TUTORIAL
LEITURA BUFFERIZADA DE SENSORES

Autor: Tiago Lone
Nível: Intermediário
Criação: 27/03/2006
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 vamos aprender como podemos fazer com que o Kit faça leituras de um canal de sensor em uma taxa que nós definimos. Em uma aquisição dessa forma, não existe a necessidade de enviar um comando para cada leitura que queiramos fazer e também não temos que nos preocupar com o intervalo entre as aquisições, pois tudo será feito pelo Kit.

Para demonstrar isso na prática, vamos criar um programa que permite que façamos a leitura de um sensor utilizando essa técnica.

2 – Material

Nesse tutorial vamos utilizar o *Módulo de Sensores* ou o *Módulo de Sensores Genérico* do KDR5000 ou as portas de sensores do MEC1000, com alguns sensores conectados a eles. Para a criação do programa será necessário o Borland Delphi 6. A seguir a imagem do *Módulo de Sensores* com vários sensores conectados a ele.

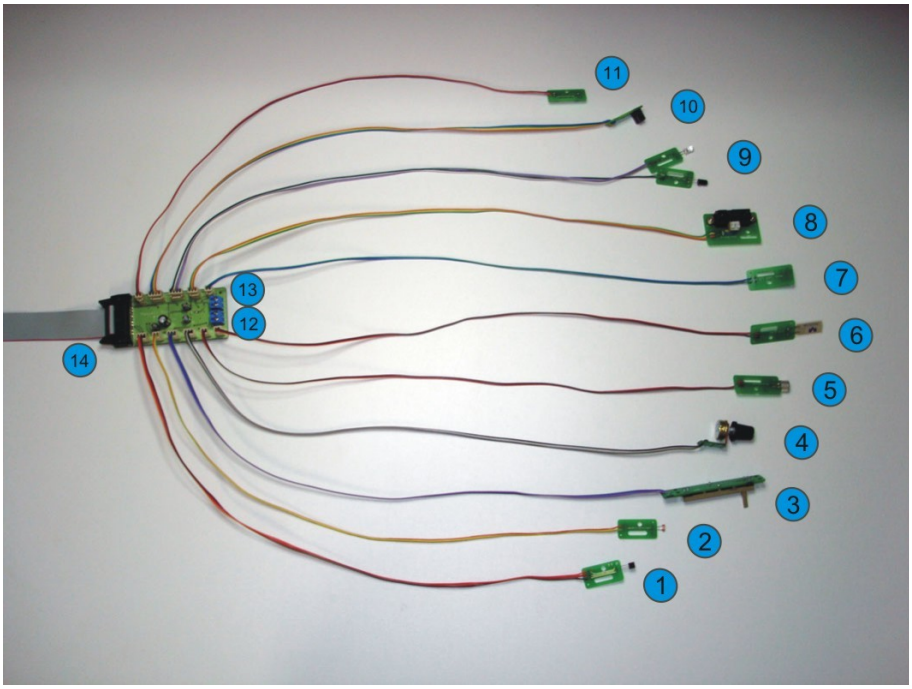

Figura 1: *Módulo de Sensores. 1-Sensor de Temperatura, 2-Sensor de Luminosidade, 3-Potenciômetro Linear, 4-Potenciômetro Angular, 5-Microfones, 6-Sensor de Vibração, 7-Sensor de Peso, 8-Sensor de Distância, 9-Par Óptico I, 10-Par Óptico II, 11-Chave Magnética, 12-Entrada auxiliar I, 13-Entrada auxiliar II, 14-Conector flat de 20 vias para conexão com o Módulo Principal.*

3 – Projeto

Nesse tutorial vamos criar um programa que utiliza os recursos de leitura bufferizada do Kit. Após o início desse tipo de aquisição, as leituras são feitas pelo Kit, de forma independente do computador. O Kit ainda poderá continuar recebendo outros comandos enquanto faz a aquisição.

Esse recurso funciona da seguinte maneira. Enviamos ao Kit um comando indicando que queremos que ele faça a aquisição de um sensor e para isso indicamos o sensor que queremos que ele leia e o intervalo em milisegundos entre essas leituras.

Após o envio desse comando o Kit iniciará a aquisição e irá armazenar esses valores lidos internamente, em uma região de memória reservada para isso. Essa região de memória é utilizada para armazenar as leituras apenas temporariamente, até que esses valores sejam lidos pelo computador. Esse tipo de região de memória temporária é normalmente denominado como "buffer" e por isso chamamos esse tipo de aquisição de "bufferizada".

O "buffer" do Kit suporta armazenar 90 leituras. Antes que esse "buffer" esteja cheio, devemos enviar um comando para que esse seja lido, o que fará com que o Kit envie todas as leituras já realizadas ao computador. Em seguida o "buffer" será esvaziado e a leitura continuará normalmente. Se o buffer encher antes que seja feita a leitura dos valores armazenados, então os valores mais antigos começarão a ser descartados para dar lugar às leituras mais recentes. Esse tipo de aquisição é mais adequada do que a leitura simples de um sensor nas seguintes situações.

- **O intervalo entre as aquisições é pequeno**: Quando queremos fazer uma aquisição com um intervalo curto entre as leituras, isto é, uma aquisição com uma taxa de leituras alta, o ideal é utilizar o modo de leituras bufferizado. Sem o uso desse recurso, não é possível obter muitas leituras por segundo, pois o tempo gasto com o envio do comando e a resposta do Kit, em uma leitura simples, inviabiliza uma aquisição rápida. A leitura simples deve ser utilizada quando o intervalo entre leituras é maior do que 50ms e mesmo assim esse intervalo pode sofrer influência de vários fatores pois dependerá totalmente do computador. Já a leitura bufferizada pode ser utilizada para intervalos de até 1ms e independe do computador. No entanto é importante lembrar que o número de aquisições que o "buffer" suporta é limitado, por isso temos que nos preocupar para que esse não fique cheio senão vamos perder leituras.

- **É necessário precisão entre os intervalos**: Na leitura simples de um sensor, o intervalo entre mais de uma leitura depende totalmente do computador, o que gera uma certa incerteza pois não há garantia nenhuma de que esse intervalo será preciso. Um computador lerdo, a execução de um outro programa ou até mesmo o Sistema Operacional pode influenciar muito no intervalo entre as aquisições. Já na aquisição bufferizada, quem controla o tempo entre as leituras é o próprio Kit, proporcionando uma precisão muito maior, pois ele irá sofrer a influência de muito menos fatores.

- **Aquisição sem que o Kit esteja conectado ao computador**: Como na aquisição bufferizada a leitura dos dados é feita de forma independente do computador, então temos a opção de desconectar o Kit da porta serial e mesmo assim a aquisição continuará sendo realizada. Dessa forma podemos programar um intervalo de alguns minutos entre as leituras

e assim monitorar por horas uma determinada variável e em seguida ler esses dados a partir de um computador. É importante lembrar que o Kit não deve ser desligado em nenhum momento para que essa aquisição não seja interrompida e perdida.

Após essa explicação sobre o que é a leitura bufferizada de sensores e de quando ela deve ser utilizada, vamos iniciar o desenvolvimento de um programa que utiliza esse tipo de leitura. Esse programa irá permitir que seja selecionado um canal de sensor e o intervalo entre leituras. Com esses parâmetros definidos podemos iniciar uma aquisição bufferizada. Além disso teremos como parar a aquisição, limpar e ler o buffer de leituras e limpar o gráfico que está sendo apresentado. A interface desse programa é mostrado na figura a seguir.

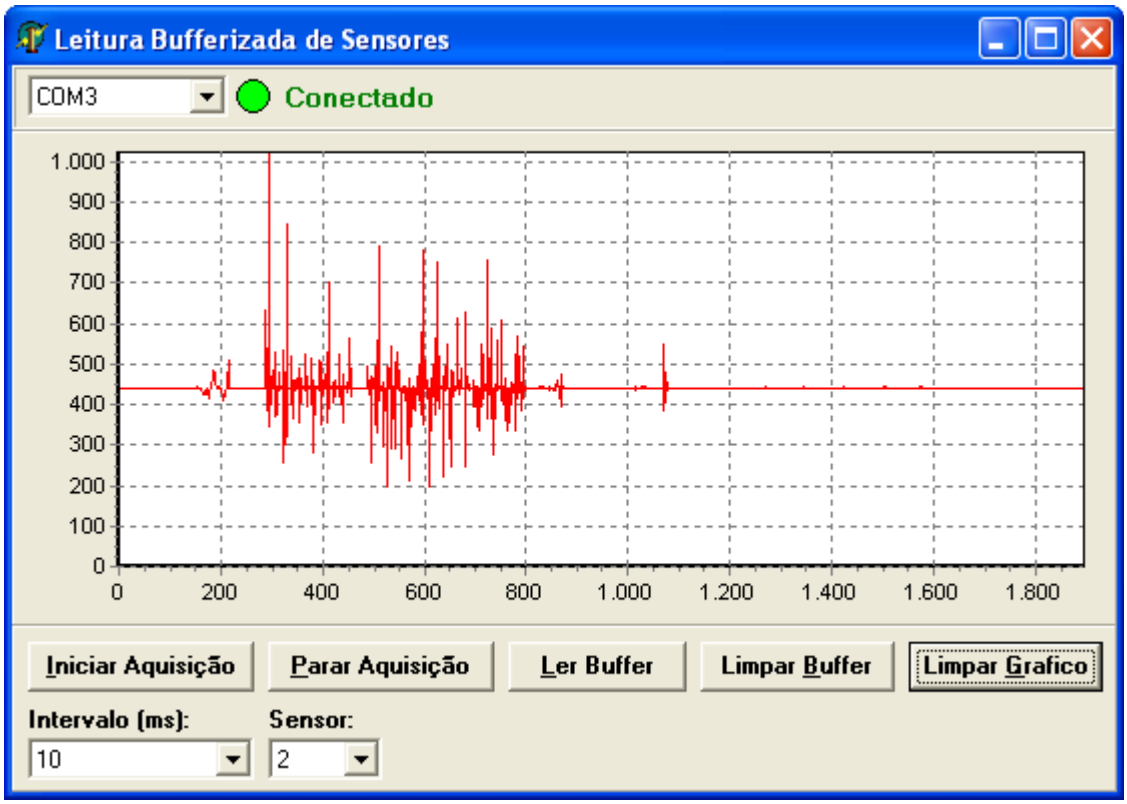

Figura 2: *Aparência do programa que será criado nesse tutorial.*

O primeiro passo é criar a interface gráfica do programa. Vamos utilizar o projeto criado no tutorial Base e em cima dele adicionar componentes e código. Para isso copiamos o projeto daquele tutorial e vamos adicionar alguns componentes gráficos extras. Vamos apenas modificar um detalhe, a propriedade Caption do Form principal ainda possui a mensagem atribuída a ela no projeto Base, então vamos modificar essa propriedade de "Projeto Base" para "Leitura Bufferizada de Sensores".

A interface do programa é composta por um componente Chart, um Panel, dois Label, dois ComboBox e cinco Button. O Chart é utilizado para apresentar, em forma de gráfco, as leituras feitas. O Panel irá agrupar os outros componentes na parte inferior do programa. Um dos Label indica que um ComboBox será utilizado para selecionar o sensor que queremos ler. O outro Label é utilizado para indicar que o ComboBox abaixo dele é utilizado para selecionar o intervalo entre as leituras. Por fim, os botões são utilizados para as várias operações que o programa irá realizar, que

são: iniciar a aquisição, finalizar a aquisição, ler o buffer de leituras, limpar o buffer de leituras e limpar o gráfico apresentado.

O Chart pode ser encontrado na aba de componentes “Additional”. Todos os outros podem ser encontrados na aba de componentes “Standard”. A seguir uma imagem da aba “Additional”.

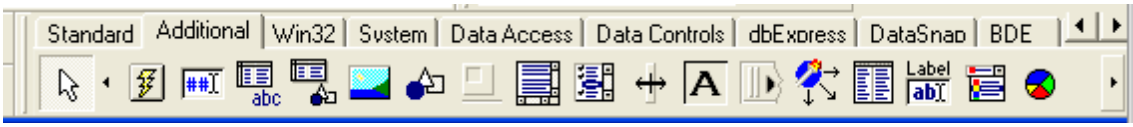

Figura 3: *Aba "Additional" da barra de componentes.*

E na próxima imagem a aba “Standard”.

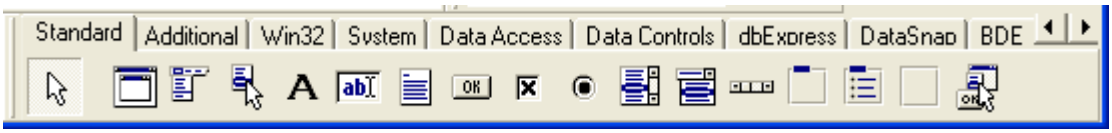

Figura 4: *Aba "Standard" da Barra de componente.*

Vamos criar nossa interface. Primeiramente adicionamos um componente Panel que pode ser encontrado na aba “Standard” e possui o seguinte ícone.

Figura 5: *Ícone do componente Panel.*

Temos que mudar as seguintes propriedades desse componente.

Name	=	PanelInferior
Align	=	alBottom
Caption	=	(deixar em branco)
Height	=	88

Dentro desse Panel adicionamos cinco componentes Button. Esse componente pode ser encontrado na aba de componente “Standard” e possui o seguinte ícone.

Figura 6: *Ícone do componente Button.*

A seguir a lista das propriedades desses cinco Button que devem ser modificadas.

Name	=	ButtonIniciarAq

Caption	=	Iniciar Aquisição
Font/Style/fsBold	=	true
Top	=	8
Left	=	8
Width	=	113

Name	=	ButtonPararAq
Caption	=	Parar Aquisição
Font/Style/fsBold	=	true
Top	=	8
Left	=	128
Width	=	113

Name	=	ButtonLer
Caption	=	Ler Buffer
Font/Style/fsBold	=	true
Top	=	8
Left	=	248
Width	=	89

Name	=	ButtonLimparBuffer
Caption	=	Limpar Buffer
Font/Style/fsBold	=	true
Top	=	8
Left	=	344
Width	=	97

Name	=	ButtonLimparGrafico
Caption	=	Limpar Grafico
Font/Style/fsBold	=	true
Top	=	8

Left	=	448
Width	=	97

Dessa forma teremos a seguinte interface até o momento.

Figura 7: *Interface gráfica até o momento.*

Agora temos que adicionar o componente Chart, que pode ser encontrado na aba "Additional" da barra de componentes.

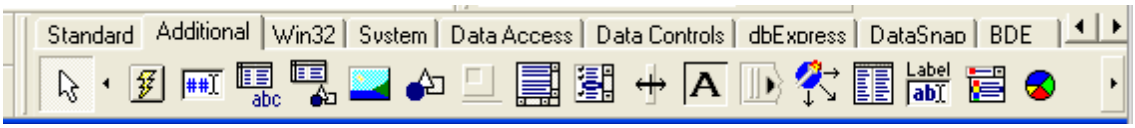

Figura 8: *Aba "Additional" da barra de componentes.*

Esse componente possui o seguinte ícone:

Figura 9: *Ícone do componente Chart.*

Adicionamos esse componente e modificamos as seguintes propriedades.

Name	=	ChartSensores
BackColor	=	clWhite
View3D	=	false
Align	=	alClient

Após modificar essas propriedades, clicamos com o botão direito do mouse sobre o componente Chart, um menu será apresentado e selecionamos a opção “EditChart” nesse menu. Com isso, será apresentada uma janela onde podemos configurar vários parâmetros do componente Chart. Essa janela tem a seguinte aparência.

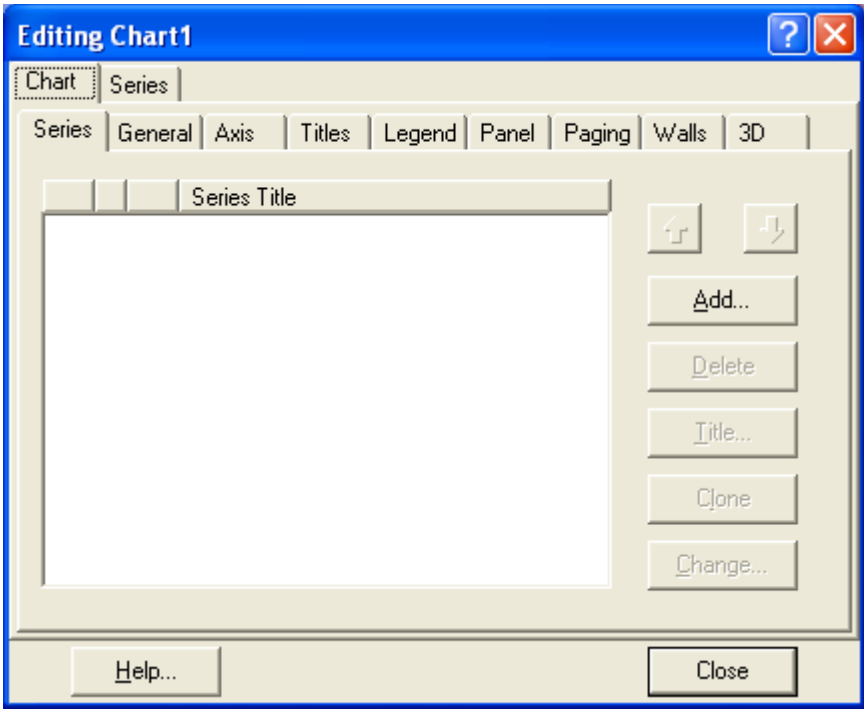

Figura 10: *Janela de configuração de um componente Chart.*

Temos que adicionar um gráfico ao componente Chart. Esse gráfico é denominado “Serie” na nomenclatura desse componente. Podemos adicionar essa “Serie” indo na aba “Chart / Series”, da janela de configuração que nos foi apresentada e nessa aba pressionar o botão “Add...”. Será apresentada uma tela onde podemos selecionar o tipo do gráfico que queremos adicionar, vamos utilizar uma “Serie” do tipo “Fast Line”.

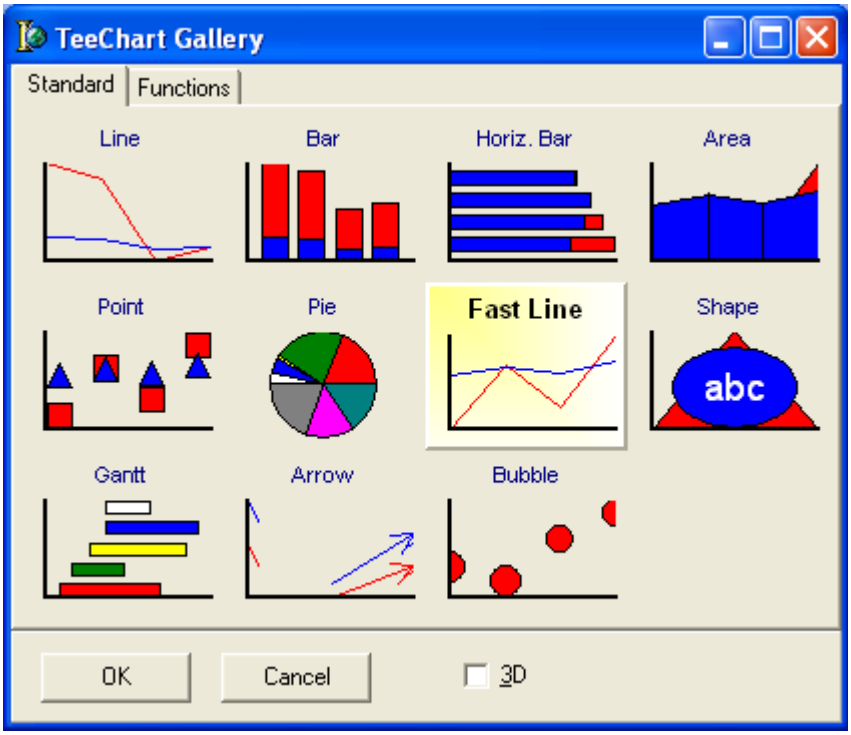

Figura 11: *Tela para a seleção do tipo de "Serie".*

Após a adição do “Serie”, a janela de configuração terá a seguinte aparência.

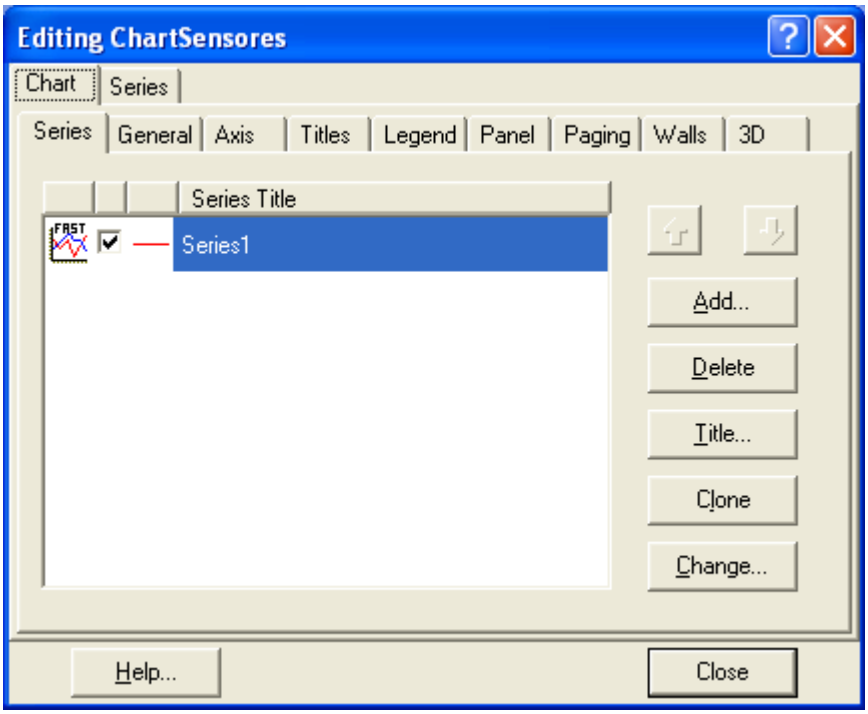

Figura 12: *Janela de configuração do Chart após adição da "Serie".*

Para uma melhor visualização dos gráficos, vamos remover o título e a legenda do Chart. Para isso temos que desmarcar a opção “Visible” na aba “Chart / Titles” e “Chart / Legend”.

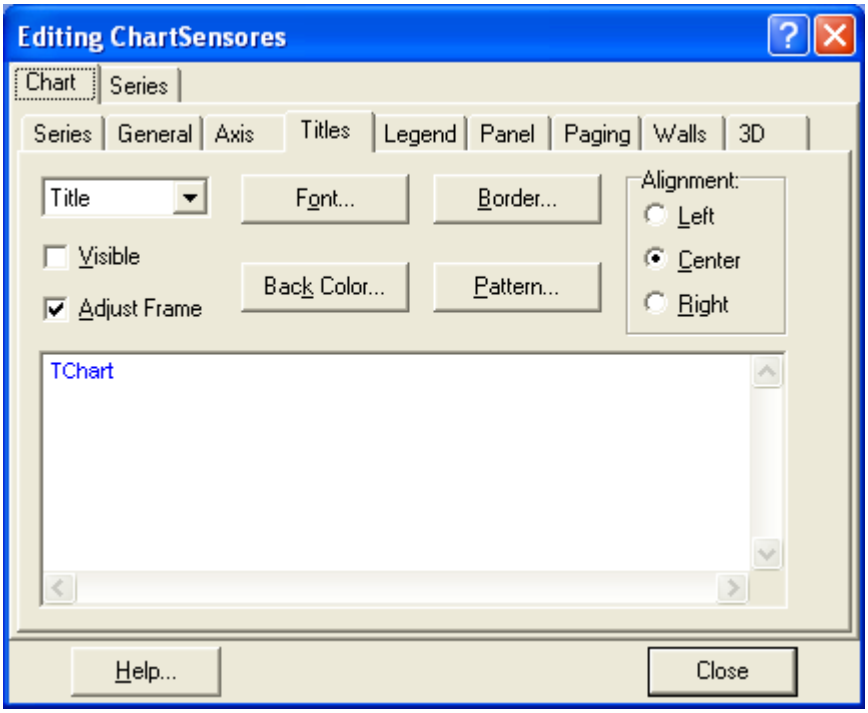

Figura 13: *Tela de configuração para remoção do título do gráfico. Observe que a opção "Visible" foi desmarcada.*

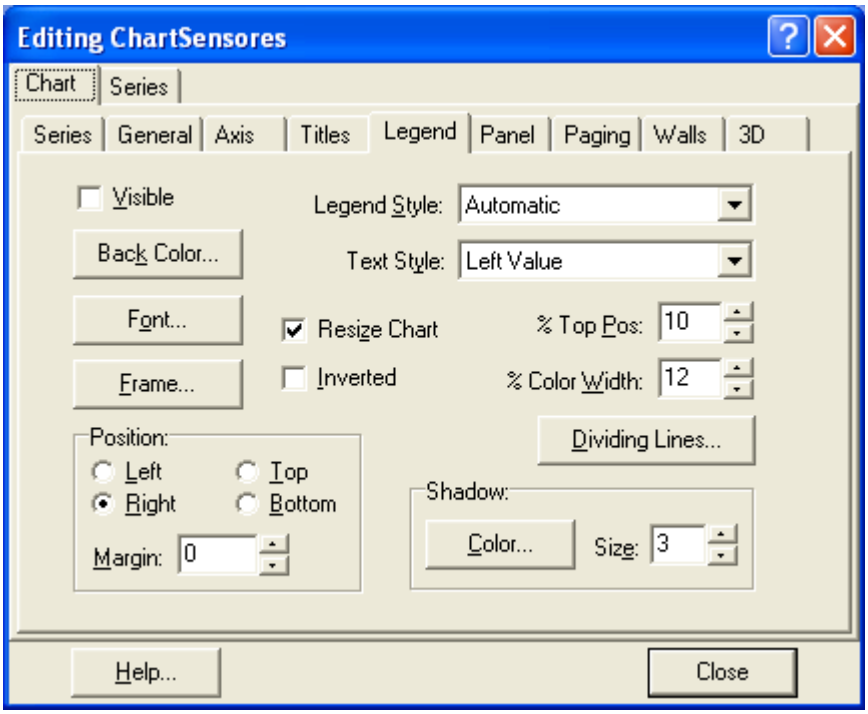

Figura 14: *Tela de configuração para remoção da legenda do gráfico. Observe que a opção "Visible" foi desmarcada.*

Outra configuração que é interessante que modifiquemos é o valor mínimo e máximo apresentados no eixo Y do gráfico. Pela configuração padrão esses valores são definidos automaticamente conforme vamos adicionando valores, no entanto, como o valor da leitura dos sensores varia de 0 à 1023, então podemos definir que o eixo Y irá apresentar valores apenas nessa faixa. Isso facilitará a interpretação do gráfico pois o valor mínimo e máximo de apresentação será sempre o mesmo.

Podemos modificar esses valores na aba “Chart/Axis”. Nessa aba selecionamos o eixo “Left”, que é o eixo Y, em seguida desmarcamos a opção “Automatic” desse eixo. As duas opções abaixo dessa opção serão habilitadas permitindo que selecionemos o valor máximo e mínimo do eixo.

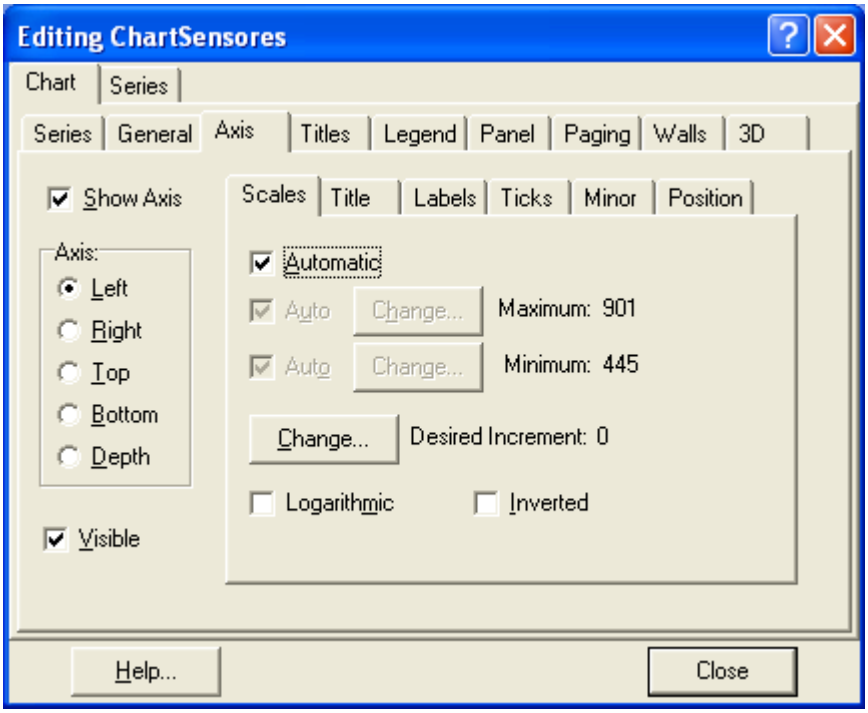

Figura 15: *Aba "Chart/Axis" com o eixo "Left" selecionado e a opção "Automatic" ainda marcada.*

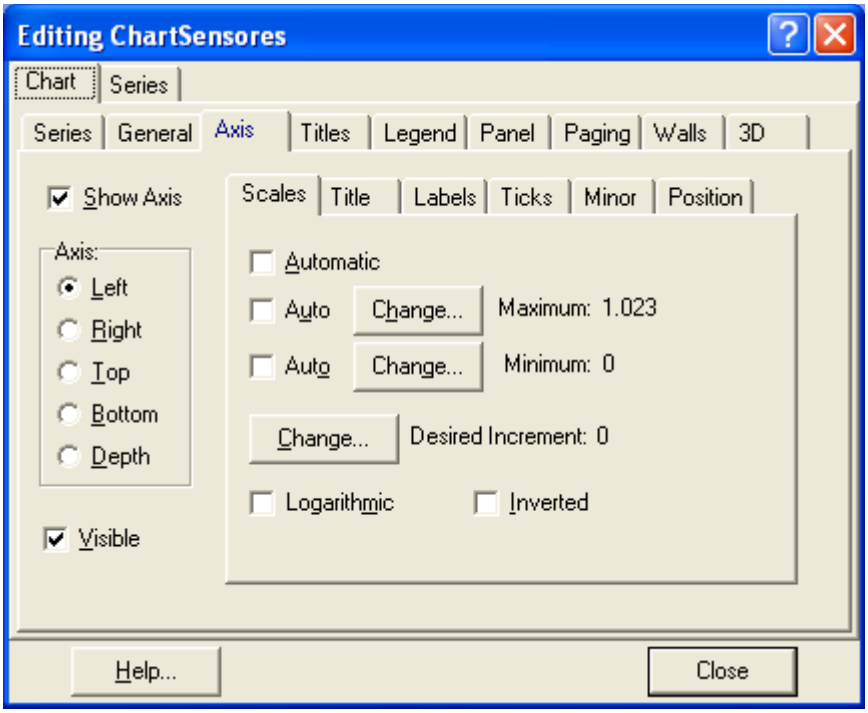

Figura 16: *Aba "Chart/Axis" com o eixo "Left" selecionado e a opção "Automatic" já desmarcada. Duas novas opções de seleção de valor máximo e mínimo foram habilitadas.*

Após fazer isso clicamos no botão “Close”. Nesse momento nosso Form principal se parecerá com algo assim.

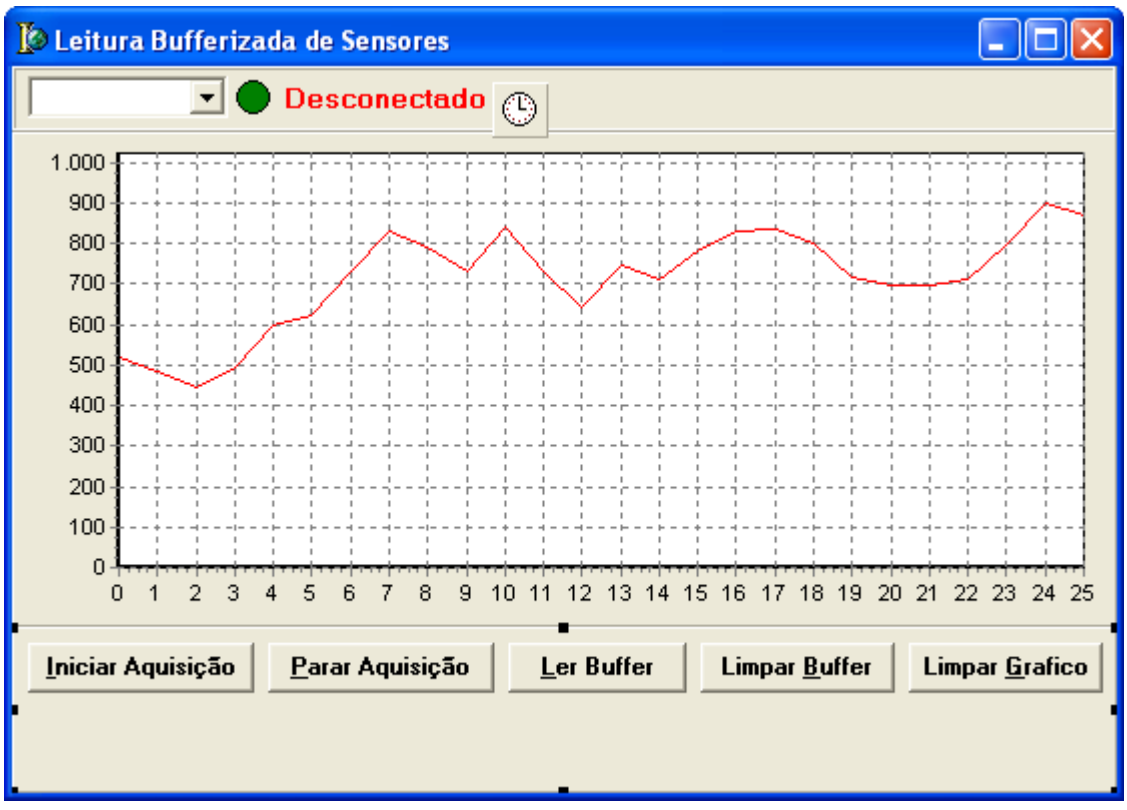

Figura 17: *Aparência do Form principal.*

Agora vamos adicionar os componentes para a seleção do intervalo entre as leituras. Para isso temos que adicionar um Label e um ComboBox. O componente Label possui o seguinte ícone.

Figura 18: *Ícone do componente Label.*

E o componente ComboBox possui o seguinte ícone:

Figura 19: *Ícone do componente ComboBox.*

Adicionamos um Label e modificamo as seguinte propriedades.

Name	=	LabelIntervalo
Caption	=	Intervalo (ms):
Font/Style/fsBold	=	true
Top	=	40
Left	=	8

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

Name	=	ComboBoxIntervalo
Style	=	csDropDownList
Items.Strings	=	1, 5, 10, 20, 50, 100, 500, 1000, 5000, 10000, 60000, 300000
ItemIndex	=	4
Top	=	56
Width	=	113

Dessa forma teremos nosso Form principal com a seguinte aparência.

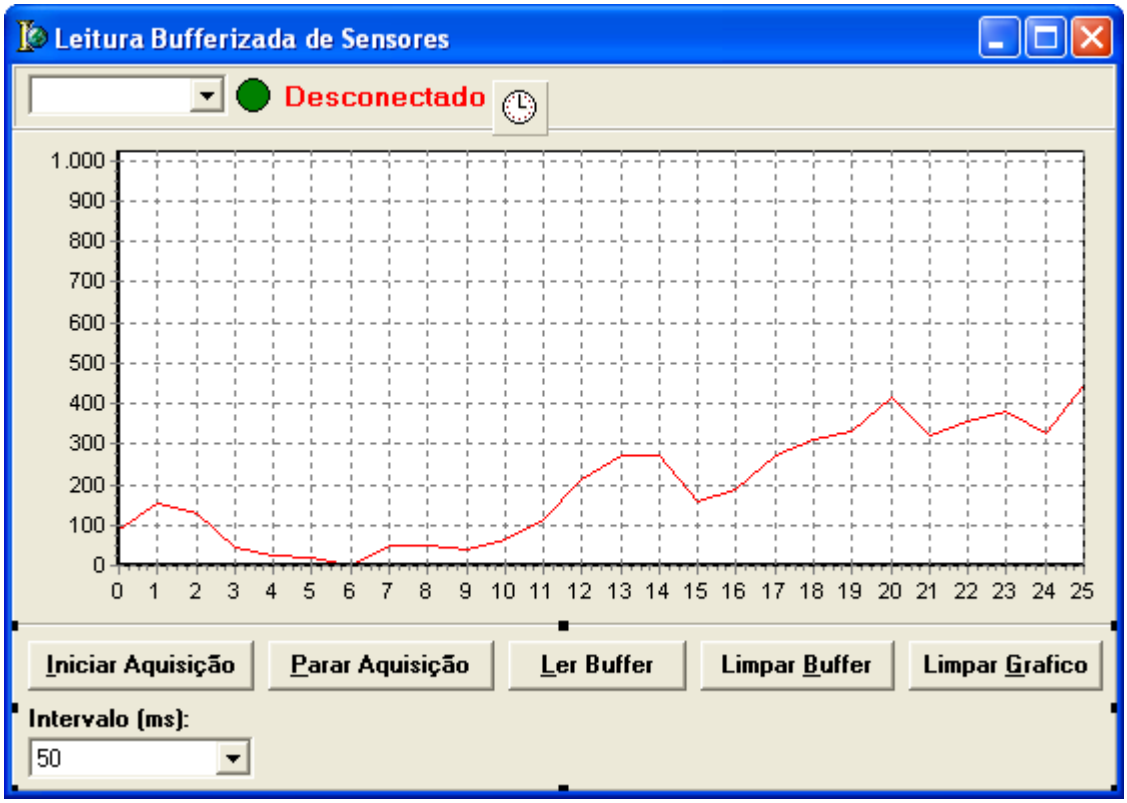

Figura 20: *Form após adicão de opção de seleção de intervalo entre leituras.*

Para selecionar o canal do sensor que será lido temos que adicionar mais um Label e um ComboBox. Adicionamos um Label e modificamos as seguinte propriedades.

Name	=	LabelSensor
Caption	=	Sensor:
Font/Style/fsBold	=	true
Top	=	40
Left	=	128

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

Name	=	ComboBoxSensor
Style	=	csDropDownList
Items.Strings	=	0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15
ItemIndex	=	0
Top	=	56
Width	=	57

Dessa forma teremos nosso Form principal com a seguinte aparência.

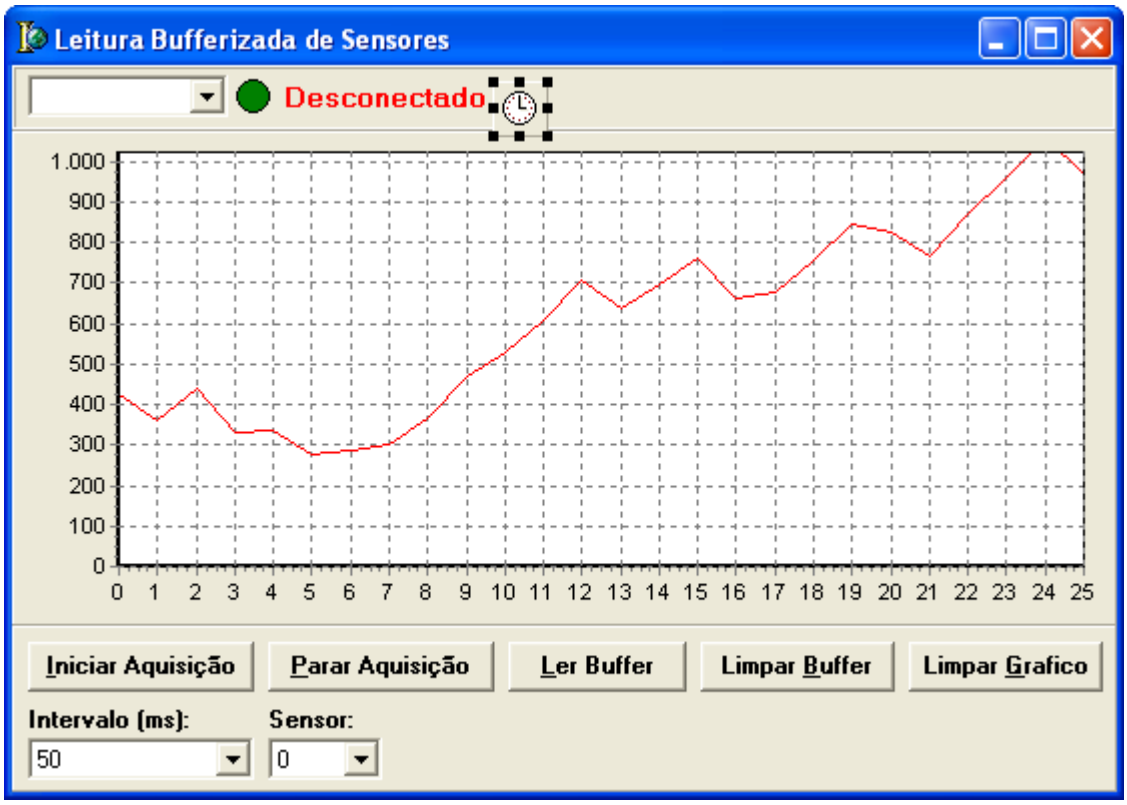

Figura 21: *Aparência final do Form.*

Assim finalizamos a interface do programa e podemos iniciar o desenvolvimento do código. O primeiro passo para isso é criar um manipulador para o evento OnClick do botão "Iniciar Aquisição". Para fazer isso podemos selecionar o componente Button correspondente, 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.ButtonIniciarAqClick(Sender:
                                                TObject);
begin
end;
```

Dentro desse manipulador adicionamos o código que irá iniciar a aquisição. Para isso, utilizaremos o método SensorBufferOn da biblioteca de controle do Kit. Esse método possui a seguinte declaração.

```
Procedure SensorBufferOn(tf : Boolean;
                         sensor, interval : Integer);
```

Esse método possui três parâmetros. O primeiro deles determina se a aquisição deve ser ligada ou desligada, o segundo indica o canal de sensor que queremos ler e o último indica o intervalo em milissegundos entre as leituras. Sabendo disso já podemos implementar o código do manipulador do evento de clique do botão "Iniciar Aquisição". A seguir, o código desse manipulador.

```
Procedure TFormMain.ButtonIniciarAqClick(Sender:
                                                TObject);
begin
  // Inicia aquisição
  kit.SensorBufferOn(true,
      StrToInt(ComboBoxSensor.Text),
      StrToInt(ComboBoxIntervalo.Text));
end;
```

Esse código é bem simples. Apenas fazemos uma chamada ao método SensorBufferOn passando como primeiro parâmetro "true" que indica que queremos ligar a aquisição. O outros dois parâmetros são definidos pelos itens selecionados no ComboBox de seleção do sensor que deve ser lido e do ComboBox para seleção do intervalo entre as leituras. Como esses dois últimos parâmetros são inteiros, então utilizamos o método StrToInt para converter o propriedade "Text" dos ComboBox, que é do tipo String, para um Integer.

O número indicado pela ComboBox dos Sensores corresponde a um canal de sensor específico, tanto do *Módulo de Sensores* quanto do *Módulo de Sensores Genérico*. A tabela contendo a relação entre os sensores e o seu canal correspondente encontra-se disponível no manual do KDR5000. Para o uso com o MEC1000, que suporta um menor número de sensores, apenas os primeiros oito itens do ComboBox serão funcionais.

A próxima funcionalidade que será implementada é a interrupção da aquisição. Temos que criar o manipulador do evento de clique do botão "Parar Aquisição". Para fazer isso podemos selecionar o componente Button correspondente, 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.ButtonPararAqClick(Sender:
                                                TObject);
begin
end;
```

Vamos utilizar o mesmo método que utilizamos para iniciar a aquisição, o SensorBufferOn. No entanto vamos passar o valor "false" como primeiro parâmetro e com isso a

aquisição será interrompida. Os outros dois parâmetros serão ignorados. O código ficará assim.

```
Procedure TFormMain.ButtonPararAqClick(Sender:
                                              TObject);
begin
  // Finaliza aquisição
  kit.SensorBufferOn(false, 0, 0);
end;
```

Agora que já podemos iniciar e parar uma aquisição, vamos implementar a parte do código que faz a leitura dos valores armazenados no buffer de leituras. Novamente, como já fizemos outras duas vezes, temos que criar o manipulador para o evento de clique para o botão "Ler Buffer". A forma mais simples de fazer isso é dar um duplo clique sobre esse botão no Form e com isso o seguinte código será criado.

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

O método que vamos utilizar para fazer a leitura do conteúdo do buffer é o SensorReadBuffer. A sua declaração é a seguinte.

```
Procedure SensorReadBuffer(
                var data : DynIntegerArray);
```

Esse método possui apenas um parâmetro passado por referência que é um vetor dinâmico de valores do tipo Integer. Nesse vetor será retornado todo o conteúdo do buffer. Após a leitura do buffer ele é apagado de modo que fique vazio para receber mais leituras. A seguir vamos iniciar a implementação do código do manipulador que acabamos de criar.

```
Procedure TFormMain.ButtonLerClick(Sender:
                                              TObject);
var
  // Vetor de inteiros que armazena as leituras
  leituras : DynIntegerArray;
```

```
  // Variável índice para o "for"
  i : integer;
begin
  // Lê buffer
  kit.SensorReadBuffer(leituras);

  ...
end;
```

Nesse trecho de código são declaradas duas variáveis, um vetor de inteiros denominado "leituras" que irá armazenar o conteúdo do buffer de leituras e um inteiro denominado "i" que servirá de índice para uma estrutura "for" que veremos a seguir. Após a declaração das variáveis, é invocado o método SensorReadBuffer que irá ler o conteúdo do buffer e armazenar os valores lidos no vetor de inteiros passado como parâmetro, no caso o vetor denominado "leituras". O código a seguir mostra como fazemos para inserir esses valores lidos no gráfico.

```
Procedure TFormMain.ButtonLerClick(Sender:
                                          TObject);
var
  // Vetor de inteiros que armazena as leituras
  leituras : DynIntegerArray;

  // Variável índice para o "for"
  i : integer;
begin
  // Lê buffer
  kit.SensorReadBuffer(leituras);

  // Insere os valores lidos no gráfico
  for i:=0 to Length(leituras)-1 do
    ChartSensores.Series[0].Add(valores[i]);
end;
```

Com isso já podemos testar nosso 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. Selecione a porta serial correta, defina como intervalo entre as leituras o valor 100, escolha uma canal de sensor qualquer, de preferência um com um sensor que tenha uma resposta dinâmica, como por exemplo o

sensor de luminosidade, e pressione o botão “Iniciar Aquisição”. Em seguida pressione o botão “Ler Buffer” algumas vezes, deverá ser apresentado um gráfico com os valores lidos. Teste alguns valores diferentes para o intervalo entre leituras e para o canal de sensor.

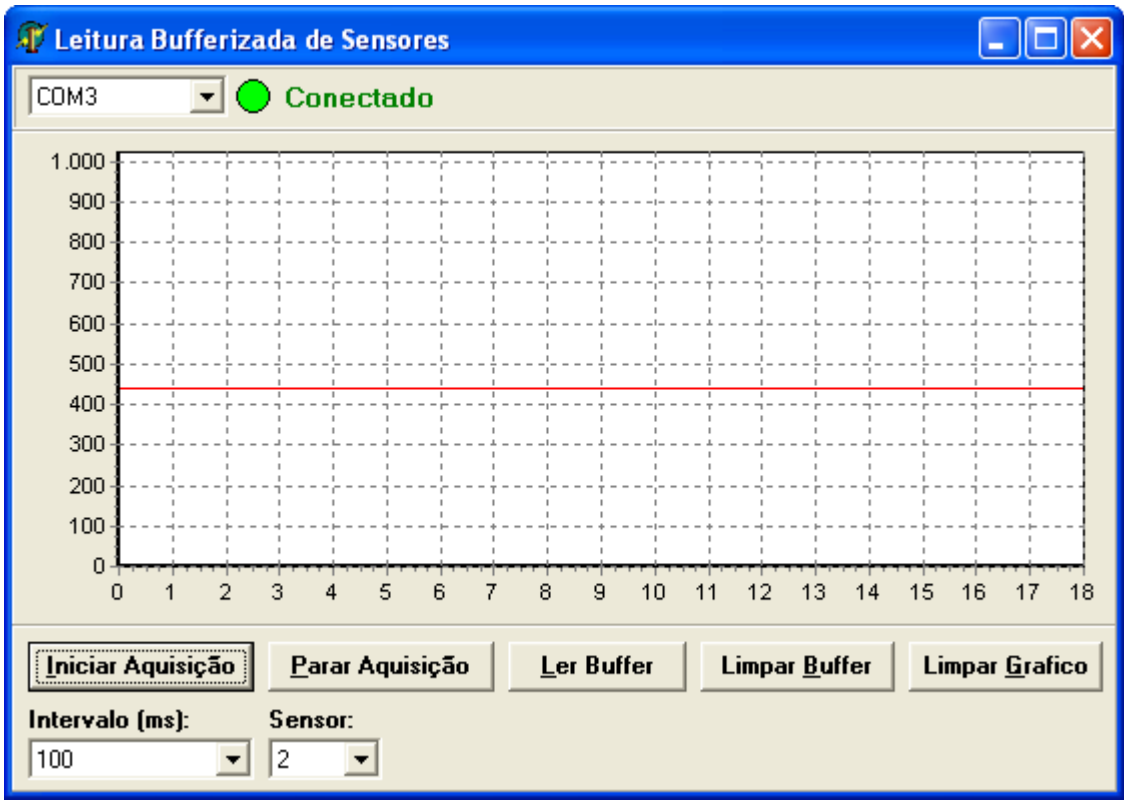

Figura 22: *Aquisição do canal 2 com um intervalo entre leituras de 100ms.*

O programa já está funcional. Vamos rapidamente implementar o código para os últimos dois botões, o “Limpar Buffer” e o “Limpar Gráfico”. Para limpar o buffer de leituras utilizamos o método SensorClearBuffer. Esse método não possui parâmetros e sua declaração é a seguinte.

```
Procedure SensorClearBuffer();
```

Temos que criar o manipulador de eventos para o clique no botão “Limpar Buffer”. A forma mais simples de fazer isso é dar um duplo clique sobre esse botão e com isso o seguinte código será criado.

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

A implementação desse manipulador ficará assim.

```
Procedure TFormMain.ButtonLimparBufferClick(Sender:
                                                TObject);
begin
  // Limpa buffer
  kit.SensorClearBuffer;
end;
```

Por fim temos que criar um manipulador para o botão “Limpar Gráfico”. Da mesma forma que fizemos anteriormente, damos um duplo clique sobre esse botão e o seguinte manipulador será criado.

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

Vamos utilizar o método “Clear” da Serie de número “0” para limpar o gráfico. A implementação desse manipulador ficará assim.

```
Procedure TFormMain.ButtonLimparGraficoClick(
                                    Sender: TObject);
begin
  // Limpa gráfico
  ChartSensores.Series[0].Clear;
end;
```

Pronto, com isso finalizamos a implementação desse programa. Sua aparência final deve ser a seguinte.

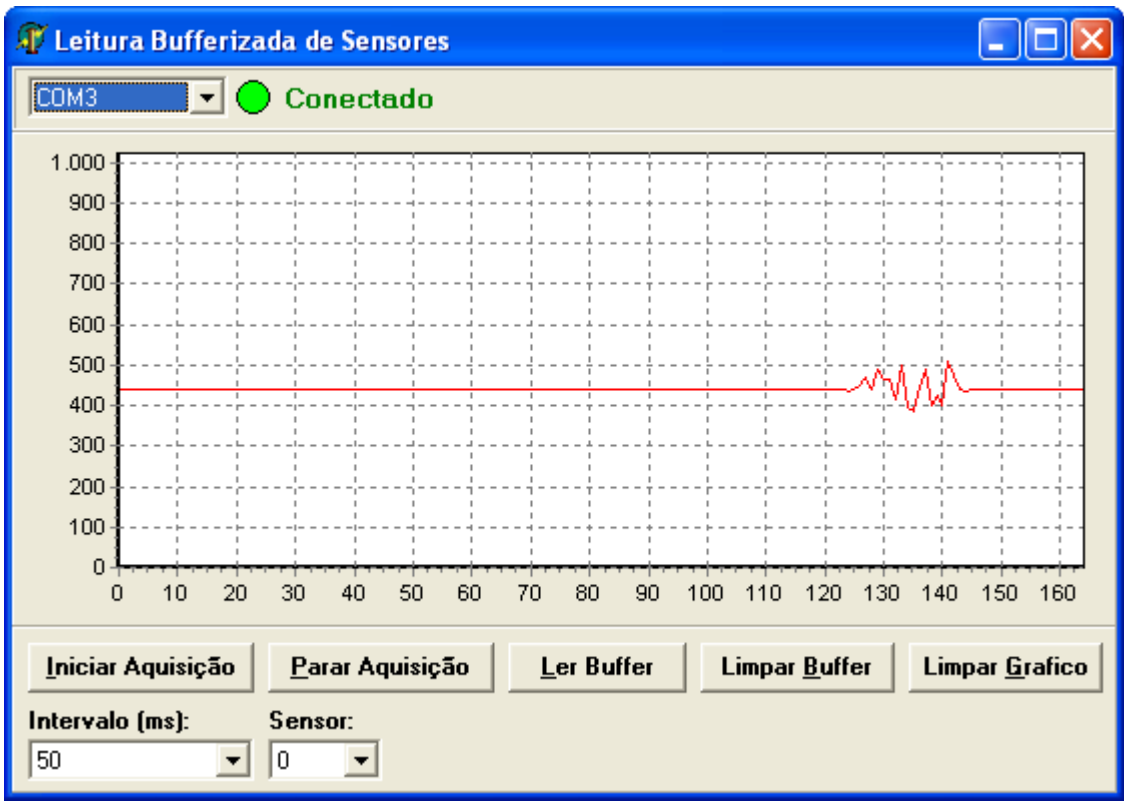

Figura 23: *Aparência final do programa.*

4 – Dicas

Uma funcionalidade interessante que podemos implementar nos programas criados nesse tutorial é uma atualização automática da leitura dos sensores. Para isso vamos criar um novo Timer e fazer com que ele atualize o gráfico sem a necessidade de pressionarmos o botão “Ler Buffer”.

O componente Timer encontra-se na aba “System” da barra de componentes. A seguir uma imagem dessa aba.

Figura 24: *Aba "System" da barra de componentes.*

O componente Timer possui o seguinte ícone.

Figura 25: *Ícone do componente Timer.*

Vamos adicionar um Timer e modificar as seguintes propriedades desse Timer.

Name = TimerSensores
Interval = 50

Agora temos que criar um manipulador para esse Timer. Isso pode ser feito dando um duplo clique com o mouse sobre ele. O seguinte código será criado.

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

Esse manipulador será executado a cada 50 milisegundos, que é um tempo pequeno o suficiente para atualizar o gráfico de forma satisfatória. A seguir o código que fará a atualização do gráfico.

```
Procedure TFormMain.TimerSensoresTimer(Sender:
                                              TObject);
begin
  try
    // Atualiza leitura de sensores
    ButtonLerClick(ButtonLer);
  except
  end;
end;
```

O que fazemos nesse manipulador é chamar o manipulador de clique no botão “Ler Buffer”. O resultado disso é como se estivéssemos pressionando o botão “Ler Buffer” a cada 50 milisegundos. A estrutura **try-except** foi utilizada para que falhas na leitura do buffer sejam ignoradas e não sejam emitidas mensagens de erro repetidas vezes.

Podemos testar os programas e verificar que agora a leitura do buffer é feita automaticamente, sem a necessidade de pressionar o botão “Ler Buffer”.

Outras funcionalidade que seria interessante é que quando modificamos o sensor lido ou o intervalo entre as leituras isso se refletisse imediatamente na aquisição. Isso é simples de implementar. Temos apenas que criar um manipulador para o evento de mudança dos dois ComboBox que controlam esses parâmetros. Isso pode ser feito dando um duplo clique sobre eles. A seguir o evento de mudanças do ComboBox de seleção de intervalo.

```
Procedure TFormMain.ComboBoxIntervaloChange(
                                    Sender: TObject);
begin
  // Atualiza intervalo da aquisição
  ButtonIniciarAqClick(ButtonIniciarAq);
end;
```

Dentro desse manipulador apenas chamamos o manipulador do evento de clique do botão “Iniciar Aquisição”, dessa forma, toda vez que mudamos o intervalo entre as leituras a aquisição será reiniciada já com o nova parâmetro. Fazemos a mesma coisa com o ComboBox de seleção do canal de sensor. O código fica assim.

```
Procedure TFormMain.ComboBoxSensorChange(
                                    Sender: TObject);
begin
  // Atualiza intervalo da aquisição
  ButtonIniciarAqClick(ButtonIniciarAq);
end;
```

5 – Conclusão

Nesse tutorial, aprendemos como o que é e como fazer a leitura de sensores bufferizada. Vimos os métodos SensorBufferOn, SensorReadBuffer e SensorClearBuffer e os utilizamos para criar um programa que faz a leitura e apresenta os dados de sensores de forma gráfica. Com os conhecimentos aqui adquiridos, podemos utilizar a leitura de sensores bufferizada em nossos programas.