

Megawin

USB EasyPOD Library for

VB

User Manual

Index

User Manual.....	1
1. Introduction	3
2. Files Needed.....	3
3. How to Use EasyPOD DLL	3
3.1. Debug Mode:	3
3.2. Execution Mode:	3
4. How to use the Internal Built Function	4
4.1. Connect.....	4
4.2. Disconnect	4
4.3. WriteData	4
4.4. ReadData	4
4.5. ClearPODBuffer	4
5. Example	5
6. Revision History	7

1. Introduction

This document explains how to use EasyPOD DLL in the Visual Basic 6.0 environment for EasyPOD data transmission and reception.

2. Files Needed

2.1. EasyPOD.DLL

2.2. EasyPOD.bas

3. How to Use EasyPOD DLL

Activate the Visual Basic Software. Use Visual Basic 6.0 to develop the application program, please be guided by the following steps:

3.1. Debug Mode:

3.1.1. Please copy EasyPOD.DLL mentioned in section #2, into your hard disk. For example, if your operating system is Windows XP, please copy the file in C:\Windows directory.

3.1.2. Create a New Project. Go to the project menu then choose Add Module. Using the Add Module menu, add the EasyPOD.Bas file.

3.2. Execution Mode:

Copy the EasyPOD.DLL into the same directory where the execution file is located.

4. How to use the Internal Built Function

The DLL functions are described as following:

4.1. Connect

Function : Public Declare Function ConnectPOD Lib "EasyPOD.DLL" (ByRef m_EasyPod As MW_EasyPOD, ByVal Index As Long) As Long
Return : ERROR_SUCCESS(0) means connection is successful otherwise connection fails.
Parameter : m_EasyPod, the reference of struct MW_EasyPOD.
Index, device index of EasyPOD, Index start at 1.

4.2. Disconnect

Function : Public Declare Function DisconnectPOD Lib "EasyPOD.DLL" (ByRef m_EasyPod As MW_EasyPOD) As Long
Return : ERROR_SUCCESS(0) means disconnection is successful otherwise disconnection fails.
Parameter : m_EasyPod, the reference of struct MW_EasyPOD.

4.3. WriteData

Function : Public Declare Function WriteData Lib "EasyPOD.DLL" (ByRef m_EasyPod As MW_EasyPOD, lpString As Byte, ByVal IToWrite As Long, IWritten As Long) As Long
Return : ERROR_SUCCESS(0) means writing is successful otherwise writing fails.
Parameter : m_EasyPod, the reference of struct MW_EasyPOD.
lpString, pointer to the buffer containing the data to write to the device.
IToWrite, number of Bytes to write.
IWritten, actual number of bytes written.

4.4. ReadData

Function : Public Declare Function ReadData Lib "EasyPOD.DLL" (ByRef m_EasyPod As MW_EasyPOD, lpString As Byte, ByVal IToRead As Long, IRead As Long) As Long
Return : ERROR_SUCCESS(0) means reading is successful otherwise reading fails.
Parameter : m_EasyPod, the reference of struct MW_EasyPOD.
lpString, pointer to the buffer containing the data to read from the device.
IToRead, number of bytes to read.
IRead, actual number of bytes read.

4.5. ClearPODBuffer

Function : Public Declare Function ClearPODBuffer Lib "EasyPOD.DLL" (ByRef m_EasyPod As MW_EasyPOD) As Long
Return : ERROR_SUCCESS(0) means buffer clearing is successful otherwise the clearing fails.
Parameter : m_EasyPod, the reference of struct MW_EasyPOD.

5. Example

```
Public Type MW_EasyPod
    VID As Long
    PID As Long
    ReadTimeOut As Long
    WriteTimeOut As Long
    Handle As Long
    FeatureReportSize As Long
    InputReportSize As Long
    OutputReportSize As Long
End Type

Public Declare Function ConnectPodA Lib "EasyPod.DLL" Alias "ConnectPod" (ByRef m_EasyPod As MW_EasyPod, ByVal Index As Long) As Long

Public Declare Function DisconnectPodA Lib "EasyPod.DLL" Alias "DisconnectPod" (ByRef m_EasyPod As MW_EasyPod) As Long

Public Declare Function WriteDataA Lib "EasyPod.DLL" Alias "WriteData" (ByRef m_EasyPod As MW_EasyPod, lpString As Byte, ByVal IToWrite As Long, IWritten As Long) As Long

Public Declare Function ReadDataA Lib "EasyPod.DLL" Alias "ReadData" (ByRef m_EasyPod As MW_EasyPod, lpString As Byte, ByVal IToRead As Long, IRead As Long) As Long

Dim m_Pod As MW_EasyPod

Private Sub Command1_Click()
    Dim IResult As Long
    Dim IReturn As Long
    Dim sOutput() As Byte
    Dim sInput() As Byte
    Dim i As Integer
    ReDim sOutput(20)
    ReDim sInput(20)

    For i = 0 To 19
        sOutput(i) = i
    Next i
    m_Pod.VID = &HE6A
    m_Pod.PID = &H317

    IResult = ConnectPod(m_Pod, 1)

    IResult = ClearPODBuffer(m_Pod)

    If (IResult = 0) Then
        m_Pod.ReadTimeOut = 1000
        m_Pod.WriteTimeOut = 1000
        IResult = WriteData(m_Pod, sOutput(0), CLng(20), IReturn)
        IResult = ReadData(m_Pod, sInput(0), 20, IReturn)
    End If
```

IReturn = DisconnectPod(m_Pod)

End Sub

6. Revision History

Revision	Description	Date
V1.00	Release version	2008/01/30
V1.10	Balance I/O in Windows 2000	2009/01/09