

Delphi: Comunicação serial com Arduino

Autor: Cleiton Bueno **Ano:** 2014

Blog: <http://www.cleitonbueno.wordpress.com>

Faz alguns anos que não desenvolvia nada em Delphi (Object Pascal), mas como no começo deste blog eu publiquei alguns programas escritos em Delphi recebo com frequência solicitações de enviar o fonte ou algum material, como eu prefiro ensinar a pescar do que dar o peixe, resolvi trabalhar em cima disso, como já escrevi sobre comunicar serial com Arduino usando Python e pretendo fazer com outras linguagens desta vez foia tão esperada vez com o Delphi.

Não tenho as versões mais recentes e acabei de usar uma já ultrapassada o Delphi 7, porém como recompensa eu fiz 3 vídeos além deste post e o fonte disponibiliza do, onde abordei desde o download do componente(biblioteca para comunicar serial), instalação, configuração e comunicação com a mesma, então vamos lá:

Foi utilizado:

IDE Borland Delphi 7.0 (Build 4.453)

Componente ComPort Library Version 4.11 by Dejan Crnila

Por que usei ComPort Library ou CPortLib? Poderia usar outros porém precisava de um que primeiro eu já tenha usado e segundo que não seja pago ou de funcionamento limitado, tem um excelente que usei a um bom tempo atras o [nrComm](#), esse é muito fera, tem várias melhorias e implementações que faltam no ComPort Lib além de toda implementação para GSM, USB HID e Bluetooth e suporte a várias versões do Delphi, mas para implementações simples e até intermediária o ComPort é perfeito, conforme vai aumentando falta recursos implementados e que você deve fazer na mão o tratamento já o nrComm praticamente entrega a informação em uma variável ou buffer pra você.

O componente pode ser baixado [aqui](#) ou em www.sourceforge.net/projects/comport/, não abordarei aqui a instalação do componente e configuração da IDE então segue o link do vídeo que gravei:

Instalando ComPort Library no Delphi 7 para comunicar com Arduino

<https://www.youtube.com/watch?v=N1amhXENGHI>

Após a instalação e configuração da IDE uma nova paleta é para surgir na IDE como Figura01.

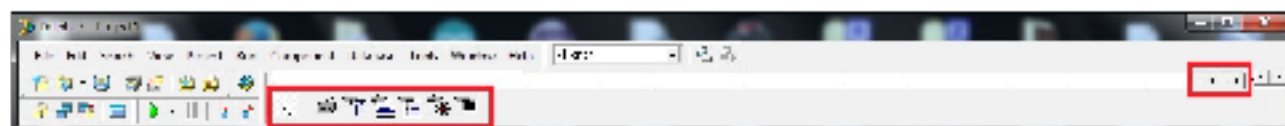


Figura01 – Novo componente instalado

Para programar é bem lá estilo Delphi, clica arrasta e solta o componente e isso pode ser feito com o primeiro componente da aba CPortLib o ComPort (CPort), cliquei em cima dele e depois clique novamente em cima do form, em seguida inserida um button ou bitbtn que daremos o nome Painel como na Figura02.

O que eu quero dizer com a Figura03 é que o Delphi com o CPortLib comunicará (Enviar/Receber) dados por comunicação serial, neste exemplo deste artigo estou usando o Arduino UNO, ok! Porém o Delphi e o CPortLib não sabe o que é Arduino, ou seja, eles não sabem o que irá ter do outro lado do cabo pois o mesmo código eu posso usar para comunicar com um Arduino, PC (Windows ou Linux), Texas LaunchPad MSP430, Raspberry PI, BeagleBone Black, microcontrolador, ARM Cortex-Mx, onde eu quero chegar é que não importa o que se tem do outro lado implementando corretamente o tratamento da comunicação serial e a correta implementação do "protocolo" maravilha comunicará, eu poderia muito bem neste artigo escrever e tratar o caractere 't' que será enviado usando um MSP430 da LaunchPad e funcionaria perfeitamente com nossa aplicação, espero ter passado a ideia e com dareza.

Vamos agora a um exemplo bem básico de uso do CPortLib ou ComPort Library o que preferir :)

Comunicação serial básica de Delphi com Arduino usando ComPort

<https://www.youtube.com/watch?v=3Xt96eG9pzg>

O que foi utilizado acima é muito simples, vamos ver do que precisei:

- 1) Componente ComPort (CPort) e o nome alterado para comport
- 2) Componente ComTerminal (CPortCtl) e nas propriedades sem ComPort escolher comport
- 3) Usar as seguintes funções do componente:

comport.Connected	\\ Retorna true se a conexão esta estabelecida
comport.Open	\\ Abre a conexão serial
comport.WriteStr('t')	\\ Escreve o caractere 't' na comunicação serial
comport.Close	\\ Fecha a conexão serial

No meu ponto de vista é o exemplo mais básico, simples e funcional para comunicar serial com Delphi, porém eu acho incomodo usar o ComTerminal salvo caso apenas para monitorar a serial, mas acho legal usar um Memo e inserir nele o que chegar pela serial além de outras informações durante a comunicação como erros, alertas e status além de usar mais recursos desse componente, vamos ao terceiro vídeo e mais completo agora.

Comunicação serial completa de Delphi com Arduino usando ComPort

<https://www.youtube.com/watch?v=Cc3Hqdvaz0>

Agora vamos explorar o que foi utilizado e como utilizado no vídeo acima:

- 1) Componente ComPort (CPort): **Name:** comport
- 2) Componente TMemo com: **Name:** MemoLog; **Lines:** Log da comunicação serial...; **ScrollBars:** ssVertical
- 3) Quatro TButton:
 - 3.1) **Name:** btnPainelConfig; **Caption:** Painel; **Cursor:** crHandPoint; **ShowHint:** True; **Hint:** Mensagem;
 - 3.2) **Name:** btnOpenPort; **Caption:** Abrir Conexão; **Cursor:** crHandPoint; **ShowHint:** True; **Hint:** Mensagem;
 - 3.3) **Name:** btnComunicar; **Caption:** Comunicar; **Cursor:** crHandPoint; **ShowHint:** True; **Hint:** Mensagem;
 - 3.4) **Name:** btnClosePort; **Caption:** Fechar Conexão; **Cursor:** crHandPoint; **ShowHint:** True; **Hint:** Mensagem;
- 4) Form: **Name:** FmPrincipal; **Caption:** Comunicação Delphi4 Arduino; **Position:** poDesktopCenter

No terceiro vídeo Comunicação serial completa de Delphi com Arduino usando ComPort eu realizei alguns tratamentos importantes como:

- 1) O botão Abrir Conexão só é habilitado quando a porta é selecionada em Painel
- 2) Os botões Comunicar e Fechar Conexão só são habilitados após abrir a conexão com sucesso
- 3) Após abrir a conexão o botão Abrir Conexão é desabilitado e a qualquer momento você pode Comunicar ou Fechar a conexão
- 4) Caso o operador esqueça de ir em Fechar Conexão no evento OnClose do Form ou seja quando o Form for encerrado o evento comPort.Close é chamado
- 5) É realizado um Try/Except quando realiza a abertura da conexão como verificar se já não está aberta ou alguma outra exceção e o resultado é reportado no MemoLog

Aplicação final Figura05.

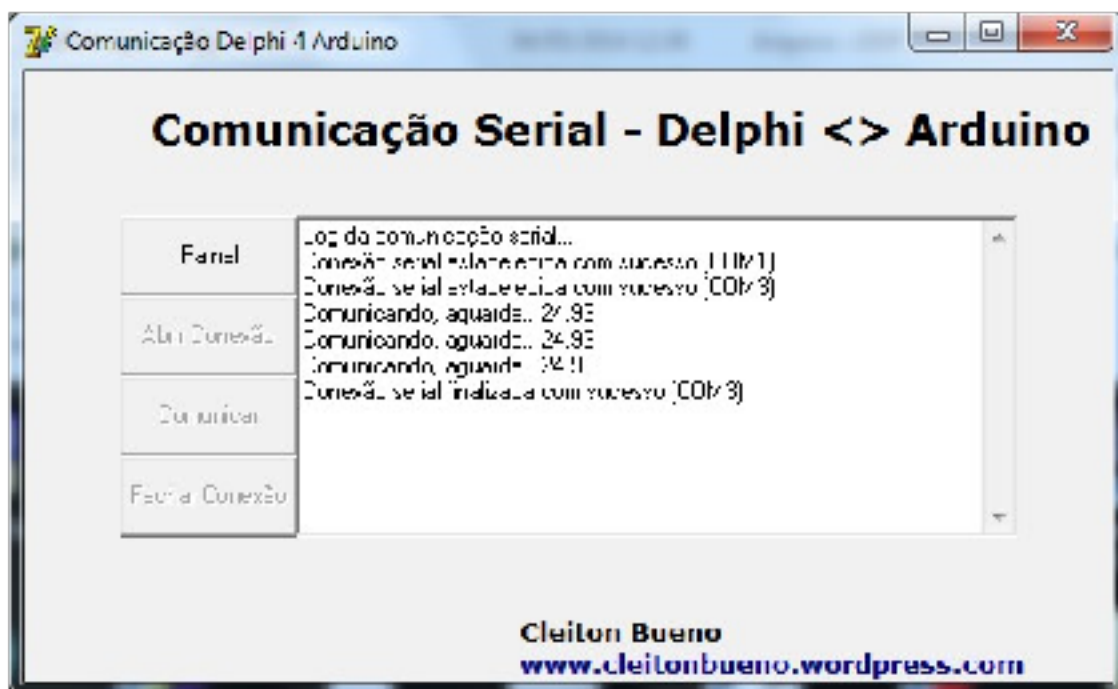


Figura05 – E todas as etapas do funcionamento e logs

Código-fonte:

<http://cleitonbueno.com/downloads/blog/Delphi4Arduino/ProjetoComunicacaoArduino4Delphi.zip>

ComPort Library

www.sourceforge.net/projects/comport/

Vídeos:

Instalando ComPort Library no Delphi 7 para comunicar com Arduino

<https://www.youtube.com/watch?v=N1amhXENGHI>

Comunicação serial básica de Delphi com Arduino usando ComPort

<https://www.youtube.com/watch?v=3Xt96eG9pzg>

Comunicação serial completa de Delphi com Arduino usando ComPort

<https://www.youtube.com/watch?v=Cc3HqdvarezQ>



Este trabalho de Cleiton Bueno, foi licenciado sob uma Licença Creative Commons Atribuição NãoComercial Compartilha Igual 3.0 Brasil.

Baseado no trabalho em <http://www.cleitonbueno.wordpress.com>.

Código-fonte Firmware Arduino

```
/* Temperatura em Celsius */

int PinAnalogLM35 = 0; //Setando Pino A0
float valAnalog = 0; //Iniciando variavel valAnalog como 0
float temp = 0; //Iniciando variavel temp como 0

void setup(){
  Serial.begin(9600);
}

void loop(){
  if (Serial.available() > 0){
    if (Serial.read() == 116){ // letra t
      // Lento o pino A0, a qui estamos obtendo o valor
      valAnalog = analogRead(PinAnalogLM35);
      temp = (valAnalog * 500) / 1023;
      Serial.println(temp);
    }
  }
}
```

Código-fonte Aplicação Delphi

```
unit UnPrincipal;

interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, StdCtrls, CPort;

type
  TFmPrincipal = class(TForm)
    comPort: TComPort;
    MemoLog: TMemo;
    BtnPainelConfig: TButton;
    btnOpenPort: TButton;
    btnComunicar: TButton;
    btnClosePort: TButton;
    Label1: TLabel;
    lbNomeRodape: TLabel;
    lbLinkRodape: TLabel;
    procedure BtnPainelConfigClick(Sender: TObject);
    procedure btnOpenPortClick(Sender: TObject);
    procedure FormClose(Sender: TObject; var Action: TCloseAction);
    procedure btnComunicarClick(Sender: TObject);
    procedure comPortRxChar(Sender: TObject; Count: Integer);
    procedure btnClosePortClick(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  FmPrincipal: TFmPrincipal;

implementation

{$R *.dfm}

{
  Titulo: Delphi: Comunicação serial com Arduino
  Autor: Cleiton Bueno
}
```

Ano: 2014
Link: <http://www.cleitonbueno.wordpress.com>
Licença: BSD

Use, divulgue, modifique mas se possível cite a fonte :)

Este trabalho de Cleiton Bueno, foi licenciado sob uma Licença Creative Commons Atribuição-NãoComercial-CompartilhaIgual 3.0 Brasil.
Baseado no trabalho em <http://www.cleitonbueno.wordpress.com>.

```
}
procedure TFmPrincipal.BtnPainelConfigClick(Sender: TObject);
begin
  comport.ShowSetupDialog;
  btnOpenPort.Enabled := True;

  // Aqui voce especifica sua porta serial, diretamente assim...
  // comport.Port := 'COM3'
  // Ou por um Edit, assim...
  // comport.Port := edtPorta.Text;
  // Na opção abaixo o BaudRate
  // Tome cuidado com o BaudRate que não é apenas para algo como:
  // 9600
  // Ele tem seu "padrão"
  // comport.BaudRate
  // StopBits
  // comport.StopBits
  // Paridade
  // comport.Parity
  // Data Bits
  // comport.DataBits
end;

procedure TFmPrincipal.btnOpenPortClick(Sender: TObject);
begin
  try
    // Abrindo a conexão serial
    comport.Open;
    if comport.Connected then
      begin
        MemoLog.Text := MemoLog.Text + 'Conexão serial estabelecida com sucesso ('+comport.Port+')';
        MemoLog.Lines.Add(""); // Gambi para ir para próxima linha, não lembro como usar #13
        btnOpenPort.Enabled := False;
        btnComunicar.Enabled := True;
        btnClosePort.Enabled := True;
      end
    else
      MemoLog.Text := MemoLog.Text + 'FALHA a abrir conexão serial com ('+comport.Port+')'
    except on E: Exception do
      begin
        MemoLog.Text := MemoLog.Text + 'ERRO a abrir conexão: Detalhes> '+E.Message;
      end
    end
  end;

procedure TFmPrincipal.FormClose(Sender: TObject);
var Action: TCloseAction;
begin
  // Caso o usuário feche o programa sem fechar a conexão
  comport.Close;
end;

procedure TFmPrincipal.btnComunicarClick(Sender: TObject);
begin
  // Sem mistério... inserindo a mensagem Comunicando, aguarde...
  // no memo e enviando o caractere 't' para a serial...
  MemoLog.Text := MemoLog.Text + 'Comunicando, aguarde...';
```

```

        comport.WriteStr('t');
    end;

    procedure TFmPrincipal.comportRxChar(Sender: TObject; Count: Integer);
    var
        RxCount: Integer;
        RxComport: String;
    begin
        // Primeiro, neste componente devemos saber
        // a quantidade de bytes recebidos
        RxCount := comport.InputCount;

        // Agora iremos chamar a função que irá receber o conteúdo
        // e informar onde será armazenado e a quantidade com a variável
        // acima
        comport.ReadStr(RxComport, RxCount);

        // Enviando para o MemoLog
        MemoLog.Text := MemoLog.Text + RxComport;
    end;

    procedure TFmPrincipal.btnClosePortClick(Sender: TObject);
    begin
        // Fechando a conexão quando o operador clicar no botão
        comport.Close;

        // Vamos trabalhar em cima desta ação
        if not comport.Connected then
        begin
            MemoLog.Text := MemoLog.Text + 'Conexão serial finalizada com sucesso (' + comport.Port + ')';
            btnClosePort.Enabled := False;
            btnComunicar.Enabled := False;
            btnOpenPort.Enabled := False;
        end
        else
            MemoLog.Text := MemoLog.Text + 'Falha ao finalizar conexão serial'
        end;
    end;
end.

```