ChnComPB

JEDNOTKA SÉRIOVÉ KOMUNIKACE RS232 / RS485 S OBVODEM I8250 S VYUŽITÍM PARITNÍHO (ADRESNÍHO) BITU

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

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

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. Opravené hlavičky metod ChSetBinParam a ChGetBinParam. Doplněné metody ChOpen a ChConnect.

1.2. Účel dokumentu

Tento dokument slouží jako popis jednotky sériové komunikace RS232 / RS485 s obvodem i8250 s využitím paritního (adresního) bitu.

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 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 ChnComPB definuje objekt **tChnComPB**, jehož instance vytváří fyzickou vrstvu v komunikačním kanálu tvořenou sériovým rozhraním RS232 nebo RS485 s obvodem i8250. Ke své činnosti využívá přerušovací systém počítače. Parametrem nastavovací metody **ChSetParam** lze určit při vysílání prvního byte zprávy vysílání paritního bitu s pevně nastavenou hodnotou (tzv. adresní bit), což znamená, že tento první vysílaný znak bude odvysílán s paritním bitem nastaveným do logické úrovně 1 nebo do logické úrovně 0. Ostatní vysílané znaky zprávy budou odvysílány vždy s paritním bitem nastaveným do logické úrovně 0. Při příjmu se v případě přijetí znaku s paritním bitem v logické úrovni 1 generuje chyba parity, která se z výsledného statusu odstraní a vyhodnotí jako příjem adresního znaku. V tomto případě se z důvodu kompatibility s knihovnou ChnComBR, kdy se přijetí adresního znaku generuje ve formě dvou znaků z nichž první určuje, že jde právě o adresní znak (viz. příručka ChnComBR), generuje také ve formě dvou znaků. Této vlastnosti využívají některé vyšší komunikační protokoly pro určení začátku zprávy. Znaky přicházející po komunikaci 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.

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

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

4. Konstanty a jednoduché typy

```
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     = 'COMPB';
```

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

4.1. Konstanty kódů pro metodu ChGetBinParam

```
cmd_GetSendPBit = $0201;
```

Kód pro vrácení používání paritního bitu při vysílání.

```
cmd_GetPBitChar = $0202;
```

Kód pro vrácení statusu paritního bitu přijatého znaku po volání metody ChReceiveChar.

4.2. Konstanty kódů pro metodu ChSetBinParam

`cmd_SetSendPBit = $0201;`

Kód pro nastavení používání paritního bitu při vysílání.

5. Objekty

5.1. tChnComPB

5.1.1. Položky

`CH_SendPBit     : Boolean;`

Položka **CH_SendPBit** definuje, jestli se při vysílání prvního znaku zprávy má použít nastavování paritního bitu do logické úrovně 1 či 0.

`CH_ReceivePBit  : Boolean;`

Položka **CH_ReceivePBit** určuje příznak přijetí znaku s opačnou paritou při posledním přijetí znaku metodou ChReceiveChar.

5.1.2. Metody

5.1.2.1. Init konstructor

`constructor Init;`

Konstruktor **Init** slouží k vytvoření a inicializaci instance komunikačního objektu. Ve svém těle zavolá zděděný konstruktor **Init** (inherited Init) z rodičovského objektu tChnCom 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(ChnCom.cName);
CH_SendPBit := True;
CH_ReceivePBit := False;
```

5.1.2.2. ChInitParam konstructor

`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 tChnComPB dekóduje tyto parametry:

PB=ON/OFF

Parametr **PB** "Use Parity Bit" určuje logickou úroveň nastavitelného paritního bitu při vysílání prvního znaku zprávy.

Příklad:

Příklad ukazuje, jak je možné v jednotce COMPB nastavit parametry komunikace na sudou paritu, přenosovou rychlost 4800 Bd, velikost vstupního vyrovnávacího bufferu na 1000 položek a vysílání prvního znaku zprávy s nastaveným paritním bitem.

```
ChSetParam('NAM=COMPB PAR=E BD=4800 LRB=1000 PB=ON');
```

Pozn: Všimněte si, že není volána metoda ChSetOneParam, ale metoda ChSetParam.

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 tChnCom kromě parametru "PAR", který určuje paritu přenášeného znaku, a poté k nim připojí seznam svých parametrů. Seznam parametrů je uveden výše u popisu metody **ChSetOneParam**.

5.1.2.5. ChSetBinParam procedura

```
procedure ChSetBinParam(NumName: tChNumName; Code: Word;
                        Param: longint);
```

Metoda **ChSetBinParam** s parametrem NumName příslušným jménu tohoto komunikačního objektu, lze podle parametru Code nastavit tyto parametry:

`Code = cmd_SetSendPBit`

Podle parametru Param nastaví logickou úroveň paritního bitu při vysílání prvního znaku zprávy. Je-li Param roven nule, bude první vysílaný znak zprávy vysílán s paritním bitem nastaveným do logické úrovně 0, v opačném případě bude vysílán s paritním bitem nastaveným do logické úrovně 1.

5.1.2.6. ChGetBinParam funkce

```
function  ChGetBinParam(NumName: tChNumName; Code: Word): longint;
```

Metoda **ChGetBinParam** s parametrem NumName příslušným jménu tohoto komunikačního objektu, lze podle parametru Code vrátit tyto nastavení:

`Code = cmd_GetSendPBit`

Vrátí příznak logické úrovně paritního bitu při vysílání prvního znaku zprávy.

`Code = cmd_GetPBitChar`

Vrátí příznak přijetí znaku s opačnou paritou při posledním přijetí znaku metodou **ChReceiveChar**. Je-li vrácená hodnota = 0, potom byl přijat znak se stejnou paritou (normální znak). Je-li vrácená hodnota = 1, potom byl přijat znak s opačnou paritou (adresní znak).

5.1.2.7. ChOpen procedura

```
procedure  ChOpen;
```

Metoda **ChOpen** provede inicializaci technického vybavení. Po úspěšném vyvolání této metody přejde komunikační kanál během několika cyklů do stavu **CHS_Open**.

5.1.2.8. ChConnect procedura

```
procedure  ChConnect;
```

Metoda **ChConnect** provede navázání komunikace mezi účastníky. Před voláním metody musí být kanál ve stavu **CHS_Open**. Po úspěšném volání metody přejde kanál po čase do stavu **CHS_Connect**, to znamená, že je možno po daném kanále komunikovat (posílat a přijímat data pomocí metod **ChSend** a **ChReceive**).

5.1.2.9. ChSendTick procedura

```
procedure ChSendTick;
```

Metoda **ChSendTick** způsobí provedení kroků vysílacích automatů. Je nutné ji periodicky volat během vysílání. **ChSendTick** je rovněž automaticky volána v metodách **ChSendReady** a **ChSend**.

5.1.2.10. ChReceiveChar funkce

```
function  ChReceiveChar: byte;
```

Pokud je kanál ve stavu **CHS_Connect** a v přijímacím bufferu jsou přijata nějaká data, navrací metoda **ChReceiveChar** jeden přijatý znak z přijímacího bufferu. Vždy se dekóduje status přijatého znaku na chybu parity. Podle tohoto testu se nastaví položka CH_ReceivePBit a bit chyby parity se ze statusu odstraní. Výsledek operace přijímače se nastaví na výsledný status přijatého znaku. Metodu **ChReceiveChar** je možno volat pouze ve stavu automatu přijímače **CHS_ReceiveReady**, proto před voláním této metody musí předcházet volání metody **ChReceiveReady** s testem na stav **CHS_ReceiveReady**, jinak v případě nepřijetí žádného znaku skončí volání metody **ChReceiveChar** chybou.

5.2. tAddChnComPB

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

6. Příklad

Příklad ukazuje použití komunikační jednotky ChnComPB. Je vytvořen komunikační kanál definovaných vlastností, po kterém je zasílána zpráva a z kterého je poté očekáván příjem zpráv.

```
uses
  uString,
  ChnVirt,
  ChnCom,
  ChnComPB,
  ...

const
  ParamStr : tParamStr =
             'NAM=COMPB COM=1 IRQ=4 BD=1200 BIT=8 STOP=2 '+
             'LRB=1000 PB=ON';
  LMess    = 40;

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

var
  Chn      : pChnVirt;
  SMess    : ^tMess;
  RMess    : ^tMess;
  LRMess   : Word;
  RecChar  : Byte;

begin
  ...
  New(SMess);
  New(RMess);
  ...
  { inicializace Chn }
  Chn:=ChnCollection^.ChNewInit(ChnComPB.cName);
  with Chn^ do
  begin
    { nastavení parametrů komunikace }
    ChSetParam(ParamStr);
    if ChResult<>res_Ok then WriteLn('Chyba');
    ChOpen;
    repeat
      if ChResult<>res_Ok then WriteLn('Chyba');
    until ChReady=CHS_Open;
    { definování místa, kam se má přijatá zpráva uložit }
    ChReceiveBuffer(RMess,SizeOf(tMess));
    if ChReceiveResult<>res_Ok then WriteLn('Chyba');
    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, LMess);
      repeat
        if ChSendResult<>res_Ok then WriteLn('Chyba');
      until ChSendReady=CHS_SendReady;
      if ChSendResult<>res_Ok then WriteLn('Chyba');
      ...
    end;
    ...
    { čekání na příjem zprávy }
    repeat
    until ChReceiveReady=CHS_ReceiveReady;
    { příjem zprávy }
    while ChReceiveReady=CHS_ReceiveReady do
    begin
      if ChReceiveResult<>res_Ok then WriteLn('Chyba')
```

```
      else
        begin
          RecChar:=ChReceiveChar;
          if ChGetBinParam(CH_NumName,cmd_ReceivePBit)<>0 then
            Write('[Addr Byte]')
          else
            Write(RecChar);
        end;
    end;
    ...
    { ukončení }
    ChDisConnect;
    if ChResult<>res_Ok then WriteLn('Chyba');
    repeat
    until ChReady=CHS_DisConnect;
    ChClose;
    if ChResult<>res_Ok then WriteLn('Chyba');
    repeat
    until ChReady=CHS_Close;
  end;
  { zrušení instance objektu }
  Dispose(Chn,Done);
  ...
end.
```