Experimento 5 – Comunicação Serial

Fundamentação

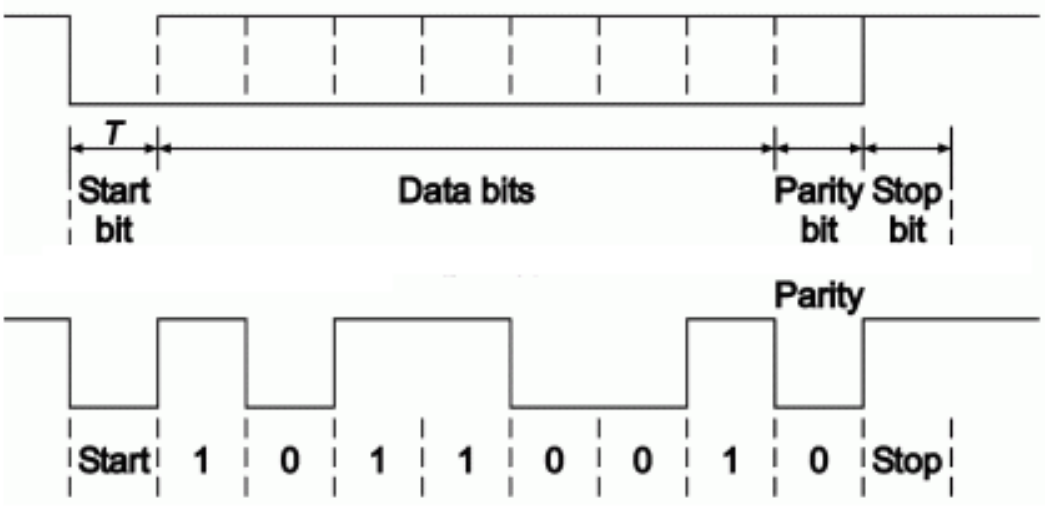
Em muitos projetos envolvendo microcontroladores é necessário estabelecer algum tipo de comunicação com outros sistemas eletrônicos. É comum por exemplo realizar a comunicação entre dois microcontroladores, entre um microcontrolador e uma interface gráfica, ou ainda entre o microcontrolador e um computador pessoal. Projetos mais complexos podem exigir ainda que o microcontrolador estabeleça conexões com a internet por exemplo. Para este tipo de conexão existem algumas formas de comunicação pré-estabelecidas, como por exemplo as comunicações seriais síncronas SPI e I2C, as comunicações seriais assíncronas como a RS232 e RS485, e a comunicação USB. Os microcontroladores possuem circuitos dedicados para realizar alguns destes tipos de comunicação. Neste experimento trataremos exclusivamente da comunicação serial assíncrona.

Comunicação serial assíncrona

A comunicação serial é o processo de enviar e receber dados um bit de cada vez, sequencialmente, em um canal de comunicação ou barramento. A comunicação serial é usada em toda comunicação de longo alcance e na maioria das redes de computadores e redes industriais.

A comunicação serial é dita síncrona quando transmite junto aos dados um sinal de *clock* e dita assíncrona quando não transmite este sinal.

A figura a seguir mostra a forma de onda de um sinal serial assíncrono.



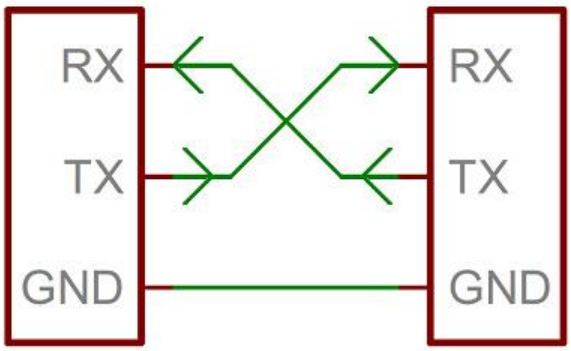
No caso da comunicação serial assíncrona, a temporização da transmissão dos dados é definida pela velocidade de transmissão em bits por segundo (bps). Existem algumas velocidades padronizadas que são utilizadas pelos equipamentos eletrônicos. As principais velocidades são apresentadas na tabela a seguir.

300 bps	600 bps	1200 bps
2400 bps	4800 bps	9600 bps
14400 bps	19200 bps	28800 bps
38400 bps	56000 bps	57600 bps
115200 bps	128000 bps	256000 bps

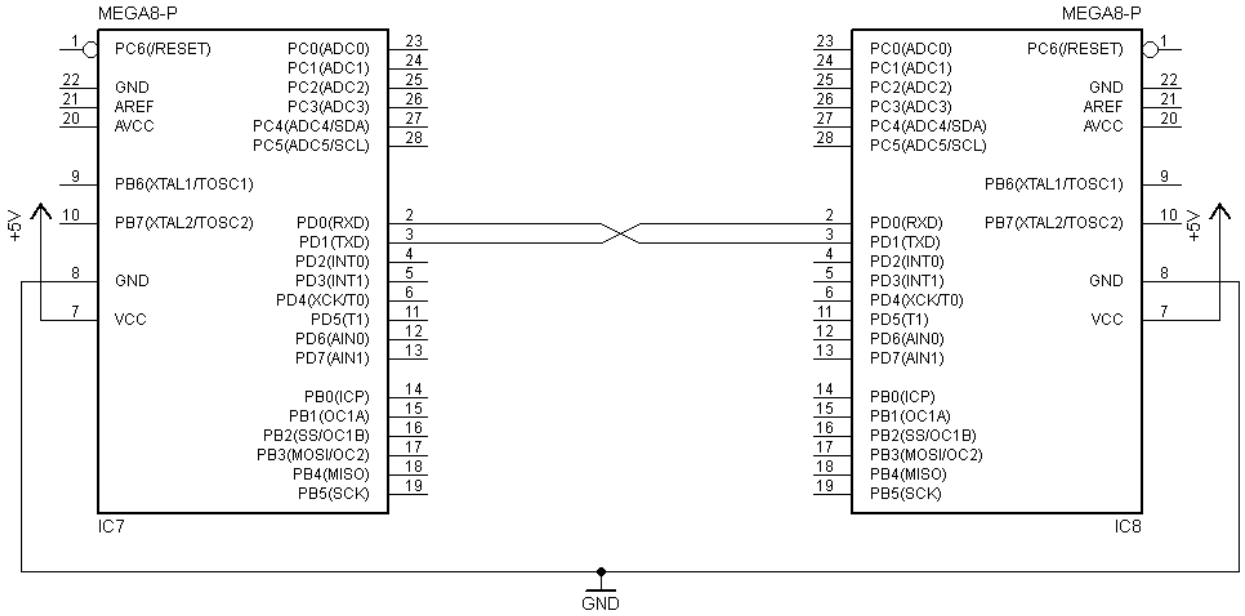
Também existem padrões no que diz respeito ao número de bits de dados e de parada que são transmitidos em cada pacote. É possível ainda configurar um bit extra de verificação de erros, chamado de bit de paridade. A tabela a seguir mostra os valores comuns para cada um destes parâmetros.

Parâmetro	**Valores Comuns**
Número de bits por pacote	5, 6, 7 e 8.
Bits de parada	1, 1,5 e 2.
Bit de paridade	Par, Impar, 0 e 1.

Para estabelecer uma comunicação serial é necessário então que os dois dispositivos estejam no mesmo potencial, ou seja, o negativo da alimentação dos dois dispositivos deve estar conectado. É necessário também que a saída de dados do transmissor seja conectada a entrada de dados do receptor. A figura a seguir mostra a conexão elétrica entre dois dispositivos para estabelecer uma comunicação de duas vias.



Neste caso, TX é a porta de transmissão e RX a porta de recepção. A figura a seguir mostra como interconectar dois microcontroladores para estabelecer uma comunicação serial.

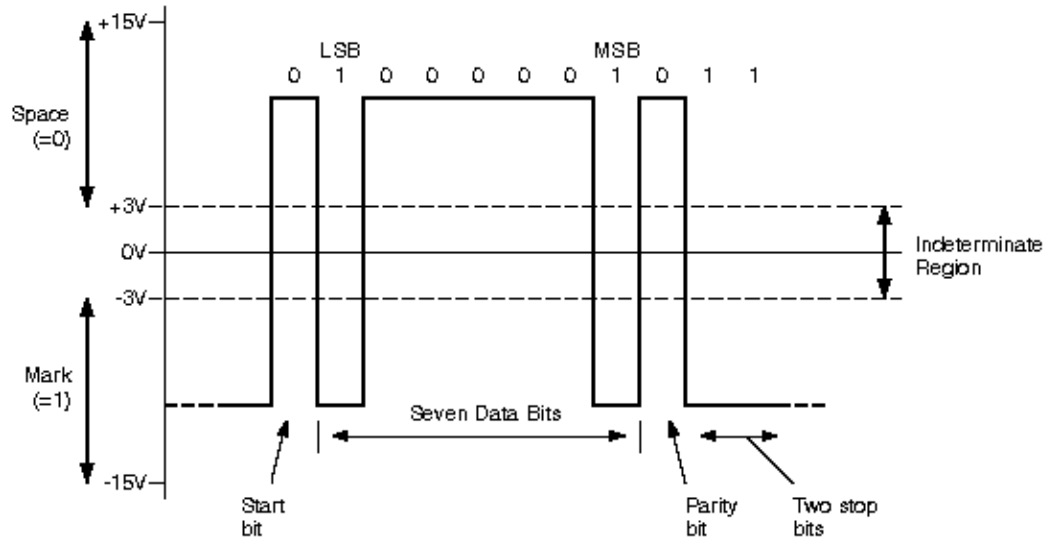


Apesar de ser possível estabelecer uma comunicação serial em curtas distâncias utilizado os sinais diretamente dos pinos do microcontrolador, este tipo de conexão é inapropriado para comunicações em distâncias maiores. Para estes casos, utiliza-se padrões de sinais elétricos diferentes para transmitir os dados. A seguir serão apresentados os padrões RS232 e RS485.

O padrão RS232

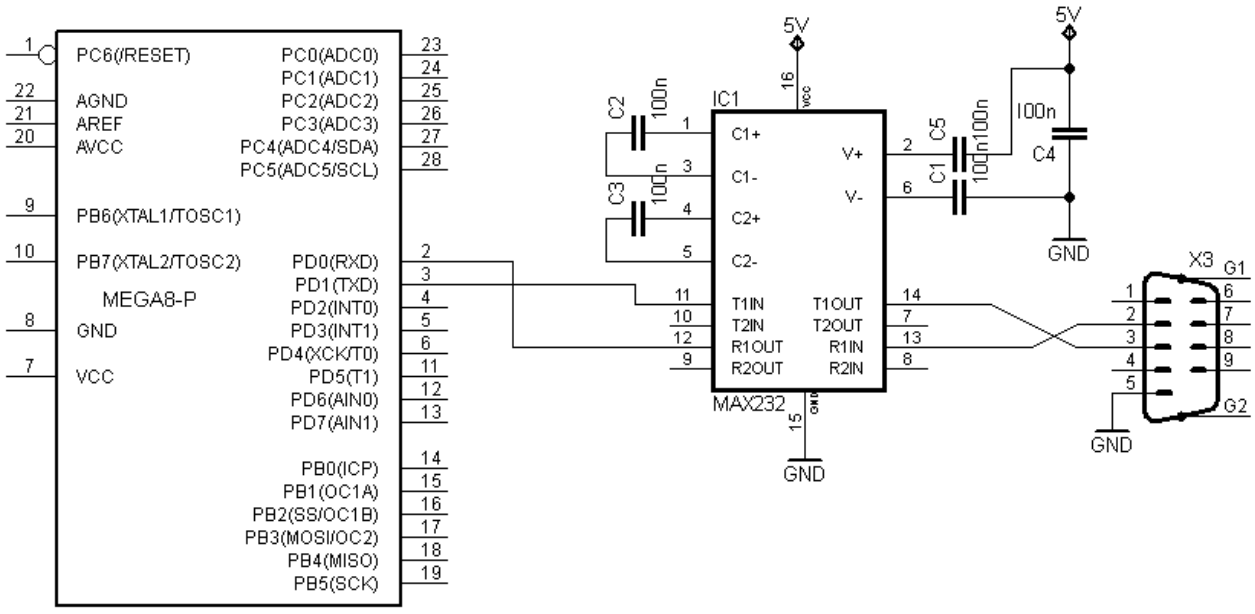
O padrão RS232 é um padrão utilizado principalmente nos computadores pessoais mais antigos, e assim é amplamente difundido em uma grande variedade de equipamentos. Este padrão foi desenvolvido para comunicação entre 2 dispositivos, não permitindo um número maior de equipamentos.

Neste padrão os sinais elétricos definidos como níveis de tensão são os seguintes. O nível lógico 0 é representado por uma tensão positiva entre 3 e 15 V, já o nível lógico 1 é representado por uma tensão negativa entre -3 e -15 V. A figura a seguir mostra esta situação.



Utilizando níveis de tensão mais elevados é possível alcançar distâncias maiores, na casa de alguns metros. Um fator importante é a blindagem do cabo, cabos blindados fornecem maior imunidade a ruído permitindo maior alcance.

Para que um microcontrolador possa se comunicar através do padrão RS232 é necessário que se adicione a ele um circuito conversor de sinais. Existem vários circuitos integrados que desenvolvem esta função, o mais comum é o MAX232. A figura a seguir mostra o circuito necessário para que o microcontrolador possa se comunicar no padrão RS232 utilizando o circuito integrado MAX232.

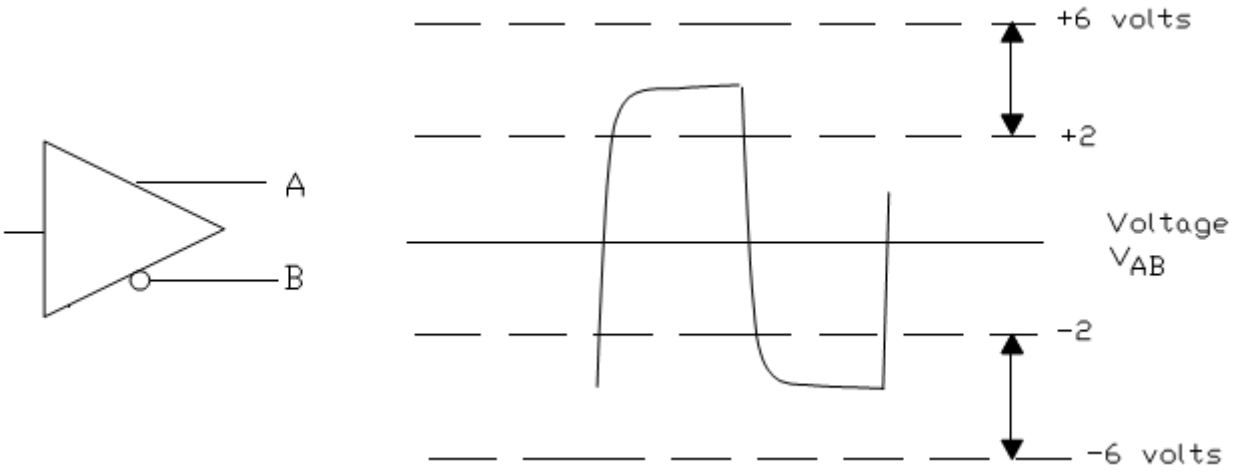


A comunicação serial permite ainda a utilização de sinais adicionais além dos sinais RX e TX, que permitem controlar o fluxo de informação de forma mais prática.

O padrão RS485

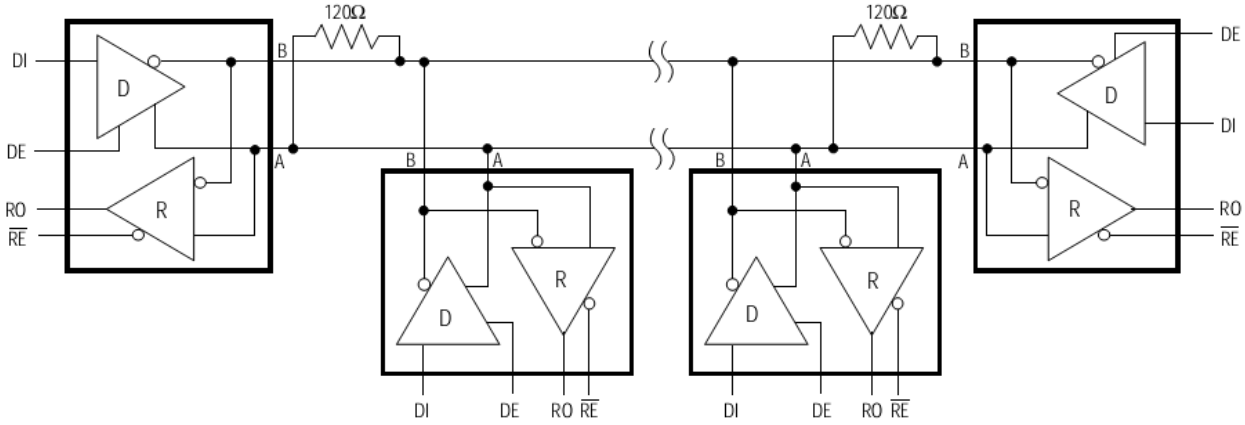
O padrão RS485 é um padrão muito utilizado nas redes industriais, diferente do padrão RS232 cada sinal é transmitido através de 2 fios e o que define o nível lógico do sinal é a diferença de tensão entre estes dois fios. Este arranjo é muito imune a ruídos, permitindo comunicações com distâncias superiores a 1 Km.

Eletricamente falando o sinal que transmite os dados na comunicação RS485 é um sinal diferencial. A figura a seguir mostra como este sinal diferencial funciona.



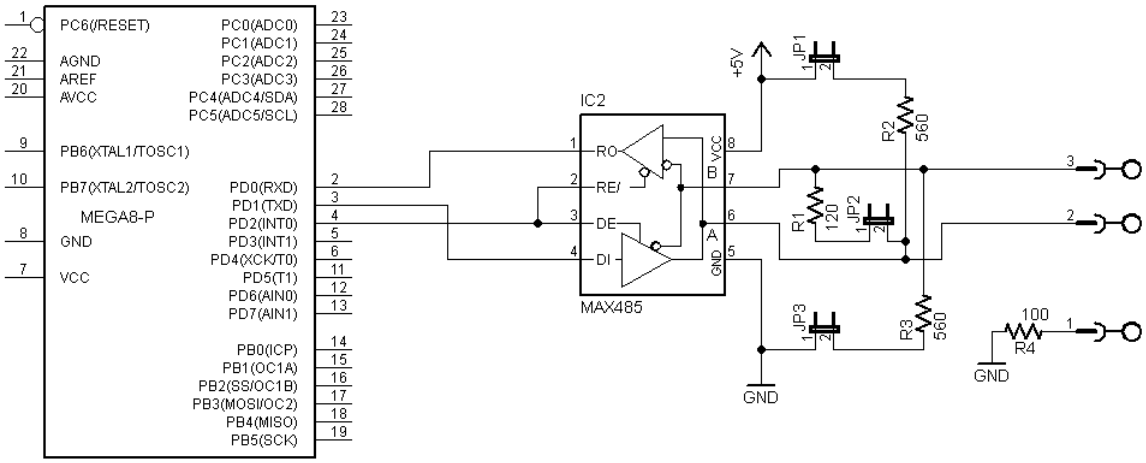
É importante salientar que os níveis de tensão indicados na figura dizem respeito a tensão entre os terminais A e B, e não com relação a referência.

O padrão de comunicação RS485 é projetado para interligar 2 ou mais dispositivos, permitindo assim redes com vários equipamentos. Porém, tanto a transmissão como a recepção são feitas no mesmo par de fios, o que não permite que um dispositivo envie e receba dados ao mesmo tempo. A figura a seguir mostra a ligação de alguns dispositivos no padrão RS485.



Observando a figura podemos notar a presença de 2 resistores de 120Ω, estes resistores são necessários nas extremidades da rede para casar a impedância dos equipamentos, tornando a mesma mais imune a ruídos e a erros de comunicação.

Para que um microcontrolador possa se comunicar através do padrão RS485 é necessário que se adicione a ele um circuito conversor de sinais. Existem vários circuitos integrados que desenvolvem esta função, o mais comum é o MAX485. A figura a seguir mostra o circuito necessário para que o microcontrolador possa se comunicar no padrão RS485 utilizando o circuito integrado MAX485.

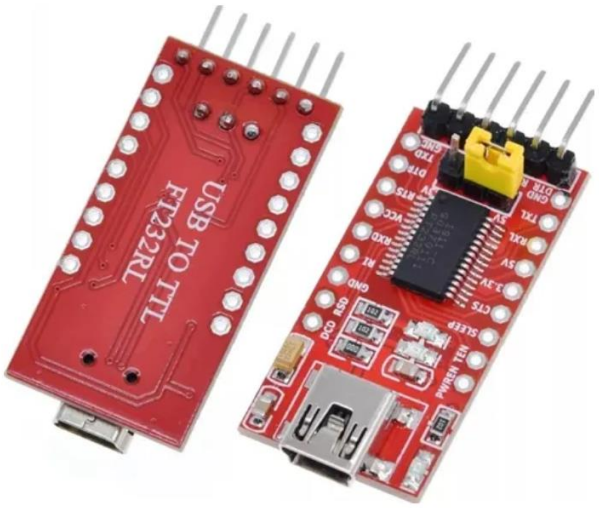


No circuito foram adicionados jumpers que permitem ligar ou desligar cada um dos resistores independentemente.

Estas figuras tratam apenas do meio físico da rede, é necessário que se estabeleça um protocolo de comunicação para que os dados façam sentido tanto para quem transmite como para quem recebe.

Comunicação serial através da USB

É bastante comum desejarmos estabelecer uma comunicação serial entre o microcontrolador e um computador pessoal, porém a maioria dos computadores pessoais não possui mais portas de comunicação serial. Nestes casos o que se costuma fazer é utilizar um conversor USB-Serial para estabelecer uma comunicação serial através da porta USB do computador. A figura a seguir mostra um conversor USB-Serial compatível com os microcontroladores.



Existem diversos modelos de conversores USB-Serial, alguns apropriados para conexão direta com o microcontrolador, outros compatíveis com os padrões RS232 ou RS485, por exemplo.

A comunicação serial nos microcontroladores AVR

Nos microcontroladores da família AVR a porta de comunicação serial é chamada de USART (Universal Synchronous and Asynchronous serial Receiver and Transmitter). Esta porta é um dispositivo de comunicação serial extremamente versátil. Suas principais características são:

• Transmissão e recepção simultânea de dados (Full Duplex).
• Operação assíncrona ou síncrona.
• Opera como mestre ou escravo em uma rede síncrona.
• Gerador de taxa de transferência de alta resolução.
• Suporta pacotes de dados de 5, 6, 7, 8 ou 9 bits com 1 ou 2 bits de parada.
• Gerador e verificador de paridade em hardware, com paridade par e ímpar.
• Detecta sobrescrita de dados.
• Detecta erro de pacote.
• Diversos filtros contra ruídos.
• Três fontes de interrupção, transmissão completa, registrador de transmissão vazio e recepção completa.
• Modo de comunicação multi-processador.
• Modo de comunicação “Double Speed” para comunicações assíncronas de alta velocidade.

Assim como nos outros periféricos do microcontrolador, a comunicação serial é controlada através de registradores. O registrador **UDR** é utilizado para armazenar os dados transmitidos e recebidos.

O primeiro registrador de controle é o **UCSRA**. A tabela a seguir detalha cada um de seus bits.

Registrador UCSRA	
Bit	**Significado**
0	MPCM: Modo multiprocessadores.
1	U2X: Dobra a velocidade da USART.
2	PE: Erro de paridade
3	DOR: Sobreposição de dados
4	FE: Erro no pacote
5	UDRE: Registrador temporário de dados vazio.
6	TXC: Transmissão completa.
7	RXC: Recepção completa.

O próximo registrador é o **UCSRB**.

Registrador UCSRB	
Bit	**Significado**
0	TXB8: Bit 8 da transmissoção.
1	RXB8: Bit 8 da recepção.
2	UCSZ2: Tamanho do pacote, bit 2.
3	TXEN: Habilita a transmissão.
4	RXEN: Habilita a recepção.
5	UDRIE: Habilita a interrupção de registrador temporário vazio.
6	TXCIE: Habilita a interrupção de transmissão.
7	RXCIE: Habilita a interrupção de recepção.

O último registrador de controle que estudaremos é o **UCSRC**.

Registrador UCSRC	
Bit	**Significado**
0	UCPOL: Polaridade do clock
1	UCSZ0: tamanho do pacote 0
2	UCSZ1: tamanho do pacote 1
3	USBS: Seleção de stop bit.
4	UPM0: Modo da paridade 0
5	UPM1: Modo da paridade 1
6	UMSEL: Seleção de modo de operação da USART
7	URSEL: Seleciona o acesso ao registrador.

Não irei detalhar cada um destes registradores aqui, o *datasheet* do microcontrolador fornece todas as informações necessárias sobre cada um deles.

Como vimos anteriormente, na comunicação serial assíncrona é necessário definir uma taxa de transferência em bits por segundo para que o transmissor e o receptor possam sincronizar os dados. Nos microcontroladores AVR a taxa de transferência é ajustada no registrador **UBRR**.

Exemplo de programa de comunicação serial:

O exemplo de programa a seguir configura a comunicação serial para uma taxa de transmissão de 1200 bps.

```
#include <avr/io.h>
#include <avr/interrupt.h>
#define F_CPU 1000000 // frequencia do microcontrolador
#include <util/delay.h>

#define USART_BAUDRATE 1200      // taxa de transmissão desejada
#define STOPBITS 2               // número de bits de parada 1 ou 2
#define BAUD_PRESCALE ((F_CPU/ (USART_BAUDRATE * (long)16))-1) //calcula
o valor do prescaler da usart

void usart_send(unsigned char data)
{
      while((UCSRA&0b00100000)==0)
      {
      }
      UDR = data;
}

void usart_init() // inicia a comunicação serial
{
      UCSRB |= (1 << RXEN) | (1 << TXEN); // Liga a transmissão e a re-
cepção
      #if STOPBITS == 2
      UCSRC |= (1 << URSEL) | (1 << UCSZ0) | (1 << UCSZ1) | (1 << USBS);
// Usa 8-bit com 2 stop bits
      #else
```

```
	UCSRC |= (1 << URSEL) | (1 << UCSZ0) | (1 << UCSZ1);  // // Usa 8-
bit com 1 stop bits
	#endif

	UBRRL = BAUD_PRESCALE;		// Parte baixa da taxa de trans-
missão
	UBRRH = (BAUD_PRESCALE >> 8);	// Parte alta da taxa de transmis-
são

	UCSRB |= (1 << RXCIE);		// Habilita a interrupção de
transmissão
}

ISR(USART_RXC_vect) // interrupção de recepção
{
	char c;
	c=UDR;
	if(c==120)
	{
		PORTB|=(1<<PB0);
	}
	else
	{
		PORTB&=~(1<<PB0);
	}
}

int main(void)
{
	DDRB|=(1<<PB0);
	unsigned char cont=0;
	usart_init();
	sei();

	while(1)
	{
		usart_send(cont);
		cont++;
		_delay_ms(1000);
	}
}
```

Este exemplo transmite pela porta serial o valor da variável “cont” de 1 em 1 segundos. Quando alguma informação é recebida pela porta serial, o microcontrolador executa a interrupção e compara o valor recebido com o número 120. Se for igual a saída PB0 é ativada, caso contrário a saída PB0 é desativada.

Parte experimental

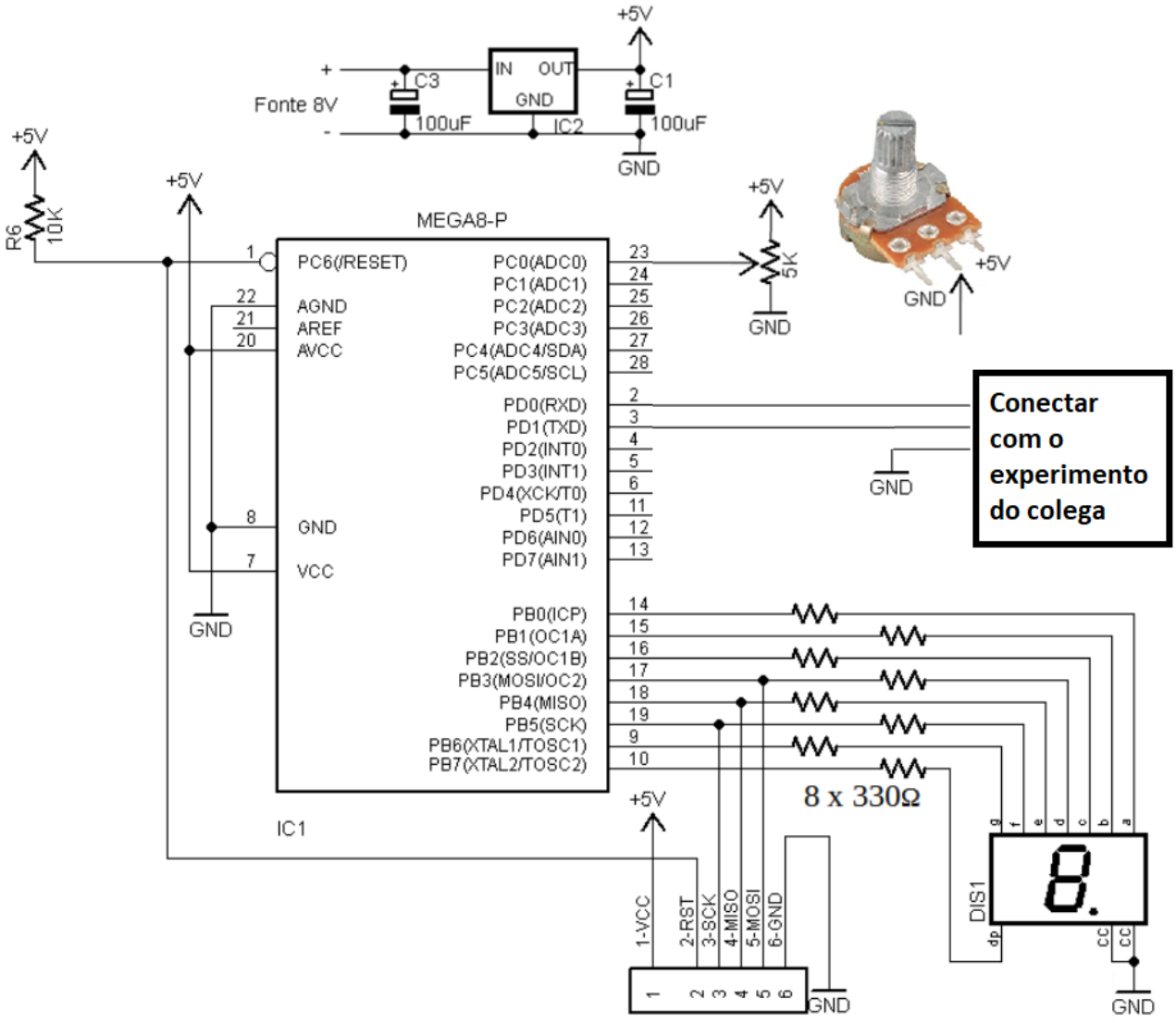
Introdução

O objetivo deste experimento é desenvolver práticas utilizando a comunicação serial de forma a preparar os alunos para estabelecer comunicação entre os microcontroladores e outros sistemas eletrônicos.

Experimento

Para o circuito da figura a seguir faça um programa que lê a tensão na entrada ADC0 do conversor A/D. O valor lido deve ser convertido para uma informação de 8 bits e transmitido pela serial 10 vezes por segundo com uma taxa de transmissão de 1200bps.

O valor transmitido deve ser proporcional a entrada analógica de forma a enviar o número 255 para uma tensão de entrada de 5V.



O programa deve também receber e processar as informações presentes na entrada serial. O valor recebido deve ser convertido para um número de 0 a 5, como na tabela a seguir, de forma a corresponder a tensão na entrada do conversor A/D do microcontrolador transmissor.

Tensão	Display
0,0-0,5V	0
0,5-1,5V	1
1,5-2,5V	2
2,5-3,5V	3
3,5-4,5V	4
4,5-5,0V	5

Desta forma dois alunos poderão testar o funcionamento de seu experimento transmitindo e recebendo dados entre si. O display conectado ao microcontrolador do experimento do primeiro aluno deve indicar a tensão na entrada analógica do microcontrolador do experimento do segundo aluno, e vice-versa.

Observação: utilize o osciloscópio para visualizar o sinal da comunicação serial.

Relatório

Após a realização dos experimentos deve ser elaborado um relatório seguindo o modelo disponibilizado. Este relatório deve ser submetido via sistema SIGAA para avaliação, em formato PDF, até a data estipulada em aula.

O modelo do relatório pode ser encontrado em: https://professor.luzerna.ifc.edu.br/ricardo-kerschbaumer/microcontroladores-experimental/

Serão avaliados os seguintes itens no relatório:

- Introdução
- Objetivo
- Fundamentação teórica
- Desenvolvimento
- Componentes utilizados
- Circuito eletrônico
- Código fonte do programa
- Resultados e discussões
- Conclusão