Catalogue

Introduction to API

❖ USBIO_GetDllVersion

Users can use this interface to get DLL version.

✓ Note

The current version is V01.00

✓ Syntax

```
BOOL USBIO_GetDllVersion(PCHAR strVersion);
```

✓ Parameters

Name	Direction	Description
strVersion	output	V01.00

✓ Return Value

Successful, return TRUE. Unsuccessful, return FALSE

✓ Example

```
CHAR chVer[32] = {0};
if (USBIO_GetDllVersion(chVer)) {
printf("\r\nLoad USBIO DLL Version: %s",chVer);
}
else printf("\r\nLoad USBIO DLL Version error");
```

❖ USBIO_GetDeviceCount

Users can use this interface to get DMX device count in system.

✓ Syntax

DWORD USBIO_GetDeviceCount(PCHAR pVID_PID);

✓ Note

The current PID&VID is vid_0483&pid_57fe

✓ Parameters

Name	Direction	Description
pVID_PID	input	vid_0483&pid_57fe

✓ Return Value

It will be considered successful if non-zero values are the current DMX device count.

✓ Example

DWORD DeviceCount = USBIO_GetDeviceCount(USBIO_VID_PID);

❖ USBIO_OpenDevice

Users can use this interface to open DMX device.

Close the port by calling USBIO_CloseDevice when operation is completed

✓ Note

One DMX device is opened at one time

✓ Syntax

HANDLE USBIO_OpenDevice(DWORD dwDeviceNum, PCHAR pVID_PID);

✓ Parameters

Name	Direction	Description
dwDeviceNum	input	Its control No of DMX devices in system, its 0 begin
pVID_PID	input	vid_0483&pid_57fe, its same with USBIO_GetDeviceCount Parameter

✓ Return Value

Successful, return effective HANDLE. Unsuccessful, return INVALID_HANLDLE_VALUE

✓ Example

```
#define USBIO_VID_PID "vid_0483&pid_57fe"
…
HANDLE m_DevHandle= USBIO_OpenDevice(0, USBIO_VID_PID);
if (m_DevHandle != INVALID_HANDLE_VALUE)
{
printf("\r\nOpen Device OK");
}
else
{
printf("\r\nOpen Device Error");
}
```

❖ USBIO_SendDmx

Users can use this interface to send data to DMX device which was opened.

✓ Syntax

USHORT USBIO_SendDmx(HANDLE hDrive, PUCHAR txBuffer, USHORT txLen);

✓ Parameters

Name	Direction	Description
hDrive	Input	Handle of the device which was opened.
txBuffer	Input	Start address of send buffer
txLen	input	Number of frames to be sent to drivers

✓ Return Value

It will be considered successful if non-zero values are real number of frames sent to driver.

✓ Example

```
USHORT sendLen = USBIO_SendDmx(m_DevHandle, sendBuf, 512);
printf("\r\nwrite USBIO_SendDmx success num=%d", sendLen);
```

❖ USBIO_RecvDmx

Users can use this interface to read data from DMX device which was opened.

✓ Syntax

```
DWORD USBIO_RecvDmx(HANDLE hDrive, unsigned char * pBuffer);
```

✓ Parameters

Name	Direction	Description
hDrive	Input	Handle of the device which was opened.
pBuffer	Output	Start address of receive buffer

✓ Return Value

It will be considered successful if non-zero values are Real number of frames received from driver.

✓ Example

```
DWORD len = 0;
UCHAR rxData[512]={0};
len = USBIO_RecvDmx(m_DevHandle, rxData);
if(len == 0)
{
printf("\r\nI2C Read Error....");
}else {
printf("\r\nread USBIO_RecvDmx: %d", len);
}
```

❖ USBIO_CloseDevice

Close the port by calling this function when operation is completed.

✓ Syntax

```
BOOL USBIO_CloseDevice(HANDLE hDriver);
```

✓ Parameters

Name	Direction	Description
hDrive	Input	Handle of the device which was opened.

✓ Return Value

Successful: return TRUE. Unsuccessful: return FALSE. Please call GetLastError function

✓ Example

```
if (!USBIO_CloseDevice(m_DevHandle))
{
printf("\r\n CloseDevice Error....");
}else {
printf("\r\n CloseDevice OK");
}
```

❖ USBIO_GetDeviceType

Users can use this interface to type from DMX device which was opened.

✓ Syntax

USHORT USBIO_GetDeviceType(HANDLE hDrive);

✓ Parameters

Name	Direction	Description
hDrive	Input	Handle of the device which was opened.

✓ Return Value

It will be considered successful if non-zero values are returned in the following situations:

X:1 to 16, its MAX Number is X*512 to be sent to drivers at one time.

✓ Example

USHORT USBType = USBIO_GetDeviceType(m_DevHandle);

❖ Guide for Visual C++ development

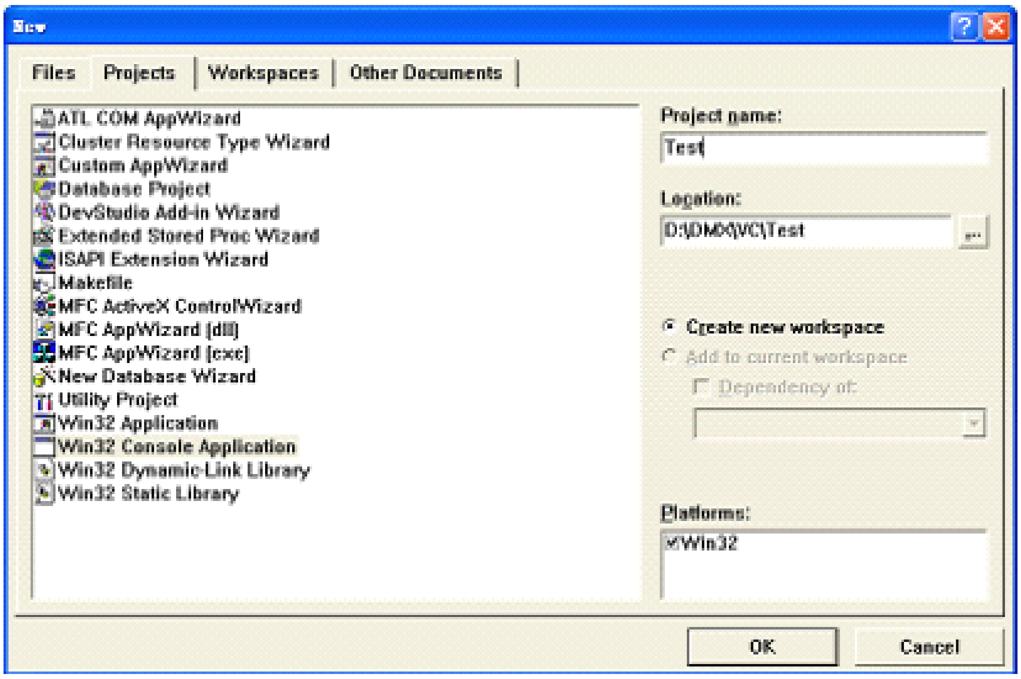
✓ Create a new VC project

Related header files must be used before using DMX Windows WDM Driver interface function. Make sure the driver had been installed correctly.

(Please refer to relevant books and documentations regarding detailed information about VC development.)

Please follow the following procedures to create a new VC++ project:

1. Select "File/New" from the main menu to create a new application project and source file. Define the type pf the new project as "Win32 Console Application", define the platform as "Win32" and select a path for files of the project.



Project name: Test

Location: D:\DMX\VC\Test

[✓ Create new workspace]

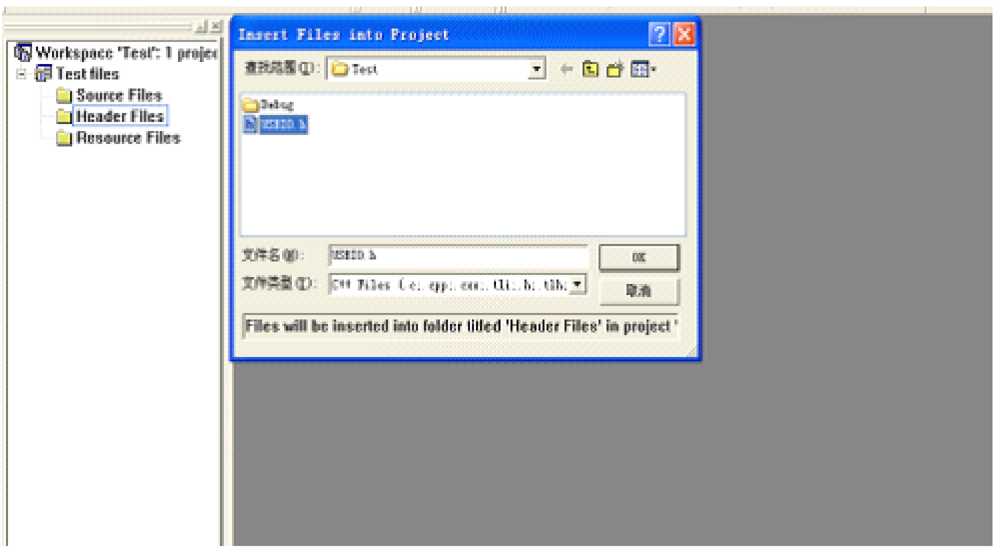
Platforms: [✓ Win32]

2. Click "OK"->"Finish"->"OK" according to the instructions on the screen.

A new VC project is created.

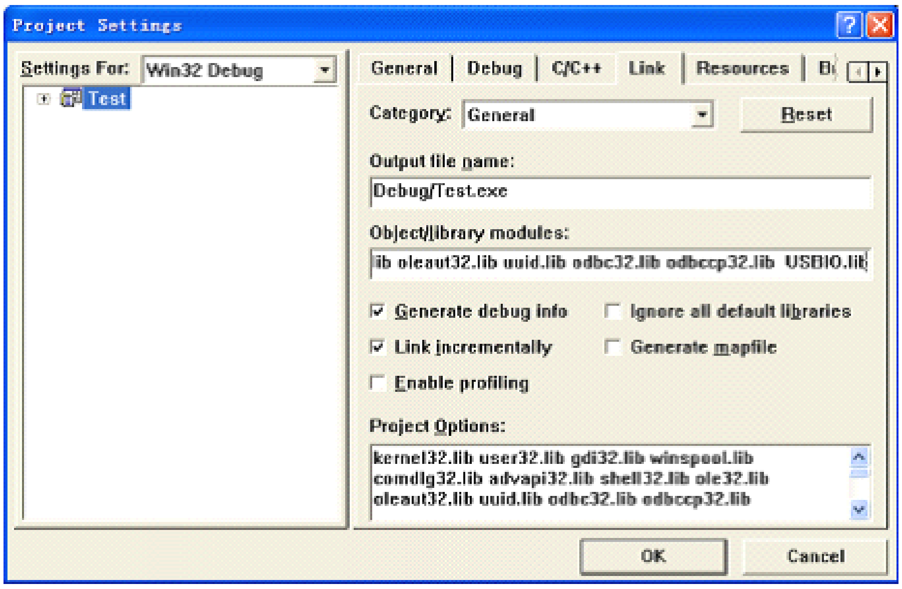
✓ Add necessary files

1、Add Include header files (USBIO.h) in DMX Windows WDM Driver. In VC++ work area, right click "Header Files", then select "Add Files to Folder" to add header files to the project.


Workspace 'Test': 1 projec
Test files
Source Files
Header Files
Resource Files

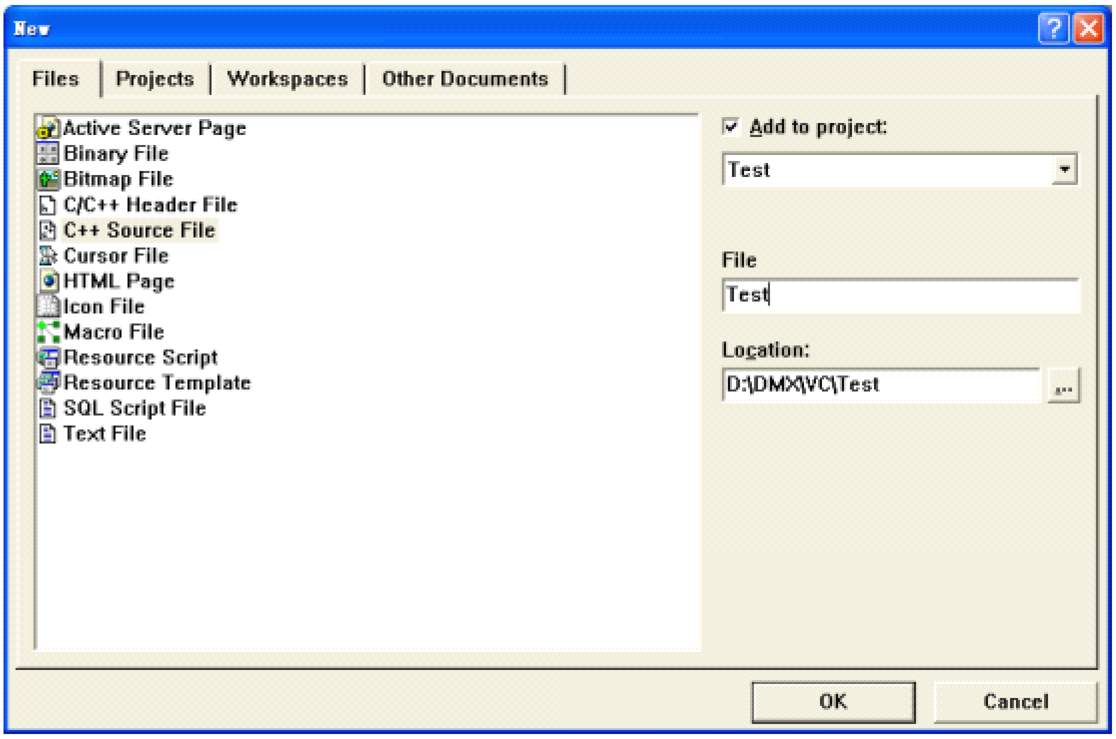
Insert Files into Project
查找范围(I): Test
Debug
USBIO.h
文件名(N): USBIO.h
文件类型(T): C++ Files (.c;.cpp;.cxx;.cc;.h;.hh;.[illegible]
Files will be inserted into folder titled 'Header Files' in project '

2、Add Lib files (USBIO.lib) in DMX Windows WDM Driver. In VC++ work area, right click "Project", then select "Settings" to add Lib files to the project.


Project Settings
Settings For: Win32 Debug
Test
General | Debug | C/C++ | Link | Resources | Bi
Category: General
Output file name: Debug/Test.exe
Object/library modules: lib oleaut32.lib uuid.lib odbc32.lib odbccp32.lib USBIO.lib
☑ Generate debug info ☐ Ignore all default libraries
☑ Link incrementally ☐ Generate mapfile
☐ Enable profiling
Project Options:
kernel32.lib user32.lib gdi32.lib winspool.lib
comdlg32.lib advapi32.lib shell32.lib ole32.lib
oleaut32.lib uuid.lib odbc32.lib odbccp32.lib

✓ **Write code**

Select "Add to Project->New" from "Project"， then select "C++ Source File".



[✓] Add to project: Test

File: Test

Location: D:\DMX\VC\Test

Write code in empty source file.

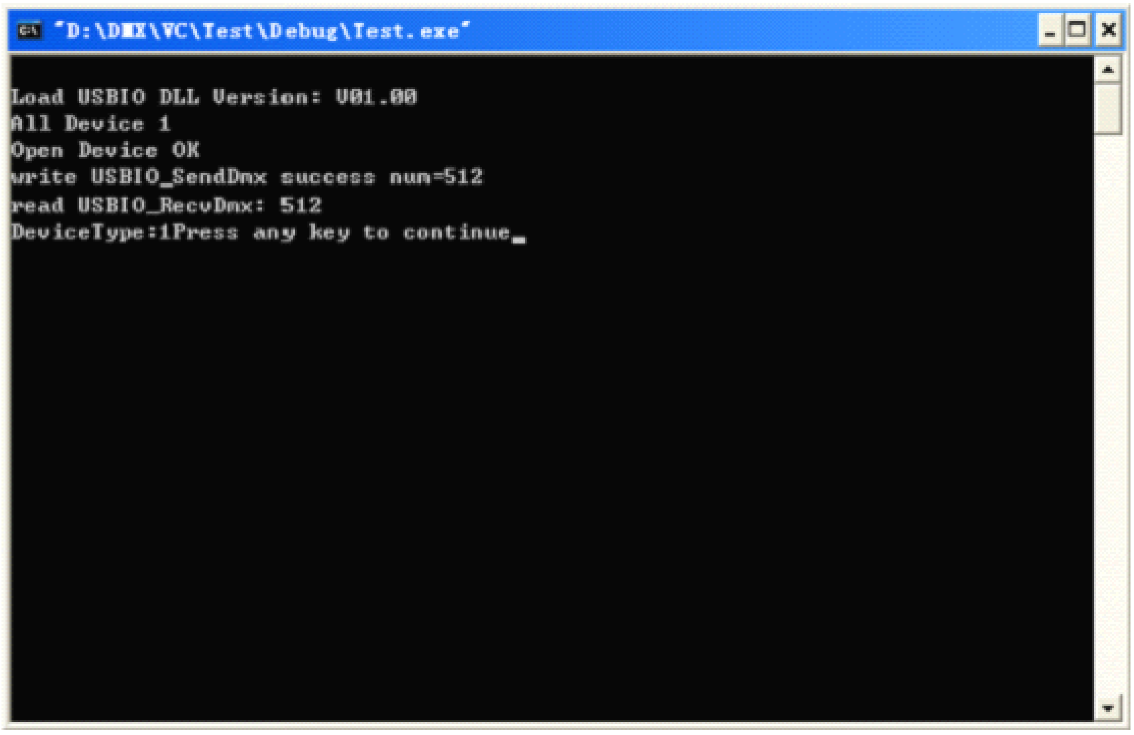
```
#include <stdio.h>
#include <windows.h>
#include "USBIO.h"

#define USBIO_VID_PID "vid_0483&pid_57fe"

void main()
{
HANDLE hDevice = NULL;

////////////////////////////////////////////////////////////////////////
//USBIO_GetDllVersion
CHAR chVer[32] = {0};
if (USBIO_GetDllVersion(chVer)) {
        printf("\r\nLoad USBIO DLL Version: %s",chVer);
}
else printf("\r\nLoad USBIO DLL Version error");
//////////////////////////////////////////////////////////////////////

//////////////////////////////////////////////////////////////////////
//USBIO_GetDeviceCount
DWORD DeviceCount = USBIO_GetDeviceCount(USBIO_VID_PID);
printf("\r\nAll Device %d", DeviceCount);
//////////////////////////////////////////////////////////////////////

//////////////////////////////////////////////////////////////////////
//USBIO_OpenDevice
hDevice= USBIO_OpenDevice(0, USBIO_VID_PID);
if (hDevice != INVALID_HANDLE_VALUE)
{
    printf("\r\nOpen Device OK");
}
else
{
    printf("\r\nOpen Device Error");
}
//////////////////////////////////////////////////////////////////////

//////////////////////////////////////////////////////////////////////
//USBIO_SendDmx
BYTE sendBuf[512]={0};
for (USHORT i=0; i<512; i++) {
    sendBuf[i] = i;
}
```

```
USHORT sendLen = USBIO_SendDmx(hDevice, sendBuf, 512);
printf("\r\nwrite USBIO_SendDmx success num=%d", sendLen);
//////////////////////////////////////////////////////////////////////////

//////////////////////////////////////////////////////////////////////////
//USBIO_RecvDmx
DWORD len = 0;
UCHAR rxData[512]={0};
len = USBIO_RecvDmx(hDevice, rxData);
if(len == 0)
{
    printf("I2C Read Error....");
}else {
    printf("\r\nread USBIO_RecvDmx: %d", len);
}
//////////////////////////////////////////////////////////////////////////

//////////////////////////////////////////////////////////////////////////
//USBIO_GetDeviceType
USHORT USBType = USBIO_GetDeviceType(hDevice);
printf("\r\nDeviceType:%d", USBType);
//////////////////////////////////////////////////////////////////////////

//////////////////////////////////////////////////////////////////////////
//USBIO_CloseDevice
if (hDevice) {
    USBIO_CloseDevice( hDevice);
    CloseHandle(hDevice);
}
```

```
////////////////////////////////////////////////////////////////////
}
```

✓ Test application

Run the application, the following result will be displayed.



```
"D:\DMX\VC\Test\Debug\Test.exe"

Load USBIO DLL Version: V01.00
All Device 1
Open Device OK
write USBIO_SendDmx success num=512
read USBIO_RecvDmx: 512
DeviceType:1Press any key to continue_
```