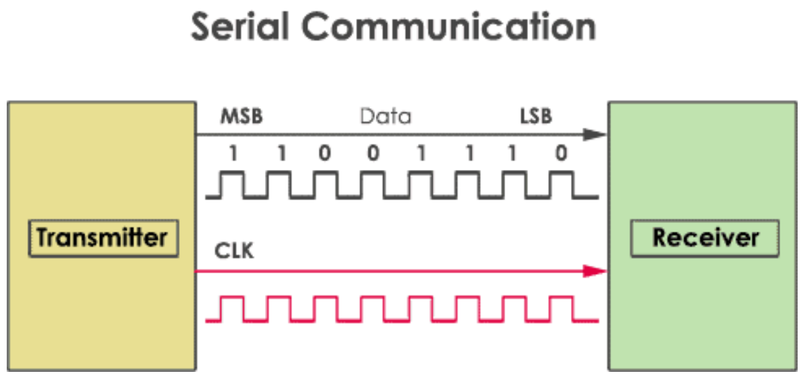
E209
Sistemas Microcontrolados e Microprocessados

Prof. João Magalhães

Comunicação Serial

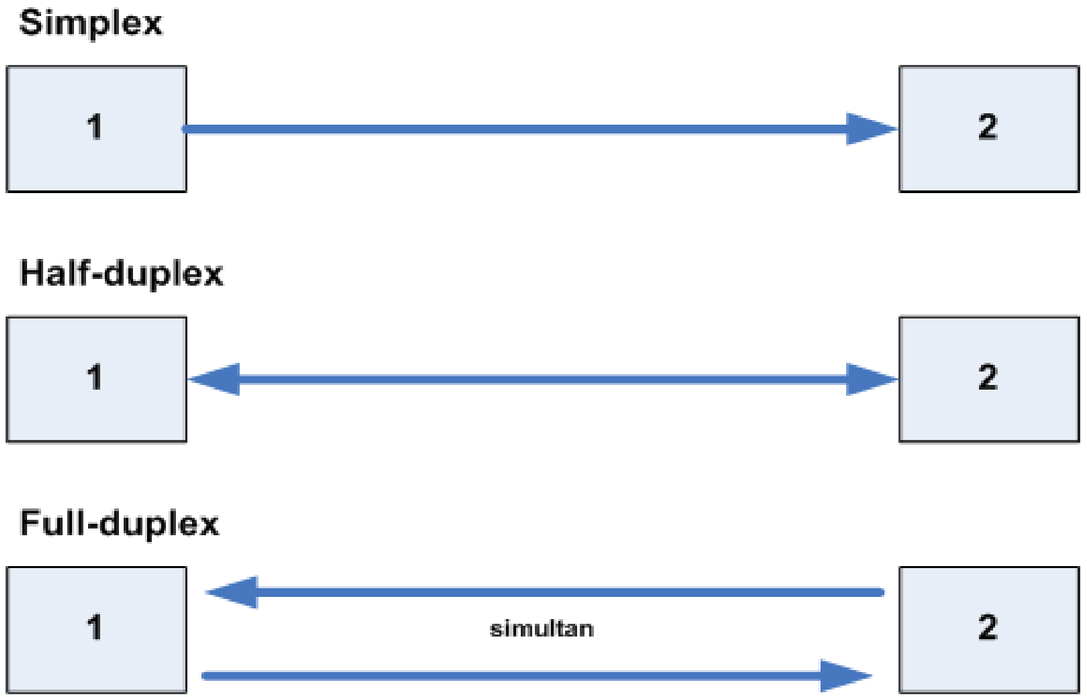
Introdução

A comunicação serial é uma das formas mais comuns de transferência de dados entre dispositivos eletrônicos.
É mais eficiente para dispositivos distantes entre si, pois demandam menos “fios” para fazer a comunicação.



Comunicação Serial

Modos de Transmissão



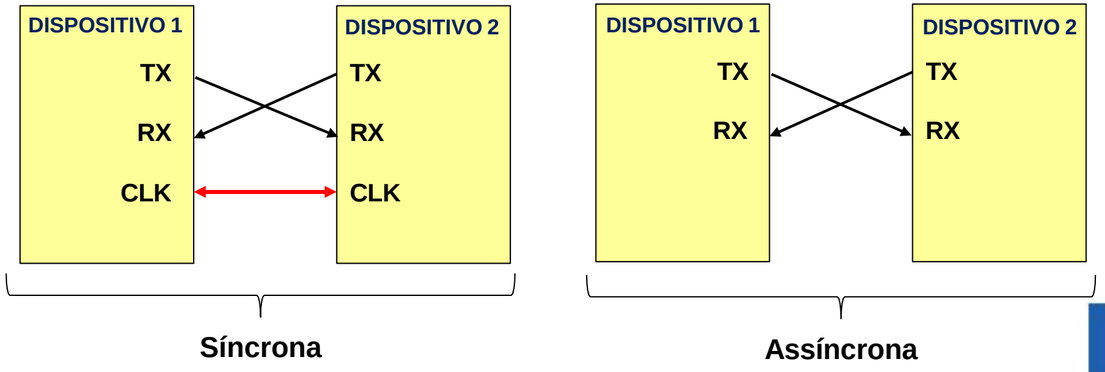
Comunicação Serial

Introdução

A comunicação serial pode ser assíncrona ou síncrona:

Assíncrona: não existe um sinal de *clock* para sincronizar a transmissão e a recepção;

Síncrona: precisa de um sinal de *clock* para sincronizar a transmissão e a recepção;



USART

Universal Synchronous Asynchronous Receiver Transceiver

É uma comunicação serial de dois protocolos.

Este protocolo é usado para transmitir e receber os dados bit a bit em relação aos pulsos de *clock* em um único fio.

Os microcontroladores AVR tem dois pinos: TX e RX, que são especialmente usados para transmitir e receber dados em série.

USART

Principais Características

- Operação Full Duplex (Registros Seriais de Recepção e Transmissão Independentes)
- Operação Assíncrona ou Síncrona com clock Mestre ou Escravo
- Gerador de Baud Rate de Alta Resolução
- Suporta Quadros Seriais com 5, 6, 7, 8 ou 9 bits de dados e 1 ou 2 bits de parada
- Geração de paridade ímpar ou par e verificação de paridade suportada pelo hardware
- Detecção de transbordamento de dados
- Detecção de erro de enquadramento

USART

Principais Características

- Filtragem de ruído, incluindo detecção de bit de início falso e filtro passa-baixa digital
- Três interrupções separadas em TX completo, Registro de dados TX vazio e RX completo
- Modo de comunicação multiprocessador
- Modo de comunicação assíncrona de velocidade dupla

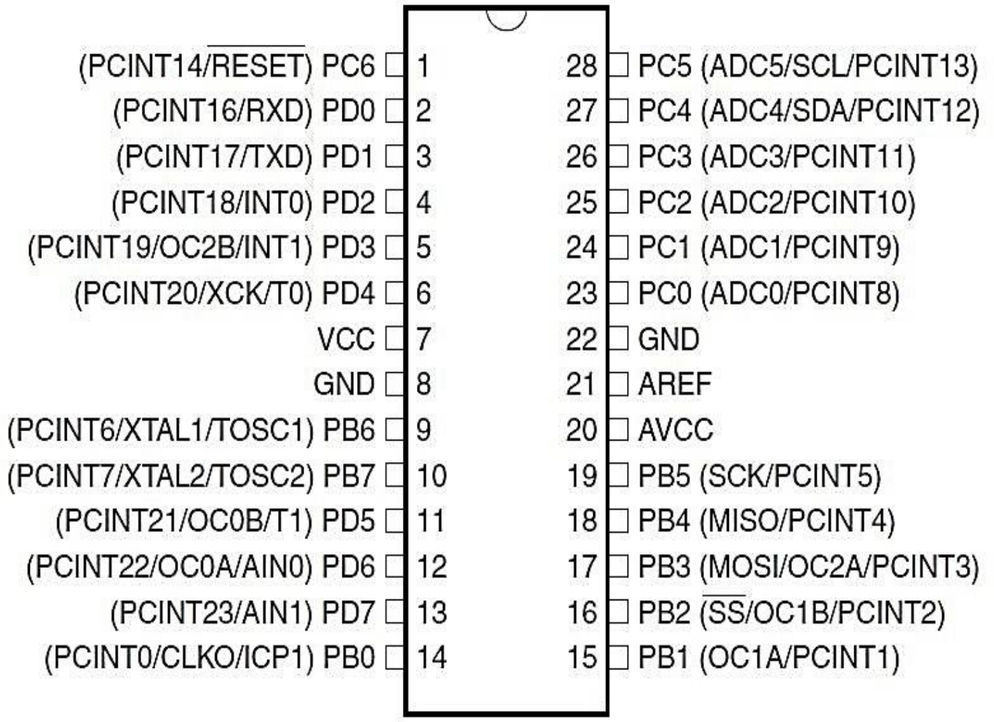
USART

Pinos no Microcontrolador

TXD – Pino do Transmissor (PD1)

RXD – Pino do Receptor (PD0)

XCK – Pino do Relógio (PD4)

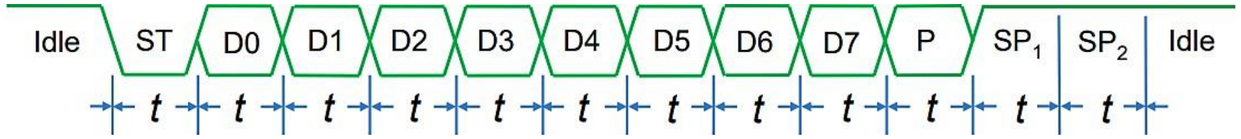


USART – Modo Assíncrono

Modo Assíncrono (UART)

Um frame UART consiste de:

- 1 start-bit;
- 7 ou 8 bits de dados;
- 0 ou 1 bit de paridade (pode ser paridade par ou paridade ímpar);
- 1 ou 2 stop-bits;
- Quando o canal está parado (sem transmissão) a linha fica em nível lógico 1;



UART

Baud Rate – Taxa de Transmissão

Define a quantidade de bits transmitidos por segundo (um frame UART tem entre 10 e 12 bits);

Alguns valores padrões de *baud rate* são: 300, 600, 1200, 2400, 4800, **9600, 14400, 19200, 28800, 38400, 56000 e 115200;**

UART

Registradores – UDR0

É o registrador de transmissão e recepção de dados.

Tem o papel de armazenar o conteúdo a ser transmitido ou recebido, dependendo da ação a ser realizada. Compartilham o mesmo endereço pois não há como transmitir e receber ao mesmo tempo.

Bit	7	6	5	4	3	2	1	0
	TXB / RXB[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 7:0 – TXB / RXB[7:0]: USART Transmit / Receive Data Buffer

UART

Registradores – UCSR0A

Bit	7	6	5	4	3	2	1	0
	RXC0	TXC0	UDRE0	FE0	DOR0	UPE0	U2X0	MPCM0
Access	R	R/W	R	R	R	R	R/W	R/W
Reset	0	0	1	0	0	0	0	0

Bit 7 – RXC0: USART Receive Complete

Bit 6 – TXC0: USART Transmit Complete

Bit 5 – UDRE0: USART Data Register Empty

Bit 4 – FE0: Frame Error

Bit 3 – DOR0: Data OverRun

Bit 2 – UPE0: USART Parity Error

Bit 1 – U2X0: Double the USART Transmission Speed

Bit 0 – MPCM0: Multi-processor Communication Mode

UART

Registradores – UCSR0B

Bit	7	6	5	4	3	2	1	0
	RXCIE0	TXCIE0	UDRIE0	RXEN0	TXEN0	UCSZ02	RXB80	TXB80
Access	R/W	R/W	R/W	R/W	R/W	R/W	R	R/W
Reset	0	0	0	0	0	0	0	0

Bit 7 – RXCIE0: RX Complete Interrupt Enable 0
Bit 6 – TXCIE0: TX Complete Interrupt Enable 0
Bit 5 – UDRIE0: USART Data Register Empty Interrupt Enable 0
Bit 4 – RXEN0: Receiver Enable 0
Bit 3 – TXEN0: Transmitter Enable 0
Bit 2 – UCSZ02: Character Size 0
Bit 1 – RXB80: Receive Data Bit 8 0
Bit 0 – TXB80: Transmit Data Bit 8 0

UART

Registradores – UCSR0C

Bit	7	6	5	4	3	2	1	0
	UMSEL01	UMSEL00	UPM01	UPM00	USBS0	UCSZ01 / UDORD0	UCSZ00 / UCPHA0	UCPOL0
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	1	1	0

UMSEL0[1:0]	Mode
00	Asynchronous USART
01	Synchronous USART
10	Reserved
11	Master SPI (MSPIM)(1)

UART

Registradores – UCSR0C

Bit	7	6	5	4	3	2	1	0
	UMSEL01	UMSEL00	UPM01	UPM00	USBS0	UCSZ01 / UDORD0	UCSZ00 / UCPHA0	UCPOL0
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	1	1	0

UPM0[1:0]	ParityMode
00	Disabled
01	Reserved
10	Enabled, Even Parity
11	Enabled, Odd Parity

UART

Registradores – UCSR0C

Bit	7	6	5	4	3	2	1	0
	UMSEL01	UMSEL00	UPM01	UPM00	USBS0	UCSZ01 / UDORD0	UCSZ00 / UCPHA0	UCPOL0
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	1	1	0

USBS0	Stop Bit(s)
0	1-bit
1	2-bit

UART

Registradores – UCSR0C

Bit	7	6	5	4	3	2	1	0
	UMSEL01	UMSEL00	UPM01	UPM00	USBS0	UCSZ01 / UDORD0	UCSZ00 / UCPHA0	UCPOL0
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	1	1	0

UCSZ0[2:0]	Character Size
000	5-bit
001	6-bit
010	7-bit
011	8-bit
100	Reserved
101	Reserved
110	Reserved
111	9-bit

UART

Registradores – UCSR0C

Bit	7	6	5	4	3	2	1	0
	UMSEL01	UMSEL00	UPM01	UPM00	USBS0	UCSZ01 / UDORD0	UCSZ00 / UCPHA0	UCPOL0
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	1	1	0

UCPOL0	Transmitted Data Changed (Output of TxD0 Pin)	Received Data Sampled (Input on RxD0 Pin)
0	Rising XCK0 Edge	Falling XCK0 Edge
1	Falling XCK0 Edge	Rising XCK0 Edge

UART

Como configurar?

- Etapa 1

A taxa de transmissão de USART / UART é definida pelo registrador **UBRR** *(Baud Rate Register)*

Este registro é usado para gerar a transmissão de dados em uma velocidade específica.

O UBRR é um registrador de 16 bits composto de “duas partes” de 8 bits: como UBRR (H), UBRR (L).

UART

Como configurar?

- Etapa 1 – Fórmula

$$UBRR = \frac{F_{osc}}{16(BAUD\ - 1)}$$

UART

Como configurar?

- Etapa 2

Seleção do modo de transmissão de dados.

- O modo de transmissão de dados, bit de início e bit de parada e o tamanho dos caracteres são definidos pelo registro de controle e status **UCSRC**
- **Para definir a comunicação serial como UART, é necessário colocar os dois primeiros bits do registrador em 0.**

UART

Como configurar?

- Etapa 3

Definição da Paridade

- Para definir a paridade, basta colocar os bits 4 e 5 do registrador UCSRC da seguinte forma.

UPM0[1:0]	ParityMode
00	Disabled
01	Reserved
10	Enabled, Even Parity
11	Enabled, Odd Parity

UART

Como configurar?

- Etapa 4

Definição dos bits de Stop

- Os microcontrolador AVR tem um bit de início e dois bits de parada.
- O bit de parada extra pode ser útil para adicionar um pouco mais de tempo de processamento de recepção, caso não seja necessário, deixe o bit 3 em 0.

UART

Como configurar?

- Etapa 5

Definição do tamanho da mensagem

- Nessa configuração, é necessário combinar os bits 1 e 2 do registrador UCSR0C e o bit 2 do registrador UCSR0B. Para definir um byte de transmissão, é necessário deixar os bits 1 e 2 do UCSR0C em 1 e o bit do UCSR0B em 0.

UART

Como configurar?

- Etapa 6

Definição do fluxo de comunicação

- Agora é necessário configurar quais os fluxos de comunicação serão utilizados, transmissão e recepção.
- Essas configurações são feitas no registrador UCSR0B, sendo o Bit 4 para recepção e o Bit 3 para transmissor.
- Caso seja necessário, pode-se habilitar o Bit 7 do registrador UCSR0B para configurar uma interrupção de finalização de recepção.

Comunicação Serial

Funções Utilizadas – Define e Variáveis

```
#define FOSC 16000000U
#define BAUD 9600
#define MYUBBR FOSC/16/BAUD - 1
#define TAMANHO 3


char msg_tx[20]
char msg_rx[32]
int pos_msg_rx = 0;
int tamanho_msg_rx = 3;
```

Comunicação Serial

Funções Utilizadas – Função de Configuração

```
void UART_config(unsigned int ubrr){

  UBBR0H = (unsigned char)(ubrr >> 8);
  UBBR0L = (unsigned char)(ubrr);

  UCSR0C = 0b00000110; //BIT2 + BIT1
  UCSR0B = 0b10011000; //BIT7 + BIT4 + BIT3

}
```

Comunicação Serial

Funções Utilizadas – Função de Transmissão

```
void UART_Transmit(char *dados){

  while(*dados != 0){

    while((UCSR0A & BIT5) == 0);

    UDR0 = *dados;

    dados++;

  }
```

```
UART_Transmit("Hello world! \n");

int x = NÚMERO;

itoa(x, msg_tx, 10);
UART_Transmit(msg_tx);
UART_Transmit("\n");
```

Comunicação Serial

Funções Utilizadas – Interrupção de Recepção

```
ISR(USART_RX_vect){

  msg_rx[pos_msg_rx++] = UDR0;

  if(pos_msg_rx == tamanho_msg_rx){

    pos_msg_rx = 0;

  }

}
```

```
if ((msg_rx[0] == 'o') &&
   (msg_rx[1] == 'l') &&
   (msg_rx[2] == 'a'))
{
```

```
valor = (msg_rx[0] - 48) * 100 +
      (msg_rx[1] - 48) * 10 +
      (msg_rx[2] - 48) * 1;
```

Prof. João Magalhães

Horário de Atendimento:

- Segunda-feira: 17h30
- Quinta-feira: 19h30

E-mail: joao.magalhaes@inatel.br

Celular: (35) 99895-4450

Linkedin: https://www.linkedin.com/in/joaomagalhaespaiva/