Ana Karolina da Silva Barbosa

Sistema de Automação Residencial de Baixo Custo com Processamento Local

Brasil

Junho, 2025

Ana Karolina da Silva Barbosa

Sistema de Automação Residencial de Baixo Custo com Processamento Local

Trabalho de Conclusão de Curso apresentado ao curso de Engenharia de Computação da Universidade Federal de Mato Grosso do Sul, como requisito parcial para a obtenção do grau de Bacharel em Engenharia de Computação.

Universidade Federal de Mato Grosso do Sul

Faculdade de Computação

Programa de Graduação

Orientador: Victor Leonardo Yoshimura

Brasil

Junho, 2025

1 Introdução

A automação residencial é uma tecnologia que permite a integração de vários circuitos distintos, como iluminação, temperatura, televisores, alarmes, atuadores e outros dispositivos, em um sistema central, permitindo ao usuário controlar uma ou mais casas remotamente, sem a necessidade de se dirigir ao dispositivo específico ou estar fisicamente presente. Além disso, a automação permite que, por meio da Internet das Coisas (IoT), uma casa inteligente seja criada. Isso significa que o sistema atua e responde a diversas situações programadas sem a necessidade de um comando direto do usuário, executando ações conforme condições previamente configuradas.

A IoT é a conexão de diversos dispositivos e objetos do cotidiano à Internet, permitindo que eles se comuniquem e troquem informações de forma autônoma. Por meio dessa integração, é possível interligar sensores, câmeras, eletrodomésticos e outros dispositivos, convertendo ambientes convencionais em sistemas automatizados e responsivos. Assim, ao fazer o monitoramento em tempo real e permitir respostas rápidas a eventos inesperados, a Internet das Coisas viabiliza a criação de casas inteligentes, onde os dispositivos trabalham de forma integrada para otimizar funções, antecipar necessidades e oferecer maior eficiência, segurança e conforto aos usuários.

Diante dos inúmeros benefícios proporcionados pela automação residencial, a demanda por esses serviços tem apresentado um crescimento expressivo nos últimos anos. Projeções indicam que o mercado global de automação residencial deverá registrar uma taxa de crescimento anual composta (CAGR) de 27,1% no período de 2023 a 2030 (Grand View Research, 2023). Esse crescimento pode ser relacionado ao avanço das tecnologias inteligentes e à busca por maior comodidade e eficiência nos processos domésticos.

Contudo, apesar de sua crescente popularidade, a automação residencial ainda é considerada um recurso de nicho. Frequentemente, está associada a altos custos, o que afasta parte da população da possibilidade de usufruir das suas funcionalidades. Ademais, muitas pessoas associam a automação à substituição de componentes como luminárias, tomadas, interruptores e dispositivos especificamente comercializados como compatíveis com sistemas de automação.

Outrossim, é comum que a automação residencial atualmente se limite a dispositivos integrados a assistentes virtuais como Alexa e Google Home. Essa dependência representa uma limitação significativa para usuários que valorizam a privacidade, a segurança e a adaptabilidade. Tais sistemas realizam o processamento de dados do usuário em servidores na nuvem, o que, além de sujeitar essas informações às políticas de privacidade das empresas, amplia a superfície de ataque e pode expor os dados a riscos como vazamentos e acessos não autorizados. Isso ocorre porque os dados trafegam pela Internet e ficam armazenados fora do controle direto do usuário. De forma complementar, esses dispositivos geralmente oferecem comandos padronizados, voltados

para atender a um público amplo, o que pode não contemplar particularidades individuais que se beneficiariam de soluções tecnológicas personalizadas.

Devido às barreiras de custo e às soluções comerciais limitadas, muitos usuários deixam de implementar automação residencial por desconhecerem alternativas mais flexíveis. Este Trabalho de Conclusão de Curso propõe uma arquitetura baseada em um microcontrolador e um computador de placa única acoplados a módulos de relés, cujo código-fonte é totalmente programável e adaptável às necessidades específicas de cada ambiente.

Além de elevar a segurança, a privacidade e a adaptabilidade, uma vez que todo o processamento e armazenamento de dados ocorrem localmente, sem depender de servidores externos, essa abordagem também contribui para a sustentabilidade ao reutilizar a maioria dos componentes já instalados (luminárias, interruptores, tomadas), evitando seu descarte prematuro. Ainda, como o sistema não requer conexão à Internet para operar, ele mantém-se plenamente funcional em locais com conectividade instável ou ausente, embora possa ser adaptado para atender a diferentes requisitos do usuário, inclusive integrando funcionalidades baseadas em nuvem caso desejado.

2 Especificações do Projeto

O presente projeto de automação residencial foi desenvolvido com foco em baixo custo, segurança, adaptabilidade e sustentabilidade. Ele conta com uma maquete demonstrativa, projetada e impressa 3D, que simula o funcionamento do sistema e um esquema elétrico real aplicável para 127V e 220V, permitindo a visualização do comportamento do sistema e disponibilizando o material necessário para que seja aplicado.

A Tabela 1 apresenta uma estimativa comparativa dos custos envolvidos entre sistemas tradicionais de automação residencial, tendo como referência a Alexa e o sistema proposto, desenvolvido com componentes de baixo custo.

Estimativa de Preços				
Objetivo	**Assistentes Tradicionais**	**Preço Médio**	**Projeto de Baixo Custo**	**Preço Médio**
Comando de Voz (Considerando 5 cômodos)	5 Alexas	R$ 2.000,00	Um Raspberry Pi 4 + 5 microfones USB + Hub USB	R$ 1.050,00
Acionamento de Luzes (Considerando 8 circuitos)	2 Módulos Inteligente com 4 Vias Wifi	R$ 160,00	2 Módulos Relé 3V com 4 Vias	R$ 110,00
Sensor de Alarme (Considerando 10 setores)	Alarme Sensor Wifi Inteligente	R$ 400,00	Sensor Abertura Magnético Com Fio	R$ 130,00
Sirene Alarme	Sirene Alarme Inteligente Tuya	R$ 130,00	Sirene Piezoelétrica + Transistor	R$ 25,00
APP Central	Grátis se houver tecnologia Tuya	R$ 0,00	Grátis se o APP for programado e conectado ao Wi-Fi	R$ 0,00

Tabela 1 – Resultado da estimativa de custo dos sistemas convencional e proposto.

Esta análise de custos reforça a tese central do projeto, demonstrando que é possível implementar um sistema de automação residencial funcional, seguro e adaptável por um valor significativamente inferior ao dos sistemas comerciais tradicionais. A Tabela 2 evidencia que o

projeto proposto equivale a aproximadamente metade do custo de um sistema que utiliza Alexa, sem comprometer as funcionalidades essenciais como controle de iluminação, acionamento de portões, monitoramento de portas e janelas para segurança e operação do sistema por comandos de voz. Essa redução de custo é alcançada por meio da seleção de componentes que combinam eficiência e baixo custo.

Resultado			
Estimativa Total do Sistema Convencional	R$ 2.690,00	**Estimativa Total do Sistema Proposto**	R$ 1.315,00

Tabela 2 – Comparativo de preços entre um sistema convencional que utiliza Alexa e o sistema proposto

O microcontrolador adotado é o ESP32, amplamente empregado em projetos de automação residencial devido a sua versatilidade, baixo custo, conectividade Wi-Fi e Bluetooth integradas, além de apresentar capacidade de processamento suficiente para o controle de sensores, atuadores e execução de lógica de controle. Trata-se de uma solução apropriada para sistemas embarcados que demandam eficiência energética e respostas em tempo real. Contudo, para a tarefa de processamento de reconhecimento de fala por meio da biblioteca VOSK, faz-se necessário um dispositivo com maior capacidade computacional. Nesse contexto, foi selecionado o Raspberry Pi 4, que oferece ótimo desempenho no modelo com 4GB de RAM, suporte a sistemas operacionais completos, como o Linux, e compatibilidade com APIs de processamento de voz. Dessa forma, ESP32 e Raspberry Pi operam de maneira complementar: enquanto o Raspberry Pi é responsável pela interpretação dos comandos de voz, o ESP32 realiza a execução das ações físicas correspondentes no ambiente automatizado.

Além dos elementos já mencionados, os demais componentes eletrônicos, materiais e sistemas necessários para a construção do projeto serão detalhados nas seções seguintes.

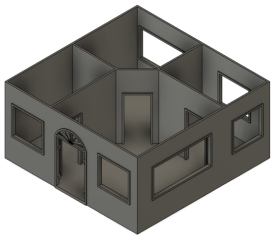
2.1 Maquete de Simulação do Sistema

A maquete utilizada para a simulação do sistema foi inteiramente projetada no software Autodesk Fusion 360, adotando-se uma escala de 1:20, em que 1 metro no projeto representa 20 metros na realidade. Com o objetivo de tornar a visualização mais clara e facilitar a manipulação física durante os testes e demonstrações, optou-se por utilizar dimensões ampliadas para portas e janelas. Todos os projetos foram impressos na impressora Creality K1 Max, utilizando filamento PLA da marca GTmax na cor cinza.

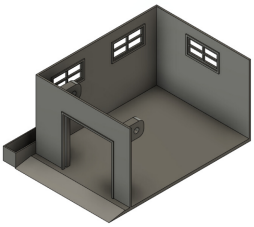
2.1.1 Projeto 3D para Impressão

A casa principal conta com quatro cômodos, sendo eles: Sala de Estar, Sala de TV, Quarto Casal e Quarto Dois. Ambientes como cozinha e banheiro não foram incluídos, considerando que,

no contexto de casas reais, apresentam menor potencial de aplicação para automações residenciais, por não possuírem dispositivos comumente automatizados, como iluminação de brilho ajustável, ar-condicionado, televisores ou sistemas de som. Além da casa principal, há uma garagem acoplada ao lado esquerdo da estrutura, a qual possui a base para o portão interno (Figura 7). Ambos os projetos podem ser visualizados na Figura 1.



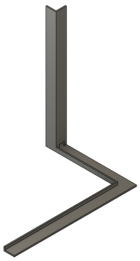
(a) Casa com 4 cômodos.



(b) Garagem lateral.

Figura 1 – Conjunto principal de cômodos para demonstração.

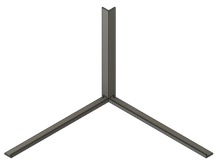
Para a estrutura de iluminação interna, foram projetadas sancas para cada um dos cômodos. Elas servem como suporte para a fixação da fita de LED, além de permitirem uma iluminação mais indireta e ocultarem os fios de alimentação e do alarme. Cada uma delas pode ser visualizada na Figura 2.



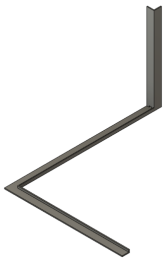
(a) Sanca - Quarto 2.

(b) Sanca - Quarto casal.

(c) Sanca - Sala TV.



(d) Sanca - Sala de Estar



(e) Sanca - Garagem

Figura 2 – Sancas para os LEDs de iluminação dos cômodos.

Para que as folhas das janelas fossem móveis na estrutura da maquete, a solução adotada foi o desenvolvimento de trilhos que permitissem o deslizamento das peças. As partes foram impressas individualmente e, posteriormente, montadas na estrutura, de modo que os trilhos fossem colados no batente com as janelas já alinhadas entre eles, como na Figura 5. Foram projetados três tamanhos diferentes de trilhos, de acordo com os três modelos de janelas utilizados no projeto, conforme ilustrado na Figura 3.

(a) Trilho de 59mm. (b) Trilho de 62mm. (c) Trilho de 120mm.

Figura 3 – Trilhos utilizados para que janelas possam se movimentar.

Foram projetados dois modelos distintos de folhas para as esquadrias das janelas: um com puxador, utilizado em todas as partes móveis, e outro sem puxador, aplicado nas partes fixas. Entre as estruturas das folhas, foi deixado um espaço destinado ao encaixe de uma lâmina de polipropileno de 18 micras, que simula o vidro.

(a) Folha fixa. (b) Folha móvel.

Figura 4 – Folhas utilizadas nas esquadrias das janelas.

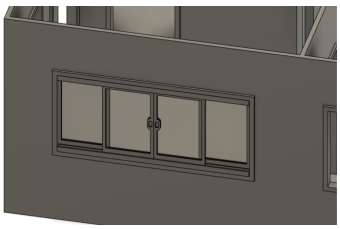
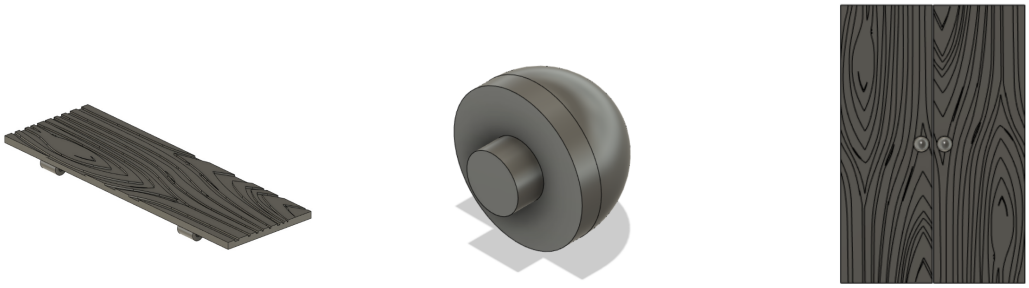


Figura 5 – Janela de 120mm montada com trilhos, parte móvel e parte fixa.

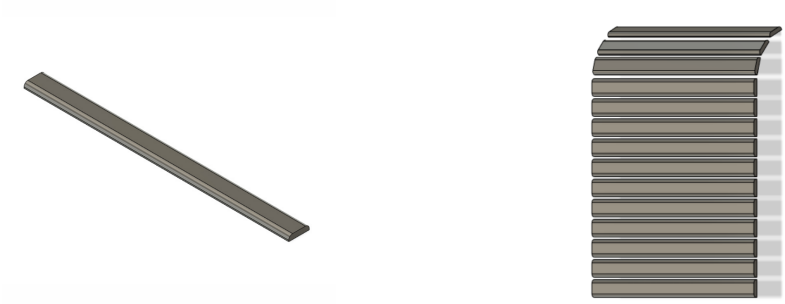
Para as portas, o eixo de movimentação foi integrado diretamente ao batente da porta na estrutura da casa principal. As portas e as maçanetas foram projetadas e impressas separadamente, sendo posteriormente encaixadas e coladas na estrutura principal. A porta, a maçaneta e a representação da montagem final podem ser observadas na Figura 6.



(a) Porta frontal. (b) Maçaneta da porta. (c) Ilustração da porta montada.

Figura 6 – Portas frontais da casa.

O Portão Interno foi desenvolvido com base no modelo de portões de enrolar, composto por lâminas articuladas com acionamento por enrolamento. Para sua construção, foi projetada uma única lâmina, posteriormente impressa em 3D quatorze vezes, como ilustrado na Figura 7. As peças foram coladas sobre uma tela mosquiteira de Nylon, mantendo um espaçamento de 1 mm entre elas. Essa configuração permite a flexibilidade necessária para o enrolamento da estrutura, ao mesmo tempo em que proporciona rigidez suficiente para resistir a forças horizontais aplicadas ao portão.

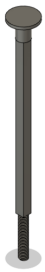


(a) Chapas do portão. (b) Ilustração do portão.

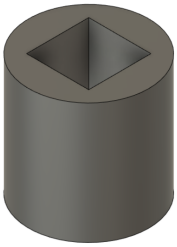
Figura 7 – Estrutura do portão interno.

Para a recepção do movimento do motor e controle do Portão Interno, foram projetadas seis peças, conforme ilustrado na Figura 8. A peça (e) foi impressa duas vezes e atua como guia

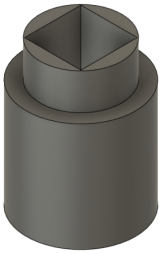
para o portão, posicionando-se nas extremidades esquerda e direita do eixo principal. A peça (a) corresponde ao eixo central do sistema, cuja geometria quadrada permite um encaixe firme, mesmo sob rotações nos sentidos horário e anti-horário. As peças (b), (c) e (d) têm a função de manter os guias (e) igualmente espaçados ao longo do eixo. A peça (f) é responsável por receber o movimento do motor por meio de uma correia dentada GT2 de 6mm (passo de 2mm), transmitindo esse movimento ao eixo para a realização dos movimentos de abertura e fechamento do portão.



(a) Eixo do rotação do portão.

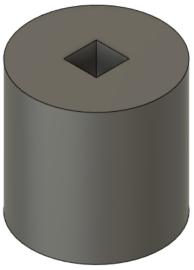


(b) Primeiro encaixe do eixo.

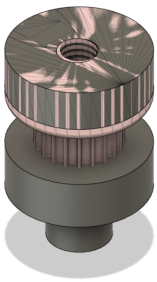


(c) Último encaixe do eixo.

(d) Encaixe central do eixo.



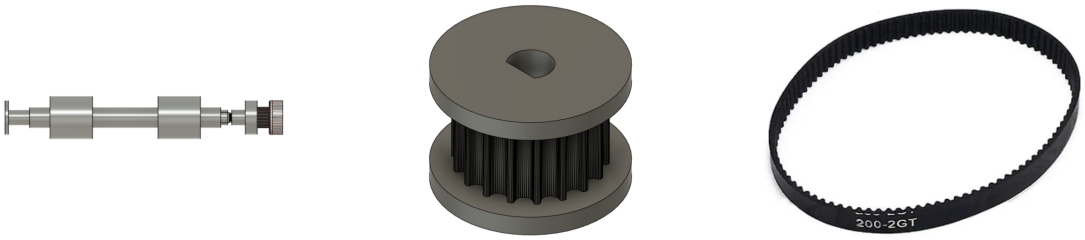
(e) Guia do portão.



(f) Transmissor do movimento.

Figura 8 – Peças de transmissão que incorporam o eixo de controle do Portão Interno.

A representação das peças montadas pode ser visualizada na Figura 9, especificamente no item (a). Na parte inferior da casa, acoplada ao eixo do motor, encontra-se a peça (b) da mesma figura, responsável por transmitir o movimento do motor para o eixo (a) por meio de uma correia dentada (c).



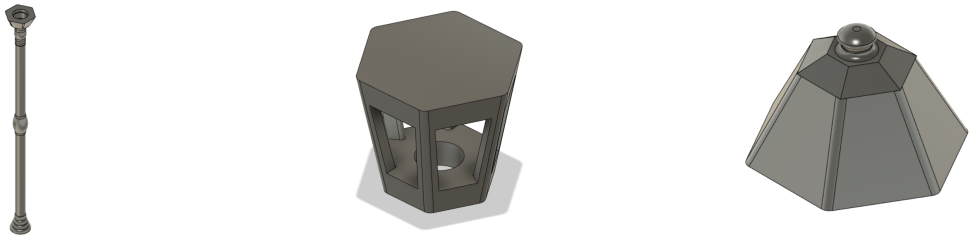
(a) Receptor - eixo do Portão Interno montado.

(b) Transmissor - adaptador para o motor.

(c) Correia dentada de passo 2mm.

Figura 9 – Peças responsáveis por abrir e fechar o Portão Interno

Para a iluminação externa, foram projetadas três peças que compõem um poste de jardim com estilo colonial. A haste possui estrutura vazada, permitindo a passagem dos fios de alimentação do LED. A luminária é encaixada ao LED, e uma tampa permite seguir a estética adotada. As três peças mencionadas, juntamente com uma ilustração do conjunto montado, estão apresentadas nas Figuras 10 e 11.



(a) Haste do poste.

(b) Luminária do poste.

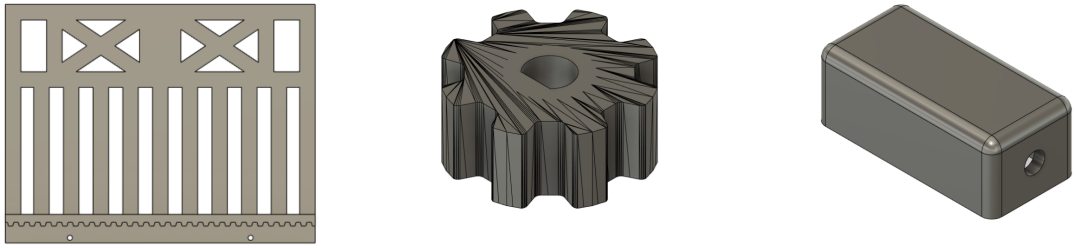
(c) Tampa da luminária.

Figura 10 – Peças necessárias para a montagem do Poste Externo.

Figura 11 – Poste montado.

O portão externo foi desenvolvido no modelo deslizante. Para sua concepção, foram utilizados como referência vídeos de profissionais da área de serralheria (CANAL OFICINA GARAGEM, 2025) (MOISES GRACIANO DA SILVA, 2025), que demonstram os mecanismos reais empregados nesse tipo de estrutura. O protótipo foi projetado com base na configuração real dos portões deslizantes, sendo adaptado apenas em aspectos específicos para sua aplicação em maquete. Como a simulação é realizada com materiais plásticos em vez de metálicos, não foi necessário manter uma estrutura tão robusta.

O portão possui um encaixe destinado a receber o movimento transmitido pela engrenagem acoplada ao motor. Além disso, foi projetada uma capa protetora para o motor, uma polia sobre a qual o portão se desloca e um guia que direciona seu movimento. As peças mencionadas estão ilustradas na Figura 12.

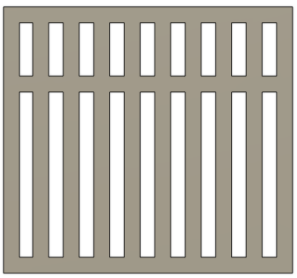


(a) Portão Externo. (b) Engrenagem para o motor. (c) Capa do motor.

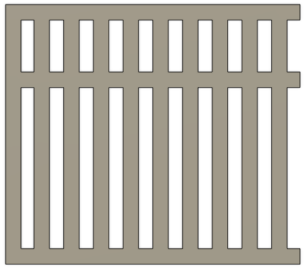
(d) Polia para mover o portão. (e) Guia para a polia.

Figura 12 – Conjunto de peças necessárias para a movimentação do Portão Externo.

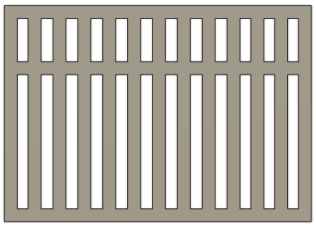
As grades que delimitam o perímetro frontal e a lateral direita da maquete foram projetadas em três tamanhos distintos: 109 milímetros, 142 milímetros e 239 milímetros. Foram desenvolvidos quatro modelos, com variações entre versões com encaixe e sem encaixe. As diferentes configurações estão representadas na Figura 13.



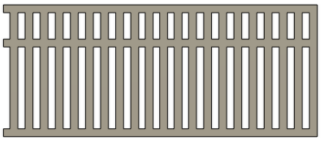
(a) Grade de 109mm.



(b) Grade de 109mm com encaixe.



(c) Grade de 142mm.



(d) Grade de 239mm com encaixe.

Figura 13 – Grades que cercam a maquete.

Além dos dois portões mencionados anteriormente, o projeto também possui um portão destinado à passagem de pedestres. Sua estrutura é simples e pode ser visualizada na Figura 14.

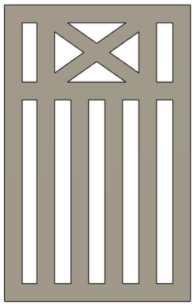


Figura 14 – Portão de Pedestres.

Com o objetivo de disponibilizar um meio de controle físico adicional ao comando por voz, bem como de exibir informações sobre o status do alarme, foi projetada uma central de

controle. Essa central possui espaço destinado à instalação de um display LCD 16×2, nove botões momentâneos e um botão com retenção. A estrutura da central está representada na Figura 15.

Figura 15 – Central de Controle.

2.1.2 Materiais Complementares Utilizados na Montagem da Maquete

Como base estrutural para a montagem da maquete, foi utilizada uma chapa de MDF cru com dimensões de 90×60 cm e espessura de 6 mm. A escolha pelo MDF deve-se à sua resistência, que permite sustentar a estrutura e facilita o transporte. Ainda, ao longo do perímetro da base, foram fixadas placas de isopor com aproximadamente 6,5 cm de altura, com o objetivo de elevar a estrutura e disponibilizar espaço na parte inferior para a montagem do circuito.

2.2 Componentes Eletrônicos e Acessórios com suas Representações no Circuito

Esta seção apresenta os componentes eletrônicos e os materiais auxiliares utilizados na construção do circuito elétrico do sistema. Para fins de simplificação, alguns componentes serão representados por blocos contendo apenas seus terminais principais, sem detalhamento do circuito interno. Essa abordagem tem como objetivo facilitar a leitura e a compreensão do diagrama, mantendo o foco no funcionamento geral do sistema. As representações adotadas encontram-se descritas a seguir.

2.2.1 LED

Um LED é um componente eletrônico que emite luz quando percorrido por corrente elétrica em um único sentido. Para a iluminação externa, acoplado ao poste colonial, foram utilizados LEDs amarelos simples de 5 milímetros, escolhidos por seu tamanho compatível com o

estilo do poste. Além disso, um LED azul foi incorporado à placa de alimentação com a finalidade de indicar a presença de tensão no circuito.

Figura 16 – LEDs amarelos

Fonte: (RoboCore, 2025b)

No diagrama esquemático, o LED é representado por seu próprio símbolo, contendo seus dois polos, conforme ilustrado na Figura 17.

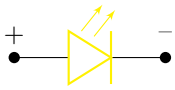


Figura 17 – Representação dos terminais de um LED

2.2.2 Fita de LED Branco

A fita de LED (Figura 18) é uma faixa flexível composta por LEDs conectados eletricamente e que compartilham o mesmo VIN e GND. Para a iluminação dos cômodos, optou-se pela utilização de uma fita de LED branca, devido à sua elevada capacidade de iluminação. Além disso, esse tipo de componente facilita a instalação por ser fino e flexível. O modelo escolhido opera em 12V e apresenta um consumo de 12W por metro. Outra vantagem é que a fita pode ser cortada em diversos pontos sem comprometer o funcionamento do restante, permitindo sua adaptação ao tamanho específico de cada cômodo.

Figura 18 – Fita de LED.

Fonte: (Lighting Philips, 2025)

No diagrama esquemático, a Fita de LED Branco é representado por um bloco retangular contendo dois pinos, correspondentes às conexões positiva e negativa de alimentação, como na Figura 19.

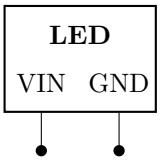


Figura 19 – Representação simplificada da fita de LED.

2.2.3 Fita de LED Endereçável RGB

Uma fita de LED RGB endereçável é composta por LEDs tricolores (vermelho, verde e azul) que possuem chips controladores integrados, permitindo que cada LED seja controlado individualmente em cor e brilho. Isso possibilita a criação de efeitos dinâmicos, como animações e transições de cores, utilizando apenas um fio de dados além da alimentação. O controle das três cores ocorre por meio de um circuito integrado presente na fita LED, que recebe os dados enviados pelo microcontrolador e gera internamente três sinais PWM, um para cada canal (vermelho, verde e azul), modulando assim a intensidade de cada cor. Para a visualização da resposta do sistema de reconhecimento de voz, foi utilizada uma fita de LED RGB endereçável, que possibilita uma análise visual do sistema, tornando possível identificar se ele está ativo e respondendo. A fita escolhida opera com alimentação de 12V e apresenta consumo de 14W por metro.

Figura 20 – Fita de LED RGB endereçável.

Fonte: (RoboCore, 2025a)

No diagrama esquemático, a fita é representada por um bloco retangular contendo três pinos, correspondentes às conexões positiva e negativa de alimentação, além do pino de entrada DATA que controla os LEDS, como na Figura 21.

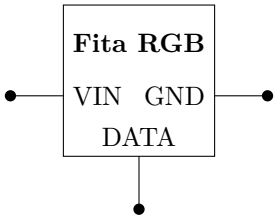


Figura 21 – Representação dos pinos de conexão de uma fita de LED RGB endereçável

2.2.4 Relé

O relé é um componente eletromecânico que funciona como um interruptor acionado por eletricidade. Ele permite ligar ou desligar um circuito de maior tensão ou corrente usando um sinal elétrico de menor intensidade, como o de uma porta lógica de um microcontrolador. Dentro dele, há uma bobina que, ao receber energia, cria um campo magnético que movimenta uma chave interna. Com isso, o relé pode abrir ou fechar conexões elétricas sem que seja necessário contato direto, como representado na Figura 22.

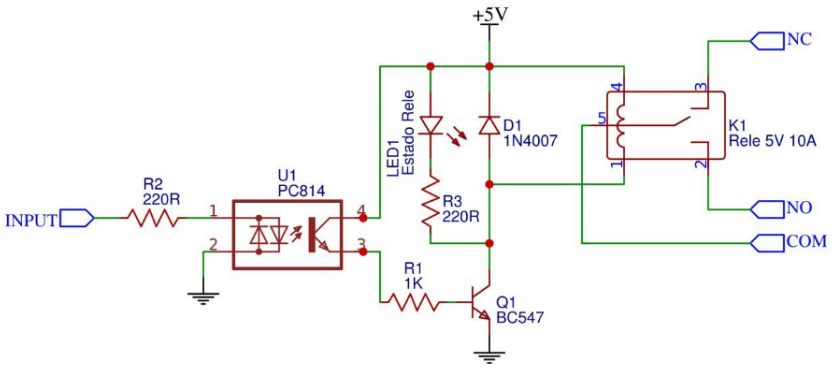


Figura 22 – Diagrama de funcionamento interno de um módulo relé com entrada de 5V

Fonte: (TEC DICAS, 2025)

O módulo relé possui 6 pinos:

- **VIN**: Utilizado para fornecer alimentação elétrica ao módulo relé.
- **GND**: Pino de referência, necessário para fechar o circuito de alimentação.
- **IN**: Responsável por receber o sinal de controle do microcontrolador, acionando ou não o relé.
- **COM (Common)**: Terminal comum da chave de comutação, por onde entra a corrente que será direcionada à carga.

- **NC (Normally Closed)**: Terminal normalmente fechado, conectado ao COM quando o relé está desligado, permitindo a passagem de corrente sem acionamento.
- **NO (Normally Open)**: Terminal normalmente aberto, que só se conecta ao COM quando o relé é acionado, permitindo o controle do fluxo de corrente conforme o sinal recebido.

Para o projeto, foi utilizado um módulo relé de quatro canais, compatível com sinais de 3,3V, permitindo seu controle direto pelo ESP32. Cada canal suporta correntes de até 10A, podendo operar com cargas em 127V ou 220V. O módulo possui circuitos de proteção integrados, incluindo diodos de *flyback*, que evitam danos causados por correntes de retorno geradas por cargas indutivas.

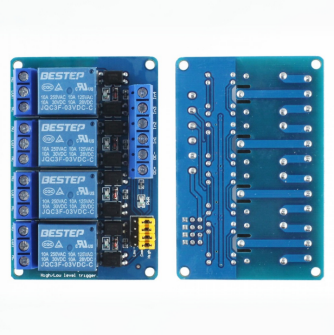


Figura 23 – Modulo relé 3V3 com quatro canais.

Fonte: (ELETROGATE, 2025b)

No diagrama esquemático, o relé é representado por um bloco retangular contendo seus seis pinos, como na Figura 24.

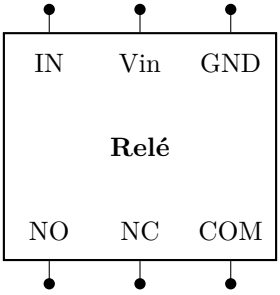


Figura 24 – Diagrama simplificado do módulo relé com indicação dos principais terminais.

2.2.5 Regulador de Tensão L7805CV

O regulador de tensão L7805CV, ilustrado na Figura 25, é um componente eletrônico utilizado para fornecer uma saída fixa de 5V a partir de uma tensão de entrada superior e variável,

garantindo estabilidade na alimentação de circuitos eletrônicos. Ele possui três pinos: VIN é o terminal de entrada de tensão, GND corresponde à referência e VOUT é a saída que fornece a tensão regulada de 5V. Neste projeto, o componente em questão foi utilizado para converter a alimentação principal de 12V para 5V, destinada a outros componentes do sistema.

Figura 25 – Regulador de tensão L7805CV.

Fonte: (STMicroelectronics, 2025)

No diagrama esquemático, o L7805CV é representado por um bloco retangular contendo seus três pinos, como na Figura 26.

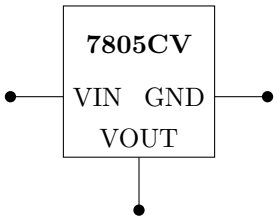


Figura 26 – Diagrama simplificado do regulador de tensão L7805CV com indicação dos pinos principais.

2.2.6 Driver de Motor TB6612FNG

O TB6612FNG (Figura 27) é um *driver* de motor projetado para controlar dois motores de corrente contínua (DC) de forma independente, permitindo a definição da direção e velocidade de rotação por meio de sinais digitais. Ele opera com tensões lógicas entre 2,7V e 5,5V, suporta até 15V na alimentação dos motores, correntes contínuas de até 1,2A por canal e picos de até 3,2A (Toshiba, 2025). No presente projeto, o componente foi utilizado para controlar os motores responsáveis pela abertura e fechamento dos dois portões para carros: um referente ao portão interno e outro ao portão externo da residência.

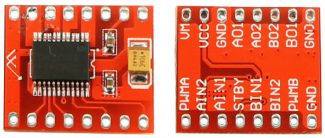


Figura 27 – Driver de motor TB6612FNG.

Fonte: (USINAINFO, 2025a)

O TB6612FNG dispõe de 14 pinos, sendo eles:

- **AIN1 e AIN2**: Entradas digitais que definem a direção de rotação do Motor A.
- **BIN1 e BIN2**: Entradas digitais que definem a direção de rotação do Motor B.
- **PWMA e PWMB**: Entradas de sinal PWM responsáveis pelo controle da velocidade dos motores A e B, respectivamente.
- **STBY**: Entrada de controle que deve ser mantida em nível lógico alto para habilitar o funcionamento do módulo.
- **A01 e A02**: Saídas conectadas aos terminais do Motor A.
- **B01 e B02**: Saídas conectadas aos terminais do Motor B.
- **VCC**: Entrada de alimentação lógica, entre 2,7 V e 5,5 V.
- **VM**: Entrada de alimentação dos motores, suportando até 15 V.
- **GND**: Pinos de referência do circuito.

O controle de cada motor é realizado por meio de dois pinos digitais (*IN1 e IN2*), que determinam o sentido de rotação ou o modo de frenagem. A combinação desses sinais define se o motor irá girar no sentido horário (*CW*), anti-horário (*CCW*), parar livremente (*stop*) ou frear rapidamente (short break). A relação entre *input* e *output* do TB6612FNG está representada na Tabela 3.

Input				Output		
IN1	**IN2**	**PWM**	**STBY**	**OUT1**	**OUT2**	**Mode**
1	1	1	1	0	0	*Short break*
0	1	1	1	0	1	*CCK*
1	0	1	1	1	0	*CK*
0	0	1	1	1	1	*Stop*

Tabela 3 – Tabela de relação entre *input* e *output* do TB6612FNG

O *driver* deve ser alimentado por dois níveis de tensão distintos: a tensão VM, que fornece energia para os motores, e a tensão VCC, que alimenta os circuitos internos do componente. O terminal GND é comum a todo o circuito e deve ser compartilhado com o restante do sistema. No diagrama esquemático, o TB6612FNG é representado por um bloco retangular contendo seus 14 pinos, como na Figura 28.

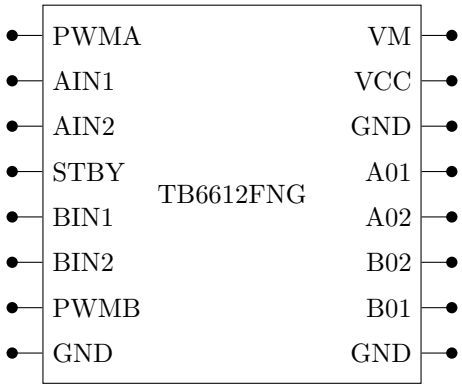


Figura 28 – Diagrama simplificado do driver de motor TB6612FNG.

2.2.7 Motor

O motor utilizado é um motor DC com escovas e caixa de redução metálica, alimentado com 12V e com rotação de 60 RPM. Esse modelo foi escolhido por oferecer torque suficiente para movimentar portões, com uma velocidade compatível com aplicações residenciais. No projeto, dois desses motores foram usados: um para o portão interno e outro para o externo.

Eles são controlados pelo *driver* TB6612FNG, que permite ajustar o sentido de rotação e a velocidade por meio de sinais PWM enviados pelo microcontrolador. A Figura 29 mostra o motor utilizado, com seus dois terminais de alimentação responsáveis por determinar o movimento de abertura ou fechamento.

Figura 29 – Motor DC 12V com redução metálica (60 RPM).

Fonte: (Pololu, 2025)

No diagrama esquemático, ele é representado por um bloco redondo contendo dois pinos, correspondentes às conexões positiva e negativa de alimentação, como na Figura 30.

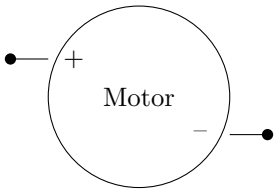


Figura 30 – Representação esquemática do motor.

2.2.8 Buzzer

O *buzzer* é um dispositivo piezoelétrico que converte sinais elétricos em som. O modelo utilizado neste projeto é um *buzzer* contínuo de 5V, que emite um som constante quando energizado. Ele é empregado para sinalizar alertas no sistema de alarme, fornecendo um aviso sonoro. A Figura 31 ilustra a aparência física do dispositivo.

Figura 31 – Buzzer contínuo de 5V utilizado no sistema de alarme.

Fonte: (TDK Product Center, 2025)

No diagrama esquemático, o *buzzer* é representado por um bloco redondo contendo dois pinos, correspondentes às conexões positiva e negativa de alimentação, como na Figura 32.

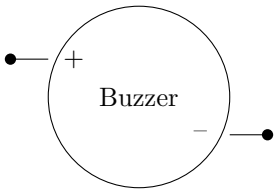


Figura 32 – Representação esquemática do buzzer com seus terminais de alimentação.

2.2.9 Resistores

O resistor é um componente eletrônico passivo cuja função é limitar o fluxo de corrente elétrica em um circuito, protegendo outros componentes e ajustando os níveis de tensão e corrente conforme necessário. Neste projeto, foram utilizados resistores de 10 kilo-ohms ($10k\Omega$), empregados como pull-up, e de 330 ohms (330Ω), utilizados para limitar a corrente nos LEDs. A Figura 33 ilustra a aparência física de um resistor.

Figura 33 – Resistor de 10 $k\Omega$ utilizado no projeto.

Fonte: (ELETROGATE, 2025c)

No diagrama esquemático, o resistor é representado por seu símbolo padrão com o seu valor indicado logo acima, como na Figura 34.

Figura 34 – Representação esquemática do resistor de 10 $k\Omega$.

2.2.10 Microfone Bluetooth

O microfone Bluetooth é o dispositivo responsável por captar o áudio do ambiente para a interpretação de comandos de voz no sistema. O modelo utilizado é um microfone duplo

profissional de lapela com conexão tipo C e tecnologia antirruído, conforme descrito em (World View, 2025). A Figura 35 apresenta a imagem do microfone utilizado no projeto.

Figura 35 – Microfone Bluetooth utilizado no projeto.

Fonte: (H'MASTON, 2025)

O microfone não será representado no circuito. Seu hub estará conectado à entrada USB tipo C do Raspberry Pi, e o microfone é Bluetooth, não possui cabos a serem conectados.

2.2.11 LCD com Módulo I2C

O display LCD é um componente eletrônico utilizado para exibir informações visuais, como textos e números, facilitando a interação com o usuário em sistemas embarcados. Quando equipado com um módulo I2C, o LCD permite comunicação simplificada com o microcontrolador, utilizando apenas dois fios (SDA e SCL) para transmissão de dados, reduzindo o número de pinos necessários para a comunicação de seis para dois. Neste projeto, foi utilizado um display LCD padrão de 16x2 caracteres com módulo I2C. A função do LCD é mostrar o status do alarme na central de controle, indicando se o sistema está armado ou não e se há violações. A Figura 36 ilustra um LCD com módulo I2C integrado.

Figura 36 – Display LCD com módulo I2C utilizado no projeto.

Fonte: (ELETROGATE, 2025a)

No diagrama esquemático, o LCD com módulo I2C é representado por um bloco retangular contendo quatro pinos, correspondentes às conexões positiva e negativa de alimentação e aos pinos SDA e SCL da comunicação, como na Figura 32.

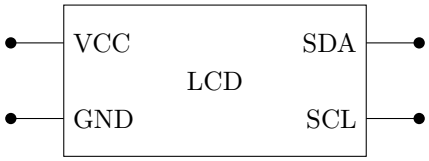


Figura 37 – Representação esquemática do display LCD com módulo I2C.

2.2.12 Botão Momentâneo

O botão momentâneo é uma chave táctil que estabelece contato elétrico apenas enquanto está sendo pressionado, permitindo a passagem de corrente temporariamente. No projeto, foram utilizados nove botões momentâneos para controlar as luzes e os portões de forma manual, pela central de controle. A Figura 38 ilustra o botão utilizado.

Figura 38 – Botão momentâneo utilizado no projeto.

Fonte: (BLOG DA ROBÓTICA, 2022)

No diagrama esquemático, o botão momentâneo é representado por um bloco redondo dentro de um bloco quadrado contendo quatro pinos, correspondentes aos dois contatos isolados do botão. Os dois pinos superiores estão sempre ligados entre si, assim como os dois inferiores. No entanto, os conjuntos superior e inferior só se conectam quando o botão é pressionado, conforme ilustrado na Figura 39.

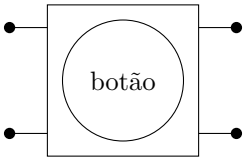


Figura 39 – Representação esquemática do botão momentâneo.

2.2.13 Botão com Retenção

O botão com retenção é um dispositivo que mantém seu estado após ser acionado, funcionando como uma chave liga/desliga. No projeto, foi utilizado um botão com retenção e LED integrado para armar e desarmar o sistema de alarme. A Figura 40 ilustra o modelo utilizado.

Figura 40 – Botão com retenção utilizado no projeto.

Fonte: (SMART KITS, 2025)

No diagrama esquemático, o botão com retenção é representado por um bloco redondo dentro de outro bloco redondo maior, contendo quatro pinos. Dois desses pinos correspondem à alimentação do LED interno do botão (VCC e GND), localizados nas posições superior e inferior. Os outros dois pinos, dispostos lateralmente, correspondem ao contato normalmente aberto, que se conectam quando o botão é pressionado. Essa configuração está ilustrada na Figura 41.

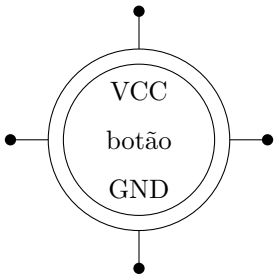


Figura 41 – Representação esquemática do botão com retenção.

2.2.14 Sensor Reed

O sensor *reed* é um dispositivo eletromecânico composto por dois contatos metálicos selados em um tubo de vidro. Quando exposto a um campo magnético, como o gerado por um ímã de neodímio, os contatos se fecham, permitindo a detecção da presença do ímã. No projeto, o sensor *reed* foi utilizado em conjunto com ímãs de neodímio para simular sensores de alarme nas portas e janelas da residência, tornando possível o monitoramento de sua abertura e fechamento. A Figura 42 ilustra o sensor empregado.

Figura 42 – Sensor reed.

Fonte: (USINAINFO, 2025b)

No diagrama esquemático, o sensor *reed* é simbolizado por uma chave normalmente aberta, como na Figura 43.

Figura 43 – Representação do sensor *reed* como uma chave normalmente aberta.

2.2.15 Eletroímã

O eletroímã é um componente eletromecânico formado por uma bobina de fio condutor enrolada em torno de um núcleo ferromagnético. Quando uma corrente elétrica passa pela bobina, gera-se um campo magnético capaz de atrair objetos metálicos, funcionando como um ímã temporário (DANTAS, 2025). No projeto, o eletroímã foi utilizado para simular mecanismos de travamento e liberação de portões residenciais, permitindo o controle remoto da ativação magnética. A Figura 44 apresenta o mini eletroímã empregado.

Figura 44 – Mini eletroíma.

Fonte: (Magnet Schultz, 2025)

No diagrama esquemático, o eletroímã é simbolizado por bobina com dois traços paralelos, como na Figura 45.

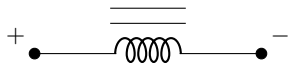


Figura 45 – Representação do eletroímã no circuito.

2.2.16 Placa de Fenolite Ilhada e Fios de Conexão

A placa de fenolite ilhada é um tipo de placa utilizada para montagem de circuitos eletrônicos, composta por uma base de fenolite revestida com pequenas ilhas de cobre isoladas entre si. Essas ilhas permitem a soldagem de componentes e a criação de conexões por meio de fios ou pontes de solda. No projeto, essa placa foi empregada na montagem dos circuitos eletrônicos, sendo escolhida por ser uma alternativa de baixo custo e de fácil acesso. Para a fiação do circuito, foram utilizados fios retirados de um cabo de rede, por fornecer oito fios internos, isolados individualmente e com bitola compatível com a furação da placa de fenolite ilhada.

2.2.17 ESP32

Fabricado pela Espressif Systems, o ESP32 é um microcontrolador que oferece suporte integrado a Wi-Fi e Bluetooth. A arquitetura do ESP32 conta com dois núcleos de processamento (dual-core), podendo operar até 240 MHz, além de contar com múltiplos pinos de entrada e saída digital (GPIOs), conversores analógico-digital (ADC), interfaces de comunicação como UART, SPI, I2C e PWM, fundamentais para a integração com sensores, atuadores e demais periféricos do sistema.

No presente projeto, foi utilizado o ESP32 DevKit, uma placa de desenvolvimento que integra o chip ESP32 juntamente com os circuitos necessários para seu funcionamento. Essa placa facilita significativamente o processo de gravação de firmware, pois já incorpora um conversor USB-serial e circuitos auxiliares, eliminando a necessidade de programadores externos. A programação é realizada diretamente via conexão USB, o que proporciona maior praticidade durante o desenvolvimento, testes e depuração do sistema. Além disso, a placa também conta com um circuito de proteção contra sobrecorrente e sobretensão, além de um regulador interno que fornece 3,3 V para o chip e também à saída VCC, permitindo a alimentação de periféricos compatíveis com essa tensão.

Na Figura 46, é possível visualizar a disposição dos pinos do ESP32 DevKit, bem como sua representação simplificada no circuito, que considerará apenas as suas portas.

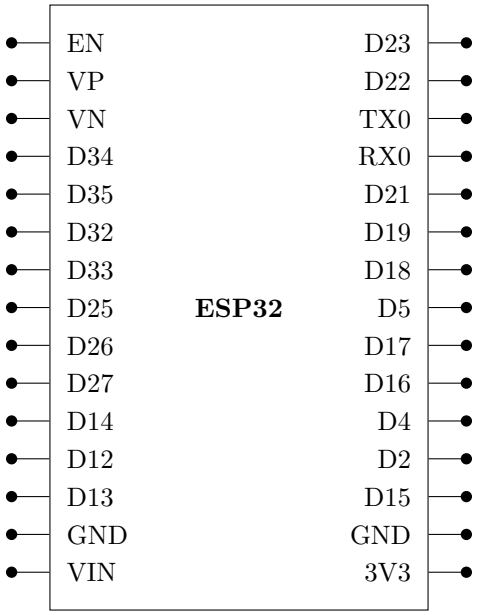


Figura 46 – Pinagem do ESP32 DevKit.

Começando pelos pinos de controle e alimentação. O pino EN (Enable) tem a função de resetar o chip: quando em nível lógico alto, o ESP32 opera normalmente; ao receber nível baixo por um breve instante, o chip é reiniciado. Do lado esquerdo, os pinos VIN e GND podem ser utilizados para alimentação do módulo, como alternativa ao conector USB. O GND corresponde à referência, enquanto o VIN recebe a tensão de entrada. Já no lado direito, há um segundo pino GND destinado à referência de circuitos externos, e o pino 3V3 fornece 3,3 V regulados, podendo ser utilizado para alimentar periféricos.

Todos os outros pinos funcionam como GPIO e PWM, alguns podem funcionar como SPI, I2C, ADC, DAC, Touch, UART e controle. Segue a lista com a funcionalidade de cada pino:

- **VP**: GPIO36, PWM e ADC1 - Apenas Input
- **VN**: GPIO39, PWM e ADC1 - Apenas Input
- **D34**: GPIO34, PWM e ADC1 - Apenas Input
- **D35**: GPIO35, PWM e ADC1 - Apenas Input
- **D32**: GPIO32, PWM, ADC1, Touch e XTAL32
- **D33**: GPIO33, PWM, ADC1, Touch e XTAL32

- **D25**: GPIO25, PWM, ADC2 e DAC1
- **D26**: GPIO26, PWM, ADC2 e DAC2
- **D27**: GPIO27, PWM, ADC2 e Touch
- **D14**: GPIO14, PWM, ADC2, Touch e HSPI_CLK
- **D12**: GPIO12, PWM, ADC2, Touch e HSPI_MISO
- **D13**: GPIO13, PWM, ADC2, Touch e HSPI_MOSI
- **D23**: GPIO23, PWM, MOSI e VSPI_MOSI
- **D22**: GPIO22, PWM e SCL
- **TX0**: GPIO1, PWM, UART TX0 e CLK3
- **RX0**: GPIO3, PWM, UART RX0 e CLK2
- **D21**: GPIO21, PWM e SDA
- **D19**: GPIO19, PWM, MISO e VSPI_MISO
- **D18**: GPIO18, PWM, SCK e VSPI_CLK
- **D5**: GPIO5, PWM e VSPI_CS
- **D17**: GPIO17, PWM e UART TX2
- **D16**: GPIO16, PWM e UART RX2
- **D4**: GPIO4, PWM, ADC2 e Touch
- **D2**: GPIO2, PWM, ADC2, Touch e CS
- **D15**: GPIO15, PWM, ADC2, Touch e HSPI_CS

Os pinos utilizados no projeto e suas aplicações estão descritos abaixo:

Função	Pino (GPIO)	Descrição
Controle de Motores – TB6612FNG		
AIN1	D27 (GPIO27)	Controle de direção do motor A
AIN2	D5 (GPIO5)	Controle de direção do motor A
BIN1	D25 (GPIO25)	Controle de direção do motor B
BIN2	D26 (GPIO26)	Controle de direção do motor B
Controle de Luzes (relés ou LEDs)		
Sala de TV	D4 (GPIO4)	Saída digital para iluminação
Quarto 2	D16 (GPIO16)	Saída digital para iluminação
Sala de Estar	D17 (GPIO17)	Saída digital para iluminação
Garagem	D18 (GPIO18)	Saída digital para iluminação
Quintal	D19 (GPIO19)	Saída digital para iluminação
Quarto Casal	D23 (GPIO23)	Saída digital para iluminação
Sistema de Alarme		
Alarme Porta	D13 (GPIO13)	Entrada digital (sensor reed)
Janela – Sala de Estar	D14 (GPIO14)	Entrada digital (sensor reed)
Janela – Sala de TV	D33 (GPIO33)	Entrada digital (sensor reed)
Janela – Quarto Casal	D34 (GPIO34)	Entrada digital (sensor reed)
Janela – Quarto 2	D35 (GPIO35)	Entrada digital (sensor reed)
Buzzer	D32 (GPIO32)	Saída digital (relé)
Controle do Portão de Pedestres		
Eletroímã	D15 (GPIO15)	Saída digital (relé)
Comunicação I2C (LCD)		
SCL	D22 (GPIO22)	Clock da comunicação I2C
SDA	D21 (GPIO21)	Dados da comunicação I2C
Comunicação UART (com Raspberry Pi)		
TX0	D1 (GPIO1)	Transmissão serial
RX0	D3 (GPIO3)	Recepção serial

Tabela 4 – Mapeamento dos pinos do ESP32 utilizados no projeto.

2.3 Raspberry Pi 4

O Raspberry Pi 4 é um minicomputador de placa única desenvolvido pela Raspberry Pi Foundation (Raspberry Pi Foundation, 2025a). Ele oferece uma solução compacta e versátil para aplicações embarcadas, automação residencial, aprendizado de programação e desenvolvimento de projetos com inteligência artificial, visão computacional, entre outros. A placa conta com um processador quad-core Cortex-A72 de 64 bits, conectividade integrada via Wi-Fi e Bluetooth,

múltiplas interfaces USB, saída de vídeo em 4K via HDMI, entrada para cartão microSD e uma interface de 40 pinos GPIO que permite a comunicação com sensores, atuadores e periféricos externos (Raspberry Pi Foundation, 2025a).

No projeto apresentado, o Raspberry Pi 4 é responsável pelo processamento do reconhecimento de voz com o auxílio da biblioteca VOSK, atuando também como central de controle dos botões físicos. O modelo escolhido foi o de 4 GB de RAM, a fim de garantir o alto desempenho do sistema.

Figura 47 – Placa Raspberry Pi 4.

Fonte: (Raspberry Pi Foundation, 2025a)

A seguir, apresenta-se a representação esquemática do Raspberry Pi 4 no diagrama do circuito, considerando as conexões da interface GPIO:

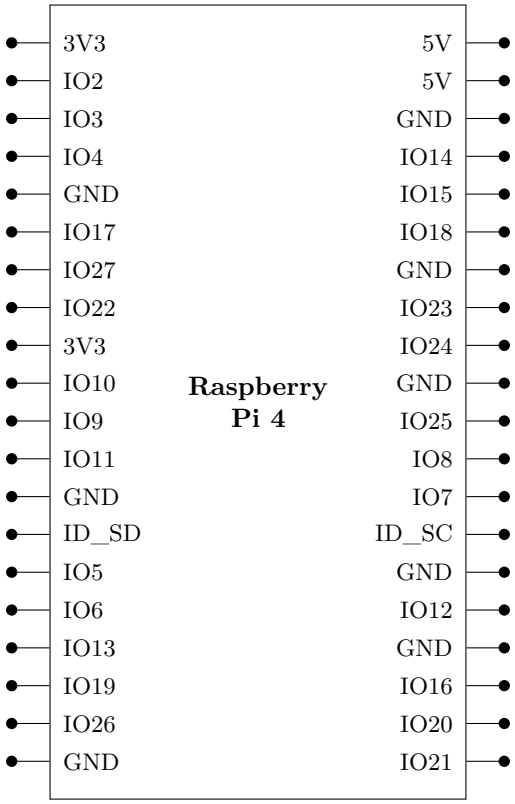


Figura 48 – Distribuição dos pinos GPIO no conector do Raspberry Pi 4.

- **3V3**: Saída de 3,3V.
- **5V**: Saída de 5V.
- **GND**: Referência (comum para todos os circuitos).
- **GPIO2**: Comunicação I2C (SDA).
- **GPIO3**: Comunicação I2C (SCL).
- **GPIO4**: GPIO geral.
- **GPIO17**: GPIO geral.
- **GPIO27**: GPIO geral.
- **GPIO22**: GPIO geral.
- **GPIO10**: SPI (MOSI).
- **GPIO9**: SPI (MISO).

- **GPIO11**: SPI (SCLK).
- **ID_SD**: Detecção de EEPROM I2C.
- **GPIO5**: GPIO geral.
- **GPIO6**: GPIO geral.
- **GPIO13**: PWM.
- **GPIO19**: PWM, SPI (MISO).
- **GPIO26**: PWM.
- **GPIO14**: UART (TXD).
- **GPIO15**: UART (RXD).
- **GPIO18**: PWM, SPI (SCLK).
- **GPIO23**: GPIO geral.
- **GPIO24**: GPIO geral.
- **GPIO25**: GPIO geral.
- **GPIO8**: SPI (CE0).
- **GPIO7**: SPI (CE1).
- **ID_SC**: Detecção de EEPROM I2C.
- **GPIO12**: PWM.
- **GPIO16**: GPIO geral.
- **GPIO20**: GPIO geral.
- **GPIO21**: GPIO geral.

2.4 Sistemas

Para o desenvolvimento do projeto, foram empregados sistemas auxiliares que serão apresentados a seguir.

2.4.1 Ubuntu

O sistema operacional utilizado no computador responsável pelo desenvolvimento e teste do sistema foi o Ubuntu 24.10, devido à sua compatibilidade com as ferramentas adotadas, como o ESP-IDF, a comunicação via SSH, a configuração de chaves criptográficas e demais recursos necessários para o desenvolvimento embarcado. Embora seja possível utilizar outros sistemas operacionais, este trabalho não aborda o passo a passo de configuração em ambientes distintos, sendo responsabilidade do usuário realizar as devidas adaptações conforme sua aplicação.

2.4.2 ESP-IDF

A ferramenta utilizada para criar os diretórios do projeto, compilar o código e realizar o upload do firmware para o microcontrolador foi o ESP-IDF (Espressif IoT Development Framework). A escolha se deu por ser a solução oficial recomendada pela própria fabricante Espressif, garantindo a compatibilidade com os recursos do ESP32 e suporte às atualizações mais recentes da plataforma (Espressif Systems, 2025).

2.4.3 VOSK

O VOSK é uma biblioteca de reconhecimento de fala *offline*, de código aberto, compatível com diversas linguagens de programação, como Python, Java, C# e Node.js. Seu principal diferencial está na capacidade de realizar a transcrição de voz localmente, sem a necessidade de conexão com a Internet, o que proporciona maior privacidade, independência de rede e menor latência nas respostas. Além disso, apresenta bom desempenho mesmo em dispositivos com recursos limitados, como o Raspberry Pi, sendo capaz de operar em tempo real. Ainda, é possível personalizar o vocabulário do modelo, de modo que a transcrição se limite apenas às palavras previamente definidas por meio do código. A versão utilizada é a 0.3.45 e a escolha pelo VOSK justifica-se pelo foco em segurança e na execução do processamento de forma local (Alpha Cephei, 2025).

2.4.4 Raspberry Pi Imager

O Raspberry Pi Imager foi utilizado para a preparação do cartão SD com o sistema operacional do Raspberry Pi. Essa ferramenta oficial permite a seleção e gravação rápida e segura de imagens de sistemas operacionais compatíveis diretamente no cartão, facilitando a configuração inicial do dispositivo. O uso do Raspberry Pi Imager garante a integridade do sistema instalado e simplifica o processo de instalação (Raspberry Pi Foundation, 2025b).

3 Implementação e Resultados

Este capítulo apresenta o processo de implementação do sistema proposto, abordando tanto a montagem física quanto o desenvolvimento dos códigos embarcados e suas integrações, além de expor os resultados obtidos a partir dos testes realizados.

3.1 Montagem da Estrutura Física da Maquete

Todos os componentes eletrônicos que precisaram ser fixados na maquete, por fazerem parte da estrutura, tiveram fios soldados aos seus terminais, permitindo o acesso por baixo do MDF para a montagem posterior do circuito, sendo eles: sensores *reed*, motores, fitas de LED branco, LEDs, eletroímã, botões e LCD. As peças projetadas em 3D foram impressas utilizando filamento PLA. A escolha desse material se deve à maior facilidade de impressão em comparação ao ABS. Mesmo sendo menos flexível e menos resistente ao calor, foi uma escolha adequada, considerando que a estrutura não será submetida a grandes forças nem exposta a fontes de calor.

Primeiramente, após a impressão das peças, as janelas foram acopladas à estrutura da casa central utilizando cola de silicone para fixar os trilhos no batente da janela, já com as folhas posicionadas entre eles. Para as janelas menores, com apenas duas folhas, foi colocada uma folha móvel em cada extremidade do trilho, com o puxador voltado para a parte externa. Para as janelas maiores, com quatro folhas, as duas folhas fixas foram coladas nos trilhos internos, encostadas ao batente da janela, enquanto as folhas móveis foram alinhadas nos trilhos externos, com os puxadores voltados para a parte interior. Com as janelas devidamente alinhadas, os trilhos foram colados ao batente, conforme ilustrado na Figura 5. Para as portas, foi suficiente encaixá-las no eixo fixado ao batente.

Após a fixação das portas e janelas na casa, ímãs foram instalados com o uso de massa adesiva Durepoxi, sendo posicionados no canto superior interno das folhas móveis e na parte inferior das duas portas. Com os ímãs fixados, as fitas de LED branco foram cortadas no tamanho de cada cômodo e acopladas às sancas utilizando cola de silicone. Os fios das fitas e dos sensores foram direcionados por trás das sancas até a parte inferior da maquete e a sanca foi colada na parede com cola de silicone. Os sensores *reed* foram posicionados com o auxílio de um multímetro no modo de teste de continuidade, garantindo que o sensor estivesse detectando o ímã. Após identificar a posição correta, o sensor foi fixado utilizando a mesma massa adesiva Durepoxi.

Para a montagem, após finalizadas, a garagem e a casa foram posicionadas sobre a base de MDF, sendo marcados e furados os locais destinado à passagem dos fios e da correia dentada. Com os furos definidos, as peças foram fixadas ao MDF utilizando cola quente. Também foram coladas as cercas nos lados frontal e esquerdo da maquete, os portões externo e de pedestres

foram instalados, e as furações para a passagem dos fios do motor e do eletroímã foram realizadas.

Ainda, o portão interno foi colado nos guias previamente fixados ao eixo (Figura 9a). A correia dentada foi unida utilizando a técnica de desgastar metade da espessura da correia na parte traseira de uma ponta e metade na parte frontal da outra, unindo-as com cola de silicone, preservando seu formato, espessura e flexibilidade (Figura (Selfcrafts, 2023)). Em seguida, a correia foi encaixada no eixo conectado ao portão (Figura 8f), conduzida pela parte inferior da maquete, e o adaptador já encaixado no motor (Figura 9b) foi acoplado a ela. O motor foi então colado na parte inferior do MDF de forma que a correia permanecesse o mais esticada possível, evitando deslizamentos durante o funcionamento. Por fim, após a realização dos furos para passagem dos fios, todos os botões foram colados na central de controle, juntamente com o display LCD. Em seguida, a central foi fixada à base de MDF com cola quente. Os postes, já contendo os LEDs, também foram colados à estrutura.

Como resultado da estrutura, as peças atenderam aos requisitos da maquete em termos de resistência e dimensões, sendo necessário apenas o corte de uma das grades para adaptação ao perímetro da maquete, além de ajustes no portão de pedestres, que não possuí, no projeto original, um espaço destinado ao eletroímã. Como sugestão de melhoria, a utilização de uma chapa de MDF mais fina, como a de três milímetros, atenderia à aplicação, facilitando a execução dos furos e reduzindo o peso da estrutura. Além disso, o aumento do diâmetro das polias dentadas contribuiria para uma maior área de contato com a correia, melhorando a aderência do conjunto e reduzindo o risco de deslizamentos. Por fim, um espaço maior para a passagem dos fios na parte superior da sanca poderia simplificar o processo de montagem.

3.2 Montagem do Circuito Eletrônico

Um aspecto fundamental na montagem dos circuitos apresentados nesta seção é a garantia de que todos os terminais de referência (GND) estejam conectados entre si. A ausência dessa unificação é um erro recorrente em montagens práticas e pode comprometer o funcionamento de todo o sistema, ocasionando comportamentos inesperados ou falhas de comunicação entre os componentes. A montagem do sistema será apresentada de forma fragmentada, destacando individualmente cada circuito com suas respectivas referências e requisitos. Dessa forma, ao realizar a montagem de cada segmento respeitando as referências estabelecidas, é garantida a integração correta dos módulos, resultando na construção funcional do sistema completo. Nos circuitos, as portas dos microcontroladores serão referenciadas com o prefixo **D** quando conectadas ao **ESP32**, e com o prefixo **IO** quando conectadas ao **Raspberry Pi**.

3.2.1 Iluminação

Para o circuito de iluminação, optou-se pela utilização de relés em vez de transistores, como o 2N2222, pois, embora os transistores atendam aos requisitos do protótipo, os relés podem

ser usados em um sistema elétrico real, conferindo maior fidelidade entre o protótipo e sua versão final. Foram utilizados dois módulos relé de quatro canais, totalizando oito canais. Desses, seis foram destinados ao controle da iluminação, cuja montagem dos respectivos circuitos será detalhada nesta seção.

Cada setor iluminado possui um circuito independente, compartilhando apenas a mesma fonte de alimentação. As fitas de LED branco utilizam um terminal COM comum de 12V, enquanto os LEDs da iluminação externa utilizam um COM de 5V. Cada área a ser iluminada está associada a um canal específico do módulo relé e a uma porta digital dedicada no ESP32. O circuito de iluminação da Sala de TV, Quarto 2 Sala de Estar, Garagem e Quarto Casal podem ser conferidos nas Figuras 49, 50, 51, 52 e 53, respectivamente.

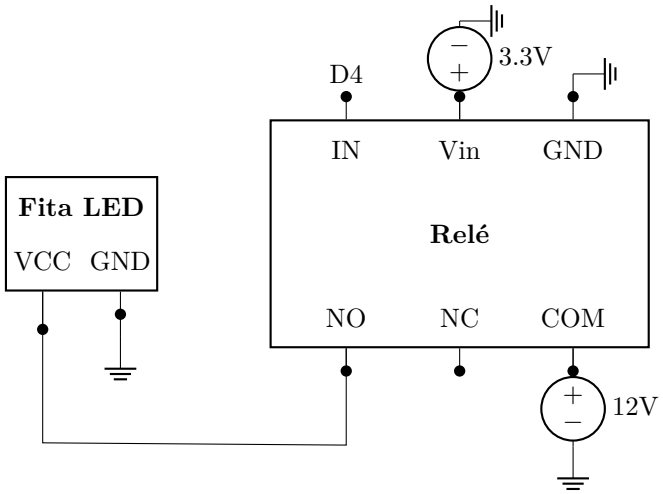


Figura 49 – Circuito de iluminação da Sala de TV.

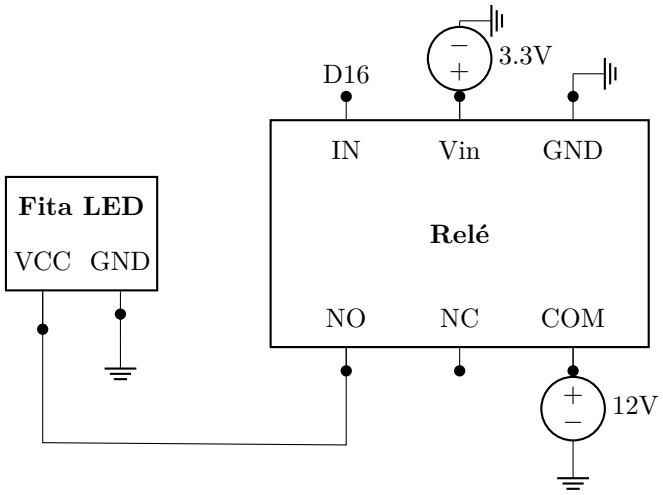


Figura 50 – Circuito de iluminação da Quarto 2.

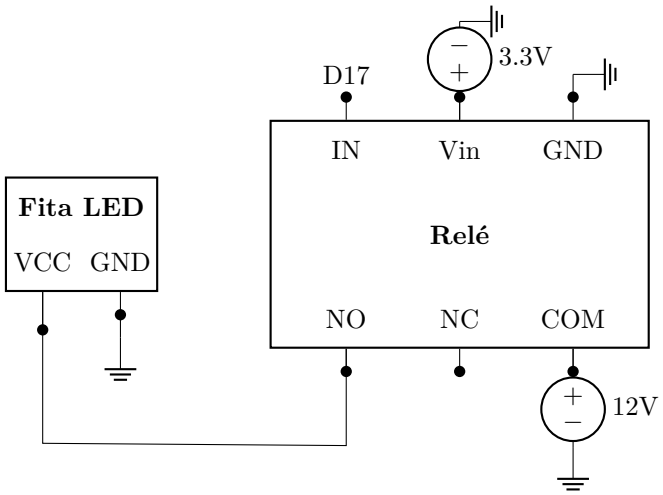


Figura 51 – Circuito de iluminação da Sala de Estar.

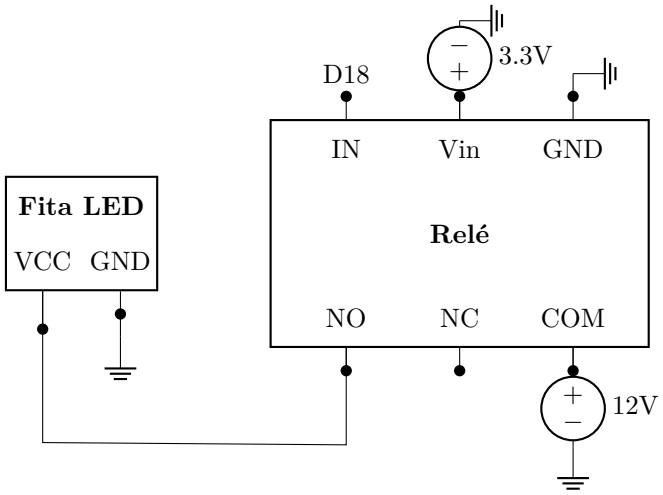


Figura 52 – Circuito de iluminação da Garagem.

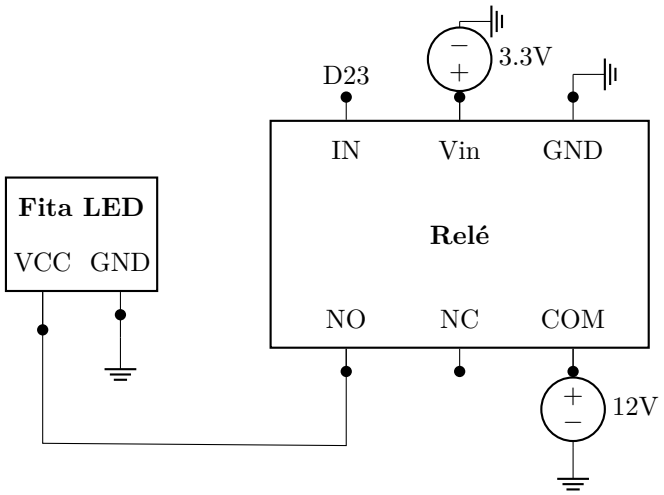


Figura 53 – Circuito de iluminação da Quarto Casal.

Para o circuito do quintal foram utilizados 8 LEDs amarelos em paralelo, pois não julgou-se necessário que cada poste do quintal tivesse um controle separado, a ligação em paralelo garante que, caso um LED do circuito queime, toda a iluminação externa não seja comprometida. Para cada LED há um resistor de 330Ω. A montagem pode ser vista na Figura 54.

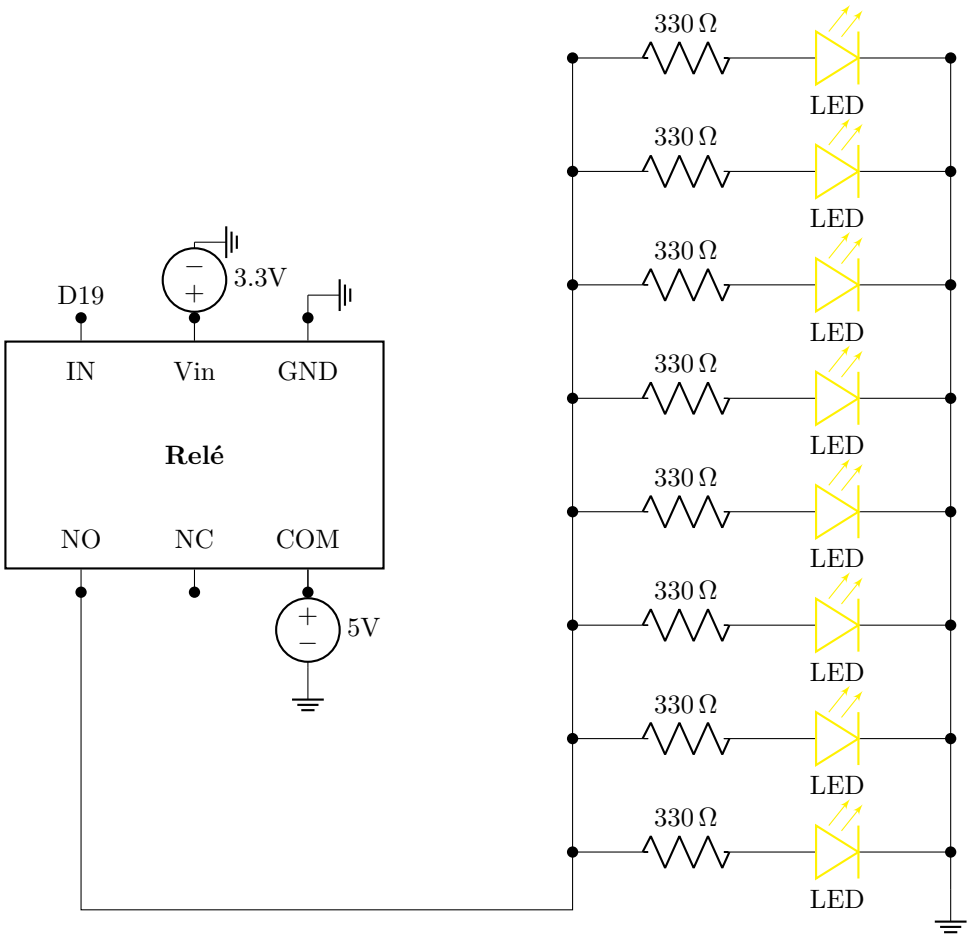


Figura 54 – Circuito de iluminação do Quintal.

Para a montagem do circuito da fita de LED RGB endereçável é necessário apenas o fornecimento de uma alimentação de 12V e um sinal de dados. Por falta de pinos do ESP32, a fita é controlada por uma porta GPIO do Raspberry Pi. O circuito segue ilustrado na Figura 55.

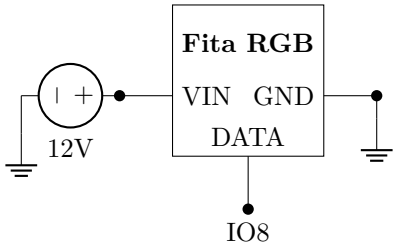


Figura 55 – Circuito da fita RGB endereçável.

Como resultado dos circuitos de iluminação, observou-se que a fonte de alimentação

de 12V e 1,5A foi insuficiente. A fita de LED RGB endereçável, não prevista no escopo inicial do projeto, foi implementada posteriormente a fim de fornecer uma resposta visual do sistema. No entanto, a fonte utilizada não foi capaz de suprir a corrente total exigida pelo circuito. Em situações em que a fita RGB estava ativa e um comando era enviado para acionar uma das fitas de LED branco, a luz branca apresentava tentativa de acionamento, mas se apagava imediatamente devido à limitação de corrente.

A solução adotada foi a limitação do brilho da fita RGB por meio do software, reduzindo assim a demanda de corrente. Como proposta de melhoria para o protótipo, recomenda-se a substituição da fonte atual por uma fonte de 12V com capacidade de 5A, conforme justificado pelas equações apresentadas a seguir.

Considerando todas as fitas de LED branco ligadas:

$$P = 12W/m \times 1,32m = 15,85W \tag{3.1}$$

$$I_B = \frac{P}{V} = \frac{15,85W}{12V} = 1,32A \tag{3.2}$$

Considerando a corrente máxima de 80mA nos motores:

$$I_M = 80mA \times 2 = 160mA = 0,16A \tag{3.3}$$

Corrente consumida da fonte 12V pelo sistema sem a fita RGB:

$$I_T = 1,32A + 0,16A = 1,48A < 1,5A \tag{3.4}$$

Corrente na fita de LED RGB:

$$P = 14W/m \times 2,1m = 29,4W \tag{3.5}$$

$$I_R = \frac{P}{V} = \frac{29,4W}{12V} = 2,45A \tag{3.6}$$

Corrente consumida da fonte 12V pelo sistema com a fita RGB:

$$I_T = 1,32A + 0,16A + 2,45A = 3,93A > 1,5A \tag{3.7}$$

Assim, uma fonte de 12V com 5A é capaz de alimentar o sistema com uma margem de segurança de aproximadamente 1A.

3.2.2 Alarme

Para o sistema de alarme, foram utilizados sensores do tipo *reed*. Os circuitos foram organizados por zonas, de modo que cada cômodo possui uma porta dedicada no ESP32. Os

sensores *reed* de cada cômodo foram conectados em série, uma vez que a intenção é que a porta digital acione o alarme assim que qualquer um dos sensores for violado e a chave *reed* for aberta. Os circuitos correspondentes estão representados nas Figuras 56, 57, 58, 59 e 60.

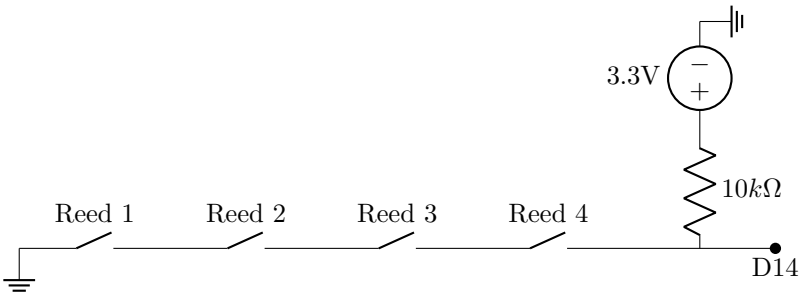


Figura 56 – Circuito de alarme das janelas da Sala de Estar.

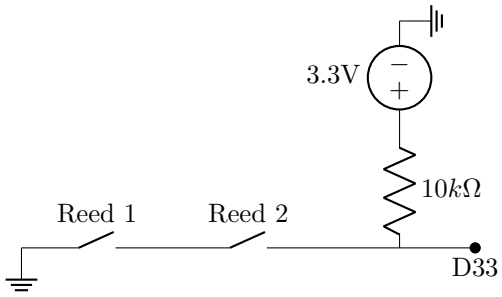


Figura 57 – Circuito de alarme das janelas do Sala de TV.

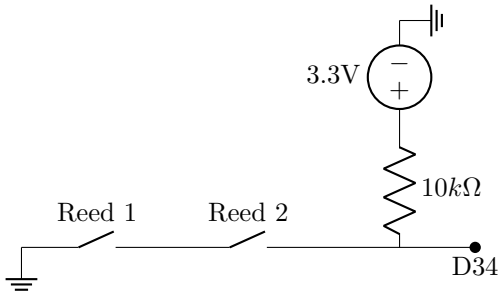


Figura 58 – Circuito de alarme das janelas do Quarto Casal.

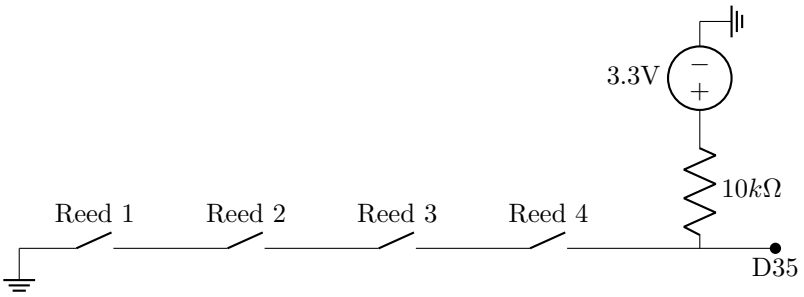


Figura 59 – Circuito de alarme das janelas da Quarto 2.

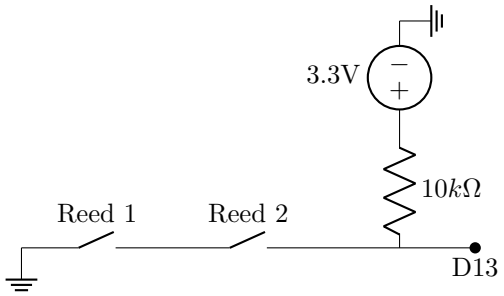


Figura 60 – Circuito de alarme das portas.

Como resultado, os testes realizados com os sensores do tipo *reed* demonstraram que o sistema de alarme é capaz de identificar violações nas janelas e portas, desde que esteja previamente ativado. A organização dos sensores em série permitiu que uma única leitura digital por cômodo fosse suficiente para detectar a abertura de qualquer uma das janelas ou portas monitoradas.

Observou-se, entretanto, que esses sensores apresentam sensibilidade devido à cápsula de vidro que os envolve, o que resultou na quebra de alguns deles durante o processo de montagem, comprometendo o funcionamento adequado do sistema. Apesar dessa fragilidade, os sensores *reed* ainda se mostraram a melhor escolha para o protótipo, especialmente por seu tamanho.

3.2.3 Motores

Os dois motores do projeto foram conectados ao mesmo *driver* de motor. Para controlar o sentido de rotação dos motores, foram utilizadas quatro portas digitais do microcontrolador, duas para cada motor. A ligação elétrica dos motores está ilustrada na Figura 61.

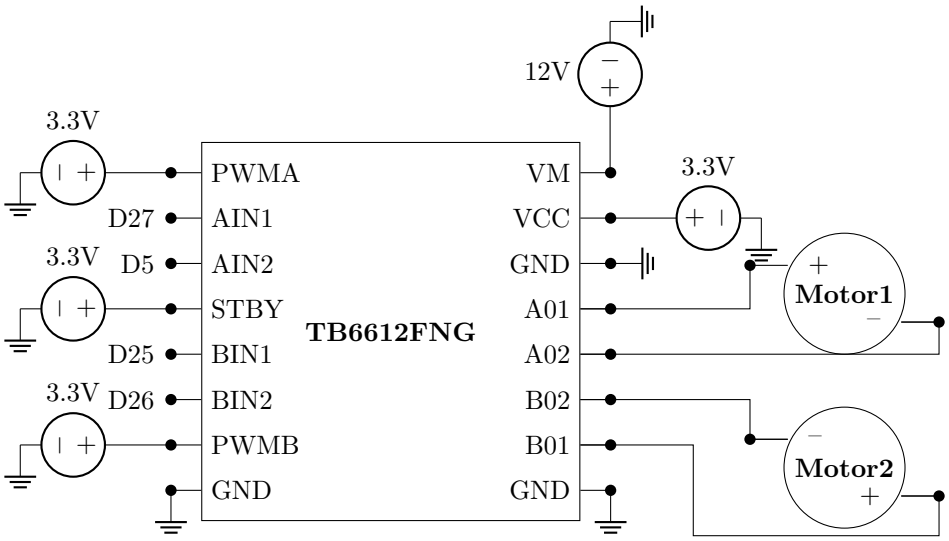


Figura 61 – Circuito dos motores com o driver.

Como resultado, o sistema controlou com sucesso os dois motores por meio do *driver*, permitindo a alteração do sentido de rotação conforme esperado. Entretanto, o protótipo utilizou motores e *driver* em escala reduzida para fins de simulação, estando distante da configuração de um sistema real de portão automático. Em um cenário real, seria possível utilizar o ESP32 para enviar pulsos ao *driver* do motor já existente na residência, simulando o controle manual do portão. Caso o *driver* não possua essa funcionalidade, ainda é viável programar um emissor de radiofrequência para replicar os comandos do controle remoto do portão.

3.2.4 Portão de Pedestres

Para o circuito do Portão de Pedestres, foram realizadas as conexões entre o eletroímã, o relé e o microcontrolador, conforme ilustrado na Figura 62.

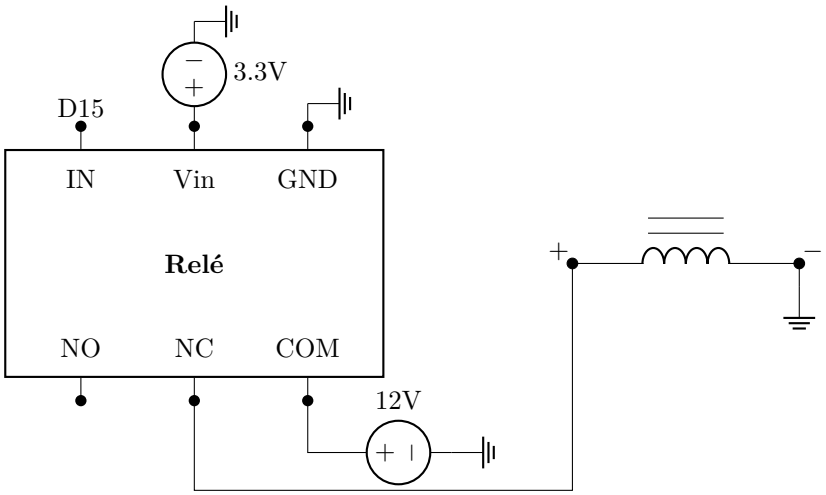


Figura 62 – Circuito do Portão de Pedestres.

Como resultado, o portão de pedestre controlado pelo sistema apresentou limitações devido ao eletroímã escolhido para a trava elétrica. Durante os testes, verificou-se que o eletroímã não forneceu força suficiente para garantir o travamento do portão. Contudo, esse componente é apenas para nível de simulação, em um projeto real seria utilizada a fechadura elétrica já existente e funcional na residência.

3.2.5 Central de Controle

Para o circuito da central de controle, os botões foram conectados às portas do Raspberry Pi no modo *pull-up*, utilizando resistores externos de $10\,k\Omega$. Todos os botões seguem a mesma configuração, diferenciando-se apenas pelas portas utilizadas. O botão de retenção, especificamente, recebe alimentação adicional para o acionamento do LED integrado. No caso do display LCD, foram utilizadas apenas as conexões de alimentação e os pinos SDA e SCL. Os circuitos estão representados nas Figuras 63, 64 e 65.

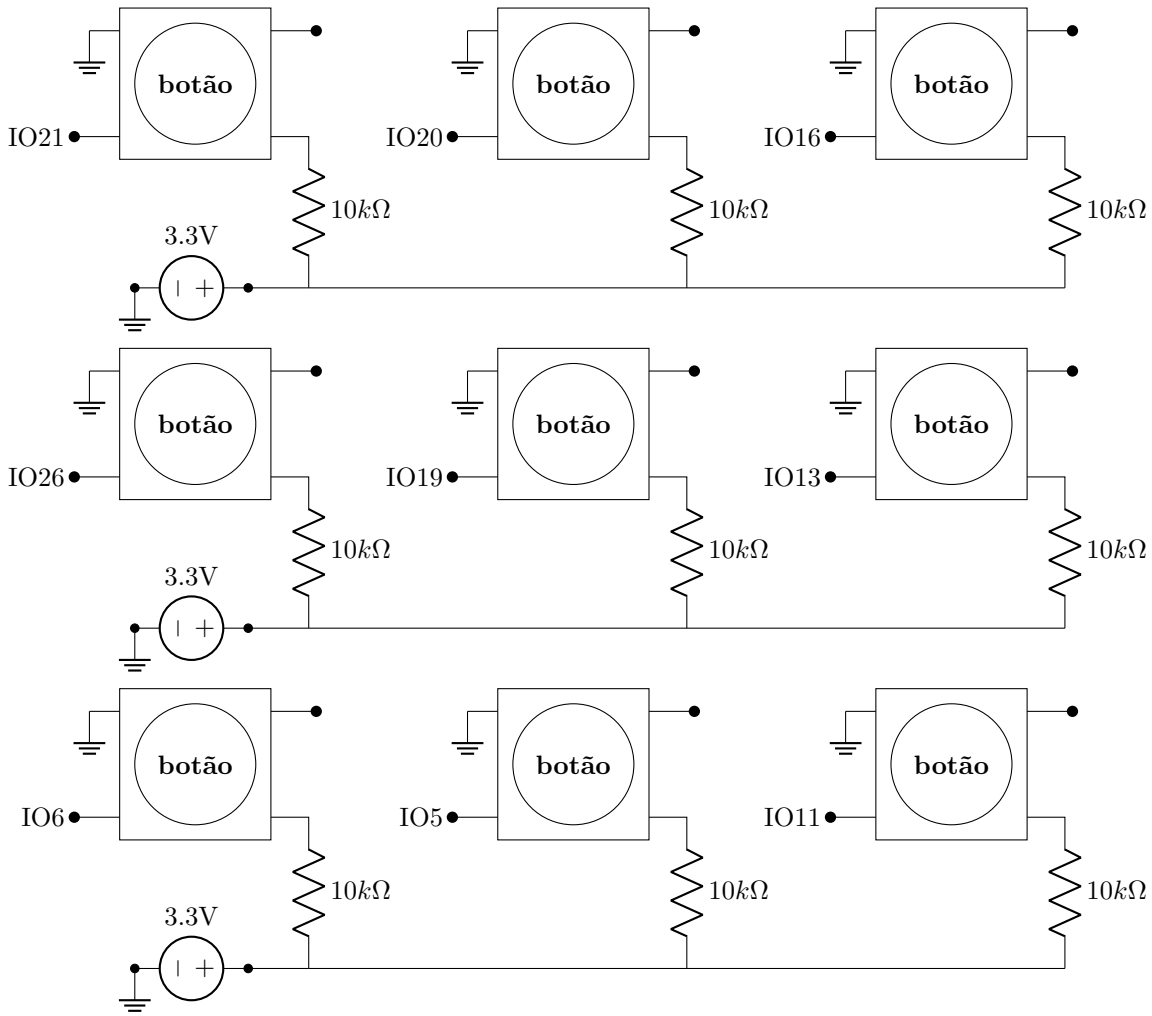


Figura 63 – Botões para controle das luzes e portões.

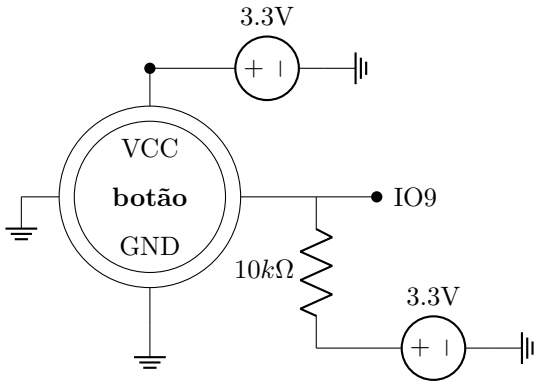


Figura 64 – Circuito do botão de arme e desarme.

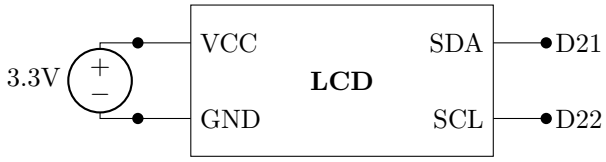


Figura 65 – Circuito do LCD com módulo I2C.

Como resultado, a central de controle comportou-se conforme o esperado, reconhecendo corretamente as interações realizadas por meio dos botões e exibindo os estados do alarme no display LCD. Os botões físicos foram utilizados apenas no protótipo, visto que, em uma residência real, o acionamento de lâmpadas e portões deve ser realizado por meio dos interruptores já existentes.

3.3 Codificação do Sistema

Para o funcionamento completo do sistema, foram desenvolvidos três módulos principais. O primeiro corresponde ao firmware do ESP32, programado em linguagem C, estruturado em quatro arquivos: um arquivo principal (main), um arquivo destinado às funções de inicialização, um arquivo responsável pelas funções de execução e tarefas (tasks), e um arquivo de cabeçalho (.h), que reúne todas as declarações e funções necessárias para serem importadas no arquivo principal.

No Raspberry Pi, foram implementados dois códigos em Python. O primeiro é responsável pelo reconhecimento de voz e leitura dos botões físicos, enquanto o segundo realiza o controle da fita de LED RGB endereçável.

Os códigos-fonte desenvolvidos podem ser acessados em (BARBOSA, 2025).

3.3.1 Arquivo `Main.c`

Este arquivo representa o núcleo da aplicação embarcada. Ele executa a configuração inicial dos periféricos, monitora os comandos recebidos pela UART e despacha tarefas de acordo com os comandos interpretados.

- Inicialmente, as funções `configure_outputs`, `configure_inputs` e `configure_uart` preparam os pinos GPIO e a comunicação serial.
- O display LCD é inicializado com a função `LCD_init`.
- Os motores e o buzzer são desativados por padrão ao iniciar o sistema.
- A tarefa `task_alarm_monitor` é criada para monitorar continuamente o status do alarme.

- Um buffer é utilizado para receber dados da UART. Os comandos são convertidos para letras minúsculas para facilitar a comparação textual.
- O sistema reconhece comandos de ativação/desativação de alarme, acionamento de botões, abertura/fechamento de portões e controle de luzes.
- A correspondência com os ambientes é realizada por meio de um mapeamento (array de estruturas `room_map`), permitindo identificar qual ambiente foi mencionado.
- Para cada comando reconhecido, uma tarefa apropriada é criada com a função `xTaskCreate`.

3.3.2 Arquivo `Functions_Init.c`

Este arquivo agrupa funções responsáveis pela configuração inicial dos periféricos do sistema.

- A função `configure_outputs` configura os pinos de saída digital (GPIOs), incluindo os motores, as luzes, o *buzzer* e o portão pequeno. A configuração é realizada com a estrutura `gpio_config_t`, ativando somente o modo de saída, sem resistores de pull-up/down.
- A função `configure_inputs` define os pinos utilizados para leitura de sensores do sistema de alarme. Todos são configurados como entrada digital simples.
- A função `configure_uart` configura a UART1 com taxa de 115200 bps, 8 bits de dados, sem paridade e um bit de parada. Também define os pinos físicos da UART e instala o *driver* UART na fila de eventos.

3.3.3 Arquivo `Functions_Exec.c`

Este arquivo contém as funções executoras responsáveis pela manipulação direta de dispositivos (luzes, motores, portões e alarme), além do display LCD.

- **Controle de iluminação:**
  - `turn_on_light` e `turn_off_light` acionam ou desligam uma luz especificada por pino.
  - `light_invert` alterna o estado de uma luz com base em um código inteiro associado a cada ambiente.
- **Controle de portões:**
  - `open_gate` e `close_gate` controlam a direção dos motores, utilizando dois pinos para cada motor (AIN1/2 ou BIN1/2), com temporizações específicas para completar a movimentação.

  - O portão pequeno é ativado com um único pino digital.
  - `gate_invert` inverte o estado atual de um portão com base em seu código, oferecendo lógica de alternância simples.

- **Controle de alarme e display:**
  - `lcd_show_status` exibe mensagens no display I2C, dependendo do estado do alarme: desarmado, armado sem violação, ou armado com violação.
  - `trigger_alarm` verifica os sensores conectados às entradas do alarme. Caso um sensor esteja ativo e o alarme esteja armado, o *buzzer* é ativado e a violação é exibida no display.

3.3.4 Arquivo `Functions.h`

O arquivo `Functions.h` define os protótipos das funções e as constantes utilizadas no sistema, além das variáveis globais compartilhadas entre os arquivos. Ele estabelece a interface para o controle dos dispositivos e a comunicação com o hardware.

- São definidas constantes referentes aos pinos GPIO usados para motores, luzes, portão pequeno, *buzzer* e sensores do alarme.
- São declaradas variáveis globais que indicam o estado atual do alarme, ativação do alarme, estado dos motores, das luzes e do portão.
- Define o mapeamento dos ambientes para facilitar o reconhecimento e controle das luzes, por meio de um array de estruturas `room_map_t`, que associa os nomes dos ambientes aos seus respectivos pinos e estados.
- Contém constantes para a configuração da UART e do display LCD I2C.
- Declara protótipos das funções de configuração, controle dos dispositivos e tarefas FreeRTOS que implementam as funcionalidades do sistema.

Esses quatro arquivos organizam a lógica do sistema em camadas claras: inicialização, leitura de comandos, execução de ações e cabeçalho, promovendo modularidade e facilitando a manutenção e expansão do sistema de automação.

3.3.5 Client.py — Reconhecimento de Voz e Monitoramento GPIO

O arquivo `Client.py` é responsável por capturar o áudio do ambiente, identificar a palavra-chave de ativação (wake word), gravar a fala do usuário até a ocorrência de silêncio, transcrever os comandos utilizando a biblioteca VOSK e enviar os comandos reconhecidos ao

ESP32 via UART. Adicionalmente, monitora 10 entradas digitais ligadas a botões físicos e sensores, enviando os respectivos eventos ao ESP32.

As tecnologias e bibliotecas utilizadas foram:

- **VOSK:** biblioteca de reconhecimento de fala *offline*.
- **sounddevice:** usada para capturar áudio da entrada do microfone com baixa latência.
- **NumPy:** utilizada para cálculo do valor de RMS e detecção de silêncio.
- **serial (pySerial):** realiza comunicação UART com o ESP32 utilizando níveis elétricos TTL.
- **socket:** realiza comunicação local com o *script* de controle da fita de LED RGB.
- **RPi.GPIO:** realiza leitura dos pinos GPIO configurados como entradas digitais.

A lógica de funcionamento é dividida em quatro etapas principais:

1. **Aguardando Wake Word:** Inicia a captura contínua de áudio e reconhece parcialmente palavras usando o VOSK. Quando a palavra-chave (`"casa"`) é detectada, inicia-se a gravação de áudio.
2. **Gravação com Detecção de Silêncio:** Grava o áudio enquanto o RMS do sinal for superior a um limiar pré-estabelecido. O áudio é salvo em um arquivo `.wav`.
3. **Transcrição e Envio do Comando:** O áudio salvo é processado novamente com o VOSK para extrair palavras relevantes, com base em um conjunto limitado de comandos definidos (`COMMANDS`). O comando é enviado ao ESP32 via UART.
4. **Monitoramento GPIO:** Uma thread paralela monitora 10 pinos GPIO associados a botões ou sensores e envia comandos no formato textual para o ESP32, indicando eventos como abertura de portões, acionamento de zonas de alarme ou acionamento de luzes por botão.

Durante as etapas, mensagens de status são enviadas via socket local para o *script* `Server.py`, alterando o estado da fita de LED RGB. Essa divisão ocorre porque, devido a peculiaridades da biblioteca *sounddevice*, o código de reconhecimento de voz deve ser executado com o usuário padrão, enquanto o controle da fita de LED RGB requer permissões de superusuário. Dessa forma, não foi possível integrar todas as funcionalidades em um único código

3.3.6 Server.py — Feedback Visual com Fita LED WS2811

O arquivo `Server.py` implementa um servidor de socket local que recebe comandos de estado do reconhecimento de voz e altera dinamicamente as cores da fita de LED WS2811. Este módulo fornece um feedback visual intuitivo ao usuário sobre o estágio atual da interação por voz.

Foi utilizada a biblioteca `rpi_ws281x`, que permite controle direto da fita de LED endereçável a partir do GPIO do Raspberry Pi, com suporte a efeitos visuais customizados.

A lógica principal é composta por:

- **Estados definidos:**
  - Estado 1: Cor rosa fixa indica sistema aguardando wake word.
  - Estado 2: Cor azul fixa indica sistema aguardando fala após wake word.
  - Estado 3: Efeito de onda amarela (progressivo) indica transcrição em andamento.
- **Servidor socket:** Recebe comandos via TCP na porta 65432. Os comandos são enviados pelo *script* `Client.py`.
- **Gerenciamento de efeitos:** O efeito de onda é executado em uma thread separada e pode ser interrompido sempre que um novo comando de estado for recebido.

Esse feedback visual facilita a interação com o sistema em ambientes residenciais, especialmente em contextos de acessibilidade e usabilidade.

3.4 Desafios de Implementação do Modelo Elétrico Real

A transição do protótipo para a aplicação em ambiente real apresenta desafios relacionados à infraestrutura física, controle de dispositivos e escalabilidade do sistema.

O controle dos motores dos portões veiculares demanda adaptações para a aplicação em escala real. No protótipo, são utilizados motores DC de 12V e 60 rpm, acionados por meio do *driver* TB6612FNG. No entanto, em cenários reais, são utilizados motores convencionais de portão eletrônico, que apresentam maior robustez, torque elevado e já incorporam controladores dedicados. A integração com o sistema proposto dependerá das características de cada modelo, sendo possível, em alguns casos, o controle por sinais de baixa tensão, comunicação via Wi-Fi ou emissor de rádio frequência.

Ainda, a implementação de um sistema de automação residencial totalmente cabeado requer o uso de fiação para todos os sensores e atuadores distribuídos pela residência. Essa abordagem demanda um planejamento elétrico e estrutural detalhado e atenção às distâncias máximas entre os componentes, a fim de preservar a integridade dos sinais. Como alternativa, a utilização de comunicação sem fio elimina a necessidade de cabeamento físico, mas introduz

ajustes ao sistema, como a adoção de protocolos de comunicação sem fio para os sensores e atuadores.

A lógica de controle das luzes utilizada no protótipo se aproxima da aplicação em escala real. As fitas de LED de 12V e os LEDs de 5V representam as lâmpadas convencionais, sendo necessário apenas ajustar a tensão no terminal comum (COM) do relé para 127V ou outro valor compatível com o sistema elétrico da residência.

A utilização de múltiplos microfones simultâneos em diferentes ambientes pode apresentar interferência na captação e interpretação dos comandos de voz. A concorrência entre entradas de áudio pode exigir a implementação de mecanismos de priorização, fusão de sinais ou centralização do processamento em um servidor.

Além disso, a alimentação dos diferentes módulos do sistema requer fontes com tensões distintas (5V, 12V e 127V), tornando necessária a separação elétrica entre os circuitos de controle e potência, bem como o uso de conversores e proteções adequadas para garantir a estabilidade e a segurança do sistema elétrico.

4 Conclusão

Este trabalho teve como objetivo o desenvolvimento de um sistema de automação residencial de baixo custo, segurança, adaptabilidade e sustentabilidade. A proposta foi implementada por meio da integração entre um Raspberry Pi, responsável pelo reconhecimento de voz com a biblioteca VOSK, e um microcontrolador ESP32, encarregado de executar os comandos físicos relacionados ao controle de luzes, alarmes e portões.

A arquitetura do sistema foi planejada de forma a permitir a comunicação eficiente entre os dispositivos por meio de conexão serial (TX/RX), possibilitando respostas rápidas e confiáveis às instruções do usuário. O reconhecimento de comandos por voz sem a necessidade de conexão à Internet garante total privacidade ao usuário.

Durante a fase de implementação, foram utilizadas tecnologias acessíveis e componentes comumente disponíveis no mercado, o que favorece a replicabilidade do projeto. Além disso, o sistema foi desenvolvido de modo modular, facilitando futuras expansões e adaptações conforme as necessidades do usuário.

Os resultados obtidos demonstraram que é possível implementar soluções de automação residencial eficientes utilizando recursos mais baratos, sem abrir mão da confiabilidade e da experiência do usuário. O sistema foi capaz de interpretar corretamente os comandos de voz e realizar as ações desejadas de forma estável, respeitando as condições pré-programadas.

Como trabalho futuro, sugere-se a integração com sensores adicionais, como sensores de presença, temperatura e gás, além da criação de um aplicativo móvel que permita a visualização em tempo real do estado dos dispositivos e o controle remoto do sistema. Também se vislumbra a possibilidade de aplicar algoritmos de aprendizado de máquina para tornar a automação ainda mais adaptável ao comportamento dos usuários.

Ademais, embora o foco principal do projeto seja o processamento local e a independência de serviços externos, a arquitetura proposta permite adaptações para usuários que priorizam a integração com dispositivos Wi-Fi ou a utilização de aplicativos móveis. Nesse contexto, é possível manter o processamento de comandos de voz de forma local, garantindo segurança e privacidade, ao mesmo tempo em que se expande a usabilidade do sistema por meio de interfaces conectadas. Tal flexibilidade reforça o caráter personalizável da solução desenvolvida, ampliando seu potencial de aplicação em diferentes cenários residenciais.

Conclui-se, portanto, que o sistema proposto representa uma alternativa viável, de baixo custo, segura, acessível e personalizável para automação residencial, sendo especialmente útil em contextos que demandam soluções econômicas e prezam pela privacidade de seus dados.

Referências

Alpha Cephei. *Vosk – Offline Speech Recognition API*. 2025. Disponível em: <https://alphacephei.com/vosk/>. 33

BARBOSA, A. *Low Cost Smart Home*. 2025. <https://github.com/Anak-Barbosa/lowCost-smart-home/tree/main>. 46

BLOG DA ROBÓTICA. *Como utilizar botão push button como interruptor para ligar LED com Arduino*. 2022. <https://www.blogdarobotica.com/2022/10/18/como-utilizar-botao-push-button-para-ligar-led-com-arduino/>. 23

CANAL OFICINA GARAGEM. *Como Fazer um Portão de Correr (Deslizante) Parte 01 - Curso de Serralheiro 82*. 2025. <https://www.youtube.com/watch?v=8J9MTJ394WY>. 10

DANTAS, R. A. *Eletroímã*. 2025. <https://brasilescola.uol.com.br/fisica/eletroima>. 25

ELETROGATE. *Display LCD 16x2 com Backlight Azul e I2C*. 2025. <https://www.eletrogate.com/display-lcd-16x2-i2c-backlight-azul>. 22

ELETROGATE. *Módulo Relé 4 Canais 3V 10A com Borne KRE para ESP32*. 2025. <https://www.eletrogate.com/modulo-rele-4-canais-3v-10a-com-borne-kre-para-esp32>. 16

ELETROGATE. *Resistor 10K 1/4W*. 2025. <https://www.eletrogate.com/resistor-10k-1-4w-10-unidades>. 21

Espressif Systems. *ESP-IDF: The official development framework for the ESP32 chip*. 2025. Disponível em: <https://docs.espressif.com/projects/esp-idf>. 33

Grand View Research. *Global Smart Home Automation Market Size & Outlook*. 2023. <https://www.grandviewresearch.com/horizon/outlook/smart-home-automation-market-size/global>. 1

H'MASTON. *Kit 2 Microfone Duplo Profissional De Lapela Tipo C Anti Ruído MK08*. 2025. <https://www.hmaston.com.br>. 22

Lighting Philips. *Fita LED Philips 15.5W 5 metros 24V 6500K 1600lm/m*. 2025. <https://www.loja.lighting.philips.com/fita-led-philips-corepro-15-5w-5-metros-24v-6500k-1600lm-929002651312/p>. 13

Magnet Schultz. *Ímã eletrônico DC 12V–24V*. 2025. <https://www.magnet-schultz.coml>. 25

MOISES GRACIANO DA SILVA. *Kit Motor Portão Eletrônico Deslizante Kdz Speed Garen 1/3hp*. 2025. <https://www.youtube.com/watch?v=RM8o42xTkCk>. 10

Pololu. *Micro Metal Gearmotor HPCB 12V*. 2025. <https://www.pololu.com/product/3039>. 20

Raspberry Pi Foundation. *Raspberry Pi 4 Model B – Technical Specifications*. 2025. Disponível em: <https://www.raspberrypi.com/products/raspberry-pi-4-model-b/>. 29, 30

Raspberry Pi Foundation. *Raspberry Pi Imager*. 2025. Disponível em: <https://www.raspberrypi.com/software/>. 33

RoboCore. *Fita LED RGB Endereçável*. 2025. <https://www.robocore.net/display-led/fita-de-led-rgb-ws2812-5050-1m>. 14

RoboCore. *LED 5mm Amarelo*. 2025. <https://www.robocore.net/display-led/led-difuso-5mm-amarelo-10-unidades>. 13

Selfcrafts. *How to Joint Timing Belt*. 2023. Disponível em: <https://youtube.com/shorts/dxuzylmGaqE?si=QQe5UyRqDGDRBAlc>. 35

SMART KITS. *Chave Botão com Trava e LED R16-503AD Vermelho*. 2025. <https://www.smartkits.com.br/chave-botao-com-trava-e-led-r16-503ad-vermelho>. 24

STMicroelectronics. *L7805CV*. 2025. <https://estore.st.com/en/l7805cv-cpn.html>. 17

TDK Product Center. *Piezoelectric Buzzers*. 2025. <https://product.tdk.com/en/products/sw_piezo/sw_piezo/piezo-buzzer/index.html>. 20

TEC DICAS. *Desenvolvendo um módulo relé 5V para sistemas embarcados*. 2025. Disponível em: <https://tecdicas.com/desenvolvendo-um-modulo-rele-5v-para-sistemas-embarcados/>. 15

Toshiba. *TB6612FNG Motor Driver - Datasheet*. 2025. Disponível em: <https://www.alldatasheet.com/datasheet-pdf/view/807693/TOSHIBA/TB6612FNG.html>. 17

USINAINFO. *Driver TB6612FNG Duplo Ponte H de Motor DC ou Passo*. 2025. <https://www.usinainfo.com.br/driver-para-motor/driver-tb6612fng-duplo-ponte-h-de-motor-dc-ou-passo-4697.html>. 18

USINAINFO. *Reed Switch Chave Magnética Ampola Reed Dourada 14x2mm*. 2025. <https://www.usinainfo.com.br/componentes-diversos/reed-switch-chave-magnetica-ampola-reed-dourada-14x2mm-3483.html>. 25

World View. *Sistema Microfone Lapela Duplo H'Maston MK08 Sem Fio USB-C 360° para Smartphone Android*. 2025. Disponível em: <https://www.worldview.com.br/sistema-microfone-lapela-duplo-hmaston-mk08-sem-fio-usb-c-360-para-smartphone-android/p>. 22

A Tutoriais para o Raspberry Pi

A.1 Tutorial de Preparação e Conexão do Raspberry Pi

A gravação da imagem do sistema operacional no cartão micro SD do Raspberry Pi pode ser realizada de forma eficiente por meio da ferramenta Raspberry Pi Imager. Além de facilitar a escolha do sistema, essa ferramenta permite que configurações básicas sejam feitas antes da gravação, como a habilitação do SSH e a adição de uma chave pública de acesso.

A.1.1 Geração de chave criptográfica SSH no Ubuntu

Antes da utilização do Raspberry Pi Imager, recomenda-se que uma chave SSH seja gerada no sistema Ubuntu, caso ainda não exista. A autenticação será feita por criptografia assimétrica, eliminando a necessidade do uso de senhas.

Esse método é eficiente pois, ao se configurar o login e a senha diretamente no Raspberry Pi Imager, é comum que o sistema substitua a senha definida por uma senha forte e aleatória, gerada automaticamente. Essa substituição pode impedir o acesso ao Raspberry Pi, uma vez que a nova senha não será conhecida. Assim, ao utilizar autenticação por chave pública, esse problema pode ser evitado, garantindo o acesso ao dispositivo.

Para que a chave seja gerada, o seguinte comando deve ser executado no terminal:

```
ssh-keygen
```

Durante o processo, a tecla `Enter` deve ser pressionada para aceitar os caminhos padrão, ou um nome específico pode ser definido para o arquivo, se desejado. Após a geração, uma imagem gráfica representando a chave (conhecida como *randomart*) será exibida. A imagem será semelhante ao exemplo abaixo:

```
The key's randomart image is:
+---[RSA 2048]----+
|   o. .+oo       |
|  . +o+..        |
| . + Eo+         |
|  o.ooo..        |
|  .oo++ S        |
|  . +*o *        |
|   o=O o .       |
```

```
|   +B@o          |
|  .=@*+.         |
+----[SHA256]-----+
```

O nome do arquivo contendo a chave pública será informado nas primeiras linhas do comando, geralmente como:

```
Generating public/private ChaveXX key pair.
Enter file in which to save the key (/home/usuario/.ssh/id_ChaveXX):
```

Para confirmação do nome, os arquivos do diretório de chaves devem ser listados com o comando:

```
ls ~/.ssh
```

É preciso procurar por um arquivo com a extensão `.pub` cujo nome corresponda ao definido e citado nas primeiras linhas da geração. Para visualização do conteúdo da chave pública, o comando abaixo deve ser utilizado:

```
cat ~/.ssh/id_rsa.pub
```

O conteúdo será semelhante ao exemplo a seguir:

```
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABAQC3vJc1x3zN9zQ5w1a9kF7Gp0O+
S1Pq5r7K8BvYr1X1pZxA5V7FvSshEXAMPLE user@meudispositivo
```

A chave pública deve ser copiada para uso posterior no Raspberry Pi Imager.

A.1.2 Instalação do Raspberry Pi Imager no Ubuntu

Para que a instalação do Raspberry Pi Imager seja realizada no Ubuntu, o Snap deve ser utilizado:

```
sudo snap install rpi-imager
```

Após a instalação, o programa pode ser executado com o comando:

```
rpi-imager
```

A.1.3 Configuração da imagem com autenticação por chave pública

Ao abrir o Raspberry Pi Imager, as seguintes etapas devem ser seguidas:

1. Clicar em **Choose Device** e selecionar o modelo correspondente do Raspberry Pi.
2. Clicar em **Choose OS** e selecionar o sistema operacional desejado. Para a versão **Lite** (sem interface gráfica), deve-se selecionar `Raspberry Pi OS (other)`.
3. Clicar em **Choose Storage** e selecionar o cartão SD que será formatado (atenção: todos os dados da unidade serão apagados).
4. Clicar em **Next**. Uma janela intitulada *Use OS customization?* será exibida. A opção **Edit Settings** deve ser marcada.

Na aba **Services**, o serviço **Enable SSH** deve ser ativado e a opção **Allow public-key authentication only** deve ser selecionada. A chave pública gerada anteriormente deve ser colada no campo exibido.

Na aba **General**, recomenda-se que a opção **Set username and password** seja desmarcada, visto que a autenticação será feita exclusivamente via chave SSH. Nessa aba, também podem ser configurados o nome da rede Wi-Fi (SSID) e a respectiva senha. No entanto, ressalta-se que pode ocorrer a sobrescrição da senha, o que impedirá o acesso ao sistema. Por esse motivo, recomenda-se o uso de cabo Ethernet na primeira inicialização do dispositivo.

Na aba **Options**, recomenda-se marcar a opção **Eject media when finished**, garantindo que o cartão SD seja ejetado com segurança após a gravação da imagem.

Após a conclusão do processo e a ejeção do cartão, o cartão deve ser removido do computador, inserido no Raspberry Pi e o dispositivo deve ser alimentado para que a inicialização seja realizada.

A.1.4 Identificação e Acesso Remoto ao Raspberry Pi

Após a gravação da imagem no cartão SD e sua inserção no Raspberry Pi, o endereço IP atribuído ao dispositivo deve ser identificado para que a conexão remota via SSH possa ser realizada.

O Raspberry Pi deve ser conectado à rede local por meio de um cabo Ethernet (LAN) e deve-se aguardar de 1 a 3 minutos para que o processo de inicialização (boot) seja concluído. Enquanto o boot ocorre, é útil iniciar a instalação do nmap, caso ainda não exista na máquina:

```
sudo apt install nmap -y
```

Após a verificação de senha do comando sudo e o fim da instalação, o comando abaixo deve ser utilizado no terminal do Ubuntu para a verificação dos dispositivos conectados à rede:

```
nmap -sn IP_Roteador/24
```

Esse comando realiza uma varredura na faixa de IP da rede local e exibe os hosts ativos. A execução deve ser repetida até que um novo endereço IP, provavelmente do Raspberry Pi, seja exibido.

Após a identificação do IP, o comando abaixo deve ser utilizado para estabelecer a conexão SSH:

```
ssh pi@IP_Raspberry
```

O endereço encontrado deve substituir `IP_Raspberry`. Caso a seguinte mensagem seja exibida:

```
ssh: connect to host IP_Digitado port XX: Connection refused
```

Isso indica que a porta acessada não corresponde à do SSH e que o IP provavelmente não pertence ao Raspberry Pi. Outros endereços devem ser testados. Se todos apresentarem a mesma mensagem, pode-se concluir que a placa não está conectada à rede.

Ao usar o IP certo, caso a autenticação por chave pública tenha sido configurada no Raspberry Pi Imager, a conexão será feita automaticamente, sem solicitação de senha. No entanto, será exibida a seguinte mensagem:

```
Are you sure you want to continue connecting (yes/no/[fingerprint])?
```

Deve-se digitar `yes` e pressionar `Enter`. Após isso, a conexão será estabelecida, desde que o procedimento tenha sido seguido corretamente. O prompt será alterado para:

```
pi@raspberrypi:~$
```

A.1.5 Solução para conflitos com a chave conhecida

Caso uma conexão anterior com o mesmo endereço IP tenha sido realizada utilizando uma chave diferente, um erro de conflito pode ser apresentado pelo SSH:

```
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
@    WARNING: REMOTE HOST IDENTIFICATION HAS CHANGED!     @
```

```
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
IT IS POSSIBLE THAT SOMEONE IS DOING SOMETHING NASTY!
Someone could be eavesdropping on you right now (man-in-the-middle attack)!
It is also possible that a host key has just been changed.
```

Para que o problema seja resolvido, o seguinte comando deve ser executado:

```
ssh-keygen -f "/home/usuário/.ssh/known_hosts" -R "IP_Raspberry"
```

O nome de usuário do sistema deve substituir `usuário` e o endereço IP do dispositivo deve substituir `IP_Raspberry`.

A.2 Configuração do Ambiente Python no Linux

Para executar o código em python em um sistema operacional Linux, como o Ubuntu, siga os passos.

Instale o python 3.12 ou superior ao 3.9, pela compatibilidade com o VOSK:

```
sudo apt update
sudo apt upgrade

sudo apt install python3.12 -y
```

Verifique a versão:

```
python3 --version
```

Instale o pip:

```
sudo apt install -y python3-pip
```

Verifique a versão:

```
pip3 --version
```

Crie um ambiente de trabalho substituindo o nome desejado em **myenv**:

```
python3.12 -m venv myenv
```

Ative o ambiente:

```
source myenv/bin/activate
```

Para encerrar o ambiente basta:

```
deactivate
```

A.3 Preparação do Raspberry Pi para Execução do Sistema

Ao inicializar o sistema pela primeira vez, é recomendada a execução dos seguintes comandos:

```
sudo apt update
sudo apt full-upgrade -y
sudo reboot
```

Após o reinício, a conexão via SSH deverá ser restabelecida manualmente. Com o Python 3.12 instalado, conforme descrito anteriormente, é necessário ativar o ambiente virtual previamente criado:

```
source myenv/bin/activate
```

Em seguida, devem ser instaladas as seguintes dependências:

```
pip install vosk
pip install sounddevice
pip install numpy
pip install RPi.GPIO
pip install pyserial
```

Também se faz necessário o download do modelo VOSK, que pode ser obtido por meio do seguinte endereço eletrônico:

```
https://alphacephei.com/vosk/models
```

Para este projeto, foi selecionado o modelo **vosk-model-small-pt-0.3**, por ser leve e compatível com o idioma português, atendendo aos requisitos de desempenho do sistema.

A.4 Comandos de Execução do Sistema

Durante a fase de desenvolvimento, os programas devem ser executados manualmente, uma vez que apenas a versão final (ou funcional) deve ser configurada para iniciar junto com o sistema. Abaixo são apresentados os comandos utilizados no Linux para executar cada parte do programa.

A.4.1 Cliente - Sistema de Reconhecimento de Voz e Leitura dos Botões no Raspberry Pi

Após alimentar o sistema elétrico e estabelecer a comunicação SSH com o Raspberry Pi, deve-se acessar o diretório onde o código-fonte está localizado, se aplicável:

```
cd pasta
```

Em seguida, é necessário ativar o ambiente virtual no qual o projeto foi configurado:

```
source myenv/bin/activate
```

A.4.2 Servidor - Sistema de Controle da Fita de LED RGB Endereçável

Por exigência da biblioteca **rpi_ws281x.h**, que requer acesso direto ao hardware para controlar o sinal PWM da fita LED RGB, é necessário executar o código com permissões de superusuário. Isso é feito utilizando o comando sudo, conforme o exemplo abaixo:

```
sudo ./myenv/bin/python3 arquivoRGB.py
```

Substitua **myenv** pelo nome do ambiente virtual no qual os pré-requisitos foram previamente instalados e **arquivoRGB** pelo nome do arquivo responsável pelo controle da fita RGB.

B Tutoriais para o ESP32

B.1 Instalação do ESP-IDF e Criação de um Projeto

Dentro do ambiente, inicie a instalação do ESP-IDF rodando o comando:

```
sudo apt-get install \
    git wget flex bison gperf python3 python3-pip python3-venv \
    cmake ninja-build ccache libffi-dev libssl-dev \
    dfu-util libusb-1.0-0
```

Clone o repositório usando os comando:

```
mkdir -p ~/esp
cd ~/esp
git clone --recursive https://github.com/espressif/esp-idf.git

cd ~/esp/esp-idf
./install.sh esp32
```

Substitua **Pasta_do_projeto** pelo nome da pasta que reúne os códigos do sistema:

```
cd Pasta_do_projeto
mkdir components
cd components
git clone https://github.com/maxsydney/ESP32-HD44780.git
```

B.2 Ativação do Ambiente ESP-IDF, Criação de um Novo Projeto e Gravação do Firmware na Placa

Para que as ferramentas do ESP-IDF fiquem acessíveis no terminal, é necessário ativar o ambiente executando o seguinte script:

```
. $HOME/esp/esp-idf/export.sh
```

Esse comando deve ser executado sempre que um novo terminal for aberto, a menos que seja adicionado ao arquivo de inicialização do shell (`.bashrc`, `.zshrc`, etc.).

B.2.1 Verificação da Ativação

Após ativar o ambiente, recomenda-se verificar se a configuração foi bem-sucedida com o comando:

```
idf.py --version
```

Se o ambiente estiver corretamente configurado, a versão do ESP-IDF instalada será exibida.

B.2.2 Criação de um Novo Projeto

A criação de um novo projeto pode ser realizada com o seguinte comando:

```
idf.py create-project Nome_Arquivo
```

Esse comando gera a estrutura inicial do projeto, contendo arquivos básicos como `main.c`, `CMakeLists.txt` e diretórios de build.

B.2.3 Acesso ao Diretório do Projeto

Após a criação, é necessário acessar o diretório do projeto para continuar o desenvolvimento:

```
cd Nome_Arquivo
```

B.2.4 Definição do Modelo de Placa

Antes da compilação, é obrigatório especificar o microcontrolador alvo. Para placas ESP32, utiliza-se:

```
idf.py set-target esp32
```

Esse comando configura a plataforma alvo para a geração do firmware.

B.2.5 Compilação do Projeto

Com o ambiente ativo e o microcontrolador definido, a compilação pode ser feita com:

```
idf.py build
```

Esse comando processa os arquivos do projeto, realiza a compilação e gera os arquivos binários necessários à gravação.

B.2.6 Gravação do Firmware

Para gravar o firmware na placa ESP32, o dispositivo deve estar conectado via USB. A seguir, utiliza-se:

```
idf.py flash -p /dev/ttyUSB0
```

O argumento `-p` indica a porta serial onde a placa está conectada. Em alguns sistemas, essa porta pode variar (por exemplo, `/dev/ttyUSB1` ou `/dev/ttyS0`).

B.2.7 Abertura do Monitor Serial

Após a gravação, é possível visualizar a saída serial da aplicação utilizando o monitor integrado:

```
idf.py monitor
```

Para encerrar o monitor, deve-se pressionar `Ctrl+]`. Também é possível realizar a gravação e abrir o monitor em um único comando:

```
idf.py -p /dev/ttyUSB0 flash monitor
```

Esse processo é útil para testes rápidos durante o desenvolvimento do firmware.