Dicas de Porta Paralela

O modelo tradicional de porta paralela, utilizado desde os tempos do XT, é conhecido como "normal" ou SPP (Single Parallel Port). Possui taxa de transferência de 150 KB/s e é unidirecional. Para a conexão micro-micro ou na conexão de equipamentos externos (como o ZIP Drive), o sistema usa transmissão 4 bits por vez, utilizando sinais de retorno como "busy", "paper out", etc. Este sistema só funciona bem mesmo com impressoras. Para a conexão de ZIP drives e até mesmo impressoras mais rápidas, a porta paralela tradicional é muito lenta, sobretudo porque é unidirecional e utiliza apenas 4 bits de retorno (ou seja, transmite a 8 bits, porém recebe informações a 4 bits por vez).

Para resolver este problema, foi desenvolvida a porta paralela avançada ou EPP (Enhanced Parallel Port). Este modelo de porta paralela é bidirecional e atinge uma taxa de transferência de 2 MB/s. Entretanto, para atingir esta taxa, necessita de um cabo especial, pois o cabo tradicional só comporta uma taxa de até 150 KB/s. Este cabo é conhecido no mercado como "cabo bidirecional", sendo que sua verdadeira característica não é ser bidirecional, mas sim permitir altas taxas de transmissão.

Aumentar a taxa de transferência trouxe um problema: a necessidade de mais atenção por parte do processador. Para resolver isto, desenvolveu-se a porta paralela ECP (Enhanced Capabilities Port). Ela é igual a EPP porém utiliza um canal de DMA, que faz com que a transmissão e recepção sejam feitas sem a intervenção do processador, aumentando o desempenho do micro.

Todos os micros novos possuem porta paralela na própria placa-mãe ("on board"), permitindo que você, através do setup do micro, configure-a a trabalhar em qualquer um dos três modos de operação. O modo que você deverá trabalhar depende do caso. A maioria dos periféricos conectados na porta paralela aceitam somente o modo normal (SPP). Alguns outros periféricos, como é o caso do ZIP Drive, funcionam perfeitamente no modo normal, mas terão sua taxa de transferência (ou seja, seu desempenho) aumentada sensivelmente se o modo da porta paralela for EPP ou ECP. Outros periféricos, como é o caso das impressoras HP série 800 e Epson Stylus Color II, necessitam obrigatoriamente que a porta paralela esteja operando em modo EPP ou ECP, necessitando, portanto, do tal "cabo bidirecional".

Controle de dispositivos externos através da porta paralela utilizando C#

Antes de começarmos precisamos entender alguns conceitos importantes.

Porta Paralela

Modos de Operação

A porta paralela atualmente possui três modos de operações. São eles:

SPP – bits de dados unidirecional
EPP – bits de dados biderecional
ECP – bits de dados bidericional

Estes modos de operação são configurados pelo BIOS Setup.
A diferença entre EPP e ECP é que esta última utiliza DMA(acesso direto a memória).

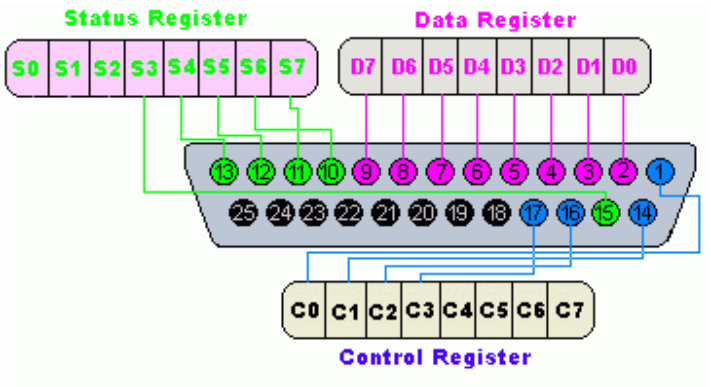
Endereços da Porta Paralela

Para enviarmos ou recebermos dados da porta paralela precisamos saber o seu endereço base.

Nome da Porta no SO	Endereço
LPT1	378 hexadecimal / 888 decimal
LPT2	278 hexadecimal / 632 decimal

Pinagem

Pinagem da porta paralela DB-25.



A pinagem no conector DB25 é dividida em três grupo, são eles:

1. Pinos de Dados (Data Register)
2. Pinos de Controle (Control Register)
3. Pinos de Status (Status Register)

Em nosso projeto utilizaremos apenas os pinos de dados (D0 a D7) para controlar dispositivos externos. E como estaremos apenas enviando sinal de saída (unidirecinoal), o projeto funcionará independente do modo de operação definido no BIOS Setup.

Observe também que D0 a D7 representam 8 bits (1 byte). Então o valor de saída pode variar entre 00000000 (0 decimal) e 11111111(255 decimal).

Software

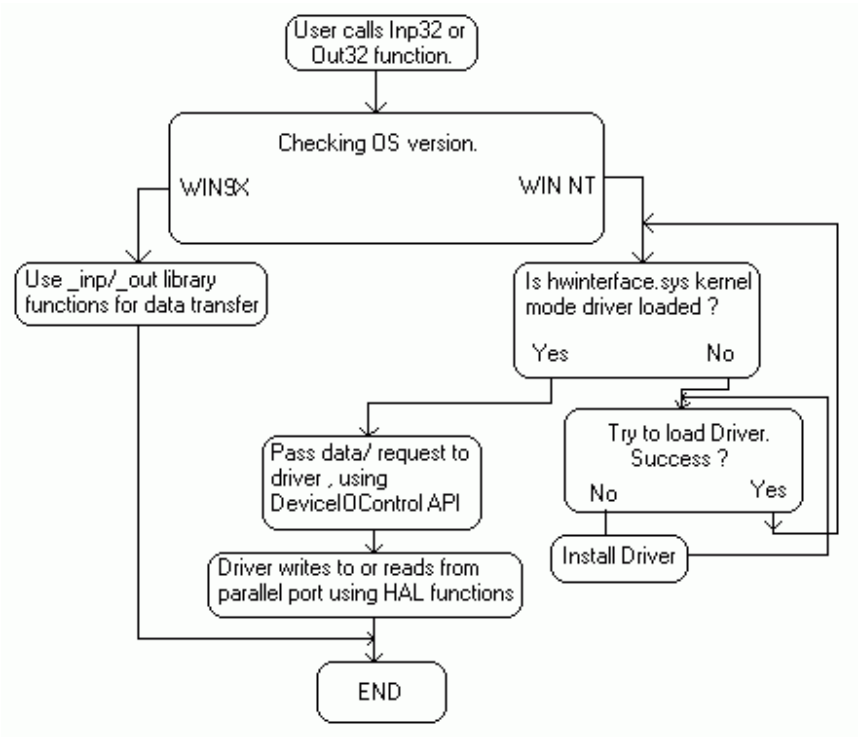
Bem, agora que já temos um embasamento técnico sobre o hardware da porta paralela, iremos nos concentrar no software que irá controlá-la.

Nos sistemas operacionais da família Windows 9x , a maioria das linguagens de programação acessavam com facilidade a porta paralela através de funções nativas da própria linguagem ou via código assembler.

Sistemas operacionais como o Windows NT/2000/XP não permitem o acesso direto a este tipo de porta. Nesse caso é necessário um driver de sistema.

Para contornar este problema usaremos a Inpout32.dll que pode ser adquirida em http://www.logix4u.net/.

Veja o funcionamente básico da Inpout32.dll na figura abaixo:

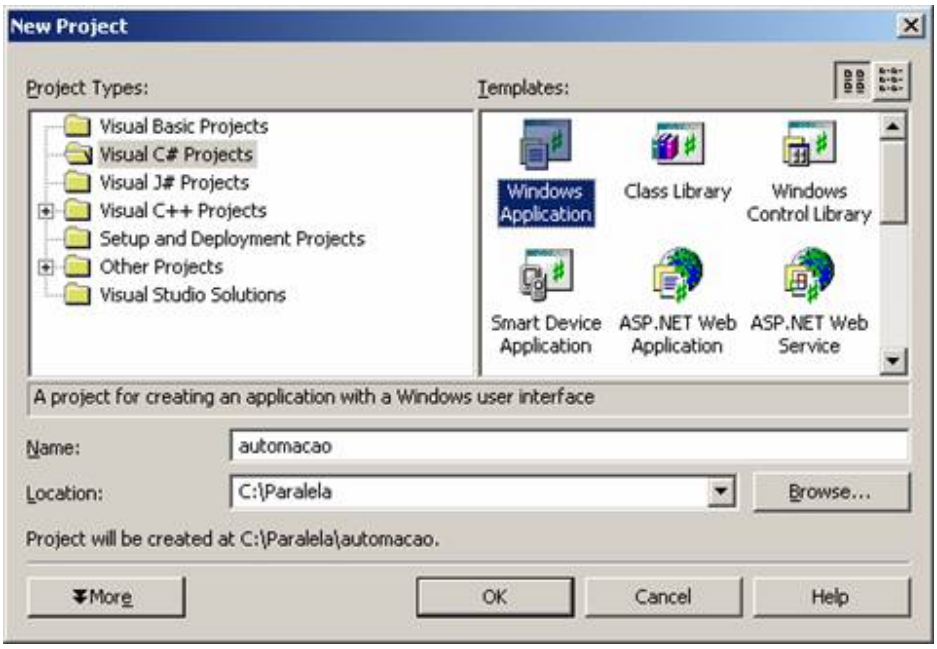


A Inpout32.dll possui duas funcões: Out32 e Inp32. A primeira para escrever um valor (byte) num endereço de I/O e a segunda para ler um valor (byte) de um endereço de I/O.

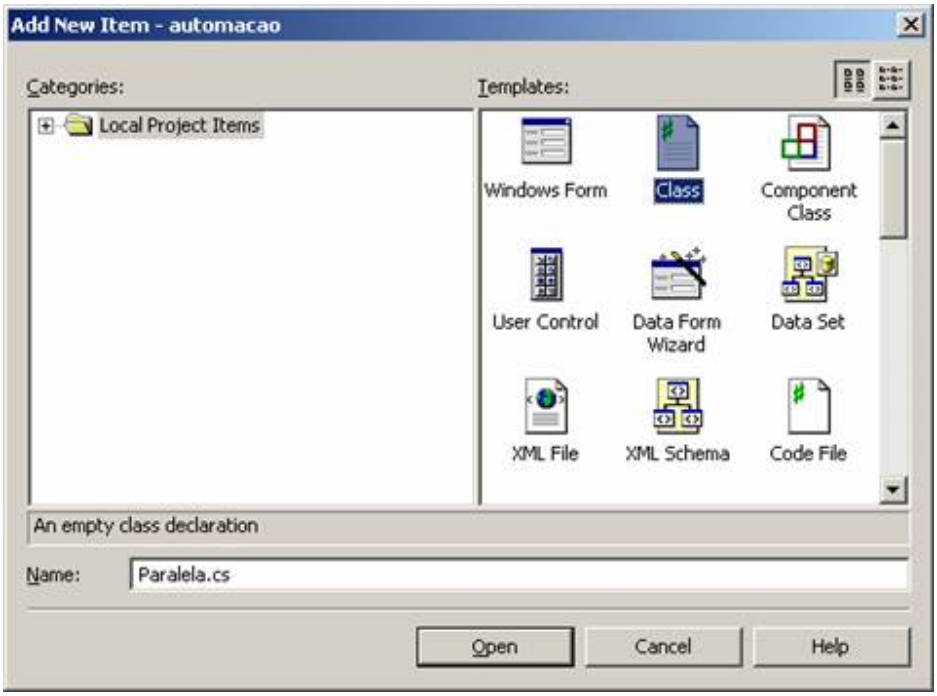
Como não se trata de uma dll .NET devemos utilizá-la como Código Não-Gerenciado (UnManage Code).

Aplicativo

Vamos então ao nosso software. Inicie o VS.NET e crie uma nova aplicação C# Windows Forms.



Vamos então criar uma classe de nome Paralela.cs para manipular a dll. Veja figura abaixo.



Escreva o seguinte código para a classe:

```
using System;
using System.Runtime.InteropServices;
```

```
namespace automacao
{
        /// <summary>
    /// Summary description for Paralela.
    /// </summary>
    public class Paralela
    {
        // Escreve um byte no endereço
       [DllImport("Inpout32.dll", EntryPoint="Out32")]
                public static extern void Escrever(int endereco, byte valor);

       // Lê um byte do endereço
       [DllImport("Inpout32.dll", EntryPoint="Inp32")]
       public static extern byte Ler(int endereco);
           }
}
```

Não esqueça de declarar o namespace que permite a utilização de dlls não gerenciadas.

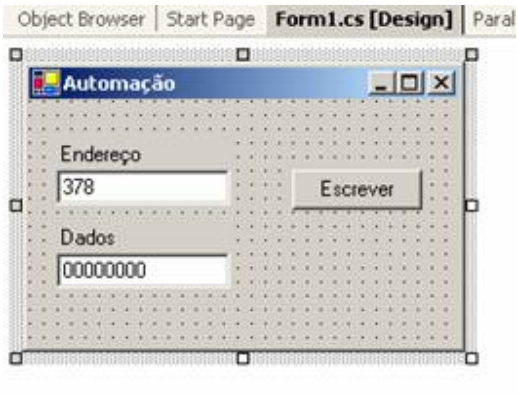
```
using System.Runtime.InteropServices;
```

Embora o nosso projeto utilize apenas o método de escrita (enviar dados para um dispositvo externo), decidi importar também o método de leitura. Então nossa classe possuirá dois métodos estáticos que são:

```
public static extern void Escrever(int endereco, byte valor);
public static extern byte Ler(int endereco);
```

Como deu pra perceber os endereços são do tipo inteiro e o valor escrito/lido do tipo byte. Dessa forma podemos garantir que o valor lido/escrito estará sempre entre 0 e 255.

Agora vamos retornar ao nosso formulário e deixá-lo conforme a figura abaixo:



Vamos definir o seguinte código para o botão Escrever:

```
private void btnEscrever_Click(object sender, System.EventArgs e)
```

```
{
        int endereco = Convert.ToInt32(txtEndereco.Text.Trim(), 16);
        byte dados = Convert.ToByte(txtDados.Text.Trim(), 2);
        Paralela.Escrever(endereco,dados);
        MessageBox.Show("O byte " + txtDados.Text.Trim() +
                " foi enviado para o endereço " +  txtEndereco.Text.Trim()
                , "Automação");
}
```

Observe que utilizamos os métodos estáticos da classe Convert para converter os valores fornecidos para inteiro e byte.

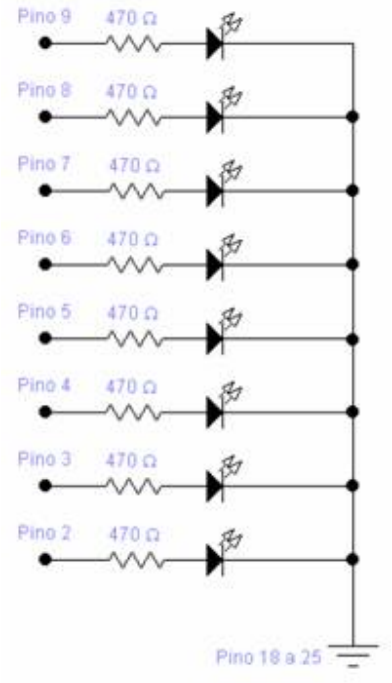
Pronto! Com isso nossa aplicação já está pronta para enviar dados para a porta paralela. Para testarmos nossa aplicação vamos construir um hardware bem simples.

Hardware

Como o nosso objetivo é apenas didático, iremos montar um circuito com 8 leds que serão acionados de acordo com a saída de dados de nossa aplicação. Observe que nada impede de utilizarmos um circuito com relés para acionar dispositivos mais potentes. Aconselho a pedir a ajuda de um técnico caso você não possua conhecimentos em eletrônica suficientes para montar o circuito. Se não desejar montar o circuito você poderá medir com um multímetro os valores de tensão diretamente nos pinos da porta paralela. Onde:

0 = 0 Volts
1 = 5 Volts

Circuito

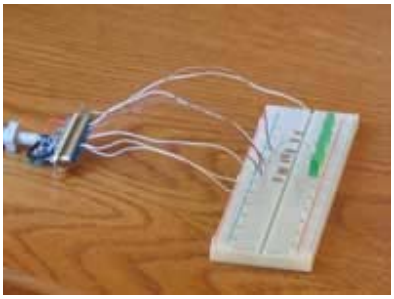


Componentes

8 Resistores de 470R
8 LEDs
1Conector DB25 macho
Cabos, placa, etc...

Exemplos do circuito montado:

1 – Circuito montado numa matriz de contato



2 – Circuito montado dentro do conector do cabo paralelo

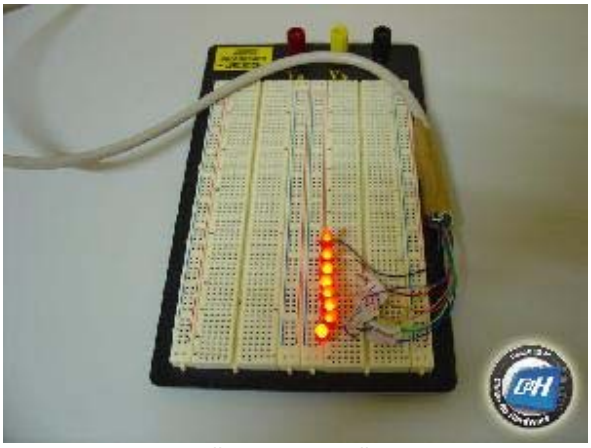
Construindo Protótipos usando a Porta Paralela

Introdução

A porta paralela do micro é o meio mais fácil para controlar dispositivos externos, como LEDs, lâmpadas e até mesmo eletrodomésticos. Neste tutorial ensinaremos a você como usar a porta paralela do micro para controlar circuitos externos.

Atualmente, as impressoras vendidas no mercado utilizam conexão USB. Com isso, na maioria dos micros modernos a porta paralela está disponível, podendo ser usada para controlar circuitos externos ao micro.

Na verdade, a idéia por trás da porta paralela é muito simples. Ela é uma interface de comunicação paralela de 8 bits, e portanto você tem oito bits disponíveis lá. Como cada bit de dados pode ser transmitido como “0” (“desligado”) ou como “1” (“ligado”), nós podemos ligar ou desligar diretamente até oito dispositivos, como LEDs, lâmpadas e até mesmo eletrodomésticos. Você pode conectar LEDs diretamente na porta paralela e brincar com eles – na verdade é exatamente isso o que iremos fazer neste tutorial, já que essa é a melhor maneira de aprender como usar a porta paralela. Mas para circuitos “pesados” como lâmpadas e eletrodomésticos, você precisará construir um circuito de potência, já que a porta paralela do micro não é capaz de fornecer corrente suficiente para dispositivos como esses. Também explicaremos como construir este tipo de circuito.

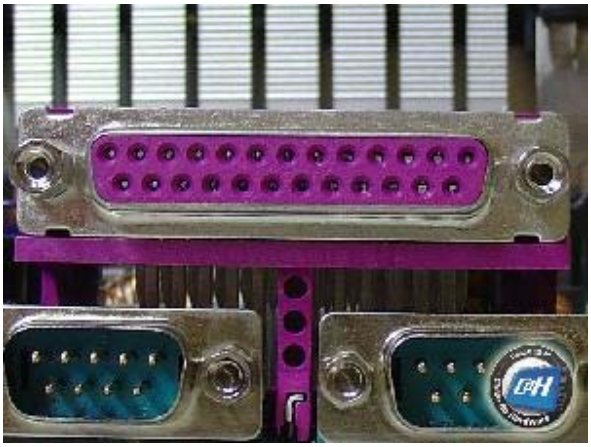


clique para ampliar

Figura 1: Conectando LEDs à porta paralela.

Entendendo a Porta Paralela

No micro a porta paralela usa um conector de 25 pinos (chamado DB-25, 25 pinos D-sub ou 25 pinos D-shell), como você pode ver na Figura 2. Nas impressoras, no entanto, é usado um tipo de conector diferente, chamado Centronics, que possui 36 pinos.



clique para ampliar

Figura 2: A porta paralela.

Além dos oito bits de dados existem mais sinais disponíveis na porta paralela. Na tabela abaixo listamos todos os sinais básicos da porta paralela e suas funções, bem como suas localizações tanto no conector padrão 25 pinos quanto no conector Centronics. A Coluna E/S indica se o sinal é de entrada (E) ou de saída (S). Entrada significa que o sinal tem que vir do dispositivo para a porta paralela (isto é, o sinal deve ser gerado pelo nosso protótipo); saída significa que o sinal vem da porta paralela.

Sinal	Nome	Pino (Conector padrão 25 pinos)	Pino (Conector Centronics 36 pinos)	E/S	Descrição
/STROBE	Strobe	01	01	S	Indica se o dado está pronto ou não para ser transmitido. (0= Dado pronto para ser transmitido, 1= Dado não está pronto para ser transmitido.)
/ACK	Acknowledge	10	10	E	Indica que a impressora está pronta para receber dados.
BUSY	Busy	11	11	E	Indica que a impressora não está pronta para receber dados.

PE	Paper Empty	12	12	E	Indica que a impressora está sem papel.
SELECT	Select	13	13	E	Indica que a impressora está “on line” pronta para receber informações.
/AUTO FD XT	Auto Feed	14	14	S	A impressora move o papel para o início da próxima linha.
/ERROR	Error	15	32	E	Aconteceu algum erro (impressora desabilitada, sem papel).
/INIT	Init	16	31	S	Reinicia a impressora e limpa seu buffer de impressão.
/SELECT INPUT	Select Input	17	36	S	Dados podem ser transferidos para impressora apenas quando esta linha estiver em “0”.
D0 até D7	D0 até D7	2 até 9	2 até 9	S	Bits de Dados.
GND	Ground	18 até 25	19 até 30	S	Terra.

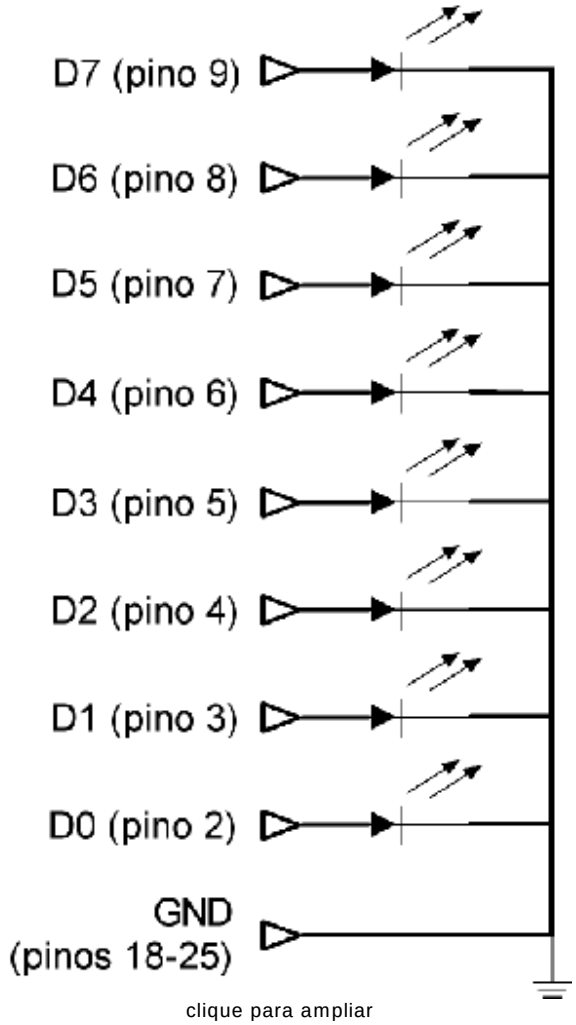
A porta paralela utiliza três endereços de E/S: dados (378h), status (379h) e controle (37Ah). Se você quer enviar dados para um dispositivo externo ao micro através da porta paralela, basta escrever os dados no endereço de dados da porta paralela. Por exemplo, se quisermos ligar todos os nossos LEDs, tudo o que temos que fazer é enviar o valor 255 (que é o decimal equivalente para 11111111, isto é, todos os bits de dados configurados como “ligado”) para o endereço 378h. Claro que explicaremos mais sobre isto e também falaremos mais sobre os endereços de status e controle.

Construindo um Protótipo Básico

Se você nunca construiu qualquer protótipo para porta paralela antes, sugerimos que você comece com o mais básico de todos: um conjunto de oito LEDs, onde cada um dos LEDs é conectado a um bit de dados da porta paralela. Com este protótipo básico você será capaz de aprender muito sobre o funcionamento da porta paralela.

Quando um pino de dados é colocado em “0”, você encontrará 0 V nele. Quando ele é colocado em “1” você encontrará 5 V. Isto é o suficiente para ligarmos os LEDs, mas não para ligar lâmpadas e eletrodomésticos; explicaremos adiante como alimentar dispositivos “pesados”.

Portanto, tudo o que você precisa fazer é conectar cada um dos pinos de dados da porta paralela (pinos 2 até 9) ao LED (em seu terminal anodo, também conhecido como “terminal positivo”) e utilizar um pino de terra (qualquer um do 18 até 25) para conectar os terminais catodo (também conhecido como “terminal negativo”) de todos os LEDs. Você pode ver o esquema na Figura 3.



clique para ampliar

Figura 3: Esquema para usar a porta paralela.

Como os LEDs possuem polaridade, você deve prestar atenção para localizar corretamente seus terminais anodo (positivo) e catodo (negativo). Se você prestar bem atenção, irá reparar que os LEDs não são completamente redondos: o lado do catodo é um pouco achatado, como você pode ver na Figura 4.

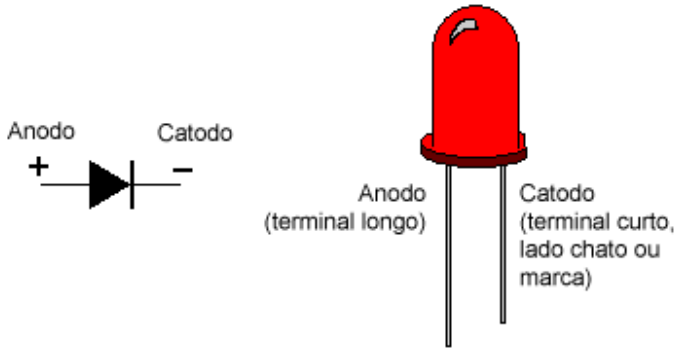
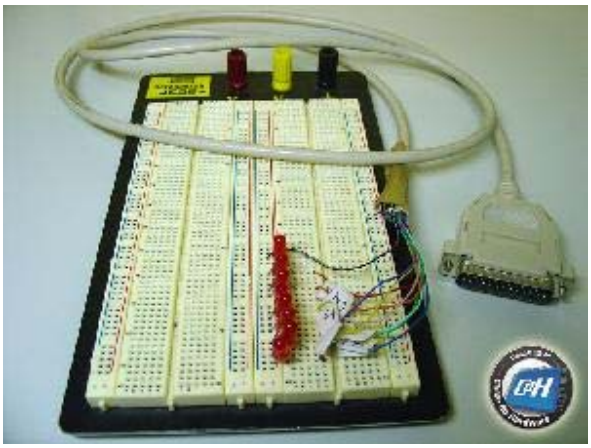


Figura 4: Terminais de um LED.

Para construir circuitos, recomendamos a utilização de um “protoboard”. Protoboards permitem a você montar protótipos sem a necessidade de nenhum tipo de solda.

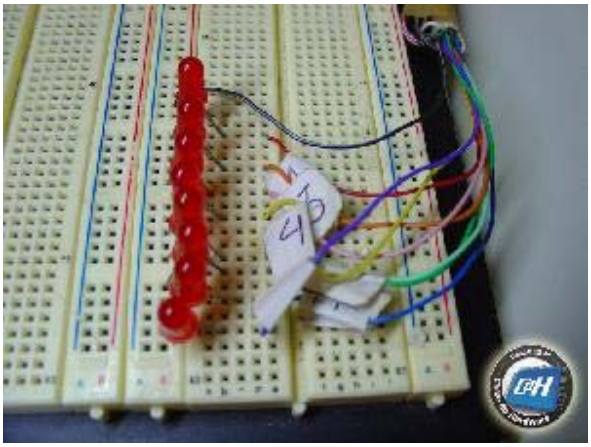


clique para ampliar

Figura 5: Usando um protoboard para construir nosso protótipo.

Construindo um Protótipo Básico (Cont.)

O modo mais fácil para construir o cabo que será utilizado para conectar a porta paralela ao seu protótipo na protoboard é pegando um cabo de impressora padrão e cortar fora o conector Centronics. Feito isso, você precisará descobrir onde cada fio está conectado. Com um multímetro na escala de resistência (ou continuidade), coloque uma das pontas no fio que você está tentando descobrir sua função e teste a outra ponta em cada um dos pinos do conector de 25 pinos do cabo. Quando a resistência for zero (ou o multímetro emitir um bipe, se você estiver usando sua escala de continuidade), você descobrirá que pino no conector de 25 pinos o fio está conectado. Rotule o fio com a função do pino (dessa forma você não precisará repetir todo o processo novamente no futuro) e vá para o próximo fio, até você ter encontrado a função de cada fio no cabo.



clique para ampliar

Figura 6: Detalhe do nosso protoboard. Veja como rotulamos os fios.

A respeito da numeração dos pinos, preste atenção no conector de 25 pinos e você verá que cada pino está numerado, veja na Figura 7.

clique para ampliar

Figura 7: Detalhe do conector de 25 pinos. Veja como cada pino está numerado.

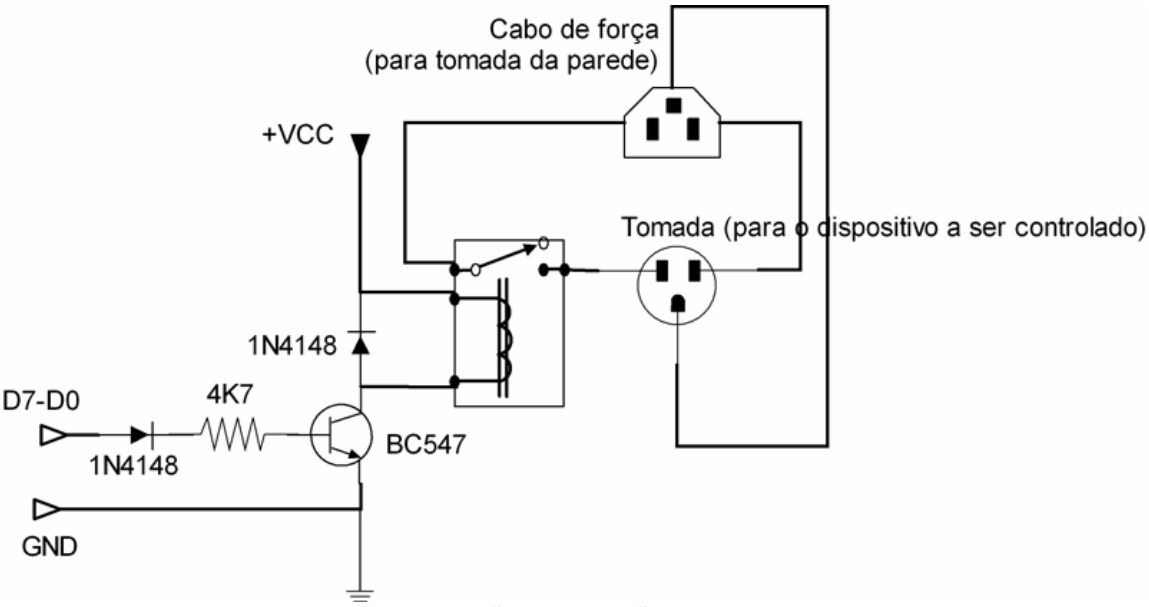
Interface de Potência

Se você precisa controlar outros dispositivos além de LEDs, você precisará projetar e construir uma interface de potência. A idéia básica é conectar um transistor agindo como uma chave na saída dos dados, e este transistor chaveando o dispositivo que você quer controlar para ligado ou desligado. Se você quer controlar circuitos AC – lâmpadas e eletrodomésticos, por exemplo – você precisará usar um relé. O relé é uma chave que liga toda vez que a corrente elétrica passa por ele.

Você pode ver uma interface de potência básica na Figura 9. Você precisará repetir este circuito para cada bit que deseja usar, isto é, se você quer usar os oito bits da porta paralela de modo a controlar até oito circuitos AC, você precisará repetir este circuito oito vezes, um para cada bit de dados.

Você precisará de uma fonte de alimentação externa com a mesma tensão do seu relé. Assim se você usar um relé de 12 V você precisará de uma fonte de alimentação de 12

V conectada ao +Vcc e terra. “Cabo de força” é o fio que será conectado na tomada da parede e “Tomada” é a tomada em seu circuito onde as lâmpadas ou eletrodomésticos serão conectados.



clique para ampliar

Figura 9: Interface de potência.

Os diodos trabalham como proteção e apesar de recomendarmos o 1N4148 qualquer outro diodo de uso geral funcionará bem. O mesmo vale para o transistor, que recomendamos o BC547, mas qualquer outro transistor NPN de uso geral também pode ser usado.

Recursos Avançados

Até agora falamos apenas sobre o envio de dados para fora da porta paralela. Na verdade, você também pode ler dados usando a porta paralela. A porta paralela padrão, também conhecida como SPP, usa dois endereços extras para status (379h) e controle (37Ah). Se você ler o conteúdo do endereço 379h você será capaz de ler o estado dos pinos busy, acknowledge, Paper Empty, Select e Error encontrados na porta paralela. Isto pode ser muito útil se você deseja construir um circuito para enviar dados para o computador. Por exemplo, se você tem algum tipo de sensor e quer um programa para ligar um alarme se este sensor disparar, esta é uma maneira de realizar isto.

Endereços de Status

Como mencionamos acima, a leitura do endereço de E/S 379h faz com que você tenha acesso aos pinos Busy, Acknowledge, Paper Empty, Select e Error. Você obtém um valor de 8 bits com o seguinte formato:

bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
/BUSY	ACK	PE	SELECT	ERROR	X	X	X

Endereço de Controle

A escrita de dados neste endereço de E/S (37Ah) permite você a controlar outras linhas disponíveis na porta paralela. Na verdade, você tem mais bits de saída na porta paralela do que os oito bits de dados padrão, mas esses bits extras são acessados em um endereço diferente. Além disso, o bit número 4 do endereço de controle mascara a IRQ7. Com este bit configurado como “1” a IRQ7 pode ser usada.

bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
X	X	X	IRQ 7	/SELECT INPUT	INIT	/AUTO FD XT	/STROBE

Modo Bi-Direcional

Se você já entendeu o básico, pode seguir adiante e estudar dois diferentes modos de operação da porta paralela: EPP (Enhanced Parallel Port, Porta Paralela Aprimorada) e ECP (Enhanced Capabilities Port, Porta com Capacidades Estendidas). Esses dois modos são genericamente chamados “modos bi-direcional”, já que nesses modos os pinos de dados podem ser usados tanto para entrada como para saída, o que não ocorre na porta paralela padrão, SPP, onde a porta pode ser usada apenas para enviar dados, mas não para receber (isto não é totalmente verdade, já que você pode usar bits de status para receber dados – esta técnica é chamada modo nibble).