

ChnV40M

JEDNOTKA SÉRIOVÉ KOMUNIKACE POMOCÍ MODEMU NA RS232 S V40

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>

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 popsáno. 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)

Upozornění:

V dokumentu použité názvy výrobků, firem apod. mohou být ochrannými známkami nebo registrovanými ochrannými známkami příslušných vlastníků.

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. tChnV40M	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	8
5.1.2.8. ChSendFlushHW procedura	8
5.1.2.9. ChSendTickHW procedura	8
5.1.2.10. ChSendReadyHW procedura	9
5.1.2.11. ChSetModem procedura	9
5.1.2.12. ChGetModem funkce	9
5.2. tAddChnV40M	9
5.2.1. Metody	9
5.2.1.1. ChInit funkce	9
6.Příklad	9

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 V40.

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 popisujícím rozhraní svých potomků 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 ChnV40M definuje objekt **tChnV40M**, jehož instance vytváří fyzickou vrstvu v komunikačním kanálu tvořenou sériovým rozhraním prostřednictvím telefonního modemu připojeném na RS232 s obvodem i8251 obsaženým v procesoru V40. 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 zaslány z výstupního bufferu rovněž s využitím přerušovacího systému.

Objekt **tChnV40M** je dědicem od komunikačního objektu **tChnMod**, to znamená, že definuje pouze metody poplatné použitému hardware. Inicializaci a nastavení modemu, navazování a rušení komunikace, přenos a příjem dat zajišťují metody rodičovského objektu **tChnMod**, které se nepředefinovávají.

Knihovna rovněž definuje objekt **tAddChnV40M**, který je dědicem od rodičovského objektu **tAddChnVirt**. Objekt **tAddChnV40M** zajistí, aby daný komunikační objekt (objekt **tChnV40M**) byl k aplikaci připojen a popřípadě zajistí vytvoření instance tohoto objektu. Po přilinkování této jednotky do aplikace (příkazem "uses ChnV40M"), se jméno objektu **tChnV40M** automaticky vloží do seznamu správců komunikačních objektů pro případné použití.

Protože je objekt **tChnV40M** 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   = 'V40M';
```

Konstanta **cName** definuje jméno komunikačního objektu **tChnV40M**.

5. Objekty

5.1. tChnV40M

5.1.1. Položky

```
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_SetModemAdr : ^byte;
```

Položka **CH_SetModemAdr** definuje externí adresu pro výstupní modemové signály.

```
CH_STickHW : Boolean;
```

Položka **CH_STickHW** na je nastavená na true pokud je vykonávána činnost HW 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_Rate      := 9600;
CH_Parity    := ParOdd;
CH_Stop      := Stop2;
CH_Length    := Bits8;
FlExtSetModem := false;
CH_SetModemAdr := nil;
CH_OldTimer  := 3;
```

5.1.2.2. ChInitParam konstruktor

```
constructor ChInitParam(const S: ParamStr);
```

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 dekodování a nastavení jednoho konkrétního parametru, který je zadán v parametru S. Tato metoda se volá v aplikaci prostřednictvím metody **ChSetParam**. Metoda **ChSetOneParam** komunikačního objektu tChnV40M dekoduje tyto parametry:

BD=aaa

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

BIT=bbb

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

PAR=ccc

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

STO=ddd

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

AMS=\$Segment:\$Offset

Parametrem **AMS** ("External Address of Modem Signals") se nastaví externí adresa pro výstupní modemové signály. Metody komunikačního objektu nebudou modemové signály nastavovat přímo na výstupní port (\$D222), ale budou nastavovat jednotlivé bity v byte na zadané externí adrese.

Příklad:

Příklad ukazuje, jak je možné v jednotce se jménem V40M nastavit parametry komunikace na sudou paritu, přenosovou rychlost 4800 Bd a velikost vstupního vyrovnávacího bufferu pro data i příkazy na 1000 položek.

```
ChSetParam('NAM=V40M PAR=E BD=4800 LRB=1000 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** navrácí 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(Buffer: Pointer; Len: Word);
```

Metoda **ChSendHW** provede započítání 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. tAddChnV40M

Typ **tAddChnV40M** je typem objektu, který slouží k definování prvku v seznamu správců komunikačních objektů (tzv. správce komunikačního objektu **tChnV40M** v seznamu správců). Objekt **tAddChnV40M** 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 **tChnV40M** a ukazatel na instanci tohoto objektu vrací jako svoji funkční hodnotu.

6. Příklad

Příklad ukazuje použití komunikační jednotky **ChnV40M** a **ChnMod**.

```
uses
  uString,
  ChnVirt,
  ChnMod,
  ChnV40M;
...

const
  ParamStr : tParamStr =
    'NAM=V40M MAS=MASTER 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(ChnV40M.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;
    if ChResult<>res_Ok then WriteLn('Chyba');
    { 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;
    if ChResult<>res_Ok then WriteLn('Chyba');
    ...
    { 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.
```