3.3 Used components

3.3.1 Submersible pump

Some key aspects were taken into consideration during the selection of a submersible pump

- it should operate on 12 V for voltage compatibility with servomotors, allowing both to be powered from a single voltage source, saving space on the control board and a manufacturing cost
- it should be low-powered to increase battery life, but also powerful enough to draw water to at least 1-meter height, making certain that the samples will be taken
- it should be lightweight for the servomotors to hold it in retracted state without too much of an effort, resulting in a decrease in a current drawn by the device

After a market research, the pump in Figure 3.4 with the following specifications (Table 3.1) was selected:

Figure 3.4: Submersible pump [14]

Specification	Value
Rated voltage	12 [V]
Rated current	350 [mA]
Power consumption	4.2 [W]
Flow rate	120 [l/h]
Transportation height	3 [m]
Operating temperature	0 - 75 [°C]
Weight	94 [g]
Liquid	water, oil, petrol, acid and base solutions

Table 3.1: Submersible pump specifications [14]

3.3.2 Dynamixel AX-12A

DYNAMIXEL AX-12A (Figure 3.5) is a smart servomotor developed by ROBOTIS. It is a widely used actuator in many robotic applications, ranging from robotic arms to complex systems like six-legged robots.

Specifications

Figure 3.5: DYNAMIXEL AX12-A [15]

Specification	Value
Input voltage	9.0 – 12.0 [V]
Stall current	1.5 [A]
Resolution	0.293 [°]
Running degree	0 – 300 [°]
Operating temperature	-5 - 75 [°C]
Weight	54.6 [g]
Communication	Half duplex Asynchronous Serial Communication
Protocol	DYNAMIXEL Protocol 1.0

Table 3.2: DYNAMIXEL AX-12A specifications [15]

Communication protocol

The main advantage of DYNAMIXEL AX-12A is that, unlike many other servomotors, which are externally controlled through a pulse width modulation (PWM), these are operated using a custom serial protocol called DYNAMIXEL Protocol 1.0 [16]. The inner circuitry receives commands sent by a microcontroller over a half-duplex universal asynchronous receiver-transmitter (UART) interface and then converts them to control actions. This approach also allows connecting multiple of these servos to one bus and control them independently, saving microcontroller pin connectors.

The packets used in DYNAMIXEL Protocol 1.0 have the following structure (Table 3.3):

Header1	Header2	Packet ID	Length	Instruction	Param 1	...	Param N	Checksum
0xFF	0xFF	Packet ID	Length	Instruction	Param 1	...	Param N	CHKSUM

Table 3.3: DYNAMIXEL protocol 1.0 packet [16]

where Packet ID is the ID of the servomotor (0x00 – 0xFD) or a broadcast address (0xFE), Length of the packet is calculated as

$$\text{Length} = \text{number of Parameters} + 2, \tag{3.1}$$

Instruction tells the inner circuitry what it should do (e.g., Read/Write to Control Table, Factory Reset etc.), Param 1 ... Param N are the instructions additional data and Checksum is calculated as

$$\text{Checksum} = \sim (\text{ID} + \text{Length} + \text{Instruction} + \text{Parameter1} + \ldots + \text{ParameterN}), \tag{3.2}$$

where $\sim$ is a bitwise negation.

An example of a packet sent over a UART can be found in Figure 3.6.

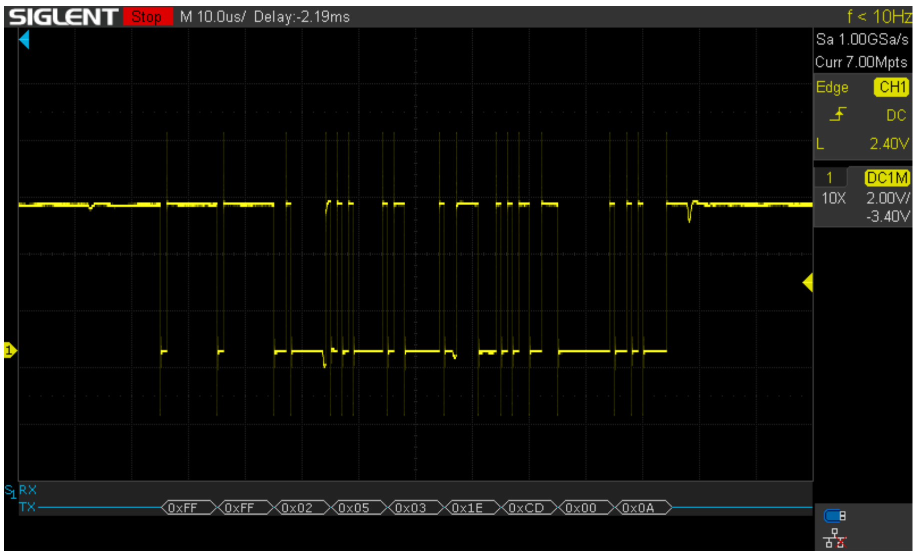


Figure 3.6: Example of a write instruction packet

Chapter 4

PCB design

This chapter is going to describe the design of a PCB we developed for controlling the water sampler. For the proper function of the device, it had to comprise several essential components:

- an MCU responsible for managing peripherals and communication with the UAV on-board computer
- a Universal Serial Bus (USB) connector for communication with the main computer
- various other connectors for programming/debugging the MCU, communication with the servomotors, and others
- a MOS transistor for switching on and off the water pump
- a 5 V to 3.3 V regulator for a stable power supply for the MCU and other integrated circuits (ICs)
- a 22 V (Battery voltage) to 12 V power management IC for powering the servomotors and the water pump

In the following parts of this chapter, the components selected to satisfy those requirements are going to be discussed.

4.1 Microcontroller unit

The central part of this circuit board is the MCU. As stated before, it is used for the control of the water sampler and the communication with the main computer. For this application, we selected the STM32F042K6T6. It is an entry-level, reasonably priced MCU with a USB peripheral needed to receive commands from the onboard computer. It is capable of a single-wire UART communication, which is used to send the instruction packets to the servomotors.

According to the datasheet [17], some external components were needed to achieve a proper function, mainly decoupling capacitors to bypass the power supply, an external reset button with a capacitor to minimize parasitic resets, and a boot pin (PB8) pull-down resistor. Moreover, an inductor was added before the analog power supply pin, which, combined with the bypass capacitors, works as an LC low-pass filter, smoothing the voltage spikes when the MCU circuitry is switched.

Although the MCU is capable of a crystal-less USB operation (i.e., USB can run from an internal RC oscillator, and an external crystal oscillator is not required), crystal oscillators generally have better accuracy over a temperature and supply voltage range. Because of that, an external 24 MHz crystal oscillator was chosen. An external capacitors' value needed to set load capacitance can be calculated using Equation 4.1

$$C_L = \frac{C_1 \cdot C_2}{C_1 + C_2} + C_s, \tag{4.1}$$

where C_L is the load capacitance (specified in datasheet [18], 10 pF for the used oscillator), C_s is stray capacitance (generally $\approx 5\,\mathrm{pF}$), C_1 and C_2 are external capacitors' capacitance [19]. Assuming $C_1 = C_2$, we can modify the Equation 4.1

$$\begin{aligned} \frac{C_1 \cdot C_2}{C_1 + C_2} + C_s &= C_L && (4.2)\\ \frac{C_1^2}{2C_1} + C_s &= C_L && (4.3)\\ \frac{1}{2}C_1 &= C_L - C_s && (4.4)\\ C_1 &= 2 \cdot (C_L - C_s) && (4.5)\\ C_1 &= 2 \cdot (10 - 5)\ \mathrm{pF} = 10\,\mathrm{pF} && (4.6) \end{aligned}$$

An external resistor for the oscillator is also included. Its purpose is to limit the drive level of the crystal oscillator [19]. The resistor's value can be calculated using Equation 4.7

$$R_{ext} = \frac{1}{2\pi F C_2}, \tag{4.7}$$

where F is oscillation frequency. By substituting the variables with given values we get 4.8

$$R_{ext} = \frac{1}{2\pi F C_2} = \frac{1}{2\pi \cdot 24 \cdot 10^6 \cdot 10 \cdot 10^{-12}} \approx 663\,\Omega \tag{4.8}$$

Last, we added a blue LED for status logging and debugging purposes.

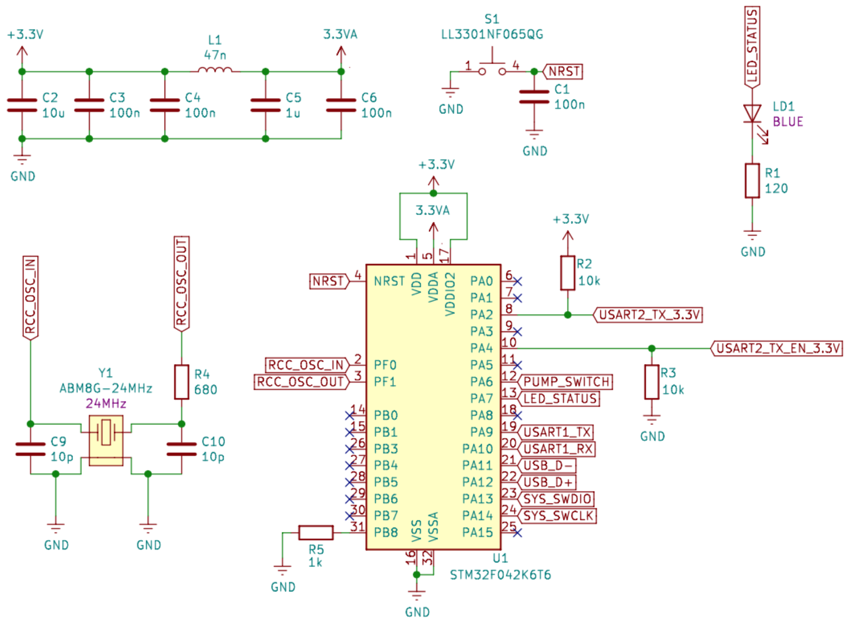


Figure 4.1: MCU schematic diagram

Pin	Function	Description
PA2	USART2 TX	servomotors control
PA4	USART2 TX EN	switch between transfer and recieve
PA6	PUMP SWITCH	turn on/off the water pump
PA7	LED STATUS	status and debugging LED
PA9	USART1 TX	not used, backup
PA10	USART1 RX	not used, backup
PA11	USB D-	USB negative differential line
PA12	USB D+	USB positive differential line
PA13	SWDIO	programming/debugging data transfer
PA14	SWCLK	programming/debugging clock
PB8	BOOT	pulled down for booting from flash memory
PF0	OSC IN	high speed crystal oscillator input
PF1	OSC OUT	high speed crystal oscillator output
NRST	NRST	reset button input

Table 4.1: MCU pinout

4.2 Power management

4.2.1 5 V to 3.3 V

Because the USB is needed for communication with the central computer and, therefore, is already included, it can also be used to power the control board's MCU. A USB 2.0 standard power line has 5 V voltage and can deliver up to 500 mA of a current, which is more than sufficient. However, the input voltage for our MCU is 3.3 V, so it needs to be stepped down. For this purpose, an AZ1117IH-3.3TRG1 [20] low-dropout linear voltage regulator is used because the control circuitry of the PCB does not draw a lot of current, and therefore, the voltage regulator will generate only a minor amount of heat. The schematics can be found in Figure 4.2.

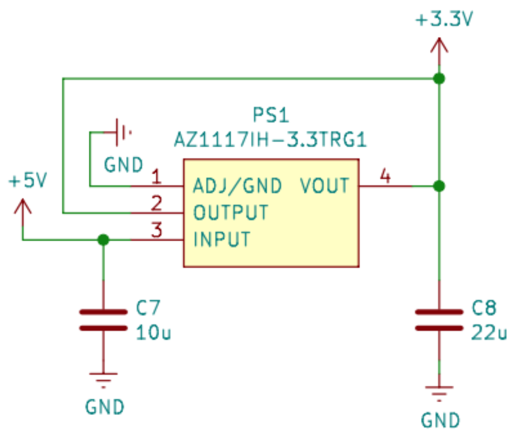


Figure 4.2: Linear voltage regulator schematic diagram

4.2.2 Battery voltage to 12 V

Unlike the control circuitry, the servomotors and the water pump are rated for a 12 V input voltage and draw a considerable amount of a current. A single AX-12A servo can take up to 1.5 A of a stall current (a maximal current when the load is too heavy or an armature is prevented from rotating), and the pump can draw up to 350 mA. Hence, a linear voltage regulator could not be used because it would require external cooling.

For this purpose, an LMZM33606RLXR [21] power module is used, a switching buck converter capable of delivering up to 6 A of a current. This is ample to drive both the servomotors and the water pump. This particular step-down converter is also convenient because it only needs a few external components for correct function, reducing the space required on the PCB. The complete schematic diagram can be found in Figure 4.3.

First, feedback resistors needed to be selected to set the output voltage. The resistor values were calculated using Equation 4.9

$$R_{FBT} = 10 \cdot (V_{out} - V_{FB})\ k\Omega, \tag{4.9}$$

where R_{FBT} is high-side feedback resistor's resistence, V_{out} is a desired output voltage, and V_{FB} is a feedback voltage (typically $1.006\,V$). Substituting our values for the variables we get

$$R_{FBT} = 10 \cdot (12 - 1.006) = 109.96\,k\Omega. \tag{4.10}$$

The recommended value for the low-side resistor is $R_{FBB} =\ 10\,k\Omega$.

Even though the LMZM33606 is internally compensated for being stable over the operating range, adding a feed-forward capacitor can improve its performance. The value of C_{FF} was calculated using Equation 4.11

$$C_{FF} = 4.3 \cdot \frac{V_{out} \cdot C_{out}}{R_{FBT}}\ pF, \tag{4.11}$$

where C_{out} is output capacitance in μF, which, in our case, is $3 \cdot 47 = 141\,\mu$F (minimum required output capacitance for 12 V voltage is 100 μF), and R_{FBT} is in kΩ. Substituting given values for the variables gives us

$$C_{FF} = 4.3 \cdot \frac{12 \cdot 141}{110} \approx 66\,pF. \tag{4.12}$$

A resistor is connected to the RT pin to set the switching frequency. A 47.5 kΩ resistor was chosen, which corresponds to the switching frequency of 800 kHz.

The LMZM33606 buck converter has an internal low-dropout linear voltage regulator powering the control circuitry. Usually, the regulator is powered by the input voltage, resulting in unnecessary heat generation for larger input voltages. An external bias voltage can be applied to the *BIAS_SEL* pin to decrease the power loss. Thus, a 5 V bias voltage from the USB is provided.

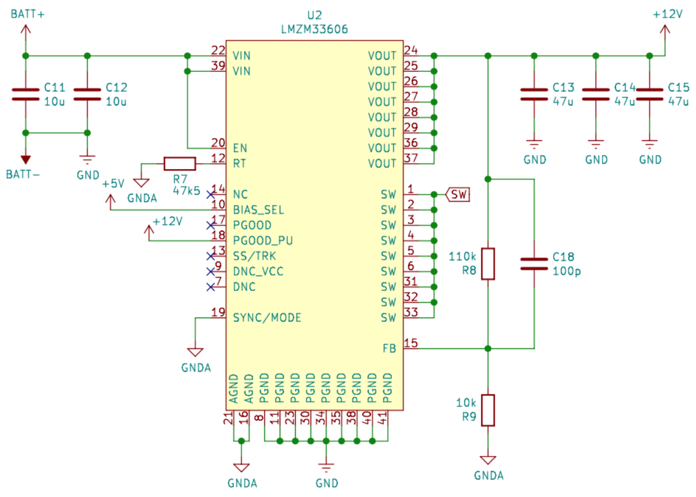
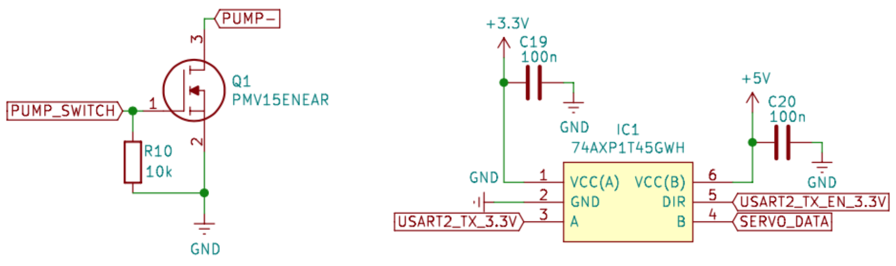


Figure 4.3: Buck converter schematic diagram

4.3 Pump switch and voltage level translating transceiver

To turn the pump on and off with the MCU, a PMV15ENEAR [22] is included, a logic-level N-channel MOS transistor with electrostatic discharge (ESD) protection, capable of driving up to 30 V/6.2 A loads with a low 20 mΩ resistance when switched on. To keep the MOSFET gate low during startup and MCU resets, a 10 kΩ pull-down resistor was added.

Because the control circuitry of servomotors runs on a 5 V voltage, a 74AXP1T45GWH [23], a 3.3 V to 5 V voltage level translating transceiver with direction control is included. However, it is unnecessary as the receiver of the servomotors functions on a TTL logic level, with a low signal defined as a voltage between 0 V and 0.8 V and a high signal defined as a voltage between 2 V and 5 V.



(a) : Pump switch

(b) : Voltage level translating transceiver

Figure 4.4: Pump switch and voltage level translating transceiver schematics

4.4 USB and other connectors

As stated before, the MCU needs to communicate with the central computer through the USB 2.0 protocol. A mini USB type B connector was selected since it is smaller than a USB type A connector, saving space on the PCB, and it is more durable than micro USB connectors.

Because the MCU is vulnerable to ESD and USB connector could potentially be exposed to a static discharge when connecting/disconnecting the USB, an IP4234CZ6 [24] integrated circuit was included, which contains TVS diodes designed to redirect static discharges.

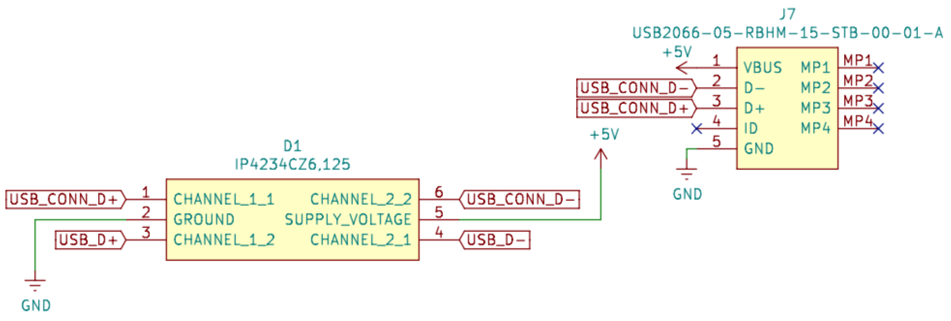


Figure 4.5: USB connector and ESD protection schematic diagram

As for the other connectors, it is necessary to connect the battery power supply, the servomotors, the water pump, and the serial wire debug (SWD) interface to the PCB.

The battery power supply uses an XT30 connector, with the XT series being used frequently on the MRS drones.

The servomotors are designed for a MOLEX Mini-SPOX 22035035 3 Pin Wire-to-Board connectors, which have a similar footprint to standard 2.5 mm header pins. Pin 1 is connected to the ground, pin 2 is connected to the 12 V supply voltage, and pin 3 is connected to the DATA line. 10 μF and 100 nF decoupling capacitors were added to increase input voltage stability, as used in an examplery circuit figure in [15]. The DATA line is driven high to 5 V with a 10 kΩ pull-up resistor on the 5 V side of the voltage level translating transceiver and to 3.3 V on the 3.3 V side.

The water pump is connected to the PCB through standard header connectors with DuPont connectors. Because the pump contains a DC motor, which is an inductive load, a flyback diode was added to suppress voltage spikes when it is switched off.

The SWD is a debugging interface developed by Arm Ltd. as an alternative to the Joint Test Action Group (JTAG) interface. It is a two-pin interface, using the SWDIO as the data transfer line and the SWDCLK as the clock line. The programmer also needs a 3.3 V reference voltage input, the ground reference, and optionally a reset input. All those are connected to the programmer through standard header pins and DuPont connectors.

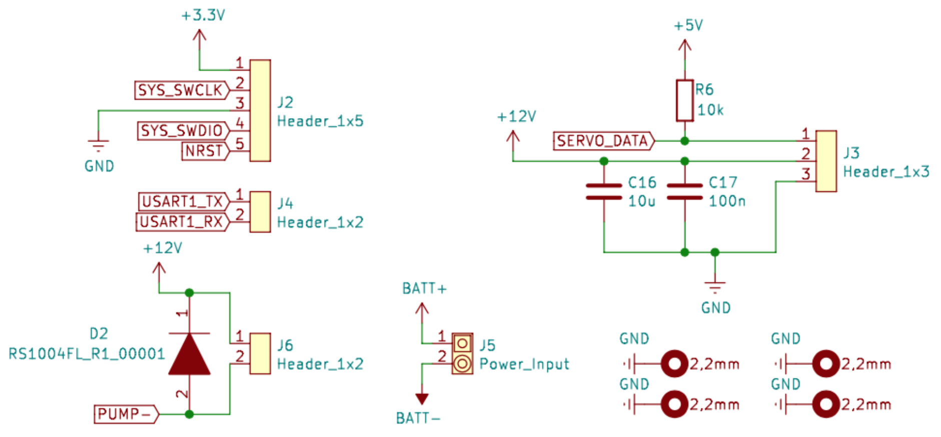


Figure 4.6: Various connectors schematic diagram

4.5 PCB layout

All the components were fitted on a 2-layer board to minimize manufacturing costs. The front side of the PCB contains all of the electronics components and most of the routing, with as few as possible routes on the bottom part of the the MCU. Both sides are filled with a ground copper pour where no routing is presented. The decoupling capacitors were put as close as possible to their respective pins. The high-speed components, like a USB and crystal oscillator, near the MCU to minimize trace length, and therefore the signal interference. Most signal routes are 0.3 mm thick. The 3.3 V and 5 V power lines are 0.5 mm thick. Since the 12 V power line will pass a high amount of a current, its traces were made 3 mm thick to decrease the PCB heating.

The PCB was designed using the KiCad software, a free electronics schematic editor, and a PCB layout editor. The designed files were then exported and sent to PCBWay, a Chinese company responsible for PCB manufacturing. The final dimensions were 54.5 mm wide and 50 mm long. The routing is shown in Figure 4.7 (top layer) and Figure 4.8 (bottom layer).

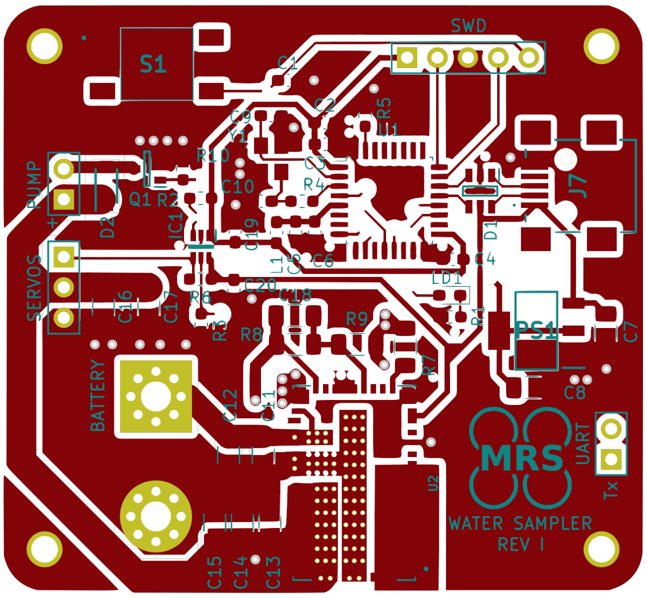


Figure 4.7: Top layer routing

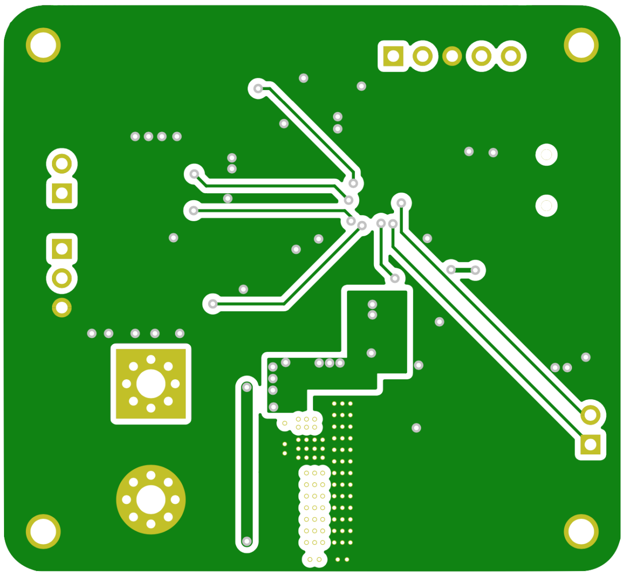


Figure 4.8: Bottom layer routing

The board was assembled by hand. Firstly, a soldering paste was applied using a stencil. Secondly, components were placed to their corresponding footprints and soldered to the PCB using an infrared reflow oven. Finally, inadequate connections were reinforced with a hot-air soldering station and a soldering iron. The assembled board can be found in Figure 4.9.

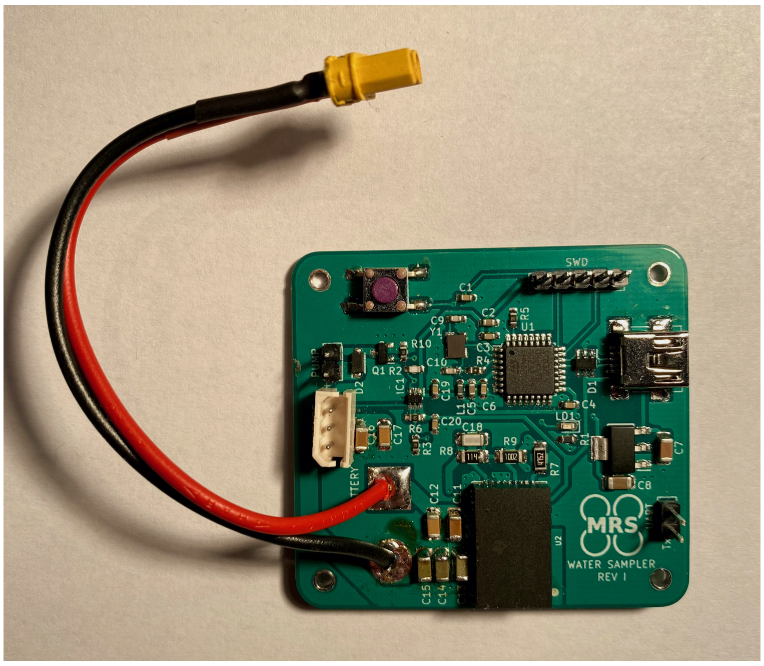


Figure 4.9: Assembled PCB

Chapter 5
Software

Up until now, we have been talking about the hardware part of the water sampler. We are going move on to the software. Since the central computer of the UAV used by MRS runs ROS [25], two separate programs were made:

- a ROS program which establishes communication between the central computer and the water sampler
- a water sampler control board program, which interprets the messages sent by ROS and gives commands to the peripherals

First, the ROS program is going to be discussed.

5.1 ROS Water Sampler program

5.1.1 ROS overview

ROS is an open-source collection of software libraries and tools curated by Open Robotics, a non-profit organization headquartered in Mountain View, California. Aimed at both commercial and research robotics applications, ROS has been proven by numerous organizations over its more than ten years of development, helping create billions of dollars in value [26].

5.1.2 ROS Package

A ROS Package is a collection of executables (called nodes), configuration files, datasets, and other useful files. Their primary purpose is to organize everything into easy-to-use bundles, making them reusable without a lot of effort.

ROS Master

The central part of the Robot Operating System is the ROS Master [27], a program providing naming and other services for other ROS nodes. The Master keeps track of publishers and subscribers to topics, enabling nodes to locate and communicate with each other.

ROS Node

A ROS Node [28] is a program written in either C++ or Python, serving as the basis for computation in the ROS environment. Individual nodes take care of a simple task, e.g., one node can control a motor while another controls an ultrasonic range sensor or a gyroscope. When connecting them, we can get a complete robot system, like a UAV.

ROS Publishers, Subscribers, Topics and Messages

Nodes communicate with each other using a structure of publishers and subscribers. When a node wants to send data it has computed, it does it by publishing a message [29] (a data structure containing typed field defined in msg files) to a topic [30], which is a named bus used for exchanging messages. Any other node which wants to receive the data must be subscribed to the same topic, and the message will be forwarded to it.

5.1.3 ROS Water Sampler node

The ROS Water Sampler node is a program written in Python used to send commands to the water sampler manipulator. It creates two custom topics, `/water_sampler/input` and `/water_sampler/output`. The input topic is used to send commands to the water sampler, providing a control interface for other nodes, which makes it easily extendable with external control logic, e.g. a neural network. To test the functionality, a node reading keyboard inputs from a user was created. The node converts the inputs to commands and publishes them to the `/water_sampler/input` topic, which are then directed to the water sampler node. If it is evaluated by the water sampler node as a correct command, the node creates a message with data required for the MCU to interpret it and publishes it to a `/uav_name/serial/send_message` topic.

This topic is provided by *mrs_serial* [31], a node that sends and receives messages from devices connected to serial lines like USB and UART. It works with its custom Baca Protocol messages with the structure defined in Table 5.1

Header	Payload size	Payload 0 (Message ID)	Payload 1	...	Payload N	Checksum
'b'	payload_size	payload_0	payload_1	...	payload_n	checksum

Table 5.1: Baca Protocol

where checksum is the sum of all previous bytes.

The water sampler node only creates the Payload part of the message, and the rest is filled by the *mrs_serial*. The water sample MCU accepts the messages defined in Table 5.2.

Header	Payload size	Payload 0 (Message ID)	Payload 1	Checksum	Description
0x62	0x02	0xAA	00	0x0E	retract arm
0x62	0x02	0xAA	01	0x0F	extend arm
0x62	0x02	0xAA	02	0x10	take sample
0x62	0x02	0xAA	03	0x11	reset samples

Table 5.2: Commands accepted by the Water Sampler MCU

5.2 STM32 Water Sampler program

The MCU on the water sampler control board is programmed using C. The initial code, which sets up the necessary peripherals needed to communicate with the UAV's central computer, servomotors, and to control the pump, is provided by using STM32CubeMX, an initialization code generator providing an easy-to-use configuration of the STMicroelectronics (STM) MCUs. It is part of the integrated development environment (IDE) STM32CubeIDE.

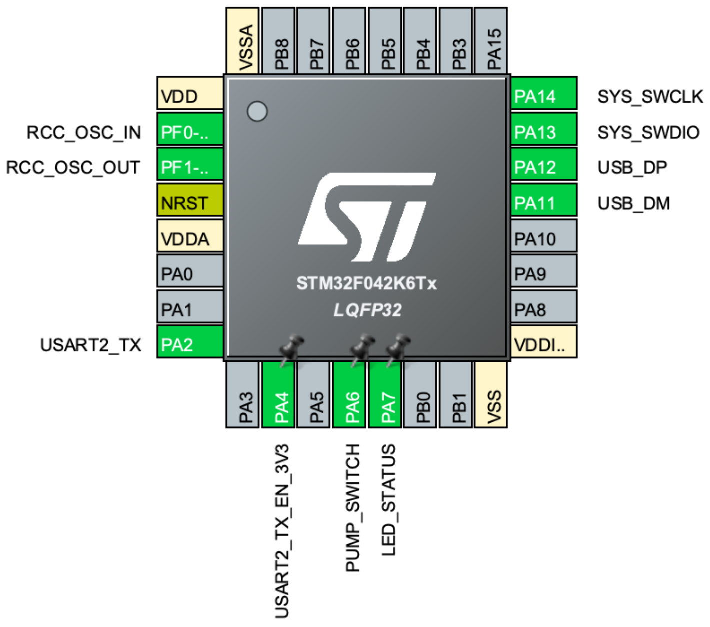


Figure 5.1: CUBEMX pin initialization

When the MCU starts, all the peripherals being used are initialized. Then the servomotors receive commands to decrease their speed and return the arm to the retracted position. After that, the program moves into a loop and waits for commands sent over the USB. If a packet is received, the MCU checks if it is a valid command. If everything is correct, the command is executed. Otherwise, MCU replies with a *bad_command_received* packet.

Whenever ROS requests a sample to be taken, the MCU first examines if the arm is retracted. If it is, then it checks whether or not the sorter has enough free cups for samples. Provided that it does, it first flushes the tube to minimize cross-contamination, and then it takes the sample and sends a packet containing information about how many cups free for samples there are. To minimize movement of the center of mass, it first takes samples into the two innermost cups and then moves on to the outer cups.

For every command, MCU sends out a corresponding reaction after finishing the action. The overview of all the MCU replies can be found in Table 5.3.

Header	Payload size	Payload 0 (Message ID)	Payload 1	Checksum	Description
0x62	0x02	0xAA	0x00-0x03	0x0E-0x11	number of free samples remaining
0x62	0x02	0xAA	0x0A	0x18	arm retracted
0x62	0x02	0xAA	0x0B	0x19	arm extanded
0x62	0x02	0xAA	0x0C	0x1A	sample taken
0x62	0x02	0xAA	0x0D	0x1B	sample reset
0x62	0x02	0xAA	0x0E	0x1C	sampler full
0x62	0x02	0xAA	0x0F	0x1D	arm not extended
0x62	0x02	0xAA	0x10	0x1E	bad command recieved

Table 5.3: MCU replies

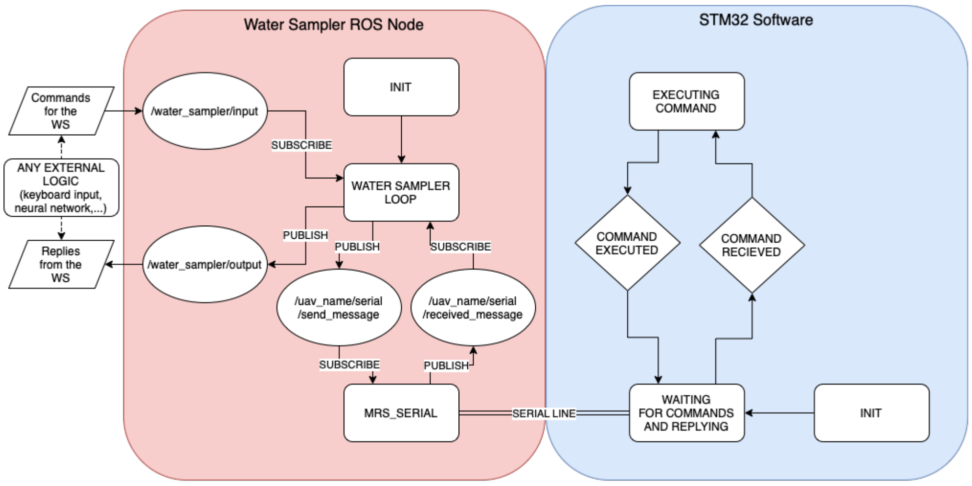


Figure 5.2: Software diagram

Chapter 6
Evaluation

After the mechanical design finished, the PCB assembled, and the software programmed, the water sampler was ready to be assessed. Firstly, the device was thoroughly tested in a lab environment to verify that every part behaves in the expected manner.

6.1 Lab testing

Firstly, a test was set up to verify that the MCU could correctly receive messages from the ROS node. To verify that a command was decoded successfully, a simple script was written to turn the LED on when MCU received a message and off when it replied. The response was then echoed in a terminal on the computer with the ROS node, verifying that communication was successfully established.

Then we moved on to the servomotors, checking that the arm and the sorter move to the desired positions. We found out that the servos on the arm were moving too quickly, which could unnecessarily stress the UAV altitude controller, so we decided to decrease their speed during initialization. After verifying that everything worked as anticipated, we began to test the pump.

Because the water sampler does not contain any sensor for measuring the sample volume, we decided to create a time-volume characteristic by taking eight samples for each half a second spaced value, starting with 1.5 seconds (which was the bare minimum for the water to flow through the whole system and create a reasonably sized sample) and ending with 5.5 seconds, which was enough to fill most of the cup. We used a granulated cylinder with a 250 ml scale and 5 ml divisions to measure the sample volume. The measured quantities can be found in Table 6.1.

Time [s]								
1.5	**2**	**2.5**	**3**	**3.5**	**4**	**4.5**	**5**	**5.5**
25	40	60	80	105	125	145	165	185
25	40	60	80	100	125	145	165	185
25	40	60	80	105	125	145	165	185
20	35	55	70	95	120	140	160	180
25	40	60	80	105	125	145	165	185
25	40	60	80	105	125	145	165	185
25	40	60	80	105	125	145	165	185
20	35	55	75	100	115	140	160	175

Table 6.1: Measured sample volume (in ml)

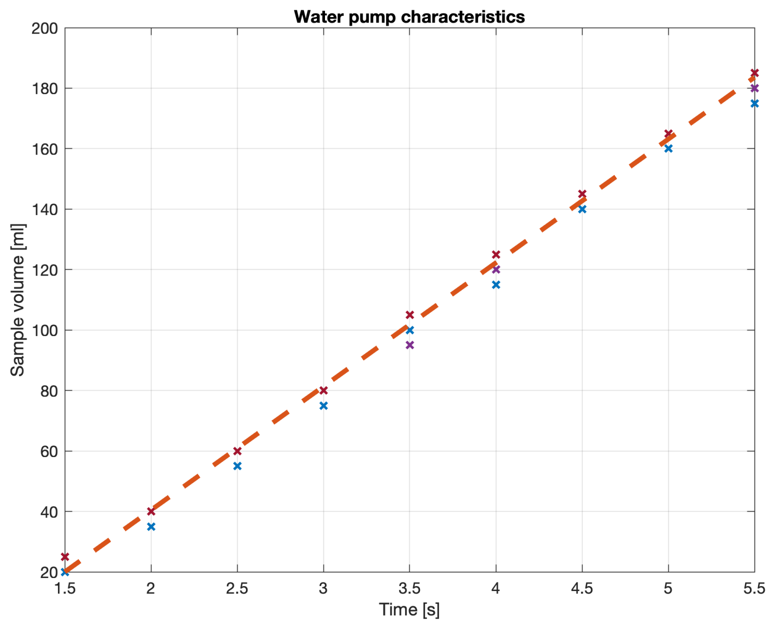


Figure 6.1: Water pump characteristics

During the test it was found out that the water pump delivered a consistent sample volume, with every fourth being around 5 ml to 10 ml smaller than the others. This is because the outer right cup gets filled every fourth sample. In this position, the tube is being bent extensively, decreasing the tubing size, hence creating resistance for the water flow.

The last experiment concluded the lab testing and the water sampler was ready to be mounted on a drone.

6.2 Testing with a drone

Due to weather unsuitable for outdoor tests, the water sampler was tested indoors. After mounting the device on a drone, it was first verified that cabling did not obstruct the movement of the socket mounted on the servomotor of the sorter. A series of tests was run to confirm that the ROS node on the UAV worked correctly and the MCU was receiving commands as expected. All the parts moved correctly and would not hinder the flight of the drone. A few samples were taken to assess the function of the pump.

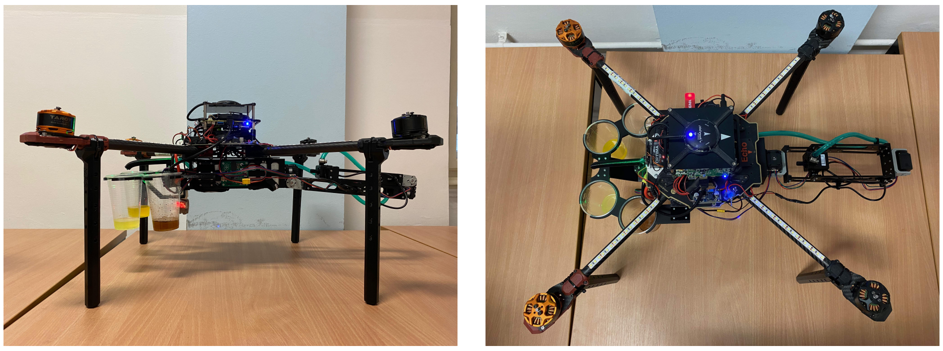

(a) : Side view **(b) :** Top view

Figure 6.2: Drone with mounted sampler

Moving on, the water sampler was ready for a test flight. After a lift-off the arm movements were tried out. During this test, it was found out that even though the speed of servomotors was decreased during programming, the arm still had a noticeable impact on the controller of the drone, so the speed was further decreased afterwards. Finally, a test where a sample was taken was conducted and experiment pass successfully. During the flight, the samples were kept inside the cup without any leak. However, during the lift-off, a bit of the outer samples were spilled by the air passing through the starting propeller.

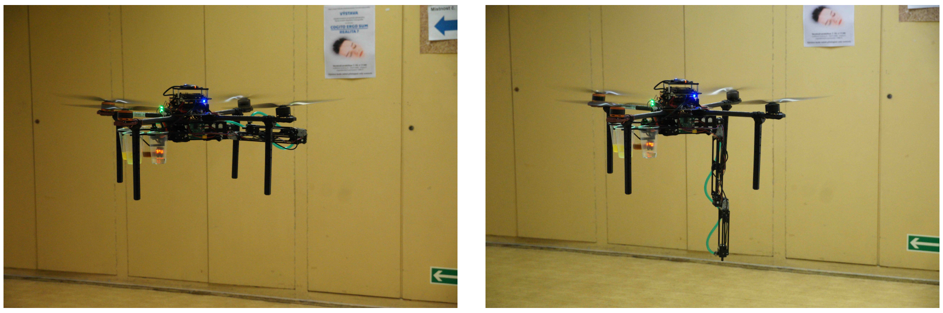

(a) : Arm retracted **(b) :** Arm extanded

Figure 6.3: In-flight arm test

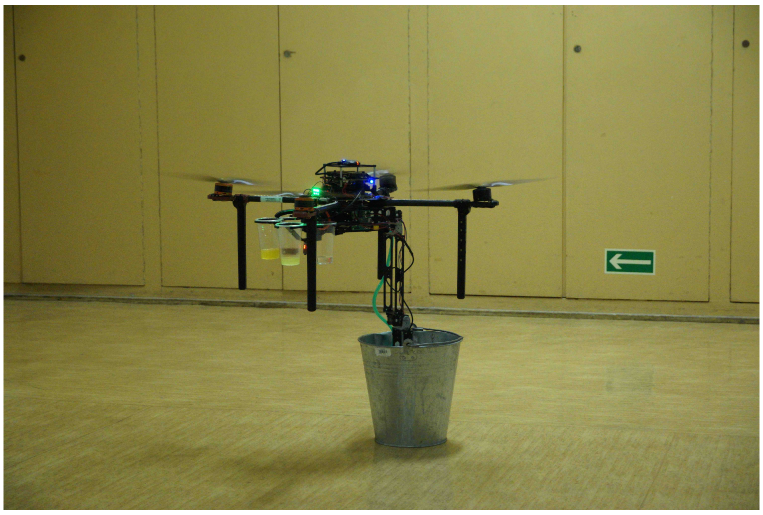


Figure 6.4: In-flight sampling

The water sampler functionality is going to be further tested outdoors during a MRS camp dedicated to testing of UAVs in Temešvár, located on the shore of the Orlík Reservoir.

Chapter 7
Conclusion

In this thesis, a water sampling manipulator for a Tarot T650 drone was developed. To conclude the thesis, a brief summary of the objectives and their solutions is provided in this chapter, with a few future ideas and improvements discussed at the end.

1. **Design the mechanical solution.**

   The mechanical design of the presented water sampling manipulator consists of two main parts: the arm, mounted at the front of the drone, and the sorter, mounted at the back. Both are attached to the battery holder of the drone. The arm is comprised of two rotary joints, allowing the device to be retractable, which permits the drone to access smaller spaces and increases safety during flights. It holds a submersible water pump, which pumps the water along the arm and battery holder to the sorter. The sorter consists of four cup holders for 200 ml plastic cups, allowing the drone to take up to four samples per flight. A servomotor in the middle directs the water flow to one of those cups. The design of the manipulator was influenced by a water sampler created by Ore et al. [11].

2. **Design the electronic control unit.**

   a. **Select appropriate parts for constructing the water sampler.**
   b. **Design a suitable electrical circuitry for the control.**
   c. **Use a STM32 family microcontroller to control the device and communicate with the onboard computer.**

   For the electronic part, a custom PCB was created. Its main part is the MCU, which is responsible for controlling the whole device. The STM32F042K6T6 was selected, an entry-level microcontroller developed by STMicroelectronics, which supports the USB 2.0 protocol necessary for establishing communication with the UAV's central computer. To power the three servomotors and the water pump, a powerful enough step-down regulator from battery voltage to 12 V was needed. For this purpose, the LMZM33606 buck converter was selected. A 5 V to 3.3 V linear regulator was used, responsible for powering the MCU. The board also contains various other connectors to properly interface the servomotors and the submersible water pump.

3. **Implement a program for the onboard computer of the UAV to control the microcontroller via USB. This program must be implemented using ROS for use with the MRS Group system.**

   To control the water sampling manipulator from a UAV's central computer, a ROS node was programmed, which subscribes and publishes to a custom topic. This allows the water sampler to be easily controlled by any external logic, e.g. a neural network, with only requirement constisted in publishing the correct commands to the `/water_sampler/input` topic. Those are then modified by the water sampler node and sent over the serial line using the *mrs_serial* node by publishing to its topic.

4. **Test the functionality of the device.**

   After assembling the manipulator, a series of tests in a lab environment was run before mounting the device on the drone. First, the USB communication between the central computer and the ROS node was tried. With MCU successfully responding to commands sent by ROS, the movement of all servomotors was checked. Furthermore, the submersible water pump was tested to accurately set the time needed for the sample to reach the desired volume. Finally, the water sampler was mounted on a Tarot 650 drone and test samples were taken in-flight.

7.1 Future work and improvement ideas

Currently, the water sampler does not possess any sensor measuring whether or not the water pump is fully submerged. The system depends on the sensors of the UAV, which could result in permanent damage to the pump in case of failure.

Adding a cover over the cups could also be beneficial to avoid spillage of water samples from the cups and also allowing the UAV to take larger samples.

Bibliography

[1] GIONES, Ferran a Alexander BREM. From toys to tools: The co-evolution of technological and entrepreneurial developments in the drone industry. Business Horizons. 2017, 60(6), 875-884. ISSN 00076813. Available from: doi:10.1016/j.bushor.2017.08.001

[2] Drinking-water [online]. Geneva, Switzerland: World Health Organization, 2019 [cit. 2021-12-08]. Available from: https://www.who.int/news-room/fact-sheets/detail/drinking-water

[3] PETERS, C.B., Y. ZHAN, M.W. SCHWARTZ, L. GODOY a H.L. BALLARD. Trusting land to volunteers: How and why land trusts involve volunteers in ecological monitoring. Biological Conservation. 2017, 208, 48-54. ISSN 00063207. Available from: doi:10.1016/j.biocon.2016.08.029

[4] HITZ, Gregory, Francois POMERLEAU, Marie-Eve GARNEAU, Cedric PRADALIER, Thomas POSCH, Jakob PERNTHALER a Ronald SIEGWART. Autonomous Inland Water Monitoring: Design and Application of a Surface Vessel. 2012, 19(1), 62-72. ISSN 1070-9932. Available from: doi:10.1109/MRA.2011.2181771

[5] EcoMapper. YSI [online]. Yellow Spring, Ohio: YSI, 2021 [cit. 2022-01-02]. Available from: https://www.ysi.com/ecomapper

[6] ELLISON, R. a D. SLOCUM. High spatial resolution mapping of water quality and bathymetry with a person-deployable, low cost autonomous underwater vehicle. OCEANS 2008. IEEE, 2008, 2008, , 1-7. ISBN 978-1-4244-2619-5. Available from: doi:10.1109/OCEANS.2008.5151978

[7] LEONARD, John J. a Alexander BAHR. Autonomous Underwater Vehicle Navigation. Springer Handbook of Ocean Engineering. Cham: Springer International Publishing, 2016, 2016, , 341-358. ISBN 978-3-319-16648-3. Available from: doi:10.1007/978-3-319-16649-0_14

[8] Nixie: Changing Water Sampling Worldwide [online]. New York: Reign Maker, 2021 [cit. 2021-12-06]. Available from: https://www.nixiedip.com

[9] KOPARAN, Cengiz, Ali KOC, Charles PRIVETTE, Calvin SAWYER a Julia SHARP. Evaluation of a UAV-Assisted Autonomous Water Sampling. Water. 2018, 10(5). ISSN 2073-4441. Available from: doi:10.3390/w10050655

[10] KOPARAN, Cengiz, A. Bulent KOC, Charles V. PRIVETTE, Calvin B. SAWYER a Julia SHARP. Adaptive Water Sampling Device for Aerial Robots. Drones. 2020, 4(1). ISSN 2504-446X. Available from: doi:10.3390/drones4010005

[11] ORE, John-Paul, Sebastian ELBAUM, Amy BURGIN, Baoliang ZHAO a Carrick DETWEILER. Autonomous Aerial Water Sampling. Field and Service Robotics. Cham: Springer International Publishing, 2015, 137-151. Springer Tracts in Advanced Robotics. ISBN 978-3-319-07487-0. Available from: doi:10.1007/978-3-319-07488-7_10

[12] Multi-robot Systems [online]. Prague: Czech Technical University, 2021 [cit. 2021-12-10]. Available from: http://mrs.felk.cvut.cz

[13] Micro Aerial Vehicles: Platforms. Multi-robot Systems [online]. Prague: Czech Technical University, 2021 [cit. 2021-12-10]. Available from: http://mrs.felk.cvut.cz/research/micro-aerial-vehicles

[14] Ponorné čerpadlo 12 V ultra-tiché. Drátek.cz [online]. Havlíčkův Brod: ECLIPSERA, 2021 [cit. 2021-12-10]. Available from: https://dratek.cz/photos/produkty/d/7/7728.jpg?m=1599736080

[15] DYNAMIXEL AX-12A e-manual. ROBOTIS [online]. Seoul: ROBOTIS Co., 2021 [cit. 2021-12-10]. Available from: https://emanual.robotis.com/docs/en/dxl/ax/ax-12a/

[16] DYNAMIXEL Protocol 1.0. ROBOTIS [online]. Seoul: ROBOTIS Co., 2021 [cit. 2021-12-10]. Available from: https://emanual.robotis.com/docs/en/dxl/protocol1/

[17] STM32F042x6 Datasheet - production data. 5th rev. Geneva, Switzerland, 2017. Also available from: https://www.st.com/resource/en/datasheet/stm32f042k6.pdf

[18] ABM8G-24 MHz crystal oscillator data sheet. Irvine, California, 2015. Also available from: https://cz.mouser.com/datasheet/2/3/ABM8G-1608872.pdf

[19] AN2867 – Oscillator design guide for STM8AF/AL/S, STM32 MCUs and MPUs: Application note. 15th rev. Geneva, Switzerland, 2021. Also available from: https://www.st.com/resource/en/application_note/cd00221665-oscillator-design-guide-for-stm8afals-stm32-mcus-and-mpus-stmicroelectronics.pdf

[20] AZ1117I – Linear voltage regulator datasheet. 1-2 rev. Plano, Texas, 2015. Also available from: https://cz.mouser.com/datasheet/2/115/DIOD_S_A0001413888_1-2541960.pdf

[21] LMZM33606RLXR datasheet. 2nd rev. Dallas, Texas, 2019. Also available from: https://www.ti.com/general/docs/suppproductinfo.tsp?distId=26&gotoUrl=http://www.ti.com/lit/ds/symlink/lmzm33606.pdf

[22] PMV15ENEAR datasheet. Nijmegen, the Netherlands, 2019. Also available from: https://cz.mouser.com/datasheet/2/916/PMV15ENEA-1604828.pdf

[23] 74AXP1T45GWH 1-bit dual supply translating transceiver datasheet. Nijmegen, the Netherlands, 2020. Also available from: https://cz.mouser.com/datasheet/2/916/74AXP1T45-1880098.pdf

[24] IP4234CZ6: Single USB 2.0 ESD protection. 3rd rev. Nijmegen, the Netherlands, 2018. Also available from: https://cz.mouser.com/datasheet/2/916/IP4234CZ6-1320204.pdf

[25] ROS: Robot Operating System [online]. California: Open Robotics, 2021 [cit. 2021-12-29]. Available from: https://www.ros.org

[26] Why ROS? ROS [online]. California: Open Robotics, 2021 [cit. 2021-12-29]. Available from: https://www.ros.org/blog/why-ros/

[27] ROS Master. ROS Wiki [online]. California: Open Robotics, 2018 [cit. 2021-12-29]. Available from: http://wiki.ros.org/Master

[28] ROS Nodes. ROS Wiki [online]. California: Open Robotics, 2021 [cit. 2021-12-29]. Available from: http://wiki.ros.org/Nodes

[29] ROS Messages. ROS Wiki [online]. California: Open Robotics, 2016 [cit. 2021-12-29]. Available from: http://wiki.ros.org/Messages

[30] ROS Topics. ROS Wiki [online]. California: Open Robotics, 2019 [cit. 2021-12-29]. Available from: http://wiki.ros.org/Topics

[31] MRS serial protocol. Multi-robot Systems GitHub [online]. Prague: Multi-robot Systems (MRS) group at Czech Technical University in Prague, 2020 [cit. 2021-12-30]. Available from: https://github.com/ctu-mrs/mrs_serial

Appendix A

Attachments

Files provided on a CD:

```
water_sampler
 └─ hardware
     ├─ pcb - PCB project files
     └─ model - STEP 3D model file
 └─ software
     ├─ ros
     │   ├─ launch - Launch file for mrs_serial
     │   └─ package - Water sampler ROS Node and Keyboard Test Node
     │      package
     └─ stm32 - Water sampler STM32CUBEIDE Project
```