TUTORIAL
BARRA DE LEDS

Autor: Tiago Lone
Nível: Básico
Criação: 19/12/2005
Última versão: 18/12/2006

Maxwell Bohr
Instrumentação Eletrônica

http://www.maxwellbohr.com.br
contato@maxwellbohr.com.br

PdP
Pesquisa e Desenvolvimento de Produtos

http://www.automato.com.br
atendimento@automato.com.br

1 – Introdução

Nesse tutorial aprenderemos a manipular as duas barras de LEDs do *Módulo de Motores e Displays*. Essas barras podem ser utilizadas para transmitir informações em uma montagem. Vamos ver o que é necessário fazer para controlar, através de um programa, o estado dessas barras. Para auxiliar nesse estudo, será desenvolvido um programa que permite controlar o estados das barras de LEDs do *Módulo de Motores e Displays*.

2 – Material

O programa desenvolvido nesse tutorial vai utilizar o *Módulo Principal* e o *Módulo de Motores e Displays* sem nenhum ítem extra, pois as barras de LEDs ficam soldadas na placa principal desse módulo. Para a criação do programa será necessário o Borland Delphi 6. A seguir a imagem da placa principal do *Módulo de Motores e Displays* do Kit onde podemos visualizar as barras de LEDs.

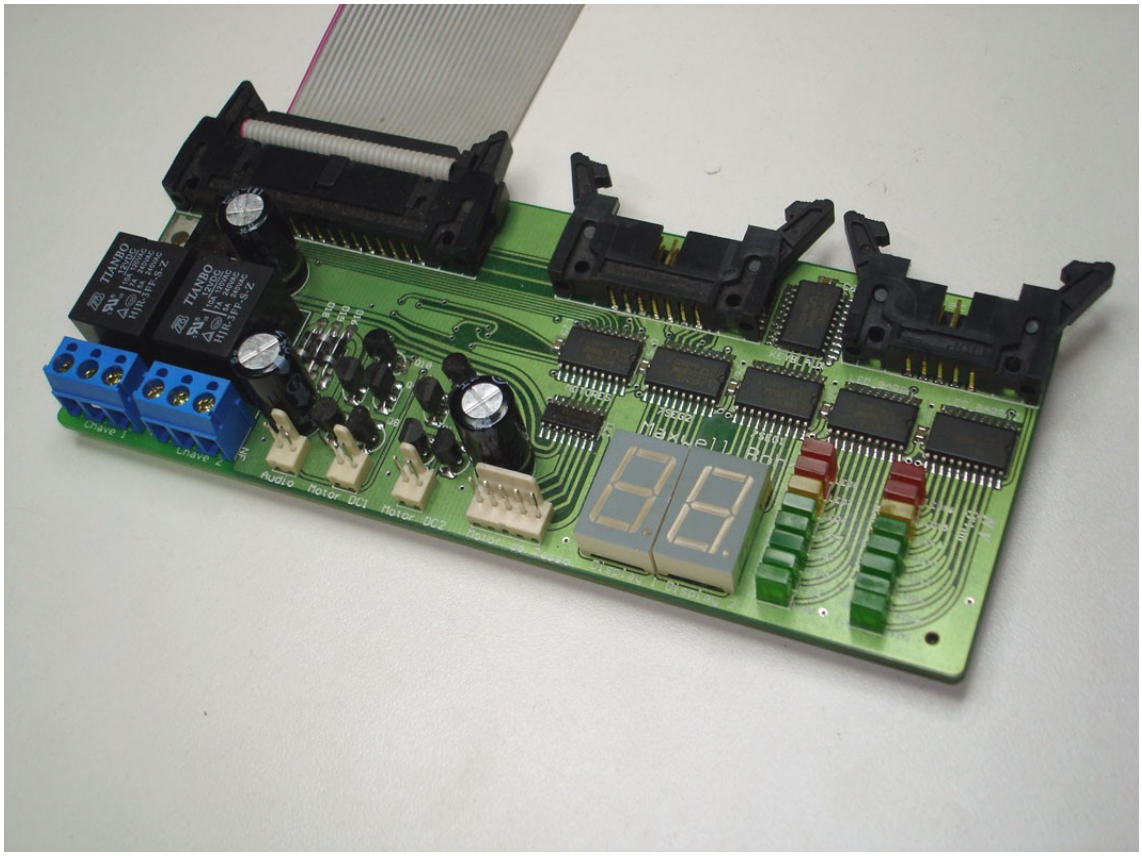

Figura 1: *Placa do Módulo de Motores e Displays.*

3 – Projeto

Vamos criar nesse tutorial um programa que permite o controle total sobre as barras de LEDs. Ele permitirá que controlemos o estado de cada um dos LEDs das barras. A aparência do programa será a seguinte.

Figura 2: *Aparência do programa que será criado nesse tutorial.*

Nosso primeiro passo na criação desse programa é desenvolver a interface gráfica. Vamos utilizar o projeto criado no tutorial Base, que já nos fornece algumas funcionalidades interessantes, e em cima dele vamos adicionar alguns componentes gráficos extra. Para isso temos apenas que copiar o projeto daquele tutorial.

A primeira coisa que modificaremos no projeto é a propriedade Caption do Form principal, que possui o texto “Projeto Base”. Vamos modificar para “Barra de LEDs”. Com isso já podemos começar a adicionar os componentes gráficos ao Form.

Para modificar a interface gráfica criada no tutorial Base para que tenhamos a interface apresentada na imagem anterior, devemos adicionar alguns componentes ao Form principal. Adicionaremos alguns Labels, ComboBox, CheckBox e um Button. Os Labels são utilizados apenas para identificar a funcionalidade de três ComboBox. Esses ComboBox são utilizados para selecionar a barra de LEDs que queremos controlar, o modo como queremos controla-lo e por fim um para selecionar o nível da barra de LEDs. Os CheckBox são utilizados para definir quais LEDs da barra serão ligados. E por fim, o componente Button, que é um botão, é utilizado para atualizar o estado das barras de acordo com a configuração atual.

Todos esses componentes, que serão adicionados, podem ser encontrados na aba “Standard” da barra de componentes.

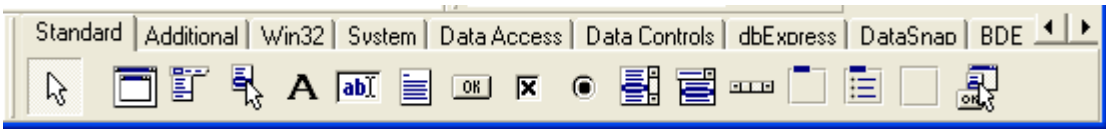

Figura 3: *Aba "Standard" da Barra de componente.*

Vamos adicionar um Label e um ComboBox para a seleção da barra de LEDs que será controlada. O componente Label possui o seguinte ícone.

Figura 4: *Ícone do componente Label.*

E o componente ComboBox possui o seguinte ícone.

Figura 5: *Ícone do componente ComboBox.*

Após adicionar os dois componentes, modificamos as seguintes propriedades do Label.

Name	=	LabelBarra
Caption	=	Barra:
Font/Style/fsBold	=	true

E as seguintes propriedades do ComboBox.

Name	=	ComboBoxBarra
Style	=	csDropDownList
Items.Strings	=	Barra I, Barra II
ItemIndex	=	0

Com isso o Form terá a seguinte aparência.

Figura 6: *Aparência do Form após a inclusão dos controles para selção da barra de LEDs que será controlada.*

Vamos adicionar mais um conjuntos de Label e ComboBox para controlar o modo de

controle das barras. Vamos oferecer dois modos de controle, um em que iremos determinar o nível da barra de LEDs e outro em que vamos controlar o estado de LED por LED da barra. Então adicionamos um Label e modificamos as propriedades a seguir.

Name	=	LabelModo
Caption	=	Modo:
Font/Style/fsBold	=	true

Em seguida adicionamos um ComboBox e modificamos as propriedades listadas abaixo.

Name	=	ComboBoxModo
Style	=	csDropDownList
Items.Strings	=	Nível, LEDs
ItemIndex	=	0

Com isso já temos na interface os controles para seleção de modo de operação. O Form deve estar semelhante ao seguinte.

Figura 7: *Interface gráfica com os controles para seleção da barra de LEDs que será controlada e do modo de operação.*

Agora vamos incluir os componentes para seleção do nível que será apresentado na barra quando o modo de operação selecionado for "Nível". Vamos utilizar novamente o conjunto Label com um ComboBox. Adicionamos o Label e modificamos as seguintes propriedades.

Name	=	LabelNivel
Caption	=	Nível:
Font/Style/fsBold	=	true

Em seguida adicionamos os ComboBox e modificamos as seguintes propriedades.

Name	=	ComboBoxNivel
Style	=	csDropDownList
Items.Strings	=	0, 1, 2, 3, 4, 5, 6, 7 e 8
ItemIndex	=	0

Assim temos a interface gráfica com a seguinte aparência.

Figura 8: *Aparência do Form após a adição dos componentes para seleção de nível.*

Agora temos que adicionar os componentes para a seleção dos LEDs que serão ligados ou que permanecerão desligados quando o modo de operação selecionado for "LEDs". Vamos utilizar para isso vários componentes do tipo CheckBox, sendo que cada um deles irá representar um LED da barra.

Esses componentes podem estar em dois estados, "Selecionados" ou "Não selecionados". Se estiverem selecionados então o LED a que eles se referem será ligado, caso contrário ele será desligado.

Podemos encontrar o componente CheckBox na aba "Standard" da barra de componentes. Ele possui o seguinte ícone.

Figura 9: *Ícone do componente CheckBox.*

Adicionaremos oito componentes desses, um para cada LED. Vamos modificar algumas propriedades deles. A seguir a lista de propriedades que vamos modificar nos oito componentes.

Name	=	CheckBoxLED0

Caption	=	LED 0:
Font/Style/fsBold	=	true

Name	=	CheckBoxLED1
Caption	=	LED 1:
Font/Style/fsBold	=	true

Name	=	CheckBoxLED2
Caption	=	LED 2:
Font/Style/fsBold	=	true

Name	=	CheckBoxLED3
Caption	=	LED 3:
Font/Style/fsBold	=	true

Name	=	CheckBoxLED4
Caption	=	LED 4:
Font/Style/fsBold	=	true

Name	=	CheckBoxLED5
Caption	=	LED 5:
Font/Style/fsBold	=	true

Name	=	CheckBoxLED6
Caption	=	LED 6:
Font/Style/fsBold	=	true

Name	=	CheckBoxLED7
Caption	=	LED 7:
Font/Style/fsBold	=	true

Após a inclusão desses componentes o Form deverá estar parecido com o seguinte.

Figura 10: *Form após a adição dos CheckBox.*

Para finalizar a interface desse programa vamos adicionar um botão para atualizar as barras de LEDs. Toda vez que ele for pressionado será enviado um comando para o Kit que irá atualizar a barra de LEDs de acordo com as seleções feitas na interface gráfica.

Para isso temos que adicionar um componente Button, que pode ser encontrado na aba "Standard" da barra de componentes. Esse componente possui o seguinte ícone.

Figura 11: *Ícone do componente Button.*

Temos que modificar as seguintes propriedades dos botões.

Name	=	ButtonAtualizar
Caption	=	Atualizar
Font/Style/fsBold	=	true

Com isso finalizamos a construção de nossa interface gráfica. Ela terá a seguinte aparência.

Figura 12: *Aparência final da interface gráfica.*

Agora vamos implementar o código que controla as barras de LEDs. Para isso precisamos criar um manipulador para o evento OnClick do botão "Atualizar". Para fazer isso podemos selecionar o componente Button com o texto "Atualizar", ir no **Object Inspector**, selecionar a aba Events e dar um duplo clique sobre a linha que está escrito OnClick. Uma forma mais fácil de fazer isso é apenas dar um duplo clique sobre o botão no Form e com isso o Delphi irá criar automaticamente um manipulador para o evento OnClick. O seguinte código será criado.

```
Procedure TformMain.ButtonAtualizarClick(Sender:
                                                TObject);
begin
end;
```

Dentro desse manipulador vamos implementar o código para controlar as barras. Quando o modo de operação selecionado for "Nível", então utilizaremos o método LEDBarLevel para atualizar a barra. Se o modo for "LEDs", então utilizaremos o método LEDBarStatus. A seguir a declaração desses dois métodos.

```
Procedure LEDBarSetLevel(bar,level : Integer);

Procedure LEDBarSetStatus(bar : Integer;
                                    status : Byte);
```

Esses dois métodos possuem como primeiro parâmetro a barra que queremos controlar. Se esse parâmetro tiver um valor "0" indica que queremos controlar a primeira barra de LEDs do *Módulo de Motores e Displays*. Se o valor for "1" então queremos controlar o segundo display.

O segundo parâmetro do método LEDBarLevel será o nível que queremos que ele apresente. Esse valor pode variar de 0 à 8. Utilizando esse método podemos utilizar as barras de LEDs para indicar um nível de forma simples. Se esse parâmetro for "0" então todos os LEDs da

barra ficarão apagados, se for “1” o primeiro LED a partir da base da barra será aceso, se for “2” os dois LEDs a partir da base serão acesos, se for “3” os três LEDs a partir da base e assim por diante.

Já o segundo parâmetro do método LEDBarStatus define quais LEDs da barra deverão ser ligados ou desligados. Ele é um valor de 8 bits sendo que cada bit desse valor irá definir o estado de um dos LEDs da barra. Cada LED da barra de LEDs está relacionado a 1 bit do valor de 8 bits desse parâmetro. Um bit de valor 1 significa que o LED relativo a ele será ligado e um bit 0 que ele será desligado.

Assim podemos ligar todos os LEDs da barra com um valor no segundo parâmetro igual a 255, que em binário é representado por 11111111b. Se o valor fosse 85, que é representado em binário por 10101010b teriamos os LEDs ligados alternadamente. Sabendo disso, vamos iniciar a implementar nosso manipulador do evento OnClick.

```
Procedure TformMain.ButtonAtualizarClick(Sender:
                                                TObject);
var
  // Armazena os LEDs que devem ser ligados
  leds : Byte;

begin
  // Verifica o modo
  if ComboBoxModo.ItemIndex = 0 then
  begin
    // Código para o modo “Nível”
  end
  else
  begin
    // Código para o modo “LEDs”
  end;
end;
```

Nesse trecho de código, que foi adicionado ao manipulador do evento OnClick, declaramos uma variável denominada “leds”, que irá armazenar quais LEDs serão ligados ou desligados, e em seguida verificamos, com uma estrutura “if”, qual modo de operação está selecionado. Vamos agora adicionar o código para quando o modo selecionado for “Nível”.

```
Procedure TformMain.ButtonAtualizarClick(Sender:
                                                TObject);
var
```

```
    // Armazena os LEDs que devem ser ligados
    leds : Byte;

begin
    // Verifica o modo
    if ComboBoxModo.ItemIndex = 0 then
    begin
      // Código para o modo "Nível"
      // Envia o número selecionado
      kit.LEDBarLevel( ComboBoxBarra.ItemIndex,
                       ComboBoxNivel.ItemIndex);
    end
    else
    begin
      // Código para o modo "LEDs"
    end;
end;
```

Essa linha de código que adicionamos utiliza o método LEDBarLevel e passa como parâmetro a barra selecionada e o nível que deve ser apresentado por essa barra.

Os ítens dos ComboBox de seleção da barra que será controlada e de nível da barra foram adicionados na ordem correta para que a posição deles dentro do ComboBox fosse equivalente ao valor que deve ser passado como parâmetro para representa-los. Dessa forma podemos utilizar diretamente como parâmetro a propriedade ItemIndex dos ComboBox.

A propriedade ItemIndex de um ComboBox indica a posição do ítem selecionado no momento. Por exemplo, no ComboBox de seleção da barra temos dois ítens, "Barra I" e "Barra II". Eles aparecem nessa ordem no ComboBox. Se o primeiro ítem estiver selcionado, isto é, o ítem "Barra I", então a propriedade ItemIndex será "0". Caso a segunda opção esteja selecionada, isto é, "Barra II", então a propriedade ItemIndex será "1".

Vamos agora implementar o código para quando o modo selecionado é "LEDs". O código será o seguinte.

```
    ...

    // Verifica o modo
    if ComboBoxModo.ItemIndex = 0 then
    begin
      // Código para o modo "Nível"
```

```
    // Envia o número selecionado
    kit.LEDBarLevel( ComboBoxBarra.ItemIndex,
                     ComboBoxNivel.ItemIndex);
end
else
begin
    // Código para o modo "LEDs"
    // Inicializa variável com "0"
    leds := 0;

    // Verifica quais LEDs devem ser ligados
     if CheckBoxLED0.Checked then
      leds := leds OR 1;

    if CheckBoxLED1.Checked then
      leds := leds OR 2;

    if CheckBoxLED2.Checked then
      leds := leds OR 4;

    if CheckBoxLED3.Checked then
      leds := leds OR 8;

    if CheckBoxLED4.Checked then
      leds := leds OR 16;

    if CheckBoxLED5.Checked then
      leds := leds OR 32;

    if CheckBoxLED6.Checked then
      leds := leds OR 64;

    if CheckBoxLED7.Checked then
      leds := leds OR 128;
```

```
        ...

      end;
    end;
```

Esse novo trecho de código inicializa a variável denominada “leds” com zero. Em seguida verifica quais LEDs devem ser ligados ou desligados e armazena essa informação na variável denominada “leds”.

Quando um LED tiver que ser ligado, temos que setar para “1” o bit correspondente a ele. Para fazer isso sem interferir nos outros bits do valor, utilizamos a operação lógica “OR”. Por exemplo, para setar o terceiro bit de uma valor para “1” fazemos uma operação “OR” desse valor com o valor 4. Utilizamos 4 porque esse valor em binário é representado por 00000100b, ou seja, possui setado para “1” apenas o terceiro bit, que é o que desejamos setar, sem interferir no valor dos outros bits. Para entender melhor a operação “OR” procure alguma documentação sobre operações lógicas, mais especificamente a operação lógica “OR”.

Para facilitar, a seguir apresentamos uma tabela com o valor que temos que utilizar com a operação “OR” para setar um determinado bit. A primeira coluna indica o bit que queremos setar e as outras 3 colunas indica o valor em decimal, hexadecimal e binário que devemos utilizar com a operação “OR”.

Bit	***Decimal***	***Hexadecimal***	***Binário***
0	1	0x01	00000001b
1	2	0x02	00000010b
2	4	0x04	00000100b
3	8	0x08	00001000b
4	16	0x10	00010000b
5	32	0x20	00100000b
6	64	0x40	01000000b
7	128	0x80	10000000b

No nosso código utilizamos os valores em decimal. Primeiro zeramos a variável denominada “leds” para garantir que todos os bits dela iniciassem com “0” e em seguida fomos verificando, um por um, qual CheckBox estava selecionado.

Se o primeiro CheckBox, que representa o primeiro LED, estiver selecionado fazemos uma “OR” da variável “leds” com o valor 1 e assim setamos o primeiro bit para “1”. Se o segundo CheckBox estiver selecionado fazemos uma “OR” do valor já existente na variável “leds” com o valor 2. Assim setamos o segundo bit para “1” sem interferir no valor dos outros bits.

Após verificar todos os CheckBox teremos na variável “leds” o valor correto para enviar como parâmetro. Então é necessário apenas chamar o método LEDBarStatus e passar esse valor como parâmetro. É o que vamos fazer a seguir.

```
        ...

        if CheckBoxLED6.Checked then
          leds := leds OR 64;

        if CheckBoxLED7.Checked then
          leds := leds OR 128;

        // Envia comando para o Kit
        kit.LEDBarStatus( ComboBoxBarra.ItemIndex,
                                                leds);
      end;
    end;
```

Com isso finalizamos o código e já podemos testar nosso programa. A aparência final do programa ficou assim.

Figura 13: *Programa finalizado.*

Se selecionarmos o modo "Nível" podemos escolher que nível a barra irá apresentar. No modo "LEDs" podemos definir um a um quais LEDs da barra serão ligados. A seguir algumas imagens dos displays ligados.

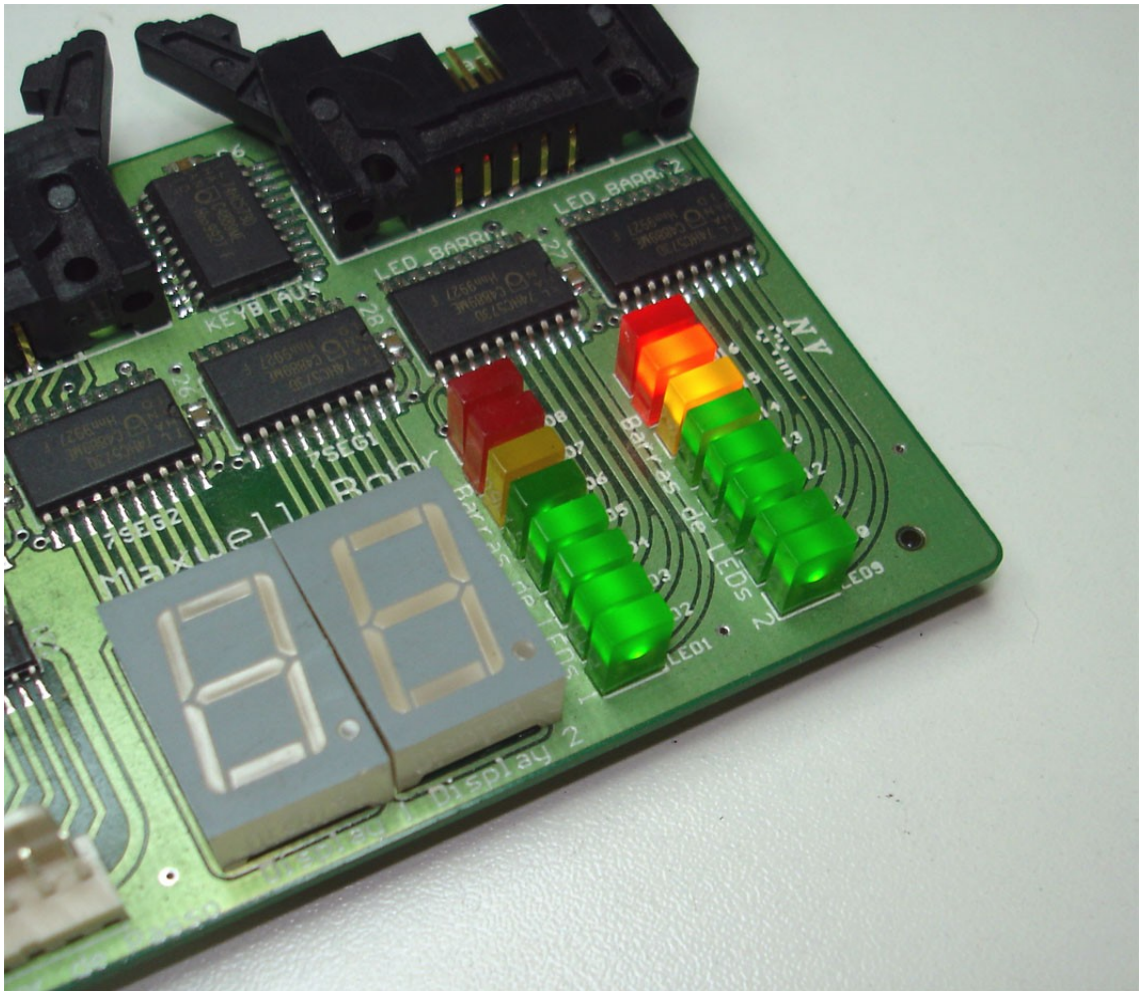

Figura 14: *Barras de LEDs com nível 4 e 7.*

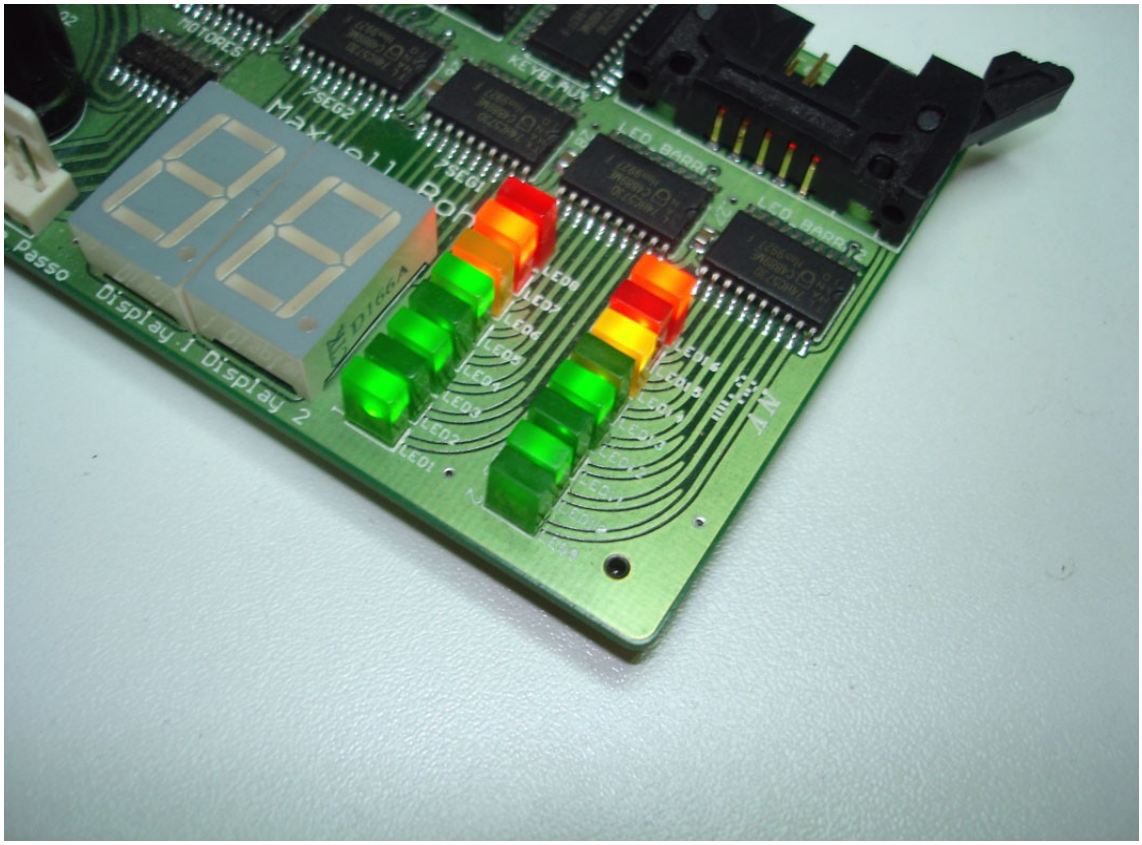

Figura 15: *Barras de LEDs com LEDs acesos alternadamente.*

Pronto, temos um programa que controla as barras de LEDs e nos oferece total controle sobre o estado dos LEDs.

4 – Dicas

Quando selecionamos o modo de operação de apresentação de níveis os CheckBox da nossa interface não tem função e quando selecionamos o modo de operação de controle dos LEDs é o ComboBox de seleção de níveis que não tem função, então seria interessante que os controles que não tenham função em um determinado modo de operação fiquem desabilitados.

Fazer isso é simples. Vamos utilizar o evento OnChange do ComboBox de seleção de modo. Criamos o manipulador para esse evento dando um duplo clique sobre o ComboBox de seleção de modo. O seguinte código será criado.

```
Procedure TformMain.ComboBoxModoChange(Sender:
                                                TObject);
begin
end;
```

Esse manipulador será chamado toda vez que o modo for alterado. Temos que colocar dentro dele as rotinas para desabilitar e reabilitar os componentes de acordo com o modo selecionado. Esse código ficará assim.

```
Procedure TformMain.ComboBoxModoChange(Sender:
                                              TObject);
begin
  // Verifica o modo
  if ComboBoxModo.ItemIndex = 0 then
  begin
    // Desabilita todos CheckBox
    CheckBoxLED0.Enabled := false;
    CheckBoxLED1.Enabled := false;
    CheckBoxLED2.Enabled := false;
    CheckBoxLED3.Enabled := false;
    CheckBoxLED4.Enabled := false;
    CheckBoxLED5.Enabled := false;
    CheckBoxLED6.Enabled := false;
    CheckBoxLED7.Enabled := false;

    // Habilita níveis
    ComboBoxNivel.Enabled := true;
  end
  else
  begin
    // Desabilita níveis
    ComboBoxNivel.Enabled := false;

    // Habilita todos CheckBox
    CheckBoxLED0.Enabled := true;
    CheckBoxLED1.Enabled := true;
    CheckBoxLED2.Enabled := true;
    CheckBoxLED3.Enabled := true;
    CheckBoxLED4.Enabled := true;
    CheckBoxLED5.Enabled := true;
```

```
        CheckBoxLED6.Enabled := true;
        CheckBoxLED7.Enabled := true;

    end;
end;
```

Com isso falta apenas um detalhe, a condição inicial dos componentes. Como nosso modo padrão quando o programa inicia é "Nível", então temos que deixar os CheckBox desabilitados na inicialização do programa. Fazemos isso modificando a propriedade Enabled de todos eles para false. Fazemos isso ainda no ambiente de desenvolvimento. Dessa forma não precisamos adicionar mais código ao nosso programa. Com isso finalizamos essa implementação. A seguir a imagem da aparência do programa nos dois modos. Observe que dependendo do modo os controles que não são utilizados ficam desabilitados.

Figura 16: *Programa no modo "Nível". Os CheckBox ficam desabilitados.*

Figura 17: *Programa no modo "LEDs". O ComboBox de seleção de nível fica desabilitado.*

5 – Conclusão

Nesse tutorial vimos como manipular as barras de LEDs do *Módulo de Motores e Displays*. Com o projeto que criamos foi possível entender o funcionamento do método LEDBarLevel e LEDBarStatus e o significado dos seus parâmetros. Assim já podemos utilizar as barras de LEDs para transmitir informações em qualquer projeto.