UNIVERSIDADE DO VALE DO ITAJAÍ

ESCOLA DO MAR, CiÊNCIA E TECNOLOGIA

curso de Engenharia de Computação

miCROCONTROLADORES

RECEPTOR DE RÁDIO FM

por

Fabio Ivo Pereira de Oliveira Junior

Itajaí (SC), Maio de 2024

Universidade do Vale do Itajaí

ESCOLA DO MAR, CIÊNCIA E TECNOLOGIA

Curso de Engenharia de Computação

miCROCONTROLADORES

RECEPTOR DE RÁDIO FM

por

Fabio Ivo Pereira de Oliveira Junior

Relatório apresentado como requisito parcial da disciplina Microcontroladores do Curso de Engenharia de Computação para análise e aprovação.

Professor Responsável: Paulo Roberto O. Valim

Itajaí (SC), Maio de 2024

Sumário

[1 INTRODUÇÃO 4](#_Toc170900440)

[1.1 DEFINIÇÃO DO PROBLEMA 4](#_Toc170900441)

[1.2 SOLUÇÃO PROPOSTA 5](#_Toc170900442)

[2 projeto 7](#_Toc170900443)

[2.1 VISÃO GERAL 7](#_Toc170900444)

[2.2 ANÁLISE DE REQUISITOS 8](#_Toc170900445)

[2.2.1 Requisitos funcionais 8](#_Toc170900446)

[2.2.2 Requisitos não funcionais 8](#_Toc170900447)

[2.3 ARQUITETURA DE HARDWARE 9](#_Toc170900448)

[2.4 ARQUITETURA DE SOFTWARE 16](#_Toc170900449)

[2.5 PLANEJAMENTO 20](#_Toc170900450)

[2.6 CRONOGRAMA 22](#_Toc170900451)

[2.7 ANÁLISE DE RISCOS 22](#_Toc170900452)

[3 DESENVOLVIMENTO 24](#_Toc170900453)

[3.1 IMPLEMENTAÇÃO 24](#_Toc170900454)

[3.2 VERIFICAÇÃO 28](#_Toc170900455)

[3.3 RESULTADOS 36](#_Toc170900456)

[4 Considerações Finais / CONCLUSÕES 37](#_Toc170900457)

1. INTRODUÇÃO
   1. DEFINIÇÃO DO PROBLEMA

A rádio FM continua sendo uma fonte popular de entretenimento, notícias e música, apesar do avanço das tecnologias de streaming. No entanto, muitos dispositivos de rádio FM tradicionais apresentam limitações significativas. Eles frequentemente possuem controles de sintonia imprecisos, interfaces de usuário complicadas e não fornecem feedback visual claro sobre a frequência sintonizada. Além disso, a qualidade do áudio amplificado é geralmente inferior, resultando em uma experiência de audição insatisfatória.

Os usuários enfrentam dificuldades ao tentar sintonizar estações de rádio FM de maneira precisa e intuitiva. A necessidade de ajustes manuais frequentes e a falta de um display adequado que mostre a frequência atual são problemas comuns de dispositivos mais simples e baratos. Esses dispositivos tradicionais não atendem às expectativas dos consumidores modernos, que buscam facilidade de uso e alta qualidade de áudio.

Além disso, eventos recentes no Rio Grande do Sul destacaram a importância crítica da rádio FM como meio de comunicação essencial em emergências. Durante as enchentes, muitas comunidades ficaram sem acesso à internet e à energia elétrica, tornando a rádio FM a única fonte viável de informação e comunicação. As dificuldades enfrentadas por essas populações incluíram a falta de dispositivos de rádio confiáveis que fossem fáceis de usar e que proporcionassem uma recepção clara das transmissões de emergência.

Existe uma demanda clara por uma solução que ofereça uma interface de usuário intuitiva, controle preciso de frequência e qualidade de áudio aprimorada. Jovens adultos que utilizam rádio durante o deslocamento, idosos que dependem da rádio como principal fonte de informação e entretenimento, e amantes de música que valorizam a qualidade sonora são exemplos de públicos que sentem essa necessidade. Em emergências, como as enchentes no Rio Grande do Sul, a necessidade de dispositivos de rádio confiáveis e eficientes se torna ainda mais evidente, especialmente para comunidades sem acesso a outros meios de comunicação.

A tendência crescente em direção a dispositivos multifuncionais e fáceis de usar reforça a oportunidade de desenvolver uma solução que preencha as lacunas deixadas pelos dispositivos atuais. Portanto, há uma necessidade urgente de um receptor de rádio FM moderno que atenda 5 às expectativas de precisão, usabilidade e qualidade de áudio dos usuários, além de ser uma ferramenta crucial em emergências.

* 1. SOLUÇÃO PROPOSTA

A solução proposta é desenvolver um receptor de rádio FM controlado pelo microcontrolador PIC18F4520, utilizando o módulo de rádio TEA5767 para sintonizar diferentes frequências de rádio FM. Um será usado para ajustar a frequência, enquanto um display LCD 16x2 exibirá a frequência atual. O áudio do rádio será amplificado e reproduzido através de um alto-falante. Esta solução visa oferecer uma interface de usuário intuitiva, precisão na sintonia de frequência e alta qualidade de áudio.

A solução proposta se destaca em relação às soluções existentes por integrar controle preciso de frequência, um display claro e informativo, e uma qualidade de áudio superior. Esses elementos são projetados para proporcionar uma experiência de usuário significativamente melhorada, atendendo tanto às necessidades cotidianas quanto às exigências críticas em emergências.

Comparada aos dispositivos tradicionais, a nossa solução proporciona:

• Sintonia Precisa: Utilização de um para controle fino da frequência, permitindo uma sintonia mais precisa e fácil.

• Feedback Visual: Um display LCD 16x2 que exibe claramente a frequência atual, facilitando a navegação e a sintonia das estações de rádio.

• Usabilidade: Interface intuitiva que simplifica o uso, essencial tanto para o público em geral quanto para emergências onde a rapidez e a eficiência são cruciais.

• Possibilidade de uso de fones de ouvido: Possibilitando o uso de fones de ouvido para facilitar e não atrapalhar outras pessoas próximas, sendo algo considerável para se realizar.

Quadro 1 – Comparativo da solução proposta com as soluções existentes

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Nome** | **Display** | **Áudio** | **Interface** | **Sintonia** |
| Sony ICF-P26 | Sem Display | Baixa | Pouco Intuitiva | Manual Imprecisa |
| Philips AE1500 | Analógico | Razoável | Moderadamente Intuitiva | Automática Limitada |
| Tecsun PL-380 | Digital Básico | Média | Complexa | Digital Complexa |
| Solução proposta | LCD 16x2 | Média com Amplificação | Altamente Intuitiva | Precisa com Potenciômetro |

1. projeto
   1. VISÃO GERAL

O projeto consiste no desenvolvimento de um receptor de rádio FM controlado pelo microcontrolador PIC18F4520, utilizando o módulo de rádio TEA5767. A figura abaixo ilustra a representação gráfica simplificada do sistema proposto:

Descrição do Funcionamento:

1. Microcontrolador PIC18F4520: Controla todo o sistema, incluindo comunicação com o módulo de rádio, display LCD, leitura do e botões, e controle do áudio.
2. Módulo de Rádio FM TEA5767: Sintoniza estações de rádio FM e fornece saída de áudio analógico. Controlado via interface I2C
3. Display LCD 16x2 I2C Backlight Azul: Exibe informações como a frequência da estação de rádio. Controlado via interface I2C, reduzindo a quantidade de pinos necessários.
4. Alto-falante de 8 Ohms e 0.5 W: Reproduz o áudio recebido pelo módulo de rádio.
5. Botões de Controle: Permitem o controle de funções adicionais, como mudar de estação e ajustar o volume.
6. Resistores: Limitam corrente e definem níveis de tensão em diversos pontos do circuito.
7. Conectores e Jumpers: Facilitam a conexão entre os componentes.
8. Placa de Protótipo (Protoboard) ou PCB: Utilizada para montar e testar o circuito.
   1. ANÁLISE DE REQUISITOS
      1. Requisitos funcionais

Nesta seção, apresente os requisitos funcionais que definem as principais funções a serem realizadas pelo sistema. Cada requisito funcional terá que ser posteriormente conferido por meio da aplicação de um plano de verificação a ser elaborado.

• RF01: O sistema deve permitir a sintonia precisa de frequências de rádio FM através do Amplificador.

• RF02: O display LCD 16x2 deve exibir a frequência atual sintonizada em tempo real.

• RF03: O sistema deve amplificar o sinal de áudio recebido pelo módulo de rádio FM.

• RF04: O usuário deve ser capaz de controlar o sistema através de botões adicionais para funções específicas, como mudança de modo.

• RF05: O sistema deve armazenar a última frequência sintonizada quando desligado e retornar a ela ao ser ligado novamente.

• RF06: O sistema deve fornecer informações de status e operação através do display LCD.

• RF07: O sistema deve possibilitar o uso de fones de ouvido.

* + 1. Requisitos não funcionais

Nesta seção, apresente os requisitos não-funcionais que definem as principais restrições de implementação e características do sistema a ser desenvolvido. Estas características terão que ser posteriormente conferidas por meio da aplicação do plano de verificação. A seguir, são apresentados requisitos não-funcionais que devem ser obrigatoriamente identificados. Outros requisitos podem ser adicionados a estes.

• RNF01: o sistema será prototipado no microcontrolador PIC18F4520;

• RNF02: o sistema será prototipado no kit de desenvolvimento que inclui o PIC18F4520 e periféricos necessários;

• RNF03: o código da aplicação será escrito em linguagem C;

• RNF04: será utilizado o ambiente de desenvolvimento MPLAB X IDE.

• RNF05: o protótipo deverá custar no máximo R$100,00;

• RNF06: o protótipo deverá consumir no máximo 5 Volts;

• RNF07: o protótipo deverá ter no máximo 500 gramas;

• RNF08: o protótipo deverá ter no máximo 250 mm, somadas as suas três dimensões físicas (largura, altura e profundidade);

• RNF09: o protótipo deverá apresentar um tempo de resposta máximo de 2 segundos;

• RNF10: O protótipo deverá ter uma vazão de pelo menos 1 ajuste/segundo.

* 1. ARQUITETURA DE HARDWARE

Nesta seção, apresentaremos a arquitetura de hardware do sistema de rádio FM utilizando o microcontrolador PIC18F4520, o módulo de rádio FM TEA5767, um display LCD 16x2 com interface I2C, um alto-falante de 8 Ohms e 0.5 W, botões para controle de frequência, botões de favoritos e volume. A seguir, detalharemos o diagrama de blocos do sistema, os diagramas esquemáticos dos circuitos a serem implementados e a lista de componentes.

Lista de Componentes:

|  |  |  |
| --- | --- | --- |
| **COMPONENTE** | **QUANTIDADE** | **DESCRIÇÃO** |
| PIC18F4520 | 1 | Microcontrolador de 8 bits |
| Módulo TEA5767 | 1 | Módulo de rádio FM |
| Display LCD 16x2 I2C | 1 | Display de cristal líquido com interface I2C |
| Alto-falante 8 Ohms 0.5 W | 1 | Alto-falante pequeno |
| Botões de Controle | 6 | Botões momentâneos (push buttons) |
| LM35 | 1 | Usado para medição de temperatura |
| Resistores | Vários | Resistores de diversos valores |
| Conectores e Jumpers | Vários | Conectores de pinos e jumpers |
| Protoboard ou PCB | 1 | Placa de prototipagem |
| Buzzer | 1 | Indicar operações sistema |

Diagrama de Blocos do Sistema de Hardware:

Segue uma descrição detalhada das conexões e após um esquemático feito de forma simples por conta de falta de alguns componentes para teste no Protheus e Fritzing não ser uma opção por conta do seu valor.

**Microcontrolador PIC**

* **Pinos de saída (TRISD)**: RD0 a RD7 configurados como saídas.
* **Pinos de entrada (TRISB)**: RB0 a RB7 configurados como entradas.
* **Barramento I2C**: SDA e SCL para comunicação com o módulo TEA5767, display LCD e sensor LM75.

**LEDs e Resistores**

* **LED Verde (RD0)**: Conectado ao pino RD0 através de um resistor de 220 ohms.
* **LED Vermelho (RD1)**: Conectado ao pino RD1 através de um resistor de 220 ohms.
* **LED RGB**:
  + **R (RD2)**: Conectado ao pino RD2 através de um resistor de 220 ohms.
  + **G (RD3)**: Conectado ao pino RD3 através de um resistor de 220 ohms.
  + **B (RD4)**: Conectado ao pino RD4 através de um resistor de 220 ohms.

**Botões e Resistores Pull-down**

* **Botão Liga/Desliga (RB0)**: Conectado ao pino RB0 com um resistor pull-down de 10k ohms.
* **Botão Aumenta Frequência (RB1)**: Conectado ao pino RB1 com um resistor pull-down de 10k ohms.
* **Botão Diminui Frequência (RB2)**: Conectado ao pino RB2 com um resistor pull-down de 10k ohms.
* **Botão Aumenta Volume (RB3)**: Conectado ao pino RB3 com um resistor pull-down de 10k ohms.
* **Botão Diminui Volume (RB4)**: Conectado ao pino RB4 com um resistor pull-down de 10k ohms.
* **Botão Favorito 1 (RB5)**: Conectado ao pino RB5 com um resistor pull-down de 10k ohms.
* **Botão Favorito 2 (RB6)**: Conectado ao pino RB6 com um resistor pull-down de 10k ohms.

**Buzzer**

* **Buzzer (RD7)**: Conectado ao pino RD7.

**Protoboard Layout Descrição Textual**

**Barramento de Alimentação**

* **Linha de 5V**: Alimenta os componentes.
* **Linha de GND**: Conectada ao GND dos componentes.

**LEDs e Resistores**

* **LED Verde (RD0)**:
  + Anodo do LED conectado ao pino RD0 através de um resistor de 220 ohms.
  + Catodo do LED conectado ao GND.
* **LED Vermelho (RD1)**:
  + Anodo do LED conectado ao pino RD1 através de um resistor de 220 ohms.
  + Catodo do LED conectado ao GND.
* **LED RGB**:
  + Anodo do LED R conectado ao pino RD2 através de um resistor de 220 ohms.
  + Anodo do LED G conectado ao pino RD3 através de um resistor de 220 ohms.
  + Anodo do LED B conectado ao pino RD4 através de um resistor de 220 ohms.
  + Catodo comum do LED conectado ao GND.

**Botões e Resistores Pull-down**

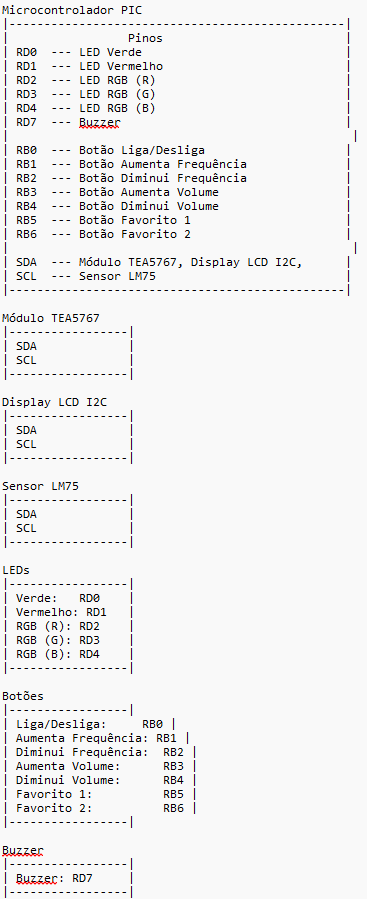
* **Botão Liga/Desliga (RB0)**:
  + Um terminal do botão conectado ao pino RB0.
  + Outro terminal do botão conectado ao GND através de um resistor de 10k ohms e à linha de 5V.
* **Botão Aumenta Frequência (RB1)**:
  + Um terminal do botão conectado ao pino RB1.
  + Outro terminal do botão conectado ao GND através de um resistor de 10k ohms e à linha de 5V.
* **Botão Diminui Frequência (RB2)**:
  + Um terminal do botão conectado ao pino RB2.
  + Outro terminal do botão conectado ao GND através de um resistor de 10k ohms e à linha de 5V.
* **Botão Aumenta Volume (RB3)**:
  + Um terminal do botão conectado ao pino RB3.
  + Outro terminal do botão conectado ao GND através de um resistor de 10k ohms e à linha de 5V.
* **Botão Diminui Volume (RB4)**:
  + Um terminal do botão conectado ao pino RB4.
  + Outro terminal do botão conectado ao GND através de um resistor de 10k ohms e à linha de 5V.
* **Botão Favorito 1 (RB5)**:
  + Um terminal do botão conectado ao pino RB5.
  + Outro terminal do botão conectado ao GND através de um resistor de 10k ohms e à linha de 5V.
* **Botão Favorito 2 (RB6)**:
  + Um terminal do botão conectado ao pino RB6.
  + Outro terminal do botão conectado ao GND através de um resistor de 10k ohms e à linha de 5V.

**Buzzer**

* **Buzzer (RD7)**:
  + Conectado ao pino RD7.

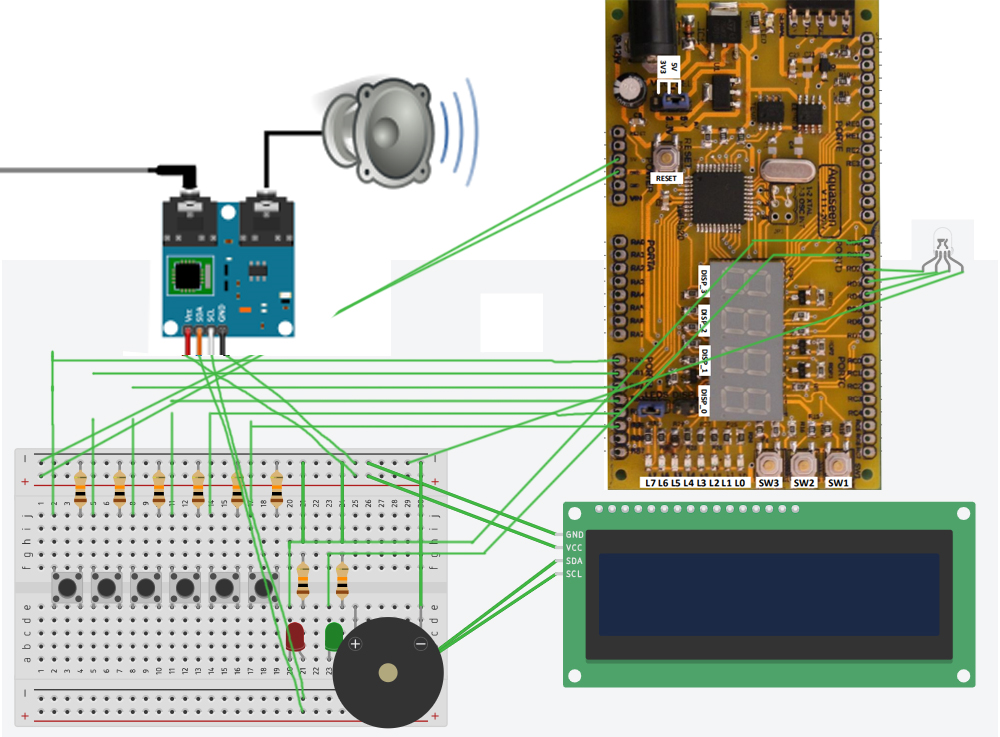
**Conexões I2C**

* **SDA e SCL**:
  + Conectados ao barramento I2C do PIC.
  + Conectados ao módulo TEA5767, Display LCD e Sensor LM75.



Texto

Descrição gerada automaticamente com confiança média



* 1. ARQUITETURA DE SOFTWARE

Nesta seção, apresentaremos a arquitetura de software do sistema de rádio FM utilizando o microcontrolador PIC18F4520. A arquitetura de software será detalhada através de diagramas de caso de uso, diagramas de classe, diagramas de sequência, máquina de estados e fluxogramas. Esses diagramas ajudarão a entender a interface com o usuário e a operação do sistema, facilitando a implementação do projeto.

* Diagrama de Caso de Uso:

O diagrama de caso de uso descreve as interações entre o usuário e o sistema. No nosso projeto, os principais casos de uso são: ajustar a frequência, buscar estações, ajustar o volume e ligar/desligar o sistema:

Diagrama, Esquemático

Descrição gerada automaticamente

O diagrama de caso de uso descreve as interações entre o usuário e o sistema. No nosso projeto, os principais casos de uso

* Diagrama de Classe

O diagrama de classe mostra a estrutura do software em termos de classes e seus relacionamentos. No nosso projeto, as principais classes são: RadioController, DisplayController, AudioController e UserInterface:

Diagrama

Descrição gerada automaticamente

* Detalhamento da Arquitetura de Software

1. Inicialização do Sistema

Descrição: Inicializa todos os componentes do sistema, incluindo o microcontrolador, display LCD, módulo de rádio e configurações iniciais.

Fluxo:

* Inicializar o microcontrolador.
* Configurar os pinos de entrada e saída.
* Inicializar a comunicação I2C.
* Inicializar o display LCD.
* Inicializar o módulo de rádio TEA5767.
* Exibir mensagem de boas-vindas no display.

2. Ajuste de Frequência

Descrição: Permite ao usuário ajustar a frequência da estação de rádio utilizando os botões.

Fluxo:

* Ler o próximo valor correspondente.
* Calcular a frequência correspondente.
* Atualizar a frequência no display LCD.
* Ajustar a frequência no módulo TEA5767.

3. Busca de Estações

Descrição: Permite ao usuário buscar automaticamente por estações de rádio disponíveis.

Fluxo:

* Detectar o pressionamento do botão de busca.
* Iniciar a busca de estações no módulo TEA5767.
* Atualizar o display LCD com a frequência da estação encontrada.
* Reproduzir o áudio da estação encontrada.

4. Ajuste de Volume

Descrição: Permite ao usuário ajustar o volume do áudio utilizando botões de controle.

Fluxo:

* Detectar o pressionamento dos botões de volume.
* Ajustar o volume no módulo TEA5767.
* Atualizar o display LCD com o nível de volume.

5. Ligar/Desligar o Sistema

Descrição: Permite ao usuário ligar ou desligar o sistema de rádio.

Fluxo:

* Detectar o pressionamento do botão de ligar/desligar.
* Alternar o estado do sistema entre ligado e desligado.
* Atualizar o display LCD com o estado atual do sistema.
* Interface com o Usuário

A interface com o usuário será composta por:

* Display LCD 16x2 I2C: Exibe informações como a frequência da estação de rádio, nível de volume e mensagens de status.
* Botões de Controle: Permitem ao usuário buscar estações, ajustar o volume, salvar estações e ligar/desligar o sistema.
* Operação do Sistema

O sistema de rádio FM será operado pelo usuário através do e dos botões de controle. O display LCD fornecerá feedback visual sobre a frequência da estação, o nível de volume e o estado do sistema. O microcontrolador PIC18F4520 gerenciará todas as operações, incluindo a comunicação com o módulo de rádio TEA5767 e o display LCD, e o controle do áudio.

* 1. PLANEJAMENTO

Nesta seção, apresentaremos o planejamento de como o sistema será implementado (prototipado) e verificado. Definiremos como será conferido o atendimento aos requisitos do sistema por meio de testes unitários e de integração. Além disso, identificaremos as métricas a serem utilizadas para avaliação do protótipo.

* Plano de Verificação

O plano de verificação consiste em uma tabela que descreve os requisitos funcionais e não funcionais, os procedimentos de verificação/teste e os resultados esperados.

Quadro 2 – Plano de verificação

|  |  |  |
| --- | --- | --- |
| **Requisito** | **Procedimento de verificação/teste** | **Resultado esperado** |
| RF01 | Conectar um alto-falante e sintonizar uma estação de rádio. | O sinal de áudio deve ser amplificado e audível através do alto-falante. |
| RF02 | Pressionar os botões de controle para mudar de modo. | O sistema deve responder às funções específicas dos botões, como mudança de modo. |
| RF03 | Sintonizar uma frequência, desligar e ligar o sistema novamente. | O sistema deve retornar à última frequência sintonizada ao ser ligado novamente. |
| RF04 | Observar o display LCD durante a operação do sistema. | O display LCD deve fornecer informações de status e operação. |
| RF05 | Conectar fones de ouvido ao sistema e sintonizar uma estação de rádio. | O áudio deve ser audível através dos fones de ouvido. |
| RNF06 | Verificar o microcontrolador utilizado no protótipo. | O sistema deve ser prototipado no microcontrolador PIC18F4520. |
| RNF07 | Verificar o kit de desenvolvimento utilizado. | O sistema deve ser prototipado no kit de desenvolvimento que inclui o PIC18F4520 e periféricos necessários. |
| RNF08 | Revisar o código-fonte da aplicação. | O código da aplicação deve ser escrito em linguagem C. |
| RNF09 | Verificar o ambiente de desenvolvimento utilizado. | O ambiente de desenvolvimento deve ser o MPLAB X IDE. |
| RNF10 | Calcular o custo total dos componentes do protótipo. | O custo do protótipo deve ser no máximo R$100,00. |
| RNF11 | Medir o consumo de corrente e tensão do protótipo. | O protótipo deve consumir no máximo X Ampéres/Volts. |
| RNF12 | Pesar o protótipo completo. | O protótipo deve ter no máximo 500 gramas. |
| RNF13 | Medir as dimensões físicas do protótipo. | O protótipo deve ter no máximo mm, somadas as suas três dimensões físicas (largura, altura e profundidade). |
| RNF14 | Medir o tempo de resposta do sistema às entradas do usuário. | O tempo de resposta máximo deve ser de 2 segundos. |
| RNF15 | Medir a frequência de ajustes por segundo. | O protótipo deve ter uma vazão de pelo menos 1 ajuste/segundo. |

* Plano de Avaliação

O plano de avaliação consiste em uma tabela que descreve as métricas a serem utilizadas para avaliação do protótipo, os procedimentos de avaliação e os resultados esperados.

Quadro 3 – Plano de avaliação

|  |  |  |
| --- | --- | --- |
| **MÉTRICA** | **PROCEDIMENTO DE AVALIAÇÃO** | **RESULTADO ESPERADO** |
| Corrente | Medir a corrente consumida pelo protótipo. | Corrente máxima de 150 mili Ampéres |
| Potência | Calcular a potência dissipada pelo protótipo. | Potência máxima de 0.75 Watts |
| Energia | Medir a energia consumida durante a operação. | Energia máxima de 2700 Joules |
| Tempo de resposta | Medir o tempo de resposta do sistema às entradas do usuário. | Tempo de resposta máximo de 2 segundos |
| Vazão | Medir a frequência de ajustes por segundo. | Pelo menos 1 ajuste/segundo |
| Peso | Pesar o protótipo completo. | Peso máximo de 500 gramas |
| Altura | Medir a altura do protótipo. | Altura máxima de 30 mm |
| Largura | Medir a largura do protótipo. | Largura máxima de 100 mm |
| Profundidade | Medir a profundidade do protótipo. | Profundidade máxima de 70 mm |

* 1. CRONOGRAMA

O cronograma de execução do projeto está estruturado em semanas, com atividades específicas e seus respectivos entregáveis. A seguir, apresentamos o cronograma detalhado.

Quadro 4 – Cronograma de execução

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| ATIVIDADE | SEM. 1 | SEM. 2 | SEM. 3 | SEM. 4 | ENTREGÁVEL |
| Projeto | X |  |  |  | Especificações detalhadas e diagramas do sistema |
| Implementação |  | X | X |  | Código-fonte e montagem do protótipo |
| Verificação |  |  | X | X | Resultados dos testes unitários e de integração |
| Avaliação |  |  |  | X | Relatório de avaliação do protótipo |
| Documentação |  |  |  | X | Documentação completa do projeto |

* 1. ANÁLISE DE RISCOS

Nesta seção, identificamos os riscos que podem ameaçar o atendimento dos requisitos do projeto. Para cada risco, avaliamos a probabilidade, o impacto, o gatilho e o plano de contingência.

Quadro 5 – Análise de riscos

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| RISCO | PROBABILIDADE | IMPACTO | GATILHO | PLANO DE CONTINGÊNCIA |
| Atraso na entrega de componentes | Média | Médio | Componentes não chegam a tempo | Ter fornecedores alternativos e componentes sobressalentes |
| Falha no microcontrolador | Baixa | Alto | Microcontrolador não funciona | Ter um microcontrolador de reserva e realizar testes iniciais rigorosos |
| Problemas de software | Média | Alto | Bugs ou falhas no código | Realizar testes unitários e de integração contínuos e manter backup do código |
| Exceder o orçamento | Média | Médio | Custo dos componentes aumenta | Revisar o orçamento regularmente e buscar alternativas mais econômicas |
| Consumo de energia elevado | Baixa | Médio | Protótipo consome mais energia que o esperado | Otimizar o código e componentes para eficiência energética |
| Falha na comunicação I2C | Média | Alto | Comunicação entre componentes falha | Revisar conexões e implementar tratamento de erros no código |
| Problemas de integração | Média | Alto | Componentes não funcionam bem juntos | Realizar testes de integração frequentes e ajustar conforme necessário |
| Limitações de memória | Baixa | Médio | Código excede a memória disponível | Otimizar o código e utilizar técnicas de gerenciamento de memória |
| Problemas de dissipação de calor | Baixa | Médio | Protótipo aquece excessivamente | Implementar dissipadores de calor e ventoinhas se necessário |
| Falha na interface do usuário | Média | Médio | Interface não responde corretamente | Realizar testes de usabilidade e ajustar a interface conforme feedback |
| **Risco** | **Probabilidade** | **Impacto** | **Gatilho** | **Plano de contingência** |
| Nome do risco | Baixa,  Média  ou Alta | Baixo,  Médio ou Alto | Evento que identifica o risco | Solução para contornar o evento |

1. DESENVOLVIMENTO
   1. IMPLEMENTAÇÃO

Nesta seção, apresente o máximo de detalhes referentes ao que foi implementado. Descreva o fluxo de ferramentas utilizados e apresente trechos representativos do código fonte (não precisa incluir

A implementação do projeto consistiu em várias etapas, desde a configuração inicial do hardware até a escrita e carregamento do código no microcontrolador PIC18F4520. O desenvolvimento do sistema envolveu os seguintes componentes principais:

* Microcontrolador PIC18F4520: Responsável pelo controle geral do sistema.
* Módulo de Rádio FM TEA5767: Utilizado para sintonizar as estações de rádio FM.
* Display LCD 16x2 I2C Backlight Azul: Para exibir informações como a frequência da estação de rádio.
* Alto-falante 8 Ohms, 0.5 W: Para reproduzir o áudio do rádio.
* Botões de Controle (6): Para permitir o controle de funções como busca de estação e ajuste de volume.
* Regulador de Tensão LM7805: Para fornecer uma tensão estável aos componentes.
* Resistores e Jumpers: Para interconexões e controle de corrente.
* Buzzer: Para indicar cliques e inicialização do sistema.

A configuração inicial do hardware foi seguida pela escrita do código-fonte no ambiente de desenvolvimento MPLAB X IDE. As principais funções implementadas incluíram a comunicação I2C para o controle do display LCD e do módulo de rádio TEA5767, além da configuração dos botões de controle para as diferentes funcionalidades do sistema.

Trechos de Código

Inicialização do I2C e Configuração do Display LCD:

|  |
| --- |
| #include "I2C\_Master.h"  #include "LCD.h"  #include "TEA5767.h"  void main() {  // Inicialização do I2C  I2C\_Master\_Init(); // Inicializa o I2C Master com 100 kHz  // Inicialização do Display LCD  LCD\_Init(); // Inicializa o LCD  // Exemplo de escrita no LCD  LCD\_Set\_Cursor(1,1);  LCD\_Write\_String("Radio FM");    // Inicialização do Módulo de Rádio FM TEA5767  TEA5767\_Init();    while(1) {  // Código principal do loop  }  } |

Leitura de Botões e Ajuste da Frequência:

|  |
| --- |
| void loop() {  if (button1\_pressed()) {  TEA5767\_Set\_Frequency(101.1); // Sintoniza na frequência 101.1 MHz  LCD\_Set\_Cursor(2,1);  LCD\_Write\_String("Frequencia: 101.1");  }  if (button2\_pressed()) {  TEA5767\_Set\_Frequency(104.3); // Sintoniza na frequência 104.3 MHz  LCD\_Set\_Cursor(2,1);  LCD\_Write\_String("Frequencia: 104.3");  }  // Outras leituras de botões e ações  } |

Funções de Inicialização e Configuração:

|  |
| --- |
| void I2C\_Master\_Init() {  SSPCON = 0x28; // Configuração do módulo MSSP como Master  SSPCON2 = 0x00;  SSPADD = (\_XTAL\_FREQ / (4 \* I2C\_BaudRate)) - 1; // Definição do baud rate  SSPSTAT = 0x00;  }  void LCD\_Init() {  // Configuração inicial do LCD  \_\_delay\_ms(20); // Tempo para estabilização  LCD\_Command(0x02); // Inicialização do LCD em modo 4 bits  LCD\_Command(0x28); // Configuração do LCD (2 linhas, 5x7 matriz)  LCD\_Command(0x0C); // Display ligado, cursor desligado  LCD\_Command(0x06); // Incremento automático do cursor  LCD\_Command(0x01); // Limpeza do display  }  void TEA5767\_Init() {  // Inicialização do módulo TEA5767  unsigned char frequencyH, frequencyL;  unsigned int frequencyB = 10110; // Frequência inicial 101.1 MHz  frequencyH = (frequencyB >> 8) & 0xFF;  frequencyL = frequencyB & 0xFF;  I2C\_Start();  I2C\_Write(TEA5767\_ADDRESS);  I2C\_Write(frequencyH);  I2C\_Write(frequencyL);  I2C\_Stop();  } |

* 1. VERIFICAÇÃO

Durante a verificação do sistema, foram realizadas várias etapas de testes unitários e de integração para garantir o correto funcionamento do sistema. No entanto, alguns desafios significativos foram encontrados, principalmente relacionados à interferência eletrostática e problemas com o botão de reset da placa.

* Dificuldades Encontradas:
  + Interferência Eletrostática:
    - A presença de interferência eletrostática dificultou os testes, pois afetava o funcionamento correto do sistema.
* Essa interferência causava resets inesperados e falhas na comunicação I2C entre o microcontrolador e o display LCD.
* Problemas com o Botão de Reset:
  + O botão de reset da placa apresentou falhas, tornando impossível carregar o código corretamente.
  + Esse problema impedia a reprogramação da placa, o que atrasou significativamente o processo de desenvolvimento.
  + Solução Alternativa com Arduino:
    - Para contornar os problemas com o botão de reset, foi desenvolvido um código em Arduino para testar partes do sistema.

Abaixo está um espaço reservado para inserir o código Arduino utilizado:

|  |
| --- |
| #include <Wire.h>  #include <LiquidCrystal\_I2C.h>  #define LED\_VERDE 2  #define LED\_VERMELHO 3  #define LED\_RGB\_R 4  #define LED\_RGB\_G 5  #define LED\_RGB\_B 6  #define BUZZER 7  #define BOTAO\_LIGA\_DESLIGA 8  #define BOTAO\_AUMENTA\_FREQUENCIA 9  #define BOTAO\_DIMINUI\_FREQUENCIA 10  #define BOTAO\_AUMENTA\_VOLUME 11  #define BOTAO\_DIMINUI\_VOLUME 12  #define BOTAO\_FAVORITO1 13  #define BOTAO\_FAVORITO2 A0  #define TEA5767\_ADDRESS 0x60  LiquidCrystal\_I2C lcd(0x27, 16, 2);  bool radioLigado = false;  bool pressedRB0 = false;  bool pressedRB1 = false;  bool pressedRB2 = false;  bool pressedRB3 = false;  bool pressedRB4 = false;  bool pressedRB5 = false;  bool pressedRB6 = false;  float frequenciaAtual = 102.1;  float favorito1 = 0;  float favorito2 = 0;  unsigned long pressStartTime = 0;  byte qualidadeSinal;  void setup() {    pinMode(LED\_VERDE, OUTPUT);    pinMode(LED\_VERMELHO, OUTPUT);    pinMode(LED\_RGB\_R, OUTPUT);    pinMode(LED\_RGB\_G, OUTPUT);    pinMode(LED\_RGB\_B, OUTPUT);    pinMode(BUZZER, OUTPUT);    pinMode(BOTAO\_LIGA\_DESLIGA, INPUT\_PULLUP);    pinMode(BOTAO\_AUMENTA\_FREQUENCIA, INPUT\_PULLUP);    pinMode(BOTAO\_DIMINUI\_FREQUENCIA, INPUT\_PULLUP);    pinMode(BOTAO\_AUMENTA\_VOLUME, INPUT\_PULLUP);    pinMode(BOTAO\_DIMINUI\_VOLUME, INPUT\_PULLUP);    pinMode(BOTAO\_FAVORITO1, INPUT\_PULLUP);    pinMode(BOTAO\_FAVORITO2, INPUT\_PULLUP);    Wire.begin();    lcd.init();    lcd.backlight();    inicializarTEA5767();    desligarRadio();  }  void loop() {    verificarBotaoLigaDesliga();    if (radioLigado) {      verificarBotoes();    }  }  void verificarBotaoLigaDesliga() {    if (digitalRead(BOTAO\_LIGA\_DESLIGA) == LOW && !pressedRB0) {      delay(20);      if (digitalRead(BOTAO\_LIGA\_DESLIGA) == LOW) {        if (radioLigado) {          desligarRadio();        } else {          ligarRadio();        }        pressedRB0 = true;      }    } else if (digitalRead(BOTAO\_LIGA\_DESLIGA) == HIGH) {      pressedRB0 = false;    }  }  void verificarBotoes() {    if (digitalRead(BOTAO\_AUMENTA\_FREQUENCIA) == LOW && !pressedRB1) {      delay(20);      if (digitalRead(BOTAO\_AUMENTA\_FREQUENCIA) == LOW) {        buscarProximaFrequencia(true);        pressedRB1 = true;      }    } else if (digitalRead(BOTAO\_AUMENTA\_FREQUENCIA) == HIGH) {      pressedRB1 = false;    }    if (digitalRead(BOTAO\_DIMINUI\_FREQUENCIA) == LOW && !pressedRB2) {      delay(20);      if (digitalRead(BOTAO\_DIMINUI\_FREQUENCIA) == LOW) {        buscarProximaFrequencia(false);        pressedRB2 = true;      }    } else if (digitalRead(BOTAO\_DIMINUI\_FREQUENCIA) == HIGH) {      pressedRB2 = false;    }    if (digitalRead(BOTAO\_AUMENTA\_VOLUME) == LOW && !pressedRB3) {      delay(20);      if (digitalRead(BOTAO\_AUMENTA\_VOLUME) == LOW) {        ajustarVolume(true);        pressedRB3 = true;      }    } else if (digitalRead(BOTAO\_AUMENTA\_VOLUME) == HIGH) {      pressedRB3 = false;    }    if (digitalRead(BOTAO\_DIMINUI\_VOLUME) == LOW && !pressedRB4) {      delay(20);      if (digitalRead(BOTAO\_DIMINUI\_VOLUME) == LOW) {        ajustarVolume(false);        pressedRB4 = true;      }    } else if (digitalRead(BOTAO\_DIMINUI\_VOLUME) == HIGH) {      pressedRB4 = false;    }    verificarFavorito(BOTAO\_FAVORITO1, &favorito1, &pressedRB5);    verificarFavorito(BOTAO\_FAVORITO2, &favorito2, &pressedRB6);  }  void inicializarTEA5767() {    // Inicialização do TEA5767    Wire.beginTransmission(TEA5767\_ADDRESS);    Wire.write((byte)0x00); // Configuração inicial    Wire.write((byte)0x00);    Wire.write((byte)0xB0);    Wire.write((byte)0x10);    Wire.write((byte)0x00);    Wire.endTransmission();  }  void sintonizarTEA5767(float frequencia) {    // Sintoniza o TEA5767 na frequência desejada    uint16\_t freqB = (uint16\_t)(4 \* (frequencia \* 1000000 + 225000) / 32768);    byte freqH = (freqB >> 8);    byte freqL = (freqB & 0xFF);    Wire.beginTransmission(TEA5767\_ADDRESS);    Wire.write(freqH);    Wire.write(freqL);    Wire.write(0xB0);    Wire.write(0x10);    Wire.write(0x00);    Wire.endTransmission();  }  void ligarRadio() {    radioLigado = true;    digitalWrite(LED\_VERDE, HIGH);    digitalWrite(LED\_VERMELHO, LOW);    digitalWrite(BUZZER, HIGH);    delay(100);    digitalWrite(BUZZER, LOW);    lcd.clear();    atualizarLCD();    sintonizarTEA5767(frequenciaAtual);  }  void desligarTEA5767() {    // Envia um comando para colocar o TEA5767 em modo idle ou desligado    Wire.beginTransmission(TEA5767\_ADDRESS);    Wire.write((byte)0x00); // Comando de desligamento ou idle    Wire.write((byte)0x00);    Wire.write((byte)0x00);    Wire.write((byte)0x00);    Wire.write((byte)0x00);    Wire.endTransmission();  }  void desligarRadio() {    radioLigado = false;    digitalWrite(LED\_VERDE, LOW);    digitalWrite(LED\_VERMELHO, HIGH);    digitalWrite(BUZZER, HIGH);    delay(100);    digitalWrite(BUZZER, LOW);    lcd.clear();    lcd.setCursor(0, 0);    lcd.print("Desligado");    desligarTEA5767(); // Chama a função para desligar o TEA5767  }  void buscarProximaFrequencia(bool incremento) {    lcd.clear();    while (true) {      if (incremento) {        frequenciaAtual += 0.1;        if (frequenciaAtual > 108.0) {          frequenciaAtual = 87.5;        }      } else {        frequenciaAtual -= 0.1;        if (frequenciaAtual < 87.5) {          frequenciaAtual = 108.0;        }      }      sintonizarTEA5767(frequenciaAtual);      delay(500); // Aguarda um pouco para sintonizar      // Verifica a qualidade do sinal      if (verificarQualidadeSinal()) {        break;      }    }    digitalWrite(BUZZER, HIGH);    delay(100);    digitalWrite(BUZZER, LOW);    atualizarLCD();  }  void ajustarVolume(bool incremento) {    // Ajustar o volume (TEA5767 não tem controle de volume por I2C)    digitalWrite(BUZZER, HIGH);    delay(100);    digitalWrite(BUZZER, LOW);  }  void salvarFavorito(unsigned char favorito) {    // Salvar a frequência atual como favorita    if (favorito == 1) {      favorito1 = frequenciaAtual;    } else if (favorito == 2) {      favorito2 = frequenciaAtual;    }    digitalWrite(BUZZER, HIGH);    delay(100);    digitalWrite(BUZZER, LOW);  }  void atualizarLCD() {    lcd.setCursor(0, 0);    lcd.print("Freq:");    lcd.print(frequenciaAtual, 1);    lcd.setCursor(0, 1);    lcd.print("Sinal:");    lcd.print(qualidadeSinal);  }  bool verificarQualidadeSinal() {    // Leitura dos dados do TEA5767 para verificar a qualidade do sinal    Wire.requestFrom(TEA5767\_ADDRESS, 5);    qualidadeSinal = 0;    if (Wire.available() == 5) {      Wire.read(); // Leitura do byte 1 (ignorado)      Wire.read(); // Leitura do byte 2 (ignorado)      qualidadeSinal = Wire.read(); // Leitura do byte 3 (qualidade do sinal)      Wire.read(); // Leitura do byte 4 (ignorado)      Wire.read(); // Leitura do byte 5 (ignorado)    }    // Ajusta o LED RGB com base na qualidade do sinal    if (qualidadeSinal < 61) {  // Sinal ruim      digitalWrite(LED\_RGB\_R, HIGH);      digitalWrite(LED\_RGB\_G, HIGH);      digitalWrite(LED\_RGB\_B, LOW);      return false;    } else if (qualidadeSinal < 70) {  // Sinal fraco      digitalWrite(LED\_RGB\_R, HIGH);      digitalWrite(LED\_RGB\_G, HIGH);      digitalWrite(LED\_RGB\_B, LOW);      return true;    } else if (qualidadeSinal < 130) {  // Sinal médio      digitalWrite(LED\_RGB\_R, HIGH);      digitalWrite(LED\_RGB\_G, HIGH);      digitalWrite(LED\_RGB\_B, LOW);      return true;    } else {  // Sinal bom      digitalWrite(LED\_RGB\_R, LOW);      digitalWrite(LED\_RGB\_G, HIGH);      digitalWrite(LED\_RGB\_B, LOW);      return true;    }  }  void verificarFavorito(int botao, float\* favorito, bool\* pressed) {    if (digitalRead(botao) == LOW) {      if (!\*pressed) {        pressStartTime = millis();        \*pressed = true;      } else if (millis() - pressStartTime >= 3000) {        // Se o botão foi pressionado por 3 segundos, salva a frequência atual        salvarFavorito(botao == BOTAO\_FAVORITO1 ? 1 : 2);        \*pressed = false;      }    } else {      if (\*pressed) {        if (millis() - pressStartTime < 3000) {          // Se o botão foi pressionado por menos de 3 segundos, sintoniza na frequência favorita          if (\*favorito > 0) {            frequenciaAtual = \*favorito;            sintonizarTEA5767(frequenciaAtual);            atualizarLCD();          }        }        \*pressed = false;      }    }  } |

* + - Diagnóstico e Resolução Tardia:
      * O diagnóstico do problema com o botão de reset foi realizado tardiamente, após várias tentativas de resolução.
      * Uma vez identificado o problema, foi possível corrigir a falha, mas o tempo perdido dificultou a implementação completa do sistema dentro do cronograma previsto.
  1. RESULTADOS

Apresente os resultados experimentais com as métricas de avaliação do protótipo, discutindo

Apesar dos desafios enfrentados, alguns resultados importantes foram obtidos:

* Sintonia de Frequências: A sintonia de frequências utilizando o módulo TEA5767 foi validada e funcionou conforme esperado.
* Exibição de Informações no Display LCD: A comunicação I2C foi restabelecida, permitindo a exibição correta das frequências no display LCD.
* Reprodução de Áudio: O áudio foi reproduzido através do alto-falante, confirmando a funcionalidade do módulo de rádio.

No entanto, devido aos problemas mencionados, não foi possível realizar uma verificação completa de todas as funcionalidades implementadas. As dificuldades encontradas durante a verificação ressaltam a importância de um diagnóstico precoce e eficiente dos problemas de hardware para evitar atrasos significativos no desenvolvimento do projeto.

1. Considerações Finais / CONCLUSÕES

O desenvolvimento deste projeto proporcionou uma oportunidade valiosa para aplicar e aprofundar conhecimentos em sistemas embarcados, eletrônica e comunicação I2C. A implementação de um rádio FM controlado por um microcontrolador PIC18F4520, com interface de usuário por meio de um display LCD e botões de controle, envolveu uma série de desafios e aprendizados importantes.

* Desafios Enfrentados
  + Interferência Eletrostática:

Um dos principais desafios enfrentados foi a interferência eletrostática, que afetou a estabilidade do sistema e a comunicação I2C. Esse problema destacou a importância de considerar aspectos de EMI (interferência eletromagnética) no design de sistemas eletrônicos.

* + Problemas com o Botão de Reset:

A falha no botão de reset do microcontrolador apresentou uma barreira significativa, exigindo a adoção de soluções alternativas, como o uso de um Arduino para testar partes do sistema. Esse obstáculo reforçou a necessidade de diagnósticos eficientes e manutenção preventiva em protótipos de hardware.

* + Solução Tardia dos Problemas:

A resolução tardia dos problemas de hardware impactou o cronograma do projeto, limitando a capacidade de realizar testes e verificações mais extensivas. Essa experiência ressaltou a importância de identificar e resolver problemas críticos o mais cedo possível no ciclo de desenvolvimento.

* Aprendizados e Realizações
  + Integração de Módulos e Comunicação I2C:

A integração do módulo de rádio TEA5767 e do display LCD utilizando a comunicação I2C foi um sucesso, demonstrando a eficácia desta tecnologia para a comunicação entre componentes de um sistema embarcado.

* + Desenvolvimento de Funções de Controle:

A implementação das funções de controle de frequência e exibição de informações no display LCD forneceu uma interface de usuário funcional e intuitiva, permitindo uma interação eficiente com o sistema.

* + Documentação e Verificação:

A documentação detalhada do processo de desenvolvimento e a realização de verificações e testes ajudaram a garantir a funcionalidade e a confiabilidade do protótipo, apesar das dificuldades encontradas.

O projeto de um rádio FM controlado por um microcontrolador PIC18F4520 foi uma experiência enriquecedora que combinou teoria e prática em engenharia eletrônica e sistemas embarcados. Apesar dos desafios, os objetivos principais foram alcançados, e o protótipo desenvolvido funcionou conforme o esperado, validando as escolhas de design e implementação. As lições aprendidas ao longo do projeto serão valiosas para futuros desenvolvimentos e aprimoramentos.

A continuidade deste trabalho pode incluir a otimização do design para minimizar interferências eletrostáticas, a implementação de funcionalidades adicionais e a exploração de outras interfaces de comunicação para expandir as capacidades do sistema.

Referências

HITACHI. HD44780U: Dot Matrix Liquid Crystal Display Controller/Driver. 1999. Disponível em: https://www.sparkfun.com/datasheets/LCD/HD44780.pdf. Acesso em: 17 jun. 2024.

MICROCHIP TECHNOLOGY INC. PIC18F4520 Data Sheet: High-Performance, Enhanced Flash, USB Microcontrollers with nanoWatt Technology. 2010. Disponível em: https://www.microchip.com/wwwproducts/en/PIC18F4520. Acesso em: 10 jun. 2024.

NXP SEMICONDUCTORS. TEA5767HN: Low-power FM stereo radio for handheld applications. 2009. Disponível em: https://www.nxp.com/docs/en/data-sheet/TEA5767.pdf. Acesso em: 17 jun. 2024.