IMPLEMENTATION OF A LOW-COST RADIO FREQUENCY DIRECTION FINDER NETWORK BASED ON TDoA TECHNIQUE

By

TSIRIGOTAKIS DIMITRIOS

Electronic Engineering TEI of Crete, 2004

A Thesis Submitted in partial fulfillment of the requirements

for the degree of

MASTER OF SCIENCE

DEPARTMENT OF ELECTRICAL AND COMPUTER
ENGINEERING SCHOOL OF APPLIED TECHNOLOGY
HELLENIC MEDITERRANEAN UNIVERSITY OF CRETE

Approved by:

Major Professor

Kornaros Georgios

Graduation Year 2022

Abstract

The origin of a Radio signal transmission is useful to be located either for “search and rescue”, for military purposes, or just for airplane orientation regarding the landing. “Time Difference of Arrival” (TDoA) is a method that has been used to locate a signal such as an acoustic wave, a seismic wave, or an RF signal transmission. Thanks to the TDoA method, the transmitted signal reaches multiple distant receiver sites with synchronized 'clocks' at a different timing. So the approximate determination of the transmitter location in two or three dimensions would become possible by just placing some pairs of receivers, at known fixed locations. TDoA uses “Trilateration” or “Multilateration” mathematical techniques and high accuracy time information in order to gain a very good bearing resolution.

In this scope, this thesis objective is the creation of a network of three or more ESP32-controlled remote Signal Receiving Stations, and a Raspberry-Pi controlled HTTP Server, providing a world map through “Node Red” environment. The user will be able to set both a radio frequency, and a modulation type (AM/FM) on the server’s page, and present the signal transmitter’s location on a map. The whole system will be based on the ‘’Trilateration’’ or ‘’Multilateration’’ technique.

Περίληψη

Ο εντοπισμός της προέλευσης ενός εκπεμπόμενου Ηλεκτρομαγνητικού σήματος, είναι συχνά χρήσιμος, είτε για «έρευνα και διάσωση», για στρατιωτικούς σκοπούς ή απλώς επικουρικά κατά το προσανατολισμό ενός αεροπλάνου κατά την διάρκεια της προσγείωσης του. Το TDoA (Time Difference of Arrival) είναι μια μέθοδος που έχει κατά καιρούς χρησιμοποιηθεί για τον εντοπισμό είτε ενός ακουστικού κύματος, ενός σεισμικού κύματος, ή ενός εκπεμπόμενου σήματος ραδιοσυχνοτήτων (RF).

Όταν εκπέμπεται ένα ραδιοφωνικό σήμα, φτάνει σε πολλαπλούς απομακρυσμένους δέκτες, σε διαφορετική χρονική στιγμή. Έτσι, ο προσδιορισμός σε δύο ή τρεις διαστάσεις της θέσης του πομπού είναι εφικτός, αν τοποθετηθούν δέκτες σε γνωστές θέσεις, και αξιοποιώντας το TDoA, τις εξισώσεις των "Trilateration" ή "Multilateration" και τις πληροφορίες χρόνου PPS, μπορεί να επιτευχθεί εντοπισμός της θέσης του Ραδιοφωνικού πομπού, με καλή ακρίβεια.

Αυτή η διπλωματική εργασία, είχε στόχο τη δημιουργία ενός δικτύου τριών ή περισσότερων απομακρυσμένων σταθμών λήψης Ραδιοφωνικού σήματος, ελεγχόμενων από μικροελεγκτές ESP32, και τη κατασκευή ενός διακομιστή HTTP πάνω σε Raspberry Pi, όπου θα παρέχει ένα παγκόσμιο χάρτη μέσω του περιβάλλοντος "Node-Red". Ο χρήστης μέσω μιας σελίδας Web, θα μπορεί να διαλέξει μια Ραδιοφωνική συχνότητα και τη διαμόρφωση (AM/FM) που τον ενδιαφέρει, και θα μπορεί μετά από επεξεργασία να αποτυπώσει αυτόματα την τοποθεσία του Ραδιοφωνικού πομπού σε έναν χάρτη. Όλο το σύστημα θα βασίζεται στην τεχνική «Trilateration» ή «Multilateration».

Keywords/Λέξεις-κλειδιά:
RDF, Radio Direction Finder, Trilateration, Multilateration, Si4730, TDoA, PPS, Node-Red, MQTT, SMTP, ESP32, Raspberry-Pi, NMEA, micro-SD, auto-correlation, Arduino, Java.

Acknowledgements

I would like to express my deep gratitude to Professor George Kornaros
I would like to thank George Kornaros, my Family, my friends and some hidden Heroes because of their confidence and their faith for the completion of this Master.
Last but not least, all the people of the forums.

Table of Contents

List of Figures

List of tables

Chapter 1: Introduction

From the "World War I" wartimes, locating the enemy transmitter's position was an advantage which could overturn the battle evolution. But except from war times, in peace times, there are daily many cases where the RDF (Radio Direction Finding) is useful.

Either during the navigation of boats, airplanes, submarines and spaceships, or locating emergency search and rescue beacon transmitters, or even discovering the location of an illegal Pirate Radio Station, or the location of a mobile phone. As the technologies evolve, and the hardware computational power is increased, the future RDF systems will tend to provide even better location accuracy. A variety of methods has been discovered from time to time in order to achieve RDF.

From the many techniques that have been used for the Discovery of the location of an unknown transmitted Radio signal, mostly known techniques are the RSS (Received Signal Strength) or POA (Power of Arrival), ToA (Time of Arrival), FoA (Frequency of Arrival), and AoA (Angle of Arrival) and TDoA (Time Difference of Arrival). With the two last methods, seem to be the more accurate. Each of it, uses different scheme of antennas, and uses either angulation or trilateration techniques, depending on the angles computing or the distances computing it uses.
A brief description of them will be described below.

1.1 The Evolution of Embedded Systems in several domains like automotive industry, technologies and trends towards integrating IoT devices.

In 1970 multiprocessors had been announced in the computer scene architecture, when Moore's law was not yet in vogue and people where totally convinced that uni-processors could not provide the necessary processing power that future applications might require. In the 1980s, the idea that we are ready to reach the physical limits of operating frequency was made even more widespread, and a number of commercial parallel processing machines were built [45].

Embedded systems were widely used in several domains such as Industrial IoT and Industry 4.0. They mark the advent of the automation that industrial connectivity needs between apparatus and equipment, cloud-based systems for harvesting processes. Also, for analyzing information faster than before and increasing the value of the business through new experience and customer service like real-time analysis, or visualization data for improving their operational efficiency [46].

Embedded systems are widely used in the automotive industry. Each car is expected to exceed 50% of electrical equipment through the next years, and each vehicle is expected to contain more than 200 sensors components that transmit signals to an ECU (Engine Control Unit) [47].

There are some candidate scenarios of a completely autonomous vehicle, like the ability to fully eliminate all the traffic lights in favor of vehicle-to-everything (V2X) communication to authorize a safe and sound transit through intersections for each and every vehicle coming from any direction, having both advanced sensing capacity and connectivity.

Though, the advancing open software-dominated services in new vehicles along with wireless network communicating vehicles is now imposing potential [50] [51]. Also CPS (Cyber-physical systems) is a new trend in the context of industrial automation automotive medical systems. CPS are devices with sensors and actuators that connect the physical to the virtual world. There is a strong trend towards open systems, which can be extended during operation by instantly adding functionalities on demand [48].

In the new reality of autonomous machines and wirelessly connected cars, the capability of cyberspace damage attacks has increased exponentially. This requires sophisticated methods of cyber defense possibilities, such as energy behavior analysis of real-time consumption or execution and anomaly detection in communication protocols and firmware [49].

As we can notice, there are several domains that the evolution of embedded systems has response to make better and safer quality of life and increase the efficiency of businesses and industries.

1.2 RSS (Received Signal Strength) technique or POA (Power of Arrival)

At this low-cost technique, the received signal strength usually expressed in dBm, is determined from the frequency, the power delivered to the transmitting antenna, its geometry, the radiation resistance, the environmental conditions, the receiving antenna, the path loss, and the distance to the receiving antenna. This signal's value is highly related to the distance, as it is expressed by the free space equation (1):

$$\mathrm{P(d)} = P_0 - 20log_{10}\frac{\mathrm{d}}{d_0} \quad \mathrm{dBm} \qquad (1)$$

By either changing the receiving antenna location, by rotating it mechanically, or by using multiple receiving antennas at different locations, the Transmitter's location can be discovered. Such an example, is the "war driving" term, used recently for locating Wi-Fi Access Points, based on their receiving Signal strength. There, by driving a car to an unknown area, and by measuring the Wi-Fi Signal strength, we reclaim that by getting closer to the transmitter, the signal strength increases. This technique, can be used if the signal energy arriving to the receiver is measurable, and preferably with outdoor "Line-of-sight" propagation at the VHF/UHF frequency range due to the luck of ionospheric refractions. As this positioning method is too inaccurate, anchor nodes [25] can be used to increase it.

1.3 AoA (Angle of Arrival) technique

Here, with the use of dedicated antenna arrays, or sometimes rotated directional antennas (e.g. Yagi), the bearing (angular directions Azimuth or even Elevation) to the transmitter's location is found from two or more receiving Stations, and is plotted to a map. The result can be more real-time, but the equipment is more demanding. On Figure 1 is shown a simplified form of AoA.

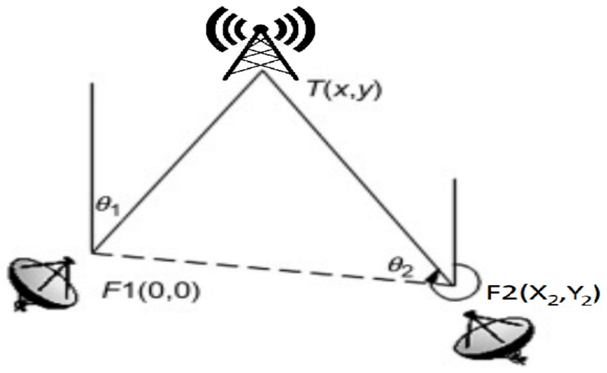


Figure 1 AoA triangulation example

The equations (1.1), (1.2) give the coordinates of the transmitter.

$$y = \frac{y_2 * \tan(\theta_2) - x_2}{\tan(\theta_2) - \tan(\theta_1)} \qquad (1.1)$$

$$x = y * \tan(\theta_1) \qquad (1.2)$$

There are also other types of AoA techniques, such as Geometric Multilateration, which don't use only angular, but also lengths (lines) as the example on Figure 2.

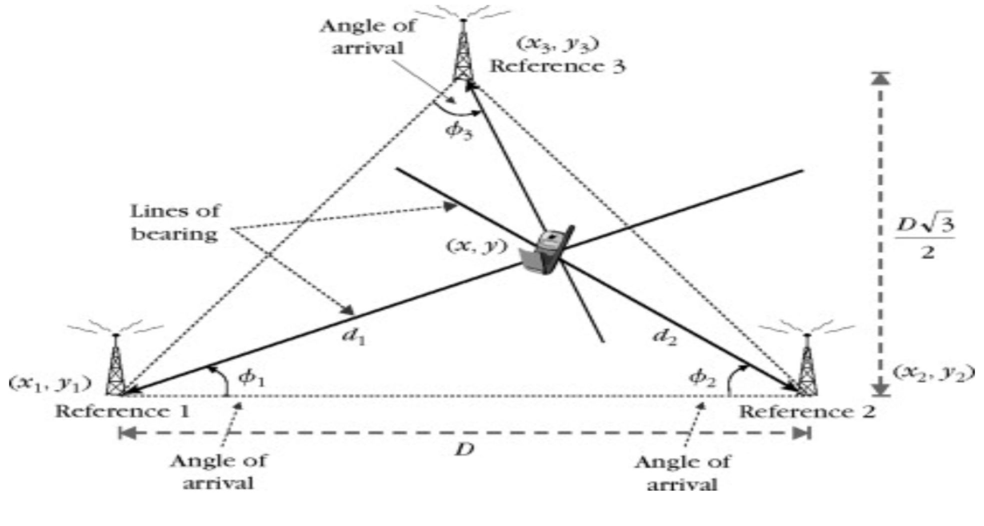


Figure 2 AoA based on Geometric Multilateration

Source: https://www.sciencedirect.com/topics/engineering/angle-of-arrival

Here, the (1.3), (1.4) equations give the coordinates of the transmitter.

$$x = d_i\cos(\phi_i) + x_i \qquad (1.3)$$

$$y = d_i\sin(\phi_i) + y_i\,,\ i = 1,2,3 \qquad (1.4)$$

The term "triangulation" is usually used to express any calculations have to be done. Early AoA Radio Direction Finders, use phased array antennas or Doppler techniques which increases the accuracy. In the case of a Periodic or a narrowband signal, AoA performance, tends to be better than TDoA. Industry 4.0, logistics and healthcare tend to use it for locating products, medical devices, tools or people [26]. For example, the tracking and the reliable identification of an aircraft engine in a huge production hall, can improve productivity and lead to a faster and more reliable restoration.

1.4 ToA (Time of Arrival) technique or TOF (Time of Flight)

This technique is used for the positioning of a variety of energy waves with a known velocity such us radio, acoustic, seismic, and is based on multiple Receiving Stations, with absolutely synchronized their clocks. The distance from the transmitter, is found by just using the equation (1.5). By using more reference points e.g. three for two-dimensional or four for three-dimensional, the exact position of the transmitter can be calculated by finding the intersection.

$$d = c * (t_{arrival} - t_{sent}) \qquad (1.5)$$

As the ToA technique presupposes the knowledge of the initial transmission time, the TDoA is preferred in its place.

1.5 TDoA, Trilateration - Multilateration

The basic idea of exploiting TDoA (Time Difference of Arrival) is that when a RF signal is been transmitted, it needs a specific time to travel with the speed of light and reach a specific point Receiving location. Thus, by measuring the difference of the arriving signal time at three different places, it is possible to draw one hyperbolic or circle at each Receiving position, corresponding to the transmitter's distance.

The accuracy of the TDoA method is related to the Time precision. For a 50nS time difference error on two Receiving stations, the transmission location accuracy could reach 12meters (distance=3*10^8m *50nS=15m).

But there are also other factors that can influent the transmission location accuracy, such as the signal Bandwidth, the signal Periodicity, obstacles presence and the position (geometry) of the receiving stations, relatively to the transmitter's position.

Each Receiving Station position is very influential. As it is shown on Figure 3 case "A" optimum example, when one RF Signal transmission appears at the Mediterranean Sea, arrives on different time base at Crete, Rhodes and Libya.

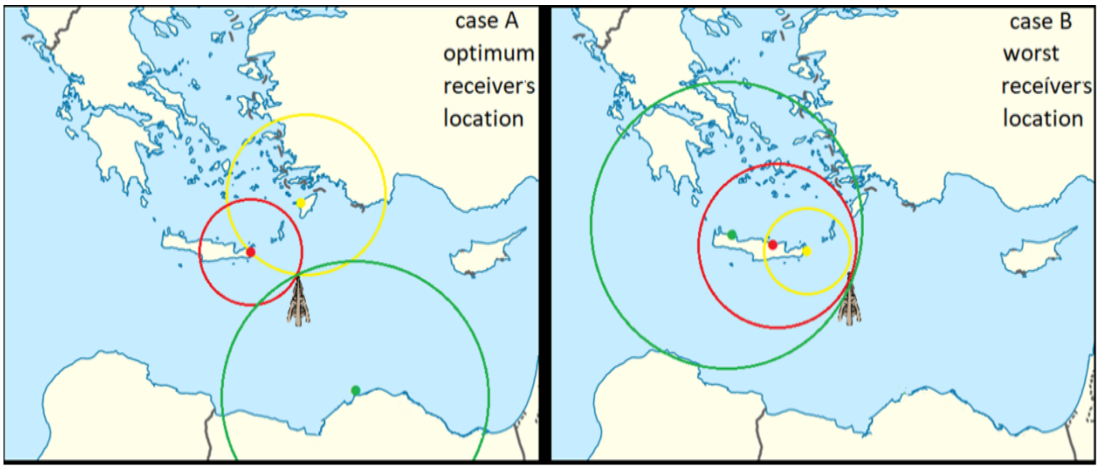
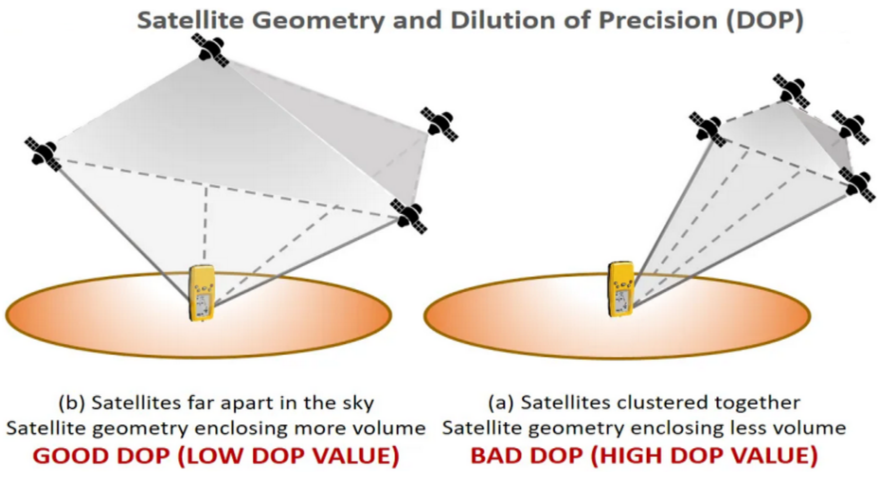


Figure 3 example of Optimum (A) & Worst (B) receiving station's location

On the other hand, on case "B" worst scenario, if all the Receiving stations are located on Crete, at the west of the target, we could not achieve good location accuracy at all for this Transmitter's location.

This situation can be compared with a satellite positioning condition, where the term GDOP (Geometric Dilution of Precision) is usually used on GNSS (Global Navigation Satellite Systems) to describe mathematically the error propagation due to a good or bad combination of satellite positioning. GDOP is usually divided on HDOP (Horizontal DOP describing Latitude and Longitude) and VDOP (for Vertical Dilution of Precision). On Figure 4 it is shown the Satellites GDOP cases.



Satellite Geometry and Dilution of Precision (DOP)
(b) Satellites far apart in the sky
Satellite geometry enclosing more volume
GOOD DOP (LOW DOP VALUE)
(a) Satellites clustered together
Satellite geometry enclosing less volume
BAD DOP (HIGH DOP VALUE)

Figure 4 Good - Bad Satellites DOP
Source: https://www.ocean-yachting.com/gps-error-sources

When just three RF Receiving Stations are used, the method is called "Trilateration", otherwise, when more than three Receiving stations are used, the method is name "Multilateration" (or Hyperbolic Positioning). At minimum three RF Receivers are required in order to locate RF transmissions at the 2D (e.g. earth surface transmissions). Otherwise, in case of locating at 3D dimensions (e.g. flying airplane transmissions) at least four RF Receivers properly placed are required. The Math expressions [27] that can be used for the Multilateration are shown in the equations (1.6), (1.7), (1.8), (1.9), (1.10), (1.11) below.

The travel time t_i of a signal from a reference station i to a mobile terminal is given by the distance divided by the signal propagation speed v:

$$t_0 = \frac{1}{\mathrm{u}}\sqrt{(x-x_0)^2+(y-y_0)^2} \qquad (1.6)$$

$$t_1 = \frac{1}{\mathrm{u}}\sqrt{(x-x_1)^2+(y-y_1)^2} \qquad (1.7)$$

$$t_2 = \frac{1}{\mathrm{u}}\sqrt{(x-x_2)^2+(y-y_2)^2} \qquad (1.8)$$

If reference station 0 is taken to be at the coordinate system origin, then the equation (1.8) can be reduced to:

$$t_0 = \frac{1}{\mathrm{u}}\sqrt{x^2+y^2} \qquad (1.9)$$

The mobile station does not know the absolute values of t_0, t_1 and t_2. It is only able to obtain the time differences:

$$\tau_1 = t_1 - t_0 = \frac{1}{\mathrm{u}}\left(\sqrt{(x-x_1)^2+(y-y_1)^2} - \sqrt{\mathrm{x}^2+\mathrm{y}^2}\right) \qquad (1.10)$$

$$\tau_2 = t_2 - t_0 = \frac{1}{\mathrm{u}}\left(\sqrt{(x-x_2)^2+(y-y_2)^2} - \sqrt{\mathrm{x}^2+\mathrm{y}^2}\right) \qquad (1.11)$$

Equation (1.10) and (1.11) must now be solved for x and y. All other vales are known.

1.6 FDoA (Frequency Difference of Arrival) or DD (Differential Doppler)

This method is analogous to TDoA and usually it is used together with TDoA to enhance accuracy and seldom alone itself. Only at the case of locating a narrowband signal having a long pulse duration, is used instead of TDoA due to its high Doppler resolution. FDoA is based on the "Doppler shifts" effects that are observed between the transmitters and the receiver's points. This means that the signal phenomenally appears to change its Frequency, as it is influenced by the transmitter's and the receiver's speed. Nonlinear equations are used to solve FDoA problems [52].

Chapter 2: State of the art - Related works

Many studies have been made, in order to improve the accuracy of TDoA method.

An efficient but more complicated method to optimize the localization is “TDoA and GRoA” (Gain Ratios of Arrival) based on Levy-GSO (Glowworm Swarm Optimization) algorithm, which is more suitable for sensor networks of dense nodes where “TSWLS” (Two-Step Weighted Least Squares) localization method is used. There, measurement information from both GRoA and TDoA are used at the same time [28].

Also, there is a new improving method which is combining the TDoA, with the De-embedding the Propagation Background method, proposed for NLOS (Non Line of Sight) environments. Here, instead of using the direct received signals, a transfer function is used for the propagation route to de-embed the NLOS effects. Thus, any propagation distortions are removed [29].

Another approach is based on RACC (RSS Assisted Cross-correlation) method to mitigate the effect of multipath interference, mostly for indoor localization systems. Thus, a RSS-assisted Cross-correlation method, mitigate the impact of multipath interference, achieving sub-meter range precision increasing localization precision [30].

By Combining TDoA and FDoA (Frequency Difference of Arrival), measurements are utilized with Taylor-series expansion and pseudo-Gaussian importance function is constructed. Thus, the named “Monte Carlo Importance Sampling” simulation algorithm is used, assigning multiple values to an uncertain variable to achieve multiple results, and then averaging the results, a global maximum convergence solution is found. Thus, the source location uncertainty is reduced [31].

Comparing to a similar method, the TDoA nonlinear equations can be solved by using the collaborative localization algorithm, which is based on PSO (Particle Swarm Optimization) and the Taylor series expansion algorithm, offering higher reliability and higher positioning accuracy [32].

On another work, SSA (Salp Swarm Algorithm) is used to solve the nonlinear problem of TDoA passive location, offering high location accuracy, less control parameters and more robust performance than PSO (Particle Swarm Optimization) and IPSO (Improved Particle Swarm Optimization) [33].

Improve performance and RM (Range Migration) can be done, by using the joint signal of TDoA and FDoA (Frequency Difference of Arrival) used during long observation time, and an improved KTM (Keystone Transform-based Method) is proposed to due to the transmitter’s high velocity movement, reducing Doppler ambiguity and increasing noise-resistance [34].

In addition, Low-Complexity WPLE (Weighted Pseudo linear Estimator) with Systematic Error Correction is proposed due to its low complexity [35].

TDoA can also be combined with RSS (Received Signal Strength) which are used based on EKF (Extended Kalman Filter) to correct the estimated position of the transmitter, achieving a large improvement in the accuracy [36].

Joint TDoA and FDoA estimation interpolation method can be used, with sub-sample accuracy based on SOCP (Second-Order Cone Programming) improving accuracy [37].

Various implementations have been done, by using different hardware each time.

At a similar implementation, A-TDoA (Asynchronous Time Difference of Arrival) was used. There with the deploying of remote nodes capable of both receiving and transmitting, and the use of algorithms such as SDP (semi-definite programming), Taylor algorithms and CLS (constrained least squares) succeed fast global convergence and high precision of about 15.2cm, to achieve superior performance [38].

With the help of a “DecaWave Trek1000” evaluation board kit, a successful implementation of UWB (Ultra-Wide Band) Tag for indoor localization system, was done. The achieved Localization accuracy was 50 cm, and more accurate for small areas, with LoS (Line-of-Sight) [39].

Thanks to the use of USRP (Universal Software Radio Peripheral) - an open source experimental SDR (Software Defined Radio), it was achieved to successfully located transmitters on another implementation [40].

By the usage of the SDR model “USRP N210”, and by using RSSI (Received Signal Strength Indicator) measurements, TDoA measurements, and joint TDoA and FDoA measurements are used, achieving a localization error of around 50m [41].

On a low cost implementation, with the usage of three receiving boards “RTL-SDRv3”, but with the utilization of NTP for the clock needs instead of GPS, but with the usage of the MQTT protocol for the communication, seems to offer notable Locating results of 15 meters accuracy after a MATLAB analysis [42].

According to another more innovating study, based on CSS (Chirp Spread Spectrum), and by using the “Nanotron nanoLOC 3.0 Development Kit” and the Chan, Kalman and Taylor improving TDoA algorithms, there will be an improved location accuracy deviation of less than 0.5 meters [43].

Chapter 3: Contribution

In this Thesis, it was made an attempt to create an operational economical AM/FM Radio Direction Finder system, which could be constructed with easy found components, and it could provide enough geolocation accuracy. For the implementation of this system, a network of remote Receiving Stations had to be constructed, and also, a Node-Red Server had to be constructed, in order to provide an interface for commanding the measurements, and for taking back the results.

Firstly, the user by accessing a Web page sets a desired frequency and its modulation type, and afterwards, after originating a measuring process of simultaneously audio capturing from the Receiving Stations, he gets the result of the AM/FM transmitter's location appearing on a map. Additionally, a free Email and a MQTT (Message Queuing Telemetry Transport) Server have to be accessed. An example of this system's overall design is shown on Figure 5.

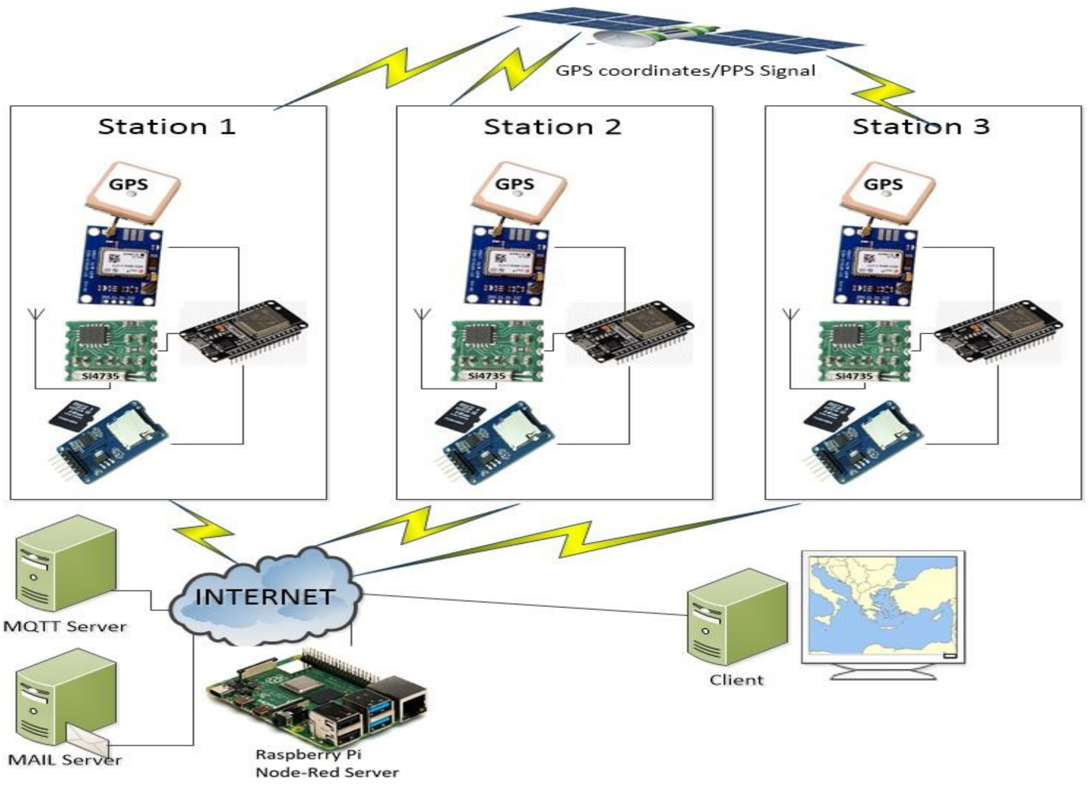


Figure 5 Thesis' overall design

Most GPS receiver modules are capable to give a time accuracy of 100nS and they provide two master outputs: The "NMEA" (National Marine Electronics Association) location-time sentence and the "PPS" (Pulse-Per-Second) signal which specify the start of a second very quarterly. The combination of these two information will be used at the project's time calculations. Each one of the remote nodes will contains mainly a microcontroller (such as ESP32), an RF receiver module (such as Si4730), a GPS receiver module (such as NEO-8M), and a micro-SD card adapter module.

The commands and the measurement results between the nodes and the server will be spread through SMTP (Simple Mail Transfer Protocol) and the MQTT protocol. In that way, communication can be additionally secured if needed to enhance authenticity, confidentiality and integrity, especially in a military environment. The whole infrastructure is implemented with the minimum cost as a target.

Chapter 3.1: Clocks - Time Accuracy

The time precision is getting continuously improved parallel with the technological evolution as it is shown on Figure 6.

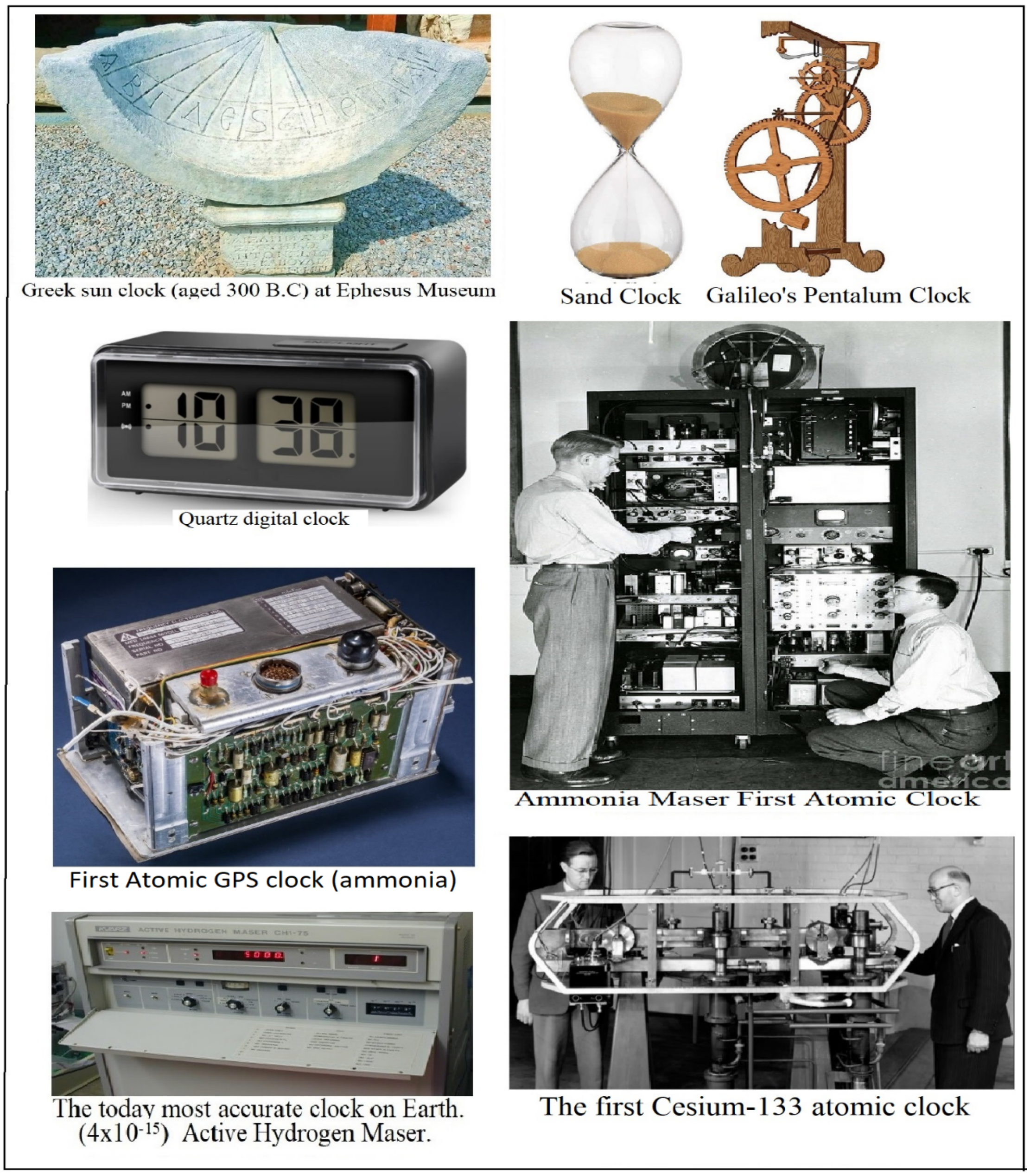

Greek sun clock (aged 300 B.C) at Ephesus Museum
Sand Clock
Galileo's Pentalum Clock
Quartz digital clock
Ammonia Maser First Atomic Clock
First Atomic GPS clock (ammonia)
The today most accurate clock on Earth. (4x10-15) Active Hydrogen Maser.
The first Cesium-133 atomic clock

Figure 6 Time accuracy evolution

We daily use clock devices with different time accuracies, such as the simple quartz-crystal clock with 20 ppm accuracy, the TCXO (Temperature compensated crystal oscillators) with 3ppm accuracy, the OCXO (Oven controlled crystal oscillators) with 0.2ppm accuracy.

Other time discovery methods are also used, such as the Radio RDS (Radio Data System) signal incorporated in a broadcasting song, offering a clock signal with accuracy of almost 100mS. The NTP (Network Time Protocol), and PTP (Precision Time Protocol) give us accuracy of 10-100 milliseconds. The RCC (radio-controlled clocks) give us accuracy of about 30-62 milliseconds. The DVB (Digital Video Broadcasting) can offer almost Stratum 1 leveled time accuracy.

Furthermore, there are molecular ammonia gas clocks, atomic heated liquid cesium-133 clocks with accuracy of $4x10^{-15}$, but the Medal goes to Optical clocks with strontium or ytterbium-171 or newer technological explorations based on Quantum Hubs clocks, offering to us tomorrow, huge accuracies of $1.6x10^{-18}$ [22]. On Table 1, it is shown a comparison of the more recent clock accuracies. Cesium and Rubidium atomic clocks are commonly used on all the GNSS satellites in our days. The more resent Galileo satellite constellation uses passive hydrogen maser and Rubidium atomic clocks [23].

	Oscillators Comparison					
	quartz			Atomic		
Specifications	TCXO	MCXO	OCXO	Rubidium	RbXO	Cesium
Accuracy/year	$2*10^{-6}$	$5*10^{-8}$	$1*10^{-8}$	$5*10^{-10}$	$7*10^{-10}$	$2*10^{-11}$
Aging/Year	$5*10^{-7}$	$2*10^{-8}$	$5*10^{-9}$	$2*10^{-10}$	$2*10^{-10}$	0
Temp Stability Range	10^{-9} (-55°C +85°C)	$3*10^{-10}$ (-55°C +85°C)	10^{-12} (-55°C +85°C)	$3*10^{-12}$ (-55°C +68°C)	$5*10^{-12}$ (-55°C +85°C)	$5*10^{-11}$ (-28°C +65°C)
Stability s_y (τ) Tα=1s	10^{-9}	$3*10^{-10}$	10^{-12}	$3*10^{-12}$	$5*10^{-12}$	$5*10^{-11}$
Size cm^3	10	30	20-200	200-800	1000	6000
Warm Up time mins	0.04	0.04	0.6	20	0.65	30
Price $	10-100	<1000	200-2000	2000-8000	<10000	<50000

Table 1 Clocks' Accuracy comparison

Chapter 3.2: Time Systems clarification

The UTC (Coordinated Universal Time) was officially formalized in 1963 and is interconnected with the Earth's rotation speed. But because of climatic and geological factors (e.g. nearby planet passages, earthquakes shifting continental plates mass, magma composition changing, Constructing Huge Dams), the Earth speed is possible to change. These Earth rotating changes can nowadays accurately be measured with methods such us VLBI (Very Long Baseline Interferometry). So, every few years, some Leap seconds are added (or also removed), in order to keep the

UTC time close to the mean solar time. When no high time precision is required, UTC is commonly used.

The GPS Time, formalized also in 1963, was firstly initialized-synchronized at 00:00am o clock of 6 Jan 1980 with the UTC time, and for now is ahead almost 18 seconds from UTC time. GPS time accuracy is ranged from 14-100 nanoseconds practically [18]. GPS time is not corrected to match the rotation of the Earth, so it does not contain leap seconds or other corrections that are periodically added to UTC.

GPS is just a “timekeeping clock”. The same behavior is also followed by the European GALILEO and the Chinese BEIDOU Systems.

Contrary, the Russian GLONASS (Global Navigation Satellite System) uses the time scale by implementing leap seconds, just like UTC does.

Thus, each GNSS constellation has its unique independent time. On Figure 7 it is shown a Comparison of time differences.

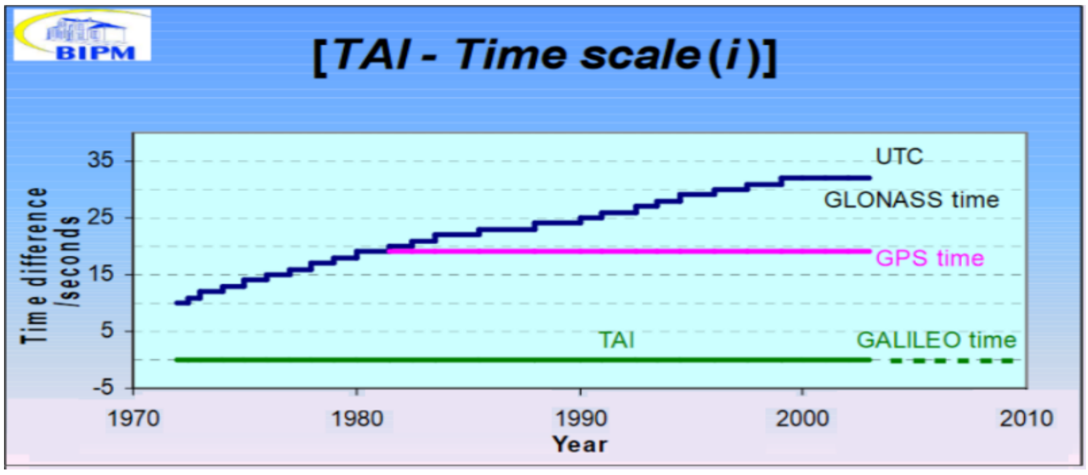


Figure 7 Comparison time difference of systems

Source: https://www.itu.int/ITUD/tech/events/2012/ ResultsWRC12_CIS_StPetersburg_June12/Presentations/Session6/S6_3_b_E.pdf

IAT (International Atomic Time) is the basis for all the world clocks, as it is keeping an average of over 200 atomic clocks in over 50 national laboratories worldwide.

As the geolocation accuracy of the TDoA technique is highly depending on time accuracy, we tent to improve geolocation by reclaiming the best supplied method of our current Times.

3.3 The GNSS (Global Navigation Satellite System)

At the elevation of 20200 Km from Earth’s surface, there are more than 135 Satellites surrounding Earth just to offer the service of geo-location, the time clock, and the PPS Signal. By using only 24 satellites it is possible to cover earth's surface area for 95% of the time [4].

By Starting counting from the American GPS (Global Positioning System) with 31 satellites, The Russian GLONASS (Global Navigation Satellite System) with 26 satellites, The Chinese BDS (BeiDou Navigation Satellite System) with 35 satellites, the European GALILEO with 30 satellites, the Indian IRNSS (Indian Regional Navigation Satellite System) with 9 satellites, and the Japanese QZSS (Quasi-Zenith Satellite System) with 4 satellites, the Navigation accuracy is continuously getting improved. As it is shown on the comparative example on Figure 8, an electromagnetic wave traveling time, is proportional of the distance.

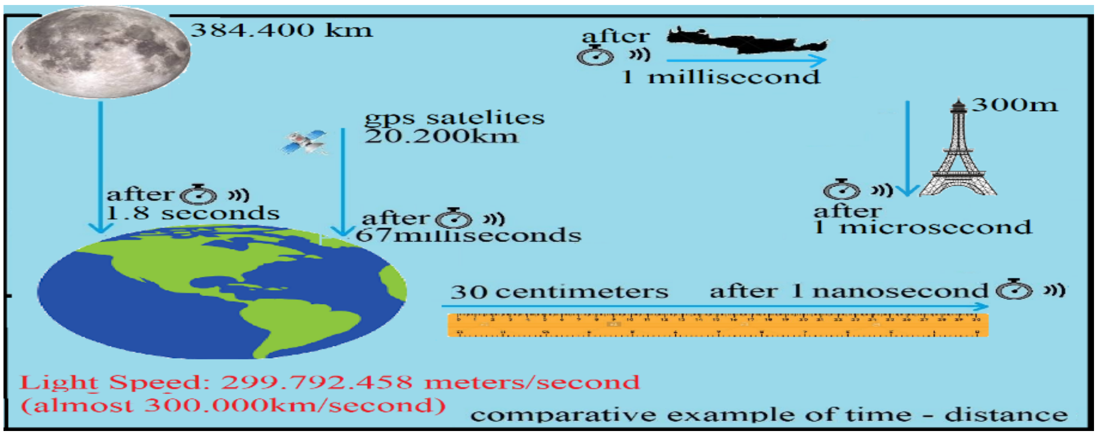


Figure 8 traveling Electromagnetic wave time comparisons

The GNSS operation is based on the Multilateration. All the GPS Satellites transmit concurrently at the same L1, L1C, L2C, L5, frequencies RF signals pointing Earth. These signals are arriving at a point on earth, at a different time from each satellite due to different distances shown On Figure 9.

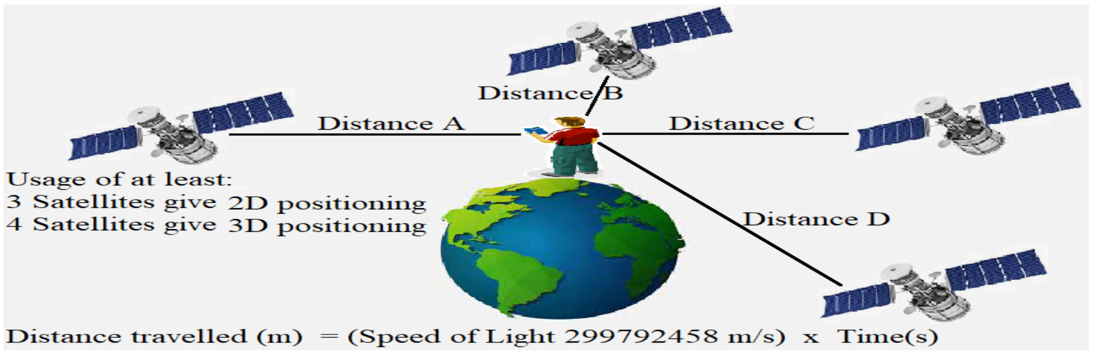

Distance B
Distance A
Distance C
Usage of at least:
3 Satellites give 2D positioning
4 Satellites give 3D positioning
Distance D
Distance travelled (m) = (Speed of Light 299792458 m/s) x Time(s)

Figure 9 GNSS trilateration example

These Signals don't conflict each other, due to their special CDMA-DSSS (Code Division Multiple Access) - (Direct Sequence Spread Spectrum) modulation.

Thanks to mathematical calculations and by knowing the exact position of each satellite, it is calculated the exact location of a specific spot.

The basic GNSS geolocation accuracy given to us by using just a single GNSS Receiver can be 2-5 meters. However, with additional methods, it can be increased even more. By using DGPS (Differential Mode DGPS), it can reach 0.7-2 meters' accuracy, or by using the complementary RTK (Real Time Kinematics), or by using CPDGPS (Carrier Phase Differential GPS), 1-2 centimeters can be achieved. Also, complementary services such as SBAS (Satellite-Based Augmentation System) e.g. EGNOS (European Geostationary Navigation Overlay Service) can be used to improve the performance even more.

However, it must be noticed that the geo-location accuracy, is influenced mostly by the ionospheres' infraction factor, than by the time accuracy factor.

3.4 The PPS (Pulse per Second) Signal

This is a repeating electrical TTL (Transistor-Transistor Logic) signal composed by a dual leveled voltage schema. Its "LOW" state is determined when its voltage is ranged 0-0.8 Volts, else its "HIGH" state is ranged 2-5 Volts. If a TTL signal appears to be ranged between 2-3 volts, it is considered to be invalid as the TTL signal purpose is to manipulate electronic circuits to behave precise predictable.

TTL is used by many devices such as precision oscillators, Radio beacons, DTV (Digital Television), Telecommunication systems and time measurement systems, where the time synchronization accuracy is very important.

Referring on the hardware specifications, the ITU (International Telecommunication Union), recommends with the G.703 G.8271/Y.1366 (4/2016) physical layer Specifications, that identically, the cable carrying the PPS Signal should not be extended more than 3 meters long. Preferably it should be a balanced-twisted 100 ohm cable (such as utp/stp cat5), which inserts a delay of almost 5 nanosecond/meter. Alternatively, an unbalanced cable 50 ohm can be used or even a coaxial. Also, ITU recommends that the Pulse width should be ranged between 100-500nS. As the sharp edges of the PPS Signal, it has a wide-band spectrum nature, it is difficult to manipulated, without distortion.

Every additive meter on the cable length, introduces a delay of almost 5 nanoseconds, and also increases the Capacitance of the cable, which in turn, is inserting deviation at the PPS signal as is shown on Figure 10.

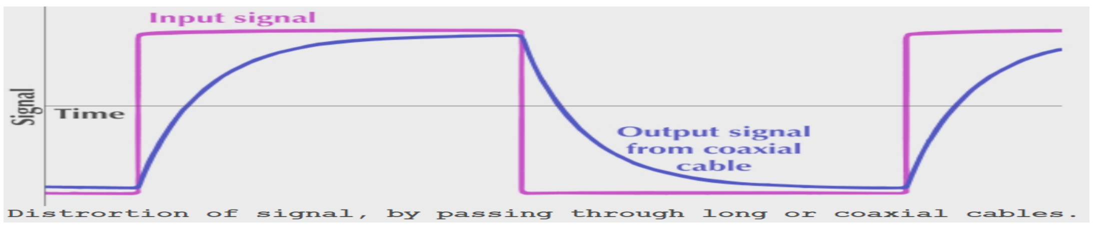


Figure 10 Deviation at the PPS signal through Coaxial cable

So, by choosing an inappropriate coaxial cable, its "velocity factor" characteristic, can influent the signal delay for 3.7-5 nanoseconds /meter. Optimally, a dedicated fiber optic cable should be used.

The PPS Signal, can exported through GNSS receiving equipment, while the GNSS Satellite constellations continuously transmit it the named “L1” and “L2” Radio Signals providing to us our Geographical location information.

The PPS Signal, which can be considered as a “sub product” of the GNSS Systems, can provide us at the worst scenario, time accuracy of a few microseconds per second [16].

On this Thesis, the PPS Signal is exported from a GPS Receiving module pin and it is connected as an input on a dedicated pin of the ESP32 Microcontroller. This pin is configured on the ESP32, as an “IRQ enabled pin”, triggered by the pulse raising state. It should be noticed that the ESP32 IRQ inserts a possible notable delay of 1.7-2μS [17].

The connection between GPS receiving module & ESP32 is established through a 5- lined 5m shielded cable. Extra precaution was taken during the cables construction, so that the PPS cables used at the project would have exactly the same length in order to equalize the line delay errors similarly to all the Receiving Stations.

This cable also contains 2 conductors for the supply of 5 volts to the GPS module and 2 conductors needed for the RX/TX RS232 Signals. By default, the PPS Signal is configured to be a rising square pulse per second (although it can reach the maximum of 1 KHz rate). By using the U-Center software or dedicated “pubx” command messages, it is possible to configure the polarity (raising/falling edge pulse), the PPS period, the PPS length, and even set a counterbalance for the antenna cable delay and set a user delay parameter.

At the moment that the PPS signal is emitted by the GPS module, concurrently and with a few milliseconds delay, is serially emitted a data message through the TX pin (of the GPS Module).

On figure 11 is presented the correlation of the PPS edge with the shifted data message serially coming out. This data message (NMEA) containing a identification of the pulse will be described at a next chapter (4.7.2).

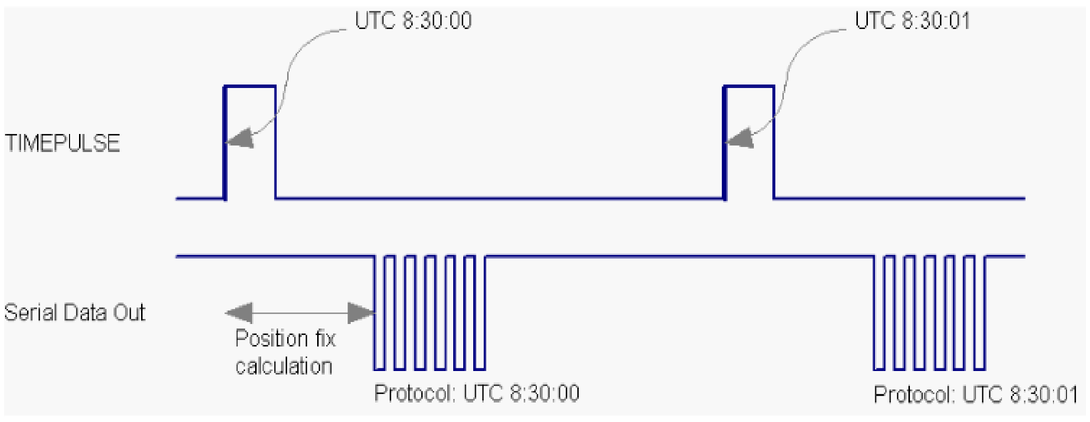


Figure 11 Correlation between the PPS edge and the data

Chapter 4: Receiving Station’s Composition

4.1 Constructing Receiving Stations

As it is shown on the figures 12, 13 below, each one of the five constructed stations, was similarly composed by this interconnected hardware:

An ESP32 microcontroller,

A GPS Receiver module NEO-8M (with its GPS antenna),

A micro-SD card reader module (with its SD card),

An Si4730 RF receiver module (with its AM/FM antennas),

A 3W audio amplifier (with its speaker),

A 7volt power supply with 2x18650 type rechargeable batteries, and a charging circuit module.

A 3.3-volt Relay to disconnect the power supply during the metering, avoiding electric noise.

The total cost of each Receiving Station, was calculated at about 40€.

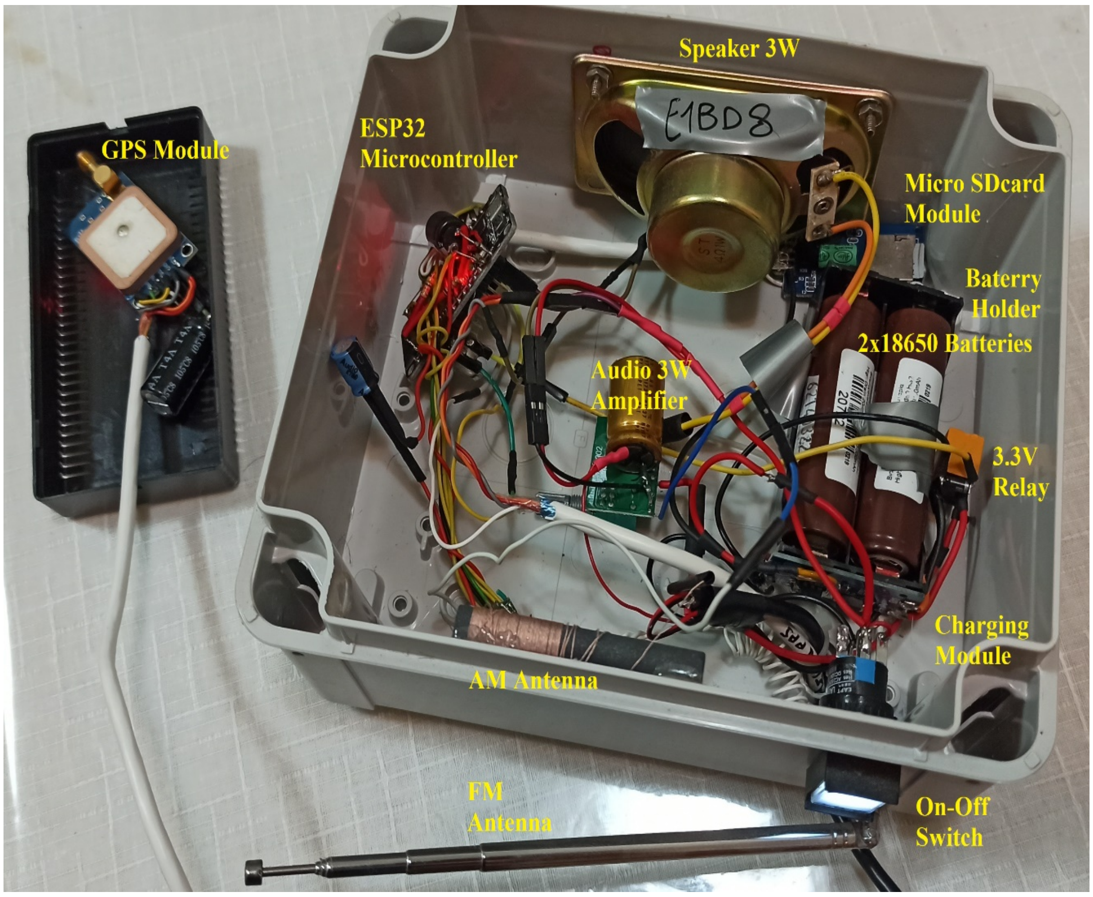


Figure 12 An assembled Receiving Station

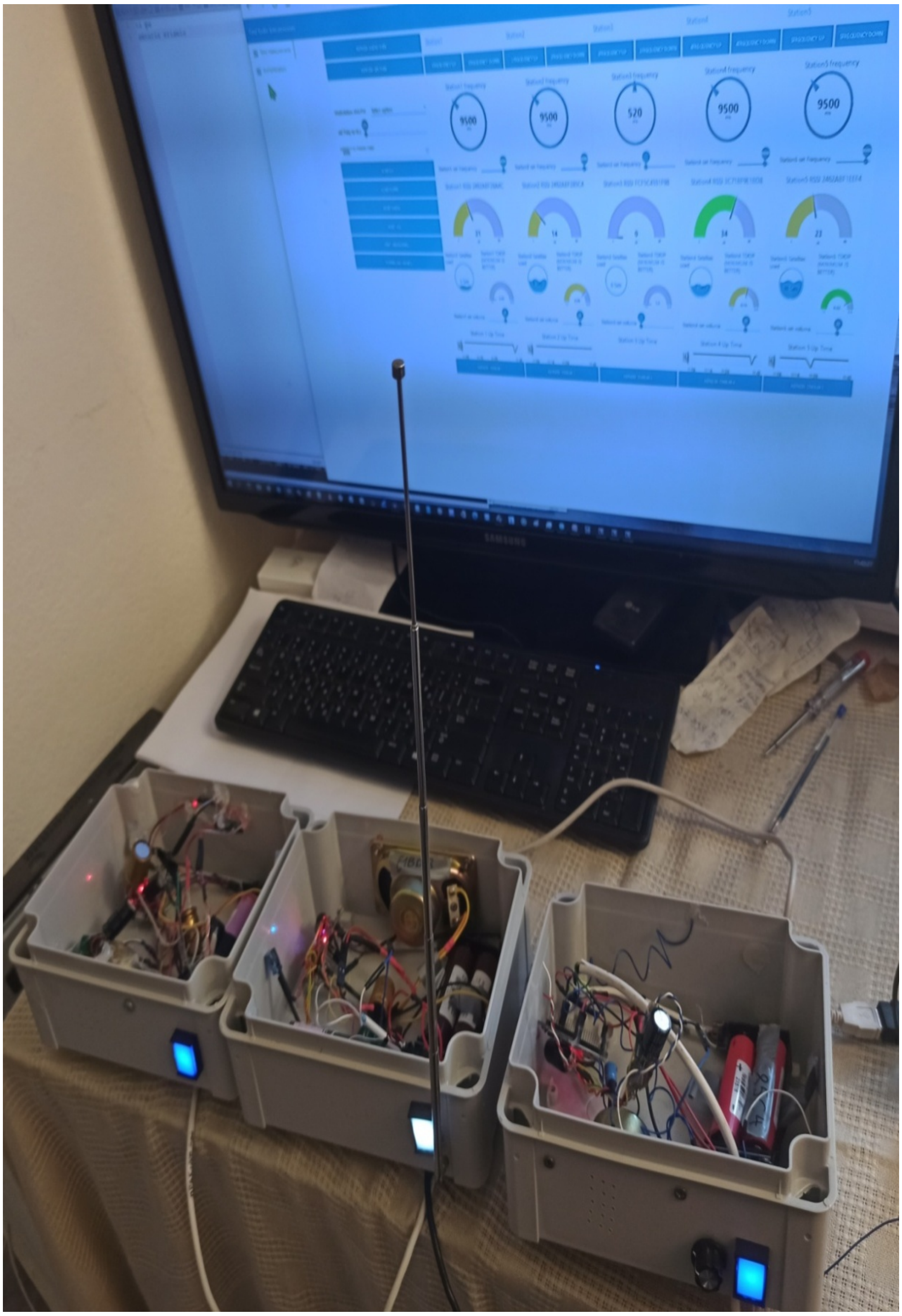


Figure 13 Three of the Receiving Stations

4.2 The ESP32 microcontroller

On this Thesis, the ESP32-Wroom-32 (30pin) was chosen, a fast, many featured, ultra-low-power, 40 nm Microcontroller Board. It is based on Espressif ‘s ESP32 Microcontroller, which uses eXtens 32-bit LX6 Dual Core microprocessor 240MHz - 600 MIPS with 4 MB of Flash and 520KB Ram Memory, which is preferably used on many IoT (Internet of Things applications). From all the ESP32 Microcontroller’s provided features, what was used was:

- The 802.11 b/g/n Wi-Fi WPA Secured communication, where the 802.11n (2.4 GHz), supporting up to 150 Mbps.
- The SPI (Serial Peripheral Interface) for the fast communication with the micro SD Card module.
- The i2C (Inter-Integrated Circuit) Interface for the communication with the Si4735 Radio Receiver Module.
- The ADC (Analog to Digital Converter) in order to convert the received audio signal to a file. At the beginning, instead of using ADC, the I2S (Inter-IC Sound) capability was intended to be used for the direct usage of the Sound signal on its Digital form. The Si4730 chip, normally can output either only analog, only digital (I2S), or either both of these audio signal formats concurrently.

  Unfortunately, although many attempts were performed, it was not possible to operate the Si4730 chip to output digital I2S signal. This way, it was necessary to use ADC, although the sound quality was lower.
- The UART2 (Universal Asynchronous Receiver/Transmitter) interface was used for the serial communication with the GPS Receiver Module.
- The UART0 interface which was used, only during the debugging time when the Microcontroller was connected with USB.

The ESP32-wroom microcontroller used and the Functional block diagram can be seen on Figure 14.

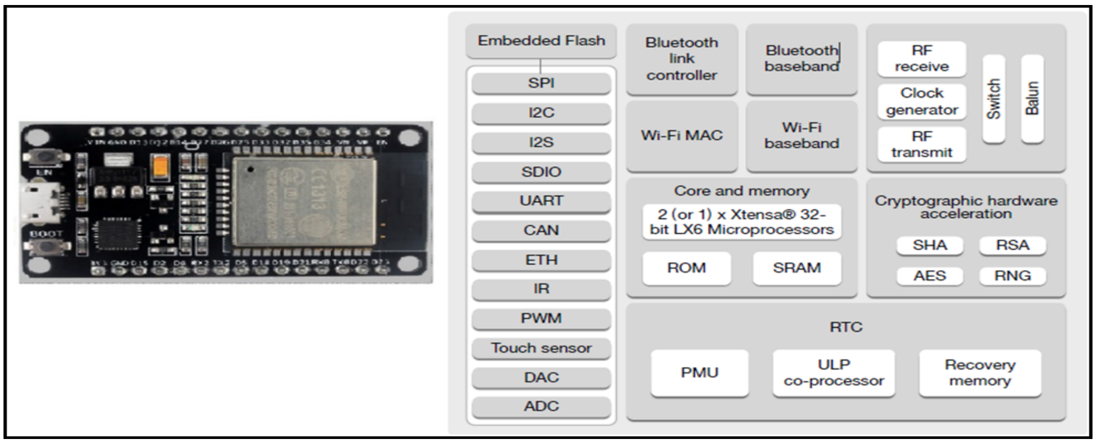
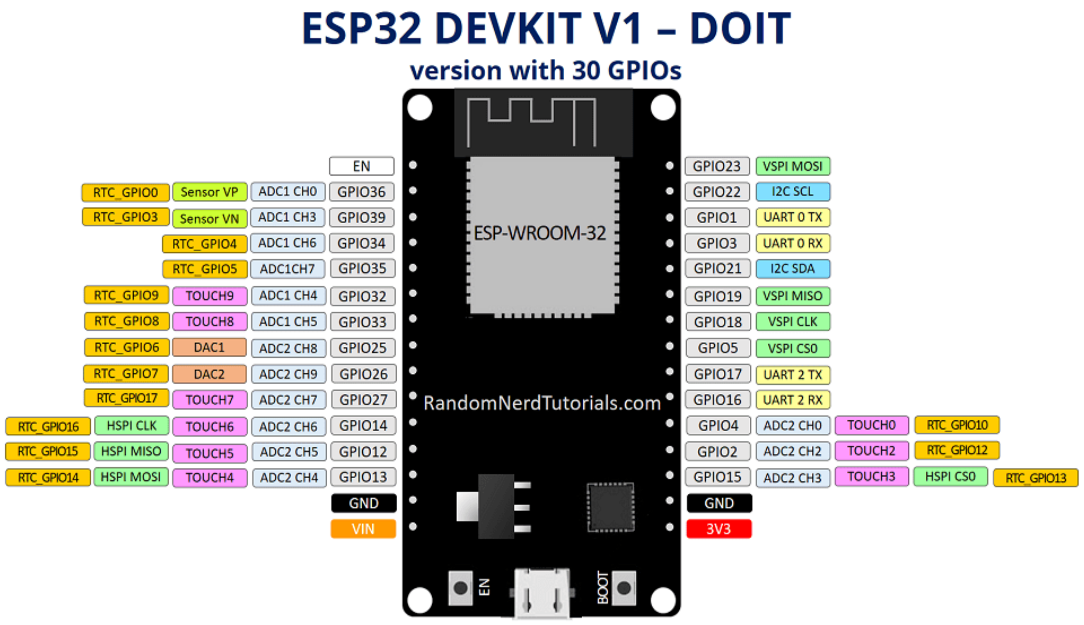


Figure 14 ESP32 wroom Chip, & Functional block Diagram

The Pins diagram of the ESP32 microcontroller can be seen on Figure 15.



ESP32 DEVKIT V1 – DOIT
version with 30 GPIOs
EN
RTC_GPIO0 Sensor VP ADC1 CH0 GPIO36
RTC_GPIO3 Sensor VN ADC1 CH3 GPIO39
RTC_GPIO4 ADC1 CH6 GPIO34
RTC_GPIO5 ADC1CH7 GPIO35
RTC_GPIO9 TOUCH9 ADC1 CH4 GPIO32
RTC_GPIO8 TOUCH8 ADC1 CH5 GPIO33
RTC_GPIO6 DAC1 ADC2 CH8 GPIO25
RTC_GPIO7 DAC2 ADC2 CH9 GPIO26
RTC_GPIO17 TOUCH7 ADC2 CH7 GPIO27
RTC_GPIO16 HSPI CLK TOUCH6 ADC2 CH6 GPIO14
RTC_GPIO15 HSPI MISO TOUCH5 ADC2 CH5 GPIO12
RTC_GPIO14 HSPI MOSI TOUCH4 ADC2 CH4 GPIO13
GND
VIN
GPIO23 VSPI MOSI
GPIO22 I2C SCL
GPIO1 UART 0 TX
GPIO3 UART 0 RX
GPIO21 I2C SDA
GPIO19 VSPI MISO
GPIO18 VSPI CLK
GPIO5 VSPI CS0
GPIO17 UART 2 TX
GPIO16 UART 2 RX
GPIO4 ADC2 CH0 TOUCH0 RTC_GPIO10
GPIO2 ADC2 CH2 TOUCH2 RTC_GPIO12
GPIO15 ADC2 CH3 TOUCH3 HSPI CS0 RTC_GPIO13
GND
3V3
ESP-WROOM-32
RandomNerdTutorials.com
EN
BOOT

Figure 15 Pins Diagram of ESP32

The ESP32 Microcontroller supports the pin multiplexing feature, meaning that the same pin can be used with many uses. On this Thesis, the pins that were used are described below:

Pin 36 was used as o voltmeter for the metering of the 5-volt power supply or the battery. With the appropriate command, the user can remotely ask for a battery or a power supply measurement. With the manipulation of a dedicated (at Pin 32) Relay, when it is not activated, the Pin36 measures the voltage while the power supply is connected to the batteries. On the other hand, when the Relay gets activated, the power supply is disconnected for a few seconds, a battery metering starts, and afterwards the power supply is connected again. As all the ESP32 pins are only 3.3Volt maximum tolerated, a voltage divider was used to make the 5-volt compatibility.

Pin 34 was used to input the IRQ (Interrupt Request) PPS signal from the GPS module. Although most of the ESP32 ports support the 45KOhm onboard pull-up/ pull-down capability, Pin 34 (as also pins 35,36,39) does not have this capability. So, an external resistor 47KOhm was used in place as a pull-down to keep the port at the "LOW" state when not PPS pulse was not present.

The IRQ is a dedicated hardware signal which when it reaches a processor, it stops temporarily any program flow, run a separate procedure called "interrupt handler", and afterwards it returns to its previous program routine. IRQ can appear in two forms: Softwarely as a Timer Interrupt/Watchdog, either hardwarely as a Change on a Pin state. The advantage of using IRQs, is that we get the fastest CPU response on an important event such as the PPS appearance. Although IRQ is very fast, however it could import a delay response at maximum of 2 microseconds (for ESP32). This delay could influent the TDoA results, injecting a distance deviation of about 600 meters at the measurements.

Pin 35 was used as an input for the sound coming from the Si4730 Receiver module. It was used as an ADC (Analog to Digital Converter) and 11Db attenuation was chosen from the library. Also a 1Kohm resistor for more attenuation, and also a 470μF Audio Capacitor was used for better sound quality and DC voltage removal. As Wi-Fi operation was continuously in use, all the ADC2 pins 15, 2, 4, 13, 12, 14, 27, 26, 25 could not be used. Only ADC1 pins 36, 39, 34, 35, 32, 33 were available.

Pin 32 was used to manipulate a 3volt Relay in order to remotely disconnect the power supply during the Station metering procedures, in order to avoid the noise passage from the AC network and improve Signal measuring quality.

Also, this Relay was used during the measuring of the Battery Voltage. By forcing a Power Supply disconnection for 4 seconds, it was possible to measure separately just the battery level (through the Pin36). Otherwise, without activating the Pin 32, the Pin 36 is measuring the Power Supply applied voltage.

As the ESP32 pin can afford only 3.3 volt, a voltage divider with resistors, was used to adapt the 5-6 voltage of the battery, to the safe 3.3-volt range.

Pin 33 was used for the RESET operation of the Si4730 Receiver module.

Pin 23 was used for the MOSI Signal of the SD-Card module (SPI).

Pin 22 was used for the CLOCK Signal of the Si4730 Receiver module (i2c).

Pin 21 was used for the DATA transfer to the Si4730 Receiver module (i2c).

Pin 19 was used for the MISO Signal of the SD-Card module (SPI).

Pin 18 was used for the CLOCK Signal of the SD-Card module (SPI).

Pin 05 was used for the CS Signal of the SD-Card module (SPI).

Pin 17 was used for the TX Hardware UART2 for the communication with the GNSS Module.

Pin 16 was used for the RX Hardware UART2 for the communication with the GNSS Module.

***The UART0 (pins 1,3)** could not be used, as they were occupied for ESP32 flashing and communication through USB connection.

4.3 The GNSS (GPS) NEO-8M module

The GNSS (Global Navigation Satellite Systems), such as GPS, is used in order to get a precise location, GPS Time, and the PPS (Pulse per Second) signal. Many GPS satellites can be seen on the sky at a huge angle tilt range (almost 80°) from the zenith axis. In any case, this Thesis' GPS modules were configured to only listen to satellites located at a little tilt angle of 15°-20° degrees from the zenith axis, in order to improve the satellite PPS signal accuracy, and eliminate the corruption, due to the huge ionospheres' propagation signal errors, Solar Flares [24], multipath reflection errors and other obstacles as shown on figure 16.

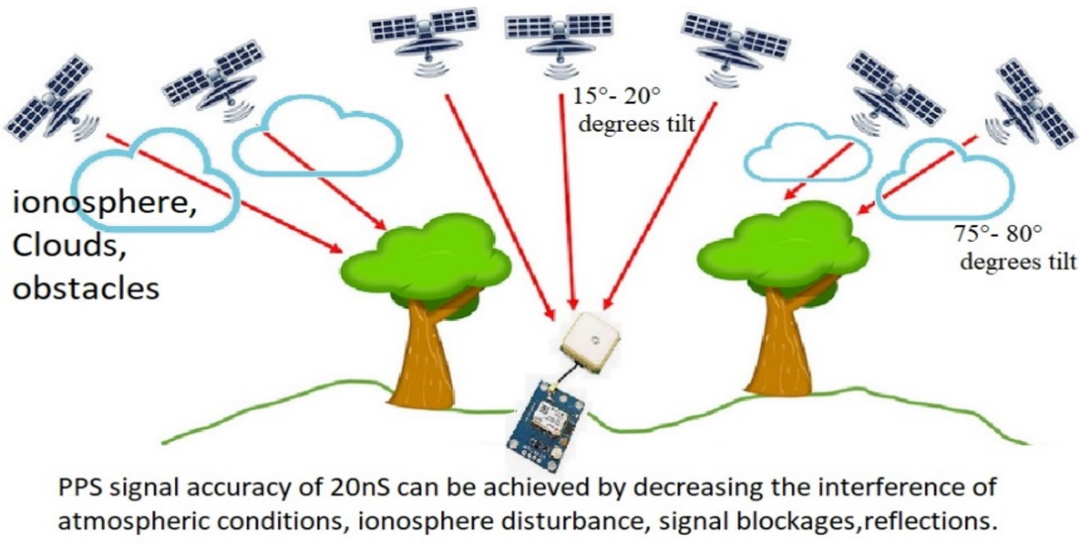

ionosphere,
Clouds,
obstacles
15°- 20°
degrees tilt
75°- 80°
degrees tilt
PPS signal accuracy of 20nS can be achieved by decreasing the interference of atmospheric conditions, ionosphere disturbance, signal blockages,reflections.

Figure 16 Possible reasons of Signal Errors

On this Thesis, the communication between GPS Module and ESP32, was done through the Serial (UART) port by using the NMEA (National Marine Electronics Association) protocol. Also, the PPS signal was taken from the dedicated pin of the GPS module and was driven to the pin 34 of ESP32 as an IRQ input. On Figure 17 is shown the NEO-8M GPS module that was used.

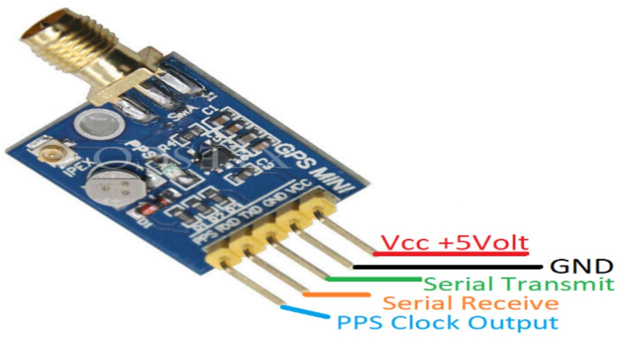


Figure 17 NEO-8M GPS module

Although this GPS module needs 5-volt supply and consumes almost 60mA, its TX output has a 3.3-volt threshold, so there is no need for voltage divider. The output signal coming out from its PPS port is a 3.3-volt pulse, and can be connected straight to ESP32 input port without need for a voltage divider. However, the ESP32 PPS input port is needed to be concurrently connected with a pull-down resistor (e.g. 47K), otherwise the port's state was floating. (ESP32 does not provide the capability of using onboard pull-down option for the pin 34). A 5 meters' cable is used to connect more distantly the GPS module with the ESP32, in order to provide flexibility to the GPS module, for a better possible placement to a clear sky view with no obstacles.

The connection is ESP32 and GPS module is shown below on figure 18.

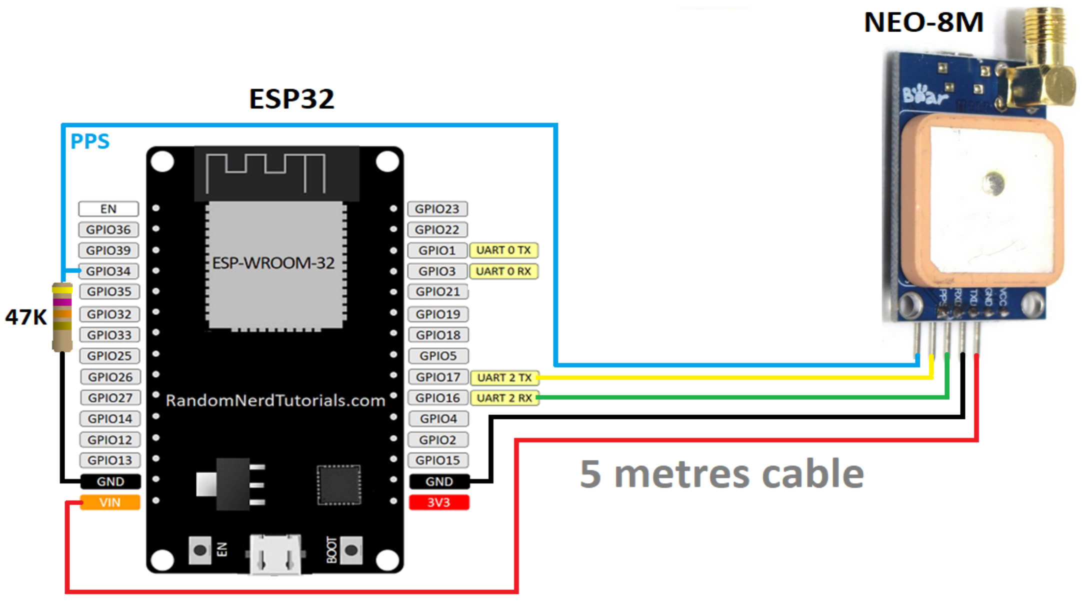
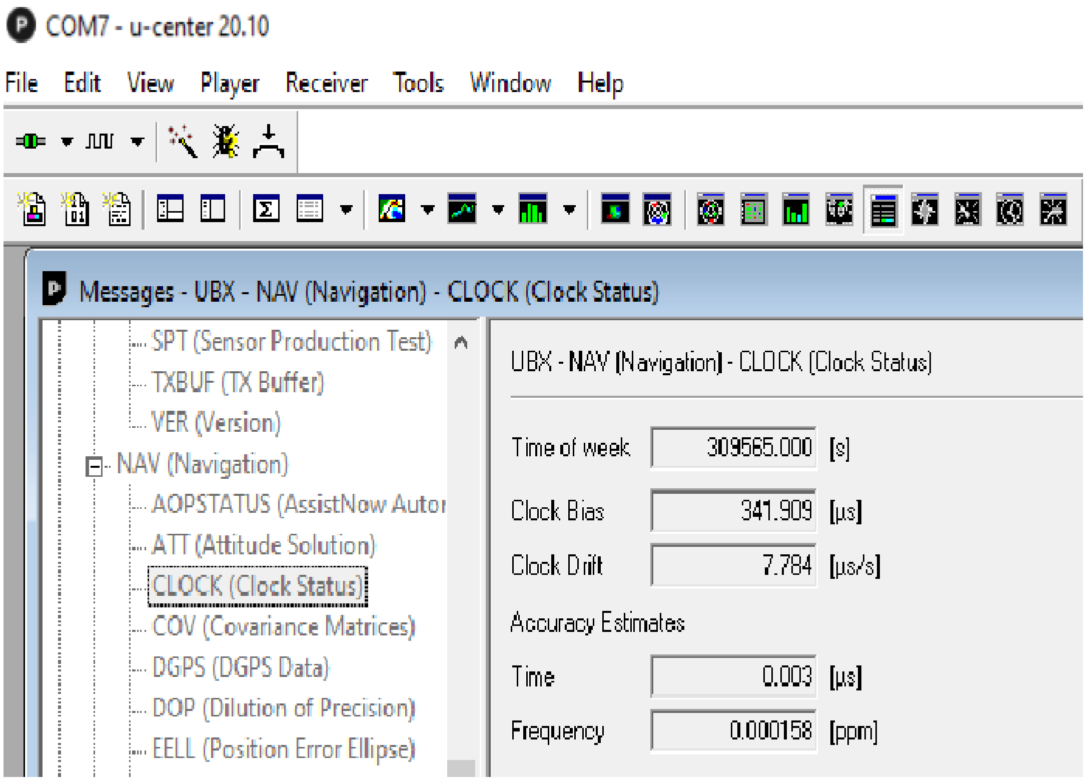


Figure 18 ESP32 & NEO-8M GPS module connection

The software which was used to watch and manage the GPS basic parameters was the "U-center" software, which is provided as freeware by the GNSS module Company. The communication with the GPS module was made through the UART Serial port. As it is shown on Figure 19 on a U-center software screenshot, after a long time of operation with good weather conditions and sky view, the GNSS module, can give time accuracy of 0.003μS (3nS) and frequency stability of 0.000158 ppm.


COM7 - u-center 20.10
File
Edit
View
Player
Receiver
Tools
Window
Help
Messages - UBX - NAV (Navigation) - CLOCK (Clock Status)
SPT (Sensor Production Test)
TXBUF (TX Buffer)
VER (Version)
NAV (Navigation)
AOPSTATUS (AssistNow Autor
ATT (Attitude Solution)
CLOCK (Clock Status)
COV (Covariance Matrices)
DGPS (DGPS Data)
DOP (Dilution of Precision)
EELL (Position Error Ellipse)

UBX - NAV (Navigation) - CLOCK (Clock Status)

Time of week 309565.000 [s]

Clock Bias 341.909 [μs]

Clock Drift 7.784 [μs/s]

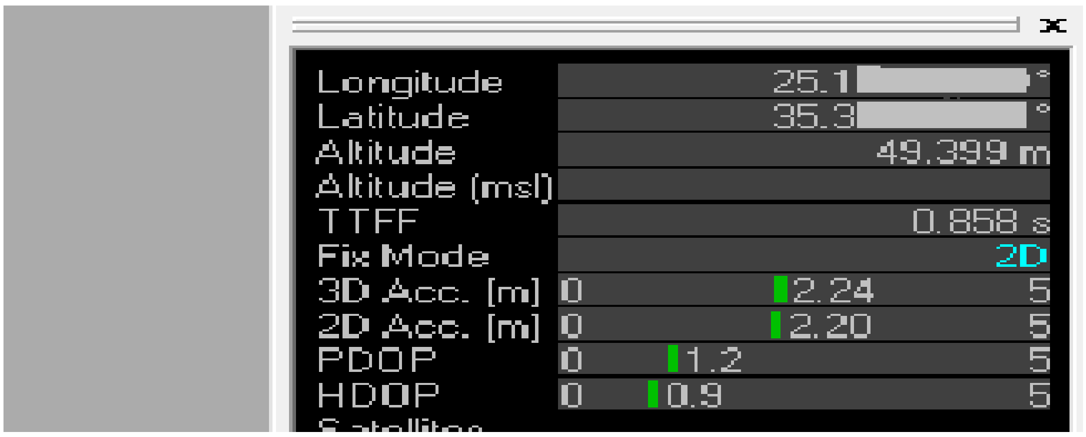
Accuracy Estimates

Time 0.003 [μs]

Frequency 0.000158 [ppm]

Figure 19 Time-Frequency accuracy U-center software screenshot

Also, as concerned the location accuracy provided by the GPS module, as it is shown on Figure 20, the PDOP (Position dilution of precision) and HDOP (Horizontal dilution of precision) parameters, appear to be very satisfying such as 1.2 meters & 0.9 meters.



```
Longitude          25.1[redacted] °
Latitude           35.3[redacted] °
Altitude                 49.399 m
Altitude (msl)
TTFF                      0.858 s
Fix Mode                       2D
3D Acc. [m]    0      2.24      5
2D Acc. [m]    0      2.20      5
PDOP           0    1.2         5
HDOP           0   0.9          5
```

Figure 20 HDOP – PDOP accuracy U-center software screenshot

4.4 The Micro SD Card Module

A micro-SD card module, as it is shown on figure 21, was used on each Station, in order to store the audio sample captured.

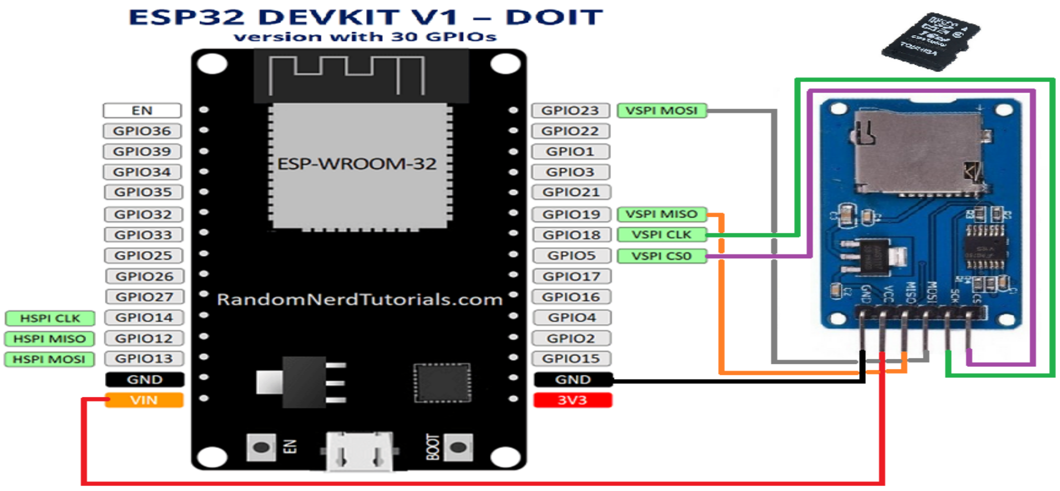


Figure 21 Micro SD card module connection

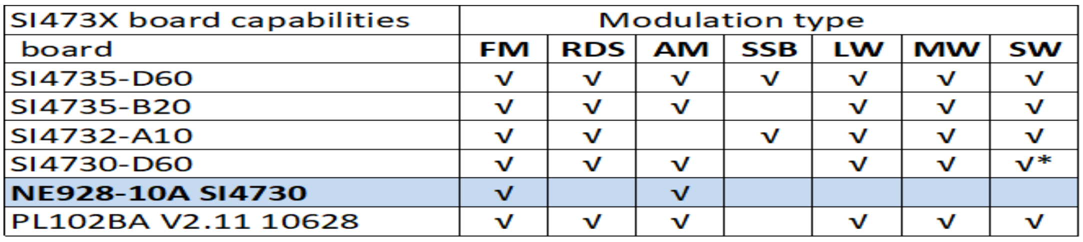
This type of storing was used because it was very fast. Although the Microcontroller's RAM memory is faster, its size is tiny and insufficient to be used. The communication of the Microcontroller with the micro-SD card module was done through SPI (Serial Peripheral Interface). SPI is a fast bus for data transfer (almost 5-50 times faster than I2C).

The Si4730 Receiver module receives the RF signal, and outputs the received analog audio signal to the ESP32. Then, the ESP32, by using its ADC, stores this audio as a file, on the micro-SD card. Afterwards, the Sound file is sent through an email server, to a Raspberry Pi Server, for further processing.

In order to minimize the card access delay, the quite fast SD card model “TRANSCEND 300S TS16GUSD300S-A 16GB MICRO SDHC UHS-I U3 V30 A1” was used, which is documented to support writing speed of 45 MB/s. However, the ESP32 SPI clock is documented to support 40MHz. The SD card module needs 5-volt source for its operation which is supplied by the ESP32.

4.5 The Si4730 Receiver module

The Si473x family electronic chips are Radio receiving chips which are able to receive many types of RF modulations such as AM, FM, SSB, LW, MW, SW as is showed on Table 2, and are capable to output either Analog, Digital, or both audio forms as is showed on its functional block diagram on Figure 22.



SI473X board capabilities	Modulation type						
board	FM	RDS	AM	SSB	LW	MW	SW
SI4735-D60	V	V	V	V	V	V	V
SI4735-B20	V	V	V		V	V	V
SI4732-A10	V	V		V	V	V	V
SI4730-D60	V	V	V		V	V	V*
NE928-10A SI4730	V		V				
PL102BA V2.11 10628	V	V	V		V	V	V

Table 2 Radio receiving chips family

The operational commands are identical for all the Si473x versions.

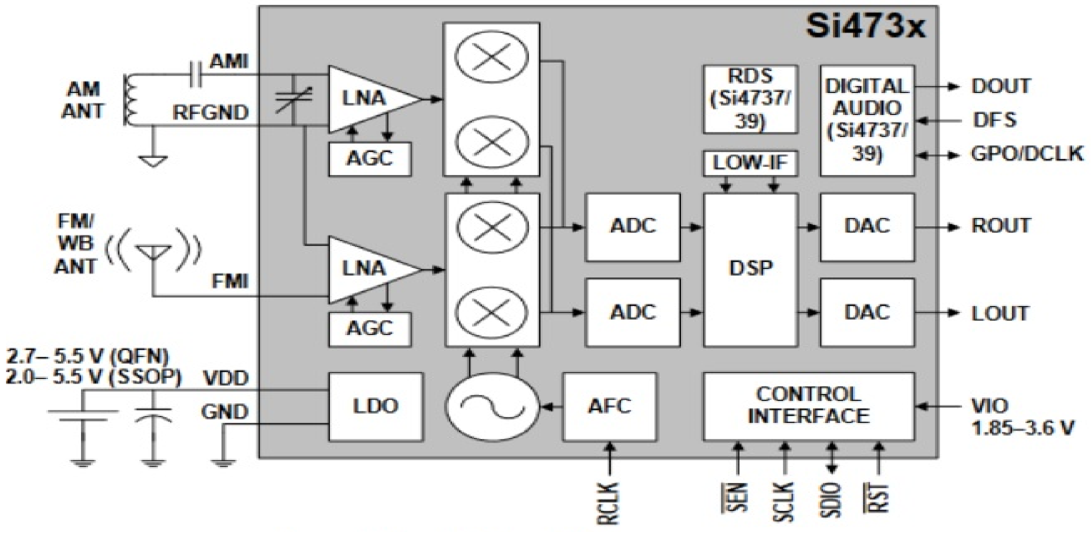


Figure 22 Si473x Functional block diagram

Even though the multi-modulation capable receiving module Si4735-D60 was searched at the shops, it was not found at a modular form, so the **NE928-10ASi4730** module was used instead. This module, shown on Figure 23, is capable of only receiving AM (520–1710 kHz) and FM (64–108 MHz) modulations.

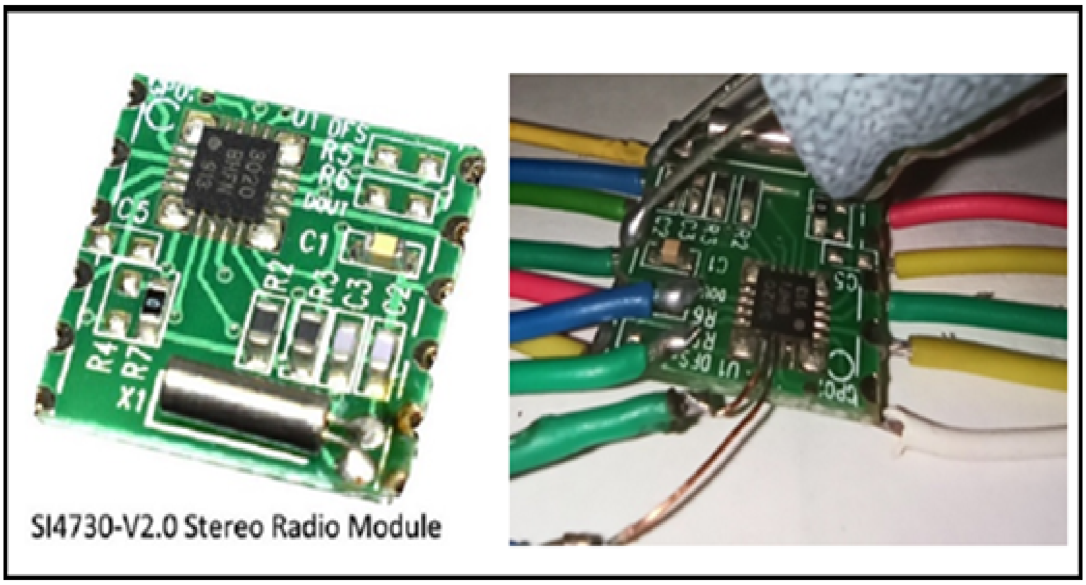


Figure 23 Si4730 Module and soldering attemption to use DOUT, DFS, DCLK

Although it was tried at the beginning, by soldering extra cables to the dedicated unconnected chip pins DOUT, DFS, DCLK of the board to extract the audio output on its digital (I2S) form, it was not achieved. This could optimally offer better audio signal quality. So, finally, the analog outputs ROUT / LOUT were used.

The communication with the ESP32 is achieved through the i2c bus by using the lines RESET=33, SCL=22, SDA=21, and SENS=none as is shown on Figure 24.

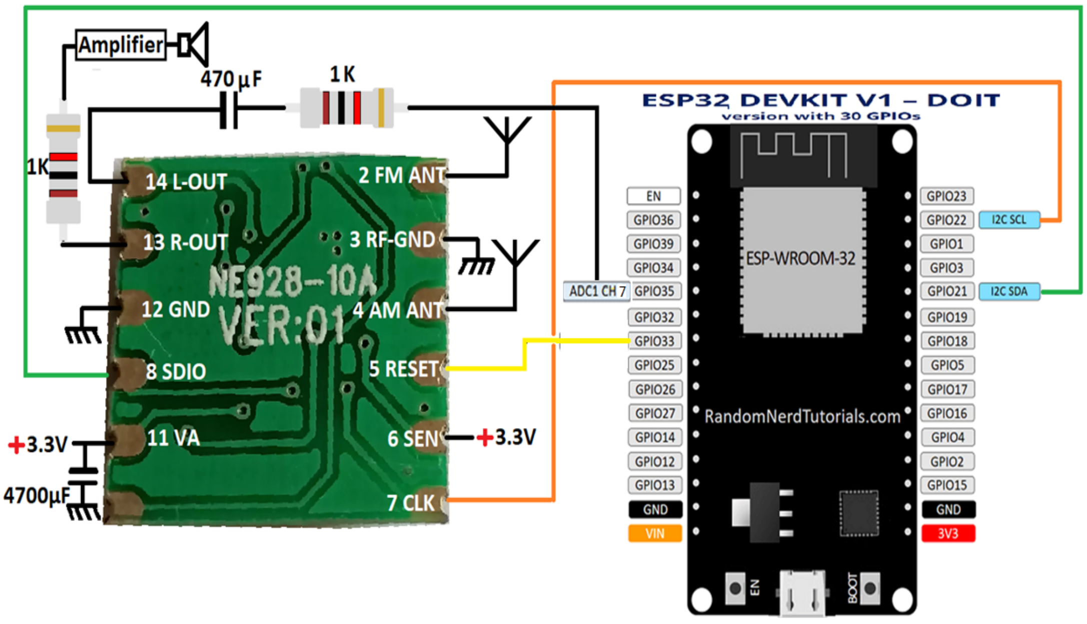


Figure 24 NE928-10A Si4730 Receiver Module connection

With the use of SEN signal state, it is supposed that the i2c address of the Si4730 module is selectable of either 0x11 or 0x63, but actually only the 0x63 address was operational. Two antennas are used, for AM and FM receiving.

The configured to MONO (not STEREO) audio signal exported from the Receiver's 13 and 14 pins, is routed to an amplifier and a speaker, but also to the ADC1 CH5 input on pin GPIO35 of the ESP32. This signal amplitude value is critical, as if it is too high, it overcomes the ADC threshold, and the recorded signal is saturated. This amplitude is configurable by the available commands e.g. "*si4735.setVolume(37)*".

4.6 Other Components that were used and Powering

Other complementary hardware that was also used at the Receiving Stations, are a simple 3W audio amplifier (a), a 3W speaker (b), a battery charging module (c), a battery case (d), two 18650 Li-Ion rechargeable batteries (e), an on-off switch (f), a 3.3v Relay (g), a 3A 7V power supply and plastic boxes. All these are shown on Figure 25.

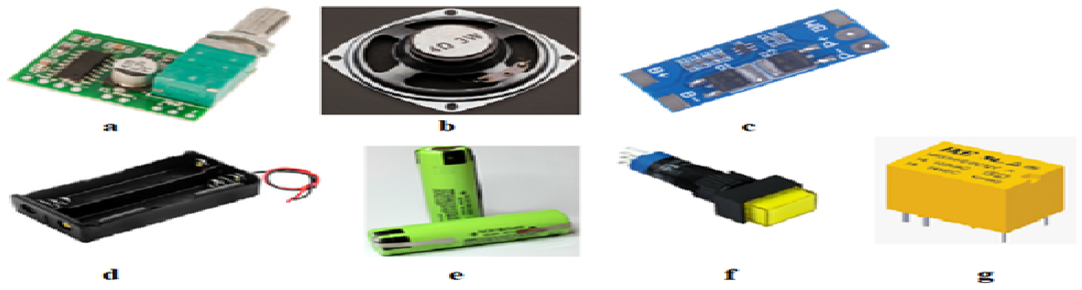


Figure 25 Other components that were used

Especially, for the power supply connection, it was programmed that during the few seconds measuring period, the GPIO32 is activated and the N.C. (Normally Closed) Relay disconnects totally the Power supply from the circuit, avoiding any AC noise passage. Thus, the circuit is powered only from the two batteries as the measuring lasts. A flyback diode is also placed to prevent overvoltage spikes. This connection is shown of Figure 26.

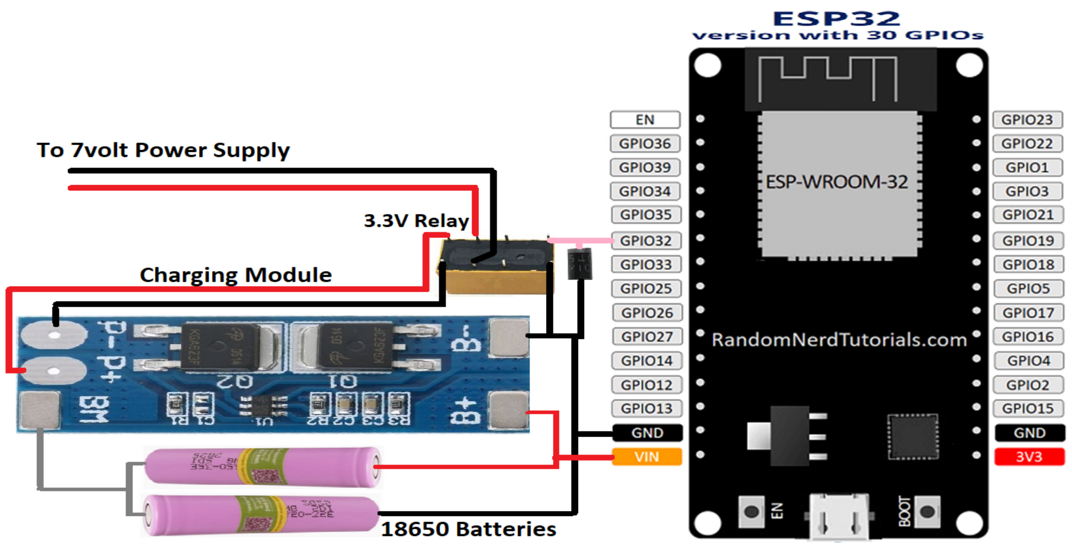


Figure 26 Power supply connection

4.7 Protocols and Libraries used

4.7.1 The UART (Serial) Protocol

This old, slow, but famous simplex/duplex communication protocol, usually known as RS-232, is broadly used despite its maximum baud rate of usually 115200 bits/sec, because of its compatibility with many devices. No extra bus cable is being need for the timing, as the Clock information can be extracted through the dedicated bits of the transferred signal. The ESP32 Microcontroller is equipped with three UART ports U**O**UXD, U**1**UXD, U**2**UXD of which, the U**O**UXD is reserved only for communication and programming through USB, and the other two are available for the user to reclaim them.

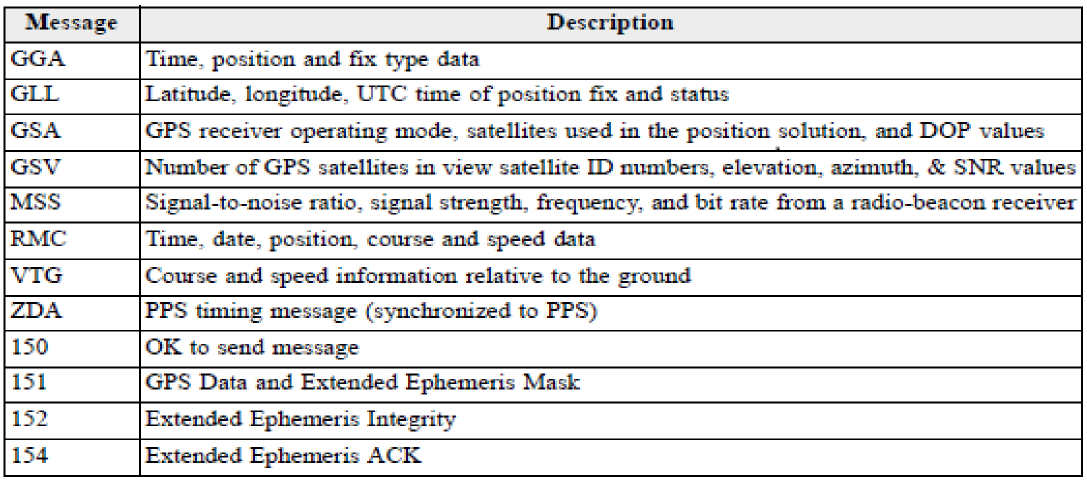
On this Thesis, the ESP32 Hardware Serial U**2**UXD on GPIO16 for RX, and GPIO17 for TX was used for the communication with the GPS module, in order to transfer the coordinates, the time information, and other initializing data. The encoding protocol for this communication is NMEA as is described later. U**O**UXD was used only during debugging through USB.

4.7.2 The NMEA (National Marine Electronics Association) Protocol

This protocol is widely used for the communication within marine electronic equipment, such as echo sounder, sonar, anemometers, gyrocompass, autopilots and mainly GPS devices. This protocol is using the UART Protocol as a carriage, but also other carriages can be used such as Bluetooth, or TCP/IP.

NMEA uses simple ASCI phrases such as: **"$GPZDA, hhmmss.ss, dd, mm ,yyyy, xx, yy*cc** " to transfer the desired information between devices.

At the beginning of each NMEA phrase, there is always a "**$**" character, and at the end there is always an asterisk "*****". The asterisk is used, to mark the separation of the usable message, from the XOR checksum. On Table 3 it is shown some of the more commonly used NMEA commands.



Message	Description
GGA	Time, position and fix type data
GLL	Latitude, longitude, UTC time of position fix and status
GSA	GPS receiver operating mode, satellites used in the position solution, and DOP values
GSV	Number of GPS satellites in view satellite ID numbers, elevation, azimuth, & SNR values
MSS	Signal-to-noise ratio, signal strength, frequency, and bit rate from a radio-beacon receiver
RMC	Time, date, position, course and speed data
VTG	Course and speed information relative to the ground
ZDA	PPS timing message (synchronized to PPS)
150	OK to send message
151	GPS Data and Extended Ephemeris Mask
152	Extended Ephemeris Integrity
154	Extended Ephemeris ACK

Table 3 Frequently used NMEA commands

All the GPS modules, are compatible with most of the available NMEA phrases.

Especially only for the “$GPZDA” phrase, must be noticed that it is always exported by the GPS modules, in correlation with its PPS Signal. Thus, just after the PPS pulse export, a few milliseconds later, the “GPZDA” message, containing the time data is following. All the other NMEA messages have no correlation to the PPS.

GPS time, is considered nowadays to be one of the best timing accuracy methods, pronounce it a “Stratum 0” time provider. By just reclaiming the NMEA messages and the PPS signals from a GPS module, the ESP Microcontroller can be considered like to be an almost “Stratum 1” operating device.

The four Stratum levels are used to describe the reliability of the UTC time and the distance from a time reference source. Thus, a Stratum 3 device can request a reliable time from Stratum 2. A Stratum 2 device can request reliable time from Stratum 1. A Stratum 1 device, can request a reliable time from Stratum 0. The Stratum 0 is only connected to primary time servers, providing the highest time accuracy.

Alternatively, to NMEA, the “UBX” binary protocol is also usually used for these equipment’s communications. Although “UBX” is more compact, fast, with 8 digits location precision, and has also XOR checksum, the NMEA protocol will be preferred in this Thesis, for the communication with the NEO-8M GPS module, in order to get the precise Station location and time. The reason is simplicity and the avoidance of the extra library’s usage.

Actually, although each GPS Module will be asked for its Location once at the beginning, this Location will not change regularly as it is placed on a stable location. So, there is no need for requesting this information regularly, flooding the communication between GPS module and ESP32. Also, a higher rate of broadcasting NMEA sentences to the Microcontroller, does not influent the precision, so it is not very important.

The “PPS” Signal is the most important factor which can influent the precision. By default, the GPS Module sends at the slow speed of 9600 bps through RS232, many NMEA such as “GGA, GLL, GSA, GSV, RMC, VTG, TXT”, and UBX packets. Some of these data take place unreasonably in the GPS module TX buffer, they force the Microcontroller to consume CPU time watching them, and they are unusable for this Thesis.

For the above reasons, the speed was configured to 115200 bps and some of these repeating transmissions e.g. GLL, RMC, GSA, GGA, VTG, GSV are disabled with the use of the appropriate commands. On the other hand, the repeating information from other sentences such as “GPZDA” packet will be preferred, as its message has a short structure, of only 8 fields as it is shown on Table 4, and contains the UTC time information needed.

Message stucture: $GPZDA,time,day,month,year,ltzh,ltzn*cs<CR><LF>
example: $GPZDA,003532.00,28,03,2021,00,00*69

Field No.	Name	Unit	Format	Example	Description
0	xxZDA	-	string	$GPZDA	ZDA Message ID (xx = current Talker ID, see NMEA Talker IDs table)
1	time	-	hhmmss.ss	082710.00	UTC Time. See the section UTC representation in the Integration manual for details.
2	day	day	dd	16	UTC day (range: 1-31)
3	month	month	mm	09	UTC month (range: 1-12)
4	year	year	yyyy	2002	UTC year
5	ltzh	-	xx	00	Local time zone hours (fixed field, always 00)
6	ltzn	-	zz	00	Local time zone minutes (fixed field, always 00)
7	cs	-	hexadecimal	*64	Checksum
8	<CR><LF>	-	character	-	Carriage return and line feed

Table 4 GPZDA message structure

After the arriving of the “GPZDA” sentence at the ESP32, it is filtered, so that from its 8 fields, only the No ‘1’ field which contains the UTC time is used. When we will need to obtain more special information, we can occasionally promote the GPS module by sending the appropriate command, in order to poll the data we need.

So, we can ask by sending “GPGSV”, to poll the number of Satellites viewed on sky, their Elevation, Azimuth, and their Signal strength. We can ask by sending “PUBX04” to poll Time of week, Receiver clock bias, Receiver clock drift, Time pulse pin quantization error. We can by sending “PUBX00” to poll Latitude, Longitude, Altitude, Navigation Status, and Dilution of Precision.

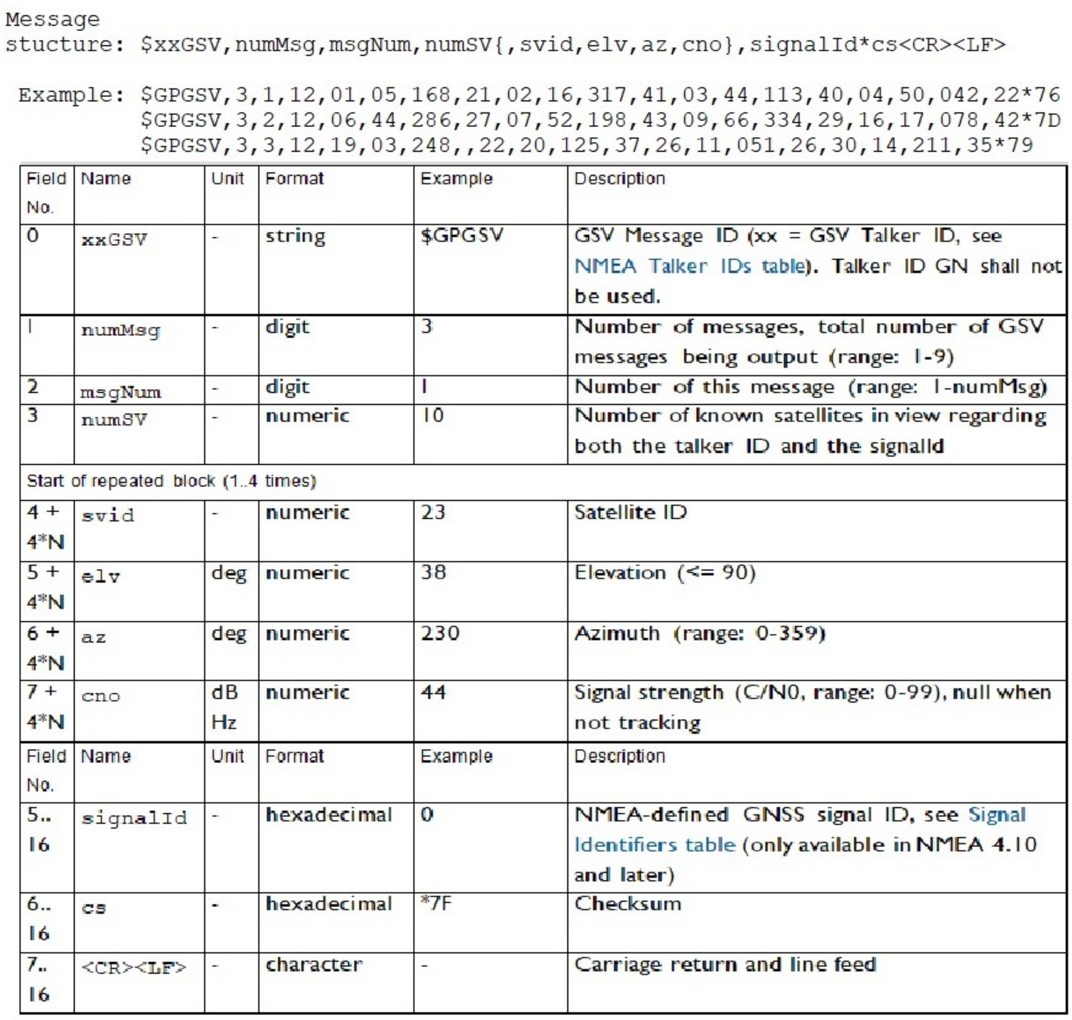
On Tables 5, 6, 7 we can see the “PUBX04”, “GPGSV”, and “PUBX00” messages structures.

Message stucture: $PUBX,04,time,date,utcTow,utcWk,leapSec,clkBias,clkDrift,
tpGran,*cs<CR><LF>

Example: $PUBX,04,073731.00,091202,113851.00,1196,15D,
1930035,-2660.664,43,*3C

Field No.	Name	Unit	Format	Example	Description
0	$PUBX	-	string	$PUBX	Message ID, UBX protocol header, proprietary sentence
1	msgId	-	numeric	04	Proprietary message identifier: 04
2	time	-	hhmmss.ss	073731.00	UTC time. See the section UTC representation in the Integration manual for details.
3	date	-	ddmmyy	091202	UTC date, day, month, year. See the section UTC representation in the Integration manual for details.
4	utcTow	s	numeric	113851.00	UTC time of week
5	utcWk	-	numeric	1196	UTC week number, continues beyond 1023
6	leapSec	s	numeric/text	15D	Leap seconds The number is marked with a D if the value is the firmware default value. If the value is not marked it has been received from a satellite.
7	clkBias	ns	numeric	1930035	Receiver clock bias
8	clkDrift	ns/s	numeric	-2660.664	Receiver clock drift
9	tpGran	ns	numeric	43	Time pulse granularity, the quantization error of the TIMEPULSE pin
10	cs	-	hexadecimal	*3C	Checksum
11	<CR><LF>	-	character	-	Carriage return and line feed

Table 5 PUBX04 message structure

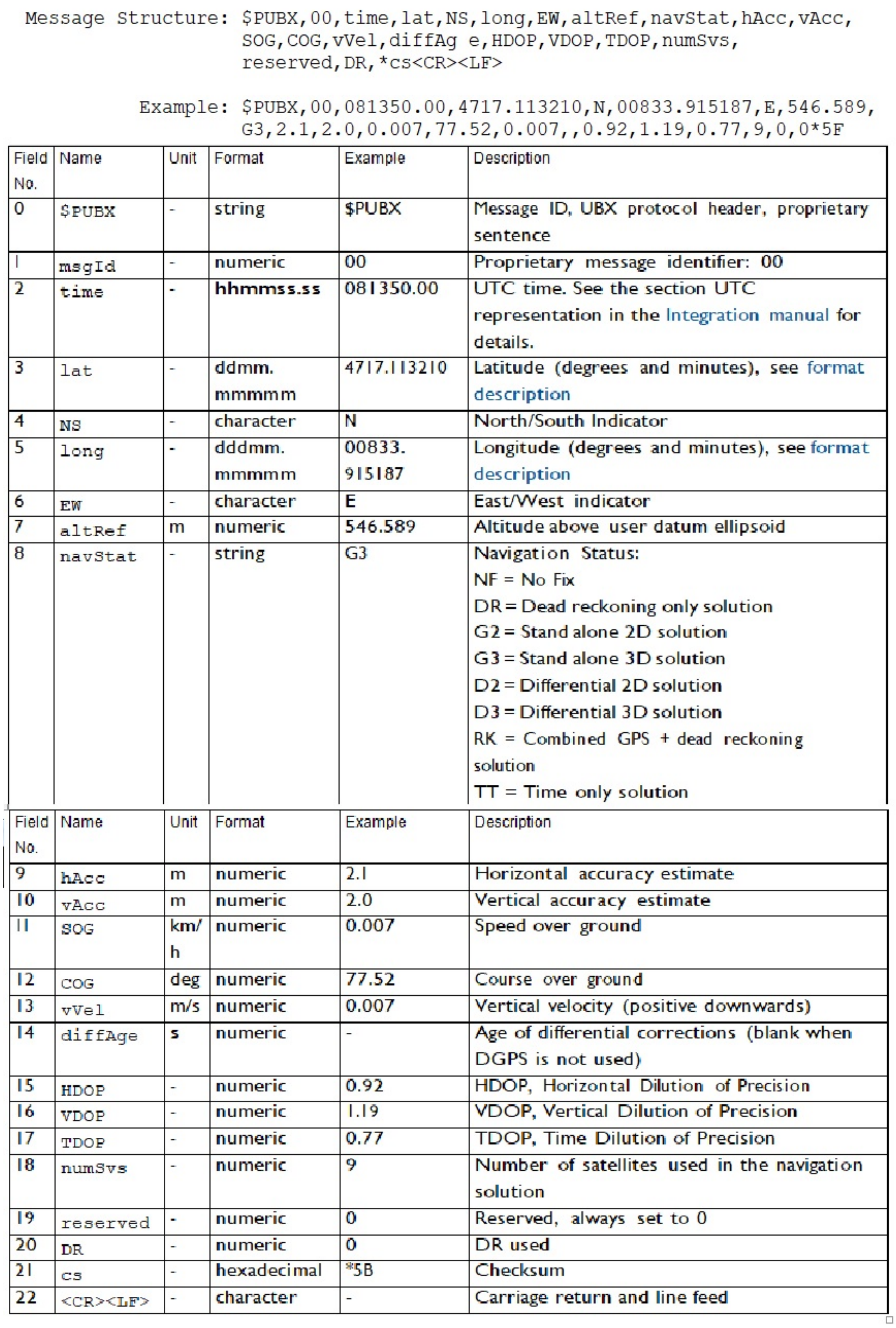


```
Message
stucture: $xxGSV,numMsg,msgNum,numSV{,svid,elv,az,cno},signalId*cs<CR><LF>

 Example: $GPGSV,3,1,12,01,05,168,21,02,16,317,41,03,44,113,40,04,50,042,22*76
          $GPGSV,3,2,12,06,44,286,27,07,52,198,43,09,66,334,29,16,17,078,42*7D
          $GPGSV,3,3,12,19,03,248,,22,20,125,37,26,11,051,26,30,14,211,35*79
```

Field No.	Name	Unit	Format	Example	Description
0	`xxGSV`	-	string	$GPGSV	GSV Message ID (xx = GSV Talker ID, see NMEA Talker IDs table). Talker ID GN shall not be used.
1	`numMsg`	-	digit	3	Number of messages, total number of GSV messages being output (range: 1-9)
2	`msgNum`	-	digit	1	Number of this message (range: 1-numMsg)
3	`numSV`	-	numeric	10	Number of known satellites in view regarding both the talker ID and the signalId
Start of repeated block (1..4 times)					
4 + 4*N	`svid`	-	numeric	23	Satellite ID
5 + 4*N	`elv`	deg	numeric	38	Elevation (<= 90)
6 + 4*N	`az`	deg	numeric	230	Azimuth (range: 0-359)
7 + 4*N	`cno`	dB Hz	numeric	44	Signal strength (C/N0, range: 0-99), null when not tracking
Field No.	Name	Unit	Format	Example	Description
5.. 16	`signalId`	-	hexadecimal	0	NMEA-defined GNSS signal ID, see Signal Identifiers table (only available in NMEA 4.10 and later)
6.. 16	`cs`	-	hexadecimal	*7F	Checksum
7.. 16	`<CR><LF>`	-	character	-	Carriage return and line feed

Table 6 GPGSV message structure



```
Message Structure: $PUBX,00,time,lat,NS,long,EW,altRef,navStat,hAcc,vAcc,
                   SOG,COG,vVel,diffAg e,HDOP,VDOP,TDOP,numSvs,
                   reserved,DR,*cs<CR><LF>

          Example: $PUBX,00,081350.00,4717.113210,N,00833.915187,E,546.589,
                   G3,2.1,2.0,0.007,77.52,0.007,,0.92,1.19,0.77,9,0,0*5F
```

Field No.	Name	Unit	Format	Example	Description
0	`$PUBX`	-	string	$PUBX	Message ID, UBX protocol header, proprietary sentence
1	`msgId`	-	numeric	00	Proprietary message identifier: 00
2	`time`	-	hhmmss.ss	081350.00	UTC time. See the section UTC representation in the Integration manual for details.
3	`lat`	-	ddmm. mmmmm	4717.113210	Latitude (degrees and minutes), see format description
4	`NS`	-	character	N	North/South Indicator
5	`long`	-	dddmm. mmmmm	00833. 915187	Longitude (degrees and minutes), see format description
6	`EW`	-	character	E	East/West indicator
7	`altRef`	m	numeric	546.589	Altitude above user datum ellipsoid
8	`navStat`	-	string	G3	Navigation Status: NF = No Fix DR = Dead reckoning only solution G2 = Stand alone 2D solution G3 = Stand alone 3D solution D2 = Differential 2D solution D3 = Differential 3D solution RK = Combined GPS + dead reckoning solution TT = Time only solution
9	`hAcc`	m	numeric	2.1	Horizontal accuracy estimate
10	`vAcc`	m	numeric	2.0	Vertical accuracy estimate
11	`SOG`	km/h	numeric	0.007	Speed over ground
12	`COG`	deg	numeric	77.52	Course over ground
13	`vVel`	m/s	numeric	0.007	Vertical velocity (positive downwards)
14	`diffAge`	s	numeric	-	Age of differential corrections (blank when DGPS is not used)
15	`HDOP`	-	numeric	0.92	HDOP, Horizontal Dilution of Precision
16	`VDOP`	-	numeric	1.19	VDOP, Vertical Dilution of Precision
17	`TDOP`	-	numeric	0.77	TDOP, Time Dilution of Precision
18	`numSvs`	-	numeric	9	Number of satellites used in the navigation solution
19	`reserved`	-	numeric	0	Reserved, always set to 0
20	`DR`	-	numeric	0	DR used
21	`cs`	-	hexadecimal	*5B	Checksum
22	`<CR><LF>`	-	character	-	Carriage return and line feed

Table 7 PUBX00 message structure

For the reading of the GPS UART data by the ESP32, this code was used, in order to minimize the CPU load:

```
if (Serial2.available(){            //if a character arrives at the GPS connected UART (GPIO16)
grammagps = Serial2.read();    //store this character
if (grammagps != 10) //ift not new line feed  (10=new line feed, 13=carriage return)
   {
    if (indexxgps<127)      //checking sentence size for overflow protection
      {
leksi_sti_siriakigps[indexxgps] = grammagps;   //add the character in the sentence
      if (leksi_sti_siriakigps[0]>35)indexxgps = indexxgps+1; //if the first char is $ or printable
     else indexxgps = 0;  //if not printable char arrives, zerozise
}
}
else  //if line feed character arrived
   {
leksi_sti_siriakigps[indexxgps] = '\0'; //remove the last stored character carriage return
 strlcpy((char*)leksi_sti_siriakigpscopy,(char*)leksi_sti_siriakigps,indexxgps);//make a
                                   //copy of the sentence so to be processed by another routine
leksi_sti_siriakigps[0]= '\0';                          //empty the whole array
indexxgps = 0;                                         //empty the char counter
 hr8e_line_feed_flaggps = true;                       // set a flag, so to be available for
processing
}
}
if (hr8e_line_feed_flaggps==true)//if a new sentence arrived from UART2
{
hr8e_line_feed_flaggps=false;          //change the flag, so not to re process the same sentence
energies_apo_diabasma_gps();   //do appropriate actions for the received sentence
}
```

A similar routine is used for the communication through the USB cable of theESP32, in the case when a debugging will be need. At this case, they are just changed the code lines "*if (Serial.available()"and "Serial.read()*", so that the USB connection is used.

For compatibility with both NEO-8M and NEO-6M GPS module versions, it is programmed that reading of both "*$GPZDA*" and "*$GNZDA*" incoming sentences are valid. So to extract the useful data from a sentence such as "$GPZDA,235828.00,27,03,2021,00,00*67" this code is used:

```
const char koma[2] = ",";                              // store the coma character
If (strstr(leksi_sti_siriakigpscopy,"$GPZDA,")||(strstr(leksi_sti_siriakigpscopy,
"$GNZDA,")))  //if the sentence begins like this
{
messarxi = strtok(leksi_sti_siriakigpscopy, koma); //store  the first field($GPZDA)
UTCtimehhmmss = strtok(NULL, koma);    //store time
UTCDay = strtok(NULL, koma);     //store day
UTCMonth = strtok(NULL, koma);              //store month
UTCYear = strtok(NULL, koma); //store year
//ltzh= strtok(NULL, koma);// ignore it
//ltzm= strtok(NULL, koma);     //ignore it
//cs = strtok(NULL, koma);      //ignore it.
}
```

It must be noticed that the three last fields including the checksum are useless and thus are ignored.

4.7.3 The SPI Protocol (Serial Peripheral Interface)

The SPI (Serial Peripheral Interface) is mainly used for the communication with SD cards, LCD displays, temperature sensors, pressure sensors, ADC converters, Real-time clocks, and FPGAs. It uses full duplex master-slave architecture, with one master and one or many slaves. Signals such as SS (Slave Select), SCLK (Serial Clock), MISO (Master-In Slave-Out) and MOSI (Master-Out Slave-In) are used. The Master is providing the clock signal to all the Slaves. As it is shown on Figure 27 on a typical usage connection, by changing the SS signal, the desirable Slave device can be chosen, without the need of a unique dedicated address. The ESP32 Microcontroller integrates four SPI channels available, of which two are dedicated entirely to the flash cache so they can’t be used.

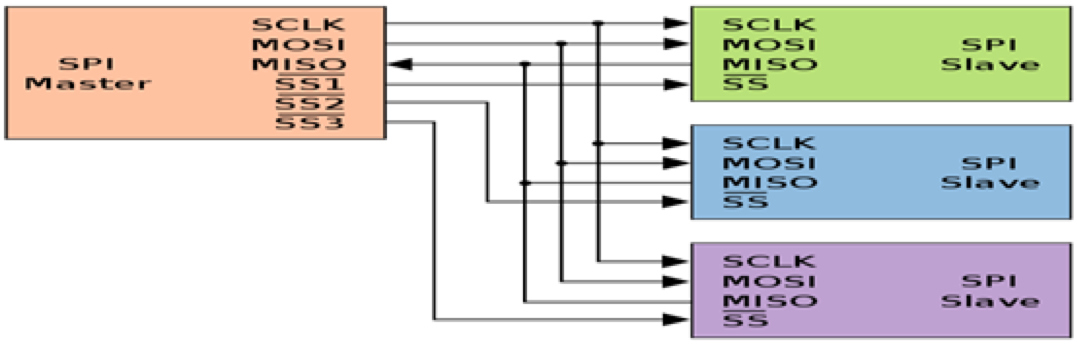


Figure 27 SPI communication bus

Source: https://en.wikipedia.org/wiki/Serial_Peripheral_Interface

On this Thesis, only one Slave Peripheral is used, the micro-SD card memory module. It is used connected to the ESP32 hardware VSPI, with the use of the pins 5=CS,18=CLK,19=MISO, and 23=MOSI. By using the appropriate libraries “SPI.h”, "FS.h" and "SD.h", the micro-SD card is behaving like a Hard disk drive. An example of code accessing it, is shown here:

```
SD.begin();          //begin communication to SD card through SPI bus
listDir(SD, "/", 0);    //show the SD Card directory contents
readFile(SD, "/hello.txt");  // print the file content
byte header[44];      // set the variable size
char file_hxoyname[] ="/sound49.wav";
```

```
File file_hxoy;        // name the new file
SD.remove(hxoyname);  //delete any previous file with the same name
file_hxoy = SD.open(hxoyname, FILE_WRITE); //open the new file with writing rights
file_hxoy.write(header, 44); //write 44 bytes of data in the file, taken by the header variable
file_hxoy.close(); //stop the file access
```

4.7.4 The i2c (Inter Integrated Circuit) Protocol

The i2c is an address-based communication protocol for short distances, where we can have one or more masters and slaves. Its Bus speed can reach the 5mbit/s and it only requires two cables SDA, SCL. It is usually used for communication with EEPROMS, ADCs, LCD Displays, Microphones, frequency synthesizers and more. With the use of 7bit addresses, i2c can communicate with up to 127 slave devices. On the Figure 28 is shown a usual Bus example.

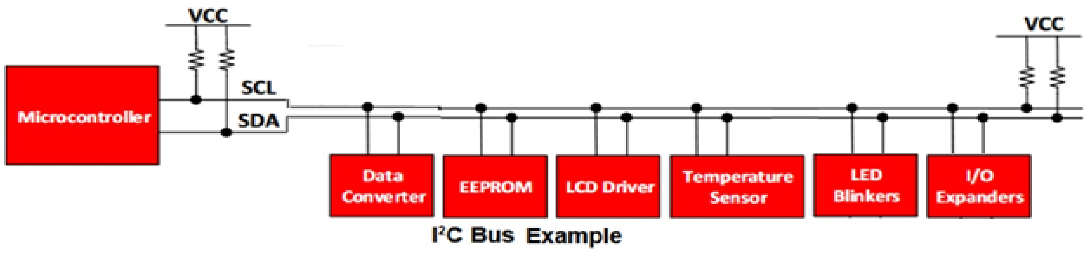


Figure 28 i2c communication bus example
Source: https://www.menloparktech.com/i2c-basics.html

On this Thesis, the one of the two ESP32 incorporated i2c buses is used to communicate with the Si4730 Receiver module. For the use of the i2c protocol, the Library "Wire.h" is included, but actually, the i2c bus manipulation is done through the "SI4735.h"library which is also included in the code. The i2c bus speed is configured slowly at just 10 KHz with the command "si4735.setI2CLowSpeedMode();". Other commands that are used to operate the Receiver Module are:

```
si4735.radioPowerUp();  // to enable the Module
si4735.powerDown();           //to disable it
si4735.volumeUp();           //to increase the volume
si4735.volumeDown();         //to decrease it
si4735.setVolume(37);        //to set a particular volume value
si4735.getCurrentVolume();   //to read the volume value
si4735.setAudioMute(true);    //to mute the audio (or un-mute with false)
si4735.setFrequency(10150);  //to go a particular frequency
previousFrequency = si4735.getFrequency(); //to read the set frequency
si4735.setFM(6400, 10800, 9500, 10); //to go an FM frequency, and set  range and step
si4735.setAM(520, 1710, 1700, 1); //to go to an AM frequency and set range and step
si4735.setSeekAmLimits(520, 1710);//to change AM seek range 1750
si4735.frequencyUp();  //increase frequency for a step
si4735.frequencyDown();//decrease frequency for a step
```

```
si4735getcurrentsnr=si4735.getCurrentSNR(); //read the SNR value
si4735getcurrentrssi=si4735.getCurrentRSSI();  //read the RSSI value
iscurrentTunefm=si4735.isCurrentTuneFM();  //find what is the modulation type
```

4.7.5 The MQTT (Message Queuing Telemetry Transport) Protocol

The MQTT is a modern TCP/IP (even UDP) protocol for fast communication of IoT (Internet of Things) devices. There are similar IoT protocols such as the STOMP (Simple or Streaming Text Orientated Messaging Protocol), CoAP (Constrained Application Protocol), AMQP (Advanced Message Queuing Protocol), XMPP (Extensible Messaging - Presence Protocol).

The MQTT was preferred as it is a lightweight protocol, it is compatible with many applications, it uses little overhead, has little packet loss, has low bandwidth consumption, and additionally, and if need, it can offers sufficient security level through SSL/TLS or AES-GCM encryption, either password or X.509 Authentication [44] or even can be upgraded to SMQTT (Secure Message Queue Telemetry Transport.

The communication is done with the usage of a dedicated Server named “MQTT Broker” where the Clients are connected, and with the use of organized “Topics”, they exchange messages of data between them. All clients are subscribed to the topic they choose. When a client writes a change to topic (publish), this change is transferred to all the others clients subscribed on the same topic. On Figure 29 is shown the MQTT architecture.

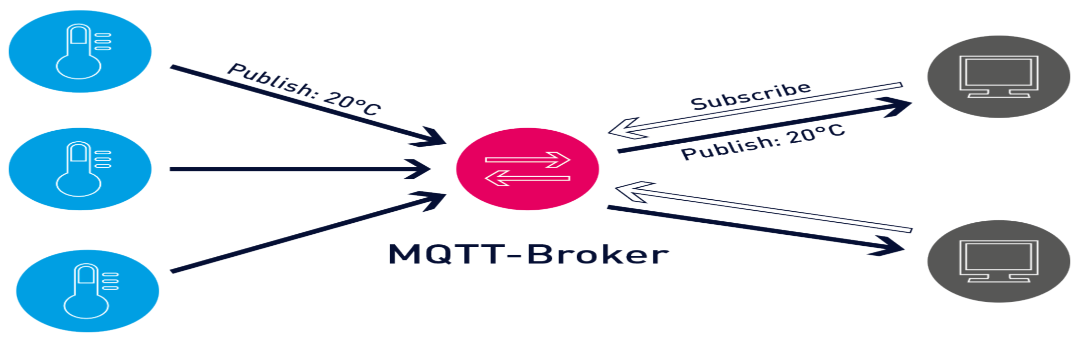


Figure 29 MQTT architecture

Source: https://www.switchdoc.com/wp-content/uploads/2020/01/mqtt-architecture.png

On this Thesis, with the usage of "EspMQTTClient.h" library, the free MQTT Server “broker.hivemq.com” on port 1883 was used. Both the Receiving Stations and the Node-Red Server, were subscribed to common MQTT Topics. Thus, it was possible from the Node-Red Page, either to force concurrency all the Receiving Stations to do an operation, or to individually manipulate a single Receiving Station to do an operation, based on its MAC address.

The Topics that were used are shown on Table 8:

Recipient's Topic	Action
esp32DimThes/ola	All the Stations are listening, waiting to receive commands
esp32DimThes/FCF5C4551F98	Only this specific Station is listening to receive commands
esp32DimThes/2462ABF28A4C	>>
esp32DimThes/2462ABF285C4	>>
esp32DimThes/2462ABF1EEF4	>>
esp32DimThes/3C71BF9E1BD8	>>
esp32DimThes/3C71BF9E1BD8/state	Each Station, sends regularly an online status informing packet
esp32DimThes/toserver	All the Stations send here their respond/answer

Table 8 MQTT Topics created

The available commands that were created to manipulate the Receiving Stations are shown on Tables 9, 10:

MQTT commands	Actions
Z	Reset the Station's Microcontroller
R	Reset the receiver module si4730
arxigps	Make an initialization to the GPS module (to set which NMEA to send, UART speed,etc.
Hotrestart	Reset the GPS module
warmrestart	>>
coldrestart	>>
paraliptis(x@x.x)	Change the recipient's email address for receiving the Email
alive	Force Stations to send a respond of which of them are connected to internet
+ or -	ChangeVolumeUp/down
S or s	ScanfrequenciesUp/down
u or d	Change Frequency Up/down a step,
volt	Learn the power supply/battery voltages (for battery, a relay disconnects the battery for 5Secs),
powersupply1/0	Manually connect/disconnect the power supply (through a relay)
ch1- ch15	Set the radio receiver to a preserved frequency of the 15 stored memories
setVol(xx)	Change the sound volume (1-63)
setFreq()	Change the frequency (6400-10800 for FM or 520-1710 for AM)
recordtime(xx)	Change the recording sound file time duration

Table 9 MQTT commands created

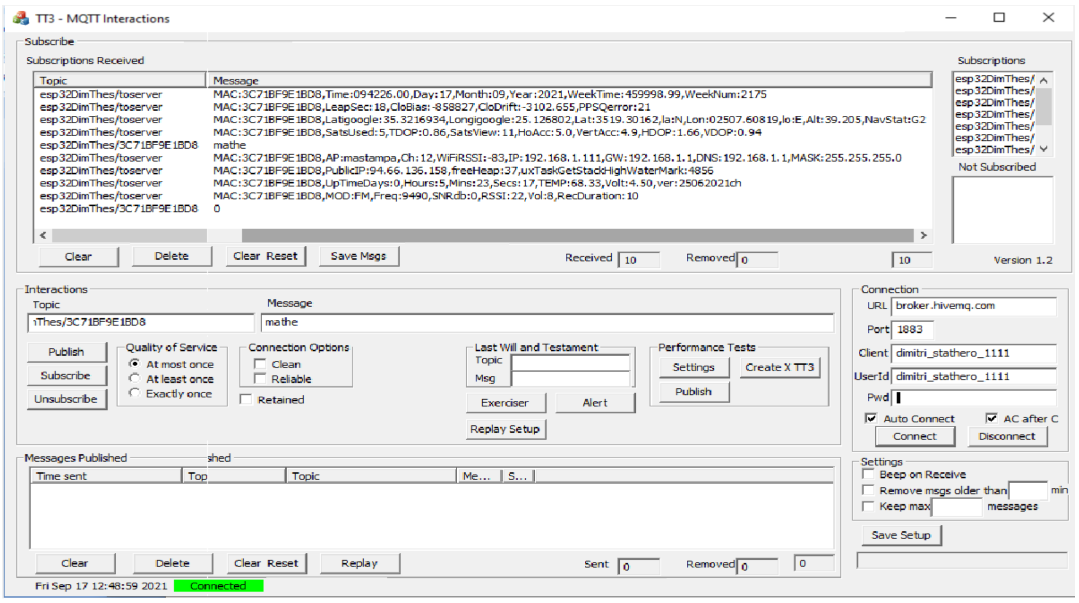
MQTT commands	Actions
x	Start immediately recording a sound file.
@@	Start recording a sound file, just when the next PPS Pulse arrives
sendd	Send by email the existing saved on SD-card sound file
@@sendd	Start recording a sound file, just on the next PPS Pulse, and email it afterwards
0	Get radio/Wi-Fi info
mathe	Get GPS Info
help	Get a memo list with the above available commands

Table 10 MQTT commands created

A remarkable feature provided by the MQTT Service, is the LWT (Last Will and Testament) feature [19]. Thus, a predefined packet is stored by the subscriber in the MQTT Broker, but it is only sent by the MQTT Broker to the Topic, only when the keep-alive interval value expires, and the client seems to be disconnected. So, if suddenly a Receiving Station loses its connection to the MQTT Broker, the Broker itself, takes over to transmit this predefined message. So as the Node-Red is subscribed on this Topic, it can be informed that this client has disconnected.

In this Thesis, the "offline" LWT messages are transmitted for each subscriber. For example, at the "esp32DimThes/3C71BF9E1BD8/state" Topic, a "offline" message will be transmitted by the subscribed client having thew MAC "3C71BF9E1BD8", after 6 seconds of no response. Afterwards, the MQTT Broker releases the expired connection. The keep-alive interval parameter can take values from a few seconds, until almost 18 Hours.

A simple free tool that was used in order to test sending and receiving MQTT commands was TT3 as is shown on figure 30.



Topic	Message
esp32DimThes/toserver	MAC:3C71BF9E1BD8,Time:094226.00,Day:17,Month:09,Year:2021,WeekTime:459998.99,WeekNum:2175
esp32DimThes/toserver	MAC:3C71BF9E1BD8,LeapSec:18,CloBias:-858827,CloDrift:-3102.655,PPSQerror:21
esp32DimThes/toserver	MAC:3C71BF9E1BD8,Latigoogle:35.3216934,Longigoogle:25.126802,Lat:3519.30162,la:N,Lon:02507.60819,lo:E,Alt:39.205,NavStat:G2
esp32DimThes/toserver	MAC:3C71BF9E1BD8,SatsUsed:5,TDOP:0.86,SatsView:11,HoAcc:5.0,VertAcc:4.9,HDOP:1.66,VDOP:0.94
esp32DimThes/3C71BF9E1BD8	mathe
esp32DimThes/toserver	MAC:3C71BF9E1BD8,AP:mastampa,Ch:12,WiFiRSSI:-83,IP:192.168.1.111,GW:192.168.1.1,DNS:192.168.1.1,MASK:255.255.255.0
esp32DimThes/toserver	MAC:3C71BF9E1BD8,PublicIP:94.66.136.158,freeHeap:37,uxTaskGetStackHighWaterMark:4856
esp32DimThes/toserver	MAC:3C71BF9E1BD8,UpTimeDays:0,Hours:5,Mins:23,Secs:17,TEMP:68.33,Volt:4.50,ver:25062021ch
esp32DimThes/toserver	MAC:3C71BF9E1BD8,MOD:FM,Freq:9490,SNRdb:0,RSSI:22,Vol:8,RecDuration:10
esp32DimThes/3C71BF9E1BD8	0

Subscriptions: esp32DimThes/ esp32DimThes/ esp32DimThes/ esp32DimThes/ esp32DimThes/ esp32DimThes/ esp32DimThes/

Received 10 Removed 0 10

Topic: ıThes/3C71BF9E1BD8
Message: mathe

Quality of Service: [✓ At most once] At least once Exactly once
Connection Options: ☐ Clean ☐ Reliable ☐ Retained

URL: broker.hivemq.com
Port: 1883
Client: dimitri_stathero_1111
UserId: dimitri_stathero_1111
Pwd:
☑ Auto Connect ☑ AC after C

Time sent	Topic	Me...	S...

Sent 0 Removed 0 0

Fri Sep 17 12:48:59 2021 Connected

Figure 30 TT3 application receiving MQTT

Explaining the two basic client responses “0” and “mathe”, when the MQTT command “0” is sent to a specific Receiving Station, then it is responding with 4 sentences, informing us about: The ESP32 MAC, the Modulation, the Frequency, the SNR, the RSSI, the Volume, the Recording duration, Uptime Duration, Temperature, Voltage, Firmware version, Free Heap memory, StackHighWatermark memory, Access Point Connected Name, Access Point Channel, Access Point RSSI, Node’s Public IP, ESP32 LAN IP, LAN Gateway, DNS, and finally, LAN Subnet Mask.

Similarly, when the MQTT command “mathe” is send to a Station, it replies to us with 4 responding sentences, informing us about: The Satellites used, the TDOP, Satellites viewed, Horizontal & Vertical accuracy, HDOP, VDOP, Latitude & Longitude, Altitude, Navigation Status, Leap Seconds, Clock Bias, Clock Drift, PPS Quality Error, Current Day-Month-Year, Week Time, and Week Number. These can be seen on Figure 30 above.

4.7.6 The ADC (Analog to Digital Conversion) Library

The ESP32 integrates two 12-bit ADC distributed on 15 channels. Theoretically all of the 15 channels, can measure concurrently but actually, due to the resources limitations, this is not possible. The 12 bits value, means that each channel input, can measure voltages from 0 volt to 3.3 volts, divided into 4096 steps (2^12=4096). Unfortunately, the behavior of the ESP32 ADC is not too linear as it shown in the Figure 31 below.

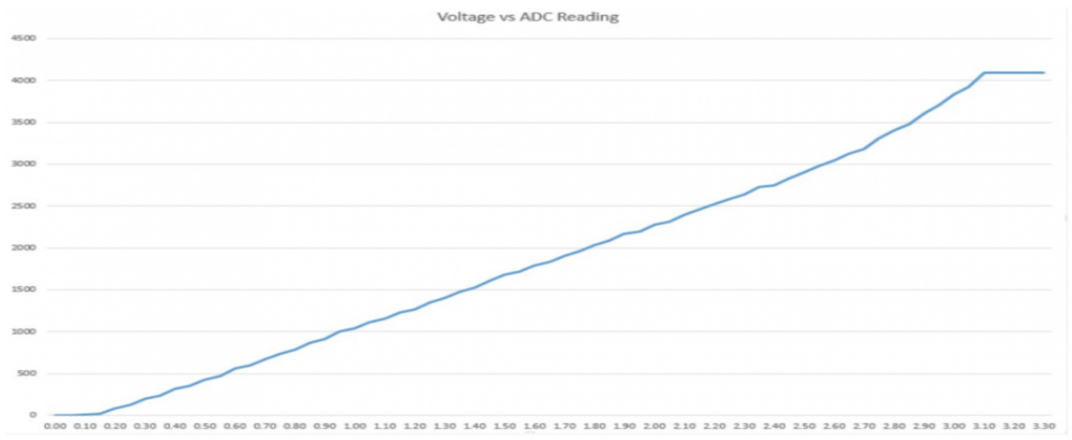


Figure 31 ESP32 ADC Nonlinear behavior

source: https://randomnerdtutorials.com/esp32-adc-analog-read-arduino-ide/

Each used channel has to be configured prior its usage, by choosing one of the four allowed software attenuation modes, depending on the input signal amplitude. The available options are ADC_0db, ADC_2_5db, ADC_6db, and ADC_11db.

In this Thesis, the option “ADC_ATTEN_DB_11” with the bigger attenuation was used, as it was better corresponding for the signal amplitude, and also as it is shown on Figure 32 below, it had the better, inclination compared to the other attenuations.

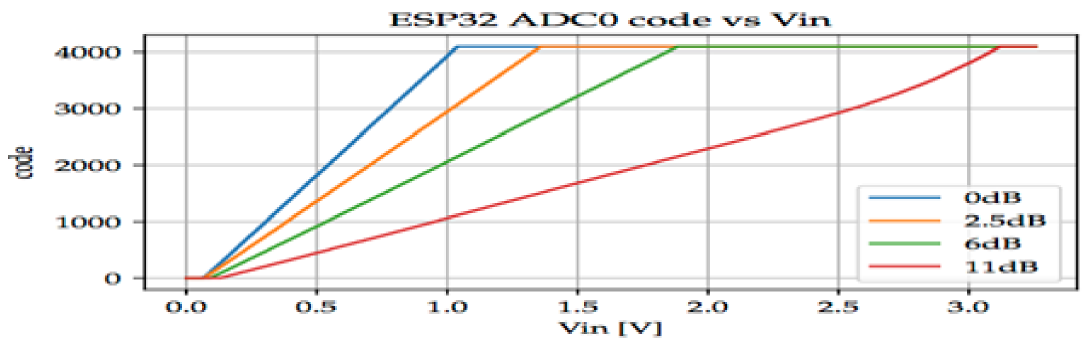


Figure 32 ADC output code versus input voltage
source: https://people.eecs.berkeley.edu/~boser/courses/49_sp_2019/N_gpio.html

Only one (ADC1_CHANNEL_7GPIO35) of the channels is used in order to capture the desirable sound to an audio file, with the usage of "I2S.h" and "Wav.h" libraries.

The audio spectrum is ranged from 20Hz το 15KHz at most for the usual FM modulations, So the audio format used was the uncompressed WAV, with sampling rate 44100Hz, and 16 bits per sample. This audio file, in order to be compatible and recognizable, has to state these previous characteristics. So it uses an RIFF (Resource Interchange File Format) tag in its structure. This tag is incorporated at the beginning of the recorded sound file as a Header. So after defining in the code “#define I2S_MODE I2S_MODE_ADC_BUILT_IN”, the audio recording is beginning with the command “I2S_Init(I2S_MODE, I2S_BITS_PER_SAMPLE_32BIT);” The audio sampling is then done with this code[14]:

```
intrecord_time = 10;  // seconds
constintheaderSize = 44;
constintwaveDataSize = record_time * 88000;
constintnumCommunicationData = 8000;
constintnumPartWavData = numCommunicationData/4;
byte header[headerSize];
char communicationData[numCommunicationData];
char partWavData[numPartWavData];

for (int j = 0; j <waveDataSize/numPartWavData; ++j)
  {
   I2S_Read(communicationData, numCommunicationData);
   for (inti = 0; i<numCommunicationData/8; ++i)
      {
partWavData[2*i] = communicationData[8*i + 1];
partWavData[2*i + 1] = communicationData[8*i +2];
      }
file_hxoy.write((const byte*)partWavData, numPartWavData);
  }
file_hxoy.close(); //close the recorded file
```

At the end, the audio file is closed, in order to be available for further handling.

4.7.7 The Wi-Fi Libraries

The “WiFi.h” and “WiFiMulti.h” libraries are used in order to activate the wireless communication. For the sake of easiness, the same firmware had to be commonly flashed in all the ESP32 of the Receiving Stations. So, in this common firmware, it had to be included credentials for all the possible Access Points. So all the SSIDs and Passwords of all the Access Points were included in the unique firmware, thus, while the booting sequence, each Receiving Station, is connecting to its own, differently named Access Point. An example of the code is:

```
wifiMulti.addAP("accessPoint1", "password1");
wifiMulti.addAP("accessPoint2", "password2");
wifiMulti.addAP("accessPoint3", "password3");
```

4.7.8 The SMTP Protocol

The SMTP (Simple Mail Transfer Protocol) is a very old protocol used for sending Emails. A dedicated port (usually 25 or 587) on a server is used. The clients start a TCP usually SSL/TLS encrypted handshake with the server in order to send the email message.

On this Thesis, the library “ESP_Mail_Client.h” was used by the Receiving Stations, providing, TLS encryption on port 587 of GMAIL. Also, the email “CHUNKING” capability was used, a method to easy send a big file by email, (such as an audio file), by splitting it and send it into parts by using the little available RAM of the ESP32 microcontroller. The recipient address is pre-defined but also configurable.

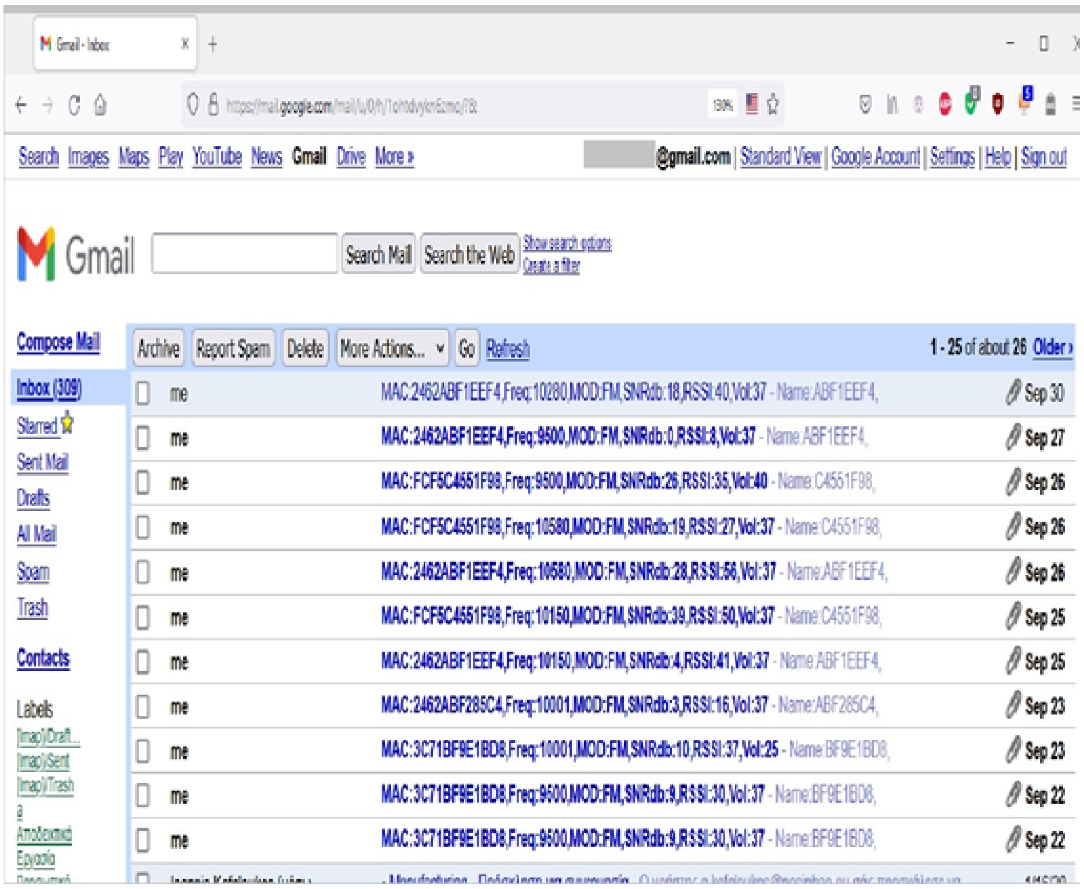
Prior the Email sending, a “thema” variable is created, containing the Station ID, the chosen Frequency, the Modulation, the SNR, and the Volume. These values are included in the Email’s Subject. So, this variable is set:

```
Stringthema="MAC:"+macstring+",Freq:"+currentFrequencystring+",MOD:"+diam+",SN
Rdb:"+si4735getcurrentsnr+",RSSI:"+si4735getcurrentrssi+",Vol:"+si4735getcurrentvolum
e;
```

Similarly, a “content” variable is created, containing not only the audio file attachment, but also other usable data such as: Latitude, longitude, NorthSouth, EastWest, NavigationStatus, TimePulseQuantizationError, SatellitesView,Satellitesused, VerticalAccuracy, Horizontal Accuracy, HDOP, VDOP, TDOP, CpuTemperature, PowerSupplyVolt, BatteryVolt, RecordingDuration, FirmwareVersion, PublicIP, UTCtimehhmmss, UTCDay, UTCMonth, and UTCYear.

These data are recently collected and incorporate by the ESP32 in the Email. During the Email sending, no other actions are done in order to reduce CPU and RAM usage. After the Email is send, the session is released, and the Station is ready for the next measurement. An example of the sent Email’s Subject is shown on Figure 33.

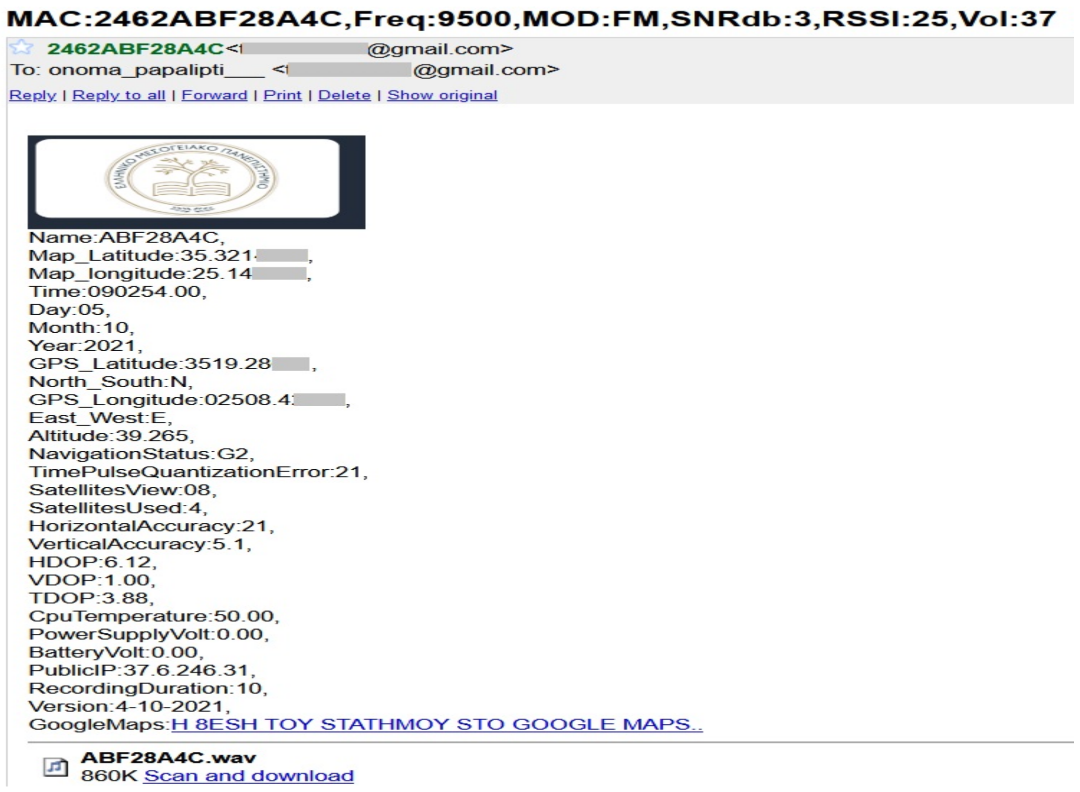
.



me	MAC:2462ABF1EEF4,Freq:10280,MOD:FM,SNRdb:18,RSSI:40,Vol:37 - Name:ABF1EEF4,	Sep 30
me	MAC:2462ABF1EEF4,Freq:9500,MOD:FM,SNRdb:0,RSSI:8,Vol:37 - Name:ABF1EEF4,	Sep 27
me	MAC:FCF5C4551F98,Freq:9500,MOD:FM,SNRdb:26,RSSI:35,Vol:40 - Name:C4551F98,	Sep 26
me	MAC:FCF5C4551F98,Freq:10580,MOD:FM,SNRdb:19,RSSI:27,Vol:37 - Name:C4551F98,	Sep 26
me	MAC:2462ABF1EEF4,Freq:10580,MOD:FM,SNRdb:28,RSSI:56,Vol:37 - Name:ABF1EEF4,	Sep 26
me	MAC:FCF5C4551F98,Freq:10150,MOD:FM,SNRdb:39,RSSI:50,Vol:37 - Name:C4551F98,	Sep 25
me	MAC:2462ABF1EEF4,Freq:10150,MOD:FM,SNRdb:4,RSSI:41,Vol:37 - Name:ABF1EEF4,	Sep 25
me	MAC:2462ABF285C4,Freq:10001,MOD:FM,SNRdb:3,RSSI:16,Vol:37 - Name:ABF285C4,	Sep 23
me	MAC:3C71BF9E1BD8,Freq:10001,MOD:FM,SNRdb:10,RSSI:37,Vol:25 - Name:BF9E1BD8,	Sep 23
me	MAC:3C71BF9E1BD8,Freq:9500,MOD:FM,SNRdb:9,RSSI:30,Vol:37 - Name:BF9E1BD8,	Sep 22
me	MAC:3C71BF9E1BD8,Freq:9500,MOD:FM,SNRdb:9,RSSI:30,Vol:37 - Name:BF9E1BD8,	Sep 22

Figure 33 Subject of sent Emails

Similarly, the Email body contains a HMU logo at top, followed by the information, and at the bottom there is a Google Map link targeting the location of this Receiving Station, and also the attached recorded file. These are shown on Figure 34.



MAC:2462ABF28A4C,Freq:9500,MOD:FM,SNRdb:3,RSSI:25,Vol:37

2462ABF28A4C <[redacted]@gmail.com>
To: onoma_papalipti___ <[redacted]@gmail.com>

Reply | Reply to all | Forward | Print | Delete | Show original

```
Name:ABF28A4C,
Map_Latitude:35.321[redacted],
Map_longitude:25.14[redacted],
Time:090254.00,
Day:05,
Month:10,
Year:2021,
GPS_Latitude:3519.28[redacted],
North_South:N,
GPS_Longitude:02508.4[redacted],
East_West:E,
Altitude:39.265,
NavigationStatus:G2,
TimePulseQuantizationError:21,
SatellitesView:08,
SatellitesUsed:4,
HorizontalAccuracy:21,
VerticalAccuracy:5.1,
HDOP:6.12,
VDOP:1.00,
TDOP:3.88,
CpuTemperature:50.00,
PowerSupplyVolt:0.00,
BatteryVolt:0.00,
PublicIP:37.6.246.31,
RecordingDuration:10,
Version:4-10-2021,
GoogleMaps:H 8ESH TOY STATHMOY STO GOOGLE MAPS..
```

ABF28A4C.wav
860K Scan and download

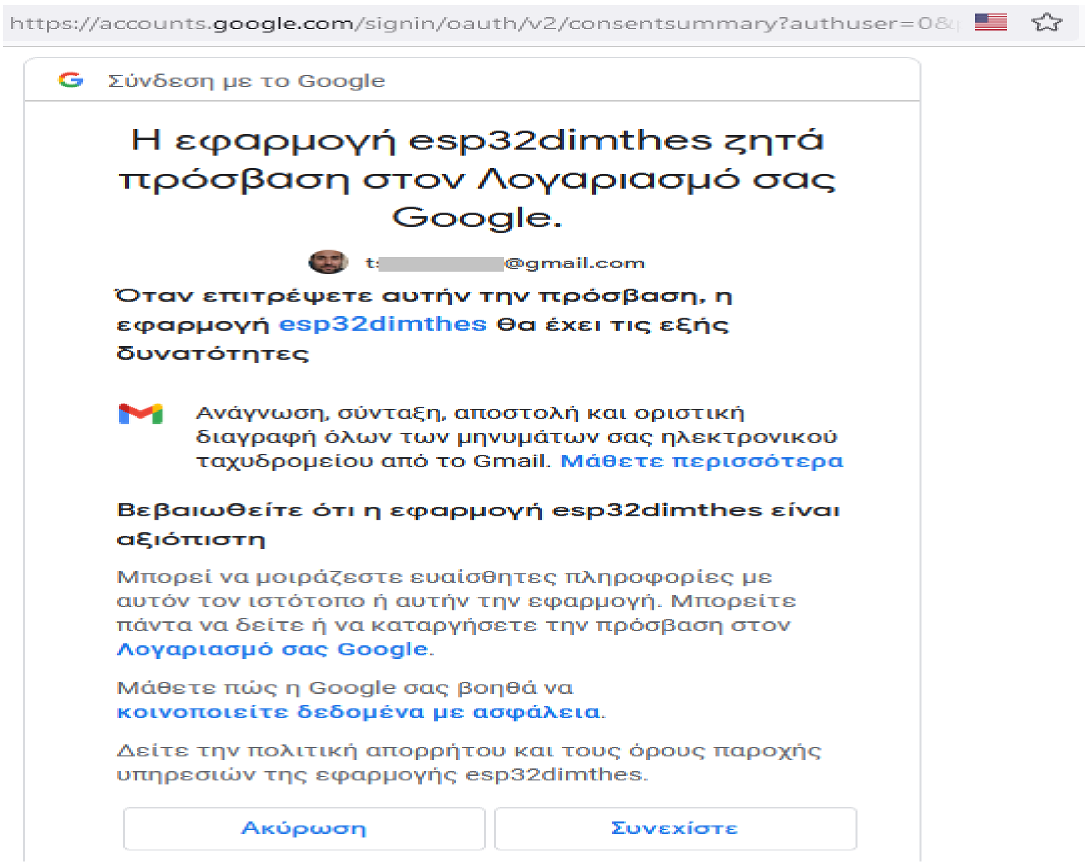
Figure 34 Email Body syntax

4.7.9 The POP3 Protocol and OAuth

The POP3 (Post Office Protocol) is an old TCP protocol used to receive emails through the dedicated usually 110 port. The client is connecting to the server, downloads the Emails, or either removes them from the Server.

On this Thesis, Node-Red environment was used to automatically connect to Gmail, to receive and store the emails, containing the measured sound files and the valuable data.

To increase security level, the Gmail OAuth (Open Authorization) protocol was used, where only with the use of a token the Node-Red application obtains email access. The enabling of this free provided API capability was done through the link: “https://console.cloud.google.com/apis/library?pli=1”as shown on Figure 35.



https://accounts.google.com/signin/oauth/v2/consentsummary?authuser=0&

Σύνδεση με το Google

Η εφαρμογή esp32dimthes ζητά πρόσβαση στον Λογαριασμό σας Google.

t[redacted]@gmail.com

Όταν επιτρέψετε αυτήν την πρόσβαση, η εφαρμογή esp32dimthes θα έχει τις εξής δυνατότητες

Ανάγνωση, σύνταξη, αποστολή και οριστική διαγραφή όλων των μηνυμάτων σας ηλεκτρονικού ταχυδρομείου από το Gmail. Μάθετε περισσότερα

Βεβαιωθείτε ότι η εφαρμογή esp32dimthes είναι αξιόπιστη

Μπορεί να μοιράζεστε ευαίσθητες πληροφορίες με αυτόν τον ιστότοπο ή αυτήν την εφαρμογή. Μπορείτε πάντα να δείτε ή να καταργήσετε την πρόσβαση στον Λογαριασμό σας Google.

Μάθετε πώς η Google σας βοηθά να κοινοποιείτε δεδομένα με ασφάλεια.

Δείτε την πολιτική απορρήτου και τους όρους παροχής υπηρεσιών της εφαρμογής esp32dimthes.

Ακύρωση | Συνεχίστε

Figure 35 Enabling access to Node-red application

The client secret file “credentials.json” had to be download, and put into the folder e.g. C:\Users\user\.node-red, and used as an authentication token.

Later through the “Authentication Tab” on Node-Red page, the secret password could be set if has expired.

4.7.10 Other Libraries used

Other libraries usage was done such as:

“freertos/queue.h" was used in order to enable dual core capability.

“HTTPClient.h” and “ESP32Ping.h”was used in order each Receiving Station, to discover, it’s Public IP.

“__cplusplusextern C" was used in order to read the ESP32 temperature.

"uptime.h" was used in order to find the ESP32 uptime.

Chapter 5: Major Principles of Operation

5.1 Receiving Stations Operation

Each Receiving Station, on its first boot up time, makes some procedures, such as: Enabling its microcontroller dual core CPU capability, enabling its ADC capability, initializing its Wi-Fi connection, learning its Public IP, initializing its GPS module, initializing its Si4730 Receiver module, initializing its SD Card Module, initializing the MQTT connections, and sends a MQTT message informing that it is now active. All these are done by CPU 0.

The CPU 1 is dedicated on two operations: Firstly, to watch the state of the first CPU core, and in case of halting, to force a CPU reboot. Secondly, when the appropriate MQTT start metering command arrives, it forces sound recording. If PPS recording is chosen, it enables an IRQ (Interrupt Request), so that when the upcoming PPS signal arrives, it will achieve a very fast response on forcing the start of metering procedure.

It must be noticed the criticalness of the point where the PPS detection is activating through the command "*attachInterrupt(digitalPinToInterrupt(34),isr, RISING);*". The "isr" function complexity has to be kept as simple as possible, in order to minimize the interrupt handler annoying time. Thus, just a flags witching is placed there. Also, the "isr" function is defined as a "*IRAM_ATTR*" type, causing compiler and linker, to mark this function's code, to be statically placed in IRAM and never swapped out resulting an even faster response [20]. Thus the "isr" function code is as simple as:

```
void IRAM_ATTR isr()
{
irthePPS = 1;    //PPS Pulse arrived
}
```

The "*irthePPS*" variable, was primarily declared as a volatile, forcing the compiler to load it from RAM instead of using a storage register, so its state changing is recognized faster in loop() [21]. Thus, the initializing code "*volatile boolirthePPS=0;*"is used.

Each Receiving Station is watching regularly on its subscribed Topics, for incoming MQTT commands to arrive. By receiving the appropriate MQTT command, each Station can either operate its Si4730 Receiving module to scan, change a frequency, change its AM or FM Modulation, change its Volume, start a sound recording, operate the SD card Module, operate the GPS Module, send an email with sound attachment, make a CPU Reset, Si4730 Reset, GPS module Reset, or perform other operations. A simplified workflow diagram of the Receiving Stations operation is shown on Figure 36.

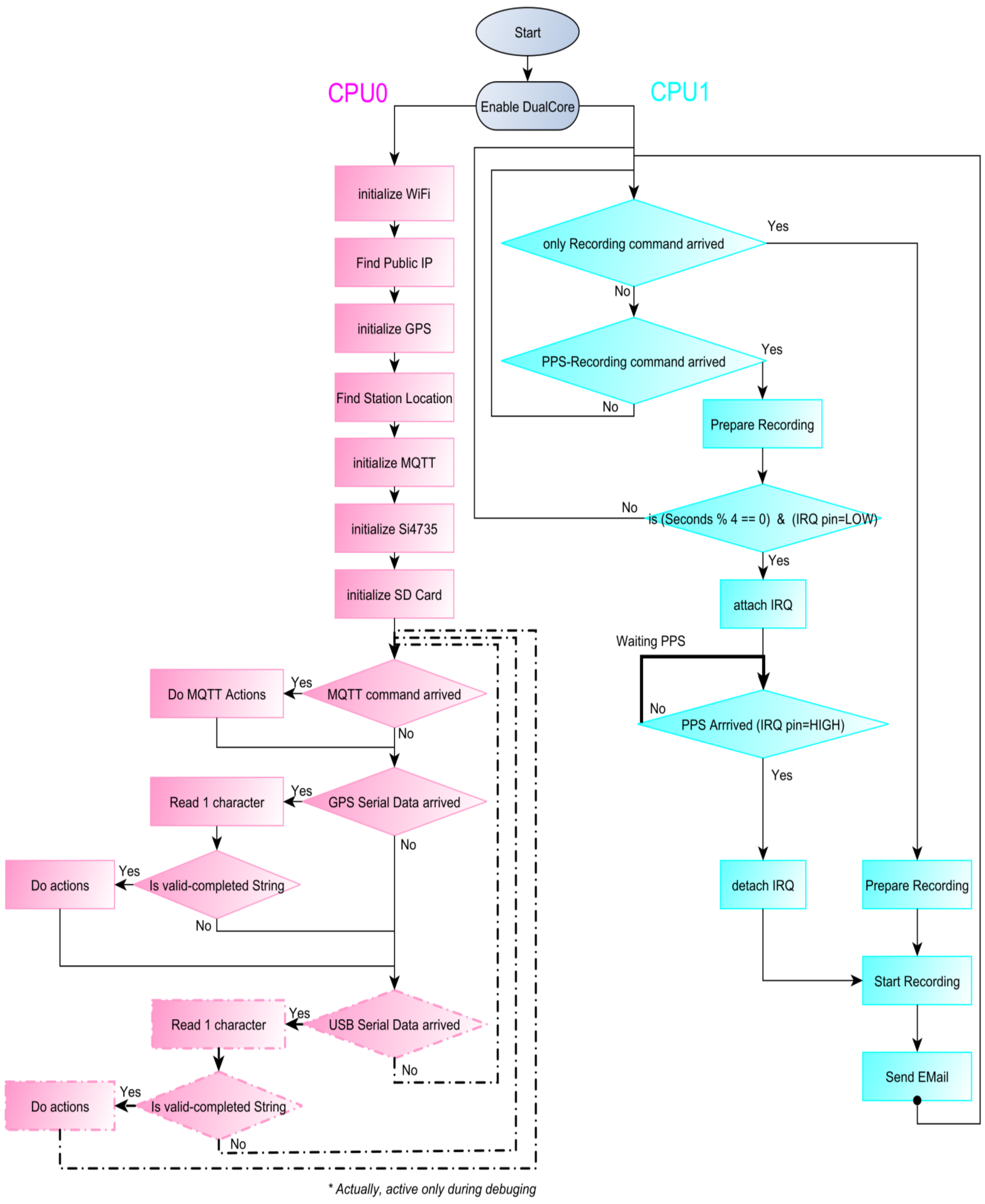


Figure 36 Simplified WorkFlow Diagram of Receiving Stations

More specifically, when the MQTT "start PPS-Recording" command arrives, the current time (already taken by the GPS clock) is checked, and the microcontroller waits until the PPS is LOW. Then, if the "seconds" field of current time's value is divided by '4', (if (

UTCtimeSeconds % 4) == 0), which means that only when the "Seconds" value is 0, 4, 8, 12, 16, 20, 24, 28, 32, 36, 40, 44, 48, 52, 56sec), then, the "attachInterrupt" is enabled, and the CPU 1 goes on standby to receive the PPS HIGH Signal within the following 3 seconds. In other words, this is translated to 15 available slots in a minute to start a measuring.

This 3 seconds of "dead time" is interfered in order to allow the MQTT commands, to reach all the distant Receiving stations, and provide plenty time for them to be prepared for starting recording (initializing SD-Card, Wav file, ADC).

After the PPS arrives, "detachInterrupt" happens (to stop watching the PPS State any more), and immediately starts an Audio recording. So, this way we can achieve that the PPS signal will force a concurrent metering beginning to all the stations, and each Station's recorded audio file will have the same beginning time moment. The recording duration is configured at 10 Seconds, but it is configurable.

5.2 User Interface Tab Operation

The user is operating the whole Direction-Finding System promoting the measurement procedure, through a web page, served by the Node-Red platform. The Measuring procedure is shown on figure 37.

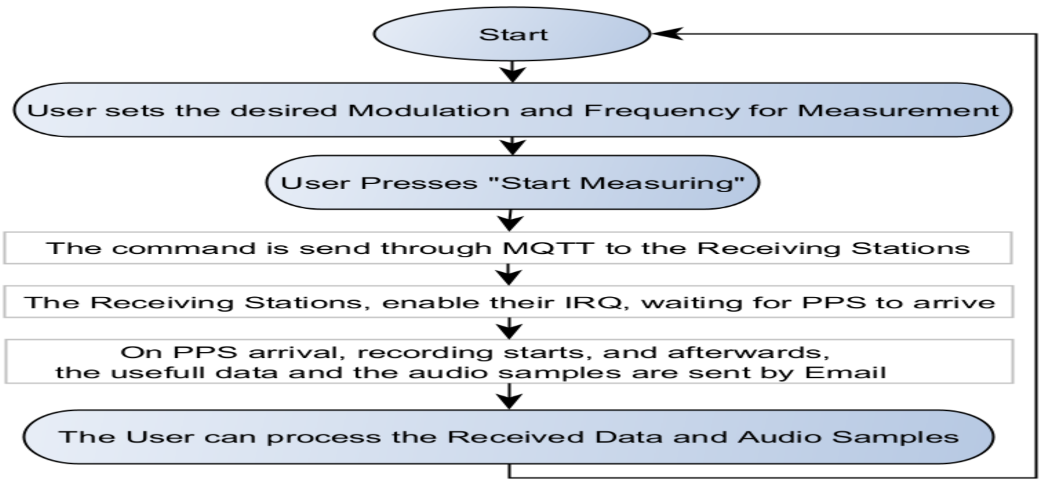

Start
User sets the desired Modulation and Frequency for Measurement
User Presses "Start Measuring"
The command is send through MQTT to the Receiving Stations
The Receiving Stations, enable their IRQ, waiting for PPS to arrive
On PPS arrival, recording starts, and afterwards, the usefull data and the audio samples are sent by Email
The User can process the Received Data and Audio Samples

Figure 37 Measuring procedure

5.2.1 The Take Measurements Tab

To operate the Direction-Finding System, the user is just accessing a Web browser served by the Raspberry Pi Node-Red Server. The User opens the "Take measurements tab" Web page as it is shown on Figure 38.

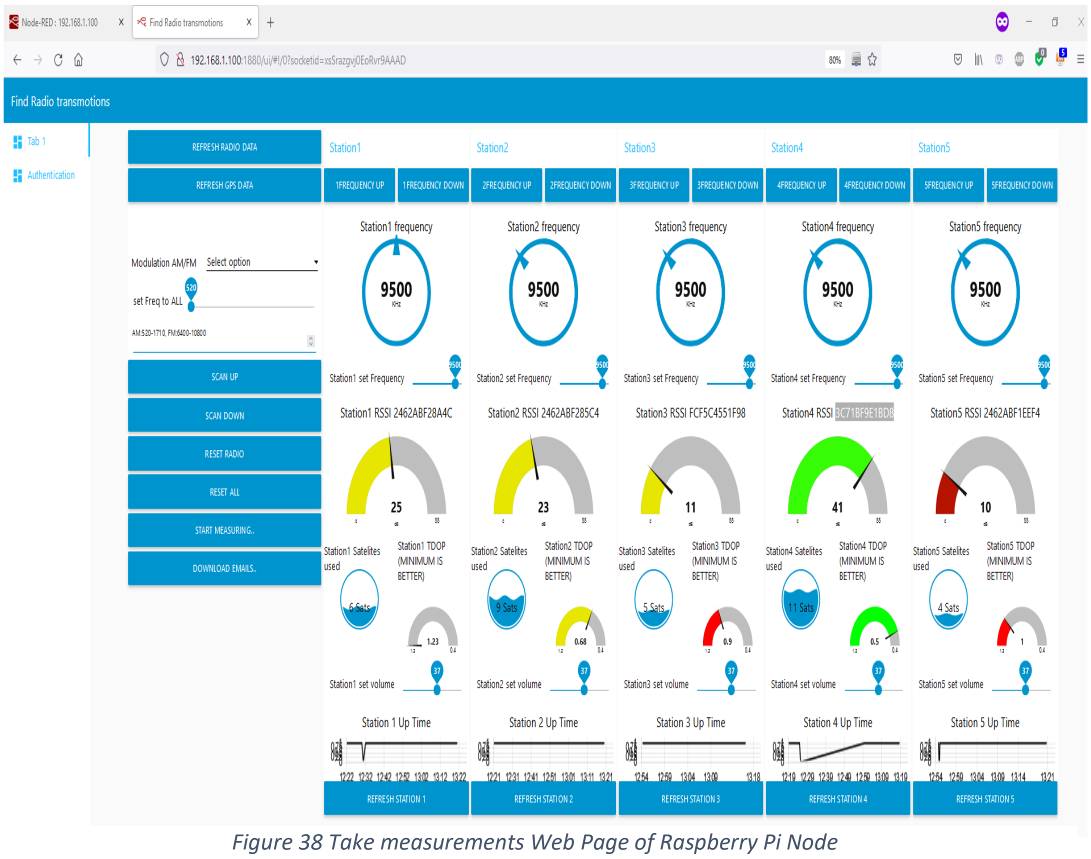


Figure 38 Take measurements Web Page of Raspberry Pi Node

There, it is represented, for each of the Stations, Basic Data such as Frequency, Signal RSSI, Satellites Used, TDOP, Station Up Time, and Volume. These data have been obtained through the MQTT "esp32DimThes/toserver" Topic.

So, through this Page, the user can either: Learn information or manipulate individual each Receiving Station separately. (By using at the background each Receiving Station's unique Topic such as "esp32DimThes/ 2462ABF28A4C"). As is shown on Figure 39.

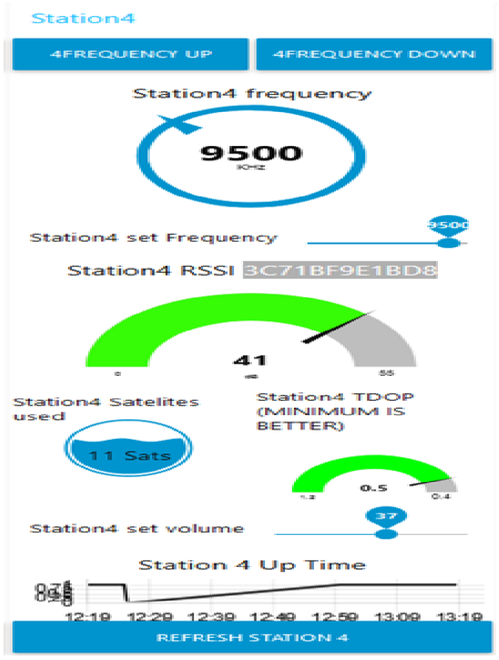


Figure 39 Separate information about each Station

Either the User can command concurrently all the available Receiving Stations by using the left aligned buttons "Refresh Radio Data", "Refresh GPS Data", "Select AM/FM Modulation", "Set Freq to All", "Scan Up/Down", "Reset Radio", "Start Measuring", "Download Emails". (Where at the background, is used the "esp32DimThes/ ola" Topic, where all the Stations have subscribed). This is shown on Figure 40.

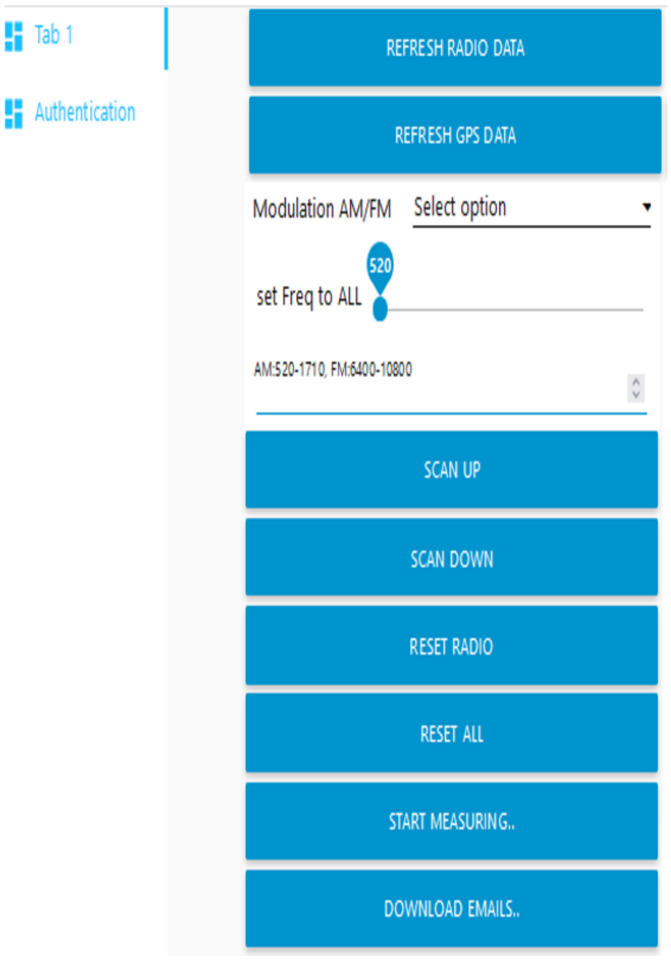


Figure 40 Concurrently manipulation of all Receiving Stations

Thus, each user's action, is interpreted to a MQTT command, which is send to the MQTT Broker, and then forwarded to all the Receiving Stations. Each Receiving Station, either responds to this command, or bypasses it, depending on his MQTT Topic subscription.

To start a Measurement, the user firstly chooses the desired frequency and modulation, and presses the "Start Measuring" button. After about 1-4 seconds, all the Receiving Stations, start concurrently, (absolutely synchronized by their PPS Signal), to record their audio file of the selected Radio Frequency.

Afterwards, when the sound recording has finished, each Receiving Station adds this audio file as an attachment in an Email and send it together with other useful data, to a common email address.

The Flow used for creation of the “Take Measurements” Dashboard Tab, is shown on Figure 41.

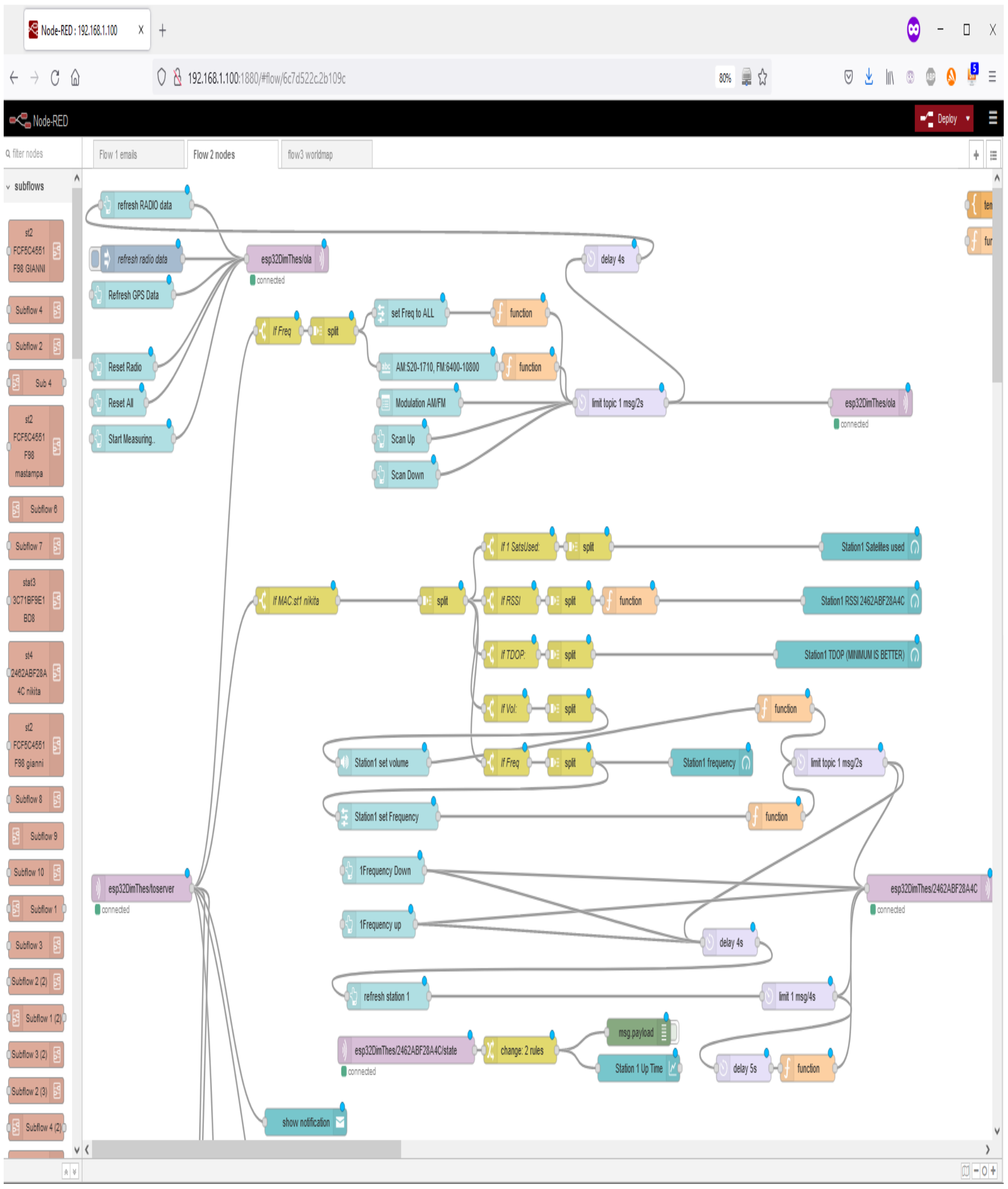


Figure 41 Flow used for the “Take Measurements” Tab

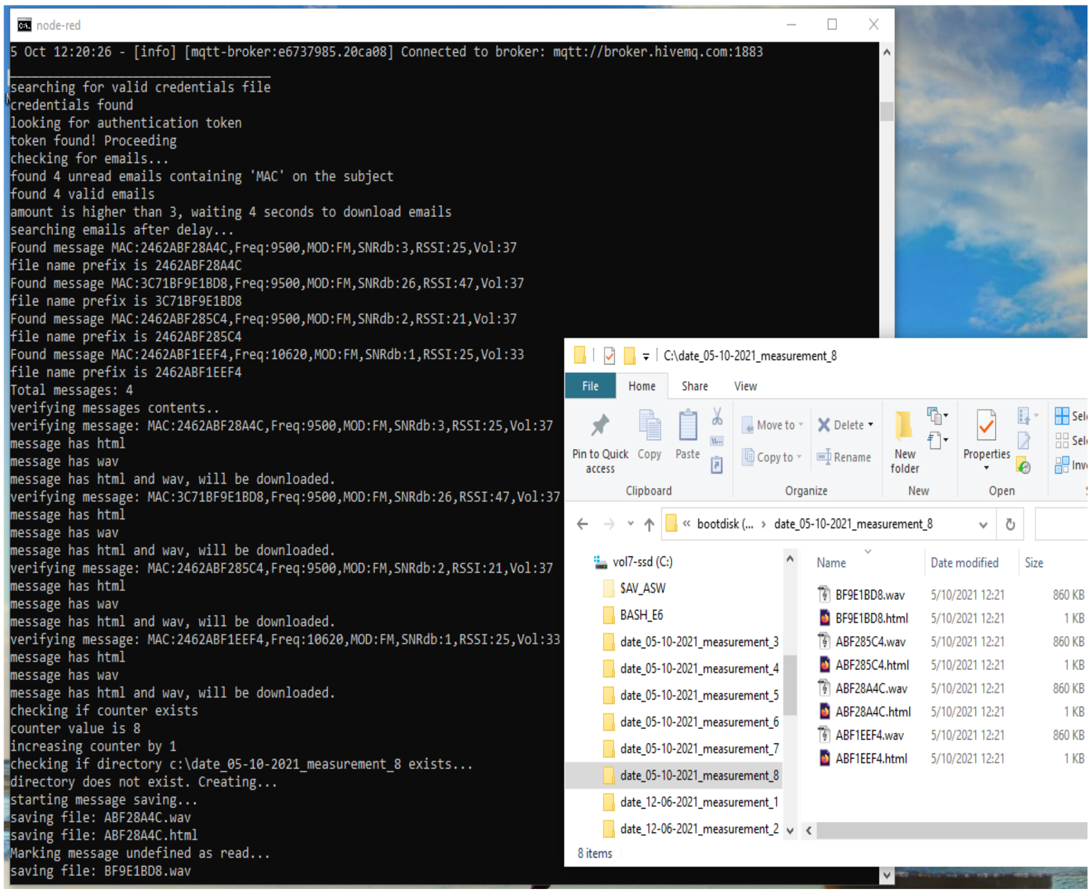
To simplify the Email Downloading process, the “Download Emails” button was also created on the “Take Measurements” tab.

When the User presses it, Node-Red is connected to the Email Server, and checks if at least three new unread emails arrived (the minimum value to do trilateration).

If true, it creates on the disk a new separate measurement folder, named as: “date_DD-MM-YYYY_measurement_X”, were X is an increasing number (counter), and there, Emails are downloaded.

Actually, each Email before saving, is sliced in two separated files, with the same name (based on the Receiving Station's ID), but with other extension.

Thus, automatically, an "xxxxxxxx.HTML", and a "xxxxxxxx.WAV" file is created for each Station's measurement, as it is show on Figure 42.



```
node-red
5 Oct 12:20:26 - [info] [mqtt-broker:e6737985.20ca08] Connected to broker: mqtt://broker.hivemq.com:1883

searching for valid credentials file
credentials found
looking for authentication token
token found! Proceeding
checking for emails...
found 4 unread emails containing 'MAC' on the subject
found 4 valid emails
amount is higher than 3, waiting 4 seconds to download emails
searching emails after delay...
Found message MAC:2462ABF28A4C,Freq:9500,MOD:FM,SNRdb:3,RSSI:25,Vol:37
file name prefix is 2462ABF28A4C
Found message MAC:3C71BF9E1BD8,Freq:9500,MOD:FM,SNRdb:26,RSSI:47,Vol:37
file name prefix is 3C71BF9E1BD8
Found message MAC:2462ABF285C4,Freq:9500,MOD:FM,SNRdb:2,RSSI:21,Vol:37
file name prefix is 2462ABF285C4
Found message MAC:2462ABF1EEF4,Freq:10620,MOD:FM,SNRdb:1,RSSI:25,Vol:33
file name prefix is 2462ABF1EEF4
Total messages: 4
verifying messages contents..
verifying message: MAC:2462ABF28A4C,Freq:9500,MOD:FM,SNRdb:3,RSSI:25,Vol:37
message has html
message has wav
message has html and wav, will be downloaded.
verifying message: MAC:3C71BF9E1BD8,Freq:9500,MOD:FM,SNRdb:26,RSSI:47,Vol:37
message has html
message has wav
message has html and wav, will be downloaded.
verifying message: MAC:2462ABF285C4,Freq:9500,MOD:FM,SNRdb:2,RSSI:21,Vol:37
message has html
message has wav
message has html and wav, will be downloaded.
verifying message: MAC:2462ABF1EEF4,Freq:10620,MOD:FM,SNRdb:1,RSSI:25,Vol:33
message has html
message has wav
message has html and wav, will be downloaded.
checking if counter exists
counter value is 8
increasing counter by 1
checking if directory c:\date_05-10-2021_measurement_8 exists...
directory does not exist. Creating...
starting message saving...
saving file: ABF28A4C.wav
saving file: ABF28A4C.html
Marking message undefined as read...
saving file: BF9E1BD8.wav
```

C:\date_05-10-2021_measurement_8

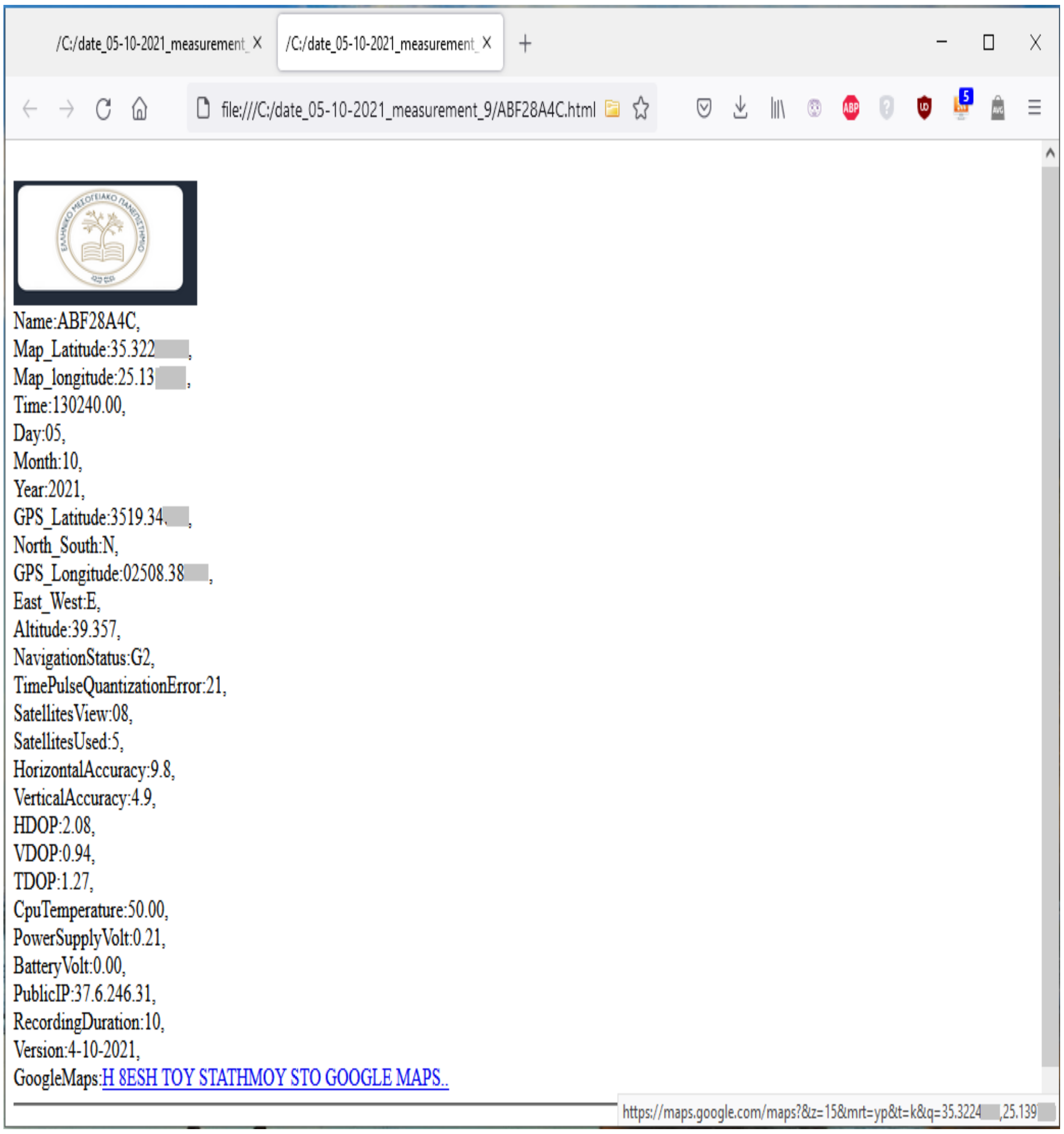
```
vol7-ssd (C:)
  $AV_ASW
  BASH_E6
  date_05-10-2021_measurement_3
  date_05-10-2021_measurement_4
  date_05-10-2021_measurement_5
  date_05-10-2021_measurement_6
  date_05-10-2021_measurement_7
  date_05-10-2021_measurement_8
  date_12-06-2021_measurement_1
  date_12-06-2021_measurement_2
8 items
```

Name	Date modified	Size
BF9E1BD8.wav	5/10/2021 12:21	860 KB
BF9E1BD8.html	5/10/2021 12:21	1 KB
ABF285C4.wav	5/10/2021 12:21	860 KB
ABF285C4.html	5/10/2021 12:21	1 KB
ABF28A4C.wav	5/10/2021 12:21	860 KB
ABF28A4C.html	5/10/2021 12:21	1 KB
ABF1EEF4.wav	5/10/2021 12:21	860 KB
ABF1EEF4.html	5/10/2021 12:21	1 KB

Figure 42 automatic Folder creation and Email downloading

Each "xxxxxxxx.html" file saved, contains on the top the HMU Unversity Logo, Receiving Station's information parameters such as: The Receiving Station Name, Google Map Latitude, Google Map Longitude, Time, Day, Month, Year, GPS Latitude, North/South GPS Longitude, East/West, Altitude, Navigation Status, TimePulseQuantizationError, SatellitesView, SatellitesUsed, HorizontalAccuracy, VerticalAccuracy, HDOP, VDOP, TDOP, CpuTemperature, PowerSupplyVolt, BatteryVolt, PublicIP, RecordingDuration, FirmwareVersion, and at the bottom a GoogleMaps link, directing to the current Receiving Station Position.

This Web Page is shown on figure 43.


/C:/date_05-10-2021_measurement_ X | /C:/date_05-10-2021_measurement_ X

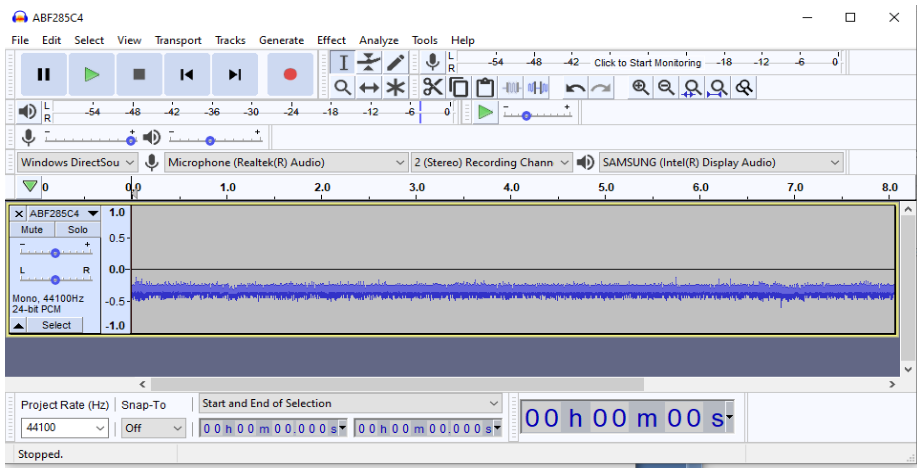
file:///C:/date_05-10-2021_measurement_9/ABF28A4C.html

Name:ABF28A4C,
Map_Latitude:35.322[redacted],
Map_longitude:25.13[redacted],
Time:130240.00,
Day:05,
Month:10,
Year:2021,
GPS_Latitude:3519.34[redacted],
North_South:N,
GPS_Longitude:02508.38[redacted],
East_West:E,
Altitude:39.357,
NavigationStatus:G2,
TimePulseQuantizationError:21,
SatellitesView:08,
SatellitesUsed:5,
HorizontalAccuracy:9.8,
VerticalAccuracy:4.9,
HDOP:2.08,
VDOP:0.94,
TDOP:1.27,
CpuTemperature:50.00,
PowerSupplyVolt:0.21,
BatteryVolt:0.00,
PublicIP:37.6.246.31,
RecordingDuration:10,
Version:4-10-2021,
GoogleMaps:H ΘΕSH TOY STATHMOY STO GOOGLE MAPS..

https://maps.google.com/maps?&z=15&mrt=yp&t=k&q=35.3224[redacted],25.139[redacted]

Figure 43 HTML file example with Station's information

Similarly, each "xxxxxxxx.wav" file saved, contains the Station's recorded sound file as shown on Figure 44. The sound quality however is not satisfying.


ABF285C4
File Edit Select View Transport Tracks Generate Effect Analyze Tools Help
Click to Start Monitoring
Windows DirectSou
Microphone (Realtek(R) Audio)
2 (Stereo) Recording Chann
SAMSUNG (Intel(R) Display Audio)
ABF285C4
Mute Solo
Mono, 44100Hz
24-bit PCM
Select
Project Rate (Hz) Snap-To
44100 Off
Start and End of Selection
00 h 00 m 00 s
Stopped.

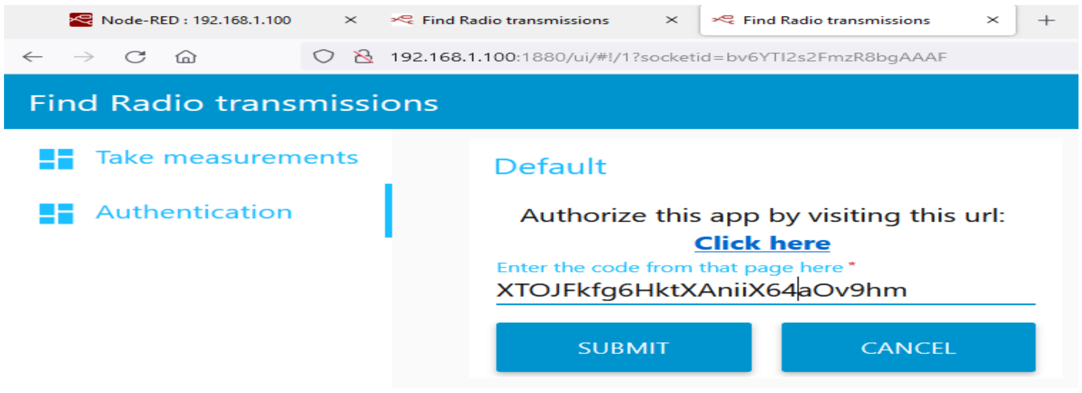
BF9E1BD8.wav

Figure 44 WAV file example recorded by a Receiving Station

5.2.2 The Authentication Tab

This Authentication Tab has to be used only when the authentication to the Gmail Server has been lost, for example after a long time period of not having use the application.

The Node-Red application establishes its authentication, by using the stored credential.jsonfile and the stored token based on OAuth (Open Authorization) service. If a no valid token is found stored, the “Authentication Tab”,asks user to connect to gmail and to download a fresh token, which is stored on disk. Afterwards, the Node-Red applicationis appearing authorized, to access the Gmail account. The Page of Authorization is shown on Figure 45.



Authorize this app by visiting this url:
Click here
Enter the code from that page here *
XTOJFkfg6HktXAniiX64aOv9hm

Figure 45 Authentication Tab

The NODE-Red Flow that was used for Authenticating and downloading mails is shown o Figure 46.

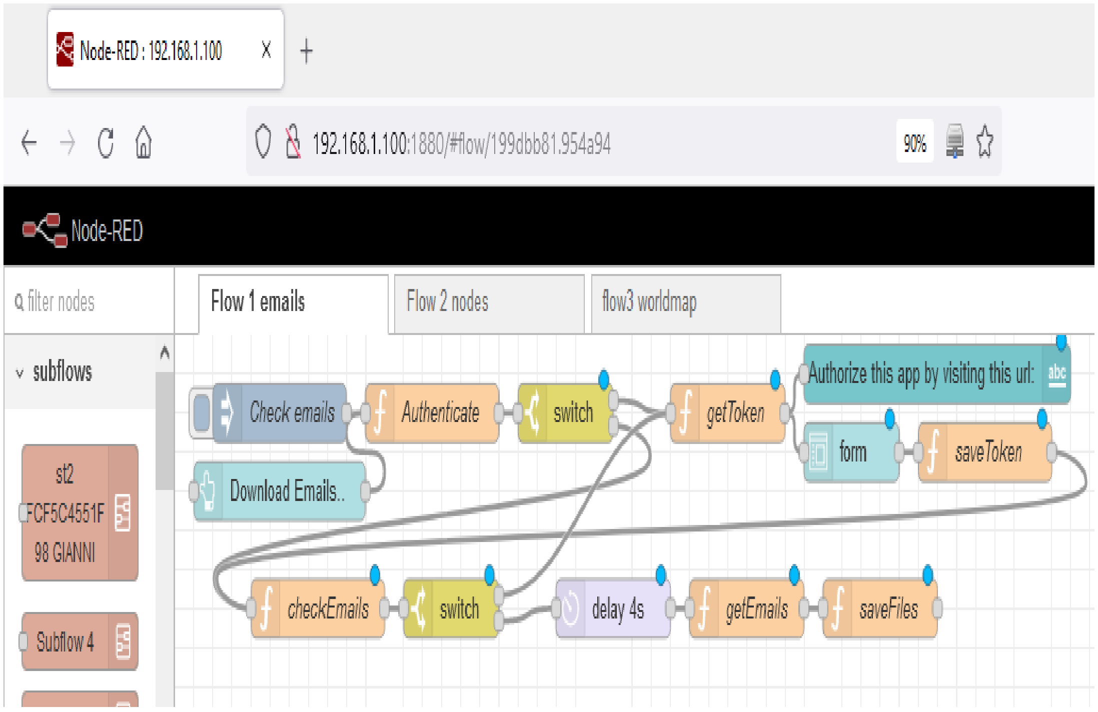


Figure 46 Authenticating and downloading emails Flow

5.2.3 The World Map Tab

Also, on another page Tab that created, (by using the node-red-contrib-web-worldmap palette node), the user can see each Receiving Station's current Position, just as it is extracted by each Receiving Station's MQTT message. On Figure 47 it is shown the five Receiving Stations location.

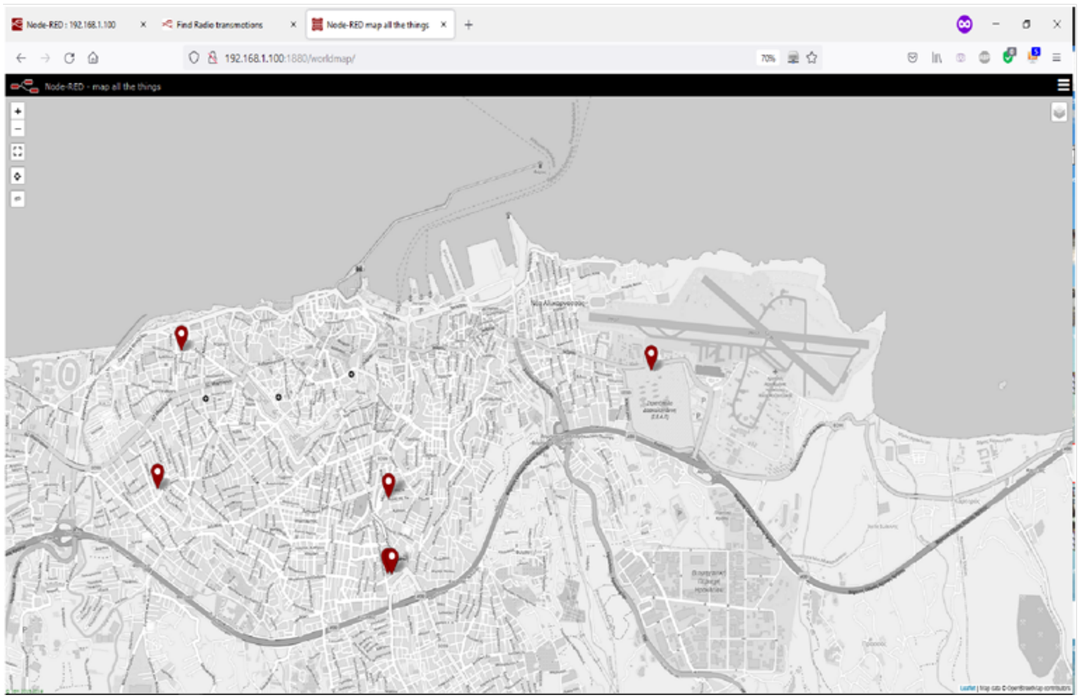

Node-RED : 192.168.1.100
Find Radio transmissions
Node-RED map all the things
192.168.1.100:1880/worldmap/
Node-RED - map all the things

Figure 47 Receiving Stations on World Map

This page was created through the World Map Flow as shown on Figure 48.

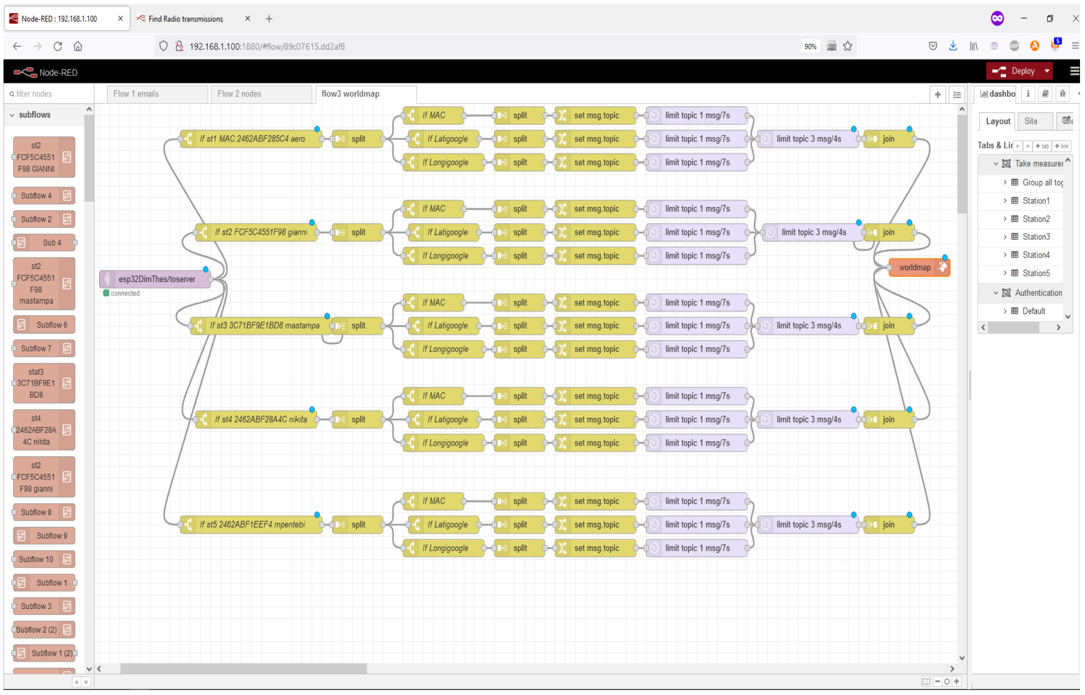


Figure 48 World Map Flow

Chapter 6: Cross-correlation, Autocorrelation, Normalization

Cross-correlation is a method to measure the similarity of two different signals.

“Autocorrelation” or “Serial Correlation” is a method to find similarity by comparing two similar signals, such as when they are time shifted. It is used for analyzing functions or series of values, such as time domain signals, or detecting repeatable patterns. The Math expression for Auto correlation “R” at lag “ℓ” for a discrete-time signal y(n) is shown on equation (1.12).

$$R_{yy}(\ell) = \sum_{n \in Z} \mathrm{y(n)}\ \overline{\mathrm{y(n}-\ell)} \qquad (1.12)$$

To have better correlation result, it is preferred the two compared signals, to have a similar amplitude range. Thus, a normalization of each signal has to be done first. Normalization means, to increase one signal by a factor, so that to have its peak level at a custom value. On Figure 49 it is shown a Signal, before, and after its Normalization.

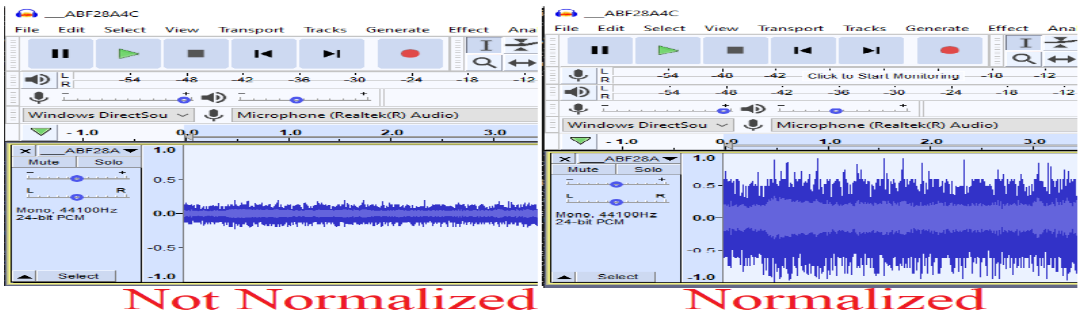


Figure 49 Signal before and after Normalization

Normalized Auto-correlation can be used, to calculate the delay of a signal while it arrives at three or more distant Receiving Stations. As the Audio Signal capturing starts at each Receiving Station concurrently due to the PPS pulse presence, then the time differences of each Station’s recording can be related to distance values as shown on simulating example on Figure 50. Thus, a hyperbolic or a circle can be draw for each Receiving Station.

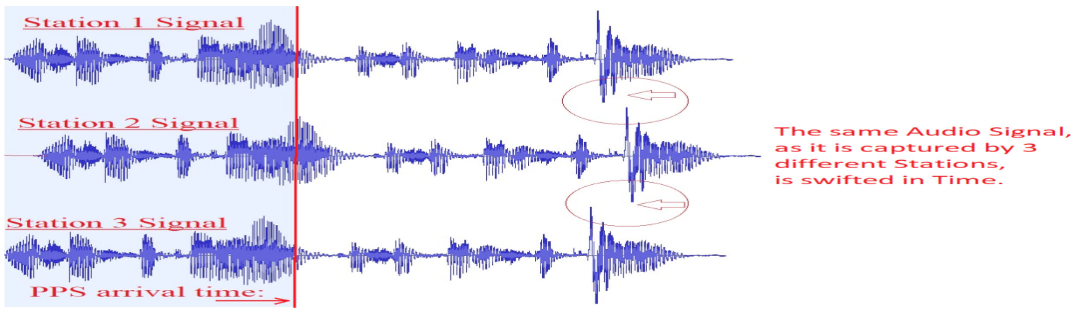
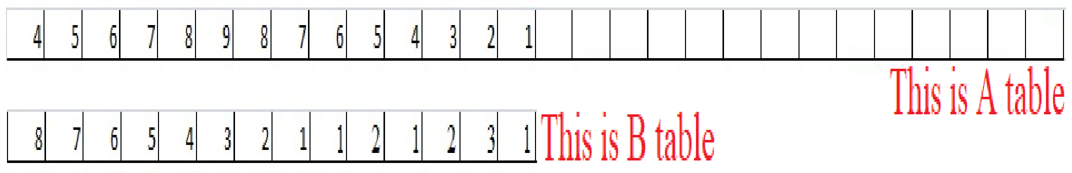

Station 1 Signal
Station 2 Signal
Station 3 Signal
PPS arrival time:
The same Audio Signal, as it is captured by 3 different Stations, is swifted in Time.

Figure 50 Simulating example of a receiving signal, while arriving at three Stations

The below procedure can be used in order to calculate the autocorrelation of a signal, which was captured concurrently by two different Stations: Each one of the two audio recordings (which have the same size), is firstly stored to a table buffer e.g. A, B.. The ‘A’ table size is twice the ‘B’ table size as it is shown on Figure 51.



4	5	6	7	8	9	8	7	6	5	4	3	2	1														

This is A table

8	7	6	5	4	3	2	1	1	2	1	2	3	1

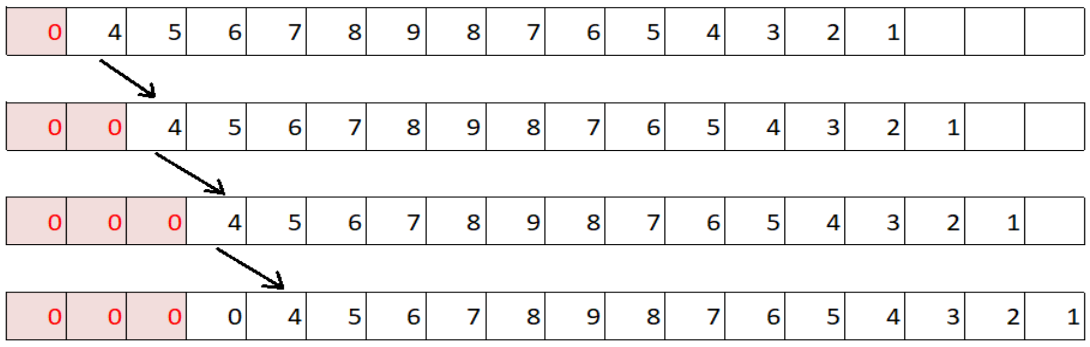
This is B table

Figure 51 tables containing A and B quantized value

Each table’s cell contains the audio file’s amplitude quantized value. We multiply each A, B common colon values, and afterwards add each result in order to find a correlation value such as:

$$(4 * 8) + (5 * 7) + (6 * 6) + (7 * 5) + (8 * 4) + (9 * 3) + \cdots = X$$

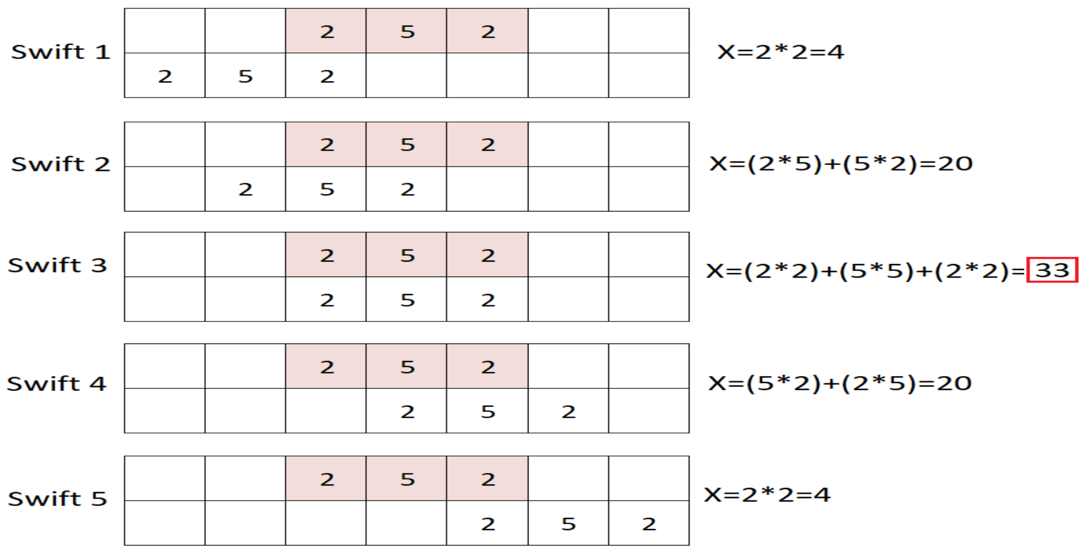
Then, we swift right for a cell position just the "A” table values as shown on Figure 52, and we calculate repeatedly **X.**



0	4	5	6	7	8	9	8	7	6	5	4	3	2	1			
0	0	4	5	6	7	8	9	8	7	6	5	4	3	2	1		
0	0	0	4	5	6	7	8	9	8	7	6	5	4	3	2	1	
0	0	0	0	4	5	6	7	8	9	8	7	6	5	4	3	2	1

Figure 52 Swift right the “A” table

After many shift repetitions, we will have many X values, of which we choose the bigger, thus that is the moment that the correlation occurs as it is shown on Figure 51.



Swift 1			2	5	2			X=2*2=4
	2	5	2					
Swift 2			2	5	2			X=(2*5)+(5*2)=20
		2	5	2				
Swift 3			2	5	2			X=(2*2)+(5*5)+(2*2)=33
			2	5	2			
Swift 4			2	5	2			X=(5*2)+(2*5)=20
				2	5	2		
Swift 5			2	5	2			X=2*2=4
					2	5	2	

Figure 53 finding correlation example

So, if we conclude e.g. that at the third position displacement the X value is the biggest, it means that at this position, the ‘A’ and ‘B’ Signals had the greater similarity, which means we had correlation.

If the table size is e.g. 44100 cells (due to the 44100 Hz sampling rate), then, each cell contains ‘1sec/44100’ audio sampling information. By having correlation on the third cell shift, then, by doing 3*(1sec/44100’), we get 132mS delay of ‘B’ signal compared to ‘A’ signal. So, this is the amount of time that the signal delayed to reach the ‘A’ position compared to “B”. By using three audio signals from three distant Receiving Stations, we have to do these three correlation combinations for finding the time differences: A-B, A-C, B-C. Otherwise, by using five Stations, these combinations can be done: A-B, A-C, A-D, A-E, B-C, B-D, B-E, C-D, C-E, D-E.

Chapter 7: Evaluation and Results

In the current Thesis, it was made an attempt to implement an economical operational RDF (Radio Direction Finder) system. Although the high effort, due to problems it was not possible to achieve a stable operation of the system and also a good sound quality of the recorded files, in order to proceed for the further processing of Normalization, Auto-Correlation and Trilateration.

The audio DC offset was randomly changing, and sometimes, the sent signal, was either missing, or was captured saturated as it is shown on Figures 54, 55.

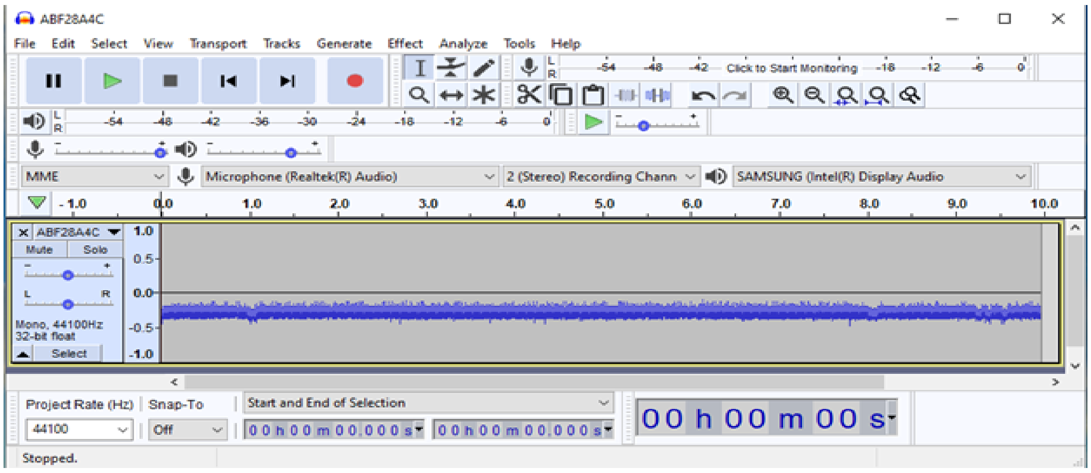

ABF28A4C
File Edit Select View Transport Tracks Generate Effect Analyze Tools Help
Click to Start Monitoring
MME
Microphone (Realtek(R) Audio)
2 (Stereo) Recording Chann
SAMSUNG (Intel(R) Display Audio
Mono, 44100Hz
32-bit float
Project Rate (Hz) 44100
Snap-To Off
Start and End of Selection
00 h 00 m 00 s
Stopped.

Figure 54 Received Audio signal with bad DC offset

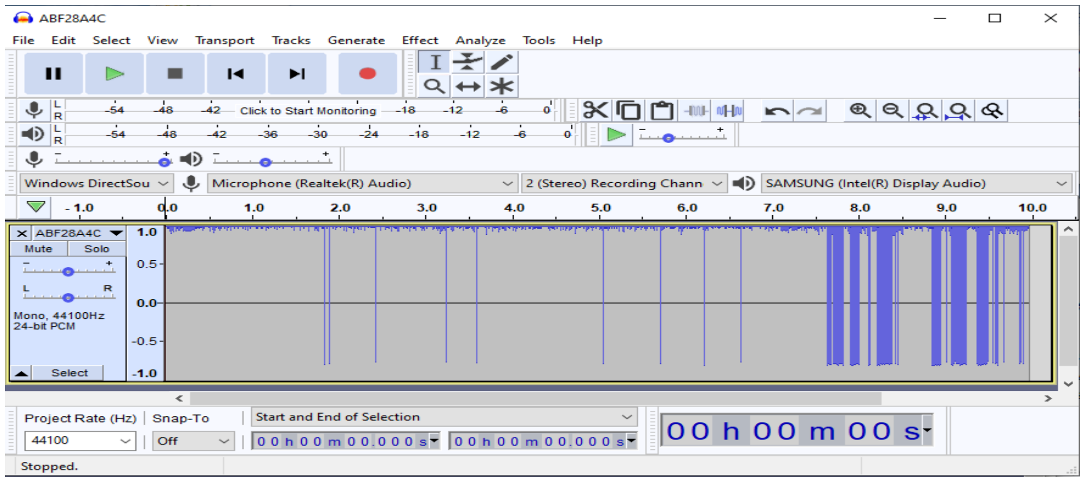

ABF28A4C
File Edit Select View Transport Tracks Generate Effect Analyze Tools Help
Click to Start Monitoring
Windows DirectSou
Microphone (Realtek(R) Audio)
2 (Stereo) Recording Chann
SAMSUNG (Intel(R) Display Audio)
Mono, 44100Hz
24-bit PCM
Project Rate (Hz) 44100
Snap-To Off
Start and End of Selection
00 h 00 m 00 s
Stopped.

Figure 55 Received Audio signal clipped

Thus, the further correct processing of the Signals could not be completed in order to accomplish the transmition location finding.

A reason for these problems' appearance seems to be the bad behavior of the ESP32 during the concurrent usage of its "ADC1" and its Wi-Fi capability, although the documentation refers that this could be an issue only for its "ADC2" channels.

Also, the electrical noise produced during the usage of the Wi-Fi module, the i2c bus and the SPI bus, seems to influent the recorded signal quality, an annoyance which can also be enriched due to the poor design quality with unshielded - floated wire connections.

Chapter 8: Future work

As a future work, for reaching better and stable results, on each Receiving Station, it is proposed the use of another Microcontroller such as Teensy 4.1, which could manipulate an external Wi-Fi module placed a little distant and not on chip, in order to minimize the RF radiation influence on the recorded signal.

The 1024KB sized RAM of this microcontroller, is enough for the temporary storing of almost 10 seconds of audio recording which demands 860 KB, and could be used in the place of the SD-Card.

An external ADC module providing higher than 12bit resolution, or alternately the usage of the digital form (I2S) of the audio exported from the Si4730 module could be preferred.

Also, a printed Board with large earth lines should be used for better noise avoidance, instead the floating cables that were used. The audio amplifiers modules that were used on this Thesis contain the chip PAM8403 which is a D class switching amplifier, which also imports electric noise and should be avoided.

It could be used another faster writing SD-card library such as “sdfat.h” instead of the “SD.h”, or either a “SD MMC” module could be used which offers a little faster response than SD.

The quantity of the Receiving Stations can increase the location accuracy. The existence of at least three stations is the minimum requirement, in order to do the math trilateration later. For 2D dimension finding, at least three Stations are required, but for 3D dimension finding at least four Stations are required. Each one of the stations, can be located at a different distant site (e.g. neighborhood, City, Municipality), depending of the RF signal Power. For example, if a low power RF signal of only 50mW has to be measured, the stations have to be located, only 100-500 meters away of the Radio transmitter, as the RF radiation is not strong enough and cannot travel farther.

For unknown reason, the GPS PPS signal sometimes disappears, and recovers later, although the clear sky. This makes difficult the correct measurement. Thus, to avoid the case of existence of multipath propagation error, it could be used for each Receiving Station, instead of a single Receiver GPS Module, a second concurrently parallelly connected GPS Receiver Module at a few meters distant, so that the PPS signal presence is more reliable. Thus, the firstly arriving PPS signal will be the more correct and will be used.

For better trilateration, it is preferred the Stations to be circularly located around the transmitter’s location. The stations, need to have access to a clear sky (to observe the GPS satellites constellation), to have access to internet through Wi-Fi, to have a reliable power supply, and to have the antennas at a non-noisy environment.

Additionally, the parameter of the 1.7-2μS IRQ transition delay of ESP32, could be considered at a better Microcontroller choice.

9. References

[1] W. Lewandowski. “Relation between GNSS system times and UTC” [available on-line] https://www.itu.int/ITU-D/tech/events/2012/ResultsWRC12_CIS_StPetersburg_June12/Presentations/Session6/S6_3_b_E.pdf (29 Jan 2022)

[2] “True-range multilateration” [available on-line] https://www.wikiwand.com/en/True-range_multilateration (29 Jan 2022)

[3] “UTC to GPS Time Correction” [available on-line] https://confluence.qps.nl/qinsy/latest/en/utc-to-gps-time-correction-271617835.html (29 Jan 2022)

[4] “Space Segment” [available on-line] https://www.gps.gov/systems/gps/space/ (29 Jan 2022)

[5] “Cesium Atoms at Work” [available on-line] http://gisweb.massey.ac.nz/topic/webreferencesites /gps/usnavy/cesium.html (29 Jan 2022)

[6] “GNSS enhancement” [available on-line] https://en.wikipedia.org/wiki/GNSS_enhancement (29 Jan 2022)

[7] “Clocks/timers, Time, and GPS”. University of Oslo. [Available on-line] https://www.uio.no/studier /emner/matnat/fys/FYS3240/v15/lectures/l11---time-timing-and-gps.pdf (29 Jan 2022)

[8] Faten Mkacher. “Optimization of Time Synchronization Techniques on Computer Networks”. [Available on-line] https://tel.archives-ouvertes.fr/tel-02988168/document#cite.Oscillators (29 Jan 2022)

[9] “About the pps pulse”. [Available on-line] http://pos.mgb-tech.com/insightpps/ (29 Jan 2022)

[10] ESP32-WROOM-32 Datasheet. [Available on-line] https://cdn.sparkfun.com/assets/learn_tutorials/8/5/2/esp32-wroom-32_datasheet_en.pdf (29 Jan 2022)

[11] “Autocorrelation”. [Available on-line] https://en.wikipedia.org/wiki/Autocorrelation (29 Jan 2022)

[12] “Understanding Cross-Correlation, Auto-Correlation, Normalization and Time Shift”. [Available on-line] https://anomaly.io/understand-auto-cross-correlation-normalized-shift/index.html (29 Jan 2022)

[14] “Esp32 Sound Recorder code”. [Available on-line] https://github.com/MhageGH/esp32_SoundRecorder (29 Jan 2022)

[15] Bernhard E. Boser. “General Purpose Input/Output (GPIO)”. [Available on-line] https://people.eecs.berkeley.edu/~boser/courses/49_sp_2019/N_gpio.html (29 Jan 2022)

[16] “Pulse-per-second signal”. [Available on-line] https://en.wikipedia.org/wiki/Pulse-per-second_signal (29 Jan 2022)

[17] “ESP32 External Interrupt Latency”. [Available on-line] https://esp32.com/viewtopic.php?f=12&t=422& sid=6f93c5817bc3f1fec1ff2605939cb40c (29 Jan 2022)

[18] Deepak Kaira. “PPS (Pulse per Second) simple as that”. [available on-line] https://www.linkedin.com/pulse/pps-pulse-per-second-simple-deepak-kaira (29 Jan 2022)

[19] “Last Will and Testament - MQTT Essentials, Part 9”. [Available on-line] https://www.hivemq.com/blog/mqtt-essentials-part-9-last-will-and-testament/ (29 Jan 2022)

[20] Ivan Voras. “Working with ESP32 Audio Sampling”. [Available on-line] https://www.toptal.com/embedded/esp32-audio-sampling (29 Jan 2022)

[21] “Volatile variable”. [Available on-line] https://www.arduino.cc/reference/en/language/variables /variable-scope-qualifiers /volatile/ (29 Jan 2022)

[22] “The physics of GPS- Trilateration”. [Available on-line] https://www.stem.org.uk/system/files/ elibrary resources /2018/08/Quantum%20Tech_Teacher %20guide_GPS%20and%20Trilateration.pdf (29 Jan 2022)

[23] “The development of a passive hydrogen maser clock for the Galileo navigation system”. [Available on-line] https://www.researchgate.net/publication/259215993_ the_development_of_a_passive_hydrogen_maser_clock_for_the_galileo_navigation_system (29 Jan 2022)

[24] A. P. Cerruti, P. M. Kintner Jr, D. E. Gary, A. J. Mannucci, R. F. Meyer, P. Doherty,d A. J. Coster, “Effect of December 2006 solar radio bursts on GPS receivers” Space Weather, vol. 6, no. 10, pp.1-10, 10/ 2008.

[25] Shahin Farahani, "Location Estimation Methods", chapter 7, Editor(s): Shahin Farahani, "ZigBee Wireless Networks and Transceivers", Newnes, 2008, Pages 225-246, ISBN 9780750683937, [Available on-line] https://www.sciencedirect.com/science/article/pii/B9780750683937000078

[26] “Angle of Arrival – High-precision positioning with the AoA method”. [Available on-line] https://www.favendo.com/angle-of-arrival (29 Jan 2022)

[27] Dr. Erwin Aitenbichler. “Telecooperation”. [Available on-line] https://www.cs.helsinki.fi/ group/cbu-ict/SummerSchool09/muc/07_Location.pdf Technische Universität Darmstadt. (29 Jan 2022)

[28] Dong-yao Zou1, Bi-wei Liu1 and Wei Yang , “An Efficient TDOA and GROA Localization Mechanism Based on GSO Algorithm with Flight Theory”., Journal of Computers Vol. 28, No. 5, 2017, Zhengzhou- China University of Light Industry

[29] Mengna Yang, David R. Jackson, Ji Chen, Zubiao Xiong, and Jeffery T. Williams. “A TDOA Localization Method Based on De-embedding the Propagation Background”. Department of ECE, University of Houston USA

[30] Tuo Xie, Chun Zhang and Zhihua Wang. “Wi-Fi TDoA Indoor Localization System Based on SDR Platform”. Institute of Microelectronics, Tsinghua University, Beijing-China. 11/2017

[31] Yunlong Wang, Ying Wu, Yuan Shen. “An efficient TDOA and FDOA based source localization algorithm via Importance Sampling”. Qingdao, China, 10/2017

[32] Li-peng Gao, Heng sun, Meng-nan liu, Yi-lin Jiang. “TDOA collaborative localization algorithm based on PSO and Newton iteration in WGS-84 coordinate system”. Harbin Engineering University China. 11/2016

[33] Xue Liu, Hongzhou Xu. “Application on Target Localization Based on Salp Swarm Algorithm (SSA)”. China 2018

[34] Zhixin Liu, Dexiu Hu, Yonziun Zhao, Yonzsheng Zhao, Rui Wang, Hongzhi Jiang. “Noise-Resistant Estimation Algorithm for TDOA and FDOA in Passive Emitter Localization China”. 5/2019

[35] Kutluyıl Dogancay, Ngoc Hung Nguyen. “Low-Complexity Weighted Pseudo linear Estimator for TDOA Localization with Systematic Error Correction”. Hungary 9/2016

[36] Julian Lategahn, Marcel Muller, Christof Rohrig. “TDoA and RSS based Extended Kalman Filter for Indoor Person Localization”

[37] Zhixin Liu, Dexiu Hu, Yongjun Zhao, Yongsheng Zhao. “A Novel Interpolation Method for TDOA and FDOA Estimation based on Second-order Cone Programming”. Germany 6/2018

[38] Shuai He. “Asynchronous Time Difference of Arrival Positioning System and Implementation”. University of Victoria. 2016

[39] Nagdeo, J.V. “Synchronization of anchors in a TDoA based ultra-wide band localization system”. Eindhoven University of Technology, 2018

[40] Michael S. Meuleners. “Design and implementation of a distributed tdoa-based geolocation system using ossie and low-cost usrp boards”. Ballston, VA. 5/2012

[41] Junming (Jamie) Wei. “RF signal sensing and source localization systems using Software Defined Radios”. Australian National University. 11/2016

[42] Yyago Lizarribar. “Signal Transmitter Localization using Low-cost SDR receivers”. Universidad Carlos III de Madrid. 2020

[43] Rui-Rong Wang, Xiao-Qing Yu1, Shu-Wang Zheng, Yang Ye. “Design of a TDOA location engine and development of a location system based on chirp spread spectrum”. 4/2016

[44] Lukas Malina, Gautam Srivastava, Petr Dzurenda, Jan Hajny, Radek Fujdiak. “A Secure Publish/Subscribe Protocol for Internet of Things”. 8/2019

[45] G. Kornaros Multi-Core Embedded Systems, CRC Press/Taylor & Francis Group, 07-April-2010, ISBN: 978-1-4398-1161-0

[46] Marcello Coppola and George Kornaros, “Automation for Industry 4.0 by using Secure LoRaWAN Edge Gateways”, in L. Andrade, F. Rousseau, (eds), Multi-Processor System-on-Chip, vol. 2., ISTE Ltd, London, and Wiley, New York, March 2021, https://iste.co.uk/book.php?id=1739, ISBN : 9781789450224

[47] Ning Lu, Nan Cheng, Ning Zhang, Xuemin Shen, Jon W.Mark “Connected vehicles: solutions and challenges,” IEEE Internet Things J. vol. 1, no. 4, pp. 289–299, Aug. 2014

[48] G. Kornaros et al., “Secure and trusted open CPS platforms,” in Handbook of Research on Solutions for Cyber–Physical Systems Ubiquity, (Advances in Systems Analysis, Software Engineering, and High Performance Computing), N. Druml et al., Eds.Hershey, PA, USA: IGI Global, 2018, pp. 301–324. [Available online] igi-global.com/chapter/secure-and-trusted-open-cps-platforms/186912

[49] G. Trouli, G. Kornaros, "Automotive Virtual In-sensor Analytics for Securing Vehicular Communication," in IEEE Design & Test, vol.37, issue 3, pp. 91-98, print ISSN: 2168-2356, online ISSN: 2168-2364, [Available online] ieeexplore.ieee.org/document/9001022

[50] C. Prehofer, G. Kornaros, M. Paolino, “Tapps-trusted apps for open cyber-physical systems”, International Conference on e-Democracy, 213-216

[51] D. Bakoyiannis, O. Tomoutzoglou, G. Kornaros, M. Coppola, “From Hardware-Software Contracts to Industrial IoT-Cloud Block-chains for Security”, 2021 Smart Systems Integration (SSI), 1-4.

[52] Karleigh J. Cameron Department of Mathematics “FDOA-BASED PASSIVE SOURCE LOCALIZATION: A GEOMETRIC PERSPECTIVE” [Available on-line] https://mountainscholar.org/bitstream/handle/10217/193165/Cameron_colostate_0053A_15200.pdf?sequence=1 , 2018