ChnComM

JEDNOTKA SERIOVÉ KOMUNIKACE POMOCÍ MODEMU NA RS232 S OBVODEM I8250

Příručka uživatele a programátora

***SofCon*®spol. s r.o.**
Střešovická 49
162 00 Praha 6
tel/fax: +420 220 180 454
E-mail: sofcon@sofcon.cz
www: http://www.sofcon.cz

Verze dokumentu 1.10

Informace v tomto dokumentu byly pečlivě zkontrolovány a SofCon věří, že jsou spolehlivé, přesto SofCon nenese odpovědnost za případné nepřesnosti nebo nesprávnosti zde uvedených informací.
SofCon negarantuje bezchybnost tohoto dokumentu ani programového vybavení, které je v tomto dokumentu popsané. Uživatel přebírá informace z tohoto dokumentu a odpovídající programové vybavení ve stavu, jak byly vytvořeny a sám je povinen provést validaci bezchybnosti produktu, který s použitím zde popsaného programového vybavení vytvořil.
SofCon si vyhrazuje právo změny obsahu tohoto dokumentu bez předchozího oznámení a nenese žádnou odpovědnost za důsledky, které z toho mohou vyplynout pro uživatele.

Datum vydání: 16.05.2003
Datum posledního uložení dokumentu: 16.05.2003
(Datum vydání a posledního uložení dokumentu musí být stejné)

Obsah :

1.O dokumentu 5
1.1. Revize dokumentu 5
1.2. Účel dokumentu 5
1.3. Rozsah platnosti 5
1.4. Související dokumenty 5
2.Termíny a definice 5
3.Úvod 6
4.Popis konstant a typů 6
5.Objekty 6
5.1. tChnMod 6
5.1.1. Položky 6
5.1.2. Metody 7
5.1.2.1. Init konstruktor 7
5.1.2.2. ChInitParam konstruktor 7
5.1.2.3. ChSetOneParam funkce 7
5.1.2.4. ChGetParam funkce 8
5.1.2.5. ChInitHW procedura 8
5.1.2.6. ChDoneHW procedura 8
5.1.2.7. ChSendHW procedura 9
5.1.2.8. ChSendFlushHW procedura 9
5.1.2.9. ChSendTickHW procedura 9
5.1.2.10. ChSendReadyHW procedura 9
5.1.2.11. ChSetModem procedura 9
5.1.2.12. ChGetModem funkce 9
5.2. tAddChnComM 9
5.2.1. Metody 9
5.2.1.1. ChInit funkce 9
6.Příklad 10

1. O dokumentu

1.1. Revize dokumentu

Verze dokumentu	Verze SW	Autor	Datum vydání	Popis změn
1.00	1.XX	Wi		První vydání
1.10	4.XX	Tu	16.05.2003	Úprava dokumentu dle ISO9000. Doplněná položka CH_STickHW. Doplněné metody ChSendTickHW a ChSendReadyHW.

1.2. Účel dokumentu

Tento dokument slouží jako popis jednotky sériové komunikace pomocí modemu na RS232 s obvodem i8250.

1.3. Rozsah platnosti

Určen pro programátory a uživatele programového vybavení SofCon.

1.4. Související dokumenty

Pro čtení tohoto dokumentu je potřeba seznámit se s manuálem ChnVirt, ChnMod a ChnTypes.

Popis formátu verze knihovny a souvisejících funkcí je popsán v manuálu LibVer.

2. Termíny a definice

Používané termíny a definice jsou popsány v samostatném dokumentu Termíny a definice.

3. Úvod

Knihovna ChnComM definuje objekt **tChnComM**, jehož instance vytváří fyzickou vrstvu v komunikačním kanálu tvořenou sériovým rozhraním prostřednictvím telefonního modemu připojeném na RS232 s obvodem i8250. Ke své činnosti využívá přerušovacího systému počítače. Znaky přicházející po komunikační lince jsou v přerušovací proceduře ukládány do vstupního kruhového vyrovnávacího bufferu, z kterého jsou předávány metodami **ChReceive** a **ChReceiveChar** k dalšímu zpracování. Znaky určené k odeslání jsou zasílány z výstupního bufferu rovněž s využitím přerušovacího systému.

Objekt **tChnComM** je dědicem od komunikačního objektu **tChnMod**, to znamená, že definuje pouze metody poplatné použitému hardware. Inicializaci a nastavení modemu, navazování a rušení spojení, přenos a příjem dat zajišťují metody rodičovského objektu tChnMod, které se nepředefinovávají.

Knihovna rovněž definuje objekt **tAddChnComM**, který je dědicem od rodičovského objektu tAddChnVirt. Objekt tAddChnComM zajistí, aby daný komunikační objekt (objekt tChnComM) byl k aplikaci připojen a popřípadě zajistí vytvoření instance tohoto objektu. Po přilinkování této jednotky do aplikace (příkazem "uses ChnComM"), se jméno objektu tChnComM automaticky vloží do seznamu správců komunikačních objektů pro případné použití.

Protože je objekt **tChnComM** dědicem rodičovského komunikačního objektu **tChnMod**, jsou v této příručce popsány jen odlišnosti a speciality pro tento druh sériové komunikace. Ostatní naleznete v příručce **ChnMod**. Některé použité konstanty a typy jsou předdefinované v jednotce **ChnTypes**.

4. Popis konstant a typů

```
cVerNo = např. $0251; { BCD formát }
cVer   = např. '02.51,07.08.2003';
```

Číslo verze jednotky v BCD tvaru a v textové podobě včetně datumu změny.

```
cName     = 'COMM';
```

Konstanta **cName** definuje jméno komunikačního objektu **tChnComM**.

5. Objekty

5.1. tChnMod

5.1.1. Položky

```
CH_Addr     : Word;
```

Položka **CH_Addr** obsahuje adresu sériového komunikačního adaptéru i8250.

```
CH_Irq      : tIrq;
```

Položka **CH_Irq** obsahuje číslo přerušení IRQ, na kterém adaptér i8250 žádá o přerušení. Může nabývat hodnot 0 až 7.

```
CH_Rate     : tRate;
```
Položka **CH_Rate** obsahuje požadovanou přenosovou rychlost v bitech za sekundu.
```
CH_Parity   : tParity;
```
Položka **CH_Parity** obsahuje požadovanou paritu přenášeného znaku.
```
CH_Stop     : tStop;
```
Položka **CH_Stop** obsahuje počet stop bitů přenášeného znaku.
```
CH_Length   : tLength;
```
Položka **CH_Length** obsahuje počet datových bitů přenášeného znaku.
```
CH_STickHW  : Boolean;
```
Položka **CH_STickHw** je určena jen pro vnitřní použití, pro určení, zda je vykonávána činnost vysílacího automatu.

5.1.2. Metody

5.1.2.1. Init konstruktor

```
constructor Init;
```
Konstruktor **Init** slouží k vytvoření a inicializaci instance komunikačního objektu. Ve svém těle nejdříve zavolá zděděný konstruktor **Init** (inherited Init) z rodičovského objektu tChnMod a poté inicializuje položky objektu. Tělo konstruktoru vypadá následovně:

```
inherited Init;
CH_Type     := cName;
CH_Name     := CH_Type;
CH_NumName  := ChNumName(CH_Type);
CH_NumNameParents := ChNumName(ChnMod.cName);
CH_Cislo    := 1;
CH_Addr     := ACom1;
CH_Irq      := IRQ4;
CH_Rate     := 9600;
CH_Parity   := ParOdd;
CH_Stop     := Stop2;
CH_Length   := Bits8;
CH_STickHW  := false;
```

5.1.2.2. ChInitParam konstruktor

```
constructor ChInitParam(const S: TParamStr);
```
Konstruktor **ChInitParam** slouží ke zkrácenému vytvoření instance komunikačního objektu s definovaným nastavením parametrů kanálu. Ve svém těle nejprve volá konstruktor **Init** a poté metodu **ChSetParam**.

5.1.2.3. ChSetOneParam funkce

```
function ChSetOneParam(const S: tWordString; var CmdL: tCmd)
         : tChResult;
```
Metoda **ChSetOneParam** slouží k dekódování a nastavení jednoho konkrétního parametru, který je zadán v parametru S. Tato metoda se volá v aplikaci prostřednictvím metody **ChSetParam**. Metoda **ChSetOneParam** komunikačního objektu tChnComM dekóduje tyto parametry:

COM=aaa

Parametr **COM** určuje číslo sériového kanálu COM. aaa může nabývat hodnot 1 až 8. Adresy odpovídající jednotlivým kanálům COM jsou uvedeny v jednotce ChnTypes.

ADD=bbb

Parametr **ADD** ("Address") určuje adresu sériového kanálu COM. Použije se jen v případě, kdy komunikační adaptér i8250 není na standardní vstupně/výstupní adrese. bbb může nabývat hodnot 0 až $ffff.

IRQ=ccc

Parametr **IRQ** určuje číslo přerušení IRQ, na kterém adaptér i8250 žádá o zpracování přerušení. ccc může nabývat hodnot 0 až 7.

BD =ddd

Parametr **BD** ("BaudRate") určuje přenosovou rychlost požadované sériové komunikace. ddd může nabývat hodnot 25 až 115200 Bd.

BIT=eee

Parametr **BIT** ("Number of Data Bits") určuje počet datových bitů v přenášeném znaku. eee může nabývat hodnot 5 až 8.

PAR=fff

Parametr **PAR** ("Parity") určuje paritu přenášeného znaku. fff může nabývat hodnot 0 "Odd" pro lichou paritu, E "Even" pro sudou paritu a N "None" pro znak bez parity.

STO=ggg

Parametr **STO** ("Number of Stop Bits") určuje počet stop-bitů v přenášeném znaku. ggg může nabývat hodnot 1 nebo 2.

Příklad:

Příklad ukazuje, jak je možné v jednotce se jménem COMM nastavit parametry komunikace na sudou paritu, přenosovou rychlost 4800 Baudů a velikost vstupního vyrovnávacího bufferu na data i na příkazy na 1000 položek.

```
ChSetParam('NAM=COMM PAR=E BD=4800 ARB=1000');
```

Pozn: Všimněte si, že není volána metoda ChSetOneParam, ale metoda ChSetParam, ve které se zadávají i parametry definované rodičovským objektem tChnMod.

5.1.2.4. ChGetParam funkce

```
function  ChGetParam(const S: TParamStr): TParamStr;
```

Metoda **ChGetParam** navrací nastavené hodnoty parametrů komunikačního objektu. Nejprve vrátí nastavení parametrů rodičovského komunikačního objektu tChnMod a poté k nim připojí seznam svých parametrů. Seznam parametrů je uveden výše u popisu metody **ChSetOneParam**.

5.1.2.5. ChInitHW procedura

```
procedure ChInitHW;
```

Metoda **ChInitHW** provede hardwarovou inicializaci (nastavení) komunikačního kanálu.

5.1.2.6. ChDoneHW procedura

```
procedure ChDoneHW;
```

Metoda **ChDoneHW** provede hardwarové uzavření komunikačního kanálu.

5.1.2.7. ChSendHW procedura

```
procedure ChSendHW(Buff: Pointer; Len:Word);
```

Metoda **ChSendHW** provede započetí fyzického vysílání zprávy do komunikační linky.

5.1.2.8. ChSendFlushHW procedura

```
procedure ChSendFlushHW;
```

Metoda **ChSendFlushHW** provede fyzické ukončení vysílání a vyprázdnění vysílacího bufferu.

5.1.2.9. ChSendTickHW procedura

```
procedure ChSendTickHW;
```

Metoda **ChSendTickHW** provede krok vysílacího automatu.

5.1.2.10. ChSendReadyHW procedura

```
procedure ChSendReadyHW:TChState;
```

Metoda ChSendReadyHW provede krok vysílacího automatu na základě volání metody **ChSendTickHw**.

5.1.2.11. ChSetModem procedura

```
procedure ChSetModem(S,R: Byte);
```

Metoda **ChSetModem** provede nastavení modemových signálů. Parametr S určuje masku signálů, které se mají nastavit, a parametr R určuje masku signálů, které se mají vynulovat.

5.1.2.12. ChGetModem funkce

```
function  ChGetModem: Byte;
```

Metoda **ChGetModem** vrátí nastavení modemových signálů. Výsledek je byte, kde jednotlivé bity určují jednotlivé modemové signály.

5.2. tAddChnComM

Typ **tAddChnComM** je typem objektu, který slouží k definování prvku v seznamu správců komunikačních objektů (tzv. správce komunikačního objektu tChnComM v seznamu správců). Objekt tAddChnComM je dědicem od rodičovského objektu **tAddChnVirt**.

5.2.1. Metody

5.2.1.1. ChInit funkce

```
function ChInit: pChnVirt;
```

Metoda **ChInit** slouží k vytvoření instance komunikačního objektu tChnComM a ukazatel na instanci tohoto objektu vrací jako svoji funkční hodnotu.

6. Příklad

Příklad ukazuje použití komunikační jednotky ChnComM a ChnMod.

```
uses
  uString,
  ChnVirt,
  ChnMod,
  ChnComM;
  ...
const
  ParamStr : tParamStr =
             'NAM=COMM MAS=MASTER COM=1 PAR=N BD=9600 BIT=8 STO=2 '+
             'LRB=1000 I1S=''ATZ'' I2S=''ATL1M3'' '+
             'PNS=''ATDP'' PNN=''242351'' REP=1';
  LMess    = 40;

type
  tMess    = array [0..LMess+10] of Byte;

var
  Chn      : pChnVirt;
  SMess    : ^tMess;
  RMess    : ^tMess;
  LSMess   : Word;
  LRMess   : Word;

begin
  ...
  New(SMess);
  New(RMess);
  ...
  { inicializace Chn }
  Chn:=ChnCollection^.ChNewInit(ChnComM.cName);
  with Chn^ do
  begin
    { nastavení parametrů komunikace }
    ChSetParam(ParamStr);
    if ChResult<>res_Ok then WriteLn('Chyba');
    { otevření hardwarového kanálu }
    ChOpen;
    repeat
      if ChResult<>res_Ok then WriteLn('Chyba');
    until ChReady=CHS_Open;
    { definování místa, kam se má přijatá zpráva uložit }
    ChReceiveBuffer(RMess,SizeOf(RMess^));
    if ChReceiveResult<>res_Ok then WriteLn('Chyba');
    { navázání spojení s volanou stanicí }
    ChConnect;
    repeat
      if ChResult<>res_Ok then WriteLn('Chyba');
    until ChReady=CHS_Connect;
    ...
    { naplnění zprávy daty }
    ...
    { vyslání zprávy }
    if ChSendReady=CHS_SendReady then
    begin
      ChSend(SMess, LSMess);
      { čekání na odvysílání zprávy }
      repeat
```

```
        if ChSendResult<>res_Ok then WriteLn('Chyba');
      until ChSendReady=CHS_SendReady;
      ...
    end;
    ...
    { čekání na příjem zprávy }
    while not ChReceiveReady=CHS_ReceiveReady do
    begin
      if ChReceiveResult<>res_Ok then WriteLn('Chyba');
    end;
    { příjem zprávy }
    ChReceive(LRMess);
    if ChReceiveResult<>res_Ok then WriteLn('Chyba');
    ...
    { ukončení }
    ChDisConnect;
    repeat
      if ChResult<>res_Ok then WriteLn('Chyba');
    until ChReady=CHS_DisConnect;
    ChClose;
    repeat
      if ChResult<>res_Ok then WriteLn('Chyba');
    until ChReady=CHS_Close;
  end;
  { zrušení instance objektu }
  Dispose(Chn, Done);
  ...
end.
```