UNIVERSIDADE FEDERAL RURAL DO SEMI-ÁRIDO
PRÓ-REITORIA DE GRADUAÇÃO
DEPARTAMENTO DE ENGENHARIAS
CURSO BACHARELADO EM ENGENHARIA ELÉTRICA

SAVIO RAGGY MENEZES DE SOUZA

# SISTEMA DE TELEMETRIA PARA VEÍCULO BAJA DA UFERSA/CAMPUS CARAÚBAS.

CARAÚBAS-RN
2023

SÁVIO RAGGY MENEZES DE SOUZA

# SISTEMA DE TELEMETRIA PARA VEÍCULO BAJA DA UFERSA/CAMPUS CARAÚBAS.

Trabalho de Conclusão de Curso apresentado à Universidade Federal Rural do Semi-Árido como requisito para obtenção do título de Bacharel em Engenharia Elétrica.

Orientador: Prof. Dr. Walber Medeiros Lima

CARAÚBAS-RN
2023

DS729s De Souza, Sávio Raggy Menezes.
Sistema de telemetria para veículo baja da UFERSA/campus Caraúbas. / Sávio Raggy Menezes De Souza. – 2023.
61 f. : il.

Orientador: Walber Medeiros Lima.
Coorientador: Charles Augusto Santos do Carmo.
Monografia (graduação) – Universidade Federal Rural do Semi-árido, Curso de Engenharia Elétrica, 2023.

1. Telemetria. 2. ESP32 LoRa. 3. Temperatura. 4. Umidade. 5. Arduino IDE. I. Medeiros Lima, Walber , orient. II. Santos do Carmo, Charles Augusto , co-orient. III. Título.

Ficha catalográfica elaborada por sistema gerador automáto em conformidade com AACR2 e os dados fornecidos pelo) autor(a).
Biblioteca Campus Caraúbas
Bibliotecária: Dalvanira Brito Rodrigues
CRB: 15/700

SÁVIO RAGGY MENEZES DE SOUZA

**SISTEMA DE TELEMETRIA PARA VEÍCULO BAJA DA UFERSA/CAMPUS CARAÚBAS.**

Trabalho de conclusão de curso apresentado a Universidade Federal Rural do Semi-Árido como requisito para obtenção do título de Bacharel em engenharia elétrica.

Defendida em: 11/10/2023.

**BANCA EXAMINADORA**

WALBER MEDEIROS LIMA:[redacted]
Assinado de forma digital por WALBER MEDEIROS LIMA:[redacted]
Dados: 2023.10.16 11:11:41 -03'00'

Prof. Dr. Walber Medeiros Lima
(UFERSA)
Presidente

HUGO MICHEL CÂMARA DE AZEVEDO MAIA
Assinado de forma digital por HUGO MICHEL CÂMARA DE AZEVEDO MAIA
Dados: 2023.10.16 14:34:01 -03'00'

Prof. Dr. Hugo Michel Camara de Azevedo Maia
(UFERSA)
Membro Examinador

FRANCISCO DE ASSIS BRITO FILHO:[redacted]
Assinado de forma digital por FRANCISCO DE ASSIS BRITO FILHO:[redacted]
Dados: 2023.10.16 15:02:45 -03'00'

Prof. Dr. Francisco de Assis Brito Filho
(UFERSA)
Membro Examinador

## AGRADECIMENTOS

Em primeiro lugar, quero agradecer a Deus pelos dias percorridos dentro da universidade, pelo conhecimento adquirido e pela força que Ele me deu para superar todas as adversidades que surgiram durante o curso.

A minha família, em especial meus pais - minha mãe, Silvanete Manú de Souza Menezes, e meu pai, Ricardo Barboza de Menezes e também meu irmão, Rian Victor de Souza Menezes, e, por fim, minha avó, Maria Dalvani de Melo. Vocês são a razão de tanto esforço e dedicação.

Ao meu orientador, Walber Medeiros de Lima, por toda a dedicação e paciência durante o projeto de TCC. Além de ser um profissional ímpar na sua área, ele é um grande amigo, sempre me ajudando quando encontrei dificuldades.

Ao meu amigo e coorientador, Charles Augusto Santos do Carmo, por todo o apoio, ajuda e paciência no decorrer da elaboração do projeto de TCC. Você foi fundamental nesse projeto.

À Banca Examinadora, por estarem disponíveis para avaliar o meu trabalho e pelos ensinamentos que compartilharam.

Aos amigos de faculdade e colegas, em destaque, Antônio Ariel Cardoso, Douglas Renan de Souza Diógenes, Pedro Victor de Morais Souza, Ian Aires Lonêto. Entramos juntos atrás desse sonho e vocês nunca deixaram faltar incentivo e amizade durante o período.

Aos meus amigos, Rafaela Emanuely de Souza Soares, Mikael da Costa Pereira, Hellen Vitória Leite da Silva, Pablo Fernandes, por todo o apoio e amizade que me deram durante o período.

À minha querida amiga e avó, Raimunda Sinezia Manú de Souza, que não está mais entre nós, por sempre acreditar na minha capacidade e potencial. Tudo isso é por você.

Por fim, à Universidade Federal do Semi-Árido (UFERSA), juntamente com seu corpo técnico e docente, por todo o conhecimento e amizade adquiridos durante essa jornada. Muito obrigado a todos que fazem parte dessa instituição.

Muito obrigado a todos!

“Sejam fortes e corajosos. Não tenham medo nem fiquem apavorados por causa delas, pois o Senhor, o seu Deus, vai com vocês; nunca os deixará, nunca os abandonará".

Deuteronômio 31:6

**RESUMO**

Diante cenário que o mundo vive em função do desenvolvimento de novas tecnologias, algumas técnicas para produção de novos meios de telemetria são desenvolvidos, como por exemplo, o uso do ESP32 LoRa. Dessa forma, esse trabalho teve como objetivo estudar o desenvolvimento de um sistema de telemetria para veículos baja com o microcontrolador ESP32 com modulo LoRa. Com isso, foi produzido o protótipo utilizando dois ESP32 LoRa, a plataforma de desenvolvimento Arduino IDE e um sensor DHT11 para um projeto de medição de temperatura e umidade através de um modulo de rádio LoRa. Posteriormente, foram realizados alguns estudos, como o teste do código fonte, tanto para o transmissor e receptor, a comunicação entre os dois códigos, teste do sensor DHT11, a prova da comunicação em pequenas distancias, longas distancias e a utilização do projeto no veículo do tipo off round BAJA. Com isso, destacando-se alguns dos resultados obtidos diante do estudo, foi possível identificar na avaliação do projeto mencionado alguns pontos importantes, caso da perfeita comunicação entre os dispositivos ESP32 LoRa, o pequeno raio de alcance com as antenas que vieram de fabrica dos dispositivos, a utilização de uma antena de melhor comunicação entre os objetos e funcionamento em perfeitas condições utilizando o veículo mencionado em movimentação com o dispositivo receptor contido em campo aberto e fechado. Dessa forma, diante dos resultados obtidos, pode-se destacar ainda, especialmente, o uso do ESP32 LoRA para comunicação de informações de suma importância para diferentes áreas e devendo-se então, realizar outros estudos para aperfeiçoar e melhorar o projeto apresentado.

**Palavras-chave:** Telemetria. ESP32 LoRa. Temperatura. Umidade. Arduino IDE.

**ABSTRACT**

In the context of the ever-evolving world driven by new technologies, various techniques for telemetry systems are being developed, one of which is the utilization of ESP32 LoRa. Accordingly, this study aimed to explore the development of a telemetry system for Baja vehicles using the ESP32 microcontroller with a LoRa module. Consequently, a prototype was created employing two ESP32 LoRa modules, the Arduino IDE development platform, and a DHT11 sensor for a temperature and humidity measurement project through a LoRa radio module. Subsequently, several assessments were conducted, including source code testing for both the transmitter and receiver, communication between the two codes, DHT11 sensor testing, short-distance and long-distance communication trials, and the use of the project on an off-road Baja-type vehicle. In light of the study's findings, key observations include the seamless communication

between ESP32 LoRa devices, the limited range with the factory antennas of the devices, the use of a more effective antenna for enhanced communication, and the successful operation in various conditions, including the vehicle's movement with the receiver device in both open and closed fields. Consequently, based on the results obtained, it is noteworthy to emphasize the utility of ESP32 LoRa for conveying critical information across diverse domains, suggesting the need for further research to refine and enhance the presented project

**Keywords:** Telemetry. ESP32 LoRa. Temperature. Humidity. Arduino IDE.

## LISTA DE FIGURAS

## LISTA DE TABELAS

## LISTA DE ABREVIAÇÕES E SIGLAS

IoT - Internet of Things

IBGE - Instituto Brasileiro de Geografia e Estatística

FIAT - Fabbrica Italiana Automobili Torino

SPARC - Scalable Processor ARChitecture

PowerPC - Conjunto de instruções projetado juntamente aos princípios RISC

GPUs - Graphics Processing Unit

LoRa - Long Range

NPUs - Unidades de processamento neural

**SUMÁRIO**

# 1 INTRODUÇÃO

Segundo Ashton (2009), a nova era da inovação tecnológica, onde ocorre a união entre o mundo físico e o meio digital, está se desenvolvendo radicalmente por meio da Internet das Coisas (IoT). A IoT representa um fenômeno que transcende as limitações convencionais da conectividade, introduzindo uma nova era na qual objetos cotidianos ganham inteligência e conectividade, tornando-se participantes ativos em um ecossistema interconectado.

Com o passar dos anos, o desenvolvimento e a evolução do conceito de IoT vêm tomando forma, trazendo à tona a grande transformação que o meio tecnológico vem experimentando. Isso é evidenciado pela importância dos componentes como sensores, microcontroladores, microprocessadores e softwares de desenvolvimento, que desempenham um papel crucial nesse avanço (Culler et al., 2004).

É notável que há algum tempo os componentes eletrônicos têm ganhado destaque na construção de diferentes sistemas integrados. Um exemplo disso são os microcontroladores, que tiveram uma grande expansão na última década devido à sua praticidade, tecnologia e baixo custo de produção (Adamoski, Furukawa, 2001).

Nos anos recentes, uma das principais preocupações da sociedade tem sido o controle tecnológico em diversos ambientes. Com o avanço das tecnologias e o desenvolvimento desses ambientes, os microcontroladores ganharam força no mercado, devido à sua aplicação em sistemas de integração (MCT, 2001).

De acordo com Souza (2005), os microcontroladores são definidos como componentes eletrônicos de alta integração, seja pelo tamanho que a maioria possui ou pela tecnologia subjacente. Eles são dotados de inteligência programável, uma variedade de dispositivos integrados, como memória, portas lógicas, contadores e conversores, entre outros.

Atualmente, com a abundância de microcontroladores disponíveis, as grandes fabricantes de automóveis estão desenvolvendo tecnologias com base nesses sistemas embarcados, aplicando-os em diferentes áreas, desde sistemas de controle de temperatura até sistemas de gerenciamento de energia, como uma forma de aprimorar seus protótipos e lançamentos no mercado automotivo (Mendes, 2015).

Entrando no mundo dos microcontroladores e sua aplicação em veículos, surge o ESP32, um dispositivo que integra um microcontrolador com Wi-Fi e Bluetooth. Este microcontrolador é amplamente utilizado em projetos relacionados à Internet das Coisas, pois pode interagir com várias bibliotecas, módulos e sensores em diversas linguagens de programação, sendo a mais comum a linguagem C++ ou C. Além disso, ele oferece um desempenho excepcional em termos de potência, velocidade e consumo de energia reduzido (Henriques, 2021).

De acordo com Pereira (2019), uma das grandes inovações tecnológicas que merece destaque quando se trata de comunicação ponto a ponto é a tecnologia ESP32 LoRa. Esses sistemas combinam um microcontrolador ESP32 com a tecnologia de comunicação LoRa (Long Range), proporcionando uma solução robusta para a construção de sistemas de Internet das Coisas (IoT) de longo alcance. O ESP32 LoRa ultrapassa os limites convencionais da conectividade, permitindo a eficiente troca de dados em distâncias consideráveis. Assim como a IoT, o ESP32 LoRa também visa transformar objetos cotidianos em elementos inteligentes.

Diante das exposições anteriores, é notória a grande importância da utilização desses componentes e da IoT nos dias atuais, seja pela necessidade de estar conectado em diversos momentos do nosso dia ou pela funcionalidade de muitos equipamentos e sistemas embarcados. Dessa forma, a telemetria e o ESP32 LoRa são de fundamental importância para medir diferentes parâmetros e contribuir para o melhor funcionamento de muitos projetos.

# 2 OBJETIVOS

## 2.1 Objetivos Geral

Este trabalho teve como objetivo o Desenvolvimento de um sistema de telemetria com esp32 com modulo LoRa.

## 2.2 Objetivos Específicos

- Projetar um projeto na plataforma de desenvolvimento Arduino IDE com ESP32 LoRa;
- Realizar testes com códigos bases para verificar funcionalidade dos ESP32 LoRa;
- Realizar testes após a confecção da etapa de desenvolvimento e com o auxílio de sensores no protótipo do veículo baja a verificação de sua funcionalidade de forma prática;
- Realizar medições de temperatura e umidade com o sensor DHT11.

# 3 REFERENCIAL TEÓRICO

## 3.1 Levantamento histórico sobre a IoT

Mediante o grande ato revolucionário chamado internet, é de grande respaldo os inúmeros benefícios que a mesma vem trazendo para a população de modo geral, seja na área da saúde, educação e indústria, mas um dos pontos que tem ganhado ênfase é a IoT (Internet of Things) ou em nossa língua a internet das coisas, podemos notar que os primeiros movimentos relacionados a IoT, veio antes mesmo da primeira geração de computadores na década de 1940 (Rowland et al., 2015).

Analisando raciocínio de McLuhan (1964) anos depois em 1960 o cientista trouxe à tona a teoria da noção de que os objetos poderiam ser conectados ao afirmar que através de meios elétricos, ia se ter a possibilidade de se criar uma dinâmica pela qual todas as tecnologias seriam traduzidas em sistemas de informação.

Segundo um dos grandes cientistas Nikola Tesla em uma entrevista à revista Colliers, mesmo em sua época a muitos anos atrás já previa a grande proporção que se tornaria a IoT e a internet de modo geral:

> "Quando a rede sem fio for perfeitamente aplicada, a Terra inteira será convertida em um cérebro enorme [...]. Nós deveremos poder nos comunicar instantaneamente, independentemente da distância. Não só isso, mas através da televisão e da telefonia, veremos e nos ouviremos tão perfeitamente como se estivéssemos cara a cara, apesar das distâncias intervenientes de milhares de quilômetros; e os instrumentos através dos quais poderemos fazer isso serão incrivelmente simples em comparação ao nosso telefone atual. Um homem poderá carregá-los no bolso do colete. (Tesla, 1926)."

Dando continuidade ao grande desenvolvimento da tecnológica durante os anos, é visto que na década de 90 se deu os primeiros experimentos de forma prática quando relacionados ao termo IoT, o termo foi tratado de forma inovadora, pois se tratava de uma visão de computadores onde os mesmos sentiam o mundo a sua volta, dessa forma destacando a capacidade de se ter uma relação mais intima entre objetos se comunicando de forma efetiva por meio da internet (Ashton, 2009).

Mas a primeira utilidade utilizando tais conceitos se deu nos anos 90, tendo como fundador o cientista da computação e inventor John Romkey que criou uma torradeira que poderia ser ligada e desligada pela internet. Onde a mesma é considerada o primeiro de muitos dispositivos de um novo mundo que viria a ser chamado de IoT (Campbell, 2014).

Assim segundo Evans (2012) o desenvolvimento da IoT é um marco continuo, pois à medida que a tecnologia é desenvolvida o seu crescimento é notório, sendo ela marcada por tecnologias como o 5G, o desenvolvimento de novos componentes eletrônicos, a introdução da inteligência artificial e a relação que esses itens vêm melhorando a cada dia, trazendo dessa forma um melhor aproveitamento e expansão da IoT.

Conforme Atzori et al. (2010) a IoT vem apresentando uma evolução natural na tecnologia que ela está inserida, seja pela comunicação sem fio, sensores ou atuadores, onde os mesmos ficam integrados entre eles fornecendo assim informações de uma ampla variedade de situações que podem atuar de forma conjunta possibilitando assim uma tomada de decisões e informações automatizadas entre si.

E ao se fazer um elo entre esses dispositivos é de suma importância levar em consideração a internet das coisas (IoT) nesse contexto, pois foi diante ela que se teve uma revolução no cotidiano das pessoas com o meio tecnológico, onde hoje é possível se conectar com diferentes aparelhos ao mesmo tempo, facilitando assim a automação e a facilidade de ser conectar com o meio digital a sensores, câmeras, lâmpadas e entre outras (Oulasvirta et al., 2012).

Segundo Sundmaeker et al. (2010) quando conectamos diferentes objetos e diferentes recursos a uma única rede surge um novo meio de aplicabilidade entre esses objetos, assim significa que esses objetos no momento que estão conectados se aumenta a comunicação entre os usuários ali presente com os objetos gerando uma nova gama de possibilidades e facilidades como sensoreamento de ambientes entre outras aplicabilidades.

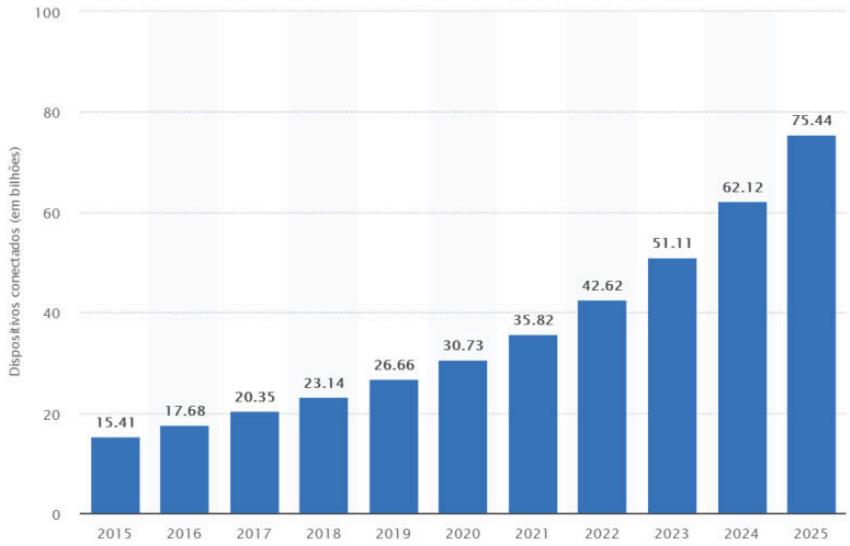
Dessa forma é de suma importância notar a grande contribuição que a internet das coisas vem proporcionando ao desenvolvimento de novos projetos, com isso criando ambientes entre dispositivos mais eficientes possibilitando um maior compartimento de informações e agindo de forma mais colaborativa, buscando desse modo uma integração mais prática entre o mundo digital e o mundo físico,

promovendo assim uma maior automação entre diferentes áreas do desenvolvimento (Gubbi et al., 2013).

Levando em consideração ao cenário brasileiro podemos atenuar que diante o quadro da revolução da internet no cotidiano do brasileiro é visto que durante o ano de 2014 o uso da internet móvel superou o uso em computadores tradicionais, trazendo à tona a versatilidade de se utilizar meios mais versáteis como forma de melhor comunicação entre as pessoas e objetos (IBGE, 2014).

Dessa forma podemos ver a seguir na figura 1, um dos panoramas que foi relatado anteriormente sobre o grande crescimento da IoT a nível mundial.

**Figura 1 -** Estimativa dos dispositivos conectados a Internet das Coisas instalados em todo o mundo de 2015 a 2025 (em bilhões).




**Fonte:** Adaptado de Startista (2019).

### 3.2 Aplicabilidade para Utilização da IoT

Segundo Itu (2005) uma das grandes utilidades da IoT é relacionado a telecomunicação onde a conexão de diferentes objetos estão interligados em nosso cotidiano e em diferentes dispositivos de redes, caso das intranets, redes peer-to-peer e a internet de modo global.

Dessa forma é notório a grande utilidade da IoT no conceito de cidades inteligentes onde a sua funcionalidade pode abranger diferentes aspectos como na coleta de resíduos, facilitando assim a identificação de compostos, níveis de carga, outros exemplos que podem ser mencionados são o monitoramento de câmeras de

segurança, sensores de diferentes aspectos onde estão inseridos em parâmetros como temperatura, umidade, movimento entre outros, que nesse aspecto traz um melhor aproveitamento desses serviços, através da comunicação entre objetos o (Zanella et al., 2014).

De acordo com Liu et al. (2015), outra aplicação que vem ganhando destaque é em relação a aplicabilidade da internet das coisas na agricultura, onde se tem um melhor controle de toda classe produtiva seja do início da sua produção, com controle e monitoramento da frota de maquinário, no controle de agrotóxicos e sua aplicabilidade na manutenção de pragas e na eficiência dos sistemas de irrigação tendo uma eficácia especifica na distribuição de água para cada cultura.

Outra situação importante na IoT é no meio educacional para o desenvolvimento de pesquisas nas universidades e nas escolas, por se tratar de um parâmetro inovador onde cada dia surge diferentes linhas de raciocínio, caso de novas utilizações em diferentes objetos (Greff, 2018).

Dessa forma uma das aplicações primárias em relação a aplicação da IoT nas instituições educacionais é a criação de ambientes de aprendizado inteligentes e conectados, com a utilização de sensores de presença e iluminação sendo possível ajustar automaticamente as condições do ambiente, proporcionando conforto e eficiência energética com o ensino desses parâmetros (Sundmaeker et al., 2010).

Nesse modelo podemos notar que a china foi um dos primeiros países do mundo a utilizar a IoT no âmbito educacional, onde desde 2011 o governo vem atuando na estruturação das escolas para uma conectividade entre os ambientes por meio de rádio frequência, como também na adição desses conhecimentos nos componentes curriculares através de componentes eletrônicos para melhor interação e desenvolvimento desse modelo (Zuim, 2016).

Mais uma aplicação da IoT é o setor de automóveis, desde o seu crescimento de produção nos anos 90 e com o desenvolvimento de seus protótipos e com a utilização e desenvolvimentos de sensores e microcontroladores, os mesmos ganharam cada vez mais espaço na construção de veículos, além de trazer um maior leque de comunicação entre os componentes os mesmos trouxeram uma melhor análise e desenvolvimento desses veículos (Sindicato dos Metalúrgicos 2012).

Dessa forma segundo Zanni (2014) um dos primeiros movimentos sobre o tema foi feito pela FIAT, com seu modelo Fiat Linea que teve seu lançamento em 2009 e

teve como inovação um assistente de Gps. Alguns anos depois a Volkswagen lançou o Virtus um de seus modelos pioneiros na utilização da inteligência artificial em seus projetos.

### 3.3 Evolução dos Microprocessadores

Mediante a história dos microprocessadores podemos ver que é mostrado um cenário que é marcado por inovações que influenciaram profundamente a computação e a sociedade de modo geral. A trajetória começou desde o advento do pioneiro Intel 4004 lá atrás em 1971 até as arquiteturas de processadores dos dias atuais, onde esse desenvolvimento mostra uma narrativa complexa que ilustra a conexão entre avanços arquiteturais, design de circuitos e demandas tecnológicas e processadores cada vez mais desenvolvidos (Smith, 2015).

A década em questão que marca anos 70, mostra o início da revolução dos microprocessadores com o lançamento do Intel 4004, onde o mesmo se tratava de um dispositivo que incorporava cerca de 2.300 transistores em um chip e que proporcionava uma capacidade de processamento centralizada. A próxima década de 1980 viu de perto à expansão da computação pessoal com a introdução dos processadores Intel 8086 e 8088, que estabeleceram a arquitetura x86 como um padrão (Peterson et al., 2013).

De acordo com Lee (2011) dando continuidade ao desenvolvimento e diante anos de 1990 até 2000, possuíram como ponto chave o desenvolvimento de processadores que tiveram como função desenvolver a busca pelo desempenho em ambientes de computação cada vez mais complexos, dessa forma uma das grandes arquiteturas no caso da RISC ganhou ênfase, culminando em inovações como o SPARC, que desafiou os limites da eficiência de processamento. Assim foram projetados os processadores do tipo multi-core, como é o caso da série Intel Core, que otimizaram a multitarefa e a eficiência energética.

Os microprocessadores dos dias atuais ganharam destaque pela especialização e pela coexistência de processadores dedicados, como é o caso das GPUs e as NPUs. Esses dispositivos altamente especializados têm revolucionado domínios como a inteligência artificial e o processamento gráfico avançado, assim ao se analisar o desenvolvimento desses microprocessadores ao olhar para o futuro e com novas tecnologias sendo desenvolvidas, é notório o grande desafio que se vem pela frente, pois os mesmos enfrentam desafios de eficiência energética, bem

como os estudos da computação quântica, que tem grande potencial de redefinir a própria natureza do processamento (Smith, 2020).

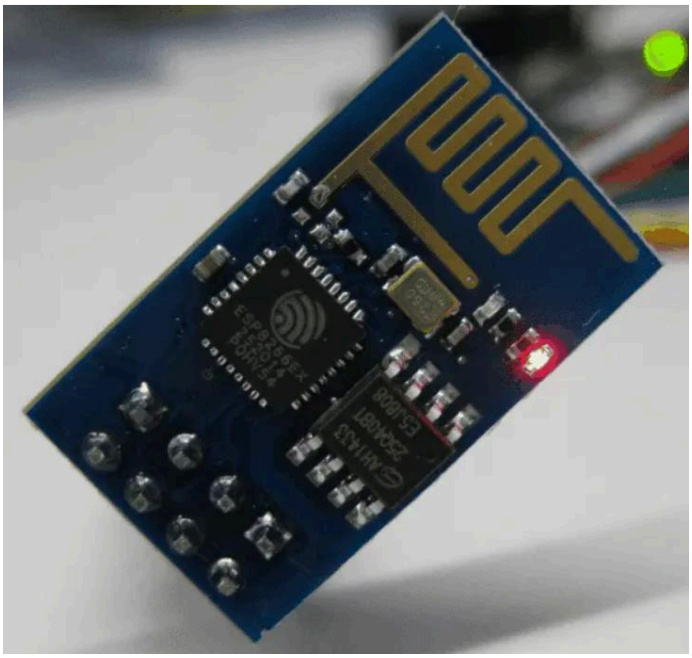
### 3.4 ESP32

Segundo Ibrahim (2017) o ESP é um robusto processador que pode ser empregado em diferentes áreas do conhecimento, principalmente no quesito da internet das coisas, assim o mesmo pode ter diversas versões e utilidades, como chegar a ter frequências de clock de até 240 MHz, contando com memorias e tantos outros periféricos.

Dessa forma é visto que um dos primeiros protótipos desenvolvidos pela empresa Espressif Systems foi o modulo ESP8266 que é composto por um System-On-Chip com Wi-Fi embutido, possui conectores GPIO, barramentos I2C, SPI, UART, entrada ADC, saída PWM e sensor interno de temperatura, CPU que opera em 80MHz, com possibilidade de operar em 160MHz, Arquitetura RISC de 32 bits, 32KBytes de RAM para instruções, entre outras especificações, onde suas versões foram desenvolvidas em 12 do ESP-01 até o ESP-12 mudando apenas o número de pinos de entrada e saída (GPIO) disponíveis, memória disponível e o espaçamento entre os pinos (Curvello, 2015).

Assim podemos ver na figura a seguir o modelo do modulo ESP8266 com algumas de suas características padrão;

**Figura 2 -** Modulo ESP8266 modelo ESP-01

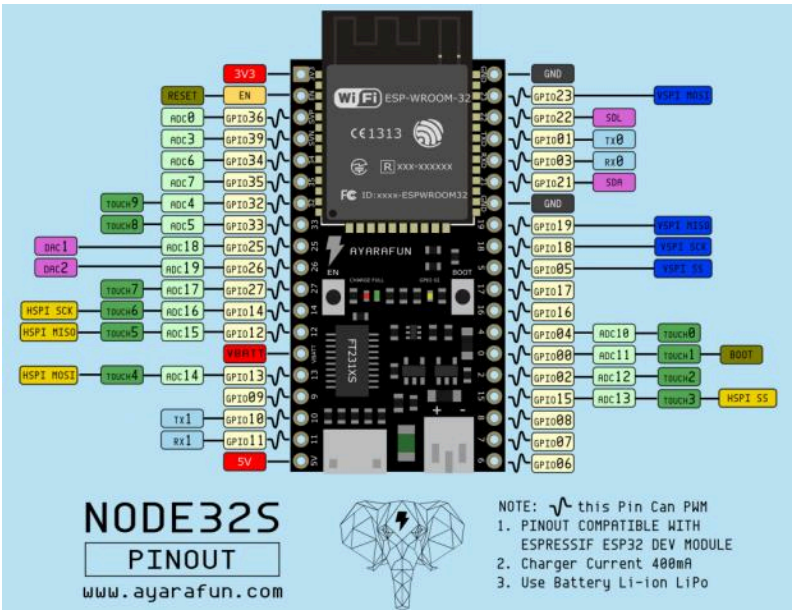


**Fonte:** Curvello (2015).

Diante o grande crescimento da IoT e mediante o grande sucesso que seus primeiros protótipos tiveram foram desenvolvidos pela Espressif Systems o sucessor do ESP8266, onde se teve um modelo mais robusto e com mais capacidade de processamento e conexão, tratando nesse quesito dos modelos ESP32, tendo como parâmetros em algumas de suas versões uma CPU Xtensa® Dual-Core 32-bit LX6, ROM 448 KBytes, RAM 520 Kbytes, clock máximo 240MHz, wireless padrão 802.11 b/g/n, conexão Wifi 2.4Ghz (máximo de 150 Mbps), antena embutida, Bluetooth BLE 4.2, além de trazer recursos para economia de energia e pinos de GPIO que têm maior precisão se comparados com outros microcontroladores (ESPRESSIF, 2018).

A figura 3 que será mostrada posteriormente irá trazer uma ilustração dos parâmetros referentes a um modelo ESP 32

**Figura 3 -** ESP-WROOM-32 e suas portas de entrada e saída




**Fonte:** Ayarafun (2017).

## 3.5 ESP32 – LoRa

A análise da evolução do ESP32 LoRa revela um percurso marcado pelo processo de inovação diante seus antecessores pois desempenharam um papel crucial no avanço das aplicações de Internet das Coisas (IoT) em ambientes de longo alcance e baixo consumo de energia. Desde a sua introdução como um módulo de comunicação sem fio de longo alcance até sua integração avançada como o ESP32 LoRaWAN (Navarro 2022).

De acordo com Navarro (2022) é visto que o protocolo LoRa é uma forma de modulação, onde sua sigla tem como significado Long Range, que significa redes de longas distancia que tem como pontos fundamentais o baixo consumo de energia, modulação simples, sua imunidade a ruídos e tem um baixo valor de aquisição, assim a mesma tem em seu diferencial a baixa taxa de transmissão de dados, sendo assim importante em comunicações remotas.

Sendo assim se tratando de dispositivos ESP 32 com módulo LoRa devemos analisar em primeiro momento a sua funcionalidade, onde o mesmo tem em sua essência a comunicação em distâncias superiores aos ESPs convencionais, além de poder se comunicar entre si, através de gateway onde o mesmo realiza comunicação com servidores da web e posteriormente armazenando e processando informações que podem ser visualizadas por aplicativos de celulares ou até computadores (Junior, 2016).

O ESP32 LoRa surge como um grande protótipo onde relaciona a IoT com os microprocessadores além de mecanismos tradicionais como Bluetooth e WiFi, dessa forma o mesmo surgiu como uma junção entre o ESP32 e o LoRa, tendo como público alvo projetistas que tem como finalidade projetos com distâncias maiores que os ESP 32 de modelos convencionais (Oliveira, 2020).

Para Oliveira (2020) quando tratamos de detalhar os elementos que compõem o ESP32 LoRa é visto que;

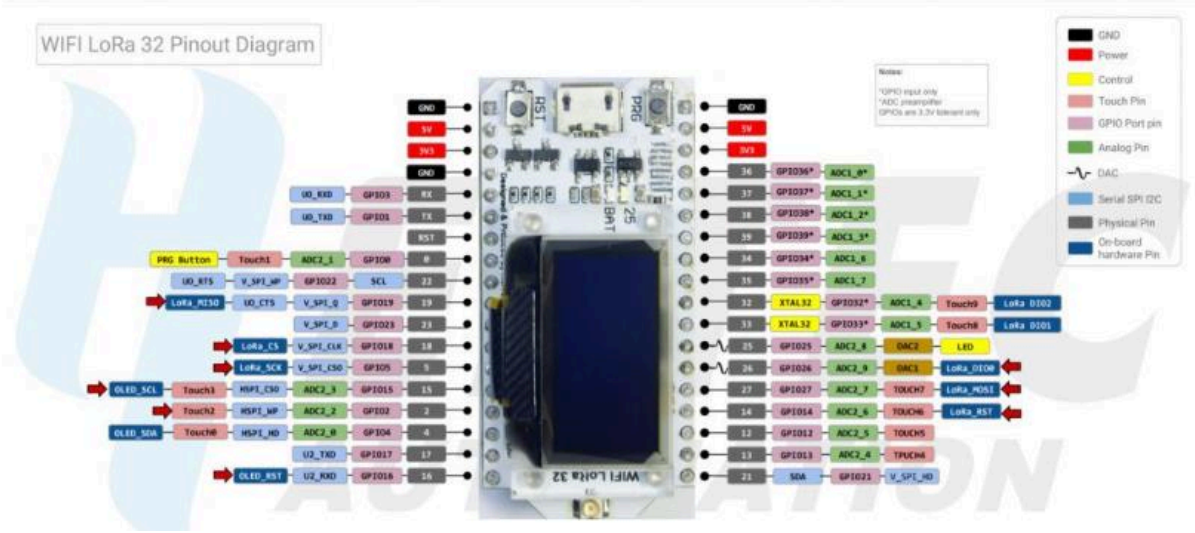
> “Possui um Display OLED de 0,96” 128X64 que permite exibir em tempo real diversas informações, logo, toda depuração de informações pode ser verificada sem o uso de uma ferramenta externa. Ao lado do display há um conector padrão UFL 2mm para conexão da antena. Na parte debaixo da placa há um conector JST para conexão de uma bateria Li-Ion ou Li-Po de 3.7V / 1000mAh para alimentação do módulo, além de uma frequência de operação do LoRa na Placa WiFi LoRa ESP32 pode ser de 433MHz, 868MHz ou 915MHz.”

Portanto um dos diferenciais desse tipo de projeto vem a ser a variabilidade de se conectar módulos, sensores na placa e, assim ter o ganho de dados e atuações de suas ações através de uma comunicação LoRa, além de se ter uma interface para se analisar seus resultados e que podem ser compartilhadas seja via bluetooth

ou WiFi e consequentemente ser mostrada em outros dispositivos como celulares e computadores (Oliveira, 2020).

Para uma melhor visualização será apresentado a seguir um diagrama da pinagem de forma prática do item mencionado anteriormente;

**Figura 4 -** Diagrama pinout.




**Fonte:** Teixeira (2019).

## 3.6 Sensores e sensoriamento

Diante pensamento de Wendling (2010) sensores são definidos como dispositivos sensíveis a qualquer formato de energia, seja ela luminosa, térmica ou cinética, que pode ser definida em diferentes grandezas como, temperatura, umidade, pressão, velocidade, corrente e aceleração, sendo dessa forma um dispositivo de grande utilidade para inúmeras finalidades.

Assim é visto que modalidade de sensoriamento é fundamental quando se trata de projetos de monitoramento de informações em ambientes, sendo que o papel dos sensores é buscar informações para serem repassadas a um circuito eletrônico, com isso sendo de suma importância a sistemas de automação pois além de facilitar o processo de informação entre os componentes e objetos (Wendling, 2010).

Segundo Patsko (2006) os sensores analógicos surgiram com o pressuposto de análise de sinais analógicos, onde esses dispositivos medem grandezas físicas de um ambiente ou de um sistema e dessa forma convertem essas medições em sinais elétricos que podem variar em uma faixa continua ou em uma faixa especifica

de medição, além de serem amplamente utilizados em diferentes aplicações como processos industriais, sistemas de automação, monitoramento ambiental, dispositivos médicos, aplicações automotivas, sistemas de segurança através de grandezas de temperatura, pressão, umidade, pressão entre outras grandezas, como também ter uma maior precisão em suas medições, caso de um termômetro de temperatura de mercúrio, que em sua saída analógica pode ser vista ao olho humano e ser analisada de forma prática.

Em outra análise agora tratando de sensores digitais podemos tratar esses componentes como sistemas que convertem informações de grandezas físicas em sinais, onde geralmente são mencionados entre 0s e 1s, onde sua conversão acontece por meio da utilização de circuitos eletrônicos e posteriormente é feita a codificação e é visto em suas saídas sinais digitais (Patsko, 2006).

De acordo com Wendling (2010) as aplicações de sensores digitais em diferentes ambientes são identificadas com muita fluidez em industrias automotivas, sistemas relacionados a medicina, casas inteligentes e eletrônica de consumo, além de apresentar alguns benefícios como imunidade a ruídos, facilidade de processamento e a sua compatibilidade com sistemas digitais.

Um dos sensores que tem grande destaque no quesito medição de temperatura e umidade é o sensor DHT11, isso se dá pela sua simplicidade ou custo de aquisição, além de seu baixo consumo enérgico, o mesmo funciona em comunicação serial, tem um sinal de transmissão de até 20 metros, sendo assim a tabela 1 a seguir irá mostrar algumas de suas especificações (Tavares, 2013).

**Tabela 1 -** Especificações sensor DHT11

| ITEM | DHT11 |
|---|---|
| **RANGER** | 20-90%RH 0-50°C |
| **PRECISÃO UMIDADE** | (+-5%RH) |
| **PRECISÃO TEMPERATURA** | (+-2 °C) |
| **RESOLUÇÃO** | 1 |
| **PINOS** | 4 |

**Fonte:** Datasheet DHT11 AOSONG (2022).

## 3.7 Plataforma de desenvolvimento Arduino IDE

Ao se fazer uma breve análise sobre a plataforma de desenvolvimento Arduino IDE é notória a sua desenvoltura durante os anos, além de um papel

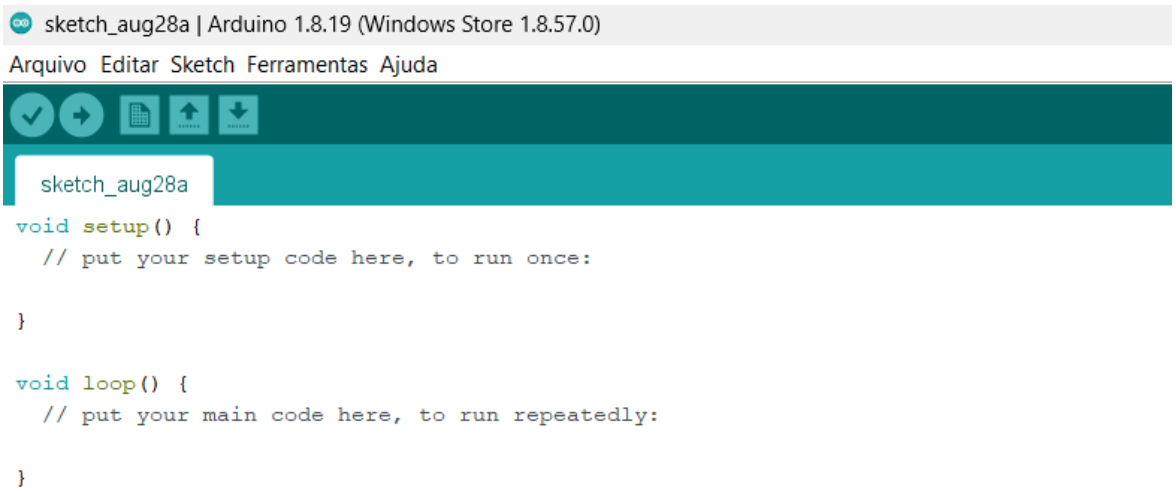
fundamental no desenvolvimento de projetos de dispositivos eletrônicos, assim a mesma é vista como uma ferramenta de código aberto, além de ser bastante acessível ao público, e assim ressaltando a junção entre, comunidade, tecnologia e acessibilidade (Mcroberts 2010).

Segundo Margolis (2011) a plataforma de desenvolvimento Arduino IDE teve como um de seus principais fundadores o italiano Massino Banzi, a mesma surgiu como uma ideia para simplificar a programação e desenvolvimento de dispositivos eletrônicos, pela sua facilidade e acessibilidade e uma abordagem de fácil acesso feita para criar projetos de fácil desenvolvimento e assim quebrando estigmas que se tinha em relação a entrada da eletrônica e programação.

A plataforma mencionada anteriormente funciona como um conjunto de informações que facilita a programação a compilação e o desenvolvimento de projetos o mesmo tem como finalidade alguns passos importantes que envolvem a escolha de placas, escrita de código, edição e depuração, compilação, upload de códigos e por fim a execução do programa além de muitas outras aplicações, a plataforma também abriu espaços para uma melhor acessibilidade com a IoT com dispositivos eletrônicos com diferentes meios e aplicabilidades (Abreu, 2012).

Dessa forma a seguir podemos ver a interface básica da plataforma Arduino IDE, para uma melhor visualização do que foi mencionado anteriormente.

**Figura 5 -** Interface da plataforma de desenvolvimento Arduino IDE




**Fonte:** Abreu (2012).

### 3.8 Veículos Baja

Ao passar dos anos é notório o grande desenvolvimento da indústria automotiva no mundo todo, dessa forma as universidades e as linhas de pesquisa são um fator primordial para esse desenvolvimento, dessa forma em 1976 na universidade da Califórnia do Sul, surgiram as primeiras linhas de pesquisa sobre os veículos do tipo Off Rolder que tinha como finalidade competições do tipo Baja SAE, que é relacionada a um competição de protótipos de carros desenvolvidos por estudantes nas universidades, tendo como objetivo a simulação de conceitos correlacionados a engenharia de mundo real (SAE Brasil, 2012).

No brasil esse tipo de competição é organizado pela SAE Brasil, que tem como função a realização de competição em todo território nacional, organizado em etapas nacionais e regionais, tendo critérios para as universidades e seus estudantes para criação de seus protótipos (SAE Brasil, 2012).

Com isso os protótipos são desenvolvidos por estudantes de engenharia do ensino superior que utilizam conceitos teóricos adquiridos em sala de aula da universidade, para produção de um veículo do tipo off-road de competição e ao mesmo tempo coloca o aluno no mercado de trabalho, correlacionado conceitos mecânicos, hidráulicos e elétricos (SAE Brasil, 2021).

Assim podemos ver posteriormente um dos modelos do tipo off-road em competição organizada pela SAE Brasil;

**Figura 6 -** Veiculo do tipo off-road em competição BAJA


**Fonte:** SAE Brasil (2021).

# 4 MATERIAIS E METODOS

Em busca de novos conhecimentos e aplicações relacionados à aplicabilidade da telemetria em veículos do tipo baja, este capítulo apresenta todos os materiais e métodos utilizados no presente trabalho. Todos os procedimentos foram conduzidos com base no datasheet do sensor de temperatura e umidade DHT11, no datasheet do módulo ESP32 LoRa e na plataforma de desenvolvimento Arduino IDE. Esses procedimentos foram executados no LAMEP - Laboratório de Automação, Microcontroladores e Eletrônica de Potência da Universidade Federal Rural do Semi-Árido (UFERSA), Campus Caraúbas.

O projeto mencionado anteriormente será dividido em várias etapas. Primeiramente, será realizado um estudo teórico abrangente relacionado à telemetria na utilização de sistemas embarcados. Em seguida, será feita a idealização de toda a parte de codificação na plataforma Arduino IDE. Posteriormente, serão realizados os devidos testes para verificar a comunicação entre os sensores e o ESP32. Por fim, os testes em campo serão conduzidos no veículo do tipo off-road baja

## 4.1 Equipamentos

Foram utilizados alguns equipamentos que foram cedidos pela UFERSA Campus Caraúbas, pelo projeto CARAUBAJA SAE como também materiais de posse do próprio aluno, os quais são apresentados na Tabela 2 juntamente com sua marca/modelo:

**Tabela 2 -** Equipamentos utilizados durante o estudo para produção do projeto em questão

| ITEM | EQUIPAMENTOS | MARCA/MODELO |
|---|---|---|
| **1** | COMPUTADOR | Lenovo / Ideapad3 |
| **2** | ESP32 - LoRa | Quimis / Q222T216 |
| **3** | SENSOR DHT11 | Akso / AK88 |
| **4** | PROTOBOARD | Del Lab / DLT-WV |
| **5** | VEÍCULO BAJA | Sem Marca / ATC |
| **6** | ANTENA EXTERNA | Sem Marca |

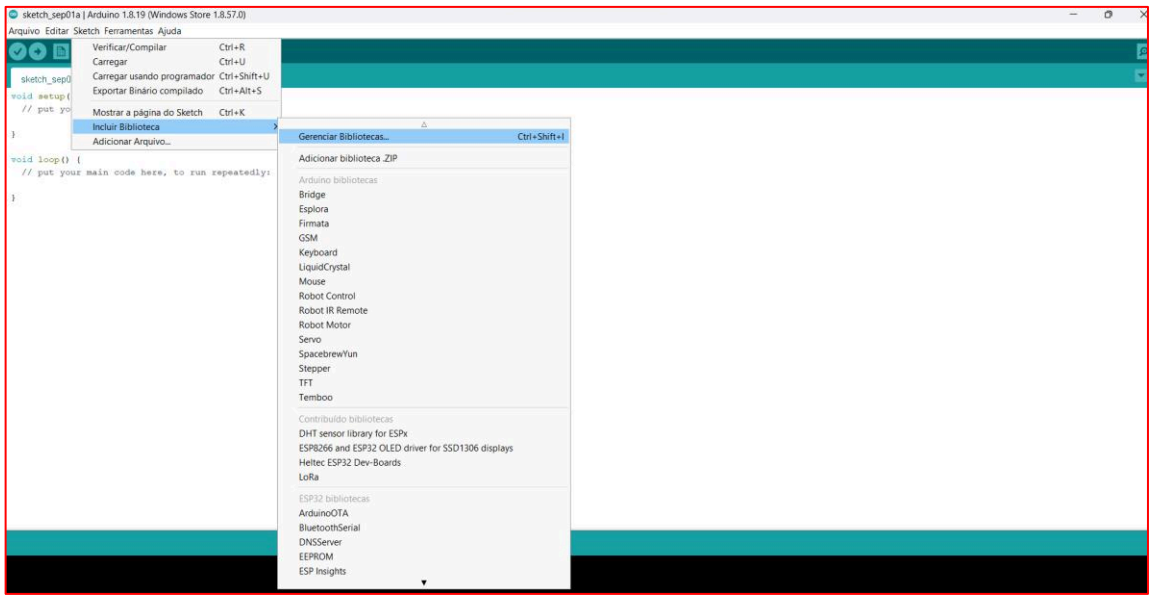
Fonte: o autor

## 4.2 Inclusão das bibliotecas ESP32 e DHT11 na plataforma Arduino IDE

A inclusão e instalação das bibliotecas ESP32 e DHT11 na plataforma de desenvolvimento foram os primeiros procedimentos realizados no projeto. O objetivo desta etapa foi simplificar e agilizar o desenvolvimento do projeto, economizando tempo e recursos, e garantindo a funcionalidade adequada. A seguir, foi realizado o procedimento em questão.

(a) Caminho para Instalação de bibliotecas;

**Figura 7 -** Caminho para Instalação de bibliotecas

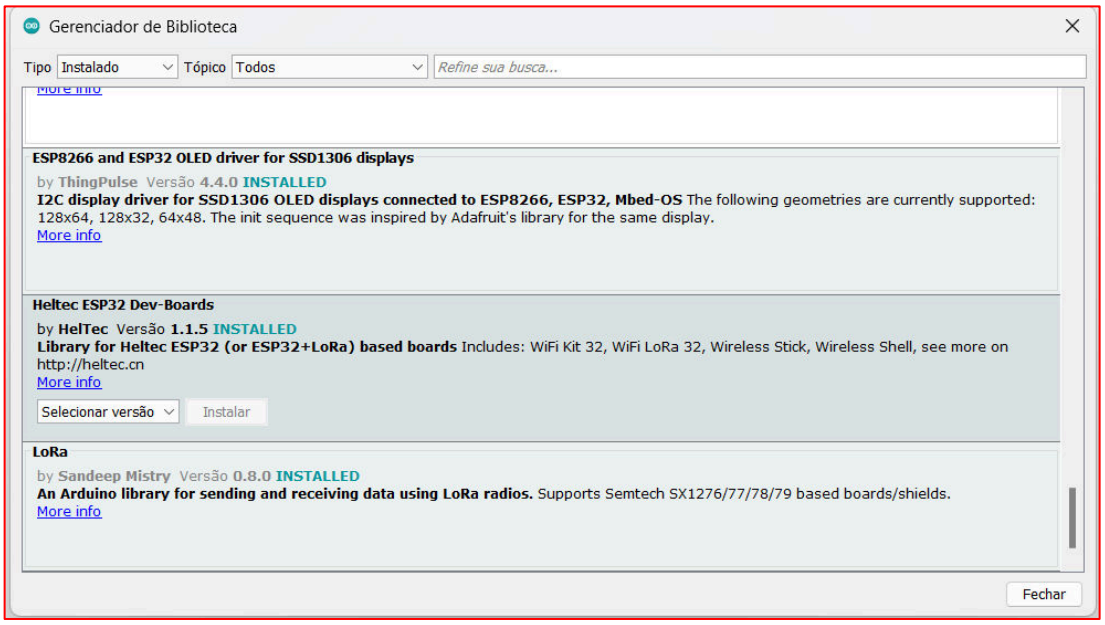



**Fonte:** Autor próprio (2023).

Nesse momento é demonstrado o caminho para se adicionar as bibliotecas na plataforma de desenvolvimento Arduino IDE. Onde passa pelo caminho Sketch=> Gerenciador de bibliotecas.

(b) Inclusão da biblioteca da Heltec ESP32 Dev-Boards;

**Figura 8 -** Heltec ESP32 Dev-Boards

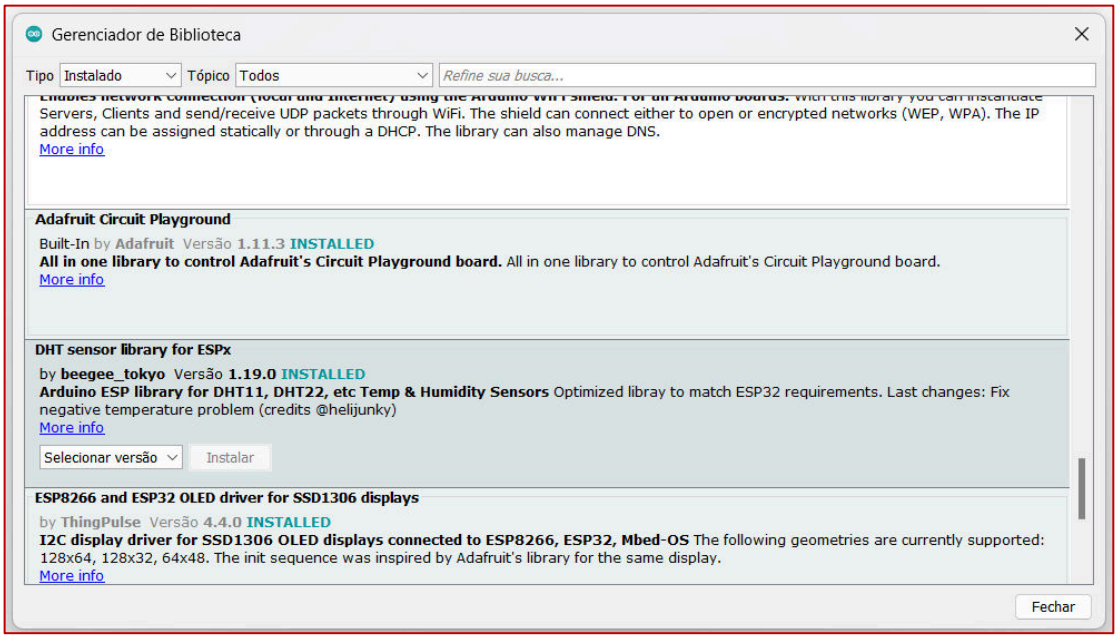



**Fonte:** Autor próprio (2023).

A seguinte biblioteca em questão é uma biblioteca especifica para placas de desenvolvimento da Heltec que tem como base o ESP32, assim a mesma fornece suporte e abstração para o hardware especifico e uma melhor inclusão com os dispositivos e a variedade da IoT.

(c) Inclusão da biblioteca DHT sensor library for ESPx;

**Figura 9 -** DHT sensor library for ESPx

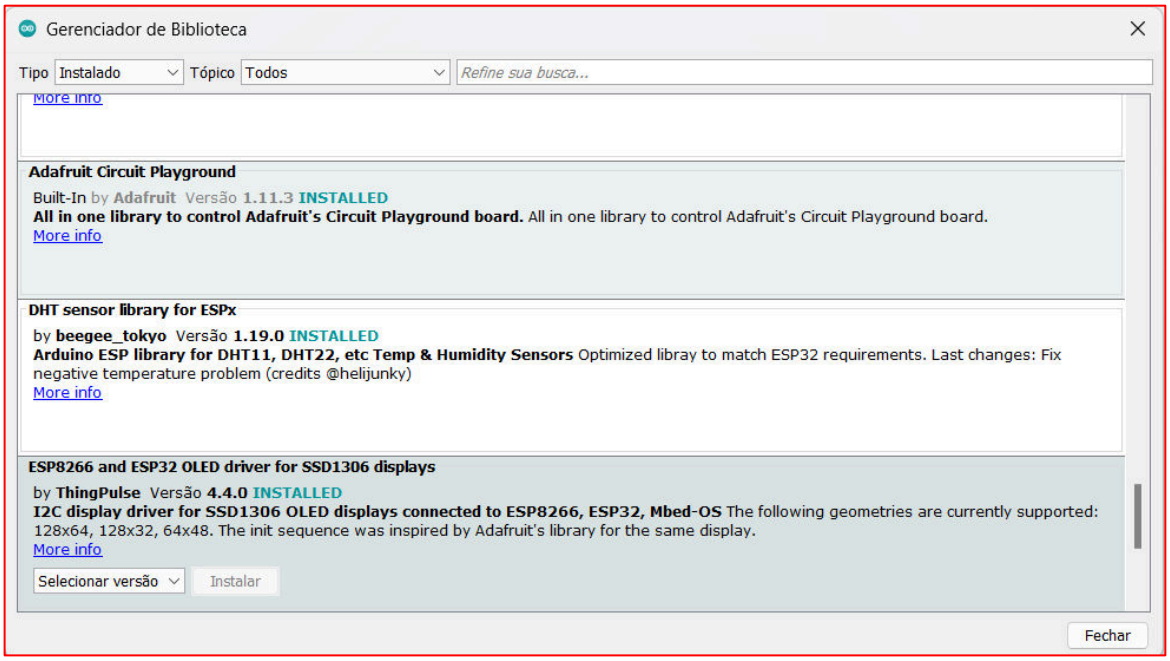



**Fonte:** Autor próprio (2023).

Dessa forma a biblioteca mencionada anteriormente simplifica o processo de leitura de dados em sensores de temperatura e umidade DHT que são utilizados em placas do tipo ESP8266 ou ESP32, a mesma é de suma importância em projetos de monitoramento que envolvem sensores DHT.

(d) Incluussão da biblioteca ESP8266 and ESP32 OLED driver for SSD1306 displays;

**Figura 10 -** ESP8266 and ESP32 OLED driver for SSD1306 displays




**Fonte:** Autor próprio (2023).

A biblioteca em questão tem como fornecer um suporte para comunicação e controle de displays OLED baseados no controlador SSD1306, quando usados com placas ESP8266 e ESP32 na plataforma Arduino IDE. O SSD1306 é um controlador bastante utilizado em displays de baixo consumo de energia.

## 4.3 Desenvolvimento do Código em C++

### 4.3.1 Código Transmissor

De forma inicial foram analisadas pesquisas de modo geral para a elaboração dos códigos fontes, onde em primeiro momento foi feita a confecção do código transmissor, assim através da plataforma de desenvolvimento Arduino IDE, foram dados os primeiros passos para o desenvolvimento do projeto.

Portanto a primeira parte do código se deu com as primeiras duas linhas incluem as bibliotecas necessárias para o programa em questão, heltec.h e para o módulo Heltec ESP32 com LoRa, enquanto DHTesp.h é para o sensor DHT11, além disso, é definida uma constante BAND com o valor da frequência LoRa como 915 MHz, como é mostrado na figura a seguir;

**Figura 11 -** Desenvolvimento inicial do código transmissor

```
#include "heltec.h" // Inclui a biblioteca Heltec para o ESP32 com LoRa
#include "DHTesp.h" // Inclui a biblioteca DHTesp para o sensor DHT11

#define BAND 915E6 // Define a frequência LoRa como 915 MHz

DHTesp dht; // Cria um objeto DHTesp para o sensor DHT11
```

**Fonte:** Autor próprio (2023).

Já o *void sendpacket (float temperature, float humidity)* é um protótipo de função, onde ele fornece uma declaração adiantada de como a função sendPacket deve ser chamada, como também, quais tipos de argumentos ela espera e qual é o tipo de retorno da função.

A função *setup* é uma função que é executada uma vez durante a inicialização, sendo que a mesma configura o LED, a comunicação serial, o módulo Heltec LoRa, o display OLED e o sensor DHT11, o display é usado para exibir "INICIADO" e "TCC SÁVIO!" durante a inicialização.

**Figura 12 -** Desenvolvimento do código transmissor função SendPacket e Setub

```
void sendPacket(float temperature, float humidity); // Protótipo da função para enviar dados via LoRa

void setup() {
  pinMode(LED, OUTPUT); // Configura o pino do LED como saída
  Serial.begin(9600); // Inicializa a comunicação serial com uma taxa de 9600 bps
  Heltec.begin(true, true, true, true, BAND); // Inicializa o módulo Heltec ESP32 LoRa com a configuração especificada,
  Heltec.display->init(); // Inicializa o display OLED
  Heltec.display->flipScreenVertically(); // Inverte a tela verticalmente (opcional, dependendo da montagem do display)
  Heltec.display->setFont(ArialMT_Plain_10); // Define a fonte para o texto como Arial, tamanho 10
  Heltec.display->clear(); // Limpa o display OLED
  Heltec.display->drawString(33, 5, "INICIADO"); // Desenha o texto "INICIADO" na posição (33, 5) do display
  Heltec.display->drawString(10, 30, "TCC SÁVIO!"); // Desenha o texto "TCC SÁVIO!" na posição (10, 30) do display
  Heltec.display->display(); // Atualiza o display
  delay(5000); // Aguarda 1 segundo
  dht.setup(17, DHTesp::DHT11); // Inicializa o sensor DHT11 no pino 17
}
```

**Fonte:** Autor próprio (2023).

Dando continuidade, à função loop é executada continuamente após a inicialização, a mesma tem como utilidade lê a temperatura e a umidade do sensor DHT11, exibindo esses valores no display OLED e envia os dados via LoRa usando a função sendPacket(), dessa forma o programa então aguarda 5 segundos antes de repetir todo processo novamente, como podemos ver a seguir;

**Figura 13 -** Desenvolvimento do código transmissor função loop

```
void loop() {
  float currentTemp = dht.getTemperature(); // Lê a temperatura atual do sensor DHT11
  float currentHumidity = dht.getHumidity(); // Lê a umidade atual do sensor DHT11

  Heltec.display->clear(); // Limpa o display OLED
  Heltec.display->setTextAlignment(TEXT_ALIGN_LEFT); // Define o alinhamento do texto como à esquerda
  Heltec.display->setFont(ArialMT_Plain_10); // Define a fonte para o texto como Arial, tamanho 10

  Heltec.display->drawString(30, 5, "TCC SÁVIO"); // Desenha o texto "TCC SÁVIO" na posição (30, 5) do display
  Heltec.display->drawString(33, 30, String(currentTemp)); // Exibe a temperatura atual na posição (33, 30) do display
  Heltec.display->drawString(78, 30, "°C"); // Exibe o símbolo de grau Celsius na posição (78, 30) do display
  Heltec.display->drawString(33, 50, String(currentHumidity)); // Exibe a umidade atual na posição (33, 50) do display
  Heltec.display->drawString(78, 50, "%"); // Exibe o símbolo de porcentagem na posição (78, 50) do display
  Heltec.display->display(); // Atualiza o display OLED

  sendPacket(currentTemp, currentHumidity); // Chama a função para enviar os dados via LoRa
  delay(5000); // Aguarda 5 segundos antes de enviar novamente
}
```

**Fonte:** Autor próprio (2023).

E por fim é utilizada a função sendPacket que é responsável por criar um pacote de dados que contém a temperatura e a umidade, dessa forma, formata os valores com texto descritivo e é enviado via comunicação LoRa, esses dados podem ser recebidos por outro dispositivo configurado para receber na mesma frequência e processar as informações da maneira desejada, caso do outro ESP32 LoRa que ficara responsável por receber dados mencionados.

**Figura 14 -** Desenvolvimento final do código transmissor

```
void sendPacket(float temperature, float humidity) {
  LoRa.beginPacket(); // Inicia um novo pacote LoRa
  LoRa.print("Temp: "); // Envia a string "Temp: "
  LoRa.print(temperature); // Envia a temperatura
  LoRa.print("°C, Umidade: "); // Envia a string ", Umidade: "
  LoRa.print(humidity); // Envia a umidade
  LoRa.print("%"); // Envia o símbolo de porcentagem
  LoRa.endPacket(); // Finaliza o pacote LoRa
}
```

**Fonte:** Autor próprio (2023).

### 4.3.2 Código Receptor

Dando continuidade no desenvolvimento do projeto, foi feito o código fonte para o ESP32 LoRa receptor, em primeiro momento foi adicionado de forma similar ao código transmissor a biblioteca heltec.h, que inclui a biblioteca Heltec, onde a mesma é usada para controlar o módulo Heltec ESP32 com suporte a LoRa.

É visto também a adição da constante chamada BAND, aliás é definida com o valor da frequência LoRa, que é configurada para 915 MHz, de forma similar ao dispositivo transmissor, além disso é de suma necessidade a implementação das variáveis de string, temperatureString e humidityString, onde as mesmas são declaradas para armazenar os valores de temperatura e umidade, assim é visto na figura a seguir tais processos mencionados;

**Figura 15 -** Desenvolvimento inicial do código receptor

```
#include "heltec.h" // Inclui a biblioteca Heltec para o ESP32 com LoRa

#define BAND 915E6 // Define a frequência LoRa como 915 MHz

String temperatureString = ""; // String para armazenar a temperatura
String humidityString = "";    // String para armazenar a umidade
```

**Fonte:** Autor próprio (2023).

Assim no desenvolver do código em questão é de suma importância adicionar dois protótipos de função e declaralos são eles o LoRaDataPrint, que será usado para atualizar o display e cbk que será usado para tratar dados recebidos via LoRa, é adicionado também a função LoRaDataPrint, que atualiza o display OLED com os valores de temperatura e umidade.

**Figura 16 -** Desenvolvimento do código receptor

```
void LoRaDataPrint() {
  Heltec.display->clear(); // Limpa o display OLED
  Heltec.display->setTextAlignment(TEXT_ALIGN_CENTER); // Define o alinhamento do texto como centralizado
  Heltec.display->setFont(ArialMT_Plain_10); // Altera o tamanho da fonte para 10 pontos

  // Exibe a temperatura e a umidade com as iniciais "TEMP" e "UMID" no centro do display
  Heltec.display->drawString(Heltec.display->width() / 2, 10, "TEMP: " + temperatureString);
  Heltec.display->drawString(Heltec.display->width() / 2, 30, "UMID: " + humidityString);

  Heltec.display->display(); // Atualiza o display OLED
}

void cbk(int packetSize) {
  String receivedData = "";

  for (int i = 0; i < packetSize; i++) {
    receivedData += (char)LoRa.read();
  }

  // Use o método indexOf para encontrar as posições de "Temp: " e ", Umidade: "
  int tempPosition = receivedData.indexOf("Temp: ");
  int humidityPosition = receivedData.indexOf(", Umidade: ");

  if (tempPosition != -1 && humidityPosition != -1) {
    // Obtém os valores de temperatura e umidade a partir dos índices encontrados
    temperatureString = receivedData.substring(tempPosition + 6, humidityPosition);
    humidityString = receivedData.substring(humidityPosition + 11);

    LoRaDataPrint(); // Atualiza o display com os novos valores
  }
}
```

**Fonte:** Autor próprio (2023).

As próximas linhas a seguir trata-se da função void setup que é o início do programa, onde é definido a configuração inicial do dispositivo, assim ela é executada uma única vez quando o dispositivo é ligado ou reiniciado, é de respaldo também que neste bloco de código, é configurado os pinos, iniciado a comunicação serial, é definido as configurações iniciais e preparado o ambiente de trabalho para o programa principal, em resumo, é onde o dispositivo se prepara para funcionar corretamente antes de entrar no ciclo principal de execução.

Dessa forma a função mencionada é uma parte de suma importância do código, pois é onde você realiza todas as preparações necessárias para o funcionamento adequado do dispositivo antes que ele comece a executar suas tarefas principais no seu loop principal, essa etapa é essencial para garantir que o dispositivo esteja em um estado apropriado antes de começar a interagir;

**Figura 17 -** Desenvolvimento do código receptor

```
void setup() {
  pinMode(LED, OUTPUT); // Configura o pino do LED como saída
  Serial.begin(9600); // Inicializa a comunicação serial com uma taxa de 9600 bps
  Heltec.begin(true, true, true, true, BAND); // Inicializa o módulo Heltec ESP32 LoRa com a configuração es
  Heltec.display->init(); // Inicializa o display OLED
  Heltec.display->flipScreenVertically(); // Inverte a tela verticalmente (opcional, dependendo da montagem
  Heltec.display->setFont(ArialMT_Plain_10); // Define a fonte para o texto como Arial, tamanho 10
  Heltec.display->clear(); // Limpa o display OLED

  // Exibe o texto "TCC Sávio" centralizado no display
  Heltec.display->drawString(Heltec.display->width() / 2, Heltec.display->height() / 2 - 10, "TCC SÁVIO");
  Heltec.display->display(); // Atualiza o display
  delay(5000); // Aguarda 5 segundos para exibir o texto

  Heltec.display->clear(); // Limpa o display
  Heltec.display->display(); // Atualiza o display
  delay(1000); // Aguarda 1 segundo
  LoRa.receive(); // Configura o dispositivo LoRa para receber dados
}
```

**Fonte:** Autor próprio (2023).

E por fim a função que é o núcleo do programa, aqui é onde o código principal é executado continuamente, dessa forma a primeira linha declara uma variável chamada packetSize, que é inicializa com o resultado da função LoRa.parsePacket, esta função verifica se há um pacote LoRa recebido e retorna o tamanho do pacote, de forma resumida esta parte do código permite que o dispositivo ESP32 LoRa receba dados remotos, interprete esses dados e exiba as informações de temperatura e umidade no display OLED, além de indicar visualmente a recepção de pacotes LoRa através de um led

**Figura 18 -** Desenvolvimento final do código Receptor

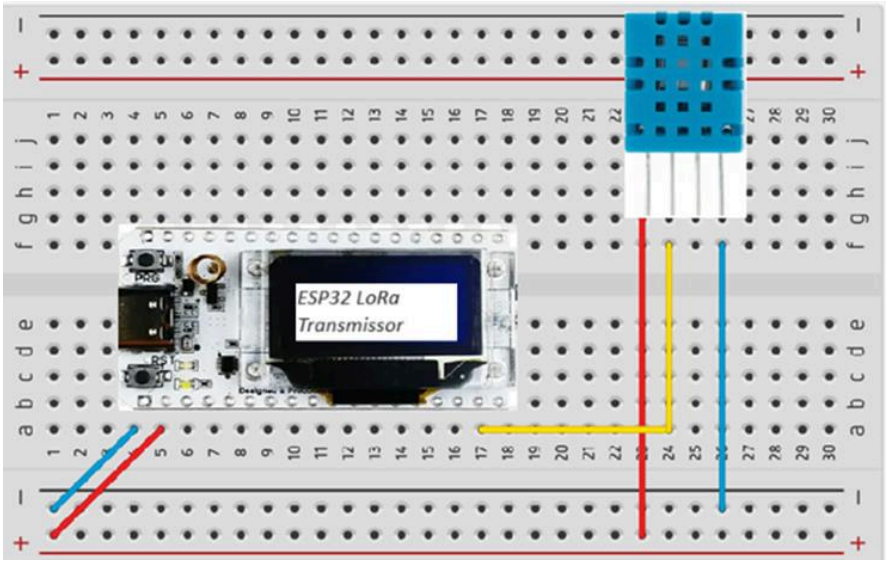
```
void loop() {
  int packetSize = LoRa.parsePacket(); // Verifica se há um pacote LoRa recebido
  if (packetSize) {
    cbk(packetSize); // Chama a função para tratar os dados recebidos via LoRa
    digitalWrite(LED, HIGH); // Liga o LED por 500ms
    delay(500);
    digitalWrite(LED, LOW); // Desliga o LED por 500ms
    delay(500);
    Serial.print("Recebendo a temperatura: ");
    Serial.print(temperatureString);
    Serial.println("°C");
    Serial.print("Recebendo a umidade: ");
    Serial.print(humidityString);
    Serial.println("%");
  }
  delay(10); // Aguarda 10ms antes da próxima iteração
}
```

**Fonte:** Autor próprio (2023).

## 4.4 Montagem do projeto em Protoboard

Quando tratamos de uma realização de determinado projeto em primeiro momento devemos fazer uma análise sobre os seus componentes e principalmente sua montagem, dessa forma de forma inicial vamos realizar a montagem do dispositivo de transmissão que é composto com um ESP32 LoRa, um sensor de temperatura DHT11, que é responsável pela medição da temperatura e umidade, uma protoboard e alguns Jumpers, assim tal procedimento é dimensionado da seguinte forma;

**Figura 19 -** Montagem do protótipo transmissor

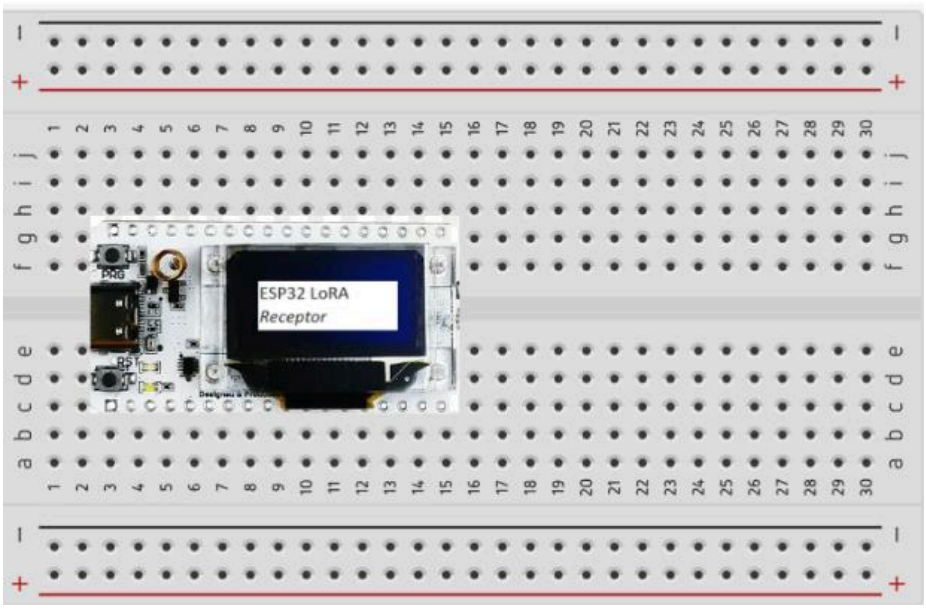



**Fonte:** Autor próprio (2023).

Assim é exemplificado na figura anterior a referida montagem, onde é possível ver no ESP32 LoRa que no pino 4 é composto o Gnd, o pino 5 o Vcc e no pino 17 a transmissão dos sinais, no componente DHT11 é visto que de forma igualitária ao ESP32, o pino 1 é composto pelo Vcc, o pino 4 o Gnd e o pino 2 a transferência de sinais.

Já o protótipo do receptor é bem mais simples que o anterior, visto que o mesmo não necessita de tantos componentes, apenas do ESP32 LoRa e uma protoboard para fixação, tal componente será utilizado apenas para recepção dos dados e sua visualização no display OLED e a utilização do seu Led para demonstrar a transmissão de dados entre os dispositivos.

**Figura 20 -** Montagem do protótipo Receptor




**Fonte:** Autor próprio (2023).

## 4.5 Utilização de uma antena externa

A fim de melhorias ao se analisar os sinais transmitidos, foram utilizadas duas antenas de uso externo para uma melhor transmissão entre os protótipos, dessa forma em primeiro momento foram realizadas algumas analises em relação a essas antenas, já que as mesmas não tinham nenhum dado especificado, assim foi realizado esse estudo para uma melhor avaliação e utilização no projeto;

**Figura 21 -** Antenas de uso externo

**Fonte:** Autor próprio (2023).

Dessa forma, as antenas utilizadas seguiram o padrão da figura anterior. Ao realizar as análises com o Vector Network Analyzer (VNA), que é um software de teste eletrônico usado para medir as características elétricas de dispositivos de RF (Radiofrequência), como antenas, filtros, cabos, circuitos integrados e outros componentes de alta frequência, foi possível observar que as antenas apresentam um casamento na faixa de 900 MHz e um RL (Return Loss) em decibéis (dB) abaixo de -10 dB. Esse valor indica uma boa correspondência de impedância entre os componentes do sistema, o que é desejável em muitas aplicações. Essa correspondência de impedância ajuda a minimizar a reflexão de energia de volta à fonte, reduzindo perdas e garantindo uma transmissão eficiente de sinais em sistemas de comunicação e transmissão de energia elétrica. A figura a seguir ilustra essas análises.

**Figura 22 -** Análise pelo Vector Network Analyzer (VNA) das características das antenas


**Fonte:** Autor próprio (2023).

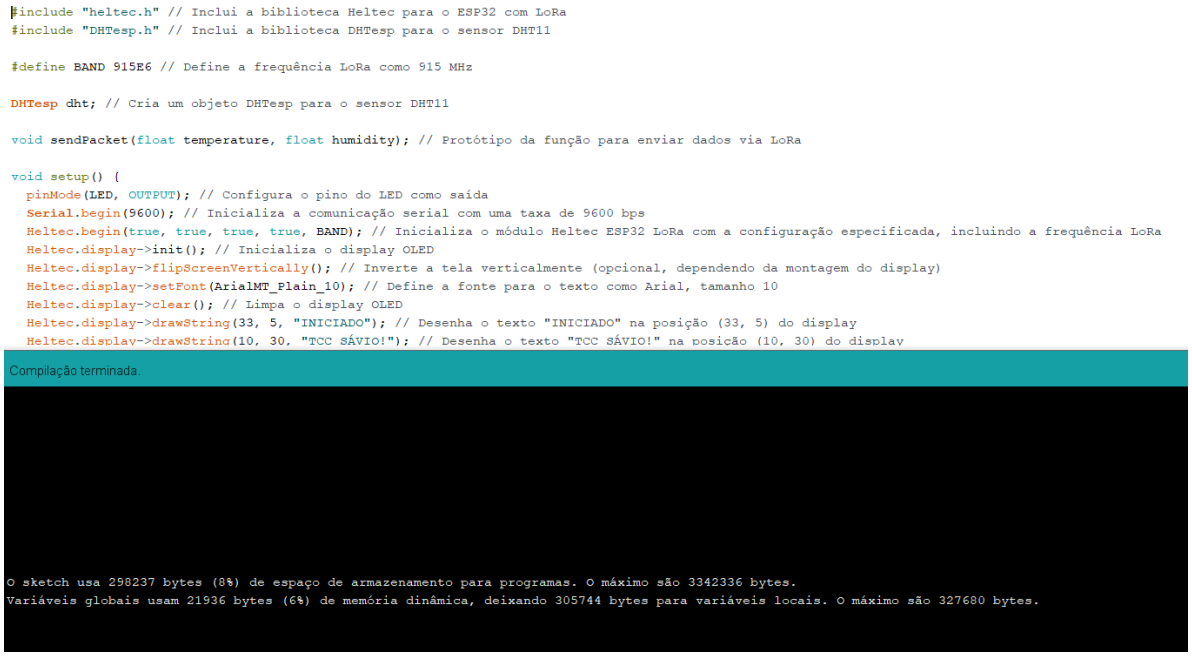
# 5 RESULTADOS E DISCUSSÃO

Este capítulo trará todos os resultados obtidos com a realização da pesquisa e as discussões que podem ser feitas acerca dos comportamentos observados.

## 5.1 Funcionamento dos códigos fontes e verificação de conexão

O primeiro passo realizado foi o teste de funcionamento de cada código individualmente, avaliando inicialmente sua eficiência na plataforma de desenvolvimento Arduino IDE. Nas figuras a seguir, será demonstrado esse procedimento;

**Figura 23 -** Testagem do Código transmissor




**Fonte:** Autor próprio (2023).

**Figura 24 -** Testagem do Código receptor

```
#include "heltec.h" // Inclui a biblioteca Heltec para o ESP32 com LoRa

#define BAND 915E6 // Define a frequência LoRa como 915 MHz

String temperatureString = ""; // String para armazenar a temperatura
String humidityString = "";    // String para armazenar a umidade

void LoRaDataPrint(); // Protótipo da função para atualizar o display
void cbk(int packetSize); // Protótipo da função para tratar dados recebidos via LoRa

void LoRaDataPrint() {
  Heltec.display->clear(); // Limpa o display OLED
  Heltec.display->setTextAlignment(TEXT_ALIGN_CENTER); // Define o alinhamento do texto como centralizado
  Heltec.display->setFont(ArialMT_Plain_10); // Altera o tamanho da fonte para 10 pontos

  // Exibe a temperatura e a umidade com as iniciais "TEMP" e "UMID" no centro do display
```

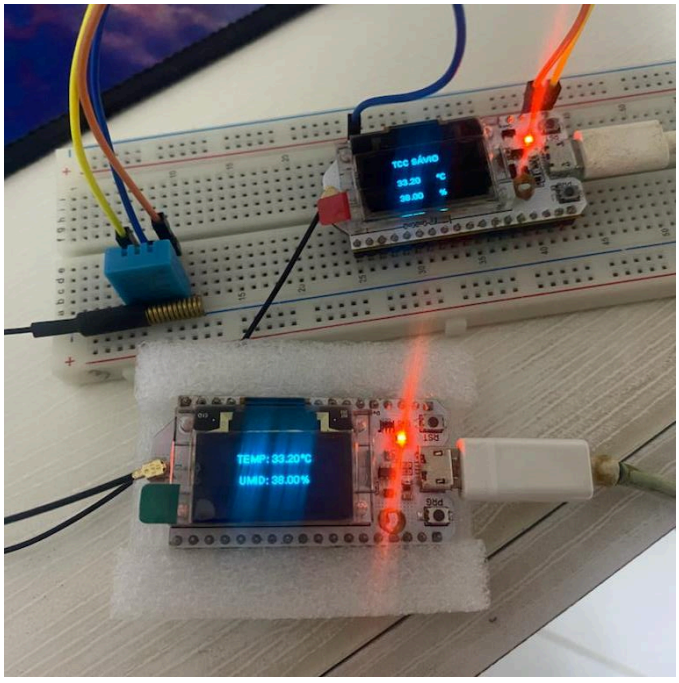
```
Compilação terminada.

O sketch usa 297141 bytes (8%) de espaço de armazenamento para programas. O máximo são 3342336 bytes.
Variáveis globais usam 21904 bytes (6%) de memória dinâmica, deixando 305776 bytes para variáveis locais. O máximo são 327680 bytes.
```

**Fonte:** Autor próprio (2023).

Como podemos observar, ambos os códigos demonstraram funcionamento satisfatório para a realização do projeto. Após essa confirmação, os programas correspondentes a cada dispositivo foram gravados. Posteriormente, realizou-se a verificação da comunicação entre os dispositivos ESP32 LoRa, como mostrado na figura a seguir;

**Figura 25 -** Verificação da comunicação entre os dispositivos



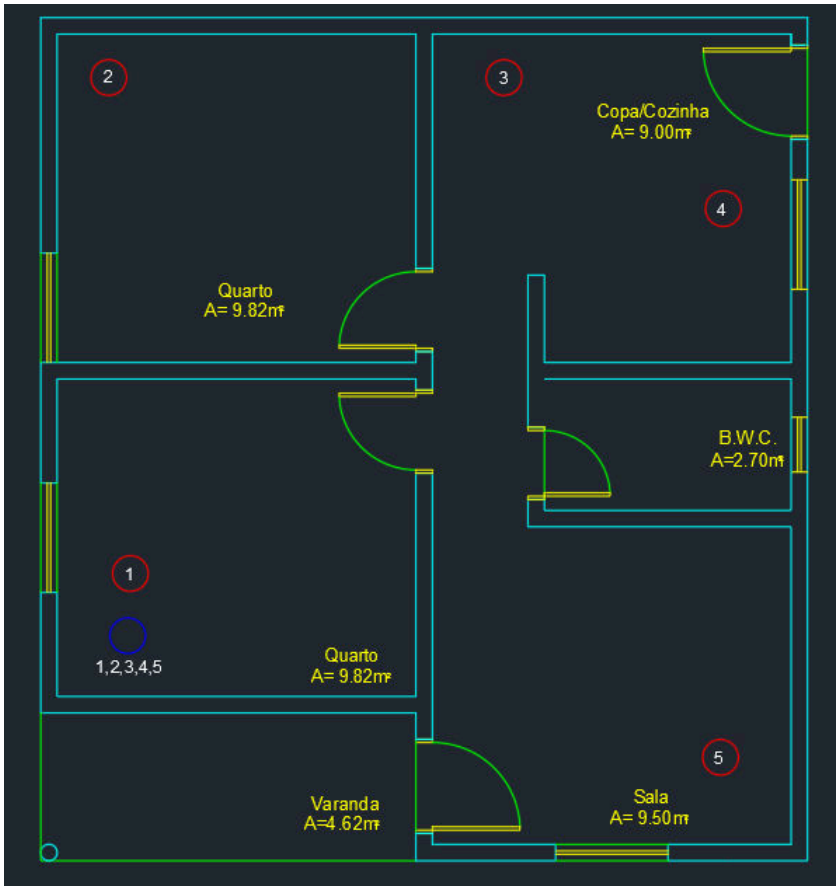

**Fonte:** Autor próprio (2023).

A partir da imagem anterior, foi possível observar que os dispositivos estavam se comunicando com sucesso, compartilhando as grandezas de temperatura e umidade. Além disso, vale destacar o acionamento do LED de cor branca, que demonstrava a troca de informações entre os equipamentos.

## 5.2 Verificação da comunicação entre os dispositivos em um ambiente casual

Continuando com a programação de verificação do projeto, foram realizados testes iniciais do funcionamento em alguns cômodos da casa. Em um cômodo ficaria o dispositivo transmissor, enquanto em outro estaria o receptor. Dessa forma, pudemos avaliar a distribuição em uma planta simples elaborada em CAD, usada apenas para fins didáticos no projeto. A simbologia em azul representa o transmissor, e em vermelho, o receptor, com um número na frente indicando o processo realizado;

**Figura 26 -** Verificação da comunicação entre os dispositivos em diferentes ambientes de uma residência




**Fonte:** Autor próprio (2023).

Ao analisar os pontos mencionados, observam-se algumas irregularidades nas comunicações. É importante ressaltar que esses experimentos foram realizados com as antenas padrão fornecidas com o ESP32 LoRa. Na primeira comunicação testada entre os pontos 1 e 1, a comunicação funcionou perfeitamente, pois os dispositivos estavam próximos um do outro. No entanto, nos demais pontos, não foi possível estabelecer a comunicação entre os dispositivos, uma vez que ao ultrapassar o cômodo onde o dispositivo transmissor estava localizado, o sinal era perdido imediatamente.

Torna-se evidente que, com as antenas de fábrica dos dispositivos ESP32 LoRa, as comunicações entre os dispositivos são fracas. Apenas os pontos 1 e 1 conseguiram realizar as trocas de dados, sendo necessário realizar ajustes para melhorar o funcionamento.

### 5.3 Verificação da comunicação com uma antena externa

Após verificar que as antenas padrão disponíveis nos dispositivos não atenderam às expectativas de desempenho, optou-se por usar uma antena externa para testar o projeto. Inicialmente, a antena externa foi conectada ao dispositivo transmissor, enquanto a do receptor permaneceu padrão. Iniciou-se a comunicação em uma pequena distância para fins de teste e, após o sucesso dessa etapa, partiu-se para testes em distâncias maiores.

O experimento foi conduzido no laboratório da UFERSA, Campus Caraúbas, onde o protótipo transmissor foi posicionado em uma sala, e o receptor percorreu vários percursos para avaliação. A figura a seguir ilustra o evento, com a ajuda do Google Earth, proporcionando uma melhor compreensão das distâncias entre os dispositivos em questão.

**Figura 27 -** Verificação da comunicação entre os dispositivos em diferentes locais no Campus da UFERSA Caraúbas


**Fonte:** Autor próprio (2023).

Com isso, vários pontos foram testados para a comunicação mencionada. A distância máxima alcançada com apenas uma antena no transmissor foi de aproximadamente 192 metros.

Da mesma forma, mas desta vez com ambas as antenas nos dois dispositivos, realizou-se o experimento. No entanto, por questões logísticas, os testes foram conduzidos na cidade de Felipe Guerra, RN. A lógica foi a mesma da realizada anteriormente, mas, desta vez, o dispositivo em trânsito foi o transmissor, e o receptor permaneceu fixo em um local, como mostrado a seguir.

**Figura 28 -** Verificação da comunicação entre os dispositivos em diferentes locais na Cidade de Felipe Guerra-RN na Avenida mira selva


**Fonte:** Autor próprio (2023).

**Figura 29 -** Verificação da comunicação entre os dispositivos em diferentes locais na Cidade de Felipe Guerra-RN no conjunto Braz Costa


**Fonte:** Autor próprio (2023).

**Figura 30 -** Verificação da comunicação entre os dispositivos em diferentes locais na Cidade de Felipe Guerra-RN no Sitio Tabuleiro, comunidade da Cidade


**Fonte:** Autor próprio (2023).

Assim, foram realizados diversos experimentos na cidade mencionada. Na Figura 28, foi feita uma análise em um ambiente aberto e com poucas casas durante uma determinada noite, o que proporcionou uma distância máxima de transmissão de dados de aproximadamente 925 metros.

Os experimentos da Figura 29 ocorreram no mesmo dia da Figura 28 e no mesmo horário, mas desta vez incluíram várias ruas com centenas de casas, todas no horário de pico. Nesse cenário, a distância máxima alcançada foi de aproximadamente 527 metros. Essa diferença pode estar relacionada aos bloqueios de sinal causados pelas numerosas casas presentes.

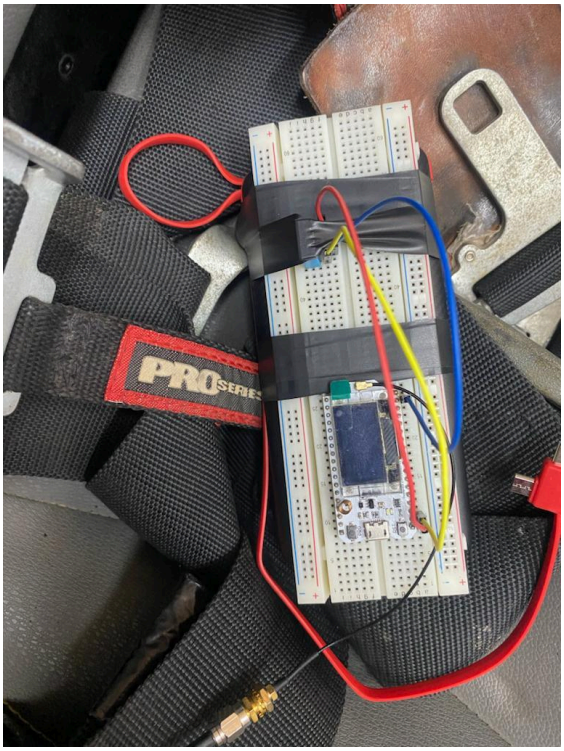
Por fim, a Figura 30 ilustra um teste de comunicação realizado em uma comunidade da cidade. O experimento ocorreu pela manhã e, devido à logística da comunidade com poucas casas e o horário não sendo o de pico, foi possível atingir uma maior distância, com um máximo de aproximadamente 1 quilômetro, tornando-se a maior distância registrada na conexão entre os protótipos.

Após esses ensaios e relacionando os conceitos adquiridos ao longo da pesquisa, chegou o momento de realizar a análise do projeto no veículo off-road da equipe Caraúbaja da universidade. Inicialmente, a antena externa foi instalada no veículo, e o protótipo do transmissor foi colocado na posse do motorista, enquanto o outro ESP32 LoRa receptor, também com a mesma antena externa, foi posicionado em outra extremidade. Para facilitar o experimento, uma bateria portátil foi utilizada. As figuras a seguir demonstram esse procedimento.

**Figura 31 -** Locação da antena no Veiculo

**Fonte:** Autor próprio (2023).

**Figura 32 -** Locação do dispositivo Transmissor no Veiculo




**Fonte:** Autor próprio (2023).

**Figura 33 -** Localização da realização do teste de comunicação entre os dispositivos em veículo Off Road em movimento


**Fonte:** Autor próprio (2023).

Diante da apresentação das figuras anteriores, os procedimentos realizados durante o período foram satisfatórios em suas análises. É importante destacar que a comunicação ocorreu corretamente, sem erros, em uma distância aproximada de 338 metros. Vale ressaltar que, mesmo com o veículo em movimento, o que poderia potencialmente causar interferência na comunicação com o dispositivo receptor, o procedimento ocorreu de forma satisfatória.

Após a realização bem-sucedida do processo em campo aberto, a mesma tarefa foi realizada com o receptor em uma sala fechada, mas os resultados foram os mesmos, sem erros de comunicação.

# 6 CONCLUSÕES

Diante do estudo realizado e dos procedimentos executados, que incluíram a avaliação da comunicação entre os ESP32 LoRa com as antenas padrões, a análise da utilização da telemetria com apenas uma antena externa entre os protótipos, bem como a utilização de duas antenas externas entre os dispositivos em diferentes localidades e fatores que podem interferir na comunicação, é possível constatar que foram apresentados bons resultados.

Com isso podemos destacar alguns pontos especiais:

- De modo geral, podemos mencionar que a comunicação entre os dispositivos foi realizada com sucesso, com isso enfatiza a construção de forma correta dos códigos fontes do projeto;
- As Antenas que vieram de fabrica nos dispositivos possuem um pequeno raio de alcance, dessa forma sendo necessário uma adaptação para um melhor alcance;
- Antenas de uso externo foram de suma importância, sendo que as mesmas colocaram o projeto em outro ponto de análise de resultados;
- A utilização do projeto em localidades onde se tem grandes bloqueios podem influenciar na comunicação dos dispositivos;
- Ao final do projeto foi possível notar a não influencia de distinção de onde essa comunicação possa ser realizada, seja ela em campo aberto ou em uma sala fechada;
- A realização do mencionado trabalho pode contribuir de forma significativa no quesito de comunicação de dados para o veículo estudado;

Assim, podemos concluir que este estudo obteve bons resultados, com uma comunicação eficaz entre os protótipos. A adição de antenas externas foi fundamental para o funcionamento em distâncias maiores, e a utilização do projeto em veículos off-road baja proporcionou números e informações positivas. No entanto, sendo este um estudo inicial, é necessário dar continuidade à pesquisa para aprimorar os dados apresentados.

## REFERÊNCIAS BIBLIOGRÁFICAS

ABREU, A D S. Arduino **– Plataforma Eletrônica Microcontrolada**. 2012. 124p. Dissertação (Bacharel) – Centro de Ciências Exatas e Tecnologia – Departamento de Engenharia de Eletricidade, Universidade Federal do Maranhão, São Luís, 2012.

AOSONG. **DHT11 SIP Packaged Temperature and Humidity Sensor.** Guangzhou Aosong Electronic Co. Ltd. Disponível em: http://www.aosong.com/en/products21.html. Acesso em: 26 de agosto de 2023.

ADAMOWSKY, Júlio Cezar; FURUKAWA, Celso Massatoshi. **Uma abordagem voltada à Automação Industrial**. Revista Mecatrônica Atual, São Paulo, v. Especial, n.1, p. 8-11, outnov. 2001.

AYARAFUN. Node32s: **Pinout Ayarafun make, creative and let's fun.** 2017. Disponível em: http://www.ayarafun.com/2016/11/esp32-node32s-pinout/. Acesso em: 25 de agosto de 2023.

ASHTON, K. **That "Internet of Things" Thing.** RFID Journal, 22 de junho de 2009.

BAJA NACIONAL, (2012). **SAE programas Estudantis.** Disponível em: http://www.saebrasil.org.br/eventos/ProgramasEstudantis/si te/baja2012/index.html Acesso em: 28 de agosto de 2023.

BAJA NACIONAL**. SAE Brasil**, São Paulo, SP. Disponível em: <https://saebrasil.org.br/programas-estudantis/baja-sae-brasil>. Acesso em: 13/07/2023.

CAMPBELL-KELLY, M.; ASPRAY, W.; ENSMENGER, N.; YOST, J. R. Computer: **A History of the Information Machine**. 3a Edição. Westview Press, 2014.

CULLER, D., Estrin, D., & Srivastava, M. **Overview of Sensor Networks.** 2004. IEEE Computer, 37(8), 41-49.

CURVELLO, Andre. Embarcados, **apresentando o módulo ESP8266,** 2016. Disponível em: https://embarcados.com.br/modulo-esp8266/, 2015. Acesso em: 25 de agosto de 2023.

ESPRESSIF. **ESP-WROOM-32 Series**: Datasheet. 3. ed. Xangai: Espressif System, 2019. Disponível em: Acesso em: 25 de agosto de 2023.

Evans, D., & Annunziata, M. Industrial internet: **Pushing the boundaries of minds and machines**. General Electric, 2012.

Gubbi, J., Buyya, R., Marusic, S., & Palaniswami, M. Internet of Things (IoT): **A vision, architectural elements, and future directions**. 2013. Future generation computer systems, 29(7), 1645-1660.

HENRIQUES, Luiz Felipe Araújo. **Implementação e monitoramento de um sistema de irrigação automatizado em iot utilizando módulo esp32 em plantio caseiro.** Manaus, 2021.

IBGE. **Pesquisa Nacional por Amostra de Domicílios** [Internet]. 2014.

IBRAHIM, Dragan. **The Complete ESP32 Projects Guide**. 1a. ed. [S.l.]: Elektor Digital, 2017. Citado na página 23.

ITU (2005). Internet reports —**The internet of things.** Disponível em: < https://www.itu.int/net/wsis/tunis/newsroom/stats/The-Internet-of-Things-2005.pdf >. Acesso em: maio 2020.

JUNIOR, V. P**. Conheça a tecnologia LoRa® e o protocolo LoRaWAN™.** [S. l.], 6 abr. 2016. Disponível em: https://www.embarcados.com.br/conheca-tecnologia-lorae-o-protocolo-lorawan/. Acesso em: 14 out. 2023.

Lee, H., & Smith, B. (2011). **Multi-Core Processors: Architectures, Performance Analysis, and Future Challenges.** ACM Transactions on Architecture and Code Optimization, 8(1), 1-22.

LIU, Yi; WANG, He; WANG, Junyu; QIAN, Kan; KONG, Ning; WANG, Kaijiang; ZHENG, Lirong; SHI, Yiwei; ENGELS, Daniel. **Enterprise-Oriented IoT Name Service for Agricultural Product Supply Chain Management. International Journal of Distributed Sensor Networks Volume.** 2015. Article ID 308165, 12 páginas. Disponível em: < http://www.hindawi.com/journals/ijdsn/2015/308165/>. Acesso em: Setembro, 2023.

MARGOLIS, Michael. **Arduino Cookbook - Recipes to Begin, Expand and Enhance Your Projects.** O'Reilly. Sebastopol: 2011.

MCT. Ministério de Ciência e Tecnologia. Ciência, tecnologia e inovação**: Desafio para a Sociedade Brasileira - Livro Verde.** Brasília: Ministério da Ciência e Tecnologia- Academia Brasileira de Ciências. 2001.

MENDES, D. R., **Redes de Computadores**. Novatec Editora, 2015.

METALÚRGICO, S. A INDÚSTRIA AUTOMOBILÍSTICA NO BRASIL: **Diagnóstico do Setor e Análise do Novo Regime Automotivo Sumário.** 2012. Disponível em: Acesso em: 7 jul. 2019.

NAVARRO, Paula Gonçalves; RODRIGUES, Ernesto Luis Malta. **O uso de redes lora para sensoriamento industrial remoto.** Paraná, 2022.

OLIVEIRA, Euller. **Conhecendo a Placa WiFi LoRa ESP32,** 2020. Disponível em: https://blogmasterwalkershop.com.br/embarcados/esp32/conhecendo-a-placa-wifi-lora-esp32-433mhz-868mhz-915mhz, 2015. Acesso em: 25 de agosto de 2023.

OULASVIRTA, A., Rattenbury, T., Ma, L., and Raita, E. 2012. **Habits make smartphone usemore pervasive. Personal Ubiquitous Comput**., 16(1):105–114.

PEREIRA, Mauricio; CRUVINEL, Paulo**. Desenvolvimento de um sistema de coleta automática de dados agrícolas baseado em rede LoRa e no microprocessador ESP32.** In: ESCOLA REGIONAL DE INFORMÁTICA DE MATO

GROSSO (ERI-MT), 10., 2019, Cuiabá. Anais [...]. Porto Alegre: Sociedade Brasileira de Computação, 2019. p. 43-48. ISSN 2447-5386. DOI: https://doi.org/10.5753/eri-mt.2019.8592.

Peterson, M., Wilson, R., & Anderson, J. (2013). **Microprocessor Developments: From 4004 to 8088.** Computing Perspectives, 42(3), 16-28.

METALÚRGICO, S. A INDÚSTRIA AUTOMOBILÍSTICA NO BRASIL: **Diagnóstico do Setor e Análise do Novo Regime Automotivo Sumário.** 2012. Disponível em: <http://www.smabc.org.br/Interag/temp_img/%7B57336FD0-AA1A4ED1-92AA DE866CE178DA%7D_diagnostico do setor automotivo.uv.pdf>. Acesso em: 7 jul. 2019.

MCLUHAN, M. **Understanding Media: The Extensions of Man.** MIT Press, 1994.

MCROBERTS, Michael. **Beggining Arduino.** New Iorque: 2010.

PATSKO, L. F. **Aplicação, funcionamento e utilização de sensores**. São Paulo: Maxwell Bohr, 2006. Apostila.

Smith, A. **The Evolution of Microprocessors. Journal of Computing Research.** 2015 27(2), 113-129.

Smith, J., & Johnson, R. **Exploring the Frontiers of Microprocessor Evolution: Quantum Computing Prospects.** 2020. Journal of Emerging Technologies, 12(3), 45-58.

SOUZA, David José. **Desbravando o PIC: Ampliado e Atualizado para PIC 16F628A.** 8ª ed. São Paulo, SP, Brasil: Érica, 2005.

STATISTA – THE PORTAL OF STATISTICS. **Internet of Things (IoT) connected devices installed base worldwide from 2015 to 2025 (in billions).** 2019, p.1. Disponível em: Acesso em: 23 mar 2019.

Sundmaeker et al. 2010 Sundmaeker, H., Guillemin, P., Friess, P., and Woelfflé, S. **Vision and challenges for realising the Internet of Things.** 2010. volume 20. EUR-OP.

TAVARES, C. A. **Controle de resfriamento de servidores utilizando um microcontrolador.** 2013. 55 f. Trabalho de Conclusão de Curso (Bacharelado) - Centro Universitário de Brasília - UniCEUB, Brasília, 2013.

TEIXEIRA, Gustavo. **ESP32 LORA WIFI SX1278,** 2019. Disponível em: https://www.usinainfo.com.br/blog/esp32-lora-wifi-sx1278/ 2019. Acesso em: 25 de agosto de 2023.

TESLA, N. When Woman is Boss. Colliers, Edição de 30 de janeiro, 1926. In: **An interview with Nikola Tesla by John B. Kennedy.** 21st Century Books, 2013. (http://www.rawscience.tv/when-woman-is-boss-tesla-on-wifi-and-gender-equality/).

WENDLING, M. **Sensores. Guaratinguetá: UNESP.** 2010. Apostila.

ZANELLA, A. et al. **Internet of Things for Smart Cities**. IEEE Internet Of Things Journal, [S. l.], v. 1, n. 1, p. 22-32, Feb. 2014. DOI: 10.1109/JIOT.2014.2306328. Disponível em: http://www.dei.unipd.it/~zanella/PAPER/CR_2014/IoTSmartCity2014_CR.pdf Acesso em: 09 de setembro, 2023.

ZANNI, Marco. **Carros inteligentes**. 2014. Revista Galileu. Disponível em: <https://revistagalileu.globo.com/Revista/noticia/2014/01/carros-inteligentes.html>. Acesso em: 7 jul. 2019.

ZUIN, A., ZUIN, A. **formação no tempo e no espaço da Internet das Coisas.** Revista Educação & Sociedade. Campinas, V. 37, N. 136, 2016.

# ANEXOS

## Código transmissor

```
#include "heltec.h" // Inclui a biblioteca Heltec para o ESP32 com LoRa
#include "DHTesp.h" // Inclui a biblioteca DHTesp para o sensor DHT11

#define BAND 915E6 // Define a frequência LoRa como 915 MHz

DHTesp dht; // Cria um objeto DHTesp para o sensor DHT11

void sendPacket(float temperature, float humidity); // Protótipo da função para enviar dados via LoRa

void setup() {
  pinMode(LED, OUTPUT); // Configura o pino do LED como saída
  Serial.begin(9600); // Inicializa a comunicação serial com uma taxa de 9600 bps
  Heltec.begin(true, true, true, true, BAND); // Inicializa o módulo Heltec ESP32 LoRa com a configuração especificada, incluindo a frequência LoRa
  Heltec.display->init(); // Inicializa o display OLED
  Heltec.display->flipScreenVertically(); // Inverte a tela verticalmente (opcional, dependendo da montagem do display)
  Heltec.display->setFont(ArialMT_Plain_10); // Define a fonte para o texto como Arial, tamanho 10
  Heltec.display->clear(); // Limpa o display OLED
  Heltec.display->drawString(33, 5, "INICIADO"); // Desenha o texto "INICIADO" na posição (33, 5) do display
  Heltec.display->drawString(10, 30, "TCC SÁVIO!"); // Desenha o texto "TCC SÁVIO!" na posição (10, 30) do display
  Heltec.display->display(); // Atualiza o display
  delay(5000); // Aguarda 1 segundo
  dht.setup(17, DHTesp::DHT11); // Inicializa o sensor DHT11 no pino 17
}

void loop() {
```

```
  float currentTemp = dht.getTemperature(); // Lê a temperatura atual do sensor DHT11
  float currentHumidity = dht.getHumidity(); // Lê a umidade atual do sensor DHT11

  Heltec.display->clear(); // Limpa o display OLED
  Heltec.display->setTextAlignment(TEXT_ALIGN_LEFT); // Define o alinhamento do texto como à esquerda
  Heltec.display->setFont(ArialMT_Plain_10); // Define a fonte para o texto como Arial, tamanho 10

  Heltec.display->drawString(30, 5, "TCC SÁVIO"); // Desenha o texto "TCC SÁVIO" na posição (30, 5) do display
  Heltec.display->drawString(33, 30, String(currentTemp)); // Exibe a temperatura atual na posição (33, 30) do display
  Heltec.display->drawString(78, 30, "°C"); // Exibe o símbolo de grau Celsius na posição (78, 30) do display
  Heltec.display->drawString(33, 50, String(currentHumidity)); // Exibe a umidade atual na posição (33, 50) do display
  Heltec.display->drawString(78, 50, "%"); // Exibe o símbolo de porcentagem na posição (78, 50) do display
  Heltec.display->display(); // Atualiza o display OLED

  sendPacket(currentTemp, currentHumidity); // Chama a função para enviar os dados via LoRa
  delay(5000); // Aguarda 5 segundos antes de enviar novamente
}

void sendPacket(float temperature, float humidity) {
  LoRa.beginPacket(); // Inicia um novo pacote LoRa
  LoRa.print("Temp: "); // Envia a string "Temp: "
  LoRa.print(temperature); // Envia a temperatura
  LoRa.print("°C, Umidade: "); // Envia a string ", Umidade: "
  LoRa.print(humidity); // Envia a umidade
  LoRa.print("%"); // Envia o símbolo de porcentagem
```

```
  LoRa.endPacket(); // Finaliza o pacote LoRa
}
```

**Código receptor**

```
#include "heltec.h" // Inclui a biblioteca Heltec para o ESP32 com LoRa

#define BAND 915E6 // Define a frequência LoRa como 915 MHz

String temperatureString = ""; // String para armazenar a temperatura
String humidityString = "";    // String para armazenar a umidade

void LoRaDataPrint(); // Protótipo da função para atualizar o display
void cbk(int packetSize); // Protótipo da função para tratar dados recebidos via LoRa

void LoRaDataPrint() {
  Heltec.display->clear(); // Limpa o display OLED
  Heltec.display->setTextAlignment(TEXT_ALIGN_CENTER); // Define o alinhamento do texto como centralizado
  Heltec.display->setFont(ArialMT_Plain_10); // Altera o tamanho da fonte para 10 pontos

  // Exibe a temperatura e a umidade com as iniciais "TEMP" e "UMID" no centro do display
  Heltec.display->drawString(Heltec.display->width() / 2, 10, "TEMP: " + temperatureString);
  Heltec.display->drawString(Heltec.display->width() / 2, 30, "UMID: " + humidityString);

  Heltec.display->display(); // Atualiza o display OLED
}

void cbk(int packetSize) {
  String receivedData = "";

  for (int i = 0; i < packetSize; i++) {
```

```
    receivedData += (char)LoRa.read();
  }

  // Use o método indexOf para encontrar as posições de "Temp: " e ", Umidade: "
  int tempPosition = receivedData.indexOf("Temp: ");
  int humidityPosition = receivedData.indexOf(", Umidade: ");

  if (tempPosition != -1 && humidityPosition != -1) {
    // Obtém os valores de temperatura e umidade a partir dos índices encontrados
    temperatureString = receivedData.substring(tempPosition + 6, humidityPosition);
    humidityString = receivedData.substring(humidityPosition + 11);

    LoRaDataPrint(); // Atualiza o display com os novos valores
  }
}

void setup() {
  pinMode(LED, OUTPUT); // Configura o pino do LED como saída
  Serial.begin(9600); // Inicializa a comunicação serial com uma taxa de 9600 bps
  Heltec.begin(true, true, true, true, BAND); // Inicializa o módulo Heltec ESP32 LoRa com a configuração especificada, incluindo a frequência LoRa
  Heltec.display->init(); // Inicializa o display OLED
  Heltec.display->flipScreenVertically(); // Inverte a tela verticalmente (opcional, dependendo da montagem do display)
  Heltec.display->setFont(ArialMT_Plain_10); // Define a fonte para o texto como Arial, tamanho 10
  Heltec.display->clear(); // Limpa o display OLED

  // Exibe o texto "TCC Sávio" centralizado no display
  Heltec.display->drawString(Heltec.display->width() / 2, Heltec.display->height() / 2 - 10, "TCC SÁVIO");
  Heltec.display->display(); // Atualiza o display
  delay(5000); // Aguarda 5 segundos para exibir o texto
```

```
  Heltec.display->clear(); // Limpa o display
  Heltec.display->display(); // Atualiza o display
  delay(1000); // Aguarda 1 segundo
  LoRa.receive(); // Configura o dispositivo LoRa para receber dados
}

void loop() {
  int packetSize = LoRa.parsePacket(); // Verifica se há um pacote LoRa recebido
  if (packetSize) {
    cbk(packetSize); // Chama a função para tratar os dados recebidos via LoRa
    digitalWrite(LED, HIGH); // Liga o LED por 500ms
    delay(500);
    digitalWrite(LED, LOW); // Desliga o LED por 500ms
    delay(500);
    Serial.print("Recebendo a temperatura: ");
    Serial.print(temperatureString);
    Serial.println("°C");
    Serial.print("Recebendo a umidade: ");
    Serial.print(humidityString);
    Serial.println("%");
  }
  delay(10); // Aguarda 10ms antes da próxima iteração
}
```