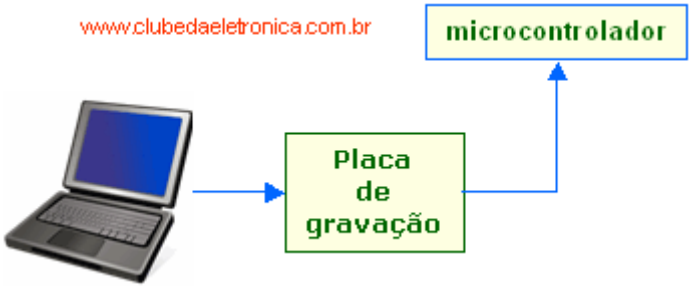
Circuito de gravação (AVR® programmer)

Introdução

Nossa proposta, nesta parte do trabalho, é apresentar um circuito para gravação ISP (In-System-Programming) para microcontroladores AVR®. Este circuito, nada mais é que uma interface entre o computador e o microcontrolador é este circuito permitirá que o código **hex** seja transferido serialmente do computador, local onde foi gerado, para a memória flash da maioria dos microcontroladores AVR®.

A idéia hardware sugerida é colocar a placa entre o PC e o microcontrolador, conforme mostrado no esquema abaixo.



Conceitos básicos

A **Transmissão** do código será **Full Duplex**, **Serial** e **Síncrona**, feita por um circuito de gravação **ISP (In-System-Programming)** utilizando um padrão de comunicação denominado **SPI (Serial Peripheral Interface)**. Acho que deu para perceber que há necessidade de conhecermos alguns termos que fazem parte do dia a dia qualquer pessoa que trabalhe com microcontrolador. Vamos à eles:

Transmissão serial de dados

Transmitir e receber dados são práticas corriqueiras entre sistemas digitalizados. Há basicamente duas maneiras para transmissão/recepção de dados, são:

- Transmissão serial (podendo ser síncrona ou assíncrona)
- Transmissão paralela.

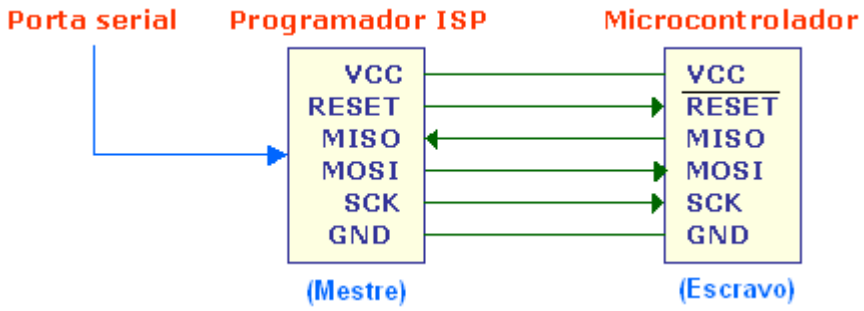
Neste momento, só nos interessa à **Serial Síncrona com transmissão Full duplex**, então vamos aos seus principais conceitos:

- A transmissão é feita serialmente, ou seja, um após o outro.
- Full duplex é um canal duplo para comunicação, ou seja, um para envio de dados e outro para recepção de dados.
- Permite a comunicação simultânea, ou seja, pode receber dados ao mesmo tempo em que transmite sem complicações.
- O sincronismo permite que a informação seja transmitida continuamente pelo canal, ou seja, sem intervalo entre os bits.

Dispositivos ISP - In-System Programming

Permite a programação e reprogramação de qualquer microcontrolador já posicionado dentro do sistema, ou seja, já soldado em uma placa de desenvolvimento ou de aplicação. Isto é bastante útil, porque podemos atualizar o programa de um microcontrolador já em campo.

Para programação **ISP (In-System Programming)** é necessário que haja um padrão pré-estabelecido entre eles, neste caso, o padrão é chamado de **SPI (Serial Peripheral Interface)** que consiste no controle de um barramento de seis vias, que fará a ligação entre o circuito de gravação e o microcontrolador. Vejamos:



Importante:

Barramento ⇒ *é um conjunto de linhas que permitem a comunicação entre dispositivos*

Protocolo ⇒ *é um padrão que controla e possibilita a conexão, comunicação ou transferência de dados entre dois sistemas dispositivos. Os protocolos podem ser implementados pelo hardware, software ou por uma combinação dos dois.*

Protocolo SPI - Serial Peripheral Interface

É um padrão para comunicação serial, desenvolvido pela Motorola, ele é necessário em qualquer tipo de comunicação, e com os microcontroladores não é diferente, ou seja, para enviar ou receber dados o microcontrolador necessita de uma via como um controle de trafego que chamamos SPI.

Outro padrão é o **I^2C**, que veremos em um outro momento.

A comunicação serial de dados via SPI

O padrão SPI consiste um dispositivo mestre (que controla a transferência) e um ou mais dispositivos escravos. Assim, para que ocorra a transferência do código **hex** (hexadecimal) para o microcontrolador, devemos torna-lo um escravo, sendo o computador, através do circuito de gravação, o mestre.

Das seis linhas do barramento apresentada, para programação In-system, duas são para alimentação (VCC e GND) as outras quatro fazem parte do protocolo SPI, sendo:

- **MISO** ⇒ Mestre na entrada e escravo na saída de dados
- **MOSI** ⇒ Mestre na saída e escravo na entrada de dados
- **SCK** ⇒ é o clock de sincronismo entre a entrada e a saída de dados
- **SS'** ⇒ seleção de escravo (em "0" será escravo em "1" será mestre)

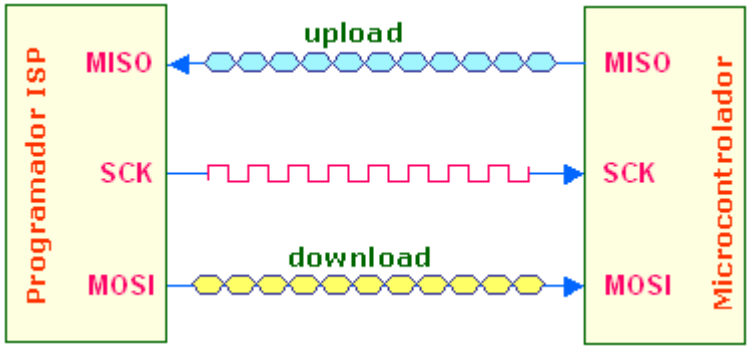
Se o microcontroladores AVR não possuir o pino SS', a seleção poderá ser feita pelo RESET, ou seja, RESET for "0" ele se torna um escravo e 1 ele será o mestre. Obviamente, esta comutação entre "0" e "1" será feita pelo circuito programador.

- **SCK ⇒ Master Clock output, Slave Clock input. (Port B, Bit 7)**

  (Se mestre envia gera clock. Se escravo recebe clock)

  Quando o dispositivo de gravação liga o pino de RESET no GND, o microcontrolador torna-se escravo e, portanto, começa a receber pulsos de clock de seu mestre o dispositivo de gravação, isso independente das configurações do PortB.7.

  Quando o Pino de RESET for à "1" o microcontrolador passa a ser mestre, neste caso, respeita-se configuração de DDRB.7 e o comando PORTB.7.



Estes pulsos são guias para o envio **(upload)** e o recebimento **(download)** de dados pelo microcontrolador.

- **MISO ⇒ Master Data Input, Slave Data Output (Port B, Bit 6)**

  (Se mestre é entrada de dados. Se escravo é saída de dados)

  Quando o dispositivo de gravação liga o pino RESET no GND, o microcontrolador passa a ser escravo e por conta disso o pino MI**SO** é configurado como saída de dados, ou seja, permite ao dispositivo de gravação a **leitura de dados** oriundos do microcontrolador, isso independente da configuração de DDRB.6.

  Quando o Pino de RESET for à "1" o microcontrolador passa a ser mestre, neste caso, respeita-se a configuração de DDRB.6 e o comando PORTB.6.

- **MOSI ⇒ Master Data Output, Slave Data Input pin for SPI channel (Port B, Bit 5)**

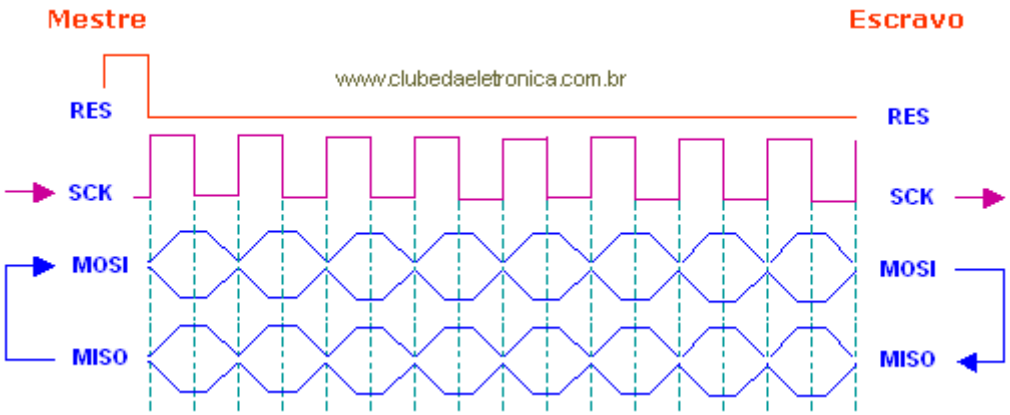
  (Se mestre é saída de dados. Se escravo é entrada de dados)

  Quando o dispositivo de gravação liga o pino de RESET no GND, o microcontrolador torna-se escravo e por conta disso o pino MO**SI** é configurado como entrada de dados no microcontrolador, ou seja, permite ao dispositivo de gravação **escrita de dados** na memória flash, isso independente da configuração de DDRB.5.

  Quando o Pino de RESET for à "1" o microcontrolador passa a ser mestre, neste caso, respeita-se configuração de DDRB.5 e o comando PORTB.5.

O protocolo SPI no diagrama do tempo

Abaixo uma ilustração sobre o comportamento do sinal no tempo, agora incluindo o reset que se estiver em zero dá inicio a transmissão.

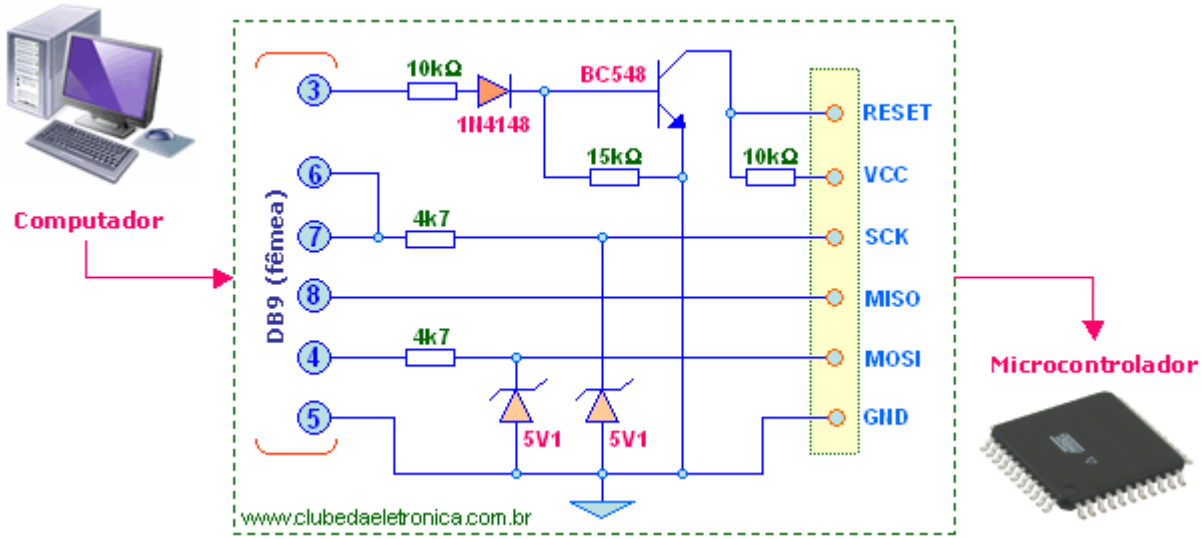


O circuito de gravação (AVR® Programmer)

Para transferência serial do código **hex** para o microcontrolador não é necessário nenhum circuito complexo, pelo contrário é bastante simples. Porém, necessita de alguns cuidados pois será conectada à porta serial do computador, ou seja, deve-se monta-lo com extremo cuidado preferencialmente sob orientação.

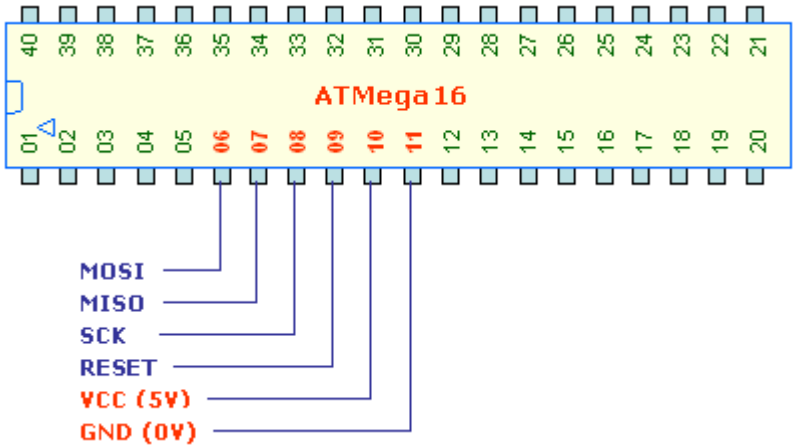
Este Programador AVR® foi extraído da página http://electronics-diy.com e é compatível com o software de gravação **Pony-Prog 2000,** que pode ser obtido gratuitamente no link http://www.lancos.com/prog.html. Seu custo é extremamente baixo se comparado a outros programadores disponíveis no mercado.

O circuito:



Conexões com o ATMEGA16

Agora, utilizaremos o Atmega 16 como exemplo, embora qualquer outro microcontrolador AVR possa ser usado, desde que observado a posição numérica dos pinos. Vejamos os pinos do Atmega16 necessários para o gravador.



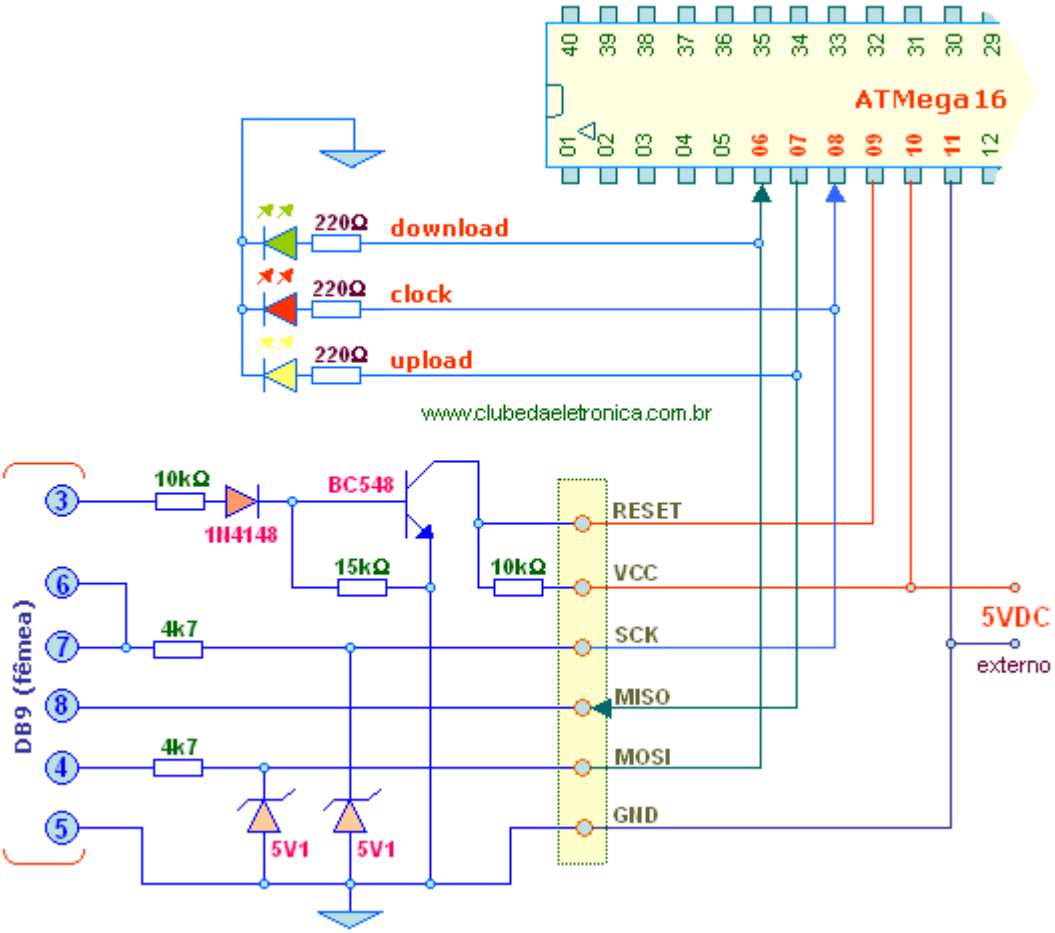
Como vantagem sobre os outros, podemos citar:

Microcontrolador	Portas de I/O	Memória flash	Conversor AD
AT90S2313	13	2kbyte	Não tem
ATTiny2313	15	2kbyte	Não tem
Atmega8	23	8kbyte	5 canais (Resol. 10bits)
Atmega16	32	16kbyte	8 canais (Resol.10bits)

Obviamente, o melhor microcontrolador esta diretamente relacionado com o seu projeto.

Conectando o Atmega16 à interface de gravação

Implementamos, três LEDs opcionais aos pinos MOSI, MISO e SCK, para indicar, o clock de sincronismo, a escrita (download) e a leitura (upload) do microcontrolador.



Uma fonte 5V externa é necessária para alimentação do Atmega16, pode também ser obtida da porta USB do PC.

Cuidado, a ansiedade e a imperícia poderão acarretar danos irreparáveis.

Continua...