Harmonichip com BitDoglab

1.Escopo do projeto

1.1. Apresentação do projeto: Este projeto consiste em um sistema embarcado capaz de identificar notas musicais e suas frequências em tempo real. Utilizando um microfone DMA, ele captura o som ambiente, processa os dados por meio de FFT e exibe no display a nota correspondente e sua frequência. Além disso, uma matriz de LEDs indica a intensidade do som detectado. O sistema também conta com um buzzer que, ao ser ativado, reproduz a melodia de "Parabéns", permitindo que o microfone registre e exiba as informações da música simultaneamente.

1.2. Objetivos do projeto:

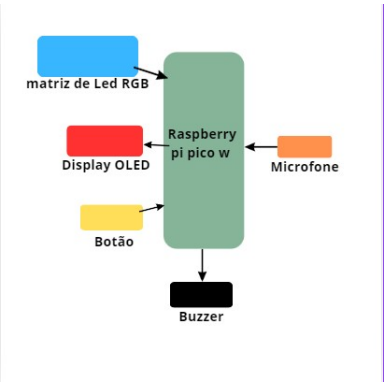
- Detectar e identificar cada nota de uma melodia com precisão.
- Compreender e evidenciar a intensidade característica de cada nota.
- Ajustar a frequência exata correspondente a cada nota.

1.3. Descrição do Funcionamento: Este projeto se justifica por sua utilidade no estudo e análise de sons em tempo real, auxiliando na identificação de notas musicais, frequências e intensidades sonoras. Sua abordagem interativa facilita o aprendizado, a afinação de instrumentos e a experimentação acústica.

1.4. Originalidade: Embora existam projetos semelhantes que utilizam a placa BitDogLab para reconhecimento de notas, como o "Notas Musicais" no GitHub, este projeto se diferencia por integrar a identificação de notas em tempo real, exibição de frequências no display, intensidade sonora em LEDs e reprodução da melodia de "Parabéns" com buzzer, permitindo a captura e exibição simultânea das informações. Isso torna o sistema mais interativo e completo.

2. Especificação do Hardware

2.1. Diagrama em Bloco:



2.2. Função de Cada Bloco:

- Bloco azul: Mostrar intensidade do som.
- Bloco vermelho: Imprimir as informações.
- Bloco amarelo: Controle manual via botões.
- Bloco preto: Emitir som.
- Bloco laranja: Capturar som.

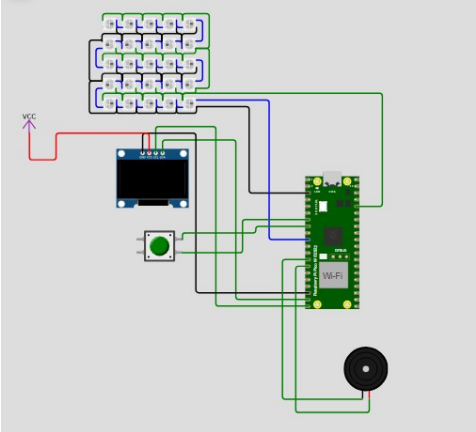
2.2. Configuração de Cada Bloco:

- Configuração do microfone para capturar sons externos e a sensibilidade.
- Configuração da Matriz de Led RGB mostrar a intensidade do som em relação a frequência recebida.
- Configuração do Buzzer para emitir a música dos parabéns com frequência e tempo determinados em cada nota.
- Botão configurado para não ficar em ponto flutuante, e ativar a musica no buzzer por meio de interrupção.
- Implementação de caracteres letras(maiúsculas e minúsculas) e números no display OLED. para imprimir
- O recurso multicore na Raspberry Pi Pico permite que o microcontrolador utilize seus dois núcleos de processamento (CPUs) de forma independente, executando tarefas paralelamente.

2.3. Descrição de Pinagem Usada:

- Pinos de entrada (microfone, botão).
- Pinos de saída (Matriz Led RGB, display OLED e buzzer).

2.4. Circuito completo do Hardware:



3. Especificação do Firmware

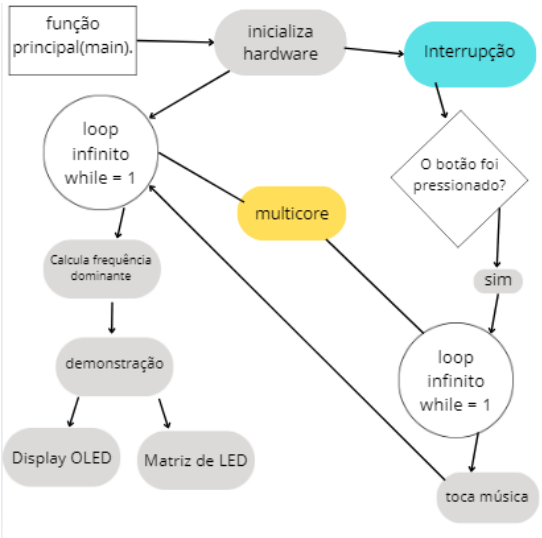
3.1. Blocos funcionais:

- Captura de som.
- Calculo da frequência dominante(processamento de sinais).
- Interrupção.
- Função multicore.
- Controle de saídas de som.

3.2. Blocos funcionais:

- Captura do som de forma contínua.
- Criar efeitos do som na matriz de LED.
- Permite que o microcontrolador responda rapidamente a eventos externos.
- Executar tarefas simultaneamente em ambos os núcleos (dois loops infinitos ao mesmo tempo).
- Controle manual por meio de botão.

3.3. Fluxograma:



3.4. Inicialização:

- Configuração de hardware e periféricos.

3.5. Configurações dos Registros:

- Definição de modo dos pinos.
- Ajuste de timers e interrupções.

3.6. Estrutura e Formato dos Dados:

- Dados em formato de sinais analógicos convertidos para digital.

3.7. Protocolo de Comunicação:

- O protocolo pode ser baseado em I2C.

4. Execução do projeto

4.1. Metodologia:

4.1.1. Pesquisas Realizadas:

- Transformada Rápida de Fourier (FFT).
- Microfones compatíveis com microcontroladores.
- Matrizes de LEDs e displays OLED.
- Protocolos de comunicação.

4.1.2. Escolha do Hardware:

- Raspberry Pi Pico.
- Microfone.
- Display OLED 128x64 (I2C).
- Matriz de LEDs WS2812B.
- Fonte de alimentação e circuitos auxiliares.

4.1.3. Definição das Funcionalidades do Software:

- Captura de áudio.
- Processamento do sinal com FFT.
- Exibição no display OLED.
- Controle da matriz de LEDs.
- Comunicação e otimização.

4.1.4. Inicialização da IDE:

- pico-sdk.
- hardware_adc, hardware_pwm, hardware_pio, hardware_I2C, hardware_dma, hardware_timer e hardware_clocks.
- ssd1306.
- Neopixel.
- pico_multicore.

4.1.5. Programação na IDE:

- Configuração do microfone e leitura do sinal de áudio.
- Implementação da FFT para análise do espectro sonoro.

- Exibição dos resultados no display OLED.
- Controle dos LEDs para indicar níveis de intensidade sonora.
- Saída de som no buzzer.

4.1.6. Depuração:

- Prints no terminal da IDE.
- Osciloscópio.
- Testes práticos
- Ajustes de parâmetros da FFT.

5. Referências

- **Datasheet do Raspberry Pi Pico**
  Disponivel em: https://datasheets.raspberrypi.com/pico/pico-datasheet.pdf

- **Blog do Raspberry Pi**
  Disponivel em: https://www.raspberrypi.org/blog/

- **SSD1306 (Display OLED)**
  Disponivel em: https://github.com/adafruit/Adafruit_SSD1306

- **Neopixel (Matriz de LEDs)**
  Disponivel em: https://github.com/adafruit/Adafruit_NeoPixel

- **Documentação do Pico SDK**
  Disponivel em: https://github.com/raspberrypi/pico-sdk

- **Repositório da Bit Dog Lab**
  Disponivel em: https://github.com/BitDogLab

6. Entrega conforme requisitado

- Repositório do projeto no GitHub: https://github.com/LuizEduardo-cyber/Projeto-final.git
- O vídeo de demonstração pode ser acessado em: https://drive.google.com/file/d/163ae8SIDxSiBRcRlbWKiaSp3NkjzKJjl/view?usp=drive_link