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

10 SÉRIOVÝ PŘENOS DAT

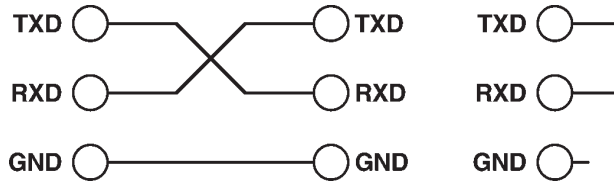
V prvých kapitolách této knihy jsme používali sériový port úplně jinak, než jeho vývojáři zamýšleli. Vlastní účel tohoto portu byl přenos dat přes modemy. Označení jako „Ring Indicator“ (indikátor zvonění) nebo „Data Terminal Ready“ (připravenost koncového zařízení) o tom ještě svědčí. V průběhu času se port stále více osvědčoval v jiných oblastech. Největší rozšíření nalezlo připojení počítačové myši na RS232. Původně si asi nikdo netroufal ani snít o tom, že na RS232 jednou bude pracovat zařízení s vlastním mikrokontrolérem, které dokonce z portu odebírá napájecí napětí.

Vlastní účel sériového portu je přenos dat na linkách TxD a RxD. Data jsou hardwarem portu převáděna na sériový tok bitů a vysílána na cestu. Na přijímací straně vyrábí druhý port ze sériového toku dat opět paralelní datové bajty, které mohou být dále zpracovávány počítačem. To celé je příkladem zdařilé dělby práce mezi softwarem a hardwarem. Vyslání jednoho znaku přes sériový port vyžaduje ve srovnání s jinými operacemi mnoho času. Jedná se zde o milisekundy, zatímco jinak jde spíše o mikrosekundy. Port v PC proto obsahuje speciální hardwarovou součástku, univerzální sériový vysílač/přijímač (Universal Serial Receiver Transmitter, UART), který je jakousi telegrafní stanicí počítače PC. Sem se předávají zprávy a samostatně se vysílají. A naopak UART samostatně přijímá zprávy a ukládá je tak, aby si je program mohl jedinou akcí vyzvednout.

Typickou úlohou pro sériový port je přenos dat z jednoho počítače do druhého. Většinou se jedná o textová data, ale také programy, obrazy, atd. Posílají se i jednotlivé bajty, tedy skupiny osmi bitů. Port je však možno nastavit i tak, že jednotlivé znaky mají jen 7 bitů. Podporovány jsou dokonce i pětibitové znaky, což je vzpomínka na doby starých dálnopisů, které mechanicky prováděly tentýž úkol, jen mnohem pomaleji a hlučněji.

10.1 Propojení nulovým modemem

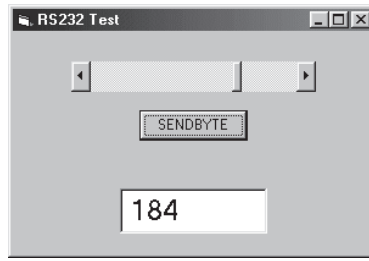
V prvním pokusu propojíme dva počítače PC, aby bylo možno posílat znaky z jednoho do druhého a naopak. Je k tomu nutné překřížené propojení linek TxD a RxD. Takovému propojení se říká nulový modem, protože zde vlastně žádný modem není použit. Zatímco na velké vzdálenosti se používají telefonní linky a modemy, na vzdálenost několika metrů vystačíme s normálními kabely. Pokus ostatně můžeme provádět i s jediným počítačem, který si přes sériový port sám posílá data.



Obr. 10.1 *Propojení nulovým modemem a test pomocí jediného portu*

Pro sériový přenos dat obsahuje knihovna PORT.DLL proceduru SENDBYTE a funkci READBYTE. Dalším předpokladem pro úspěšný přenos dat je, aby porty obou partnerů byly inicializovány se stejnými parametry. Pomocí OPENCOM „COM2:1200:N, 8,1“ se sjednotíme na těchto parametrech: přenosová rychlost 1200 bitů za sekundu (1200 baudů), žádný paritní (kontrolní) bit, osm bitů na znak a jeden stopbit. Těmito parametry se budeme ještě podrobněji zabývat.

Program v prvním příkladu přenáší při každém příkazu jen jeden bajt. Partnerovi se v tomto případě bude předávat poloha posuvníku jako číselná hodnota mezi 0 a 255. Obráceně se každý přijatý bajt zobrazí v textovém okně.



Obr. 10.2 *Vysílání a příjem jednotlivých bajtů*

```
Private Sub Form_Load()
 OPENCOM "COM2"
End Sub

Private Sub Form_Unload(Cancel As Integer)
 CLOSECOM
End Sub

Private Sub Command1_Click()
 d = HScroll1.Value
 SENDBYTE d
```

```
End Sub

Private Sub Timer1_Timer()
  d = READBYTE
  If d > -1 Then Text1.Text = Str$(d)
End Sub
```

Listing 10.1 *Vysílání a příjem jednotlivých bajtů (rsbyte1.vbp)*

10.2 Univerzální terminálový program

Pro práci s portem RS232 často potřebujeme terminálový program, který zobrazuje přijatá data a umožňuje výstup dat. Pod Windows již jeden takový terminál máme – Hyperterminal. Pomocí něho lze např. přenášet data přes modem. Zde však přesto vyvineme terminálový program, který je možno co nejuniverzálněji používat speciálně pro obecné vývojové úlohy.

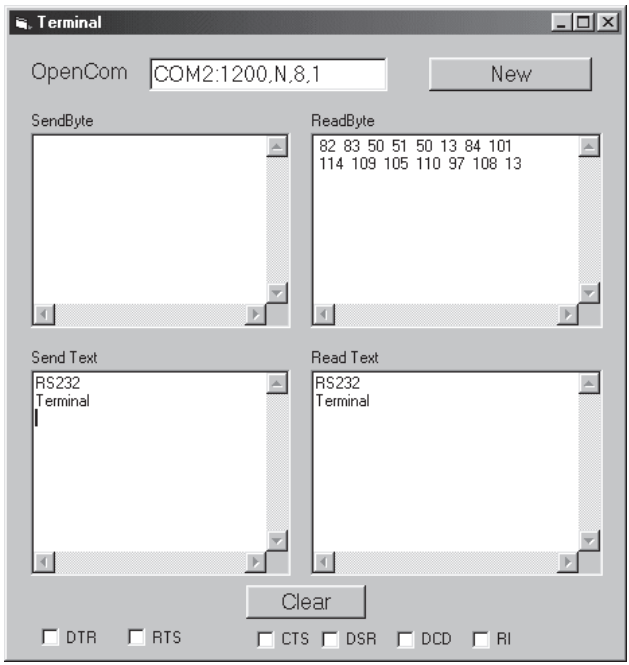
Uživatel může přímo zadávat řetězec pro otevření portu a otevřít port s požadovanými daty. V těch případech, kdy přenosové charakteristiky zařízení nejsou přesně známy, je možno je rychle vyzkoušet. Po každé změně otvíracího řetězce a stisknutí tlačítka „New“ se port zavře a otevře se s novými parametry.

Tatáž sériová data je možno interpretovat dvěma zcela rozdílnými způsoby. První způsob se jedná o řídicí data v bajtovém formátu, tedy jednoduše o číselné hodnoty v rozsahu 0 až 255. V tomto případě hovoříme o binárním formátu. Příkladem byl přenos polohy posuvníku v předchozím odstavci. Tento režim využívají četné měřicí přístroje a zařízení jako počítačové myši nebo jiná ukazovací zařízení.

Druhý způsob interpretuje přenášená data jako text. Textové znaky jsou také vždy bajty. *Obr. 10.3* ukazuje přenos textových znaků. Linky TxD a RxD byly pro tento test přímo propojeny, takže zadávaný vysílaný text byl týmž portem opět přijímán. Znaková sada ASCII, jak známo, definuje, jakou číselnou hodnotou je reprezentován každý textový znak. Například číselná hodnota 65 odpovídá velkému A, 49 číslici 0. Ten, kdo přenáší data, se obvykle nestará o ASCII kód. Jsou však i případy, u nichž má podrobnější posuzování smysl. Máme-li např. za úkol dotazovat se programem na stav měřicího přístroje vybaveného sériovým portem, často se v návodu k obsluze ocitneme v džungli nejasných informací. Textová a binární data se často míchají dohromady.

Terminálový program podle *Listingu 10.2* podporuje od počátku obě interpretace znaků. Přicházející data se současně zobrazují jako bajty i jako tex-

tové znaky. Pak velmi rychle poznáme, co je míněno. Také při vysílání máme možnost použít oba formáty. Ve spodním okně se jednoduše naťuká text, který se znak po znaku přenáší. Naproti tomu v horním okně se zadávají číselné hodnoty od 0 do 255 a potvrzují se klávesou Enter.



Obr. 10.3 *Přenos textu pomocí terminálového programu*

```
Dim n

Private Sub Check1_Click()
 If Check1.Value Then DTR 1 Else DTR 0
End Sub

Private Sub Check2_Click()
 If Check2.Value Then RTS 1 Else RTS 0
End Sub

Private Sub Command1_Click()
 CLOSECOM
 OPENCOM Text1.Text
 DTR 0
```

```
 Check1.Value = False
 RTS 0
 Check2.Value = False
End Sub

Private Sub Command2_Click()
 Text2.Text = " "
 Text3.Text = " "
 Text4.Text = " "
 Text5.Text = " "
End Sub

Private Sub Form_Load()
 OPENCOM "COM2:1200,N,8,1"
 DTR 0
 Check1.Value = False
 RTS 0
 Check2.Value = False
End Sub
Private Sub Form_Unload(Cancel As Integer)
 CLOSECOM
End Sub
Private Sub Text2_Change()
 If Text2.Text > "" Then Char = Asc(Right$(Text2.Text, 1))
 If Char = 10 Then
   Byt$ = (Right$(Text2.Text, 5))
   dat = Val(Byt$)
   dat = dat And 255
   SENDBYTE dat
 End If
End Sub

Private Sub Text4_KeyPress(KeyAscii As Integer)
 SENDBYTE KeyAscii
End Sub

Private Sub Timer1_Timer()
 Do
   dat = READBYTE
   If dat > -1 Then
     Text3.Text = Text3.Text + Str$(dat) + " "
     n = n + 1
 If n = 8 Then n = 0: Text3.Text = Text3.Text + Chr(13) + Chr(10)
   Text3.Refresh
```

```
  End If
  If (dat > -1) And (dat <> 13) And (dat <> 19) Then
  Text5.Text = Text5.Text + Chr$(dat)
  End If
  If dat = 13 Then
    Text5.Text = Text5.Text + Chr$(13) + Chr$(10)
    Text5.Refresh
  End If
  Loop Until dat = -1
  If CTS = 1 Then Check3.Value = 1 Else Check3.Value = 0
  If DSR = 1 Then Check4.Value = 1 Else Check4.Value = 0
  If DCD = 1 Then Check5.Value = 1 Else Check5.Value = 0
  If RI = 1 Then Check6.Value = 1 Else Check6.Value = 0
End Sub
```

Listing 10.2 *Terminál RS232(Terminal.vbp)*

Při vysílání textu zadávaného ve spodním okně (Text4) se beze změny přenášejí znaky, které odpovídají stisknutým tlačítkům. Visual Basic dodává hodnotu znaku pro přenos v proměnné KeyAscii, kterou je možno přímo předat proceduře SENDBYTE. Také klávesa Enter dává znak, a to znak posuv o řádek (Line Feed, LF) s hodnotou 10.

Interpretace číselných hodnot v horním textovém okně (Text2) je trochu pracnější. Program musí nejprve počkat, až uživatel potvrdí zadání čísla klávesou Enter. Text pak na posledním místě obsahuje znak LF = 10. Potom se oddělí poslední čtyři znaky a převedou se z textu na číselné hodnoty. Ty musí být v rozsahu 0 až 255. K zabezpečení proti chybnému vstupu větších čísel se předaná hodnota omezí pomocí „AND 255“.

Příjem znaků probíhá v proceduře timeru. Každých 100 ms se zjišťuje, zda přišel znak. Aby nedošlo k „zácpě“, všechny příchozí znaky se pokaždé načtou a zpracují, dokud READBYTE nedá výsledek –1, tj. dokud nenarazí na prázdný buffer. Přijaté znaky se zobrazí v obou výstupních oknech Text3 a Text5.

Při výstupu v číselném formátu v textovém okně Text3 jsou číselné hodnoty uspořádány do řádků vždy po osmi bajtech. Potom následuje zlom řádku se speciálními znaky Line Feed (LF, ASCII 10) a Carige Return (CR, ASCII 13).

Výstup textu v okně Text5 musí rovněž zvláštním způsobem zpracovávat konec řádku. Speciální znak LF se zachytí a nahradí znaky LF a CR. Všechny ostatní znaky se přímo přidají k textu. Výsledkem je zobrazení textu přesně tak, jak byl zadán ve vstupním okně.