TTLAPI Programming Examples

Microsoft Visual C++ 6

BatteryLogger

An MFC dialog based application. This simple example logs the encoder's battery level to a text file. Demonstrates usage of TTLLive control when dragged onto a form. It also uses the OnEncoderStateChange event.

TTLExplorer

A full featured WTL 7.1 based application (see sourceforge.net/projects/wtl/). It displays most of the encoder and channel properties, and allows live sample data from multiple channels to be displayed on simple line graphs. By default, it forces detection of FlexComp Infiniti encoders. To use this example with a ProComp encoder, unselect "Force Protocol" in the Options menu.

Tutorial

10_Simplest

This is an example of how to use the TTLLive control in its simplest, most concise way. It creates the object and then uses few basics properties. To keep it simple, the example uses smart pointers and wrapper class generated by the Visual Studio environment.

11_Simplest

Very similar to 10_Simplest, printing out some results and also querying for the ITTLLive2 interface.

20_CreatingInstance

This example takes a few steps back and demonstrates functions similar to 11_Simplest, without using smart pointers or the wrapper class. Shows a little bit more of the magic hidden in the Visual Studio-generated helper code.

Directly uses CoCreateInstance to create an object instance and get an ITTLLive interface pointer. CoCreateInstance does an AddRef() call for the client; it is then not necessary for the client to call pTTLLive2->AddRef(). However the client must still call pTTLLive2->Release().

The example also uses the object's QueryInterface to attempt to get an ITTLLive2 interface pointer. When using QueryInterface it is necessary to AddRef() the received pointer and consequently Release() it when finished.

30_Connecting_Encoder

This example demonstrates connecting to encoders and printing out their properties. The code takes advantage of the ITTLLive2 interface if it is available; otherwise it prints the subset of information obtainable through the ITTLLive interface.

In this example, the wrapper class CreateIntance method is used; therefore it is necessary to call Release to properly balance interface reference counting.

40_Reading_Data

This example demonstrates a simple application that connects to discovered encoder(s), creates channels, starts data acquisition and finally reads sample data from connected encoder(s).

41_Reading_Data

This example is identical to 40_Reading_Data except for the contents of the read_data() procedure. In this example the read_data() procedure uses the TTLLive ReadChannelDataVT method to demonstrate use of the VARIANT data type within a C++ client. The VARIANT type, while easy to use under scripting languages or Visual Basic, is more complex to deal with in Visual C++.

42_Reading_Data

From the 41_Reading_Data example, we now start a new project which includes all of the code from the previous example, but this time including MFC support. This example demonstrates simpler manipulation of the VARIANT type using built-in MFC classes such as COleSafeArray, at the cost of taking more memory and having more dependencies.

Microsoft Visual Basic for Applications

Microsoft Excel 2002

TTLLive Simple Capture Client.xls

This simple example conveniently captures some sample data in an Excel worksheet for later analysis.

Microsoft Visual Basic 6

SimpleClient

Simple client application which shows the TTLLive version, connects to an encoder and displays other information, counts data errors by receiving connection point events, and plots sample data obtained in a VARIANT array through ReadChannelDataVT.

MathWorks MatLab

This set of three Matlab scripts was tested with Matlab Release 14. They should be called from the Matlab command window in the order in which they are described below. ttltestNew is a "function", the other two are m-file scripts.

The mInitTTL.m script initializes the ActiveX object and returns a handle called TTLLive to the global workspace.

The ttltestNEW function takes that handle as input parameter, connects to the encoder and reads and plots streaming data from channels C and D. The output of the function is a data array (allData) containing the samples collected from both channels.

The mDestroyTLL.m script releases the handle to the ActiveX object.

National Instruments LabVIEW 7.1

Although we have some TTLLive sample code for the National Instruments LabView environment, it was not included in this distribution package. Please contact the Thought Technnology Engineering department to request sample code.