

DEWESoft C++ DLL to Trenz Electronic C++ DLL

Introduction



How to write C++ programs using the new DLL starting from the old DLL.






There are some major differences between the two DLLs.



feature	Dewesoft C++ DLL	Trenz Electronic C++ DLL
programming language	C++	C++
architecture	standard (TE0300DLL.dll)	stacked (TE_USB_FX2_CyAPI.dll requires Cypress CyAPI.dll);
Handles	present	absent
structures	embedded	defined in Cypress CyAPI.h
parameters*	less	more
freedom*	less	more
buffer size	2 Kbyte (fixed)	4 Kbyte or more (it can be changed)

Feature of Dewesoft C++ DLL and Trenz Electronic C++ DLL

Function translation

Dewesoft C++ DLL	Trenz Electronic C++ DLL
<code>HANDLE m_handle = 0;</code>	<code>CCyUSBDevice *USBdevList = new CCyUSBDevice((HANDLE)0,CYUSBDRV_GUID,true);</code> <div><div> The handles are internally managed by CyAPI.lib and there is no need to expose them to the user.</div><div> <code>CyUSBDevice</code> <code>TE_USB_FX2_USBDevice</code> doesn't explicitly exist in C++, unlike the C# case; this is a shortcoming of CyAPI.lib. <code>d=0; USBdevList->Open(d); // Open automatically calls Close() if necessary</code> <code>vID = USBdevList->VendorID; // not (USBdevList->TE_USB_FX2_USBDevice).VendorID as logically expected</code> <code>pID = USBdevList->ProductID; // not (USBdevList->TE_USB_FX2_USBDevice).ProductID as logically expected</code></div></div>
<code>cout << endl << TE0300_ScanCards() << endl;</code>	<code>cout << endl << TE_USB_FX2_ScanCards(USBdevList) << endl;</code>

TE0300_Open (&m_handle, 0) !=0;	TE_USB_FX2_Open(USBdevList, 0)!=0; <div data-bbox="386 352 1479 457">  TE_USB_FX2_Open() In the code, it is possible to call TE_USB_FX2_Open() where TE0300_Open() is used. </div> <div data-bbox="386 485 1479 726">  TE_USB_FX2_Open() act as Close() for other TE USB handles already open When a new handle to the device driver is open (TE_USB_FX2_Open() run successfully) other internal handles (inside USBDeviceList) are automatically closed by TE_USB_FX2_Open() function. No more than one handle can be active on the same time. It is a behavior inherited by CyAPI.dll Open() function. For this reason, TE_USB_FX2_Close() function is almost useless. d=0; USBdevList->Open(d); // Open automatically calls Close() if necessary (from the Cypress CyAPI documentation) </div> <div data-bbox="386 751 1479 877">  TE_USB_FX2_Open() as SelectCard() TE_USB_FX2_Open(USBdevList, x) acts more as a SelectCard() function because the list of USB devices is already created in USBdevList. </div>
TE0300_Open (&m_handle, 1)!=0;	TE_USB_FX2_Open(USBdevList, 1)!=0
TE0300_Close (&m_handle);	TE_USB_FX2_Close(USBdevList); <div data-bbox="386 1035 1479 1392">  This function closes all internal handles of USBDeviceList. This function does NOT differ much from its homonym of the previous TE0300DLL.dll; the only difference is that this function closes a handle (like TE0300DLL.dll) to the driver but the handle is not exposed to user because it is not exposed by USBDeviceList (unlike TE0300DLL.dll). In the code, it is possible to call TE_USB_FX2_Close() where TE0300_Close() is used, but <ul style="list-style-type: none"> • it is rare that you would ever need to call TE_USB_FX2_Close() explicitly (though doing so would not cause any problems). • TE_USB_FX2_Open() realize automatically much of the TE_USB_FX2_Close() work. Close is automatically carried out by the TE_USB_FX2_Open() function, if another handle to the same device driver is already open (i.e. a TE_USB_FX2_Open() has been successfully used before). </div> <div data-bbox="386 1419 1479 1570">  Warning about derived variables If TE_USB_FX2_Close() is called, then dynamically allocated members of the CCyUSBDevice class are de-allocated. And, all "shortcut" pointers to elements of the EndPoints array (ControlEndPt, IsoIn/OutEndPt, BulkIn/OutEndPt, InterruptIn/OutEndPt) are reset to NULL. </div>
TE0300_SendCommand (handle, cmd, cmd_length, reply, &reply_length, timeout)	TE_USB_FX2_SendCommand(USBdevList, cmd, cmd_length, reply, reply_length, timeout)

Equivalent code doesn't exist	TE_USB_FX2_SetData_InstanceDriverBuffer (USBdevList, &BulkOutEP, PI_EP8, timeout, DeviceDriverBufferSize); <div>  <p>Example: in TE0300DLL.dll, the SET buffer size is fixed to 2 Kbyte, while in TE_USB_FX2_CyAPI.dll you are free to choose 4 Kbyte or more.</p> <p>BufferSize has a strong influence on DataThroughput. If BufferSize is too small, the data throughput can be 1/3 to 1/2 of the maximum value (33-36 Mbyte/s for read transactions).</p> <p>You should instance the driver buffer only one time and not for every transmission, otherwise you could half your data throughput.</p> </div>
Equivalent code doesn't exist	TE_USB_FX2_GetData_InstanceDriverBuffer (USBdevList, &BulkInEP, PI_EP6, timeout, DeviceDriverBufferSize); <div>  <p>Example: in TE0300DLL.dll, the GET buffer size is fixed to 2 Kbyte, while in TE_USB_FX2_CyAPI.dll you are free to choose 4 Kbyte or more.</p> <p>BufferSize has a strong influence on DataThroughput. If BufferSize is too small, the DataThroughput can be 1/3 to 1/2 of the maximum value (25-28 Mbyte/s for write transactions).</p> <p>You should instance the driver buffer only one time and not for every transmission, otherwise you could half your data throughput.</p> </div>

Function Translation from DEWESoft C++ to Trenz Electronic C++

Examples

Dewesoft C++ DLL	Trenz Electronic C++ DLL
<pre>//test code, not production code int packetlen = 512; byte data[512];</pre>	<pre>//test code, not production code int packetlen = 512; byte data[512];</pre>
	<pre>CCyBulkEndPoint *BulkOutEP = NULL; TE_USB_FX2_SetData_InstanceDriverBuffer (USBdevList, &BulkOutEP, PI_EP8, timeout, DeviceDriverBufferSize);</pre>
<pre>for (int i = 0; i < 10; i++) { packetlen = 512; for (int j = 0; j < packetlen; j++) data[j] = j; if (TE0300_SetData(handle, data, packetlen, PI_EP8)) { cout << "ERROR" << endl; return; } }</pre>	<pre>for (int i = 0; i < 10; i++) { packetlen = 512; for (int j = 0; j < packetlen; j++) data[j] = j; if (TE_USB_FX2_SetData(&BulkOutEP, data, packetlen)) { cout << "ERROR" << endl; return; } }</pre>

Simplified Example 1

Dewesoft C++ DLL	Trenz Electronic C++ DLL
<pre>int packetlen = 512; byte data[512];</pre>	<pre>int packetlen = 512; byte data[512];</pre>

	<pre>CCyBulkEndPoint *BulkInEP = NULL; TE_USB_FX2_GetData_InstanceDriverBuffer (USBdevList, &BulkInEP, PI_EP6, timeout, DeviceDriverBufferSize);</pre>
<pre>for (int i = 0; i < 10; i++) { packetlen = 512; if (TE0300_GetData(handle, data, &packetlen, PI_EP6, 1000)) { cout << "ERROR" << endl; return; } for (int j = 0; j < packetlen; j++) cout << data[j]; cout << endl; }</pre>	<pre>for (int i = 0; i < 10; i++) { packetlen = 512; if (TE_USB_FX2_GetData(&BulkInEP, data, packetlen)) { cout << "ERROR" << endl; return; } for (int j = 0; j < packetlen; j++) cout << data[j]; cout << endl; }</pre>

Simplified Example 2

Dewesoft C++ DLL	Trenz Electronic C++ DLL
<pre>void ReadData(unsigned int handle) {</pre>	<pre>void ReadData (CCyUSBDevice *USBdevList, unsigned int DeviceDriverBufferSize, int RX_PACKET_LEN, unsigned long TIMEOUT) {</pre>
<pre>int packetlen = RX_PACKET_LEN; unsigned int packets = 1200; //allocate memory byte * data; data = new byte [RX_PACKET_LEN*packets];</pre>	<pre>long packetlen = RX_PACKET_LEN; unsigned int packets = 1200; //allocate memory byte * data; byte * data_temp = NULL; data = new byte [RX_PACKET_LEN*packets];</pre>
<pre>unsigned int total_cnt = 0; unsigned int errors = 0;</pre>	<pre>unsigned int total_cnt = 0; unsigned int errors = 0;</pre>
	<pre>//Instantiate driver buffer CCyBulkEndPoint *BulkInEP = NULL; TE_USB_FX2_GetData_InstanceDriverBuffer (USBdevList, CardNo, &BulkInEP, PI_EP6, TIMEOUT, DeviceDriverBufferSize);</pre>
<pre>//starts test ResetFX2FifoStatus(handle); SendFPGAcommand(handle, FX22MB_REG0_START_TX);</pre>	<pre>//starts test ResetFX2FifoStatus(USBdevList); SendFPGAcommand(USBdevList, FX22MB_REG0_START_TX);</pre>
<pre>//StopWatch starts ElapsedTime.Start(); for (unsigned int i = 0; i < packets; i++) { packetlen = RX_PACKET_LEN;</pre>	<pre>//StopWatch starts ElapsedTime.Start(); for (unsigned int i = 0; i < packets; i++) { packetlen = RX_PACKET_LEN;</pre>
	<pre>data_temp = &data[total_cnt];</pre>

<pre> if (TE0300_GetData(handle, data+total_cnt, &packetlen, PI_EP6,TIMEOUT_MS)) { cout << "ERROR read" << endl; errors++; break; } total_cnt += packetlen; } TheElapsedTime = ElapsedTime.Stop(false); //DEBUG StopWatch timer SendFPGAcommand(handle, FX22MB_REG0_STOP); //stops test delete data; } </pre>	<pre> if (TE_USB_FX2_GetData(&BulkInEP,data_temp, packetlen)) { cout << "ERROR read" << endl; errors++; break; } total_cnt += (packetlen); } TheElapsedTime = ElapsedTime.Stop(false); //DEBUG StopWatch timer SendFPGAcommand(USBDevice, FX22MB_REG0_STOP); //stops test delete data; } </pre>
---	---

Read Data Test Example

Dewesoft C++ DLL	Trenz Electronic C++ DLL
<pre> void WriteData(unsigned int handle) { </pre>	<pre> void WriteData(CCyUSBDevice *USBdevList, unsigned int DeviceDriverBufferSize, int TX_PACKET_LEN, unsigned long TIMEOUT) { </pre>
<pre> int packetlen = TX_PACKET_LEN; unsigned int packets = 1200; //allocate memory byte * data; data = new byte [TX_PACKET_LEN*packets]; </pre>	<pre> long packetlen = TX_PACKET_LEN; unsigned int packets = 1200; //allocate memory byte * data; byte * data_temp = NULL; data = new byte [TX_PACKET_LEN*packets]; </pre>
<pre> unsigned int total_cnt = 0; unsigned int errors = 0; </pre>	<pre> unsigned int total_cnt = 0; unsigned int errors = 0; </pre>
	<pre> CCyBulkEndPoint *BulkOutEP = NULL; TE_USB_FX2_SetData_InstanceDriverBuffer (USBdevList, CardNo, &BulkOutEP, PI_EP8,TIMEOUT, DeviceDriverBufferSize); </pre>
<pre> SetData (data); ResetFX2FifoStatus(handle); //starts test SendFPGAcommand(handle, FX22MB_REG0_START_RX); </pre>	<pre> SetData (data); ResetFX2FifoStatus(USBdevList); //starts test SendFPGAcommand(USBdevList, FX22MB_REG0_START_RX); </pre>
<pre> //StopWatch starts ElapsedTime.Start(); for (unsigned int i = 0; i < packets; i++) { packetlen = RX_PACKET_LEN; </pre>	<pre> //StopWatch starts ElapsedTime.Start(); for (unsigned int i = 0; i < packets; i++) { packetlen = RX_PACKET_LEN; </pre>
	<pre> data_temp = &data[total_cnt]; </pre>

<pre> if (TE0300_SetData(handle, data+total_cnt, &packetlen, PI_EP8, TIMEOUT_MS)) { cout << "ERROR write" << endl; errors++; break; } total_cnt += packetlen; } TheElapsedTime = ElapsedTime.Stop(false); //DEBUG StopWatch timer SendFPGAcommand(handle, FX22MB_REG0_STOP); //stops test delete data; } </pre>	<pre> if (TE_USB_FX2_SetData(&BulkOutEP, data_temp, packetlen)) { cout << "ERROR write" << endl; errors++; break; } total_cnt += (packetlen); } TheElapsedTime = ElapsedTime.Stop(false); //DEBUG StopWatch timer SendFPGAcommand(USBDevice, FX22MB_REG0_STOP); //stops test delete data; } </pre>
---	--

Write Data Test Example