Human Benchmark

Introdução:

O "Human Benchmark" consiste no desenvolvimento de um sistema embarcado destinado a medir o tempo de reação de um usuário a estímulos visuais e auditivos. O principal objetivo é avaliar a rapidez com que uma pessoa responde a determinados sinais, fornecendo feedback imediato sobre seu desempenho.

Para a implementação deste sistema, foram estabelecidos requisitos essenciais:

Requisitos Funcionais:

- **Medição do Tempo de Reação:** O sistema deve detectar o momento em que o usuário responde a um estímulo e calcular o intervalo de tempo decorrido.
- **Feedback Imediato:** Após cada tentativa, o dispositivo deve exibir o tempo de reação medido na tela OLED.

Requisitos Não Funcionais:

- **Usabilidade:** A interface deve ser intuitiva, permitindo que usuários de diferentes idades e habilidades utilizem o sistema sem dificuldades.

O funcionamento do sistema inicia com uma tela de boas-vindas, orientando o usuário a pressionar um botão para começar. Após a confirmação, uma contagem regressiva de três segundos é exibida, acompanhada por sinais sonoros e visuais. Em seguida, após um intervalo aleatório, um estímulo é apresentado, e o usuário deve responder o mais rápido possível pressionando o botão correspondente. O tempo decorrido entre o estímulo e a resposta é calculado e mostrado na tela, permitindo que o usuário avalie seu desempenho.

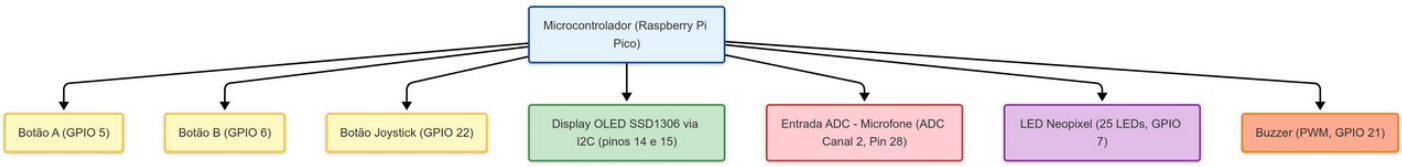
A justificativa para a realização deste projeto reside na crescente demanda por ferramentas que auxiliem no monitoramento e aprimoramento das capacidades cognitivas e motoras. Medir o tempo de reação é uma prática comum em diversas áreas, como esportes, medicina e ergonomia, sendo um indicador relevante da saúde neuromuscular e da prontidão mental. Além disso, o projeto serve como uma plataforma educacional para demonstrar a integração de diferentes componentes de hardware e software em sistemas embarcados.

Embora existam projetos similares disponíveis, como cronômetros de reação baseados em microcontroladores—por exemplo, o desenvolvimento de um periodímetro microcontrolado para aplicações em física experimental (REFERENCIA) —este projeto se destaca por sua abordagem didática e pela combinação específica de componentes utilizados, proporcionando uma experiência única tanto para o usuário quanto para o desenvolvedor.

Adicionalmente, este sistema pode ser particularmente útil para indivíduos com condições de saúde que afetam os reflexos, como doenças neurológicas ou decorrentes do envelhecimento. Ao medir o tempo de reação, é possível estimar o grau de comprometimento dos reflexos e monitorar a progressão ou melhoria da condição ao longo do tempo. Ferramentas que avaliam o tempo de reação fornecem informações sobre os reflexos e a velocidade de tomada de decisões, sendo úteis em avaliações de saúde para detectar condições como a doença de Parkinson.

Hardware:

Diagrama de blocos:



Função de Cada Bloco

Cada componente do sistema desempenha uma função específica na medição do tempo de reação do usuário. O **Raspberry Pi Pico** atua como a unidade de processamento central, gerenciando os periféricos e executando a lógica de controle do sistema. Os **botões físicos (GPIO 5, GPIO 6 e GPIO 22)** são usados para capturar a interação do usuário e registrar o tempo de resposta com precisão. O **display OLED SSD1306**, operando via barramento **I2C**, exibe mensagens de inicialização, contagem regressiva e o resultado do tempo de reação. O **microfone (ADC Canal 2, Pino 28)** é utilizado como fonte de entropia para a inicialização do gerador randômico do sistema, garantindo variação nos tempos aleatórios de estímulo. Além disso, possibilita futuras expansões do sistema para medições baseadas em estímulos sonoros. O **LED Neopixel (GPIO 7)** gera estímulos visuais sincronizados com os eventos do sistema, enquanto o **buzzer (PWM, GPIO 21)** emite sinais acústicos que auxiliam na marcação do tempo de resposta do usuário.

Configuração de Cada Bloco

A configuração dos componentes foi realizada considerando os requisitos de tempo real e eficiência do sistema:

- **Raspberry Pi Pico:** Configurado para operar com múltiplos periféricos simultaneamente, utilizando **GPIOs** para controle digital, **I2C** para comunicação serial com o display, **PWM** para controle de sinal sonoro e **ADC** para leitura de entrada analógica.
- **Botões (GPIO 5, GPIO 6, GPIO 22):** Configurados como **entradas digitais** com resistores **pull-up internos** ativados para garantir níveis de tensão estáveis e evitar leituras espúrias.
- **Display OLED SSD1306 (I2C, GPIO 14 e GPIO 15):** Configurado como **slave I2C**, operando a uma taxa de comunicação de **400 kHz** para garantir atualização rápida da interface gráfica.
- **Microfone (ADC, Pino 28 – Canal 2):** Configurado como **entrada analógica** utilizando o **Conversor Analógico-Digital (ADC) de 12 bits** do RP2040, permitindo medições precisas do sinal de entrada. O valor lido do microfone é utilizado como **seed do gerador randômico**, garantindo que os tempos aleatórios de exibição do estímulo não sigam um padrão fixo e sejam mais imprevisíveis.
- **LED Neopixel (GPIO 7):** Controlado por **modulação de largura de pulso (PWM Serial)**, utilizando um protocolo específico de comunicação que permite o controle individual de cada LED no barramento.

- **Buzzer (PWM, GPIO 21):** Operando em **modo PWM**, com frequência ajustável entre **1 kHz e 4 kHz**, permitindo a geração de sinais sonoros diferenciados para cada estado do sistema.

Especificações Técnicas e Atendimento aos Requisitos

O projeto foi desenvolvido para atender aos requisitos de tempo de resposta rápido e comunicação eficiente entre os componentes. O **Raspberry Pi Pico**, com seu processador **dual-core ARM Cortex-M0+ a 133 MHz**, garante baixa latência na execução do código e resposta imediata aos estímulos. O uso do **barramento I2C** permite uma comunicação eficiente com o display OLED sem comprometer os demais periféricos. O **ADC de 12 bits** proporciona medições de alta precisão, garantindo a captura exata de sinais analógicos. O microfone, além de permitir futuras expansões para estímulos sonoros, melhora a aleatoriedade da geração de tempos de reação, tornando o teste mais confiável. Os botões e o joystick foram configurados para minimizar ruídos elétricos, garantindo leituras consistentes. O uso de **PWM** para o controle do buzzer e dos LEDs Neopixel possibilita um controle eficiente dos sinais sonoros e visuais.

Esse conjunto de especificações garante que o sistema opere dentro dos requisitos estabelecidos, possibilitando a medição precisa do tempo de reação e proporcionando uma interface responsiva ao usuário.

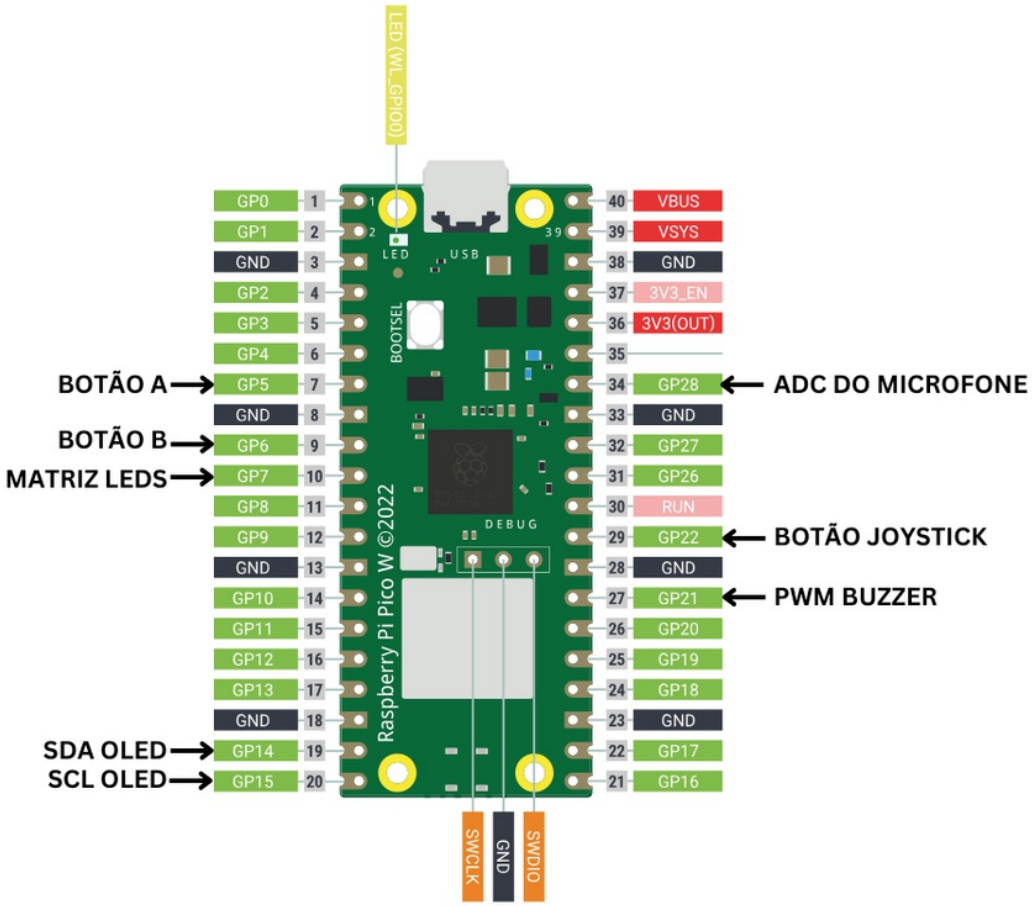
Lista de Materiais e Descrição

- **Raspberry Pi Pico W**
  Microcontrolador baseado no ARM Cortex-M0+ com 2MB de Flash e suporte a comunicação sem fio, utilizado como unidade de processamento central do sistema para gerenciar periféricos e processar os dados do tempo de reação.
- **Display OLED SSD1306**
  Tela monocromática de 128x64 pixels operando via barramento I2C a 400 kHz, utilizada para exibir informações como instruções, contagem regressiva e o tempo de reação do usuário.
- **Botões tácteis (3 unidades)**
  Interruptores mecânicos momentâneos conectados aos GPIOs do microcontrolador, responsáveis pela interação do usuário, permitindo iniciar o teste e registrar a resposta ao estímulo.
- **Buzzer Piezoelétrico**
  Componente que opera entre 3V e 5V, controlado via sinal PWM, utilizado para gerar alertas sonoros que indicam diferentes estados do sistema, como início do teste e exibição do resultado.
- **LED (25 LEDs)**
  Conjunto de LEDs RGB endereçáveis individualmente, controlados via protocolo WS2812, responsáveis por fornecer estímulos visuais sincronizados com os eventos do sistema.
- **Microfone Analógico**
  Sensor de captação de áudio com sensibilidade de -42 dBV/Pa, utilizado para leitura de sinais analógicos e como fonte de entropia para inicialização do gerador randômico do sistema, garantindo tempos de estímulo imprevisíveis.

Descrição da Pinagem Usada

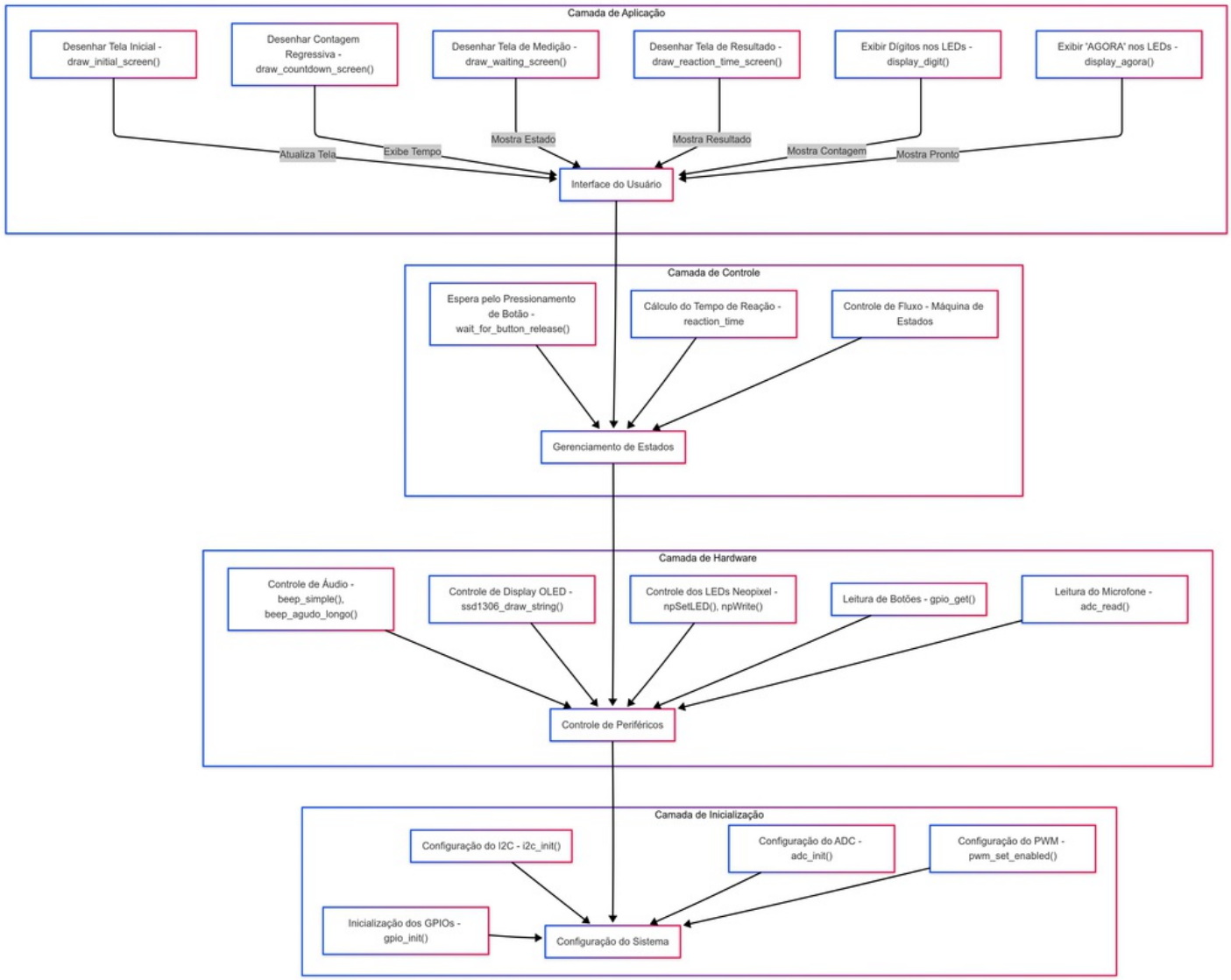
- **GPIO 5** – Entrada digital conectada ao **Botão A**, utilizado para iniciar o teste ou interagir com o sistema.
- **GPIO 6** – Entrada digital conectada ao **Botão B**, permitindo outra opção de interação do usuário.
- **GPIO 22** – Entrada digital conectada ao **Botão do Joystick**, utilizado para registrar a resposta ao estímulo.
- **GPIO 14 (SDA)** – Linha de dados do barramento **I2C**, utilizada para comunicação com o **Display OLED SSD1306**.
- **GPIO 15 (SCL)** – Linha de clock do barramento **I2C**, responsável pela sincronização da comunicação com o **Display OLED SSD1306**.
- **GPIO 28 (ADC Canal 2)** – Entrada analógica conectada ao **Microfone**, utilizado tanto para captação de áudio quanto como seed para o gerador randômico.
- **GPIO 7** – Saída digital conectada ao **LED Neopixel**, permitindo o controle individual dos LEDs via protocolo WS2812.
- **GPIO 21** – Saída PWM conectada ao **Buzzer**, utilizada para gerar sinais sonoros de diferentes frequências e durações.

Circuito:



Software:

Blocos funcionais:



Descrição de funcionalidades e definição das variáveis:

Camada de Aplicação

Descrição das funcionalidades:

- Gerencia a interface com o usuário, exibindo informações no display OLED e controlando os LEDs Neopixel.
- Monitora a interação do usuário por meio dos botões e do joystick.
- Atualiza a tela com mensagens visuais e exibe os tempos de reação.

Definição das variáveis:

- `ssd` → Buffer do display OLED.
- `linha1`, `linha2`, `texto_piscar` → Strings exibidas na tela.
- `x1`, `x2`, `y_texto`, `x_piscar`, `y_piscar` → Coordenadas para exibição no display.
- `countdown` → Controla a contagem regressiva antes do estímulo.

Camada de Controle

Descrição das funcionalidades:

- Implementa a máquina de estados do sistema, controlando a progressão das telas.
- Aguarda entradas do usuário e gerencia o tempo de reação.
- Calcula o tempo de resposta e decide as transições entre estados.

Definição das variáveis:

- `current_screen` → Estado atual do sistema (inicial, contagem regressiva, medição, resultado).
- `start_time`, `end_time` → Armazenam os tempos de início e fim da medição.
- `reaction_time` → Tempo de reação calculado do usuário.
- `exibir_piscar` → Controla o efeito de piscar na tela inicial.

Camada de Hardware

Descrição das funcionalidades:

- Controla os periféricos do sistema, incluindo buzzer, display OLED, LEDs Neopixel e botões.
- Gera estímulos sonoros e visuais para o usuário.
- Obtém leituras dos botões e sensores.

Definição das variáveis:

- `BUTTON_A`, `BUTTON_B`, `JOYSTICK_BUTTON` → Definem os pinos GPIO dos botões.
- `LED_PIN`, `LED_COUNT` → Configuram os LEDs Neopixel.
- `MIC_PIN`, `MIC_CHANNEL` → Definem a entrada analógica do microfone.
- `BUZZER_PIN` → Pino do buzzer para sinais sonoros.

Camada de Inicialização

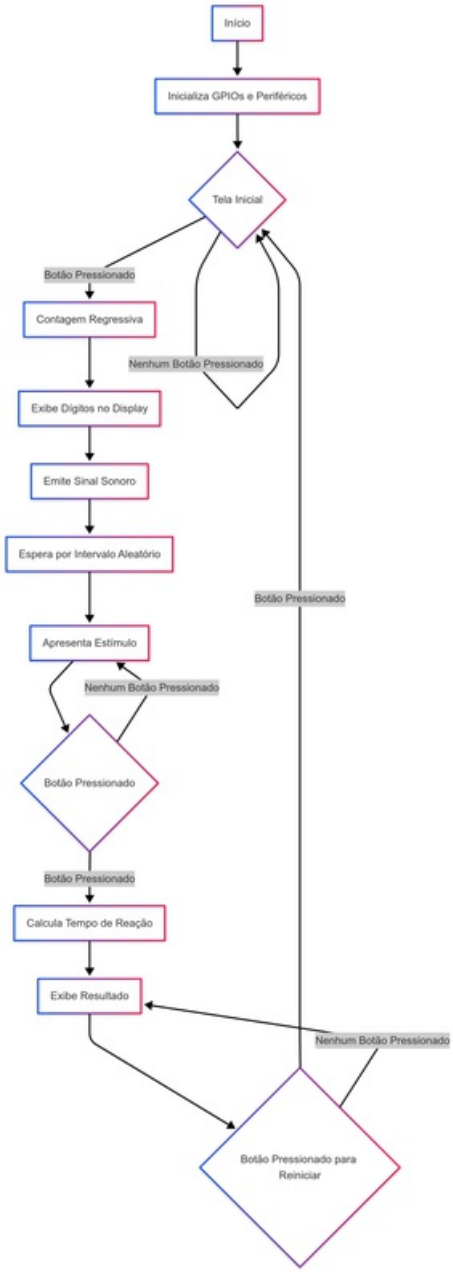
Descrição das funcionalidades:

- Configura os periféricos, incluindo GPIOs, ADC, PWM e I2C.
- Define as funções dos pinos de entrada e saída.
- Prepara o sistema para execução contínua.

Definição das variáveis:

- `ssd1306_i2c_clock` → Define a frequência da comunicação I2C.
- `frame_area` → Estrutura para renderização do display OLED.
- `last_press`, `last_blink` → Controlam os tempos de resposta e efeitos visuais.
- `slice_num`, `wrap`, `divider` → Configuram os sinais PWM do buzzer.

Fluxograma:



Inicialização – Processo de Inicialização do Software

O processo de inicialização ocorre dentro da função `main()`, onde os periféricos são configurados e as variáveis são inicializadas antes da execução do loop principal. Os seguintes passos são executados:

1. **Configuração dos GPIOs:**
    - Os botões (`BUTTON_A`, `BUTTON_B`, `JOYSTICK_BUTTON`) são configurados como entradas digitais com resistores de pull-up ativados (`gpio_pull_up()`).
    - O pino do buzzer (`BUZZER_PIN`) é configurado como saída PWM para geração de sinais sonoros.
    - Os LEDs Neopixel (`LED_PIN`) são inicializados para permitir o controle individual dos LEDs RGB.
2. **Inicialização dos periféricos:**
    - A interface I2C (`i2c1`) é configurada e habilitada para comunicação com o display OLED SSD1306.
    - O ADC é ativado (`adc_init()`) e configurado para ler o sinal do microfone conectado ao pino correspondente (`adc_select_input(MIC_CHANNEL)`).
3. **Inicialização das variáveis globais:**
    - `current_screen` é definido como `SCREEN_INITIAL`, estabelecendo o estado inicial do sistema.
    - `reaction_time`, `start_time` e `end_time` são inicializados para armazenar o tempo de reação do usuário.
    - O buffer do display `ssd` é zerado (`memset()`) antes da primeira atualização da tela.

Após essas etapas, o sistema entra em um **loop infinito**, onde a máquina de estados gerencia a execução e a transição entre diferentes telas e funcionalidades.

Configuração dos Registros – Funções de Configuração dos Registros

A configuração de registradores do microcontrolador RP2040 é realizada por meio das funções da **Pico SDK**, que ajustam o comportamento dos periféricos:

- **PWM (Controle do Buzzer):**
  - `pwm_set_clkdiv(slice_num, divider)` : Define o divisor de clock para ajustar a frequência do PWM.
  - `pwm_set_wrap(slice_num, wrap)` : Configura o período do PWM.
  - `pwm_set_gpio_level(pin, wrap / 2)` : Define o ciclo de trabalho do PWM.
- **GPIOs (Botões e LEDs):**
  - `gpio_init(pin)` : Inicializa os pinos GPIO necessários.
  - `gpio_set_dir(pin, GPIO_IN)` : Configura os botões como entradas digitais.
  - `gpio_pull_up(pin)` : Habilita resistores de pull-up internos para garantir leituras estáveis.
- **ADC (Leitura do Microfone):**
  - `adc_select_input(MIC_CHANNEL)` : Seleciona o canal ADC correspondente ao microfone.
  - `adc_read()` : Obtém a leitura do valor analógico e converte para digital utilizado para seed do gerador de números pseudorrandômicos.
- **I2C (Comunicação com o Display OLED SSD1306):**
  - `i2c_init(i2c1, ssd1306_i2c_clock * 1000)` : Inicializa o barramento I2C.
  - `gpio_set_function(14, GPIO_FUNC_I2C)` e `gpio_set_function(15, GPIO_FUNC_I2C)` : Configuram os pinos GPIO 14 e 15 como SCL e SDA.

Estrutura e Formato dos Dados – Dados Específicos Usados no Software

O software utiliza diferentes tipos de dados para armazenamento e processamento:

- **Enumeração dos Estados do Sistema:**

  ```
  enum Screen { SCREEN_INITIAL, SCREEN_COUNTDOWN, SCREEN_MEASURING, SCREEN_RESULT };
  ```

  - Define os estados da máquina de estados do software.
- **Buffer do Display OLED:**

  ```
  uint8_t ssd[ssd1306_buffer_length];
  ```

  - Armazena os dados que serão renderizados na tela OLED antes de serem transmitidos via I2C.
- **Variáveis de Tempo:**

  ```
  absolute_time_t start_time, end_time; float reaction_time;
  ```

  - Armazena os tempos de início e término da medição, além do tempo de reação do usuário.
- **Dados para Controle dos LEDs Neopixel:**

  ```
  uint8_t pattern[LED_COUNT];
  ```

  - Define padrões de exibição de LED para representar números ou estados do sistema.

Organização da Memória – Endereços de Memória Usados

A memória é organizada conforme a seguinte estrutura:

- **RAM:**
  - **Variáveis globais e buffers** são armazenados na RAM, incluindo `ssd`, `reaction_time` e `current_screen`.
  - **Buffers temporários** usados na comunicação I2C e controle de LEDs.
- **Flash (ROM):**
  - Código do programa armazenado na memória flash.
  - Strings de mensagens fixas (`"Pressione algo"`, `"AGORA"`) são mantidas na memória de programa.
- **Registradores do Hardware:**
  - **GPIOs, PWM, ADC e I2C** são configurados e manipulados diretamente nos registradores do RP2040, via chamadas da Pico SDK.

Protocolo de Comunicação – Descrição do Protocolo

O sistema utiliza **I2C** para comunicação entre o microcontrolador e o display OLED SSD1306.

- **Configuração do I2C:**
  - Interface configurada como mestre (`i2c1`).
  - Taxa de comunicação definida (`ssd1306_i2c_clock * 1000`).
  - Pinos GPIO 14 (SDA) e 15 (SCL) utilizados para transmissão de dados.
- **Processo de Comunicação:**
  - O microcontrolador envia comandos e dados gráficos ao display via I2C.

- O barramento segue o protocolo padrão: `Start -> Endereço do Slave -> Comando/Dados -> Stop` .

- **Formato do Pacote de Comando:**

  `[Start] [Endereço do Display] [Byte de Controle] [Byte de Comando] [Stop]`

  - Utilizado para configurar propriedades do display, como brilho e rolagem de tela.

- **Formato do Pacote de Dados:**

  `[Start] [Endereço do Display] [Byte de Controle] [Dados da Imagem] [Stop]`

  - Contém os bits de imagem/texto a serem exibidos.

Formato do Pacote de Dados – Formação dos Pacotes

Os pacotes de dados transmitidos seguem o formato especificado pelo protocolo I2C:

- **Comandos de Controle (Pacote de 2 bytes)**

```
uint8_t command[] = {0x00, 0xAE}; // 0x00 = Byte de Controle, 0xAE = Comando Display OFF

i2c_write_blocking(i2c1, SSD1306_ADDRESS, command, 2, false);
```

  - O primeiro byte (`0x00`) indica que se trata de um comando.
  - O segundo byte (`0xAE`) é o comando específico (nesse caso, desligar o display).

- **Envio de Dados para Renderização (Pacote de múltiplos bytes)**

```
uint8_t data_packet[128];  // Buffer de dados para um segmento da tela

data_packet[0] = 0x40;      // Byte de Controle para indicar dados

memcpy(&data_packet[1], buffer, 127);

i2c_write_blocking(i2c1, SSD1306_ADDRESS, data_packet, 128, false);
```

  - O primeiro byte (`0x40`) indica que os bytes seguintes contêm dados de imagem.
  - O restante do pacote transmite os pixels a serem exibidos.

Execução do projeto:

A metodologia aplicada na execução deste projeto seguiu uma abordagem estruturada baseada no desenvolvimento modular e na integração progressiva dos componentes de hardware e software. Inicialmente, foi realizada a especificação dos requisitos funcionais e não funcionais, garantindo que todas as funcionalidades estivessem alinhadas com os objetivos do sistema. A implementação seguiu uma abordagem bottom-up, onde os módulos de hardware foram configurados e testados individualmente antes da integração com a lógica principal do sistema. A configuração do barramento I2C foi validada isoladamente para garantir a comunicação correta entre o microcontrolador RP2040 e o display OLED SSD1306. Da mesma forma, os sinais de entrada e saída foram testados separadamente, incluindo a resposta dos botões, o acionamento do buzzer via PWM e a geração de padrões luminosos nos LEDs Neopixel. A lógica do software foi estruturada utilizando uma máquina de estados finitos, garantindo transições bem definidas entre os diferentes estágios do sistema, desde a tela inicial até a exibição dos resultados de tempo de reação.

Os testes de validação foram conduzidos de maneira empírica, executando o dispositivo pelo menos 30 vezes para garantir a precisão e a confiabilidade do sistema. Essa abordagem permitiu identificar e corrigir bugs e comportamentos inesperados que poderiam comprometer a funcionalidade do sistema. A resposta dos botões foi avaliada verificando a detecção de pressões sucessivas e garantindo que o debounce estivesse adequado para evitar leituras inconsistentes. A funcionalidade do display foi testada enviando diferentes padrões gráficos e mensagens para confirmar a correta renderização das informações. O tempo de reação do usuário foi medido comparando os valores registrados pelo software com medições externas realizadas por um cronômetro digital, assegurando que os cálculos internos estivessem corretos e coerentes. O buzzer foi testado para confirmar a geração dos diferentes tons necessários para os alertas sonoros e sua sincronização com os estímulos visuais apresentados. Durante os testes, foi possível verificar se o comportamento do sistema era consistente em diferentes execuções e se a apresentação dos estímulos ocorria de maneira uniforme.

A análise dos resultados demonstrou que o sistema foi capaz de medir com precisão o tempo de reação do usuário, apresentando variações consistentes entre diferentes execuções. O desempenho do sistema mostrou-se adequado, com transições suaves entre os estados da máquina de estados e uma interface responsiva às interações do usuário. Durante os testes, foi observado que o tempo de reação dos participantes variou conforme o estímulo apresentado, validando a funcionalidade do sistema para medir tempos de resposta neuromotores. Além disso, a repetição dos testes possibilitou a identificação de falhas sutis, como pequenas inconsistências na detecção do pressionamento dos botões e na sincronização dos estímulos sonoros e visuais, permitindo ajustes para aumentar a confiabilidade do sistema.

Diante dos resultados obtidos, conclui-se que o sistema atende aos requisitos estabelecidos e demonstra grande potencial para aplicações práticas na avaliação de tempos de reação humana. A capacidade de medir com precisão a resposta a estímulos visuais e sonoros, aliada à facilidade de uso e portabilidade do dispositivo, faz com que este projeto tenha aplicabilidade em diversas áreas, incluindo pesquisa acadêmica, esportes e avaliações médicas. A modularidade do sistema permite futuras expansões e melhorias, como a inclusão de armazenamento de dados para análise estatística ou a integração com redes sem fio para monitoramento remoto. Além disso, a abordagem utilizada na implementação deste sistema pode servir de referência para projetos semelhantes, reforçando a importância de uma metodologia estruturada no desenvolvimento de sistemas embarcados. Assim, o projeto demonstrou ser uma ferramenta eficaz para medições de tempo de reação, proporcionando uma base sólida para aprimoramentos e aplicações futuras.

Vídeo demonstrativo: https://www.youtube.com/watch?v=UB9qdzY9bRk Repositório: https://github.com/PedroSampaioDias/Human-Benchmark