

**UNIVERSIDADE DO ESTADO DO AMAZONAS
ESCOLA SUPERIOR DE TECNOLOGIA – EST**

MARIO RUBEN LIMA DE OLIVEIRA

**DESENVOLVIMENTO DE PROTÓTIPO DE AQUISIÇÃO DE DADOS
DE LOCALIZAÇÃO DE TRANSPORTES FLUVIAIS NA REGIÃO
AMAZÔNICA UTILIZANDO RÁDIO LORA**

Manaus

2022

MARIO RUBEN LIMA DE OLIVEIRA

**DESENVOLVIMENTO DE PROTÓTIPO DE AQUISIÇÃO DE DADOS
DE LOCALIZAÇÃO DE TRANSPORTES FLUVIAIS NA REGIÃO
AMAZÔNICA UTILIZANDO RÁDIO LORA**

Pesquisa desenvolvida durante a disciplina de Trabalho de Conclusão de Curso II e apresentado à banca avaliadora do Curso de Engenharia Eletrônica da Escola Superior de Tecnologia da Universidade do Estado do Amazonas, como pré-requisito para a obtenção do título de Engenheiro em Eletrônica.

Orientador: Edgard Luciano Oliveira da Silva, Dr.

Manaus

2022

Universidade do Estado do Amazonas – UEA
Escola Superior de Tecnologia - EST

Reitor:

André Luiz Nunes Zogahib

Vice-Reitor:

Kátia do Nascimento Couceiro

Diretor da Escola Superior de Tecnologia:

Ingrid Sammyne Gadelha Figueiredo

Coordenador do Curso de Engenharia Eletrônica:

Bruno da Gama Monteiro

Banca Avaliadora composta por:

Data da defesa: <18/10/2022>

Prof. Edgard Luciano Oliveira da Silva, Dr (Orientador)

Prof. Raimundo Cláudio Souza Gomes, Dr

Prof. Fábio de Sousa Cardoso, Dr

CIP – Catalogação na Publicação

Oliveira, Mario Ruben Lima

Desenvolvimento de Protótipo de Aquisição de Dados de Localização de Transportes Fluviais na Região Amazônica Utilizando Rádio Lora / Mario Ruben Lima de Oliveira; [orientado] por Edgard Luciano Oliveira da Silva. – Manaus: 2022.

78 p.: il.

Trabalho de Conclusão de Curso (Graduação em Engenharia Eletrônica). Universidade do Estado do Amazonas, 2022.

1. LoRa. 2. Geolocalização. 3. Internet das Coisas. 4. Redes de Comunicação. I. Luciano, Edgard Oliveira da Silva.

MARIO RUBEN LIMA DE OLIVEIRA

**DESENVOLVIMENTO DE PROTÓTIPO DE AQUISIÇÃO DE DADOS
DE LOCALIZAÇÃO DE TRANSPORTES FLUVIAIS NA REGIÃO
AMAZÔNICA UTILIZANDO RÁDIO LORA**


Pesquisa desenvolvida durante a disciplina de Trabalho de Conclusão de Curso II e apresentada à banca avaliadora do Curso de Engenharia Eletrônica da Escola Superior de Tecnologia da Universidade do Estado do Amazonas, como pré-requisito para a obtenção do título de Engenheiro em Eletrônica.


Nota obtida: 9,3 (Nove vírgula três)

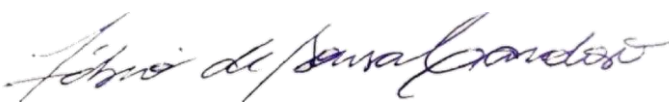
Aprovada em 18 / 10 / 2022.

Área de concentração: Internet das Coisas, Protocolos de comunicação, Microcontroladores, Sistemas de Geolocalização.

BANCA EXAMINADORA


Orientador: Edgard Luciano Oliveira da Silva, Dr.


Avaliador: Raimundo Cláudio Souza Gomes, Dr.


Avaliador: Fábio de Sousa Cardoso, Dr.

Manaus
2022

“Este trabalho é dedicado aos meus pais, irmãos e a toda minha família por todo o apoio recebido, meu muito obrigado.”

AGRADECIMENTOS

Agradeço à Deus, por iluminar meu caminho e o da minha família e por estar comigo nos momentos de maior dificuldade;

Aos meus pais, Mário Sérgio e Alessandra Anne, que muito além de provedores, são grandes amigos, incentivadores e conselheiros que sempre me incentivaram e estiveram ao meu lado.

Aos meus irmãos, Erik Gustavo e Alan Daniel, por todo o apoio e incentivo à realização deste projeto.

À minha namorada e companheira, Daniela Moura, pela motivação, carinho, incentivo e contribuição no processo de realização deste trabalho.

Ao Prof. Dr. Edgard Luciano, orientador e amigo que aceitou este desafio e me motivou do início ao fim dessa jornada. Obrigado pela confiança e pelo apoio em todos os momentos. Sua liderança e incentivo foi essencial para que o trabalho fosse finalizado com êxito.

Ao Prof. Me. Bruno Monteiro, pelas orientações e revisões técnicas na escrita deste trabalho e por todo suporte dado durante o percurso dessa jornada.

Aos colegas de curso que me acompanharam na vida acadêmica: Derik Adan, Mayumi Ono, Danilo Cunha, Darc Pabla e Estevão Assis. Especialmente aos colegas Rafael Baima e Danilo Frazão pela ajuda com a programação do protótipo.

À Universidade Do Estado do Amazonas e à Coordenação de Engenharia Eletrônica, e, pelo apoio e suporte prestado a todos os discentes e pela oportunidade de realização desse importante passo na minha vida acadêmica e profissional.

RESUMO

Este trabalho apresenta a viabilidade do uso da tecnologia LoRa para obtenção da localização de embarcações e monitoramento destes na região Amazônica. O foco está no desenvolvimento de um protótipo físico de envio das coordenadas de localização para outra embarcação que esteja nas proximidades, via rádio LoRa, a ser instalado em embarcações para utilização em situações de emergências que possam vir a ocorrer como os naufrágios, contribuindo para a segurança da navegação e das embarcações nas regiões, consequentemente reduzindo os impactos financeiros, ambientais e à vida humana que um naufrágio pode causar, pois o tempo é crucial em situações de emergências. Em função disso foi apresentada uma revisão bibliográfica sobre o contexto das embarcações na Amazônia, a fim de entender de forma geral as características de navegações. Além disso foram apresentados conceitos tecnológicos pertinentes ao desenvolvimento do trabalho nas áreas de microcontroladores, rede sem fio, Lora e LoraWan, Internet das coisas e Sistemas de Geolocalização. Por fim foi gerado um protótipo com o funcionamento do trabalho proposto.

Palavras-chaves: Lora, localização de embarcações, redes sem fio, internet das coisas.

ABSTRACT

This work presents the viability of using Lora technology to obtain the location of vessels and their monitoring in the Amazon region. The focus is on the development of a physical prototype for sending the location coordinates to another vessel that is nearby, via LoRa radio, to be installed on vessels for use in emergency situations that may occur, such as shipwrecks, contributing to the safety of navigation and vessels in the regions, consequently reducing the financial, environmental and human life impacts that a shipwreck can cause as time is crucial in emergency situations. As a result, a bibliographic review was presented on the context of vessels in the Amazon, in order to understand in general the characteristics of navigation, in addition, technological concepts relevant to the development of work in the areas of microcontrollers, wireless network, Lora and LoraWan, Internet of Things and Geolocation Systems. Finally, a prototype was generated with the functioning of the proposed work.

Keywords: Lora, Vessels traceability, wireless, Internet of Things.

LISTA DE FIGURAS

Figura 1: Modalidades de Transportes Fluviais no Amazonas.....	12
Figura 2: Navio Anna Karoline III retirado do fundo do Rio Amazonas.....	14
Figura 3: Microcontrolador PIC12F675	15
Figura 4: Microcontrolador INTEL 8051	15
Figura 5: Diagrama Funcional do ESP32	16
Figura 6: Evolução Histórica da IoT	18
Figura 7: Comunicação Dispositivo para Dispositivo.....	19
Figura 8: Comunicação Dispositivo para Nuvem	20
Figura 9: Comunicação Dispositivo para Gateway	20
Figura 10: Arquitetura dos Dispositivos.....	21
Figura 11: Vantagens Sigfox, Lora e NB-IoT na internet das coisas	24
Figura 12: Estrutura de uma rede LoRaWAN	26
Figura 13: Estrutura de uma rede LoRaWAN	27
Figura 14: Módulo Wifi LoRa 32.....	28
Figura 15: Diagrama de Pinos ESP 32 LoRa	29
Figura 16: Modulação DSSS (Dynamic Sequence Spread Spectrum)	31
Figura 17: Ilustração do Espectro de Propagação de LoRa Chirp.....	32
Figura 18: Sinais upchirp e downchirp.....	33
Figura 19: Fatores de Espelhamento Lora	33
Figura 20: Comunicação entre 2 módulos Wifi LoRa 32.....	37
Figura 21: Caminho para instalação da biblioteca ESP32.....	38
Figura 22: Gerenciador de Placas na Arduino IDE	38
Figura 23 Biblioteca instalada do módulo WiFi LoRa 32(V2)	39
Figura 24: Configuração de frequência de Modulação na IDE.....	39
Figura 25: Módulo transmissor enviando pacotes via LoRa	40
Figura 26: Módulo receptor recebendo pacotes via LoRa.....	40
Figura 27: Repositório da Biblioteca TinyGPSPlus	41
Figura 28: Repositório da Biblioteca EspSoftwareSerial	41
Figura 29: Caminho para instalação de bibliotecas na IDE Arduino	42
Figura 30: Circuito montado GPS + Wifi Lora 32	43
Figura 31: Dados extraídos do Módulo GPS GY-GPS6MV2.....	43
Figura 32: Teste do transmissor de coordenadas geográficas	44
Figura 33: Teste do receptor de coordenadas geográficas.....	44
Figura 34: Placa do Hardware Transmissor	46
Figura 35: Desenho das caixas Transmissor e Receptor	46
Figura 36: Parâmetros para corte de acrílico	47
Figura 37: Confecção em CNC a Laser	47
Figura 38: Protótipo final do transmissor	48
Figura 39: Protótipo final do receptor	48
Figura 40: Local de Teste do protótipo	49
Figura 41: Ponto de deslocamento da unidade móvel.....	50
Figura 42: Configuração dos parâmetros do Rádio LoRa	50

LISTA DE SIGLAS

AMB	Autoridade Marítima Brasileira
ARSEPAM	Agência Reguladora de Serviços Públicos Delegados e Contratados do Estado do Amazonas
CEASA	Centrais Estaduais de Abastecimento
CNC	Controle Numérico Computadorizado
CSS	Chirp Spread Spectrum
DSS	Dynamic Sequence Spread Spectrum
FDMA	Frequency Division Multiple Access
IOT	Internet Of Things, Internet das Coisas
IP	Internet Protocol
ISM	Internet Safety Management
LORA	Long Range
LPWAN	Low Power Wide Area Network
LTE	Long Term Evolution
MAC	Controle de Acesso ao Meio
NORMAN	Norma da Autoridade Marítima
OFDM	Orthogonal Frequency Division Multiplexing
PCB	Printed Circuit Board
RF	Rádio Frequência
RFID	Radio Frequency Identification
RTOS	Real Time Operating System
SAS	Software As Service
SDK	Software Development Kit
SS	Spread Spectrum
ULA	Unidade Lógica e Aritmética
TSMC	Taiwan Semiconductor Manufacturing Company
TTL	Transistor-Transistor Logic
WAN	Wide Area Network

SUMÁRIO

INTRODUÇÃO	9
REFERENCIAL TEÓRICO	11
1.1 O TRANSPORTE FLUVIAL	11
1.1.1 Características do Transporte Fluvial no Amazonas.....	11
1.1.2 Principais acidentes envolvendo embarcações.....	13
1.2 MICROCONTROLADOR ESP32	15
1.2.1 Especificações técnicas ESP32.....	16
1.2.2 Programação e Sistema Operacional do ESP32	17
1.3 INTERNET DAS COISAS	18
1.3.1 Modelos de comunicação.....	19
1.3.2 Arquitetura Básica de dispositivos IoT	21
1.3.3 Tecnologias de Comunicação LPWAN.....	22
1.4 LORA E LORAWAN	25
1.4.1 Módulo ESP32 WiFi Lora	28
1.4.2 Modulação Rádio Lora	29
1.4.3 Sinais e parâmetros Lora	32
2 MATERIAIS E MÉTODOS	35
2.1 MÉTODO PROPOSTO.....	35
2.2 MATERIAIS UTILIZADOS.....	35
3 IMPLEMENTAÇÃO DO PROJETO	37
3.1 VALIDAÇÃO DOS MÓDULOS.....	37
3.1.1 Validação do rádio LoRa	37
3.1.2 Validação do módulo GPS	40
3.1.3 Validação da integração GPS + LoRa	44
3.2 DESENVOLVIMENTO DO PROTÓTIPO	45
4 RESULTADOS OBTIDOS.....	49
4.1 CENÁRIO DE TESTE	49
CONCLUSÃO.....	54
REFERÊNCIAS	56
APÊNDICE A – CÓDIGO TRANSMISSOR	59
APÊNDICE A – CÓDIGO RECEPTOR	68

INTRODUÇÃO

O transporte fluvial no Amazonas é predominantemente usado devido à grande extensão do rio Amazonas e sua alta mobilidade. Além de promover a fomentação da economia da região atendendo às necessidades das comunidades mais afastadas da capital, compreende o deslocamento de passageiros e cargas nessas embarcações. No entanto, fatores como o mau tempo, excesso de passageiros e de carga devido à falta de fiscalização podem ocasionar um naufrágio.

O transporte de passageiros e cargas pelas vias fluviais vem sendo realizado através de embarcações mistas que são tradicionalmente utilizadas para interligar os municípios mais afastados da capital de Manaus para promover o deslocamento de produtos, serviços e passageiros. No entanto, nos últimos anos têm ocorrido muitos acidentes com um elevado número de vítimas. Dados mostram que pelo menos mil pessoas morreram nos rios da região amazônica desde 1981 (FOLHA DE SÃO PAULO, 2017).

Segundo Ferreira (2016), o setor mostra-se parcialmente ineficaz pois a fiscalização é dificultosa devido à falta de recursos e a alta quantidade de embarcações. O licenciamento das embarcações e da tripulação é regulamentado pela Marinha e sua fiscalização compete à Capitania dos Portos de Manaus, porém existe um número significativo de embarcações que trafegam sem as devidas autorizações e de forma irregular.

Ressalta-se a importância do tema em pauta devido a sua relevância em nossa atualidade pelo fato que o setor de transportes é vital para o crescimento do país. Tanto no transporte de passageiros quanto no transporte de cargas há inúmeros desafios a serem superados. A tecnologia deve prover solução efetiva de menor custo que ofereça maior segurança com o menor impacto ambiental possível (SANCHES, 2019).

O projeto de pesquisa explorará conhecimentos adquiridos durante o curso de Engenharia Eletrônica, principalmente das disciplinas: Microprocessadores e Microcontroladores, Introdução às Redes de Comunicação, Sistemas de Geo-Localização baseados em Satélites, Propagação e Antenas para Sistemas Embarcados e Linguagem de Programação I e II.

A falta do monitoramento e da rastreabilidade de embarcações podem ser cruciais em casos de naufrágios ou acidentes resultando em impactos financeiros, ambientais e à vida humana. Além disso verifica-se também a inexistência de um sistema integrado que alerte pânico ou perigo e interaja através de uma tecnologia de comunicação de dados.

Esta pesquisa tem como objetivo o desenvolvimento de um protótipo de sistema em laboratório, localizado na Escola Superior de Tecnologia (EST) da Universidade do Estado do Amazonas, para realizar o monitoramento remoto da posição e localização de transportes fluviais do Rio Amazonas, baseado na plataforma ESP32 LoRa para envio da localização entre barcos utilizando Rádio Lora.

Nesse contexto o desenvolvimento de uma revisão da literatura sobre o tema proposto contribuiu como parte da solução destes problemas, uma vez que as revisões têm a função de possibilitar uma análise sobre um determinado assunto para que o objetivo final seja alcançado no desenvolvimento de um protótipo de rastreamento de localização e posição de transportes fluviais.

Este trabalho está dividido em quatro capítulos, os quais são: referencial teórico, materiais e métodos, implementação e análise dos resultados obtidos.

O capítulo 1 apresenta uma revisão bibliográfica sobre os principais conceitos e tecnologias que serviram de base para o desenvolvimento do projeto. Neste capítulo é descrito o funcionamento do transporte fluvial e suas características; além disso a visão geral de microcontroladores Esp32, especificações técnicas e pinagem; conceitos de internet das coisas bem como os modelos de comunicação, das redes LPWAN, assim como o estudo a respeito da tecnologia LoRa e do protocolo LoRaWan.

No capítulo 2 são mostrados os materiais utilizados para a implementação do trabalho, bem como o método proposto. Onde são descritas as características dos componentes usados para a confecção do protótipo e os softwares de apoio.

O capítulo 3 contém a descrição detalhada da implementação do projeto, contendo as etapas de validação individuais e conjuntas dos componentes. As etapas de confecção do protótipo final também estão descritas neste capítulo.

O capítulo 4 apresenta o resultado do teste de campo, os dados coletados do indicador de intensidade do sinal recebido, assim como os resultados obtidos durante a etapa de validação do sistema.

Por fim, no último capítulo, estão descritas as considerações finais e sugestões de melhorias para trabalhos futuros.

REFERENCIAL TEÓRICO

Neste capítulo, serão abordados os aspectos teóricos dos assuntos relacionados à pesquisa. Inicialmente, será feita uma contextualização sobre a importância, a dinâmica e estrutura do transporte fluvial no Amazonas. Em seguida, serão abordados os conceitos introdutórios sobre microcontrolador ESP32, Internet das coisas, protocolos de comunicação e a modulação LoRa.

1.1 O TRANSPORTE FLUVIAL

O transporte fluvial no Amazonas está bastante presente devido à imensa extensão do rio Amazonas e a sua mobilidade interestadual. Essa é uma grande vantagem hidrográfica que esse bem natural dispõe para fomentar a economia da região ao mesmo tempo que provê a necessidade dos povos e comunidades que estão mais afastadas da capital.

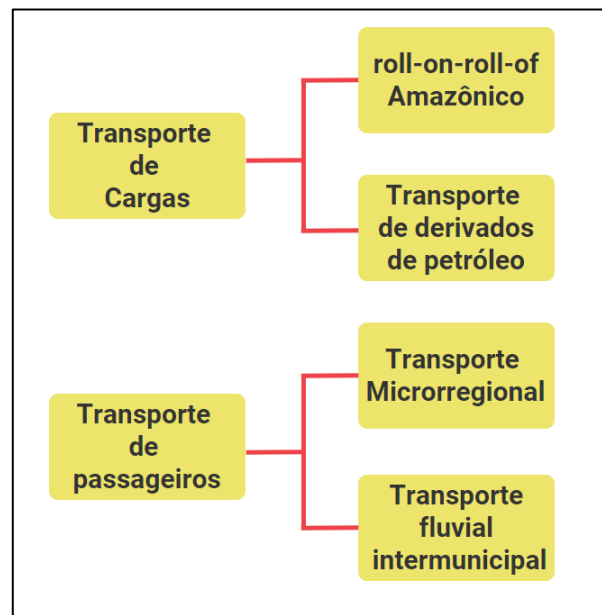
O transporte hidroviário, também chamado de transporte aquaviário, refere-se a todos os tipos de transporte realizados sobre a água com o intuito de transportar cargas ou pessoas. Isso inclui tanto o transporte fluvial em rios e em lagos, que são denominados de aquaviário interior ou transporte marítimo. Este último pode ser dividido em duas características de transporte, sendo eles: Internacional - marítimo de longo curso e Nacional – cabotagem (NOVAES, 2004).

1.1.1 Características do Transporte Fluvial no Amazonas

O transporte fluvial no Amazonas possui algumas características bem peculiares em relação às outras regiões do Brasil. Isso se deve às características únicas como ser o maior rio em volume de água do mundo, possuindo 6992 metros de extensão e uma vazão média de 210 mil m^3/s (SUÇUARANA, 2022).

Podemos classificar, para melhor entendimento, no âmbito da divisão dos transportes fluviais no Amazonas da seguinte forma: Transportes de cargas, subdividindo-se em *roll-on-roll-off* amazônico e transportes de derivados de petróleo; e transporte de passageiros, que por sua vez, subdivide-se em transporte microrregional e transporte fluvial intermunicipal de passageiros (NOGUEIRA, 1999). A figura 1 resume essa subdivisão:

Figura 1: Modalidades de Transportes Fluviais no Amazonas



Fonte: Autoria Própria

O roll-on-off amazônico foi uma alternativa rodofluvial que começou a ganhar destaque por meados da década de 80. Este modelo surgiu como “adaptação do modelo internacional de roll-on-roll-off, transporte de contêineres em navios, carga unitizada, sem a necessidade de grande número de carregadores, como o exige a carga solta” (NOGUEIRA, 1999), denominado, na Amazônia, de “roro caboclo”: consiste em colocar caminhões e carretas sobre um comboio de balsas, que são transportadas por um barco potente, denominado empurrador.

O transporte de derivado de petróleo é um tipo de transporte fluvial especializado. As relações econômicas para a região são evidenciadas nessa modalidade de transporte de carga, em virtudes das características da carga transportada, por ser uma carga perigosa e ter grande importância para as comunidades do interior da Amazônia, onde constituirá fonte de energia para geração de energia elétrica para as comunidades, além de ser fonte de energia para motores de luz para uso privado (DAVID, 2010).

O transporte microrregional também é um tipo de transporte com característica de baixa renda. Segundo David (2010), este tipo de transporte é realizado em sua quase totalidade por ribeirinhos. Os operadores deste empreendimento também se apresentam como pessoas de baixa renda, seus lucros são irrisórios e, em geral, não têm capital para manter seu próprio barco. Nesse meio de transporte, as viagens são de curta distância e os proprietários das embarcações utilizadas quase sempre realizam atividades complementares como agricultura, pesca ou venda de produtos agrícolas para Manaus.

O transporte fluvial intermunicipal de passageiros tem uma característica bastante peculiar, que é o transporte de passageiros e cargas ao mesmo tempo. Essas cargas são mais

leves do que as transportadas por embarcações que trabalham exclusivamente com carga. No entanto, ocupam um espaço considerável nos porões dos barcos e, muitas vezes, até em locais destinados aos passageiros, tornando a viagem mais longa (DAVID, 2010).

O transporte de produtos e pessoas na Amazônia é feito por via fluvial, diferentemente de outros estados, que têm seu transporte principal por vias rodoviárias. Isso é comprovado pelo grande número de pessoas, produtos agrícolas e peixes que chegam diariamente a Manaus vindos de vários municípios.

Segundo dados da Agência Reguladora de Serviços Públicos Delegados e Contratados do Estado do Amazonas (ARSEPAM) em Manaus, o Terminal das Centrais Estaduais de Abastecimento (CEASA) absorve o transporte rodofluvial, realizado por balsas e *ferryboats*, para complementar o transporte rodoviário da BR-319, que atualmente é descontínuo pela intercessão do Rio Negro e Rio Solimões. Esse terminal atende, também, aos passageiros, por meio de lanchas rápidas e expressas, caracterizando o transporte transversal para a outra margem do rio, também em continuidade ao percurso da BR-319, localizada no município Careiro da Várzea.

1.1.2 Principais acidentes envolvendo embarcações

Segundo um levantamento realizado pelo portal uol (2020), pelo menos 182 pessoas morreram em 142 naufrágios de barcos desde 2017 nos 16 mil km de rios na Amazônia. As informações foram fornecidas pelas seis Capitânicas dos Portos que fiscalizam a malha hidroviária da Amazônia.

O Estado do Amazonas aparece em primeiro lugar em número de mortes por acidentes com embarcações na última década. Os dados são do Ministério da Marinha. E de acordo com o Ministério da Saúde, o Amazonas se mantém na dianteira em 2017 (BNC AMAZONAS, 2020).

O debate sobre a segurança da navegação e das embarcações na região Norte voltou à tona com o naufrágio, no dia 29 de fevereiro, da embarcação Anna Karoline 3. O barco de médio porte transportava cerca de 60 pessoas. Esse naufrágio foi um desastre ocorrido na madrugada de 29 de fevereiro de 2020 no Rio Amazonas que deixou ao menos 40 mortos. A embarcação saiu por volta das 18h de sexta-feira, 28 de fevereiro da cidade de Santana, Amapá, com destino à cidade de Santarém no Pará.

Figura 2: Navio Anna Karoline III retirado do fundo do Rio Amazonas



Fonte: (BNC AMAZONAS, 2020)

Outro caso mais recente foi em Nhamundá, no interior do Amazonas, na madrugada de 20 de agosto de 2022, onde três pessoas morreram após um barco naufragar. De acordo com as informações preliminares, a embarcação saiu da comunidade do Laguinho e afundou por volta das 3h30, no rio Paraná, afluente do rio Nhamundá, na zona rural do município, que fica a 380 km de distância da capital Manaus (YAHOO NOTÍCIAS, 2022).

Uma das causas é a superlotação de cargas e passageiros conforme visto no capítulo 1. O Portal da uol (2020) reforça que a região amazônica tem várias localidades remotas, que não contam com estradas e por meio das quais só se chega a partir de uma embarcação. É comum constatar barcos transportando pessoas com a lotação acima de suas capacidades.

A Marinha do Brasil por ser Autoridade Marítima Brasileira (AMB), cabe a ela orientar e fiscalizar o tráfego aquaviário, por meio das suas Capitânicas, Delegacias e Agências distribuídas por todo o Brasil. Antes de sair para navegar, é importante que os tripulantes e passageiros estejam atentos às regras e aos procedimentos estipulados nas Normas da Autoridade Marítima (NORMAM), que podem ser consultadas por meio do site da Diretoria de Portos e Costas no *link* “Normas e Legislações” (MARINHA DO BRASIL, 2022).

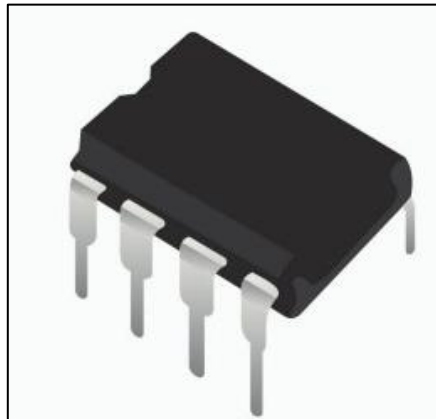
O documento da embarcação deve estar em dia e a bordo. O condutor precisa portar a sua habilitação e não pode fazer uso de bebida alcoólica. Os extintores de incêndio têm que estar dentro do prazo de validade e em local de fácil acesso. Toda embarcação deve possuir material de salvatagem, tais como coletes e boias de acordo com a dotação da embarcação. Estas são algumas recomendações a serem seguidas pelos condutores de embarcações que podem ser verificadas, também, pelos passageiros (MARINHA DO BRASIL, 2022).

Em caso de acidentes da navegação, os procedimentos a serem adotados também estão regidos em Norma da Autoridade Marítima (NORMAM-09), que determina a instauração de inquérito sobre o ocorrido.

1.2 MICROCONTROLADOR ESP32

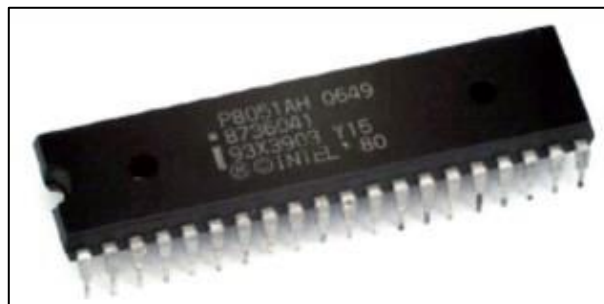
Microcontroladores são circuitos integrados que possuem em seu interior todos os componentes necessários ao seu funcionamento dependendo unicamente da fonte de alimentação externa. Para ÉDILUS & RONALDO (2019) um microcontrolador é, em última análise, um computador em um único chip (Figuras 3 e 4). Esse chip contém um processador de Unidade Lógica e Aritmética (ULA), memória, periféricos de entrada e de saída, temporizadores, dispositivos de comunicação serial, dentre outros.

Figura 3: Microcontrolador PIC12F675



Fonte: (ÉDILUS & RONALDO, 2016)

Figura 4: Microcontrolador INTEL 8051



Fonte: (ÉDILUS & RONALDO, 2016)

Conforme informações da fabricante ESPRESSIF SYSTEMS (2022), o ESP32 é um único chip Wi-Fi e Bluetooth de 2,4 GHz criado usando a tecnologia de ultrabaixa potência de 40 nm da empresa Taiwan Semiconductor Manufacturing Company (TSMC). Este é um projeto

para alcançar a potência ideal e desempenho de Rádio Frequência (RF), exibindo resiliência, adaptabilidade e confiabilidade em uma ampla gama de aplicações e condições de energia.

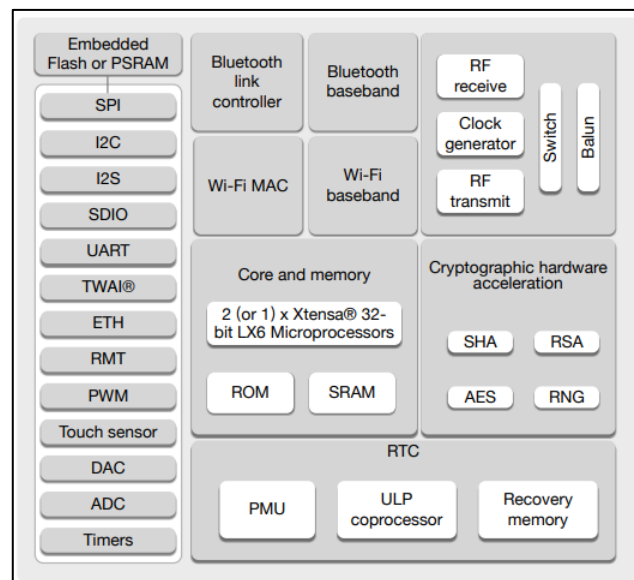
O ESP32 foi projetado para dispositivos móveis, eletrônicos vestíveis e aplicativos de Internet das Coisas (IoT). Ele apresenta todas as características de última geração de chips de baixa potência, incluindo ajuste fino de clock, vários modos de energia, e dimensionamento dinâmico de energia (ESPRESSIF SYSTEMS, 2022).

Segundo FERNANDO K (2017), atualmente existem 2 formas de se encontrar o ESP32: na forma de chip e na forma de módulo. O ESP32 chip e o ESP32 módulo possuem diferentes tamanhos e números de pinos, logo a escolha da forma que será o ESP dependerá dos requisitos de design e objetivos do projeto.

1.2.1 Especificações técnicas ESP32

O ESP32 é um chip que oferece conectividade (Wi-fi e Bluetooth, com frequência 2,4Ghz), poder computacional (CPU + memórias), I/Os, RTC, suporte à comunicações diversas (SPI, I2C, I2S, etc). Suporte à operação de baixo consumo e blocos de *hardware* dedicado à segurança em um único chip (Bertoleti, 2019). Na figura 5 pode-se observar o diagrama de blocos do ESP32.

Figura 5: Diagrama Funcional do ESP32



Fonte: (ESPRESSIF SYSTEMS, 2022)

O ESP32 apresenta vantagens em suas especificações quando comparado a outras placas com microcontroladores do mercado apresentando como diferencial a presença da tecnologia

Bluetooth, muito mais GPIOs e ADCs, além da presença de 2 DACs e frequência de clock de 160 MHz. O quadro 1 demonstra um comparativo entre os microcontroladores.

Quadro 1: Comparação entre recursos de microcontroladores

	ESP32	ESP8266	Arduino Uno R3
Cores	2	1	1
Arquitetura	32 bits	32 bits	8 bits
Clock	160Mhz	80MHz	16 MHz
Wifi	Sim	Sim	Não
Bluetooth	Sim	Não	Não
Ram	512kB	160kB	2kB
Flash	16Mb	16Mb	32kB
GPIO	36	17	14
Interfaces	SPI/I2C/UART/I2S/CAN	SPI/I2C/UART/I2S	SPI/I2C/UART
ADC	18	1	6
DAC	2	0	0

Fonte: Adaptado de (FERNANDO K, 2017)

1.2.2 Programação e Sistema Operacional do ESP32

O ESP32 pode ser programado de várias formas, com destaque via *Software Development Kit* (SDK) oficial da *Espressif Systems*, em linguagem C. Uma das alternativas de software mais utilizadas no desenvolvimento de programações em placas ESP tem sido a plataforma de desenvolvimento Arduino, pois esta possui uma grande comunidade de desenvolvedores, que costumam disponibilizar diversos exemplos e projetos de forma *Open Source* (gratuita). A transferência de programações a qualquer microcontrolador, exigirá a seleção de pelo menos um dos modelos de placas compatíveis com o software, no caso de placas ESP32 ou similares, será preciso adicionar um novo conjunto nas opções de gerenciamento de placas, no menu *file >>preferences* (BERTOLETI, 2019).

Quanto ao sistema operacional, o ESP32 utiliza um sistema operacional de tempo real chamado na língua estrangeira de *Real Time Operating System* (RTOS) que é um sistema operacional projetado para executar multitarefas onde cada evento tem um padrão de tempo pré-definido. Bertoleti (2019) cita que uma das vantagens do ESP32 em relação ao seu sucessor em sistemas operacionais é a utilização do famoso e consolidado FreeRTOS trazendo como vantagem a facilidade de migração/portabilidade de outros projetos feitos utilizando FreeRTOS em outras plataformas ESP32.

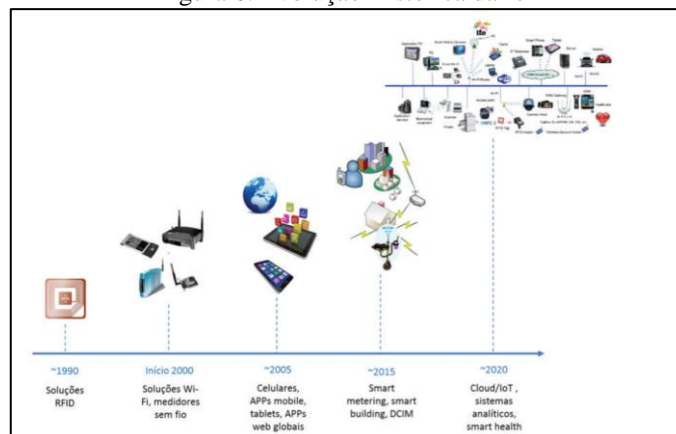
1.3 INTERNET DAS COISAS

A Internet das Coisas, do inglês *Internet of Things* (IoT), emergiu dos avanços de várias áreas como sistemas embarcados, microeletrônica, comunicação e sensoriamento. A internet das coisas é uma nova visão para a internet, em que a internet passa a conectar não apenas computadores mas também objetos do dia a dia. Segundo Faccioni (2016), são inúmeras as aplicações para IoT. Atualmente, muito se fala em telemetria, aplicações com coleta de dados em ambientes diversos, possibilidade de atuação direta sobre objetos de todos os tipos, relacionamento em rede e interação de objetos entre si, interação entre objetos e pessoas.

A história da IoT começa muito antes da internet quando Kevin Ashton, em junho de 2009, utilizou o termo *Internet of Things* pela primeira vez em seu trabalho intitulado “*I made at Procter & Gamble*” em 1999. Na época, a IoT era associada ao uso da tecnologia *Radio Frequency identification* – RFID. O propósito dos seus estudos era de conectar as etiquetas de RFID, chamadas de “tags”, com a internet, o que mudou profundamente a ideia das companhias em relação a esse sistema de identificação, pois possibilitaria conhecer, em tempo real e *on-line*, toda a movimentação de cargas e produtos.

E a partir de então, a história da IoT se confirma, sendo que em 2008 acontece a primeira conferência internacional sobre o tema da internet das coisas em Zurich na Suíça: *First International Conference, IOT 2008*. Nessa conferência, são discutidos, em sessões científicas, temas como RFID, sensoriamento, aspectos de negócios, tecnologias de conexão e conversão de protocolos, dando origem a um enorme campo de debates e evoluções técnicas, consolidando o cenário da “internet das coisas”, cuja história está resumida na figura 6 abaixo:

Figura 6: Evolução Histórica da IoT



Fonte: (FACCIONI FILHO, 2016)

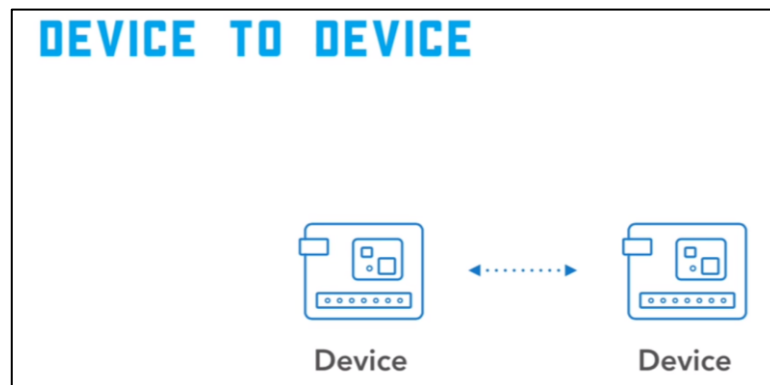
1.3.1 Modelos de comunicação

Com tanto avanço e evolução no campo da comunicação, é importante entender como os dispositivos IoT se conectam e se comunicam. A LoraWan Academy, programa educacional individualizado para estudantes e profissionais interessados na criação de soluções de IoT de longo alcance, define 3 modelos diferentes de comunicação IoT:

- Comunicação de dispositivo para dispositivo (*Device to Device*);
- Comunicação de dispositivo para nuvem (*Device to Cloud*);
- Comunicação de dispositivo para gateway (*Device to Gateway*).

Na comunicação dispositivo para dispositivo (*Device to Device*), figura 7, dois ou mais modelos são diretamente conectados um ao outro. Este modo de comunicação é frequentemente usado em automação residencial onde pequenos pacotes de dados de informação são enviados entre os dispositivos. Exemplos de aplicações estão no uso de lâmpadas, termostatos e trancas de portas. Estes dispositivos usam alguns protocolos como: Bluetooth, Z-Wave ou Zigbee para estabelecer a comunicação dispositivo para dispositivo.

Figura 7: Comunicação Dispositivo para Dispositivo

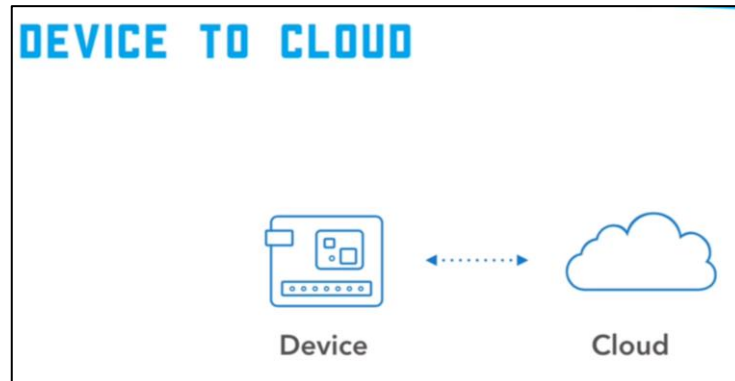


Fonte: (SEMTECH , 2022)

Na comunicação de dispositivo para nuvem (*Device to Cloud*), figura 8, o dispositivo é conectado diretamente a um serviço de nuvem para troca e controle do tráfego de dados. Geralmente se faz o uso de mecanismos de comunicação existentes como a tradicional Ethernet ou Wi-fi para estabelecer uma conexão entre o dispositivo e o protocolo IP. Vantagens desse método de comunicação é a função de obter acesso remoto aos dispositivos via *smartphone* ou interface web e fazer atualizações de *software* posteriormente. Exemplos desta comunicação é a de um termostato inteligente, este manda dados para a nuvem e a energia da casa é analisada.

Outro exemplo é a *Smart TV* em que a televisão usa a conexão da internet para enviar dados sobre o comportamento do usuário para a nuvem para análises posteriores.

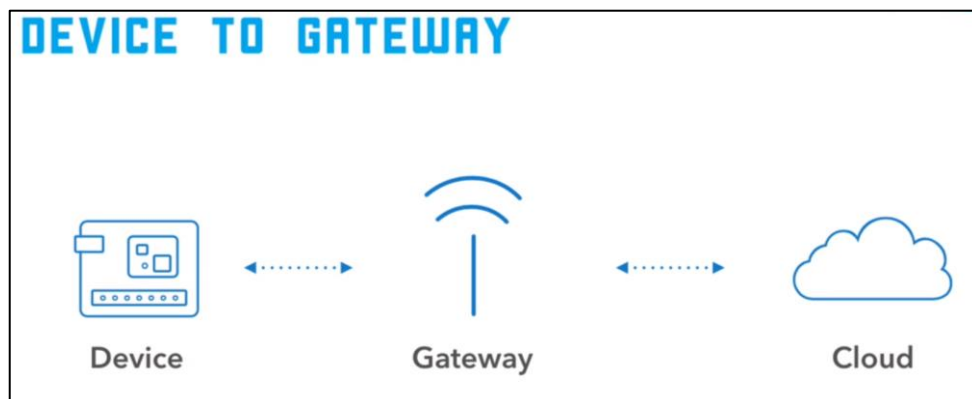
Figura 8: Comunicação Dispositivo para Nuvem



Fonte: (SEMTECH ,2022)

O último modelo, figura 9, é a Comunicação de dispositivo para *Gateway (Device to Gateway)*. Os dispositivos IoT nestes casos se conectam através de um serviço intermediário para alcançar um serviço de nuvem. Em termos simples, isto significa que há uma aplicação de software operando em um *gateway* local que atua como intermediário entre o dispositivo e o serviço de nuvem e provê segurança junto com outras funcionalidades. É frequentemente usado para desacoplar uma conexão específica da conexão *wireless* da conectividade da internet. Um exemplo dessa comunicação é um dispositivo rastreador *fitness* que é conectado a um telefone que nesse caso atua como um *gateway*.

Figura 9: Comunicação Dispositivo para Gateway



Fonte: (SEMTECH, 2022)

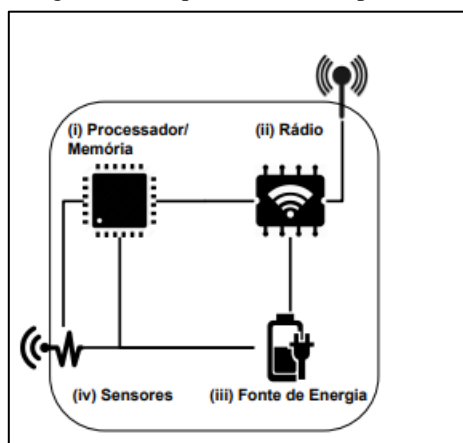
1.3.2 Arquitetura Básica de dispositivos IoT

Nesta seção veremos em mais detalhes a arquitetura básica dos dispositivos inteligentes. Segundo Santos (2016) a arquitetura básica dos dispositivos inteligentes é composta por quatro unidades:

1. Processamento/memória;
2. Comunicação;
3. Energia;
4. Sensores/atuadores.

A Figura 10 mostra uma visão geral da arquitetura de um dispositivo e sua interligação entre seus componentes.

Figura 10: Arquitetura dos Dispositivos



Fonte: (SANTOS, 2016)

Unidades de Processamento/Memória: Este dispositivo é composto por uma memória interna para armazenamento de dados e programas, um microcontrolador e um conversor analógico-digital para receber os sinais dos sensores. As CPUs utilizadas nestes dispositivos são tipicamente as mesmas utilizadas em sistemas embarcados e comumente não possuem alto poder computacional. Muitas vezes existe uma memória externa do tipo flash, que serve como memória secundária, por exemplo, para manter um *log* de dados. As vantagens dessas unidades são o baixo consumo de energia e pelo fato de ocupar o menor espaço possível.

Unidades de comunicação são as que se utilizam para transmitir informações entre os sistemas. Estas unidades podem ser tanto de alta gama (rádio e fibra óptica) como de baixa gama (internet). A maioria das plataformas usam rádio de baixo custo e baixa potência, o que

faz com que a comunicação seja de curto alcance. Como consequência, as perdas frequentes ocorrem em todo o processo de transmissão da informação.

A fonte de energia é responsável por fornecer energia elétrica aos componentes de um dispositivo inteligente. Normalmente, a fonte de alimentação é composta por uma bateria (recarregável ou não) e um conversor AC/DC que tem a função de alimentar os componentes. No entanto, existem outras fontes de energia, como energia elétrica, solar e até mesmo captação de energia do meio ambiente por meio de técnicas de conversão (por exemplo, energia mecânica em energia elétrica).

Sensores são dispositivos que capturam informações sobre o ambiente em que estão inseridos. Neste caso, o sensor é um dispositivo que mede grandezas físicas como temperatura, umidade e pressão. Existem literalmente centenas de sensores diferentes disponíveis hoje no mercado que podem medir essas variáveis físicas. Já os atuadores são dispositivos que executam alguma ação atendendo a comandos manuais elétricos ou mecânicos.

1.3.3 Tecnologias de Comunicação LPWAN

Low power wide area (LPWA) é um termo genérico para um grupo de tecnologias com as principais características de longa vida útil da bateria com conectividade de área ampla. O custo dos chipsets é baixo com uma largura de banda de comunicação de dados limitada. No início de 2013, o termo 'LPWA' nem existia. O fato de que o espaço LPWA desde então se tornou um dos aspectos de desenvolvimento mais rápido do mercado da Internet das Coisas é um testemunho do incrível potencial das tecnologias LPWA (SEMTECH, 2022).

As redes LPWA representam um novo paradigma de comunicação, que complementarmente tecnologias celulares e sem fio de curto alcance no endereçamento de diversos requisitos de aplicativos de IoT. Tecnologias LPWA oferecem conjuntos exclusivos de recursos, incluindo conectividade de área ampla para dispositivos de baixa potência e baixa taxa de dados (U. RAZA et al, 2017).

Uma rede de baixo consumo (chamada LPWAN, do inglês Low Power Wide Area Network) foi escolhida para realizar a comunicação entre os microcontroladores. Segundo Bardyn et al (2016), o LPWAN abrange aplicativos de longa distância de baixo consumo de energia, que têm um alcance superior a alguns quilômetros e velocidades de dados que variam de 10bps a alguns kbps.

A maioria dos aplicativos LPWAN serão novos, pois as tecnologias LPWAN conectarão coisas para as quais nenhuma solução de conexão viável existia anteriormente. Essa definição

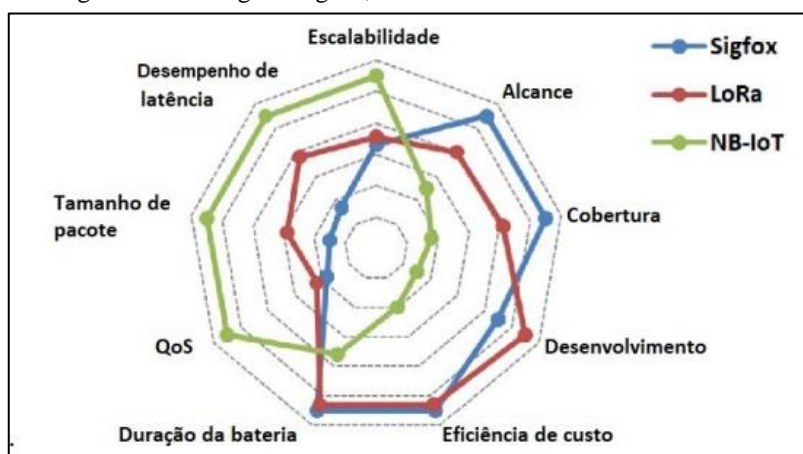
faz com que a rede LPWAN cubra uma grande variedade de aplicações, porém de acordo com Bardyn et al. (2016) há algumas características essenciais para ser observadas que irão definir o projeto de uma rede LPWAN:

- Baixo consumo de energia elétrica;
- O custo deve ser reduzido através de dispositivos mais baratos, com fácil instalação na rede e redução de manutenções. Além da complexidade de *hardware* e *software* que devem se manter simples;
- O dispositivo deve apenas enviar ou receber dados de outro dispositivo somente quando for necessário, evitando assim o gasto excessivo de energia;
- A infraestrutura de rede deve ser simples de estabelecer em escala nacional, com forte cobertura e capacidade de portabilidade entre países;
- A transferência de dados entre o usuário final e o dispositivo devem ser completamente seguros e confiáveis;
- Em grande parte das aplicações o dispositivo necessita ser facilmente localizado com baixo consumo de energia;
- A modulação deve ser robusta, para suportar um suposto enfraquecimento de sinal;
- Do ponto de vista da aplicação, os dispositivos fornecerão dados que serão usados para construir uma variedade de serviços.

Boa parte das aplicações *Machine-to-Machine* (M2M) requerem taxa de transmissão de dados relativamente baixa e a partir disso, foram desenvolvidas tecnologias que priorizam menor consumo e maior alcance.

As principais LPWANs existentes no mercado são: LoRaWAN, Sigfox e NB-IoT. Sigfox, da empresa homônima, segue o modelo *Software as a Service* (SaS), com um custo mensal para uso da rede. NB-IoT é integrada à rede *Long Term Evolution* (LTE). Portanto, sua operação utiliza banda licenciada e depende de um contrato com operadoras de telefonia. Cada uma das tecnologias tem suas características em comum e específicas. A figura 11 demonstra um comparativo entre as 3 principais LPWAN's.

Figura 11: Vantagens Sigfox, Lora e NB-IoT na internet das coisas



Fonte: Adaptado de Mekki et.al (2019)

Comparando LoraWan e Sigfox elas se encontram na categoria de padrões que fazem uso da faixa de frequências não licenciadas *Internet Safety Management* (ISM). Esta escolha traz uma vantagem que é a diminuição dos custos de desenvolvimento e de operação de ambos sistemas (DING et al., 2020). Quanto à modulação, as tecnologias empregam algumas diferenças no uso de banda. Sigfox trabalha com modulação GFSK (*Gaussian Frequency Shift Keying*)/DBPSK (*Differential Binary Phase Shift Keying*) em *Ultra Narrow Band* (UNB) com canais de 100Hz, o LoRa, CSS em *Ultra Wide Band* (UWB) com canais de 125kHz e 500kHz. Naturalmente, essa diferença confere ganhos e perdas a cada uma das técnicas (MEKKI et al., 2019).

O Sigfox apresenta vantagens a respeito do alcance e da cobertura, variando de 10km (urbano) a 40km (rural), enquanto o LoRa atinge de 5km (urbano) a 20 km (rural) (MEKKI et al., 2019). Neste cenário, o LoRa se sai melhor em comparação com as tecnologias UNB, podendo suportar velocidades de até 40km/h, enquanto o Sigfox perde estabilidade com velocidades por volta de 5 a 10 km/h (DMITRIEVICH, 2019).

O Sigfox possui uma estratégia voltada para *software* e serviço. Desta forma, apesar de trabalhar na banda ISM, livre de custos, o Sigfox requer a contratação de um serviço anual, por dispositivo, para a utilização da rede. O LoRaWAN, por sua vez, oferece uma estrutura de rede mais flexível pois as especificações de protocolo e o *software* para *gateways* e nós são disponibilizados gratuitamente pelas próprias Semtech e LoRa Alliance. Isso dá ao usuário a opção de montar uma rede totalmente privada, sem custos de serviço, desenvolvida a partir da documentação ou das diversas opções comerciais disponíveis (SIMONI, 2021).

Diferentemente do LoRaWAN e do Sigfox, o padrão NB-IoT é utilizado em faixa licenciada, por fazer parte do padrão LTE. Por apresentar uma banda estreita, o NB-IoT permite a cobertura de construções mais densa e uma maior penetração nas mesmas. Por isso, na

construção, ele é mais indicado do que outras tecnologias. Além disso, a baixa taxa de dados permite que as redes tenham uma maior cobertura em relação às redes LTE (DING et al., 2020).

O LoRaWAN e o NB-IoT vêm ganhando popularidade nos últimos anos, mas o LoRaWAN se destaca quando se trata de custo de implementação, capacidade de rede e consumo de energia. Enquanto isso, o NB-IoT tem melhores métricas de qualidade de serviço (QoS), menor latência e taxas de dados mais altas (MEKKI et al., 2019).

A bateria dos dispositivos LoRaWAN pode durar mais do que a maioria dos outros dispositivos, porque é possível entrar em modo de baixo consumo de energia em qualquer momento por utilizar um método de comunicação assíncrono baseado no protocolo ALOHA. Já o Padrão NB-IoT necessita de uma sincronização regular, além de utilizar de técnicas OFDM (*Orthogonal Frequency Division Multiplexing*) e FDMA (*Frequency Division Multiple Access*). Dessa forma, a tecnologia NB-IoT atinge maior latência porém maior consumo energético (DING et al., 2020).

Um fator a se levar em conta, comparando NB-IoT com LoraWan, é pelo fato da NB-IoT ser limitado a se comunicar com estações LTE que acaba sendo não aplicável para áreas rurais ou suburbanas que não possuam essa cobertura dependendo da disponibilidade do serviço de operadoras de telefonia. O que faz com que o NB-IoT não seja útil neste projeto, pois áreas dos rios tem limitado acesso à rede de telefonia. Por outro lado, a rede LoraWan, além de utilizar a banda ISM, possui *gateways* de baixo custo, facilitando a implementação em qualquer local. Logo podemos concluir que o NB-IoT se destina a aplicações que precisam de uma baixa latência e alta taxa de dados, com maior QoS, sacrificando um pouco de consumo energético que ainda é baixo e classificado como uma LPWAN.

Sendo assim, por se tratar de uma empresa com foco na pesquisa e desenvolvimento de projetos, a implementação de uma rede LoRaWAN apresenta uma oportunidade maior para o futuro, por oferecer mais flexibilidade, possibilidade de expansão do número de dispositivos sem custos e implantação em áreas rurais e suburbanas. A seguir veremos mais sobre a tecnologia Lora, seu protocolo LoraWan e suas especificações técnicas.

1.4 LORA E LORAWAN

LoRa é uma tecnologia de modulação de Radio Frequência para redes de longas áreas e de baixa potência (LPWANs). O nome, LoRa, é uma referência ao transporte de dados em longo alcance (*Long Range*) que essa tecnologia permite. Criada pela Semtech para padronizar as LPWANs, a tecnologia LoRa fornece comunicações de longo alcance de até 5 quilômetros em áreas urbanas, e até 15 quilômetros ou mais em áreas rurais (linha de visada).

Uma característica fundamental das soluções baseadas em LoRa são os requisitos de energia ultra-baixo, que permitem a criação de dispositivos em que sua bateria que podem durar até 10 anos (SEMTECH, 2022). A figura abaixo destaca algumas vantagens importantes da implantação de uma rede LoRaWAN.

Figura 12: Estrutura de uma rede LoRaWAN



Fonte: Adaptado de (SEMTECH, 2022).

Com relação ao alcance, um único *gateway* baseado em LoRa pode receber e transmitir sinais a uma distância de mais de 15 quilômetros em áreas rurais. Mesmo em ambientes urbanos densos, as mensagens são capazes de viajar até cinco quilômetros, dependendo de quão longe os dispositivos finais (*end nodes*) estão localizados.

No que diz respeito à duração da bateria, a energia necessária para transmitir um pacote de dados é baixa, visto que os pacotes de dados são muito pequenos e só são transmitidos algumas vezes por dia. Além disso, quando os *end nodes* estão em *stand-by*, o consumo de energia é medido em miliwatts (mW), permitindo que a bateria de um dispositivo dure muitos anos.

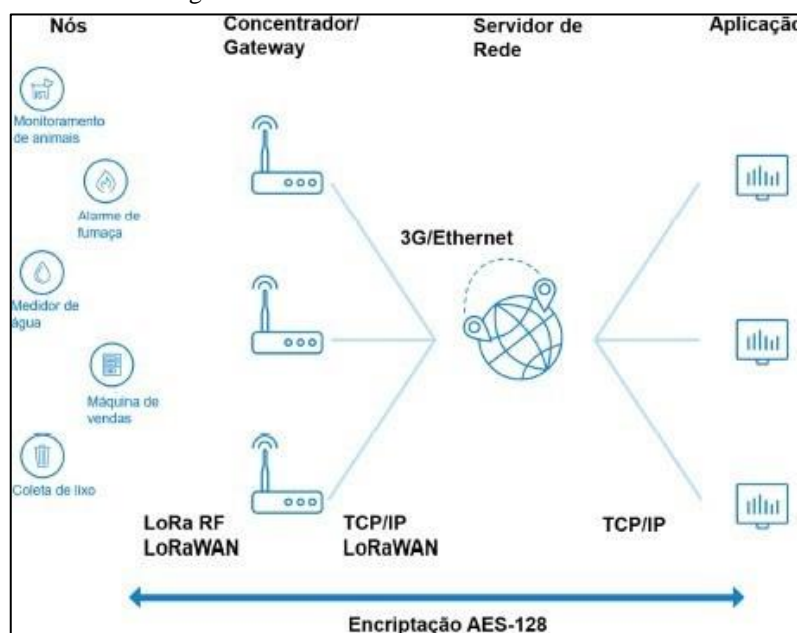
Quando se trata de capacidade, uma rede LoRaWAN pode suportar milhões de mensagens. No entanto, o número de mensagens suportadas em qualquer implementação depende do número de *gateways* instalados. Um único *gateway* de oito canais pode suportar algumas centenas de milhares de mensagens ao longo de um período de 24 horas. Se cada dispositivo final enviar 10 mensagens por dia, esse *gateway* pode suportar cerca de 10.000 dispositivos. Se a rede incluir 10 desses *gateways*, a rede pode suportar cerca de 100.000

dispositivos e um milhão de mensagens. Se for necessário mais capacidade, é só acrescentar mais gateways à rede.

A LoRa Alliance recomenda que seja utilizada a faixa de frequência entre 902 a 928 MHz na América Latina. Porém, pelo Ato número 14448 (ANATEL, 2017), publicado em quatro de dezembro de 2017, a faixa entre 907,5 e 915 MHz possui utilização proibida. Sendo assim, Souza e Rabello (2017) alertam para que a configuração seja corretamente feita para o perfeito funcionamento da rede.

Segundo Kageyama (2019), comumente as redes LoRa utilizam-se de uma topologia estrela similar à tecnologia celular. Para o funcionamento da rede são necessários basicamente 3 tipos de dispositivos, sendo eles: o nó final, normalmente um sensor, o *gateway* e o servidor. O sensor tem a finalidade de coletar os dados, podendo ser temperatura, fumaça, umidade e inúmeros outros tipos, além da possibilidade de exercerem a função de atuadores. Com estes dados coletados, o nó final cria um pacote e o envia *ao gateway*. A arquitetura de uma rede LoRa com protocolo LoRaWAN está apresentada na Figura 13.

Figura 13: Estrutura de uma rede LoRaWAN



Fonte: (NETWORK, 2022)

É importante destacar que a camada física LoRa é completamente independente das camadas superiores do protocolo LoRaWAN, podendo ser utilizada em aplicações que não sejam compatíveis com outras tecnologias de rede, transporte e aplicação.

1.4.1 Módulo ESP32 WiFi Lora

Neste trabalho foi utilizado o módulo microcontrolado WiFi LoRa 32, figura 14, que se trata de uma placa de desenvolvimento IoT projetada e produzida pela Heltec Automation. É um produto altamente integrado baseado em ESP32 + SX127x, possui funções Wi-Fi, Bluetooth, LoRa, sistema de gerenciamento de bateria Li-Po, e o OLED de 0,96" também está incluído. É a melhor escolha para cidades inteligentes, casas inteligentes e projetistas de soluções de IoT (Heltec, 2022).

Figura 14: Módulo Wifi LoRa 32



Fonte: (Heltec, 2022)

A tecnologia LoRa revolucionou a *Internet of Things* (IoT), ao permitir que os dispositivos tenham uma vasta solução de aplicações. Assim, ela transmite vários pacotes com informações importantes. Dentre os principais benefícios citados por LOGPYX (2022) são:

- Baixo custo de implementação e operação;
- Baixo consumo de potência;
- Longo Alcance;
- Mobilidade;
- Padronização da faixa de frequência.

Na figura 15 pode-se ver com mais detalhes o diagrama de pinos do ESP 32 Lora.

O teorema define o valor quantitativo do limite da taxa de transmissão de dados, ou capacidade do canal, com base na relação sinal-ruído e largura de banda, para se alcançar uma transmissão livre de erros em um canal com ruído aditivo branco e gaussiano (SIMONI, 2021).

$$C = BW \times \log_2 \left(1 + \frac{S}{N} \right) \quad (1)$$

Onde:

C = Capacidade [bits/s]

BW = Largura de banda [Hz]

S = Potência média do sinal [W] N

N = Potência média do ruído [W]

Isolando a Equação 1 em termos do logaritmo natural e aproximando por série de Taylor, dado que o sinal *spread spectrum* se encontra normalmente abaixo do nível de ruído, chega-se na Equação 2 (SEMTECH, 2022).

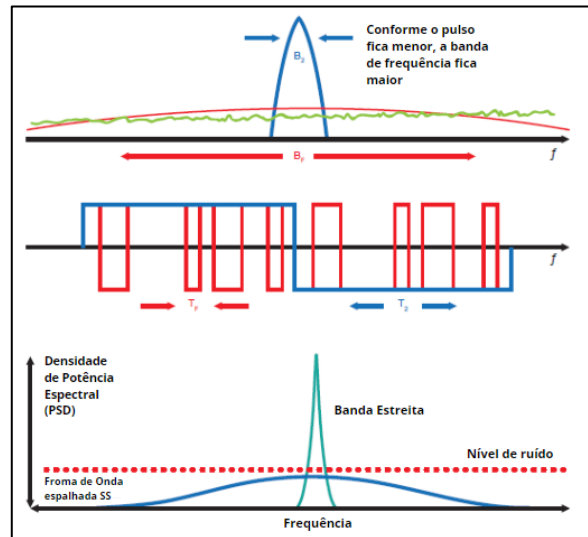
$$\frac{N}{S} \approx \frac{B}{W} \quad (2)$$

Com base na baixa relação sinal-ruído e largura de banda limitada disponível para transmissão, pode-se concluir que os ganhos de robustez vêm em desfavor da transmissão de informações, tornando este tipo de modulação mais adequado para aplicações que não requerem um alto tráfego de dados.

Além disso, LoRa usa *orthogonal spreading factor*. Isso permite que a rede preserve a vida útil da bateria dos nós finais conectados, fazendo otimizações adaptativas dos níveis de energia e taxas de dados de um nó final individual. Por exemplo, um dispositivo final localizado perto de um *gateway* deve transmitir dados em um baixo SF, já que muito pouca taxa de dados são necessários. No entanto, um dispositivo final localizado a vários quilômetros de um *gateway* precisará transmitir com um SF muito maior. Esse maior SF proporciona maior ganho de processamento e maior sensibilidade ao acolhimento, embora a taxa de dados seja, necessariamente, menor (SEMTECH, 2022).

Em um sistema DSSS (*Dynamic Sequence Spread Spectrum*, espectro de propagação de sequência direta) ou *direct sequence*, a fase portadora do sinal do transmissor muda de acordo com uma sequência de código, conforme mostrado na figura 16.

Figura 16: Modulação DSSS (Dynamic Sequence Spread Spectrum)



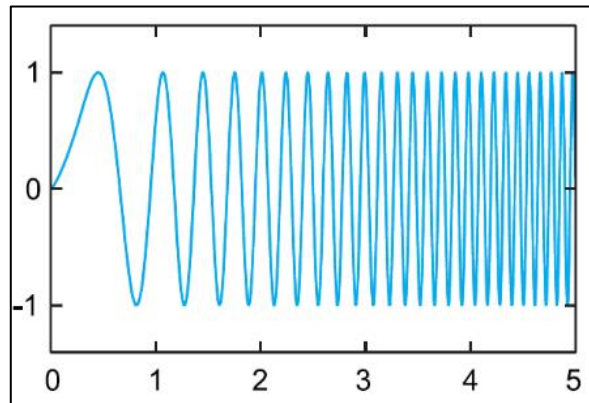
Fonte: Adaptado de (SEMTECH, 2022).

Ao multiplicar o sinal de dados com um padrão de bit pré-definido a uma taxa muito maior, também conhecido como código de disseminação (ou sequência de chip), um sinal é criado com componentes de frequência mais altos do que o sinal de dados original. Isso significa que a largura de banda do sinal está espalhada além da largura de banda do sinal original. Na terminologia RF, os bits da sequência de código são chamados chips (para distinguir entre os bits mais longos, não codificados, do sinal de dados original) (SIMONI, 2021).

Pode surgir um questionamento do porquê não transmitir o sinal de dados original em vez de passar por essa multiplicação de sequência de código. A resposta é que passar por essa multiplicação de sequência de código deixa o canal de RF mais alto, para que possamos transmitir por um intervalo mais longo.

Uma das desvantagens de um sistema DSSS é o fato de que ele requer um *clock* de referência altamente preciso e caro. A tecnologia Lora CSS da Semtech oferece uma alternativa DSSS de baixo custo e de baixa potência, mas robusta, que não requer um *clock* de referência altamente preciso. Na modulação lora, a disseminação do espectro do sinal é alcançada através da geração de um sinal de chirp que varia continuamente em frequência, como é observado na Figura 17.

Figura 17: Ilustração do Espectro de Propagação de LoRa Chirp



Fonte: (SEMTECH, 2022).

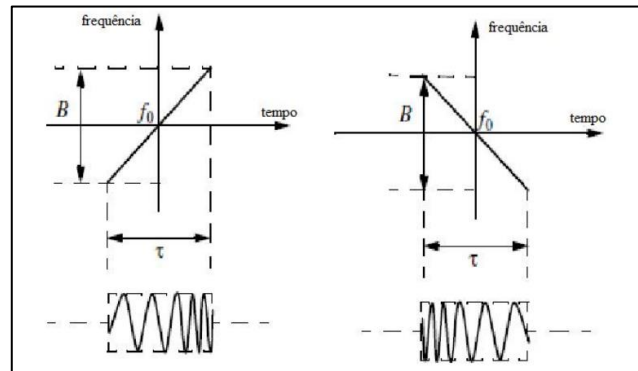
Uma vantagem deste método é que os deslocamentos de tempo e frequência entre transmissor e receptor são equivalentes, reduzindo consideravelmente a complexidade do *design* do receptor. A largura de banda de frequência deste chirp é equivalente à largura de banda espectral do sinal. O sinal de dados que transporta os dados de um dispositivo final para um *gateway* é transmitida a uma taxa de dados mais alta e modulado no sinal do portador chirp. A modulação LoRa também inclui um esquema de correção de erro variável que melhora a robustez do sinal transmitido. Para cada quatro bits de informação enviados, um quinto bit de informações de paridade é enviado (SEMTECH, 2022).

LoRa é puramente uma implementação de camada física (PHY), ou "bits", conforme definido pelo modelo de rede de sete camadas da OSI. Em vez de cabeamento, o ar é usado como um meio para transportar ondas de rádio LoRa de um transmissor RF em um dispositivo IoT para um receptor RF em um *gateway*, e vice-versa.

1.4.3 Sinais e parâmetros Lora

O sinal LoRa é uma onda senoidal modulada em frequência. Este sinal apresenta uma variação ao longo do tempo que pode ser crescente ou decrescente, dando origem a um *upchirp* ou *downchirp* conforme demonstrado na figura 18 (SIMONI, 2021).

Figura 18: Sinais upchirp e downchirp



Fonte: (SIMONI, 2021)

O sinal LoRa é enviado pelo ar em uma banda que se estende por toda a largura da frequência da portadora. A etapa na qual a frequência varia é definida pelo fator de espalhamento (SF). Quanto maior o SF, menor essa variação. Um símbolo é descrito por duas etapas, ou chips com duração de $1/BW$ segundos (SEMTECH, 2022). Cada símbolo é espalhado por um código com comprimento de 2^{SF} chips. A figura 19 demonstra os possíveis valores do SF e os tamanhos dos CHIP' s.

Figura 19: Fatores de Espelhamento Lora

Spreading Factor (For UL at 125 KHz)	Bit Rate	Range (Depends on Terrain)	Time on Air for an 11-byte payload
SF10	980 bps	8 km	371 ms
SF9	1760 bps	6 km	185 ms
SF8	3125 bps	4 km	103 ms
SF7	5470 bps	2 km	61 ms

Fonte: (SEMTECH, 2022)

O Tempo no Ar pode ser considerado como tempo de duração do pacote. Em geral o *Time on Air* (TOA) sempre aumentará ao mesmo tempo que o SF aumentar, pois o pacote demorará mais para chegar ao destino.

Cada símbolo LoRa pode ser descrito pela Equação 3, na qual cada chip é representado por um índice $n \in \{0,1,2,\dots,2^{SF} - 1\}$ (SEMTECH, 2022).

$$S_k = \sqrt{\frac{E_s}{2^{SF}}} \exp[j2\pi * (k + n) \bmod 2^{SF} * \frac{1}{2^{SF}} * n] \quad (3)$$

sk = Símbolo

E_s = Energia de símbolo

SF = Fator de espalhamento

k = Valor do símbolo

n = Índice do chip

2 MATERIAIS E MÉTODOS

2.1 MÉTODO PROPOSTO

Este estudo trata-se de uma revisão da literatura com o objetivo de obter um protótipo de envio de dados de posição via Radio Lora. As revisões são publicações amplas com a função de discutir o desenvolvimento de um assunto sob pontos de vista diferentes. Esse tipo de estudo constitui basicamente da análise da literatura publicada em artigos científicos, livros, revistas impressas ou eletrônicas na interpretação e análise crítica do autor, com o objetivo de permitir ao leitor uma atualização do seu conhecimento sobre um determinado tema.

Para essa revisão, foi realizada uma busca por artigos, livros, dissertações e teses nas bases de dados Google Acadêmico, *Institute of Electrical and Electronics Engineers* (IEEE) e Periódico Capes. Além da busca nas bases de dados, também foram realizadas pesquisas em sites, jornais e revista. As palavras-chave utilizadas na busca foram: Localização de embarcações, Rádio Lora, LoraWan, Internet das Coisas.

Como critério de inclusão dos materiais literários neste estudo, definiu-se o período de publicação de 10 anos pela possibilidade de poder ser encontrado um maior número de artigos científicos sobre o tema. Além disso, incluíram-se apenas artigos disponibilizados em português e inglês, dissertações, teses, livros, matérias de revistas eletrônicas e sites. Como critérios de exclusão, foram rejeitados os materiais literários que não tinham relação direta com o tema proposto pelo trabalho.

2.2 MATERIAIS UTILIZADOS

Para os testes e a confecção do protótipo foram utilizados as seguintes ferramentas, softwares e materiais:

- Arduino IDE 1.8.13 para Windows 10 de 64 bits;
- LaserCAD V7.98.19;
- CorelDRAW Versão 24.0.0.301;
- Ferro de solda de 127V/40 watts;
- Tubo de solda estanho de 1,0 mm;
- Alicates de corte;
- 1 *proto-board* de 830 pinos;
- 1 *proto-board* de 400 pinos;
- 2 Módulos ESP32 Wifi LoRa;

- 1 botão *push button* vermelho;
- 1 botão *push button* amarelo;
- 1 botão *push button* verde;
- 1 led vermelho;
- 1 resistor 220 Ω (1/8W);
- Barra 40 pinos macho;
- 1 placa de fenolite 5x9cm;;
- 1 módulo GPS modelo GY-GPS6MV2;
- 2 Cabos Micro-USB;
- 1 multímetro digital Minipa ET-2042D;
- Acrílico cristal de 3mm de espessura.

3 IMPLEMENTAÇÃO DO PROJETO

Neste capítulo é mostrado o desenvolvimento do projeto desde a pesquisa teórica em conjunto com as primeiras medições em laboratório, passando pela montagem do protótipo e testes de comunicação entre o módulo ESP 32 LoRa, até a criação do sistema e sua integração.

3.1 VALIDAÇÃO DOS MÓDULOS

A validação dos módulos foi feita independentemente como uma etapa antecessora à montagem do protótipo. O objetivo desses testes, além da validação do funcionamento, é estudar e compreender as bibliotecas de cada um, além de fazer a verificação das pinagens e as conexões entre elas para posteriormente ser feita a integração entre os módulos.

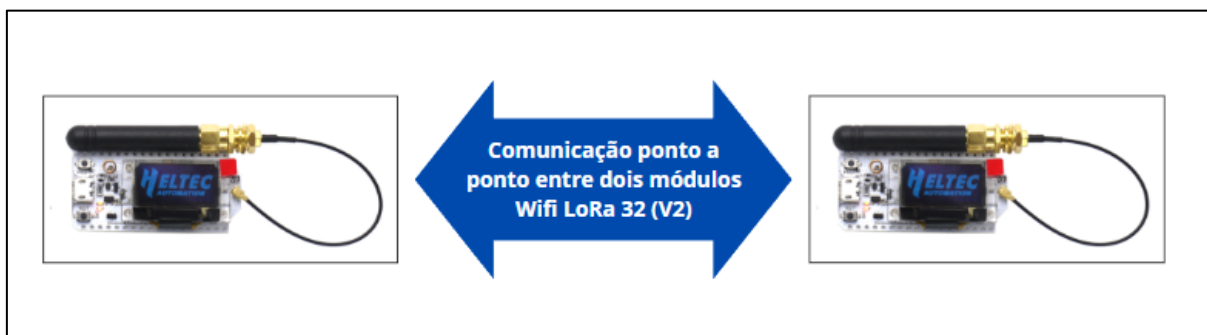
3.1.1 Validação do rádio LoRa

Esta será a primeira etapa do projeto, onde será validado a comunicação ponto a ponto via Rádio Lora. Esta etapa tem como objetivo entender a utilização dos parâmetros da modulação LoRa antes de implementar no projeto de envio das coordenadas de localização. Para esta etapa foram utilizados os seguintes materiais:

- 2 módulos Wifi LoRa 32 (V2);
- 1 fonte micro-usb de 5V/2^a;
- 1 cabo micro-usb (Para programação de ambos os módulos),

Na figura 20, podemos ver a comunicação entre dois módulos na topologia ponto a ponto. Nesta figura há dois módulos se comunicando via rádio-Lora, trocando informações de forma síncrona podendo ser duplex ou full-duplex.

Figura 20: Comunicação entre 2 módulos Wifi LoRa 32

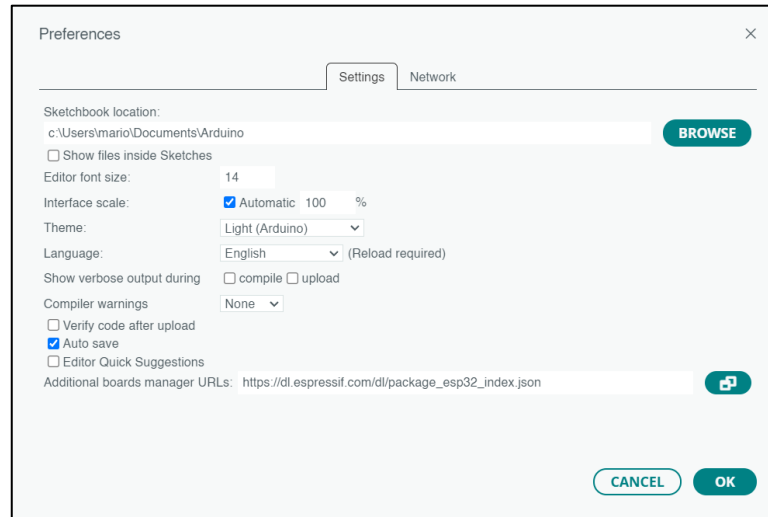


Fonte: Autoria Própria

Para que seja utilizado o módulo, primeiramente foi feita a instalação das bibliotecas básicas para o Esp32. Para realizar a instalação bastou seguir os seguintes passos: Na plataforma arduino IDE vá em *File>Preferences>* Em *Additional boards manager URLs* inserir:

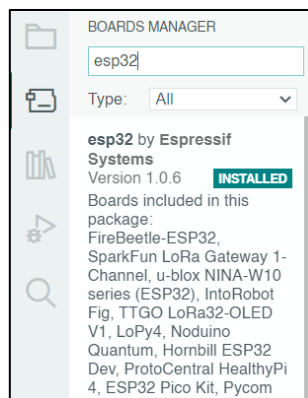
https://dl.espressif.com/dl/package_esp32_index.json. Conforme figura 21 e clicar em ok. Usar o atalho CTRL + SHIFT + B para ir até “*Boards Manager*”. Na janela do gerenciador de placas escrever “esp32”. Clicar na opção “*Esp32 by Espressif Systems*” e depois no botão *install* aguardar o download e instalação da biblioteca (Figura 22).

Figura 21: Caminho para instalação da biblioteca ESP32



Fonte: Autoria Própria

Figura 22: Gerenciador de Placas na Arduino IDE

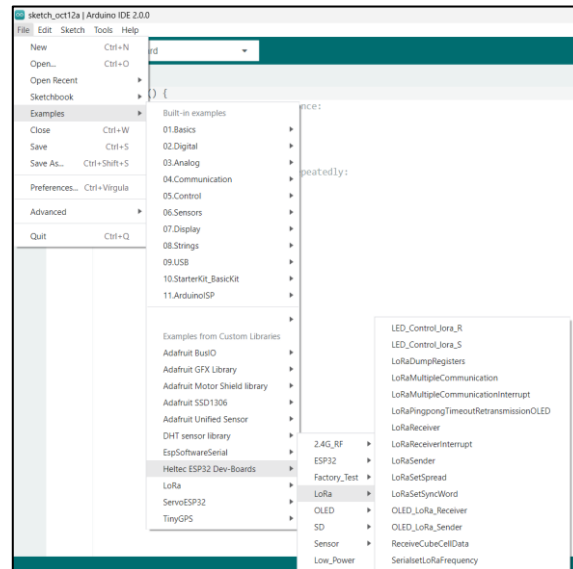


Fonte: Autoria Própria

Após o procedimento acima, foi feita a instalação das bibliotecas de LoRa e display OLED. O procedimento de instalação da está disponível no tutorial disponibilizado pela Heltec disponível em: https://heltec-automation-docs.readthedocs.io/en/latest/esp32/quick_start.html.

Finalizado os procedimentos de instalações de bibliotecas, o módulo esteve pronto para ser conectado ao computador via porta serial e ser programado. Foram verificados os códigos exemplos disponibilizados pela Heltec conforme mostra a figura 23.

Figura 23 Biblioteca instalada do módulo WiFi LoRa 32(V2)



Fonte: Autoria própria

Foram utilizados os códigos: “OLED_LoRa_Sender” e “OLED_LoRa_Receiver” para testar o envio de dados utilizando a modulação LoRa. Neste código por padrão a modulação está configurada em 868Mhz, foi feita a alteração para 915Mhz tanto no código “*sender*” quanto no “*receiver*” conforme observa-se na figura 24.

Figura 24: Configuração de frequência de Modulação na IDE

```

21 #include "heltec.h"
22 #include "images.h"
23
24 #define BAND 915E6 // alterado de 868Mhz para 915Mhz
25

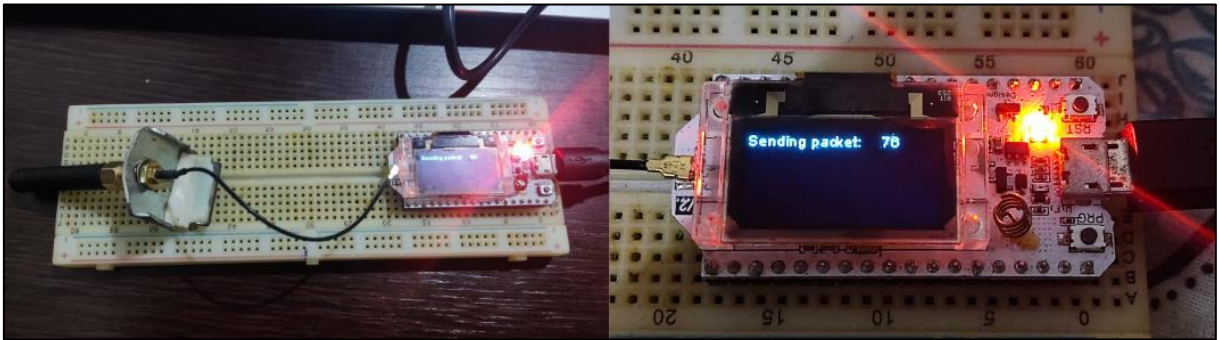
```

Fonte: Autoria própria

No módulo transmissor (Figura 25) foi compilado o código *OLED_LoRa_Sender*, que envia um pacote com a mensagem “*Hello World*” com um incremento crescente começando em 0 para fazer a contagem do envio dos pacotes.

No módulo receptor (Figura 26) foi compilado o código *OLED_LoRa_Receiver*, que recebe os pacotes enviados pelo Módulo 1 com a informação do nível do sinal recebido (RSSI) com a contagem do recebimento dos pacotes e suas respectivas quantidade de bytes recebidos.

Figura 25: Módulo transmissor enviando pacotes via LoRa



Fonte: Autoria própria

Figura 26: Módulo receptor recebendo pacotes via LoRa

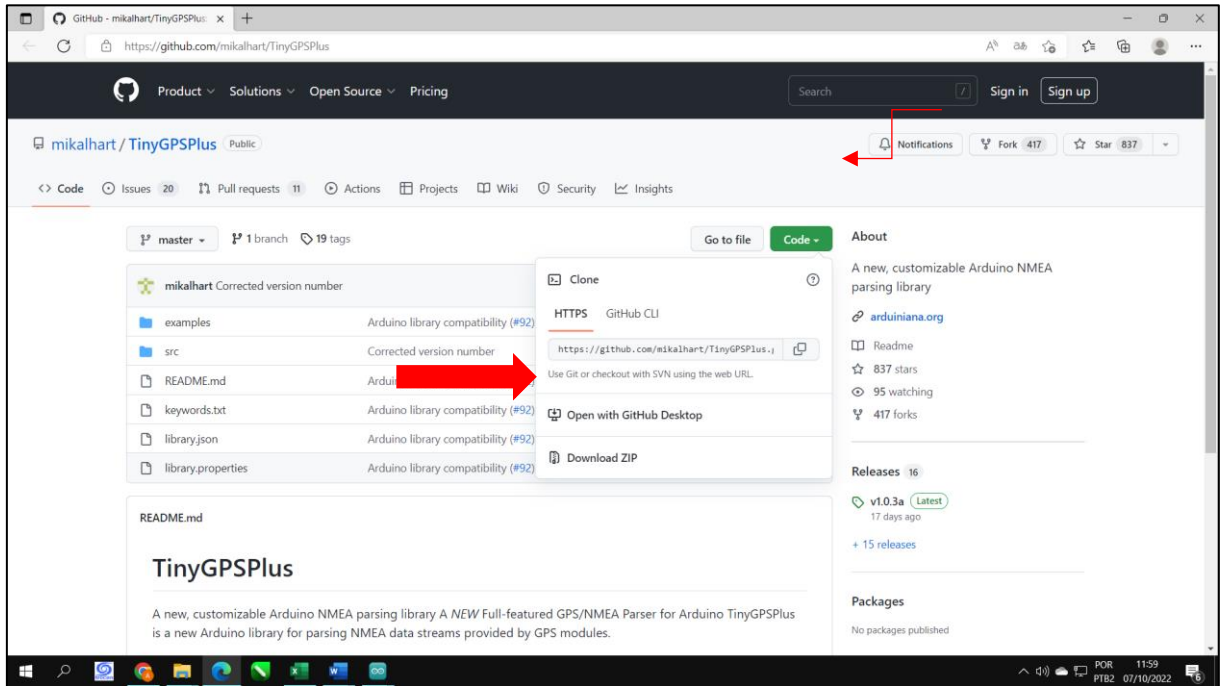


Fonte: Autoria própria

3.1.2 Validação do módulo GPS

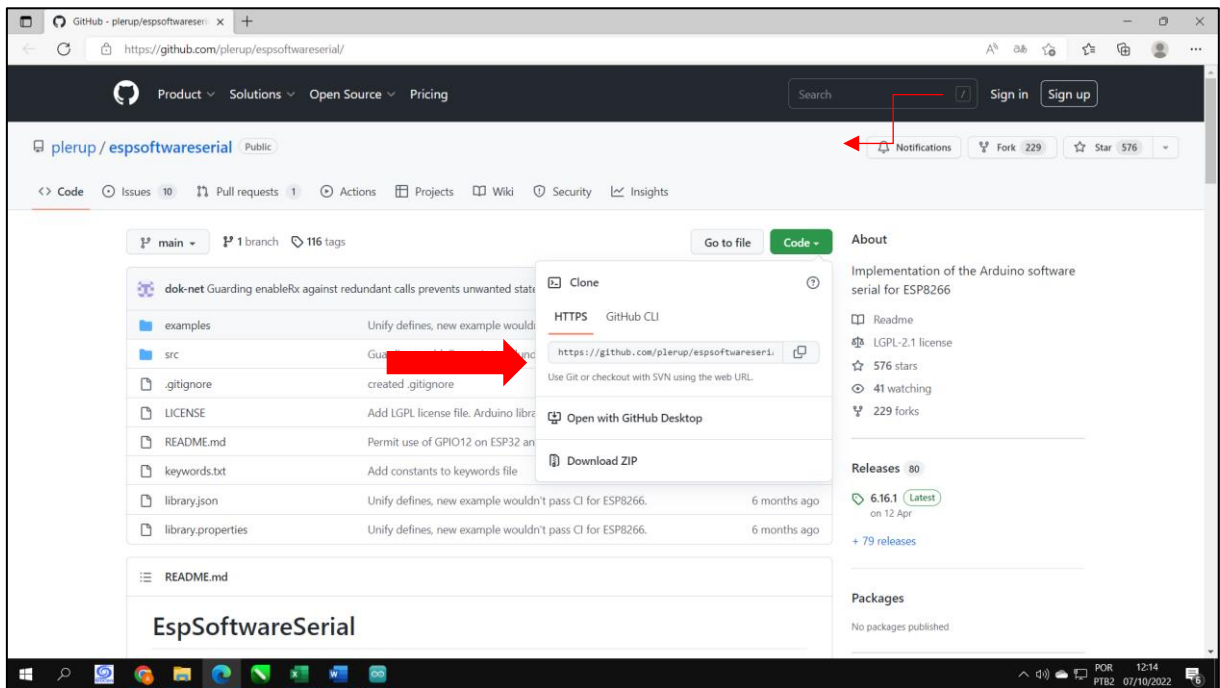
Nesta etapa do projeto, o módulo GPS de modelo GY-GPS6MV2 foi testado para validar o funcionamento junto ao módulo ESP32 Lora V2. Inicialmente foi feita uma busca nas bibliotecas do repositório github, onde foi utilizada a área *Clone* para *download* das bibliotecas TinyGPSPlus e EspSoftwareSerial conforme mostra a figura 27 e 28.

Figura 27: Repositório da Biblioteca TinyGPSPlus



Fonte: Autoria Própria.

Figura 28: Repositório da Biblioteca EspSoftwareSerial



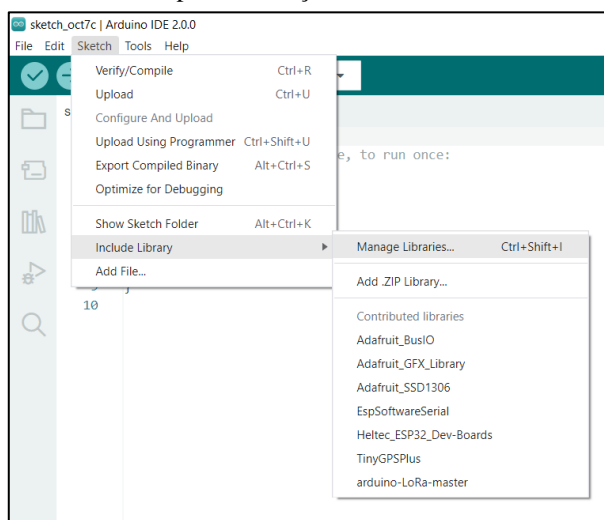
Fonte: Autoria Própria.

A biblioteca foi desenvolvida por Peter Lerup e está disponível no *link* <https://github.com/plerup/espsoftwareserial/> no repositório GitHub.

A biblioteca TinyGPSPlus foi desenvolvida por Mikal Hart e está disponível no [link https://github.com/mikalhart/TinyGPSPlus](https://github.com/mikalhart/TinyGPSPlus) no repositório GitHub.

Após o download dos arquivos da biblioteca, foi aberto a IDE Arduino para executar a instalação da biblioteca do módulo do receptor GPS. Para realizar a instalação bastou seguir o diretório *Sketch>Include Library>Add .ZIP Library* e selecionar o diretório onde foi salvo o arquivo baixado do diretório GitHub. A Figura 29 mostra o caminho para realizar a instalação destas bibliotecas.

Figura 29: Caminho para instalação de bibliotecas na IDE Arduino



Fonte: Autoria Própria.

O próximo passo foi a montagem física do módulo GPS junto a placa de desenvolvimento Wifi Lora Esp 32 V2. Para isso foi feita uma consulta no datasheet do módulo receptor GPS GY-GPS6MV2 onde se encontram os seguintes pinos: Vcc, Rx, Tx e GND. Então foi definido a conexão entre o módulo GPS e o módulo Esp32 Lora conforme mostra o Quadro 2:

Quadro 2: Conexões entre GPS e Lora

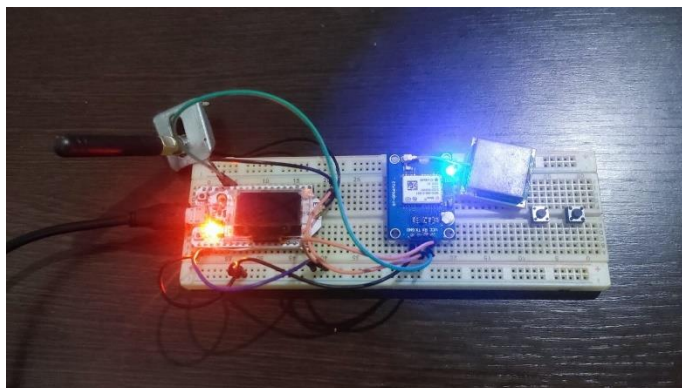
Módulo GPS GY-GPS6MV2	Módulo Wifi Lora 32
VCC	5V
Rx	36
Tx	37
GND	GND

Fonte: Autoria Própria.

Após a montagem do circuito conforme figura 30, e a instalação da biblioteca do módulo, foi feita a análise dos programas básicos da biblioteca do módulo receptor GPS com o intuito de extrair apenas as informações necessárias, como data, horário, latitude e longitude, sendo separados pelo sinal de ponto e vírgula entre cada parâmetro. Além disso, foram feitos

os ajustes de *baudrate* do GPS para 9600 e ajustado no programa os pinos 36 e 37 (Conforme Quadro 2) para ser o Rx e Tx respectivamente.

Figura 30: Circuito montado GPS + Wifi Lora 32



Fonte: Autoria Própria.

Antes de carregar o programa para o módulo foram testadas as conexões com o auxílio de um multímetro para certificar que havia continuidade entre as ligações e que estavam todas ligadas corretamente. Verificado a condição acima, o programa foi compilado e executado no microcontrolador ESP32 da placa Wifi Lora 32(V2). A figura 31 demonstra o resultado na saída do monitor serial da IDE do Arduino.

Figura 31: Dados extraídos do Módulo GPS GY-GPS6MV2

```
Message (Ctrl + Enter to send message to 'Heltec WiFi LoRa 32(V2)' on 'COM3')
-----
Localizacao: -3.071059,-59.942607 Data/Hora: 10/7/2022 17:34:44.00
Localizacao: -3.071059,-59.942607 Data/Hora: 10/7/2022 17:34:44.00
Localizacao: -3.071059,-59.942607 Data/Hora: 10/7/2022 17:34:44.00
Localizacao: -3.071059,-59.942607 Data/Hora: 10/7/2022 17:34:44.00
Localizacao: -3.071036,-59.942627 Data/Hora: 10/7/2022 17:34:52.00
Localizacao: -3.071036,-59.942627 Data/Hora: 10/7/2022 17:34:52.00
Localizacao: -3.071036,-59.942627 Data/Hora: 10/7/2022 17:34:52.00
Localizacao: -3.071036,-59.942627 Data/Hora: 10/7/2022 17:34:52.00
Localizacao: -3.071036,-59.942627 Data/Hora: 10/7/2022 17:34:52.00
Localizacao: -3.071036,-59.942627 Data/Hora: 10/7/2022 17:34:52.00
Localizacao: -3.071036,-59.942627 Data/Hora: 10/7/2022 17:34:52.00
Localizacao: -3.071036,-59.942627 Data/Hora: 10/7/2022 17:34:52.00
Localizacao: -3.071036,-59.942627 Data/Hora: 10/7/2022 17:34:52.00
Localizacao: -3.071051,-59.942620 Data/Hora: 10/7/2022 17:35:02.00
-----
Ln 45, Col 29 UTF-8 Heltec WiFi LoRa 32(V2) on COM3 4
```

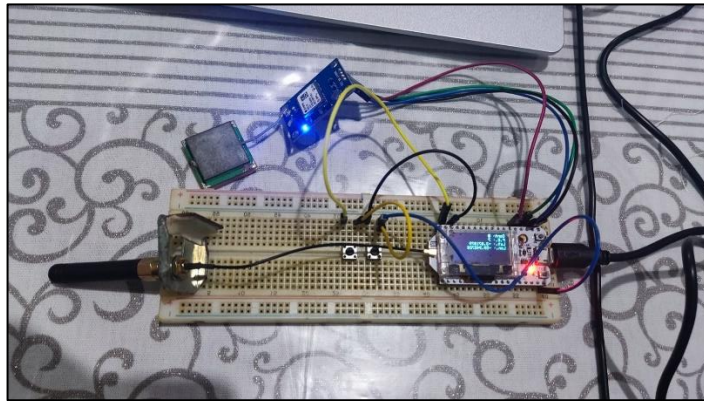
Fonte: Autoria Própria.

O programa foi compilado e carregado no microcontrolador ESP32 da placa Wifi LoRa 32(V2). Em seguida, o receptor GPS ficou aguardando alguns segundos, até começar a receber os pacotes NMEA contendo as informações de latitude, longitude, data e horário.

3.1.3 Validação da integração GPS + LoRa

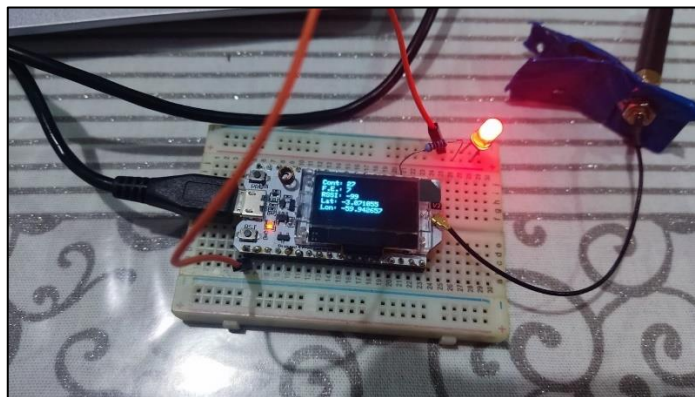
Após os testes individuais dos módulos, foi realizado a integração do módulo GPS com o rádio LoRa para envio das coordenadas. Para isso foi definido 2 *hardwares*: um transmissor e um receptor conforme podemos observar nas figuras 32 e 33 abaixo:

Figura 32: Teste do transmissor de coordenadas geográficas



Fonte: Autoria Própria.

Figura 33: Teste do receptor de coordenadas geográficas



Fonte: Autoria Própria.

Para a montagem do transmissor de coordenadas de localização foram utilizados os seguintes materiais:

- 1 Wifi Lora Esp 32;
- 2 botões *Push-button* com as funções “transmitir coordenadas” e “parar transmissão”;
- 1 GPS de modelo GY-GPS6MV2;;
- Protoboard de 830 pinos;
- Cabo micro-usb para programação.

Para a montagem do receptor de coordenadas de localização foram utilizados os seguintes materiais:

- 1 Wifi Lora Esp 32;

- 1 led vermelho;
- 1 resistor de 220 Ω .

As conexões físicas do GPS se mantiveram conforme descrito no quadro 2 da seção 3.1.2 deste capítulo. E o botão com a função “transmitir” e “parar transmissão” e “acréscimo do fator do espalhamento” foram definidas as portas conforme vemos no quadro 3 abaixo.

Quadro 3: Definições de GPIO para integração GPS-LoRa

Botão	Pino
Transmitir	12
Parar transmissão	13
Fator de Espalhamento	0

Fonte: Autoria Própria.

Vale ressaltar que o GPIO “0” utiliza o próprio botão do Módulo Wifi LoRa denominado “PRG Button”.

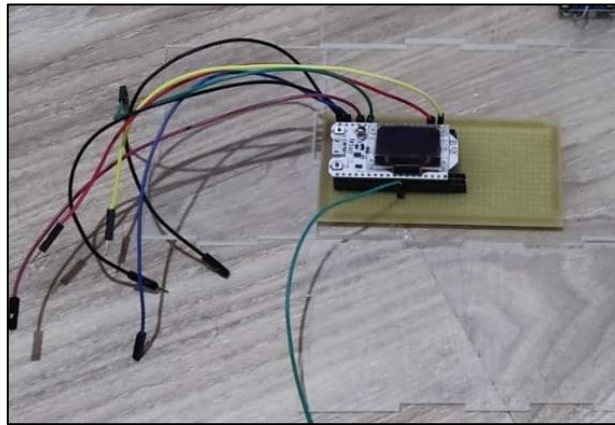
A porta utilizada para acender o led apenas quando receber dados das coordenadas de localização no receptor foi o GPIO número 12. Para mais informações do código consultar apêndice B – Código Receptor.

Ao final dos testes nos códigos do transmissor e receptor, esta etapa foi concluída com êxito. A próxima etapa será o desenvolvimento do protótipo.

3.2 DESENVOLVIMENTO DO PROTÓTIPO

Nesta etapa desenvolveu-se o protótipo final do transmissor. Primeiramente a barra de 40 pinos foi dividida ao meio para utilização de encaixe do módulo Wifi Lora na placa PCB perfurada. Depois foram soldados os fios que interligará o módulo GPS e os botões de “transmissão”, “parar transmissão” e “fator de espalhamento”, pode-se observar esta montagem na figura 44.

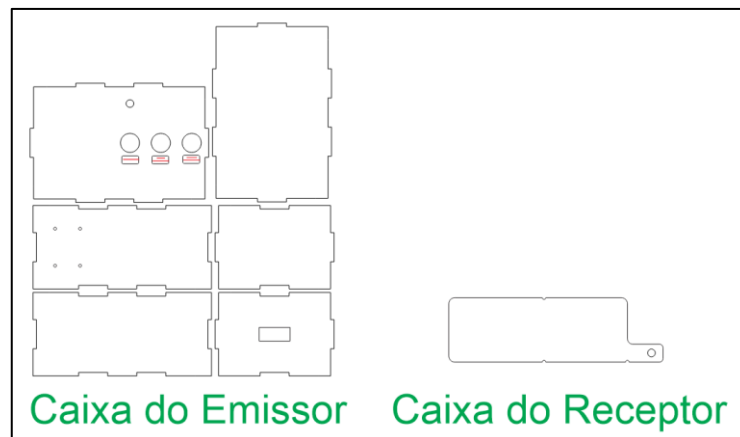
Figura 34: Placa do Hardware Transmissor



Fonte: Aatoria Própria.

Em seguida foi projetado o encaixe para montagem das caixas em material acrílico transparente de espessura de 3mm, tanto para o transmissor quanto para o receptor (ver figura 35) utilizando o *software* CorelDRAW. a caixa do transmissor possui as seguintes dimensões: 100mm x 150mm x 70mm (Comprimento x largura x altura), e a caixa do receptor com as seguintes dimensões: 80mm x 54mm x 70mm (Comprimento x largura x altura).

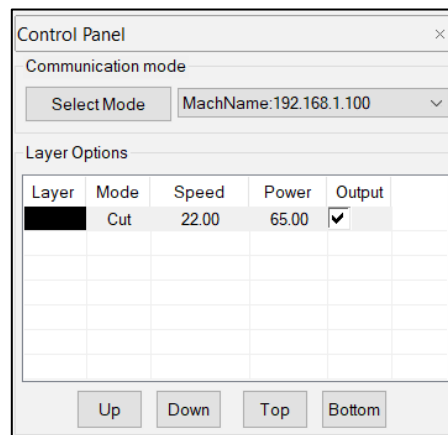
Figura 35: Desenho das caixas Transmissor e Receptor



Fonte: Aatoria Própria.

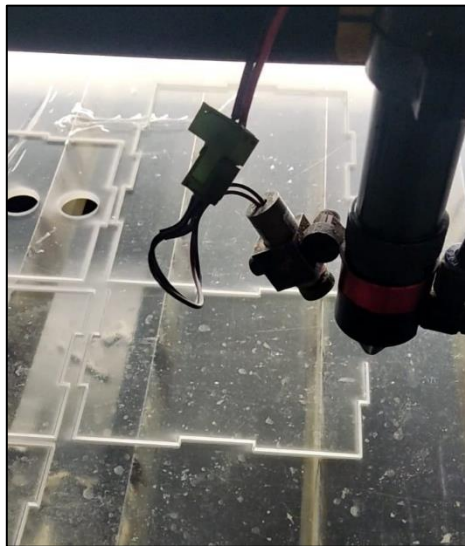
Com o desenho finalizado, a caixa foi exportada em formato .dxf e enviado ao programa LaserCad, *software* de máquina de corte de Controle Numérico Computadorizado (CNC) a laser, cujo parâmetros utilizados para corte estão especificados na figura 36. Após a conversão do desenho para a linguagem de programação em Código G, a máquina iniciou os cortes para a confecção das caixas. A figura 37 mostra a máquina realizando o corte das peças.

Figura 36: Parâmetros para corte de acrílico



Fonte: Autoria Própria.

Figura 37: Confeção em CNC a Laser



Fonte: Autoria Própria.

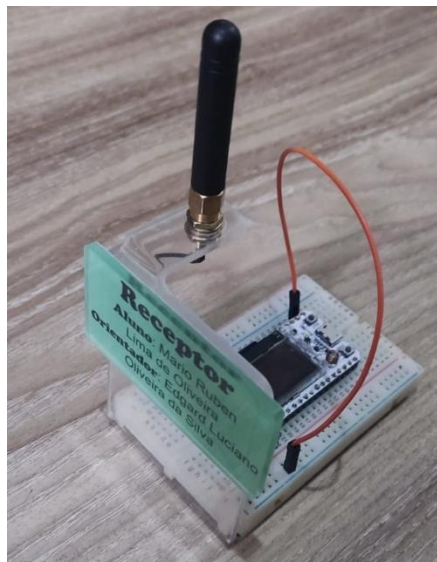
Com os cortes finalizados, foi efetuado a colagem das peças acrílicas e a fixação do módulo GPS, da antena e da placa PCB na caixa. Por fim foram feitas as soldagens dos cabos nos conectores dos botões *push-button*. Além disso, foram feitas impressões no acrílico utilizando impressora ultra-violeta para identificações do transmissor, do receptor e das identificações das funções dos botões. As figuras 38 e 39, demonstram respectivamente o protótipo final do transmissor e do receptor.

Figura 38: Protótipo final do transmissor



Fonte: Autoria Própria.

Figura 39: Protótipo final do receptor



Fonte: Autoria Própria.

Os protótipos finais do emissor e do receptor foram novamente testados o seu funcionamento na transmissão de dados de localização obtendo-se êxito em seu funcionamento.

4 RESULTADOS OBTIDOS

Neste capítulo serão mostrados os resultados obtidos após a implementação do projeto. Essa etapa de teste mostra a funcionalidade do protótipo para atingir o objetivo de implementar o envio das coordenadas de localização utilizando Rádio Lora.

4.1 CENÁRIO DE TESTE

Com o protótipo pronto, é preciso atestar o funcionamento do sistema com um teste de campo o mais próximo possível. Esse experimento consistiu em levar os dois protótipos: O protótipo transmissor e o protótipo receptor das coordenadas de GPS nas dependências da faculdade de tecnologia da Universidade Federal do Amazonas por possuir bastante vegetação similar a dos rios conforme mostra a Figura 40.

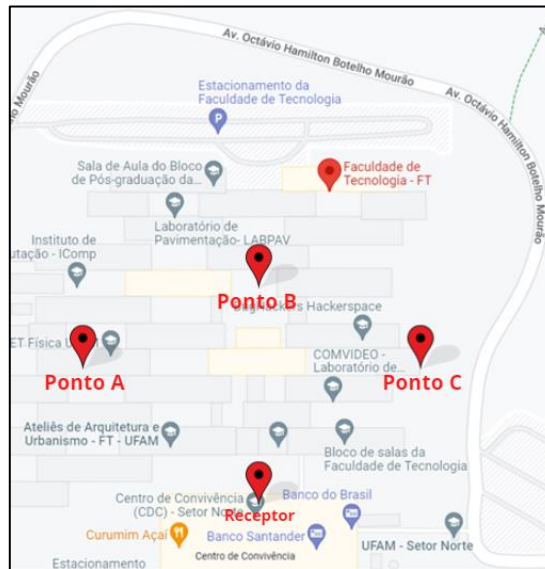
Figura 40: Local de Teste do protótipo



Fonte: Autoria Própria.

Neste cenário, o módulo receptor ficou fixo perto das proximidades do Centro de Convivência (CDC) – setor norte da Universidade Federal do Amazonas. Em seguida, a unidade móvel foi deslocada para os pontos especificados e foi transmitido 5 pacotes em cada ponto conforme mostra a figura 41.

Figura 41: Ponto de deslocamento da unidade móvel



Fonte: Autoria Própria.

Para desenvolver o experimento, foi necessário efetuar algumas preparações. Uma delas foi a configuração dos parâmetros do rádio. A transmissão foi configurada com a potência de transmissão em 20 dBm, a largura de banda de 125 kHz e o Code Rate (CR) de 5/5 com a palavra de sincronização em (0x55) tanto no transmissor quanto no receptor conforme figura 42.

Figura 42: Configuração dos parâmetros do Rádio LoRa

```
{ // setar os parâmetros do rádio
  LoRa.setTxPower(HIGH_GAIN_LORA);
  LoRa.setSignalBandwidth(125E3);
  LoRa.setCodingRate4(5);
  LoRa.setSyncWord(0x55);
```

Fonte: Autoria Própria.

O fator de espalhamento espectral (*Spreading Factor*) foi variado de 7 a 10 com o intuito de avaliar a influência desse parâmetro na transmissão dos dados, no alcance máximo atingido e na quantidade de pacotes perdidos. No quadro 3 podemos ver os resultados coletados:

Quadro 3: Nível de RSSI com SF=7

Ponto	Amostra	Frequência	SF (Fator de Espalhamento)	Bandwidth (BW)	RSSI (dBm)
Ponto A	1	915MHz	7	125kHz	-75
	2	915MHz	7	125kHz	-73
	3	915MHz	7	125kHz	-78
	4	915MHz	7	125kHz	-75
	5	915MHz	7	125kHz	-73
Ponto B	1	915MHz	7	125kHz	-76
	2	915MHz	7	125kHz	-75
	3	915MHz	7	125kHz	**
	4	915MHz	7	125kHz	-80
	5	915MHz	7	125kHz	-82
Ponto C	1	915MHz	7	125kHz	-107
	2	915MHz	7	125kHz	**
	3	915MHz	7	125kHz	**
	4	915MHz	7	125kHz	-100
	5	915MHz	7	125kHz	-98

Fonte: Autoria Própria.

É possível observar no quadro 3 que houve perdas de pacotes durante a transmissão das mensagens em alguns pontos definidos. Os níveis de RSSI obtidos no ponto A e ponto B foram os mais altos devido à proximidade com o receptor e a linha de visada quase sem obstrução.

Após a conclusão desse primeiro teste, foi alterado o valor do espalhamento espectral para SF=8. O quadro 4 apresenta o resultado desse teste.

Quadro 4: Nível de RSSI com SF=8

Ponto	Amostra	Frequência	SF (Fator de Espalhamento)	Bandwidth (BW)	RSSI (dBm)
Ponto A	1	915MHz	8	125kHz	**
	2	915MHz	8	125kHz	-75
	3	915MHz	8	125kHz	-80
	4	915MHz	8	125kHz	-78
	5	915MHz	8	125kHz	**
Ponto B	1	915MHz	8	125kHz	-80
	2	915MHz	8	125kHz	-73
	3	915MHz	8	125kHz	**
	4	915MHz	8	125kHz	-78
	5	915MHz	8	125kHz	-78
Ponto C	1	915MHz	8	125kHz	**
	2	915MHz	8	125kHz	**
	3	915MHz	8	125kHz	**
	4	915MHz	8	125kHz	-98
	5	915MHz	8	125kHz	-101

Fonte: Autoria Própria.

Nos respectivos quadros 5 e 6 são apresentados os resultados recebidos no módulo receptor com SF = 9 e SF= 10, respectivamente.

Quadro 5: Nível de RSSI com SF=9

Ponto	Amostra	Frequência	SF (Fator de Espalhamento)	Bandwidth (BW)	RSSI (dBm)
Ponto A	1	915MHz	9	125kHz	-87
	2	915MHz	9	125kHz	-80
	3	915MHz	9	125kHz	-84
	4	915MHz	9	125kHz	-85
	5	915MHz	9	125kHz	-82
Ponto B	1	915MHz	9	125kHz	-100
	2	915MHz	9	125kHz	-105
	3	915MHz	9	125kHz	-103
	4	915MHz	9	125kHz	-106
	5	915MHz	9	125kHz	-103
Ponto C	1	915MHz	9	125kHz	**
	2	915MHz	9	125kHz	**
	3	915MHz	9	125kHz	-113
	4	915MHz	9	125kHz	-109
	5	915MHz	9	125kHz	-110

Fonte: Autoria Própria.

Quadro 2: Nível de RSSI com SF=10

Ponto	Amostra	Frequência	SF (Fator de Espalhamento)	Bandwidth (BW)	RSSI (dBm)
Ponto A	1	915MHz	10	125kHz	-86
	2	915MHz	10	125kHz	-85
	3	915MHz	10	125kHz	-78
	4	915MHz	10	125kHz	-80
	5	915MHz	10	125kHz	-105
Ponto B	1	915MHz	10	125kHz	-104
	2	915MHz	10	125kHz	-105
	3	915MHz	10	125kHz	-103
	4	915MHz	10	125kHz	**
	5	915MHz	10	125kHz	-102
Ponto C	1	915MHz	10	125kHz	-98
	2	915MHz	10	125kHz	-97
	3	915MHz	10	125kHz	-98
	4	915MHz	10	125kHz	-99
	5	915MHz	10	125kHz	-98

Fonte: Autoria Própria.

Os resultados do Quadro 5 e Quadro 6 mostram que SF=9 e SF=10 apresentaram maior estabilidade no nível de recepção do sinal. Conforme mencionado na seção 1.4.3 o valor do *Spreading Factor* define quantos *chirp's* são enviados por segundo, logo aumentando o valor

de SF gera um tempo de recepção mais prolongado possibilitando receber o sinal mais longe e assim obter maior cobertura.

CONCLUSÃO

Este trabalho realizou uma revisão da literatura sobre o tema desenvolvimento de protótipo de aquisição de dados de localização de transportes fluviais na região amazônica utilizando rádio Lora. O projeto desenvolvido mostra que é possível implementar um sistema de monitoramento de embarcações, além de ser eficaz a transmissão de dados de localização para uma embarcação que esteja nas proximidades.

A implementação do projeto foi realizada com êxito, de acordo com as etapas descritas no capítulo de materiais e métodos. Com base nisso, foi possível realizar os testes de campo de comunicação entre o módulo transmissor e o módulo receptor.

Estes sistemas de localização funcionam com base nos conceitos de triangulação e tempo de chegada (TOA) para determinação das coordenadas de geolocalização. Por meio de do tempo de chegada da mensagem e sabendo-se a velocidade de propagação da onda de rádio, é possível calcular a distância entre o satélite e o dispositivo e com isso obter as coordenadas de localização que o módulo GPS GY-GPS6MV2 nos fornece.

Os resultados obtidos durante os testes no ambiente urbano mostram que o aumento do fator de espalhamento (*spreading factor*) e da taxa de codificação, bem como a diminuição da largura de banda, acarretam diminuição da taxa de dados. Com os testes, foi possível verificar também os limites de alcance da rede LoRa conforme o fator de espalhamento era alterado e também na influência que os obstáculos na linha de visada impactam a qualidade da transmissão do sinal.

Há alguns pontos que podem ser explorados para trabalhos futuros. É sugerido fazer testes de comunicação entre os módulos no modo de transmissão *Full-Duplex* onde o modo de transmissão se dá em sentido duplo ou bidirecional simultâneo, ou seja, os módulos serem capazes de transmitir e receber os dados das coordenadas de localização ao mesmo tempo. Outra sugestão é de realizar a implementação de *gateways* para transmissão de dados via internet com a implementação de um sistema supervisorio disponibilizando uma API com banco de dados em nuvem de maneira escalável e publicação do aplicativo.

Um ponto de atenção é que o LoRa não oferece (de forma nativa) a segurança dos dados trafegados. Ou seja, fica a cargo do projetista desenvolver um *software* que faça a encriptação dos dados para transmissão e desencriptação na recepção. Este acaba sendo um ponto de atenção relevante em projetos de internet das coisas, uma vez que brechas de segurança podem causar altos impactos negativos para um negócio. Nesse contexto sugere-se um estudo do uso

de tecnologia emergente utilizando a criptografia do blockchain no registro de dados como ferramenta de segurança para armazenar e autenticar dados trafegados pela rede.

REFERÊNCIAS

- ANATEL. Ato número 14448. 2017. Disponível em:
<<https://informacoes.anatel.gov.br/legislacao/atos-de-certificacao-de-produtos/2017/1139-ato-14448>>. Acesso em: 07 de agosto de 2022.
- BNC AMAZONAS. “**Anna Karoline**” navega no rio Amazonas como navio-fantasma, e assusta. 2020 Disponível em: < <https://bncamazonas.com.br/municipios/anna-karoline-amazonas-navio-fantasma/>> Acesso em: 10 de setembro de 2022.
- BERTOLETI, Pedro. **Projetos com ESP32 e Lora. [eBook]** - Editora NCB; 1ª edição (15 julho 2019).
- BARDYN, J. P. et al. IoT: **The era of LPWAN is starting now.** In: ESSCIRC Conference 2016: 42nd European Solid-State Circuits Conference.
- DAVID, Robert carvalho de Azevedo. **As dinâmicas do transporte fluvial de passageiros no Estado do Amazonas. 2010.** 121f. Dissertação (Mestrado em Geografia) - Universidade Federal do Amazonas, Manaus, 2010.
- DING, Jie; NEMATI, Mahyar; RANAWEERA, Chathurika; CHOI, Jinho. **Iot connectivity technologies and applications:a survey.** 2020.
- DMITRIEVICH, Ilin Alexander; DMITRIEVICH, Belogurov Bogdan. **Assessing the possibility of integration lpwan technology into tracking and monitoring systemsfor special vehicles.** International Siberian Conference on Control and Communications (SIBCON), 2019.
- ÉDILUS & RONALDO. **Microcontroladores** / – Ouro Preto : Instituto Federal de Educação, Ciência e Tecnologia de Minas Gerais ; Santa Maria : Universidade Federal de Santa Maria, ColégioTécnico Industrial de Santa Maria ; Rede e-Tec Brasil, 2013.
- ESPRESSIF SYSTEMS. **ESP32 Series Datasheet, 2021.** Disponível em:
<https://www.espressif.com/sites/default/files/documentation/esp32_datasheet_en.pdf>
Acesso em: Acesso em: 15 de agosto de 2022.
- FERNANDO K. **Introdução ao Esp32.** 2017. Disponível em:
<<https://www.fernandok.com/2017/11/introducao-ao-esp32.html>>. Acesso em: 15 de agosto
- FACCIONI FILHO, Mauro. **Internet das coisas:** livro digital / Mauro Faccioni Filho ; design instrucional Marina Cabeda Egger Moellwald. – Palhoça: UnisulVirtual, 2016. 56 p. : il. ; 28 cm. Acesso em: 18 de setembro de 2022
- FERREIRA, Márcio Antônio Couto. **Transporte fluvial por embarcações mistas no Amazonas: uma análise do trecho Manaus-Coari e Manaus- Parintins.** 2016. 164 f. Tese (Doutorado em Ciências do Ambiente e Sustentabilidade na Amazônia) - Universidade Federal do Amazonas, Manaus. 2016.
- FOLHA DE SÃO PAULO, EDITORIAL. **Pelo Menos Mil Pessoas Morreram Nos Rios Da Região Amazônica Desde 1981.** Disponível em <

<https://m.folha.uol.com.br/cotidiano/2017/08/1912508-pelo-menos-mil-pessoas-morreram-nos-rios-da-regiao-amazonica-desde-1981.shtml> >. Acesso em: 05 agosto de 2022.>

HELTEC. **WiFi LoRa 32 (V2.1) Phaseout**. Disponível em: < <https://heltec.org/project/wifi-lora-32/>>. Acesso em: 15 de agosto de 2022.

KAGEYAMA, Matheus Kenji Glassey; MENEZES, Victor Augusto Domingos Calixto de. **Compartilhamento de recursos em redes LoRa**. 2019. Trabalho de Conclusão de Curso (Sistemas de Informação) - Universidade Tecnológica Federal do Paraná, Curitiba, 2019.

LOGPYX. **LoRa: o que é essa solução para a conectividade das coisas**. Disponível em: < <https://logpyx.com/blog/lora-solucao-para-conectividade-das-coisas/>>. Acesso em: 20 de setembro de 2022.

MARINHA DO BRASIL. **Segurança do Tráfego Aquaviário: preocupação constante da Autoridade Marítima Brasileira**. 2022. Disponível em: <https://www.marinha.mil.br/noticias/seguranca-do-trafego-aquaviario-preocupacao-constante-da-autoridade-maritima-brasileira>. Acesso em: 16 de setembro de 2022.

MEKKI, Kais; BAJIC, Eddy; CHAXEL, Frederic; MEYER, Fernand. **A comparative study of LPWAN technologies for large-scale IoT deployment**. ICT Express, Elsevier BV, v. 5, n. 1, p. 1–7, mar 2019.

NETWORK, The Things. **Building a global open LoRaWAN® network**. 2021. The Things Network. Disponível em: <https://www.thethingsnetwork.org/>. Acesso em: 20 de setembro de 2022.

NOGUEIRA, Ricardo José Batista. **Amazonas, um estado ribeirinho**. Manaus: EDUA, 1999

NOVAES, A. G. Logística e Gerenciamento da Cadeia de Distribuição. Editora Elsevier, Rio de Janeiro, 2004.

PROAKIS, John. Digital communications. In: . 5. ed. Boston: McGraw-Hill, 2008. cap. **Chapter 12: Spread Spectrum Signals for Digital Communications**. ISBN 9780072957167.

SANCHES, Eduardo. **O desenvolvimento sustentável do sistema de transportes do Brasil**.

SEMTECH. **LoRaWan Academy**. Disponível em: <<https://learn.semtech.com/course/index.php?categoryid=2>>. Acesso em: 20 de setembro de 2022.

SOUZA, F. V. M. de; RABELLO, R. dos S. **Desenvolvimento de um protótipo com utilização de LoRaWAN como Solução de Comunicação de Baixo Custo**. 2017.

SUÇUARANA, Monik da Silveira. **Rio Amazonas**. Disponível em: < <https://www.infoescola.com/hidrografia/rio-amazonas/>>. Acesso em: 08 de agosto de 2022

SANTOS, Bruno Pereira et al. **Internet das coisas: da teoria à prática**. XXXIV Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos. Disponível em: <http://www.sbrc2016.ufba.br/minicurso/minicurso-1/>. Acesso em: 18 de setembro de 2022

SIMONI, Gabriel Mendes; SCARAMELLA, Geovana. **Monitoramento de veículos alternativos: desenvolvimento de um protótipo com tecnologia LoRaWAN**. 2021. Trabalho de Conclusão de Curso (Bacharelado em Engenharia Eletrônica) - Universidade Tecnológica Federal do Paraná, Curitiba, 2021.

U. Raza, P. Kulkarni and M. Sooriyabandara, "**Low Power Wide Area Networks: An Overview**," in *IEEE Communications Surveys & Tutorials*, vol. 19, no. 2, pp. 855-873, Secondquarter 2017, doi: 10.1109/COMST.2017.2652320.

UOL. **Rios da Amazônia desafiam autoridades e já somam 142 naufrágios desde 2017. 2020**. Disponível em: < <https://noticias.uol.com.br/meio-ambiente/ultimas-noticias/redacao/2020/03/07/rios-da-amazonia-desafiam-autoridades-e-ja-somam-142-naufragios-desde-2017.htm> >. Acesso em: 16 de setembro de 2022.

USINAINFO. **ESP32 Lora Wifi SX1278**. Disponível em: <<https://www.usinainfo.com.br/blog/esp32-lora-wifi-sx1278/>> Acesso em: 18 de setembro de 2022.

YAHOO NOTÍCIAS. **“Três pessoas morrem após barco naufragar no interior do AM”**.2022. Disponível em: < <https://br.noticias.yahoo.com/tres-pessoas-morrem-apos-barco-naufragar-no-interior-do-am-204328949.html/>>. Acesso em: 05 de agosto de 2022.

APÊNDICE A – CÓDIGO TRANSMISSOR

```

/* Bibliotecas para o Display OLED*/
#include <Wire.h>
#include <Adafruit_GFX.h>
#include <Adafruit_SSD1306.h>

/* Bibliotecas para comunicação LoRa */
#include <LoRa.h>
#include <SPI.h>

/* Header-file com as funções utilizadas para manipulação da partição NVS */
#include "nvs_flash.h"

/* Bibliotecas para o módulo de GPS */
#include <SoftwareSerial.h>
#include <TinyGPS.h>

/* Pinagem para o Display Oled */
#define OLED_SDA 4
#define OLED_SCL 15
#define OLED_RST 16
#define SCREEN_WIDTH 128
#define SCREEN_HEIGHT 64

/* Pinagem para comunicação com radio LoRa */
#define SCK_LORA          5
#define MISO_LORA         19
#define MOSI_LORA         27
#define RESET_PIN_LORA   14
#define SS_PIN_LORA       18
#define HIGH_GAIN_LORA    20 /* dBm */
#define BAND               915E6 /* 915MHz de frequencia */

/* Chave atribuida ao valor a ser escrito e lido da partição NVS */
#define CHAVE_NVS  "fe"

/* Pinagem para o módulo de GPS */
const int RX_PIN = 36; /* Ligar no TX do GPS */
const int TX_PIN = 37; /* Ligar no RX do GPS */
const int BAUD_RATE = 9600;

/* Pinagem para o botão do fator de espalhamento */
uint32_t fatorE = 7;
int botao = 23 ;

/* Variáveis globais para o módulo de GPS */
TinyGPS gps;

```

```

SoftwareSerial ss(RX_PIN, TX_PIN);

Adafruit_SSD1306 display(SCREEN_WIDTH, SCREEN_HEIGHT, &Wire, OLED_RST); /*
Definições do Display OLED */

/* Estrutura do pacote de dados */
typedef struct __attribute__((__packed__))
{
    int contador;
    byte hora;
    byte minuto;
    byte segundo;
    byte dia;
    byte mes;
    int ano;
    float f_latitude;
    float f_longitude;
    float temperatura;
    float umidade;
} TDadosLora;

void aguardando_dados_display();
void envia_medicoes_serial(TDadosLora dados_lora);
void escreve_medicoes_display(TDadosLora dados_lora);
void envia_informacoes_lora(TDadosLora dados_lora);
bool init_comunicacao_lora(void);

void aguardando_dados_display() {
    /* Imprimir mensagem dizendo que está aguardando o funcionamento do GPS */
    display.clearDisplay();
    display.setCursor(0, 0);
    display.println("Aguardando o GPS...");
    display.setCursor(0, 10);
    display.print("F.E.: ");
    display.println(fatorE);
    display.display();

    delay(200);
}

void escreve_medicoes_display(TDadosLora dados_lora)
{
    display.clearDisplay();

    display.setCursor(0, 0);
    display.print("Cont: ");
    display.println(dados_lora.contador);

    display.setCursor(0, 10);

```

```

    display.print("F.E.: ");
    display.println(fatorE);

    char str_flat[11] = {0};
    char str_flon[11] = {0};
    sprintf(str_flat, "%.6f", dados_lora.f_latitude);
    sprintf(str_flon, "%.6f", dados_lora.f_longitude);

    display.setCursor(0, 20);
    display.print("Lat.: ");
    display.println(str_flat);

    display.setCursor(0, 30);
    display.print("Lon.: ");
    display.println(str_flon);

    display.display();
}

void envia_medicoes_serial(TDadosLora dados_lora)
{
    char mensagem[200];

    Serial.print("Contador: ");
    Serial.println(dados_lora.contador);

    memset(mensagem,0,sizeof(mensagem));
    sprintf(mensagem,"Hora: %d:%d:%d", dados_lora.hora, dados_lora.minuto,
dados_lora.segundo);
    Serial.println(mensagem);

    Serial.print("Fator de Espalhamento: ");
    Serial.println(fatorE);

    memset(mensagem,0,sizeof(mensagem));
    sprintf(mensagem,"Lat: %.6f", dados_lora.f_latitude);
    Serial.println(mensagem);

    memset(mensagem,0,sizeof(mensagem));
    sprintf(mensagem,"Lon: %.6f", dados_lora.f_longitude);
    Serial.println(mensagem);

    Serial.println(" ");
}

void envia_informacoes_lora(TDadosLora dados_lora)
{
    LoRa.beginPacket();
    LoRa.write((unsigned char *)&dados_lora, sizeof(TDadosLora));
}

```



```

    LoRa.endPacket();
}

bool init_comunicacao_lora(void)
{
    bool status_init = false;
    Serial.println("[LoRa Emissor] Tentando iniciar comunicacao com o radio
LoRa...");
    SPI.begin(SCK_LORA, MISO_LORA, MOSI_LORA, SS_PIN_LORA);
    LoRa.setPins(SS_PIN_LORA, RESET_PIN_LORA, LORA_DEFAULT_DIO0_PIN);

    display.clearDisplay();

    if (!LoRa.begin(BAND))
    {
        Serial.println("[LoRa Emissor] Comunicacao com o radio LoRa falhou. Nova
tentativa em 1 segundo...");
        status_init = false;

        display.setCursor(0, 0);
        display.println("Radio LoRa");
        display.setCursor(0, 10);
        display.println("Status: Conectando...");
        display.setCursor(0, 20);
        display.println("Tentativas: Cada 1s");
        display.display();

        delay(1000);
    }
    else
    { // setar os parâmetros do rádio
        LoRa.setTxPower(HIGH_GAIN_LORA); /* Configura o ganho do receptor LoRa
para 20dBm, o maior ganho possível (visando maior alcance possível) */
        LoRa.setSignalBandwidth(125E3); /* Largura de banda fixa de 125 kHz -
Suporta valores: 7.8E3, 10.4E3, 15.6E3, 20.8E3, 31.25E3, 41.7E3, 62.5E3,
125E3, 250E3 e 500E3 */
        LoRa.setCodingRate4(5); /* Taxa de código - Suporta valores
entre 5 e 8 */
        LoRa.setSyncWord(0x55); /* Palavra de sincronização. Deve ser a
mesma no transmissor e receptor */

        Serial.println("[LoRa Emissor] Comunicacao com o radio LoRa ok");
        status_init = true;

        display.setCursor(0, 0);
        display.println("Radio LoRa");
        display.setCursor(0, 10);
        display.println("Status: Ok");
        display.display();
    }
}

```

```

    }

    return status_init;
}

/* Função: grava na NVS um dado do tipo interio 32-bits sem sinal, na chave
definida em CHAVE_NVS */
void grava_dado_nvs(uint32_t dado)
{
    nvs_handle handler_particao_nvs;
    esp_err_t err;

    err = nvs_flash_init_partition("nvs");

    if (err != ESP_OK)
    {
        Serial.println("[ERRO] Falha ao iniciar partição NVS.");
        return;
    }

    err = nvs_open_from_partition("nvs", "ns_nvs", NVS_READWRITE,
&handler_particao_nvs);
    if (err != ESP_OK)
    {
        Serial.println("[ERRO] Falha ao abrir NVS como escrita/leitura");
        return;
    }

    /* Atualiza valor do horimetro total */
    err = nvs_set_u32(handler_particao_nvs, CHAVE_NVS, dado);

    if (err != ESP_OK)
    {
        Serial.println("[ERRO] Erro ao gravar horimetro");
        nvs_close(handler_particao_nvs);
        return;
    }
    else
    {
        Serial.println("Dado gravado com sucesso!");
        nvs_commit(handler_particao_nvs);
        nvs_close(handler_particao_nvs);
    }
}

/* Função: le da NVS um dado do tipo interio 32-bits sem sinal, contido na
chave definida em CHAVE_NVS */
uint32_t le_dado_nvs(void)
{

```

```

nvs_handle handler_particao_nvs;
esp_err_t err;
uint32_t dado_lido;

err = nvs_flash_init_partition("nvs");

if (err != ESP_OK)
{
    Serial.println("[ERRO] Falha ao iniciar partiç o NVS.");
    return 0;
}

err = nvs_open_from_partition("nvs", "ns_nvs", NVS_READWRITE,
&handler_particao_nvs);
if (err != ESP_OK)
{
    Serial.println("[ERRO] Falha ao abrir NVS como
escrita/leitura");
    return 0;
}

/* Faz a leitura do dado associado a chave definida em CHAVE_NVS */
err = nvs_get_u32(handler_particao_nvs, CHAVE_NVS, &dado_lido);

if (err != ESP_OK)
{
    Serial.println("[ERRO] Falha ao fazer leitura do dado");
    return 0;
}
else
{
    Serial.println("Dado lido com sucesso!");
    nvs_close(handler_particao_nvs);
    return dado_lido;
}
}

//Pino ligado no bot o
int btn_on = 12; //aqui
int btn_off = 13; //aqui
bool estado = 0;
void setup()
{
    //Seta os pinos dos bot es como entrada //aqui
    pinMode(btn_on, INPUT_PULLUP); //aqui
    pinMode(btn_off, INPUT_PULLUP);
    pinMode(botao, INPUT_PULLUP);
}

```

```

/* Inicialização do módulo GPS */
ss.begin(BAUD_RATE);

/* Monitor Serial */
Serial.begin(115200);

/* Preparando a inicialização do display OLED */
pinMode(OLED_RST, OUTPUT);
digitalWrite(OLED_RST, LOW);
delay(20);
digitalWrite(OLED_RST, HIGH);

/* Inicialização do display OLED */
Wire.begin(OLED_SDA, OLED_SCL);
if(!display.begin(SSD1306_SWITCHCAPVCC, 0x3c, false, false)) { // Address
0x3C for 128x32
  Serial.println(F("Falha no Display Oled"));
  for(;;); // Don't proceed, loop forever
}

/* Mensagem inicial */
Serial.println("EMISSOR LORA");
display.clearDisplay();
display.setTextColor(WHITE);
display.setTextSize(1);
display.setCursor(0,0);
display.print("Emissor LoRa");
display.display();
delay(2000);

/* Leitura da memória flash que guarda o valor do Fator de Espalhamento */
uint32_t dado_lido = le_dado_nvs();

/* Lendo ou gravando o Fator de Espalhamento na memória flash do Esp32 */
if (dado_lido < 7 || 12 < dado_lido){
  grava_dado_nvs(fatorE);
} else {
  fatorE = dado_lido;
}

while(init_comunicacao_lora() == false); /* Tenta, até obter sucesso na
comunicacao com o chip LoRa */

delay(2000);

/* Imprimir mensagem dizendo que está aguardando o funcionamento do GPS */
aguardando_dados_display();

/* Pino de entrada do botão de mudança do fator de espalhamento*/

```

```

    //pinMode(botao, INPUT);
}

int cont = 0;
unsigned long tempoAntes = millis();

void loop()
{
    TDadosLora dados_lora;
    dados_lora.contador = cont;
    unsigned long idade;
    bool newData = false;

    /* Realiza a leitura do módulo GPS */
    for (unsigned long start = millis(); millis() - start < 1000;)
    {
        while (ss.available())
        {
            char c = ss.read();
            if (gps.encode(c)) {
                newData = true;
                break;
            }
        }
    }

    /* Fator de Espalhamento */
    if (millis() - tempoAntes > 20) /* Executar a cada 20 milisegundos devido
a uma oscilação do pino ao ligar o dispositivo */
    {
        if (digitalRead(botao) == LOW)
        {
            fatorE = fatorE + 1;
            if (fatorE < 7 || 12 < fatorE)
            {
                fatorE = 7;
            }
            grava_dado_nvs(fatorE);          /* Gravando o Fator de Espalhamento
na memória flash do Esp32 */
            LoRa.setSpreadingFactor(fatorE); /* Configurando o Fator de
Espalhamento */
            aguardando_dados_display();
        }

        tempoAntes = millis();
    }
}

/* Verifica se a leitura do módulo de GPS foi bem sucedida */

```

```
if (newData)
{

    gps.f_get_position(&dados_lora.f_latitude, &dados_lora.f_longitude,
&idade);

    byte centesimo;
    gps.crack_datetime(&dados_lora.ano, &dados_lora.mes, &dados_lora.dia,
&dados_lora.hora, &dados_lora.minuto, &dados_lora.segundo, &centesimo,
&idade);

    envia_medicoes_serial(dados_lora);
    escreve_medicoes_display(dados_lora);

    if (digitalRead(btn_on) == LOW){
    estado = 1;
    }

    if (digitalRead(btn_off) == LOW){
    estado = 0;
    cont=0;
    }

    if (estado == 1 ) //Caso o botão 1 foi pressionado //aqui
    { //aqui
    //Exibe no monitor serial a mensagem entre aspas
    Serial.println("Botão pressionado");
    envia_informacoes_lora(dados_lora);
    cont++;

    }
}
}
```

APÊNDICE A – CÓDIGO RECEPTOR

```

//receptor
/* Bibliotecas para o Display OLED*/
#include <Wire.h>
#include <Adafruit_GFX.h>
#include <Adafruit_SSD1306.h>

/* Bibliotecas para comunicação LoRa */
#include <LoRa.h>
#include <SPI.h>

/* Header-file com as funções utilizadas para manipulação da partição NVS */
#include "nvs_flash.h"

/* Pinagem para o Display Oled */
#define OLED_SDA 4
#define OLED_SCL 15
#define OLED_RST 16
#define SCREEN_WIDTH 128
#define SCREEN_HEIGHT 64

/* Pinagem para comunicação com radio LoRa */
#define SCK_LORA 5
#define MISO_LORA 19
#define MOSI_LORA 27
#define RESET_PIN_LORA 14
#define SS_PIN_LORA 18
#define HIGH_GAIN_LORA 20 /* dBm */
#define BAND 915E6 /* 915MHz de frequencia */

/* Chave atribuida ao valor a ser escrito e lido da partição NVS */
#define CHAVE_NVS "teste"

/* Pinagem para o fator de espalhamento */
uint32_t fatorE = 7; /* Valor do fator de espalhamento */
int pinoBotao = 0; /* Valor do pino do botão */

/* Definições do Display OLED */
Adafruit_SSD1306 display(SCREEN_WIDTH, SCREEN_HEIGHT, &Wire, OLED_RST);

/* Estrutura do pacote de dados */
typedef struct __attribute__((__packed__))
{
    int contador;
    byte hora;
    byte minuto;
    byte segundo;
}

```

```

byte dia;
byte mes;
int ano;
float f_latitude;
float f_longitude;
} TDadosLora;

void aguardando_dados_display();
void escreve_medicoes_display(TDadosLora dados_lora, int lora_rssi);
void envia_medicoes_serial(TDadosLora dados_lora, int lora_rssi, int
tam_pacote);
bool init_comunicacao_lora(void);
void grava_dado_nvs(uint32_t dado);
uint32_t le_dado_nvs(void);

void aguardando_dados_display() {
    /* Imprimir mensagem dizendo para esperar a chegada dos dados */
    Serial.print("Aguardando dados... F.E: ");
    Serial.println(fatorE);

    display.clearDisplay();
    display.setCursor(0, 0);
    display.println("Aguardando dados...");
    display.setCursor(0, 10);
    display.print("F.E.: ");
    display.println(fatorE);
    display.display();
    digitalWrite(12,LOW);

    delay(200);
}

void escreve_medicoes_display(TDadosLora dados_lora, int lora_rssi)
{
    char str_rssi[11];
    char str_flat[11];
    char str_flon[11];

    memset(str_rssi,0,sizeof(str_rssi));
    memset(str_flat,0,sizeof(str_flat));
    memset(str_flon,0,sizeof(str_flon));

    sprintf(str_rssi, "%d", lora_rssi);
    sprintf(str_flat, "%.6f", dados_lora.f_latitude);
    sprintf(str_flon, "%.6f", dados_lora.f_longitude);

    display.clearDisplay();

    display.setCursor(0, 0);

```



```

//Serial.println("[LoRa Receptor] Tentando iniciar comunicacao com o radio
LoRa...");
SPI.begin(SCK_LORA, MISO_LORA, MOSI_LORA, SS_PIN_LORA);
LoRa.setPins(SS_PIN_LORA, RESET_PIN_LORA, LORA_DEFAULT_DIO0_PIN);

if (!LoRa.begin(BAND)) {
    status_init = false;

    Serial.println("[LoRa Receptor] Comunicacao com o radio LoRa falhou.
Nova tentativa em 1 segundo...");

    display.clearDisplay();
    display.setCursor(0, 0);
    display.println("Radio LoRa");
    display.setCursor(0, 10);
    display.println("Status: Conectando...");
    display.setCursor(0, 20);
    display.println("Tentativas: Cada 1s");
    display.display();

    delay(1000);
} else {
    status_init = true;
    LoRa.setSpreadingFactor(fatorE); /* Fator de Espalhamento */
    LoRa.setTxPower(HIGH_GAIN_LORA); /* Configura o ganho do receptor LoRa
para 20dBm, o maior ganho possível (visando maior alcance possível) */
    LoRa.setSignalBandwidth(125E3); /* Largura de banda fixa de 125 kHz
/**/ Suporta valores: 7.8E3, 10.4E3, 15.6E3, 20.8E3, 31.25E3, 41.7E3, 62.5E3,
125E3, 250E3 e 500E3 */
    LoRa.setCodingRate4(5); /* Taxa de código - Suporta valores
entre 5 e 8 */
    LoRa.setSyncWord(0x55); /* Palavra de sincronização. Deve ser a
mesma no transmissor e receptor */
}

return status_init;
}

/* Função: grava na NVS um dado do tipo interio 32-bits sem sinal, na chave
definida em CHAVE_NVS */
void grava_dado_nvs(uint32_t dado)
{
    nvs_handle handler_particao_nvs;
    esp_err_t err;

    err = nvs_flash_init_partition("nvs");

    if (err != ESP_OK)
    {

```

```

        Serial.println("[ERRO] Falha ao iniciar partiçao NVS.");
        return;
    }

    err = nvs_open_from_partition("nvs", "ns_nvs", NVS_READWRITE,
&handler_particao_nvs);
    if (err != ESP_OK)
    {
        Serial.println("[ERRO] Falha ao abrir NVS como escrita/leitura");
        return;
    }

    /* Atualiza valor do horimetro total */
    err = nvs_set_u32(handler_particao_nvs, CHAVE_NVS, dado);

    if (err != ESP_OK)
    {
        Serial.println("[ERRO] Erro ao gravar horimetro");
        nvs_close(handler_particao_nvs);
        return;
    }
    else
    {
        Serial.println("Dado gravado com sucesso!");
        nvs_commit(handler_particao_nvs);
        nvs_close(handler_particao_nvs);
    }
}

/* Função: le da NVS um dado do tipo interio 32-bits sem sinal, contido na
chave definida em CHAVE_NVS */
uint32_t le_dado_nvs(void)
{
    nvs_handle handler_particao_nvs;
    esp_err_t err;
    uint32_t dado_lido;

    err = nvs_flash_init_partition("nvs");

    if (err != ESP_OK)
    {
        Serial.println("[ERRO] Falha ao iniciar partiçao NVS.");
        return 0;
    }

    err = nvs_open_from_partition("nvs", "ns_nvs", NVS_READWRITE,
&handler_particao_nvs);
    if (err != ESP_OK)
    {

```

```

        Serial.println("[ERRO] Falha ao abrir NVS como
escrita/leitura");
        return 0;
    }

    /* Faz a leitura do dado associado a chave definida em CHAVE_NVS */
    err = nvs_get_u32(handler_particao_nvs, CHAVE_NVS, &dado_lido);

    if (err != ESP_OK)
    {
        Serial.println("[ERRO] Falha ao fazer leitura do dado");
        return 0;
    }
    else
    {
        Serial.println("Dado lido com sucesso!");
        nvs_close(handler_particao_nvs);
        return dado_lido;
    }
}

void setup()
{
    pinMode(12, OUTPUT);
    /* Monitor Serial */
    Serial.begin(115200);

    /* Preparando a inicialização do display OLED */
    pinMode(OLED_RST, OUTPUT);
    digitalWrite(OLED_RST, LOW);
    delay(20);
    digitalWrite(OLED_RST, HIGH);

    /* Inicialização do display OLED */
    Wire.begin(OLED_SDA, OLED_SCL);
    if(!display.begin(SSD1306_SWITCHCAPVCC, 0x3c, false, false)) { /* Endereço
0x3C para 128 x 32 */
        Serial.println(F("Falha no Display OLED"));
        for(;;); /* Loop infinito */
    } else {
        display.clearDisplay();
        display.setTextColor(WHITE);
        display.setTextSize(1);
    }
}

/* Leitura da memória flash que guarda o valor do Fator de Espalhamento */
uint32_t dado_lido = le_dado_nvs();

/* Lendo ou gravando o Fator de Espalhamento na memória flash do Esp32 */

```

```

    if (dado_lido < 7 || 12 < dado_lido){
        grava_dado_nvs(fatorE);
    } else {
        fatorE = dado_lido;
    }
}

while(init_comunicacao_lora() == false); /* Tenta, até obter sucesso na
comunicacao com o chip LoRa */

/* Pino de entrada do botão */
pinMode(pinoBotao, INPUT);

/* Imprimir mensagem dizendo para esperar a chegada dos dados */
aguardando_dados_display();
}

unsigned long tempoAntes = millis();
unsigned long tempoRecepcao = millis();
bool canRestart = false;

void loop()
{
    TDadosLora dados_lora;
    char byte_recebido;
    int tam_pacote = 0;
    int lora_rssi = 0;
    char * ptInformacaoRecebida = NULL;
    unsigned long idade;
    bool newData = false;

    /* Executar a cada 20 milisegundos devido a uma oscilação do pino ao ligar o
dispositivo */
    if (millis() - tempoAntes > 20)
    {
        /* Mudando o fator de espalhamento */
        if (digitalRead(pinoBotao) == LOW) {
            fatorE = fatorE +1;

            if (fatorE < 7 || 12 < fatorE) {
                fatorE = 7;
            }

            grava_dado_nvs(fatorE);          /* Gravando o Fator de Espalhamento na
memória flash do Esp32 */
            LoRa.setSpreadingFactor(fatorE); /* Configurando o Fator de Espalhamento
*/
            aguardando_dados_display();
        }
        tempoAntes = millis();
    }
}

```

```

    /* Verifica se é necessário reiniciar. Caso passe mais de 5 segundos sem
receber dados */
    if(millis() - tempoRecepcao > 5000) {
        if (canRestart) {
            Serial.println("Reiniciando ESP32");
            ESP.restart();
        }
    }
}

tam_pacote = LoRa.parsePacket(); /* Verifica se chegou alguma informação do
tamanho esperado */

if (tam_pacote == sizeof(TDadosLora)) {
    ptInformacaoRecebida = (char *)&dados_loRa; /* Recebe os dados conforme
protocolo */
    while (LoRa.available())
    {
        byte_recebido = (char)LoRa.read();
        *ptInformacaoRecebida = byte_recebido;
        ptInformacaoRecebida++;
    }

    lora_rssi = LoRa.packetRssi(); /* Escreve RSSI de recepção e informação
recebida */

    envia_medicoes_serial(dados_loRa, lora_rssi, tam_pacote);
    escreve_medicoes_display(dados_loRa, lora_rssi);

    /* Tratamento para reiniciar o dispositivo */
    tempoRecepcao = millis();
    canRestart = true;
}
}

```