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

9.3 SÉRIOVÝ PORT MIKROKONTROLÉRU 8051

Interní sériový port mikrokontroléru 8051 umožňuje jednoduché a bezproblémové spojení mikrokontroléru s počítačem PC. Protože je port navržen jako autonomní hardwarový UART, je procesor volný pro jiné úkoly.

Pro přesnější pochopení programování je třeba se podívat na registr speciálních funkcí sériového portu. K inicializaci portu se musí naplnit vhodnými parametry registr SCON.

SCON (SFR 98h)

7	6	5	4	3	2	1	0
SM0	SM1	SM2	REN	TB8	RB8	T1	Ri

SM0, SM1: Bity sériového režimu (serial mode) určují pracovní režim:
- 0 0 režim 0: 8bitový posuvný registr
- 0 1 režim 1: 8bitový UART, přenosová rychlost určená časovačem Timer 1
- 1 0 režim 2: 9bitový UART, 375 kBaudů při 12 MHz
- 1 1 režim 3: 9bitový UART, přenosová rychlost určená časovačem Timer 1

SM2: vícepočítačový režim
REN: Receiver Enable, povolení přijímače
TB8: Transmitted bit 8, pro 9bitový režim
RB8: Received bit 8, pro 9bitový režim
TI: Transmitter Interrupt, 1 při úspěšném ukončení přenosu:
RI: Receiver Interrupt, 1 po příjmu znaku

Datový registr sériového portu je SBUF (SFR 099h). Ve skutečnosti se za SBUF skrývají dva registry, a to vysílací (výstupní) datový registr a přijímací (vstupní) datový registr. Vysílání se prostě zahájí zápisovým přístupem na registr SBUF. Naopak přijatý byte může být ze SBUF přečten. V 9bitovém režimu je vždy třeba brát ohled na 9. bit (TB8 nebo TR8) v registru SMOD.

Ve většině případů se používá pracovní režim 1, tedy 8bitový UART s přenosovou rychlostí danou časovačem 1. Osmibitový datový tok je zahajován startbitem a uzavřen stopbitem. Přenosová rychlost je 1/16 (SMOD = 1) nebo 1/32 (SMOD = 0) frekvence přeběhů časovače 1. SMOD je bit s nejvyšší váhou v registru PCON (Power Control, SFR 87h), který je jinak určen k řízení režimu power-down (snížení napájecího výkonu).

PCON (SFR 87h)

7	6	5	4	3	2	1	0
SMOD	–	–	–	GF1	GF2	PD	IDL

SMOD: 1 = nízká přenosová rychlost , 0 = vysoká přenosová rychlost
GF1, GF2: Volně použitelná návěští (flags)
PD: Režim snížení příkonu (power-down)
IDL: Režim dynamického zastavení (idle modus)

Výroba hodinového signálu pro sériový port vyžaduje nejprve naprogramování časovače 1 (timer 1). Oba časovače obvodu 8051 mají každý dva 8bitové čítací registry, které je možno plnit a přečíst.

TL0 (SFR 8Ah): časovač 0, nižší byte
TH0 (SFR 8Ch): časovač 0, vyšší byte
TL1 (SFR 8Bh): časovač 1, nižší byte
TH1 (SFR 8Dh): časovač 1, vyšší byte

Vlastnosti časovačů jsou řízeny registry TCON (SFR 88h) a TMOD (SFR 89h). Pomocí TMOD se nastavuje pracovní režim.

TMOD (SFR 89h)

7	6	5	4	3	2	1	0
Gate	C/T	M1	M0	Gate	C/T	M1	M0
Čítač 1				Čítač 0			

Gate: Příslušný čítač se uvolňuje vývodem INT0, popř. INT1.
C/T: 0: časovač, 1: čítač
M1 a M0: Pracovní režim
0 0 13bitový časovač/čítač
0 1 16bitový časovač/čítač
1 0 8bitový časovač/čítač, automatické plnění
1 1 Jen pro čítač 0: dva oddělení 8bitové čítače

Časovač 1 má jako volnoběžný čítač načítat interní hodinové impulsy 8051. Nastavíme jej proto jako časovač v režimu 2. Při každém přeběhu se čítač znovu naplní předem nastavenou hodnotou.

Každý čítač se startuje svým bitem TR v registru TCON. TCON obsahuje i flagy (příznaky) přeběhů obou časovačů a čtyři řídicí bity k řízení přerušení pomocí vývodů INT0 a INT1.

TCON (SFR 88h)

7	6	5	4	3	2	1	0
TF1	TR1	TF0	TR0	IE1	IT1	IE0	IT0

TF1: příznak přeběhu pro časovač 1
TR1: zapnutí časovače 1
TF0: příznak přeběhu pro časovač 0
TR0: zapnutí časovače 0
IE1: přerušení na INT1 se spouštěním hranou
IT1: příznak přerušení pro INT1
IE0: přerušení na INT0 se spouštěním hranou
IT0: příznak přerušení pro INT0

Pro provoz čítače ve funkci generátoru přenosové rychlosti je pouze třeba nastavit spouštěcí bit TR1. Ten je bitově adresovatelný a dosažitelný pod bitovou adresou 8 Eh. Nejprve se však musí při zastaveném čítači nastavit potřebný dělicí poměr. V režimu 2 obsahuje vyšší byte čítače TH1 hodnotu pro opakované plnění, která se pak při každém přeběhu znovu zavede do vlastního registru čítače TL1. Pro přenosovou rychlost 9600 baudů je při nahozeném bitu SMOD potřebná hodinová frekvence 16 × 9,6 kHz = 153,6 kHz. Protože čítač načítá interní hodinovou frekvenci příkazů rovnou 1/12 frekvence krystalu, je při frekvenci krystalu rovné 11,0592 MHz třeba nastavit dělení číslem 6:

11059,2 kHz / 12 / 6 / 16 = 9,6 kHz

Dělicí faktor 6 dostaneme při hodnotě opakovaného plnění čítače 256 – 6 = 250 = FAh, protože se čítá nahoru.

Listing COM51.ASM ukazuje kompletní program. K inicializaci se nejprve příslušné registry naplní svými řídicími parametry. Hlavní program pak používá již jen dva podprogramy SEND (vysílání) a EMPF (příjem).

Přijímací rutina EMPF se nejprve dotazuje na bit RI, dokud nepřijde znak. Potom se načte přijímací buffer SBUF a RI se shodí. Vysílací rutina SEND se naopak dotazuje na bit TI, dokud se úplně nezpracuje poslední vysílaný znak. Pak se vysílaný byte zapíše do vysílacího bufferu SBUF a tím se nastartuje vysílač UART.

```
;Seriovy port mikrokontroleru 8051       (COM51.ASM)
;11,059MHz, 9600 Baud
#include 8051.H
         .org 0000H

ANFANG   mov   SP,#60H          ;nastavit stackpointer
INIT     clr   TR1              ;zastavit Timer 1
         mov   TH1,#0FAH        ;6 do prebehu: 9600 Baud
         mov   TL1,#0FAH
         anl   TMOD,#0FH        ;Timer1: 8 bitů, auto-reload
         orl   TMOD,#20H
         setb  TR1              ;spustit Timer 1
         mov   SCON,#50H        ;InitRS232
         setb  TI
         orl   PCON,#80H        ;SMOD=1

NEXT     acall EMPF
         mov   P1,A             ;vystup portu
         anl   P3,#0EFH         ;strob-impuls na P3.4
         orl   P3,#10H
         Mov   A,P1             ;cist stav portu
         acall SEND
         sjmp  NEXT

EMPF     jnb   RI,EMPF
         mov   A,SBUF
         clr   RI
         ret
```

```
SEND      jnb   TI,SEND
          clr   TI
          mov   SBUF,A
          ret
          .end
```

Listing 9.4 Použití sériového portu
CD:\Kniha\Kap09\TasmKap9\COM51.ASM

Každý přijatý byte se předá na port 1. Následuje krátký strobovací impuls na P3.4 (linka T0). Potom program přečte stav portu 1 a vyšle přečtený byte přes sériový port zpět. Tak je realizován paralelně-sériový převodník, který je možno použít buď jen pro výstupy portu, jen pro čtecí přístupy na port nebo pro smíšený provoz. Všechny vstupní linky musí být při vysílání nahozeny na vysokou úroveň a tím být ve stavu s vysokou impedancí. Vysílaná hodnota 255 (FFh) například způsobí, že celý port bude k dispozici jako vstupní port a bude se provádět čtení. Funkci programu je možno otestovat pomocí programů pro použití se sériovými vysílači a přijímači představených již v kapitole 4.

9.4 KONTROLÉROVÝ UART

Mikrokontrolérový UART byl představen v praktické aplikaci již v kapitole 4.5. Zde uvedeme jen potřebný pracovní program.

Jako náhrada za UART 6042 musí mít kontrolér oddělené vývody pro vstup a výstup. Port 3 se částečně používá pro sériovou komunikaci a není volně k dispozici. Protože však není nutná žádná paměť RAM, zůstává celá datová a adresní sběrnice volná. Tím je opět k dispozici port 0 a port 2. Port 0 se svými elektrickými parametry liší od ostatních portů, protože nemá žádné interní zdvihací odpory. Pro použití jako výstup by bylo nutno zapojit externí zdvihací odpory. Chybějící zdvihací obvody mohou být pro digitální vstupy nevýhodou, protože není možno přímo připojovat kontakty nebo spínače. Z toho důvodu zde jako vstupní port použijeme port 1. Port 2 je výstupní port mikrokontrolérového UART. I zde je nutno pro některé aplikace zapojovat zdvihací odpory, protože porty dodávají jen malý proud. Výstupní budič ULN2803 (viz kap. 4.6) např. vyžaduje dodatečné odpory 3,3 kΩ proti Vcc, aby byl k dispozici dostatečný proud.

Mikrokontrolérový UART je možno pomocí propojek, zapojených na P3.2 až P3.4 proti zemi, nakonfigurovat celkem pro čtyři přenosové rychlosti. Bez propojek se použije 1200 baudů, na P3.2 lze zvolit 9600 baudů, na P3.3 19200 a na P3.4 nejvyšší možnou přenosovou rychlost 57600 baudů. P3.5 slouží jako strobovací výstup se zápornými impulsy pro každý přijatý byte.

```
;UART-Controller 8051        (UART51.ASM)
;11,059MHz, 9600 Baud
#include 8051.H
          .org 0000H

ANFANG    mov   SP,#60H          ;nastavit stackpointer
INIT      clr   TR1              ;zastavit Timer 1
B1200     mov   TH1,#0D0H        ;delic 48, 1200 baudu
```

```
          mov   TL1,#0D0H
B9600     jb    P3.2,B19200      ;Jumper 9600?
          mov   TH1,#0FAH        ;delic 6, 9600 baudu
          mov   TL1,#0FAH
B19200    jb    P3.3,B57600      ;Jumper 19200?
          mov   TH1,#0FDH        ;delic 3, 19200 baudu
          mov   TL1,#0FDH
B57600    jb    P3.4,INIT2       ;Jumper 57600?
          mov   TH1,#0FFH        ;delic 1, 57600 baudu
          mov   TL1,#0FFH
INIT2     anl   TMOD,#0FH        ;Timer1: 8-Bit-Auto-Reload
          orl   TMOD,#20H
          setb  TR1              ;spustit Timer
          mov   SCON,#50H        ;InitRS232
          setb  TI
          orl   PCON,#80H        ;SMOD=1

NEXT      acall EMPF
          mov   P2,A             ;vystup portu Port 2
          clr   P3.5             ;strobovaci impuls na P3.5
          setb  P3.5
          Mov   A,P1             ;cist stav portu 1
          acall SEND
          sjmp  NEXT

EMPF      jnb   RI,EMPF
          mov   A,SBUF
          clr   RI
          ret

SEND      jnb   TI,SEND
          clr   TI
          mov   SBUF,A
          ret
          .end
```

Listng 9.5 Mikrokontrolérový UART (Uart51.asm)
CD:\Kniha\Kap09\TasmKap9\Uart51.asm

9.5 PARALELNÍ PŘÍSTUPY NA SBĚRNICI

K buzení periferních zařízení na adresní a datové sběrnici procesoru přes sériový port jsme v kapitole 5 představili paralelní sběrnici rozhraní (Interface Bus). Použitý řadič sběrnice jsme pokládali za speciální integrovaný obvod, aniž bychom se zabývali programem kontroléru. Zde nyní představíme použitý software.

Program Pibus.asm byl vyvinut pomocí vývojového systému ES52-Flash. Všechny vývody sběrnice jsou na 40pólové kontaktní liště na kraji desky, takže k pohodlnému propojení s přídavnými zařízeními je možno použít plochý kabel. Mikrokontrolér AT89S8252

je ale možno naprogramovat i ve vývojovém systému a pak jej umístit na speciální desce spolu s přídavnými zařízeními. Vyvinutý program je mimo to vhodný i pro levnější mikrokontrolér 8751.

Obvykle se kontrolér 8051 obrací na sběrnici prostřednictvím příkazů MOVX. Signály $\overline{WR}$ a $\overline{RD}$ se přitom generují automaticky. Data a spodních osm adres se multiplexovaně vydávají na datovou sběrnici, kdežto horních osm adresních signálů se vytváří staticky na portu 2. Mohli bychom použít adresní střadač, např. 74HC573, který je k dispozici pro buzení RAM na vývojové desce. Abychom mohli tuto součástku ušetřit a realizovat opravdové jednočipové řešení, vytváří se všechny signály pomocí přístupů na porty. Tím máme také možnost vyrábět i další předdekódované povolovací signály s požadovaným časováním. Všechny signály jsou pomalejší než signály generované pomocí MOVX, to však v důsledku spojení přes sériový port, které je stejně pomalejší, nehraje žádnou roli.

Port 2 slouží jako adresní sběrnice pro dolních sedm adresních linek A0 … A6. Přijatý adresní byte je celý přiveden na port, takže se na P2.7 objeví i bit směru dat. Port 0 slouží jako datová sběrnice. Musí se ovšem brát v úvahu, že port nemá obvyklé zdvihací (pullup) odpory. V závislosti na aplikaci mohou tedy být nutné externí zdvihací odpory. Všechny řídicí signály se vytvářejí na portu 3. P3.7 a P3.6 slouží jako linky $\overline{RD}$ a $\overline{WR}$. Protože na portu 3 jsou i sériové datové linky, jsou k dispozici ještě čtyři linky jako povolovací linky Y0 až Y3. Všechny signály jsou v klidu na úrovni High.

```
;Seriovy port kontroleru 8051        (PIbus.ASM)
;11,059MHz, 9600 baudu
;Rozhrani s paralelni sbernici
#include 8051.H
         .org 0000H

ANFANG   mov   SP,#60H          ;nastavit stackpointer
                                          (Anfang = zacatek)
INIT     clr   TR1              ;zastavit Timer 1
         mov   TH1,#0FAH        ;6 do prebehu: 9600 baudu
         mov   TL1,#0FAH
         anl   TMOD,#0FH        ;Timer1: 8-Bit-Auto-Reload
         orl   TMOD,#20H
         setb  TR1              ;spustit Timer
         mov   SCON,#50H        ;InitRS232
         setb  TI
         orl   PCON,#80H        ;SMOD=1

NEXT     acall EMPF             ;prijem adresy
         mov   r3,A             ;ulozeni adresy
         jb    ACC.7,Lesen      ;bit cteni?  (lesen = cteni)
         mov   P2,A             ;staticka adresa
         acall EMPF             ;prijem dat
         mov   P0,A             ;vystup dat
         mov   A,r3             ;adresa
         acall CS               ;/CS na Port 1
         clr   P3.6             ;/WR = 0
         nop
```

```
          setb  P3.6            ;/WR = 1
          mov   P3,#255         ;vsechny /CS = 1
          sjmp  NEXT

Lesen     mov   DPL,A           ;A0...A6     (lesen = cteni)
          mov   DPH,A           ;+ A8...A14
          mov   P2,A            ;staticka adresa
          mov   P0,#255         ;datova sber. s vys. impedanci
          mov   A,r3            ;adresa
          acall CS              ;/CS na Port 1
          clr   P3.7            ;/RD = 0
          nop
          mov   A,P0            ;cist data
          setb  P3.7            ;/RD = 1
          mov   P3,#255         ;vsechny /CS = 1
          acall SEND            ;vyslat vysledek
          sjmp  NEXT

CS        anl   A,#127          ;oblast 0...127
          mov   P1,#255         ;vsechny linky /CS 1
          anl   A,#0F0h         ;oblasti 16 adres
          jnz   Y1              ;0...15
          clr   P3.2            ;Y0 = 0
Y1        subb  A,#16           ;dalsi skupina
          jnz   Y2              ;16...31
          clr   P3.3            ;Y1 = 0
Y2        subb  A,#16           ;dalsi skupina
          jnz   Y3              ;32...47
          clr   P3.4            ;Y2 = 0
Y3        subb  A,#16           ;dalsi skupina
          jnz   Y4              ;48...63
          clr   P3.5            ;Y3 = 0
Y4        ret

EMPF      jnb   RI,EMPF                      (empfangen = prijimat)
          mov   A,SBUF
          clr   RI
          ret

SEND      jnb   TI,SEND
          clr   TI
          mov   SBUF,A
          ret
          .end
```

Listing 9.6 Řídicí program pro rozhraní s paralelní sběrnicí (Pibus.asm) CD:\Kniha\Kap09\TasmKap9\PIbus.ASM

Hlavní program pracuje v nekonečné smyčce, v níž se vždy nejprve přijímá adresní byte. V závislosti na bitu s nejvyšší váhou se provede buď operace zápisu nebo po skoku na „Lesen“ (čtení) operace čtení. Společně využívaný podprogram „CS“ zajišťuje předdekódování čtyř adresních oblastí a vytvoření signálu Y. Při zápisu se jako druhý

byte přijme datový byte a vydá se na P0. Teprve potom se vytvoří signál CS a $\overline{WR}$. Výpočet adresních signálů po příjmu datového bytu vyžaduje dočasné uložení přijatého adresního bytu.

Při čtení bytu ze sběrnice se nejprve vygeneruje adresa, potom signál $\overline{RD}$ a případný signál Y. Tím obvod přídavného zařízení přivede na sběrnici svá data, která se přečtou čtecím přístupem mikrokontroléru na port 0.

Program při opakovaných zápisových přístupech provádí prakticky průběžně přepínání mezi adresními a datovými byty. V principu při tom může dojít k záměně například v důsledku programového přerušení na straně PC. Při novém startu by se datový byte interpretoval jako adresa. Tomu však lze snadno zabránit provedením prázdného (dummy) čtecího přístupu při startu programu. Protože při čtení je v každém směru jen jeden byte, je po něm kontrolér v definovaném stavu. *Listing 9.7* ukazuje jednoduchý příklad programu v jazyce Visual Basic. Je možno popsat a načítat všechny adresy v oblasti 0 až 127. Další příklady jsou uvedeny v kapitole 5.

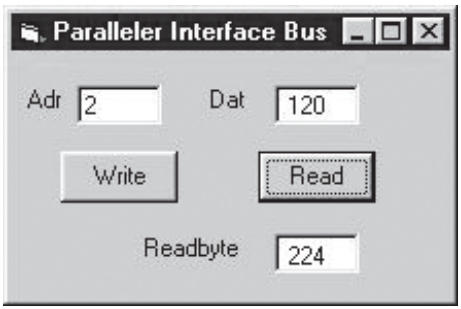
```
Private Sub Form_Load()
  OPENCOM "COM2:9600,N,8,1"
  SENDBYTE 128
  Dummy = READBYTE
  SENDBYTE 128
  Dummy = READBYTE
End Sub

Private Sub Form_Unload(Cancel As Integer)
  CLOSECOM
End Sub

Private Sub Read_Click()
  SENDBYTE Val(Text1.Text) + 128 'cteni adresy
  Text3.Text = Str$(READBYTE)
End Sub

Private Sub Write_Click()
  SENDBYTE Val(Text1.Text)    'Adresa
  SENDBYTE Val(Text2.Text)    'Data
End Sub
```

Listing 9.7. Zápis a čtení přes sběrnici



Obr. 9.7 Přístup na sběrnici programem v jazyce Visual Basic