EMBARCATECH E INSTITUTO HARDWARE BR
RESIDÊNCIA TECNOLÓGICA EM SISTEMAS EMBARCADOS

ARTHUR FRANCO NETO

SISTEMA PARA CONTROLE DE ALUNOS DO ENSINO INFANTIL NO TRANSPORTE ESCOLAR

CAMPINAS
2025

ESCOPO DO PROJETO

Apresentação

Para quem tem filho pequeno que estuda em escola sempre tem aquela preocupação por eles estarem fora de casa. E uma das maiores preocupações está relacionada ao transporte escolar, seja ao fato de alguma criança ser esquecida dentro do ônibus ou van, ou de acabar não sendo enviada de volta pra casa por algum esquecimento por parte dos profissionais na escola. Diante dessa preocupação foi desenvolvido uma proposta de sistema embarcado que possa auxiliar no controle dos alunos e diminuir o risco de falhas humanas.

Objetivos

O objetivo do projeto é desenvolver uma solução embarcada que proporcione o monitoramento dos alunos no transporte escolar. Para isso alguns requisitos foram definidos para atender a demanda:

Principais Requisitos

- O sistema deve ter um alarme sonoro para indicar a falta de algum aluno;
- O sistema deve indicar uma forma de reconhecimento do aluno pendente;
- Visualização rápida da quantidade de alunos.
- Deve ter fácil visualização e fácil operação

Tentando atender os requisitos foi pensado um sistema embarcado que utilizará tecnologia RFID para controle dos alunos dentro do transporte escolar.

Descrição do Funcionamento

Os alunos receberão etiquetas RFID personalizadas (podendo ser em forma de crachá ou em pulseiras) com dados que possam garantir a identificação por parte do sistema. O sistema embarcado fica encarregado de fazer a leitura das etiquetas e gerenciar os alunos nos momentos de embarque e desembarque.

O sistema terá uma etiqueta do Motorista (denominada também como “ADMIN”) que define a Linha/Percurso do ônibus e somente alunos que tiverem a mesma informação de Linha/Percurso será contabilizado no sistema.

Através de uma visualização simples, será possível identificar a quantidade de alunos em cada etapa, e caso haja alguma divergência na informação de algum aluno perante o transporte será disparado um alarme e uma visualização da identificação do aluno. Para parar o alarme, somente através da etiqueta do próprio aluno ou através da etiqueta do motorista. Abaixo uma representação das etapas do transporte:

- Etapa 1: Embarque – reconhecimento dos alunos para ir para escola através do transporte escolar;
- Etapa 2: Desembarque na escola – todos os alunos que deram check in no embarque devem dar saída nessa etapa;
- Etapa 3: Reembarque – reconhecimento dos alunos que embarcaram para reembarque;
- Etapa 4: Ponto final – todos os alunos que deram check in no reembarque devem dar saída nessa etapa;

Durante a Etapa 1 após todos os embarques o operador deve apertar o Botão “AVANÇAR” do equipamento para finalizar a etapa de embarque, já que o número de alunos pode variar. Depois cada etapa é finalizada automaticamente se todas as condições de embarque e desembarque forem satisfeitas. Se houver alguma divergência o operador do sistema pode apertar o Botão “AVANÇAR” para fazer a verificação dos alunos.

Justificativa

A segurança dos nossos filhos é prioridade absoluta e o transporte escolar é crucial nessa cadeia de cuidados, por isso o projeto se faz necessário para trazer uma alternativa e auxílio tecnológico para profissionais e escola evitando assim que possíveis falhas humanas possam gerar transtornos como esquecimento de crianças na escola ou até mesmo dentro do transporte.

Originalidade

Existem muitos sistemas baseados em tecnologia RFID para controle de acessos, outros tantos utilizam todo o recurso que smartphones possuem para trazer soluções integradas com mapas mas nada focado especificamente no controle de crianças pequenas para evitar esquecimento. Além disso a ideia foi criar uma interface simples e intuitiva

HARDWARE EMBARCADO

Para definição do hardware os requisitos de funcionamento são fundamentais para escolha do hardware. Como um dos principais requisitos do projeto é a identificação do aluno, optou-se por utilizar a tecnologia RFID para gerenciamento dos alunos através de TAG's (em forma de crachá, ou inseridas em um relógio). A utilização das etiquetas RFID se dá pelo baixo custo das TAG's e uma maior simplicidade na implementação em relação a outros métodos como reconhecimento biométrico ou facial.
A utilização da matriz NeoPixel, junto com o LED RGB e o buzzer e apenas um Push Button tentam trazer simplicidade no uso e fácil visualização do sistema.

Como proposta para o hardware embarcado, definimos o seguinte diagrama em blocos exibido na figura abaixo.

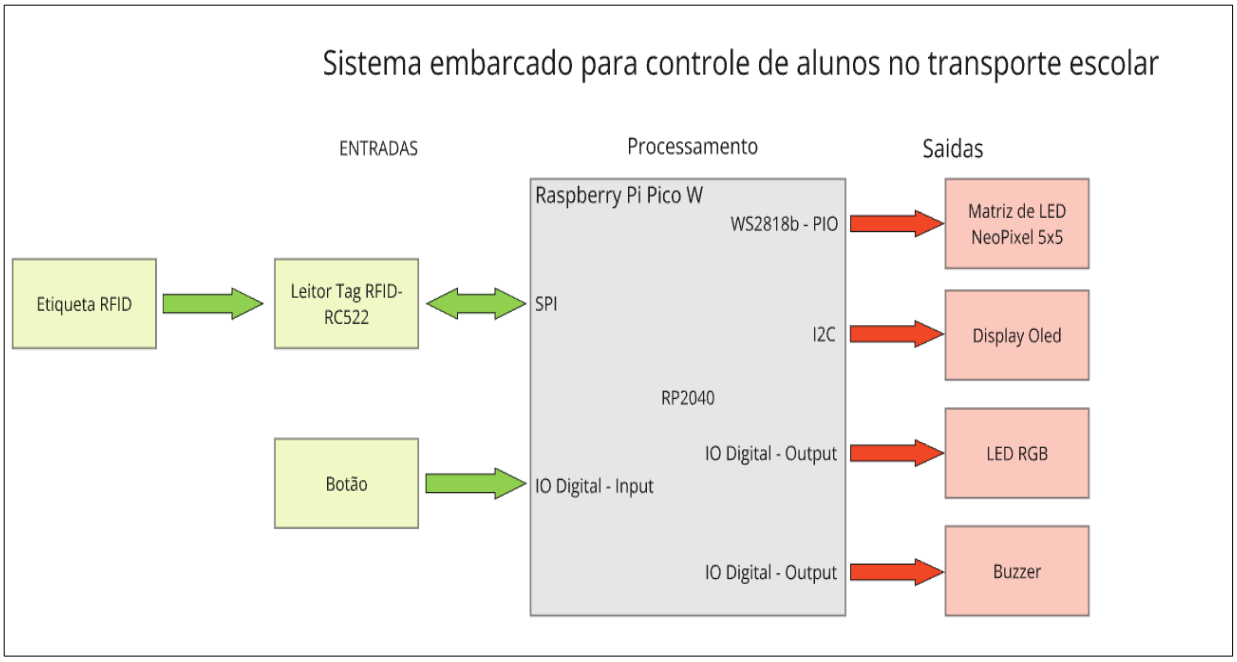

Figura 1: Diagrama em Blocos do Hardware

Podemos separar nosso hardware embarcado em três blocos funcionais: Entradas, processamento e saídas

1. Entradas
    a) Etiqueta RFID – Componentes passivos que transportam informações e que são lidas através da Rádio Frequência. Para o projeto utilizamos TAG's MIFARE na frequência de 13,56MHz, essas etiquetas permitem que as informações do aluno sejam lidas pelo sistema;
    b) Leitor Tag RFID-RC522 – tem a função de fazer as leituras das TAG's RFID's através de rádio frequência e enviar os dados para o microcontrolador;

c) Botão – permite a interação entre operador e sistema.

2. Processamento – O microcontrolador é responsável por todo gerenciamento dos periféricos de entrada e saída, protocolos de comunicação e processamento dos dados.
3. Saídas:
   a) Matriz de LED NeoPIXEL – tem como objetivo trazer a indicação da quantidade de alunos no veículo;
   b) Display OLED – através do display podemos exibir informações dos alunos que estão faltando, ou outras mensagens para o operador;
   c) LED RGB – utilizado para criar uma visualização do status da leitura das TAG's RFID;
   d) Buzzer – tem a função de emitir sinais sonoros sobre status de operações ou alarmes.

Cada um desses blocos tem sua configuração correspondente, que será detalhada abaixo:

Etiqueta RFID – A etiqueta (TAG) utilizada trabalha na frequência de 13,56Mhz e é do tipo MIFARE com 1KB de memória EEPROM. A EEPROM na Tag é importante porque conseguimos definir alguns dados para personalização.

Leitor RFID-RC522 – O Leitor de RFID trabalha com TAG's de 13,56Mhz e será interfaceado através do protocolo SPI com nosso microcontrolador. Existe ainda um pino de Reset que será controlado por uma saída Digital do micro.

Botão – O Botão é um Push Button normalmente aberto, é configurado como uma entrada Digital, com resistor de pull up ativado, para chavear o sinal GND quando pressionado.

Microcontrolador – O microcontrolador é o RP2040, que através das rotinas já definidas trabalha a 125Mhz.

Matriz LED Neopixel – A matriz neopixel utiliza o protocolo de comunicação ws2818b que permite através de 1 único pino controlar todos os leds da matriz, para isso exige uma comunicação com temporização muito rápida e critica. Para contornar esse problema utilizamos a máquina PIO (Programable Input/Output) do RP2040.

Display OLED – Para comunicação com o display OLED utilizamos o protocolo I2C.

LED RGB – Para o LED RGB são utilizados resistores para limitação da corrente do LED e o controle é feito através de saídas digitais (um para cada cor do LED).

Lista de Materiais utilizados no desenvolvimento:

Descrição	Quantidade
Placa Raspberry Pi Pico W	1
Matriz LED Neo Pixel (5x5)*	1
Buzzer	1
Botão Push Button	1
Display Oled 128x64	1
Leitor RFID-RC522	1
LED RGB*	1
Etiquetas RFID Mifare 13,56Mhz	10

*Desconsiderados resistores para ligação dos leds por já estarem na placa da BitDogLab
Também foi utilizado um mini protoboard e 7 jumpers fêmea-fêmea para ligação do Leitor RFID na placa da BitDogLab

Descrição da Pinagem:

Componente	GPIO	Função
Leitor RFID-RC522	GP17	SPI – CS (Chip Select)
	GP18	SPI – SCLK (Serial Clock)
	GP19	SPI – MOSI (Master Out Slave In)
	GP16	SPI – MISO (Master In Slave Out)
	GP20	Output – Controle de Reset
LED RGB	GP11	Output – Controle LED Verde
	GP12	Output – Controle LED Azul
	GP13	Output – Controle LED Vermelho
Matriz LED NeoPixel	GP7	Output – Din (Data Input) Protocolo ws2818b através de PIO (Programable Input Output)
Buzzer	GP21	Output – Controle via PWM
Display OLED	GP14	I2C – SDA (Serial Data)
Display OLED	GP15	I2C – SCL (Serial Clock)
Botão	GP5	Input – Resistor Pull up

Abaixo segue esquema do circuito completo do hardware, o esquema de ligação pode ser consultado através do simulador Wokwi no link: https://wokwi.com/projects/423236220655967233

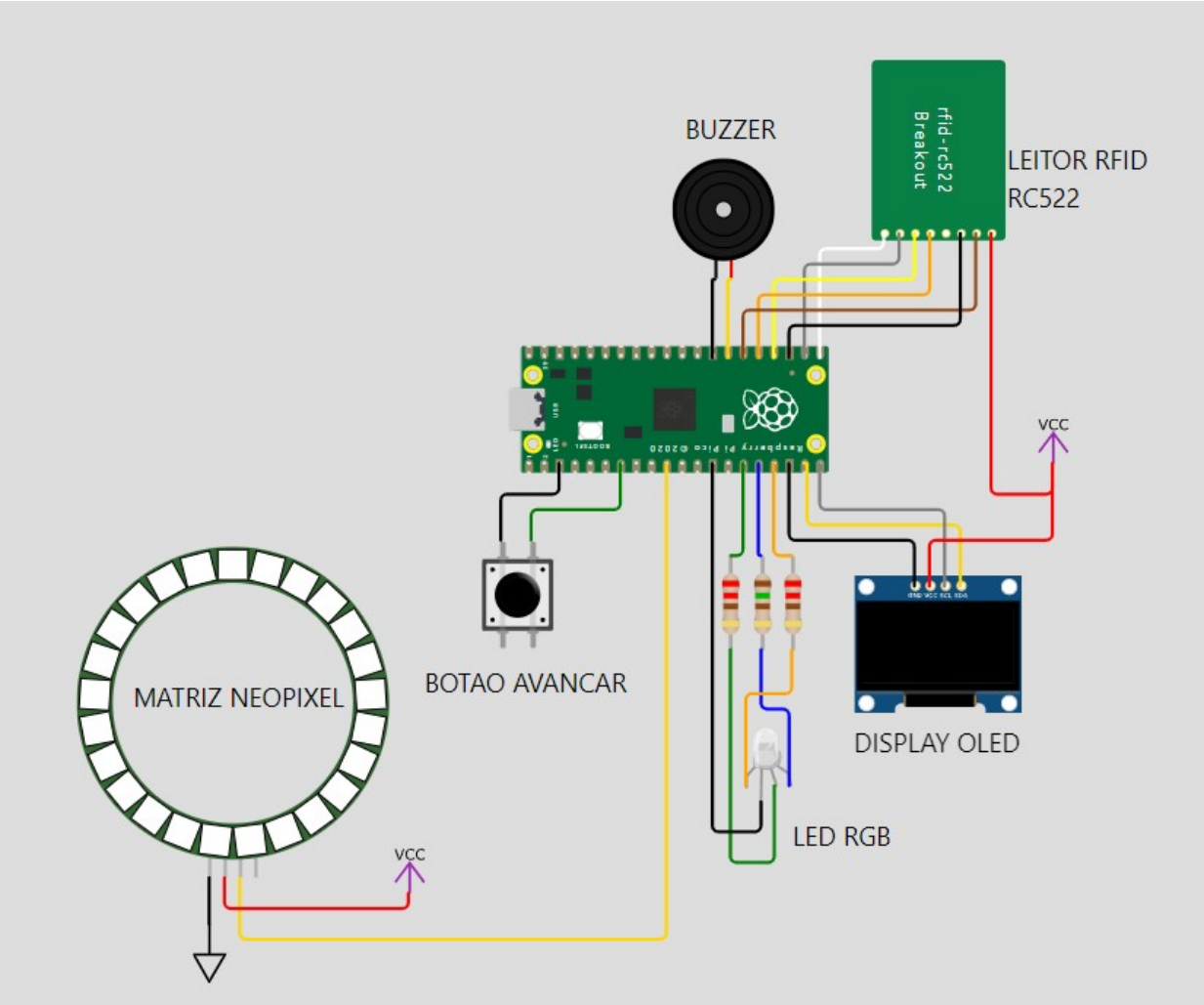

Figura 2: Esquema de Ligações do Hardware

SOFTWARE EMBARCADO

O projeto é baseado na leitura de informações das TAG's RFID, como ela possui uma memória EEPROM acabamos utilizando algumas posições da memória para armazenar dados para o sistema. As TAG's utilizadas possuem 16 Setores de 4 blocos, onde cada Bloco possui 16 bytes de tamanho.

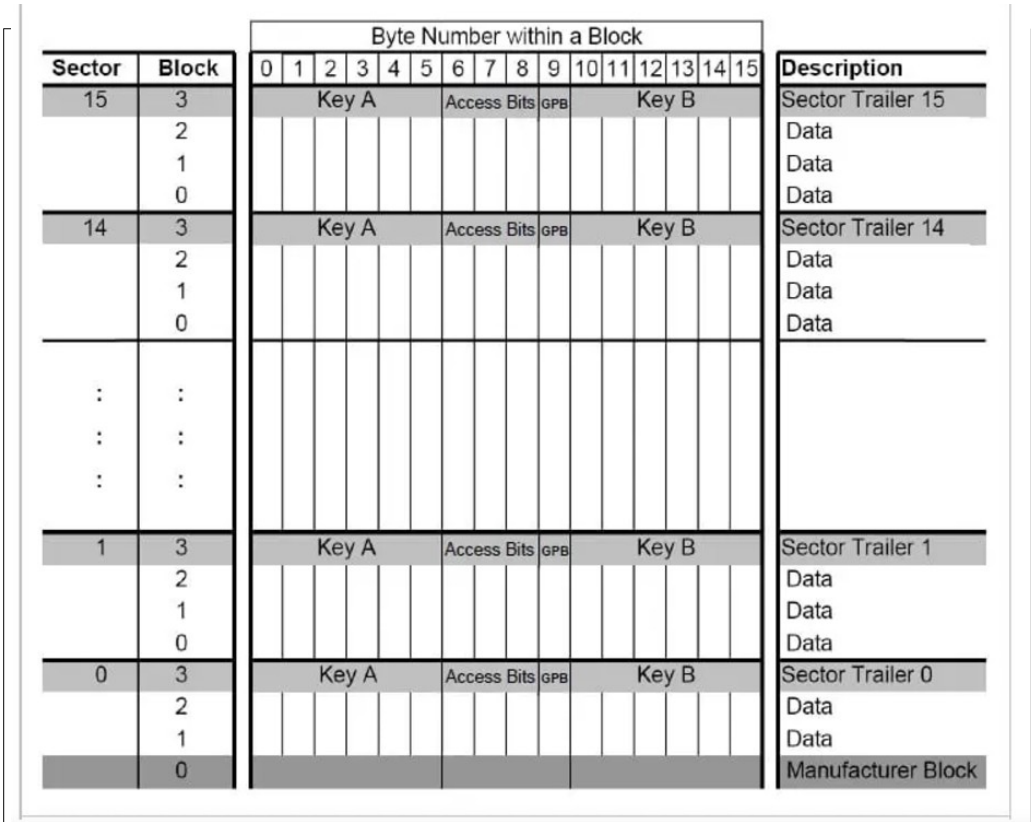

Figura 3: EEPROM TAG MIFARE

No projeto utilizamos apenas o setor 0, que armazena os dados do fabricante como o ID e tem os blocos 1 e 2 disponíveis para dados:

Bloco 0 – ID da TAG (4 primeiros Bytes)

Bloco 1 – Nome do Aluno ou Nome do Administrador (quando cartão do Motorista)

Bloco 2 – Linha / Percurso (Foi utilizado apenas os 4 primeiros bytes, sendo o primeiro para validação devendo ter o valor 0xAA e os três demais para determinar a linha do ônibus.

Na figura abaixo é exibida uma configuração da TAG no setor 0

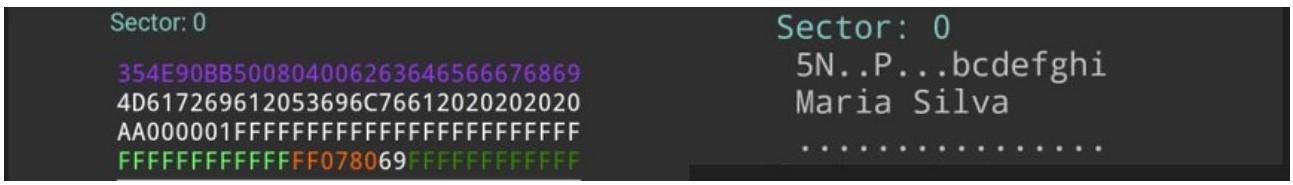

Figura 4: Dados EEPROM da TAG - Setor 0

Através da leitura das informações das Tags são realizados todos os gerenciamentos necessários para controle dos alunos, detecção de erros, exibição de status e outros.

No desenvolvimento do software embarcado podemos separá-los em alguns blocos funcionais para uma visão genérica do funcionamento do sistema, logo abaixo é detalhado a função de cada bloco, apresentando pontos como variáveis, inicialização, configurações e protocolos utilizados.

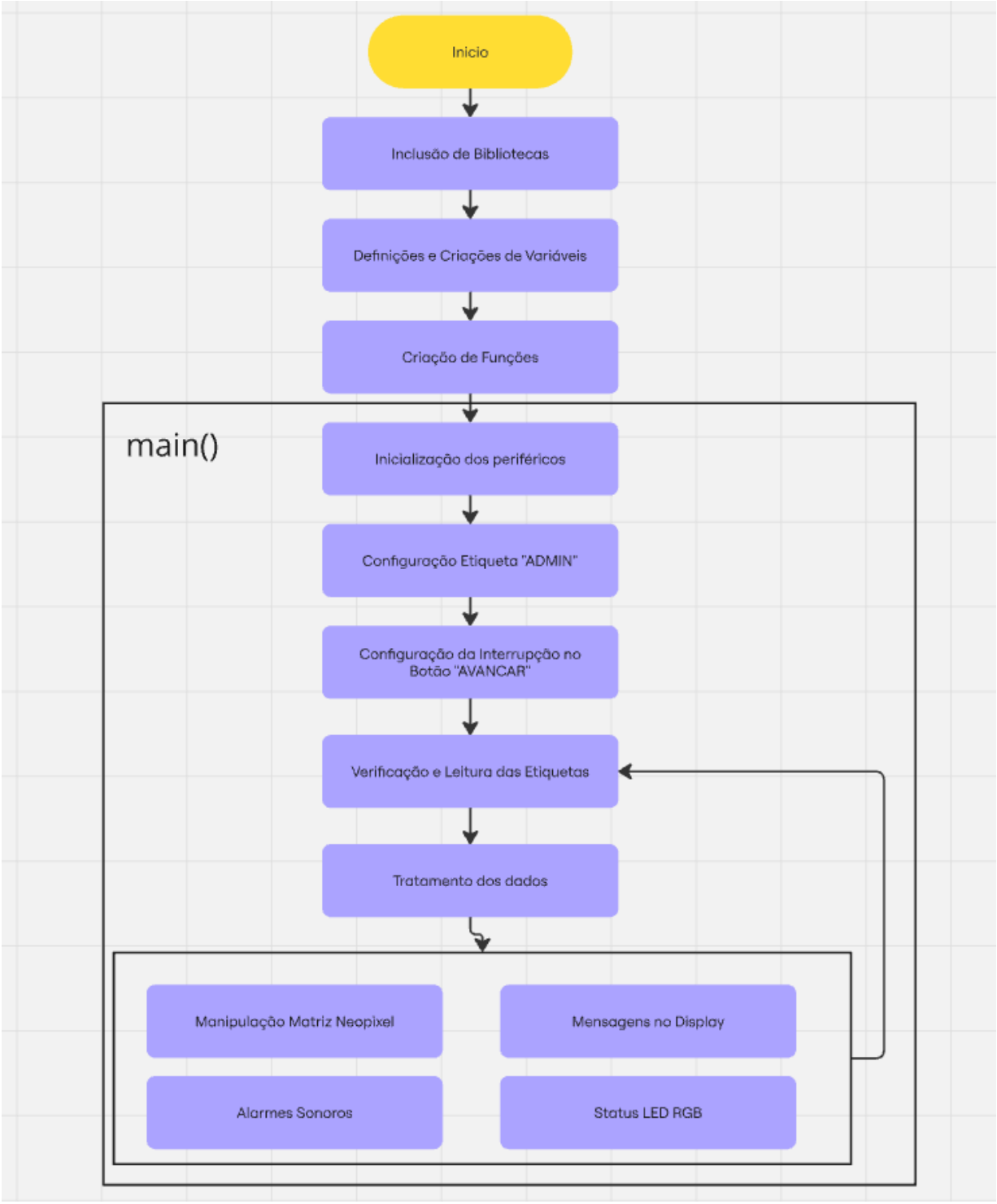

Figura 5: Blocos de funcionamento do software

Inclusão de Bibliotecas

As bibliotecas permitem trabalhar com funções pré definidas para periféricos como PIO, PWM e protocolos como SPI e I2C além do display OLED.

Uma biblioteca particular que permite a comunicação com o leitor RFID-RC522 e consequentemente a leitura das etiquetas é a `#include "mfrc522.c"` que faz utilização do protocolo SPI e tem inúmeras funções para leitura, autenticação e escrita das Etiquetas.

Definições e Criação de Variáveis

São criadas definições que facilitam a visualização do código e variáveis para controle do sistema.

Algumas das principais variáveis são descritas abaixo:

Variáveis inteiras:

```
uint8_t n_alunos_embarc = 0; // Controle do número de alunos para embarque
uint8_t n_alunos_desemb = 0; // Controle do número de alunos para manipulação dos dados
volatile uint8_t Etapa = 1; // Gerenciamento da Etapa da Viagem / Fluxo do sistema
```

Variáveis booleanas como flags de controle:

```
volatile bool led_on_off = false; //Flag para indicar estado do led RGB
volatile bool led_ativo = false;  //Flag para indicar se o sistema pode trabalhar com o LED RGB (true - ON / false - OFF)
volatile bool botao_liberado = false;   //Flag para indicar se o BOTAO AVANCAR tem efeito, ou se será ignorado
volatile bool alarme_disparado = false; //flag para indicar se o alarme esta disparado ou não
volatile bool alarme_on = false;    //Flag para indicar se o alarme esta ligado ou não
volatile bool mudanca_etapa = false;    //Flag Indicando mudanca de etapa do sistema
```

Estrutura de dados para informações do aluno

```
//Definição de estrutura para salvar os dados das Tags dos Alunos
typedef struct {
  char nome[10];  //Utilizado para salvar os dados lidos da EEPROM da TAG
  char escola_linha[10]; //Utilizado para salvar os dados lidos da EEPROM da TAG
  uint8_t tag[4]; //Utilizado para salvar os dados lidos da EEPROM da TAG
  bool embarque; // Flag de status do embarque do aluno
} meusAlunos;
meusAlunos alunos[max_alunos];
```

Criação de Funções

Criação de Funções que serão utilizadas no decorrer do programa.

- Manipulação da Matriz, através do PIO;
- Manipulação do LED RGB;

- Manipulação do PWM e Buzzer;
- Função de Alarme sonoro;
- Funções de manipulação do Display;
- Função de Callback da interrupção do Botão por borda de descida
- Funções para exibição de mensagens no Display

Função Principal main()

- **Inicialização dos Periféricos**

Nessa etapa são feita as configurações dos pinos e inicialização dos periféricos.

Inicialização dos Pinos para controle do LED RGB que indica o Status do Leitor

```
config_GPIO_OUT(LED_RGB_G); //Led Verde
config_GPIO_OUT(LED_RGB_R); //Led Vermelho
config_GPIO_OUT(LED_RGB_B); //Led Azul
```

Configuração do botão como entrada com resistor de pull-up interno.

```
gpio_init(BOTAO_AVANCAR);                  // inicia botao A
gpio_set_dir(BOTAO_AVANCAR, GPIO_IN);      // configura GPIO do botao como entrada
gpio_pull_up(BOTAO_AVANCAR);               // Habilita o resistor pull-up interno
```

Configuracao do PWM do Buzzer

```
setup_pwm_buzzer();
```

Inicializa matriz de LEDs NeoPixel.

```
npInit(LED_PIN);
npClear();
```

Inicialização do Módulo RFID

```
MFRC522Ptr_t mfrc = MFRC522_Init();
PCD_Init(mfrc, spi0);
```

Inicialização do i2c para Display OLED

```
i2c_init(i2c1, ssd1306_i2c_clock * 1000);  //Inicia I2C 1
gpio_set_function(I2C_SDA, GPIO_FUNC_I2C); //Configura GPIO como I2C
gpio_set_function(I2C_SCL, GPIO_FUNC_I2C); //Configura GPIO como I2C
gpio_pull_up(I2C_SDA);                      //Ativa resistor de Pull up
gpio_pull_up(I2C_SCL);                      //Ativa resistor de Pull up
ssd1306_init();  // Processo de inicialização completo do OLED SSD1306
calculate_render_area_buffer_length(&frame_area);
zerar_display();
```

- **Configuração da Etiqueta “ADMIN”**

  Para fazer a leitura das Etiquetas, temos que enviar comandos seriais através do protocolo SPI para o leitor, que depois fará o papel de tranceiver e gerar a comunicação com a TAG.

  `!PICC_IsNewCardPresent(mfrc);` – Detecta se existe uma etiqueta presente no leitor
  `PICC_ReadCardSerial(mfrc);` - Realiza a Leitura do ID da etiqueta

`PICC_ReadBlock(mfrc, &(mfrc->uid), 1);` - Realiza a Leitura do Bloco da EEPROM para pegar informações como “Nome” e “Linha/Percurso”

Depois de lido os tags da Etiqueta “ADMIN” os mesmos são comparados com os padrões pré estabelecidos:

```
if(memcmp(mfrc->uid.eeprom, NomeAdmin, 10) == 0) && mfrc->uid.eeprom[10] == 0xAA) -
    uint8_t NomeAdmin[10] = {'A', 'D', 'M', 'I', 'N', '*', '*', '*', '*', '*'}; //Validacao da TAG ADMIN
uint8_t LinhaOnibus[4] = {}; //4 bytes reservados para a linha, o primeiro obrigatoriamente deve ser 0xAA
```

- **Configuração da Interrupção no Botão “AVANÇAR”**

  `gpio_set_irq_enabled_with_callback(BOTAO_AVANCAR, GPIO_IRQ_EDGE_FALL, true, &gpio_callback);` - Configura uma interrupção por borda de Descida do Botão Avançar e que chamara a rotina gpio_callback

Na função de callback é checado se é permitido naquele momento fazer o incremento da Etapa. Situações como alarme disparado e execução de funções criticas não permitem a alteração

```
void gpio_callback(uint gpio, uint32_t events) {
  if (botao_liberado == true && alarme_disparado == false && mudanca_etapa == false){     if(Etapa < 8){
      Etapa++;     }
    else{   Etapa = 8; }
    mudanca_etapa = true;
    botao_liberado = false; }
}
```

- **Verificação e Leitura das Etiquetas**

  `!PICC_IsNewCardPresent(mfrc); -` Detecta se existe uma etiqueta presente no leitor
  `PICC_ReadCardSerial(mfrc);` - Realiza a Leitura do ID da etiqueta, retorna os dados do ID em `mfrc->uid.uidByte`
  `PICC_ReadBlock(mfrc, &(mfrc->uid), 1);` - Realiza a Leitura do Bloco da EEPROM para pegar informações como “Nome” e “Linha/Percurso”, retorna os dados em `mfrc->uid.eeprom`

- **Tratamento dos Dados**

  O bloco de tratamento dos dados faz o gerenciamento das informações lidas das etiquetas dos alunos para cadastros, comparações e verificações para poder exibir em alguma das saídas disponíveis. A maior parte da estrutura utiliza comandos de comparações para determinar a ação a ser tomada

Saídas

- **Manipulação Matriz Neopixel**

  A matriz de LED Neopixel é responsável por exibir as informações quantitativas dos alunos no sistema. Através do protocolo w2818b e da máquina PIO enviamos dados para

controlar um led em qualquer posição da matriz. A função `entrada_matriz_verd(n_alunos_desemb);` permite configurar um pino especifico na matriz para exibir a contagem crescente ou decrescente dos alunos. O seguinte padrão de cores foi seguido:

Verm – Etapa 1 – Embarque dos alunos
Amar – Etapa 2 – Desembarque dos alunos na escola
Verd – Etapa 3 – Reembarque dos alunos
Desl – Etapa 4 – Ponto Final

- **Mensagens no Display OLED**

Algumas informações são exibidas no Display OLED, como Etapa do Sistema, Erros, ou o nome dos Alunos que estão pendentes no Sistema. Os dados são enviados para o display através de comunicação serial I2C.

Por exemplo, para exibir o nome do aluno no display utilizamos a função `ssd1306_draw_char(ssd, x, y, alunos[i].nome[j]);` para desenhar os caracteres, onde os parâmetros x e y indicam os eixos de posicionamento inical no display e o nome do aluno é escrito byte a byte em um loop for.

A função `render_on_display(ssd, &frame_area);` renderiza efetivamente a imagem na tela do display.

- **Alarmes sonoros**

No bloco de alarme sonoros definimos dois tipos de funções de saída:

1. Função de Status – que representa alguma operação do sistema como “Operação OK”, “Error” ou “Mudança de Etapa”. Para esses três status foram definidos temporização e quantidades de bips para indicação sonora.

Função para indicar Sucesso, 2 bips curtos

```
void bipar_ok(){
  beep(2, 50);
}
```

Função para indicar mudança de Etapa, 3 bips médios

```
void bipar_mudanca_etapa(){
beep(3, 100);
}
```

Função para indicar Erro, 3 bips longos

```
void bipar_erro(){
  beep(3, 200);
}
```

2. Alarme para casos de identificação de divergência da quantidade de alunos em relação ao embarque e desembarque

A função `add_alarm_in_ms(10, disparar_alarme, NULL, false);` dispara um alarme sonoro que só para após as pendências dos sistemas serem resolvidas.

- **Status LED RGB**

  Outra forma de visualização da operação da leitura das Etiquetas se dá através do LED RGB, e alguns padrões de cores foram definidos.

  LED Amarelo – Aguardando TAG “ADMIN”

  LED Azul – Aguardando TAG “Aluno”

  LED Verde – pisca para indicar uma leitura com sucesso

  LED Vermelho – pisca para indicar uma falha na leitura

Abaixo segue um fluxograma completo do software

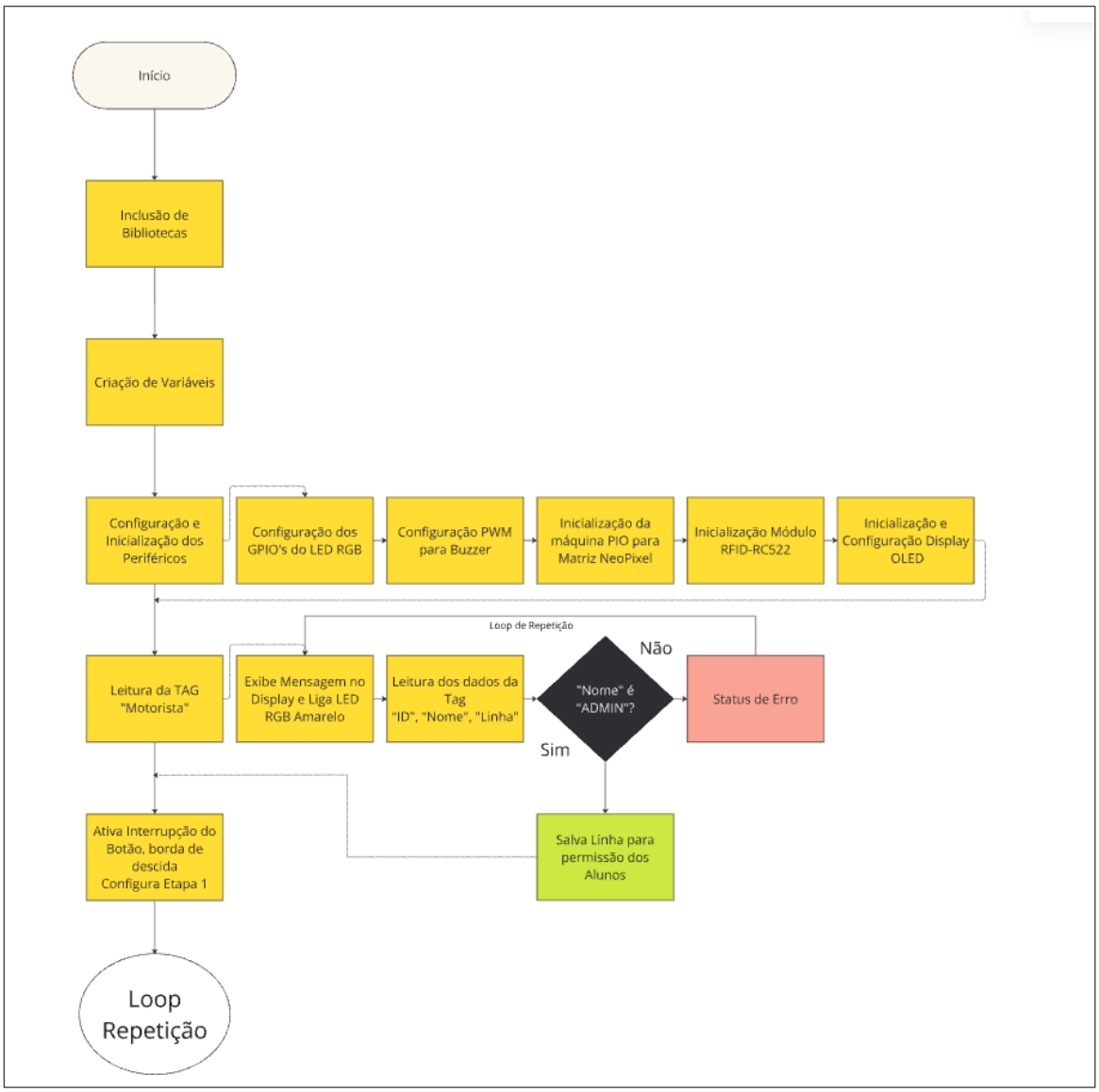

Figura 6: Fluxograma - Configurações e Inicialização

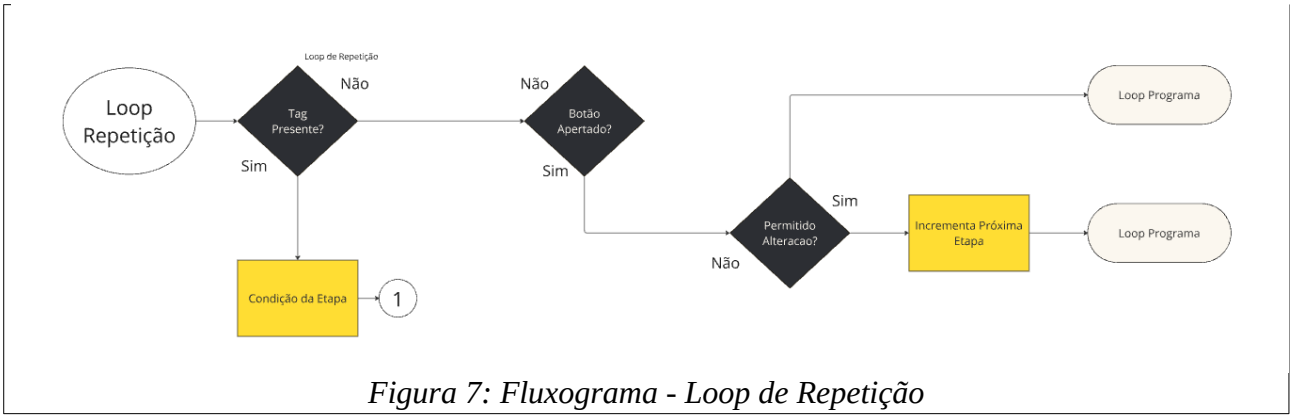

Figura 7: Fluxograma - Loop de Repetição

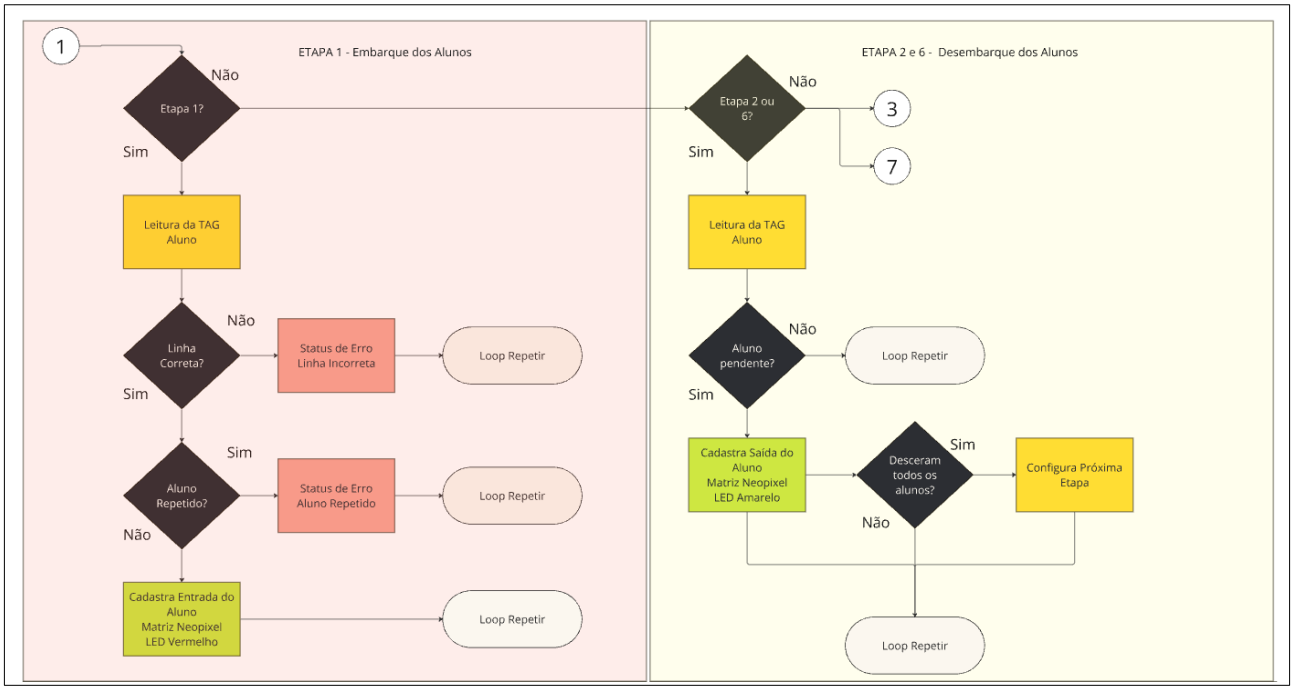

Figura 8: Fluxograma - Switch Case 1, 2 e 6

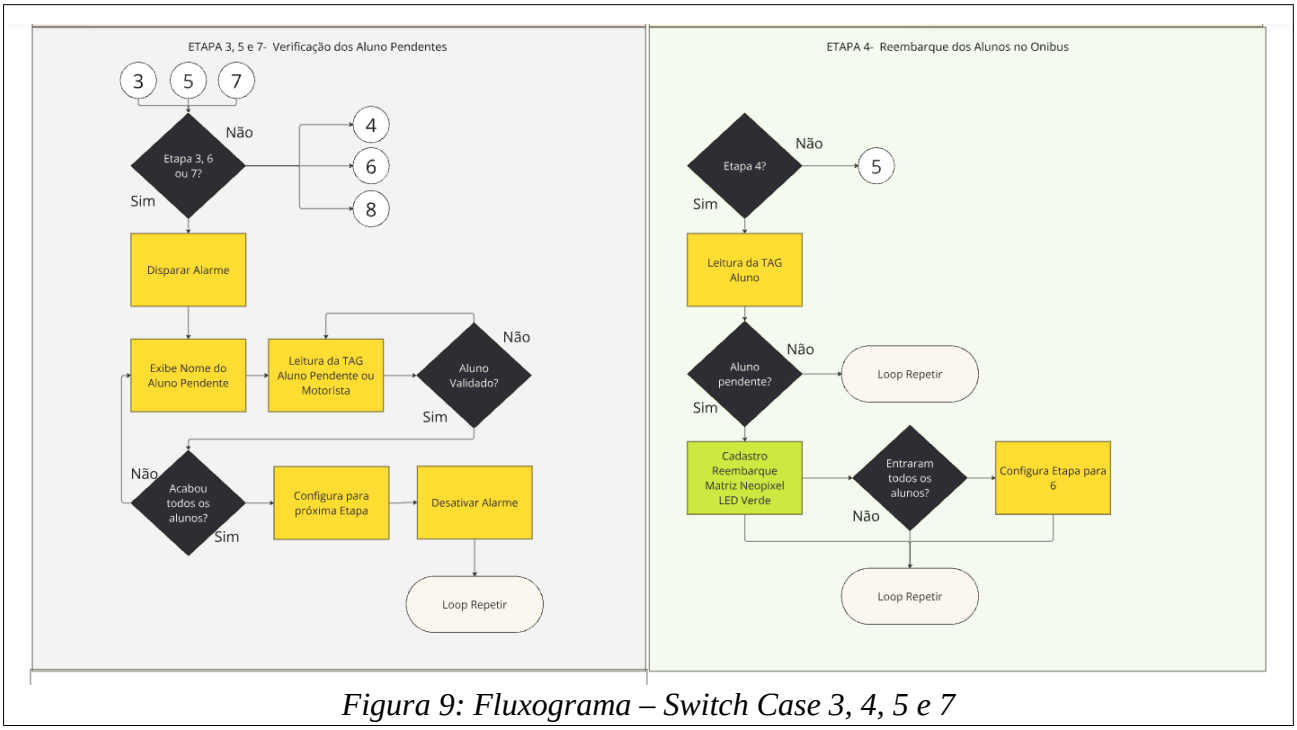

Figura 9: Fluxograma – Switch Case 3, 4, 5 e 7

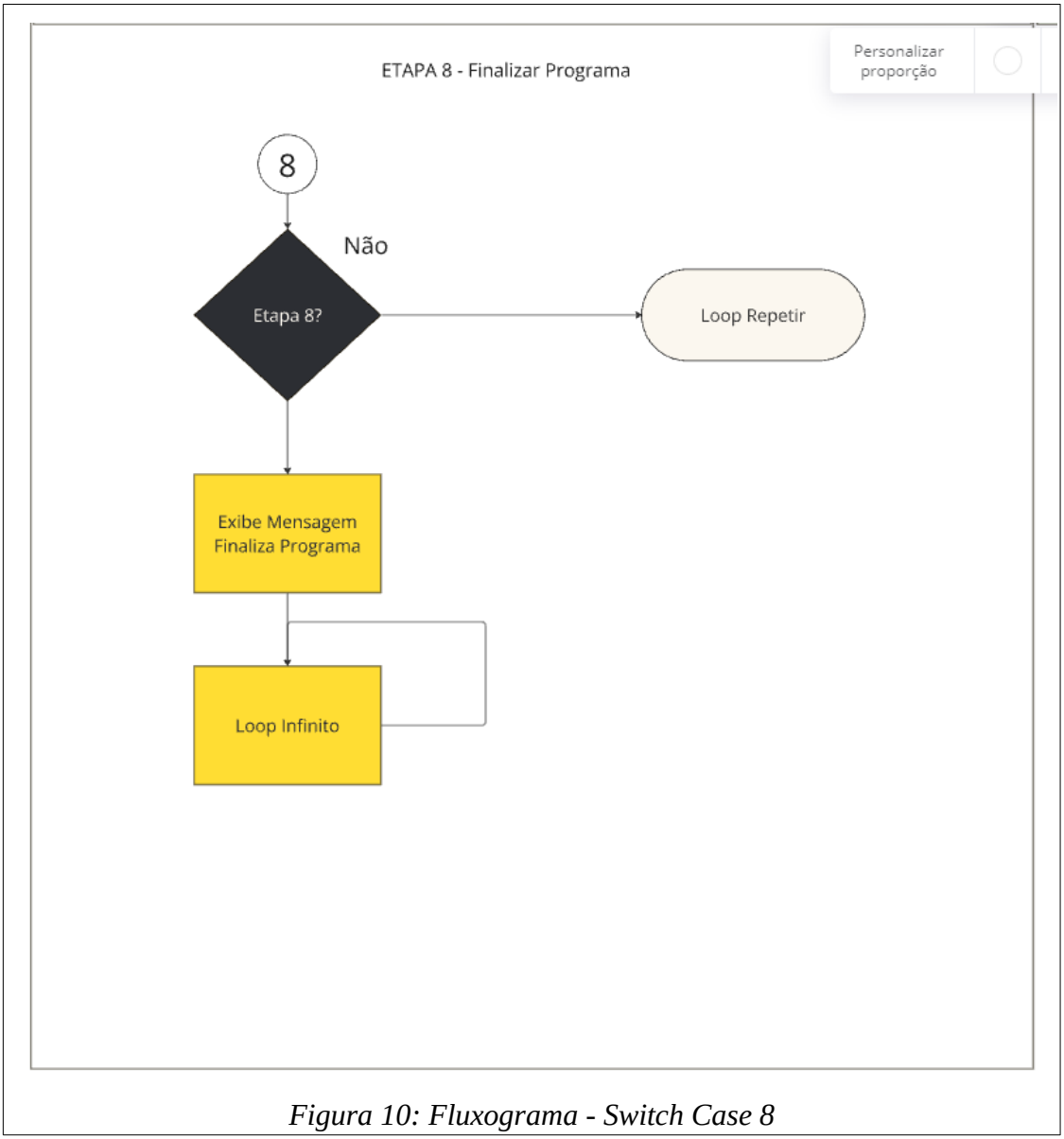

Figura 10: Fluxograma - Switch Case 8

Estrutura e formato dos dados

Durante o projeto, trabalhos com diversos tipos de dados. A principal fonte de informação dos dados vêm das TAG's lidas pelo leitor, esses dados são interpretados em bytes e são tratados de diferentes formas pelo software. Dentre os bytes da TAG, utilizamos:

4 bytes para ID (identificação) da TAG – esses dados são utilizados apenas para comparações

10 bytes contendo o nome do dono do cartão – esses dados são gravados em representação "ASCII" byte a byte, e estão preparados para serem escritos no display através de rotinas byte a byte.

4 bytes para controle da Linha / Percurso – esses dados também são utilizados para controle e não tem saída

Para contagem dos alunos são utilizadas variáveis do tipo inteiro, essas variáveis também são utilizadas para controle da matriz neopixel.
Outro tipo de variáveis bastante utilizadas no projeto são do tipo char, para armazenas as strings (mensagens) que são exibidas no display OLED.

Protocolo de Comunicação

Alguns protocolos de comunicação serial são utilizados no projeto:
I2C – Utilizado para comunicação do micro com o Display OLED para exibição de informações pertinentes. Utiliza 2 pinos para comunicação (Dados e Clock).
SPI – Utilizado para comunicação do micro com o leitor RFID, Utiliza 4 pinos para comunicação e controle (Dados de Saída, Dados de Entrada, Clock e Seleção do dispositivo).
Ainda temos a tecnologia RFID (por radiofrequencia) utilizada para fazer a leitura das etiquetas.

EXECUÇÃO DO PROJETO

Para execução do projeto foi definido a tecnologia RFID com etiquetas Mifare pelo baixo custo do hardware de leitura além das etiquetas que permitem uma implementação barata e com opção de personalização. A escolha pelas interfaces de saídas como os LEDs e buzzer tentaram trazer um tom de fácil utilização ao sistema. A definição das funcionalidades do software foram baseadas através das experiências do autor desse projeto em relação ao transporte escolar de sua filha e de conhecidos na cidade em que ele habita.

No desenvolvimento do projeto foi necessário utilizar um protoboard para ligação do leitor RFID de forma externa a placa da BitDogLab, conforme imagem abaixo.

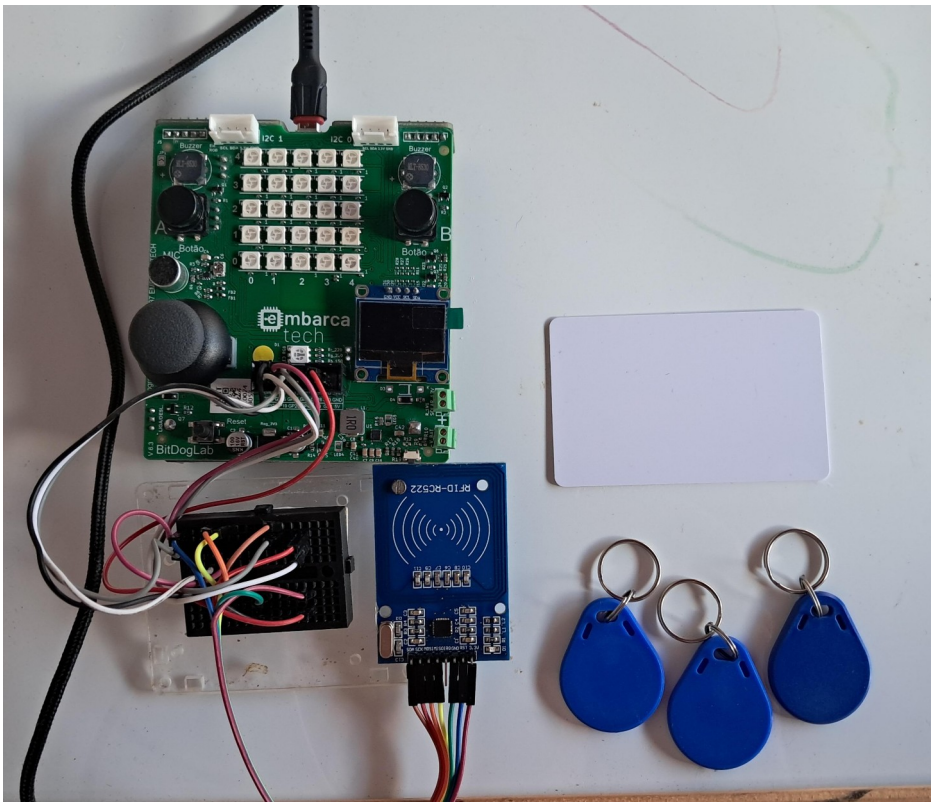

Figura 11: Projeto Final

Para desenvolvimento do projeto foi utilizado o VS Code como editor de código e utilizado o plugin SDK da Raspberry Pico na versão 2.1.0. O código foi desenvolvido em linguagem C e utilizado como principal forma de depuração o monitor serial do próprio VSCode para melhor visualização do comportamento do código como informações, dados das Tags e todas as informações pra conseguir desenvolver o código. Para validação final do projeto foi necessário configurar algumas Tag's com dados pré-determinados a fim de passar por todas as situações previstas. A configuração das TAG's foi feita através do Aplicativo MIFARE Classic Tool disponível na Play

Store em um smartphone com suporte a tecnologia NFC. Através do aplicativo era possível verificar se os dados de leitura estavam corretos e era possível alterar os dados das TAG's para os testes.

Os resultados dos testes em bancada se mostraram positivos, pois nessas situações o sistema embarcado conseguiu se comportar de maneira esperada, tanto em situações positivas, como em situações de erros e se mostrou bem confiável. Entretanto precisamos de testes em ambiente real e solucionar algumas condições que podem não ter ficado tão claras ainda, como por exemplo a criação de um sistema de armazenamento dos dados através de um SDCard ou na própria Flash da Raspberry Pi Pico W para poder desligar o equipamento e quando religar os dados estarem salvos (principalmente na condição do Reembarque já que o veículo ficará por um bom tempo com o motor desligado, o que dependendo do consumo do sistema poderia consumir muito a bateria do veículo a qual ele está conectado. Também poderiam haver otimizações para controles de mais de uma Linha simultânea.

Vídeo do Projeto disponível em:
https://youtu.be/i6PaPgTRhtI

Repositório do Projeto disponível em:
https://github.com/Netofranco/Embarcatech.git

REFERENCIAS

Pico SDK

https://www.raspberrypi.com/documentation/pico-sdk/

BitDogLab

https://github.com/BitDogLab/BitDogLab-C/tree/main

RP2040

https://datasheets.raspberrypi.com/rp2040/rp2040-datasheet.pdf

Simulador

https://wokwi.com/

Display OLED

https://cdn-shop.adafruit.com/datasheets/SSD1306.pdf

https://www.makerhero.com/blog/controlando-um-display-oled-com-a-biblioteca-ssd1306/?srsltid=AfmBOoosfW-zzEGQmDlxv9YAUFDiFzdztdONVBVsz-stovaT4_vFc6jz

Leitor RFID-RC522

https://github.com/BenjaminModica/pico-mfrc522/blob/main/mfrc522.c

https://blogmasterwalkershop.com.br/arduino/como-usar-com-arduino-kit-rfid-mfrc522

https://www.nxp.com/docs/en/data-sheet/MFRC522.pdf

Cartão RFID Mifare

https://www.nxp.com/products/rfid-nfc/mifare-hf/mifare-classic:MC_41863

https://www.nxp.com/docs/en/data-sheet/MF1S70YYX_V1.pdf

https://embarcados.com.br/rfid-cartoes-mifare/