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

6 DĚDĚNÍ 2

Doposud jsme používali jen existující třídy. Nyní bychom chtěli na příkladu teplotního čidla LM75 popsat, jak se navrhuje vlastní třída.

6.1 NÁVRH VLASTNÍ TŘÍDY: TTF

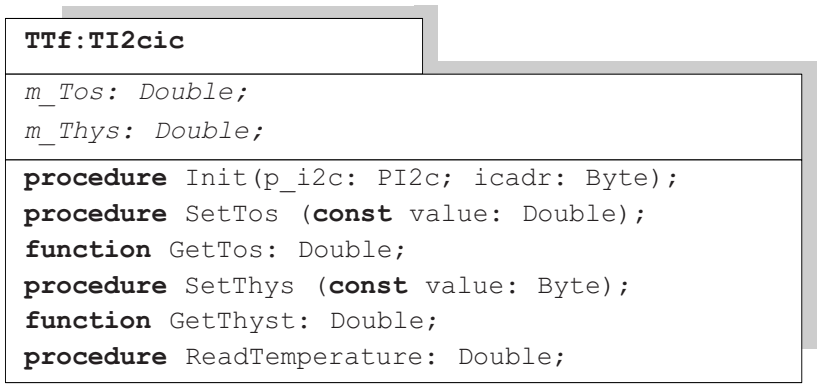
Podobně jako výše nejprve v diagramu třídy a v části věnované rozhraní popíšeme, co nová třída umí. Teprve potom se budeme zabývat tím, jak je možno tyto schopnosti implementovat.

6.1.1 Vlastnosti nového zařízení

Výrobce National Semiconductor popisuje součástku v katalogovém listu (viz doprovodné CD) takto:

- LM75 je teplotní čidlo, AD převodník a digitální hlídač teploty s rozhraním I²C. Obvod LM75 je možno kdykoliv vyzvat k měření teploty. Teplotní hlídač s otevřeným kolektorem (drain) se aktivuje, když teplota překročí nastavitelnou hodnotu. Tento vývod může pracovat v režimu komparátoru („comparator“) nebo v režimu přerušení („interrupt“).
- Je možno naprogramovat jak prahovou teplotu poplachu T_{OS}, tak teplotu (T_{HYST}), při které se poplach opět vypne. Navíc je možno nastavené hodnoty T_{OS} a T_{HYST} přečíst. Obvod má tři vývody (A0, A1 a A2) pro nastavení adresy I²C. Při zapnutí snímače se nastaví režim komparátoru a T_{OS} se nastaví na 80 °C, T_{HYST} na 75 °C.

S tímto popisem výrobce je jasné, **co** snímač umí. Nyní můžeme popsat i uživatelské rozhraní pro třídu teplotního snímače TTf:



6.1.2 Zavedení nové třídy

Jak můžeme třídu s těmito vlastnostmi implementovat?

Vyjdeme z toho, že jste otevřeli projekt, do kterého se má integrovat teplotní snímač (pokud ne, vytvořte nový projekt jako konzolovou nebo dialogovou aplikaci). Zaveďte do svého projektu pomocí File: **New** z karty **New** objektové galerie jednotku (Unit). Uložte novou unit nejlépe hned do C:\MSRDelphi\Units pod jménem TF.pas (teplotní čidlo).

Doplňte nejprve část pro rozhraní takto:

```
unit TF;

interface

uses I2C, I2Cic;
type TTf = class(TI2cic)
           public
                procedure Init (p_i2c:PI2C; icadr: Byte);
                procedure SetTos(const Value: Double);
                function GetTos: Double;
                procedure SetThyst(const Value: Double);
                function GetThyst: Double;
                function GetTemperature: Double;
           private
                property m_Tos: Double read GetTos
                                write SetTos;
                property m_Thyst: Double read GetThyst
                                write SetThyst;
                function BufferToReal
                     (buffer: TbyteArray): Double
      end;
```

V následující implementační části jednotky TF.pas najdete zdrojový text pro třídu TTf a její metody. Text je podrobně komentován, takže spolu s katalogovým listem pro teplotní snímač LM75 bude snadno srozumitelný. Přesto uveďme dopředu dvě poznámky:

- Metoda `Init (…)` byla v obou třídách TIoexp a TAda jednoduše zděděna od třídy TI2cic beze změn. To zde nejde, protože je navíc třeba do příslušných vlastností převzít normální hodnoty nastavené výrobcem.

```
m_Tos := 80;     // normální hodnoty při
m_Thyst := 75;   // zapnutí snímače
```

Mimo to se přirozeně musí vyvolat metoda `Init(…)` ze základní (bázové) třídy. Musí se však nyní výslovně volat jako metoda základní třídy. To se dělá slovem inherited, což znamená „zděděný“.

```
procedure TTf.Init(p_I²C: PI2C; icadr: Byte);
begin
  inherited Init(p_I²C, icadr);
  m_Tos := 80;
  m_Thyst := 75;
end;
```

Zde se tedy volá zděděná metoda `Init(…)`. Kdybychom vynechali slovo `inherited`, vyvolala by Init sama sebe! (S fatálním následkem nekonečného opakování!)

- Teploty se v čidle ukládají v 9bitovém formátu. Při teplotě například 23,5 °C je v prvním bajtu 23 a ve druhém bajtu je nahozen bit s nejvyšší váhou, protože za desetinnou čárkou je číslice 5, obsahuje tedy hodnotu 128. Při 23,0 °C by ve druhém bajtu byla 0. Z obsahů obou bajtů (v `buffer[0]` a `buffer[1]` je tedy možno teplotu vypočítat následujícím způsobem, pokud teplota není záporná:

```
Tos := (buffer[0] * 2 + buffer[1] / 128) ) / 2;
```

Je-li teplota < 0 °C, (`buffer[0] < 128`), musí se první bajt napřed negovat

Tyto převody se provádějí ve `funkci BufferToReal (..)`.

Úplný Listing jednotky TF pro třídu TTf:

```
unit TF;

interface                    // viz výše

implementation

const TEMPREG    = $00;    // registr teploty
      TCONFREG   = $01;    // konfigurační registr
      THYSTREG   = $02;    // registr pro Thyst
      TOSREG     = $03;    // registr pro Tos

function TTF.BufferToReal(buffer: TByteArray): Double;
begin
  if (buffer[0] > 127) then
    begin
      buffer[0] = - buffer[0] ;
```

```
      BufferToReal := - (buffer[0] * 2 - buffer [1] / 128) / 2;
    end
  else
      BufferToReal := (buffer[0] * 2 - buffer [1] / 128) / 2;
end;

procedure TTf.Init(p_i2c: PI2C; icadr: Byte);
begin
  inherited Init(p_I²C, icadr);
  m_Tos := 80;
  m_Thyst := 75;
end;

procedure TTf.SetTos(const Value: Double);
var buffer: array[0..2] of Byte;
begin
  buffer[0] := TOSREG;         // nastavit registr pro Tos
  buffer[1] := Trunc(Value) ;  // 1. bajt obsahuje hodnotu
Wert
if Trunc(2*Value - 2*Trunc(Value) ) <> 0 then
  buffer [2] := $80  // Byte2 je $80, když Temp =.5
else
    buffer [2] := 0; // a $00, je-li Temp = .0
  WriteToIC(buffer, 3);   // vyslat všechny 3 bajy
  WriteToIC(TEMPREG);// znovu nastavit registr teploty
end;

function TTf.GetThyst: Double;
var buffer: TByteArray;
begin
  SetLength(buffer, 2);
  WriteToIC(THYSTREG);    // nastavit registr pro Tos
  ReadFromIC(buffer. 2);  // číst teplotu (2 bajty)
  GetHyst:= BufferToReal(buffer);
end;

function TTf.GetTemperature: Double;
var buffer: TByteArray;
begin
  SetLength(buffer, 2);
  ReadFromIC(buffer, 2);  // číst teplotu (2 bajty)
```

```
    GetTemperature := BufferToReal(buffer) ;
end;

end.
```

Protože novou třídu TTf pravděpodobně nebudeme chtít používat jen v tomto projektu, uložte soubor TF.pas nejlépe do adresáře C:\MSRmitDelphi\UNITS.

6.2 POUŽITÍ NOVÉ TŘÍDY TTf

Uvedeme ještě jeden příklad použití třídy TTf, abychom na něm ukázali, jak lze s novou třídou zacházet.

```
unit Temperaturdlg;

interface

uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls,
  Forms, Dialogs, I2c, TF, StdCtrls, ExtCtrls, Buttons;

type
  TForm1 = class(TForm)
    Timer1: TTimer;
    EditTemp: TEdit;
    EditTos: TEdit;
    EditThyst: TEdit;
    Label1: TLabel;
    Label2: TLabel;
    Label3: TLabel;
    Label4: TLabel;
    Label5: TLabel;
    Label6: TLabel;
    Label7: TLabel;
    BitBtnAbbrechen: TBitBtn;
    Bevel1: TBevel;
    procedure Timer1Timer(Sender: TObject);
    procedure FormCreate(Sender: TObject);
    procedure BitBtnAbbrechenClick(Sender: TObject);
    procedure OnClose(Sender: TObject; var Action:
                        TCloseAction);
```

```
    procedure EditTosOnEnter(Sender: TObject);
    procedure EditThystOnEnter(Sender: TObject);
    procedure EditThystOnExit(Sender: TObject);
    procedure EditTosOnExit(Sender: TObject);
  private
      I2c1: TI2c;
      Tf1: TTf;
    public
      { Public deklarace }
    end;

var
  Form1: TForm1;

implementation

{$R *.DFM}

procedure TForm1.FormCreate(Sender: TObject);
var s: String;
begin
  I2c1:= TI2c.Create;
  I2c1.Init(1);
  Tf1:= TTf.Create();
  Tf1.Init(@I2c1, $96);
end;

procedure TForm1.Timer1Timer(Sender: TObject);
var s: String;
begin
  str(Tf1.GetTemperature:6:1, s);
  EditTemp.Text:= s;
  str(Tf1.GetTos:6:1, s);
  EditTos.Text:= s;
  str(Tf1.GetThyst:6:1, s);
  EditThyst.Text:= s;
end;

procedure TForm1.EditTosOnEnter(Sender: TObject);
begin
    Timer1.Enabled := False;
end;
```

```
procedure TForm1.EditThystOnEnter(Sender: TObject);
begin
  Timer1.Enabled := False;
end;

procedure TForm1.EditThystOnExit(Sender: TObject);
var  temp: real;
     code: Integer;
begin
  Val(EditThyst.Text, temp, code);
  if code = 0 then Tf1.SetThyst(temp);
  Timer1.Enabled := True;
end;

procedure TForm1.EditTosOnExit(Sender: TObject);
var  temp: real;
     code: Integer;
begin
     Val(EditTos.Text, temp, code);
     if code = 0 then Tf1.SetTos(temp);
     Timer1.Enabled := True;
end;

procedure TForm1.BitBtnAbbrechenClick(Sender: TObject);
begin
  Close;
end;

procedure TForm1.OnClose(Sender: TObject;
                    var Action: TCloseAction);
begin
     Tf1.Destroy;
     I2c1.Exit;
     I2c1.Destroy;
     Action := caFree // Odstranit formulář z paměti
end;

end.
```