TUTORIAL
Display de Cristal Líquido

Autor: Tiago Lone
Nível: Básico
Criação: 09/03/2006
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 iremos abordar o controle do display de cristal líquido (LCD) que pode ser conectado ao *Módulo de Motores e Displays* ou ao *Módulo de Entradas, Saídas e Servo Motores*.

Esse display possui uma estrutura formada por 4 linhas e 20 colunas em que podem ser apresentados caracteres, o que permite a apresentação de textos relativamente grandes. Outro recurso interessante que esse display possui é uma luz de fundo que permite a visualização das informações mesmo em ambientes escuros.

Veremos, nesse tutorial, como controlar esses recursos a partir de um programa no computador. Para auxiliar nesse estudo, será desenvolvido um programa que permite controlar todos os recursos do LCD.

2 – Material

O programa desenvolvido nesse tutorial vai precisar do *Módulo Principal* e o *Módulo de Motores e Displays* ou do *Módulo de Entradas, Saídas e Servo Motores* com o display de cristal líquido conectado. 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 KDR5000 com o display de cristal líquido.

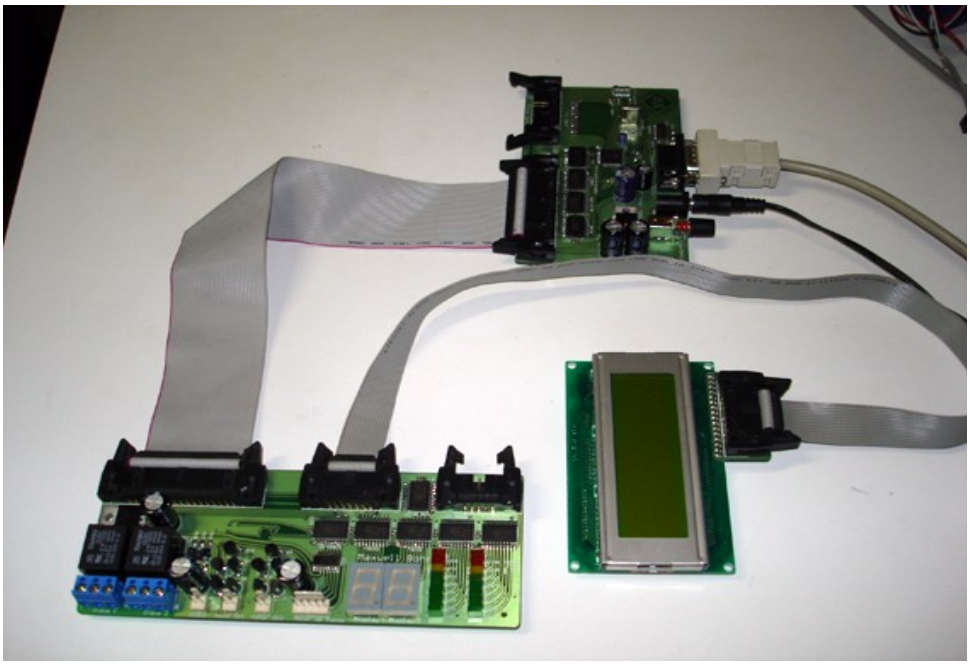

Figura 1: *Montagem do KDR5000 utilizada nesse tutorial.*

3 – Projeto

Nesse tutorial vamos criar um programa que permite total controle sobre o display de cristal líquido. Ele irá permitir o envio de textos, controle de visualização e posicionamento do cursor, acionamento da luz de fundo e limpeza o conteúdo do LCD. A interface gráfica desse programa será a seguinte.

Figura 2: *Interface gráfica 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 extras. 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 "Display de Cristal Líquido". Com isso já podemos começar a adicionar os componentes gráficos ao Form.

Para modificar a interface gráfica criada no tutorial Base para a interface do programa desse tutorial vamos precisar adicionar alguns componentes ao Form principal. Adicionaremos três Labels, dois ComboBox, dois CheckBox, um Edit e dois Button. Os Labels são utilizados apenas para identificar a funcionalidade dos ComboBox e do Edit. Os ComboBox são utilizados para selecionar a linha e coluna do cursor no display. O Edit é utilizado para que possamos escrever o texto que vamos enviar ao LCD. Os CheckBox controlam visualização do cursor e o acionamento da luz de fundo do display. Por fim temos um botão para enviar o texto do Edit para o display e um botão para limpar o conteúdo no LCD.

Todos esses componentes podem ser encontrados na aba "Standard" da barra de componentes.

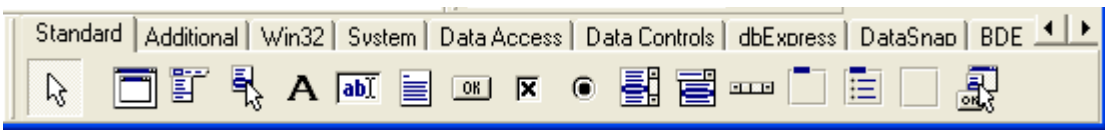

Figura 3: *Aba "Standard" da Barra de componente.*

Vamos adicionar dois Labels e dois ComboBox para o controle da posição do cursor no display. 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 Label, modificamos as seguintes propriedades do primeiro.

Name	=	LabelLinha
Caption	=	Linha:
Font/Style/fsBold	=	true

E as seguintes propriedades do segundo.

Name	=	LabelColuna
Caption	=	Coluna:
Font/Style/fsBold	=	true

Em seguida vamos modificar as propriedades dos ComboBox. Modificamos as seguintes propriedades do primeiro ComboBox.

Name	=	ComboBoxLinha
Style	=	csDropDownList
Items.Strings	=	0, 1, 2, 3
ItemIndex	=	0

E as seguintes propriedades do segundo ComboBox.

Name	=	ComboBoxColuna
Style	=	csDropDownList

Items.Strings	=	0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19
ItemIndex	=	0

Com isso o Form terá a seguinte aparência.

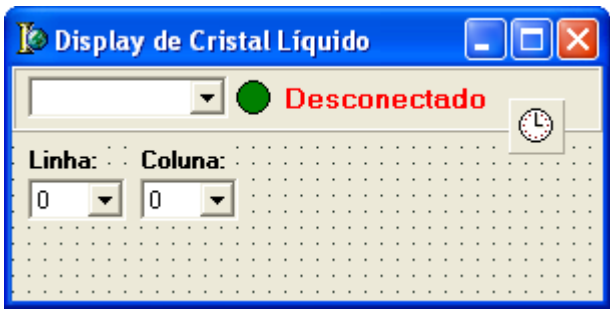

Figura 6: *Aparência do Form após a inclusão dos controles para o controle da linha e coluna do cursor no displays de cristal líquido.*

Agora vamos adicionar dois componentes CheckBox, um para controle da visualização do cursor e outro para acionamento da luz de fundo do LCD. O ícone do componente CheckBox é o seguinte.

Figura 7: *Ícone do componente CheckBox.*

Temos que modificar as seguintes propriedades do primeiro CheckBox que for adicionado.

Name	=	CheckBoxLuzFundo
Caption	=	Luz de Fundo
Font/Style/fsBold	=	true

E as seguintes propriedades do segundo CheckBox.

Name	=	CheckBoxCursor
Caption	=	Cursor
Font/Style/fsBold	=	true

Dessa forma teremos a seguinte interface gráfica até o momento.

Figura 8: *Interface gráfica após a adição de dois CheckBox.*

Agora temos que adicionar mais um Label e um Edit. O ícone do componente Edit é o seguinte.

Figura 9: *Ícone do componente Edit.*

Assim que adicionarmos esses componente temos que modificar as seguintes propriedades do componente Label.

Name	=	LabelTexto
Caption	=	Texto:
Font/Style/fsBold	=	true

E as seguintes propriedades do componente Edit.

Name	=	EditTexto
Text	=	(deixar em branco)

Com isso teremos a seguinte interface gráfica.

Figura 10: *Interface gráfica após adição de um Label e um Edit.*

Por fim vamos adicionar dois componentes Button. Um para enviar o texto do Edit para o display e outro para limpar o conteúdo do LCD. O ícone do componente Button é o seguinte.

Figura 11: *Ícone do componente Button.*

Temos que modificar as seguintes propriedades do primeiro Button.

Name	=	ButtonEnviar
Caption	=	Enviar
Font/Style/fsBold	=	true

E as seguintes do segundo.

Name	=	ButtonLimpar
Caption	=	Limpar
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 irá controlar o LCD. A primeira funcionalidade que vamos implementar é o envio de textos para o display. Para isso precisamos criar um manipulador para o evento OnClick do botão "Enviar". Para fazer isso podemos selecionar o componente Button com o texto "Enviar", 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 que enviará o texto ao LCD. Para isso utilizaremos o método LCDWriteText. A seguir a declaração desse método.

```
Procedure LCDWriteText(txt : String);
```

Esse método possui um parâmetro que é a String de texto que será escrita no display na posição atual do cursor. O código do manipulador do evento OnClick do botão "Enviar" ficará da seguinte maneira.

```
Procedure TFormMain.ButtonEnviarClick(Sender:
                                                TObject);
begin
  // Envia texto ao LCD
  kit.LCDWriteText(EditTexto.Text);
end;
```

Como podemos ver, passamos diretamente a propriedade Text do Edit como parâmetro do método LCDWriteText e dessa forma o texto do Edit é enviado ao display de cristal líquido. Nesse momento já podemos testar o programa. 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. Escreva algum texto no edit e pressione o botão "Enviar". O texto escrito no Edit deverá ser apresentado no display.

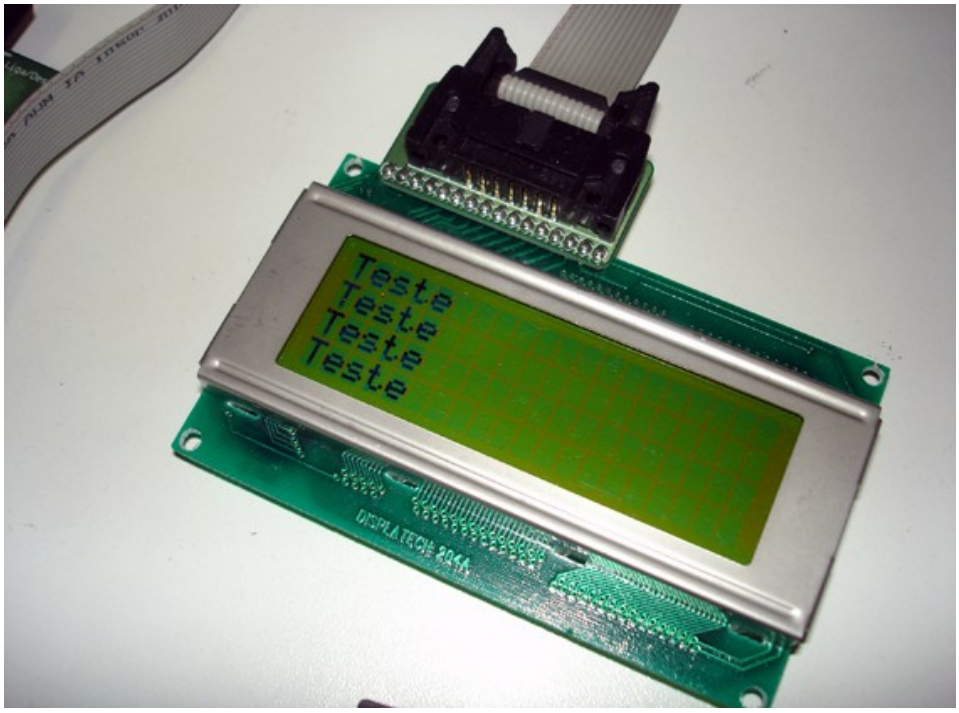

Figura 13: *Display de Cristal Líquido apresentando texto.*

Vamos agora implementar o código que limpa o conteúdo da tela do display. Para isso precisamos criar um manipulador para o evento OnClick do botão "Limpar". Para fazer isso podemos selecionar o componente Button com o texto "Limpar", 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.ButtonLimparClick(Sender:
                                              TObject);
begin
end;
```

Dentro desse manipulador vamos implementar o código que limpa o conteúdo do display. Utilizaremos o método LCDClear. A seguir a declaração desse método.

```
Procedure LCDClear;
```

Esse método não possui parâmetros nem retorno e assim que executado irá limpar todo o conteúdo do display de cristal líquido. O código do manipulador do evento de clique do botão "Limpar" ficará como a seguir.

```
Procedure TFormMain.ButtonLimparClick(Sender:
```

```
                                                    TObject);
begin
  // Limpa LCD
  kit.LCDClear;
end;
```

Para testar essa nova funcionalidade implementada, compile e execute o programa como foi explicado anteriormente, envie algum texto ao display e em seguida pressione o botão “Limpar”. O texto presente no display deverá ser apagado.

Vamos agora implementar o código para o controle da posição do cursor do display. Esse controle é feito através de dois ComboBox. Temos que criar manipuladores para os eventos de mudança de valor nos dois ComboBox. A forma mais fácil de fazer isso é dar um duplo clique sobre cada um desses componentes gráficos. Vamos fazer isso passo a passo, dando um duplo clique sobre o ComboBox de seleção de linha será criado o seguinte manipulador.

```
Procedure TFormMain.ComboBoxLinhaChange(Sender:
                                                    TObject);
begin
end;
```

Dentro desse manipulador temos que inserir o código que posiciona o cursor. Para isso vamos fazer uso do método LCDGotoXY, encontrado na biblioteca de controle do Kit. Esse método possui a seguinte declaração.

```
Procedure LCDGotoXY(lin, col : Integer);
```

Como podemos ver, esse método possui dois parâmetros do tipo Integer. O primeiro indica a linha onde o cursor será posicionado e pode variar de 0 à 3. O segundo indica a coluna onde o cursor deverá ser posicionado e pode variar de 0 à 19.

O código do manipulador do evento OnChange do ComboBox de seleção de linha deverá ficar da seguinte maneira.

```
Procedure TFormMain.ComboBoxLinhaChange(Sender:
                                                    TObject);
begin
  // Posiciona cursor
  kit.LCDGotoXY(ComboBoxLinha.ItemIndex,
                ComboBoxColuna.ItemIndex);
```

```
end;
```

Os itens dos dois ComboBox foram inseridos na ordem de forma que a posição deles corresponda ao valor apresentado, isto é, o valor 0 está no posição 0, o valor 1 na posição 1 e assim por diante. Por isso, passamos como parâmetro de linha e coluna, do método LCDGotoXY, a posição do ítem selecionado no ComboBox de seleção de linha e de coluna.

Temos que inserir esse código, sem nenhuma alteração, no manipulador de eventos do ComboBox de seleção de coluna. Dessa forma toda vez que houver uma alteração tanto na linha quanto na coluna selecionadas, esse código será executado e a posição do cursor será atualizada.

Para criar o manipulador do evento OnChange do ComboBox de seleção de coluna damos um duplo clique sobre ele fazendo com que o seguinte código seja gerado.

```
Procedure TFormMain.ComboBoxColunaChange(Sender:
                                              TObject);
begin
end;
```

Em seguida, inserimos o código que vimos anteriormente para atualizar a posição do cursor toda vez que a seleção de coluna mude. O código ficará assim.

```
Procedure TFormMain.ComboBoxColunaChange(Sender:
                                              TObject);
begin
  // Posiciona cursor
  kit.LCDGotoXY(ComboBoxLinha.ItemIndex,
                ComboBoxColuna.ItemIndex);
end;
```

Pronto, a funcionalidade de posicionamento de cursor está pronta e já pode ser testada. Para isso compile e execute o programa, modifique a linha e coluna e envie um texto ao display. Observe que o texto irá aparecer na posição selecionada, isto é, na linha e coluna que foram selecionadas nos ComboBox.

Já temos boa parte das funcionalidades do programa implementadas, faltando apenas implementar a funcionalidade dos dois CheckBox. Como acabamos de implementar o código para posicionamento do cursor, vamos implementar primeiro o código do CheckBox que habilita e desabilita a visualização do cursor. Dessa maneira poderemos visualizar melhor o posicionamento do cursor pois poderemos deixá-lo visível.

Temos que criar um manipulador para o evento OnClick do CheckBox que controla a visualização do cursor. Para isso damos um duplo clique sobre ele e o seguinte código será criado.

```
Procedure TFormMain.CheckBoxCursorClick(Sender:
                                        TObject);
begin
end;
```

Dentro desse manipulador vamos utilizar o método LCDCursorOn que permite habilitar e desabilitar a visualização do cursor do LCD. A declaração desse método é a seguinte.

```
Procedure LCDCursorOn(tf : Boolean);
```

Esse método possui um parâmetro do tipo Boolean que indica se a visualização do cursor de ser ligada ou desligada. O código do manipulador do evento de clique que acabamos de criar ficará como a seguir.

```
Procedure TFormMain.CheckBoxCursorClick(Sender:
                                        TObject);
begin
  // Liga/Desliga cursor
  kit.LCDCursorOn(CheckBoxCursor.Checked);
end;
```

Nesse código passamos como parâmetro para o método LCDCursorOn o valor da propriedade “Checked” do componente CheckBox que controla a visualização do cursor. Essa propriedade é um valor Boolean e se o CheckBox estiver marcado então ela será igual a “verdadeiro”, caso contrário ela será igual a “falso”.

Para finalizar nossa implementação vamos criar um um manipulador para o evento OnClick do CheckBox que controla o acionamento da luz de fundo do display. Para isso damos um duplo clique sobre esse componente e o seguinte código será criado.

```
Procedure TFormMain.CheckBoxLuzFundoClick(Sender:
                                          TObject);
begin
end;
```

Utilizaremos o método LCDBacklightOn para controlar o acionamento da luz de fundo do display. A declaração desse método é a seguinte.

```
Procedure LCDBacklightOn(tf : Boolean);
```

Assim como o método anterior, esse possui um único parâmetro do tipo Boolean que indica se a luz de fundo deve ser ligada ou desligada. O código do manipulador do evento de clique sobre o CheckBox de acionamento da luz de fundo ficará assim.

```
Procedure TFormMain.CheckBoxLuzFundoClick(Sender:
                                          TObject);
begin
  // Liga/Desliga luz de fundo
  kit.LCDBacklightOn(CheckBoxLuzFundo.Checked);
end;
```

Passamos como parâmetro para o método LCDBacklightOn a propriedade “Checked” do CheckBox de controle do acionamento da luz de fundo. Essa lógica já foi utilizada no CheckBox de controle da visualização do cursor.

Com isso finalizamos a implementação de todas as funcionalidades. A aparência final do programa ficou assim.

Figura 14: *Programa finalizado.*

Abaixo a imagem do LCD com visualização do cursor acionada, luz de fundo ligada e uma mensagem de texto “Teste” na linha 2 e coluna 7.

Figura 15: *LCD com visualização do cursor acionada, luz de fundo ligada e uma mensagem de texto "Teste" na linha 2 e coluna 7.*

Pronto, temos um programa que controla todas as funcionalidades do display de cristal líquido.

4 – Conclusão

Nesse tutorial vimos como controlar o display de cristal líquido que pode ser conectado tanto no *Módulo de Motores e Displays* quanto no *Módulo de Entradas, Saídas e Servo Motores*. Com o projeto que criamos foi possível entender o funcionamento dos métodos LCDWriteText, LCDClear, LCDGotoXY, LCDCursorOn e LCDBacklightOn. Assim já podemos utilizar o display de cristal líquido em qualquer projeto.