Nome: LUCAS DIAS DA SILVA

Unidade 7|Capítulo 1 - Projeto de Sistema Embarcado

Sistema Embarcado para Monitoramento de Níveis de Ruído

Plataforma: BitDogLab (baseada no Raspberry Pi Pico W / RP2040)

Objetivo Geral:

Desenvolver um sistema embarcado que monitore continuamente os níveis de ruído ambiente, oferecendo feedback visual e sonoro, realizando autotestes, registrando os dados em formato JSON e preparando o sistema para futura integração com comunicação Wi-Fi para monitoramento remoto.

---

1. Introdução e Justificativa

Ruídos elevados em ambientes industriais e corporativos podem impactar significativamente a saúde dos trabalhadores e a eficiência das operações. Além dos efeitos físicos – como estresse, perda auditiva e problemas cardiovasculares – ruídos também podem interferir na comunicação interna, causando erros e atrasos, e afetando a reputação da empresa.

Os ruídos nas empresas podem ser classificados em duas categorias:

- **Ruídos Ambientais:**
    - A exposição a sons elevados pode causar estresse e problemas de saúde.
    - A Organização Mundial da Saúde (OMS) considera 50 dB como limite de conforto.
    - No Brasil, o limite legal para ruído contínuo e intermitente é de 85 dB.
    - Para cada 5 dB acima de 80 dB, o tempo de exposição permitido diminui, e atividades com mais de 115 dB(A) sem proteção adequada oferecem risco grave.
- **Ruídos de Comunicação:**

- São interferências que prejudicam a comunicação entre colegas e setores, podendo originar erros, mal-entendidos, atrasos e retrabalhos.
- Tais interferências podem estar ligadas à estratégia, operação ou à forma de transmitir informações, impactando negativamente a eficiência e a imagem corporativa.

Como reduzir o ruído:

Utilização de plantas bem-posicionadas, instalação de materiais de isolamento acústico, uso de cabines à prova de som e portas acústicas são medidas que podem mitigar esses problemas.

Diante desse cenário, é fundamental dispor de um sistema de monitoramento que forneça dados em tempo real para alertas imediatos e permita análises históricas, contribuindo para a segurança ocupacional e a melhoria dos processos internos.

Com o avanço dos sistemas embarcados, torna-se viável criar soluções de monitoramento contínuo que não só alertem os operadores em tempo real, mas também armazenem os dados para análise histórica e suporte à manutenção preditiva. A utilização da BitDogLab, com seu ADC de 12 bits e a possibilidade de integração Wi-Fi, torna esse projeto acessível e expansível para aplicações industriais.

---

2. Objetivos do Projeto

Objetivos Específicos

- **Monitoramento Contínuo:**
  - Capturar, em tempo real, os níveis de ruído ambiente através de um microfone de eletreto.
  - Converter a leitura do ADC (0 a 4095) para um sinal AC centrado em 0 (subtraindo o offset de 1,65 V, que equivale aproximadamente a 2048 contagens) e calcular o nível em dB SPL com base na sensibilidade do microfone.
- **Feedback Visual e Sonoro:**
  - Exibir as medições em um display OLED (SSD1306, 128x64) via I2C.
  - Acionar LEDs:

        - **LED Verde:** Indica operação normal (níveis de ruído abaixo do limiar de alerta).
        - **LED Azul:** Pode ser utilizado para indicar status ou futuras funcionalidades.
        - **LED Vermelho:** Acionado quando o nível de ruído ultrapassa o limiar crítico.
    - Emitir alerta sonoro por meio de buzzer (configurado via PWM) quando o ruído atinge níveis perigosos.
- **Autoteste Integrado:**
    - Ao pressionar o Botão A (configurado com pull-up), o sistema deverá:
        - Realizar uma medição inicial do ruído ambiente;
        - Acionar o buzzer para gerar um tom controlado;
        - Realizar uma nova medição pelo microfone para confirmar a captação do som;
        - Comparar os resultados e exibir no OLED se o autoteste foi bem-sucedido ("Self-Test: OK") ou não ("Self-Test: FAIL").
- **Registro de Dados:**
    - Armazenar cada leitura (por exemplo, número da leitura, valor bruto do ADC, amplitude filtrada, nível em dB SPL e timestamp) em um arquivo JSON (por exemplo, "noise_data.json") para análise histórica e monitoramento preditivo.
- **Expansibilidade e Comunicação Remota:**
    - Preparar o firmware para futura integração com a conectividade Wi-Fi nativa do Pico W, possibilitando:
        - Envio dos dados via MQTT ou HTTP para um servidor/dashaboard remoto;
        - Implementação de alertas remotos e registro em tempo real na nuvem.

---

3. Descrição do Funcionamento e Fluxo do Sistema

3.1 Inicialização

- **Configuração dos Periféricos:**

- Inicialização do ADC para leitura do microfone (GPIO28 – canal ADC2).
- Configuração do display OLED via I2C (GPIO14 – SDA e GPIO15 – SCL).
- Inicialização dos LEDs (vermelho, verde e azul) e do buzzer (configurado com PWM, por exemplo, para 2000 Hz).
- Configuração dos botões (Botão A para autoteste e Botão B para reinicialização/BOOTSEL).

- **Calibração do Ruído Ambiente:**
  - Leitura de várias amostras do microfone para definir o valor de offset (ruído_base), que idealmente representa o sinal em ausência de som (aproximadamente 1,65 V ou 2048 contagens).

3.2 Autoteste (Acionado pelo Botão A)

- O sistema:
  1. Realiza uma medição do sinal (obtendo o valor bruto do ADC e removendo o offset para gerar o sinal AC).
  2. Aciona o buzzer para emitir um tom controlado.
  3. Efetua nova medição para confirmar a resposta do microfone.
  4. Compara os valores medidos e exibe no OLED o resultado do autoteste.

3.3 Monitoramento Contínuo

- **Leitura e Processamento:**
  - Leitura periódica do ADC.
  - Conversão do valor lido para sinal AC (subtraindo o offset) e filtragem (média móvel) para reduzir ruídos momentâneos.
  - Conversão do valor filtrado para tensão e, com base na sensibilidade do microfone, para pressão sonora e, finalmente, para dB SPL.
- **Feedback e Controle:**
  - Exibição dos valores (ADC, amplitude filtrada e dB SPL) no display OLED.
  - Acionamento dos LEDs e do buzzer conforme os limiares definidos:

        - Se o nível for normal (abaixo do limiar crítico), acende LED verde (ou azul conforme a função definida).
        - Se o nível exceder o limiar definido, acende LED vermelho e emite alerta sonoro.

3.4 Registro dos Dados

- Cada leitura é armazenada em um arquivo JSON com o formato, por exemplo:

{ "reading": 1, "adc": 2048, "filtered": 45, "dbSPL": 85.3, "timestamp": "2025-02-17T19:00:00Z" }

O armazenamento pode ser feito em um cartão SD (via SPI) ou na memória flash interna, utilizando bibliotecas FATFS ou similares.

3.5 Comunicação Wi-Fi (Expansão Futura)

- **Preparação do Firmware:**
    - Inicializar a conexão Wi-Fi do Pico W.
    - Formatar e enviar os dados em formato JSON para um servidor remoto (usando protocolos como MQTT ou HTTP) para monitoramento e análise em dashboard web ou mobile.

---

4. Especificação do Hardware

- **Placa BitDogLab (Raspberry Pi Pico W):**
    - Base do sistema com processador RP2040 e conectividade Wi-Fi (para expansão futura).
- **Microfone de Eletreto:**
    - Conectado ao GPIO28 (ADC2).
    - Características:
        - **Offset:** 1,65 V (valor de repouso, configurável via trimpot).
        - **Amplitude máxima:** Varia de 0 a 3,3 V (deslocamento máximo de ±1,65 V).
- **Display OLED (SSD1306, 128×64):**
    - Comunicação via I2C (SDA em GPIO14, SCL em GPIO15).
- **LED RGB:**
    - **LED Vermelho (alerta):** GPIO13

    - **LED Verde (operação normal):** GPIO11
    - **LED Azul:** GPIO12 (uso atual ou para futuras funcionalidades)
- **Buzzer:**
    - Conectado ao GPIO21, acionado via PWM para alertas e autoteste.
- **Botão A:**
    - Conectado ao GPIO5 com pull-up interno, acionando o autoteste.
- **Módulo Wi-Fi:**
    - Integrado ao Raspberry Pi Pico W, para futuras expansões.
- **Sistema de Armazenamento de Dados:**
    - Pode ser implementado com um cartão SD (via SPI) ou com sistema de arquivos na memória flash.

---

5. Especificação do Firmware

Principais Blocos de Software:

- **Leitura do ADC e Processamento do Sinal:**
    - Remoção do offset (valor de 1,65 V ≈ 2048 contagens) para obtenção do sinal AC.
    - Aplicação de filtro móvel para suavização do sinal.
    - Conversão da amplitude do sinal (em contagens) para tensão, pressão sonora e, finalmente, para dB SPL.
- **Controle do Display OLED:**
    - Exibição dos valores lidos e mensagens do autoteste.
- **Gerenciamento dos LEDs e Buzzer:**
    - Acionamento dos LEDs (vermelho para alerta, verde para operação normal, azul para funções futuras) e do buzzer de acordo com o nível do ruído e o autoteste.
- **Rotina de Autoteste:**
    - Executada quando o Botão A é pressionado, para validar o funcionamento dos sensores e do buzzer.
- **Registro de Dados em JSON:**

    - Armazenamento de cada leitura com identificação (número da leitura, valores medidos, timestamp, etc.) em um arquivo JSON.
- **Módulo de Comunicação Wi-Fi (Opcional/Futuro):**
    - Inicialização da conexão de rede e envio de payloads JSON para monitoramento remoto.

---

6. Metodologia e Desenvolvimento

Etapas de Execução:

1. **Planejamento e Definição:**
    - Levantamento dos requisitos funcionais e não funcionais.
    - Pesquisa de projetos similares e seleção dos componentes disponíveis na BitDogLab.
2. **Especificação de Hardware e Software:**
    - Elaboração dos diagramas de blocos e esquemáticos do circuito.
    - Definição detalhada das funcionalidades do firmware (leitura do ADC, processamento, autoteste, feedback, registro e comunicação).
3. **Implementação:**
    - Desenvolvimento do código em C para o RP2040, integrando os módulos (ADC, OLED, LEDs, buzzer, botões, armazenamento de dados).
    - Testes iniciais e ajustes na calibração (obtendo o valor de offset do microfone).
4. **Testes e Validação:**
    - Execução de testes de autoteste para verificar a integridade dos sensores.
    - Validação do monitoramento contínuo e dos alertas (visual e sonoro).
    - Testes do registro de dados (geração e leitura do arquivo JSON).
5. **Documentação e Apresentação:**
    - Preparação de um relatório final com escopo, diagramas, fluxogramas, código-fonte, resultados dos testes e análises dos dados.

- Publicação do código-fonte em repositório (por exemplo, GitHub) e produção de um vídeo demonstrativo do sistema.

---

7. Aplicações Industriais e Benefícios

- **Segurança Ocupacional**: Monitoramento contínuo dos níveis de ruído para garantir que os colaboradores não sejam expostos a níveis que possam causar perda auditiva ou estresse.
- **Manutenção Preditiva:** A identificação de anomalias sonoras pode antecipar falhas em máquinas e permitir intervenções preventivas.
- **Controle de Qualidade:** Verificação dos padrões sonoros em linhas de produção, ajudando a identificar problemas ou irregularidades no funcionamento de equipamentos.
- **Registro Histórico e Monitoramento Remoto:** Com o armazenamento em JSON e a futura integração com Wi-Fi, os dados poderão ser enviados para um dashboard remoto, facilitando análises e ações corretivas a longo prazo.

---

8. Possíveis Expansões Futuras

- **Integração Completa com IoT:** Desenvolvimento da funcionalidade Wi-Fi para envio contínuo dos dados a um servidor ou dashboard remoto (via MQTT/HTTP).
- **Dashboard Interativo:** Criação de uma interface web ou mobile para visualização em tempo real e análise histórica dos níveis de ruído.
- **Aprimoramento da Calibração:** Implementação de algoritmos de calibração automática (talvez com um sensor de referência) e inclusão de timestamps com RTC para um registro mais preciso.
- **Feedback Multimodal:** Uso adicional dos LEDs (por exemplo, LED azul para indicar status de comunicação ou para outras funções de diagnóstico) e integração com sistemas de alarme.

Referências

1. **BitDogLab Datasheet** – Documento técnico fornecido pelo CEPEDI, contendo as especificações e o pinout da placa BitDogLab.
2. **Raspberry Pi Pico W Datasheet** – Informações oficiais sobre o microcontrolador RP2040 e as funcionalidades de rede do Pico W (disponível no site oficial da Raspberry Pi).
3. **CUGNASCA – Metodologia de Projeto em Sistemas Embarcados** – Referência utilizada durante a capacitação, que simplifica os passos do desenvolvimento de sistemas embarcados.
4. Ganssle, J. “The Art of Designing Embedded Systems” – Referência clássica para o desenvolvimento e melhores práticas em sistemas embarcados.
5. Wolf, W. “Embedded Systems Design” – Referência adicional sobre técnicas e ferramentas no desenvolvimento de sistemas embarcados.
6. **Documentação do FATFS e bibliotecas de arquivos em sistemas embarcados** – Referências para a implementação de sistemas de arquivos em microcontroladores, essenciais para o armazenamento de dados em JSON.