Einführung zu der DLL für die USB Experiment Interface Board K8055

Die K8055 Interface Board hat 5 digitale Eingangskanäle und 8 digitale Ausgangskanäle. Zusätzlich gibt es 2 analoge Eingänge, 2 analoge Spannungsausgänge und 2 PWM (Pulse Width Modulation)-Ausgänge mit 8-Bit Auflösung.
Die Anzahl der Ein-/Ausgänge können Sie noch vergrößern, indem Sie (bis maximal 4) Karten an die USB-Stecker des Computers anschließen. Jeder Karte wird eine eigene Identifikationsnummer mittels 2 Steckbrücken, SK 5 und SK6, zugewiesen (siehe Tabelle 1 unten für die Nummerierung).

Alle Kommunikationsprogramme sind in einer Dynamic Link Library (DLL) K8055D.DLL gespeichert.

Dieses Dokument beschreibt alle Arbeitsweisen und Funktionen der DLL, die für Ihre Applikationsprogramme verfügbar sind. Mithilfe dieser Arbeitsweisen und Funktionen, die durch die DLL exportiert werden, können Sie maßgeschneiderte Windows-Applikationen (98SE, 2000, Me, XP) in Delphi, Visual Basic, C++ Builder oder jedem anderen 32-Bit Windows-Hilfsprogramm für Applikationen, das DLL unterstützt, schreiben.

Eine vollständige Übersicht dieser Funktionen und Methoden, die durch K8055D.DLL exportiert werden, können Sie weiter in diesem Dokument finden. Am Ende dieses Dokumentes gibt es eine Liste von Programmbeispielen. Diese werden Ihnen helfen, einen Einblick in die Art und Weise zu gewinnen, auf die Sie Ihre eigenen Applikationen entwickeln können. Die Beispiele sind in Delphi, Visual Basic und C++ Builder geschrieben. In der Übersicht werden die Funktionen und die Methoden der DDL erklärt.

Bitte achten Sie darauf, dass die Beispiele in den Beschreibungen der Funktionen und Methoden für Delphi geschrieben sind.

SK5	SK6	CARD ADDRESS (ADRESSE DER KARTE)
EIN	EIN	0
AUS	EIN	1
EIN	AUS	2
AUS	AUS	3

TABELLE 1: Steckbrücke SK5, SK6: Einstellungen

Anmerkung: Die Einstellungen müssen bestimmt werden, bevor das USB-Kabel an die K8055-Karte angeschlossen ist oder der PC angeschaltet wird.

Übersicht der Arbeitsweisen und Funktionen der K8055D.DLL

Allgemeine Arbeitsweisen

`OpenDevice(CardAddress)`	*Öffnet den Kommunikationslink zu dem K8055-Gerät*
`CloseDevice`	*Schließt den Link zu dem K8055-Gerät*

Analog in Digital konvertieren: Arbeitsweise

`ReadAnalogChannel(Channelno)`	*Liest den Status eines analogen Eingangskanals*
`ReadAllAnalog(Data1, Data2)`	*Liest den Status der beiden analogen Eingangskanäle*

Digital in Analog konvertieren: Arbeitsweise

`OutputAnalogChannel(Channel,Data)`	*Stellt den analogen Ausgangskanal nach den Daten ein*
`OutputAllAnalog(Data1,Data2)`	*Stellt die beiden Ausgangskanäle nach den Daten ein*
`ClearAnalogChannel(Channel)`	*Stellt den analogen Ausgangskanal auf das Minimum ein*
`ClearAllAnalog`	*Stellt alle Ausgangskanäle auf das Minimum ein*
`SetAnalogChannel(Channel)`	*Stellt den analogen Ausgangskanal auf das Maximum ein*
`SetAllAnalog`	*Stellt alle Ausgangskanäle auf das Maximum ein*

Digitaler Ausgang: Arbeitsweise

`WriteAllDigital(Data)`	*Stellt alle digitalen Ausgänge nach den Daten ein*
`ClearDigitalChannel(Channel)`	*Löscht den Ausgangskanal*
`ClearAllDigital`	*Löscht alle Ausgangskanäle*
`SetDigitalChannel(Channel)`	*Stellt den Ausgangskanal ein*
`SetAllDigital`	*Stellt alle Ausgangskanäle ein*

Digitaler Eingang: Arbeitsweise und Funktionen

`ReadDigitalChannel(Channel)`	*Liest den Status des Eingangskanals*
`ReadAllDigital(Buffer)`	*Liest den Status aller Eingangskanäle*

Zählerfunktionen und Arbeitsweise

`ResetCounter(CounterNr)`	*Stellt den 16-bit Pulszähler 1 oder 2 auf Null*
`ReadCounter(CounterNr)`	*Liest den Inhalt des Pulszählers 1 oder 2*
`SetCounterDebounceTime(CounterNr, DebounceTime)`	*Stellt die Entprellung (debounce time) für den Zähler ein*

Funktionen und Arbeitsweisen der K8055D.DLL

OpenDevice

Syntax
```
FUNCTION OpenDevice(CardAddress: Longint): Longint;
```

Parameter
CardAddress: Wert zwischen 0 und 3, der mit der Steckbrücke (SK5, SK6) auf K8055 übereinstimmt (siehe Tabelle 1).

Ergebnis
Longint: Bei Erfolg wird der Rückgabewert die Adresse der Karte sein, die von der K8055-Hardware gelesen wird.

Beschreibung
Öffnet den Kommunikationslink mit der K8055-Karte. Ladet die Treiber, die zur Kommunikation über den USB-Port erforderlich sind. Dieses Verfahren soll erst erledigt sein, bevor Sie versuchen eine Kommunikation mit K8055 zustande zu bringen.

Diese Funktion kann auch verwendet werden um die tätige K8055-Karte zu selektieren und sie zu lesen und die Daten zu schreiben. Alle Kommunikationsroutinen nach der Selektion werden dieser bestimmten Karte zugewiesen bis eine andere Karte selektiert wird.

Beispiel
```
  var h: longint;
  BEGIN
     h:=OpenDevice(0); // Opens the link to card number 0
  END;
```

CloseDevice

Syntax
```
PROCEDURE CloseDevice;
```

Beschreibung
Entladet die Kommunikationsroutinen für die K8055-Karte und entladet den Treiber für die Kommunikation über den USB-Port. Diese Handlung ist der letzte Schritt im Applikationsprogramm vor Beendigung.

Beispiel
```
    BEGIN
     CloseDevice; // The communication to the K8055 device is closed
  END;
```

ReadAnalogChannel

Syntax
```
FUNCTION ReadAnalogChannel (Channel: Longint): Longint;
```

`Channel:` Wert zwischen 1 und 2, der mit dem zu lesenden Status des AD-Kanals, übereinstimmt.

Ergebnis
`Longint:` Die entsprechenden Daten vom 'Analogue to Digital Converter' werden gelesen.

Beschreibung
Die Eingangsspannung des selektierten 8-Bit 'Analogue to Digital Converter' –Kanals wird in einen Wert, der zwischen 0 und 255 liegt, umgesetzt.

Beispiel
```
  var data: longint;
  BEGIN
      data := ReadAnalogChannel(1);
      // AD channel 1 is read to variable 'data'
  END;
```

ReadIAllAnalog

Syntax
```
PROCEDURE ReadAllAnalog(var Data1, Data2: Longint);
```

Parameter
`Data1, Data2:` 'Pointers' auf die 'long integers' wo die Daten gelesen werden.

Beschreibung
Der Status von den beiden 'Analogue to Digital Converters' wird in einem Datenfeld von 'long integers' gelesen.

Beispiel
```
procedure TForm1.Button1Click(Sender: TObject);
var Data1, Data2: Longint;
begin
  ReadAllAnalog(Data1, Data2);  // Read the data from the K8055
  Label1.caption:=inttostr(Data1); // Display CH1 data
  Label2.caption:=inttostr(Data2); // Display CH2 data
end;
```

OutputAnalogChannel

Syntax
```
PROCEDURE OutputAnalogChannel(Channel: Longint; Data: Longint);
```

Parameter
`Channel:` Wert zwischen 1 und 2, der mit der 8-Bit DA Kanal-Nummer, deren Daten eingestellt werden sollen, übereinstimmt.
`Data:` Wert zwischen 0 und 255, der zu dem 8-Bit 'Digital to Analogue Converter' geschickt werden soll.

Beschreibung
Der angegebene 8-Bit 'Digital to Analogue Converter'-Kanal ist nach den neuen Daten geändert worden. Das bedeutet, dass die Daten einer spezifischen Spannung entsprechen. Der Wert 0 stimmt mit der minimalen Ausgangsspannung überein (0 Volt) und der Wert 255 ist die maximale Ausgangsspannung (+5V).

Ein Wert von 'Data' zwischen diesen 2 Äußersten kann mit der folgenden Formel wiedergegeben werden: Data / 255 x 5V.

Beispiel

```
    BEGIN
    OutputAnalogChannel (1,127);
    // DA channel 1 is set to 2.5V
  END;
```

OutputAllAnalog

Syntax

```
PROCEDURE OutputAllAnalog(Data1: Longint; Data2: Longint);
```

Parameter

`Data1, Data2:` Wert zwischen 0 und 255, der zu dem '8-Bit Digital to Analogue Converter' geschickt werden soll.

Beschreibung

Die beiden 'Digital to Analogue Converter'-Kanäle sind nach diesen neuen Daten geändert worden. Das heißt, dass die Daten einer spezifischen Spannung entsprechen. Der Wert 0 stimmt mit der minimalen Ausgangsspannung (0 Volt) überein und der Wert 255 stimmt mit der maximalen Ausgangsspannung (+5V) überein. Der Wert 'Data1' oder 'Data2' zwischen diesen 2 Äußersten kann mit der folgenden Formel wiedergegeben werden: Data / 255 x 5V.

Beispiel

```
    BEGIN
    OutputAllAnalog(127, 255);
    // DA channel 1 is set to 2.5V and channel 2 is set to 5V
  END;
```

ClearAnalogChannel

Syntax

```
PROCEDURE ClearAnalogChannel(Channel: Longint);
```

Parameter

`Channel:` Der Wert zwischen 1 und 2, der mit der '8-Bit DA' Kanalnummer, in der die Daten gelöscht werden sollen, übereinstimmt.

Beschreibung

Der ausgewählte DA-Kanal ist auf die minimale Ausgangsspannung (0 Volt) eingestellt.

Beispiel

```
  BEGIN
    ClearAnalogChannel (1); // DA channel 1 is set to 0V
  END;
```

ClearAllAnalog

Syntax
```
PROCEDURE ClearAllAnalog;
```

Beschreibung
Die beiden DA-Kanäle sind auf die minimale Eingangsspannung (0 Volt) eingestellt.

Beispiel
```
   BEGIN
    ClearAllAnalog; // All DA channels 1 and 2 are set to 0V
  END;
```

SetAnalogChannel

Syntax
```
PROCEDURE SetAnalogChannel(Channel: Longint);
```

PParameter
`Channel:` Wert zwischen 1 und 2, der mit der '8-Bit DA' Kanalnummer, in der die Daten auf Maximum eingestellt werden sollen, übereinstimmt.

Beschreibung
Der ausgewählte 8-Bit 'Digital to Analogue Converter'-Kanal ist auf die maximale Ausgangsspannung eingestellt.

Beispiel
```
  BEGIN
    SetAnalogChannel(1); // DA channel 1 is set to +5V
  END;
```

SetAllAnalog

Syntax
```
PROCEDURE SetAllAnalog;
```

Beschreibung
Alle Kanäle der '8-Bit Digital to Analogue Converters' sind auf die maximale Ausgangsspannung eingestellt.

Beispiel
```
  BEGIN
    SetAllAnalog; // DA channels 1 and 2 are set to +5V
  END;
```

WriteAllDigital

Syntax

```
PROCEDURE WriteAllDigital(Data: Longint);
```

Parameter
`Data:` Wert zwischen 0 und 255, der zu dem Ausgangsport geschickt wird (8 Kanäle).

Beschreibung
Die Kanäle des digitalen Ausgangsports werden mit dem Status der entsprechenden Bits im Datenparameter aktualisiert. Ein hoher Pegel (1) bedeutet, dass der Mikrocontroller IC1-Ausgang bestimmt ist, ein niedriger Pegel (0) bedeutet, dass der Ausgang freigegeben ist.

Beispiel
```
  BEGIN
    WriteAllDigital(7);
    // Output channels 1...3 are on, output channels 4...8 are off
  END;
```

ClearDigitalChannel

Syntax
```
PROCEDURE ClearDigitalChannel(Channel: Longint);
```

Parameter
`Channel:` Wert zwischen 1 und 8, der mit dem zu löschenden Ausgangskanal übereinstimmt.

Beschreibung
Der selektierte Kanal wird freigegeben (gelöscht).

Beispiel
```
  BEGIN
    ClearIOchannel(4); // Digital output channel 4 is OFF
  END;
```

ClearAllDigital

Syntax
```
PROCEDURE ClearAllDigital;
```

Ergebnis
Alle digitalen Ausgänge werden freigegeben.

Beispiel
```
  BEGIN
    ClearAllDigital; // All Output channels 1 to 8 are OFF
  END;
```

SetDigitalChannel

Syntax

```
PROCEDURE SetDigitalChannel(Channel: Longint);
```

Parameter
`Channel:` Wert zwischen 1 und 8, der mit dem einzustellenden Ausgangskanal übereinstimmt.

Beschreibung
Der gewählte Ausgangskanal wird eingestellt.

Beispiel
```
  BEGIN
    SetDigitalChannel(1); // Digital output channel 3 is ON
  END;
```

SetAllDigital

Syntax
```
PROCEDURE SetAllDigital;
```

Beschreibung
Alle digitalen Ausgangskänale werden eingestellt.

Beispiel
```
  BEGIN
    SetAllDigital; // All Output channels are ON
  END;
```

ReadDigitalChannel

Syntax
```
FUNCTION ReadDigitalChannel(Channel: Longint): Boolean;
```

Parameter
`Channel:` Wert zwischen 1 und 5, der mit dem zu lesenden Eingangskanal übereinstimmt.

Ergebnis
`Boolean`: TRUE bedeutet, dass der Kanal eingestellt ist und FALSE bedeutet, dass der Kanal freigegeben ist.

Beschreibung
Der Status des gewählten Eingangskanals wird gelesen.

Beispiel
```
  var status: boolean;
  BEGIN
    status := ReadIOchannel(2); // Read Input channel 2
  END;
```

ReadAllDigital

Syntax
```
FUNCTION ReadAllDigital: Longint;
```

Ergebnis
`Longint:` Die 5 LSB stimmen mit dem Status der Eingangskanäle überein. 'Hoch' (1) bedeutet das, dass der Kanal HIGH ist, 'niedrig' (0) bedeutet, dass der Kanal LOW ist.

Beschreibung
Die Funktion gibt den Status der digitalen Eingänge wieder.

Beispiel
```
  var status: longint;
  BEGIN
    status := ReadAllDigital; // Read the Input channels
  END;
```

ResetCounter

Syntax
```
PROCEDURE ResetCounter(CounterNumber: Longint);
```

Parameter
`CounterNumber:` Wert 1 oder 2, der mit dem 'auf Null' zu setzenden Zähler übereinstimmt.

Beschreibung
Der gewählte Zähler wird auf Null gesetzt.

Beispiel
```
  BEGIN
    ResetCounter(2); // Reset the counter number 2
  END;
```

ReadCounter

Syntax
```
FUNCTION ReadCounter(CounterNumber: Longint): Longint;
```

Parameter
`CounterNumber:` Wert 1 oder 2, der mit dem zu lesenden Zähler übereinstimmt.

Ergebnis
`Longint:` Der Inhalt des 16-Bit-Impulszählers.

Beschreibung
Die Funktion gibt den Status des 16-Bit-Impulszählers wieder.
Der Zähler 1 zählt die Impulse, die in Eingang I1 eingegeben werden; Zähler 2 zählt die Impulse, die in Eingang 2 eingegeben werden.

Beispiel

```
var pulses: longint;
BEGIN
  pulses := ReadCounter(2); // Read the counter number 2
END;
```

SetCounterDebounceTime

Syntax

```
PROCEDURE SetCounterDebounceTime(CounterNr, DebounceTime: Longint);
```

Parameter

`CounterNumber:` Wert 1 oder 2, der mit dem zu einstellenden Zähler übereinstimmt.
`DebounceTime:` 'Debounce' (Entprellung) Zeit für den Impulszähler.
Der `DebounceTime` –Wert stimmt mit der für den Impulszähler einzustellenden 'Debounce' Zeit in Millisekunden (ms) überein. Der Wert der 'Debounce Time' kann zwischen 0 und 5000 liegen.

Beschreibung

Die Zähler-Eingänge werden in der Software entprellt um falsches Auslösen, wenn mechanische Schalter oder Relaiseingänge verwendet werden, zu vermeiden. Die 'Debounce' Zeit ist dieselbe für ansteigende und fallende Flanken. Die Standard 'Debounce' Zeit ist 2ms. Das heißt, dass der Zähler-Eingang mindestens 2 ms stabil sein muss, bevor er anerkannt wird; und wenn man die maximale Zählrate von ungefähr 200 Zählungen pro Sekunde berücksichtigt.

Wenn die 'Debounce-Zeit auf 0 eingestellt wurde, dann ist die maximale Zählrate ungefähr 2000 Zählungen pro Sekunde.

Beispiel

```
BEGIN
  SetCounterDebounceTime(1,100);
  // The debounce time for counter number 1 is set to 100ms
END;
```

Die K8055D.DLL in Delphi verwenden

In diesem Anwendungsbeispiel werden die Arbeitsweisen und die Funktionen der K8055.DLL erklärt. Auch werden die 2 wichtigsten DLL -Signale: **`OpenDevice`** und **`CloseDevice`** verdeutlicht.

```
unit K8055;

interface

uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
  StdCtrls, ExtCtrls, ComCtrls;

type
  TForm1 = class(TForm)
    GroupBox1: TGroupBox;
    SK6: TCheckBox;
    SK5: TCheckBox;
    Button1: TButton;
    Label1: TLabel;
    procedure FormClose(Sender: TObject; var Action: TCloseAction);
    procedure Button1Click(Sender: TObject);

  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Form1: TForm1;
  timed:boolean;

implementation

{$R *.DFM}
function OpenDevice(CardAddress: Longint): Longint; stdcall; external 'K8055d.dll';
procedure CloseDevice; stdcall; external 'K8055d.dll';
function ReadAnalogChannel(Channel: Longint):Longint; stdcall; external 'K8055d.dll';
procedure ReadAllAnalog(var Data1, Data2: Longint); stdcall; external 'K8055d.dll';
procedure OutputAnalogChannel(Channel: Longint; Data: Longint); stdcall; external
'K8055d.dll';
procedure OutputAllAnalog(Data1: Longint; Data2: Longint); stdcall; external 'K8055d.dll';
procedure ClearAnalogChannel(Channel: Longint); stdcall; external 'K8055d.dll';
procedure ClearAllAnalog; stdcall; external 'K8055d.dll';
procedure SetAnalogChannel(Channel: Longint); stdcall; external 'K8055d.dll';
procedure SetAllAnalog; stdcall; external 'K8055d.dll';
procedure WriteAllDigital(Data: Longint);stdcall;  external 'K8055d.dll';
procedure ClearDigitalChannel(Channel: Longint); stdcall; external 'K8055d.dll';
procedure ClearAllDigital; stdcall; external 'K8055d.dll';
procedure SetDigitalChannel(Channel: Longint); stdcall; external 'K8055d.dll';
procedure SetAllDigital; stdcall; external 'K8055d.dll';
function ReadDigitalChannel(Channel: Longint): Boolean; stdcall; external 'K8055d.dll';
function ReadAllDigital: Longint; stdcall; external 'K8055d.dll';
function ReadCounter(CounterNr: Longint): Longint; stdcall; external 'K8055d.dll';
procedure ResetCounter(CounterNr: Longint); stdcall; external 'K8055d.dll';
procedure SetCounterDebounceTime(CounterNr, DebounceTime:Longint); stdcall; external
'K8055d.dll';

procedure TForm1.FormClose(Sender: TObject; var Action: TCloseAction);
begin
  CloseDevice;
end;

procedure TForm1.Button1Click(Sender: TObject);
var h,CardAddr:longint;
begin
  CardAddr:= 3-(integer(SK5.Checked) + integer(SK6.Checked) * 2);
  h:= OpenDevice(CardAddr);
  case h of
    0..3: label12.caption:='Card '+ inttostr(h)+' connected';
```

```
-1: label12.caption:='Card '+ inttostr(CardAddr)+' not found';
  end;
end;

end.
```

Die K8055D.DDL in Visual Basic verwenden

In diesem Anwendungsbeispiel werden die Arbeitsweisen und die Funktionen der K8055.DLL erklärt. Auch werden die 2 wichtigsten DLL -Signale: **`OpenDevice`** and **`CloseDevice`** verdeutlicht.

Hinweis: Achten Sie darauf, dass Sie die Datei in den Windows' SYSTEM32'-Ordner kopieren.

```
Option Explicit
Private Declare Function OpenDevice Lib "k8055d.dll" (ByVal CardAddress As Long) As Long
Private Declare Sub CloseDevice Lib "k8055d.dll" ()
Private Declare Function ReadAnalogChannel Lib "k8055d.dll" (ByVal Channel As Long) As Long
Private Declare Sub ReadAllAnalog Lib "k8055d.dll" (Data1 As Long, Data2 As Long)
Private Declare Sub OutputAnalogChannel Lib "k8055d.dll" (ByVal Channel As Long, ByVal Data As
Long)
Private Declare Sub OutputAllAnalog Lib "k8055d.dll" (ByVal Data1 As Long, ByVal Data2 As
Long)
Private Declare Sub ClearAnalogChannel Lib "k8055d.dll" (ByVal Channel As Long)
Private Declare Sub SetAllAnalog Lib "k8055d.dll" ()
Private Declare Sub ClearAllAnalog Lib "k8055d.dll" ()
Private Declare Sub SetAnalogChannel Lib "k8055d.dll" (ByVal Channel As Long)
Private Declare Sub WriteAllDigital Lib "k8055d.dll" (ByVal Data As Long)
Private Declare Sub ClearDigitalChannel Lib "k8055d.dll" (ByVal Channel As Long)
Private Declare Sub ClearAllDigital Lib "k8055d.dll" ()
Private Declare Sub SetDigitalChannel Lib "k8055d.dll" (ByVal Channel As Long)
Private Declare Sub SetAllDigital Lib "k8055d.dll" ()
Private Declare Function ReadDigitalChannel Lib "k8055d.dll" (ByVal Channel As Long) As
Boolean
Private Declare Function ReadAllDigital Lib "k8055d.dll" () As Long
Private Declare Function ReadCounter Lib "k8055d.dll" (ByVal CounterNr As Long) As Long
Private Declare Sub ResetCounter Lib "k8055d.dll" (ByVal CounterNr As Long)
Private Declare Sub SetCounterDebounceTime Lib "k8055d.dll" (ByVal CounterNr As Long, ByVal
DebounceTime As Long)


Private Sub Connect_Click()
    Dim CardAddress As Long
    Dim h As Long
    CardAddress = 0
    CardAddress = 3 - (Check1(0).Value + Check1(1).Value * 2)
    h = OpenDevice(CardAddress)
    Select Case h
        Case 0, 1, 2, 3
            Label1.Caption = "Card " + Str(h) + " connected"
        Case -1
            Label1.Caption = "Card " + Str(CardAddress) + " not found"
    End Select
End Sub


Private Sub Form_Terminate()
    CloseDevice
End Sub
```

Die K8055D.DLL in Borland C++ Builder verwenden

In diesem Anwendungsbeispiel werden die Arbeitsweisen und die Funktionen der K8055.DLL erklärt. Auch werden die 2 wichtigsten DLL -Signale: **`OpenDevice`** and **`CloseDevice`** verdeutlicht.

```
//Listing K8055D.h
#ifdef __cplusplus
extern "C" {
#endif

#define FUNCTION __declspec(dllimport)

FUNCTION long __stdcall OpenDevice(long CardAddress);
FUNCTION __stdcall CloseDevice();
FUNCTION long __stdcall ReadAnalogChannel(long Channel);
FUNCTION __stdcall ReadAllAnalog(long *Data1, long *Data2);
FUNCTION __stdcall OutputAnalogChannel(long Channel, long Data);
FUNCTION __stdcall OutputAllAnalog(long Data1, long Data2);
FUNCTION __stdcall ClearAnalogChannel(long Channel);
FUNCTION __stdcall ClearAllAnalog();
FUNCTION __stdcall SetAnalogChannel(long Channel);
FUNCTION __stdcall SetAllAnalog();
FUNCTION __stdcall WriteAllDigital(long Data);
FUNCTION __stdcall ClearDigitalChannel(long Channel);
FUNCTION __stdcall ClearAllDigital();
FUNCTION __stdcall SetDigitalChannel(long Channel);
FUNCTION __stdcall SetAllDigital();
FUNCTION bool __stdcall ReadDigitalChannel(long Channel);
FUNCTION long __stdcall ReadAllDigital();
FUNCTION long __stdcall ReadCounter(long CounterNr);
FUNCTION __stdcall ResetCounter(long CounterNr);
FUNCTION __stdcall SetCounterDebounceTime(long CounterNr, long DebounceTime);

#ifdef __cplusplus
}
#endif


//Listing Unit1.cpp
//---------------------------------------------------------------------------

#include <vcl.h>
#pragma hdrstop

#include "Unit1.h"
#include "K8055D.h"
//---------------------------------------------------------------------------
#pragma package(smart_init)
#pragma resource "*.dfm"
TForm1 *Form1;
//---------------------------------------------------------------------------

__fastcall TForm1::TForm1(TComponent* Owner)
        : TForm(Owner)
{

}
//---------------------------------------------------------------------------

void __fastcall TForm1::Connect1Click(TObject *Sender)
{
  int CardAddr = 3 - (int(CheckBox1->Checked) + int(CheckBox2->Checked) * 2);
  int h = OpenDevice(CardAddr);
  switch (h) {
    case  0 :
    case  1 :
    case  2 :
    case  3 :
      Label1->Caption = "Card " + IntToStr(h) + " connected";
      break;
    case  -1 :
```

```
Label1->Caption = "Card " + IntToStr(CardAddr) + " not found";
  }
}
//---------------------------------------------------------------------------

void __fastcall TForm1::FormClose(TObject *Sender, TCloseAction &Action)
{
  CloseDevice;
}
//---------------------------------------------------------------------------
```