uIOT10

JEDNOTKA PRO OVLÁDÁNÍ DESKY IOTERM10

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: 22.05.2003
Datum posledního uložení dokumentu: 22.05.2003
(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

1.1. Revize dokumentu

Verze dokumentu	Verze SW	Autor	Datum vydání	Popis změn
1.00	1.XX	Če		První vydání.
1.10	2.XX	Tu	22.05.2003	Úprava dokumentu dle ISO9000.

1.2. Účel dokumentu

Tento dokument slouží jako popis jednotky pro ovládání desky IOTERM10.

1.3. Rozsah platnosti

Určen pro programátory a uživatele programového vybavení SofCon.

1.4. Související dokumenty

Pro čtení tohoto dokumentu není potřeba číst žádný další manuál, ale je potřeba se orientovat v používání programového vybavení SofCon.

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

Jednotka slouží k obsluze desky IOTERM10. Deska je obsahuje tyto periferie:

- 16 binárních opticky oddělených vstupů
- 16 binárních opticky oddělených výstupů
- 24 bitový sigma-delta AD převodník a vstupní analogový multiplexer s 8 unipolárními analogovými vstupy a 8 bipolárními analogovými vstupy s můstky pro připojení odporového čidla.
- 6 analogových výstupů s 12 bitovými DA převodníky
- integrovaný teploměr
- sériovou paměť EEPROM

Pro obsluhu uvedených periferií je v jednotce deklarován objektový typ **tIOT10**..

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.

```
const
  MaxDataAD                =$FFFFFF;
```

MaxData je maximální číslo, které lze dostat z A/D převodníku, odpovídá dekadickému číslu 16777216.

```
  HalfDataAD               =$800000;
```

HalfDataAD je číslo, které odpovídá polovičnímu rozsahu A/D převodníku, odpovídá dekadickému číslu 8388608.

```
  ModeAD_Normal            =0;
  ModeAD_SelfCalib         =1;
  ModeAD_SystemCal0        =2;
  ModeAD_SystemCalFull     =3;
  ModeAD_SystemOffsCal     =4;
  ModeAD_BackgroundCal     =5;
  ModeAD_RW0CalCoef        =6;
  ModeAD_RWFullCalCoef     =7;
```

Konstanty **ModeAD_xxxx** jsou konstanty režimů AD převodníku. Význam - viz popis metody **tIOT10.SetAD.**

```
const
  ADGain_1                  =0;
  ADGain_2                  =1;
  ADGain_4                  =2;
  ADGain_8                  =3;
  ADGain_16                 =4;
  ADGain_32                 =5;
  ADGain_64                 =6;
  ADGain_128                =7;
```

Konstanty **AdGain_xxxx** jsou konstanty pro řízení zisku AD převodníku. Význam - viz popis metody **tIOT10.SetAD.**

```
const
  PIOT10_In1    =0;{offset adresy binárního vstupu  X4}
  PIOT10_In2    =1;{offset adresy binárního vstupu  X5}
  PIOT10_Out1   =4;{offset adresy binárního výstupu X7}
  PIOT10_Out2   =5;{offset adresy binárního výstupu X6}
  PIOT10_EOut   =3;{offset adresy uvolnění binárních výstupů}
```

Konstanty **PIOT10_xxxx** udávají offsety adres jednotlivých periferií.

const	čtení / zápis	význam
PIOT10_In1	čtení	první osmice vstupů
PIOT10_In2	čtení	druhá osmice vstupů
PIOT10_Out1	zápis	první osmice výstupů
PIOT10_Out2	zápis	druhá osmice výstupů
PIOT10_EOut	zápis	připojení výstupů.

```
type
  tTransPar=record
              KLimitP       : Longint;
              KLimitM       : Longint;
              KAdd0         : Longint;
              KAddP         : Longint;
              KAddM         : Longint;
              KMultP1       : Longint;
              KMultP2       : Longint;
              KMultM1       : Longint;
              KMultM2       : Longint;
            end;
```

Typ **tTransPar** obsahuje korekční koeficienty pro korekci chyby můstku při snímání odporového čidla. Význam položek záznamu: viz popis metody **tIOT10.SetRezist**.

5. Popis objektu tIOT10

```
type
  pIOT10 = ^tIOT10;
  tIOT10 = object(tObject)
```

Objektový typ **tIOT10** implementuje metody pro práci s periferiemi desky IOTERM10. Kromě metod pro obsluhu binárních vstupů a výstupů nejsou metody objektu reentrantní a je na uživateli, aby zajistil, že v době výkonu některé z metod nebude volána žádná další metoda tohoto objektu.

5.1. Proměnné

```
BaseAddr : Word;
```

Proměnná **BaseAddr** uchovává bázovou adresu desky IOTERM10 v I/O prostroru procesoru.

```
LKanal   : Byte;
```

Proměnná **LKanal** uchovává číslo kanálu nastavené na vstupním analogovém multiplexeru.

```
LGain    : Byte;
```

Proměnná **LGain** uchovává nastavený zisk AD převodníku.

```
Wait90Cykl: Word;
```

Proměnná **Wait90Cykl** uchovává potřebný počet průchodů čekací smyčkou pro zpoždění 90 μs.

```
Wait2Cykl: Word;
```

Proměnná **Wait2Cykl** uchovává potřebný počet průchodů čekací smyčkou pro zpoždění 2 μs.

```
Wait8Cykl: Word;
```

Proměnná **Wait8Cykl** uchovává potřebný počet průchodů čekací smyčkou pro zpoždění 8 μs.

```
ATransPar: array [8..15] of tTransP;
```

Proměnná **ATransPar** je pole s korekčními koeficienty pro korekci chyby můstku při snímání odporového čidla.

```
StatOut: Word;
```

Proměnná **StatOut** uchovává stav binárních výstupů.

```
EEPROMAdresBits: Byte;
```

Proměnná **EEPromAdresBits** obsahuje zjištěný počet adresových bitů osazené sériové paměti EEPROM.

5.2. Metody

5.2.1. Init

```
constructor Init(NAddr:Word);
```

Konstruktor **Init** inicializuje objekt a hardware. Během inicializace hardware se provádí měření rychlosti procesoru, které trvá asi 1.1 s. Po tuto dobu by neměl být přepínán kontext ani by neměl být procesor příliš zahlcen obsluhou přerušení. Přerušení od systémového časovače (INT 8) však nesmí být zakázáno. Podle výsledku měření rychlosti je nastavena proměnná **Wait90Cykl**. V Initu se voláním metody SetRezist nastavují proměnné ATransPar tak, aby odpovídaly vstupnímu děliči pro PT100, to jest odporům 2M, 100Ω a 100kΩ.

5.3. Metody pro práci s binárními vstupy

5.3.1. ReadInB1

```
function ReadInB1:Byte;
```

Metoda **ReadInB1** vrací stav první osmice binárních vstupů, které se nacházejí na konektoru X4. Metodu lze volat reentrantně.

5.3.2. ReadInB2

```
function ReadInB2:Byte;
```

Metoda **ReadInB2** vrací stav druhé osmice binárních vstupů, které se nacházejí na konektoru X5. Metodu lze volat reentrantně.

5.3.3. ReadInW

```
function ReadInW:Word;
```

Metoda **ReadInW** vrací stav všech 16 binárních vstupů. Metodu lze volat reentrantně.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
druhá osmice konektor X5								první osmice konektor X4							

5.4. Metody pro práci s binárními výstupy

5.4.1. WriteOutB1

```
procedure WriteOutB1(On,Off:Byte);
```

Metoda **WriteOutB1** nastavuje první osmici binárních výstupů, které se nacházejí na konektoru X7. Nastavené bity parametru **On** udávají které výstupy budou sepnuty a nastavené bity parametru **Off** udávají, které výstupy budou rozepnuty. Stav výstupů je uchován v proměnné **StatOut**. Metodu lze volat reentrantně.

*Pozor ! Metoda připojí všech 16 binárních výstupů. Při prvním zápisu do výstupů je třeba použít metodu **WriteOutW**, aby byl při připojení výstupů již definován stav všech 16 výstupů.*

5.4.2. WriteOutB2

```
procedure WriteOutB2(On,Off:Byte);
```

Metoda **WriteOutB2** nastavuje druhou osmici binárních výstupů, které se nacházejí na konektoru X6. Nastavené bity parametru **On** udávají které výstupy budou sepnuty a nastavené bity parametru **Off** udávají, které výstupy budou rozepnuty. Stav výstupů je uchován v proměnné **StatOut**. Metodu lze volat reentrantně.

*Pozor ! Metoda připojí všech 16 binárních výstupů. Při prvním zápisu do výstupů je třeba použít metodu **WriteOutW**, aby byl při připojení výstupů již definován stav všech 16 výstupů.*

5.4.3. WriteOutW

```
procedure WriteOutW(On,Off:Word);
```

Metoda **WriteOutW** nastavuje všech 16 binárních výstupů. Nastavené bity parametru **On** udávají které výstupy budou sepnuty a nastavené bity parametru **Off** udávají, které výstupy budou rozepnuty. Stav výstupů je uchován v proměnné **StatOut**. Metodu lze volat reentrantně. *Pozor ! Metoda připojí všech 16 binárních výstupů.*

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
druhá osmice konektor X6								první osmice konektor X7							

5.4.4. Tick

```
procedure Tick;virtual;
```

Metoda **Tick** je určena k periodickému volání. Metoda obnoví stav binárních výstupů podle proměnné **StatOut**. Před prvním voláním metody **Tick** je třeba definovat stav výstupů, např. voláním metody **WriteOutW**.

5.5. Metody pro práci s analogovými vstupy

5.5.1. SetAD

```
procedure SetAD(N: byte; M: byte;
                G: byte; F: Integer);
```

Metoda **SetAD** je určena k nastavení AD převodníku a vstupního analogového multiplexeru. Parametr **N** definuje číslo kanálu, který bude zpracováván, v rozsahu 0 až 15 (0 až 7 - unipolární kanály, 8 až 15 - bipolární kanály). Parametr **M** definuje režim AD převodníku (pro jednotlivé režimy jsou definovány konstanty **ModeAD_xxxx**):

číslo	konst.	název	popis
0	ModeAD_Normal	Normal Mode	Normální režim, převodník snímá nastavený kanál.
1	ModeAD_SelfCalib	Activate Self-Calibration	Převodník se zkalibruje pro nulu a plný rozsah pomocí interního spojení vstupu s GND a V_{REF}. Po kalibraci přejde do normálního režimu.
2	ModeAD_SystemCal0	Activate System Calibration - zero-scale	Převodník se zkalibruje pro nulu. Tento režim předpokládá externí připojení vstupu na nulový potenciál. Po kalibraci přejde do normálního režimu.
3	ModeAD_SystemCalFull	Activate System Calibration - full-scale	Převodník se zkalibruje pro plný rozsah. Tento režim předpokládá externí připojení vstupu na potenciál odpovídající plnému rozsahu převodníku. Po kalibraci přejde do normálního režimu.
4	ModeAD_SystemOffsCal	Activate System Offset Calibration	Převodník se zkalibruje pro nulu a plný rozsah. Tento režim předpokládá externí připojení vstupu na nulový potenciál. Kalibrace pro plný rozsah se provede na základě interního připojení vstupu na

číslo	konst.	název	popis
			V_{REF}. Po kalibraci přejde do normálního režimu.
5	ModeAD_BackgroundCal	Activate Background Calibration	Kalibrace na pozadí. Převodník snímá nastavený kanál a průběžně provádí kalibraci pomocí interního spojení vstupu na GND a V_{REF}. ***Po nastavení tohoto režimu je první vzorek neplatný !***
6	ModeAD_RW0CalCoef	Read/Write Zero-Scale Calibration Coefficients	V tomto režimu lze pomocí metod **ReadCal** a **WriteCal** číst a zapisovat 24 bitový kalibrační koeficient pro nulu.
7	ModeAD_RWFullCalCoef	Read/Write Full-Scale Calibration Coefficients	V tomto režimu lze pomocí metod **ReadCal** a **WriteCal** číst a zapisovat 24 bitový kalibrační koeficient pro plný rozsah převodníku.

Parametr **G** definuje zesílení na vstupu AD převodníku podle následující tabulky (pro jednotlivé hodnoty zesílení jsou definovány konstanty **ADGain_xxxx**):

G	konst.	zesílení
0	ADGain_1	1
1	ADGain_2	2
2	ADGain_4	4
3	ADGain_8	8
4	ADGain_16	16
5	ADGain_32	32
6	ADGain_64	64
7	ADGain_128	128

Parametr **F** definuje nastavení notch frekvence číslicového filtru v AD převodníku a to:

$F = \frac{19531.25}{f_{notch}}$, kde f_{notch} je první notch frekvence filtru [Hz].

F může nabývat hodnot od 19 do 2000, jinak není chování převodníku specifikováno. Filtr má útlum -3dB na frekvenci:

$f_{-3dB} = 0.262 \cdot f_{notch}$

5.5.2. ReadyAD

```
function ReadyAD:Boolean;
```

Metoda **ReadyAD** vrací true pokud je připravena hodnota ke čtení z AD převodníku. V opačném případě vrací false.

5.5.3. ReadAD

```
function ReadAD:Longint;
```

Metoda **ReadAD** slouží k přečtení hodnoty z AD převodníku. Přečtená hodnota je vrácena jako funkční hodnota metody. Pokud není v převodníku hodnota k dispozici, metoda čeká dokud nebude v převodníku hodnota připravena. Doporučuje se volat metodu **ReadAD** až tehdy, když metoda **ReadyAD** vrátí true. Zamezí se tak zbytečnému čekání v metodě **ReadAD**.

5.5.4. WriteCal

```
procedure WriteCal(Dat:Longint);
```

Metoda **WriteCal** slouží k zápisu kalibračních koeficientů v režimech 6 a 7. Parametrem **Dat** je předáván zapisovaný koeficient.

5.5.5. ReadCal

```
function ReadCal:Longint;
```

Metoda **ReadCal** slouží ke čtení kalibračních koeficientů v režimech 6 a 7. Přečtený koeficient je vracen jako návratová hodnota metody.

5.5.6. ReadRezist

```
function ReadRezist:Longint;
```

Metoda **ReadRezist** přečte hodnotu z AD převodníku metodou **ReadAD**. Je-li čten některý z bipolárních kanálů (8 až 15) provede korekci chyby můstku podle nastavených korekčních koeficientů. Jako funkční hodnotu vrátí v tomto případě odpor v mΩ. Korekční koeficienty se nastavují metodou **SetRezist**. Pro unipolární kanály (0 až 7) je vracena hodnota přečtená metodou **ReadAD**.

5.5.7. SetRezist

```
procedure SetRezist(N:Byte;R1,R2,Rmin,Rmax:Longint);
```

Metoda **SetRezist** vypočte korekční koeficienty a uloží je do proměnné pro kanál definovaný parametrem **N** (pouze kanály 8 až 15). Korekční koeficienty jsou vypočteny ze zadaných hodnot parametrů **R1, R2, Rmax, Rmin.** Hodnoty parametrů se zadávají v mΩ.

Jejich význam je na obrázku:

Nelineární převodní křivka je nahrazena lomenou čarou podle obrázku. Parametry **R1** a **R2** udávají velikost odporů v můstku. Parametry **Rmin** a **Rmax** se zadávají krajní body zlomu.

5.5.8. SetNulMOST

```
procedure SetNulMOST;
```

Metoda nuluje vstupní můstky všech bipolárních vstupů.

5.5.9. ResNulMOST

```
procedure ResNulMOST;
```

Metoda ruší nulování vstupních můstků všech bipolárních vstupů.

5.5.10. ResetAD_DA

```
procedure ResetAD_DA;
```

Metoda generuje resetovací impuls pro A/D a D/A převodník.

5.5.11. Příklad čtení analogových vstupů

```
var IOT10  :pIOT10;
    ADData :array [0..15] of Longint;   {data ze vstupu}
...
IOT10:=New(pIOT10,Init(cAdrIOT10));  {vytvoření objektu}
...
procedure PReadAd;far;             {proces čtení vstupů}
var L:Longint;                              {pomocná proměnná}
    N:Byte;                       {číslo čteného kanálu}
begin
  Start(...);Exit;            {spuštění procesu čtení vstupů}
  with IOT10^ do
  begin
    N:=0;
    SetAD(N,ModeAD_BackGroundCal,ADGain_1,900);
    repeat
      while not ReadyAD do Wait(1);      {čeká na Ready}
      L:=ReadAD;   {první vzorek v módu 5 je neplatný}
      while not ReadyAD do Wait(1);      {čeká na Ready}
      LockKernel;   {zajištění konzistence dat v ADData}
        ADData[N]:=ReadAD;  {druhý vzorek je již platný}
      UnLockKernel;
      N:=(N+1) and 15;                     {další kanál}
      SetAD(N,ModeAD_BackGroundCal,ADGain_1,900);
    until FlEnd;
  end;
end;
```

5.6. Metody pro práci s analogovými výstupy

5.6.1. WriteDA

```
procedure WriteDA(N: Byte; W: Integer);
```

Metoda **WriteDA** slouží k zápisu hodnoty do DA převodníku. Parametr **N** definuje výstupní kanál (0 až 5) a parametr **W** obsahuje zapisovanou hodnotu (0 až 4095).

5.7. Metody pro práci s integrovaným teploměrem

5.7.1. GetTemp

```
function GetTemp(var Res:Boolean):Integer;
```

Metoda **GetTemp** slouží ke čtení teploty z integrovaného teploměru. Vrací teplotu v 0.01 °C (12345 = 123.45°C). V parametru **Res** je vráceno true pokud bylo čtení teploty úspěšné. V opačném případě je vrácená hodnota neplatná a čtení je třeba opakovat.

5.8. Metody pro práci se sériovou pamětí EEPROM

5.8.1. WriteEEPROM

```
procedure WriteEEPROM(A,D:Word);
```

Metoda **WriteEEPROM** slouží k zápisu slova do sériové paměti EEPROM. Parametr **A** obsahuje adresu slova a parametr **D** zapisované slovo.

5.8.2. ReadEEPROM

```
function ReadEEPROM(A:Word):Word;
```

Metoda **ReadEEPROM** slouží ke čtení slova ze sériové paměti EEPROM. Parametr **A** obsahuje adresu slova. A je z intervalu <0; $2^{EEpromAresBits+1}$-1>. Přečtené slovo je vráceno jako funkční hodnota.

5.8.3. DetectEEPROM

```
function DetectEEPROM:Byte;
```

Metoda **DetectEEPROM** zjistí jak velká paměť EEPROM je osazena a vrátí jako funkční hodnotu počet adresových bitů paměti. Tato metoda je volána konstruktorem **Init** pro nastavení proměnné **EEpromAdresBits**. Bez správného nastavení této proměnné nebudou funkční metody pro zápis a čtení paměti. Není-li na desce paměť EEProm osazena vrací metoda 0. Potom je EEPromAdresBits=0.