Dewesoft C++ DLL
to
Trenz Electronic C# DLL
Porting Guide

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

1 Introduction

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_CyUSB.dll requires Cypress CyUSB.dll);
Handles	present	absent
structures	embedded	defined in Cypress CyAPI.h
parameters*	less	more
freedom*	less	more

* Example: in TE0300DLL.dll, the buffer size is fixed to 2 kbyte, while in TE_USB_FX2_CyAPI.dll you are free to choose 4 kbyte or more.

2 Function translation

Dewesoft C++ DLL	Trenz Electronic C# DLL
`HANDLE m_handle = 0;`	`CyUSBDevice TE_USB_FX2_USBDevice = null;` `USBDeviceList USBdevListToDispose = new USBDeviceList(CyConst.DEVICES_CYUSB);`
`cout << endl << TE0300_ScanCards() << endl;`	`int NumberOfCardAttached = TE_USB_FX2.TE_USB_FX2.TE_USB_FX2_ScanCards(ref USBdevList);` `Console.WriteLine(" {0} ", NumberOfCardAttached);`
`TE0300_Open(&m_handle, 0)!=0`	`TE_USB_FX2.TE_USB_FX2.TE_USB_FX2_Open (ref TE_USB_FX2_USBDevice, ref USBdevList, 0) == false`
`TE0300_Open(&m_handle, 1)!=0`	`TE_USB_FX2.TE_USB_FX2.TE_USB_FX2_Open (ref TE_USB_FX2_USBDevice, ref USBdevList, 1) == false`
`TE0300_Close(&m_handle);`	`TE_USB_FX2.TE_USB_FX2.TE_USB_FX2_Close (ref USBdevListToDispose);`
`TE0300_SendCommand(handle, cmd, cmd_length, reply, &reply_length, timeout)`	`TE_USB_FX2.TE_USB_FX2.TE_USB_FX2_SendCommand (ref TE_USB_FX2_USBDevice, ref cmd, ref cmd_length, ref reply, ref reply_length, TIMEOUT_MS)`

Dewesoft C++ DLL	Trenz Electronic C# DLL

void ResetFX2FifoStatus(HANDLE handle)
{
 	
  cout << endl << "Resetting all FIFOs" << endl;
  byte cmd[64], reply[64];
  int cmd_length = 64;
  int reply_length = 64;
 	
  cmd[0] = 0xA4;//command RESET_FIFO_STATUS
  cmd[1] = 0;//RESET all FIFOs
 	
  if (TE0300_SendCommand(handle, cmd, cmd_length, reply, &reply_length, 1000))
  cout << "Error" << endl;
 	
  cmd[0] = 0xA0;//command INITIALIZE
  cmd[1] = 1;//FIFO mode
 	
  if (TE0300_SendCommand(handle, cmd, cmd_length, reply, &reply_length, 1000))
  cout << "Error" << endl;
}
	static void ResetFX2FifoStatus(CyUSBDevice TE_USB_FX2_USBDevice)
	{
	
	  if (TE_USB_FX2_USBDevice == null)
	  {
	  Console.WriteLine("Error,no device is selected");
	  return;
	  }
	  Console.WriteLine("Resetting all FIFOs");
	  byte[] cmd = new byte[64];
	  byte[] reply = new byte[64];
	  int cmd_length = 64;
	  int reply_length = 64;
	
	  uint TIMEOUT_MS = 100000;
	
	  cmd[0] = (byte)FX2_Commands.RESET_FIFO_STATUS;
	  cmd[1] = 0; //reset all fifos
	
	  if (TE_USB_FX2.TE_USB_FX2.TE_USB_FX2_SendCommand(ref TE_USB_FX2_USBDevice, ref cmd, ref cmd_length, ref reply, ref reply_length, TIMEOUT_MS) == false)
	  Console.WriteLine("Error Send Command Reset all fifos");
	
	  cmd[0] = (byte)FX2_Commands.INITALIZE;
	  //0xA0;//command INITIALIZE
	  cmd[1] = 1;//FIFO mode
	
	  if (TE_USB_FX2.TE_USB_FX2.TE_USB_FX2_SendCommand(ref TE_USB_FX2_USBDevice, ref cmd, ref cmd_length, ref reply, ref reply_length, TIMEOUT_MS) == false)
	  Console.WriteLine("Error Switch Mode Fifo Mode");
	}

Dewesoft C++ DLL	Trenz Electronic C# DLL
void ReadData(unsigned int handle) {	static void ReadDataFPGAIntegrity(CyUSBDevice TE_USB_FX2_USBDevice, int BUFFER_SIZE, uint TIMEOUT_MS) {

	if (TE_USB_FX2_USBDevice == null)
	{
	Console.WriteLine("Error,no device is selected");
	return;
	}

int packetlen = RX_PACKET_LEN;
unsigned int packets = 1200;
byte * data;
 	
unsigned int total_cnt = 0;
unsigned int errors = 0;
data = new byte [RX_PACKET_LEN*packets];
//allocate memory
	int packetlen = RX_PACKET_LEN;
	int packets = 1200;
	byte[] data = new byte[packetlen*packets];
	byte[] buffer = new byte[packetlen];
	int total_cnt = 0;
	int errors = 0;
	int PI_EP6 = 6;
	
	bool bResultXfer = false;
	ResetFX2FifoStatus(TE_USB_FX2_USBDevice);
	SendFPGAcommand(ref TE_USB_FX2_USBDevice,
	MB_Commands.FX22MB_REG0_START_TX, TIMEOUT_MS);
	//starts test
	test_cnt = 0;
	total_cnt = 0;
	Stopwatch stopWatch = new Stopwatch();
	stopWatch.Start();

ResetFX2FifoStatus(handle); SendFPGAcommand(handle,FX22MB_REG0_START_TX); //starts test	
ElapsedTime.Start(); //StopWatch start	

for (unsigned int i = 0; i < packets; i++)
{
packetlen = RX_PACKET_LEN;
	for (int i = 0; i < packets; i++)
	{
	packetlen = RX_PACKET_LEN;
	//fixed (byte* buf = &data[total_cnt])
	//bResultXfer =
	TE_USB_FX2.TE_USB_FX2.TE_USB_FX2_GetDataP(ref
	inEndpointPipeNo, buf, ref packetlen, PI_EP6, TIMEOUT_MS);

if (TE0300_GetData(handle, data+total_cnt, &packetlen,
PI_EP6,TIMEOUT_MS))
{
	bResultXfer =
	TE_USB_FX2.TE_USB_FX2.TE_USB_FX2_GetData(ref
	TE_USB_FX2_USBDevice, ref buffer, ref packetlen, PI_EP6,
	TIMEOUT_MS,BUFFER_SIZE);
	Buffer.BlockCopy(buffer,0, data, total_cnt, packetlen);

```
      cout << "ERROR" << endl;
      errors++;
      break;
    }
    total_cnt += packetlen;
  }
  TheElapsedTime = ElapsedTime.Stop(false);
  //DEBUG StopWatch timer

  SendFPGAcommand(handle,FX22MB_REG0_STOP);
  //stops test
  delete data;
}
```

```
    if (bResultXfer == false)
    {
      Console.WriteLine("Error Get Data");
      errors++;
      break;
    }
    total_cnt += packetlen;
  }
  stopWatch.Stop();
  TimeSpan ts = stopWatch.Elapsed;

  SendFPGAcommand(ref TE_USB_FX2_USBDevice,
MB_Commands.FX22MB_REG0_STOP, TIMEOUT_MS);
  //stops test
}
```

Dewesoft C++ DLL	Trenz Electronic C# DLL

void WriteData(unsigned int handle)
{
 	
  int packetlen = TX_PACKET_LEN;
  unsigned int packets = 1200;
  byte * data;
 	
  unsigned int total_cnt = 0;
  unsigned int errors = 0;
  data = new byte [TX_PACKET_LEN*packets];
  //allocate memory
  //SetData (data);
  ResetFX2FifoStatus(handle);
  SendFPGAcommand(handle,FX22MB_REG0_START_RX);
  //starts test
 	
  ElapsedTime.Start();
  //StopWatch start
 	
  for (unsigned int i = 0; i < packets; i++)
  {
  packetlen = TX_PACKET_LEN;
  if (TE0300_GetData(handle, data+total_cnt, &packetlen, PI_EP8,TIMEOUT_MS))
  {
	static void WriteData(CyUSBDevice TE_USB_FX2_USBDevice, int BUFFER_SIZE, uint TIMEOUT_MS)
	{
	
	  if (TE_USB_FX2_USBDevice == null)
	  {
	  Console.WriteLine("Error,no device is selected");
	  return;
	  }
	
	  int packetlen = TX_PACKET_LEN;
	  int packets = 1200;
	  byte[] data = new byte[packetlen*packets];
	  byte[] buffer = new byte[packetlen];
	  int total_cnt = 0;
	  int errors = 0;
	  int PI_EP8 = 8;
	  //SetData (data);
	  bool bResultXfer = false;
	  ResetFX2FifoStatus(TE_USB_FX2_USBDevice);
	  SendFPGAcommand(ref TE_USB_FX2_USBDevice, MB_Commands.FX22MB_REG0_START_RX, TIMEOUT_MS);
	  //starts test
	  test_cnt = 0;
	  total_cnt = 0;
	  Stopwatch stopWatch = new Stopwatch();
	  stopWatch.Start();
	
	  for (int i = 0; i < packets; i++)
	  {
	  packetlen = TX_PACKET_LEN;
	  Buffer.BlockCopy(data,total_cnt, buffer, 0, packetlen);
	  bResultXfer = TE_USB_FX2.TE_USB_FX2.TE_USB_FX2_GetData(ref TE_USB_FX2_USBDevice, ref buffer, ref packetlen, PI_EP6, TIMEOUT_MS,BUFFER_SIZE);

```
      cout << "ERROR" << endl;
      errors++;
      break;
    }
    total_cnt += packetlen;
  }
  TheElapsedTime = ElapsedTime.Stop(false);
  //DEBUG StopWatch timer

  SendFPGAcommand(handle,FX22MB_REG0_STOP);

  //stops test
  delete data;
}
```

```
    if (bResultXfer == false)
    {
      Console.WriteLine("Error Get Data");
      errors++;
      break;
    }
    total_cnt += packetlen;
  }
  stopWatch.Stop();
  TimeSpan ts = stopWatch.Elapsed;

  SendFPGAcommand(ref TE_USB_FX2_USBDevice,
MB_Commands.FX22MB_REG0_STOP, TIMEOUT_MS);
  //stops test
}
```

3 Document Change History

version	date	author	description
0.9	2012-06-01	SP, FDR	Public preview.
1.0			Initial release.