Vážení zákazníci,

dovolujeme si Vás upozornit, že na tuto ukázku knihy se vztahují autorská práva, tzv. copyright.

To znamená, že ukázka má sloužit výhradně pro osobní potřebu potenciálního kupujícího (aby čtenář viděl, jakým způsobem je titul zpracován a mohl se také podle tohoto, jako jednoho z parametrů, rozhodnout, zda titul koupí či ne).

Z toho vyplývá, že není dovoleno tuto ukázku jakýmkoliv způsobem dále šířit, veřejně či neveřejně např. umisťováním na datová média, na jiné internetové stránky (ani prostřednictvím odkazů) apod.

redakce nakladatelství BEN – technická literatura
redakce@ben.cz

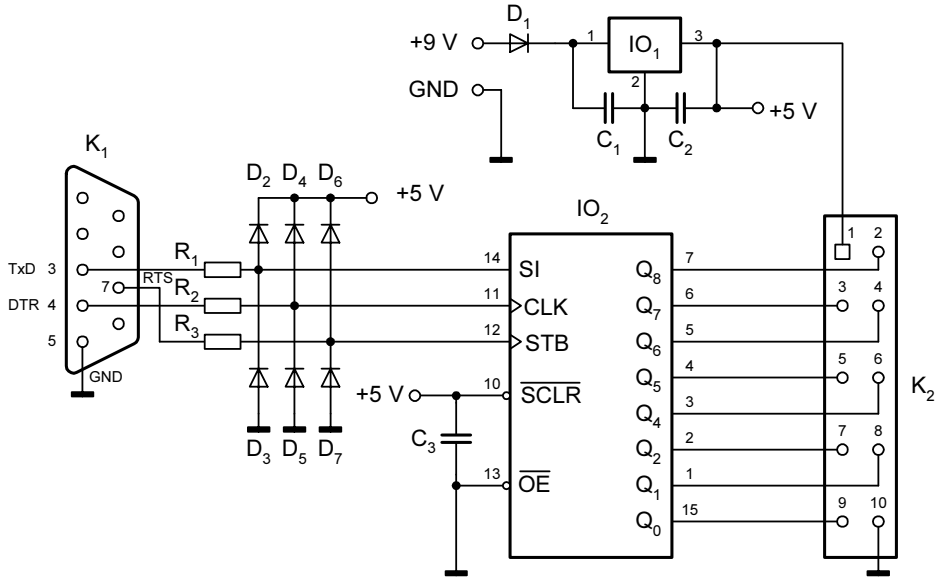
*V této kapitole si ukážeme, jak ovládat přípravek **ATLCDTX2** pomocí sériového portu počítače. Velmi podstatnou výhodou sériového portu je skutečnost, že používá napěťové úrovně RS-232C. Signály jsou pak více odolné rušení než například u paralelního portu (ten používá napěťové úrovně TTL). Použití sériového portu tak vede na jednodušší propojovací kabel (data se přenášejí sériově), který může mít délku až 10 m.*

*Ovládání displeje je řešeno pomocí speciální třídy **TATLCDTX2COM** implementované ve vývojovém prostředí **Delphi 7.0**.*

4.1 PŘÍPRAVEK COM2PSL

Vzhledem k tomu, že sériový port není vybaven dostatečným počtem výstupních linek, musí se pro připojení LCD displeje použít posuvný registr. Tak vznikl i přípravek **COM2PSL**.

Schéma zapojení je uvedeno na *obr. 4.1*.

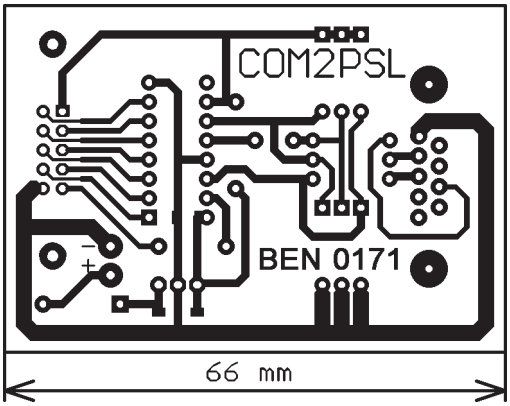


*Obr. 4.1 Schéma zapojení přípravku **COM2PSL***

Linky sériového kanálu: TxD, DTR a RTS jsou nejdříve pomocí diodových omezovačů (jsou tvořeny rezistory R_1 až R_3 a diodami D_2 až D_7) upraveny na napěťové úrovně srovnatelné s TTL. Poté vstupují do posuvného registru (typu SIPO) IO_2 (**74HCT595**). Linka TxD ovládá sériový vstup dat, linka DTR představuje hodinový signál a linka RTS je strobovací signál. Bity se tedy odesílají linkou TxD a potvrzují náběžnou hranou DTR. Po zápisu celého bajtu způsobí náběžná hrana RTS přepis přijatých bitů na výstupy Q_0 až Q_7 (podrobnější popis obvodu 74HCT595 najdete například v [1] nebo [5]).

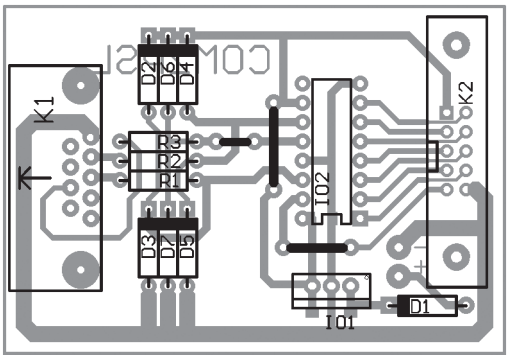
Zbytek zapojení představuje stabilizátor IO_1 (7805) spolu s ochrannou diodou D_1 (1N4007, která brání přepólování) a blokovacími kondenzátory C_1 až C_3 (pro úsporu místa jsou použity kondenzátory v provedení SMD).

Výkres desky plošných spojů je uveden na *obr. 4.2*.

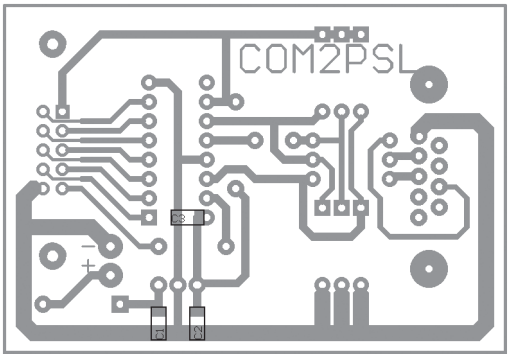


Obr. 4.2 *Výkres desky plošných spojů přípravku* ***COM2PSL (BEN 0171)***

Osazovací plánky jsou dva, protože některé součástky jsou zapájeny ze strany spojů (viz *obr. 4.3* a *obr. 4.4*).



Obr. 4.3 *Osazovací plánek přípravku* ***COM2PSL*** *(strana součástek)*



Obr. 4.4 *Osazovací plánek přípravku* ***COM2PSL*** *(strana spojů)*

Rozpis součástek pro přípravek COM2PSL (cena asi 60 Kč):

C_1 až C_3	CK+ 100N X7R	3 ks
D_1	1N4007	1 ks
D_2 až D_7	1N4148	6 ks
IO_1	7805 + chladič DO1A	1 ks
IO_2	74HCT595	1 ks
K_1	CAN 9 Z 90	1 ks
K_2	PSL10	1 ks
R_1 až R_3	RR 1K	3 ks

4.2 PROPOJOVACÍ KABELY

Pro připojení přípravku **COM2PSL** k sériovému portu počítače je zapotřebí kabel. Lze jej vyrobit z plochého 10žilového vodiče **AWG28-10** (9žilový kabel se totiž neprodává, takže je třeba jednu žílu odstřihnout) délky asi 1 m a samořezných konektorů **CAN 9 V S** a **CAN 9 Z S**. Kabel vložíme označenou žílou na pozici 1 prvního z konektorů a zmáčkeneme ve svěráku. Postup opakujeme pro druhý konektor (označená žíla bude opět na pozici 1).

Konstrukce propojovacího kabelu **PSLKAB** mezi přípravky **COM2PSL** a **ATLCDTX2** byla popsána již v kapitole 3.2.

4.3 TŘÍDA TATLCDTX2COM

Pro ovládání přípravku **ATLCDTX2** přes sériový port (a přípravek **COM2PSL**) byla vytvořena speciální třída **TATLCDTX2COM**. Tato třída vychází ze stejné báze (**ATLCDTX2BASE**), jako dříve popsaná třída **TATLCDTX2LPT** (viz kapitolu 3.4), která zajišťuje ovládání pomocí paralelního portu.

Z tohoto důvodu je popis ovládacího rozhraní stejný, jako na konci kapitoly 3.3.

Implementace

Z nově zavedených metod je důležité především překrytí metody báze **Send** a dále metoda **SendToSIPO**, která zajišťuje tzv. serializaci (odesílání dat do posuvného registru). Komentáře jsou přímo ve zdrojovém textu.

Pro řízení sériového portu byla použita třída **TSPort**, která byla implementována již v [7]. Pro účely této knihy nebyla nijak upravována.

Zdrojový text třídy **TATLCDTX2COM** najdete na doprovodném CD-ROM v adresáři **PROGRAMY\PC\ATLCDTX2COM**.

ATLCDTX2COM.PAS:

```
unit ATLCDTX2COM;
interface
```

```
uses
  ATLCDTX2BASE, SPort, Windows, SysUtils;

type
  //třída pro ovládání displeje přes COM:
  TATLCDTX2COM = class(TATLCDTX2BASE)
  protected
    //vložená instance třídy TSPort:
    Port:TSPort;
    //překrytí metody Send báze:
    procedure Send(
      Data:Byte;RS:TRSType;uDelay:UINT);override;
    //pomocná metoda pro zápis do SIPO:
    procedure SendToSIPO(Data:Byte);
  public
    constructor Create(Number:Word);
    destructor Destroy; override;
end;

implementation

//konstruktor:
constructor TATLCDTX2COM.Create(Number:Word);
begin
  inherited Create;
  //založí instanci TSPort:
  Port:=TSPort.Create(Number);
  try
    //vyvolá inicializaci:
    Init;
  except
    raise
  end;
end;

//destruktor:
destructor TATLCDTX2COM.Destroy;
begin
  //uvolní port:
  Port.Free;
  inherited Destroy;
end;
```

```
//pošle 4 bity:
procedure TATLCDTX2COM.Send(
  Data:Byte;RS:TRSType;uDelay:UINT);
begin
  //posuň data o 3 bity vlevo,
  //E=0, RS dle zadání:
  SendToSIPO((Data shl 3) + 0 + Byte(RS));
  //počkej 1 µs (ustálení RS před E=1):
  Delay(1);

  //posuň data o 3 bity vlevo,
  //E=1, RS dle zadání:
  SendToSIPO((Data shl 3) + 4 + Byte(RS));
  //počkej 1 µs (ustálení dat před E=0):
  Delay(1);

  //posuň data o 3 bity vlevo,
  //E=0, RS dle zadání:
  SendToSIPO((Data shl 3) + 0 + Byte(RS));
  //počkej 1 µs (potvrzení dat):
  Delay(1);

  //počkej na provedení:
  Delay(uDelay);
end;

//pomocná metoda pro zápis do SIPO:
procedure TATLCDTX2COM.SendToSIPO(Data:Byte);
var  Mask:Byte;
     i:Integer;
begin
  //TxD je DIN
  //DTR je CLK
  //RTS je STB
  Mask:=$80;                    //maska pro bity
  Port.RTS:=false;              //STB=0
  Delay(1);

  //odešle 8 bitů:
  for i:=0 to 7 do begin
    Port.DTR:=false;            //CLK=0
    Delay(1);                   //ustálení
    Port.TxD:=(Data and Mask)>0; //data na SI
    Mask:=Mask shr 1;           //změna masky
```

```
    Delay(1);                    //ustálení
    Port.DTR:=true;              //CLK=1
    Delay(1);                    //ustálení
  end;
  //přepis na výstupy:
  Port.RTS:=true;
end;

end.
```

Uživatelský popis

Kromě metod báze **TATLCDTX2BASE** je uživateli dostupný pouze konstruktor a destruktor (více není třeba):

- **constructor Create(Number:Word)** – konstruktor. Při konstrukci instance dojde také k inicializaci displeje. Parametr **Number** udává číslo sériového portu (například pro COM1 zadáme Number = 1),
- **destructor Destroy** – destruktor,
- metody báze: **ClearDisplay**, **CursorHome**, **SetIncrementMode**, **SetDisplayMode**, **SetCGRAMAddr**, **SetDDRAMAddr**, **WriteData**.

4.4 TEST PŘÍPRAVKU

Zdrojový text testovací aplikace je velmi podobný jako v kapitole 3.5, kde bylo řešeno ovládání přes paralelní port. Vlastně se jen změní název používané třídy z **ATLCDTX2LPT** na **ATLCDTX2COM** a pochopitelně také parametry konstruktoru. Zbytek změn zůstává důmyslně ukryt pod ovládacím rozhraním, takže z toho důvodu se výpisy obou příkladů velmi podobají. Příklad najdete na doprovodném CD-ROM v adresáři **PROGRAMY\PC\ATLCDTX2COM**.

LCDTEST.PAS:

```
unit LCDTest;
interface

uses
  ATLCDTX2COM,
  Windows, Messages, SysUtils, Variants,
  Classes, Graphics,
  Controls, Forms, Dialogs, StdCtrls, ExtCtrls;

type
  TTestForm = class(TForm)
    Panel: TPanel;
```

```
    Smaz: TButton;
    ZapniBlikani: TButton;
    VypniBlikani: TButton;
    KurzorDomu: TButton;
    PrvniRadek: TButton;
    DruhyRadek: TButton;
    Edit: TEdit;
    Label1: TLabel;
    Posli: TButton;
    procedure FormCreate(Sender: TObject);
    procedure FormDestroy(Sender: TObject);
    procedure ZapniBlikaniClick(Sender: TObject);
    procedure VypniBlikaniClick(Sender: TObject);
    procedure SmazClick(Sender: TObject);
    procedure KurzorDomuClick(Sender: TObject);
    procedure PrvniRadekClick(Sender: TObject);
    procedure DruhyRadekClick(Sender: TObject);
    procedure PosliClick(Sender: TObject);
  private
    Displej:TATLCDTX2COM;
  end;

var
  TestForm: TTestForm;

implementation
{$R *.dfm}

//OnCreate:
procedure TTestForm.FormCreate(Sender: TObject);
//definice "českých" znaků:
const CGRAMData:array[0..15] of Byte =
  ($0A,$04,$0E,$11,$1F,$10,$0E,$00, //ě
  $02,$04,$11,$11,$0E,$01,$0E,$00); //ý
var i:Integer;
begin
  try
    ClientHeight:=Panel.Height;
    //založ instanci rozhraní displeje:
    Displej:=TATLCDTX2COM.Create(1);
    //pošli data do CGRAM:
    for i:=0 to sizeof(CGRAMData) do begin
      Displej.SetCGRAMAddr(i);
```

```
        Displej.WriteData(CGRAMData[i]);
      end;
      //kurzor zase na začátek DDRAM:
      Displej.CursorHome;
    except
      on E: Exception do begin
        Application.MessageBox(
          PChar(E.Message),
          'ATLCDTX2',
          MB_ICONHAND);
        Application.Terminate;
      end;
    end;
end;

//OnDestroy:
procedure TTestForm.FormDestroy(Sender: TObject);
begin
  Displej.Free;
end;

//Smaž displej:
procedure TTestForm.SmazClick(Sender: TObject);
begin
  Displej.ClearDisplay;
end;

//Kurzor na začátek:
procedure TTestForm.KurzorDomuClick(
  Sender: TObject);
begin
  Displej.CursorHome;
end;

//Zapni blikání:
procedure TTestForm.ZapniBlikaniClick(
  Sender: TObject);
begin
  Displej.SetDisplayMode(true,true,true);
end;

//Vypni blikání:
procedure TTestForm.VypniBlikaniClick(
  Sender: TObject);
```

```
begin
  Displej.SetDisplayMode(true,false,false);
end;

//1. řádek:
procedure TTestForm.PrvniRadekClick(
  Sender: TObject);
begin
  Displej.SetDDRAMAddr(0);
end;

//2. řádek:
procedure TTestForm.DruhyRadekClick(
  Sender: TObject);
begin
  Displej.SetDDRAMAddr($40);
end;

//Pošli text na displej:
procedure TTestForm.PosliClick(
  Sender: TObject);
var i:Integer;
begin
  //procházení řetězce:
  for i:=1 to Length(Edit.Text) do
    //test znaku,
    //nahradí 'ě' 0 a 'ý' 1:
    case Edit.Text[i] of
      'ě':Displej.WriteData(0);
      'ý':Displej.WriteData(1)
      else Displej.WriteData(ord(Edit.Text[i]));
    end;
end;

end.
```

Funkce tlačítek:

- **Smaž displej** – smaže displej (to zároveň způsobí přesun kurzoru na začátek displeje),
- **Kurzor na začátek** – přesune kurzor na začátek (bez smazání displeje),
- **Zapni blikání/vypni blikání** – zapne/vypne blikání kurzoru. Po stisku tlačítka **Zapni blikání** se tedy objeví kurzor a začne blikat. Po stisku tlačítka **Vypni blikání** kurzor zmizí,