Autonomous Collaboration Between Underwater and Aerial Drones

Group 22

Antony Brittain, Nikhil Saroya
Ethan Wilson, Xinkai Xu

I. Introduction and Background

Background – Our client is working on a project with the final goal of developing a system that is capable of launching an aerial drone from a surfaced underwater vehicle, navigating the aerial drone to retrieve an object, landing the aerial drone back on the underwater vehicle, and doing this all autonomously. Before this system can be fully integrated, a reliable and robust communication system is needed to facilitate autonomous collaboration between the drones.

Statement of the Problem – The client would like to establish a communication method between underwater and aerial drones. The submersible drone must be capable of autonomously transporting an aerial drone to a desired location, and then launching it for the purpose of retrieving a remote object. The communication system should be robust enough to facilitate this degree of collaboration.

Objective – The goal of our project is to design a reliable and robust long range communication system that allows a surfaced underwater vehicle to exchange data with an aerial drone. The drones should be able to exchange simple information at a long range. The drones should also be able to exchange complicated information such as audio and video at close range. The system should handle challenges such as signal attenuation and latency while minimizing error rates. It must be simple to use and should be specialized to integrate into the drones. The communication between drone and the submersible will be done only when the submersible resurfaces. Therefore, communication interference caused by water bodies is not in the scope of this project. The designed transceivers will interface with the Raspberry Pi units on the submersible and the drone via a serial interface.

II. Design Overview

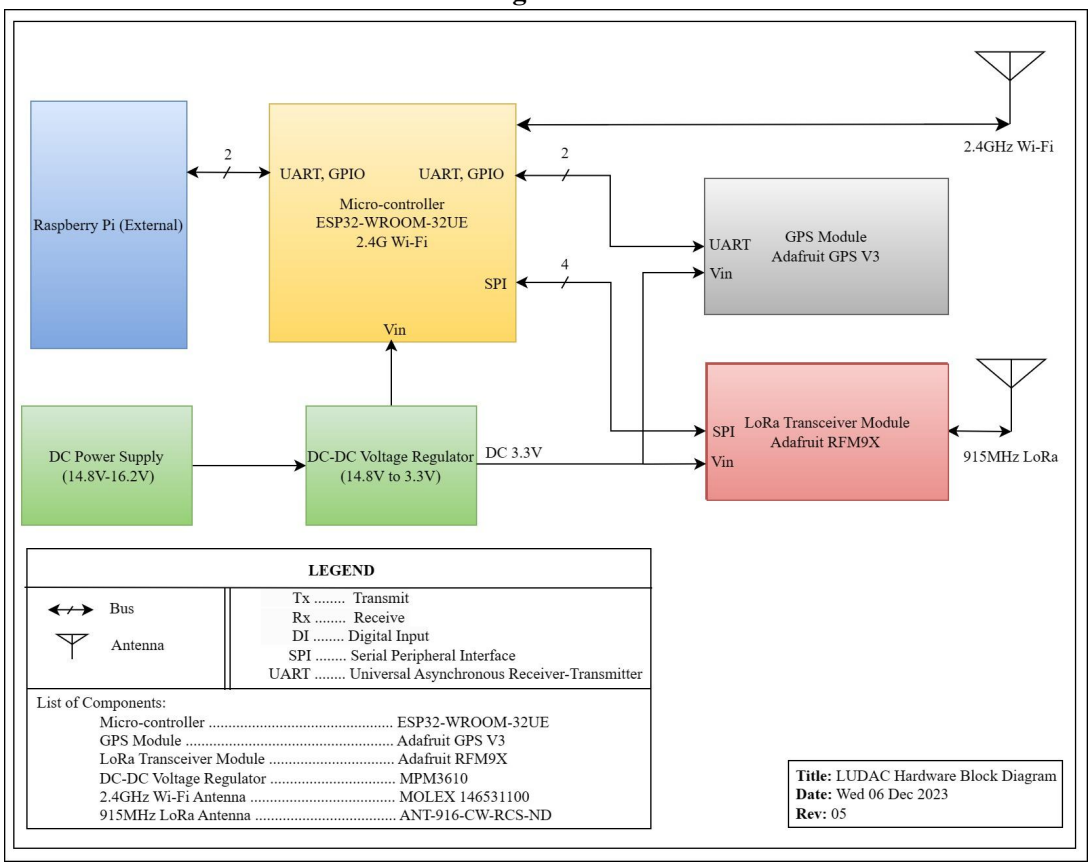


Figure 1 – Hardware block diagram for the proposed design.

Our proposed design, Long Range Underwater Drone to Aerial Communication (LUDAC), is a pair of GPS-capable integrated transceivers that achieve the design objectives through the use of several peripherals managed by a central ESP32 microcontroller. The ESP32 is responsible for coordinating the action of a GPS module (MTK3339), a LoRa module (RFM95W), and its own internal WiFi hardware. The ESP32 will pass information between the onboard computer and relevant transmitters and receivers in order to transmit and receive information across a range of distances. Power will be supplied by the vehicles and stepped down with a buck converter (MPM3610). See Section IV for details on parts selection. A detailed schematic is available at the end of this document.

III. Project Specifications

The transceiver design is consistent with wireless communication standards. Wireless communication will use legal frequency bands over appropriate duty cycles (RSS-Gen, RSS-132, IEEE 802.11), and the radiated power will be within the legal limits for those bands (RSS-102, RSS-247 5.4, RSS-247 6.2.5.2). Interfacing with onboard computers will use UART protocol. Detailed design specifications are available in Table 1.

Table 1: LUDAC Specifications

Parameter	**Conditions**	**Min**	**Typ**	**Max**	**Unit**
Supply Voltage	–	4.5	14.8	21	V
Supply Current	Idle (Standby) Mode	–	149.6	–	mA
	Active Mode	159.1	–	548	mA
Total Power	Idle (Standby) Mode	–	0.49	–	W
	Active Mode	0.53	–	1.81	W
Temperature Range	Operational temperature range	0	30	40	℃
Dimensions	PCB Dimensions	–	9x5x1	10x6x1	cm
	Aerial Housing Dimension	10x6x1	13x3.5x8	15x4x10	cm
Distance Mode via LoRa protocol:					
Parameter	**Conditions**	**Min**	**Typ**	**Max**	**Unit**
Center Frequency	–	–	915	–	MHz
Channel Bandwidth	BW correlates to associated SF configuration	7.8	125	500	kHz
Payload Length	Preamble Length = 12 symbols (programmed register PreambleLength=8)	–	64	–	Bytes
Transmit Bitrate	From SF6, BW = 500kHz to SF12, BW=7.8kHz	0.02	–	37.5	kbps

Range	Antenna design constraint from radial (preferred) to directional	–	1.5	2.5	Km
RF Power Output	In transmit mode	5	13	20	dBm
RF Sensitivity	From SF6, BW = 500kHz to SF12, BW=7.8kHz	-136	-128	-118	dBm
Co-channel rejection	Interferer is a LoRa signal using the same BW and SF. Pw = sensitivity + 3dB	–	-6	–	dBm
Maximum tolerated frequency offset	Between transmitter and receiver, no sensitivity degradation, SF6 thru 9 (BW ranges from 10.4kHz - 500kHz)	2.5	30	120	kHz
Dynamic Range	AGC enabled	–127	-	0	dBm
Proximity Mode via WiFi protocol:					
Parameter	**Conditions**	**Min**	**Typ**	**Max**	**Unit**
Center Frequency	–	2.41	2.44	2.48	GHz
Transmit power	Standard 11n BW 20-40MHz with MCS0/MCS7 scheme	13	15	18	dBm
RF Sensitivity	Standard 11n BW 20-40MHz with MCS0/MCS7 scheme	-69	-85	-92	dBm
Adjacent channel rejection	Standard 11n BW 20-40MHz with MCS0/MCS7 scheme	7	16	27	dB
Transmit Bitrate	Using ESP-NOW protocol	–	1	–	Mbps
Payload Length	Using ESP-NOW protocol	–	250	–	Bytes

IV. Parts Selection and Project Budget

MCU: ESP32-WROOM-32UE was chosen as the microcontrollers for the following reasons [2]:

1. The board has built-in 802.11n (2.4 GHz) WiFi modules with support for external antennas.
2. The board supports SPI, I2C, and UART communication.
3. The board can be powered by a 3.3 V, which is consistent with the GPS and LoRa modules on board.
4. The board has great documentation over hardware and firmware.

LoRa module: RFM95W transceiver (Adafruit breakout) is chosen for the following reasons [3]:

1. The module demonstrates good sensitivity down to -148 dBm.
2. The module operates at 915 MHz, which is the legal North American LoRa frequency.
3. The module interfaces with the MCU over SPI.

GPS module: MTK3339 (Adafruit breakout) is chosen because [4]:

1. The module comes with an internal patch antenna with support for external antenna connection.
2. The module has a built-in voltage converter with a quiet 3.3 V output.
3. The module interfaces with the MCU over serial communication.

Voltage converter: MPM 3610 3.3 V buck converter (Adafruit breakout) was chosen because [5]:

1. The converter has a wide voltage input from 4 V to 21 V.
2. The converter outputs up to 1.2 A of current.
3. Overcurrent protection and thermal shutdown feature.

Table 2: LUDAC Budget Outline

Part Description	Model	Supplied By	Quantity	Price ($CAD)
GPS Board	Adafruit Ultimate GPS V3	Digikey	2	46.78
ESP32	ESP32-WROOM-32UE	Digikey	2	15.62
LoRa Antenna (helical)	ANT-916-CW-RCS-ND	Digikey	2	11.95
WiFi Antenna* (Stick)	Bingfu dual band	Amazon	1 (2-pack)	14.69
WiFi Antenna (Patch)	MOLEX 146531100	Digikey	2	4.30
LoRa Module	Adafruit RFM95W	Amazon	2	36.80
SMA RF Connector	931-1175-ND	Digikey	4	5.95
Push Buttons	PTS526 SM15 SMTR2 LFS	Digikey	10	0.22
Voltage Regulator	MPM3610	Adafruit	2	8.15
PCB Fabrication	N/A	Via Capstone Lab	1	15.00
SMA F to SMA M	900-0732305300-ND	Digikey	2	20.05
TOTAL:	–	–	–	**342.99**

**Will not be ordered at first, but it will be considered if the patch antenna for our WiFi communication does not function as per required.*

This design is covered by the budget provided by the University of Alberta's Capstone Lab. This budget consists of $100 per team member, which equates to $400. Since the budget is $342.99, we will have $57.01 left for shipping and other unforeseen expenses. If during testing we determine that the patch antennas are not sufficient for the required signal strength, stick antennas will be ordered. This cost is included in the $342.99 to account for design revisions.

V. LoRa Configuration

LoRa is a patented long-range and low power proprietary radio communication technique using chirp spread spectrum technology (CSS). CSS is a frequency modulation technique that encodes bits of

information in linear frequency sweeps of the carrier signal, called “chirps”. CSS is a wideband technique, its use of the entire bandwidth allows it to resist noise and the doppler effect. The spread factor (SF) is correlated to the slope of a chirps’ frequency/time graph; higher SF trades bitrate for bandwidth gives better sensitivity and noise resistance, but it sacrifices data rates.

The device chosen for LoRa communication is the RFM95W. It operates at 915MHz and can communicate at ranges greater than 2 km. It can achieve sensitivity down to -148 dBm, and is compatible with SMA and SPI interfacing. The RFM95W has an internal memory of 256 bytes, 252 of which constitute a FIFO buffer for data that is transmitted and received. This buffer is partitioned into Tx and Rx portions, and the base address of each portion is configurable. To use the maximum FIFO data buffer size for Tx or Rx, the base addresses *FifoTxBaseAddr* and *FifoRxBaseAddr* can be set at the bottom of the memory (0x00) by the ESP32 [3]. Maximum bitrate can be achieved by reallocating the FIFO data buffer each time the RFM95W switches between transmitting and receiving. An illustration of this loop is shown in Figure 2.

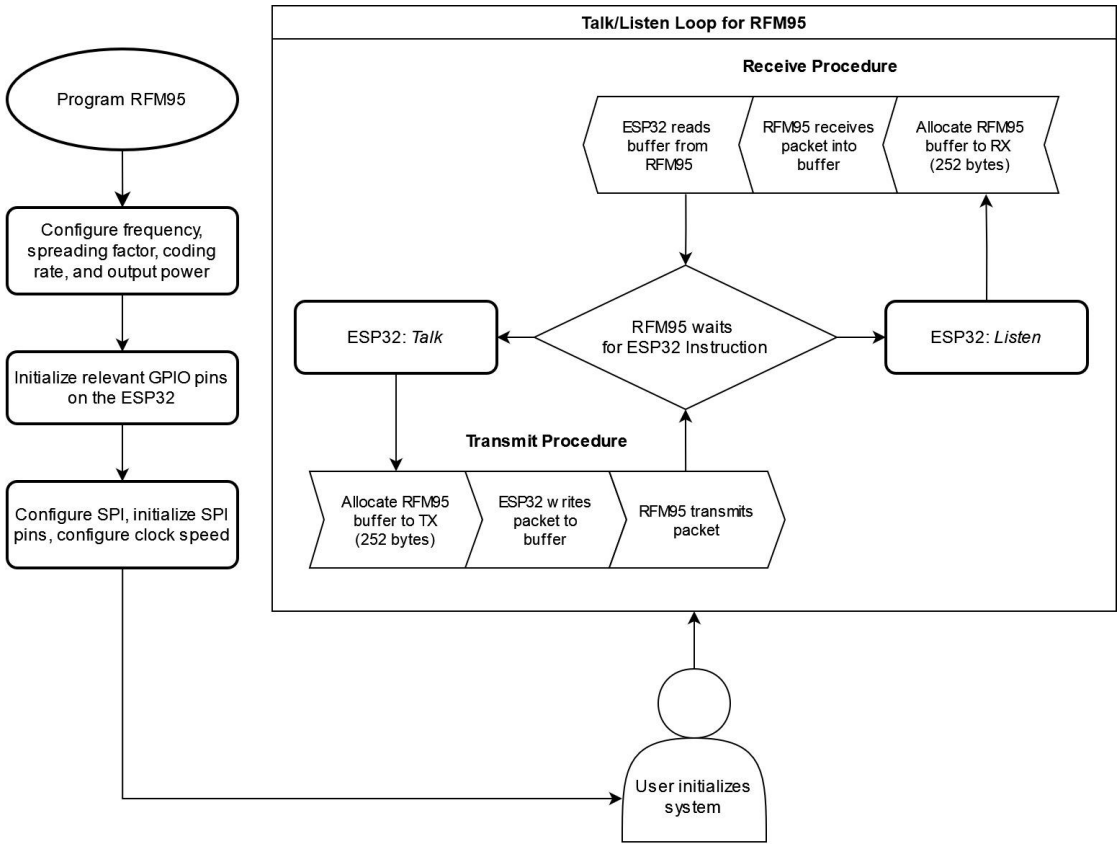


Figure 2 – LoRa Routine

Desired settings, including frequency, power output, modulation parameters, FIFO buffer setup, spreading factor, bandwidth, and coding rate can be programmed by configuring RFM95W internal registers. For example, to utilize the maximum bitrate, the procedure is as follows [3]:

1. Ensure header is set to Implicit mode.
2. Set SpreadingFactor = 6 in RegModemConfig2.
3. Write bits 2-0 of register address 0x31 to value "0b101".
4. Write register address 0x37 to value 0x0C.

Similar procedures are available in the RFM95W datasheet [3]. Communication with the ESP32 microcontroller is handled via SPI. Pin connections are shown in the hardware schematic.

VI. WiFi Configuration

Direct, quick, and low-power peer-to-peer communication will be enabled between multiple ESP32 devices using ESP-NOW, a connectionless WiFi communication protocol defined by Espressif. In ESP-NOW, application data is encapsulated in a vendor-specific action frame and then transmitted from one WiFi device to another without connection [6]. CTR with CBC-MAC Protocol (CCMP) is used to protect the action frame for security [6]. The format of the vendor-specific action frame is as follows:

MAC Header	Category	Identifier	Random Values	Vendor Specific Content	FCS
24 bytes	1 byte	3 bytes	4 bytes	7-257 bytes	4 bytes

where ‘Category’ is set to the value (127) to indicate the vendor-specific category, ‘Organization Identifier’ is a three byte MAC address applied by Espressif, ‘Random value’ is used to prevent relay attacks, and, lastly, ‘Vendor Specific Content’ contains the vendor-specific fields as follows:

Element ID	Length	Identifier	Type	Version	Body
1 byte	1 byte	3 byte	1 byte	1 byte	0-250 byte

where ‘Element ID’, ‘Length’, ‘Organization Identifier’, ‘Type’, and ‘Version’ are ESP-NOW specific identifiers, and the ‘Body’ of the frame can contain up-to 250 bytes of information. The default ESP-NOW bit rate is 1 Mbps [6]. Concerning the security, ESP-NOW uses the CMPP method (as described in IEEE Std. 802.11-2012) to protect the vendor-specific action frame. This can be optionally enabled and hence, requires a Local Master Key (LMK) to be set.

The firmware routine for WiFi would consist of the following key components: Initialization, Add Paired Device, Config ESP-NOW Rate, Config ESP-NOW Power-saving Parameter (optional), Send ESP-NOW Data, Receive ESP-NOW Data, and lastly Reinitialize the protocol [6]. It is recommended to start WiFi before initializing the ESP-NOW and setup WiFi after de-initializing ESP-NOW [6]. Before sending the ESP-NOW data, one must configure the ESP32 in either Station (STA) or Access Point (SoftAP) mode [7]. For this application, we will be configuring the ESP32 as an access point, essentially creating a local network to which devices can connect. In such configuration, the ESP32 broadcasts its own WiFi network with a specific SSID (Service Set Identifier) and password. Furthermore, a device with a broadcast MAC address must be added before broadcast data. Before sending the ESP-NOW data, the sending callback function must be initialized and similar for receiving the data. If necessary, back acknowledgement data can be sent when receiving ESP-NOW data and receiving acknowledgement data timeout can be set, if violated, the ESP-NOW data can be retransmitted. There are several reasons that can lead to ESP-NOW to fail data transmission such as the destination device does not exist, the channels of the devices are not the same, the action frame is lost when transmitting on the air, and so on [6].

Another consideration taken when designing the firmware routine is in the case if there is a lot of ESP-NOW data to send. The data packets send only 250 bytes of data at a time, and once has to avoid too short an interval between sending two ESP-NOW data to prevent disorder of the sending callback function. Hence, it is recommended that sending the next ESP-NOW data after the sending calling function of the previous sending has returned [6]. Note that sending callback function runs from a high-priority WiFi task and it is recommended to avoid lengthy operations in the callback function.

VII. LoRa Synchronization and Transmission Scheme

The "echo" half-duplexing method for LoRa transmission:
This half-duplexing communication method is straightforward to implement and provides information on how long each packet transmission takes. If not eventually implemented in the final product, this could serve as an interim testing LoRa algorithm to evaluate the optimal distance of LoRa transceiving. In this communication scheme, the drone transceiver (DT) acts as the "master" and the submersible transceiver (ST) "slave" [8]. DT periodically (every 1s) sends out messages containing GPS (9 bytes), landing request (1 byte), landing coordinate (6 bytes), landing countdown (1 byte), and the message timestamp (4 bytes), and other auxiliary bytes, in a total of less than 50 bytes, which are below the LoRa packet limit of 251 bytes [9]. ST is on default in RX mode. Once a message is received by ST, it switches to TX mode and "echos" the same message back to DT which is in RX mode after first sending the message. Once DT receives the successfully echoed message, the next packet will be prepared for sending at the beginning of the next period. The time required for the completion of one such cycle of message delivery and echoing must be less than the period, which is set to 1 second in this case, and can be extended to 10 seconds or longer as frequency long-range communication is not required for this application. If no echoed message is received within the period, the program will break out to a timeout and reset routine. The flowchart Figure 3 illustrates the described LoRa half-duplexing procedure.

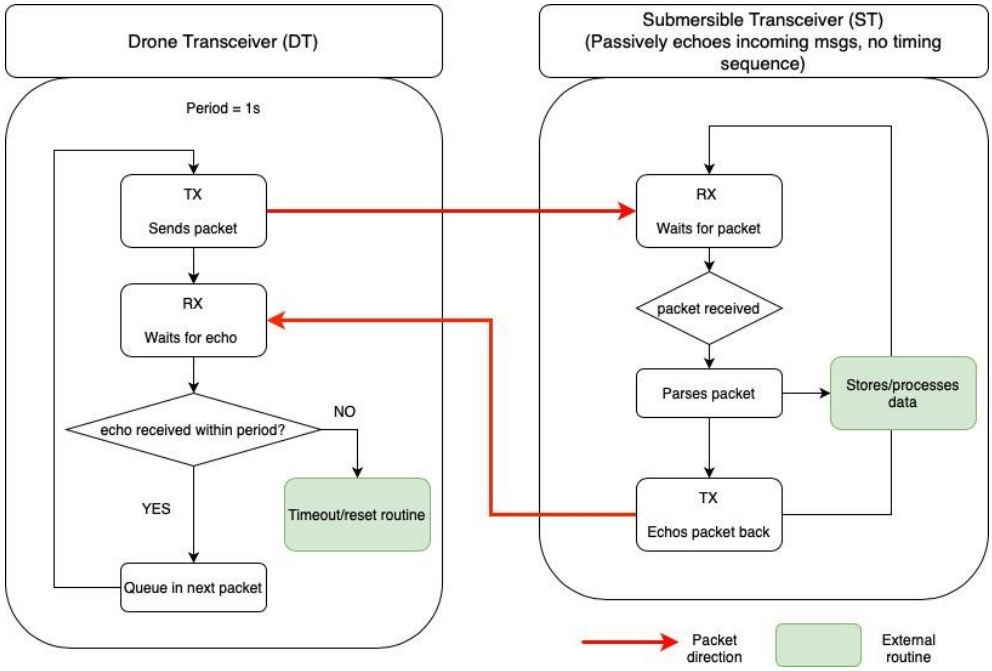


Figure 3 – Flowchart of "echo" half-duplexing.

The "randomness" half-duplexing method for LoRa transmission:
Alternatively, a half-duplexing algorithm incorporating a random timing sequence is worth trying and benchmarked against the "echo" method. A time-domain diagram showing the two-way communication between the transceivers in a half-duplex manner is shown in Figure 4, and a high level flowchart of this method is shown in Figure 5 [10]. Compared to the "echo" method, this protocol does not show confirmation of packet reception, but it may be more efficient.

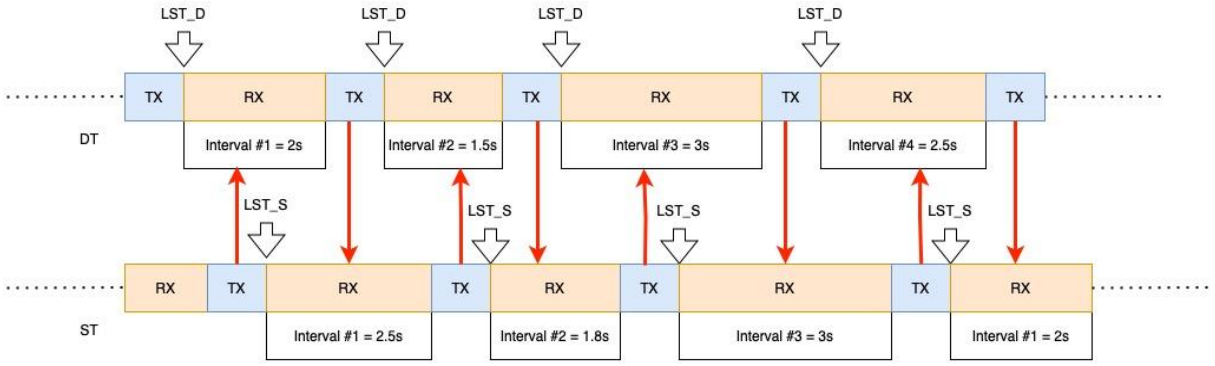


Figure 4 – Time-domain diagram showing two-way half-duplex communication between transceivers.

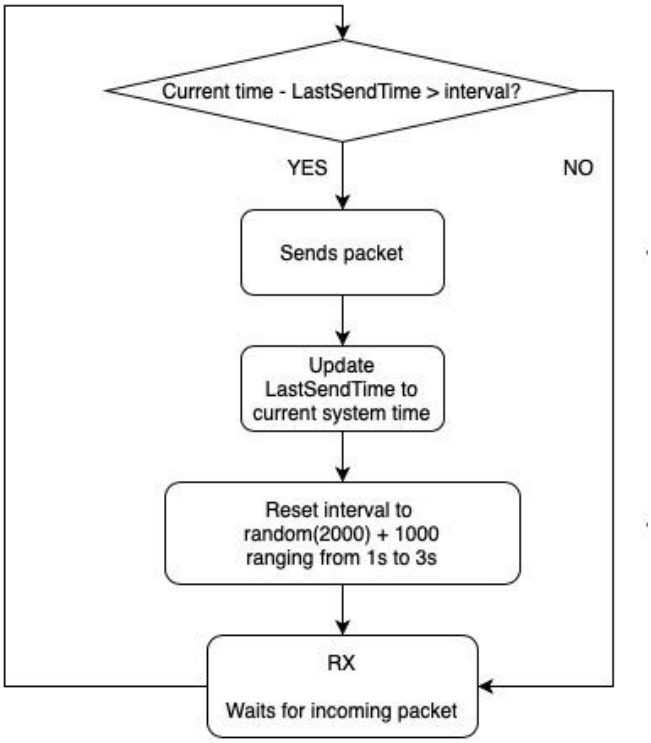


Figure 5 – Flow chart showing "randomness" half duplexing.

VIII. WiFi/LoRa Switching Protocol

WiFi and LoRa switching is determined by the relative distance between the pair of transceivers and the received signal strength indication (RSSI) readings. The flowchart demonstrating this switching mechanism is shown in Figure 6. WiFi communication is first established when the drone is dispatched from the re-surfaced submersible. A PWM signal triggered distance-checking script runs in the background to evaluate the relative distance between transceivers. The distance calculation will be addressed in section XI. When the distance reaches 30 m, the program proceeds to initiate the LoRa modules. The RSSI value is then checked in a PWM triggered script until it reaches the critical value that ensures satisfactory quality of connection. The WiFi mode is then turned off, and the LoRa mode takes over for low-volume information transmission.

When the transceivers are in LoRa mode, the distance-checking script continues running to detect when the transceivers are within 30 m of distance, at which point the ESPNOW (WiFi) protocol will be initiated as described in section VI. The RSSI value of the WiFi connection is then checked against the critical value, at which point the LoRa mode is turned off, and WiFi mode takes over data transmission.

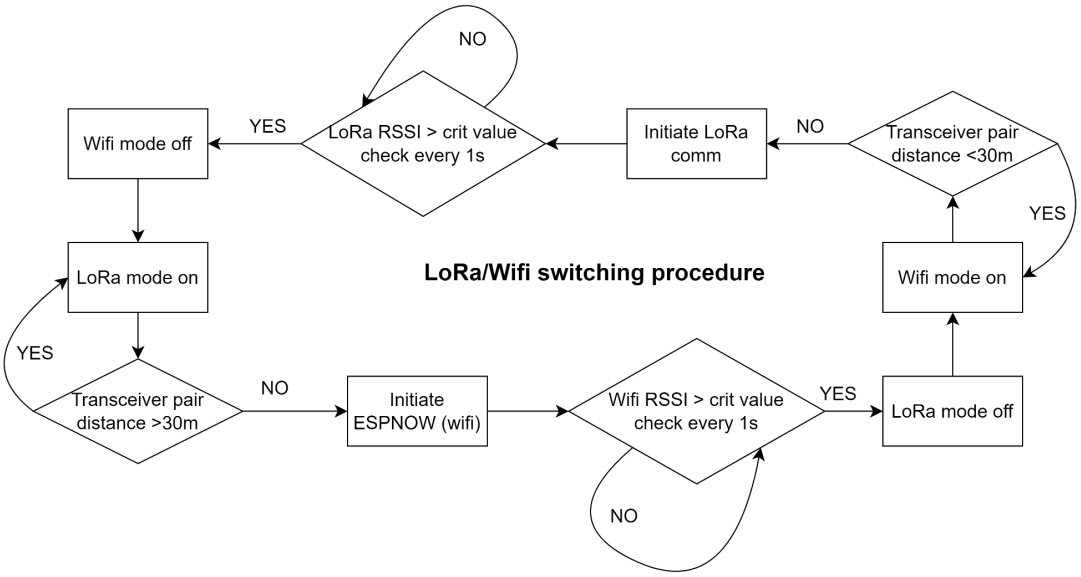


Figure 6 – LoRa/Wifi protocol switching process.

IX. GPS Integration

GPS will be integrated into the design for the purposes of obtaining the relative distance between the drones. This will be accomplished through the use of the GPS module, MTK3339, and the NMEA Parser library. The GPS will be receiving GPS messages in the Global Positioning System Fixed Data (GGA) format [11]. This message format consists of a variety of useful information ranging from coordinates to the number of satellites used in determining location [11][12]. This design will only require the altitude, longitude coordinate, and latitude coordinate.

To extract these three variables from the GPS module, the NMEA Parser library will be used [13]. The GPS module Rx pin will be connected to the RXD0/IO3 pin on the ESP32. Through the project configuration menu, the NMEA Parser Ring Buffer Size, NMEA Parser Task Stack Size, NMEA Parser Task Priority, and the NMEA Statement Support will need to be configured [13]. After setting these parameters, the program can be run to generate the Latitude, Longitude, and Altitude [13]. These three variables will be stored and used in deriving the distance between both transceivers.

X. Distance Calculations

The latitude and longitude can be used to derive the distance between two objects on a spherical plane using the haversine formula. Even though the earth isn't a perfect sphere and more of an ellipse shape, this formula will be sufficient for this application [13]. The haversine formula only requires the earth's radius (r) the latitude (Lat) and longitude (Long) coordinates [14]:

$$Distance_{Spherical} = 2r \times sin^{-1}(\sqrt{sin^2(\frac{\Delta Lat}{2}) + cos(Lat_1) \cdot cos(Lat_2) \cdot sin^2(\frac{\Delta Long}{2})}) \quad (1)$$

Formula 1 is used in radians to produce the spherical plane distance. Since the drones won't be at the same elevation and they will be within 2 km, a more accurate distance is derived using the elevation data obtained from the GPS message. Formula 2 demonstrates how the elevation data can be used in conjunction with formula 1's output:

$$Distance_{Corrected} = \sqrt{(Distance_{Spherical})^2 + (\triangle Elevation)^2} \quad (2)$$

When the transceivers require the distance between them, they will obtain the GPS data and then use formulas 1 & 2 to infer the information.

XI. Detailed Test and Quality Control Plan

Our development plan is divided into two stages. In the first stage, the primary focus will be ensuring that individual components work as intended. Modular development will proceed until all peripherals are independently functional, at which point we will proceed to the second stage. The goal of the second stage is to integrate modular functionality into a cohesive, autonomous system. Details are as follows:

Stage I: Modular Development

1. *Range and reliability* – Gradually increase the distance between the modules in a controlled environment. Test data transmission success rates at predefined distances (e.g., 10 meters, 50 meters, 1 kilometer, etc). Record success rates and latency at each distance. One successful packet delivery and echo (if using the echo half-duplexing method) should be less than 1 second.
2. *Diverse Operating Conditions* – Test under various environmental conditions, such as indoor and outdoor scenarios.Validate that data is properly encrypted and secured during transmission. Ensure data is made available to the on-board for further analysis. A Raspberry Pi test is planned for integration and testing purposes. Verify that data logging and monitoring tools provide real-time information on system status.
3. *Data Format and Reception* – Verify that the data from the receiver is correctly formatted and adhere to the specified protocol. Ensure that the LoRa packets can be successfully interpreted and benchmark the time cost of each packet interpretation to make sure it does not exceed 1/10 of a duplex cycle time (e.g., 100 ms for a 1 s cycle). Validate that data is properly encrypted and secured during transmission. Test if data is made available to the on-board for further analysis. For this purpose, a Raspberry Pi will be incorporated into the test plan to log the transmitted data and display it on a LCD screen for user interpretation. Note that the Raspberry Pi is not part of the design scope, and is only incorporated for testing purposes.
4. *Power Efficiency* – Measure the power consumption of IoT devices during LoRa and WiFi transmission, both in active and standby modes.

Stage II: Integration

1. *ESP32 Mock LoRa and Wifi Data Generation* – The ESP32 will be provided mock raw data such as GPS readings, mission status, landing requests, landing coordinates, etc, and it must be able to process the raw data and construct LoRa packets each with 251 bytes of data. The time cost of this data processing and packet generation will be recorded. Also, the ESP32 will be provided mock images to transmit under Wifi mode. The time cost of image transmission will be recorded. All the mock data will be provided by the Raspberry Pi via serial communication.
2. *LoRa and Wifi Distance-Based Switching* – The LoRa/Wifi module switching function will be benchmarked with varying distance from or the time cost for switching, packet losses and switching distance tolerance. First, the switching mechanism will be tested with no transmission between boards and only the time cost and success rate will be recorded. Then, the board will be loaded with a series of testing information to transmit during this transition to benchmark how the

boards perform to mitigate packet losses.

3. *Autonomous Operation* – Ween the system off of user dependence to ensure it is able to operate autonomously. Testing will involve allowing the system to operate independently and correcting any complications that arise when it does so. This will require coordination with the drone team.
4. *Quality Control* – Repeat the Stage I procedures for the integrated system to ensure that components have integrated correctly.

XII. Packaging

The aerial drone transceiver will be situated outside of the drone's housing, and therefore will need to be waterproof. Our proposed design for a waterproof transceiver housing, and the lid for that housing, is shown in Figures 7 and 8 respectively. Waterproof seals will be embedded in the lid groove, and silicone paste will be applied to the seams. The hole on the bottom of the housing was requested by the drone team to feed a USB cable connecting our MCU to their Raspberry Pi. Mechanical integration with the drone will be reassessed and refined once the drone team finalized their housing design. As of now, the transceiver housing is expected to be attached to the drone via industrial velcro, and is designed around a PCB sized 10 cm x 6 cm x 2 cm. The dimensions of the housing (13 cm x 3.5 cm x 8 cm) will be altered once the PCB is fabricated and components are integrated. Refer to Figure 7 for more detail.

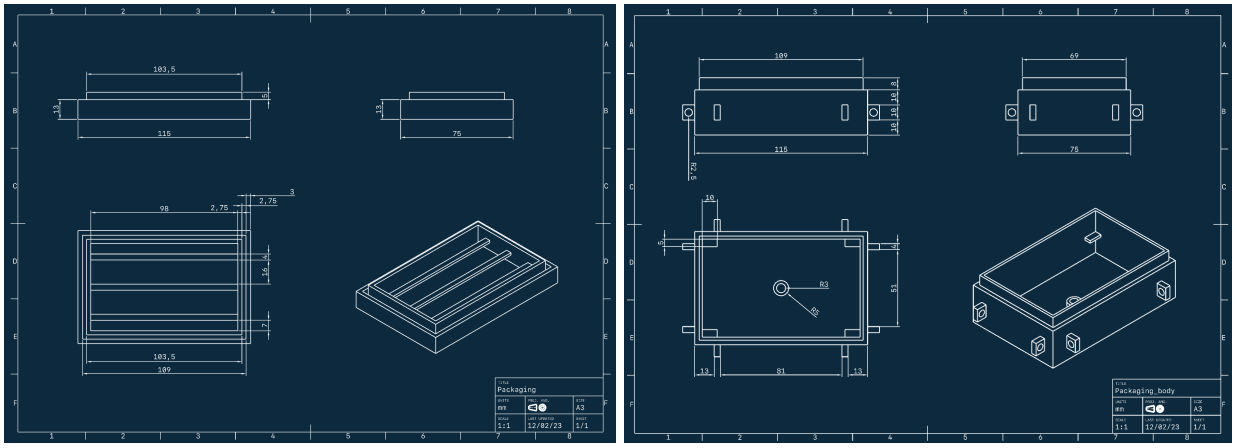


Figure 7 – The housing engineering drawings (left: lid; right: body) showing top and side views.

XIII. Development Schedule

There are six remaining major milestones that we would like to achieve during the course of this project. These milestones are highlighted in red in Figure 8. All of the milestones are within the ECE 491 portion of the Capstone, due to the function testing beginning in ECE 491. Obtaining the relative distance between both transceivers on January 20th 2024 is our first milestone. This date was decided since we would like to complete the most fundamental aspect of the communication protocols prior to WIFI & LoRa.

	Tasks	Expected Start Date	Due Date	Responsible for Task
490	Research ESP32, GPS, and LoRa technologies	2023-10-10	2023-12-08	Team
	Complete Parts Order 1	2023-11-27	2023-12-07	Team
	Set up development environment	2023-12-20	2023-12-25	Nikhil
	Develop ESP32 firmware to read GPS data	2023-11-27	2024-01-15	Antony & Nikhil
	Develop Wi-Fi communication protocol	2023-11-27	2024-01-30	Nikhil & Xinkai
	Develop LoRa communication protocol	2023-11-27	2024-01-30	Xinkai & Ethan
	Implement data processing and storage functionality	2023-11-27	2024-01-30	Antony & Ethan
	Test and debug software components	2023-11-27	2024-03-18	Team
	Complete the Revised Design	2023-11-27	2023-12-07	Team
	Designing the PCB Layout (Initial revision)	2023-11-27	2023-12-31	Antony
491	Designing the housing	2024-01-01	2024-01-08	Xinkai
	Designing the PCB Layout (Final Revision)	2024-01-01	2024-02-01	Antony
	MILESTONE: Achieve distance between drones via GPS		2024-01-20	Team
	Progress Report 2	2024-01-15	2024-01-26	Team
	Complete the Parts Order 2	2024-01-22	2024-02-01	Team
	Revise the PCB Design then submit	2024-02-05	2024-02-14	Team
	Progress Report 3	2024-02-19	2024-03-01	Team
	Complete an inventory of the parts recevied	2024-02-12	2024-02-16	Team
	MILESTONE:Achieve 1 way communication via WIFI		2024-02-18	Team
	MILESTONE:Achieve 1 way communication via LoRa		2024-02-18	Team
	Perform unit testing for individual components	2024-01-13	2024-02-19	Antony & Xinkai
	Perform system testing on the breadboard prototype	2024-01-13	2024-02-19	Ethan & Nikhil
	Perform system testing for the entire IoT system	2024-01-13	2024-02-27	Team
	Debug and fix any issues found during testing	2024-01-16	2024-03-12	Team
	Assembling the PCBs	2024-02-19	2024-02-23	Team
	Prototype the PCBs	2024-02-21	2024-03-04	Team
	MILESTONE: Achieve half duplex communication via WIFI		2024-03-18	Team
	MILESTONE: Achieve half duplex communication via LoRa		2024-03-18	Team
	Integrate the PCB and software components	2024-02-25	2024-03-22	Team
	Construct the 3d-printed housing	2024-02-26	2024-02-29	Xinkai
	Ensure all inventory is acquired	2024-03-01	2024-03-06	Antony
	Conduct final testing and quality assurance of the PCB	2024-03-10	2024-03-23	Team
	Create user documentation	2024-03-11	2024-03-20	Ethan
	Constucting a Presentation Poster	2024-03-18	2024-03-27	Team
	MILESTONE: Integrate the PCB with the drone/submersible computer		2024-03-31	Team
	Complete the Final Report	2024-03-25	2024-04-10	Team
	Practise Showcasing the Project	2024-03-29	2024-04-02	Team

Figure 8 – Project development schedule.

Our next two milestones pertain to achieving 1 way communication via WIFI and LoRa on February 18th 2024. This date was chosen based on anticipating that achieving half duplex communication will require at least a month. Achieving half duplex communication via WIFI and LoRa is set for March 18th 2024. This date was chosen to allow for 1 month to achieve half duplex and provide enough time afterwards to debug any remaining issues. The final milestone of Integrating the transceiver PCBs with the drone computer is on March 31st 2024. This date allows us a few days to prepare for presentations afterwards.

XIII. Conclusion

This report details the design and future implementation of a Long Range Underwater Drone to Aerial Communication (LUDAC) system for use in systems of autonomously operating aerial and underwater vehicles. The proposed system, anchored by an ESP32 microcontroller, integrates GPS, LoRa, and WiFi modules to enable information exchange over varying distances and environmental conditions. The LoRa configuration employs the RFM95W transceiver for long range operation. In this mode, information transfer will be limited, but the range will extend over kilometers. The WiFi configuration employs the ESP-NOW protocol for low-power peer-to-peer communication. This configuration will be used in proximity to enable the transfer of complex data. The envisaged LUDAC system incorporates a dynamic switching protocol, transitioning between WiFi and LoRa modes based on relative distance and signal strength. This is accomplished through integration with a MTK3339 GPS module, which should enable spatial awareness by discerning the relative positions of collaborating drones. A comprehensive testing plan was developed, as well as a detailed project schedule and project budget. The total cost of this project will be $342.99. A detailed schematic of the proposed design is appended to the end of this document. The project is on track for completion by March 31st, 2024.

References and Citations

[1] BlueROV2 - affordable and capable underwater ROV - Blue Robotics, https://bluerobotics.com/store/rov/bluerov2/ (accessed Dec. 6, 2023).

[2] ESP32-WROOM-32E ESP32-WROOM-32UE Datasheet, Version 1.6, Espressif Systems, 2023, pp.1-33

[3] HOPERF ELECTRONIC, "RFM95/96/97/98(W) - Low Power Long Range Transceiver Module V1.0," [Online]. Available: https://cdn.sparkfun.com/assets/learn_tutorials/8/0/4/RFM95_96_97_98W.pdf. [Accessed 6 12 2023].

[4] Adafruit Ultimate GPS Breakout -66 channel w/10 Hz updates, Version 3, Adafruit, 2015, pp. 1-3

[5] Adafruit. "MPM3610 3.3V Buck Converter Breakout - 21V in 3.3V Out at 1.2A." Adafruit. Accessed: Dec 06, 2023. [Online] Available: https://www.adafruit.com/product/4683

[6] Espressif Systems (Shanghai) Co., Ltd., "ESP-NOW," [Online]. Available: https://docs.espressif.com/projects/esp-idf/en/latest/esp32/api-reference/network/esp_now.html. [Accessed 29 11 2023].

[7] Espressif Systems (Shanghai) Co., Ltd., "ESP-IDF Programming Guide - WiFi," Espressif Systems (Shanghai) Co., Ltd., [Online]. Available: https://docs.espressif.com/projects/esp-idf/en/latest/esp32/api-reference/network/esp_wifi.html. [Accessed 29 11 2023].

[8] "Intro to Lora with Arduino, Long Range Wireless for Makers (RFM95 maduino)," YouTube, https://www.youtube.com/watch?v=YhilJivurVQ&t=216s (accessed Dec. 6, 2023).

[9] Arjan, "Best practices when sending GPS location data [HOWTO]," The Things Network, https://www.thethingsnetwork.org/forum/t/best-practices-when-sending-gps-location-data-howto/1242 (accessed Dec. 6, 2023).

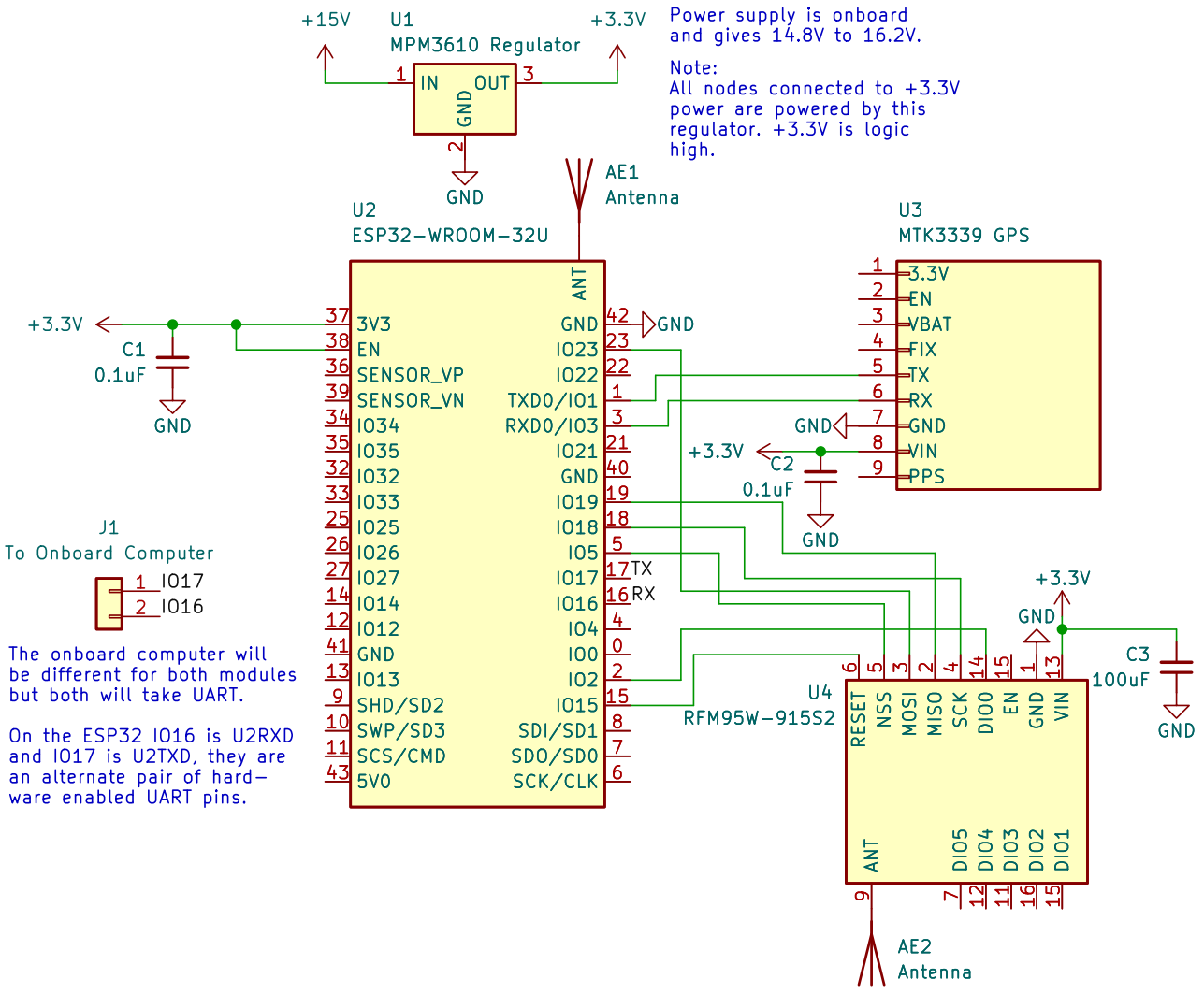
[10] "Lora duplex with ESP32 T-beam," Hutscape, https://hutscape.com/tutorials/lora-duplex-a-esp32-t-beam (accessed Dec. 6, 2023).

[11] Eric Gakstatter. "What Exactly Is GPS NMEA Data?" GPS WORLD. Accessed: Nov 29, 2023. [Online] Available:https://www.gpsworld.com/what-exactly-is-gps-nmea-data/

[12] SiRF Technology, Inc. NMEA Reference Manual, Rev 2.1 (2007). Accessed: Nov 29 , 2024. [Online]. Available: https://www.sparkfun.com/datasheets/GPS/NMEA%20Reference%20Manual-Rev2.1-Dec07.pdf

[13] Espressif/ esp-idf. "NMEA Parser Example." Github. Accessed: Nov 29, 2023. [Online] Available: https://github.com/espressif/esp-idf/tree/c8243465e45/examples/peripherals/uart/nmea0183 _parser

[14] Wikipedia. "Haversine Formula." Wikipedia. Accessed: Nov 29, 2023. [Online] Available: https://en.wikipedia.org/wiki/Haversine_formula



Decoupling capacitor values were chosen in accordance with recommendations made in the data sheets for all components with ICs. AE1 is a wire antenna tuned to 915MHz, AE2 is a patch antenna tuned to 2.4 GHz. Ground for both peripherals is ESP32 ground.

Sheet: /
File: diagram_v1.kicad_sch

Title: LUDAC Integrated Transceiver with GPS

Size: A5	Date: 2023-11-19	**Rev:**
KiCad E.D.A. kicad (6.0.11)		Id: 1/1