Departamento de Engenharia Elétrica
Universidade Federal do Ceará *Campus* Sobral

(SBL0082) Microprocessadores

Prof. Me. Alan Marques da Rocha
Prof. Dr. Marcelo Marques Simões de Souza

8

Prática 08: Comunicação Serial Síncrona

Introdução

Nos sistemas embarcados modernos, a comunicação eficiente entre diferentes dispositivos é um requisito fundamental para aplicações que vão desde automação industrial até sistemas de monitoramento remoto e dispositivos Internet of Things (IoT). Um dos protocolos mais amplamente utilizados para essa troca de informações é a comunicação serial *Universal Asynchronous Receiver-Transmitter*, devido à sua simplicidade, baixo custo e facilidade de implementação em microcontroladores.

No contexto dos microcontroladores da família PIC18F, a UART é um recurso nativo que permite transmitir e receber dados assíncronos por meio dos pinos RC6 (`TX`) e RC7 (`RX`), com suporte para diferentes taxas de transmissão (*baud rates*). A comunicação ocorre de forma ponto a ponto, ou seja, entre dois dispositivos, em que um atua como transmissor (`TX`) e o outro como receptor (`RX`). É uma solução amplamente empregada para envio de dados de sensores, controle remoto, diagnósticos e integração com periféricos como módulos *Bluetooth*, Wi-Fi ou até mesmo computadores.

Além da comunicação serial, esta prática integra dois recursos fundamentais em sistemas embarcados: o conversor Analógico-Digital (ADC) e o display LCD. O ADC permite converter sinais analógicos contínuos – como a tensão de um potenciômetro em valores digitais manipuláveis pelo microcontrolador. Já o *display* LCD é utilizado para exibir informações relevantes ao usuário, sendo um meio eficaz de retroalimentação visual em sistemas interativos.

Dessa forma, esta prática visa consolidar os seguintes conceitos fundamentais:

- Leitura de sinais analógicos utilizando o ADC do PIC18F4520;
- Transmissão serial UART entre dois microcontroladores PIC;
- Manipulação e exibição de dados em tempo real utilizando display LCD 16×2;
- Integração entre *hardware* (circuito físico/simulado) e *software* (*firmware* em C).

Na implementação proposta, um microcontrolador atuará como transmissor, enviando os dados de tensão lida em um potenciômetro via UART. Um segundo microcontrolador, atuando como receptor, fará sua própria leitura de tensão e exibirá no LCD tanto o valor local quanto o valor recebido, permitindo a visualização simultânea de duas grandezas analógicas em um único *display*. Tal abordagem simula cenários reais de monitoramento distribuído, como sistemas mestre-escravo ou sensores remotos conectados a um concentrador de dados.

Objetivos

- Configurar comunicação UART entre dois microcontroladores PIC18F4520.
- Utilizar o conversor A/D para leitura de tensões analógicas.
- Realizar a transmissão de dados numéricos via UART.
- Interpretar e exibir dados recebidos em um *display* LCD 16x2.
- Simular o circuito no Proteus e realizar a montagem física posteriormente.

Lista de Material

- 2 kits da placa UFC PICLAB-4520;
- 1 gravador PICKit2 ou PICKit3 para programação do microcontrolador;
- 02 microcontroladores PIC18F4520;
- 01 display LCD 16×2 (modelo LM016L ou compatível);
- Fonte de alimentação de 5 V (ou via USB);
- Protoboard e *jumpers*;
- Três potenciômetros de 1kΩ ou similar;
- *Software* MPLAB-X com compilador XC8;
- *Software* Proteus para simulação do circuito;
- LEDs verde, amarelo e vermelho.

Procedimento Experimental

Etapa 1: Criação do projeto no MPLAB-X para `PIC_TX`, `PIC_RX` e configuração do dispositivo

1. Crie dois projetos independentes no MPLAB-X: um para o transmissor (`PIC_TX`) e outro para o receptor (`PIC_RX`).

2. Selecione o microcontrolador PIC18F4520 em ambos os projetos.

Etapa 2: Desenvolvimento do código-fonte do `PIC_TX`

1. Realize o *download* do arquivo `main_TX.c` disponível no link (*https* : *//abre.ai/oeYx*) e faça o *upload* na pasta *Source* do projeto do MPLAB `PIC_TX` criado anteriormente.

2. Realize a leitura e interpretação do código, verifique as rotinas e estrutura antes de realizar a compilação.

3. Após a interpretação do código, realize a compilação do mesmo.

Etapa 3: Desenvolvimento do código-fonte do `PIC_RX`

1. Realize o *download* do arquivo `main_RX.c` disponível no link (*https* : *//abre.ai/oeYK*) e faça o *upload* na pasta *Source* do projeto do MPLAB `PIC_TX` criado anteriormente.

2. Realize a leitura e interpretação do código, verifique as rotinas e estrutura antes de realizar a compilação.

3. Após a interpretação do código, realize a compilação do mesmo.

Etapa 4: Montagem do circuito no Proteus

1. Realize a montagem do circuito seguindo o modelo do esquemático representado na Figura 8.1.

2. Como segunda opção, realize o *download* do esquemático no repositório do Github (`P08_comunicacao_sinc_C.pdsprj`) disponível no link (*https* : *//abre.ai/oeYM*).

Etapa 5: Carregar o *firmware* `.hex` no `PIC_TX` e `PIC_RX` dentro do Proteus

- Compile os projetos no MPLAB-X e gere os arquivos `.hex`.
- Como segunda opção, realize o *download* dos arquivos `main_TX.hex` e `main_RX.hex` nos *links* disponibilizados anteriormente.
- Atribua os arquivos correspondentes a cada microcontrolador `PIC_TX` e `PIC_RX` dentro do Proteus.

Etapa 6: Execução da simulação

- Execute a simulação no Proteus.
- Varie os potenciômetros e observe a leitura no *display* LCD.

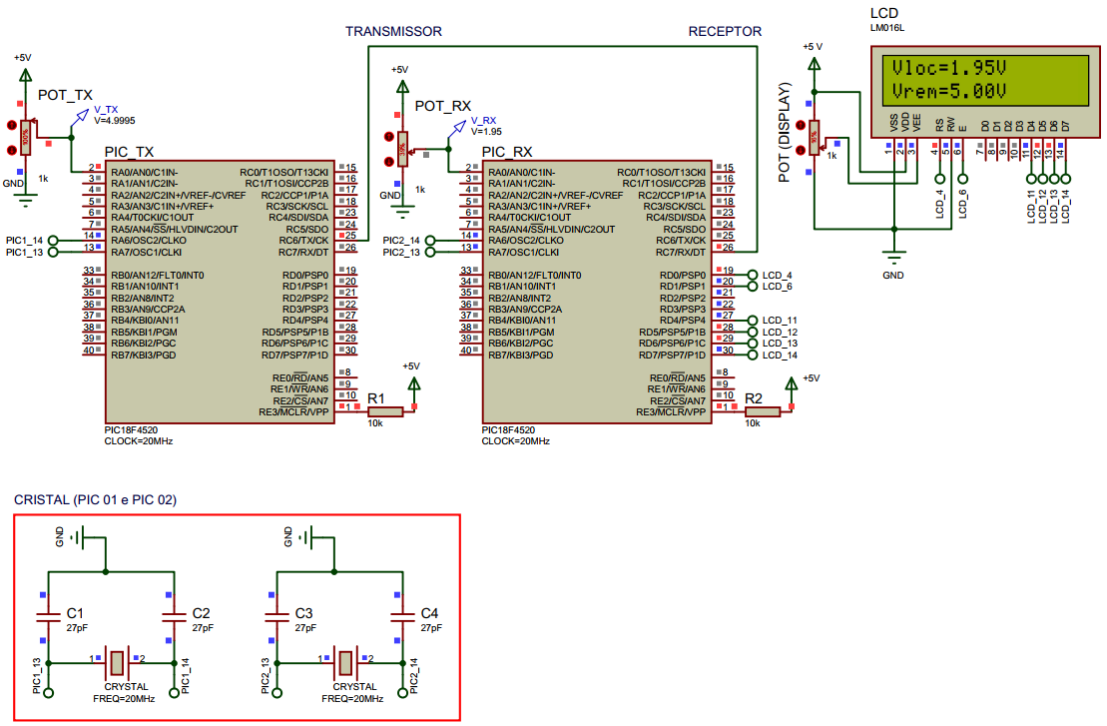

Figure 8.1: Circuito esquemático da prática 08.

- Verifique se os valores estão sendo atualizados corretamente em ambas as linhas.

Etapa 7: Montagem do circuito físico

- Reproduza o circuito da simulação em uma protoboard.
- Grave os microcontroladores com os arquivos `.hex` utilizando o PICKit2 ou PICKit3.
- Realize os testes práticos e compare com a simulação.

Questionário

1. Explique o funcionamento da comunicação UART entre os dois PICs.
2. Qual a importância do protocolo assíncrono nesta prática?
3. O que aconteceria se a taxa de transmissão entre os dois PICs fosse diferente?
4. O valor de tensão exibido no *display* corresponde exatamente ao valor do potenciômetro? Justifique.

5. Modifique o código para exibir os valores em milivolts (Encaminhar simulação+código).

6. Seria possível estender este projeto para transmitir outros tipos de sensores além de potenciômetros? Dê exemplos.

7. Que modificações seriam necessárias para exibir os dados de múltiplos transmissores em um único receptor?