ELECTRONICS AND EMBEDDED FIRMWARE FOR UAVs

A Graduate Project Report submitted to Manipal Academy of Higher Education in partial fulfillment of the requirements for the award of the degree of

BACHELOR OF TECHNOLOGY

in

Mechatronics

Submitted by

Mukul Yadav
190929042

Under the guidance of

Internal Guide
Dr. Nikhil Pachauri
Department of Mechatronics
MANIPAL INSTITUTE OF TECHNOLOGY

External Guide
Varun Raghavendra
AIR Labs, IISc Bangalore

MANIPAL INSTITUTE OF TECHNOLOGY
MANIPAL
(A constituent unit of MAHE, Manipal)

July 2023

MANIPAL INSTITUTE OF TECHNOLOGY
MANIPAL
(A constituent unit of MAHE, Manipal)

Manipal
12/7/ 2023

CERTIFICATE

This is to certify that the project titled Electronics and Embedded Firmware for UAVs is a record of the bonafide work done by Mukul Yadav **(190929042**) submitted in partial fulfillment of the requirements for the award of the degree of **BACHELOR OF TECHNOLOGY** in **MECHATRONICS of** Manipal Institute of Technology, Manipal, Karnataka (A constituent unit of Manipal Academy of Higher Education, Manipal) during the year 2022-2023.

Dr. Nikhil Pachauri
Assistant Professor
Department of Mechatronics
MIT, Manipal

Dr. D.V. Kamath
Professor & HOD
Department of Mechatronics
MIT, Manipal

INTERNSHIP COMPLETION CERTIFICATE

Dr. Suresh Sundaram, Associate Professor
Artificial Intelligence and Robotics Lab (AIRL)
Department of Aerospace Engineering
Indian Institute of Science
Bangalore 560012

+91-80-2293 2755 (Phone/Office)
+91-80-23600134 (Fax/Office)

vssuresh@iisc.ac.in
http://aero.iisc.ac.in/index.php/people/suresh-sundaram/

Date: 3rd May 2023

CERTIFICATE

This is to certify that the internship project entitled "Electronics and Embedded Firmware for UAVs" has been carried out by Mr. Mukul Yadav bearing, Manipal Institute of Technology, Udupi, Karnataka. The project was carried out at Artificial Intelligence and Robotics Laboratory, Department of Aerospace Engineering Located at Indian Institute of Science (IISc), Bangalore effect from 10th December 2022 to 30th April 2023. It is certified that he has completed the project successfully.

Suresh Sundaram

Prof. Suresh Sundaram
Associate Professor
Department of Aerospace Engineering
Indian Institute of Science, B'lore - 12.

ACKNOWLEDGEMENTS

I hereby express my gratitude and extend my heartfelt thanks to all those who have supported and guided me during my internship at AIR Labs. Firstly, I would like to thank my supervisor, Mr. Varun Raghavendra, for providing me with the opportunity to work with him and his support during the internship. I also thank the other folks at AIR labs for assisting me in times of doubt and making me feel welcome. I am grateful to all the workmates who generously shared their knowledge and expertise with me, helping me learn and grow both personally and professionally.

A special mention goes out to Professor Suresh Sundaram from the Department of Aerospace, IISc Bangalore, who, despite not being my direct supervisor, has enabled the work to happen at AIR Labs. None of this would be possible without him.

I am also thankful to my friends and family for their support and encouragement throughout my internship. Without their constant motivation, I would not have been able to make the most of this opportunity.

I must also mention Dr. Nikhil Pachauri from the Department of Mechatronics. He was my internal guide during this time and has provided valuable and constructive feedback for the work I did.

Finally, I would like to express my deepest appreciation to MIT Manipal for providing me with the impulse to pursue this internship.

ABSTRACT

This internship report provides an overview of the work done in electronics, with afocus on the embedded software, flight control algorithms, and device drivers. The report begins by introducing UAVs and their use in various industries, including surveillance, search and rescue, and agriculture. The report then delves into the technical aspects of the project, describing the hardware and software components used for the tasks carried out. The embedded software for a UAV, which runs on a microcontroller, is developed using the C programming language. The report covers the design and implementation of flight control algorithms, including design principles of embedded software, and several custom PCBs. Furthermore, the report highlights the importance of device drivers in a UAV's operation. Device drivers are used to interface the microcontroller with sensors, actuators, and other peripheral devices used in the UAV. The report concludes with a summary of the major results achieved and the lessons learned during the internship. This internship report provides an in-depth insight into the tasks carried out, and the challenges encountered while developingthe embedded software, flight control algorithms, and device drivers.

CONTENTS

LIST OF FIGURES

NOMENCLATURE

UAV	Unmanned Aerial Vehicle
PCB	Printed Circuit Board
ESC	Electronic Speed Controller
IMU	Inertial Measurement Unit
NMEA	National Marine Electronics Association
GPS	Global Positioning System
TOF	Time of Flight
RTOS	Real Time Operating System

CHAPTER 1
INTRODUCTION

1.1 UAVs and Their Relevance in the Modern World:

Unmanned aerial vehicles (UAVs), or drones, are revolutionizing the way we live, work, and interact with the world around us [1]. With their ability to fly and collect data from hard-to-reach places, they have become increasingly popular in recent years [2]. Their versatility and range of applications have made them an indispensable tool in various industries, including agriculture, construction, mining, search and rescue, and filmmaking [3].

In agriculture, drones are used to monitor crop growth and detect crop diseases, providing farmers with valuable information to improve crop yields and reduce crop losses [4]. In construction, they are used to survey sites and provide real-time updates on the progress of the project, making the construction process more efficient and cost-effective [5]. In search and rescue, drones are being used to locate missing persons and provide aerial views of the terrain, allowing rescuers to plan and execute rescue operations more effectively [6].

The use of drones has also extended to the military, where they are used for surveillance and reconnaissance, and in some cases, for targeted attacks [7]. However, there are ethical concerns regarding the use of drones in warfare and the potential for civilian casualties [8]. The use of drones has also raised privacy concerns, with many individuals worried about the possibility of drones being used for surveillance [9].

Despite these concerns, the potential benefits of drones are significant, and their use is likely to continue to grow in the years to come [10]. As the technology continues to improve, drones will become even more versatile and useful, providing new opportunities for businesses, researchers, and individuals [11].

1.2 Components of a UAV:

Frame: The frame is the basic structure of the UAV and provides support for all the other components. It is typically made of lightweight materials, such as carbon fiber or aluminum, to minimize weight and increase maneuverability.

Motors and propellers: The motors and propellers are used to generate lift and provide propulsion for the UAV. They are typically electric and can be controlled using a flight controller to adjust speed and direction.

Flight controller: The flight controller is the "brain" of the UAV and is responsible for controlling the motors and other systems. It uses sensors, such as accelerometers and gyroscopes, to measure the orientation and movement of the UAV and adjust the motor speeds accordingly.

Battery: The battery provides power for the motors and other systems. It is typically a rechargeable lithium-ion battery that is chosen based on the size and weight of the UAV.

Sensors: Sensors are used to provide information about the UAV's environment and orientation. This can include GPS for location tracking, barometers for altitude measurement, and cameras for visual information.

Communication system: The communication system allows the UAV to transmit and receive information with ground control stations and other UAVs. This can include radios, Wi-Fi, and cellular communication.

Payload: The payload is any additional equipment or sensors that are carried by the UAV. This can include cameras, sensors, and other equipment required for specific applications.

1.3 About Embedded Software:

Embedded software is a type of software that is designed to run on a specific hardware platform, typically a microcontroller or microprocessor. It is designed to perform a specific function, such as controlling a motor or reading data from a sensor. Embedded software is used in a wide range of applications, including consumer electronics, automotive systems, medical devices, and industrial automation.

One of the primary features of embedded software is its ability to run efficiently on low-power devices. Unlike general-purpose software, embedded software is optimized to run on a specific hardware platform, providing a high level of performance and efficiency. It typically uses low-level programming languages like C and assembly, which allow the software to interact directly with the hardware without the overhead of a higher-level programming language.

Embedded software is also highly reliable, as it is designed to operate in real-time environments where failures can have serious consequences. For example, embedded software used in medical devices must be highly reliable to ensure patient safety. Similarly, embedded software used in automotive systems must be highly reliable to ensure the safety of the driver and passengers.

Another key feature of embedded software is its ability to work with limited resources. Embedded devices often have limited memory, processing power, and storage space, which means that the software must be designed to work within these constraints. This can be a challenging task for developers, as they must find ways to optimize the software to work with limited resources while still providing the necessary functionality.

Embedded software is essential to the operation of many modern devices, including UAVs, medical devices, and automotive systems. In UAVs, embedded software is used to control the flight of the drone, stabilize the drone in flight, and communicate with sensors and other peripheral devices. In medical devices, embedded software is used to monitor patient health and deliver appropriate treatments. In automotive systems, embedded software is used to control the engine, transmission, and other systems in the vehicle.

1.4 Control Algorithms for UAVs

Control algorithms are an essential component of the flight control system of unmanned aerial vehicles (UAVs). They are used to stabilize and control the flight of the UAV, ensuring that it can fly safely and accurately. Control algorithms work by taking input from sensors on the UAV and processing it to generate control signals that adjust the position, speed, and direction of the UAV.[12]

There are several types of control algorithms used in UAV flight control systems, each with its own strengths and weaknesses. The most common types of control algorithms are proportional-integral-derivative (PID) controllers and model-based controllers.[13]

PID controllers are the most basic type of control algorithm used in UAV flight control systems. They work by calculating an error signal that represents the difference between the desired position or velocity of the UAV and its current position or velocity. The error signal is then used to adjust the UAV's control inputs to bring it closer to the desired position or velocity. PID controllers are simple and reliable, but they can struggle to handle complex dynamics or external disturbances.[14]

Model-based controllers, on the other hand, use mathematical models of the UAV's dynamics to generate control inputs. These models consider the physical properties of the UAV, such as its mass, aerodynamic properties, and propulsion system. Model-based controllers are more complex than PID controllers, but they can provide more accurate and robust control in complex environments or when dealing with external disturbances.[15]

Other types of control algorithms used in UAV flight control systems include adaptive control, nonlinear control, and optimal control. These algorithms can provide more advanced capabilities, such as automatic tuning, adaptive response to changing conditions, and the ability to optimize performance under specific constraints.[16]

1.5 Pixhawk: A Case Study

Pixhawk is an open-source hardware and software platform used in unmanned aerial vehicles (UAVs) for flight control and navigation. It is one of the most widely used flight control systems in the UAV industry and is popular among hobbyists, researchers, and commercial operators alike.

Pixhawk was developed by the open-source community as an alternative to commercial flight control systems that were often expensive and proprietary. The first version of Pixhawk was released in 2011, and since then, it has undergone several iterations, with the latest version being Pixhawk 4. Pixhawk is designed to be highly modular, allowing users to customize and expand the system as needed.

Pixhawk hardware consists of a main board and a set of peripheral boards, including GPS, sensors, and communication modules. The main board contains the central processing unit (CPU), memory, and interfaces for peripheral boards. The peripheral boards are connected to the main board via a standardized interface, allowing for easy customization and expansion of the system.

Pixhawk software is based on the open-source ArduPilot firmware, which provides a comprehensive suite of flight control and navigation algorithms. ArduPilot is compatible with a wide range of UAV platforms, including fixed-wing aircraft, quadcopters, and even ground-based robots. It provides a range of features, including autonomous flight, mission planning, and telemetry data logging.

One of the key features of Pixhawk is its support for a wide range of peripheral devices and sensors. This includes GPS, inertial measurement units (IMUs), barometers, and airspeed sensors, among others. Pixhawk also supports a variety of communication protocols, including MAVLink, which is a common protocol used in the UAV industry.

Pixhawk has become an industry standard in the UAV industry, with a large and active user community contributing to its ongoing development and improvement. It has been used in a wide range of applications, from hobbyist drones to commercial surveying and mapping operations.

The modular design and open-source software of Pixhawk make it an attractiveoption for developers and operators looking for a customizable and flexible flight control system for their UAVs.

CHAPTER 2
LITERATURE REVIEW

2.1 Introduction:

Unmanned aerial vehicles (UAVs) have witnessed significant advancements in recent years, enabling their widespread applications across various industries. At the core of UAV flight control systems lie control algorithms, which play a vital role in stabilizing and manoeuvring the UAVs with safety and precision. This literature review aims to explore the findings of several key studies and their methodologies related to control algorithms for UAV flight control systems.

2.2 Methodology:

To conduct this literature review, a comprehensive search was performed across academic databases and relevant sources. Five key studies were selected based on their relevance, methodology, and contributions to the field of UAV control algorithms. The selected studies include works by Valavanis [12], Yang et al. [13], Raffo et al. [14], Elbouchikhi and Veluvolu [15], and Ding et al. [16].

2.3 Key Findings:

The reviewed studies reveal various control algorithm types employed in UAV flight control systems, each offering unique strengths and weaknesses. Proportional-integral-derivative (PID) controllers, as discussed by Valavanis (2017) and Raffo et al. (2017), represent a fundamental and widely used algorithm. PID controllers calculate error signals to adjust control inputs, allowing UAVs to approach desired positions or velocities. While PID controllers are simple and reliable, they may struggle with complex dynamics or external disturbances.

Model-based controllers, explored by Valavanis (2017), Elbouchikhi and Veluvolu (2020), and Ding et al. (2018), utilize mathematical models of UAV dynamics. These models consider physical properties like mass, aerodynamics, and propulsion, enabling more accurate and robust control in complex environments. Model-based controllers offer enhanced performancebut are more complex than PID controllers.

Additionally, adaptive control, nonlinear control, and optimal control algorithms are mentioned in Valavanis (2017) and Ding et al. (2018). Adaptive control algorithms exhibit the ability to

dynamically adjust control parameters based on changing conditions. Nonlinear control algorithms handle UAV dynamics that deviate from linear behavior. Optimal control algorithms optimize performance based on specific constraints.

2.4 Methodologies Employed:

The methodologies employed in these studies involve theoretical analysis, mathematical modeling, and experimental evaluations. Valavanis (2017) provides an extensive handbook encompassing UAV control algorithms, incorporating theoretical concepts and practical examples. Yang et al. (2016) focus on nonlinear flight control for small-scale UAVs, emphasizing theoretical analysis and control algorithm design. Raffo et al. (2017) investigate robust control for UAVs, encompassing theoretical insights and flight experiments. Elbouchikhi and Veluvolu (2020) present a survey of nonlinear model-based control methods, highlighting theoretical discussions and case studies. Ding et al. (2018) propose an optimal control approach for UAV trajectory tracking, employing theoretical analysis and simulations.

2.5 Conclusion:

This literature review reveals that control algorithms for UAV flight control systems encompass a range of methodologies and algorithmic types. The PID controllers represent a fundamental approach, while model-based controllers offer enhanced accuracy and robustness. Adaptive, nonlinear, and optimal control algorithms provide advanced capabilities to address dynamic conditions and optimize performance. The findings of the reviewed studies contribute to the understanding and development of UAV control algorithms, supporting the continued advancement of UAV technology and applications in diverse fields.

CHAPTER 3
OBJECTIVES AND METHODOLOGY

3.1 Objectives

Following are the objectives of this internship:

- To gain an understanding ofthe principles of embedded software development and how they apply to UAVs and associated tools.
- To understand and apply common control algorithms used in UAVs, such as PID (Proportional Integral Derivative) and Kalman filtering.
- To design components that aid in building UAVs and other robots, such as landing gears, PCB enclosure, camera mounts and other parts.
- To learn how to design PCBs (Printed Circuit Boards) as flight controllers for UAVs.

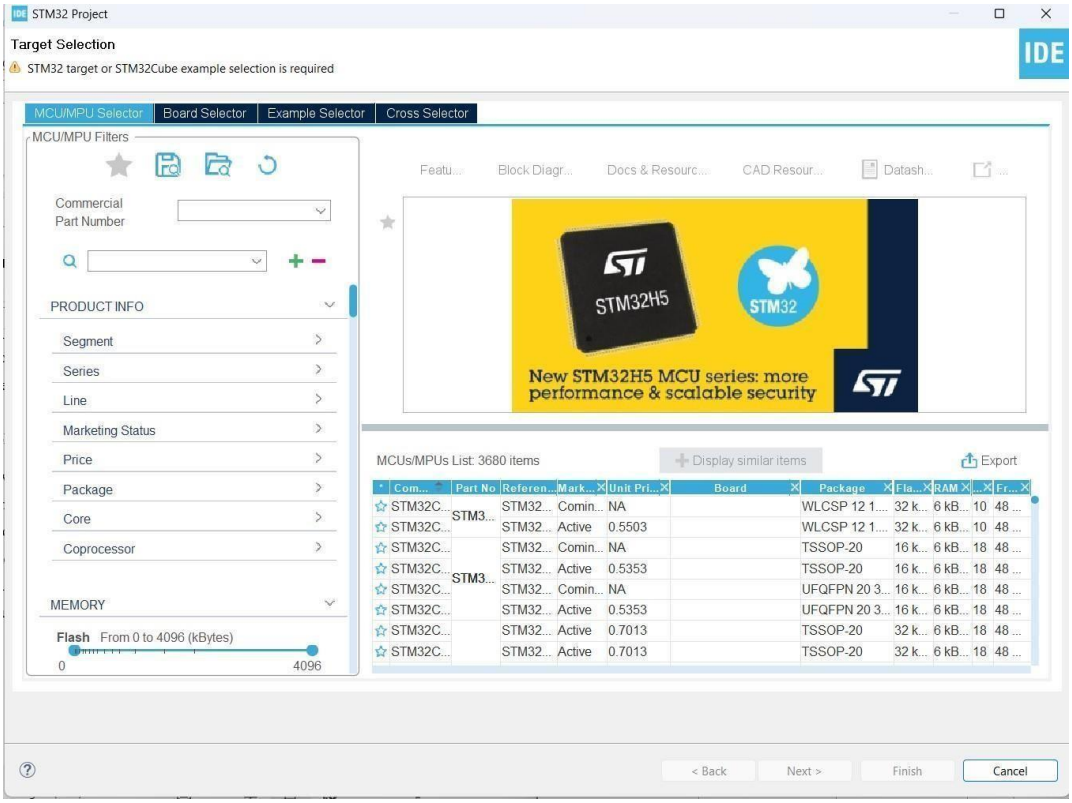
3.2 Methodology

To be able to make contributions in the area of embedded software, familiarity with a number of tools is required. The bulk of the work being done involves STMicroelectronics' STM32 line of microcontrollers, hence one must be familiar with the relevant toolchains, IDE and the debugging tools.

Embedded Software:

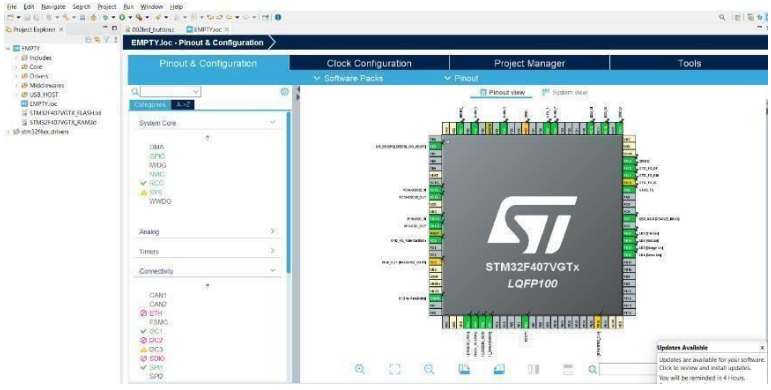
The STM32CubeIDE provides a comprehensive and user-friendly platform for software development, debugging, and programming of STM32-based projects. The IDE provides a set of software libraries, middleware, and examples for STM32 microcontrollers. This integration simplifies the development process by providing ready-to-use code and peripheral drivers, allowing one to focus on their application-specific functionality. The C programming language shall be used for all purposes.

Upon creating a new project, the IDE prompts the user to select the MCU/Board they are working with. This allows the IDE to create the necessary startup and linker scripts.



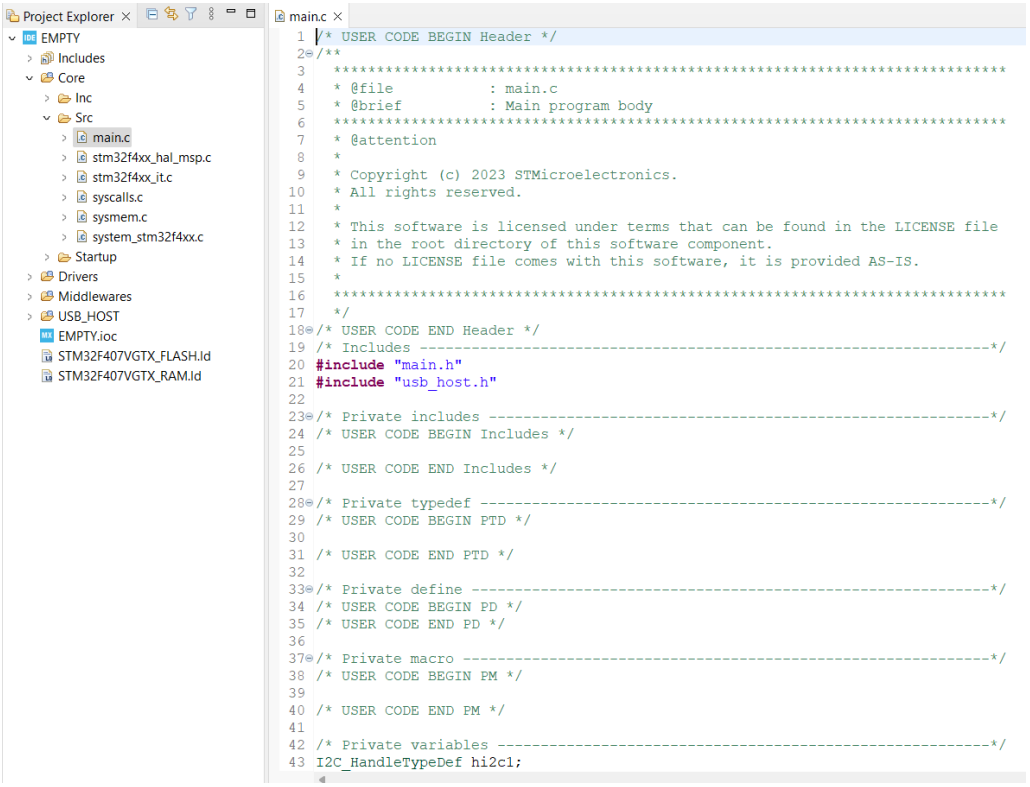
3.1 CubeIDE MCU Selection Page

The illustration below is the screen encountered should the user decide to graphically configure the microcontroller. This screen generates code to initialise peripherals, interrupts, and the clock. Alternatively, the user could create an empty project and write all the code from scratch. The latter approach was used by the author while writing embedded drivers for a better understanding of the internal workings of the MCU.



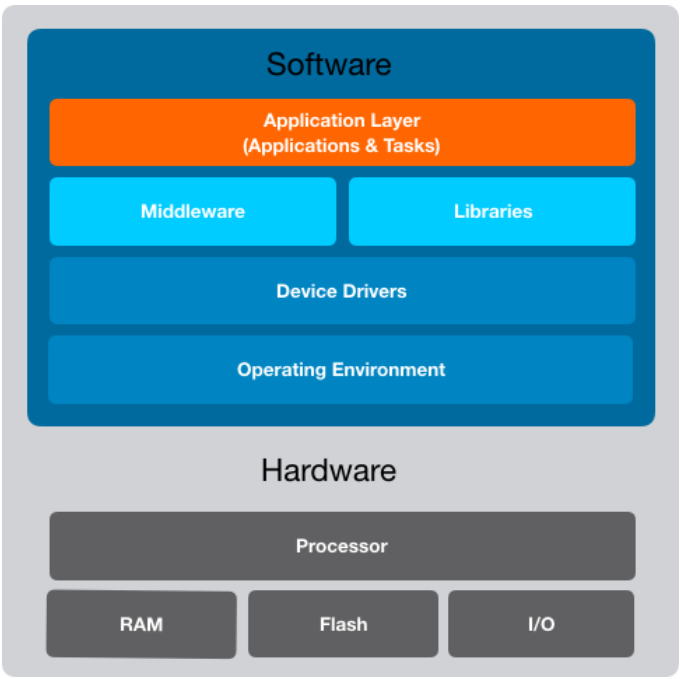
3.2 MCU Configuration Page

In case of auto initialised code, the user simply has to write the application in “main.c”. A hardware abstraction layer is provided by STMicroelectronics to ease interaction with the MCU’s peripherals.



3.3 Project Tree

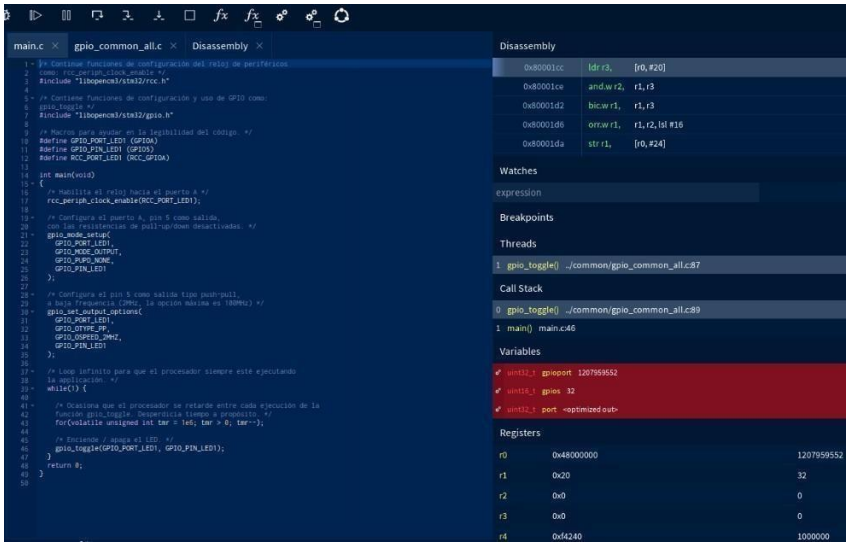
During the course of this internship, work was carried out on both the application layer and the device driver layer. An illustration of the typical embedded software stack is shown below:



3.4 Embedded Software Layers

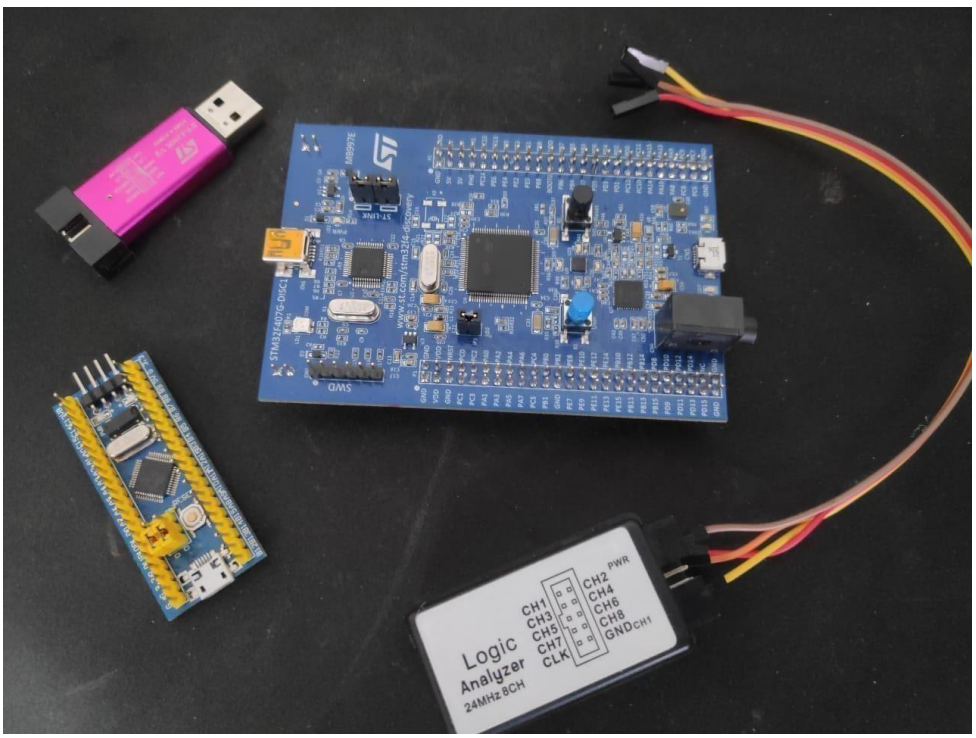
The STM32CubeIDE provides several options for debugging a user's application. However, a device to bridge the host (User PC) to the target (MCU) is needed while using a few STM32

boards. This device is the STLink programmer. Inexpensive clones of this device are often sold by online retailers. In case of the programmer being a clone, it may be incompatible with the STM32CubeIDE. This is when a general debugging environment such as OpenOCD is useful. It allows breakpoints, probing of expressions, and step-by-step execution. Once set up, it serves as a highly versatile tool for finding elusive bugs.



3.5 OpenOCD with STM32

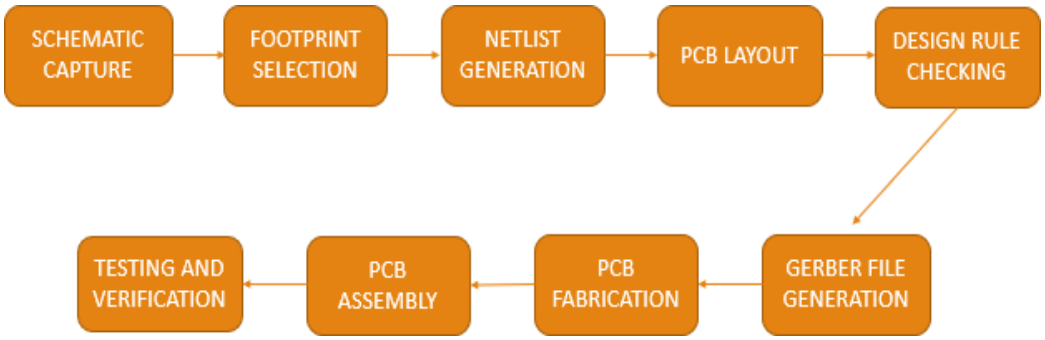
Following are the major hardware tools used to aid in embedded software development. Starting at the top left and scanning counterclockwise, they are: STLink programmer, STM32F1 MCU, USB logic analyser and an STM32F4 MCU.



3.6.Tools Used for Software Development

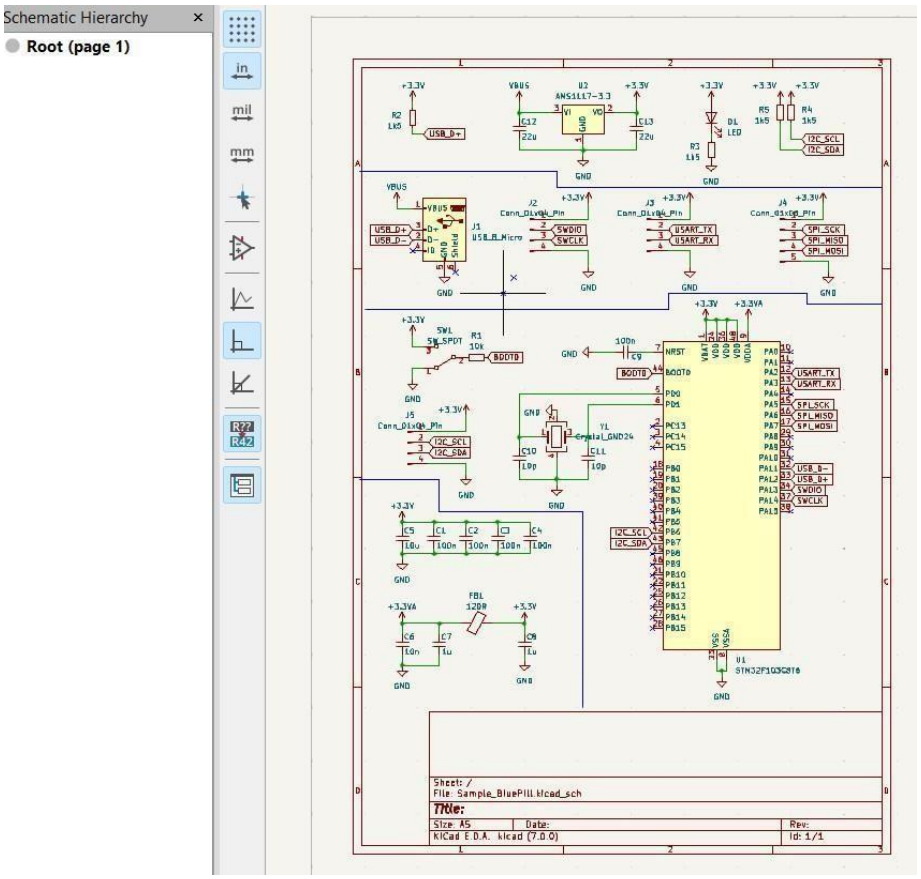
PCB Design:

Several PCBs were designed during the course of this internship, and KiCAD was used for this purpose. Following is the workflow for PCB design with KiCAD.



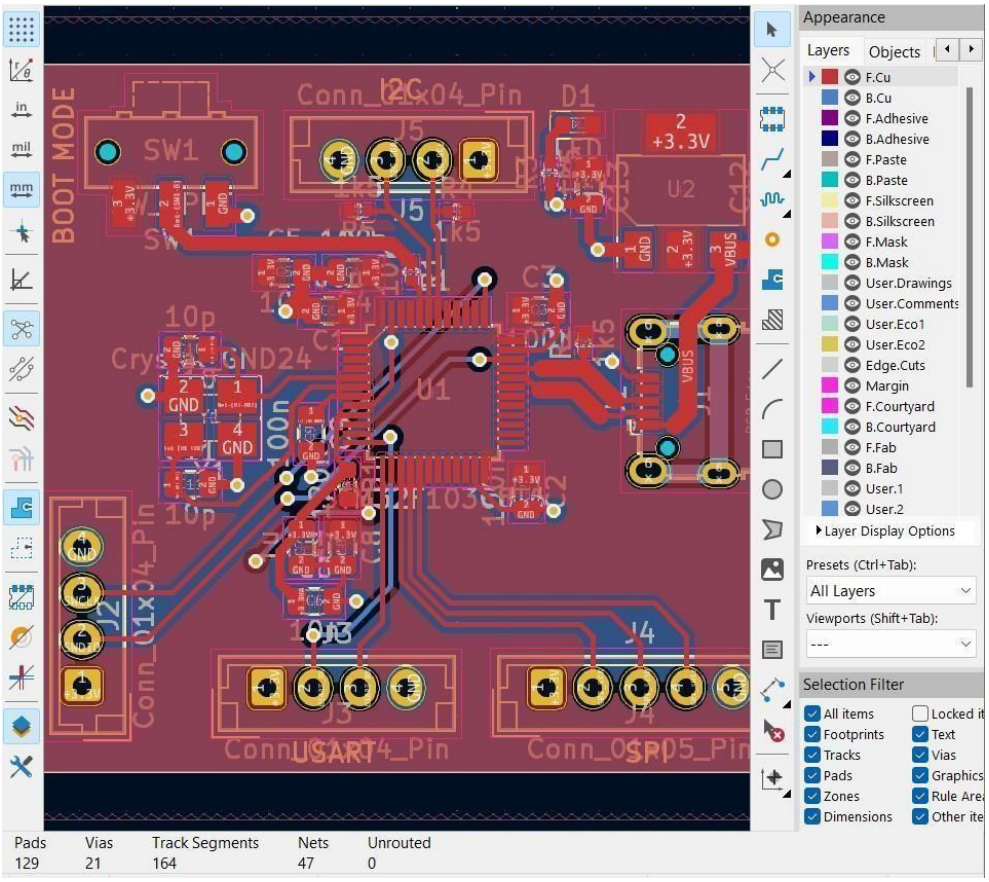
3.7.PCB Design Workflow

Following is the schematic view, any idea for a circuit must be drafted here first.



3.8.Schematic Draft

The schematic is then translated into a netlist, which dictates how the components are linked. This netlist is then imported to a board layout screen where routing can be completed.



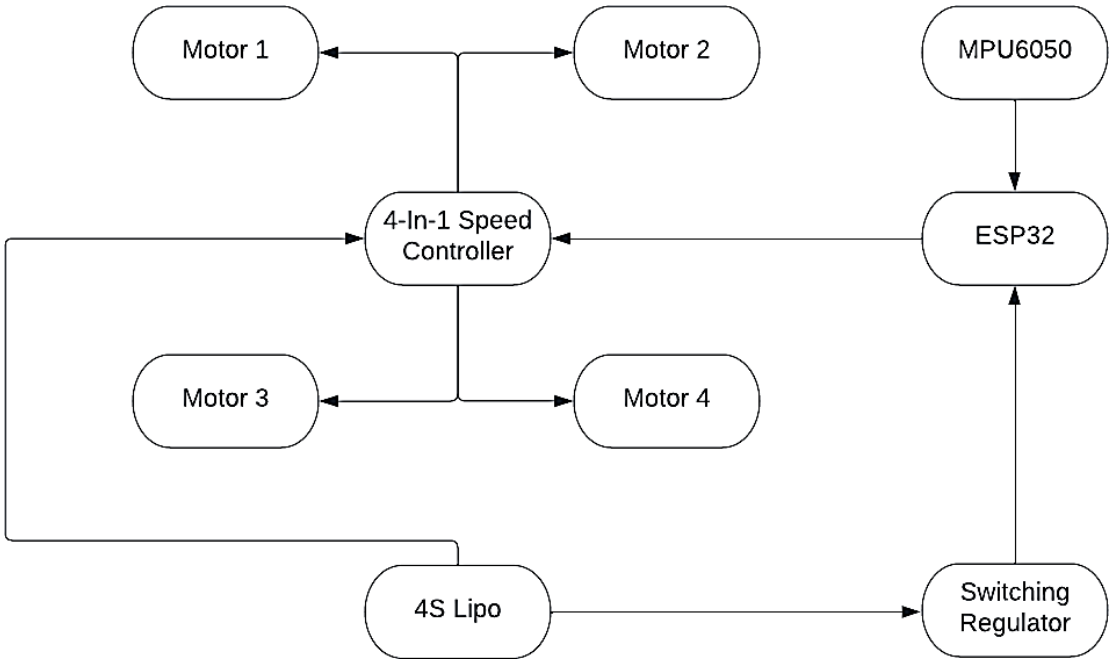
3.9.PCB Layout

After a design rule check, Gerber files are generated which can be sent to a manufacturer to get the PCB completed. This concludes the methodology used for completing repeated tasks during the course of this internship. It is important to note that several other tools and methods may have been used, yetthey were not significant/frequent enough to detail here. The author shall cover them while speaking of all tasks individually.

CHAPTER 4
RESULTS AND DISCUSSION

4.1 Building and Assembling Drones:

Several drones were worked on during the internship. For example, a simple drone was assembled using an ESP32 as a flight controller. The block diagram below illustrates the system architecture:

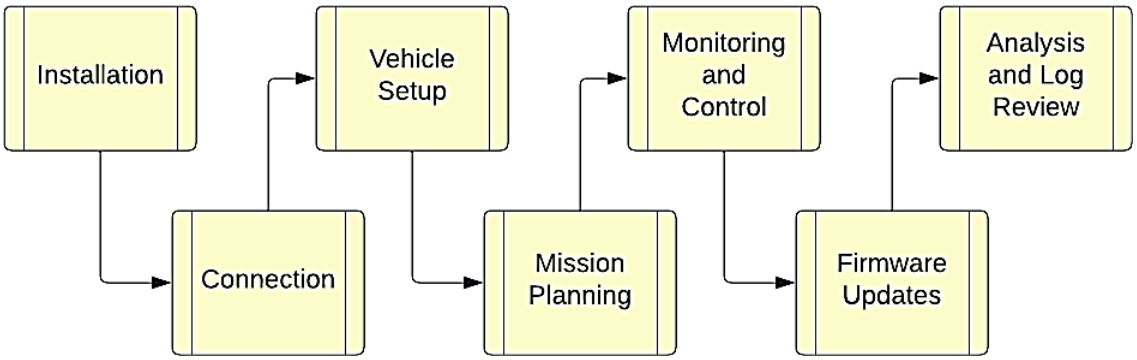


4.1 Drone System Block Diagram

The ESP32 is equipped with wireless capabilities such as Wi-Fi and BLE. This makes control of the drone possible without investing in a relatively expensive transceiver kit. The speed controller receives PWM signals from the ESP32. It is worth noting that motors used in this configuration require a PWM frequency of 500 Hz. The speed of the motors is mapped to a pulse width of 1100 us to 2000 us. An MPU 6050 IMU captures attitude data and the ESP32 implements the appropriate control algorithm. A downside of using this system architecture is that the drone stays feature limited.

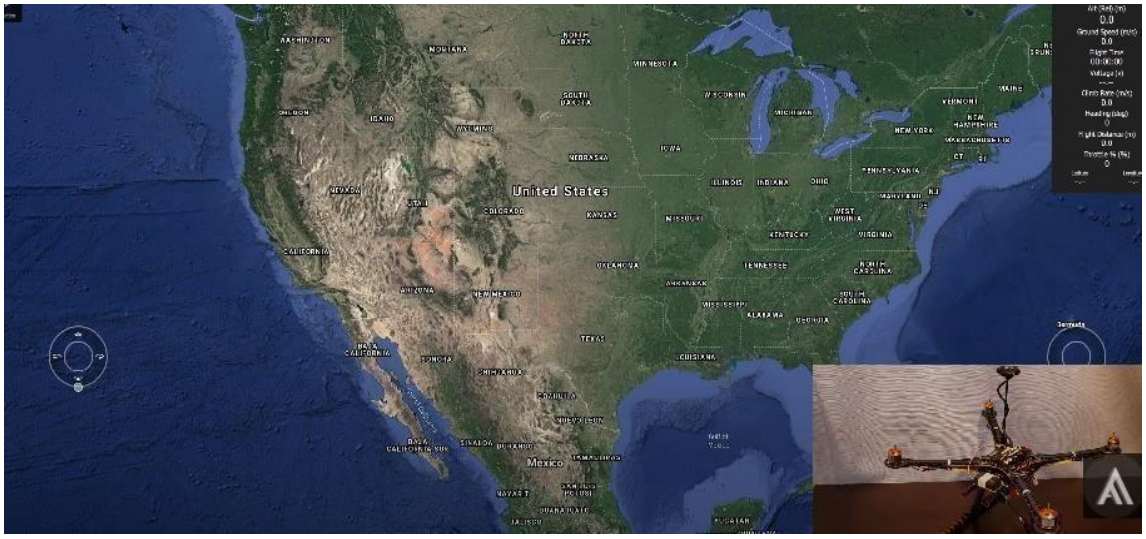
4.2 Q Ground Control and Pixhawk 4:

Q Ground Control is an open-source ground control station (GCS) software designed for unmanned aerial vehicles (UAVs) and other autonomous systems. It provides a user-friendly interface for mission planning, vehicle control, and monitoring of various aspects of autonomous vehicles. Q Ground Control is widely used in the field of robotics, particularly in the development and operation of drones. It features motion planning, vehicle control, telemetry, monitoring, geofencing, among others. Q Ground Control is fully compatible with Pixhawk flight controllers and using them together allows for a powerful combination for controlling and monitoring unmanned vehicles.



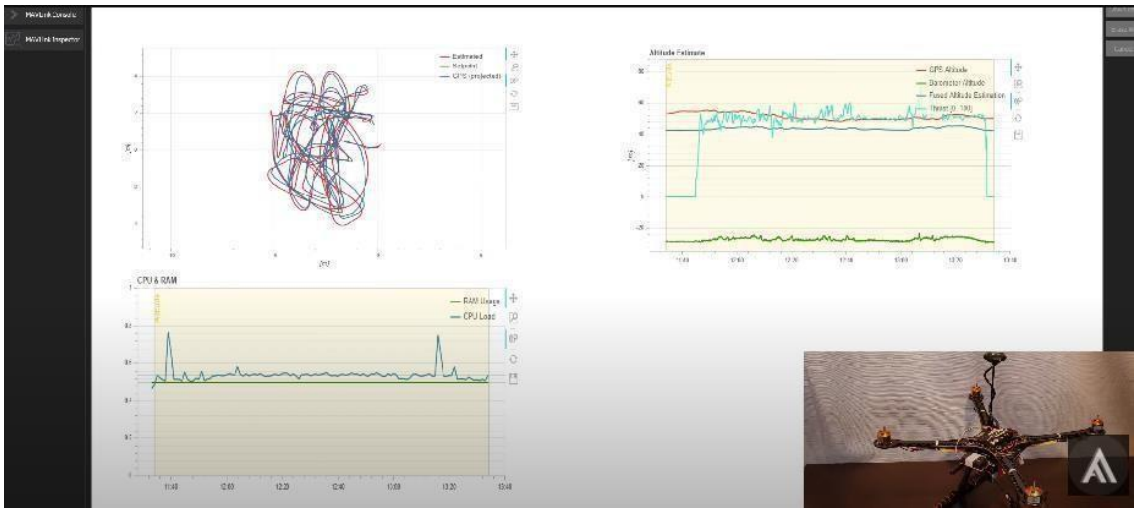
4.2 Q Ground Control Setup

The section below depicts Pixhawk and Q Ground Control in use and is similar to what the author encountered while interacting with the combined system. Upon connection, the Q Ground Control interface detects user location.

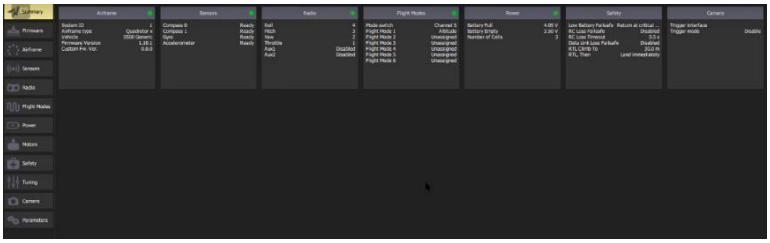


4.3 Triangulation

One can monitor multiple statistics in real time, including attitude, CPU and RAM usage and temperature.

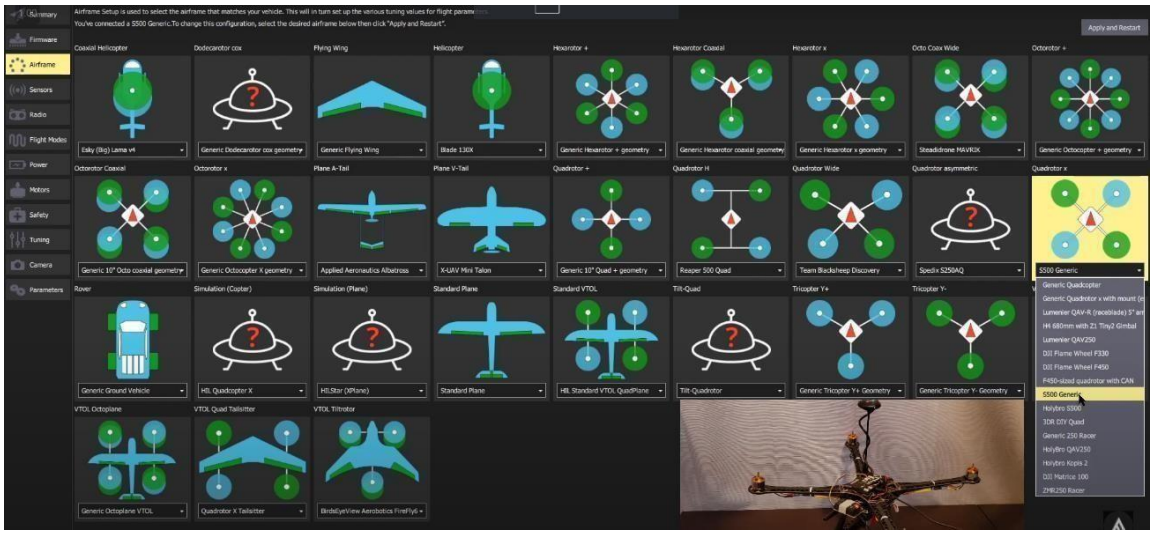


4.4 Analysis



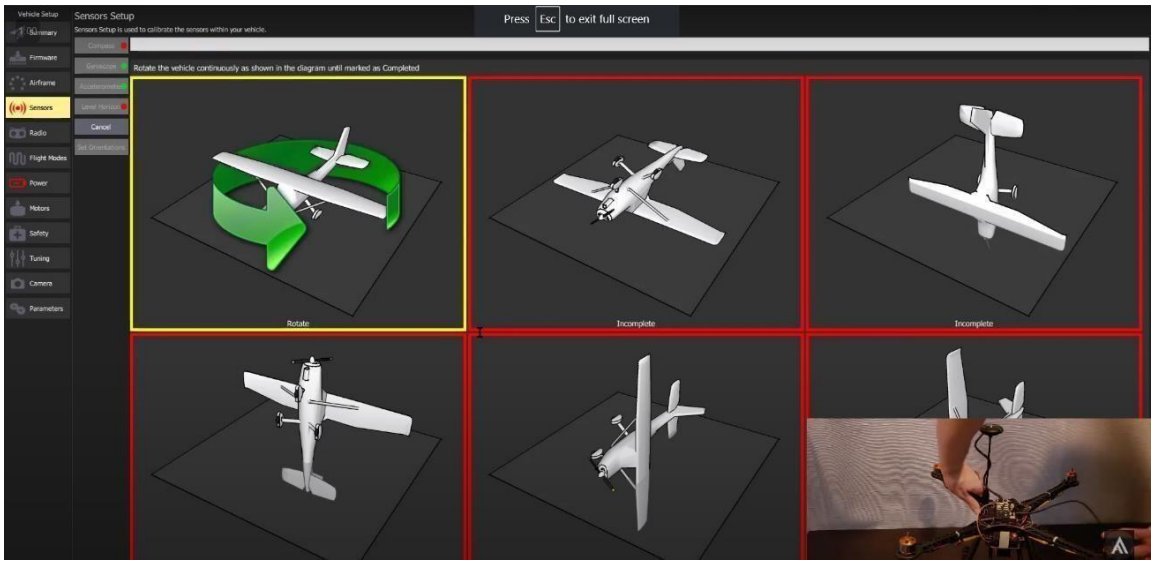
4.5 Flight Configuration Summary

The same flight control firmware does not work for all UAV configurations. Our specific UAV type can be selected as shown below.



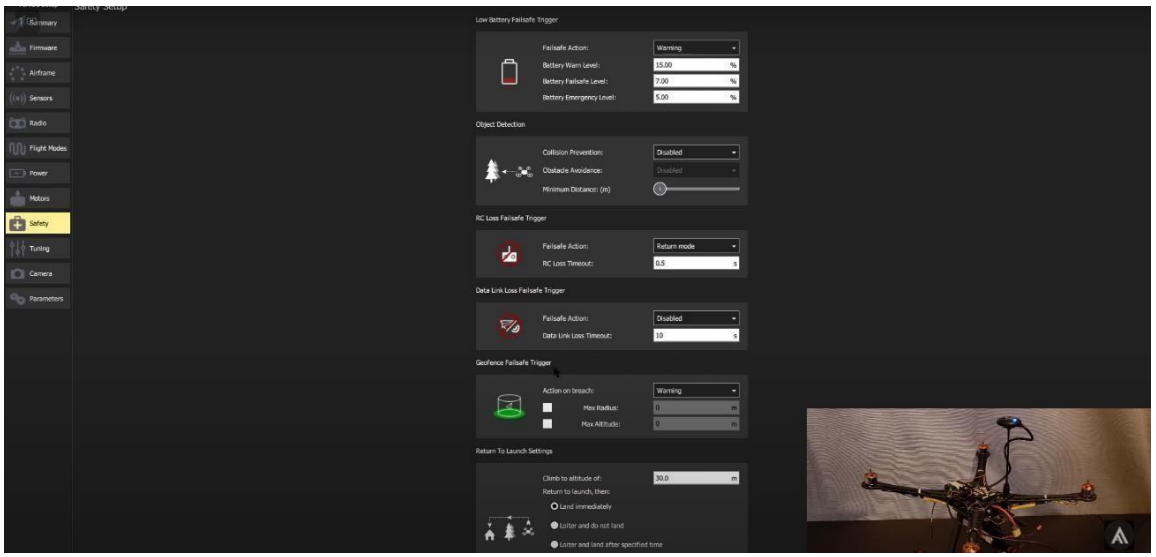
4.6 UAV Configuration Options

Sensor calibration is an important setup step for any UAV.



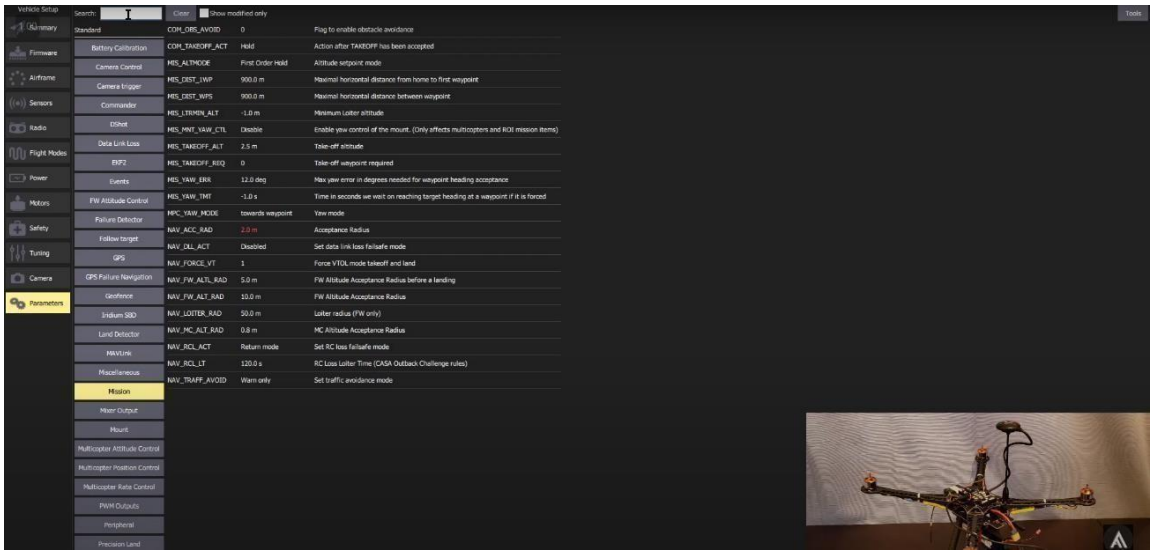
4.7 Sensor Calibration

Multiple safety features are built-in to the flight control firmware instructing the vehicle what to do in case of failures such as telemetry loss, geofence breach and power interruptions.



4.8 Falisafes

In some cases, fine tuning of parameters may be needed. The interface provides this feature as well. For example, PID sliders are provided for seamless tuning of the control algorithm.



4.9 Parameters

It can be seen how such projects make UAVs much more accessible for newbies. Pixhawk and Q Ground Control provide a robust system for configuring a UAV for a wide range of parameters.

4.3 Influence of High Currents and Inductive Loads on PCB design:

In one instance, the author was tasked with designing a PCB which can handle high currents and serves an inductive load, such as a solenoid. There are several factors to be considered while designing PCBs for such a task. [W1]

1. Trace Width and Thickness: High currents flowing through PCB traces generate heat due to the resistance of the copper conductors. To prevent excessive temperature rise and potential damage, the trace width and thickness should be carefully chosen to minimize resistance and provide adequate current-carrying capacity. Design guidelines, such as IPC-2152, can be followed to determine suitable trace dimensions based on current levels.

2. Copper Pour and Power Planes: To distribute high currents evenly across the PCB, copper pours and power planes are often employed. These large copper areas help reduce the resistance and improve thermal dissipation. Adequate copper thickness and sufficient clearance between copper pours and adjacent signal traces should be maintained to avoid interference and signal degradation.

3. Voltage Drops and Power Integrity: High currents flowing through PCB traces can cause voltage drops along the traces, especially for long or narrow traces with high resistance. These voltage drops can negatively affect the performance of the circuit and

may lead to erroneous behaviour or damage. Proper power distribution strategies, including minimizing trace lengths, using wider traces, and placing decoupling capacitors near power pins, help mitigate these voltage drops and maintain power integrity.

4. Electromagnetic Interference (EMI): Inductive loads, such as motors, solenoids, and relays, generate magnetic fields that can induce noise and interference in nearby traces or components. To minimize the impact of EMI, proper trace routing techniques should be employed. Signal traces should be routed away from high-current paths or inductive components, and differential pair routing or shielding techniques can be applied to reduce the susceptibility to electromagnetic noise.
5. Magnetic Fields and Component Placement: Inductive loads produce magnetic fields that can affect nearby sensitive components, including sensors, analogue circuitry, or communication modules. Careful component placement and separation between inductive loads and sensitive components are essential to minimize the coupling of magnetic fields. Shielding techniques, such as using magnetic shields or placing sensitive components on separate PCB layers, can provide additional protection.
6. Grounding and Return Paths: High currents require robust grounding strategies to ensure proper return paths for the currents. Separate ground planes for high-current and low-current sections, as well as the use of multiple ground vias, can help reduce ground impedance and minimize ground loops. Adequate decoupling and bypass capacitors should be placed near high-current components to maintain a stable ground reference.
7. Heat Dissipation: High currents flowing through PCB traces or components cangenerate significant heat. Proper heat dissipation measures, such as adding thermal vias,heat sinks, or thermal pads, should be implemented to prevent overheating and ensure the reliability of the circuitry. Thermal simulations and calculations can be employed to optimize heat dissipation and prevent temperature-related issues.

4.4 Black Box Systems in Modern UAVs:

Black box systems, also referred to as flight data recorders (FDRs) or flight recorders, are electronic devices installed in UAVs to capture and store critical flight data and information during their operation. These systems are designed to withstand extreme conditions and provide invaluable insights in the event of accidents, incidents, or system failures. Let's delve into some key aspects of black box systems in modern UAVs:

1. Data Recording: Black box systems are equipped to record a vast array of data parameters, including flight control inputs, sensor readings, flight trajectory, altitude, airspeed, engine parameters, and communication data. This comprehensive data collection offers investigators a detailed understanding of the UAV's behaviour and performance leading up to an incident or accident.
2. Safety and Accident Investigation: Black box systems play a crucial role in accident investigation and safety analysis. In the event of a mishap, the recorded data can provide valuable insights into what caused the incident, aiding in the determination of contributing factors, system failures, or human errors. This information is instrumental in improving UAV design, identifying safety measures, and preventing future accidents.
3. System Performance Analysis: Black box systems provide a means to assess the performance and efficiency of UAV systems. By analyzing the recorded data, manufacturers and engineers can evaluate the behaviour of the aircraft under different conditions, identify areas for improvement, and enhance system reliability and performance.
4. Regulatory Compliance: Many regulatory authorities mandate the use of black box systems in UAVs, similar to their requirement in manned aircraft. These regulations ensure that critical flight data is captured and can be utilized for safety analysis and compliance verification. Black box systems aid in meeting these regulatory requirements and help enhance the overall safety standards of UAV operations.
5. Training and Simulation: The recorded flight data from black box systems can be valuable for training purposes. UAV operators and pilots can use the data to analyze their performance, evaluate decision-making processes, and enhance their skills through simulation and training exercises. This enables operators to learn from past experiences and develop strategies for improved flight operations.
6. Real-Time Monitoring: In some cases, black box systems are equipped with real-time monitoring capabilities, allowing operators to receive live flight data and monitor UAV performance during operations. Real-time data streaming can assist in making informed decisions, detecting anomalies, and responding promptly to critical situations.

4.5 Flight Controller:

Flight controllers play a crucial role in the operation of Unmanned Aerial Vehicles (UAVs), commonly known as drones. These electronic devices serve as the "brain" of the UAV, responsible for controlling and managing its flight dynamics, stability, and autonomous

functions. In this write-up, we will explore the key features and functions of flight controllers in UAVs.

1. Control and Navigation: The primary function of a flight controller is to provide precise control and navigation capabilities to the UAV. It interprets pilot inputs or commands. from an autonomous system and translates them into control signals for the various components of the aircraft, such as motors, servos, and actuators. Flight controllers ensure the UAV responds accurately to control inputs, allowing it to perform manoeuvres, maintain stability, and follow desired flight paths.

2. Sensor Integration: Flight controllers integrate various sensors to gather critical data about the UAV's orientation, motion, and environmental conditions. These sensors typically include accelerometers, gyroscopes, magnetometers, barometers, and Global Navigation Satellite Systems (GNSS) receivers. By continuously monitoring sensor data, the flight controller can make real-time adjustments to stabilize the UAV and enable accurate position hold, altitude hold, and other flight modes.

3. Flight Modes and Autonomy: Flight controllers offer different flight modes to cater to various operating scenarios. These modes can include manual mode (direct pilot control), stabilize mode (self-levelling and stability-assisting), altitude hold mode, position hold mode (using GPS or other positioning systems), autonomous waypoint navigation, and even follow-me mode. These modes provide flexibility and autonomy, allowing UAVs to perform a wide range of tasks, from aerial photography and videography to surveying and inspection missions.

4. Flight Performance Optimization: Flight controllers employ sophisticated algorithms and control schemes to optimize flight performance and stability. These algorithms utilize sensor data and input signals to calculate control outputs that ensure smooth flight, precise maneuverability, and resistance to disturbances like wind or turbulence. PID (Proportional-Integral-Derivative) controllers are commonly used to adjust control outputs based on the error between desired and actual states.

5. Telemetry and Communication: Flight controllers often support telemetry systems, enabling real-time data transmission between the UAV and a ground control station. Telemetry data includes vital information such as GPS coordinates, altitude, battery voltage, motor RPM, and sensor readings. This data allows operators to monitor the UAV's status, make informed decisions, and troubleshoot issues remotely. Communication protocols like Wi-Fi, Bluetooth, or radio frequencies are used to

establish a reliable link between the flight controller and ground station.

6. Redundancy and Fail-Safe Measures: To enhance safety and mitigate potential failures, advanced flight controllers incorporate redundancy and fail-safe features. Redundancy involves redundant sensors, processors, or communication channels, ensuring the availability of critical systems even in the event of a component failure. Fail-safe mechanisms can trigger predefined actions, such as returning to a specified location, descending to a safe altitude, or initiating an emergency landing, in case of communication loss or critical system malfunctions.
7. Customization and Development: Many flight controllers offer open-source platforms or software development kits (SDKs) that allow enthusiasts, researchers, and developers to customize and extend their capabilities. This flexibility enables the UAV community to experiment with new features, integrate additional sensors, or develop specialized applications using the flight controller as a foundation.

4.10 Examples of Flight Controllers

4.6 Flysky Transceiver:

Flysky transceivers, also known as transmitters, are designed to provide a stable and responsive connection between the operator and the RC vehicle. These transceivers typically operate on the 2.4GHz frequency, which offers excellent signal strength and reduced interference compared to lower frequency options.

One of the key features of Flysky transceivers is their compatibility with Flysky receivers. This allows users to easily pair the transmitter with the corresponding receiver, ensuring seamless communication between the operator and the RC vehicle. Flysky transceivers often incorporate multiple channels, giving users the flexibility to control various functions of their RC vehicles simultaneously.

4.11 Flysky Transceiver

Binding Procedure:

Prepare the transmitter and receiver: Ensure that both the transmitter and receiver are powered off and have fully charged batteries installed. Make sure that any switches or knobs on the transmitter that affect binding or pairing are in the correct position as per the manufacturer's instructions.

Identify the binding procedure: Consult the user manual or documentation provided with the transmitter and receiver to determine the specific binding procedure. Different brands and models may have different methods for binding.

Put the receiver into binding mode: Most receivers have a small button, or a bind plug that needs to be pressed or inserted to put them into binding mode.

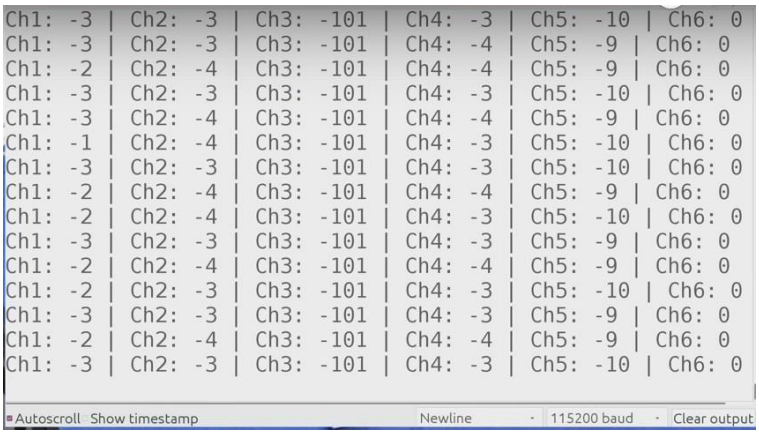
Power on the receiver: Connect the receiver to the power source, usually the ESC (Electronic Speed Controller) or a separate battery pack. The receiver's LED indicator lights may start flashing rapidly to indicate it is in binding mode.

Power on the transmitter: Turn on the transmitter while holding down the specific button or switch that initiates the binding process. The transmitter's LED or LCD screen may display a message indicating it is in binding mode.

Establish the connection: Once the transmitter is in binding mode, it will attempt to establish a connection with the receiver. The receiver's LED indicator lights may change their pattern or stop flashing altogether, indicating a successful bind.

Verify the connection: Test the controls on the transmitter to ensure that they are properly communicating with the receiver. Move the control sticks, switches, and knobs to verify that they correspond to the movements or changes on the RC vehicle.

Finalize the binding process: Once you have confirmed that the transmitter and receiver are bound successfully and the controls are working correctly, power off both the transmitter and receiver.



4.12 Flysky Channel Readings with Arduino

4.7 ESC Arming and Calibration:

Arming and calibrating electronic speed controllers (ESCs) is an essential process in setting up and configuring your RC vehicle's motor system. Arming refers to the procedure of enabling the ESC to provide power to the motor, while calibration ensures proper synchronization between the ESC and the transmitter's throttle input. Here is a general overview of the arming and calibration process for ESCs:

1. ESC Connection: Ensure that your ESC is correctly connected to the power source, which is typically a battery pack. Verify that all connections between the ESC, motor, and battery are secure.

2. Throttle Position: Set the throttle stick or trigger on your transmitter to its lowest position. Make sure the throttle trim is centered or set to its lowest position as well.

3. Power on the Transmitter: Turn on your transmitter, ensuring that it is set to the correct model or vehicle profile if applicable. This step is necessary for establishing communication between the transmitter and the ESC.

4. Power on the ESC: Connect the battery to the ESC. Some ESCs may require specific sequences or timing for power-on procedures, so refer to the ESC manual for any specific instructions. Generally, you will connect the battery and wait for the ESC to initialize.

5. Arming Confirmation: Once the ESC initializes, it will usually provide an audible beep or visual indicator (such as LED flashing patterns) to indicate that it is armed and ready for operation. This confirmation may vary depending on the brand and model of the ESC.

6. Throttle Calibration: ESCs often require calibration to synchronize the transmitter's throttle input range with the ESC's throttle response. The calibration process typically involves the following steps:

a. Move the throttle stick or trigger on your transmitter to its highest position (full throttle). The ESC may emit a specific beep or LED pattern to indicate that it recognizes the maximum throttle input.

b. Move the throttle stick or trigger to its lowest position (idle or zero throttle). Again, the ESC may provide a confirmation signal, usually a different beep or LED pattern, to indicate recognition of the minimum throttle input.

c. Some ESCs may require an additional step, such as moving the throttle stick or trigger to the neutral position (half throttle) and waiting for another confirmation signal.

7. Calibration Confirmation: After completing the throttle calibration process, the ESC will usually emit a final confirmation signal, such as a series of beeps or LED patterns, indicating that the calibration was successful.

8. Motor Testing: To ensure proper calibration and functionality, test the motor by gradually increasing the throttle input from its idle position. Observe the motor's response and verify that it accelerates smoothly and without any unexpected behaviour.

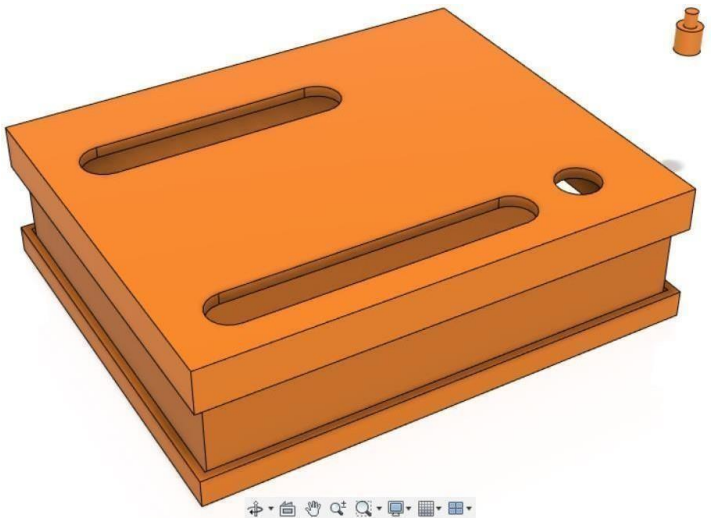
4.8 Designing and 3D Printing:

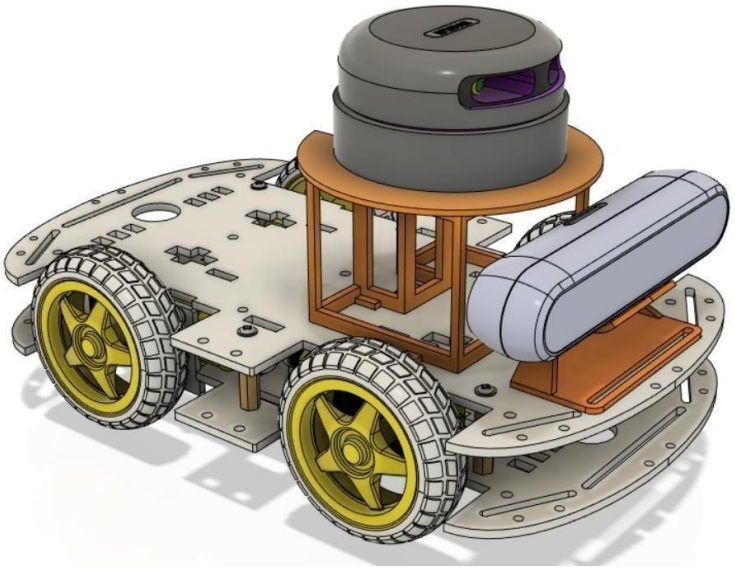
For a period, the author was tasked with designing and 3D printing components required for several tasks. This could entail enclosures for custom PCBs, landing gears for drones and camera mounts, etc…. The design was completed using Fusion360 and the slicer used was Creality.

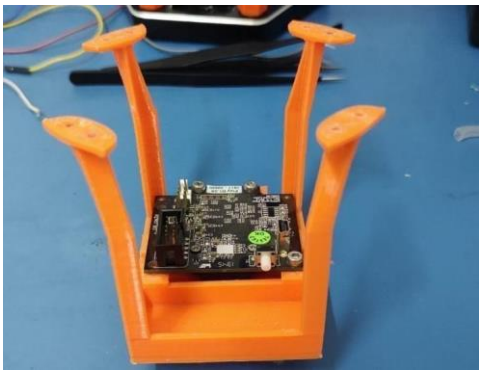
Some measures were needed to ensure the prints succeeded. These included:

- Printing of a raft or brim for adhesion.
- Low printing speed
- Higher bed temperatures

Following are some of the designs completed for the team:





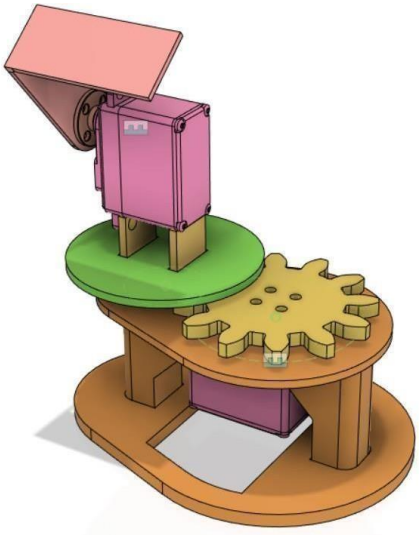


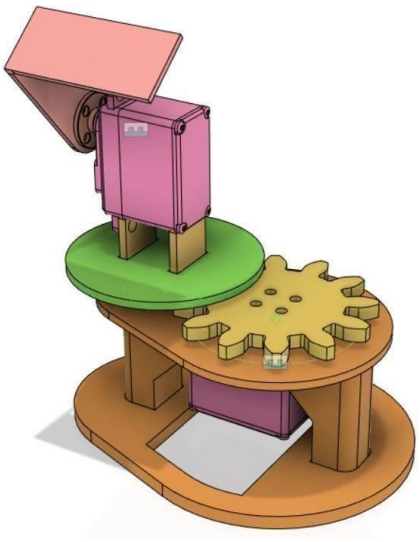
4.13. 3D Printing Work

4.9 IMU Testing Rig:

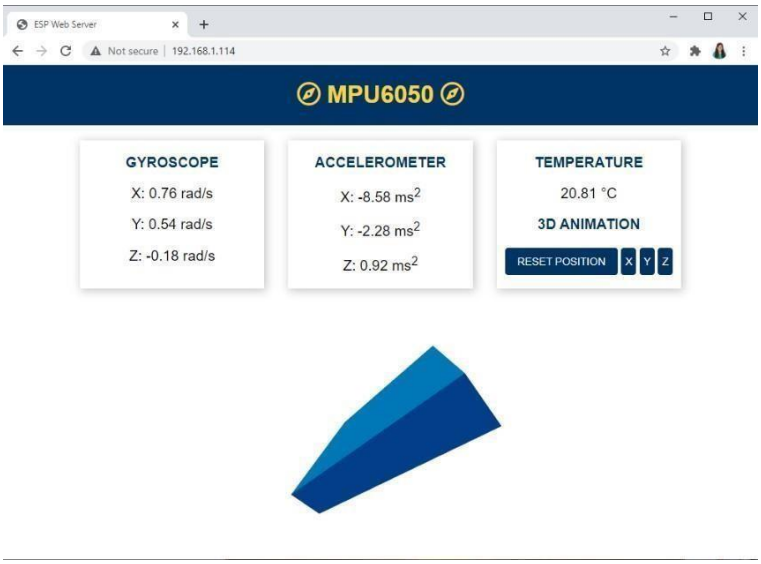
To understand the limitations of a 6 Axis IMU, a mount previously designed for another application was repurposed and used as a 2 DoF setup for the MPU 6050 to sit atop of. [W2]

The results were visualised using a webpage designed on the ESP32.





4.14 IMU Testing Rig



4.15 IMU Visualization

Observations:

- Movement of the cuboid is highly jittery. This implies the presence of high frequency noise in the readings. A low pass filter can mitigate the issue.
- The cuboid is being drawn based on the readings from the gyroscope. Because of this, the cube's orientation drifts from the actual orientation over time. Readings from the accelerometer need to be considered.
- Despite being in a stationery, the gyroscope readings are not zero. This implies the presence of sensor bias which needs to be corrected.

Possible Remedies:

1. Filtering Techniques: Implementing filtering algorithms can help reduce noise in IMU readings. One common approach is to use a low-pass filter to smooth out high-frequency noise while preserving the essential characteristics of the motion. This can be achieved by applying techniques such as moving average filters, exponential filters, or Kalman filters. Each filtering method has its own advantages and considerations, so it's important to understand the specific requirements of your application and choose the appropriate filter accordingly.

2. Sensor Calibration: Calibrating the IMU can significantly improve the accuracy of the measurements and reduce noise. This involves compensating for any biases, offsets, or non-linearities in the sensor readings. Calibration techniques often involve collecting data from the IMU in a controlled environment and applying mathematical models or calibration algorithms to estimate and correct the sensor errors. This can include calibrating sensor biases, scale factors, and misalignments.

3. Sensor Fusion: Combining data from multiple sensors can help mitigate noise and improve the overall accuracy of the measurements. Sensor fusion algorithms integrate data from complementary sensors, such as gyroscopes, accelerometers, and magnetometers, to obtain a more robust and accurate estimation of the device's orientation and motion. Techniques like the Madgwick or Mahony filters, which utilize sensor fusion algorithms like the Kalman filter or complementary filter, are commonly employed to enhance the accuracy of IMU readings.

4. Sampling Rate and Data Processing: Adjusting the sampling rate of the IMU can have an impact on the noise level. In some cases, reducing the sampling rate can help mitigate high-frequency noise, as it effectively filters out rapid variations. Additionally, applyingsuitable signal processing techniques, such as data smoothing or interpolation, can helpreduce noise and improve the quality of the measurements.

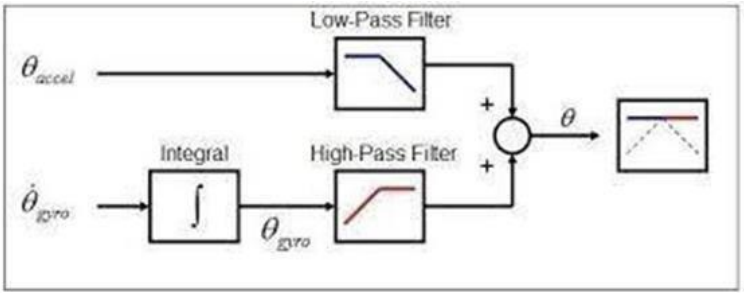
5. Mechanical Isolation: Vibration and mechanical disturbances can introduce noise into IMU readings. Mounting the IMU on a vibration-isolated platform or using anti-vibration measures can minimize external disturbances and improve the accuracy of the measurements.

4.10 Complementary and Kalman Filter:

The basic idea behind the complementary filter is to take advantage of the strengths of different types of sensors by combining their measurements in a way that compensates for their individual weaknesses. Typically, it involves combining the high-frequency response of one sensor with the low-frequency response of another sensor to create a composite signal that is more accurate across a wide range of frequencies.

In practice, the complementary filter works by applying a weighted sum of the sensor measurements, where the weights are determined based on the desired frequency response characteristics. For example, if one sensor provides accurate measurements at high frequencies but is prone to noise or drift at low frequencies, while another sensor has a good low-frequency response but is less accurate at high frequencies, the complementary filter can be designed to emphasize the high-frequency measurements from one sensor and the low-frequency measurements from the other sensor.

The filter is called "complementary" because it complements the strengths and weaknesses of the individual sensors, combining their outputs in a way that compensates for their limitations. By properly tuning the filter parameters, it is possible to achieve a more accurate and robust estimation of the desired variable or state.



4.16 Diagrammatic Representation of Complementary Filter

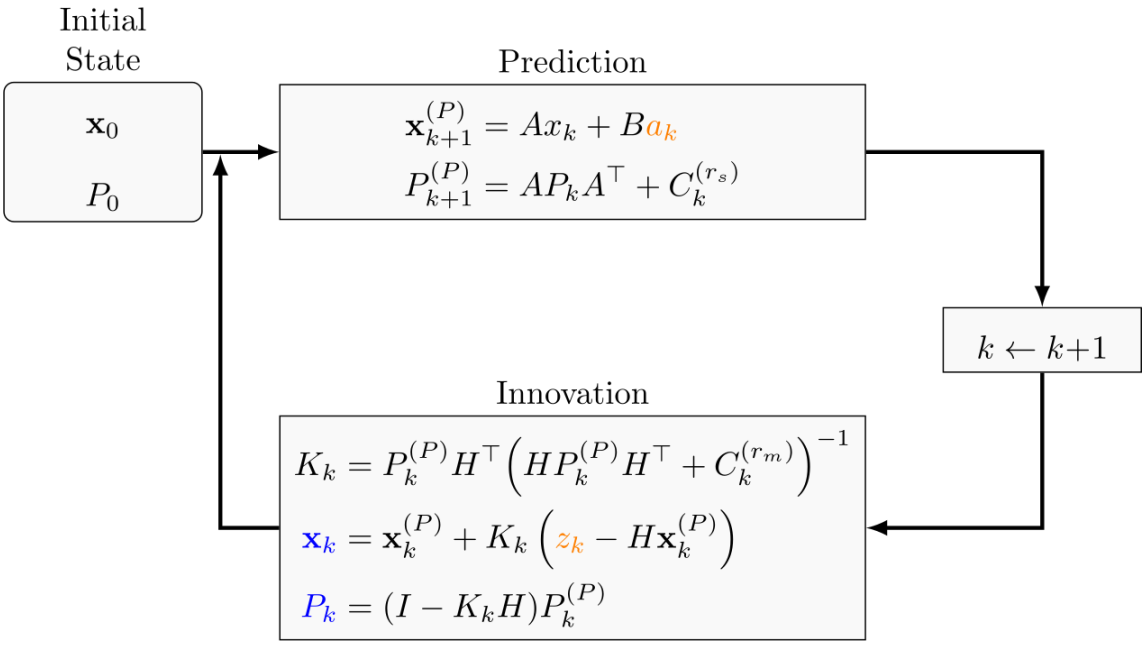
Unlike the complementary filter, which is a relatively simple blending technique, the Kalman filter is based on a rigorous mathematical framework and statistical principles. It is designed to handle both deterministic and stochastic systems, taking into account the uncertainties and noise present in the measurements and the system dynamics. [W3]

The Kalman filter operates in two main steps: the prediction step and the update step.

1. Prediction Step: In this step, the Kalman filter uses the system's dynamic model to predict the state of the system at the next time step. It estimates the future state based on the previous state estimate and the control input, considering the system's dynamics. The prediction step also provides an estimate of the error covariance, whichrepresents

the uncertainty associated with the predicted state.

2. Update Step: In this step, the Kalman filter incorporates the measurements from the sensors to update the state estimate. It compares the predicted state with the actual measurements, taking into account the noise characteristics of the sensors. The update step adjusts the state estimate based on the reliability of the measurements and the predicted state. It also updates the error covariance to reflect the updated estimate's uncertainty.



4.17 Kalman Algorithm

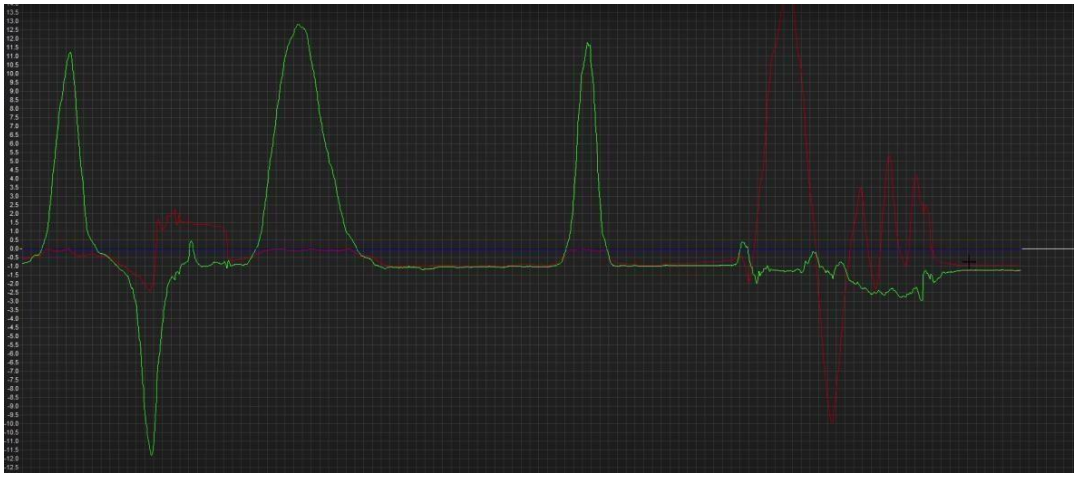
Compared to the complementary filter, the Kalman filter offers several advantages:

Optimality: The Kalman filter is an optimal estimator in the sense that it minimizes the mean square error between the estimated state and the true state, given the available measurements and system dynamics.

Statistical Modelling: The Kalman filter explicitly models the noise and uncertainties associated with the measurements and system dynamics. It uses statistical properties, such as mean and covariance, to account for these uncertainties.

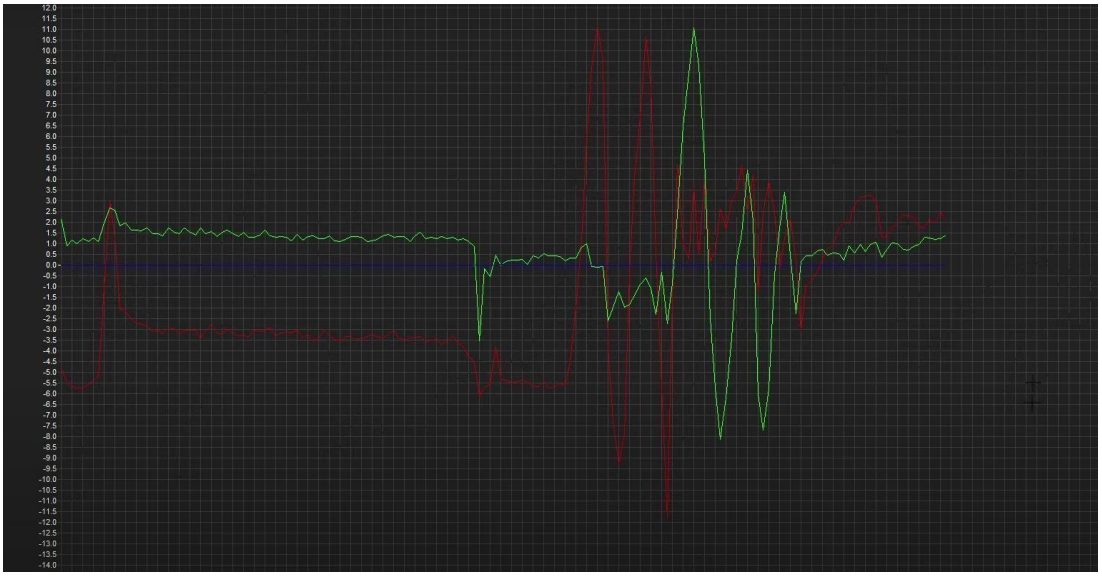
Dynamic Adaptation: The Kalman filter adapts to changes in the system and the measurement characteristics by continuously updating its estimates and adjusting the blending of information based on the measurements' reliability.

Robustness: The Kalman filter is robust to noise and disturbances in both the measurements and the system dynamics. It can effectively filter out noise and provide accurate estimates even in the presence of significant disturbances.



4.18 Degrees v Time Results of the Complementary Filter applied on IMU readings. Pitch (Green) and Roll (Red). Complementary coefficient – 0.02

4.19 Degree v Time Results of the Kalman Filter applied on IMU readings. Pitch (Green) and Roll (Red). First Order Low Pass Filter: 0.01 Gyro Alpha 0.1 Accel Alpha



4.20 Degree v Time Results of the Kalman Filter applied on IMU readings. Pitch (Green) and Roll (Red). First Order Low Pass Filter: 0.1 Gyro Alpha 0.5 Accel Alpha

4.11 NMEA GPS Standard:

The NMEA GPS standard (NMEA 0183) defines several types of sentences, also known as data messages, that convey specific information related to navigation and positioning. These sentences are transmitted via the NMEA protocol and allow different marine electronic devices to exchange data seamlessly. Following are some of the commonly used NMEA GPS sentences [W4]:

1. GGA (Global Positioning System Fix Data): The GGA sentence provides essential information about the GPS receiver's position and fix quality. It includes data such as latitude, longitude, altitude, time, and the number of satellites in view.

Example GGA sentence:
$GPGGA,123519,4807.038,N,01131.000,E,1,08,0.9,545.4,M,46.9,M,,*47

2. RMC (Recommended Minimum Navigation Information): The RMC sentence provides essential navigation data, including the GPS receiver's position, velocity, and time. It also indicates the status of the fix and the mode of operation.

Example RMC sentence:
$GPRMC,123519,A,4807.038,N,01131.000,E,022.4,084.4,230394,003.1,W*6A

3. GSA (GPS DOP and Active Satellites): The GSA sentence provides information about the GPS receiver's overall satellite fix and the Dilution of Precision (DOP) values. It includes details about the satellites being used for the fix and their relative contribution.

Example GSA sentence:

$GPGSA,A,3,04,05,,09,12,,,24,,,,,2.5,1.3,2.1*39

4. GSV (GPS Satellites in View): The GSV sentence reports information about the satellites in view of the GPS receiver. It provides data on the total number of satellites in view, as well as details about each satellite's identification, elevation, azimuth, and signal strength.

Example GSV sentence (part 1):
$GPGSV,3,1,11,01,40,083,46,02,19,308,41,06,19,001,48,07,06,266,44*71

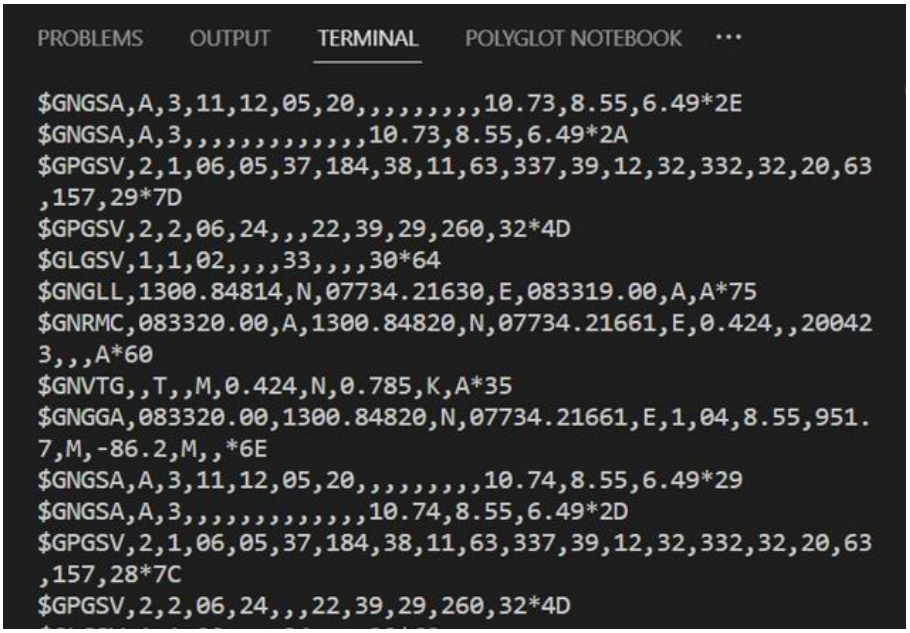
5. VTG (Course Over Ground and Ground Speed): The VTG sentence provides information about the vessel's course over ground (COG) and ground speed. It includes data on the true heading, magnetic heading, ground speed, and speed units.

Example VTG sentence: $GPVTG,054.7,T,034.4,M,005.5,N,010.2,K*48

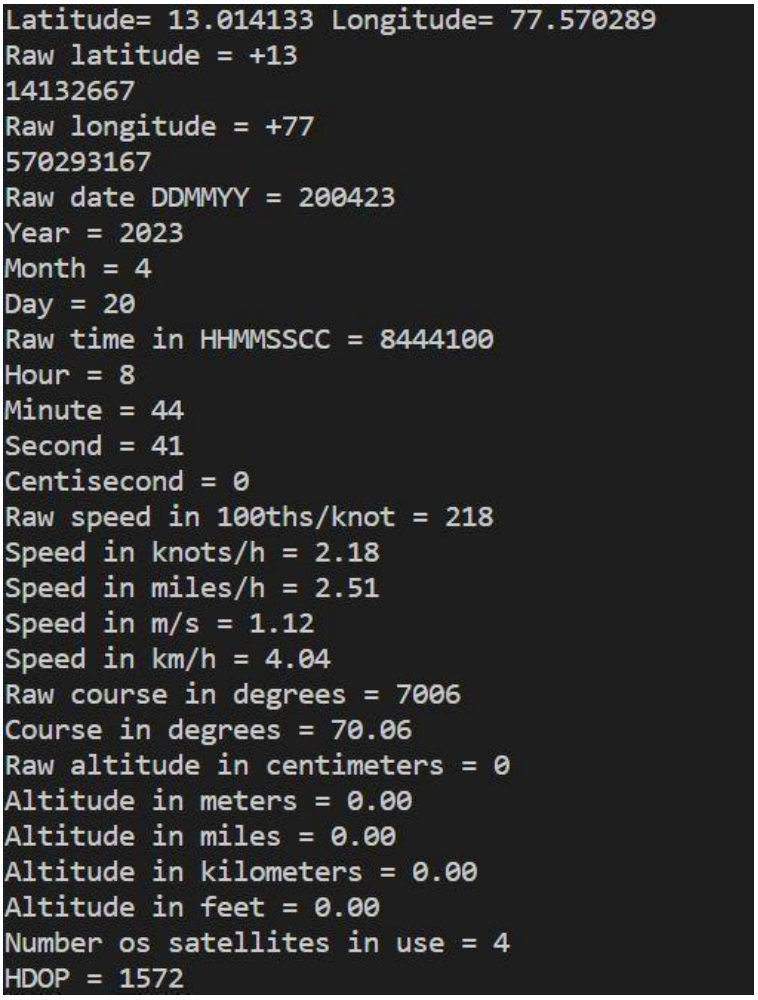
6. GLL (Geographic Position - Latitude/Longitude): The GLL sentence reports the current geographic position, including latitude and longitude, along with the time of the fix. It does not provide information about the fix quality or the number of satellites in view.

Example GLL sentence: $GPGLL,4916.45,N,12311.12,W,225444,A,*1D

These are just a few examples of the NMEA GPS sentences commonly used in marine navigation systems. Each sentence contains specific information that facilitates accurate positioning, course calculation, and other navigation-related tasks.



4.21 NEO-6M GPS Data



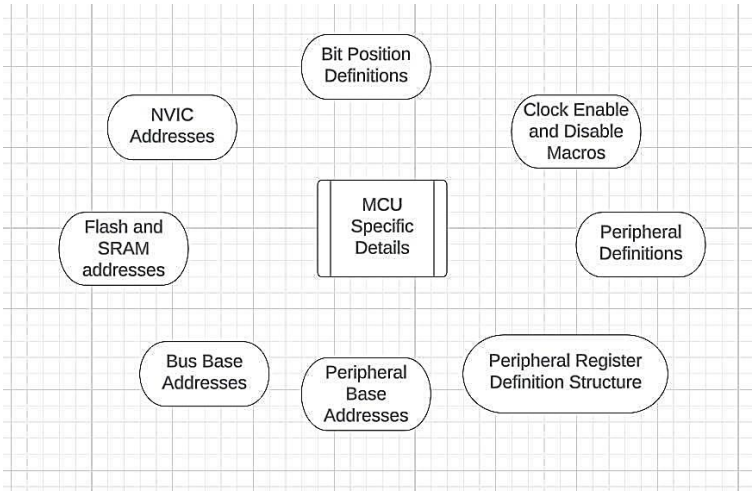
4.22 Translation of NMEA Sentences

4.12 Embedded Driver Development:

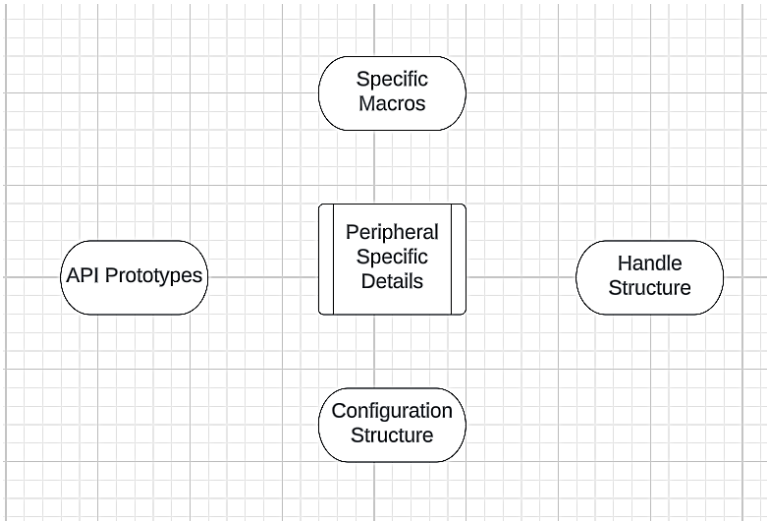
Embedded driver development is a critical aspect of software development for embedded systems. It involves the creation of software drivers that enable the system to communicate with various hardware components, such as sensors, actuators, and other peripheral devices. Without proper driver development, embedded systems may not be able to function correctly or may experience stability issues, which can lead to system failures and safety hazards.

The importance of embedded driver development can be seen in a variety of applications, including UAVs, automotive systems, medical devices, and industrial control systems. In these applications, the embedded system must communicate with a range of sensors, motors, and other devices, and the drivers must be designed to handle the specific requirements of each device. The drivers must also be optimized for performance and efficiency, as embedded systems often have limited processing power and memory resources.

Embedded driver development is a complex and challenging process that requires specialized knowledge and skills. Developers must have a deep understanding of the hardware components, as well as the software programming languages and tools used for embedded systems. They must also be able to debug and troubleshoot driver issues, which can be challenging in complex systems.



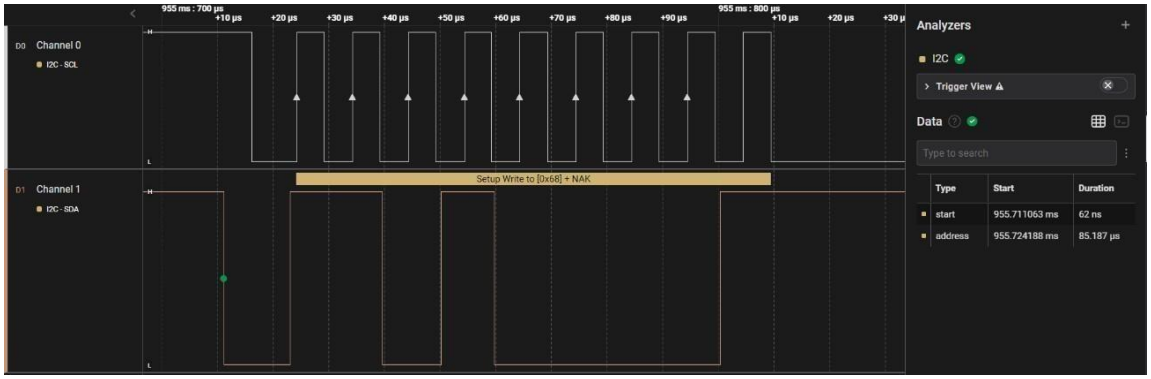
4.23 Inclusions in MCU Header File



4.24 Inclusions in Peripheral Specific Header File

4.25 GPIO Test

4.26 SPI Transmission Testing



4.27 I2C Transmission Testing

4.13 Micro Drones:

Micro drones, also known as miniature drones or nano drones, are small unmanned aerial vehicles (UAVs) that are characterized by their compact size and lightweight construction. These drones are designed to be highly portable, manoeuvrable, and capable of performing a variety of tasks in both indoor and outdoor environments.

The size of micro drones can vary, but they are typically small enough to fit in the palm of a hand or even smaller. They are equipped with miniaturized components, including motors, sensors, cameras, and batteries, which enable them to fly and carry out specific functions.

One of the key advantages of micro drones is their versatility. Due to their small size, they can access areas that are challenging or impossible for larger drones or humans to reach. They can navigate through tight spaces, fly close to objects, and even perform tasks in hazardous environments, such as search and rescue missions in collapsed buildings or monitoring dangerous chemical spills.

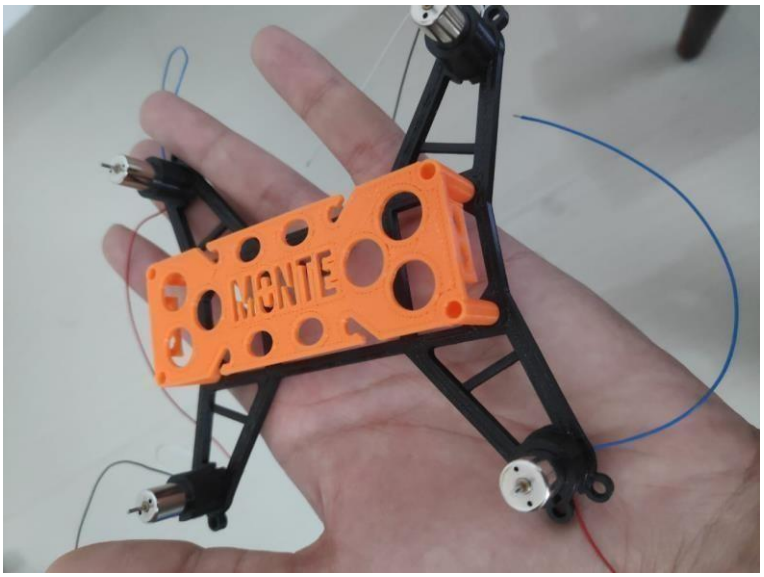
Micro drones find applications in various fields. In the military sector, they are used for reconnaissance, surveillance, and intelligence gathering. They can provide real-time video footage, capture images, and gather data from remote locations without endangering human lives. Law enforcement agencies also utilize micro drones for monitoring crowds, conducting aerial surveillance, and assisting in search operations.

Beyond military and law enforcement applications, micro drones have a range of civilian uses. They are employed in aerial photography and videography, allowing photographers and filmmakers to capture unique perspectives and stunning shots. Additionally, they have applications in agriculture for crop monitoring, in construction for site inspections, and in scientific research for studying wildlife and environmental monitoring.

Technological advancements have led to the development of sophisticated micro drones with enhanced capabilities. Some micro drones are equipped with obstacle avoidance systems, GPS navigation, autonomous flight modes, and advanced imaging technologies. These features enable them to fly autonomously, track targets, and perform complex missions with minimal human intervention.

However, the use of micro drones also raises concerns related to privacy and security. Their small size and maneuverability make it easier for them to go unnoticed or be used for

unauthorized surveillance. This has prompted the need for regulations and guidelines to ensure responsible and ethical use of micro drones.



4.28 Scale of Micro Drone

4.14 VL53 Series TOF Sensors:

The VL53 series of Time-of-Flight (ToF) sensors represents a breakthrough in distance and proximity measurement technology. Developed by STMicroelectronics, these sensors have gained significant popularity and have been widely adopted in various industries due to their exceptional performance, reliability, and compact design.

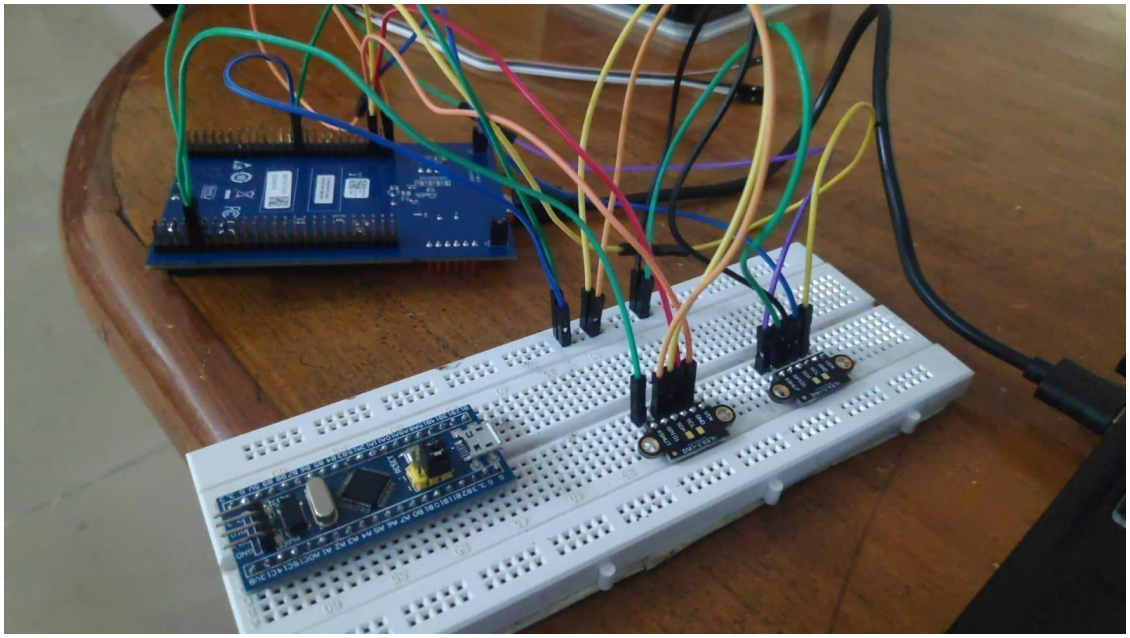
The VL53 sensors utilize a specialized technique known as Single-Photon Avalanche Diode (SPAD) array combined with advanced algorithms to accurately measure the distance of an object with millimeter precision. By emitting an invisible laser pulse and measuring the time it takes for the pulse to reflect back to the sensor, the VL53 sensors can determine the distance to the target, even in challenging environments.

One of the key advantages of the VL53 series is its high accuracy. The sensors can measure distances from a few millimetres up to several meters, offering precise and reliable results across a wide range of applications. Whether it's object detection, gesture recognition, robotics, or autonomous navigation, the VL53 sensors deliver consistent and dependable measurements, enabling enhanced functionality and performance.

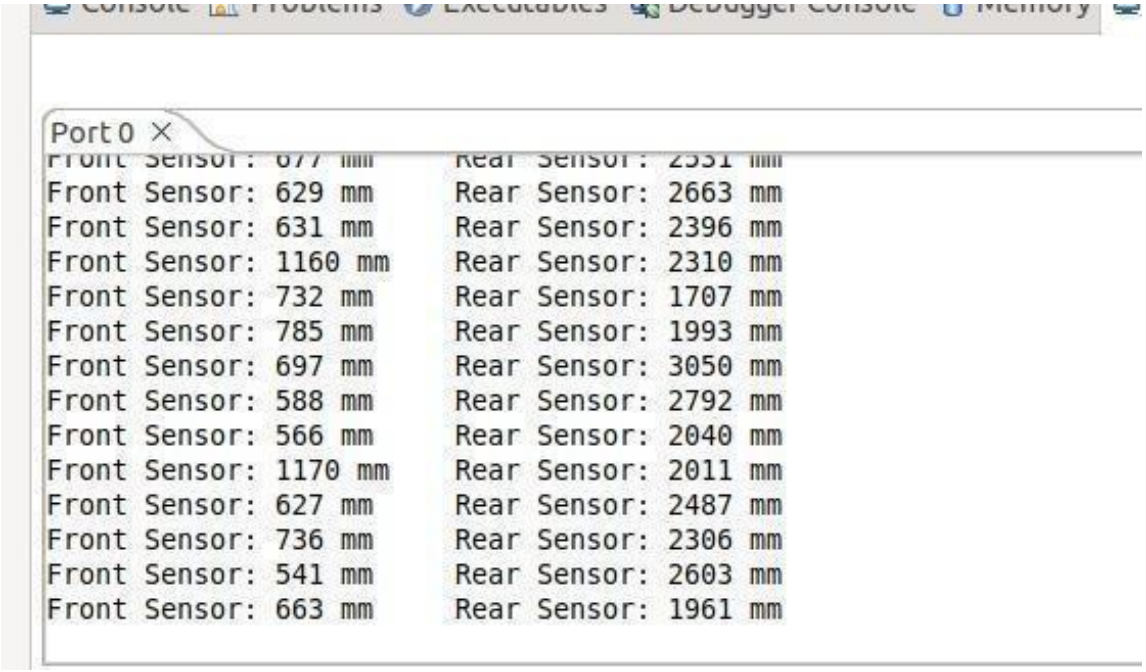
Speed is another remarkable feature of the VL53 series. These sensors can perform distance measurements in a fraction of a second, allowing for real-time monitoring and response. This rapid response time makes them suitable for applications that require quick and dynamic measurements, such as fast-moving objects or high-speed automation processes.

Additionally, the compact form factor of the VL53 sensors makes them easy to integrate into various devices and systems. With their small footprint and low power consumption, these sensors can be seamlessly integrated into smartphones, tablets, wearables, drones, robotics, and IoT (Internet of Things) devices. This versatility makes the VL53 sensors highly adaptable to a wide range of applications and industries.

Furthermore, the VL53 series offers a comprehensive set of features and functionalities to cater to different application requirements. These include multi-zone operation, signal filtering, ambient light suppression, and programmable thresholds. The sensors can also provide additional information such as signal strength and target presence detection, enabling developers to create sophisticated and intelligent applications.



4.29 TOF Sensors with STM32



4.30 ITM Output

4.15 ESP-NOW Protocol:

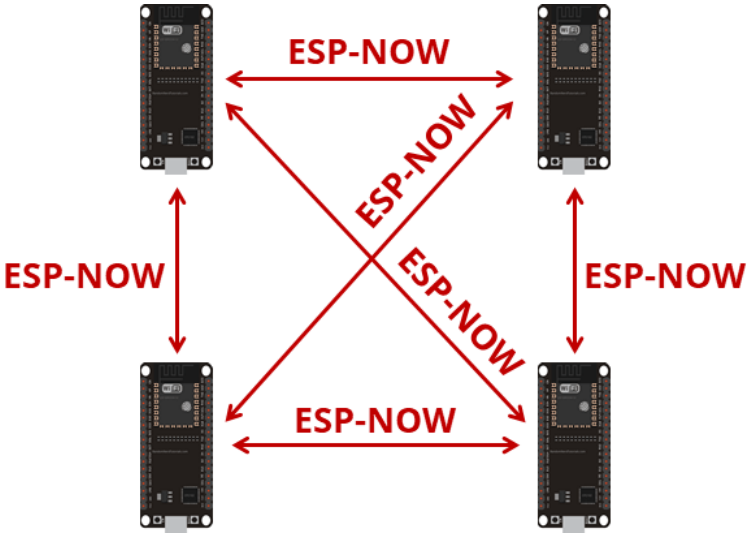
The ESP-NOW protocol is a communication protocol developed by Espressif Systems specifically for their ESP8266 and ESP32 series of microcontrollers. It provides a simple and efficient way to establish wireless communication between ESP-NOW-enabled devices, making it ideal for applications requiring low-power, low-latency, and peer-to-peer communication.

ESP-NOW operates in the 2.4 GHz frequency band and utilizes the Wi-Fi hardware of ESP8266 and ESP32 chips to establish a direct link between devices without the need for a Wi-Fi access point or router. This peer-to-peer communication eliminates the complexities associated with traditional Wi-Fi networks, simplifying the implementation and reducing power consumption.

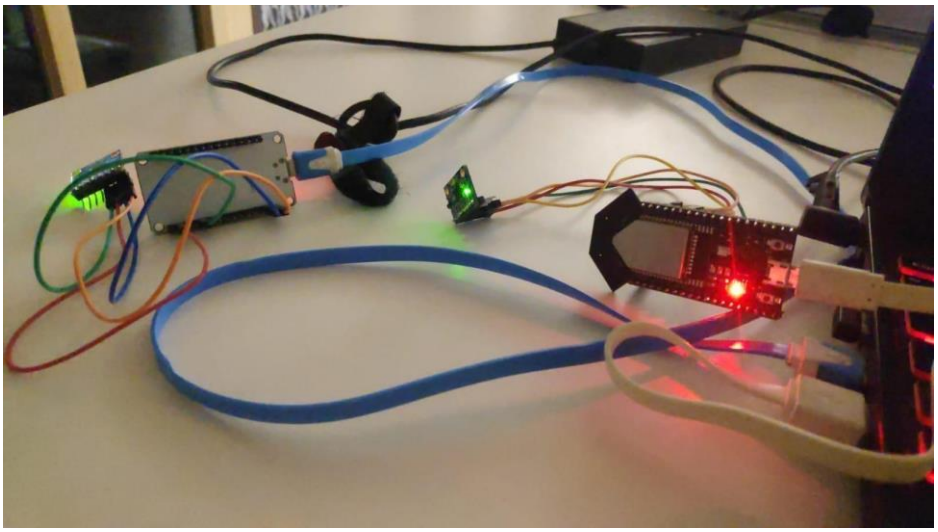
One of the key advantages of ESP-NOW is its low-latency communication. The protocol is designed for near-real-time data exchange, making it suitable for applications that require quick response times, such as sensor networks, home automation, remote control systems, and IoT devices. The minimal overhead and optimized data transmission enable rapid and efficient data transfer between devices.

Another notable feature of ESP-NOW is its low-power operation. The protocol is designed to minimize power consumption, allowing devices to operate on battery power for extended periods. By leveraging the sleep modes and power-saving capabilities of the ESP8266 and ESP32 chips, ESP-NOW devices can conserve energy while still maintaining a reliable communication link.

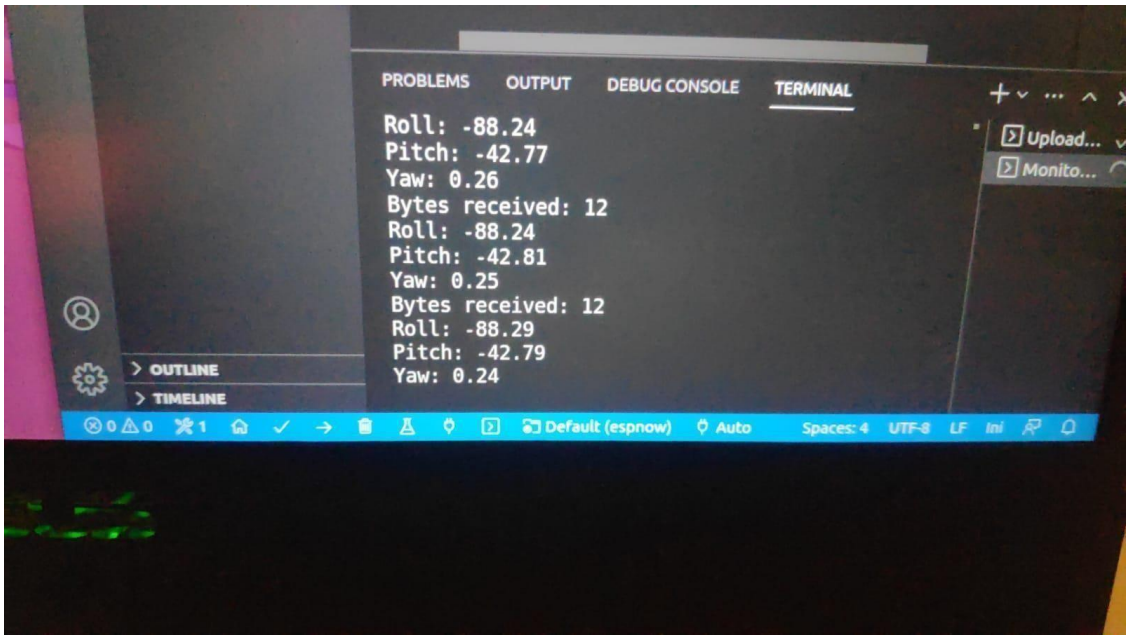
Furthermore, ESP-NOW supports multicast and broadcast communication. This enables a device to send data to multiple recipients simultaneously, simplifying scenarios where multiple devices need to receive the same data or command. The multicast and broadcast capabilities of ESP-NOW enhance scalability and flexibility in applications with multiple nodes or devices. [W5]



4.31 One possible ESP-NOW Configuration



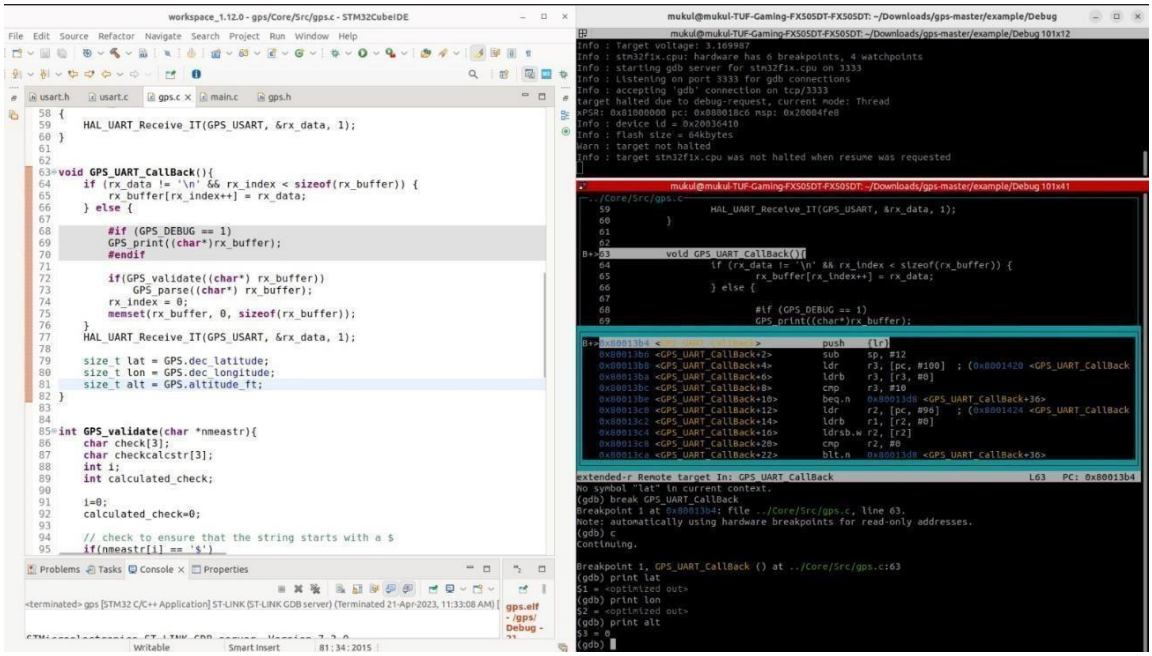
4.32 Pair of ESP32s sending IMU data back and forth



4.33 IMU Data Wirelessly Transmitted

4.16 OpenOCD:

Debugging using OpenOCD (Open On-Chip Debugger) is a powerful and versatile approach to hardware debugging and programming of embedded systems. OpenOCD is an open-source project that provides a debugging and programming interface for a wide range of microcontrollers and processors.



4.34 OpenOCD Debugging Interface

4.17 NuttX RTOS:

NuttX is a real-time operating system (RTOS) designed to provide a lightweight, scalable, and reliable platform for embedded systems. It is open-source software released under the permissive BSD license, making it accessible and customizable for a wide range of applications. [W6]

Key Features of NuttX:

1. Small Footprint: NuttX is designed to be highly efficient in terms of both memory usage and execution speed. It has a small footprint, typically requiring as little as 10KB of RAM and 50KB of ROM, making it suitable for resource-constrained devices.

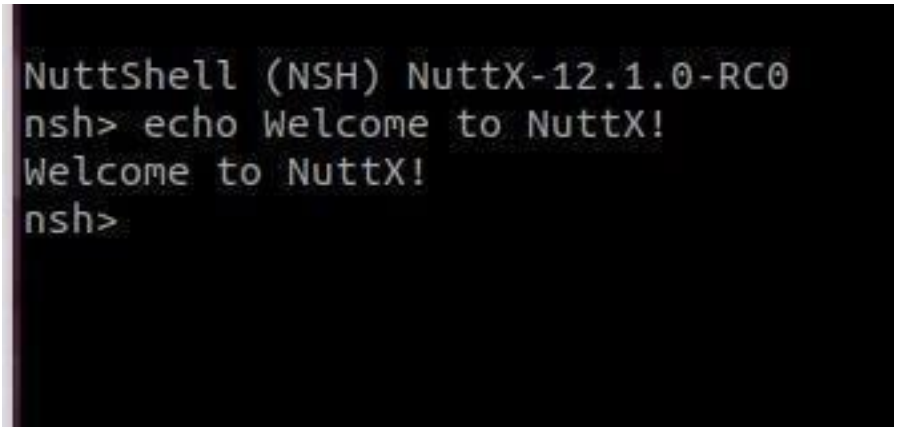
2. POSIX Compliance: NuttX aims to provide a high degree of compatibility with the Portable Operating System Interface (POSIX) standard. This allows developers familiar with POSIX APIs to easily port and develop applications for NuttX.

3. Real-Time Capabilities: As an RTOS, NuttX provides real-time scheduling and deterministic behavior. It offers a priority-based preemptive scheduler, allowing critical tasks to execute within defined time constraints. It supports features like task scheduling, inter-task communication, and synchronization mechanisms, such as semaphores and message queues.

4. Modularity and Extensibility: NuttX follows a modular architecture, where functionalities are organized into individual components. This modular design allows developers to select and include only the necessary components, reducing the overall memory footprint. Additionally, NuttX supports loadable kernel modules, enabling the dynamic addition of new features without requiring a complete recompilation.

5. Device Support: NuttX provides support for a wide range of microcontrollers and microprocessors, including ARM, MIPS, PowerPC, and more. It offers device drivers for various hardware peripherals like UARTs, SPI, I2C, Ethernet, USB, and file systems. This broad device support makes it versatile and applicable to diverse embedded systems.

4.35 Building NuttX RTOS



4.36 NuttX Shell

Process to Build NuttX for any Application:

1. Select Target Hardware: Determine the target hardware platform for your application. NuttX supports a wide range of microcontrollers and microprocessors. Make sure to choose a platform that is compatible with NuttX and suitable for your application's needs.

2. Choose the Configuration File: NuttX provides a configuration file for each supported hardware platform. Locate the appropriate configuration file for your target platform. These files are typically found in the **configs** directory of the NuttX source code.

3. Customize the Configuration: Open the configuration file for your target platform and

modify the settings according to your application requirements. The configuration file contains various options and macros that can be adjusted. Some common configuration options include:

- System Clock: Set the system clock frequency to match your hardware configuration.
- Memory Configuration: Specify the available RAM and ROM sizes.
- Scheduler and Task Configuration: Configure the task scheduling algorithm, priority levels, and stack sizes.
- Device Drivers: Enable or disable the device drivers required for your hardware peripherals.
- File Systems: Choose the appropriate file system support for your application.
- Networking: Enable or disable the networking stack and protocols as needed.
- Real-Time Features: Configure the real-time capabilities, such as timers and synchronization mechanisms.

4. Build and Flash NuttX: Once you have customized the configuration file, build NuttX using the provided build system (usually based on Makefiles or CMake). Ensure that the build process includes your modified configuration file. After a successful build, you will have an executable image ready to be flashed onto the target hardware.

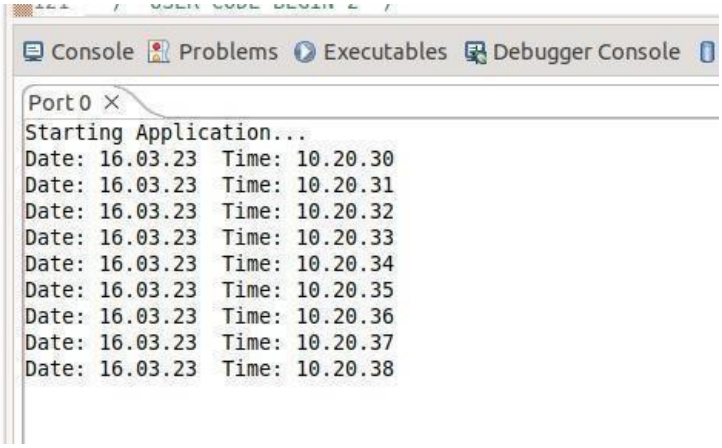
4.18 Real Time Clock:

The STM32 features a Real Time Clock peripheral. The RTC provides accurate timekeeping and calendar functions, making it useful in applications where keeping track of time is essential. [W7]

To use the STM32's RTC, the following steps are typically involved:

1. RTC Initialization: Configure the RTC module by setting the appropriate registers and enabling the necessary functionalities. This includes setting the clock source, enabling calendar mode, configuring alarms, and adjusting calibration settings.
2. Time and Date Setting: Set the initial time and date values for the RTC. This involves configuring the hours, minutes, seconds, as well as the day, month, year, and day-of-the-week.

3. Alarm Configuration: If needed, configure the alarm(s) by specifying the desired time at which the alarm should trigger.
4. Interrupt or Event Handling: Configure interrupt or event handlers to respond to RTC events, such as alarm triggers or tamper detection. These handlers can perform specific actions or wake up the system from low-power modes.
5. Time Retrieval and Updates: Read the current time and date values from the RTC registers whenever required. Update the RTC with the current time periodically or as needed.
6. Backup Power Supply: Connect a backup battery or an external power source to the RTC's backup power domain to ensure continuous timekeeping during power disruptions.
7. Calibration: Optionally, perform RTC calibration procedures to adjust the clock frequency and maintain accurate timekeeping.



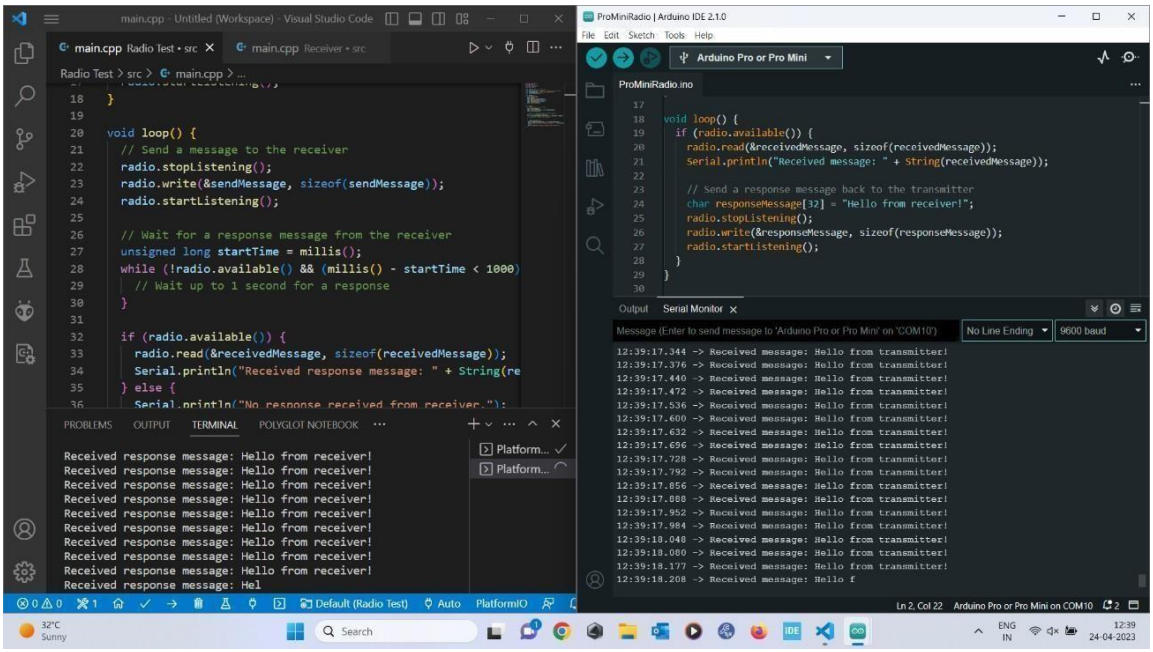
4.37 RTC Implementation

4.19 NRF24L01 Radio:

The NRF24L01 is a popular and versatile radio frequency (RF) module used for wireless communication in various applications. Manufactured by Nordic Semiconductor, the NRF24L01 module offers reliable and low-power communication capabilities, making it suitable for battery-operated devices and low-power wireless applications. [W8]

To use the NRF24L01 radio module, the following steps are typically involved:

1. Hardware Connection: Connect the NRF24L01 module to the host microcontroller using the SPI interface. Ensure that the necessary power supply and signal lines are properly connected. Configuration: Initialize the NRF24L01 module by configuring various settings, such as data rate, channel, addressing mode, and power-saving features. This is usually donethrough SPI commands and configuration registers.
2. Data Transmission: Prepare the data to be transmitted and send it using the appropriate NRF24L01 commands and functions. The module handles packetization, error checking, and retransmission automatically.
3. Data Reception: Set up the NRF24L01 module to receive data by configuring the receiver mode and address. Retrieve the received data from the module's receive buffer using the host microcontroller.
4. Error Handling and Acknowledgment: Handle potential transmission errors by checking the module's status registers and utilizing the automatic acknowledgment feature. This ensures reliable data transfer and can trigger retransmission if necessary.
5. Power Management: Utilize the power-saving features of the NRF24L01 module to optimize power consumption. Enable sleep modes during idle periods and wake the module when necessary to conserve energy.
6. Testing and Optimization: Test the wireless communication between the NRF24L01 module and other devices to ensure reliable operation. Fine-tune parameters such as channel selection, data rate, and antenna configuration for optimal performance in the specific application environment.



4.38 Two-Way Radio

CHAPTER 5
CONCLUSIONS AND SCOPE FOR FUTURE WORK

The engineering process is a crucial framework that aligns with the development of embedded software. Embedded systems refer to the integration of hardware and software components into a dedicated function or application. As such, the engineering process provides a systematic approach to designing, developing, and maintaining software for these embedded systems, ensuring efficient and reliable functionality.

The first step in the engineering process is requirements analysis. This phase involves understanding the objectives and constraints of the embedded software project. Engineers gather and analyze the functional and non-functional requirements to define the scope and purpose of the software. By clearly defining these requirements, engineers can establish a solid foundation for the subsequent stages of development.

Following requirements analysis, the system design phase begins. This step involves architecting the software system, including selecting appropriate hardware components and determining the software architecture. Engineers consider factors such as system performance, power consumption, memory utilization, and real-time constraints during this phase. They create a detailed design that maps out the structure, interfaces, and behavior of the software, ensuring it aligns with the overall embedded system's objectives.

Once the system design is in place, the implementation phase commences. Engineers start coding the software based on the design specifications. They use programming languages and tools suitable for embedded systems, taking into account resource limitations, real-time requirements, and hardware integration. Thorough documentation and modular coding practices are essential to facilitate maintenance and future updates.

Testing is a critical aspect of the engineering process in embedded software development. Engineers perform various tests, including unit testing, integration testing, and system testing, to verify the software's functionality, performance, and reliability. Testing helps identify and rectify software defects, ensuring that the embedded system operates as intended.

Finally, the deployment phase involves the installation and configuration of the embedded software on the target hardware.

REFERENCES

Journals/ Research Articles:

[1] Perez, D., & Blank, D. (2017). Drones: The Good, the Bad, and the Unknown. In The Drone Debate (pp. 3-18). Routledge.

[2] Tomic, T., & Schiller, S. (2018). A UAV Revolution: A Socio-Technical Analysis of the Drone Industry. In Human Aspects of IT for the Aged Population. Applications in Health, Assistance, and Entertainment (pp. 237-249). Springer.

[3] Palanisamy, K., & Veeraraghavan, A. (2018). Drones for Smart Cities: Issues in Cyber-Physical Systems. IEEE Internet of Things Journal, 5(3), 1923-1933.

[4] Giannakos, I., Diakakis, J., & Vozikis, A. (2020). Integrating drone and wireless sensor network technologies for smart precision agriculture: System design and performance evaluation. Sustainable Cities and Society, 61, 102334.

[5] Teizer, J., Cheng, T., Fang, Z., & Scherer, R. (2019). Adrone safety index to benchmark safety performance for US construction projects. Journal of Management in Engineering, 36(2), 04019077.

[6] Ahad, M. A. R., & Kadir, A. (2019). A Review of Unmanned Aerial Vehicle for Search and Rescue Operation. IOP Conference Series: Earth and Environmental Science, 260(1), 012002.

[7] Kreps, S., & Kaag, J. (2019). Drones and the Future of Armed Conflict: Ethical, Legal, and Strategic Implications. International Security, 43(4), 7-73.

[8] Kreps, S., & Kaag, J. (2019). Drones and the Future of Armed Conflict: Ethical, Legal, and Strategic Implications. International Security, 43(4), 7-73.

[9] Calo, R. (2013). The drone as privacy catalyst. Stanford Law Review Online, 64, 29-33.

[10] Tomic, T., & Schiller, S. (2018). AUAV Revolution: A Socio-Technical Analysis of the Drone Industry. In Human Aspects of IT for the Aged Population. Applications in Health, Assistance, and Entertainment (pp. 237-249). Springer.

[11] Bonney, J. A. (2020). Policy Implications ofthe Use of Drones for Environmental Research. In Ecological Implications of Drone Ecology (pp. 265-285). Springer.

[12] Valavanis, K. P. (2017). Handbook of unmanned aerial vehicles. Springer.

[13] Yang, H., Xia, X., & Fu, M. (2016). Nonlinear flight control for small-scale unmanned aerial vehicles. Control Engineering Practice, 50, 45-56.

[14] Raffo, G., Quang Do, M., Castillo, P., & Lozano, R. (2017). Robust controlof UAVs: A flight control perspective. Annual Reviews in Control, 44, 294-308.

[15] Elbouchikhi, E., & Veluvolu, K. C. (2020). Nonlinear model-based control for quadrotor unmanned aerial vehicles: A survey. Nonlinear Dynamics, 101(1-2), 97-120.

[16] Ding, Y., Liu, J., Ding, Z., Li, H., & Na, W. (2018). An optimal control approach for trajectorytracking of unmanned aerial vehicles. Aerospace Science and Technology, 80, 110-117.

Images:

Layers of Embedded Software: https://www.microcontrollertips.com/how-does-embedded-software-work/

OpenOCD with STM32: https://elrobotista.com/en/posts/stm32-debug-linux/

Pixhawk and QGroundControl: https://www.youtube.com/watch?v=BNzeVGD8IZI&t=556s

Flight Controller:

https://www.unmannedtechshop.co.uk/product/foxeer-f722-flight-controller/

https://www.fruugoindia.com/fast-delivery-fast-delivery-pixhawk-px4-autopilot-pix-248-32-bit-flight-controller-safety-switch-buzzer-4g-sd-i2c-splitter-expand-module-usb/p-154687319-327653921?language=en

Flysky Transceiver: https://www.youtube.com/watch?v=BACBNgaCnJU&t=1572s

Complementary Filter: https://www.researchgate.net/figure/Complementary-Filter-anglet-a-anglet-1-gyro-dt-b-accelero-1_fig2_265915080

Kalman Filter: https://commons.wikimedia.org/wiki/File:Kalman-filter_en.svg

ESP-NOW Protocol: https://randomnerdtutorials.com/esp-now-esp32-arduino-ide/

Websites:

[W1] High current PCB Design: https://www.ourpcb.com/high-current-pcb.html

[W2] MPU6050 Visualiser: https://randomnerdtutorials.com/esp32-mpu-6050-web-server/

[W3] Filters: https://robottini.altervista.org/kalman-filter-vs-complementary-filter

[W4] NMEA GPS: https://www.gpsworld.com/what-exactly-is-gps-nmea-data/

[W5] ESP-NOW: https://randomnerdtutorials.com/esp-now-esp32-arduino-ide/

[W6] NuttX RTOS: https://nuttx.apache.org/docs/latest/

[W7] STM32 RTC: https://embedded-lab.com/blog/stm32s-internal-rtc/

[W8] NRF24L01: https://howtomechatronics.com/tutorials/arduino/arduino-wireless-communication-nrf24l01-tutorial/

Annexure 1

PO &PSO Mapping

Student Name: Mukul Yadav Register no: 190929042

Note: use a tick mark if you have addressed that PO in your report

PO	✓ Tick	Pg. No	Section No	Guides Observation
PO1	✓	14-49	4	
PO2	✓	6-7	2	
PO3	✓	1	1.1	
PO4	✓	31-34	4.10	
PO5	✓	8-13	3	
PO6				
PO7	✓	1	1.1	
PO8				
PO9	✓	i	i	
PO10	✓	50	5	
PO11	✓	14-49	4	
PO12	✓	50	5	

PSO	✓ Tick	Pg. No	Section No	Guides Observation
PSO1	✓	14-49	4	
PSO2	✓	8-49	3,4	
PSO3				

Signature of Student: Name and Signature of Guide:

Date: 12/07/23 Dr. Nikhil Pachauri

Annexure 2

PLO Mapping

Student Name: Mukul Yadav Register no: 190929042

Sl	PLO	✓ Tick	Pg. No	Section No	Guides Observation
1	C1.	✓	14-49	4	
2	C2.	✓	6-7, 14-49	2,4	
3	C3.	✓	8-13	3	
4	C4.	✓	6-7	2	
5	C5.				
6	C6.	✓	14-49	4	
7	C7.	✓	1-5	1	
8	C8.	✓	1	1.1	
9	C9.				
10	C10.	✓	14-49	4	
11	C11.	✓	14-49	4	
12	C12.	✓	8-13	3	
13	C13.	✓	8-49	3,4	
14	C14.	✓	50	5	
15	C15.	✓	14-49	4	
16	C16.	✓	14-49	4	
17	C17.	✓	14-49	4	
18	C18.				

Note: use a tick mark if you have addressed that LO in your report

Signature of Student: Name and Signature of Guide:

Date: 12/07/23 Dr. Nikhil Pachauri

Annexure 3

Address of IET learning outcomes during project period

Answer the following questions with relevant to your Practice Scool work.

1. Explain the steps you followed to Investigate and define the problem in your project work (C4, evaluate level)
   - Understand the basics of UAVs.
   - Use common hardware and tools used in that domain.
   - Revisit the principles of embedded development.
   - Tryto apply those principles from the ground up to UAV applications.

2. What is the science, mathematics, statistics, engineering principles and other basic technology you identified for design (Mechanical, Electronic, Physics, Chemistry, Automation) in your project work? (C1, C2, C3, Application, Analysis, Evaluation of Science and Mathematics in the project)
   - Mechanical design in 3D printing
   - Electronic circuit design
   - Automation using embedded software

3. Have you considered the Environmental and sustainability limitations in your project work? (C7, evaluate)
   - There is an implicit consideration of the environment and sustainability while discussing the current relevance of UAVs.

4. Have you considered ethical dilemma, health, safety, security, and risk issues; intellectual property; codes of practice and standards? Did you address any of these issues in your project work? If so, Explain in detail. (C5, create)
   - UAVs have many ethical and social dilemmas accompanying them. Since they have a wide range of applications, it is important to design safeguards for that specific application.

5. What were the aesthetic issues faced and how it is addressed in your project in the design phase? (C5, analysis)
   - The design of UAV components and the form factor of custom PCBs affect aesthetics and had to be considered.

6. Were there any health issues considered during design process. How it is addressed in your project in the design phase? (C5, create)
   - No specific health issues were encountered during the process.

7. What were the safety, security and risk issues needed to be taken care of in the design stage? (C10, create)
   - There was some light deliberation about the design of a practical Black Box System for a UAV.
   -

8. Were there any intellectual property issues needed to be taken care off? Have you come across IP issues in the project phase? (C5, create)
   - No IP issues were encountered in the project phase.

9. What are the codes of conduct and standards you needed to use in design phase and in other phases of your project as well? (It may include codes of practice and standards for safety, security, health, risk) Explain the legal issues, ISO standards, IEC standards, etc. (C8, evaluate)
   - No standards were applied because of the nature of the work, but theoretical discussions regularly reinforced the Drone Laws of India.

10. What is the general safety measure regulated in the industry where you did the project work? (C8, evaluate)
    - The software component of the products in this industry has to be highly secure to prevent hacking and misuse.

11. What were the professional ethics needed to be followed in general while you are doing the project? (C8, evaluate)
    - Punctuality
    - Documentation
    - Weekly Meets

12. Do you think ethics and professionalism needs to be paid attention by students during study? If, yes, explain how it can be inculcated/introduced/implemented? (C8, evaluate)
    - Ethics and professionalism will be beneficial to students in their future careers. The enforcement of the same lies with the management.

13. Do you think environmental and sustainability limitations; ethical, health, safety, security, and risk issues; intellectual property; codes of practice and standards are sufficiently covered in the courses you have studied in your curriculum? (C8, evaluate)
    - The above-mentioned issues have been discussed, although concentrated within a handful of subjects.

14. Have you gone through online classes, or a crash course in which you are familiarized with intellectual property rights as well as risk issues in professional environment? (C8, evaluate)
    - On my own, I have viewed a video series covering IP laws.

15. In the beginning of your project did you evaluate environmental effects and sustainability factors in your work? (C7, evaluate)
    - No such specific discussions were carried out.

16. During your stay in the industry, have you realized the need for professional and ethical conduct? Quote the context and explain. (C8, evaluate)
    - Competency in work
    - Clear communication with higher ups
    - Punctuality

17. What are the professional codes of conduct you needed to imbibe during your stay in the industry? (C8, evaluate)

- Competency in work
- Clear communication with higher ups
- Punctuality

18. Did you address any of limitations of your project work and have you improved the results through continuous improvements in your project work? (C5, create)
    - The work carried out is not suitable for industry grade applications. The purpose of the work carried out is to learn and develop skills.

19. How did you plan your project, deadlines, maintaining dairy of each stage and improved the quality of the project (C14, understand)
    - The project was planned by the management. The schedule was not harsh in any manner. It was easy to adhere to deadlines.

20. While having Industrial training/ internship, what were the college practices which helped you to abide by the professional ethics of the company environment? (C8, evaluate)
    - No specific practices to be cited, but the experience in the college aided confidence.

21. During your stay in the industry, did you observe how the teamwork plays a role in engineering process? (C16, apply)
    - Yes, specialization is extremely important for a successful team.

22. Are you aware of the ethical clearance when you work in the field of health/medical applications.? (C8, evaluate)
    - Not relevant to the work done.

23. During your stay in industry, are you able to observe and understand certain management techniques practiced in that industry. Explain in detail. (C14, understand)
    - Apart from weekly meetings, no management techniques can be cited.

24. Could you understand how they tackle project management and what tools and techniques is adopted? (C14, understand)
    - Platforms such as Slack and Kredily are used for managing projects. Tasks are created by the management and assigned to everyone.

25. During your stay in the industry, did you observe any engineering activity implemented to promote sustainable development? (C7, evaluate)
    - Not applicable here.

26. Did you adopt any quantitative technique for any engineering activity related to your project? (C3, evaluate)
    - Not applicable here.

27. What are the elements of your project work which addresses sustainable development and were you able to apply quantitative techniques to analyze and achieve your project goals? (C7, evaluate)
    - Not applicable here.

28. How the company takes green initiative, environment related factors. (C7, evaluate)
    - Not applicable here.

29. During your stay in the industry, have you observed/sensitized about legal requirements governing such activities in that industry? Explain. (C8, evaluate)
    - There are plenty of legalities required to establish and operate a UAV business. This is a highly sensitive domain since misuse is incredibly common. The relevant parties need to be aware of UAV laws of the territory.

30. Did your project need the understanding of relevant legal requirements governing engineering activities you carried out as a part of your project work? Explain in detail. (C8, evaluate)
    - The work done was rather elementary, meaning no such legal requirements were considered.
31. What are the legal, ethical practices you followed while working on project? (C8, evaluate)
    - Using existing software, one needs to be aware of the license afforded to the user by the author.The BSD license is one of the most unrestrictive licenses.

32. Are you sure that you abide IPR/copy right issues? (C15, apply)
    - Being aware of IP laws, it is reasonable to affirm this position.

33. Have you observed any national/international standards in the workplace? How many are relevant to your project work? List them. (C8, evaluate)
    - Not applicable here.

34. What online course you attended to improve your communication skills. Report writing, Oral presentation, Software used for writing report. (c17, apply)
    - Not applicable here.

35. In your project, was it needed to tackle risk issues, including health & safety, environmental and commercial risk, and of risk assessment and risk management techniques? Explain in detail. (C5, create)
    - Apart from some general discussions, issues related to the same did not come up.

36. What are the cyber safety rules and precaution you were sensitized with, when you started practice school, or started industrial training? (C9, evaluate)
    - When it comes to cyber safety, organizations often use proprietary softwares to ensure the integrity of their work. No such software was used here.

37. How is an organization addressing a fire accident/human safety when working with machines? (C9, evaluate)
    - Not applicable here.

38. Process of teamwork. How each of you are involved in the team? What part the work is addressed by you.? (C16, evaluate)
    - My role was testing and development of embedded software/ basic electronics.

39. Have you filed patent, IPR, or published your work? Give more details. (C17, evaluate)
    - Not applicable here.

40. How you documented the literature review, your analysis on their results, discussion with the guide and team members, provide the documents on weekly basis. Put as one chapter in final report. (C4, evaluate).
    - All documents have been consistently and sincerely handed to the internal guide.

41. Have you sensitized about inclusion and diversity in the team? If yes, what are the diversification in the team in terms of religion, gender, ethnicity, etc. What challenges you come across in the team. (C11, apply). Indian constitution and acts related to caste, gender, race discrimination.
    - Not applicable here.

42. How were you able to keep yourself updated with the technology? How you incorporated advanced technology in your project. (C18, lifelong learning)
    - Online courses and staying aware of industry news assist in this matter.

43. Which are the laboratory skills you found applicable to your project. Explain. (C12, apply)
    - Patience, unit testing and concentration are most applicable here.

Annexure 4

Project/practice school classification

Student Name: Mukul Yadav Register no: 190929042

Note: Use a tick mark to specify under which domain your practice school work falls into.
Table 1: classification based on project domain classification

Domain	✓ **Tick**
Product	
Application	✓
Review	
Research	
Management	

Note: Use a tick mark to specify Societal impacts you considered during your practice school.
Table 2: classification based on societal consideration

Societal Impact	✓ **Tick**
ethics	✓
safety	✓
environmental	
commercial	✓
economic	✓
social	✓

Signature of Student: Name and Signature of Guide:

Date: 12/07/23 Dr. Nikhil Pachauri

Annexure 5

Company Details

Student Details			
Student Name	**Mukul Yadav**		
Register Number	190929042	Section / Roll No	B/12
Email Address	mukuly.2001@gmail.com	Phone No (M)	8094021222
Practice School Details			
Title	**Electronics and Embedded Firmware for UAVs**		
Practice School start Date	10 Dec 2022	Practice School End Date	30 April 2023
Organization (Company) Details			
Organization Name	**AIR Labs, IISc Bangalore**		
Type of Organization **(Public Listed, Private, PSU, Govt, cooperative)**	**Govt.**		
Full postal address with **pin code**	CV Raman Rd, Bengaluru, Karnataka 560012		
Website address	https://iisc.ac.in/		
Name of the CEO of the Organization	-		

Supervisor Details			
Supervisor Name	**Varun Raghavendra**		
Designation	Technical Associate		
Full contact address with pin code	AIR Labs, CV Raman Rd, Bengaluru, Karnataka 560012		
Email address	varuncr@iisc.ac.in	Phone No (M)	96321 37643
Internal Guide Details			
Faculty Name	**Dr. Nikhil Pachauri**		
Full contact address with pin code	Department of Mechatronics, Manipal Institute of Technology, Manipal – 576 104 (Karnataka State), INDIA		
Email address	nikhil.pachauri@manipal.edu		

project

ORIGINALITY REPORT

7%	4%	1%	5%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	Submitted to Manipal Academy of Higher Education (MAHE) Student Paper	1%
2	Submitted to Manipal University Student Paper	<1%
3	homepage.eircom.net Internet Source	<1%
4	Submitted to NIIT University Student Paper	<1%
5	Submitted to Universidad Carlos III de Madrid Student Paper	<1%
6	Submitted to Cranfield University Student Paper	<1%
7	irm.am.szczecin.pl Internet Source	<1%
8	promicro.tech.blog Internet Source	<1%
9	Submitted to Üsküdar Üniversitesi Student Paper	<1%

10	Submitted to Cankaya University Student Paper	<1%
11	www.nymtc.org Internet Source	<1%
12	Submitted to University of Queensland Student Paper	<1%
13	Submitted to Independent University Bangladesh Student Paper	<1%
14	Submitted to King's College Student Paper	<1%
15	Submitted to University of North Texas Student Paper	<1%
16	Submitted to Virginia Polytechnic Institute and State University Student Paper	<1%
17	Submitted to Cheshire College South and Wes Student Paper	<1%
18	Hanafy M. Omar, Rizwan Akram, Saad M.S. Mukras, Ahmed Alaa Mahvouz. "Recent advances and challenges in controlling quadrotors with suspended loads", Alexandria Engineering Journal, 2023 Publication	<1%

19	Submitted to University of Technology, Sydney Student Paper	<1%
20	dalspace.library.dal.ca Internet Source	<1%
21	Submitted to Associatie K.U.Leuven Student Paper	<1%
22	Submitted to Dhirubhai Ambani Institute of Information and Communication Student Paper	<1%
23	Submitted to University of Newcastle upon Tyne Student Paper	<1%
24	Jian Zhang, Lifeng Hao, Fan Yang, Weicheng Jiao, Wenbo Liu, Yibin Li, Rongguo Wang, Xiaodong He. "Biomimic Hairy Skin Tactile Sensor Based on Ferromagnetic Microwires", ACS Applied Materials & Interfaces, 2016 Publication	<1%
25	Submitted to Liverpool John Moores University Student Paper	<1%
26	Submitted to University of Bristol Student Paper	<1%
27	Submitted to Bloomington High School North Student Paper	<1%

28	Submitted to Frederick University Student Paper	<1%
29	site-7238175-2806-1438.mystrikingly.com Internet Source	<1%
30	Submitted to Central Queensland University Student Paper	<1%
31	bulletin.sfsu.edu Internet Source	<1%
32	www-d0.fnal.gov Internet Source	<1%
33	Henghui Deng, Fei Xie, Hebo Shi, Yufeng Li, Shuoyan Liu, Chaoqun Zhang. "UV Resistance, Anticorrosion and High Toughness Bio-based Waterborne Polyurethane enabled by a Sorbitan Monooleate", Chemical Engineering Journal, 2022 Publication	<1%
34	population.pwv.gov.za Internet Source	<1%
35	rsucon.rsu.ac.th Internet Source	<1%
36	www.airlive.com Internet Source	<1%
37	www.coursehero.com Internet Source	<1%

38	www.researchgate.net Internet Source	<1%
39	Amin Al-Habaibeh, Bubaker Shakmak, Anup Athresh, Keith Parker, Omar Hamza. "Chapter 20 Extracting Energy from Flooded Coal Mines for Heating and Air-Conditioning of Buildings: Opportunities and Challenges", Springer Science and Business Media LLC, 2022 Publication	<1%
40	Dogan Ibrahim. "Event groups", Elsevier BV, 2021 Publication	<1%

Exclude quotes	On	Exclude matches	Off
Exclude bibliography	On		