TUTORIAL
Motor DC

Autor: Tiago Lone
Nível: Básico
Criação: 15/12/2005
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 controlar, através de um programa, os motores DC do *Módulo de Motores e Displays*. Para auxiliar nessa tarefa criaremos um projeto que possibilita a seleção do motor DC que queremos controlar e o ajuste dos parâmetros para acionamento desse. Com isso serão vistos todos os detalhes de programação necessários para o controle desses dispositivos.

2 – Material

O programa desenvolvido nesse tutorial vai utilizar o *Módulo Principal* e o *Módulo de Motores e Displays* com apenas os motores DC conectados à placa principal desse módulo. Para a criação do programa será necessário o Borland Delphi 6. A seguir a imagem da montagem do KDR5000 necessária para esse tutorial.

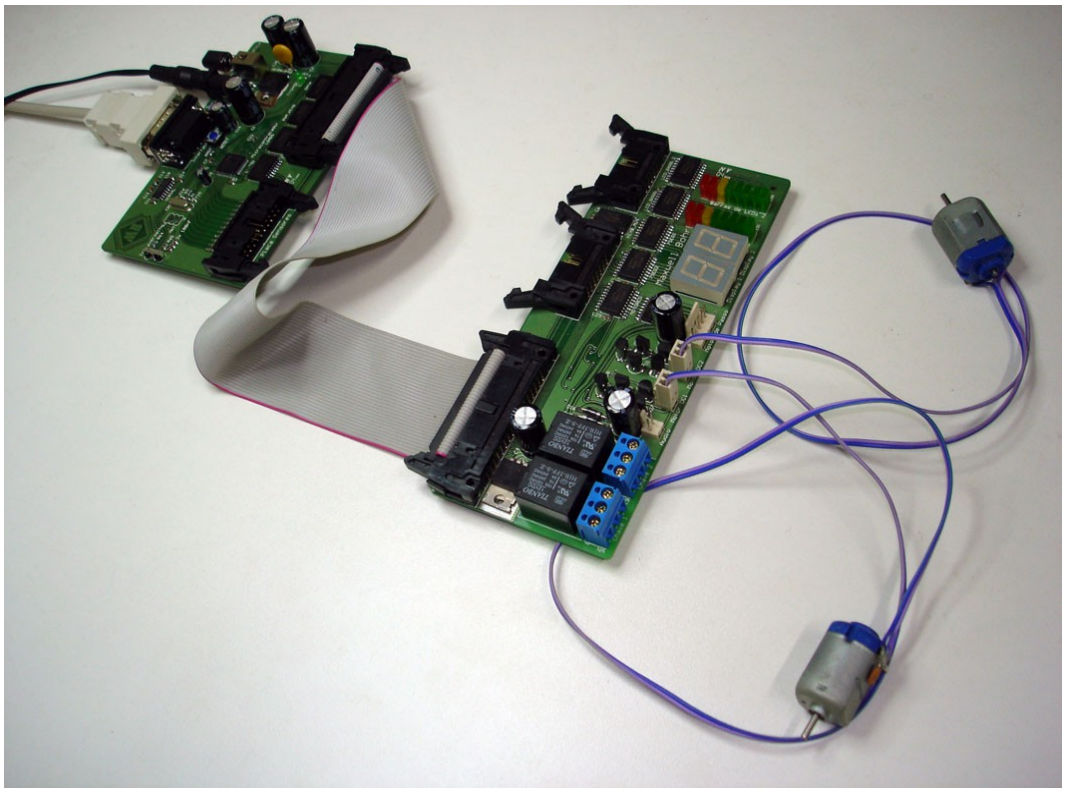

Figura 1: *Montagem do KDR5000 utilizada nesse tutorial.*

3 – Projeto

O projeto desse tutorial será um programa que controla os motores DC do *Módulo de*

Motores e Displays permitindo o ajuste do sentido e da velocidade da rotação. Esse programa terá a seguinte interface gráfica.

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

Nosso primeiro passo é criar essa interface gráfica. Utilizaremos o projeto criado no tutorial Base que já nos fornece algumas funcionalidades interessantes. Para isso copiamos o projeto daquele tutorial e em cima dele vamos adicionar alguns componentes gráficos extras.

A interface é simples, possuindo três componentes Labels, dois ComboBox, um ScrollBar e dois botões. Os ComboBox são utilizados para selecionar o motor e o sentido da rotação, o ScrollBar ajusta a velocidade do movimento e um botão para enviar o comando que acionar o motor e outro para desligá-lo. Todos esses componentes podem ser encontrados na aba "Standard" da barra de componentes.

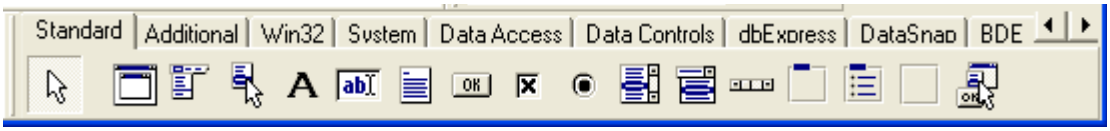

Figura 3: *Aba "Standard" da Barra de componente.*

Vamos adicionar um Label e um ComboBox para a seleção do motor. 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.*

Adicionamos um Label e modificamos as seguintes propriedades.

Name	=	LabelMotor
Caption	=	Motor:
Font/Style/fsBold	=	true

Em seguida adicionamos um componente ComboBox e modificamos as propriedades a seguir.

Name	=	ComboBoxMotor
Style	=	csDropDownList
Items.Strings	=	Motor DC I, Motor DC II
ItemIndex	=	0

Com isso teremos os componentes utilizados para a seleção do motor que queremos controlar. Nosso Form irá se parecer com o seguinte.

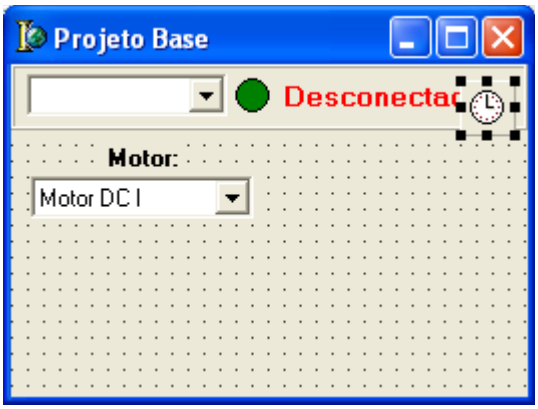

Figura 6: *Form após a inclusão dos componentes para seleção de motor.*

Agora adicionaremos os componentes para a seleção do sentido da rotação. Utilizaremos os mesmos componentes que utilizamos para a seleção do motor, um Label e um ComboBox. Adicionamos um Label e modificamos as seguintes propriedades.

Name	=	LabelSentido
Caption	=	Sentido:
Font/Style/fsBold	=	true

Em seguida adicionamos o ComboBox e modificamos as seguintes propriedades.

Name	=	ComboBoxSentido
Style	=	csDropDownList
Items.Strings	=	Anti-Horário, Horário
ItemIndex	=	0

Dessa forma já temos os componentes necessários para o ajuste do motor que queremos controlar e do sentido da rotação. Nosso Form se parecerá com o seguinte.

Figura 7: *Form com componentes para seleção de motor e direção da rotação.*

Agora adicionaremos os componentes para o ajuste da velocidade de rotação. Vamos utilizar um componente Label e um ScrollBar. O componente ScrollBar, assim como o Label, encontra-se na aba "Standard" da barra de componentes e possui o seguinte ícone.

Figura 8: *Ícone do componente ScrollBar.*

Adicionamos os dois componentes e modificamos as seguintes propriedades do Label.

Name	=	LabelVelocidade
Caption	=	Velocidade:
Font/Style/fsBold	=	true

E as seguintes propriedades do ScrollBar.

Name	=	ScrollBarVelocidade
Max	=	255

Position = 127

Assim teremos a seguinte aparência no nosso Form principal.

Figura 9: *Form após a adição de componentes para o ajuste de velocidade da rotação.*

Nesse ponto só nos falta adicionar os dois botões para iniciar e para parar o giro do motor. Para isso adicionamos dois componentes Button, que podem ser encontrados na aba "Standard" da barra de componentes. Esse componente possui o seguinte ícone.

Figura 10: *Ícone do componente Button.*

Temos que modificar as seguintes propriedades dos botões. Vamos modificar primeiro as propriedades do botão para início da rotação.

Name = ButtonIniciar
Caption = Iniciar
Font/Style/fsBold = true

Em seguida as propriedades do botão para parar a rotação.

Name = ButtonParar
Caption = Parar
Font/Style/fsBold = true

Um último detalhe que vamos modificar é a propriedade Caption do Form principal. Como copiamos o projeto do tutorial Base, essa propriedade possui o valor "Projeto Base". Vamos

modificar essa propriedade para “Motor DC”. Com isso finalizamos a construção de nossa interface gráfica. A seguir a imagem dessa interface finalizada.

Figura 11: *Interface final.*

Nosso objetivo agora é implementar o código para controle dos motores. Para isso precisamos criar um manipulador para o evento OnClick do botão de início e um para o botão de parada. Vamos criar primeiro o manipulador do botão para início da rotação. Para fazer isso podemos selecionar o componente Button, que possui o texto “Iniciar”, 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.ButtonEnviarClick(Sender: TObject);
begin
end;
```

Dentro desse manipulador vamos implementar o código para preparar os parâmetros da rotação e enviar o comando para o Kit. O método que vamos utilizar para acionar o motor DC é o DCMotorOn. Esse método possui a seguinte declaração.

```
Procedure DCMotorOn(motor, dir, speed : Integer);
```

Esse método possui três parâmetros, um indicando qual motor deve ser controlado, outro indicando a direção da rotação e por fim um que define a velocidade da rotação. Para indicar qual motor deve ser controlado podemos passar os valores “0” ou “1” no primeiro parâmetro, indicando o primeiro ou o segundo motor DC respectivamente. No parâmetro de direção passamos um valor igual a “0” para indicar um movimento anti-horário ou um valor “1” para indicar sentido horário. E por fim, no parâmetro de velocidade podemos passar valores na faixa de 0 a 255, sendo que quanto maior o valor, maior será a velocidade da rotação. Se o valor da velocidade for zero então o motor será desligado.

Vamos implementar o código para acionar um motor DC. O primeiro passo é preparar os parâmetros. A seguir o início do código do manipulador do botão “Iniciar”.

```
Procedure TformMain.ButtonEnviarClick(Sender:
                                          Tobject);
var
  // Armazena o motor
  motor : Integer;

  // Armazena sentido
  sentido : Integer;

  // Armazena velocidade
  velocidade : Integer;

begin
  // Recupera qual motor deve ser controlado
  if ComboBoxMotor.ItemIndex = 0 then
    motor := 0
  else
    motor := 1;

  ...
end;
```

Nesse trecho de código criamos três variáveis do tipo Integer para armazenar os parâmetros de motor, sentido e velocidade. Em seguida verificamos, através da propriedade ItemIndex do ComboBox de seleção do motor, qual é o motor que foi selecionado e setamos a variável "motor" com o valor correspondente a ele.

Em seguida verificamos a seleção do sentido da rotação e setamos a variável "sentido" de acordo. O código fica o seguinte.

```
Procedure TformMain.ButtonEnviarClick(Sender:
                                          Tobject);
var
  // Armazena o motor
  motor : Integer;
```

```
    // Armazena o sentido
    sentido : Integer;

    // Armazena velocidade
    velocidade : Integer;

begin
    // Recupera qual motor deve ser controlado
    if ComboBoxMotor.ItemIndex = 0 then
      motor := 0
    else
      motor := 1;

    // Recupera o sentido
    if ComboBoxSentido.ItemIndex = 0 then
      sentido := 0
    else
      sentido := 1;

    ...
end;
```

Agora vamos recuperar o último parâmetro, que é a velocidade. O código que faz isso é o seguinte.

```
    ...

    // Recupera o sentido
    if ComboBoxSentido.ItemIndex = 0 then
      sentido := 0
    else
      sentido := 1;

    // Recupera a velocidade
    velocidade := ScrollBarVelocidade.Position;
```

```
  ...
end;
```

O parâmetro de velocidade será o valor da propriedade Position do ScrollBar. Ele sempre estará na faixa de 0 a 255 porque definimos que a propriedade Max, que define o valor máximo de retorno da posição desse componente, fosse 255 e por padrão o valor mínimo já é 0.

Com esse código temos nosso parâmetro de velocidade armazenado na variável de nome "velocidade". Com isso temos todos nossos parâmetros armazenados em variáveis. Falta apenas chamar o método DCMotorOn e passar essas variáveis como parâmetro. É o que vamos fazer a seguir.

```
  ...

  // Recupera a velocidade
  velocidade := ScrollBarVelocidade.Position;

  // Envia o comando para ligar o motor
  Kit.DCMotorOn(motor, sentido, velocidade);

  ...
end;
```

Essa última linha que adicionamos apenas chama o método DCMotorOn com os parâmetros que armazenamos anteriormente. Com isso já podemos fazer um teste de controle dos motores DC do *Módulo de Motores e Displays*. Para isso, vamos no menu **Run – Run** ou pressionamos F9. Se não houver nenhum erro o programa será compilado e executado. Com um Kit conectado em alguma porta serial podemos testar se o programa está funcionando. Selecione a porta serial correta, modifique alguns parâmetros e pressione o botão "Enviar". O motor selecionado deverá se movimentar. Teste algumas vezes com parâmetros diferentes.

Vamos implementar agora o código do manipulador de eventos do botão "Parar". Como vimos anteriormente, para parar um motor temos que passar como parâmetro de velocidade o valor "0". Então vamos criar o manipulador do evento OnClick do botão "Parar" dando dois cliques sobre ele e depois inserir o seguinte código.

```
Procedure TformMain.ButtonPararClick(Sender:
                                          TObject);
begin
  // Desliga motor
```

```
    Kit.DCMotorOn(ComboBoxMotor.ItemIndex , 0, 0);
end;
```

Pronto, temos um programa que controla os motores DC e ajusta todos os parâmetros da rotação. A aparência final do programa ficou assim.

Figura 12: *Aparência final do programa.*

Podemos selecionar nessa interface o motor controlado, o sentido e a velocidade da rotação. Em seguida é preciso apenas pressionar o botão enviar e o motor irá se movimentar. Para interromper o movimento do motor podemos pressionar o botão "Parar" e o movimento será interrompido.

4 – Dicas

Uma funcionalidade interessante que podemos implementar é que ao modificar a velocidade do motor através do ScrollBar essa fosse refletida automaticamente no motor sem a necessidade de pressionar o botão "Iniciar".

Para isso vamos utilizar o evento do ScrollBar denominado OnChange, que é executado toda vez que há uma mudança na posição do ScrollBar. Capturando esse evento podemos tomar alguma atitude perante qualquer mudança na velocidade. Para criar o manipulador para esse evento, dê um duplo clique sobre o componente ScrollBar responsável pelo ajuste da velocidade. O seguinte código será criado.

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

Para atualizar a velocidade do motor temos que utilizar o método DCMotorOn passando como parâmetro a nova velocidade juntamente com os parâmetros antigos. Isso pode ser feito de

uma maneira bem simples. Como nosso método ButtonEnviarClick, que é o manipulador do evento OnClick do botão “Enviar”, lê todos os parâmetros selecionados atualmente na interface e chama o método DCMotorOn com esses parâmetros, então podemos apenas chamar esse método toda vez que houver uma mudança no ajuste de velocidade e com isso essa será atualizada imediatamente no motor. Precisamos de apenas uma linha de código para fazer isso e é o que mostramos a seguir.

```
Procedure TformMain.ScrollBarVelocidadeChange
                                    (Sender: TObject);
begin
  // Atualiza a velocidade do motor imediatamente
  // após essa ser modificada no ScrollBar.
  ButtonEnviarClick(Sender);
end;
```

Teste selecionar um motor e modificar a velocidade. Perceba que agora a velocidade da rotação do motor varia conforme o ScrollBar é modificado, sem a necessidade de pressionar o botão “Enviar”. Para parar o movimento pressione o botão “Parar”.

Uma outra dica interessante é desligar os motores toda vez que o programa for fechado. Para fazer isso podemos implementar um código para desligar os motores no evento OnDestroy do Form principal, que é executado quando o programa fecha. Já criamos um manipulador para esse evento no tutorial Base. Como copiamos aquele projeto, o manipulador para esse evento já existe nesse código fonte, e seu código está a seguir.

```
Procedure TFormMain.FormDestroy(Sender: TObject);
begin
  kit.CloseCommunication;
end;
```

Nesse manipulador adicionaremos o código para desligar os motores. Isso é feito com o seguinte código.

```
Procedure TFormMain.FormDestroy(Sender: TObject);
begin
  // Desliga o primeiro motor
  Kit.DCMotorOn(0, 0, 0);

  // Desliga o segundo motor
  Kit.DCMotorOn(1, 0, 0);
```

```
    // Fecha a comunicação
    kit.CloseCommunication;
end;
```

Agora, quando fechamos nosso programa, os motores irão interromper seu movimento imediatamente.

5 – Conclusão

Nesse tutorial vimos como controlar os motores DC do *Módulo de Motores e Displays*. Com o projeto que criamos foi possível entender o funcionamento do método DCMotorOn e o significado de todos os seus parâmetros. Com isso já é possível explorar ao máximo tudo que os motores DC do Kit Didático de Robótica podem oferecer.