

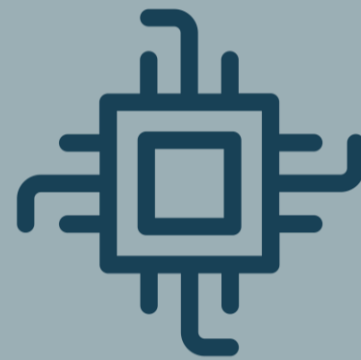


Universidade Federal
de Campina Grande



USART

ATMEGA328P



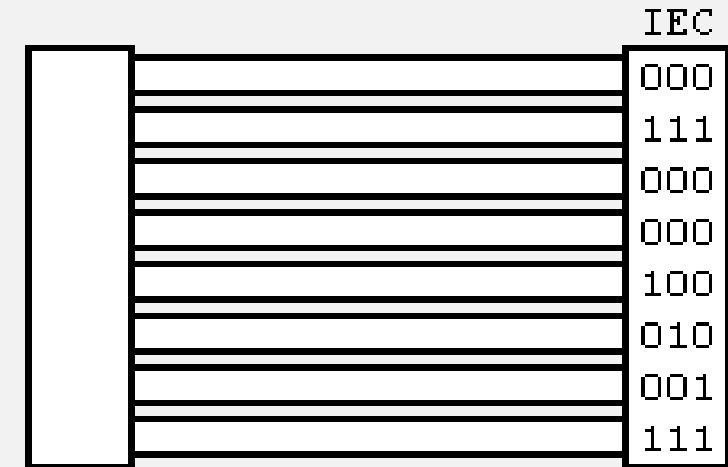
Prof.

**Rafael
Lima**

COMUNICAÇÃO: SERIAL X PARALELO

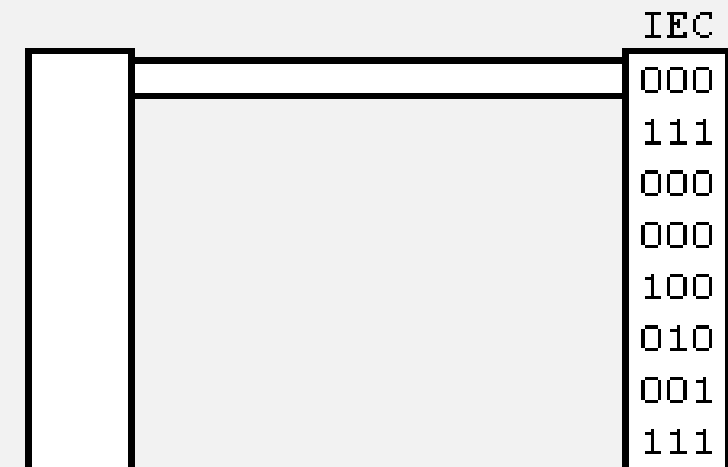
PARALELO

- *Maior Velocidade* ✓
- *Maior Custo* ✓
- *Mais susceptível a ruídos* ✓
- *Curtas distâncias* ✓



SERIAL

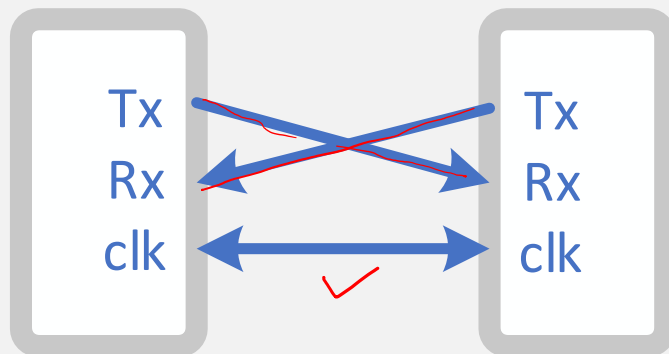
- *Menor Velocidade* ✓
- *Menor Custo* ✓
- *Menos susceptível a ruídos* ✓
- *Longas distâncias* ✓



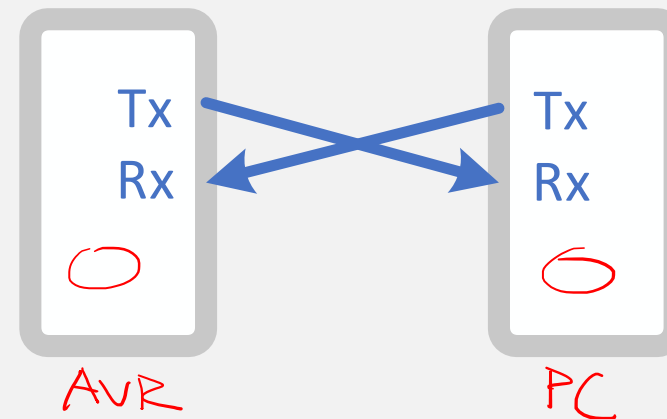
USART

USART (Universal **Synchronous** and **Asynchronous** Receiver and Transmitter)

Síncrono



Assíncrono

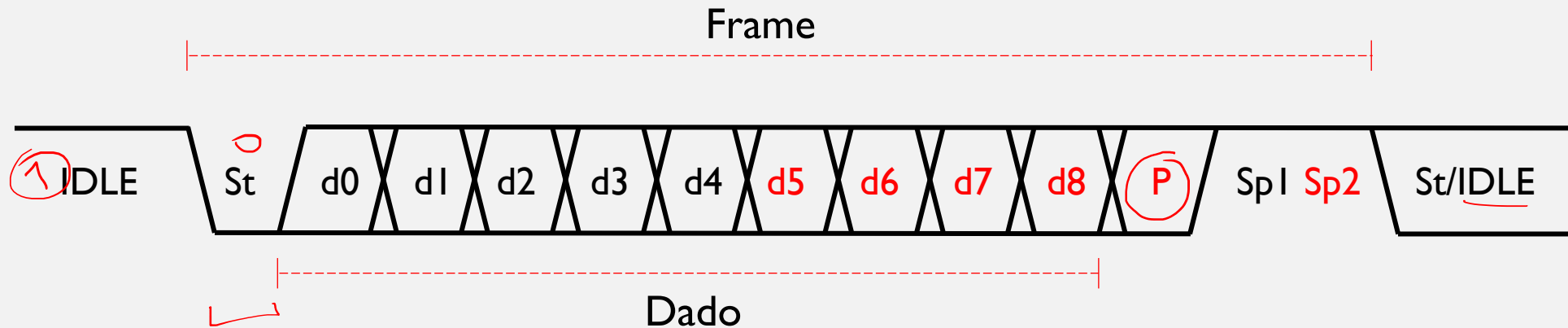


Full DUPLEX: Capaz de transmitir e receber simultaneamente

FRAME USART

Formato dos Frames

■ : Opcional



St: Start Bit (Sempre 0)

P: Bit de paridade (Par ou Impar)

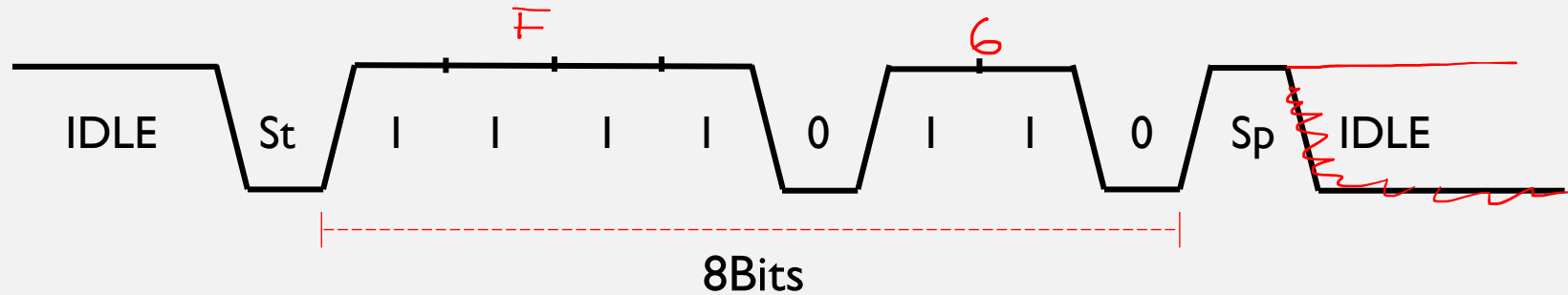
Sp: Stop Bit (Sempre 1)

IDLE: Estado de espera (Sempre 1)

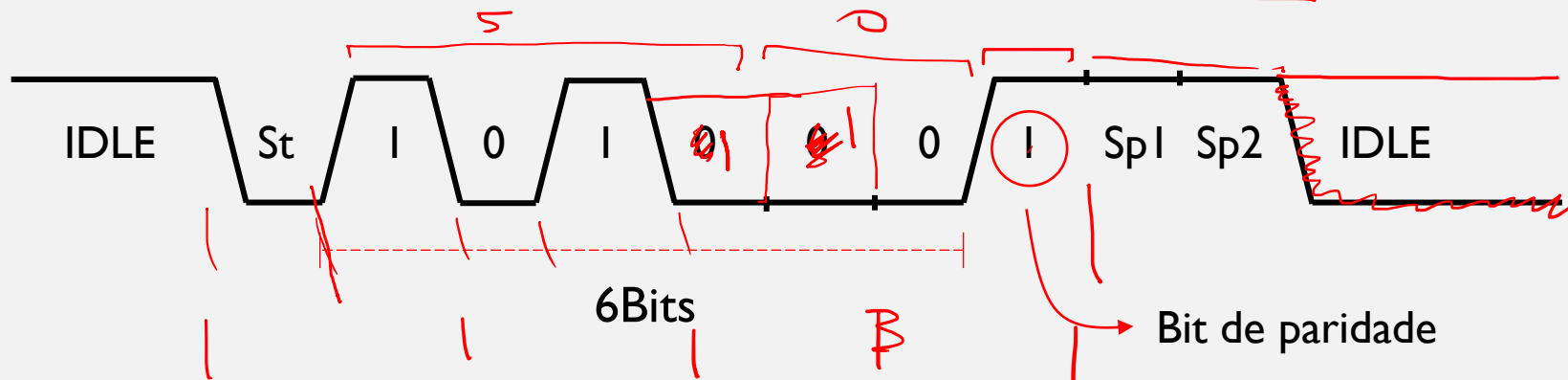


FRAME USART

Exemplo A: Transmitir 0x6F, 8bits, sem paridade, 1 Stop bit.



Exemplo B: Transmitir 0x05, 6bits, paridade ímpar, 2 Stop bits.



USART NO ATMEGA328P

- ✓ Operação Full Duplex (registradores independentes de recepção e transmissão).
- ✓ Operação síncrona ou assíncrona.
- ✓ Operação síncrona com clock mestre ou escravo.
- ✓ Gerador de taxa de comunicação de alta resolução (*Baud Rate Generator*)
- ✓ Suporta frames seriais com 5, 6, 7, 8 ou 9 bits de dados e 1 ou 2 bits de parada.
- ✓ Gerador de paridade par ou ímpar e conferência de paridade por hardware.
- ✓ Detecção de colisão de dados e erros de frames.
- ✓ Filtro para ruído, incluindo bit de início falso e filtro digital passa baixa.
- ✓ Três fontes separadas de interrupção (transmissão completa, recepção completa e esvaziamento do registrador de dados).
- ✓ Modo de comunicação assíncrono com velocidade duplicável.
- ✓ Pode ser utilizada como interface SPI mestre.

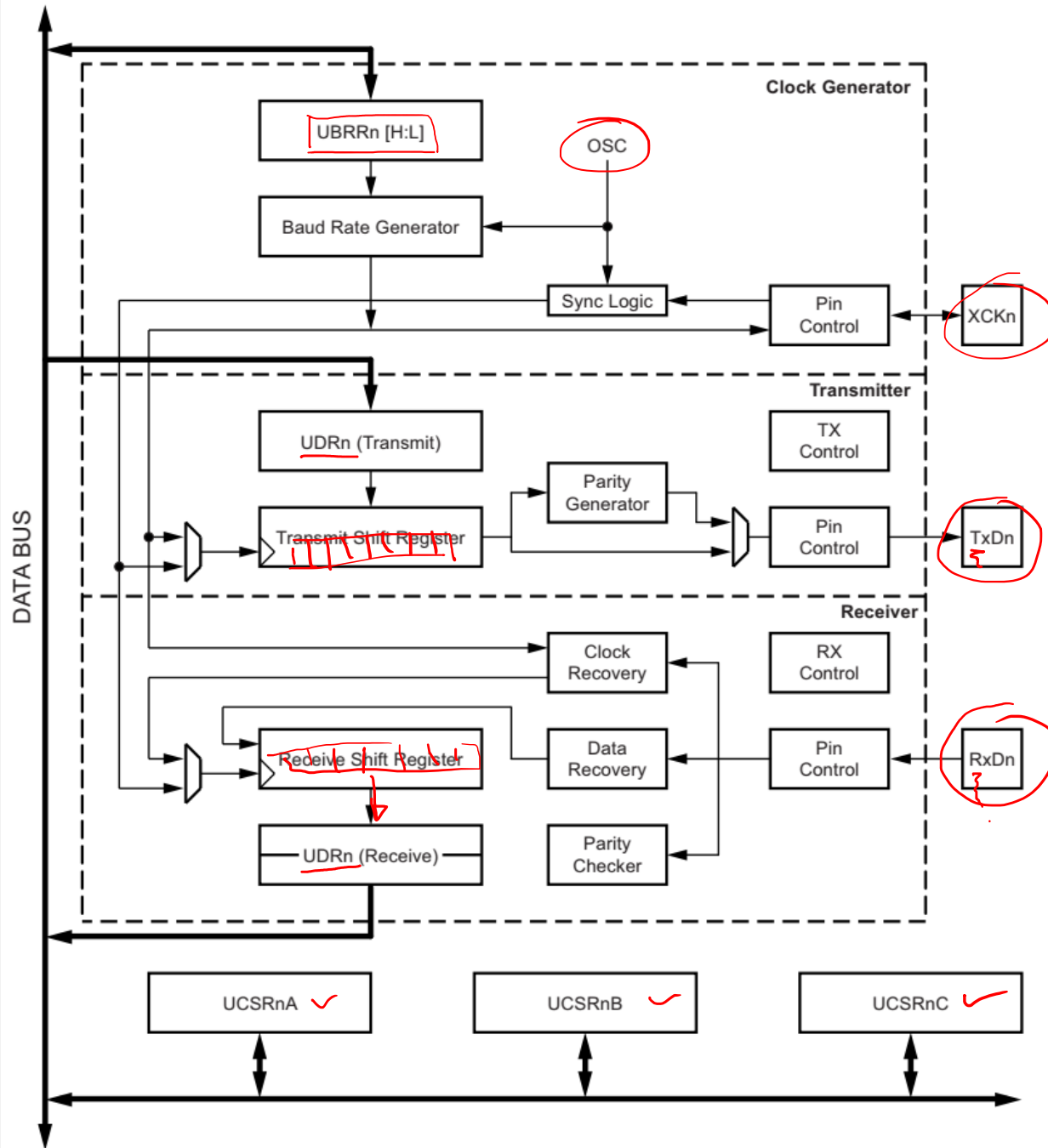
USART NO ATMEGA328P

Baud Rate

- A taxa de transmissão deve ser a mesma no transmissor e no receptor

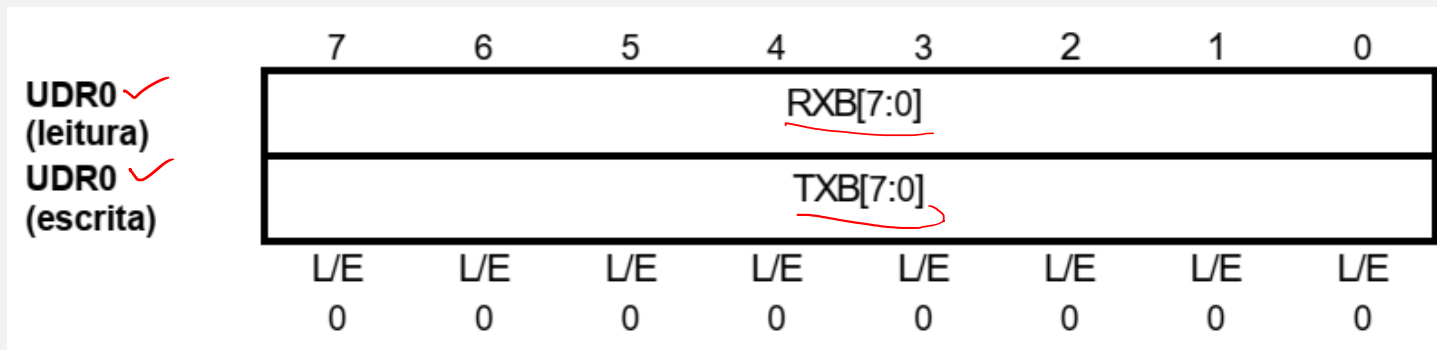
Modo de operação	Equação para o cálculo da taxa de transmissão	Equação para o cálculo do valor de UBRR0
Modo Normal Assíncrono (U2X0 = 0)	$TAXA = \frac{f_{osc}}{16(\underline{UBRR0} + 1)}$	$\underline{UBRR0} = \frac{f_{osc}}{16.TAXA} - 1$
Modo de Velocidade Dupla Assíncrono (U2X0 = 1)	$TAXA = \frac{f_{osc}}{8(UBRR0 + 1)}$	$UBRR0 = \frac{f_{osc}}{8.TAXA} - 1$
Modo Mestre Síncrono	$TAXA = \frac{f_{osc}}{2(UBRR0 + 1)}$	$UBRR0 = \frac{f_{osc}}{2.TAXA} - 1$

DIAGRAMA DE BLOCOS DA USART



REGISTRADORES DA USART

UDR0 (USART I/O Data Register)



Os registradores de **envio** e **recepção** compartilham o mesmo endereço. O acesso deles dependerá do contexto, se leitura ou escrita.

REGISTRADORES DA USART

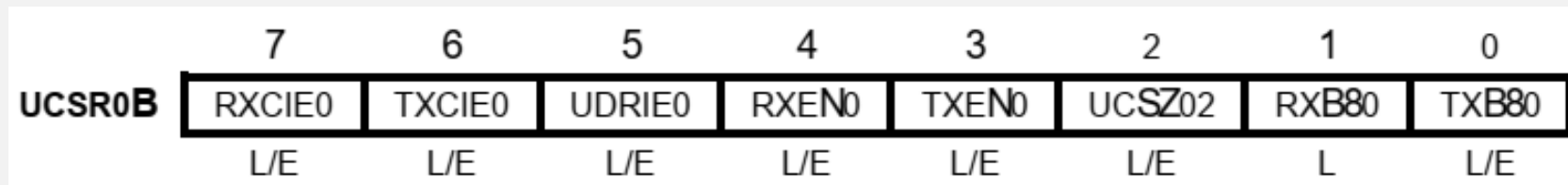
UCSR0A (USART Control and Status Register A)

	7	6	5	4	3	2	1	0
UCSR0A	RXC0	TXC0	UDRE0	FE0	DOR0	UPE0	U2X0	MPCM0
	L	L/E	L	L	L	L	L/E	L/E

- 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 ✓

REGISTRADORES DA USART

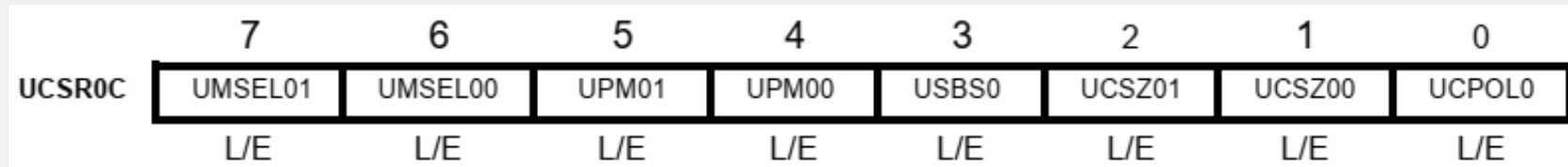
UCSR0B (USART Control and Status Register B)



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

REGISTRADORES DA USART

UCSR0C (USART Control and Status Register C)



- Bit 7:6 – **UMSEL01:0** – USART Mode Select
- Bits 5:4 – **UPM01:0** – Parity Mode
- Bit 3 – **USBS0** – Stop Bit Select

UMSEL01	UMSEL00	Modo de operação
0	0	assíncrono ✓
0	1	síncrono ✓
1	0	reservado
1	1	SPI mestre ✓

USBS0	Stop Bit
0	1 Stop bit ✓
1	2 Stop bit ✓

UPM01	UPM00	Modo de Paridade
0	0	Desabilitado ✓
0	1	Reservado
1	0	Habilitado, paridade par ✓
1	1	Habilitado, paridade ímpar ✓

REGISTRADORES DA USART

- Bits 2:1 – **UCSZ01:0** – Character Size

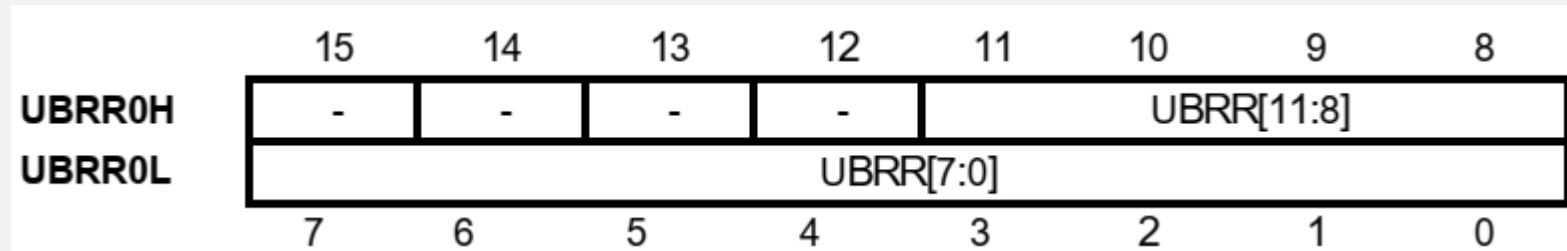
UCSZ02	UCSZ01	UCSZ00	Tamanho do Caractere
0	0	0	5 bits
0	0	1	6 bits
0	1	0	7 bits
0	1	1	8 bits
1	0	0	reservado
1	0	1	reservado
1	1	0	reservado
1	1	1	9 bits

- Bit 0 – **UCPOL0** – Clock Polarity

UCPOL0	Mudança do Dado Transmitido (saída do pino TxD0)	Amostragem do Dado Recebido (entrada do pino RxD0)
0	borda de subida de XCK	borda de descida de XCK
1	borda de descida de XCK	borda de subida de XCK

REGISTRADORES DA USART

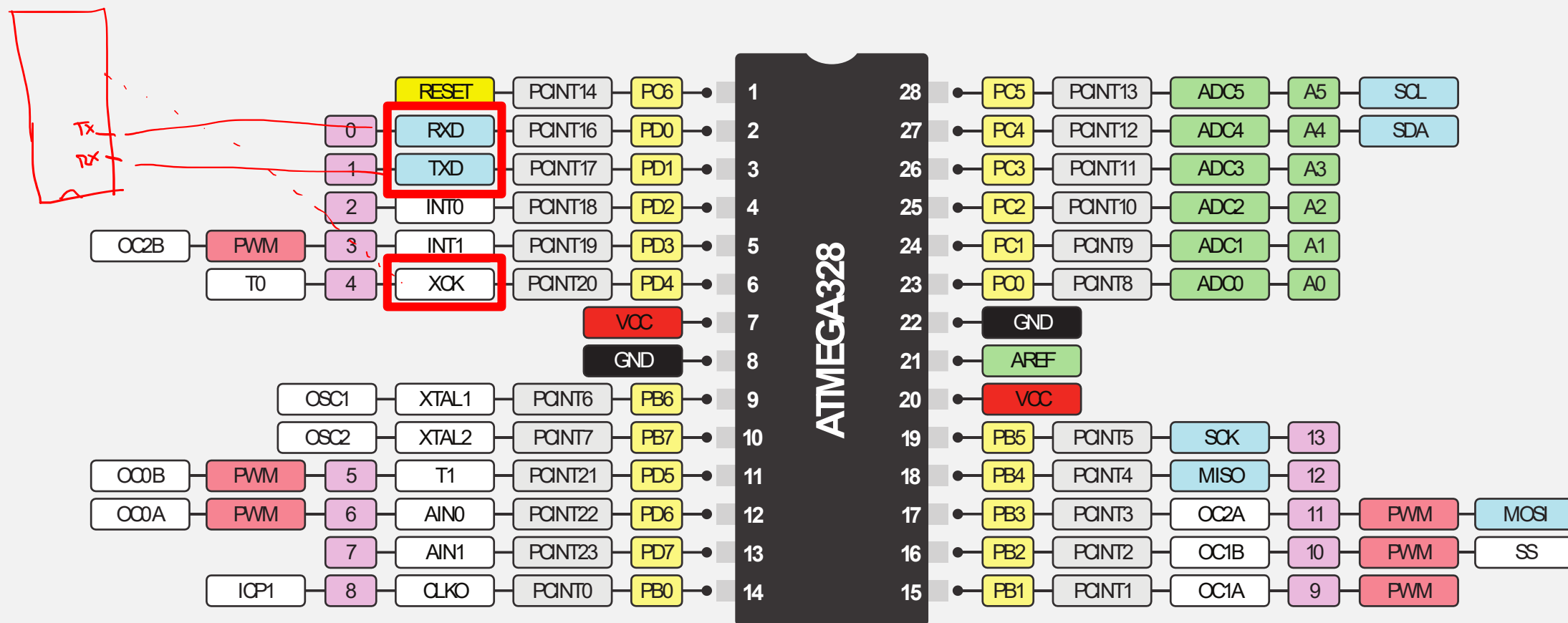
UBRR0L e UBRR0H (USART Baud Rate Register)



9600 bps

MAPA DE PINOS DO ATMEGA328P

- Pinos da USART



```
#define F_CPU 16000000UL //Frequência de trabalho da CPU
#define BAUD 9600
#define MYUBRR F_CPU/16/BAUD-1
#include <avr/io.h>
#include <util/delay.h>
```

```
// ||Função para inicialização da USART||
```

```
void USART_Init(unsigned int ubrr)
```

```
{
```

```
    UBRR0H = (unsigned char)(ubrr>8); //Ajusta a taxa de transmissão
```

```
    UBRR0L = (unsigned char)ubrr;
```

```
    UCSRB = (1<<RXEN0)|(1<<TXEN0); //Habilita o transmissor e o receptor
```

```
    UCSRC = (1<<USBS0)|(3<<UCSZ0); //Ajusta o formato do frame: 8 bits de dados e 2 de parada
```

```
}
```

```
// ||Função para envio de um frame de 5 a 8bits||
```

```
void USART_Transmit(unsigned char data)
```

```
{
```

```
    while(!(UCSR0A & (1<<UDRE0))); //Espera a limpeza do registr. de transmissão
```

```
    UDR0 = data; //Coloca o dado no registrador e o envia
```

```
}
```

```
// ||Função para recepção de um frame de 5 a 8bits||
```

```
unsigned char USART_Receive(void)
```

```
{
```

```
    while(!(UCSR0A & (1<<RXC0))); //Espera o dado ser recebido
```

```
    return UDR0; //Lê o dado recebido e retorna
```

```
}
```

```
void main(void)
```

```
{
```

```
    USART_Init(MYUBRR);
```

```
    while(1)
```

```
    {
```

```
        USART_Transmit('u');
```

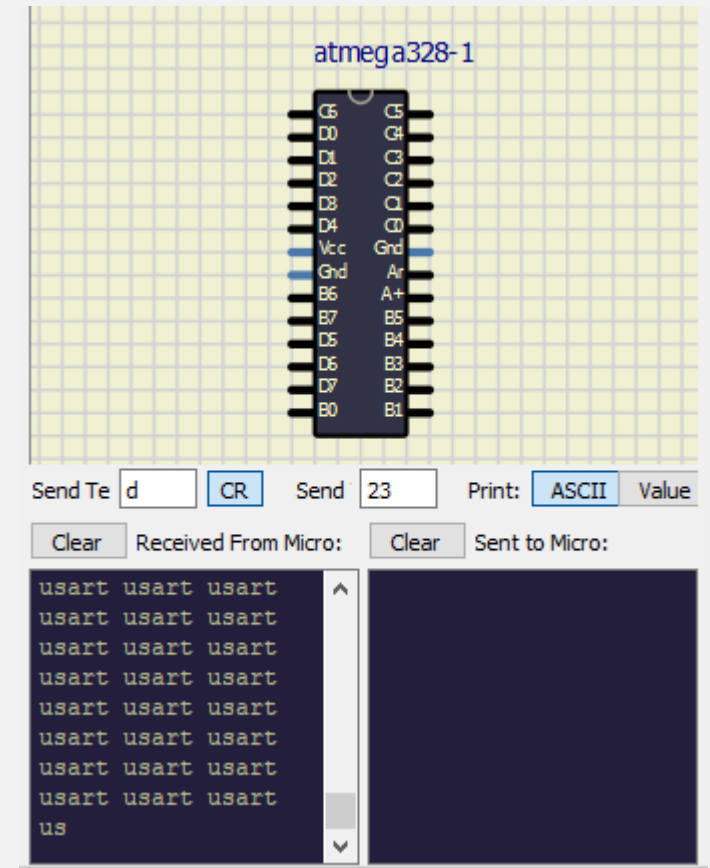
```
        _delay_ms(200);
```

```
        ...
```

```
    }
```

```
}
```

EXEMPLO: USART POLLING



EXEMPLO: USART INTERRUPÇÃO

```
#include <avr/interrupt.h>
```

```
// ||Função de tratamento de interrupção - Recepção USART||
```

```
ISR(USART_RX_vect)
```

```
{
```

```
    char recebido;
```

```
    recebido = UDR0; //UDR0 contém o dado recebido via USART
```

```
    USART_Transmit(recebido);
```

```
}
```

```
UCSR0B |= (1<<RXCIF0); //Habilita a interrupção de recepção da USART
```

```
sei(); //Habilita as interrupções globais
```

REFERÊNCIAS

IDE

- Atmel Studio 7 (gratuito) <https://www.microchip.com/mplab/avr-support/atmel-studio-7>

Simuladores

- <https://www.simulide.com/p/blog-page.html>
- <https://github.com/lcgamboa/picsimlab/releases>
- <https://www.labcenter.com/downloads/>

Material de referência:

- Datasheet do Atmega 328p: <https://www.microchip.com/wwwproducts/en/ATmega328p#datasheet-toggle>
- Livro texto: <http://borgescorporation.blogspot.com/2012/05/avr-e-arduino-tecnicas-de-projeto.html>