DMX Controller K8062

The K8062 is an USB connected DMX lighting controller.

Features:
- 512 DMX output channels
- USB bus powered (<100mA)
- Output: 3-pin XLR connector
- LED indicators for the presence of USB voltage and DMX data transfer
- Supports Windows 98SE, ME, Windows 2000, Windows XP
- Stand-alone battery operated test mode for DMX fixture testing

About the DMX

DMX is a standard protocol for lighting equipment. You can use DMX to control light equipment remotely. DMX can be used to control dimmers, colour scrollers, scanners, strobes, projectors, smoke machines and many other stage equipment.

Every DMX device has two XLR connectors, input and output. The DMX cable from the DMX controller (K8062) goes to the input of the first light device. The output connector of the light device is used to connect the next device in the chain.

Every DMX device has its own DMX address and a DMX range. If you have a fixture that performs many different functions it will require multiple DMX channels.

The K8062 controller supports up to 512 addresses (or channels).

The DMX address indicates the starting channel for control of that device. Each piece of equipment receiving the DMX signal has to be told which of the 512 channels it is responding to. This is done by setting the DMX address on each receiving unit. On most units there are small DIP switches to set the start address.

If your first device has a start address of 1 and uses 6 channels. The next DMX device should start at address 7 or greater to prevent a conflict.

To synchronize the operation of any two or more similar fixtures they may share a single address.

The controlled light devices may be connected in any order. You do not have to connect the units in the same order as the DMX address.

All the data for the individual 512 DMX channels is sent one after other, beginning with channel 1. Not all the 512 channels need to be output per packet. As soon as one packet is finished, another can begin. If nothing has changed the same data will be sent out over and over again. The fewer channels are used, the higher is the refresh rate of information.

Each DMX channel contains a level represented by an 8-bit word. The 8-bit word allows 256 individual levels for each device to be transmitted ("0" = channel off, "255" = channel full on). Many moving lights for instance use 2 channels to provide a 16-bit resolution.

The DMX line must be correctly terminated. Some equipment has a switchable line terminating resistor. In these case make sure that only the last item in the chain has its terminating switch set on. If the last item does not have a terminating switch then an external line terminating plug is needed.
The following parts are needed for the terminator plug:

- One male 3 Pin XLR connector
- One 120 ohm 0.5W resistor

To make the line terminator, solder the resistor across pins 2 and 3 of the XLR connector.

Make your own software for the K8062 with the Dynamic Link Library K8062D.DLL

The K8062D.DLL is a 32 bit Windows DLL. This document describes the procedures provided by the DLL that are available for your application programme. Calling the functions and procedures exported by the DLL, you may write custom Windows (98SE, 2000, Me, XP) applications in Delphi, Visual Basic, C++ Builder or any other 32-bit Windows application development tool that supports calls to a DLL.

A complete overview of the procedures that are exported by the K8062D.DLL follows. At the end of this document there are listings of example programmes in order to gain an insight as to how to construct your own application programmes. The examples are written in Delphi, Visual Basic, Visual Basic.NET and C++ Builder. In the listings there are full declarations for the DLL function and procedures.

Note that all the examples in the function and procedure description section are written for Delphi.

Overview of the Procedures of the K8062D.DLL

General procedures

`StartDevice`	*Opens the communication link to the K8062 device*
`StopDevice`	*Closes the link to the K8062 device*

Output procedures

`SetChannelCount(Count)`	*Sets the maximum DMX channel in use*
`SetData(Channel, Data)`	*Sets the data value for the DMX channel*

Procedures of the K8062D.DLL

StartDevice

Syntax
```
PROCEDURE StartDevice;
```

Description
Opens the communication link to the K8062 card. Loads the drivers needed to communicate via the USB port. This procedure must be performed before any attempts to communicate with the K8062 card.

Example
```
BEGIN
   StartDevice; // Opens the link to the card
END;
```

StopDevice

Syntax
```
PROCEDURE StopDevice;
```

Description
Unloads the communication routines for K8062 card and unloads the driver needed to communicate via the USB port. This is the last action of the application program before termination.

Example
```
BEGIN
   StopDevice; // The communication to the K8062 device is closed
END;
```

SetChannelCount

Syntax
```
PROCEDURE SetChannelCount(Count: Longint)
```

Description
This function sets the number of the max. DMX channel in use.
Count is in the range 8 ... 512.
This function allows the application to significantly improve the execution time of SetData when less than 512 channels are required.

Example
```
BEGIN
    SetChannelCount(32); // Sets the max. DMX channel in use to 32
END;
```

SetData

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

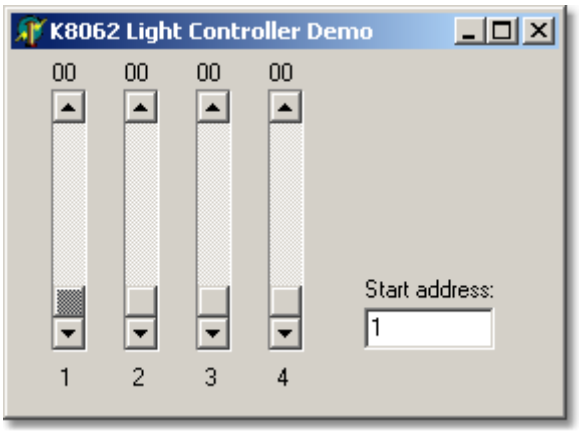
Description
This function sets the level of a DMX channel.
Channel is in the range 1 to Count.
Data is in the range 0 to 255.

Example
```
BEGIN
    SetData(1,127); // Sets channel #1 data to 127 (50% of the max. value)
END;
```

Application examples

In the following application examples there are the declarations of the K8062D.DLL procedures for Delphi, Visual Basic and Borland C++Builder.
The listings show the use of the `StartDevice` and `StopDevice` procedures and how to output DMX data to individual DMX channels and how to set the number of the DMX channels in use.

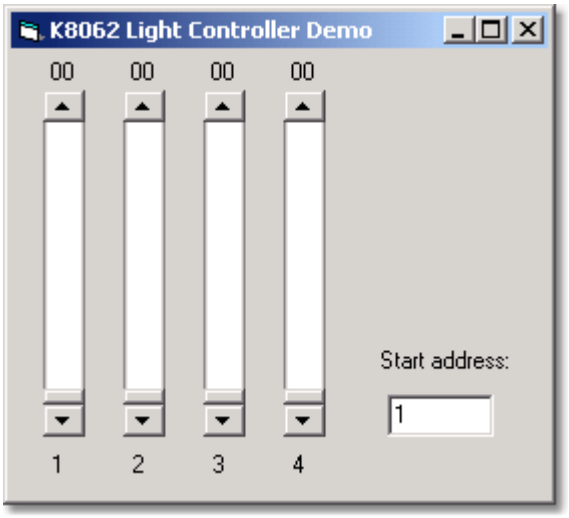
Using the K8062D.DLL in Delphi




```
unit K8062_1;

interface

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

type
  TForm1 = class(TForm)
    ScrollBar1: TScrollBar;
    ScrollBar2: TScrollBar;
    ScrollBar3: TScrollBar;
    ScrollBar4: TScrollBar;
    Edit1: TEdit;
    Label1: TLabel;
    Label2: TLabel;
    Label3: TLabel;
    Label4: TLabel;
    Label5: TLabel;
    Label6: TLabel;
    Label7: TLabel;
    Label8: TLabel;
    Label9: TLabel;
    procedure Edit1Change(Sender: TObject);
    procedure FormCreate(Sender: TObject);
    procedure FormClose(Sender: TObject; var Action: TCloseAction);
    procedure ScrollBar1Change(Sender: TObject);
```

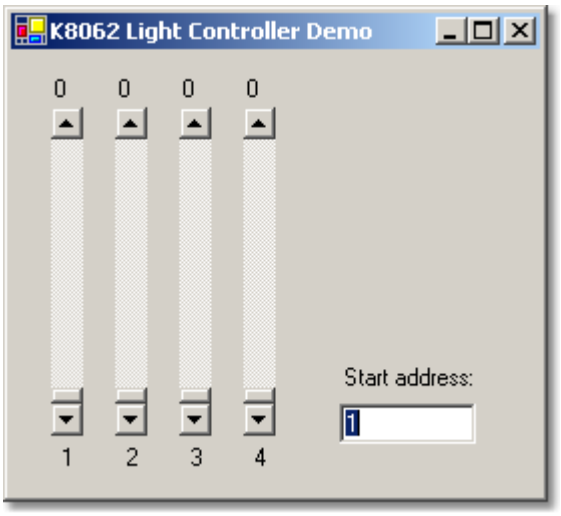
```
procedure ScrollBar2Change(Sender: TObject);
    procedure ScrollBar3Change(Sender: TObject);
    procedure ScrollBar4Change(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Form1: TForm1;
  StartAddress: Longint;

implementation

{$R *.DFM}
PROCEDURE StartDevice; stdcall; external 'K8062d.dll';
PROCEDURE SetData(Channel: Longint ; Data: Longint); stdcall; external 'K8062d.dll';
PROCEDURE SetChannelCount(Count: Longint); stdcall; external 'K8062d.dll';
PROCEDURE StopDevice; stdcall; external 'K8062d.dll';


procedure TForm1.FormCreate(Sender: TObject);
begin
  StartDevice;
  StartAddress:=1;
end;

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

procedure TForm1.Edit1Change(Sender: TObject);
begin
  if (StrToInt(Edit1.Text)>0) and (StrToInt(Edit1.Text)<509) then
  begin
    SetChannelCount(StartAddress+3);
    StartAddress:=StrToInt(Edit1.Text);
    Label5.Caption:=IntToStr(StartAddress);
    Label6.Caption:=IntToStr(StartAddress+1);
    Label7.Caption:=IntToStr(StartAddress+2);
    Label8.Caption:=IntToStr(StartAddress+3);
  end;
end;

procedure TForm1.ScrollBar1Change(Sender: TObject);
begin
  Label1.Caption:=IntToStr(255-ScrollBar1.Position);
  SetData(StartAddress, 255-ScrollBar1.Position);
end;

procedure TForm1.ScrollBar2Change(Sender: TObject);
begin
  Label2.Caption:=IntToStr(255-ScrollBar2.Position);
  SetData(StartAddress+1, 255-ScrollBar2.Position);
end;

procedure TForm1.ScrollBar3Change(Sender: TObject);
begin
  Label3.Caption:=IntToStr(255-ScrollBar3.Position);
  SetData(StartAddress+2, 255-ScrollBar3.Position);
end;

procedure TForm1.ScrollBar4Change(Sender: TObject);
begin
  Label4.Caption:=IntToStr(255-ScrollBar4.Position);
  SetData(StartAddress+3, 255-ScrollBar4.Position);
end;

end.
```

Using the K8062D.DLL in Visual Basic




```
Option Explicit
Private Declare Sub StartDevice Lib "k8062d.dll" ()
Private Declare Sub SetData Lib "k8062d.dll" (ByVal Channel As Long, ByVal Data As Long)
Private Declare Sub SetChannelCount Lib "k8062d.dll" (ByVal Count As Long)
Private Declare Sub StopDevice Lib "k8062d.dll" ()

Private Sub Form_Load()
    StartDevice
End Sub

Private Sub Form_Terminate()
    StopDevice
End Sub

Private Sub UpdateLabels()
    Dim i As Integer
    Dim n As Integer
    n = 0
    If (Val(Text1.Text) > 0) And (Val(Text1.Text) < 510) Then
        For i = Val(Text1.Text) To Val(Text1.Text) + 3
            Label2(n) = Str(i)
            n = n + 1
        Next i
        SetChannelCount Val(Text1.Text) + 3
    End If
End Sub

Private Sub Text1_Change()
    UpdateLabels
End Sub

Private Sub VScroll1_Change(Index As Integer)
    Label3(Index) = Str(255 - VScroll1(Index).Value)
    SetData Val(Label2(Index)), Val(Label3(Index))
End Sub

Private Sub VScroll1_Scroll(Index As Integer)
    Label3(Index) = Str(255 - VScroll1(Index).Value)
    SetData Val(Label2(Index)), Val(Label3(Index))
End Sub
```

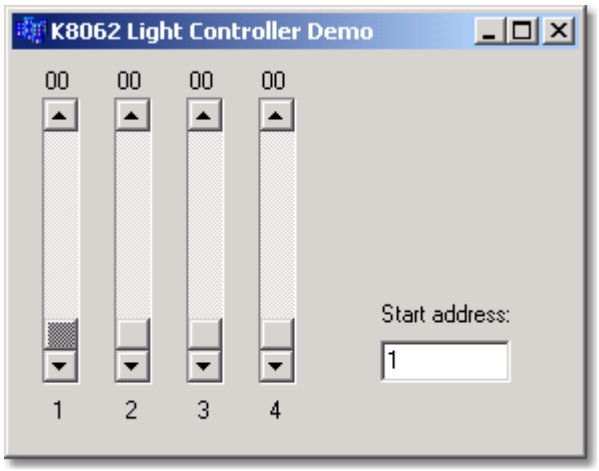
Using the K8062D.DLL in Visual Basic.NET




```
Public Class Form1
    Inherits System.Windows.Forms.Form

    Private Declare Sub StartDevice Lib "k8062d.dll" ()
    Private Declare Sub SetData Lib "k8062d.dll" (ByVal Channel As Integer, ByVal Data As
Integer)
    Private Declare Sub SetChannelCount Lib "k8062d.dll" (ByVal Count As Integer)
    Private Declare Sub StopDevice Lib "k8062d.dll" ()
    Dim StartAddress As Integer

    Private Sub Form1_Load(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles
MyBase.Load
        StartDevice()
        StartAddress = 1
    End Sub

    Private Sub Form1_Closed(ByVal sender As Object, ByVal e As System.EventArgs) Handles
MyBase.Closed
        StopDevice()
    End Sub
    Private Sub VScrollBar1_Scroll(ByVal sender As System.Object, ByVal e As
System.Windows.Forms.ScrollEventArgs) Handles VScrollBar1.Scroll
        Label1.Text = Str(255 - VScrollBar1.Value)
        SetData(StartAddress, 255 - VScrollBar1.Value)
    End Sub

    Private Sub VScrollBar2_Scroll(ByVal sender As System.Object, ByVal e As
System.Windows.Forms.ScrollEventArgs) Handles VScrollBar2.Scroll
        Label2.Text = Str(255 - VScrollBar2.Value)
        SetData(StartAddress + 1, 255 - VScrollBar2.Value)
    End Sub

    Private Sub VScrollBar3_Scroll(ByVal sender As System.Object, ByVal e As
System.Windows.Forms.ScrollEventArgs) Handles VScrollBar3.Scroll
        Label3.Text = Str(255 - VScrollBar3.Value)
        SetData(StartAddress + 2, 255 - VScrollBar3.Value)
    End Sub

    Private Sub VScrollBar4_Scroll(ByVal sender As System.Object, ByVal e As
System.Windows.Forms.ScrollEventArgs) Handles VScrollBar4.Scroll
        Label4.Text = Str(255 - VScrollBar4.Value)
        SetData(StartAddress + 3, 255 - VScrollBar4.Value)
    End Sub

    Private Sub TextBox1_TextChanged(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles TextBox1.TextChanged
        If Val(TextBox1.Text) > 0 And Val(TextBox1.Text) < 510 Then
            StartAddress = Val(TextBox1.Text)
            SetChannelCount(StartAddress + 3)
            Label5.Text = Str(StartAddress)
            Label6.Text = Str(StartAddress + 1)
```

```
Label7.Text = Str(StartAddress + 2)
            Label8.Text = Str(StartAddress + 3)
        End If
    End Sub
End Class
```

Using the K8062D.DLL in Borland C++Builder




```
//Listing K8062D.h
#ifdef __cplusplus
extern "C" {          /* Assume C declarations for C++ */
#endif

#define FUNCTION __declspec(dllimport)

FUNCTION __stdcall StartDevice();
FUNCTION __stdcall SetData(long Channel, long Data);
FUNCTION __stdcall SetChannelCount(long Count);
FUNCTION __stdcall StopDevice();

#ifdef __cplusplus
}
#endif

//Listing K8062.cpp
//---------------------------------------------------------------------------

#include <vcl.h>
#pragma hdrstop
#include "K8062D.h"
#include "K8062.h"
//---------------------------------------------------------------------------
#pragma package(smart_init)
#pragma resource "*.dfm"
TForm1 *Form1;
int StartAddress = 1;
//---------------------------------------------------------------------------
__fastcall TForm1::TForm1(TComponent* Owner)
  : TForm(Owner)
{
}
//---------------------------------------------------------------------------
void __fastcall TForm1::FormCreate(TObject *Sender)
{
  StartDevice();
}
//---------------------------------------------------------------------------
void __fastcall TForm1::FormClose(TObject *Sender, TCloseAction &Action)
{
```

```
StopDevice();
}
//---------------------------------------------------------------------------
void __fastcall TForm1::Edit1Change(TObject *Sender)
{
  if (StrToInt(Edit1->Text)>0)
  {
    if (StrToInt(Edit1->Text)<509)
    {
      SetChannelCount(StartAddress+3);
      StartAddress = StrToInt(Edit1->Text);
      Label5->Caption = IntToStr(StartAddress);
      Label6->Caption = IntToStr(StartAddress+1);
      Label7->Caption = IntToStr(StartAddress+2);
      Label8->Caption = IntToStr(StartAddress+3);
    }
  }
}
//---------------------------------------------------------------------------
void __fastcall TForm1::ScrollBar1Change(TObject *Sender)
{
  Label1->Caption = IntToStr(255-ScrollBar1->Position);
  SetData(StartAddress, 255-ScrollBar1->Position);
}
//---------------------------------------------------------------------------
void __fastcall TForm1::ScrollBar2Change(TObject *Sender)
{
  Label2->Caption = IntToStr(255-ScrollBar2->Position);
  SetData(StartAddress + 1, 255-ScrollBar2->Position);
}
//---------------------------------------------------------------------------
void __fastcall TForm1::ScrollBar3Change(TObject *Sender)
{
  Label3->Caption = IntToStr(255-ScrollBar3->Position);
  SetData(StartAddress + 2, 255-ScrollBar3->Position);
}
//---------------------------------------------------------------------------
void __fastcall TForm1::ScrollBar4Change(TObject *Sender)
{
  Label4->Caption = IntToStr(255-ScrollBar4->Position);
  SetData(StartAddress + 3, 255-ScrollBar4->Position);
}
//---------------------------------------------------------------------------
```