INSTITUTO HARDWARE BR PROGRAMA EMBARCATECH

PROJETO FINAL - UNIDADE 7

TÍTULO:

Aplicação Didática dos Recursos da BitDogLab no Ensino de Sistemas Embarcados

Aluno:

Gabriel da Conceição Miranda

Brasília - DF
Fevereiro/2025

Sumário

1. Escopo do Projeto

O projeto consiste no desenvolvimento de um software bem documentado e didático, voltado para auxiliar o Projeto Escola 4.0 da Unicamp e outros programas educativos que utilizam a BitDogLab como ferramenta de ensino. A BitDogLab é uma plataforma educacional baseada na Raspberry Pi Pico H ou W, projetada para proporcionar uma experiência prática de aprendizado em eletrônica e computação. Com componentes integrados e organizados de forma segura, ela permite que estudantes explorem conceitos de programação e sistemas embarcados de maneira progressiva e intuitiva.

Esse projeto tem um papel fundamental na modernização do ensino, especialmente no Brasil, onde a formação de profissionais na área tecnológica ainda enfrenta desafios. Ao trazer experiências com novas tecnologias para o sistema educacional básico, iniciativas como esta preparam os alunos para o futuro, incentivando o desenvolvimento de habilidades essenciais para a inovação e a independência tecnológica. Em um cenário onde grande parte das tecnologias utilizadas no país são importadas, projetos como o BitDogLab contribuem para fomentar o conhecimento e a capacitação local, preparando estudantes para atuar no desenvolvimento de novas soluções tecnológicas.

Com a crescente demanda por profissionais qualificados em tecnologia e engenharia, o projeto busca tornar o ensino mais acessível, interativo e eficaz, conectando teoria e prática de forma integrada. Assim, ele não apenas fortalece o Projeto Escola 4.0, mas também amplia seu impacto em outras iniciativas educacionais, consolidando a importância da eletrônica e dos sistemas embarcados no aprendizado de jovens e futuros profissionais.

O principal objetivo deste projeto é desenvolver um exemplo didático que vá além de uma simples demonstração dos componentes da BitDogLab. Ele deve possuir utilidade prática, permitindo que os alunos compreendam e experimentem os principais recursos da placa de maneira integrada e aplicada. O projeto deve explorar simultaneamente diversos periféricos da BitDogLab, incluindo buzzer, microfone, matriz de LEDs RGB 5x5, LED RGB, display OLED, joystick e botões, além de demonstrar comunicações via USB, UART e I2C. Além disso, conceitos mais avançados como DMA (Direct Memory Access), PIO (Programmable IO) com máquinas de estado, conversão analógico-digital (ADC), PWM (duty cycle, wrap e divisão de clock) e temporização também serão abordados. Outro aspecto fundamental do projeto é demonstrar a utilização de bibliotecas na linguagem C/C++, permitindo que os alunos compreendam como importar, configurar e utilizar códigos reutilizáveis para facilitar o desenvolvimento de aplicações embarcadas.

Como requisitos, o exemplo desenvolvido deve ser claro, bem documentado e reprodutível, possibilitando que estudantes que possuam a BitDogLab ou desejem montar um hardware similar adquirindo os componentes individualmente consigam replicar e aprender com a implementação. O projeto foi pensado especialmente para ser aplicado em programas educativos que utilizam a BitDogLab, garantindo uma experiência prática e estruturada com a plataforma. No entanto, essa restrição não impede que qualquer interessado explore o projeto, já que ele pode ser reproduzido por aqueles que optarem por montar sua própria estrutura de hardware com os componentes necessários, tornando o aprendizado mais acessível e flexível.

O projeto consiste no desenvolvimento de um aplicativo musical interativo que utiliza os recursos da BitDogLab para oferecer uma experiência didática e prática. Ao ligar o dispositivo, o display OLED exibe um menu interativo que permite ao usuário navegar entre as funcionalidades disponíveis utilizando o joystick. Atualmente, duas funções principais foram implementadas: um afinador de instrumentos musicais e um treinamento de ouvido. A seleção entre essas opções é feita movimentando o joystick para os lados, enquanto a confirmação ocorre ao pressionar o botão do joystick.

O afinador utiliza o microfone da BitDogLab para capturar periodicamente o som ambiente. O sistema calcula a frequência predominante do som captado e a associa à nota musical mais próxima, além de exibir a diferença entre a frequência detectada e a nota de referência. Essas informações são apresentadas no display OLED, sendo constantemente atualizadas, permitindo que o usuário ajuste a afinação do instrumento com precisão. Para validar o funcionamento do afinador, foram realizados testes utilizando instrumentos musicais afinados, aplicativos geradores de notas e outros afinadores de referência, garantindo a confiabilidade dos cálculos. Caso deseje retornar ao menu principal, o usuário pode sair da funcionalidade pressionando

novamente o botão do joystick.

Na funcionalidade de treinamento de ouvido, o usuário pode desenvolver sua percepção auditiva e aprimorar a familiaridade com notas musicais. O display OLED inicialmente exibe uma nota e sua frequência correspondente, junto com instruções sobre a atividade. Para indicar o início do exercício, a matriz de LEDs 5x5 emite um alerta visual, preparando o usuário para a reprodução sonora. Em seguida, os buzzers tocam as notas musicais uma de cada vez, e uma seta amarela aparece na matriz de LEDs, apontando para o buzzer ativo no momento, facilitando a identificação do som emitido.

Se houver uma segunda BitDogLab conectada via UART, a funcionalidade se expande automaticamente. Nesse caso, os buzzers e a matriz de LEDs 5x5 da segunda placa também participam da atividade, tornando o exercício mais desafiador ao dobrar a quantidade de sons possíveis. O display OLED da segunda BitDogLab também espelha as informações da primeira, garantindo sincronia entre os dispositivos. O usuário deve então pressionar o botão correspondente ao buzzer que tocou a nota indicada no display OLED (o botão que fica logo abaixo do buzzer que tocou a nota). Após a escolha, a matriz de LEDs exibe a resposta correta, utilizando uma seta verde para indicar acerto e uma seta vermelha para indicar erro. O resultado também é apresentado no display OLED.

Caso o usuário esteja utilizando duas BitDogLab, a comunicação entre os dispositivos permite que qualquer botão pressionado em uma das placas seja reconhecido imediatamente pela outra. Para sair da funcionalidade de treinamento de ouvido, basta manter o joystick pressionado durante a resposta. Se houver uma segunda BitDogLab conectada, a comunicação UART será encerrada automaticamente, e a segunda placa retornará ao modo de busca por conexão. Se a conexão física entre os dispositivos for mantida, a comunicação poderá ser restabelecida automaticamente caso o usuário selecione novamente o modo de treinamento. Com a utilização de uma segunda placa, o nível de dificuldade aumenta, pois o número de opções cresce, reduzindo a probabilidade de acerto e tornando o exercício mais desafiador. Embora essas funcionalidades tenham aplicações práticas no mundo real, o principal objetivo do projeto é demonstrar a utilização integrada dos recursos da BitDogLab.

A justificativa para a realização deste projeto está diretamente relacionada à necessidade de oferecer materiais didáticos mais completos e acessíveis para o aprendizado com a BitDogLab e a Raspberry Pi Pico W. Essas tecnologias são relativamente novas e muito bem projetadas para o ensino de eletrônica e programação, porém, ainda existem poucos exemplos disponíveis que auxiliem os alunos a utilizá-las de maneira fluida e eficiente. Embora já exista um repositório de programas para a BitDogLab, muitos exemplos são incompletos ou pouco explicativos, dificultando a compreensão e a aplicação prática dos conceitos, isso será mostrado mais abaixo. Como a BitDogLab foi desenvolvida especificamente para fins educativos, a proposta deste projeto se alinha perfeitamente com seu propósito, contribuindo para tornar o aprendizado mais acessível e estruturado.

Além disso, durante o desenvolvimento do projeto, foi possível observar que muitos alunos enfrentavam dificuldades para configurar bibliotecas externas e utilizar corretamente os recursos da placa. Algumas das dúvidas frequentemente encontradas incluíam questões como:

- "Estou há um tempo tentando configurar as bibliotecas externas, como a ssd1306, e isso está sendo bem difícil, pois encontro diversos tutoriais, mas às vezes funciona e outras não. Existe algum projeto que já contenha as bibliotecas com o CMakeLists bem configurado, onde a parte principal seja o código do projeto final?"

- "Alguém conseguiu ligar toda a matriz de LEDs?"

- "Pessoal, estou com uma dúvida básica. Estou tentando compilar um projeto que usa o Display OLED, mas aparece a mensagem de erro 'hardware/i2c.h: No such file or directory'. Alguém saberia informar como resolver?"

Esses questionamentos demonstram que há uma demanda real por projetos bem documentados e estruturados que facilitem a adoção da BitDogLab no ensino. Este projeto se justifica justamente por fornecer um exemplo prático, detalhado e funcional, que não apenas exemplifica o uso dos componentes da BitDogLab, mas também oferece um código bem organizado, com bibliotecas corretamente configuradas e explicações claras, permitindo que novos alunos avancem com mais facilidade em seus estudos.

Este projeto se destaca por sua originalidade ao integrar diversos recursos da BitDogLab em um único aplicativo didático e funcional. Como mencionado anteriormente, já existe um repositório de exemplos de programas para a BitDogLab, mas eles são, em sua maioria, fragmentados e abordam os componentes de forma isolada. Isso dificulta a compreensão dos alunos sobre como combinar diferentes funcionalidades em um projeto coeso e aplicado a um contexto real.

A necessidade de um projeto mais abrangente pode ser observada em questionamentos frequentes dos alunos, como:

- "Pessoal, por exemplo, para juntar dois arquivos seria possível, por exemplo, joystick com LED e outro arquivo com display? Alguém sabe como fazer isso ou terei que criar um arquivo com todas essas funções?"
- "Boa tarde @Timóteo Altoé e @Fabiano Fruett, vocês poderiam pedir para o Embarcatech disponibilizar um ebook mais completo sobre a parte de interfaces de comunicação? Os PDFs de vocês estão bem legais e com muitos exemplos, mas o de comunicação foi o único que ficou meio 'pobre'. Vocês poderiam preparar um segundo ebook com vários exemplos práticos como os ebooks dos outros temas?"

Esses comentários indicam que os exemplos disponíveis atualmente não oferecem uma didática plena aos alunos. O grande diferencial deste projeto está na abordagem integrada, pois ele explora simultaneamente os principais recursos da BitDogLab em um único programa, ao invés de tratá-los separadamente. Além disso, sua aplicação prática no contexto musical confere um propósito real ao aprendizado, tornando a experiência mais significativa e engajadora. Outro aspecto inovador do projeto é a implementação da comunicação entre duas BitDogLabs via UART. Nenhum dos exemplos disponíveis no repositório atual demonstra essa funcionalidade, o que limita a compreensão dos alunos sobre a integração entre múltiplos dispositivos embarcados. A proposta desenvolvida não apenas incorpora essa comunicação, mas também detalha a montagem do arquivo CMakeLists.txt para a utilização conjunta de vários recursos da placa, facilitando o aprendizado sobre estruturação de projetos embarcados mais complexos.

Dessa forma, o projeto não apenas preenche lacunas na documentação existente, mas também proporciona um exemplo prático e bem documentado, permitindo que os alunos compreendam melhor a BitDogLab e suas possibilidades de aplicação no mundo real.

2. Hardware

A parte de hardware deste projeto destaca-se pela integração de três componentes essenciais que se complementam de forma única. No centro dessa configuração está o microprocessador RP2040, desenvolvido pela Raspberry Pi, que reúne dois núcleos ARM Cortex-M0+ operando até 133 MHz, 264 KB de memória SRAM distribuída em bancos independentes e suporte para memória flash externa. Essa combinação oferece desempenho robusto e flexibilidade para diversas aplicações em sistemas embarcados, tornando-o ideal para lidar com múltiplas tarefas e interfaces simultâneas. Complementando o RP2040, a placa Raspberry Pi Pico W incorpora esse microcontrolador e amplia suas capacidades ao adicionar conectividade Wi-Fi 2.4 GHz – e, opcionalmente, Bluetooth –, mantendo a compatibilidade com acessórios e facilitando o desenvolvimento de soluções IoT. Por fim, o BitDogLab se apresenta como uma plataforma de hardware de código aberto. Baseada nas placas Raspberry Pi Pico (H ou W), ela integra diversos componentes e periféricos de maneira inovadora, permitindo que os usuários experimentem e desenvolvam projetos de forma integrada e colaborativa. As especificações dos periféricos da placa BitDogLab podem ser observadas no material complementar [1], assim como a descrição mais aprofundada dos componentes de hardware, envolvendo o processador RP2040 e Raspberry Pi Pico W [2].

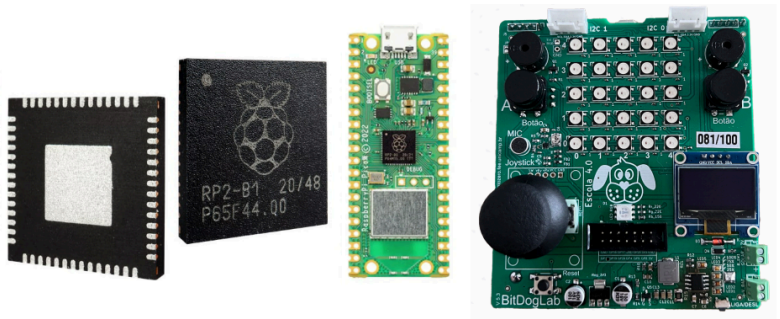

Figura 1 - Processador RP2040, Raspberry Pi Pico W e BitDogLab.

A Figura 2 ilustra os elementos de hardware da BitDogLab empregados no projeto, destacando-se quatro categorias de blocos, cada uma sinalizada por uma cor específica. Em verde, estão os componentes de saída de sinal, responsáveis por comunicar informações ao usuário durante a execução do programa. Em azul, encontram-se os elementos de entrada, por meio dos quais o usuário interage com o aplicativo musical. Em marrom, aparecem os recursos fundamentais incorporados diretamente à Raspberry Pi Pico W, essenciais para o funcionamento geral do sistema. Por fim, em laranja, há um bloco de comunicação bidirecional, capaz de tanto enviar quanto receber sinais. A seguir está um detalhamento da função e configuração utilizada em cada bloco.

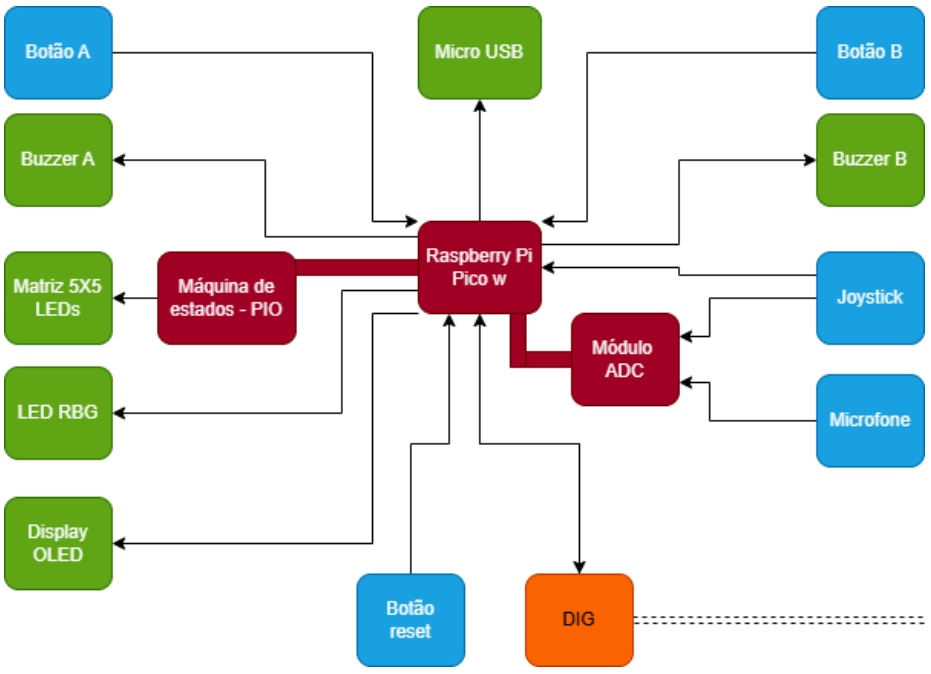

Figura 2 - Diagrama de blocos do hardware utilizado no projeto.

Botões A e B: os botões são utilizados para registrar o palpite do usuário. Como cada botão está fisicamente próximo a um buzzer, eles representam uma ferramenta ideal para essa funcionalidade. A configuração dos botões é realizada por meio da função setup_buttons(), que inicializa seus pinos e os configura como entradas com resistores de pull-up. Dessa forma, seu estado natural, quando não pressionados, é 1, mudando para 0 quando pressionados.

Buzzers A e B: os buzzers são responsáveis por emitir os sons das notas que o usuário deve identificar corretamente ao observá-las no display OLED. A configuração desses componentes é feita por meio da função setup_buzzers(), que inicialmente define os pinos dos buzzers para a função PWM, obtém o número do slice (gerador de PWM) e define o divisor de clock (DIVISOR_CLK_PWM) como 16. Essa configuração garante valores aceitáveis de wrap para as frequências das notas musicais reproduzidas.

Micro USB: embora a placa da BitDogLab possua suporte para bateria recarregável, tornando o uso do micro USB dispensável para alimentação, esse recurso é essencial para depuração. Ele permite a emissão de mensagens para o terminal durante a execução do código, auxiliando no monitoramento do funcionamento do sistema. Durante o desenvolvimento, a depuração foi crucial para confirmar qual nota estava sendo reproduzida por cada buzzer e garantir que a resposta do programa estivesse condizente com a realidade. Assim, foi possível assegurar que o usuário recebesse corretamente a informação sobre acertos e erros. A configuração desse recurso é feita por meio do comando stdio_init_all(), permitindo o uso de funções como printf.

Matriz de LEDs 5X5 e PIO: a matriz de LEDs desempenha um papel fundamental na comunicação com o usuário, sinalizando em qual etapa do ciclo o programa se encontra. Antes da reprodução das notas pelos buzzers, a matriz emite um sinal de alerta, indicando que o usuário deve prestar atenção. Durante a reprodução, uma seta amarela é exibida para indicar visualmente qual buzzer está tocando, facilitando a identificação. Após o palpite do usuário, a matriz exibe uma seta para indicar qual buzzer reproduziu a nota correta: verde para acerto e vermelha para erro. A matriz de LEDs 5×5 é controlada por meio de uma máquina PIO (Programmable Input/Output) da Raspberry Pi Pico, que foi configurada para enviar sinais com a temporização exata exigida pelos LEDs do tipo WS2818B. Essa abordagem permite gerar, com alta precisão, os pulsos que codificam os bits de dados que definem as cores de cada pixel da matriz. A configuração utiliza a diretiva .side_set 1 para gerar sinais precisos através de um loop com wrap que incorpora delays dos estados 2, 1 e 4, garantindo que cada bit seja transmitido em exatamente 10 ciclos de clock. A função ws2818b_program_init inicializa o pino e a máquina de estados, configurando a transferência de dados em blocos de 8 bits – cada LED é composto por 24 bits (8 bits para cada cor: G, R e B). O divisor de clock é calculado pela expressão prescaler = clock_get_hz(clk_sys) / (10.f * 800000.f), ajustando a temporização para uma frequência de 800 kHz. Além disso, o uso do TX FIFO para envio dos dados, seguido por uma espera de 100 µs para o pulso de RESET, assegura a correta comunicação e funcionamento da matriz de LEDs.

Módulo ADC: o módulo de conversão analógico-digital (ADC) tem a função de transformar sinais analógicos em valores digitais compatíveis com o processamento pelo microcontrolador. O ADC utilizado possui uma resolução de 12 bits, permitindo a conversão de um sinal analógico variando entre 0 e 3,3 V em um valor digital no intervalo de 0 a 4095. A inicialização desse módulo é realizada pelo comando adc_init(), que configura o periférico para iniciar a conversão analógica-digital. Configurações adicionais, como a seleção do canal ADC e a definição do pino de entrada, são realizadas em funções específicas que serão abordadas nos blocos seguintes.

Joystick: o joystick permite a interação do usuário com o sistema, sendo utilizado para navegação no menu e escolha de funcionalidades, como o afinador e o treinamento de ouvido. Além disso, possibilita a saída dos loops de cada funcionalidade e o encerramento da conexão UART com um segundo dispositivo quando essa estiver ativa. A configuração do botão do joystick segue o mesmo procedimento adotado para os demais botões físicos: o pino correspondente é inicializado e configurado como entrada com resistor de pull-up por meio da função setup_buttons(). Para a leitura analógica do joystick, é utilizada a função setup_adc_joystick(), responsável por inicializar o pino correspondente, selecionar o canal ADC adequado e configurar o sistema para interpretar corretamente os valores lidos.

Microfone: no modo afinador, o sistema utiliza um microfone para capturar amostras de áudio e determinar a frequência predominante no ambiente. Esse processo requer a conversão dos sinais analógicos captados pelo microfone em valores digitais, sendo realizada por meio do módulo ADC. A configuração do microfone é feita na função setup_adc_mic(), que inicializa o pino associado para leitura analógica, seleciona o canal ADC apropriado para conversão, configura uma estrutura FIFO para evitar perdas de amostras e define um divisor de clock com valor 24.576. Esse valor foi escolhido para equilibrar a taxa de amostragem e a resolução espectral da Transformada Rápida de Fourier (FFT) aplicada posteriormente. Dessa forma, o sistema pode analisar um período maior antes de realizar a FFT, garantindo uma precisão aproximada de 0,45 Hz na estimativa da frequência capturada, o que melhora a exatidão da análise em relação ao som real. Além disso, o microfone é configurado para utilizar DMA (Direct Memory Access), um recurso de hardware que permite a transferência direta de dados entre a memória e os periféricos sem a necessidade de intervenção constante do processador. Essa configuração otimiza o desempenho e reduz a carga de processamento sobre os núcleos do RP2040. Para garantir a eficiência na transferência de dados do ADC para o buffer, um canal DMA é alocado dinamicamente por meio da função dma_claim_unused_channel(true). O canal é configurado para realizar transferências de 16 bits, possibilitando adequação à resolução do ADC, mantendo o ponteiro de leitura fixo para garantir que os dados sejam lidos sempre da FIFO do ADC, enquanto o ponteiro de escrita é incrementado para armazenar os dados sucessivamente no buffer. Além disso, a sincronização com as requisições do ADC é feita através da configuração channel_config_set_dreq(&dma_cfg, DREQ_ADC), garantindo que cada amostra convertida seja automaticamente transferida para o buffer sem atrasos desnecessários. Essas configurações tornam a aquisição de áudio mais eficiente e confiável, permitindo ao sistema capturar, processar e analisar sinais sonoros com alta precisão, sem comprometer o desempenho geral do microcontrolador.

LED RGB: O LED RGB tem a função de indicar o status da conexão entre as placas BitDogLab, fornecendo uma representação visual do estado da comunicação. Quando o LED está vermelho, a BitDogLab correspondente está operando de forma independente, sem comprometimento do desempenho, mas com uma quantidade reduzida de recursos disponíveis. A cor amarela indica que a placa está em processo de busca por uma conexão, testando a comunicação via UART. Já a cor verde sinaliza que a conexão foi estabelecida com sucesso e que as duas BitDogLabs estão se comunicando corretamente. A configuração do LED RGB é realizada por meio da função setup_rgb(), que inicializa os três pinos correspondentes aos canais vermelho, verde e azul. Esses pinos são configurados como saídas digitais para permitir o controle individual de cada canal de cor. Inicialmente, todos os pinos são definidos com valor 0, garantindo que o LED permaneça desligado até que uma mudança de status exija a ativação de uma das cores.

Botão reset: o botão reset possui a funcionalidade de reiniciar o microcontrolador, permitindo que o sistema retome a execução do programa ou entre no modo bootloader sem a necessidade de desconectar fisicamente o cabo USB. Essa funcionalidade é especialmente útil durante o desenvolvimento e a depuração, pois possibilita uma recuperação rápida do sistema caso ocorra algum travamento ou seja necessário atualizar o firmware, além de evitar o desgaste dos conectores de alimentação. Em termos de configuração, o botão reset é implementado de forma simples na Raspberry Pi Pico, estando conectado entre o pino RUN (pino número 30) e o GND da placa. Ao pressionar o botão, esse pino é curto-circuitado ao GND, forçando o microcontrolador a reiniciar.

Display OLED: tem a função de exibir informações para o usuário durante a execução do programa. Inicialmente, ele é utilizado na implementação do menu, que é composto por três telas: uma tela de início, uma tela de seleção do treinamento de ouvido e uma tela para a escolha do afinador. Dentro de cada funcionalidade, o display também desempenha um papel fundamental ao fornecer informações relevantes. No modo afinador, ele exibe a frequência capturada do som ambiente, a nota musical mais próxima dessa frequência e a diferença, em Hz, entre a frequência detectada e a frequência da nota mais próxima. Já no modo treinamento de ouvido, o display apresenta a instrução “ACERTE O BOTÃO DA NOTA” e exibe a nota musical gerada, como “C5#”, por exemplo, juntamente com sua frequência correspondente, como 523 Hz. Quando o usuário faz um palpite, o display informa se a resposta está correta ou errada. O display OLED utiliza o protocolo de comunicação I2C, sendo um exemplo prático da implementação desse tipo de comunicação. Sua configuração é realizada pela função setup_oled(), que inicializa o módulo i2c1 com uma frequência de 400.000 Hz, configura os pinos SCL e SDA para a comunicação com resistores de pull-up e estabelece a comunicação por meio da função ssd1306_init() da biblioteca ssd1306. A interação entre o processador e o display é feita utilizando um buffer de memória denominado ssd.

DIG: sigla "DIG" é uma abreviação da palavra "Digital" e se refere a um módulo adicional da placa BitDogLab, projetado para permitir conexões mais robustas, como soldagem ou ligações por cabos jacaré. Os terminais do DIG são diretamente conectados aos pinos 1, 2, 3 e 4 da Raspberry Pi Pico W, facilitando conexões temporárias no dispositivo. Os orifícios desse módulo são utilizados para estabelecer uma comunicação UART entre duas placas BitDogLab por meio de cabos jacaré. No entanto, também é possível realizar essa conexão de forma alternativa, utilizando um cabo JST de 4 pinos. A configuração do DIG é feita pela função setup_uart(), que inicializa a UART0 com uma taxa de transmissão de 115200 baud, configura os pinos do transmissor e do receptor e habilita o FIFO (First In, First Out) da UART para evitar a perda de dados quando o buffer está cheio. Vale destacar que o programa desenvolvido é capaz de operar de forma independente em uma única BitDogLab, tornando o uso desse módulo opcional. No entanto, sua utilização é recomendada, pois ele facilita o entendimento da comunicação UART e proporciona uma experiência mais completa ao usuário.

A especificação técnica deste projeto foi cuidadosamente escolhida para atender aos requisitos do usuário, garantindo uma experiência fluida e didática. Como o objetivo principal é demonstrar o uso dos principais recursos da BitDogLab, a própria placa se apresenta como a solução de hardware ideal para a implementação, pois já integra os componentes necessários para as funcionalidades propostas.

A velocidade de processamento de 125 MHz do microcontrolador RP2040, juntamente com as taxas de transmissão de 115200 baud para comunicação UART e 400000 baud para comunicação I²C, são adequadas para a proposta do projeto. Como não se trata de uma aplicação de alto desempenho, essas especificações garantem execução suave e tempos de resposta confortáveis para o usuário, permitindo uma interação ágil e intuitiva.

O sistema de áudio, composto por buzzers ativos, é capaz de produzir sons com uma corrente superior a 12 mA, garantindo que os usuários consigam ouvir claramente as notas, mesmo em ambientes barulhentos. A funcionalidade de afinador e treinamento de ouvido depende dessa característica para proporcionar uma experiência de aprendizado eficaz e imersiva.

A bateria de 3,7V 2000mAh confere autonomia e portabilidade, permitindo que o usuário utilize o dispositivo sem a necessidade de estar conectado a uma fonte de energia externa. Essa característica é essencial para um projeto educacional, pois facilita sua utilização em diferentes ambientes, como salas de aula, laboratórios e até mesmo apresentações.

A interface gráfica do sistema é composta por dois elementos principais: a matriz de LEDs 5x5 e o display OLED de 60x128 pixels. A matriz de LEDs permite a exibição de símbolos claros e perceptíveis, como as setas que indicam qual buzzer está ativo. Sua taxa de transmissão, baseada no protocolo do LED WS2818B, opera a 800 kHz, garantindo uma atualização suficientemente rápida para proporcionar animações fluidas e uma sensação de movimento contínuo. Com uma frequência de clock de 8 MHz, o sistema consegue realizar múltiplas instruções por bit enviado, assegurando uma resposta visual precisa e sincronizada com os eventos do programa.

O display OLED, por sua vez, desempenha um papel fundamental na exibição de informações como notas musicais detectadas pelo afinador, frequências captadas pelo microfone e instruções para o usuário. Sua resolução de 60x128 pixels é suficiente para apresentar textos e símbolos com clareza e boa legibilidade, mesmo em uma tela compacta.

A captação de áudio é feita por meio de um microfone com amplificador operacional MAX4466, que oferece alta sensibilidade e boa resposta para diferentes fontes sonoras. Esse componente permite que o sistema identifique frequências musicais com precisão, possibilitando tanto a afinação de instrumentos de corda quanto a captação de sons de dispositivos eletrônicos e da própria voz humana.

Com essas especificações, o projeto consegue atingir seu objetivo principal de demonstrar os recursos da BitDogLab de forma clara, interativa e funcional, proporcionando uma experiência educacional rica e intuitiva para os usuários.

Para a execução do projeto, os seguintes materiais foram utilizados:

Componentes principais

- 2 Placas BitDogLab (*sendo a segunda opcional, utilizada para uma experiência aprimorada*)
- 1 Bateria recarregável 3,7V 2000mAh 18650 Lithium (*caso seja desejado operar o dispositivo de forma portátil, sem conexão direta à alimentação; se 2 placas forem utilizadas 2 baterias seriam necessárias*)
- 1 Cabo micro USB de 1m, 2400mA (*para alimentação e programação do(s) dispositivo(s)*)

Conectores e acessórios

- 3 Conectores jacaré (*para conexão UART entre duas BitDogLab, se necessário*)
  - Abertura máxima da garra: ~8mm
  - Comprimento total: 50cm
  - Material do fio condutor: Cobre
  - Dimensões da garra (CxLxE): ~28x7x2mm

Componentes eletrônicos alternativos (*caso a BitDogLab não esteja disponível, permitindo a montagem de um circuito equivalente*)

Áudio

- 2 Buzzers passivos (*modelo utilizado: MLT-8530*)
  - Nível de pressão sonora (SPL): 80dB@5V,10cm
  - Tensão operacional: 2,5V~4,5V
  - Frequência: 2700Hz

Interface de usuário

- 1 Joystick genérico (*para navegação no menu e seleção de opções*)
- 2 Botões genéricos (*para seleção de notas no treinamento auditivo*)

Display e Iluminação

- 1 Display OLED 64x128 pixels (*modelo utilizado: SSD1306, interface I²C*)
  - Tensão de operação: 3V a 5V
  - Nível lógico compatível: 3,3V ou 5V
  - Dimensão do módulo: 27x26 mm
  - Dimensão útil da tela: 25x14 mm
- 26 LEDs RGB WS2818B (*25 para a matriz de LED 5x5 e 1 LED individual para alertas visuais*)
  - Tensão de alimentação: 6 ~ 7V DC
  - Tipo de LED: 5050 RGB
  - Velocidade de transmissão de dados: até 800Kbps
  - Taxa de atualização: 30fps
  - Dimensões unitárias: 5x5x0,5mm

Processamento e Controle

- 1 Raspberry Pi Pico W (caso a BitDogLab não esteja disponível, essa placa pode ser usada como alternativa para o controle do sistema)
  - Tensão de alimentação: 1,8 - 5,5V DC
  - Microcontrolador: RP2040 Dual-core Arm Cortex-M0+ @ 133MHz
  - Memória SRAM: 264KB

- Memória Flash: 2MB QSPI
- Interfaces disponíveis: 26 GPIO, 3 ADC 12 bits, 2 SPI, 2 I²C, 2 UART, 8 PIO, 16 canais PWM
- Conectividade: Wi-Fi 2,4GHz IEEE 802.11b/g/n e Bluetooth 5.2
- Dimensões: 21 x 51 x 4mm

Captação de áudio

- 1 Microfone com amplificador operacional MAX4466 ou similar
  - Offset de tensão: 1,65V (centro da faixa do ADC)
  - Amplitude máxima do sinal: 1,65V (variação entre 0V e 3,3V)

Esta lista garante que o projeto possa ser montado tanto utilizando a BitDogLab original quanto através de uma montagem manual com componentes eletrônicos equivalentes, possibilitando maior flexibilidade na replicação e estudo do sistema.

A tabela apresenta a descrição da pinagem utilizada na BitDogLab, sendo válida tanto para uma única placa quanto para um sistema composto por duas unidades. Os pinos IO são destinados à entrada e leitura de sinais digitais, como os botões e o controle do LED RGB. Já os pinos ADC são responsáveis pela leitura de sinais analógicos, sendo utilizados para captar informações do microfone e do joystick. O pino PIO é programado por uma máquina de estado para controlar a matriz de LEDs, garantindo precisão na temporização dos sinais enviados. Além disso, o GND serve como referência elétrica do circuito e pode ser compartilhado entre duas placas para garantir um nível de tensão comum entre elas.

Numeração do Pino	Uso	Função
1	UART0 TX	Comunicação UART com uma segunda BitDogLab quando disponível.
2	UART0 RX	
3	GND	
7	IO	Entradas digitais para leitura do status dos botões
9	IO	
29	IO	
10	PIO	Controle da matriz de LEDs
14	PWM	Reprodução de sons com buzzers
27	PWM	
15	IO	Controle do LED RBG
16	IO	
17	IO	
19	I2C SDA	Controle do display OLED
20	I2C SLC	
34	ADC	Captação do microfone
32	ADC	Captura do eixo x do joystick
30	IO	Reinicialização do sistema

A comunicação entre os componentes da placa ocorre por meio de diferentes interfaces, como PWM, utilizado para a reprodução de sons nos buzzers, e I²C, empregado no controle do display OLED. A interface UART, presente nos pinos 1 e 2, é opcional e só é utilizada quando há duas placas BitDogLab em funcionamento simultâneo, permitindo a troca de informações entre elas. Por fim, o pino 30 é dedicado à reinicialização do sistema, possibilitando que o microcontrolador seja reiniciado sem a necessidade de desligamento manual, facilitando a depuração e atualização do firmware.

A Figura 3 apresenta o desenho completo do circuito de hardware, permitindo visualizar a numeração e a função de cada pino da Raspberry Pi Pico W, bem como o local onde cada componente está conectado. A primeira BitDogLab, além de desempenhar as funções principais do programa, também envia mensagens para o terminal via USB, proporcionando uma visão mais detalhada do algoritmo em execução. Quando a placa está conectada via USB, o uso da bateria torna-se dispensável, facilitando testes e depurações do sistema.

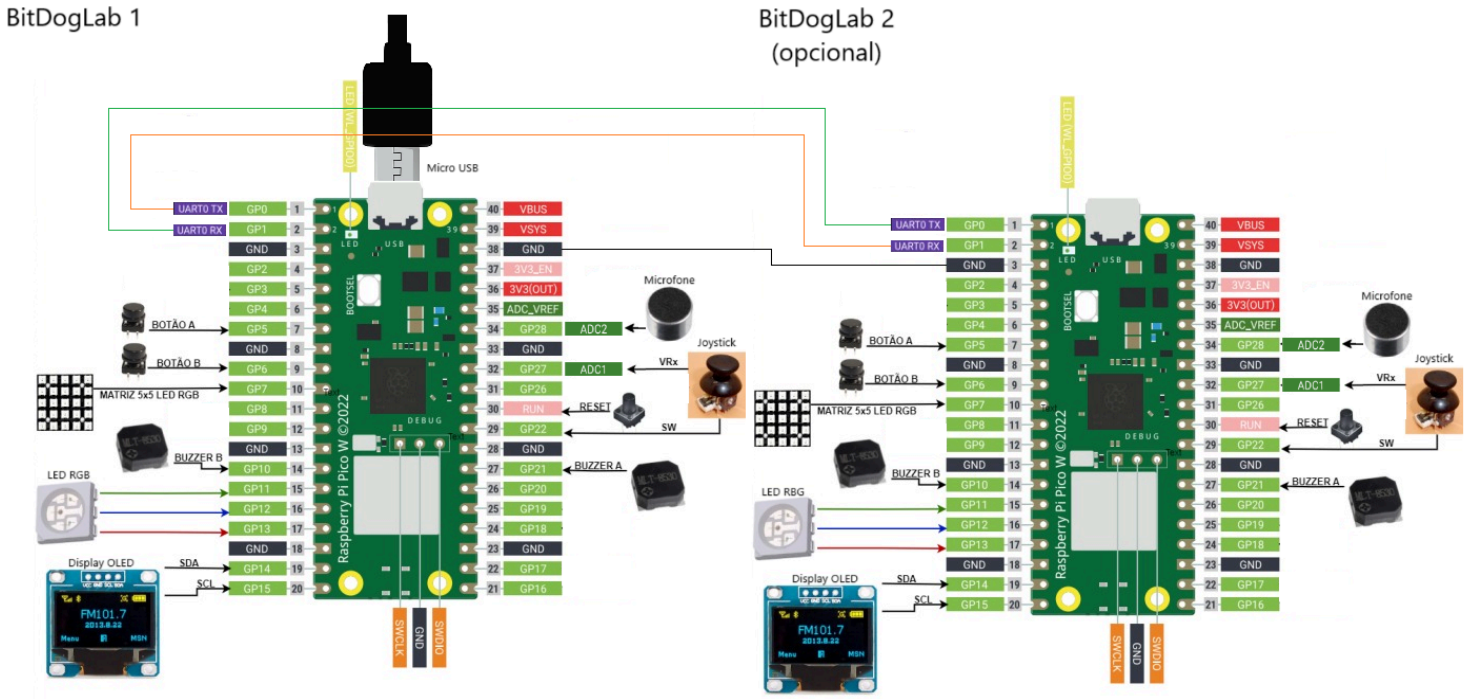

Figura 3 - Circuito completo do hardware.

A comunicação UART entre as placas é estabelecida por meio de três cabos jacaré. O pino 1 da primeira placa é conectado ao pino 2 da segunda, e o pino 1 da segunda é ligado ao pino 2 da primeira, permitindo a troca de mensagens entre ambas. Para garantir que as placas compartilhem a mesma referência elétrica, um terceiro cabo conecta os pinos GND das duas placas. Na figura, essa conexão é representada entre os pinos 38 e 3, mas pode ser feita em qualquer par de pinos correspondentes ao GND.

3. Software

O algoritmo music_app, responsável pela interface musical e interação com o usuário, integra uma arquitetura de software estratificada. Essa hierarquia permite que cada camada se concentre em funções específicas, desde a inicialização do hardware até a implementação das funcionalidades de alto nível, facilitando o desenvolvimento, a manutenção e a evolução do sistema.

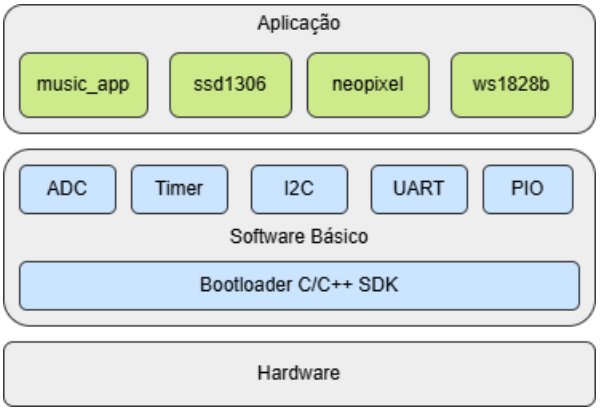

Figura 4 - Camadas de software do projeto.

A primeira camada, denominada camada de software básico, corresponde ao firmware e aos drivers de baixo nível. Essa camada representa a fundação do sistema, onde reside o firmware – baseado no bootloader e no C/C++ SDK da Raspberry Pi Pico W. Durante o processo de inicialização, o firmware executa diversas funções essenciais, como a configuração dos principais periféricos e o microcontrolador, garantindo que todos os recursos, como GPIO, ADC e timers, sejam devidamente inicializados. Além disso, o firmware ativa a interface USB, permitindo que a placa seja programada via conexão USB e apareça como um dispositivo de armazenamento. Essa funcionalidade facilita a transferência de arquivos e atualizações de firmware, além de possibilitar a depuração por meio do envio de mensagens para o terminal, utilizando funções como stdio_init_all(). Por fim, essa camada prepara a base necessária para a execução do software de aplicação, assegurando que o hardware esteja em um estado estável e configurado.

Os módulos de software abstraem os detalhes de controle dos periféricos, oferecendo interfaces de alto nível para o desenvolvedor. Entre esses módulos, destaca-se o módulo ADC, responsável pela conversão dos sinais analógicos em digitais. Esse módulo gerencia a configuração dos canais ADC, como os utilizados pelo microfone e pelo joystick, e oferece funções como adc_init(), permitindo que o software de aplicação obtenha leituras precisas sem precisar lidar com a complexidade da conversão em si. Outro módulo importante é o módulo Timer, que coordena a temporalização do sistema, essencial para o agendamento de tarefas e a sincronização de eventos. Com ele, funções como get_absolute_time() e delayed_by_us() tornam-se acessíveis, possibilitando a implementação de delays e a medição precisa de intervalos de tempo. O módulo I2C também compõe essa camada e abstrai os detalhes da comunicação com dispositivos que utilizam o protocolo I²C, como o display OLED. Ao utilizar funções de inicialização e transmissão, como i2c_init(), o desenvolvedor pode interagir com o display sem precisar se preocupar com o controle dos sinais elétricos e a sincronização dos dados. De maneira semelhante, o módulo UART é fundamental para a comunicação serial na troca de dados entre duas BitDogLabs. Apesar de a camada de aplicação poder especificar parâmetros como a taxa de transmissão, o módulo UART gerencia toda a lógica de envio e recepção dos dados em nível baixo, garantindo uma comunicação robusta e sincronizada. Por fim, o módulo PIO (Programmable Input/Output) oferece uma maneira extremamente flexível de controlar sinais digitais com precisão, independentemente da carga de processamento dos núcleos principais. Cada bloco PIO dispõe de quatro máquinas de estado (state machines) que operam de forma independente e comunicam-se com o processador via filas (FIFO). Essa funcionalidade é crucial para controlar dispositivos com requisitos temporais rigorosos, como a matriz de LEDs, onde cada bit de informação deve ser transmitido com uma temporização exata.

A camada de aplicação concentra a interface interativa do sistema, sendo composta por quatro módulos de software que colaboram para oferecer uma experiência completa ao usuário. O primeiro módulo, denominado ws2818b.pio, é responsável pelo controle da matriz de LEDs. Ele contém o código em PIO assembly,

estruturado com a diretiva de wrap, side-set e instruções de saída e salto, que garantem a temporização precisa necessária para a comunicação com os LEDs do tipo WS2818B. Em complemento, a função C ws2818b_program_init realiza a configuração do pino, inicializa o estado da máquina PIO e ajusta o divisor de clock de forma a assegurar que cada bit seja transmitido com a exatidão requerida para a operação do LED. O segundo módulo, o neopixel.c, gerencia a matriz de LEDs RGB de maneira integrada. Esse módulo define a estrutura de dados para representar cada pixel, armazena os valores de cor (R, G e B) em um buffer alocado dinamicamente e implementa funções que permitem a inicialização (através de npInit), a atribuição de cores individuais com npSetLED, a limpeza do buffer com npClear e a escrita dos dados nos LEDs por meio de npWrite. Ao utilizar o programa PIO configurado no módulo anterior, o neopixel.c assegura a transmissão correta e a sincronização dos sinais, permitindo animações e efeitos visuais com precisão.

O terceiro componente da camada de aplicação é o módulo ssd1306, proveniente de uma biblioteca consolidada para o controle de displays OLED baseados no driver SSD1306. Esse módulo é responsável pela inicialização, configuração e atualização do display, facilitando a exibição de menus e informações interativas. Para atender às especificidades do projeto, os arquivos originais da biblioteca foram editados para incluir novos caracteres, ampliando assim as possibilidades de representação gráfica e personalização da interface do usuário. Por fim, o módulo music_app implementa o algoritmo principal do aplicativo musical, onde toda a lógica de interação com o usuário é definida. Nele, são integradas as funcionalidades dos demais módulos, como o controle visual via display e LED, a captação de áudio e a leitura dos comandos de entrada, para oferecer uma experiência prática e didática. No caso da utilização de uma segunda BitDogLab, as estruturas das camadas de aplicação e de software básico permanecem as mesmas, porém o módulo que implementa o algoritmo principal passa a ser denominado coop_music_app. Essa versão apresenta uma lógica diferenciada, adaptada para a cooperação entre as duas placas, permitindo uma expansão das funcionalidades e uma interação mais dinâmica entre os dispositivos.

Devido à extensão do fluxograma, sua representação foi dividida em quatro partes, apresentadas nas Figuras 5, 6, 7 e 8. A Figura 5 ilustra o processo de inicialização do software, detalhando as ações executadas e as decisões tomadas até que um dos loops principais seja acessado. A Figura 6 descreve o funcionamento interno do loop de treinamento auditivo, evidenciando o processo de conexão automática entre as duas placas e o mecanismo que permite ao usuário optar por permanecer ou sair desse modo. A Figura 7 foca no loop correspondente ao afinador, cuja estrutura é mais simples em comparação com os demais. Por fim, a Figura 8 apresenta o funcionamento do programa secundário executado na segunda placa BitDogLab, que, embora compartilhe semelhanças com o software principal, possui diferenças significativas em sua lógica de operação.

Principais variáveis

O software utiliza diversas constantes, macros, variáveis globais e estruturas que permitem a configuração e o controle dos periféricos, além de organizar os dados para processamento e exibição. As constantes definidas com #define especificam, por exemplo, o canal e o pino do microfone (através de MIC_CHANNEL e MIC_PIN), a frequência base do ADC (ADC_CLOCK) e seu divisor (ADC_DIV), que juntos determinam a taxa de amostragem real (SAMPLE_RATE). Outras constantes definem o número de amostras (SAMPLES), os pinos e a quantidade de LEDs (através de LED_PIN e LED_COUNT), bem como os pinos dos buzzers (BUZZER_A e BUZZER_B). Macros adicionais, como NUM_NOTES, calculam automaticamente o tamanho dos arrays de notas musicais, enquanto outras constantes definem os pinos para comunicação I²C (I2C_SDA e I2C_SCL), os parâmetros para a interface do display OLED (como TEXT_LINES e TEXT_LENGTH), e a largura da “barra” do joystick (BAR_WIDTH). Pinos específicos são definidos para os botões de interação (joystick, A e B), para a comunicação UART (UART_TX e UART_RX) e para os LEDs RGB (RED_PIN, GREEN_PIN e BLUE_PIN).

No nível das variáveis globais, encontram-se parâmetros críticos para o funcionamento dos módulos: o divisor do clock para o PWM (DIVISOR_CLK_PWM) e o valor de wrap para ajustar o duty cycle dos buzzers (wrap_div_buzzer). A variável num_BitDogLabs armazena quantos dispositivos estão conectados, permitindo a alternância entre operações com um ou dois dispositivos. As variáveis dma_channel e dma_cfg guardam, respectivamente, o identificador e a configuração do canal DMA utilizado para transferir as amostras do ADC para o buffer adc_buffer. Este buffer, por sua vez, armazena as amostras de áudio que serão processadas pela FFT, cujo vetor de entrada é o array de estruturas do tipo Complexo (fft_in). Outras variáveis, como note, diff e frequency, armazenam a nota musical detectada, a diferença entre a frequência medida e a nota ideal e a frequência dominante identificada, respectivamente. O array text é utilizado para formatar as informações que serão exibidas no display OLED, enquanto os arrays notes e note_wraps definem os dicionários de notas

musicais e os respectivos valores de wrap para o PWM. Por fim, as matrizes tela1, tela2 e tela0 armazenam as strings pré-definidas para as telas de menu, e a estrutura frame_area delimita a área do display onde o conteúdo será renderizado.

Inicialização

O programa começa com a inclusão das bibliotecas essenciais, que fornecem os recursos necessários para lidar com os periféricos utilizados, como a comunicação serial, display OLED, LEDs RGB, matriz de LEDs e joystick. Essas bibliotecas garantem a interface correta entre o software e o hardware. Na sequência, ocorre a definição da pinagem, configurando quais pinos serão usados para cada componente. Isso abrange o joystick, botões de entrada, microfone, buzzers e os pinos de controle dos LEDs. Também são definidas as constantes do programa, que representam limites e parâmetros específicos, como posições do joystick e definições relacionadas ao display. Em seguida, são declaradas variáveis globais que armazenam estados e informações necessárias para o funcionamento contínuo do programa. Após as definições básicas, o programa estabelece funções específicas para os módulos e componentes. Funções relacionadas ao afinador são configuradas para processar o áudio captado e determinar a frequência dominante, exibindo a nota correspondente no display OLED. No módulo de treinamento de ouvido, as funções permitem tocar notas aleatórias e avaliar se o usuário identificou corretamente essas notas. Funções para monitorar os estados do joystick e dos botões são configuradas para capturar cliques e leituras de posição, e as funções da matriz de LEDs e LEDs RGB permitem controlar diferentes padrões e cores. Também há funções específicas para o display OLED, que garantem a exibição das informações de maneira clara, com a possibilidade de renderizar texto e gráficos. A comunicação UART é configurada para garantir a troca de dados entre diferentes módulos e placas, permitindo sincronização em tempo real.

Com todos os recursos iniciais configurados, o programa exibe a tela inicial do menu, que apresenta ao usuário as opções de entrada no modo afinador ou no modo de treinamento de ouvido. O display OLED é limpo e atualizado para garantir a exibição correta das informações. O programa então entra no loop principal, onde monitora continuamente o joystick e os botões. Ele lê a posição x do joystick e ajusta a tela exibida conforme o movimento detectado. Se o joystick for movido para uma nova posição, o programa muda para a tela correspondente, alternando entre as telas de afinador e treinamento de ouvido. A verificação de pressionamento do botão do joystick é um ponto crítico no programa. Se o botão não foi pressionado, o programa permanece no loop de monitoramento. Se o botão for pressionado, o programa identifica a tela exibida no momento e toma a ação apropriada. Se a tela atual for a do afinador, o programa ativa o microfone para captura de som e entra no loop do afinador. Nesse loop, as amostragens de áudio são processadas para calcular a frequência dominante, que é comparada com as notas musicais definidas. A nota correspondente é exibida no display OLED, auxiliando o usuário no processo de afinação. Por outro lado, se a tela exibida for a de treinamento de ouvido, os buzzers são configurados para tocar notas, e um gerador de números aleatórios é ativado para selecionar notas aleatórias para o treino. O programa entra então no loop de treinamento, onde notas são tocadas e o usuário precisa identificá-las corretamente para continuar. Esse processo de inicialização garante que todos os módulos estejam configurados corretamente e que o programa esteja pronto para responder conforme a escolha do usuário, proporcionando uma experiência de uso fluida e interativa, seja no modo afinador ou no modo de treinamento de ouvido.

Configurações dos registros

A configuração dos registradores é realizada por meio de funções específicas que ajustam os parâmetros dos diversos periféricos do microcontrolador RP2040. Para a aquisição de áudio, a função setup_adc_mic inicializa o ADC: ela configura o pino do microfone como entrada analógica, seleciona o canal desejado e prepara o FIFO do ADC com o threshold adequado. A taxa de amostragem é ajustada com adc_set_clkdiv(ADC_DIV), e a transferência dos dados é otimizada através da configuração do DMA, que é preparado para realizar transferências de 16 bits do FIFO para o buffer adc_buffer, com incremento apenas no ponteiro de escrita e sincronização com o ADC via DREQ_ADC.

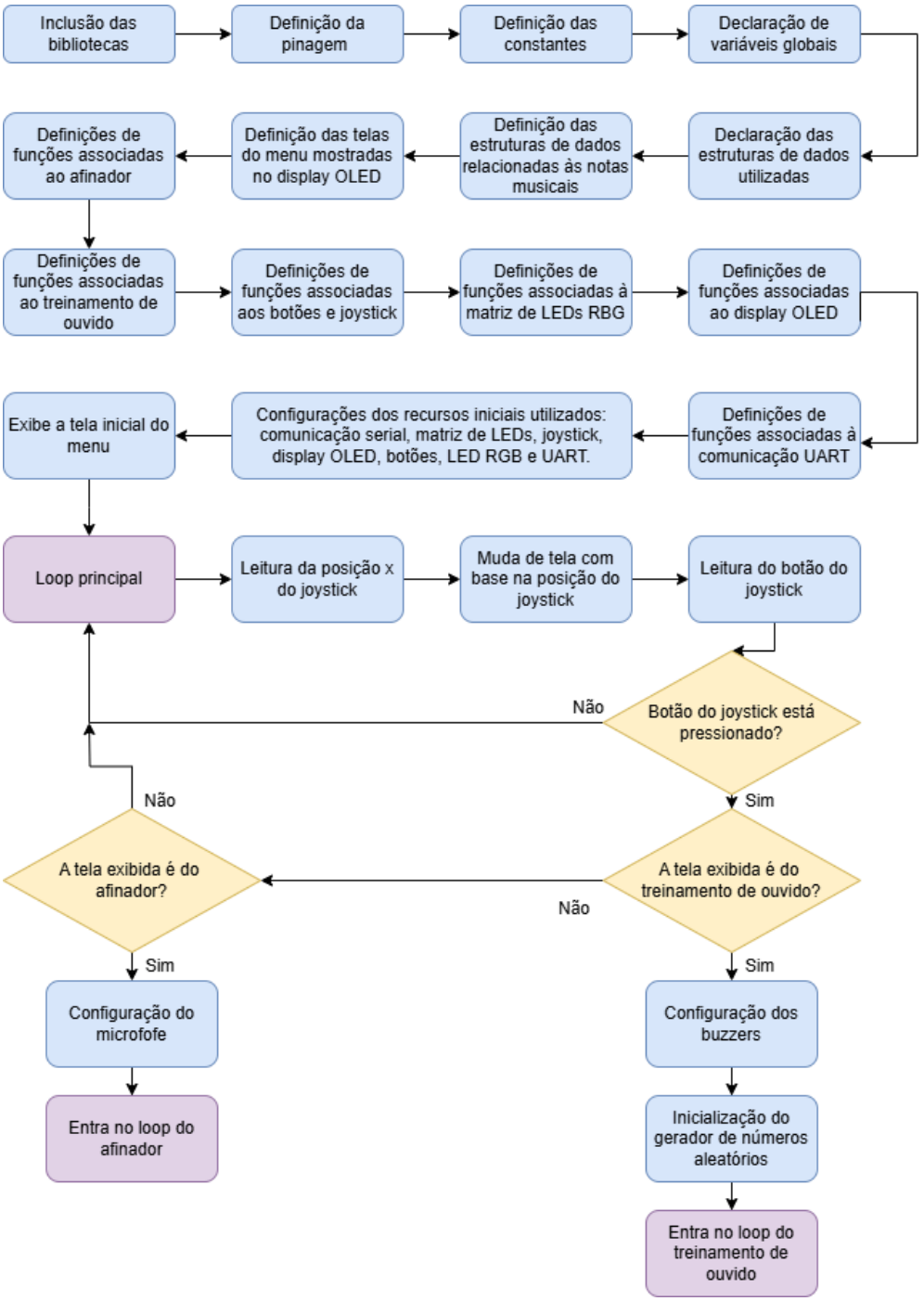

Figura 5 - Fluxograma do software (inicialização).

A configuração dos buzzers é realizada na função setup_buzzers, que utiliza a função gpio_set_function para configurar os pinos em modo PWM. Em seguida, a função pwm_gpio_to_slice_num identifica o slice correspondente a cada buzzer, e o divisor do clock do PWM é ajustado com pwm_set_clkdiv, utilizando o valor definido em DIVISOR_CLK_PWM. Para a comunicação, a função setup_uart inicializa a UART0 com uma taxa de 115200 baud e configura os pinos de transmissão e recepção para a função UART. O FIFO da UART é ativado para garantir uma transmissão sem perda de dados, e a função setup_oled configura o barramento I²C, definindo os pinos SDA e SCL com resistores de pull-up, para a comunicação com o display OLED. Outras funções, como setup_buttons e setup_adc_joystick, configuram os pinos dos botões e do joystick como entradas com resistores de pull-up, garantindo estados lógicos estáveis. Por fim, a função setup_rbg inicializa os pinos dos LEDs RGB como saídas e os coloca inicialmente no estado desligado.

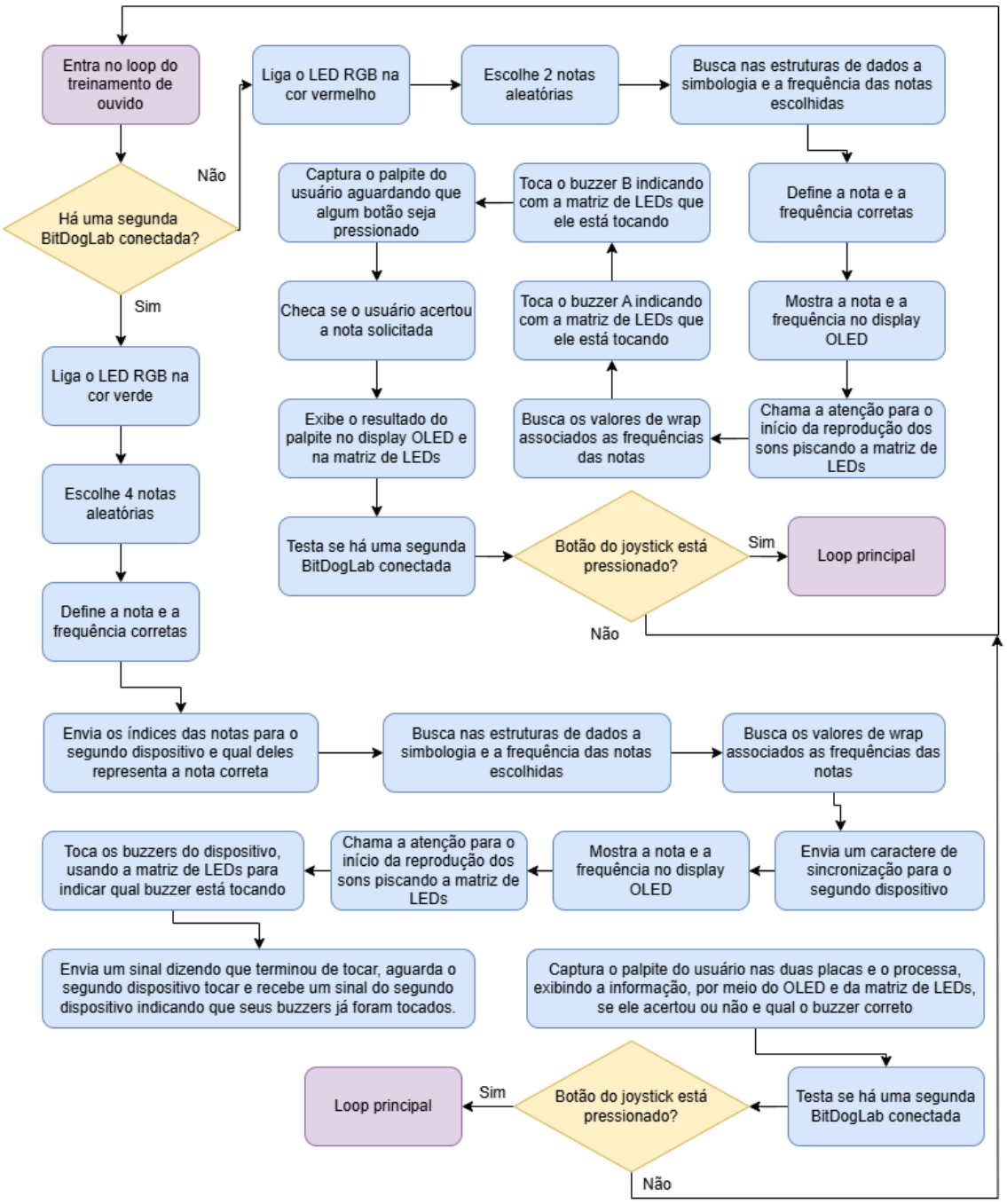

Figura 6 - Fluxograma do software (treinamento de ouvido).

Estrutura e formato dos dados

O software organiza os dados críticos por meio de estruturas e arrays que permitem a manipulação e o processamento dos sinais e das informações. A estrutura Complexo é definida para armazenar números complexos, com campos para as partes real e imaginária, sendo essencial para a realização da Transformada Rápida de Fourier (FFT) dos sinais de áudio. A estrutura Note associa o nome de uma nota musical a um valor numérico, que pode representar tanto a frequência em Hertz quanto o valor de wrap para o PWM. Essa estrutura é utilizada para definir os arrays notes e note_wraps, que funcionam como dicionários para a identificação das notas e para a configuração dos buzzers. Além dessas estruturas, o software utiliza arrays para armazenar amostras de áudio (adc_buffer), dados de FFT (fft_in) e textos que serão exibidos no display OLED (text). As matrizes de strings que compõem as telas de menu (tela1, tela2, tela0) facilitam a navegação do usuário pelo sistema. A estrutura frame_area delimita a área de renderização do display, definindo as colunas e páginas a serem atualizadas, o que assegura que as informações sejam apresentadas de maneira organizada e centralizada.

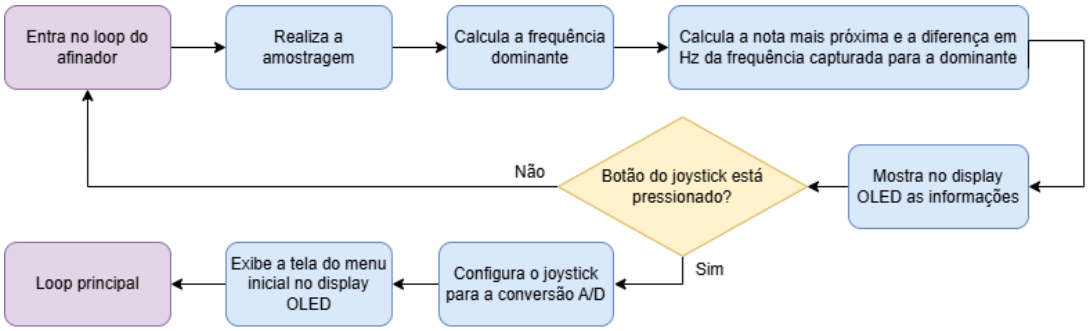

Figura 7 - Fluxograma do software (afinador).

Organização da memória

A organização da memória no software se dá principalmente por meio do uso de memória estática e dinâmica, bem como do acesso a registradores mapeados. As variáveis globais e arrays, como adc_buffer, fft_in, text, notes, note_wraps e as matrizes de telas, são alocados na SRAM e permanecem disponíveis durante toda a execução do programa, proporcionando acesso rápido e consistente. Durante a execução da função fft, a memória dinâmica é utilizada para alocar temporariamente os arrays par e impar com a função malloc(), que são posteriormente liberados com free() para evitar vazamentos e otimizar o uso do heap. Além disso, o acesso aos periféricos – ADC, PWM, UART, I²C e DMA – é realizado por meio de bibliotecas do SDK do RP2040, que escrevem nos registradores mapeados em endereços específicos do microcontrolador. Embora esses endereços não sejam manipulados explicitamente no código, as funções de inicialização e configuração garantem que os parâmetros de operação dos módulos sejam corretamente ajustados, permitindo a comunicação eficiente entre o hardware e o software.

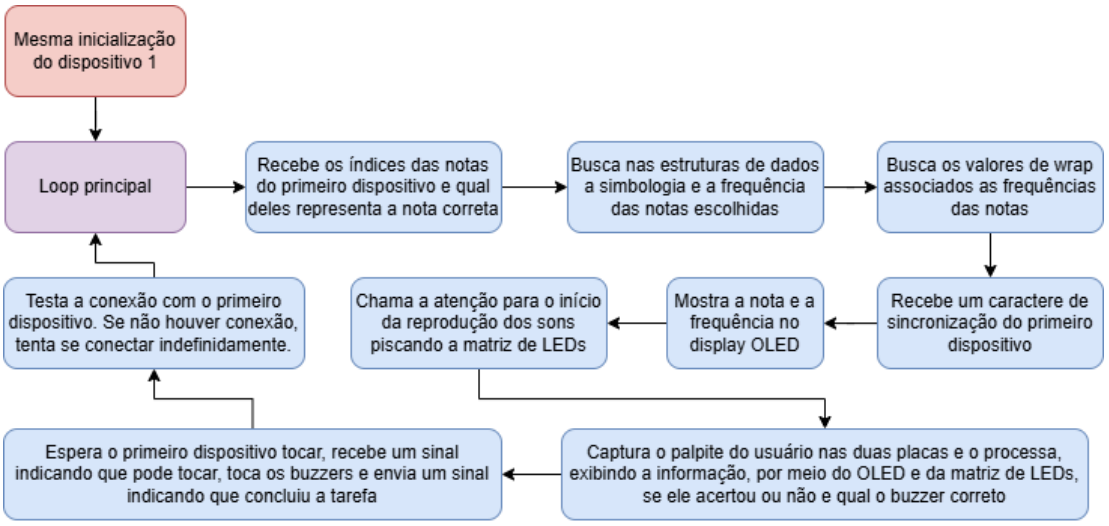

Figura 8 - Fluxograma do software (segunda BitDogLab).

Protocolos de comunicação

O sistema utiliza dois protocolos principais para a comunicação: o UART para a sincronização e troca de dados entre as BitDogLabs e o I²C para a comunicação com o display OLED. A comunicação UART opera a 115200 baud, permitindo o envio de dados simples, byte a byte, como índices de notas, identificadores de buzzer e sinais de sincronização (por exemplo, o caractere 's' para indicar prontidão ou confirmar ações). Esse protocolo robusto possibilita a troca de informações em tempo real, garantindo que os dispositivos trabalhem em sincronia, especialmente durante as funcionalidades de treinamento de ouvido, onde os palpites do usuário e os comandos de controle são imediatamente transmitidos entre as placas. O protocolo I²C é utilizado para controlar o display OLED. Configurado com uma velocidade adequada de 400 kHz e utilizando os pinos SDA e SCL com resistores de pull-up, o I²C permite a transmissão dos dados gráficos e textuais para o display. As funções do driver SSD1306 cuidam da formatação e envio dos comandos, organizando os bytes que compõem o conteúdo

visual (como textos e barras) de forma síncrona com as demais operações do sistema.

Formato dos pacotes de dados

No que diz respeito ao formato dos dados transmitidos, o protocolo UART adota um esquema simples e sequencial. Em cenários onde duas BitDogLabs estão conectadas, o dispositivo mestre envia inicialmente uma sequência de cinco bytes. Os quatro primeiros bytes correspondem aos índices das notas selecionadas no array de notas, enquanto o quinto byte indica qual buzzer contém a nota correta. Sinais adicionais, como o caractere 's', são enviados para sincronizar a reprodução dos sons ou confirmar a conclusão de uma etapa da operação, facilitando a comunicação de comandos e a captura de palpites do usuário. Já na comunicação via I²C com o display OLED, os dados não são estruturados como pacotes complexos com cabeçalhos e trailers, mas sim organizados em um buffer que representa o conteúdo visual a ser exibido. Esse buffer contém os bytes que definem o estado de cada pixel do display e é transmitido de forma contínua pelo barramento I²C. A biblioteca SSD1306 é responsável por interpretar esses dados, atualizar as áreas definidas do display e garantir que as informações sejam renderizadas de forma clara e sincronizada com as demais operações do sistema. Dessa forma, o formato dos dados tanto no UART quanto no I²C é adaptado às necessidades específicas de cada função, mantendo a simplicidade e eficiência na transmissão.

Repositórios

Os códigos para as placas BitDogLab principal e secundária estão disponíveis nos seguintes repositórios:

- Repositório da placa principal
- Repositório da placa secundária

Esses dois projetos são essencialmente idênticos, diferindo apenas no programa principal. A separação em dois repositórios foi feita para facilitar o processo de desenvolvimento, embora ambos os programas pudessem estar contidos em um único repositório. Nesse caso, bastaria selecionar o programa principal correspondente no momento da compilação e transferência para a placa apropriada.

4. Execução do projeto

A execução do projeto iniciou-se com a identificação da problemática. Em meio a inúmeras áreas de aplicação dos sistemas embarcados – que vão desde a agropecuária até a medicina – a área educacional muitas vezes é negligenciada. Observou-se que, com a evolução da Internet das Coisas, há uma crescente demanda por profissionais especializados em sistemas embarcados. Nesse contexto, o curso Embarcatech surgiu como uma iniciativa que visa fomentar a formação desses profissionais. Durante as pesquisas realizadas por meio de fóruns, grupos de WhatsApp e outros canais de comunicação utilizados pelos alunos, constatou-se que a BitDogLab, uma placa eletrônica de intuito educativo, ainda é relativamente nova e não conta com uma comunidade consolidada ou uma ampla oferta de códigos exemplares. Além disso, os exemplos existentes demonstram o funcionamento de seus componentes de forma isolada – como o simples acender e apagar de LEDs – sem apresentar uma aplicação prática que se relacione diretamente com o cotidiano dos usuários. Essa carência evidenciou a necessidade de desenvolver um projeto que integrasse diversos recursos da placa e apresentasse utilidade prática, solucionando as dificuldades apontadas pelos alunos, como a integração dos componentes e a migração do simulador para o hardware real.

Com base nas pesquisas, optou-se pela utilização da própria BitDogLab como hardware de escolha. Essa decisão foi motivada pela capacidade da placa de integrar, de forma coerente, um conjunto diversificado de componentes eletrônicos – como ADC, PWM, I²C, UART, PIO, matriz de LEDs, buzzer, joystick e display OLED – que são essenciais para o desenvolvimento de aplicações interativas e lúdicas. A proposta visava utilizar o máximo desses recursos, garantindo que cada componente desempenhasse um papel significativo no funcionamento do sistema. Adicionalmente, considerando a aplicação em ambientes educativos, foi prevista a possibilidade de expansão para duas BitDogLabs, permitindo que os alunos experimentem a comunicação via protocolo UART e desenvolvam projetos cooperativos, sem prejudicar aqueles que dispõem de apenas uma placa.

No que tange à definição das funcionalidades do software, o afinador foi concebido para demonstrar uma amostragem precisa do áudio, utilizando o ADC em conjunto com o DMA para coletar e processar sinais captados pelo microfone, aplicando a Transformada Rápida de Fourier (FFT) para identificar a frequência dominante. Essa funcionalidade não só evidenciou o uso de técnicas avançadas de processamento de sinais, como também possibilitou a comparação dos resultados com instrumentos musicais ou aplicativos de emissão de som, proporcionando uma validação prática do funcionamento. Por sua vez, o modo de treinamento de ouvido foi elaborado para integrar e demonstrar o uso dos demais periféricos da BitDogLab – como buzzers, matriz de LEDs RGB 5×5, LED RGB, display OLED, joystick e botões – além de enfatizar o uso dos protocolos de comunicação (USB, UART e I²C) e de conceitos avançados como o PIO para controle preciso de sinais digitais, PWM e temporização.

A etapa seguinte concentrou-se na preparação do ambiente de desenvolvimento, etapa essencial para garantir um fluxo de trabalho estável e produtivo. Inicialmente, a IDE escolhida foi o Visual Studio Code (VS Code), reconhecido por sua robustez e flexibilidade na programação em C/C++ no contexto do Raspberry Pi Pico. Foram instalados o Arm GNU Toolchain, a última versão do compilador ARM GCC e o Raspberry Pi Pico SDK, seguindo um conjunto de instruções detalhadas e adaptadas para o projeto, o que incluiu a configuração de variáveis de ambiente como PATH, PICO_SDK_PATH e PICO_TOOLCHAIN_PATH, assegurando que todas as ferramentas estivessem devidamente acessíveis.

Paralelamente à instalação das ferramentas essenciais, foram integradas diversas extensões no VS Code – especificamente as extensões C/C++, CMake, CMake Tools e a extensão oficial do Raspberry Pi Pico – que facilitaram a estruturação, a compilação, a depuração e o gerenciamento do projeto. Essas extensões contribuíram para uma organização mais eficiente do código, permitindo um gerenciamento integrado das tarefas e uma configuração padronizada do ambiente de desenvolvimento, o que foi crucial para a manutenção e evolução do projeto.

Além disso, para garantir a segurança do código e facilitar o controle de versões, todo o projeto foi integrado a um repositório no GitHub. Esse uso estratégico do versionamento permitiu salvar etapas importantes do desenvolvimento, evitando perdas e facilitando a recuperação de versões anteriores em caso de falhas ou mudanças significativas. A integração do Git ao VS Code possibilitou um gerenciamento eficiente dos commits, branches e pull requests, garantindo que cada modificação fosse registrada e revisada, contribuindo para a estabilidade e evolução contínua do projeto.

A última etapa do projeto concentrou-se na depuração. Esse processo foi realizado de forma iterativa, utilizando as ferramentas de monitoramento e diagnóstico integradas ao Visual Studio Code. Por meio do terminal integrado do VS Code, foram exibidas mensagens de depuração – utilizando funções como printf – que permitiram verificar a execução de cada trecho do código e identificar problemas em tempo real. Além disso, essa abordagem facilitou a validação das funcionalidades implementadas, desde a operação dos buzzers e da matriz de LEDs até a correta comunicação via UART entre duas placas, assegurando que o comportamento do sistema atendesse às expectativas definidas durante a fase de levantamento de requisitos.

Após a implementação correta das funcionalidades relativas ao display OLED, esse recurso também foi empregado, juntamente com os LEDs, como ferramentas visuais para auxiliar na compreensão da lógica do programa. A exibição de informações no display OLED, combinada com os feedbacks visuais proporcionados pelos LEDs, possibilitou uma análise mais intuitiva e imediata do fluxo de execução do código, permitindo que o desenvolvedor observasse em tempo real o comportamento do sistema e facilitasse a identificação e correção de eventuais falhas.

Para validar o funcionamento do projeto, foi adotada uma abordagem de testes em etapas e partes. Inicialmente, cada componente da BitDogLab foi testado individualmente. Foram verificadas, de forma isolada, as funcionalidades da matriz de LEDs, do microfone, do display OLED e dos botões, assegurando que cada um apresentasse o comportamento esperado sem interferência externa. Após essa verificação unitária, foram realizados testes de integração para confirmar a coordenação entre os módulos, garantindo que todos os recursos pudessem ser utilizados simultaneamente sem conflitos ou perda de desempenho.

Em seguida, a função de afinador foi rigorosamente testada. Para isso, utilizou-se um aplicativo de smartphone emissor de som com frequência ajustável, permitindo confirmar se o afinador era capaz de detectar com precisão a frequência emitida e identificar corretamente a faixa de frequências que o sistema suportava. Complementarmente, instrumentos musicais e afinadores de referência foram empregados para validar a utilidade prática do afinador desenvolvido, especialmente no ajuste de instrumentos musicais. Quanto ao treinamento de ouvido, os testes envolveram a verificação de que a nota emitida pelos buzzers correspondia exatamente à frequência configurada no software, através de comparações com instrumentos musicais e afinadores de referência, bem como a confirmação via mensagens no terminal de que o buzzer designado estava emitindo a nota correta. Por fim, a sincronização entre as duas BitDogLabs foi testada, verificando-se que os sinais enviados e recebidos via UART eram consistentes, assegurando uma comunicação confiável entre as placas.

Os testes demonstraram que o afinador possui uma alta capacidade de identificar corretamente as frequências, conseguindo detectar as cordas de violão e outros instrumentos musicais com precisão notável. Com uma resolução de FFT próxima a 0,45 Hz, o sistema manteve uma taxa de acerto de 100% quando a fonte sonora estava próxima ao microfone, mesmo em ambientes com ruídos ou com sinais sonoros fracos, registrando erros iguais ou inferiores a 1 Hz. Contudo, devido à taxa de amostragem empregada, o afinador apresentou limitações na identificação de frequências muito altas (superiores a 1000 Hz) ou muito baixas (inferiores a 50 Hz), embora a maioria das notas convencionais se encontre dentro da faixa em que o sistema se mostrou altamente preciso.

Em relação ao treinamento de ouvido, os testes evidenciaram que os sons emitidos pelos buzzers são funcionais para fins didáticos e para músicos iniciantes, apesar de apresentarem uma certa imprecisão. A nota emitida pelos buzzers teve dificuldade de ser captada por aplicativos de afinação em smartphones, e comparada com as notas de teclados portáteis, as diferenças eram perceptíveis ao ouvido de músicos mais experientes. Além disso, a lógica do programa – incluindo a sequência de etapas, a comunicação UART entre os dispositivos e a interação com o usuário – demonstrou um desempenho robusto, contribuindo para a confiabilidade e aplicabilidade geral do projeto. Dessa forma, o sistema se mostra apto para uso educativo, proporcionando uma ferramenta prática e integrada para o ensino de conceitos de sistemas embarcados, mesmo que haja algumas limitações na precisão sonora dos buzzers para aplicações que exijam uma fidelidade maior.

Foi gravado um vídeo demonstrando o funcionamento completo do projeto, evidenciando o desempenho de cada funcionalidade testada, incluindo a precisão do afinador e o treinamento de ouvido. O vídeo pode ser acessado pelo link: https://youtu.be/KLigtz6VqDY.

5. Referências

1. YOUTUBE. *Demonstration of Project A*. Disponível em: https://www.youtube.com/watch?v=aS0tE-y4iuQ. Acesso em: 17 fev. 2025.
2. YOUTUBE. *Project B - Live Demonstration.* Disponível em: https://www.youtube.com/live/KC0SelbhEhg. Acesso em: 17 fev. 2025.
3. GITHUB. *BitDogLab.* Disponível em: https://github.com/BitDogLab/BitDogLab. Acesso em: 17 fev. 2025.
4. LOKER, David R. *Raspberry Pi Pico as an IoT Device.* In: 2023 ASEE Annual Conference & Exposition. 2023.
5. CUGNASCA, C. E. *Projetos de Sistema Embarcados.* Departamento de Engenharia de Computação e Sistemas Digitais. Escola Politécnica da USP, 02/2018.