Projeto 2 – Sintetizador de Áudio com a BitDogLab

Desenvolvimento em C estruturado (VS Code)

O projeto do *Sintetizador de Áudio* é uma atividade focada em compreender, estruturar e implementar conceitos referentes à captura, armazenamento, processamento e reprodução de áudio digital. Para isso, deve-se dominar o funcionamento de conversores (ADC), modulação de sinais (no caso, PWM) e manipulação de memória (RAM ou via DMA).

1. Funcionamento do Sintetizador

Por definição, o sintetizador deve ser capaz de – com o microfone da BitDogLab – gravar um áudio (de duração pré-definida) e, feito isso – empregando os Buzzers ou um alto-falante externo –, reproduzir as informações de som armazenadas. Com o uso de botões, é possível controlar as funções de gravação e reprodução arbitrariamente.

Etapas de Implementação

- **Aquisição do Sinal:** O sinal analógico detectado pelo microfone (por exemplo, voz ou sons ambientes) é convertido em digital pelo ADC do microcontrolador. Este processo exige atenção à taxa de amostragem, para que o áudio seja capturado de forma clara.
- **Processamento e Armazenamento:** Após a digitalização, os dados de áudio são armazenados em memória (RAM ou cartão SD, quando desejado) e podem passar por etapas de processamento, como filtragem básica, normalização ou até compressão simples.
- **Síntese e Reprodução:** O áudio é então reproduzido por meio de PWM ou I2S. Para sinais de melhor qualidade, emprega-se um amplificador classe D conectado a um alto-falante externo. Já com o buzzer passivo, o experimento demonstra sons mais simples, mas ainda ilustra perfeitamente os princípios de geração de onda por PWM.

Conceitos de Eletrônica e Processamento de Sinais

- **Conversão Analógico-Digital (ADC):** Transformação de sinais de áudio em dados binários, fundamental para todo processamento digital subsequente.
- **Taxa de Amostragem:** Conceito crucial para garantir que o áudio capturado seja fiel à fonte original; determina o intervalo entre leituras sucessivas do sinal.
- **Filtragem e Efeitos de DSP:** Técnicas como filtros passa-baixas ou normalização evidenciam como se pode manipular digitalmente o áudio, removendo ruídos e destacando frequências de interesse.

- **Geração de Sinais (PWM/I2S):** Mostra como o microcontrolador pode recriar ou sintetizar sons, seja na forma de ondas senoidais simples ou até em sinais de áudio mais complexos.
- **Amplificação de Áudio:** Com o uso de um amplificador classe D, o sinal digital gerado se converte em potência adequada para ser reproduzido em alto-falantes, permitindo maior volume e melhor fidelidade.

O experimento une conhecimentos de eletrônica, programação, física (ondas sonoras) e matemática (frequências, amostragem e transformadas de sinal), assim como ilustra a relevância de etapas como aquisição de sinais, processamento em tempo real e otimização de recursos (DMA, buffers, memória limitada).

2. Som com PWM

PWM (Modulação de Largura de Pulso, do inglês Pulse Width Modulation), é uma técnica usada para controlar dispositivos analógicos, usando sinais digitais. Com o PWM, um dispositivo digital, como um microcontrolador, pode gerar sinais que simulam diferentes níveis de tensão analógica, permitindo o controle preciso de motores, lâmpadas, atuadores, alto-falantes, buzzers, entre outros dispositivos.

A técnica funciona variando o ciclo de trabalho (*duty cycle*) de um sinal digital, que corresponde à proporção do tempo em que o sinal permanece ativo (nível lógico alto) em relação ao tempo total do ciclo. Ao controlar essa proporção, é possível ajustar a corrente média fornecida ao componente, influenciando diretamente a potência entregue a ele.

Por exemplo, imagine uma chave elétrica conectada entre um dispositivo e uma fonte de corrente contínua. Ao abrir e fechar rapidamente essa chave de forma repetida, controlamos o tempo em que a corrente é fornecida ao dispositivo. Quanto maior o tempo em que a chave permanece fechada (maior ciclo de trabalho), maior a potência entregue; quanto menor esse tempo, menor a potência disponibilizada. Ao controlar o tempo em que a chave permanece fechada e aberta, é possível variar a largura do pulso emitido pelo dispositivo:

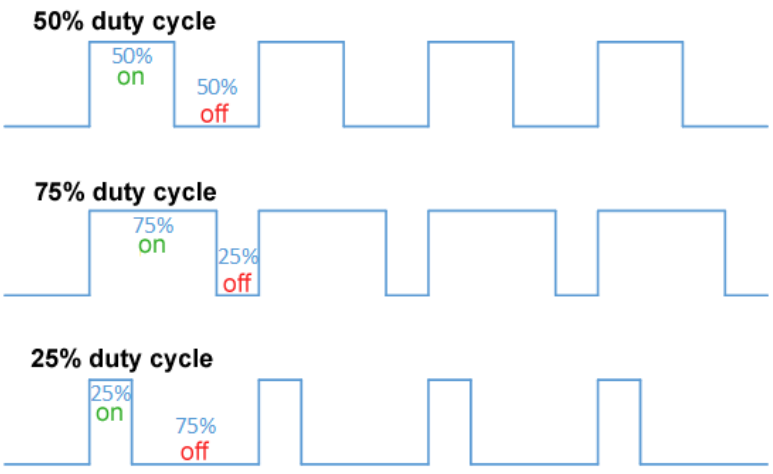


Figura – Alteração de duty cycle em PWM.

Note que o ciclo de trabalho (duty cycle) é o tempo (em relação à duração total do pulso) em que o sinal permanece no nível lógico alto. Para empregar a técnica de PWM na reprodução de áudio, basta estabelecer uma relação entre a amplitude da onda sonora (a ser emitida pelo buzzer/alto-falante) e o duty cycle do PWM – num processo similar ao usado por arquivos WAV. Ou seja, a depender da amplitude da amostra de áudio capturada em relação ao espectro, o duty cycle do PWM deve ser ajustado apropriadamente.

Na figura a seguir, o sinal senoidal (em azul) representa uma forma de onda analógica, como um sinal de áudio. As barras verticais (em verde) ilustram os pulsos gerados por modulação por largura de pulso (PWM), cujo ciclo de trabalho varia ao longo do tempo para seguir o formato da onda analógica. Ao filtrar esse sinal PWM com um circuito passivo (como um filtro RC), é possível reconstruir uma forma de onda analógica contínua, aproximando-se do sinal original.

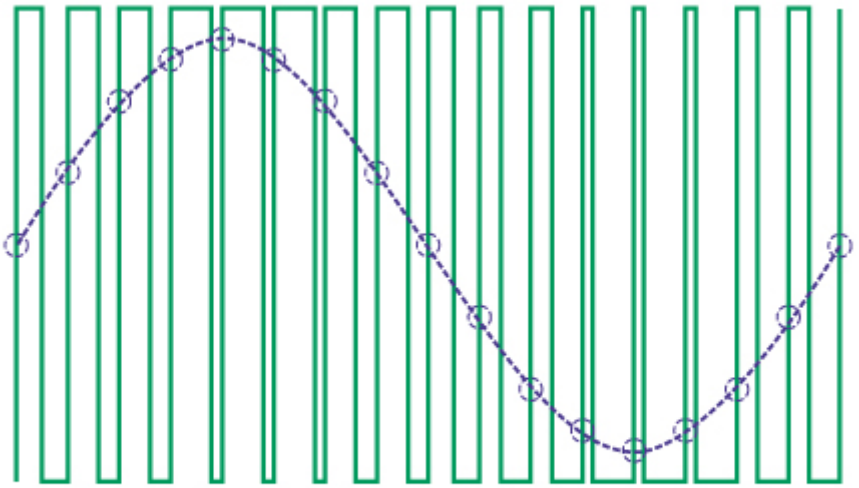


Figura – Exemplo de PWM moldando um sinal analógico.

3. Desenvolvimento do Projeto

Objetivo

Desenvolver um Sintetizador de Áudio, capaz de gravar e reproduzir áudio, dentro das especificações do sistema – duração do áudio (em segundos), taxa de amostragem, frequência do áudio reproduzido, etc.

Sugerimos a seguir algumas etapas para o desenvolvimento desta atividade:

Etapas do Desenvolvimento

1. **Aquisição do sinal do microfone (via ADC)**

    - Desenvolva uma função que aquisite o sinal do microfone (via ADC) e o envie (na forma de valor numérico) ao terminal, permitindo sua visualização. Atente para uma taxa de amostragem adequada.

- **Teste essa função:** Use um loop para chamar a função repetidamente, detectando vários valores em sequência (recomenda-se um pequeno *delay* entre cada registro). Observe a variação dos números impressos no terminal.

2. **Armazenamento dos dados de ADC num buffer**

    - Altere a função anterior para que ela, ao invés de imprimir o valor no terminal, o armazene em um buffer (tipo de lista ou array, de tamanho pré-definido) – contanto que ainda haja espaço vago nele. Pode-se usar um contador para garantir que o buffer esteja cheio ao final da gravação.

    - **Teste essa função:** Similar ao feito no item 1, use um loop para detectar (e armazenar) vários valores em sequência. Feito isso, imprima o conteúdo do buffer no terminal para comparar os valores armazenados com os números mostrados no passo anterior.

3. **Configuração da taxa de amostragem**

    - Dentro do loop, modifique o valor do delay entre os registros sequenciais, até atingir uma taxa de amostragem apropriada para uma gravação (recomendam-se frequências de 8 kHz, 11 kHz, 16 kHz ou 22 kHz, a depender da qualidade desejada).

4. **Configuração do período de gravação**

    - Modifique as especificações do buffer para que ele armazene a quantidade adequada de valores (para a taxa de amostragem escolhida) para um áudio de duração pré-definida.

    - Para decidir as especificações, tenha em mente o tamanho de cada amostra de áudio (em bytes), a taxa de amostragem (em bytes/s), a duração desejada (em s) e o espaço disponível na memória.

5. **Manipulação do sinal do PWM**

    - Desenvolva uma função que envie os sinais (gravados no buffer) ao PWM (dos Buzzers). Faça os ajustes necessários para que os valores armazenados sejam convertidos em sinais audíveis.

6. **Reprodução completa do áudio armazenado**

    - Modifique a função anterior para que ela reproduza o áudio, via PWM, dentro da frequência desejada (ajustada de acordo com a taxa de amostragem). Recomenda-se o uso de um contador, para garantir que todos os valores do buffer sejam devidamente enviados ao PWM.

- **Teste essa função:** Altere a função main() para que ela realize todas as operações (gravação, armazenamento e reprodução) em intervalos de tempo pré-programados. Ouça o áudio sendo reproduzido nos Buzzers e faça os ajustes necessários para garantir uma maior fidelidade à gravação original.

7. **Controle das funções com os botões**

    - Modifique o fluxo de controle da função principal para que, em vez de as funções serem chamadas automaticamente, cada um dos dois botões controle uma função separada. Por exemplo: ao apertar o Botão A, a BitDogLab começa a gravar o áudio, e ao apertar o Botão B (após o fim da gravação), a BitDogLab reproduz o áudio.

8. **Retorno visual de ativação (com LEDs)**

    - Adicione suporte ao LED RGB da BitDogLab – para feedback visual. Modifique as funções de gravação e reprodução para que, ao iniciar a gravação, a luz vermelha do LED acenda, e, ao iniciar a reprodução, a luz verde do LED acenda.

9. **Visualização da forma de onda no Display OLED**

    - Desenvolva uma função que, para cada amostra do áudio, desenha uma coluna de pixels no display OLED – cuja altura varia de acordo com o valor detectado pelo ADC.

10. **Aprimore a saída de áudio (opcional)**

    - Caso tenha acesso a um amplificador Classe-D, faça uma conexão usando os pinos da placa para que o áudio seja reproduzido em um alto-falante externo.

11. **Refine o processamento de áudio (opcional)**

    - O toque final para aperfeiçoar este projeto é desenvolver métodos para diminuir o ruído no áudio. Busque formas de filtrar o áudio (via software, durante a captura via ADC) e observe os efeitos que isso tem na reprodução dos sons.

Entrega usando o GitHub

Lembre-se de criar um Readme com a introdução do projeto.

É desejável que seu programa apresente:

- Funções de gravação e reprodução de áudio em C estruturado;
- Interatividade com os botões da BitDogLab;

- Feedback visual de ativação com o LED RGB;
- Visualização da forma de onda no Display OLED.

Carregue seu projeto no Github Classroom.

Dicas:

- Empenhe-se em apresentar um código limpo, modularizado e comentado;
- Faça um vídeo curto da demonstração. Inclua o link do vídeo no GitHub

Referências

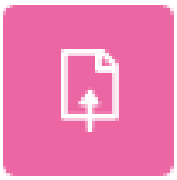
Para saber mais sobre os conceitos abordados neste projeto, recomendamos a leitura dos seguintes materiais:

- Sampling Rate – Analog Devices
- Analog-To-Digital Converters: How Does An ADC Work? – Arrow Electronics
- What is a PWM signal? – CircuitBread

Reflexão Final

- Quais técnicas de programação podemos usar para melhorar a gravação e a reprodução do áudio?

- Como é possível gravar áudios mais extensos, sem prejudicar a qualidade da gravação?



TAREFA

Projeto Sintetizador de Áudio