

**CENTRO PAULA SOUZA
FATEC SANTO ANDRÉ
Tecnologia em Mecatrônica Industrial**

**Gustavo Cavedon Rodrigues
Luka Nagayassu Aguiar
Vitor Micael Araujo
Pablo Vinícius Silva**

**DESENVOLVIMENTO DE UM DRONE CONTROLADO ATRAVÉS
ESP-32 COM COMUNICAÇÃO VIA MODBUS COM SUPERVISÓRIO**

Santo André

2024

Gustavo Cavedon Rodrigues

Luka Nagayassu Aguiar

Vitor Micael Araujo

Pablo Vinícius Silva

**DESENVOLVIMENTO DE UM DRONE CONTROLADO ATRAVÉS
ESP-32 COM COMUNICAÇÃO VIA MODBUS COM SUPERVISÓRIO**

Trabalho de Conclusão de Curso
apresentado ao Curso Tecnólogo em
Mecatrônica Industrial da Fatec de Santo
André, orientado pelo Mestre Luíz Vasco
Puglia como requisito parcial para
obtenção do título de tecnólogo em
Mecatrônica Industrial.

Santo André

2024

LISTA DE PRESENÇA

Santo André, 29 de junho de 2024.

LISTA DE PRESENÇA REFERENTE À APRESENTAÇÃO DO
TRABALHO DE CONCLUSÃO DE CURSO COM O TEMA:
“DESENVOLVIMENTO DE UM DRONE CONTROLADO” DOS
ALUNOS DO 6º SEMESTRE DESTA U.E.

BANCA

PRESIDENTE:

PROF. JHONNY FRANK SOUSA JOCA _____

MEMBROS:

PROF. GLAUCIO SANTOS _____

PROF. NOURIANDRES LIBORIO SILVA _____

ALUNOS:

GUSTAVO CAVEDON RODRIGUES _____

LUKA NAGAYASSU AGUIAR _____

PABLO VINICIUS SILVA _____

VITOR MICAEL ARAUJO _____

AGRADECIMENTO

Agradecemos primeiramente aos nossos orientadores/professores Luíz Vasco Puglia e Pedro Adolfo Galani pelas orientações valiosas, apoio contínuo e incentivo durante todo o desenvolvimento deste trabalho.

Expressamos também nossa gratidão aos colegas de classe e amigos, cujo apoio e encorajamento foram fundamentais para superar os desafios enfrentados ao longo desta jornada acadêmica.

À nossa família, dedicamos um profundo agradecimento pelo amor incondicional, apoio emocional e compreensão demonstrados ao longo dos anos de estudo. Seu apoio foi essencial para alcançar este marco em nossa vida acadêmica.

Por fim, agradecemos a todas as fontes de conhecimento, literatura e recursos consultados ao longo deste trabalho, bem como às instituições que proporcionaram o ambiente propício para o desenvolvimento acadêmico.

Este trabalho não teria sido possível sem o apoio e contribuição de cada uma dessas pessoas e entidades. Nosso mais sincero obrigado a todos

“Inventar é imaginar o que ninguém pensou; é acreditar no que ninguém jurou; é arriscar o que ninguém ousou; é realizar o que ninguém tentou. Inventar é transcender.”

Santos Dumont

RESUMO

O projeto intitulado "Desenvolvimento de um Drone controlado através de um ESP-32" tem como principal objetivo a criação completa de um quadricóptero, com controle via Supervisório. Para a concretização desse empreendimento, é essencial iniciar com a análise detalhada de um drone genérico, a fim de obter uma compreensão profunda dos motores e de toda a dinâmica subjacente ao funcionamento de um quadricóptero. Através da implementação de códigos de programação em linguagem C++, é conferido ao microcontrolador a capacidade de executar as diversas funções necessárias para o correto funcionamento do quadricóptero. O controle remoto do drone é realizado através de um supervisório no software AVEVA, com o ESP-32 conectado via Modbus. No que concerne à parte estrutural, toda a configuração física do drone é elaborada usando software de modelagem 3D. A representação virtual permite a criação minuciosa da estrutura, que posteriormente é concretizada por meio de impressão em formato STL. Esse procedimento garante um controle preciso das dimensões do drone, assegurando a qualidade e eficácia do projeto em sua totalidade.

Palavras-chave: Drone. Microcontrolador ESP-32. C++. Modbus. AVEVA. Impressão STL.

ABSTRACT

The project entitled "Development of a Drone controlled through an ESP-32" has as its main objective the complete creation of a quadcopter, with control via Supervisory. To carry out this undertaking, it is essential to start with a detailed analysis of a generic drone, in order to obtain a deep understanding of the engines and all the dynamics underlying the operation of a quadcopter. Through the implementation of programming codes in C++ language, the microcontroller is given the ability to perform the various functions necessary for the correct operation of the quadcopter. Remote control of the drone is carried out through a supervisor in AVEVA software, with the ESP-32 connected via Modbus. Regarding the structural part, the entire physical configuration of the drone is created using 3D modeling software. The virtual representation allows the detailed creation of the structure, which is later implemented through printing in STL format. This procedure guarantees precise control of the drone's dimensions, ensuring the quality and effectiveness of the project in its entirety.

Keywords: Drone. ESP-32 microcontroller. C++. Modbus. AVEVA. STL printing.

LISTA DE FIGURAS

Figura 1 – Abraham Karen ao lado do Amber.....	15
Figura 2 – Drone brasileiro BQM-1BR.....	16
Figura 3 – Anatomia de um Drone	19
Figura 4 – Sentido de Direção das Hélices	20
Figura 5 – Rotação Entorno do Próprio Eixo.....	20
Figura 6 – Movimento Frente e Trás	21
Figura 7 – Movimento Subir e Descer	21
Figura 8 – Movimento de Rolagem	22
Figura 9 – Movimento de Arfagem	22
Figura 10 – Supervisório de uma Cervejaria	25
Figura 11 – Monitoramento de Temp.	26
Figura 12 - Quadro de Mensagens para Protocolo Modbus.....	28
Figura 13 – Projeto 3D	33
Figura 14 – Impressora 3D.....	34
Figura 15 – Base Central do Drone.....	36
Figura 16 – Os Braços do Drone.....	37
Figura 17 – Motor Brushless	38
Figura 18 – Modulador de Velocidade ESC	39
Figura 19 – Hélice de PETG.....	39
Figura 20 – Bateria de Lítio	40
Figura 21 – Localização da Bateria	40
Figura 22 – Carregador da Bateria.....	41
Figura 23 – Tabela Comparativa	42
Figura 24 – Microcontrolador ESP-32	42
Figura 25 – Circuito Elétrico do Drone	43
Figura 26 – Software Arduino IDE.....	44
Figura 27 – Programa Biblioteca Modbus	45
Figura 28 – Programa 4 Leds.....	46
Figura 29 – Programa 4 Leds.....	47
Figura 30 – Teste Holding Registers	48
Figura 31 – Teste Holding Registers	49

Figura 32 – Controle De Velocidade Pwm	50
Figura 33 – Controle De Velocidade Pwm	51
Figura 34 – Adicionando Input Register	52
Figura 35 – Adicionando Input Register	53
Figura 36 – Adicionando Input Register	54
Figura 37 – Código Final	55
Figura 38 – Código Final	56
Figura 39 – Código Final	57
Figura 40 – Código Final	58
Figura 41 – Tags do Supervisório	59
Figura 42 – Tela Experimental do Supervisório	60
Figura 43 – Coils Aveva.....	58
Figura 44 – Holding Registers Aveva.....	59
Figura 45 – Lista Completa de Tags	62
Figura 46 – Adição de Eixos.....	63
Figura 47 – Tela Supervisório	64
Figura 48 – Circuito de Testes	65
Figura 49 – Foto Real do Circuito.....	65
Figura 50 – Regulador de Tensão Dc-Dc.....	66
Figura 51 – Drone Completo	67
Figura 52 – Frente Drone Completo.....	68
Figura 53 – Traseira Drone Completo	68
Figura 54 – Braço Drone Completo.....	69
Figura 55 – Braço 2 Drone Completo.....	69
Figura 56 – Braço 3 Drone Completo.....	70
Figura 57 – Centro Drone Completo	70
Figura 58 – Superior do Centro	71
Figura 59 – Inferior do Centro	71
Figura 60 – Impressão do Centro Inferior.....	72
Figura 61 – Impressão do Centro Superior	72
Figura 62 – Impressão dos Braços.....	73

SUMÁRIO

1	INTRODUÇÃO	13
1.1	Industria 4.0	13
1.2	História dos Drones.....	14
1.2.1	História dos Drones no Brasil.....	15
1.2.2	Uso de Drone no Brasil.....	16
1.3	Como um drone funciona?	18
1.3.1	Dinâmica de Voo.....	20
2	FUNDAMENTAÇÃO TEÓRICA	23
2.1	Microcontrolador	23
2.1.1	Microcontrolador X Microprocessador	23
2.2	ESP-32.....	23
2.3	Sistema Supervisório: Definição e Aplicações	24
2.3.1	Definição de Sistema Supervisório	24
2.3.2	Aplicações de Sistemas Supervisórios	25
2.4	AVEVA HISTORIAN.....	26
2.5	Modbus TCP/IP.....	27
2.5.1	Funcionamento do Protocolo Modbus	28
2.5.2	Comunicação TCP	30
3	DESENVOLVIMENTO DO PROJETO	32
3.1	Estrutura principal	32
3.1.1	Planejamento e Design	32
3.1.2	Materiais	33
4	MONTAGEM	35
4.1	Montagem da Estrutura.....	35
4.1.1	A Base do Drone.....	35

4.1.2	Materiais	37
4.2	Hardware	41
4.2.1	ESP-32	41
4.3	Circuito elétrico	43
4.4	Processo de Programação.....	44
4.5	Comunicação no AVEVA	59
4.5.1	Lógica de Voo por Modbus	62
4.5.2	Tela Principal do Supervisório	63
5	TESTES E RESULTADOS	65
5.1	Testes do Programa.....	65
5.2	Testes do Drone.....	66
5.3	Resultados	67
5.3.1	Projeto 3D	71
6	CONCLUSÃO E PROPOSTAS FUTURAS	74
7	REFERÊNCIAS	75

1 INTRODUÇÃO

Com o mundo se movimentando para a Quarta Revolução Industrial, conhecida como Indústria 4.0, tecnologias já exploradas vêm se modificando para se adequar aos novos conceitos propostos. O seguinte projeto visa unificar tecnologias já existentes, para adaptá-las à nova era industrial, explorando os recursos ensinados na instituição FATEC Santo André e aplicando-os em todo o desenvolvimento.

Os drones são uma tecnologia que vem ganhando destaque e sendo utilizada cada vez mais frequentemente em diversas áreas, como entregas, monitoramento de áreas urbanas e rurais, segurança, agricultura de precisão, e até em missões de resgate. A popularidade desses dispositivos se deve à sua versatilidade, eficiência e capacidade de alcançar locais de difícil acesso de maneira rápida e segura.

A utilização de modelos amplamente conhecidos, como os drones, facilita a visualização prática de como sistemas de monitoramento e controle industrial funcionam. Eles oferecem uma plataforma acessível para demonstrar a aplicação de tecnologias avançadas em diversos setores, promovendo uma compreensão mais clara e prática dos processos envolvidos.

1.1 Indústria 4.0

Durante o meio milênio que se passaram, tiveram três grandes marcos na indústria que mudaram o mundo até chegar na forma em que conhecemos.

O primeiro grande marco na era industrial foi a Primeira Revolução Industrial, na metade do século XVIII, representada pela confecção de teares (aparelho para tecer tecidos) movidos a máquinas a vapor.

Já durante a Segunda Revolução Industrial que ocorreu aproximadamente entre meados do século XIX e o início do século XX, tiveram diversas implementações tecnológicas, como motores elétricos, produção de aço em larga escala, descoberta de novos processos de produção de produtos químicos e a síntese de novos materiais, a exploração, extração e refino de petróleo, e um grande destaque ao modelo de produção proposto por Henry Ford: a Linha de Montagem (SEBRAE, 2018).

Na Terceira Revolução Industrial, também chamada de Revolução Técnico-Científica, iniciada no século XX, em um período sucessor a Segunda Guerra Mundial, foi feita a implementação de tecnologias e informática para automatizar e otimizar os processos já existentes. Com a obtenção de dados informatizados, também foi possível realizar um melhor controle dos recursos da empresa. Foi durante esse período que introduziu o uso de CLPs no ramo industrial.

Por fim a Quarta Revolução Industrial, ou Indústria 4.0 caracteriza-se pelo uso de novas tecnologias na revolução antecessora.

Indústria 4.0, também chamada de Quarta Revolução Industrial, é um conceito, apresentado ao público pela primeira vez na Feira de Hannover de 2011, que representa a automação industrial e a integração de soluções tecnológicas inovadoras, como inteligência artificial, internet das coisas, robótica e utilização de dados em nuvem. (SIEMENS. Um guia prático sobre a Indústria 4.0.)

Ao contrário das demais, a Indústria 4.0 é um evento que está sendo estudado durante seu acontecimento, garantindo assim uma evolução mais concreta e otimizada.

1.2 História dos Drones

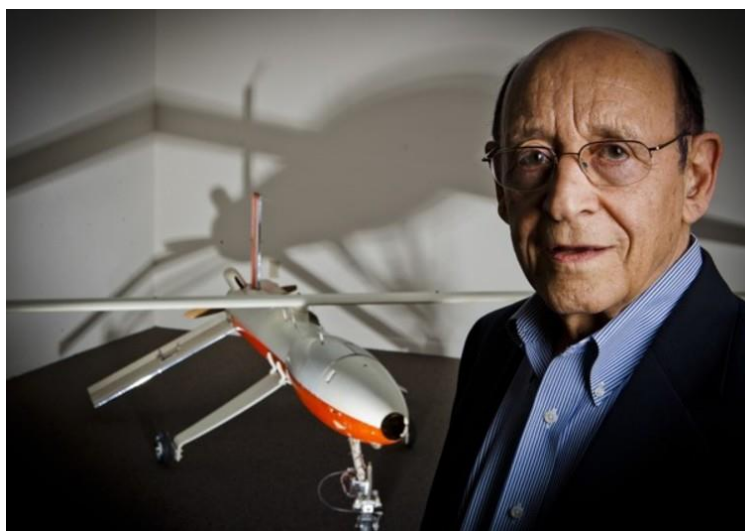
Assim como grande parte das tecnologias presentes nos tempos modernos, a criação dos drones iniciou-se em períodos de guerra, especificamente durante a Segunda Guerra Mundial. Foram inspirados em bombas voadoras alemãs chamados de *Buzz Bomb* (recebe esse nome por conta do som que fazia enquanto voava), que eram VANTs (Veículo Aéreo Não Tripulado) ou da sigla em inglês UAV (*Unmanned Aerial Vehicles*), capazes de voar em velocidade constante em linha reta (ODRONES, 2015).

A partir disso, um engenheiro espacial israelita, apaixonado por aeronaves, chamado Abraham Karen, passou a desenvolver VANT's em sua empresa '*Leading System*' no ano de 1977 nos EUA (ITARC). Inicialmente eram construídos com poucos

recursos, como fibra de vidro caseira e restos de madeira, o que deu origem ao VANT Albatross. (CONCURSOSNOBRASIL, 2023)

Posteriormente Abraham Karen recebeu um financiamento da empresa DARPA, e assim implementou melhorias no seu novo modelo VANT, o Amber, capaz de voar 56 horas sem recarga de bateria e com três pessoas operando. (FIGURA 1)

FIGURA 1 – ABRAHAM KAREN AO LADO DO AMBER.



Fonte: LUCAS BUZZO. ODrones, História dos Drones: do início aos dias de hoje, c2015. Disponível em: <<https://odrones.com.br/historia-dos-drones/>>. Acesso em: 03 de Mar de 2023

1.2.1 História dos Drones no Brasil

O Brasil também não ficou de fora na criação de drones, em território brasileiro teve o BQM-1BR, desenvolvido pela empresa CBT (Companhia Brasileira de Tratores) na década de 80, com o propósito de servir como alvo de treinamento para militares (FARMFOR, 2021).

A aeronave voava a partir de uma turbina a jato chamada de Tietê JT-2, que funcionava com querosene aeronáutico, álcool ou gás natural. O drone possuía 3,89 m de comprimento e uma envergadura de 3,38 m.

Apresentado como alvo aéreo, o avião tem 3m91 de comprimento e envergadura de 3m48. Pode voar até a 700 quilômetros por hora a uma altitude de 6000 metros, e com uma autonomia de voo de 45 minutos. Leva consigo uma carga útil de no máximo 30 quilos, que pode ser empregada tanto em missões de reconhecimento aéreo, feito por equipamento de televisão, como até servir de míssil, se tiver uma carga detonante. Em teste a primeira turbina a jato nacional. (Jornal Estado de São Paulo, São Paulo, 27, ago. 1983.)

Até hoje é possível encontrar um modelo do BQM-1BR no museu da TAM. (FIGURA 2)

FIGURA 2 – DRONE BRASILEIRO BQM-1BR.



Fonte: FARMFOR. Famfor, BQM-1BR, o drone fabricado pela Companhia Brasileira de Tratores – CBT, c2021. Disponível em: <<https://www.farmfor.com.br>>. Acesso em: 12 de jun de 2023

Outra empresa que se arriscou no desenvolvimento de VANT's foi a Embravant, que produziu o Gralha Azul, que possuía 4m de envergadura e podia realizar voos de até 3 horas. Infelizmente não é possível encontrar imagens do mesmo em sites de busca (DEFESACIVIL.ES, 2023).

1.2.2 Uso de Drone no Brasil

De acordo com o site do governo Federal, Pelo regulamento da ANAC, “Aeromodelos são as aeronaves não tripuladas remotamente pilotadas usadas para recreação e lazer e as aeronaves remotamente pilotadas (RPA) são as aeronaves não

tripuladas utilizadas para outros fins como experimentais, comerciais ou institucionais.” (GOV.BR, 2023).

Os dois tipos (aeromodelos e RPA) só podem ser operados em áreas com no mínimo 30 metros horizontais de distância das pessoas não anuentes ou não envolvidas com a operação e cada piloto remoto só poderá operar um equipamento por vez.” Ou seja, embora tenhamos vários tipos de drones, deve-se sempre manter atualizado sobre a legislação nacional, pois sobrevoar com o uso do drone de maneira irregular, além de ter o equipamento apreendido, poderá ser multado de R\$ 3.200,00 a R\$ 40.000,00 (ASAS, 2018).

No Brasil, a Agência Nacional de Aviação Civil (ANAC) é a entidade responsável por regular o uso de drones. Em 2021, a ANAC atualizou a legislação que versa sobre o tema. De acordo com as novas regras, os drones portáteis, assim como outros tipos de drones, não podem ser utilizados em áreas próximas a aeroportos e helipontos. Ademais, é obrigatório o registro de todos os drones com peso superior a 250g no site da ANAC, mediante pagamento de uma taxa.

Outras restrições incluem o não uso em áreas onde haja grande concentração de pessoas, em locais onde ocorram eventos públicos ou em áreas restritas, como presídios e bases militares. A altura máxima de voo é de 120 metros, e é proibido voar sobre propriedades privadas sem a permissão do proprietário (ANAC, 2021).

No que tange à privacidade, o uso de drones portáteis para espionagem ou vigilância de pessoas é proibido e pode ser enquadrado como crime de invasão de privacidade. Além disso, é importante ressaltar que o uso de drones para fins comerciais ou profissionais, como registro de imagens aéreas para veículos de comunicação ou para inspeções de segurança, exige autorização prévia da ANAC.

Ressalta-se que o não cumprimento das regulamentações pode resultar em multas e apreensão do drone, além de possíveis sanções criminais. Portanto, é fundamental que os usuários de drones portáteis conheçam as leis e regulamentações aplicáveis e utilizem o dispositivo de forma responsável e segura, respeitando a privacidade e segurança de terceiros (ED ROTA CULTURAL, 2018).

1.3 Como um drone funciona?

Os modelos mais comuns de drone possuem normalmente quatro motores elétricos localizados nas extremidades da estrutura do drone. Esses motores giram hélices que possuem um funcionamento parecido com um helicóptero, as hélices giram em sentido horário e anti-horário em forma de xis, para que assim garanta a sustentação de voo. (TECHTUDO, 2015)

Sua estrutura geralmente é feita a partir de fibra de carbono, por conta da sua leveza e resistência, utiliza-se plásticos nas regiões que não necessitam tanto de resistência mecânica.

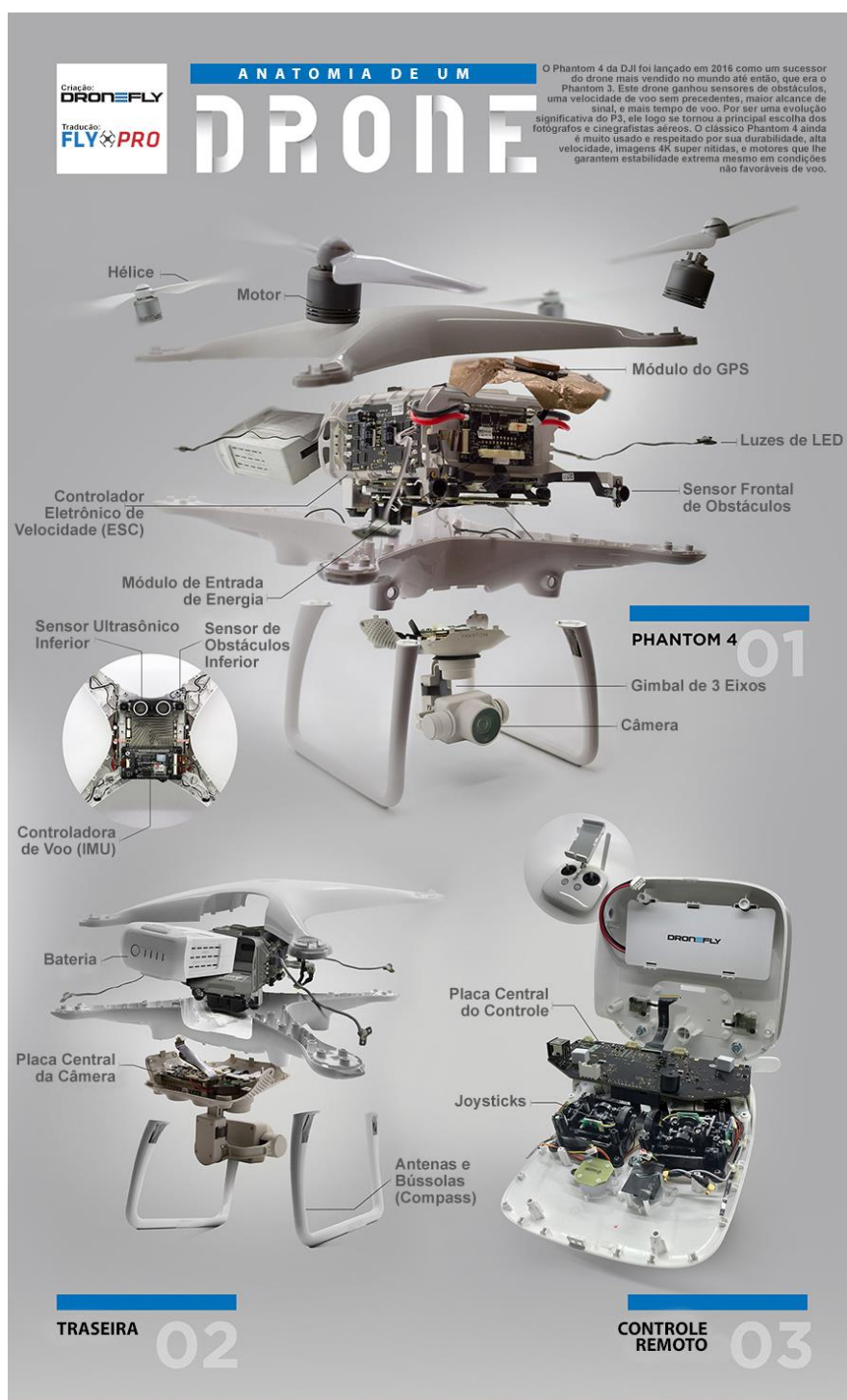
Dentro de sua carcaça estão as baterias, que precisam ser leves e pequenas para que facilite a decolagem do voo, porém isso acaba por influenciar no tempo de voo da aeronave. Até mesmos os melhores drones do mercado possuem poucos minutos de bateria para voo. (FLYPRO, 2024)

Seu funcionamento se deve a partir de uma placa lógica, também localizada na carcaça do drone. Alguns modelos apresentam recursos como GPS para navegação autônoma a partir de sinal de satélite (ARAÚJO, 2012).

A placa lógica também é responsável por transmitir aos motores a velocidade requisitada no controle manual para os moduladores de velocidade ESC.

No geral os modelos mais simples de drones apresentam esses recursos, porém a modelos mais complexos que possuem mais recursos como câmeras, sensores e recursos militares, como mostrado na FIGURA 3.

FIGURA 3 – ANATOMIA DE UM DRONE



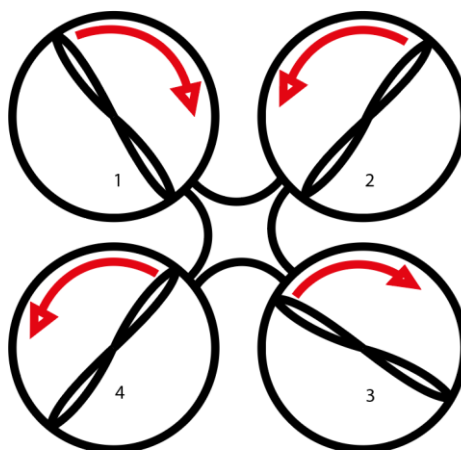
Fonte: FLYPRO, FlyPro: Anatomia de um drone. Disponível em:

<<https://www.flypro.com.br/pagina/anatomia-de-um-drone.html>>. Acesso em: 20 de jan. 2024

1.3.1 Dinâmica de Voo

Toda a elevação e propulsão de um quadricóptero só é possível ser realizada graças aos diferentes sentidos de direção de suas hélices, como mostra a FIGURA 4.

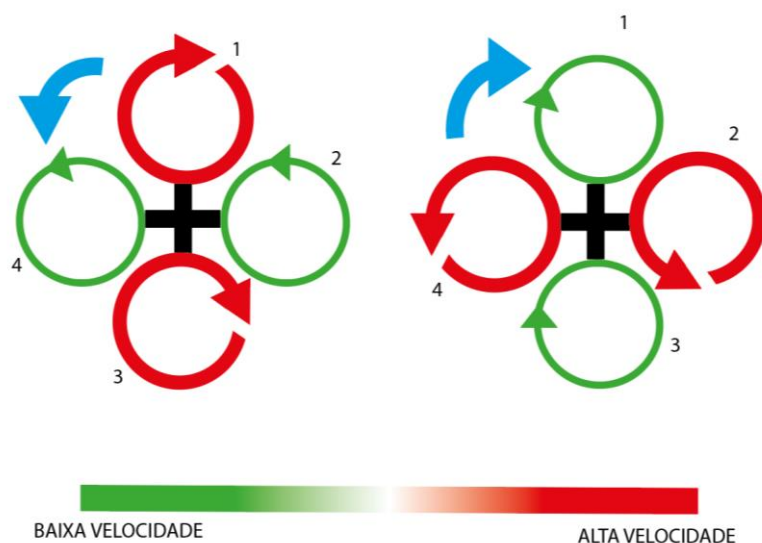
FIGURA 4 – SENTIDO DE DIREÇÃO DAS HÉLICES



Fonte: Autoria própria

Para realizar o movimento de giro em torno do seu eixo vertical de sustentação, é necessário que seus motores rotacionem diagonalmente em velocidades angulares diferentes, como mostra a FIGURA 5.

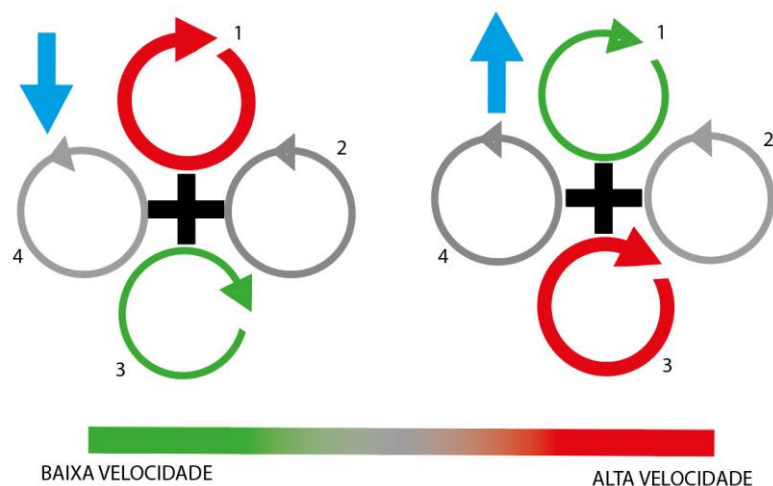
FIGURA 5 – ROTAÇÃO ENTORNO DO PRÓPRIO EIXO



Fonte: Autoria própria

Já para realizar o movimento de avanço será necessário aumentar a velocidade dos motores traseiros e diminuir a velocidade dos motores dianteiros. Para recuar o oposto deve ser realizado, assim como na FIGURA 6.

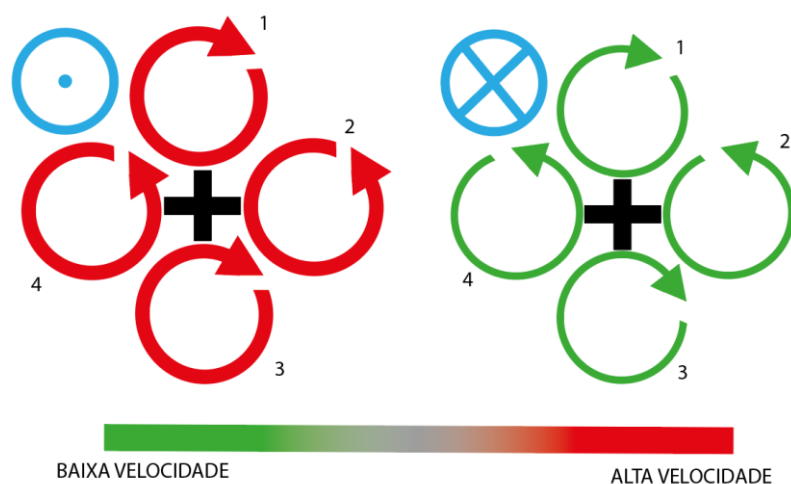
FIGURA 6 – MOVIMENTO FRENTE E TRÁS



Fonte: Autoria própria

O movimento de subir e descer pode ser realizado aumentando ou diminuindo a velocidade angular dos motores equitativamente. E para mantê-lo no ar, a força de sustentação precisa ser igual a força da gravidade. Como mostrado na FIGURA 7.

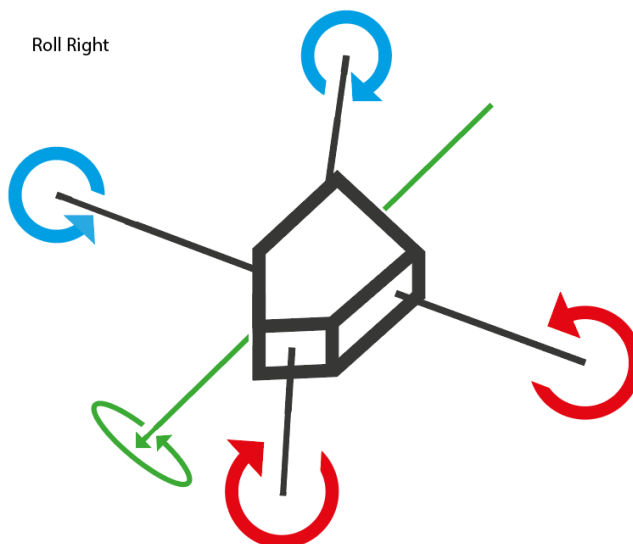
FIGURA 7 – MOVIMENTO SUBIR E DESCER



Fonte: Autoria própria

Para realizar o movimento de rolagem (*roll*) é necessário alterar as velocidades dos motores localizados à direita e à esquerda fazendo com que um lado do drone tenha mais sustentação que o outro. Mostrado na FIGURA 8.

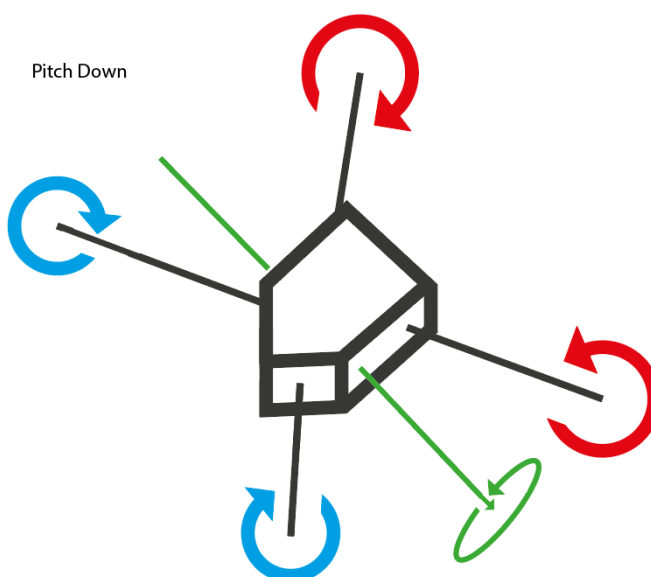
FIGURA 8 – MOVIMENTO DE ROLAGEM



Fonte: Autoria própria

O movimento de arfagem (*pitch*) é realizado ajustando a velocidade relativa dos motores traseiros e dianteiros do quadricóptero. Como a FIGURA 9 (ARAÚJO, 2012).

FIGURA 9 – MOVIMENTO DE ARFAGEM



Fonte: Autoria própria

2 FUNDAMENTAÇÃO TEÓRICA

2.1 Microcontrolador

Um microcontrolador se trata de um circuito integrado que possui em sua estrutura um núcleo de processador, memórias voláteis e não voláteis, e diferentes periféricos de entrada e saída de dados. Em resumo ele trabalha como um minicomputador, capaz de realizar determinadas tarefas (IFC, 2018).

2.1.1 Microcontrolador X Microprocessador

Apesar de possuírem nomes parecidos, podendo assim ser facilmente confundível, os microprocessadores não possuem os periféricos que os microcontroladores dispõem, sendo assim só podem realizar tarefas determinadas pela programação (UFC, 2000).

2.2 ESP-32

O ESP32 é um microcontrolador que surgiu como uma evolução do ESP8266, ambos desenvolvidos pela empresa chinesa Espressif Systems.

O ESP8266 foi lançado pela Espressif Systems em 2014. Ele era inicialmente concebido como um módulo para adicionar conectividade Wi-Fi a dispositivos eletrônicos de forma fácil e econômica. O ESP8266 ganhou popularidade na comunidade de desenvolvedores de hardware e IoT devido ao seu baixo custo, tamanho compacto e recursos de conectividade.

Embora o ESP8266 tenha sido muito bem recebido, a comunidade de desenvolvedores expressou o desejo de mais recursos, como mais GPIOs (pinos de entrada/saída), suporte a Bluetooth e maior poder de processamento. Em resposta a essa demanda, a Espressif Systems começou a trabalhar no desenvolvimento do ESP32 (OLIVEIRA, 2017).

O ESP32 foi oficialmente lançado em setembro de 2016. Ele apresentava uma arquitetura mais poderosa, incluindo dois núcleos de processamento Xtensa LX6,

conectividade Wi-Fi e Bluetooth, maior quantidade de GPIOs, capacidade de processamento gráfico e outros recursos avançados (OMEGA, 2021).

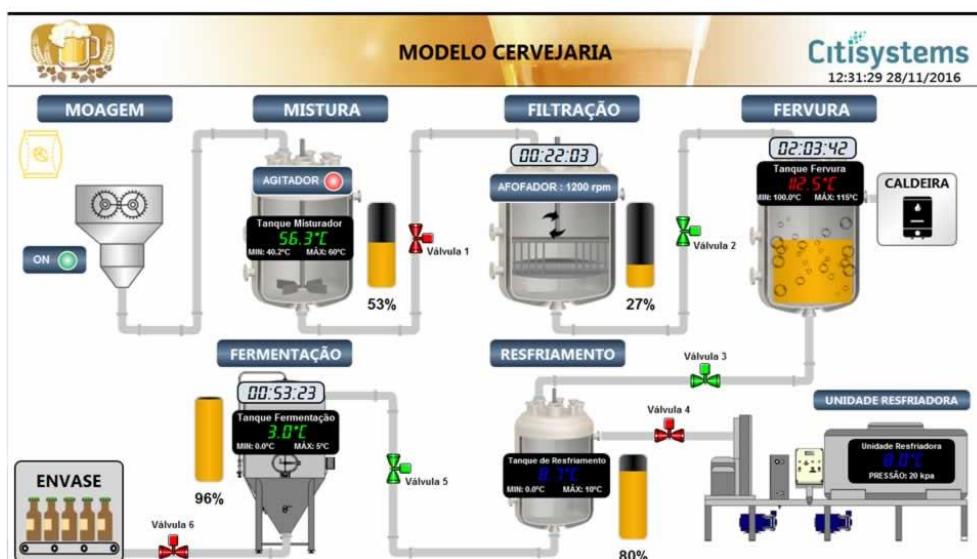
2.3 Sistema Supervisório: Definição e Aplicações

2.3.1 Definição de Sistema Supervisório

Os sistemas supervisórios, também conhecidos como SCADA (*Supervisory Control and Data Acquisition*), são plataformas de software utilizadas para monitorar e controlar processos industriais e infraestrutura como no modelo da FIGURA 10. Esses sistemas coletam dados em tempo real de sensores e dispositivos localizados em campo, processam e apresentam essas informações em interfaces gráficas amigáveis aos operadores, permitindo o controle e a tomada de decisões em tempo real (TOTVS, 2023).

Um sistema supervisório é composto por diversos componentes. Entre eles, destacam-se as Unidades de Aquisição de Dados, como RTUs (*Remote Terminal Units*) e PLCs (*Programmable Logic Controllers*), que coletam dados diretamente dos sensores e equipamentos. O Servidor SCADA, por sua vez, processa e armazena os dados recebidos, executando algoritmos de controle e análise. A Interface Homem-Máquina (IHM) apresenta os dados de forma gráfica aos operadores, permitindo a interação e controle dos processos. Além disso, as Redes de Comunicação conectam os diversos componentes do sistema, possibilitando a transmissão de dados em tempo real (TOTVS, 2023).

FIGURA 10 – SUPERVISÓRIO DE UMA CERVEJARIA

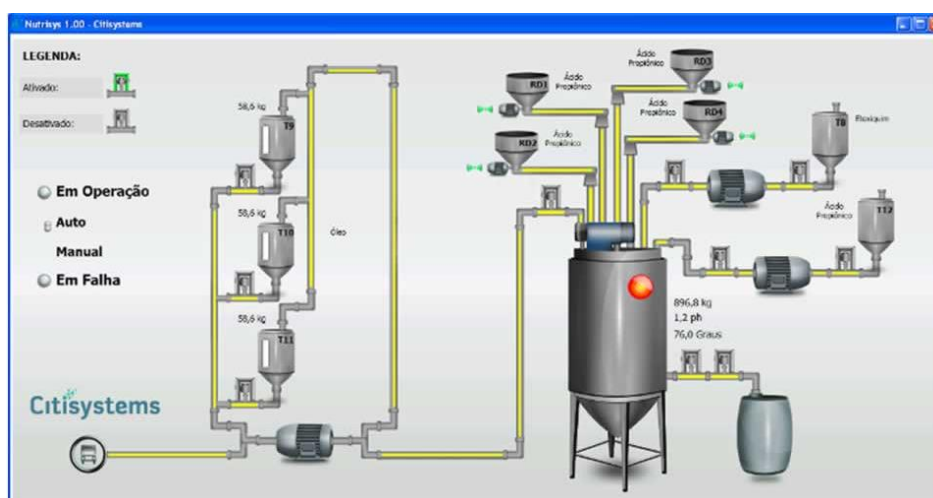


Fonte: Citisystems. Disponível em: < <https://www.citisystems.com.br/portfolio-items/supervisorio-cervejaria-imperatriz/> >. Acesso em: 10 de Mar. 2024

2.3.2 Aplicações de Sistemas Supervisórios

Os sistemas supervisórios têm uma vasta gama de aplicações em diferentes setores industriais e de infraestrutura. Na indústria de manufatura, por exemplo, são utilizados para o monitoramento e controle de linhas de produção, gestão de qualidade e eficiência operacional. No setor de energia e utilidades, são essenciais para a supervisão de redes de distribuição de eletricidade, água, gás e sistemas de energia renovável. No setor de petróleo e gás, são usados para o gerenciamento de operações em plataformas de petróleo, oleodutos e refinarias. Além disso, no tratamento de água e esgoto, controlam os processos de tratamento, distribuição e coleta. No setor de transportes, monitoram sistemas ferroviários, metroviários e de tráfego rodoviário. Finalmente, na automação predial, auxiliam na gestão de sistemas HVAC (*Heating, Ventilating and Air Conditioning*), segurança e iluminação em edifícios.

FIGURA 11 – MONITORAMENTO DE TEMP.



Fonte: Citisystems. Disponível em: <<https://www.citisystems.com.br/portfolio-items/supervisorio-dosagem-qualimix/>>. Acesso em: 10 de Mar. 2024

Esses sistemas são fundamentais para a modernização e eficiência operacional, proporcionando vantagens como a redução de custos, aumento da produtividade, melhoria na qualidade dos produtos e serviços, e maior segurança operacional.

2.4 AVEVA HISTORIAN

A AVEVA é uma empresa global que se destaca no fornecimento de soluções de software para automação industrial e gestão de ativos. Fundada em 1967 como CADCentre, a empresa tem uma longa história de inovação e liderança no setor. Ao longo dos anos, a AVEVA tem desempenhado um papel fundamental no avanço das tecnologias industriais, oferecendo soluções que abrangem desde o design de plantas e processos até a operação e otimização de ativos.

Alguns marcos notáveis na história da AVEVA incluem o pioneirismo na introdução de técnicas de design assistido por computador (CADD, *computer-aided design and drafting*) na década de 1960, o lançamento do primeiro sistema de design de planta em 3D em 1976 e a formação de parcerias estratégicas, como a aliança com a Schneider Electric.

A AVEVA tem se dedicado não apenas a impulsionar a eficiência e a inovação nas operações industriais, mas também a promover a sustentabilidade e a

responsabilidade ambiental. A empresa tem sido reconhecida por suas iniciativas inovadoras e sustentáveis, como o apoio à redução de gases de efeito estufa em projetos de plantas e a otimização da produção de energia solar.

Com uma ampla gama de soluções de software, a AVEVA atende a diversos setores industriais, incluindo petróleo e gás, energia, água e serviços públicos, manufatura e muitos outros. Seu compromisso com a inovação tecnológica e a busca por soluções sustentáveis a posicionam como uma líder no campo da automação industrial e gestão de ativos (AVEVA, 2024).

2.5 Modbus TCP/IP

Criado pela empresa Modicon em 1979, o protocolo Modbus foi inicialmente utilizado em CLPs. Por ser um protocolo simples e aberto, podia ser facilmente implementado na maioria dos equipamentos industriais com comunicação em rede.

Para aplicação em arquiteturas com meio físico Ethernet, o Modbus conta com um protocolo de comunicação TCP (*Transmission Control Protocol*), uma sigla em inglês que significa Protocolo de Controle de Transmissão. Amplamente utilizado em equipamentos de automação, este modelo opera através de uma estrutura de aplicação que distribui tarefas entre fornecedores de serviço, os “servidores”, e aqueles que necessitam dessas informações, os “clientes” (ALTUS, 2017).

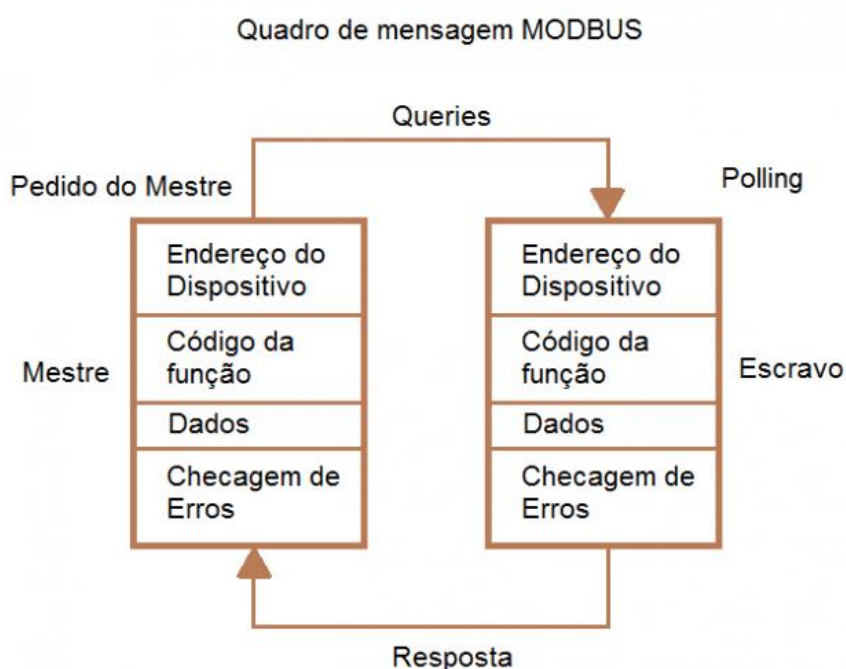
Sua implementação em uma infraestrutura pode ser feita através de switches ou hubs industriais, utilizando cabeamento de par trançado com blindagem (FTP - *Foil Twisted Pair*) e conectores RJ45. Isso garante uma comunicação de 100Mbps a 10Gbps, com uma distância máxima que varia de 100m a 200m, dependendo do tipo de cabo e método de instalação escolhido.

A comunicação é feita sem restrições de autenticação, permitindo que cada usuário se conecte diretamente ao servidor, estabelecendo uma comunicação um a um, na qual o servidor aguarda os pedidos para atendê-los. Esse protocolo pode ser utilizado em Controladores Lógicos Programáveis (CLPs) ou em sistemas supervisórios, como no caso abordado neste projeto.

2.5.1 Funcionamento do Protocolo Modbus

O protocolo Modbus opera com base na lógica de mestre e escravo. (FIGURA 12) O mestre envia solicitações de dados aos escravos, que respondem com as informações solicitadas. Esses dados podem ser discretos (0 e 1, "On" e "Off", "High" e "Low") ou valores numéricos, como leituras de sensores de temperatura, presença, pressão, entre outros (EMBARCADOS, 2014).

FIGURA 12 - QUADRO DE MENSAGENS PARA PROTOCÓLO MODBUS



Fonte: EMBARCADOS, 2014. Disponível em: <<https://embarcados.com.br/protocolo-modbus/>>.

Acesso em: 07 de Jun. 2024

Ao utilizar o meio físico Ethernet, o protocolo MODBUS opera com o mecanismo de controle de acesso CSMA/CD (*Carrier Sense Multiple Access with Collision Detection*), característico das redes Ethernet, e adota um modelo de comunicação cliente-servidor (EMBARCADOS, 2014).

2.5.1.1 Modos de transmissão

O protocolo Modbus pode se comunicar de duas maneiras: RTU e ASCII. Esses modos determinam como os bytes são transmitidos e como as informações são empacotadas e descompactadas. Não é possível utilizar ambos os modos simultaneamente em uma única rede.

2.5.1.2 Endereçamento

O protocolo Modbus possui um total de 256 endereços, distribuídos da seguinte maneira:

- 0: Endereço de broadcast, utilizado quando o mestre envia uma mensagem que deve ser recebida por todos os escravos.
- 1 a 247: Endereços disponíveis para os escravos.
- 248 a 255: Endereços reservados.

O mestre não é atribuído a um endereço específico, apenas os escravos devem ter um endereço definido.

2.5.1.3 Código da Função

O protocolo Modbus atua com diferentes funções para acessar um tipo específico de dado.

TABELA 1 - FUNÇÕES NO PROTOCOLO MODBUS

Código da função	Descrição
1	Leitura de bloco de bits do tipo coil(saída discreta).
2	Leitura de bloco de bits do tipo entradas discretas.
3	Leitura de bloco de registradores do tipo holding.

4	Leitura de bloco de registradores do tipo input.
5	Escrita em um único bit do tipo coil(saída discreta).
6	Escrita em um único registrador do tipo holding.
7	Ler o conteúdo de 8 estados de exceção.
8	Prover uma série de testes para verificação da comunicação e erro internos.
11	Modbus: Obter o contador de eventos.
12	Modbus: Obter um relatório de eventos.
15	Escrita em bloco de bits do tipo coil(saída discreta).
16	Escrita em bloco de registradores do tipo holding.
17	Ler algumas informações do dispositivo.
20	Ler informações de um arquivo.
21	Escrever informações em um arquivo.
22	Modificar o conteúdo de registradores de espera através de operações lógicas.
23	Combina ler e escrever em registradores numa única transação.
24	Ler o conteúdo da fila FIFO de registradores.
43	Identificação do modelo do dispositivo.

Fonte: EMBARCADOS, 2014. Disponível em: <<https://embarcados.com.br/protocolo-modbus/>>.

Acesso em: 07 Jun. 2024

2.5.2 Comunicação TCP

O protocolo Modbus TCP utiliza a pilha TCP/IP para comunicação e adiciona ao quadro Modbus um cabeçalho específico chamado MBAP (*MODBUS Application Protocol*). O modelo de mensagem Modbus TCP/IP é composto pelos seguintes elementos: MBAP, código de função e dados

MBAP: Cabeçalho de 7 bytes, incluindo:

- Identificador de transação: 2 bytes para identificar a transação.
- Identificador de protocolo: 2 bytes, onde 0 (zero) indica Modbus.

- Comprimento: 2 bytes para contagem dos bytes seguintes.
- Identificador de unidade: 1 byte, utilizado para identificar o escravo remoto em uma rede Modbus RTU. (EMBARCADOS, 2014)

O protocolo não inclui um campo para verificação de erros no quadro, pois o frame Ethernet já contém um CRC-32, eliminando a necessidade de um campo de checagem adicional. O cliente Modbus TCP inicia a comunicação TCP com o servidor para realizar as requisições. A porta padrão para conexão com os servidores é a porta TCP 502.

3 DESENVOLVIMENTO DO PROJETO

Nesta seção, é detalhado os processos envolvidos na criação e implementação do produto, abrangendo desde a escolha dos materiais até a construção final. Explorando os métodos de fabricação, as ferramentas utilizadas e os desafios encontrados ao longo do caminho. Além disso, é discutido as considerações ergonômicas, estéticas e funcionais que guiaram nas decisões, assegurando que o resultado atenda tanto às especificações técnicas quanto às necessidades dos usuários.

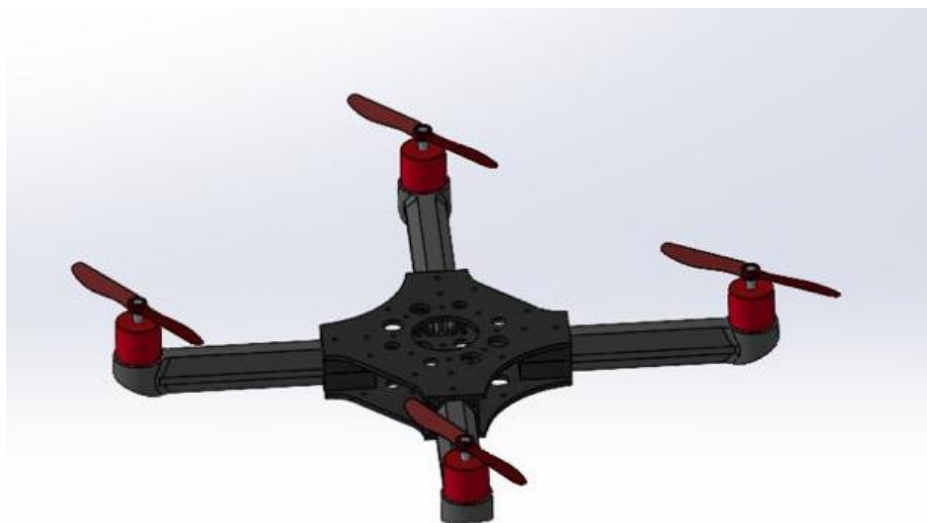
3.1 Estrutura principal

O objetivo é criar uma base sólida e eficiente que não só suporte os desafios do voo, mas também permita a integração harmoniosa de sistemas como motores, controladores e sensores.

3.1.1 Planejamento e Design

Primeiramente, decidir em conjunto o tipo de drone a construir, optando por um quadricóptero devido à sua simplicidade e estabilidade. Utilizar o software SolidWorks para modelar o design do drone, criando componentes principais como as armações, a placa central e os suportes para a bateria. (FIGURA 13)

FIGURA 13 – PROJETO 3D



Fonte: Autoria própria

3.1.1.1 Modelagem Detalhada

Para garantir um encaixe perfeito, ajustar meticulosamente as dimensões dos braços, da placa central e de outros componentes para acomodar os motores e a eletrônica planejada. Projetar espaços específicos para a controladora de voo, os receptores e os ESCs (*Eletronic Speed Control*), com o objetivo de otimizar a organização interna. Focar em reduzir o peso da estrutura utilizando designs vazados e aplicando reforços apenas onde necessário, assegurando a resistência sem comprometer a leveza.

3.1.2 Materiais

3.1.2.1 Material PETG

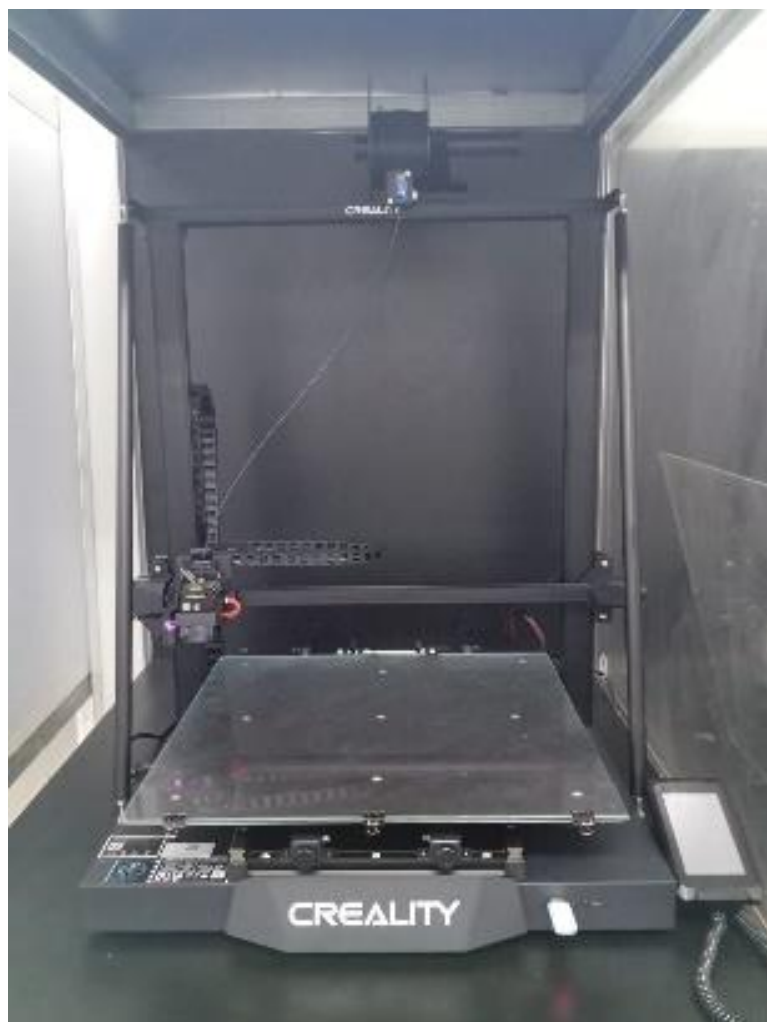
A escolha foi o PETG (Polietileno Tereftalato Glicol) devido à sua durabilidade, resistência ao impacto e flexibilidade, características essenciais para a estrutura do drone. Além disso, o PETG é mais fácil de imprimir do que o ABS, apresentando menos problemas de deformação e não necessitando de uma câmara fechada, o que facilita o processo de fabricação e garante uma maior precisão nas peças produzidas.

3.1.2.2 Impressão 3D

Configurar a impressora 3D com a temperatura do bico entre 230-250°C e a cama aquecida a 70-90°C. Definir a espessura da camada em 0,2 mm, equilibrando qualidade e tempo de impressão, e configurar o preenchimento entre 20-50%. Ajustar a velocidade de impressão para 40-60 mm/s. Para melhorar a aderência do PETG à cama, aplicar spray adesivo, garantindo uma impressão mais estável e precisa.

Modelo da impressora: Creality CR – M4. (FIGURA 14)

FIGURA 14 – IMPRESSORA 3D



Fonte: Autoria Própria

4 MONTAGEM

4.1 Montagem da Estrutura

Com as peças impressas, iniciar a montagem do drone. Fixar os braços na placa central utilizando parafusos e porcas, garantindo uma conexão firme e segura. Em seguida, instalar a controladora de voo, os ESCs e os receptores em suas posições designadas, organizando cuidadosamente os cabos para evitar interferências e garantir um fluxo de sinal eficiente. Antes de integrar toda a eletrônica, testar a robustez da estrutura para assegurar que não há falhas ou pontos de fragilidade, garantindo a integridade do drone durante o voo.

Nesta etapa, o drone está em fase de testes e preparação de estruturas para impactos. Testar protetores helicoidais, ferros rígidos e leves, além de protetores de componentes que são bons receptores de impactos. Optar pela utilização de silicone PU, uma cola altamente resistente, à prova d'água, de fácil aplicação e modelagem, e extremamente eficaz na absorção de impacto.

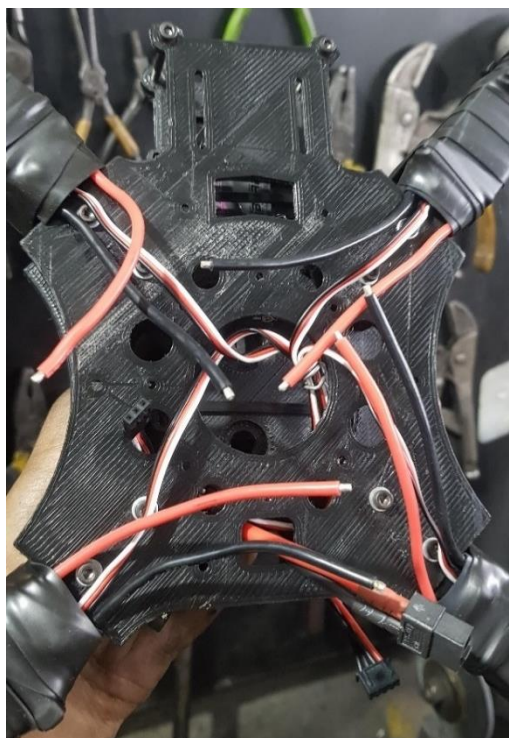
4.1.1 A Base do Drone

A base do drone atua como a espinha dorsal da estrutura, oferecendo suporte e fixação segura para todos os componentes vitais. Os braços, onde os motores e hélices são montados, são fixados à base de maneira a garantir rigidez e minimizar vibrações, o que é essencial para o controle preciso do voo. (FIGURA 15) O microcontrolador, que é o "cérebro" do drone, precisa estar firmemente fixado para evitar deslocamentos que possam afetar os sensores e a eletrônica de controle. Além disso, a bateria, uma das partes mais pesadas do drone, deve ser fixada de forma segura para evitar mudanças no centro de gravidade durante o voo.

A simetria na construção de um drone é fundamental para garantir seu equilíbrio e estabilidade durante o voo. Isso se refere à distribuição uniforme dos componentes em torno do centro de gravidade do drone. Quando a base do drone é projetada de maneira simétrica, cada motor e hélice geram forças que são

equilibradas em relação ao centro de gravidade, contribuindo para um voo estável e controlado.

FIGURA 15 – BASE CENTRAL DO DRONE



Fonte: Autoria Própria

4.1.1.1 Os Braços

Além de sua função estrutural, os braços do drone desempenham um papel fundamental na dissipação de vibrações. Durante o voo, os motores e hélices geram vibrações que, se não forem adequadamente controladas, podem interferir nos sensores e no sistema de controle do drone. Braços bem projetados ajudam a minimizar essas vibrações, garantindo um voo mais estável e um melhor desempenho dos sistemas eletrônicos.

Os braços do drone são estruturas que conectam os motores e hélices à base principal do drone. Eles têm a responsabilidade de transmitir as forças geradas pelos motores e hélices, possibilitando a sustentação e o movimento do drone. Os braços estão ilustrados na FIGURA 16.

FIGURA 16 – OS BRAÇOS DO DRONE



Fonte: Autoria Própria

4.1.2 Materiais

Aqui estão listados os componentes que compõem o drone, bem como os instrumentos utilizados para sua montagem.

4.1.2.1 Motores

Os motores são mais que fundamentais para o drone. Eles são responsáveis por converter energia elétrica em movimento mecânico, gerando a força necessária para erguer e mover o drone no ar. A eficiência, potência e qualidade dos motores influenciam diretamente a capacidade de voo, estabilidade, velocidade e manobrabilidade do drone.

A potência dos motores determina a capacidade de sustentação do drone. Motores mais potentes permitem que o drone carregue cargas maiores e alcance maiores altitudes. Além disso, a potência dos motores influencia a capacidade de resposta do drone aos comandos de controle. Motores de alta qualidade permitem mudanças rápidas de velocidade e direção, melhorando a agilidade e precisão das manobras.

No caso específico, optar por usar motores de 920 Kv (FIGURA 17), que são geralmente utilizados em drones de corrida. Determinar que essa seria a melhor opção para garantir controle de estabilidade, confiabilidade e eficiência energética, evitando perdas e desperdícios de energia, entre outras vantagens.

FIGURA 17 – MOTOR BRUSHLESS



Fonte: Autoria Própria

4.1.2.2 Moduladores de Velocidade

Para controlar as direções que o drone irá se movimentar, é necessário controlar as velocidades dos motores de suas hélices, para isso então, utilizar moduladores de velocidade ESC, que já possuem compatibilidade com os motores brushless, e são muito utilizados na confecção de drones com o propósito de controlar direções. (FIGURA 18)

FIGURA 18 – MODULADOR DE VELOCIDADE ESC



Fonte: Autoria Própria

4.1.2.3 Hélices

As hélices que serão acopladas aos motores serão feitas de PETG, pois é um material leve, o que reduziria o peso do projeto e por consequência exigiria menos potência dos motores, acrescentando o custo-benefício que esse material apresenta em relação aos outros presentes no mercado. (FIGURA 19)

O Drone conta com uma medida de 25cm entre cada eixo. Para que não haja nenhuma interferência e contato entre as hélices.

FIGURA 19 – HÉLICE DE PETG



Fonte: Autoria Própria

4.1.2.4 Bateria de Íons de Lítio

A bateria de íons de lítio é uma fonte de energia recarregável comumente usada em dispositivos eletrônicos devido à sua alta densidade de energia e longa vida útil. Esta bateria específica é a Bateria de Íons de Lítio CNHL (Marca da bateria) 2200mAh 3S 30C, fornecendo 11.1V e 24.42Wh de energia. Produzida pela marca CNHL, é reconhecida pela sua confiabilidade e desempenho ilustrada na FIGURA 20.

FIGURA 20 – BATERIA DE LÍTIO



Fonte: Autoria própria

A bateria, assim como todos os componentes, é de suma importância. No entanto, ela é um dos fatores que mais dificultam a manutenção do voo do drone. O peso da bateria é considerado grande para um componente de pequeno porte. Por isso, a bateria deve ser posicionada de maneira estratégica para não comprometer o desempenho do drone. De acordo com testes realizados, concluiu-se que a melhor posição para a bateria é na base inferior do drone. (FIGURA 21)

FIGURA 21 – LOCALIZAÇÃO DA BATERIA



Fonte: Autoria Própria

4.1.2.5 Carregador

O carregador é um dispositivo utilizado para recarregar a bateria de íons de lítio. Neste caso, o carregador é o IMAXRC B3, compatível com uma ampla faixa de entrada de tensão de 110-240V AC e uma corrente máxima de carga de 800mA. O "Carregador B3" é uma escolha popular para recarregar baterias de drones de forma segura e eficiente. (FIGURA 22)

FIGURA 22 – CARREGADOR DA BATERIA



Fonte: Autoria própria

4.2 Hardware

4.2.1 ESP-32

O cérebro responsável pelo controle das ações do drone será um microcontrolador ESP-32 (FIGURA 24). Também seria possível utilizarmos outros microcontroladores como um Arduino UNO R3 por exemplo, ou até mesmo adquirimos um controlador pronto específico para drones, mas como o objetivo do projeto é aplicarmos o conhecimento adquirido no curso, optamos por utilizar o ESP-32, que é leve, possui um bom custo-benefício em comparação com os recursos oferecidos (FIGURA 23) e pode utilizar a linguagem de programação c++ pelo mesmo software do Arduino, uma linguagem que vai atender aos nossos requisitos. (ELETRONICA OMEGA, 2021)

FIGURA 23 – TABELA COMPARATIVA

	ESP32	ESP8266	ARDUINO UNO R3
Cores	2	1	1
Arquitetura	32 bits	32 bits	8 bits
Clock	160MHz	80MHz	16MHz
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: Euler Oliveira, MasterWalker. Disponível em:

<<https://blogmasterwalkershop.com.br/embarcados/esp32/conhecendo-o-nodemcu-32s-esp32>>. Acesso em: 24 de Jul. 2023.

FIGURA 24 – MICROCONTROLADOR ESP-32



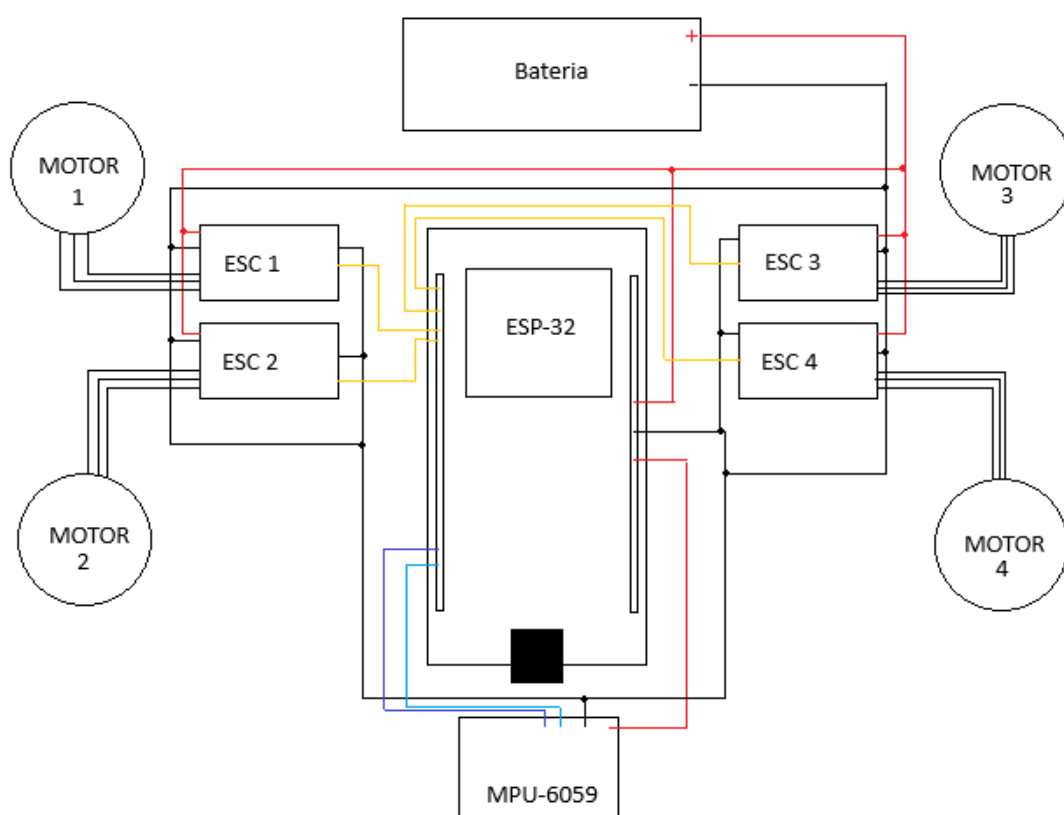
Fonte: Autoria Própria

4.3 Circuito elétrico

O circuito elétrico do nosso drone (FIGURA 25) é projetado para integrar e controlar vários componentes essenciais, garantindo um funcionamento eficiente e seguro. Utilizando um microcontrolador ESP32, o circuito gerencia quatro motores brushless através de seus respectivos ESCs (Electronic Speed Controllers), que são responsáveis por converter sinais PWM (Pulse Width Modulation) em controle de velocidade para os motores. Além disso, um giroscópio MPU6050 é incorporado para fornecer dados de orientação e estabilização, comunicando-se com o ESP32. A alimentação do sistema é fornecida por uma bateria, que é conectada diretamente aos ESCs para fornecer a energia necessária aos motores.

Abaixo um modelo simples de como é o nosso circuito elétrico. (FIGURA 25)

FIGURA 25 – CIRCUITO ELÉTRICO DO DRONE

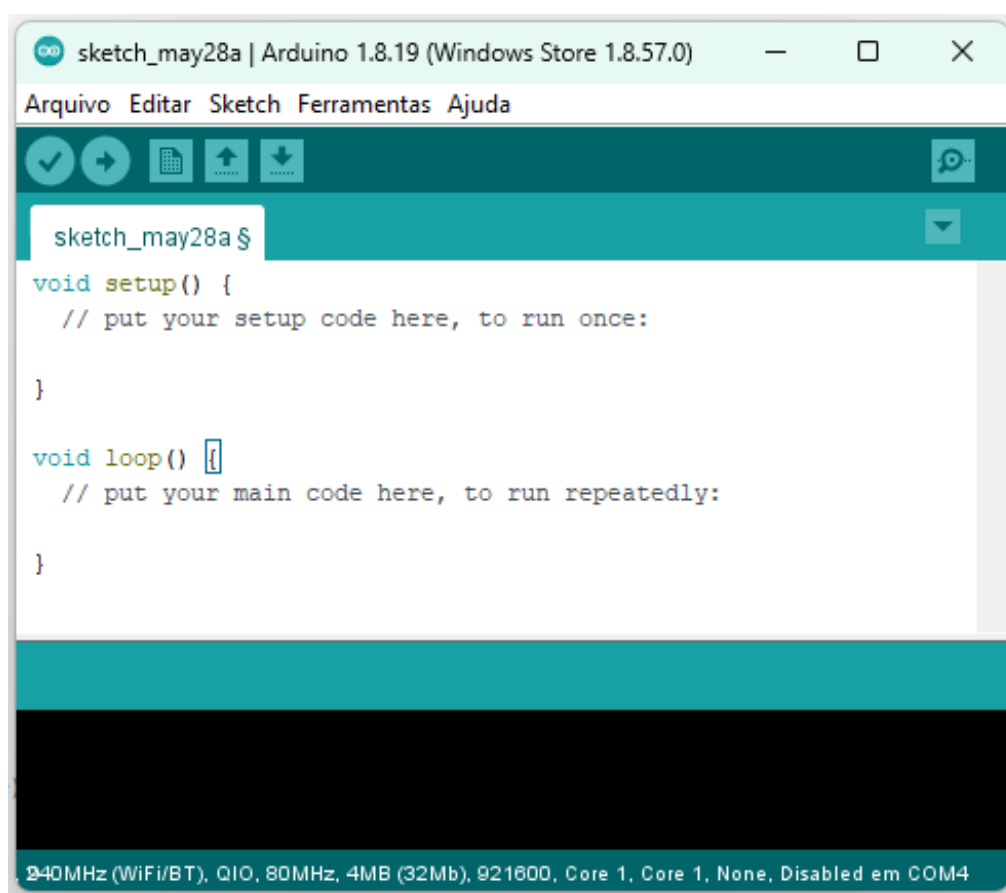


Fonte: Autoria Própria

4.4 Processo de Programação

Todo o funcionamento do drone será programado em C++, que é uma linguagem de programação aceita pelo ESP-32 através do software “Arduino IDE”, além de ser simples e flexível de se utilizar. (FIGURA 26)

FIGURA 26 – SOFTWARE ARDUINO IDE



Fonte: Autoria Própria

Para iniciar a programação do drone, é essencial entender os conceitos do Modbus TCP/IP. Para isso, utilizar um código disponível na biblioteca modbus-esp8266, desenvolvida pelo programador André Sarmento. (FIGURA 27)

FIGURA 27 – PROGRAMA BIBLIOTECA MODBUS

```

Modbus IP - Test LED

/*
  Modbus-Arduino Example - Test Led (Modbus IP ESP8266)
  Control a Led on GPIO0 pin using Write Single Coil Modbus Function
  Original library
  Copyright by André Sarmento Barbosa
  http://github.com/andresarmiento/modbus-arduino

  Current version
  (c)2017 Alexander Emelianov (a.m.emelianov@gmail.com)
  https://github.com/emelianov/modbus-esp8266
*/

#ifdef ESP8266
  #include <ESP8266WiFi.h>
#else //ESP32
  #include <WiFi.h>
#endif
#include <ModbusIP_ESP8266.h>

//Modbus Registers Offsets
const int LED_COIL = 100;
//Used Pins
const int ledPin = 0; //GPIO0

//ModbusIP object
ModbusIP mb;

void setup() {
  Serial.begin(115200);

  WiFi.begin("your_ssid", "your_password");

  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }

  Serial.println("");
  Serial.println("WiFi connected");
  Serial.println("IP address: ");
  Serial.println(WiFi.localIP());

  mb.server();

  pinMode(ledPin, OUTPUT);
  mb.addCoil(LED_COIL);
}

void loop() {
  //Call once inside loop() - all magic here
  mb.task();

  //Attach ledPin to LED_COIL register
  digitalWrite(ledPin, mb.Coil(LED_COIL));
  delay(10);
}

```

Fonte: GitHub. André Sarmento, modbus-esp8266, c2015. Disponível em
 <<https://github.com/andresarmiento/modbus-esp8266/tree/master/arduinoIDE>>. Acesso em: 10 Fev.

Modificar o código para que ele possa controlar quatro LEDs, permitindo testar quatro Coils no sistema. (FIGURA 28 e 29)

FIGURA 28 – PROGRAMA 4 LEDS

```
Modbus IP - Test 4 LEDS

#ifdef ESP8266
#include <ESP8266WiFi.h>
#else //ESP32
#include <WiFi.h>
#endif
#include <ModbusIP_ESP8266.h>

// Modbus Coils
const int LED_COIL = 100; // Endereçamento Modbus do LED1
const int LED_COIL_1 = 101; // Endereçamento Modbus do LED2
const int LED_COIL_2 = 102; // Endereçamento Modbus do LED3
const int LED_COIL_3 = 103; // Endereçamento Modbus do LED4

// Pinos utilizados
const int ledPin = 4; // GPIO4 para o LED1
const int ledPin1 = 15; // GPIO15 para o LED2
const int ledPin2 = 32; // GPIO32 para o LED3
const int ledPin3 = 18; // GPIO18 para o LED4

// Cria objeto ModbusIP
ModbusIP mb;

void setup() {
  Serial.begin(115200);

  // Aqui define-se a rede utilizada "ssid" e "senha"
  WiFi.begin("SEU-SSID", "SUA-SENHA");

  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }
}
```

Fonte: Autoria Própria

FIGURA 29 – PROGRAMA 4 LEDS

```
// No monitor serial você encontrará o IP para o servidor Modbus
Serial.println("");
Serial.println("WiFi connected");
Serial.println("IP address: ");
Serial.println(WiFi.localIP());

mb.server();

// Definir os pinos como saída
pinMode(ledPin, OUTPUT);
pinMode(ledPin1, OUTPUT);
pinMode(ledPin2, OUTPUT);
pinMode(ledPin3, OUTPUT);

// Adiciona Coils para o servidor Modbus
mb.addCoil(LED_COIL);
mb.addCoil(LED_COIL_1);
mb.addCoil(LED_COIL_2);
mb.addCoil(LED_COIL_3);
}

void loop() {
    // Garante o funcionamento do servidor
    mb.task();

    // Atribui os LedPin aos LED_COIL
    digitalWrite(ledPin, mb.Coil(LED_COIL));
    digitalWrite(ledPin1, mb.Coil(LED_COIL_1));
    digitalWrite(ledPin2, mb.Coil(LED_COIL_2));
    digitalWrite(ledPin3, mb.Coil(LED_COIL_3));

    delay(10);
}
```

Fonte: Autoria Própria

Com isso, é concluído que nosso projeto suportaria a entrada de quatro registradores Coils. Em seguida, os testes dos *Holding Registers*, já que os ESCs (*Electronic Speed Controllers*) trabalham com grandezas analógicas. Durante a

simulação, utiliza-se um servo motor controlado por PWM (*Pulse Width Modulation*) para testar a funcionalidade e a precisão dos comandos analógicos.

FIGURA 30 – TESTE HOLDING REGISTERS

```
Modbus IP - Teste V2 - Hreg

#ifdef ESP8266
#include <ESP8266WiFi.h>
#else //ESP32
#include <WiFi.h>
#include <ESP32Servo.h>
#endif
#include <ModbusIP_ESP8266.h>

// Endereçamentos dos registradores Modbus
const int LED_COIL = 100; // Endereçamento Modbus do LED1
const int LED_COIL_1 = 101; // Endereçamento Modbus do LED2
const int LED_COIL_2 = 102; // Endereçamento Modbus do LED3
const int LED_COIL_3 = 103; // Endereçamento Modbus do LED4
const int SERVO_SPEED_REGISTER = 104; // Endereçamento Modbus do Servo Motor

// Pinos usados
const int ledPin = 4; // GPIO4 para o LED1
const int ledPin1 = 15; // GPIO15 para o LED2
const int ledPin2 = 32; // GPIO32 para o LED3
const int ledPin3 = 18; // GPIO18 para o LED4
const int servoPin = 5; // GPIO5 para o servo motor

// Cria o objeto Servo
Servo servo;

// Cria o objeto ModbusIP
ModbusIP mb;

void setup() {
  Serial.begin(115200);

  // Aqui define-se a rede utilizada "ssid" e "senha"
  WiFi.begin("VIVOFIBRA-BFE8", "14491303");

  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }
}
```

Fonte: Autoria Própria

FIGURA 31 – TESTE HOLDING REGISTERS

```

// No monitor serial você encontrará o IP para o servidor Modbus
Serial.println("");
Serial.println("WiFi connected");
Serial.println("IP address: ");
Serial.println(WiFi.localIP());

mb.server();

// Definir os pinos como saída
pinMode(ledPin, OUTPUT);
pinMode(ledPin1, OUTPUT);
pinMode(ledPin2, OUTPUT);
pinMode(ledPin3, OUTPUT);

// Atribui o servo ao pino correto
servo.attach(servoPin);

// Adiciona Coils e Holding Registers para o servidor Modbus
mb.addCoil(LED_COIL);
mb.addCoil(LED_COIL_1);
mb.addCoil(LED_COIL_2);
mb.addCoil(LED_COIL_3);
mb.addHreg(SERVO_SPEED_REGISTER); // Registrador para o Servo Motor
}

void loop() {
  // Garante o funcionamento do servidor
  mb.task();

  // Atribui os LedPin aos LED_COIL
  digitalWrite(ledPin, mb.Coil(LED_COIL));
  digitalWrite(ledPin1, mb.Coil(LED_COIL_1));
  digitalWrite(ledPin2, mb.Coil(LED_COIL_2));
  digitalWrite(ledPin3, mb.Coil(LED_COIL_3));

  // Lê a velocidade a partir do registrador Modbus
  int speed = mb.Hreg(SERVO_SPEED_REGISTER);
  speed = map(speed, 0, 1023, -90, 90); // Mapeia o alcance da velocidade (-90 a 90)

  // "Seta" a velocidade do servo
  int servoSpeed = 90 + speed; // 90 é a posição parado, ajuste para aumentar a velocidade
  servo.write(servoSpeed); // Movimenta o servo de acordo com a velocidade

  delay(10);
}

```

Fonte: Autoria Própria

Finaliza-se atribuindo o controle de velocidade PWM aos ESCs, utilizando o servo motor como base para os testes (FIGURA 32 e 33).

FIGURA 32 – CONTROLE DE VELOCIDADE PWM

```

Modbus IP - Teste V3 - ESCs

#ifdef ESP8266
#include <ESP8266WiFi.h>
#else //ESP32
#include <WiFi.h>
#include <ESP32Servo.h>
#endif
#include <ModbusIP_ESP8266.h>

// Endereçamentos dos registradores Modbus
const int LED_COIL = 100; // Endereçamento Modbus do LED1
const int LED_COIL_1 = 101; // Endereçamento Modbus do LED2
const int LED_COIL_2 = 102; // Endereçamento Modbus do LED3
const int LED_COIL_3 = 103; // Endereçamento Modbus do LED4
const int ESC_SPEED_REGISTER_1 = 104; // Endereçamento Modbus da velocidade do ESC 1 (PWM)
const int ESC_SPEED_REGISTER_2 = 105; // Endereçamento Modbus da velocidade do ESC 2 (PWM)
const int ESC_SPEED_REGISTER_3 = 106; // Endereçamento Modbus da velocidade do ESC 3 (PWM)
const int ESC_SPEED_REGISTER_4 = 107; // Endereçamento Modbus da velocidade do ESC 4 (PWM)

// Pinos usados
const int ledPin = 4; // GPIO4 para o LED1
const int ledPin1 = 15; // GPIO15 para o LED2
const int ledPin2 = 32; // GPIO32 para o LED3
const int ledPin3 = 18; // GPIO18 para o LED4
const int escPin1 = 5; // GPIO5 para o ESC 1 (PWM)
const int escPin2 = 6; // GPIO6 para o ESC 2 (PWM)
const int escPin3 = 7; // GPIO7 para o ESC 3 (PWM)
const int escPin4 = 8; // GPIO8 para o ESC 4 (PWM)

// Cria o objeto ModbusIP
ModbusIP mb;

void setup() {
  Serial.begin(115200);

  // Aqui define-se a rede utilizada "ssid" e "senha"
  WiFi.begin("VIVOFIBRA-BFE8", "14491303");

  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }

  // No monitor serial você encontrará o IP para o servidor Modbus
  Serial.println("");
  Serial.println("WiFi connected");
  Serial.println("IP address: ");
  Serial.println(WiFi.localIP());

  mb.server();
}

```

Fonte: Autoria Própria

FIGURA 33 – CONTROLE DE VELOCIDADE PWM

```

// Definir os pinos como saída
pinMode(ledPin, OUTPUT);
pinMode(ledPin1, OUTPUT);
pinMode(ledPin2, OUTPUT);
pinMode(ledPin3, OUTPUT);

// Adiciona Coils e Holding Registers para o servidor Modbus
mb.addCoil(LED_COIL);
mb.addCoil(LED_COIL_1);
mb.addCoil(LED_COIL_2);
mb.addCoil(LED_COIL_3);
mb.addHreg(ESC_SPEED_REGISTER_1); // Registrador para o ESC 1 (PWM)
mb.addHreg(ESC_SPEED_REGISTER_2); // Registrador para o ESC 2 (PWM)
mb.addHreg(ESC_SPEED_REGISTER_3); // Registrador para o ESC 3 (PWM)
mb.addHreg(ESC_SPEED_REGISTER_4); // Registrador para o ESC 4 (PWM)
}

void loop() {
  // Garante o funcionamento do servidor
  mb.task();

  // Atribui os LedPin aos LED_COIL
  digitalWrite(ledPin, mb.Coil(LED_COIL));
  digitalWrite(ledPin1, mb.Coil(LED_COIL_1));
  digitalWrite(ledPin2, mb.Coil(LED_COIL_2));
  digitalWrite(ledPin3, mb.Coil(LED_COIL_3));

  // Lê a velocidade desejada dos registradores Modbus
  int speed1 = mb.Hreg(ESC_SPEED_REGISTER_1);
  int speed2 = mb.Hreg(ESC_SPEED_REGISTER_2);
  int speed3 = mb.Hreg(ESC_SPEED_REGISTER_3);
  int speed4 = mb.Hreg(ESC_SPEED_REGISTER_4);

  // Mapeia a a faixa de velocidade PWM (1000 a 2000)
  speed1 = map(speed1, 0, 1023, 1000, 2000);
  speed2 = map(speed2, 0, 1023, 1000, 2000);
  speed3 = map(speed3, 0, 1023, 1000, 2000);
  speed4 = map(speed4, 0, 1023, 1000, 2000);

  // "Seta" a velocidade dos ESCs
  analogWrite(escPin1, speed1);
  analogWrite(escPin2, speed2);
  analogWrite(escPin3, speed3);
  analogWrite(escPin4, speed4);

  delay(10);
}

```

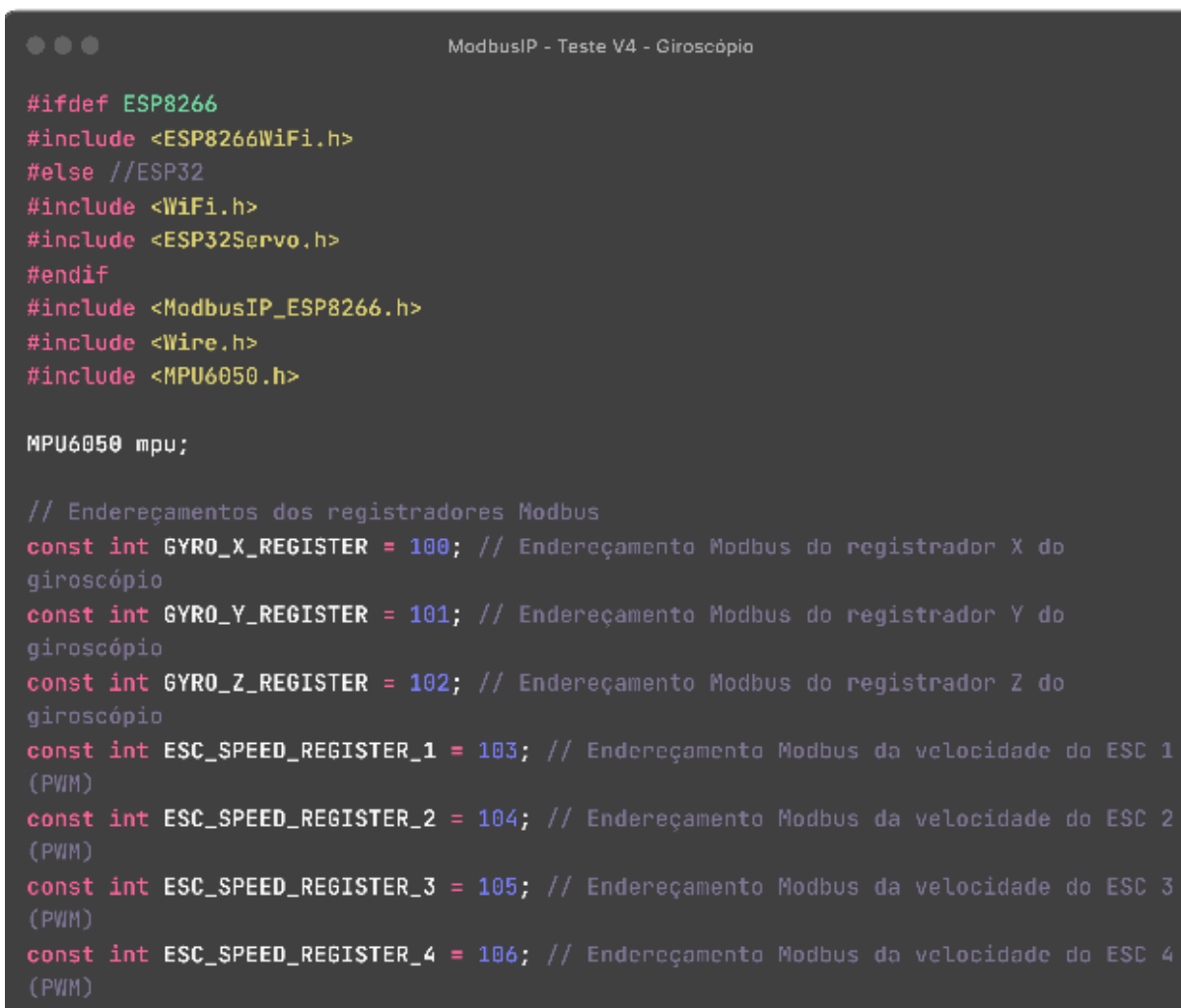
Fonte: Autoria Própria

Ao finalizar estes testes, é tomada a decisão de remover os LEDs, uma vez que não seriam necessários para a funcionalidade final do programa. Inicialmente, esses

LEDs foram empregados exclusivamente para a realização de testes preliminares e de validação do sistema.

Além disso, como parte da otimização do sistema e visando aprimorar a estabilidade do voo, procede com a adição de registros de entrada específicos para o giroscópio. Essa medida foi implementada com o objetivo de assegurar um desempenho consistente e seguro durante a operação do dispositivo. Dessa forma, a integração dos *inputs registers* proporcionará um controle mais preciso e confiável sobre as variáveis relacionadas ao movimento e orientação do equipamento, contribuindo significativamente para sua eficácia e eficiência operacional.

FIGURA 34 – ADICIONANDO INPUT REGISTER



```
ModbusIP - Teste V4 - Giroscópio

#ifdef ESP8266
#include <ESP8266WiFi.h>
#else //ESP32
#include <WiFi.h>
#include <ESP32Servo.h>
#endif
#include <ModbusIP_ESP8266.h>
#include <Wire.h>
#include <MPU6050.h>

MPU6050 mpu;

// Endereçamentos dos registradores Modbus
const int GYRO_X_REGISTER = 100; // Endereçamento Modbus do registrador X do
giroscópio
const int GYRO_Y_REGISTER = 101; // Endereçamento Modbus do registrador Y do
giroscópio
const int GYRO_Z_REGISTER = 102; // Endereçamento Modbus do registrador Z do
giroscópio
const int ESC_SPEED_REGISTER_1 = 103; // Endereçamento Modbus da velocidade do ESC 1
(PWM)
const int ESC_SPEED_REGISTER_2 = 104; // Endereçamento Modbus da velocidade do ESC 2
(PWM)
const int ESC_SPEED_REGISTER_3 = 105; // Endereçamento Modbus da velocidade do ESC 3
(PWM)
const int ESC_SPEED_REGISTER_4 = 106; // Endereçamento Modbus da velocidade do ESC 4
(PWM)
```

Fonte: Autoria Própria

FIGURA 35 – ADICIONANDO INPUT REGISTER

```

// Pinos usados
const int escPin1 = 13; // GPIO5 para o ESC 1 (PWM)
const int escPin2 = 12; // GPIO6 para o ESC 2 (PWM)
const int escPin3 = 14; // GPIO7 para o ESC 3 (PWM)
const int escPin4 = 27; // GPIO8 para o ESC 4 (PWM)
const int i2cSDA = 21; // GPIO21 para SDA do I2C
const int i2cSCL = 22; // GPIO22 para SCL do I2C

// Cria o objeto ModbusIP
ModbusIP mb;

void setup() {
  Serial.begin(115200);

  // Aqui define-se a rede utilizada "ssid" e "senha"
  WiFi.begin("VIVOFIBRA-BFE8", "14491303");

  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }

  // No monitor serial você encontrará o IP para o servidor Modbus
  Serial.println("");
  Serial.println("WiFi connected");
  Serial.println("IP address: ");
  Serial.println(WiFi.localIP());

  mb.server();

  // Inicializa o I2C e o MPU6050
  Wire.begin(i2cSDA, i2cSCL);
  mpu.initialize();
  if (mpu.testConnection()) {
    Serial.println("MPU6050 conectado");
  } else {
    Serial.println("Falha na conexão com MPU6050");
  }
}

```

Fonte: Autoria Própria

FIGURA 36 – ADICIONANDO INPUT REGISTER

```

// Adiciona Holding Registers para o servidor Modbus
mb.addIreg(GYRO_X_REGISTER); // Registrador para o eixo X do giroscópio
mb.addIreg(GYRO_Y_REGISTER); // Registrador para o eixo Y do giroscópio
mb.addIreg(GYRO_Z_REGISTER); // Registrador para o eixo Z do giroscópio
mb.addHreg(ESC_SPEED_REGISTER_1); // Registrador para o ESC 1 (PWM)
mb.addHreg(ESC_SPEED_REGISTER_2); // Registrador para o ESC 2 (PWM)
mb.addHreg(ESC_SPEED_REGISTER_3); // Registrador para o ESC 3 (PWM)
mb.addHreg(ESC_SPEED_REGISTER_4); // Registrador para o ESC 4 (PWM)
}

void loop() {
  // Garante o funcionamento do servidor
  mb.task();

  // Lê os valores do giroscópio
  int16_t gx, gy, gz;
  mpu.getRotation(&gx, &gy, &gz);

  // Atualiza os registradores Modbus com os valores do giroscópio
  mb.Ireg(GYRO_X_REGISTER, gx);
  mb.Ireg(GYRO_Y_REGISTER, gy);
  mb.Ireg(GYRO_Z_REGISTER, gz);

  // Lê a velocidade desejada dos registradores Modbus
  int speed1 = mb.Hreg(ESC_SPEED_REGISTER_1);
  int speed2 = mb.Hreg(ESC_SPEED_REGISTER_2);
  int speed3 = mb.Hreg(ESC_SPEED_REGISTER_3);
  int speed4 = mb.Hreg(ESC_SPEED_REGISTER_4);

  // Mapeia a faixa de velocidade PWM (1000 a 2000)
  speed1 = map(speed1, 0, 1023, 1000, 2000);
  speed2 = map(speed2, 0, 1023, 1000, 2000);
  speed3 = map(speed3, 0, 1023, 1000, 2000);
  speed4 = map(speed4, 0, 1023, 1000, 2000);

  // "Seta" a velocidade dos ESCs
  analogWrite(escPin1, speed1);
  analogWrite(escPin2, speed2);
  analogWrite(escPin3, speed3);
  analogWrite(escPin4, speed4);

  delay(10);
}

```

Fonte: Autoria Própria

A versão final do código incorpora uma sofisticada lógica de controle de voo, que é fundamental para a operação precisa e estável do drone. Esta lógica de voo é responsável por manipular as variáveis dos Controladores Eletrônicos de Velocidade (ESCs), permitindo assim o controle eficaz das direções e movimentos do drone durante o voo.

Além disso, uma importante adição ao código é a implementação de uma velocidade base para os ESCs. Essa velocidade base desempenha um papel crucial na determinação da altitude de voo do drone, proporcionando um referencial fundamental para o controle vertical do dispositivo.

Dessa forma, ao integrar essa funcionalidade ao código, conseguimos não apenas controlar as direções e movimentos do drone de forma precisa, mas também garantir uma altura de voo consistente e estável, contribuindo assim para uma experiência de voo segura e controlada (FIGURA 37 a 40).

FIGURA 37 – CÓDIGO FINAL

```

#ifdef ESP8266
#include <ESP8266WiFi.h>
#else //ESP32
#include <WiFi.h>
#include <ESP32Servo.h>
#endif
#include <ModbusIP_ESP8266.h>
#include <Wire.h>
#include <MPU6050.h>

MPU6050 mpu;

// Endereçamentos dos registradores Modbus
const int GYRO_X_REGISTER = 100; // Endereçamento Modbus do registrador X do giroscópio
const int GYRO_Y_REGISTER = 101; // Endereçamento Modbus do registrador Y do giroscópio
const int GYRO_Z_REGISTER = 102; // Endereçamento Modbus do registrador Z do giroscópio
const int ESC_SPEED_REGISTER_1 = 103; // Endereçamento Modbus da velocidade do ESC 1 (PWM)
const int ESC_SPEED_REGISTER_2 = 104; // Endereçamento Modbus da velocidade do ESC 2 (PWM)
const int ESC_SPEED_REGISTER_3 = 105; // Endereçamento Modbus da velocidade do ESC 3 (PWM)
const int ESC_SPEED_REGISTER_4 = 106; // Endereçamento Modbus da velocidade do ESC 4 (PWM)
const int COMMAND_UP_DOWN = 107; // Comando Modbus para subir/descer
const int COMMAND_FORWARD = 108; // Comando Modbus para frente
const int COMMAND_BACKWARD = 109; // Comando Modbus para trás
const int COMMAND_LEFT = 110; // Comando Modbus para esquerda
const int COMMAND_RIGHT = 111; // Comando Modbus para direita

```

Fonte: Autoria Própria

FIGURA 38 – CÓDIGO FINAL

```

// Pinos usados
const int escPin1 = 15; // GPIO5 para o ESC 1 (PWM)
const int escPin2 = 2; // GPIO6 para o ESC 2 (PWM)
const int escPin3 = 4; // GPIO7 para o ESC 3 (PWM)
const int escPin4 = 5; // GPIO8 para o ESC 4 (PWM)
const int i2cSDA = 21; // GPIO21 para SDA do I2C
const int i2cSCL = 22; // GPIO22 para SCL do I2C

// Cria o objeto ModbusIP
ModbusIP mb;
Servo esc1, esc2, esc3, esc4; // Cria objetos Servo para controlar os ESCs

void setup() {
    Serial.begin(115200);

    // Aqui define-se a rede utilizada "ssid" e "senha"
    WiFi.begin("SEU-SSID", "SUA-SENHA");

    while (WiFi.status() != WL_CONNECTED) {
        delay(500);
        Serial.print(".");
    }

    // No monitor serial você encontrará o IP para o servidor Modbus
    Serial.println("");
    Serial.println("WiFi connected");
    Serial.println("IP address: ");
    Serial.println(WiFi.localIP());

    mb.server();

    // Inicializa o I2C e o MPU6050
    Wire.begin(i2cSDA, i2cSCL);
    mpu.initialize();
    if (mpu.testConnection()) {
        Serial.println("MPU6050 conectado");
    } else {
        Serial.println("Falha na conexão com MPU6050");
    }

    // Adiciona Holding Registers para o servidor Modbus
    mb.addIreg(GYRO_X_REGISTER); // Registrador para o eixo X do giroscópio
    mb.addIreg(GYRO_Y_REGISTER); // Registrador para o eixo Y do giroscópio
    mb.addIreg(GYRO_Z_REGISTER); // Registrador para o eixo Z do giroscópio
    mb.addHreg(ESC_SPEED_REGISTER_1); // Registrador para o ESC 1 (PWM)
    mb.addHreg(ESC_SPEED_REGISTER_2); // Registrador para o ESC 2 (PWM)

```

Fonte: Autoria Própria

FIGURA 39 – CÓDIGO FINAL

```

mb.addHreg(ESC_SPEED_REGISTER_3); // Registrador para o ESC 3 (PWM)
mb.addHreg(ESC_SPEED_REGISTER_4); // Registrador para o ESC 4 (PWM)
mb.addHreg(COMMAND_UP_DOWN); // Comando para subir/descer
mb.addHreg(COMMAND_FORWARD); // Comando para frente
mb.addHreg(COMMAND_BACKWARD); // Comando para trás
mb.addHreg(COMMAND_LEFT); // Comando para esquerda
mb.addHreg(COMMAND_RIGHT); // Comando para direita

// Anexa os objetos Servo aos pinos ESC
esc1.attach(escPin1);
esc2.attach(escPin2);
esc3.attach(escPin3);
esc4.attach(escPin4);

// Inicializa os ESCs com valor mínimo
esc1.writeMicroseconds(1000);
esc2.writeMicroseconds(1000);
esc3.writeMicroseconds(1000);
esc4.writeMicroseconds(1000);

delay(2000); // Espera para garantir a inicialização dos ESCs
}

void loop() {
    // Garante o funcionamento do servidor
    mb.task();

    // Lê os valores do giroscópio
    int16_t gx, gy, gz;
    mpu.getRotation(&gx, &gy, &gz);

    // Atualiza os registradores Modbus com os valores do giroscópio
    mb.Ireg(GYRO_X_REGISTER, gx);
    mb.Ireg(GYRO_Y_REGISTER, gy);
    mb.Ireg(GYRO_Z_REGISTER, gz);

    // Lê a velocidade base desejada do registrador Modbus
    int baseSpeed = mb.Hreg(COMMAND_UP_DOWN);

    // Lê os comandos de movimentação
    int forward = mb.Hreg(COMMAND_FORWARD);
    int backward = mb.Hreg(COMMAND_BACKWARD);
    int left = mb.Hreg(COMMAND_LEFT);
    int right = mb.Hreg(COMMAND_RIGHT);

```

FIGURA 40 – CÓDIGO FINAL

```

// Inicializa os valores de velocidade dos motores com a velocidade base
int speed1 = baseSpeed;
int speed2 = baseSpeed;
int speed3 = baseSpeed;
int speed4 = baseSpeed;

// Ajusta as velocidades dos motores com base nos comandos de movimentação
if (forward > 0) {
    speed1 += forward;
    speed2 += forward;
    speed3 -= forward;
    speed4 -= forward;
}

if (backward > 0) {
    speed1 -= backward;
    speed2 -= backward;
    speed3 += backward;
    speed4 += backward;
}

if (left > 0) {
    speed1 -= left;
    speed2 += left;
    speed3 += left;
    speed4 -= left;
}

if (right > 0) {
    speed1 += right;
    speed2 -= right;
    speed3 -= right;
    speed4 += right;
}

// Mapeia a faixa de velocidade PWM (1000 a 2000)
speed1 = map(speed1, 0, 1023, 1000, 2000);
speed2 = map(speed2, 0, 1023, 1000, 2000);
speed3 = map(speed3, 0, 1023, 1000, 2000);
speed4 = map(speed4, 0, 1023, 1000, 2000);

// "Seta" a velocidade dos ESCs
esc1.writeMicroseconds(speed1);
esc2.writeMicroseconds(speed2);
esc3.writeMicroseconds(speed3);
esc4.writeMicroseconds(speed4);

delay(10);
}

```

4.5 Comunicação no AVEVA

A comunicação com a AVEVA, no contexto dos "coils" (bobinas), envolve a integração e troca de informações entre sistemas de automação industrial e a plataforma AVEVA. Coils são pontos de dados binários utilizados em controladores lógicos programáveis (PLCs) para monitorar e controlar dispositivos e processos. A AVEVA facilita a supervisão e o gerenciamento desses pontos de dados, permitindo a visualização em tempo real e o controle remoto dos sistemas. A comunicação eficiente garante que os dados dos coils sejam transmitidos de forma precisa e rápida, suportando decisões operacionais e manutenção preditiva. Através de protocolos de comunicação padronizados, a integração entre coils e a plataforma AVEVA otimiza a operação industrial, melhora a eficiência e reduz o tempo de inatividade.

No caso foi definido quatro Coils, que serão os quatro motores do nosso drone (FIGURA 43). Essas quatro Coils serviram para realizar o primeiro teste, com a tela que seria utilizada para testar a comunicação com o software (FIGURA 42).

FIGURA 41 – TAGS DO SUPERVISÓRIO

Tags do Projeto x MOTCP004.DRV MOTCP003.DRV						
	Nome	Tamanho do Vetor	Tipo	Descrição	Escopo	Disponibilidade do UAExteno
2	Do2	0	Booleana	LED 2	Servidor	Desabilitado
3	Do3	0	Booleana	LED 3	Servidor	Desabilitado
4	Do4	0	Booleana	LED 4	Servidor	Desabilitado
5	Hreg1	0	Inteira	ESC 1	Servidor	Desabilitado
6	Hreg2	0	Inteira	ESC 2	Servidor	Desabilitado
7	Hreg3	0	Inteira	ESC 3	Servidor	Desabilitado
8	Hreg4	0	Inteira	ESC 4	Servidor	Desabilitado
9	SubirDescer	0	Inteira	Sobe ou desce	Servidor	Desabilitado
10	Frente	0	Inteira	Vai para frente	Servidor	Desabilitado
11	Tras	0	Inteira	Vai para trás	Servidor	Desabilitado
12	Esquerda	0	Inteira	Vai para a esquerda	Servidor	Desabilitado
13	Direita	0	Inteira	Vai para a direita	Servidor	Desabilitado
14	EixoY	0	Inteira	Giroscópio Eixo Y	Servidor	Desabilitado
15	EixoX	0	Inteira	Giroscópio Eixo X	Servidor	Desabilitado
16	EixoZ	0	Inteira	Giroscópio Eixo Z	Servidor	Desabilitado

Fonte: Autoria Própria

FIGURA 42 – TELA EXPERIMENTAL DO SUPERVISÓRIO



Fonte: Autoria Própria

FIGURA 43 – COILS AVEVA

Descrição:

Coils ☐ Aumentar prioridade

Disparo de Leitura: Habilita Leitura: Leitura Concluída: Estado da Leitura:

Disparo de Escrita: Habilita Escrita Automática: Escrita Concluída: Estado da Escrita:

Estação: Cabeçalho: ☐ Min: Máx:

	Nome da Tag	Endereço	Div	Adicionar
	Filtro de Texto	Filtro de Texto	Filtro de Te:	Filtro de Te:
1	Do1	101	0.000000	
2	Do2	102		
3	Do3	103		
4	Do4	104		
*				
*				
*				

Fonte: Autoria Própria

A comunicação com a AVEVA, no contexto dos "*holding registers*" (registros de retenção), envolve a integração e troca de informações entre sistemas de automação industrial e a plataforma AVEVA. *Holding registers* são pontos de dados de 16 bits

utilizados em controladores lógicos programáveis (PLCs) para armazenar informações numéricas, como *setpoints*, contadores e valores de configuração. A AVEVA facilita a supervisão e o gerenciamento desses registros, permitindo a visualização em tempo real e o ajuste remoto dos parâmetros operacionais. Uma comunicação eficiente garante que os dados dos *holdings registers* sejam transmitidos de forma precisa e rápida, suportando decisões operacionais e manutenção preditiva. Através de protocolos de comunicação padronizados, a integração entre *holding registers* e a plataforma AVEVA otimiza a operação industrial, melhora a eficiência e reduz o tempo de inatividade.

Neste projeto é utilizado 4 desses para que possa ser supervisionado o estado dos motores no primeiro teste, para testar se seria possível utilizar 4 sinais de *holding registers*.

FIGURA 44 – HOLDING REGISTERS AVEVA

Descrição:
 Leitura e escrita em registradores ☒ Aumentar prioridade

Disparo de Leitura: Habilita Leitura: Leitura Concluída: Estado da Leitura:

Disparo de Escrita: Habilita Escrita Automática: Escrita Concluída: Estado da Escrita:

Estação: Cabeçalho: ☐ Mín: Máx:

	Nome da Tag	Endereço	Div	Adicionar
	 Filtro de Texto	 Filtro de Texto	 Filtro de Te:	 Filtro de Te:
1	Hreg1	104		
2	Hreg2	105		
3	Hreg3	106		
4	Hreg4	107		
*				
*				
*				

Fonte: Autoria Própria

4.5.1 Lógica de Voo por Modbus

Inicialmente, foi planejado empregar a tabela matemática disponível no software para implementar a lógica de voo do drone. No entanto, durante a fase de desenvolvimento, teve-se alguns conflitos relacionados ao recebimento de sinais, o que levou a reconsiderar essa abordagem. Diante disso, o melhor foi integrar a lógica diretamente no código principal, como ilustrado na FIGURA 45. Essa mudança exigiu a inclusão de novas tags e endereços para acessar e manipular a tabela de leitura e escrita em registradores, garantindo assim a integração eficaz da lógica de voo ao sistema geral do drone.

FIGURA 45 – LISTA COMPLETA DE TAGS

Tags do Projeto **MOTCP004.DRV** x **MOTCP003.DRV**

Descrição:
 Leitura e escrita em registradores ☒ Aumentar prioridade

Disparo de Leitura: Habilita Leitura: Leitura Concluída: Estado da Leitura:

Disparo de Escrita: Habilita Escrita Automática: Escrita Concluída: Estado da Escrita:

Estação: Cabeçalho: ☐ Min: Máx:

	Nome da Tag	Endereço	Div	Adicionar
	Filtro de Texto	Filtro de Texto	Filtro de Te:	Filtro de Te:
1	Hreg1	103		
2	Hreg2	104		
3	Hreg3	105		
4	Hreg4	106		
5	SubirDescer	107		
6	Frente	108		
7	Tras	109		
5	Esquerda	110		
6	Direita	111		

*

Fonte: Autoria Própria

Como parte do processo de aprimoramento do sistema de controle do drone, foi implementada a adição dos eixos do giroscópio como *input registers*. Estes registros foram especificamente designados para monitorar os movimentos e orientações do drone ao longo dos diferentes eixos, contribuindo assim para assegurar a estabilidade e precisão durante o voo.

A inclusão desses *inputs registers* representa um avanço significativo no monitoramento e controle das variáveis críticas de voo, proporcionando ao sistema uma capacidade mais refinada de resposta aos estímulos externos e internos. Dessa forma, ao integrar essa funcionalidade ao sistema, estamos garantindo não apenas um voo mais estável e controlado, mas também uma maior segurança e confiabilidade operacional do drone em diversas condições de voo.

FIGURA 46 – ADIÇÃO DE EIXOS

Tags do Projeto MOTCP004.DRV MOTCP003.DRV x

Descrição:
 Leitura de registradores ☒ Aumentar prioridade

Disparo de Leitura: Habilita Leitura: Leitura Concluída: Estado da Leitura:

Disparo de Escrita: Habilita Escrita Automática: Escrita Concluída: Estado da Escrita:

Estação: Cabeçalho: ☐ Mín: Máx:

	Nome da Tag	Endereço	Div	Adicionar
	Filtro de Texto	Filtro de Texto	Filtro de Te:	Filtro de Te:
1	EixoX	100		
2	EixoY	101		
3	EixoZ	102		
*				

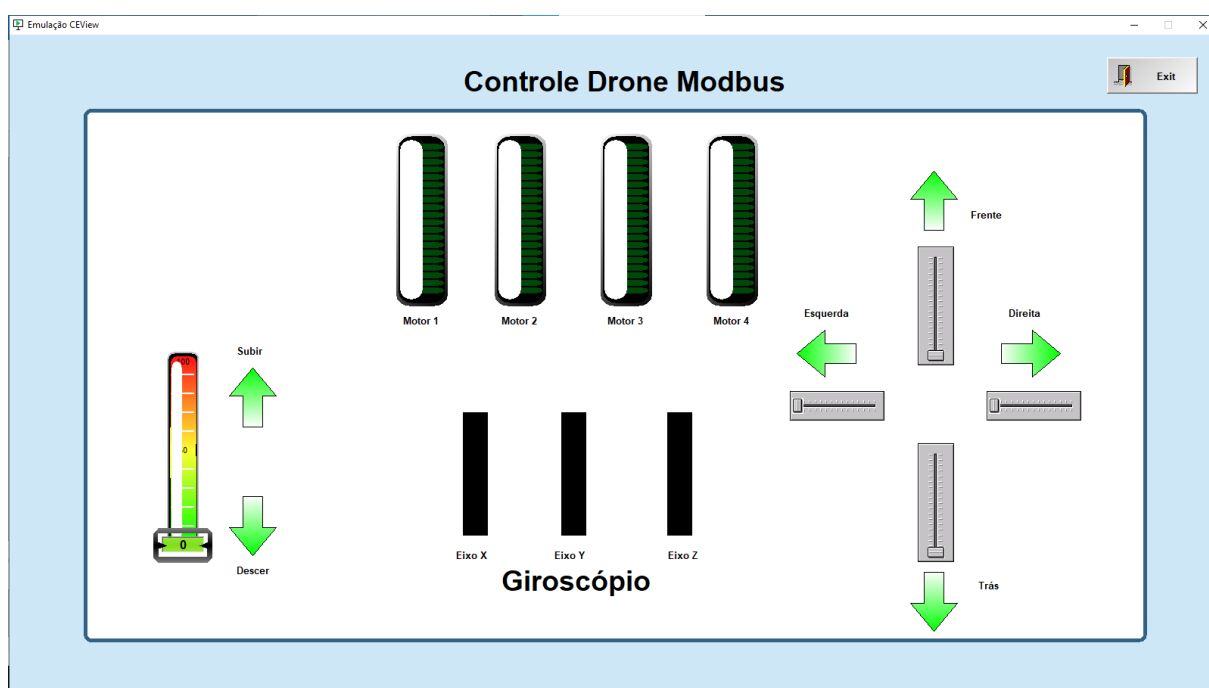
Fonte: Autoria Própria

4.5.2 Tela Principal do Supervisório

O design final selecionado para a tela principal de controle do drone apresenta um *layout* intuitivo e funcional. (FIGURA 47) No lado esquerdo, há um *slider* dedicado

ao controle do movimento de subida e descida do drone, enquanto no lado direito, encontram-se quatro *sliders* individualizados para cada direção de voo. Além disso, são incorporadas quatro barras de LED, as quais indicam a potência de cada motor do drone, proporcionando uma visualização rápida e precisa do seu desempenho. Complementando o *layout*, são incluídas três barras de LED destinadas a mostrar os sinais dos eixos do giroscópio, contribuindo para o monitoramento e a estabilidade durante o voo.

FIGURA 47 – TELA SUPERVISÓRIO



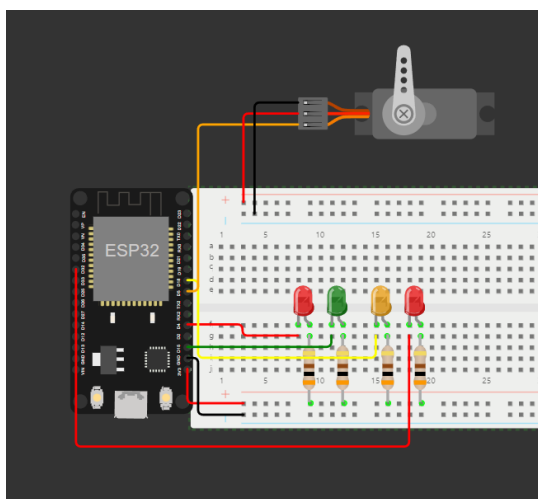
Fonte: Autoria Própria

5 TESTES E RESULTADOS

5.1 Testes do Programa

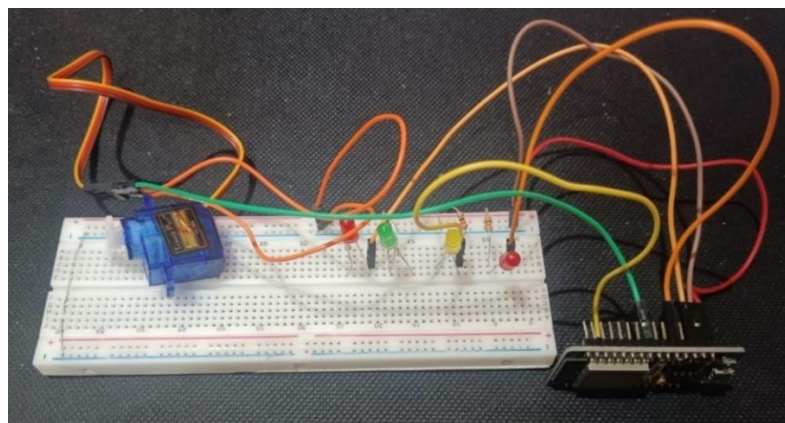
Durante a pesquisa e decisão dos códigos, foram realizados alguns testes de simulação com LEDs e motores servo (FIGURA 48 e 59) para garantir que não ocorressem imprevistos com nossos motores principais. Esses testes permitiram validar a lógica de controle e a comunicação entre os componentes, assegurando que o sistema funcionasse corretamente antes de ser implementado com os motores principais. Dessa forma, foi possível identificar e corrigir possíveis problemas antecipadamente, garantindo um funcionamento seguro e eficiente do drone.

FIGURA 48 – CIRCUITO DE TESTES



Fonte: Autoria Própria

FIGURA 49 – FOTO REAL DO CIRCUITO



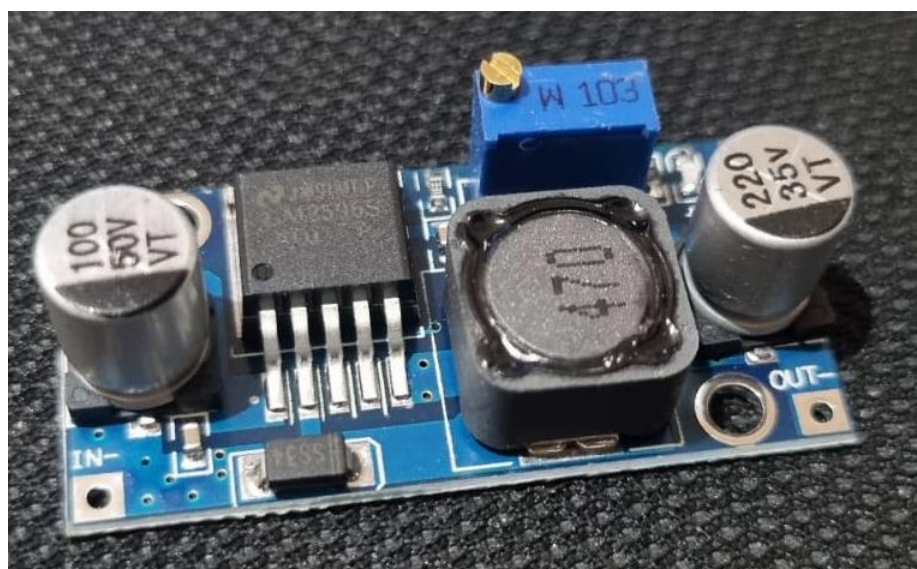
Fonte: Autoria Própria

5.2 Testes do Drone

Durante os primeiros testes com o drone completo, foi cometido um erro que comprometeu o microcontrolador ESP32. A bateria fornece uma tensão de 11V, que é significativamente superior à suportada pelo ESP32. Infelizmente, não foi considerada esse detalhe crucial durante a montagem inicial e foi conectado ao ESP32 diretamente à bateria, resultando na queima do microcontrolador devido à sobrecarga de tensão.

Essa falha mostra a importância de verificar minuciosamente as especificações de todos os componentes envolvidos no projeto. Para corrigir o problema e evitar danos futuros, deve-se implementar um regulador de tensão DC-DC. Este componente é essencial para converter a tensão de 11V da bateria para um nível seguro e adequado para o ESP32. O regulador DC-DC (FIGURA 50) não só protege o microcontrolador contra sobrecargas de tensão, mas também garante um fornecimento de energia estável e eficiente, essencial para o funcionamento confiável do drone.

FIGURA 50 – REGULADOR DE TENSÃO DC-DC

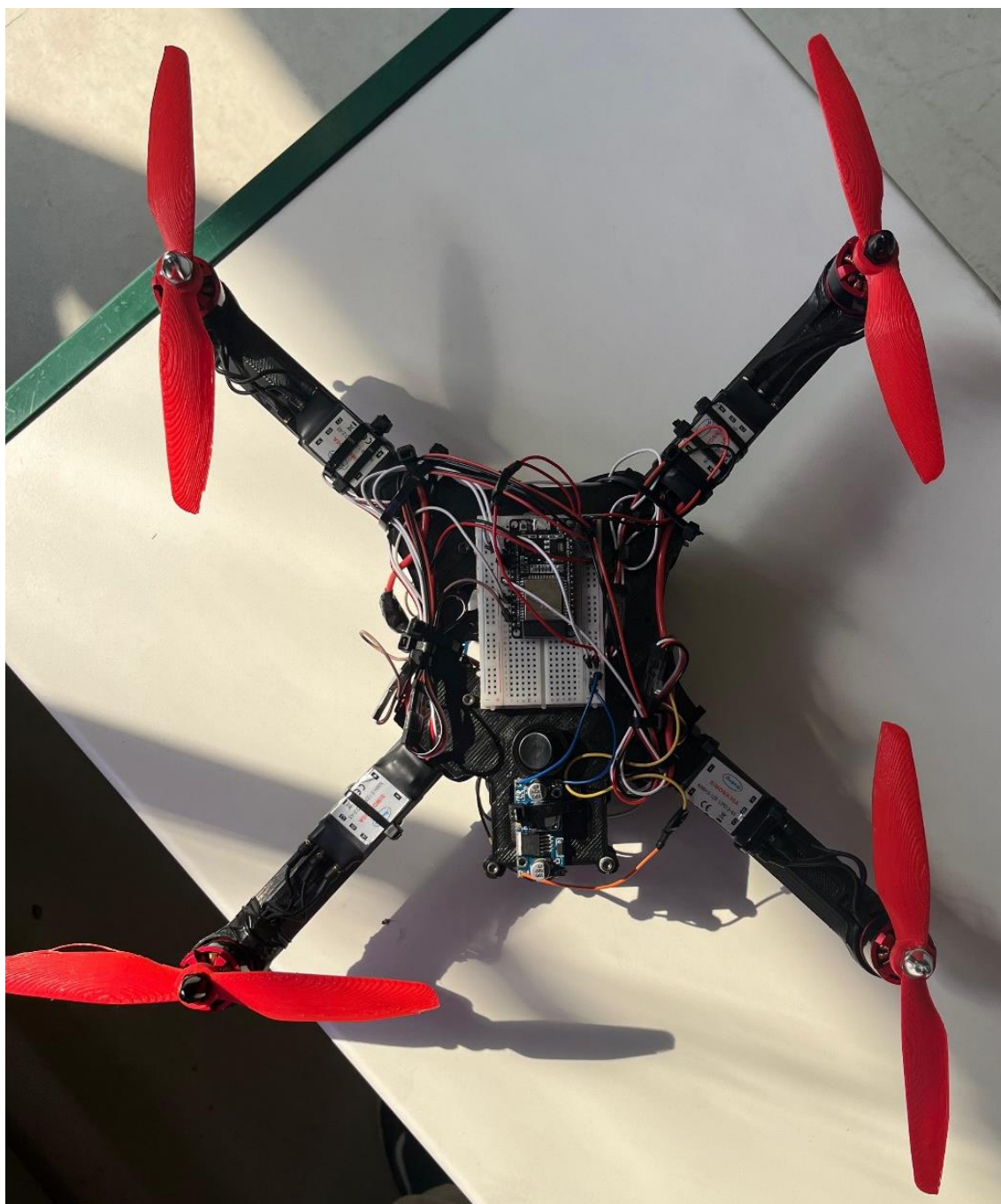


Fonte: Autoria Própria

5.3 Resultados

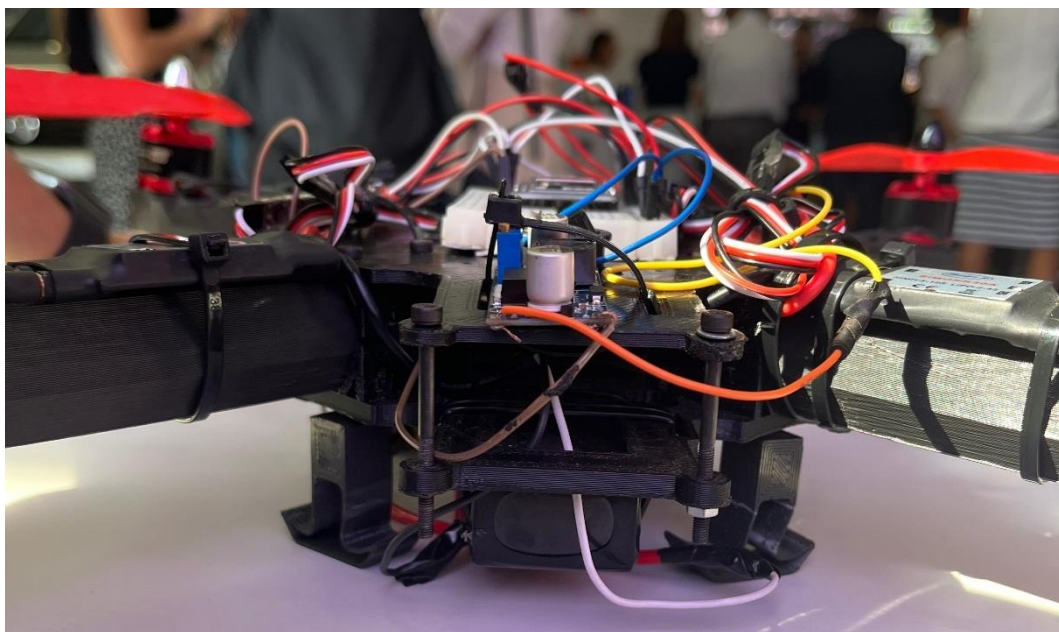
Após realizar diversos testes no programa e na parte física do projeto, incluindo a ligação dos componentes, testes de sentido de rotação do motor e testes de força para levantar o peso do drone, se tem o modelo final concluído. (FIGURA 51 A 57)

FIGURA 51 – DRONE COMPLETO



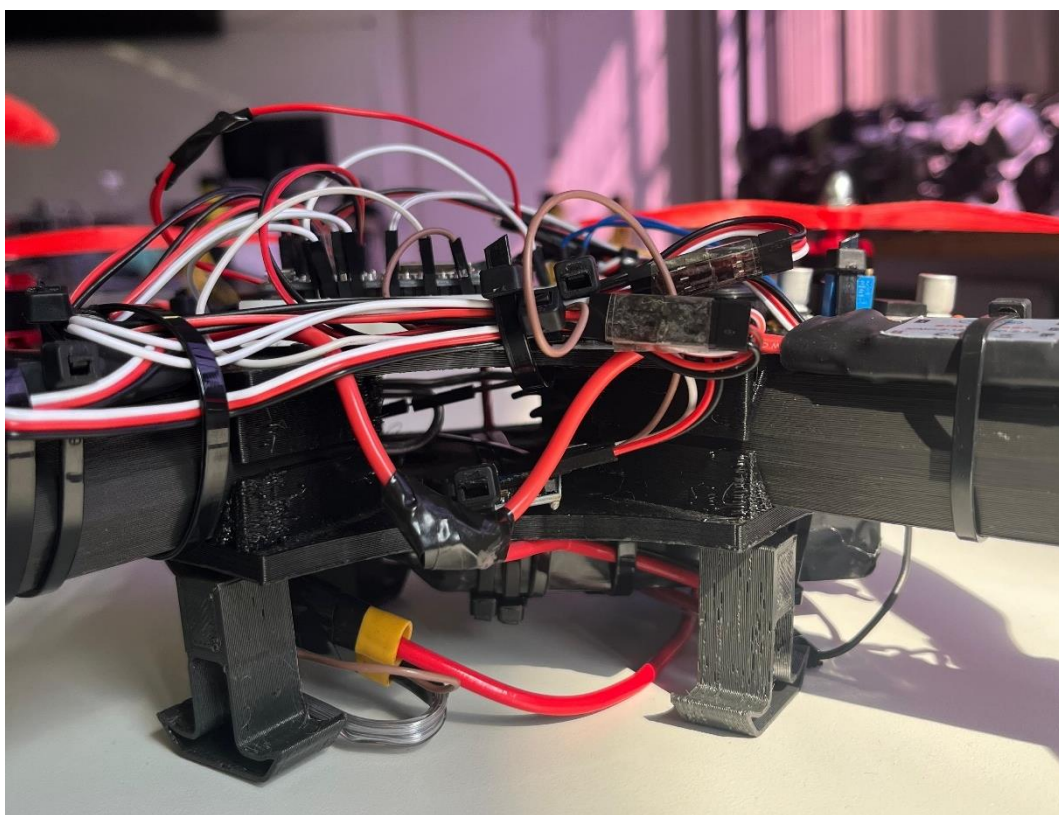
Fonte: Autoria Própria

FIGURA 52 – FRENTE DRONE COMPLETO



Fonte: Autoria Própria

FIGURA 53 – TRASEIRA DRONE COMPLETO



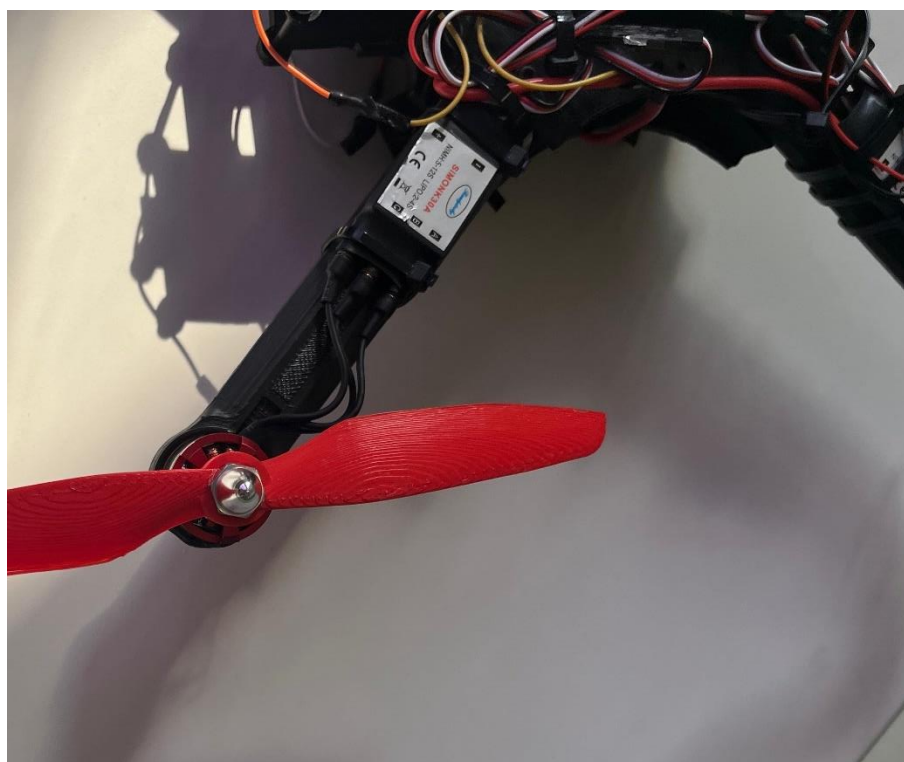
Fonte: Autoria Própria

FIGURA 54 – BRAÇO DRONE COMPLETO



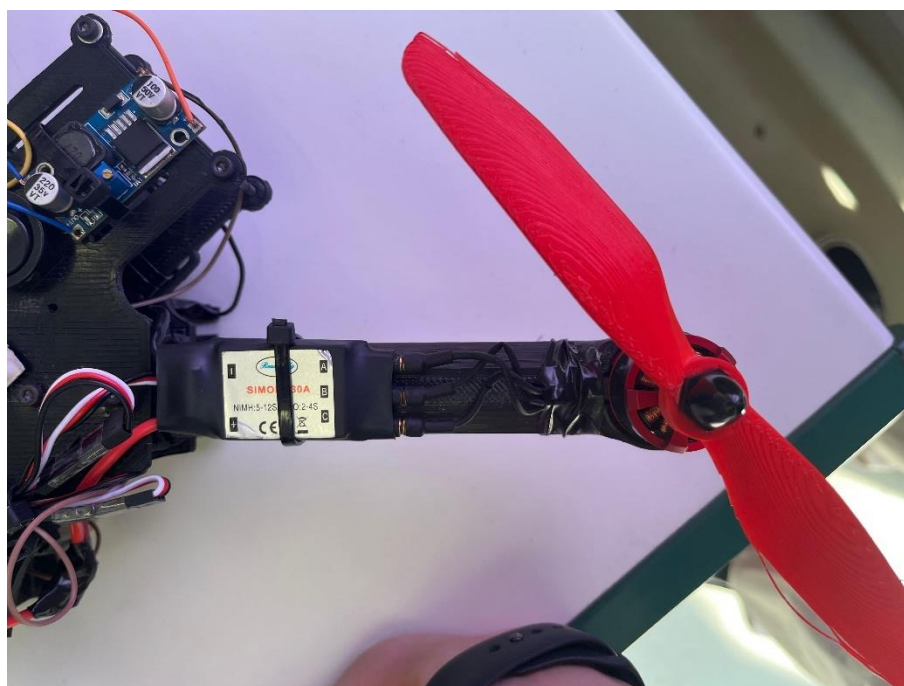
Fonte: Autoria Própria

FIGURA 55 – BRAÇO 2 DRONE COMPLETO



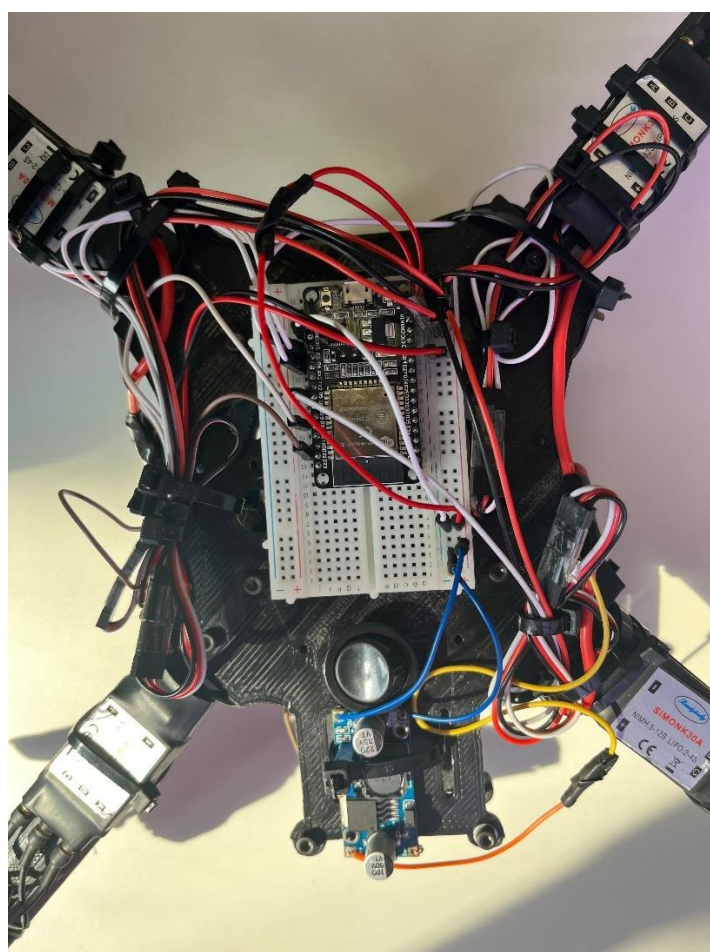
Fonte: Autoria Própria

FIGURA 56 – BRAÇO 3 DRONE COMPLETO



Fonte: Autoria Própria

FIGURA 57 – CENTRO DRONE COMPLETO

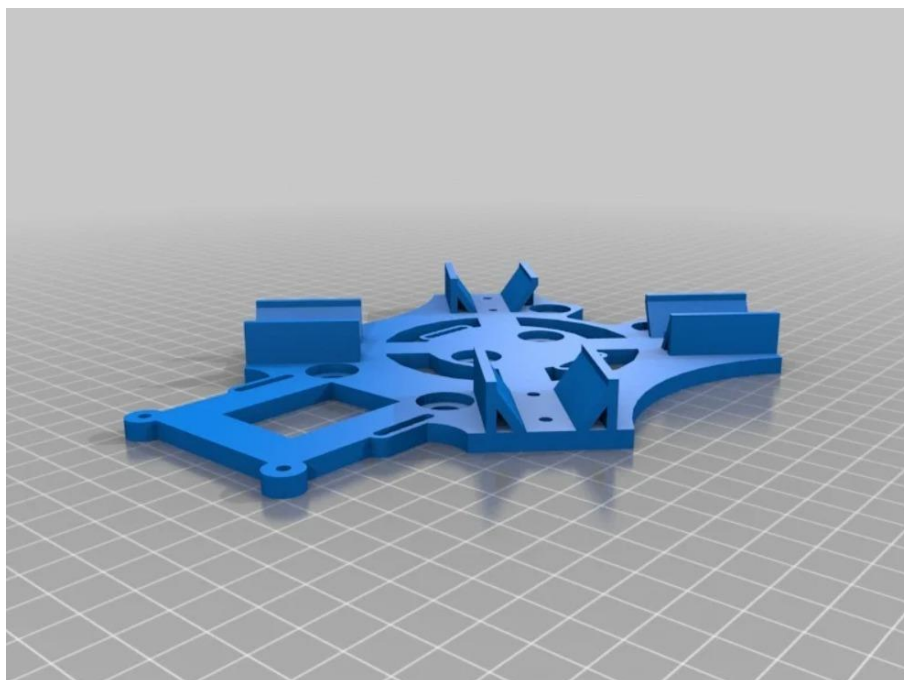


Fonte: Autoria Própria

5.3.1 Projeto 3D

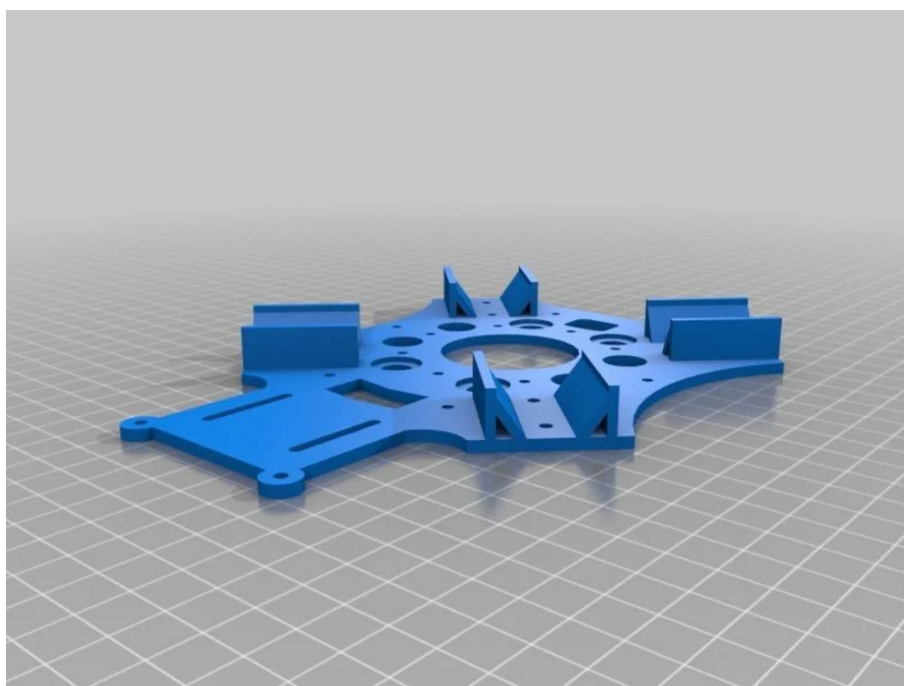
Abaixo alguns dos testes e imagens dos modelos e prints da impressão em 3D do centro e braços do drone nas FIGURAS 58 a 62:

FIGURA 58 – SUPERIOR DO CENTRO



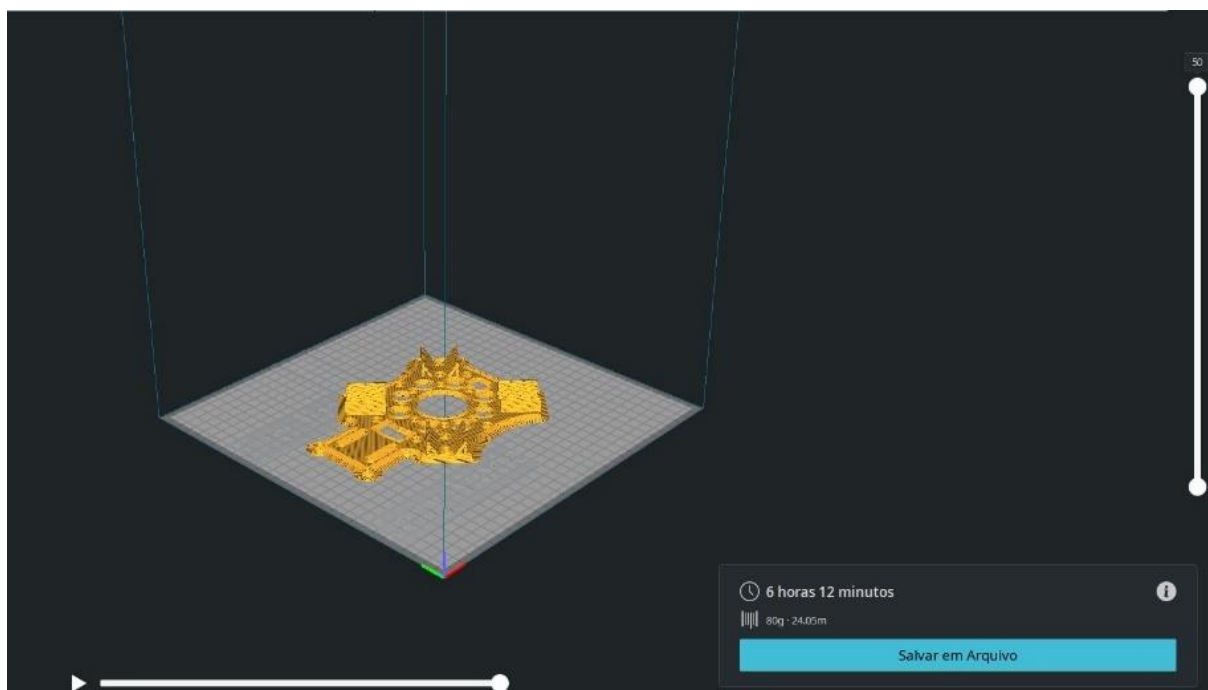
Fonte: Autoria Própria

FIGURA 59 – INFERIOR DO CENTRO



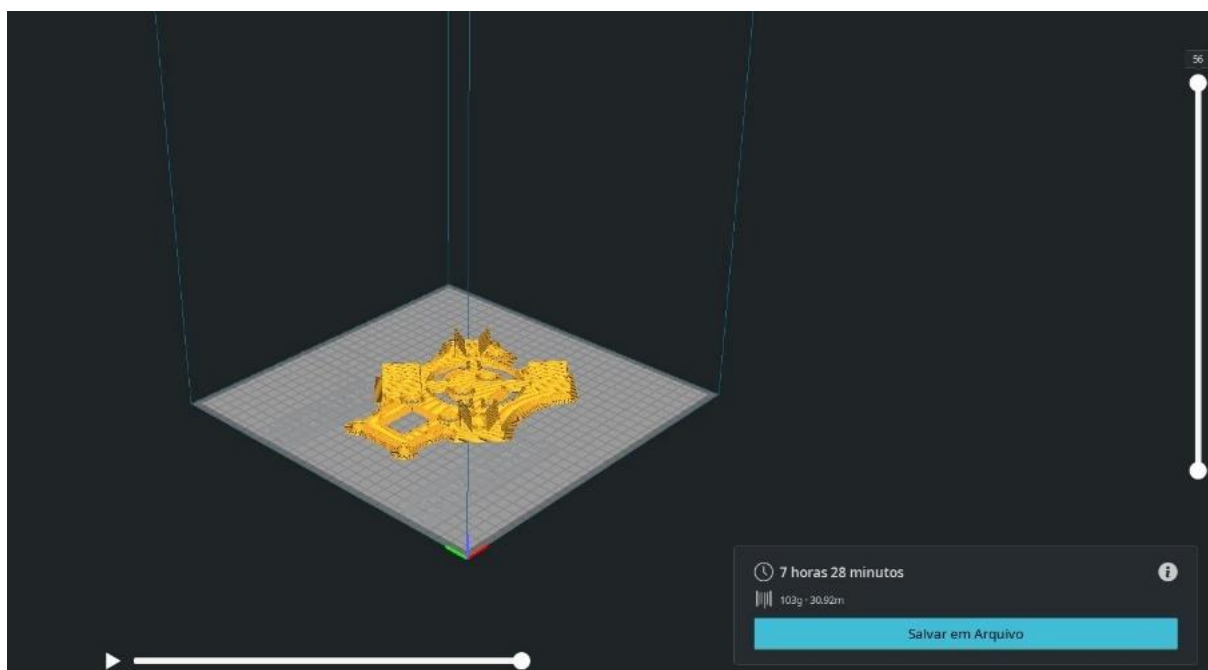
Fonte: Autoria Própria

FIGURA 60 – IMPRESSÃO DO CENTRO INFERIOR



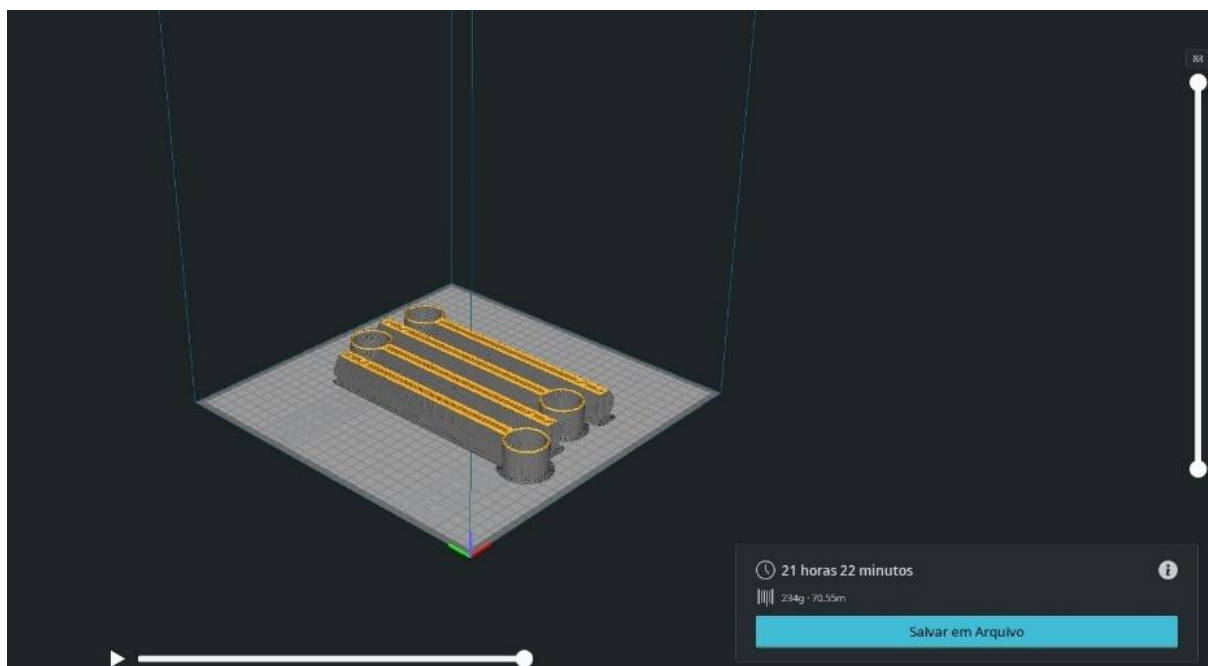
Fonte: Autoria Própria

FIGURA 61 – IMPRESSÃO DO CENTRO SUPERIOR



Fonte: Autoria Própria

FIGURA 62 – IMPRESSÃO DOS BRAÇOS



Fonte: Autoria Própria

6 CONCLUSÃO E PROPOSTAS FUTURAS

O desenvolvimento deste drone controlado por ESP-32 com comunicação via Modbus e supervisor AVEVA representou uma aplicação prática dos conhecimentos adquiridos ao longo do curso de Tecnologia em Mecatrônica Industrial. O projeto integrou diversas áreas do conhecimento, incluindo eletrônica, programação, modelagem 3D e sistemas supervisórios, resultando em um dispositivo funcional e eficiente.

A escolha dos componentes, como os motores de 920 kV, o giroscópio e os ESCs, aliada à programação em C++ e à comunicação via Modbus, permitiu a criação de um sistema robusto e estável. Os testes realizados com LEDs e motores servo foram fundamentais para garantir a confiabilidade e a segurança do drone antes de sua aplicação final.

A estrutura física do drone, desenvolvida com software de modelagem 3D e materializada por impressão STL, assegurou precisão e qualidade na construção. A comunicação com o supervisor AVEVA proporcionou um controle remoto eficaz, ampliando as possibilidades de aplicação industrial do drone.

Em suma, o projeto demonstrou a viabilidade e a importância de integrar tecnologias modernas e inovadoras, contribuindo para o avanço da automação e controle na Indústria 4.0. Este trabalho não apenas alcançou seus objetivos técnicos, mas também reforçou a importância da interdisciplinaridade e do trabalho em equipe na realização de projetos complexos e desafiadores.

7 REFERÊNCIAS

SEBRAE. Indústria 4.0. Disponível em: <[https://sebrae.com.br/Sebrae/Portal%20Sebrae/UFs/RJ/Anexos/Industria%204 0%20-%20WEB.PDF](https://sebrae.com.br/Sebrae/Portal%20Sebrae/UFs/RJ/Anexos/Industria%204%20-%20WEB.PDF)>. Acesso em: 15 de Out. 2023.

ODRONES. História dos Drones. Disponível em: <<https://odrones.com.br/historia-dos-dronesk/>>. Acesso em: 03 de Mar 2023.

ITARC. História dos Drones. Disponível em: <<https://itarc.org/historia-dos-drones/#:~:text=O%20surgimento%20do%20drone,durante%20a%20Segunda%20Guerra%20Mundial>>. Acesso em: 03 de Mar 2023.

FARMFOR. BQM-1BR: O drone fabricado pela Companhia Brasileira de Tratores (CBT). Disponível em: <<https://www.farmfor.com.br/posts/bqm-1br-o-drone-fabricado-pela-companhia-brasileira-de-tratores-cbt/>>. Acesso em: 03 Mar 2023.

CONCURSOS NO BRASIL. Afinal, quem criou o primeiro drone? Quando a tecnologia surgiu?, Disponível em: <<https://concursosnobrasil.com/afinal-quem-criou-o-primeiro-drone-quando-a-tecnologia-surgiu/>>. Acesso em: 03 de Mar 2023.

DEFESA CIVIL ES. Drone: Introdução. Disponível em: <https://defesacivil.es.gov.br/Media/DefesaCivil/Material%20Didático/01.Drone_introdução.pdf>. Acesso em: 05 de Jun 2023.

ANAC. Drones. Disponível em: <<https://www.anac.gov.br/assuntos/setor-regulado/drones>>. Acesso em: 05 Jun 2023.

EDROTA CULTURAL. Pilotos de drones começam a levar multas no Brasil. Disponível em: <<https://www.edrotacultural.com.br/pilotos-de-drones-comecam-a-levar-multas-no-brasil/>>. Acesso em: 05 Jun de 2023.

ARAÚJO, R. C. S. A. Estudo do uso de drones em aplicações civis. 2012. Dissertação (Mestrado) - Universidade Federal do Ceará. Disponível em: <https://repositorio.ufc.br/bitstream/riufc/4097/1/2012_dis_rcsa.pdf>. Acesso em: 11 de Out. 2023.

DIGITAL TRENDS. Drone Reviews. Disponível em: <<https://www.digitaltrends.com/drone-reviews/>>. Acesso em: 28 maio 2023.

DRONE DJ. Disponível em: <<https://dronedj.com/>>. Acesso em: 28 maio 2023.

DRONE RUSH. Disponível em: <<https://dronerush.com/>>. Acesso em: 28 maio 2023.

FAA. Unmanned Aircraft Systems (UAS). Disponível em: <<https://www.faa.gov/uas/>>. Acesso em: 16 de Fev 2023.

FORBES. Drones. Disponível em: <<https://www.forbes.com/drones/>>. Acesso em: 11 de Nov 2023.

OLHAR DIGITAL. Drones. Disponível em: <<https://olhardigital.com.br/tag/drones/>>.

TECHTUDO. Como funciona um drone: entenda a tecnologia por trás desses robôs. Disponível em: <<https://www.techtudo.com.br/noticias/2015/04/como-funciona-um-drone-entenda-tecnologia-por-tras-desses-robos.ghtml>>. Acesso em: 11 de Nov 2023.

FLYPRO. Anatomia de um drone. Disponível em: <[https://www.flypro.com.br/pagina/anatomia-de-um-drone.html#:~:text=Drones%20\(quadrícópteros\)%20tem%20%20motores,o posta%20e%20de%20mesma%20intensidade](https://www.flypro.com.br/pagina/anatomia-de-um-drone.html#:~:text=Drones%20(quadrícópteros)%20tem%20%20motores,o posta%20e%20de%20mesma%20intensidade)>. Acesso em: 20 de Jan 2024.

IEEE UFCG RAS. O que é um microcontrolador? Disponível em: <<https://edu.ieee.org/br-ufcgras/o-que-e-um-microcontrolador/>>. Acesso em: 22 de Jan 2024.

VICTOR VISION. O que é um microcontrolador? Disponível em: <<https://victorvision.com.br/blog/o-que-e-um-microcontrolador/>>. Acesso em: 22 de Jan 2024.

IFC LUZERNA. Apostila Microcontroladores. Disponível em: <<https://professor.luzerna.ifc.edu.br/ricardo-kerschbaumer/wp-content/uploads/sites/43/2018/02/Apostila-Microcontroladores.pdf>>. Acesso em: 03 Dez 2023.

UFC SOBRAL. Apostila Microprocessadores. Disponível em: <http://engeletrica.sobral.ufc.br/professores/marcelo/Micro/Apostila_Microprocessadores.pdf>. Acesso em: 03 Dez 2023.

BLOG ARDUINO OMEGA. Eletrônica Omega IoT Iniciantes ESP32. Disponível em: <<https://blog.arduinoomega.com/ebooks/Eletronica-Omega-Iot-Iniciantes-Esp32.pdf>>. Acesso em: 31 Mar 2023.

BLOG MASTER WALKER SHOP. Conhecendo o NodeMCU 32S ESP32. Disponível em: <<https://blogmasterwalkershop.com.br/embarcados/esp32/conhecendo-o-nodemcu-32s-esp32>>. Acesso em: 24 de Jul 2023.

TOTVS. SCADA: O que é e como funciona? Disponível em: <<https://www.totvs.com/blog/gestao-industrial/scada/>>. Acesso em: 15 Abr 2024.

AVEVA. Sobre a AVEVA. Disponível em: <<https://www.aveva.com/en/about/about-aveva/>>. Acesso em: 13 de Fev 2024.

EMBARCADOS. Protocolo Modbus. Disponível em:
<<https://embarcados.com.br/protocolo-modbus/>>. Acesso em: 7 Jun. 2024.

ALTUS. Conhecendo os protocolos Modbus TCP, Ethernet/IP e Profinet. Disponível em: <<https://www.altus.com.br/post/411/conhecendo-os-protocolos-modbus-tcp-2c-ethernet-2fip-e-profinet>>. Acesso em: 7 Jun. 2024.