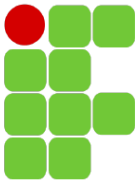


Instituto Federal de Educação, Ciência e Tecnologia do Ceará

Campus Fortaleza

EmbarcaTech

Bruno Lemoel Costa Batista

Smart Clock: Alarme inteligente para otimização de atividades

Fortaleza (CE) - 2025

Bruno Lemoel Costa Batista

Smart Clock: Alarme inteligente para otimização de atividades

Projeto desenvolvido para conclusão do curso apresentado ao EmbarcaTech do Instituto Federal do Ceara como requisito para obtenção de certificação em sistemas embarcados

Fortaleza (CE) - 2025

RESUMO

Nos últimos tempos tem sido essencial a criação de tecnologias para otimização de tarefas do usuário em diversas áreas como domesticas, profissionais e entre outras, porem principal causa para não execução das atividades rotineiras podemos destacar os atrasos e esquecimentos. Estudos revelam que atrasos podem instigar a dificuldade de relacionamentos pessoais e profissionais e indivíduos com déficit em gestão de tempo são mais propensos a se atrasar, no que pode acabar resultando em sentimento de culpa, estresse e ansiedade. Com o Smart Alarm o objetivo desse projeto é garantir a execução da atividade do usuário e auxilia-lo de forma autônoma para visando assim a eficiência e otimização máxima da tarefa para que seja executada no tempo ideal ou caso deseje somente alertar ao usuário de forma simples e objetiva da atividade programada.

Palavras chave: Atrasos. Smart Alarm. atividade.

Sumário

1 – Introdução

A evolução da tecnologia ao longo das décadas transformou profundamente a maneira como as pessoas executam suas atividades diárias. Desde a invenção de ferramentas básicas até a era da inteligência artificial e da Internet das Coisas (IoT), a tecnologia tem sido um motor de eficiência, permitindo que os usuários realizem tarefas de forma mais rápida, precisa e com menos esforço. Nos primórdios do avanço tecnológico para otimização de rotina do usuário podemos citar a revolução digital e a automação iniciada na metade do século XX, no que trouxe consigo a automação de processos que antes demandavam tempo e esforço humano. Computadores pessoais, softwares de produtividade e sistemas operacionais simplificaram tarefas como redação de documentos, cálculos complexos e organização de informações. Programas como o Microsoft Office e ferramentas de edição de texto eliminaram a necessidade de máquinas de escrever, acelerando a produção de conteúdo.

Estudos sobre atrasos como "Procrastination and Tardiness: A Behavioral Analysis" (Psychological Reports, 2018) relatam a dificuldade com relação a procrastinação e atrasos, onde os resultados indicam que indivíduos que procrastinam têm maior probabilidade de se atrasar para compromissos, o que pode ser um reflexo de dificuldades em priorizar tarefas e gerenciar prazos. Para evitar problemas com relação ao atraso de pessoas a suas atividades o Smart Clock foi inventado como forma melhorada do alarme convencional no qual apenas verifica a hora atual e compara com horário cadastrado pelo usuário e emite um alarme ou lembrete, porém, nesse projeto iremos apresenta algo que vai além do convencional que consiste em alertar ao usuário de sua atividade é a capacidade de se comunicar com outros dispositivos através de um modulo relé, e se caso necessário ira ligar um ambiente de trabalho ou aparelhos específicos no local, em casa ou etc... mas ainda sim não somente isso, incluímos no projeto funções como a conexão a rede wi-fi de preferência do usuário, informe de temperatura atual do ambiente para o usuário. Atuando com interface de forma simples e objetiva, disponibiliza ao usuário espaço onde podemos informar nossa atividade de forma escrita e em seguida informar data e hora para que o alarme seja ativado e a descrição da sua atividade, ao ser acionado o alarme no dispositivo ele irá ler a atividade descrita pelo usuário e identificar se ira ser utilizado um ambiente de trabalho ou aparelho especifico e se caso estiverem desligados ira ligá-los com sinal que será emitido ao módulo relé instalado na tomada e emitira um sinal sonoro e visual com leds e mensagens no próprio alarme para o usuário afim de alertar sobre sua atividade e auxiliar na execução.

De forma cronológica o alarme ao ser ligado ira verificar a conexão com a rede wi-fi, emitira temperatura do ambiente e informar a data e hora atual, após as principais funções ira ser permitido ao usuário o fornecimento da descrição da sua atividade e horário que deseja programar, constantemente o alarme irá comparar o horário fornecido pelo usuário com o horário atual, se os dois horários forem iguais o alarme será acionado, com o alarme acionado ele irá ler a descrição da atividade e se necessário irá se comunicar com outros aparelhos no local através do módulo relé para liga-los ou desliga-los e no mesmo momento

emitira um alerta com sinal sonoro e visual através de luzes e enviará uma mensagem ao usuário para que seja iniciada a atividade programada, também disponibilizara de um botão para que seja interrompido o alarme.

O alarme inteligente proposto combina várias funcionalidades essenciais em um único dispositivo, essa integração de funcionalidades torna o dispositivo versátil e capaz de atender a múltiplas necessidades do usuário, desde segurança até automação residencial. Além disso, a praticidade de ter um sistema que monitora temperatura, data e hora em um único dispositivo elimina a necessidade de múltiplos equipamentos, reduzindo custos e simplificando a instalação. A escolha de um sistema embarcado é fundamental para o sucesso desse projeto para serem altamente eficientes, consumindo pouca energia. Isso é crucial para dispositivos que precisam operar continuamente, como um alarme inteligente, sistema embarcado garante que o dispositivo seja compacto, de baixo custo e altamente eficiente, atendendo às demandas de um mundo cada vez mais conectado e automatizado.

No mercado atual que também utilizam sistemas embarcados e tecnologias IoT para oferecer funcionalidades de segurança, monitoramento e automação já temos projetos similares que servem de referência para o alarme inteligente como o primeiro projeto, Home Automation System com Raspberry Pi, este projeto utiliza uma Raspberry Pi como controlador central para automação residencial, incluindo funcionalidades de segurança, controle de iluminação e monitoramento ambiental. Funcionalidades, Alarme de Segurança: Utiliza sensores de porta/janela e sensores de movimento para detectar intrusões, Controle de Iluminação: Permite ligar/desligar luzes remotamente ou com base em horários pré-definidos, Monitoramento Ambiental: Mede temperatura, umidade e qualidade do ar utilizando sensores como DHT22 e MQ-135, Interface Web: Oferece uma interface web para controle e monitoramento do sistema. O segundo projeto, Smart Security System with IoT Integration é um sistema de segurança inteligente que combina hardware (sensores e atuadores) com software (conectividade Wi-Fi e plataformas IoT) para monitorar um ambiente, detectar eventos de segurança (como intrusão ou incêndio) e permitir o controle remoto de dispositivos. Ele é altamente personalizável e pode ser adaptado para residências, escritórios ou até mesmo pequenas indústrias.

2 - Objetivos

2.2 – Objetivo Geral

Desenvolver um alarme inteligente baseado no microcontrolador Raspberry Pi Pico W é criar um sistema integrado, eficiente e acessível que combine praticidade, monitoramento ambiental e automação residencial em um único dispositivo. Esse projeto visa oferecer uma solução moderna e personalizável para melhorar a qualidade de vida do usuário, garantindo conveniência, praticidade e controle sobre o ambiente em que ele está inserido.

2.3 – Objetivo Específicos

I. Baixo custo no que torna o equipamento acessível;
II. Comunicação com dispositivos;
III. Interação pratica e direta com atividades do usuário;

3 – Fundamentos Teóricos

Nessa parte será apresentando o conceito teórico sobre dispositivos I/O, wi-fi, sensores, módulo relé, e sistema embarcado para explicação do projeto que são necessários para esse projeto.

3.1 – Dispositivos I/O

Os dispositivos de entrada (input) e saída (output), conhecidos como I/O, são componentes essenciais em qualquer sistema embarcado, como no projeto de alarme inteligente. Eles permitem a interação entre o microcontrolador (Raspberry Pi Pico W) e o ambiente externo, possibilitando que o sistema receba informações (input) e execute ações (output). Abaixo, detalhamos a importância e a descrição dos dispositivos I/O utilizados no meu projeto: LED, buzzer e botão.

3.1.1 - LED (Dispositivo de Saída - Output)

O LED (Light Emitting Diode) é um componente eletrônico que emite luz quando uma corrente elétrica passa por ele. No seu projeto, o LED é utilizado como um indicador visual. O LED pode ser usado para mostrar o status do alarme (ligado, desligado, disparado).

3.1.2 - Buzzer (Dispositivo de Saída - Output)

O buzzer é um dispositivo eletrônico que emite sons quando é acionado. Ele pode gerar tons contínuos ou intermitentes, dependendo da aplicação. Emite um som alto e perceptível quando o alarme é acionado, O som é uma forma poderosa de chamar a atenção, especialmente em situações críticas. Enquanto o LED fornece feedback visual, o buzzer oferece feedback auditivo, aumentando a eficácia do sistema.

3.1.3 - Botão (Dispositivo de Entrada - Input)

O botão é um componente de entrada que permite ao usuário interagir com o sistema. Quando pressionado, ele fecha a execução do circuito, enviando um sinal ao microcontrolador. O botão pode ser usado para parar o alarme (desligar o buzzer e o LED) após ser acionado. Proporciona uma forma simples e direta de controle manual, essencial para situações em que o usuário precisa intervir rapidamente.

3.2 – Wi-Fi

O Wi-Fi no Raspberry Pi Pico W é implementado através do chip wireless integrado, que suporta padrões IEEE 802.11 b/g/n. Ele opera na frequência de 2.4 GHz e oferece uma conexão estável e confiável para comunicação com a rede local e internet. No meu projeto, o Wi-Fi será responsável por: Enviar e receber dados: Como notificações, comandos de controle, e informações de temperatura, data e hora.

3.3 – Sensores

3.3.1 - RTC (Real-Time Clock)

O RTC é um circuito integrado que mantém o controle preciso da data e hora, mesmo quando o dispositivo principal (como o Raspberry Pi Pico W) está desligado ou sem energia. Ele possui uma bateria de backup (geralmente uma bateria de célula de moeda) que permite que ele continue funcionando independentemente do estado do sistema principal.

3.3.2 - Sensor de Temperatura

Um sensor de temperatura é um dispositivo que mede a temperatura do ambiente, O sensor de temperatura permite que o alarme monitore a temperatura do ambiente em tempo real, o que pode ser útil para detectar condições anormais, como superaquecimento ou incêndios.

3.4 – Módulo Relé

O módulo relé é um dispositivo eletrônico que funciona como um interruptor controlado eletricamente. Ele permite que um circuito de baixa potência (como o do Raspberry Pi Pico W) controle um circuito de alta potência (como lâmpadas, motores, ou outros dispositivos elétricos). O relé age como uma ponte entre os dois circuitos, isolando eletricamente o controlador (microcontrolador) dos dispositivos de alta potência. O relé possui uma bobina que, quando energizada, gera um campo magnético que move um contato mecânico, fechando ou abrindo o circuito de alta potência. No meu projeto de alarme inteligente, o módulo relé é essencial para permitir que o Raspberry Pi Pico W controle dispositivos externos de forma segura e eficiente

3.5 – Sistema Embarcado

Um sistema embarcado é um sistema computacional projetado para executar funções específicas dentro de um dispositivo maior. Diferente de um computador de propósito geral, como um PC ou notebook, um sistema embarcado é otimizado para uma tarefa específica, com hardware e software integrados para atender a requisitos de desempenho, consumo de energia, tamanho e custo. Projetado para realizar uma ou poucas tarefas específicas, como controlar um alarme, monitorar sensores ou gerenciar a comunicação entre dispositivos e interage diretamente com sensores, atuadores e outros dispositivos físicos, coletando dados e controlando processos. No meu projeto de alarme inteligente, o uso de um sistema embarcado é fundamental para garantir que o dispositivo seja eficiente, confiável e capaz de executar suas funções específicas de forma autônoma com baixo custo e consumo de energia, ao mesmo tempo em que oferece flexibilidade para personalização e expansão.

3.4 – Projetos Similares

3.4.1 - Home Automation System com Raspberry Pi

Este projeto geralmente envolve o uso de um Raspberry Pi (ou similar) para controlar dispositivos domésticos, como luzes, ventiladores, sensores de temperatura, câmeras de segurança, etc. Ele se conecta à internet via Wi-Fi e permite que o usuário controle esses dispositivos remotamente por meio de um aplicativo ou interface web.

Características principais:

I. Comunicação Wi-Fi: O Raspberry Pi se conecta à rede Wi-Fi para enviar e receber dados,
II. Controle de dispositivos: Ativa/desativa dispositivos com base em entradas do usuário ou condições pré-definidas (como horários ou sensores).
III. Interface de usuário: Uma interface web ou app móvel para interação.
IV. Integração com sensores: Sensores de temperatura, umidade, movimento, etc., são usados para automatizar decisões.

Relação com seu projeto:

I. Comunicação Wi-Fi: Assim como no seu projeto, o Raspberry Pi Pico W usa Wi-Fi para se comunicar com outros dispositivos e a internet.
II. Controle de dispositivos: Seu alarme inteligente pode ativar outros dispositivos (como luzes ou sirenes) com base em entradas do usuário ou condições do ambiente.
III. Sensores: Você está usando um sensor de temperatura, assim como em sistemas de automação residencial.

IV. Interface de usuário: Pode ser útil criar uma interface web ou app para que o usuário configure o alarme e visualize dados como temperatura, data e hora.

3.4.2 - Smart Security System with IoT Integration

Este projeto é um sistema de segurança inteligente que usa sensores para detectar intrusões (como movimento ou abertura de portas/janelas) e monitora condições ambientais (como temperatura e umidade). Ele se conecta à internet via Wi-Fi para enviar alertas em tempo real para o usuário e permite o controle remoto de dispositivos, como sirenes, luzes ou câmeras.

Componentes principais:

I. Microcontrolador: Raspberry Pi Pico W, NodeMCU (ESP8266) ou ESP32.
II. Sensores:
Sensor de movimento PIR.
Sensor magnético para portas/janelas.
Sensor de temperatura e umidade (DHT11/DHT22 ou BME280).
III. Atuadores:
Sirene ou buzzer.
Relé para controlar luzes ou outros dispositivos.
IV. Conectividade: Wi-Fi para enviar alertas e permitir controle remoto.
V. Plataforma IoT: Blynk, ThingSpeak, MQTT ou Firebase.
VI. Interface de usuário: Aplicativo móvel ou dashboard web.

Funcionalidades:

I. Detecção de intrusão e envio de alertas em tempo real.
II. Monitoramento de temperatura e umidade.
III. Controle remoto de dispositivos (ligar/desligar sirene, luzes, etc.).
IV. Registro de eventos em um banco de dados na nuvem.
V. Integração com outros dispositivos IoT.

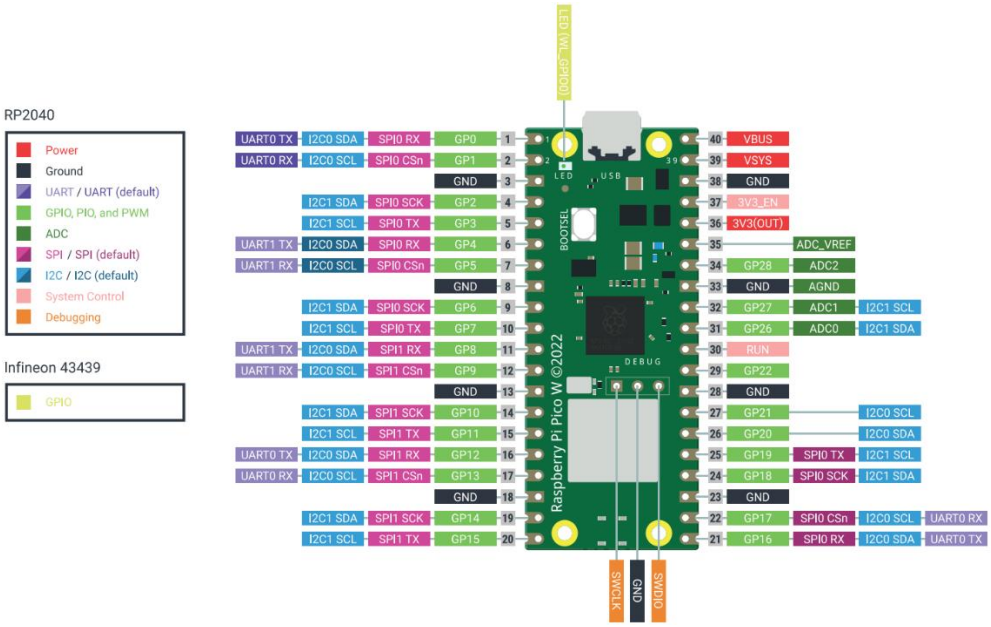
Bastante similar ao meu projeto, combinando funcionalidades de segurança, monitoramento ambiental e controle remoto de dispositivos. Ele pode servir como uma excelente referência para expandir seu alarme inteligente, adicionando sensores de movimento, integração com IoT e uma interface de usuário mais robusta.

4 – Metodologia

A criação do alarme inteligente tem como objetivo promover a interação com o ambiente de forma eficiente, com baixo custo e consumo de energia reduzido. O sistema é baseado em um sistema embarcado, que permite a execução de tarefas específicas de maneira autônoma e confiável. A escolha por um sistema embarcado se justifica pela sua capacidade de integrar hardware e software de forma otimizada, garantindo desempenho, eficiência energética e custo-benefício. O funcionamento do alarme inteligente consiste na programação de uma tarefa e seu horário de execução. Quando a data e hora atual coincidem com o horário programado, o sistema embarcado aciona o alarme, gerando um sinal de alerta para o usuário. Além disso, o sistema permite a interação com outros dispositivos por meio de um módulo relé, que pode ligar ou desligar equipamentos de acordo com a programação definida. Um sistema embarcado no desenvolvimento do alarme inteligente não apenas garante eficiência e confiabilidade, mas também abre portas para a integração com outras tecnologias, tornando-o uma solução versátil e adaptável às necessidades modernas. Na pratica realizamos toda programação no ambiente virtual Wokwi que nos permitiu interligar todo circuito de componentes a placa Raspberry Pi Pico W em linguagem C, tornando possível a visualização sobre como foi planejado o funcionamento do alarme inteligente, permitindo a interação com usuário de forma fácil e ágil. Vejamos a baixo os componentes integrados a placa e sua funcionalidade:

4.1 – Raspberry pi pico W

O projeto se utiliza de um microcontrolado Raspberry Pi Pico W que inclui conectividade Wi-Fi. Ele foi projetado para oferecer uma solução acessível e flexível para projetos de IoT (Internet das Coisas), automação, robótica e muito mais. A placa funciona como coração do projeto onde foram conectados os componentes na GPIO e fontes de alimentação para pino 3v3 e terra para GND.



Esquema de entradas placa raspberry pi pico w

4.1.1 - Circuito

Nesta parte do projeto iremos abordar o circuito utilizado no projeto, segue a baixo a lista de componentes:

I. Módulo RTC (Real Time Clock) com interface I2C e 56 bytes de NV SRAM
II. Sensor analógico de temperatura: termistor NTC (coeficiente de temperatura negativo)
III. Um LCD com 2 linhas, 16 caracteres por linha
IV. LED 5mm padrão
V. Resistor de 330 ohms
VI. Um buzzer piezoelétrico
VII. Botão interruptor tátil de 12 mm (botão momentâneo)
VIII. Módulo Relé

4.1.2 – Módulo RTC

O uso do módulo RTC (DS3231) foi essencial para garantir a precisão e confiabilidade do sistema. Ele permite que o alarme seja acionado exatamente no horário programado, além de fornecer a data e hora atual para exibição no LCD. A independência do RTC em relação ao microcontrolador garante que o sistema continue funcionando corretamente mesmo em caso de reinicializações ou falta de energia, tornando-o ideal para aplicações que exigem temporização precisa.

4.1.3 - Sensor analógico de temperatura: termistor NTC

O sensor analógico de temperatura (termistor NTC) foi utilizado para monitorar a temperatura ambiente de forma precisa e eficiente. Sua resposta rápida e baixo custo o tornam ideal para aplicações que exigem medições confiáveis de temperatura. No sistema, o termistor NTC permite a leitura da temperatura em tempo real, fornecendo dados essenciais para o funcionamento do alarme inteligente e possibilitando a expansão do sistema para funcionalidades como controle automático de dispositivos ou monitoramento ambiental.

4.1.4 - LCD com 2 linhas e 16 caracteres por linha

O display LCD foi utilizado para fornecer feedback visual ao usuário, exibindo informações essenciais como a hora atual e a data. Sua facilidade de uso e baixo consumo de energia o tornam ideal para sistemas que precisam ser acessíveis e eficientes. No projeto, o LCD permite que o usuário interaja com o sistema de forma intuitiva, visualizando relevantes em tempo real.

4.1.5 - LED 5mm padrão

O LED 5mm foi utilizado como um indicador visual para fornecer feedback rápido e eficiente sobre o estado do sistema. Sua simplicidade, baixo custo e consumo reduzido de energia o tornam ideal para aplicações que exigem comunicação visual imediata, como o acionamento do alarme. No projeto, o LED é utilizado para indicar o status do alarme, alertando o usuário de forma clara e intuitiva.

4.1.6 - Resistor de 330 ohms

O resistor de 330 ohms foi utilizado em série com o LED 5mm para limitar a corrente que passa pelo componente, garantindo seu funcionamento seguro e eficiente. Esse resistor é essencial para proteger o LED contra danos causados por corrente excessiva e para manter o brilho do LED estável e adequado. Além disso, o resistor protege o microcontrolador de possíveis sobrecargas, assegurando a integridade do circuito.

4.1.7 - Buzzer piezoelétrico

O buzzer piezoelétrico foi utilizado para fornecer feedback auditivo ao usuário, emitindo um sinal sonoro quando o alarme é acionado. Sua simplicidade, baixo custo e capacidade de gerar diferentes frequências o tornam ideal para aplicações que exigem alertas sonoros claros e eficientes. No projeto, o buzzer complementa o feedback visual do LED, garantindo que o usuário perceba o alarme mesmo que não esteja olhando diretamente para o sistema.

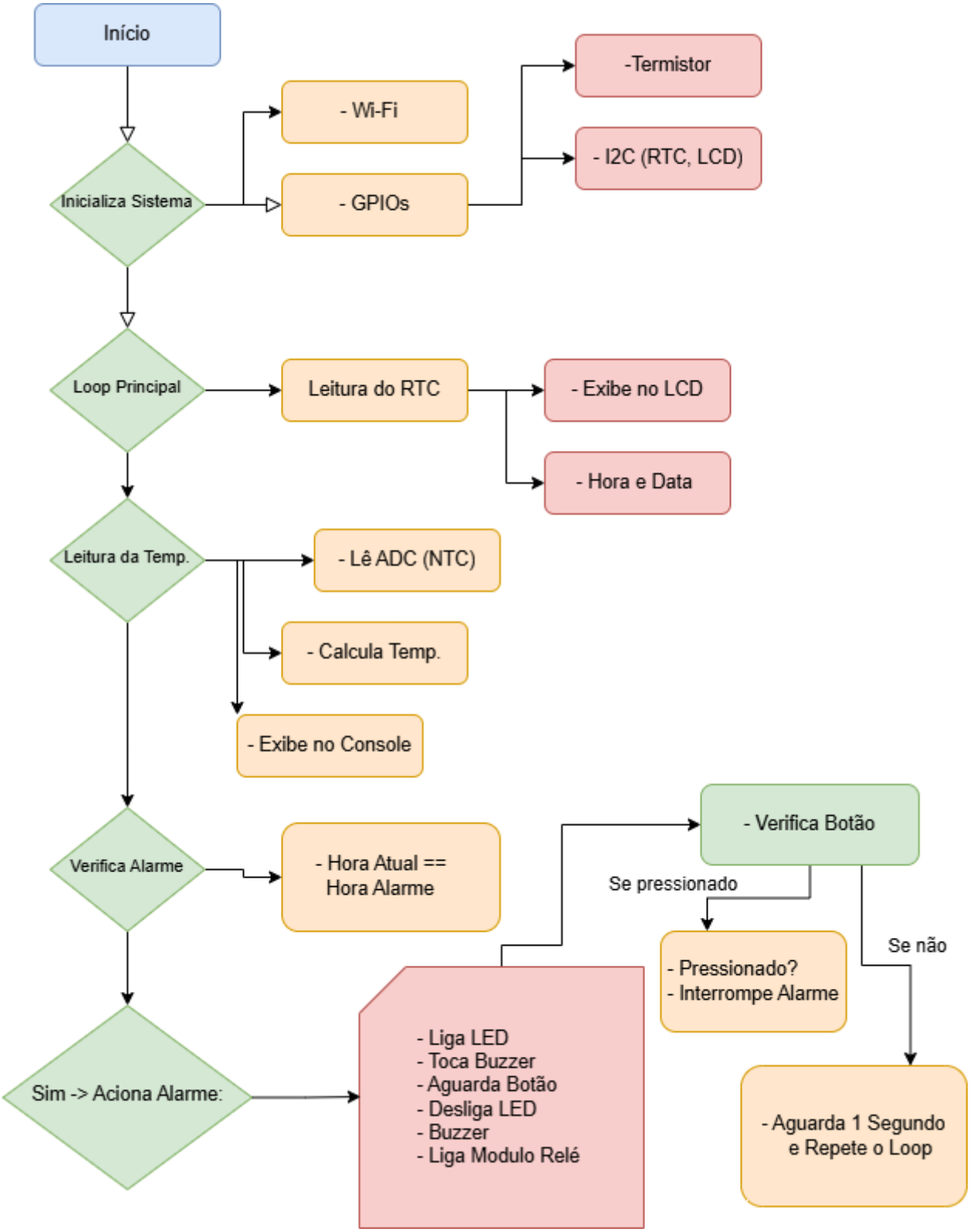
4.1.8 - Botão interruptor tátil de 12 mm

O botão interruptor tátil de 12 mm foi utilizado como interface de entrada para permitir a interação do usuário com o sistema. Sua simplicidade, confiabilidade e facilidade de uso o tornam ideal para aplicações que exigem controle manual, como a interrupção do alarme. No projeto, o botão permite que o usuário desative o alarme de forma rápida e intuitiva, melhorando a usabilidade e a experiência geral do sistema.

4.1.8 – Módulo Relé

o módulo relé é essencial para permitir que o Raspberry Pi Pico W controle dispositivos externos de forma segura e eficiente. Abaixo estão os principais motivos pelos quais o relé é importante: controle de dispositivos externos de alta potência, segurança na qual protege o micro controlador pico de altas tensões, integração com alarme e fácil implementação.

4.2 – Fluxograma



Esquema de funcionamento

4.2.1 – Inicialização do sistema

Nessa parte temos a inicialização do sistema embarcado onde envolve parâmetros cruciais para funcionamento de todo restante do alarme, a começarmos por pela inicialização do Wi-Fi que que permite a placa Raspberry Pi Pico W uma conexão sem fio com a rede local, primeiro definimos a biblioteca #include "pico/cyw43_arch.h" no simulador Wokwi e definimos no código o SSID e senha do wi-fi local com #define WIFI_SSID "Seu_SSID" // Nome da rede do seu wi-fi e #define WIFI_PASSWORD "Sua_Senha" // Senha da rede do seu wi-fi, para checagem da comunicação com a rede Wi-Fi local utilizamos a função:

```
void connectToWiFi() { //Função que conecta ao Wi-Fi
      printf("Conectando ao Wi-Fi: %s\n", WIFI_SSID);
      if (cyw43_arch_init()) {
      printf("Falha na inicialização do módulo Wi-Fi.\n");
return;
      }
      cyw43_arch_enable_sta_mode();
      if (cyw43_arch_wifi_connect_timeout_ms(WIFI_SSID, WIFI_PASSWORD,
CYW43_AUTH_WPA2_AES_PSK, 10000)) {
      printf("Falha ao conectar ao Wi-Fi.\n");
} else {
      printf("Conexão Wi-Fi estabelecida!\n");
      }
}
```

O alarme fara a leitura e indicara no console da plataforma Wokwi se a conexão foi bem sucedida ou se caso não com PRINTF.

4.2.2 – GPIOs

Nessa parte do código temos a inicialização de informações que são apresentadas ao usuário assim que a conexão do Wi-Fi e bem estabelecida, sendo ela a comunicação com componentes de verificação de temperatura e I2C que checa a comunicação com RTC para leitura de data e hora e com o nosso LCD para apresentar no visor os dados coletados por nosso RTC.

4.2.3 – Loop Principal

Chegamos na leitura do RTC, primeiro definimos as portas GPIOs respectivas para portas I2C SDA e SCL sendo elas 26 e 27 da placa Raspberry Pi Pico W e definição de seu endereço 0x68 especifico para uso no DS3231 que encontramos no wokwi, para seu pleno funcionamento chamamos a função void rtc_read_date para leitura da data atual e void rtc_read_time para leitura do horário atual de Brasília e assim sendo ele impresso no console para visualização do usuário. No mesmo pino 26 e 27 temos a conexão do LCD para I2C e se comunica com RTC para leitura de dados e exibição no painel a informação de data e hora atual.

Função utilizada para leitura de data atual:

```
void rtc_read_date(uint8_t *day, uint8_t *month, uint8_t *year) {

      uint8_t buffer[3];
      i2c_write_blocking(I2C_PORT, RTC_I2C_ADDRESS, (uint8_t[]){0x04}, 1, true);
      i2c_read_blocking(I2C_PORT, RTC_I2C_ADDRESS, buffer, 3, false);

      *day = ((buffer[0] >> 4) * 10) + (buffer[0] & 0x0F);
      *month = ((buffer[1] >> 4) * 10) + (buffer[1] & 0x0F);
      *year = ((buffer[2] >> 4) * 10) + (buffer[2] & 0x0F);
}
```

Função utilizada para leitura de hora atual:

```
void rtc_read_time(uint8_t *hours, uint8_t *minutes, uint8_t *seconds) {
      uint8_t buffer[3];
      i2c_write_blocking(I2C_PORT, RTC_I2C_ADDRESS, (uint8_t[]){0x00}, 1, true);
      i2c_read_blocking(I2C_PORT, RTC_I2C_ADDRESS, buffer, 3, false);

      *seconds = ((buffer[0] >> 4) * 10) + (buffer[0] & 0x0F);
      *minutes = ((buffer[1] >> 4) * 10) + (buffer[1] & 0x0F);
      *hours = ((buffer[2] >> 4) * 10) + (buffer[2] & 0x0F);
}
```

4.2.4 – Leitura de temperatura

No código, o termissor NTC está conectado ao pino analógico (ADC) do microcontrolador (GPIO28). A função calculate_temperature converte o valor lido pelo ADC em uma temperatura em graus Celsius, utilizando a equação do termistor e os parâmetros fornecidos (como o coeficiente beta e o resistor fixo).

Função usada para cálculo de temperatura:

```
float calculate_temperature(uint16_t adc_value) {
float voltage = adc_value * (V_REF / ADC_MAX); // Converte o valor ADC em tensão
if (voltage == 0) return -273.15;            // Evita divisão por zero
float resistance = (R_FIXED * voltage) / (V_REF - voltage); // Calcula a resistência do NTC
float temperature = 1 / (log(resistance / R_FIXED) / BETA + 1 / 298.15)      - 273.15; // Fórmula para Celsius
return temperature;
}
```

4.2.5 – Verificação do alarme

Compara a hora atual com a hora programada para o alarme, após o usuário fornecer a hora desejada para acionar o alarme o código irá armazenar o dado e compara-lo com a data e hora atual, se o as duas se coincidirem o alarme será acionado.

4.2.6 – Acionamento do alarme

Caso o alarme acione será gerado uma serie de ações para alertar ao usuário da sua atividade programada, primeiro vamos aos componentes envolvidos sendo eles: led, buzzer, botão e modulo relé. A começarmos pelo LED, conectamos ele a placa Raspberry Pi Pico W no GPIO 15 e definimos no código em #define LED_PIN 15, sua função foi atrelada a função de acionamento do alarme e caso for ligado irá ficar acesso por 10 segundos e se apagara por 3 segundos ficando nesse ciclo por até 6 vezes até se auto desligar. No momento em que o LED é aceso também definimos o BUZZER para funcionar em diferentes frequências durante 6 ciclos que o alarme estiver ativo, conectamos ele na GPIO 2 e definimos como #define BUZZER_PIN 2. Durante o inicio do primeiro ciclo temos a iniciação do nosso modulo relé, essencial para conexão com outros dispositivos que o usuário desejar para ser ligado juntamente com o smart clock, com alarme acionado cargas elétricas são enviadas ao módulo relé ligando componentes, como lâmpadas, motores e aquecedores, através de sinais digitais e dentre outras, mas, para simulação no wokwi demonstramos seu funcionamento com

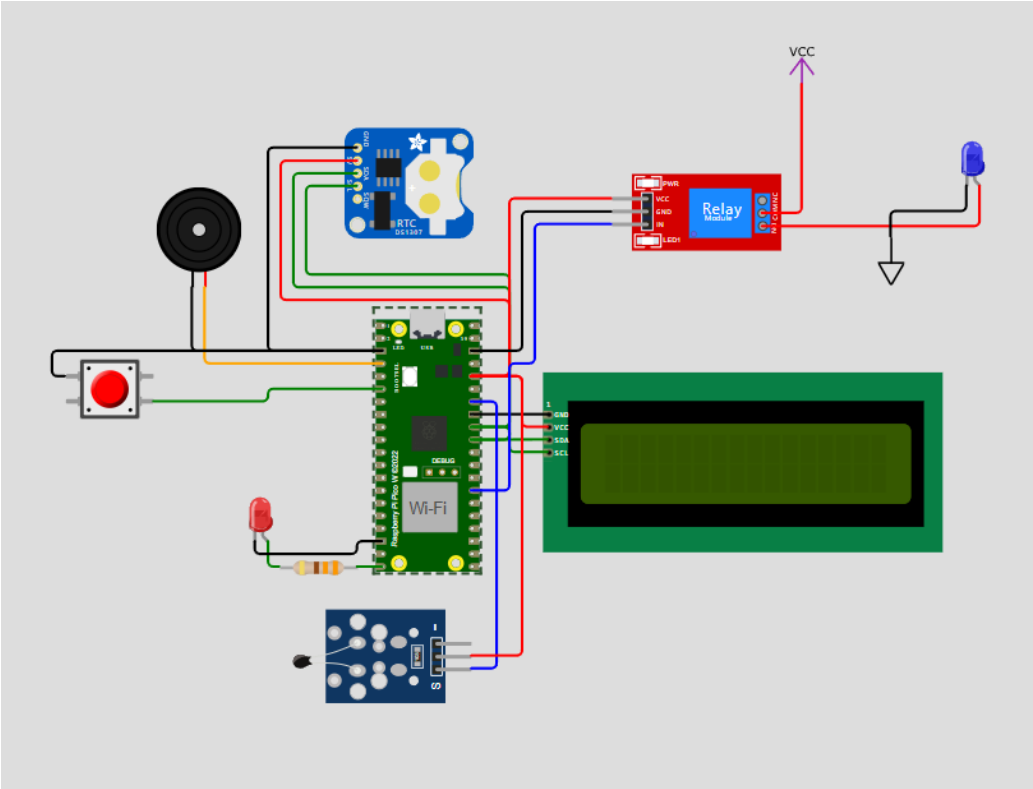
led conectado ao modulo, na placa encontraremos ele conectado a GPIO 21 e definido com #define RELAY_PIN 21. Para pararmos todo acionamento do alarme implementamos um botão conectado a GPIO 4 e definido no código como #define BUTTON_PIN 4, ao ser pressionado quando o alarme for acionado ele irá parar a execução do ciclo e enviar uma mensagem ao console de “Alarme desativado”.

Função para acionamento do alarme:

```
  void activate_alarm() {
    alarm_active = true;
    uint slice_num = pwm_gpio_to_slice_num(BUZZER_PIN);
    uint channel = pwm_gpio_to_channel(BUZZER_PIN);
    gpio_set_function(BUZZER_PIN, GPIO_FUNC_PWM); // Define função PWM para o buzzer
    // Aciona o relé
    gpio_put(RELAY_PIN, 1);  // Liga o relé
    // Sequência de frequências para o buzzer (imita um alarme)
    float frequencies[] = {1000, 1200, 800, 1500, 1000, 1200, 900};
    int num_frequencies = sizeof(frequencies) / sizeof(frequencies[0]);
    for (int cycle = 0; cycle < 6; cycle++) {
        if (!alarm_active) { // Sai do loop se o botão for pressionado
            printf("Alarme interrompido pelo botão.\n");
            break;
        }
        // Liga LED e inicia o ciclo do alarme
        gpio_put(LED_PIN, 1);
        for (int i = 0; i < num_frequencies; i++) {
            if (!alarm_active) { // Verifica novamente o botão durante o ciclo
                printf("Alarme interrompido pelo botão.\n");
                break;
            }
            buzzer_play(slice_num, channel, frequencies[i]);
            sleep_ms(500); // Toca cada frequência por 500ms
            // Verifica se o botão foi pressionado
            if (is_button_pressed(BUTTON_PIN)) {
                alarm_active = false;
                printf("Alarme desativado. \n");
                break;}
        }
        if (!alarm_active) break; // Verifica antes de desligar o LED
        // Desliga LED e buzzer
        gpio_put(LED_PIN, 0);
        buzzer_stop(slice_num, channel);
        sleep_ms(3000); // Pausa de 3 segundos entre os ciclos
    }
    // Garante que o LED e o buzzer estejam desligados no final
    gpio_put(LED_PIN, 0); //Desliga o led
    gpio_put(RELAY_PIN, 0); // Desliga o relé
    buzzer_stop(slice_num, channel); // Desliga o buzzer
    alarm_active = false; // Reseta o estado do alarme
}
```

5 – Interligação dos componentes

Realizamos no wokwi a ligação de todos os componentes necessários para conecta-los a placa Raspberry Pi Pico W, o wokwi é uma ferramenta excelente para simular projetos com microcontroladores como o Raspberry Pi Pico W, e permite visualizar as conexões de forma clara, nosso diagrama esquemático ficou da seguinte maneira:



(Esquemático do projeto)

5.1 - Função de Cada Bloco

5.1.1 - Raspberry Pi Pico W

Microcontrolador principal que coordena todos os componentes. Suas principais tarefas são:

Ler dados do RTC (DS3231).
Controlar o LCD via I2C.
Medir temperatura via NTC (ADC).
Acionar relé, buzzer, LED e botão.
Gerenciar Wi-Fi (Pico W).

5.1.2 - DS3231 (RTC)

- Mantém a hora/data precisa mesmo sem alimentação.
- Fornece hora atualizada para o sistema.

5.1.3 - LCD 16x2 I2C

- Exibir informações de hora, data e status do sistema.

5.1.4 – NTC (Sensor de Temperatura)

- Mede temperatura ambiente via divisor resistivo.

5.1.5 - Módulo Relé

- Controla dispositivos de alta potência (ex: lâmpadas).

5.1.6 – Buzzer

- Gerar alertas sonoros (alarme).

5.1.7 – LED

- Indicador visual de status/alarme.

5.1.8 – Botão

- Interrompe alarme ou configurar horários.

5.1.9 - Wi-Fi (CYW43)

- Conectar à rede para possíveis atualizações remotas (Pico W).

5.2 - Configuração de Cada Bloco

5.2.1 - DS3231 (RTC)

Protocolo: I2C (100 kHz).
Endereço: 0x68 (hex).
Registros:
- 0x00: Segundos
- 0x01: Minutos
- 0x02: Horas
- 0x04: Dia
- 0x05: Mês
- 0x06: Ano

5.2.2 - LCD 16x2 I2C

Protocolo: I2C (mesmo barramento do RTC).
Endereço: 0x27 (hex).
Modo: 4 bits, 2 linhas.
Comandos utilizados:

```
LCD_CLEAR_DISPLAY    // Limpa o display
LCD_RETURN_HOME      // Retorna cursor ao início
LCD_DISPLAY_CONTROL  // Controla visibilidade do cursor
```

5.2.3 - NTC (Sensor de Temperatura)

Configuração ADC:
- Pino: GPIO28 (ADC2).
- Resolução: 12 bits (ADC_MAX = 4095).

Fórmula de conversão:

```
// Resistência do NTC: R = (R_FIXED * V_adc) / (V_REF - V_adc)
// Temperatura: 1 / (ln(R/R_FIXED)/BETA + 1/298.15) - 273.15
```

5.2.4 – Buzzer

Configuração PWM:
Pino: GPIO2.
Slice PWM: pwm_gpio_to_slice_num(BUZZER_PIN).
Frequência ajustável via pwm_set_clkdiv().

5.2.5 – Relé

Controle Digital:
Pino: GPIO21.
Lógica: 1 = Ativado, 0 = Desativado.

5.3 - Comandos e Registros Utilizados

5.3.1 - I2C (RTC e LCD)

```
i2c_init(I2C_PORT, 100000);      // Inicializa I2C a 100 kHz
i2c_read_blocking()              // Leitura de registros do RTC
i2c_write_blocking()             // Escrita no LCD
```

5.3.2 - ADC (NTC)

```
adc_init();                  // Inicializa hardware ADC
adc_gpio_init(NTC_PIN);      // Configura GPIO28 como ADC
adc_select_input(2);         // Seleciona canal ADC2
```

5.3.3 - PWM (Buzzer)

```
pwm_set_clkdiv(slice_num, div);  // Define divisão de clock
pwm_set_wrap(slice_num, wrap);   // Define período do PWM
pwm_set_chan_level(...);         // Define duty cycle
```

5.4 - Descrição da Pinagem

Pino Pico	*Componente*	*Função*
GPIO26	SDA (I2C)	Dados para RTC e LCD
GPIO27	SCL (I2C)	Clock para RTC e LCD
GPIO28	NTC	Leitura analógica (ADC2)
GPIO21	Relé	Controle digital (saída)
GPIO2	Buzzer	Sinal PWM (saída)
GPIO15	LED	Saída digital
GPIO4	Botão	Entrada digital (pull-up)
3.3V	Todos	Alimentação lógica
GND	Todos	Terra comum

6 – Conclusão

O projeto Smart Clock, desenvolvido no Raspberry Pi Pico W em linguagem C e simulado no Wokwi, representa uma solução inovadora e versátil para automação residencial ou monitoramento de ambientes. Destaco os pontos-chave que reforçam sua relevância, resultados positivos e confiabilidade, a combinação de monitoramento de temperatura, sincronização de data/hora via RTC e controle remoto de dispositivos externos cria um sistema modular e adaptável a diferentes cenários, como segurança, conforto térmico ou eficiência energética. A implementação do uso do Wi-Fi para atualizar dados (como temperatura e horário) e enviar comandos a outros dispositivos garante que o sistema opere com informações atualizadas, essencial para decisões rápidas em situações críticas. A escolha do Raspberry Pi Pico W, aliada à linguagem C, oferece um equilíbrio entre desempenho, eficiência energética e custo reduzido, tornando o projeto acessível e escalável. A simulação no Wokwi permitiu testar a lógica do código, a conexão Wi-Fi e a interação com sensores/atuadores virtualmente, reduzindo riscos de falhas antes da implementação física. O projeto pode facilmente ser utilizado em projetos futuros garantindo segurança residencial para acionar luzes ou câmeras em caso de invasão detectada, controle ambiental para ligar ar-condicionado se a temperatura ultrapassar limites pré-definidos, automação industrial para monitorar máquinas e desligá-las em situações de superaquecimento. Melhorias que podem ser implementadas no código podemos dizer que seriam elas adicionar um dashboard remoto para visualização de dados e controle dos dispositivos, implementar criptografia (TLS/SSL) na comunicação Wi-Fi para proteger dados sensíveis, usar algoritmos para prever padrões de temperatura ou detectar anomalias. É possível criar sistemas IoT confiáveis e de baixo custo com o Raspberry Pi Pico W, combinando hardware acessível, software eficiente (C) e simulação prévia (Wokwi). A confiabilidade é reforçada pela escolha de protocolos estáveis, calibração de sensores e código otimizado, enquanto a flexibilidade permite adaptação a diversas necessidades. Com testes contínuos e expansões futuras, essa solução tem potencial para se tornar uma ferramenta indispensável em automação inteligente.

LINK DA SIMULAÇÂO VIA WOKWI:

https://wokwi.com/projects/420993025459313665

7 – Referências

1. *repositorio.ufpb.br/jspui/bitstream/123456789/32557/1/Mateus%20Antonio%20da%20Silva_TCC.pdf*

2. *repositorio.ufc.br/bitstream/riufc/49723/1/2019_tcc_fafelixneto.pdf*

3. *docs.wokwi.com/pt-BR/*

4. *forums.raspberrypi.com/*

5. *www.clubedohardware.com.br/*

6. *github.com/BitDogLab/BitDogLab-C/blob/main/wifi_button_and_led/pico_w_wifi_complete_example.c*

7. *github.com/raspberrypi/pico-examples/tree/master?tab=readme-ov-file#pico-networking*