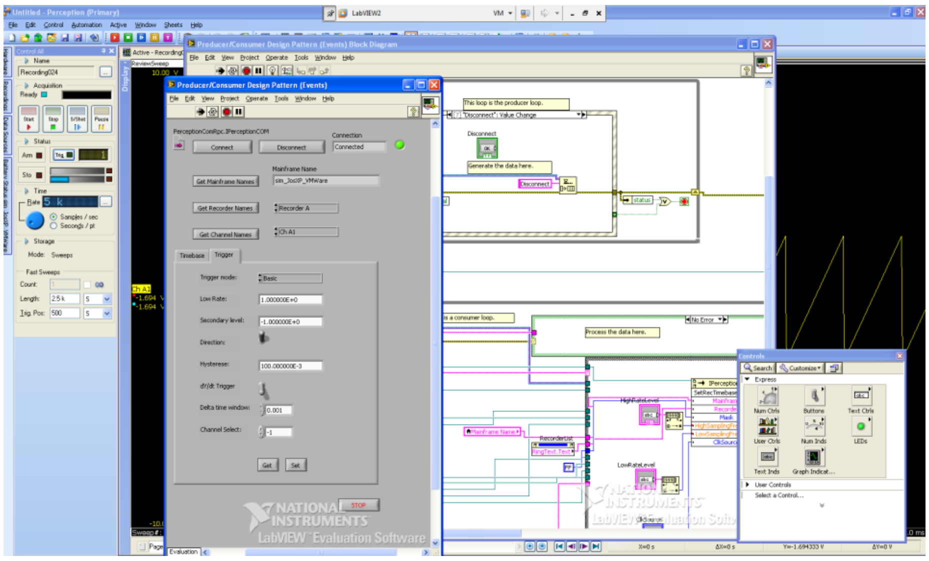
Genesis HIGH SPEED

User Manual

English



Perception remote control using LabVIEW

HBM

I3290-1.2_en

Document version 1.2 – October 2012

For Perception 6.30 or higher

For HBM's Terms and Conditions visit www.hbm.com/terms

HBM GmbH
Im Tiefen See 45
64293 Darmstadt
Germany
Tel: +49 6151 80 30
Fax: +49 6151 8039100
Email: info@hbm.com
www.hbm.com/highspeed

Table of Contents

1 Getting Started

Welcome to the Perception remote control manual using LabVIEW. This manual describes how you can use the COM/RPC interface of Perception from within LabVIEW. For more information on the COM/RPC interface we refer to the user manual called **“Programmers Reference Perception RPC interface”** (I2699_1.0en)

1.1 Introduction

The LabVIEW program of National Instruments can be used to control Perception. Perception has implemented an RPC interface, to simplify the usage of this interface HBM has designed a COM wrapper around the RPC client. This COM wrapper can be used as an ActiveX from within LabVIEW. For more information we refer to the appendix called **Using RPC-COM wrapper and C#** in the Perception RPC interface manual.

This document will describe how you can use this COM wrapper, it demonstrates how you can start from scratch and build your first simple LabVIEW application communicating with Perception.

1.2 Intended audience

This documentation assumes you have sufficient knowledge of LabVIEW, this manual is NOT a tutorial on how to use LabVIEW.
This documentation also assumes you understand your HBM equipment, software, and basic acquisition terminology.
Understanding acquisition terminology is vital to understanding digital recordings: trigger, sample rate, pre-/post trigger, etc.

1.3 Requirements and installation

The HBM RPC Interface is an option that is enabled through the use of the HASP®
4 USB Token. When this option is installed, a colored icon is shown on the splash screen at start-up.
When this icon is grayed you should contact your local dealer for more information on how to obtain this option.
The HBM Remote API is an option that is enabled through the use of the HASP®4 USB Token.

This option is also listed as **Remote API: control Perception using the SOAP interface or using RPC calls** in the Perception menu:
Help > About Perception > More... > Options page

We assume you have installed LabVIEW.
In addition you must install the required software modules as described below.

1.3.1 System requirements

- HBM Perception software with Remote API option enabled

1.3.2 Supported hardware

- HBM GEN Series Modular Data Acquisition System
- HBM Liberty Ruggedized In-vehicle Data Acquisition System

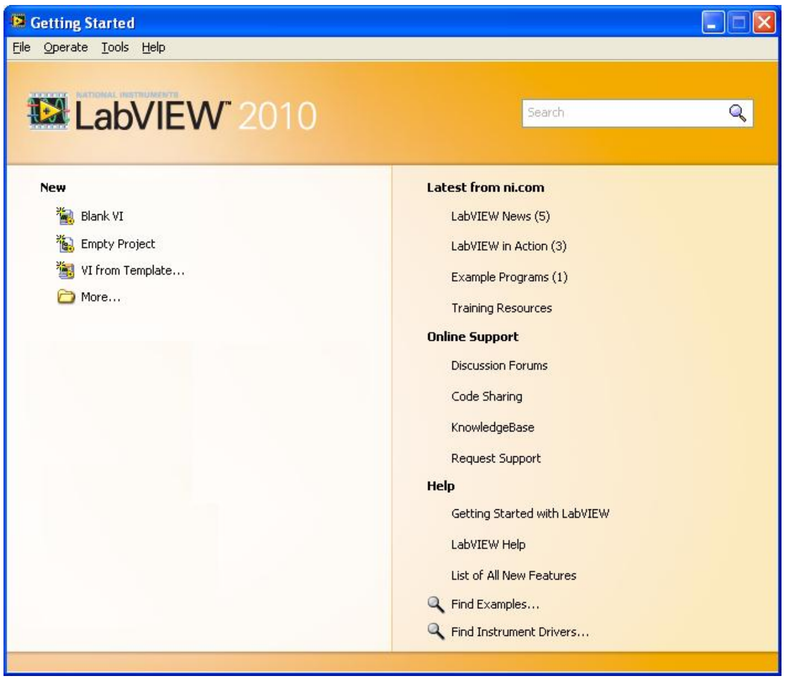
1.3.3 *Installation*

For installation information we refer to the appendix called **Using RPC-COM wrapper and C#** in the Perception RPC interface manual.

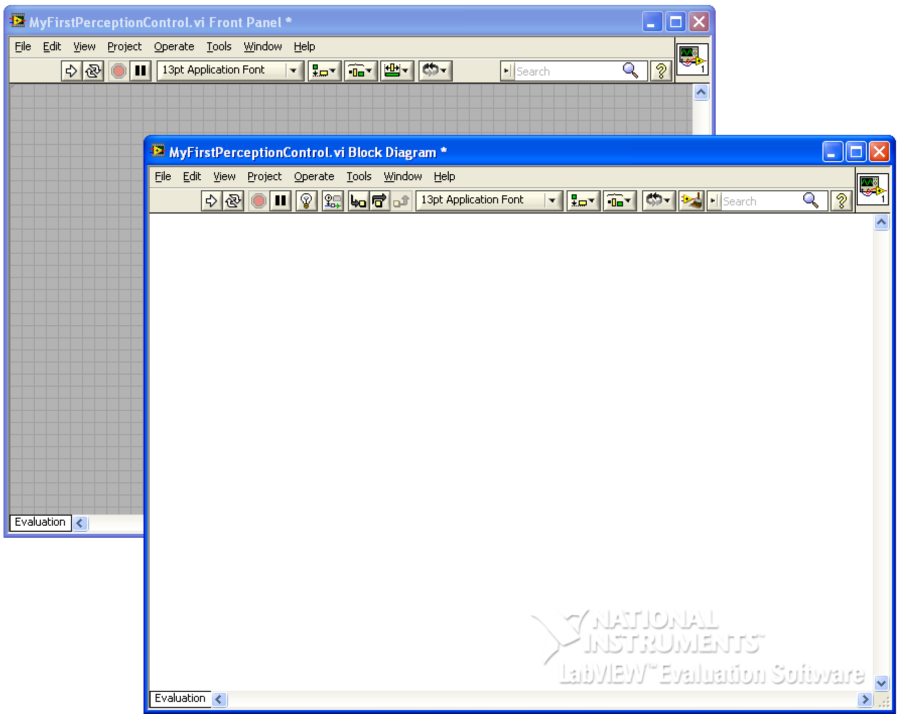
2 Creating your first LabVIEW Perception controlling VI

In this section we show you how you can use the Perception COM wrapper from within LabVIEW. A simple VI is created which will connect to Perception and shows the acquisition state.

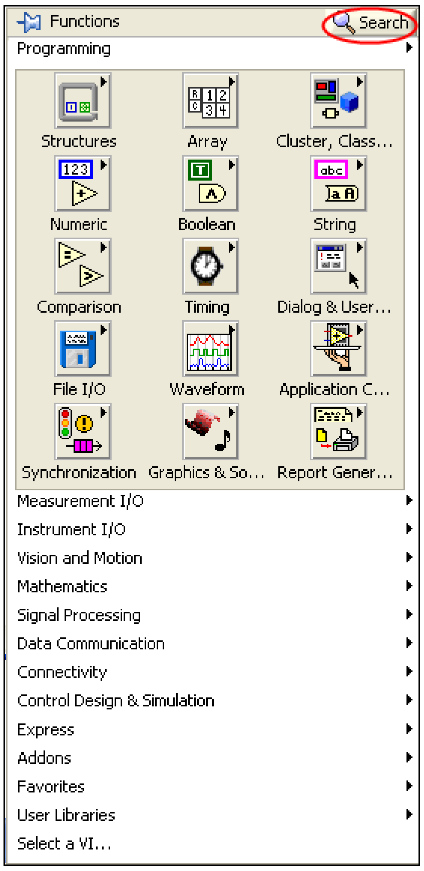
- Start LabVIEW



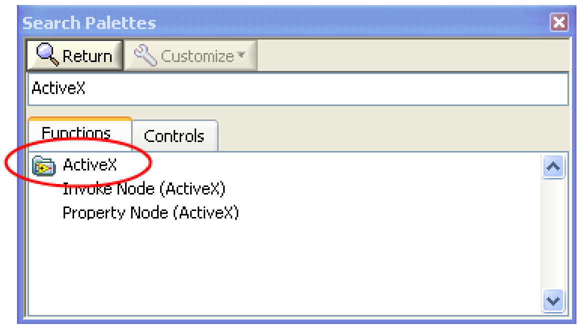
- Select Blank VI
- Save the VI as MyFirstPerceptionControl.


MyFirstPerceptionControl.vi Front Panel *
MyFirstPerceptionControl.vi Block Diagram *
File Edit View Project Operate Tools Window Help
13pt Application Font
Search
Evaluation
NATIONAL INSTRUMENTS
LabVIEW Evaluation Software

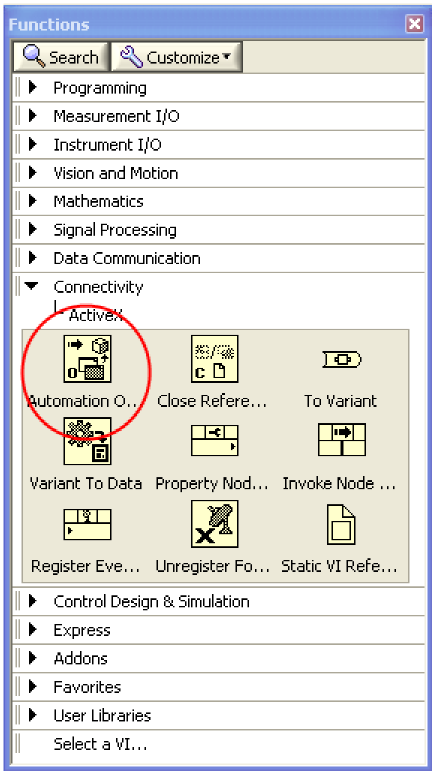
- Right mouse click on the Block Diagram area.



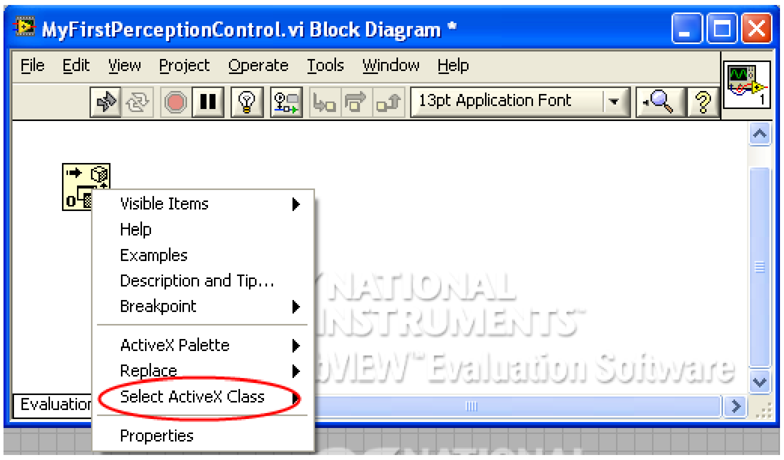
- Use the Search to find the ActiveX functions



- Select the Automation Open function and add it to your block diagram

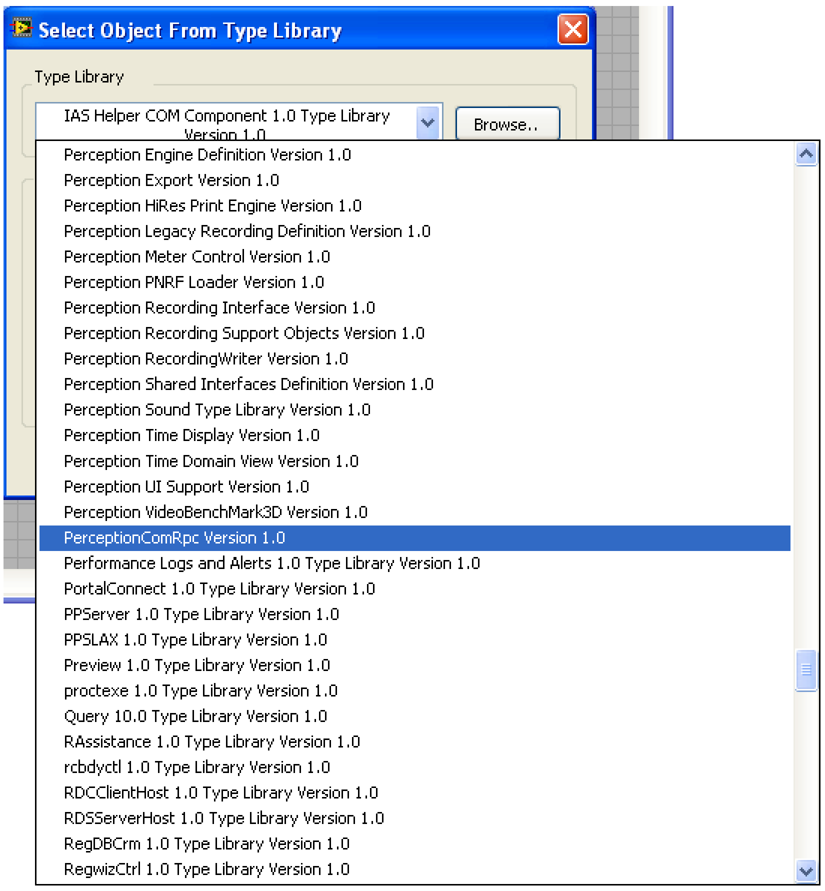


- Right mouse click on the function and select the menu **Select ActiveX Class**



- In this dialog you have to select the **PerceptionComRPC** type library. If you cannot find this type library, you either should install Perception or install the Perception COM-RPC wrapper, see: **Getting Started** of the Appendix **Using RPC-COM wrapper and**

C# of the **Programmers Reference Perception RPC interface.**


Select Object From Type Library

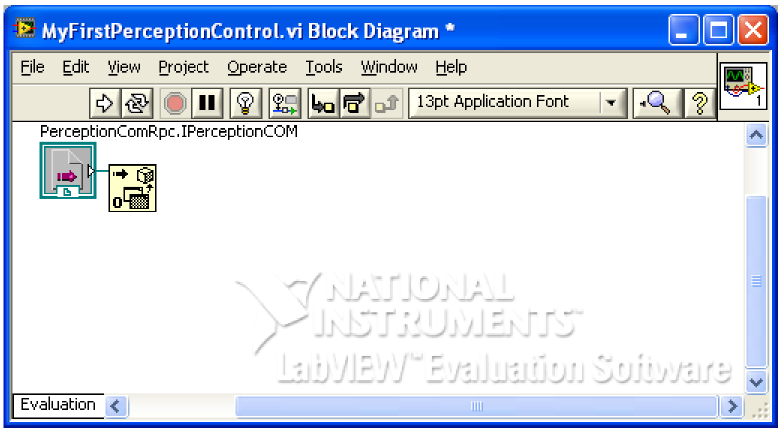
Type Library

IAS Helper COM Component 1.0 Type Library Version 1.0

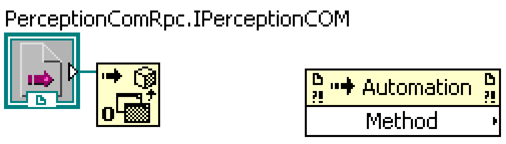
Browse..

Perception Engine Definition Version 1.0
Perception Export Version 1.0
Perception HiRes Print Engine Version 1.0
Perception Legacy Recording Definition Version 1.0
Perception Meter Control Version 1.0
Perception PNRF Loader Version 1.0
Perception Recording Interface Version 1.0
Perception Recording Support Objects Version 1.0
Perception RecordingWriter Version 1.0
Perception Shared Interfaces Definition Version 1.0
Perception Sound Type Library Version 1.0
Perception Time Display Version 1.0
Perception Time Domain View Version 1.0
Perception UI Support Version 1.0
Perception VideoBenchMark3D Version 1.0
PerceptionComRpc Version 1.0
Performance Logs and Alerts 1.0 Type Library Version 1.0
PortalConnect 1.0 Type Library Version 1.0
PPServer 1.0 Type Library Version 1.0
PPSLAX 1.0 Type Library Version 1.0
Preview 1.0 Type Library Version 1.0
proctexe 1.0 Type Library Version 1.0
Query 10.0 Type Library Version 1.0
RAssistance 1.0 Type Library Version 1.0
rcbdyctl 1.0 Type Library Version 1.0
RDCClientHost 1.0 Type Library Version 1.0
RDSServerHost 1.0 Type Library Version 1.0
RegDBCrm 1.0 Type Library Version 1.0
RegwizCtrl 1.0 Type Library Version 1.0

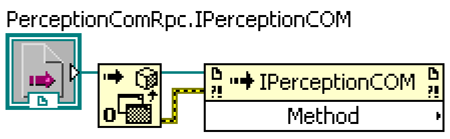
- If you all do this than your block diagram will look like:


MyFirstPerceptionControl.vi Block Diagram *
File Edit View Project Operate Tools Window Help
13pt Application Font
PerceptionComRpc.IPerceptionCOM
NATIONAL INSTRUMENTS
LabVIEW Evaluation Software
Evaluation

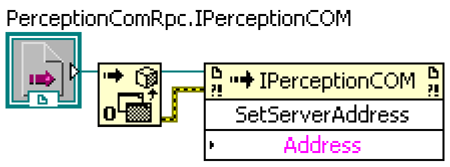
- Add the function **Invoke Node** from the **Functions** toolbox onto your block diagram.



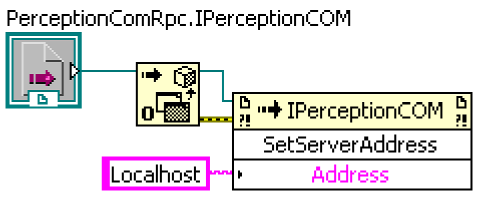
- Connect the Reference input of **the Invoke Node** function to the **Automation Refnum** of the **Automation Open** function. Also connect the **Error in** and **Error out**.



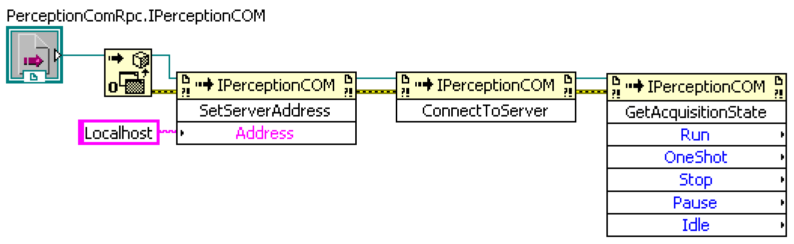
- The name IPerceptionCOM will now be shown. When you click on the Method field all the methods of the IPerceptionCOM interface will be shown. Select the method **SetServerAddress**.



- If Perception is running on the same machine as the one you are working with then use a **constant** with value **Localhost** as input for the Address. If however Perception is running on another machine you should enter the IP address of the remote PC.



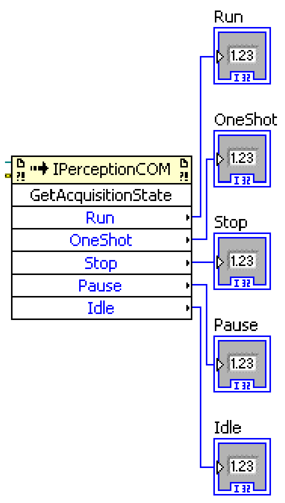
- Add two more methods to the block diagram, see picture below:



- The first new function is used to connect to the RPC Perception server. The last one is used to query the acquisition state, so we can see that we are connected.

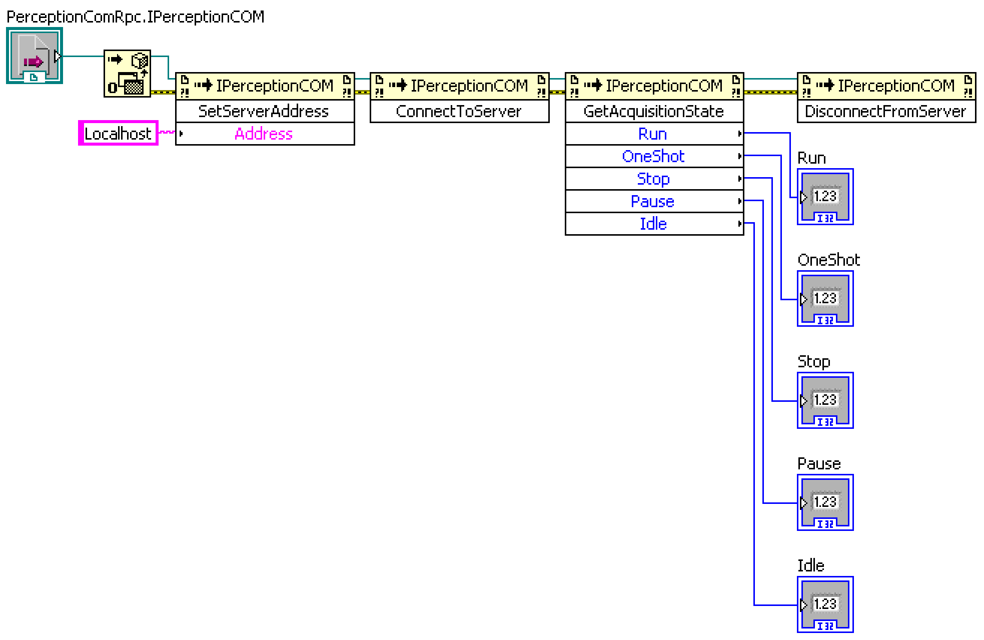
Note: We assume Perception is running and connected to at least one mainframe

- To see the values of the **Run**, **OneShot**, **Stop**, **Pause** and **Idle** states we will add indicators to them. Therefore right mouse click on one of those parameters and select Create -> Indicator.

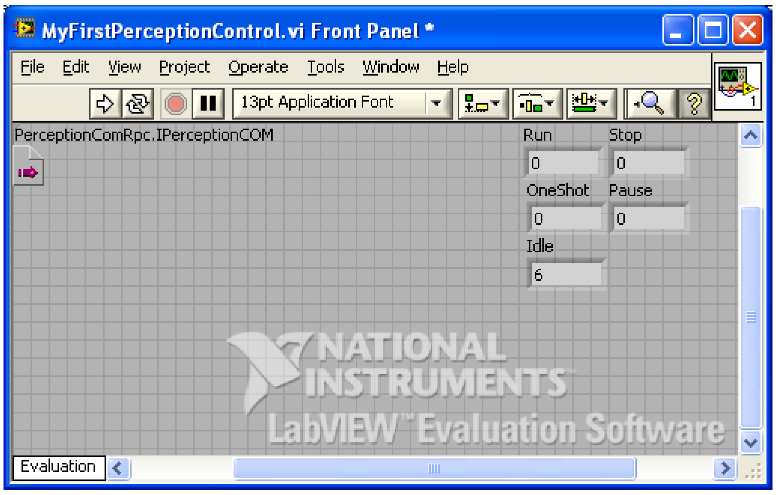


- The last function we will add is **DisconnectFromServer**. Now our first VI is ready. The complete block diagram looks like:

Tip: Use Clean **Up Diagram** (Ctrl+U) when things get messy, use **Remove Broken Wires** (Ctrl+B) to cleanup your block diagram.



- Switch to the Front Panel view and click the Run button.


MyFirstPerceptionControl.vi Front Panel *
File Edit View Project Operate Tools Window Help
13pt Application Font
NATIONAL INSTRUMENTS
LabVIEW Evaluation Software
Evaluation

PerceptionComRpc.IPerceptionCOM

Run 0
Stop 0
OneShot 0
Pause 0
Idle 6

- The indicators now show the acquisition state of the recorders, in the above case Perception was connected to a system with 6 recorders and they are all in idle mode. To test if your program is working you should start a recording in Perception by clicking the Start button in the acquisition control. Rerun your VI and restart it check if the run indicator shows a non zero number.

You can find this VI in the examples, its name is **MyFirstPerceptionControl.vi**

If this all is working and you know how to use LabVIEW than you are ready to create the applications as you want. The description of the IPerceptionCOM interface can be found in the user manual called **"Programmers Reference Perception RPC interface"**. The RPC install also comes with various examples in C++ and C#. We recommend to have a look at the C# examples because they are also using the IPerceptionCOM interface. In the next chapter we will work out some more complicated demos.

3 Simple Acquisition Control

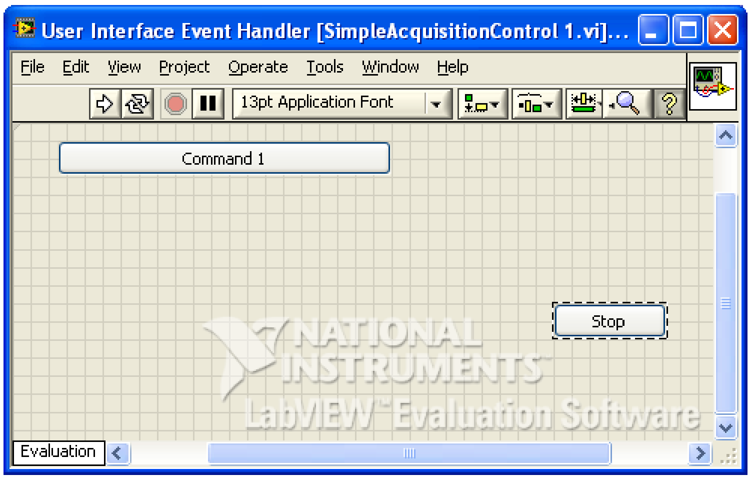
This demo will demonstrate how to create a VI where we can start and stop a recording in Perception.

- We will start by using the default design patterns coming with LabVIEW. For this example you have to select New -> VI from Template...

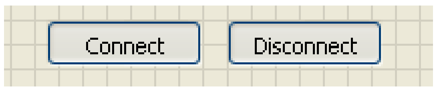
- Select User Interface Event Handler

- Save the VI as SimpleAcquisitionControl 1

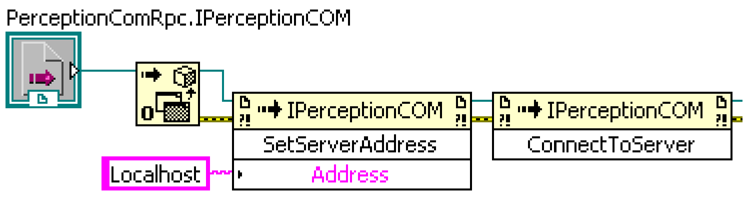
- The front panel looks like:


User Interface Event Handler [SimpleAcquisitionControl 1.vi]...
File Edit View Project Operate Tools Window Help
13pt Application Font
Command 1
Stop
NATIONAL INSTRUMENTS
LabVIEW Evaluation Software
Evaluation

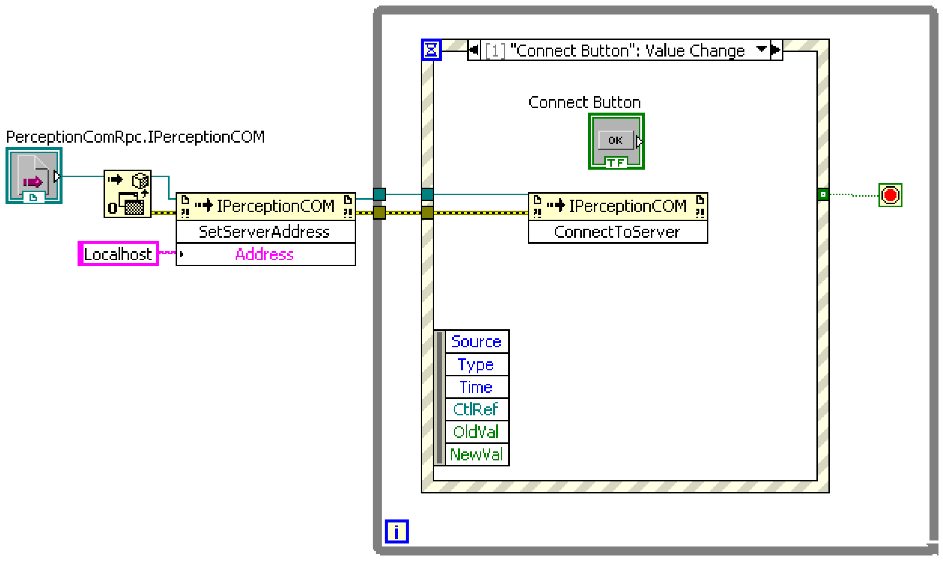
- Resize the button and rename it to Connect Button and change the text to Connect
- Copy this button and rename it to **Disconnect Button** and change the text to **Disconnect**



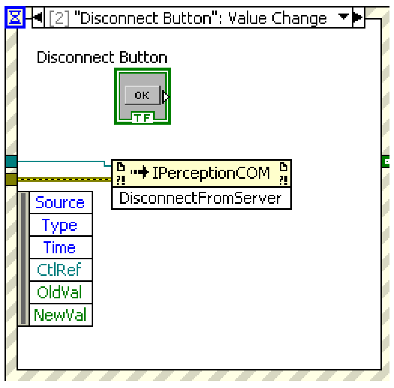
- Switch to the block diagram.
- Here you see a While Loop with an event structure inside.
- Create or copy the following functions from the previous VI into the block diagram:



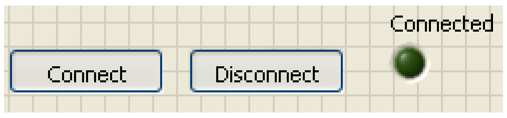
- Add this as follows into the block diagram:



- Add a new event case to the event structure by right mouse click on the header of the structure. Select the **Disconnect Button** and take the value **Change event**



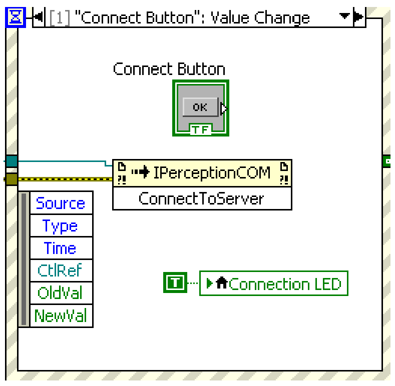
- To give the user visual feedback whether we are connected or not you have to add a connection LED. The name is Connection LED and the label Connected



- Create a Local Variable of the Connection LED. Right click on the function and select Create -> Local Variable

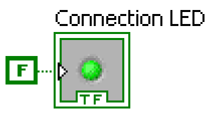
- Locate this local variable into the Connect Button event case.

- Add a constant to its input (Create -> Constant) and set the constant to True (Toggle its value by double click on the constant)



- Repeat this also for the other two event cases but now use False as the input constant.

- Connect a False constant to the Connection LED



- Keep this function out of the loop, this will force the LED to be switched off when starting the VI

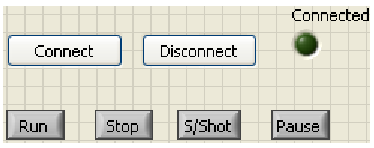
- The VI is ready to run. Go back to the front panel and Run the VI to test if it is working.

- You can find this VI in the examples, its name is SimpleAcquisitionControl 1.vi

- Now we will extend this VI with 4 buttons to control the acquision state of Perception.

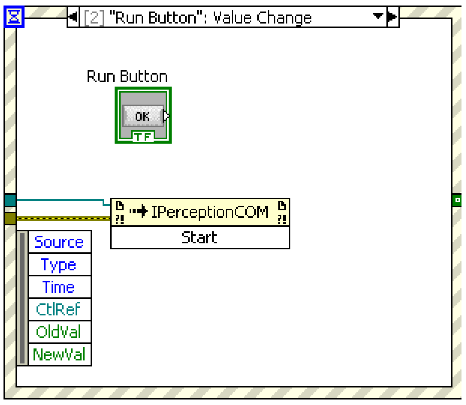
- Continue with the previous example add four new buttons to the front panel

- Name the buttons as follow: Run Button, Stop Button, SingleShot Button and Pause Button



- Create event cases for each button in the event structure.

- Add an Invoke node for each new event case, the picture below shows the event case for the run button:

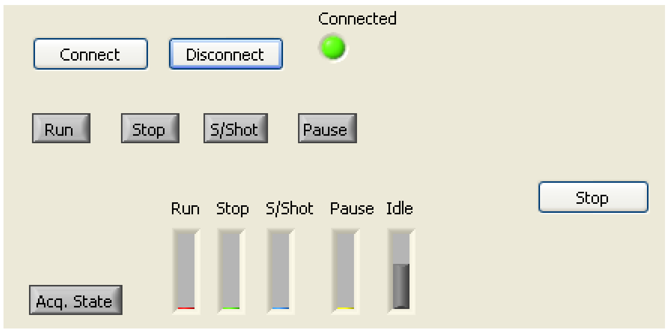


- For the other buttons you should select the following methods: *Stop Button* -> **Stop**, *SingleShot Button* -> **OneShot** and the *Pause Button* -> **Pause**
- The VI is ready to run. Go back to the front panel and Run the VI to test if it is working.

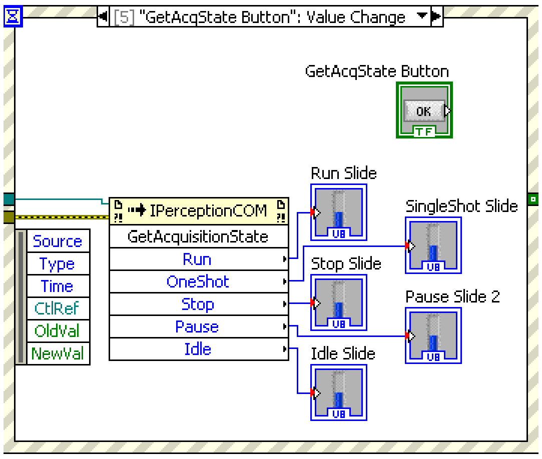
You can find this VI in the examples, its name is **SimpleAcquisitionControl 2.vi**

We now will extend the demo with the possibility to get the acquisition state and show it in the user interface.

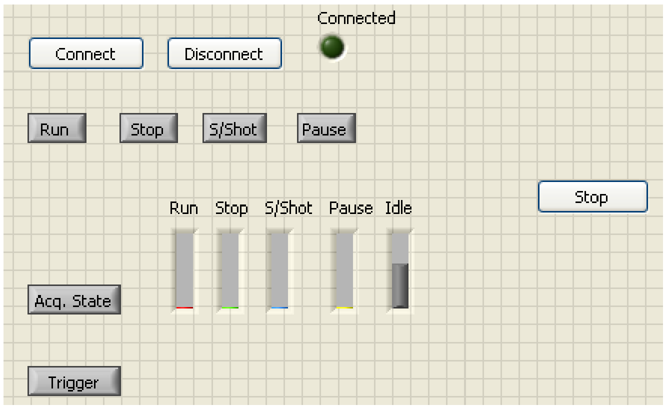
- Continue with the previous example add a new button to the front panel
- Name the buttons as GetAcqState Button.
- Add 5 slide controls set their scale range from 0 to 10.
- Define different fill colors.
- The control panel should look like:



- Switch back to the block diagram and add a new event case for the GetAcqState Button
- Add the function GetAcquisitionState into it.
- Connect the output parameters of this method to the various sliders.

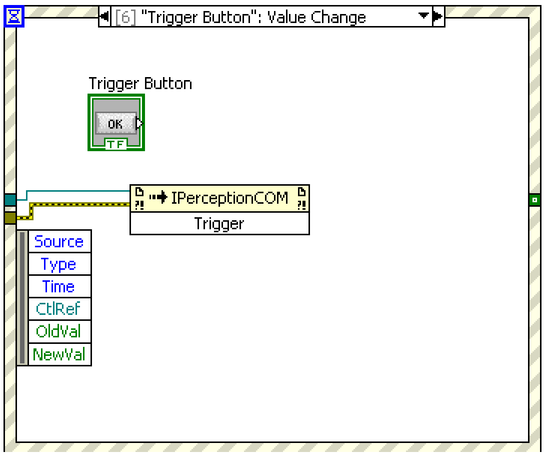


- Now you have to add a button called Trigger Button. This button will be used to generate a manual trigger.



- Add a new event case for this trigger button in the block diagram

- Call the Trigger function in this event case.



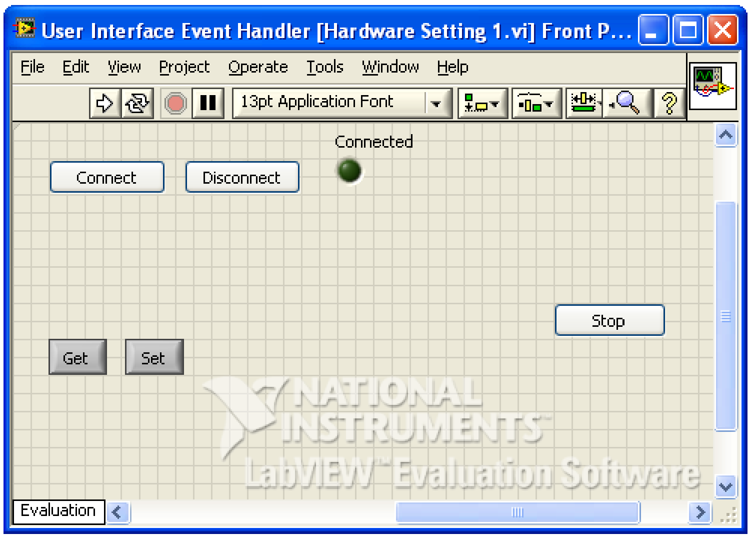
- The VI is ready to run. Go back to the front panel and Run the VI to test if it is working.

You can find this VI in the examples, its name is **SimpleAcquisitionControl 3.vi**

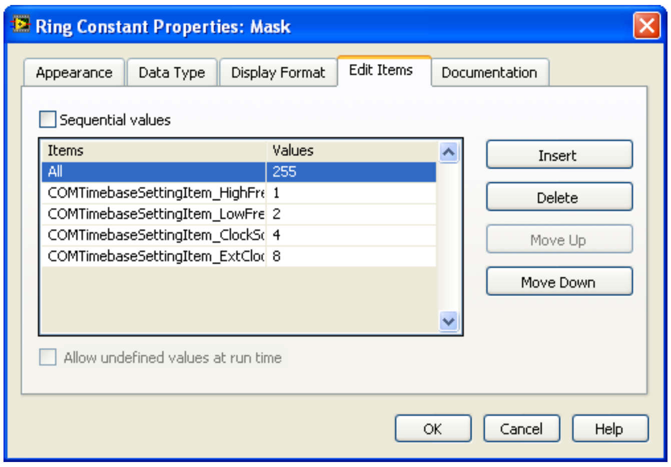
4 Hardware settings

In this example we show you how to get and set the acquisition rate of a recorder. Actually Perception can work with a dual sample rate, **High** and **Low sample rate**. Normally the **High** sample rate is used when you select a recording mode with dual sample rates then the **Low** sample rate is used as well. However when you use the RPC/COM interface the **GetGroupTimebaseSettings** and **SetGroupTimebaseSettings** functions always passes the High and Low sample rates.
The sample rates are set per group, therefore the Timebase setting functions have also the input parameter *Group.*
We will start by copying the previous created VI called **SimpleAcquisitionControl 1.vi** to **Hardware Setting 1.vi**.

- Add a Get and a Set button to the front panel, label them GetTimeBaseBtn and SetTimeBaseBtn.


User Interface Event Handler [Hardware Setting 1.vi] Front P...
File Edit View Project Operate Tools Window Help
13pt Application Font
Connected
Connect
Disconnect
Stop
Get
Set
NATIONAL INSTRUMENTS
LabVIEW Evaluation Software
Evaluation

- Go to the block diagram and add event cases for both buttons.
- Add the SetGroupTimebaseSettings function to the SetTimeBaseBtn event case, see picture below.
- Right mouse click at the Group field and select Create -> Constant and enter “Group1”
- Right mouse click at the Mask field and select Create -> Constant.
- Add a mask value of 255 to the list items and name it “All”



Ring Constant Properties: Mask

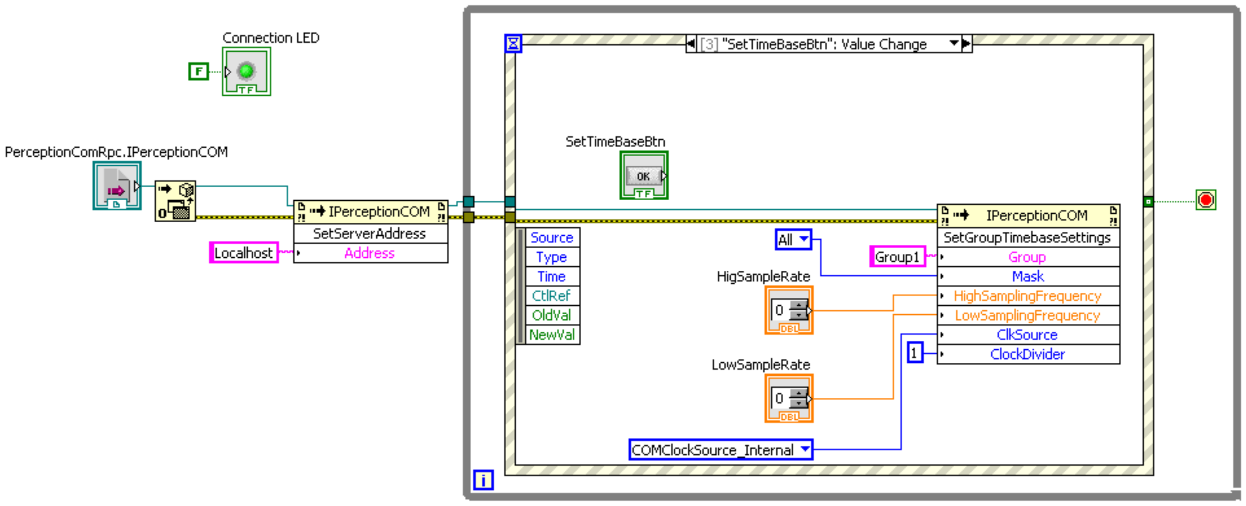
Appearance | Data Type | Display Format | Edit Items | Documentation

☐ Sequential values

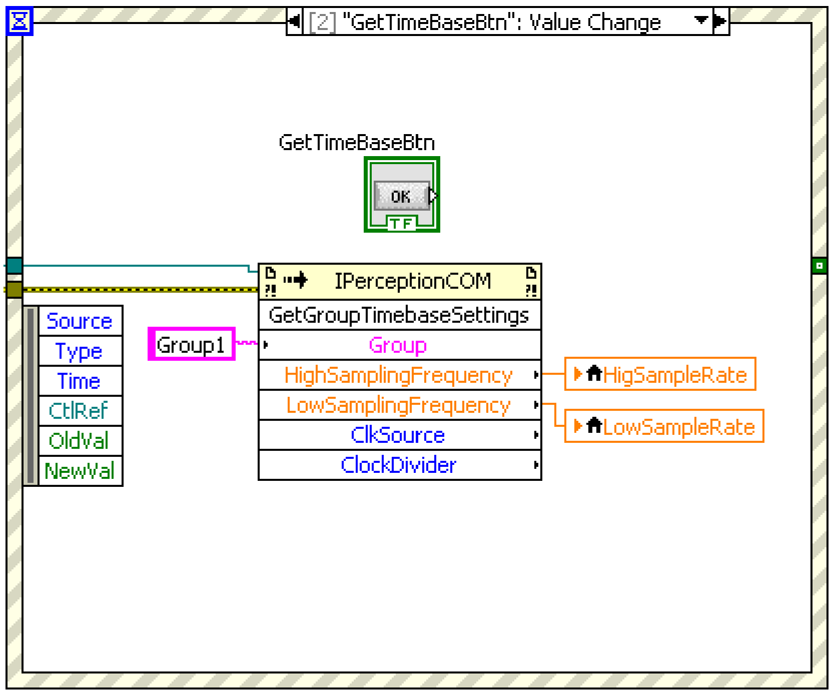
Items	Values
All	255
COMTimebaseSettingItem_HighFre	1
COMTimebaseSettingItem_LowFre	2
COMTimebaseSettingItem_ClockSo	4
COMTimebaseSettingItem_ExtCloc	8

☐ Allow undefined values at run time

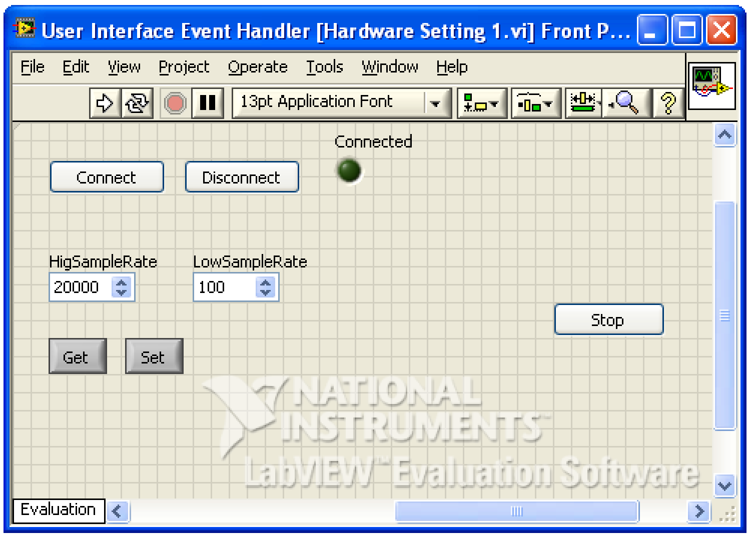
- Right mouse click at the HighSampleRate field and select Create -> Control.
- Do the same for the LowSampleRate.
- Repeat this for the input parameters ClkSource and ClockDivider.
- Create the correct wires and the block diagram should look like:



- Before you go to the GetTimeBaseBtn event case create local variables of the HighSampleRate and LowSampleRate controls.
- Use these local variables in the GetTimeBaseBtn event case, see picture below.



- Now you should go back to the front panel and move the High and Low sample rate controls to a nice position. For example:


User Interface Event Handler [Hardware Setting 1.vi] Front P...
File Edit View Project Operate Tools Window Help
13pt Application Font
Connected
Connect
Disconnect
Stop
Get
Set
NATIONAL INSTRUMENTS LabVIEW Evaluation Software
Evaluation

HigSampleRate 20000

LowSampleRate 100

- Your VI is ready save it and run it, make sure Perception is running and connected to at least one mainframe.

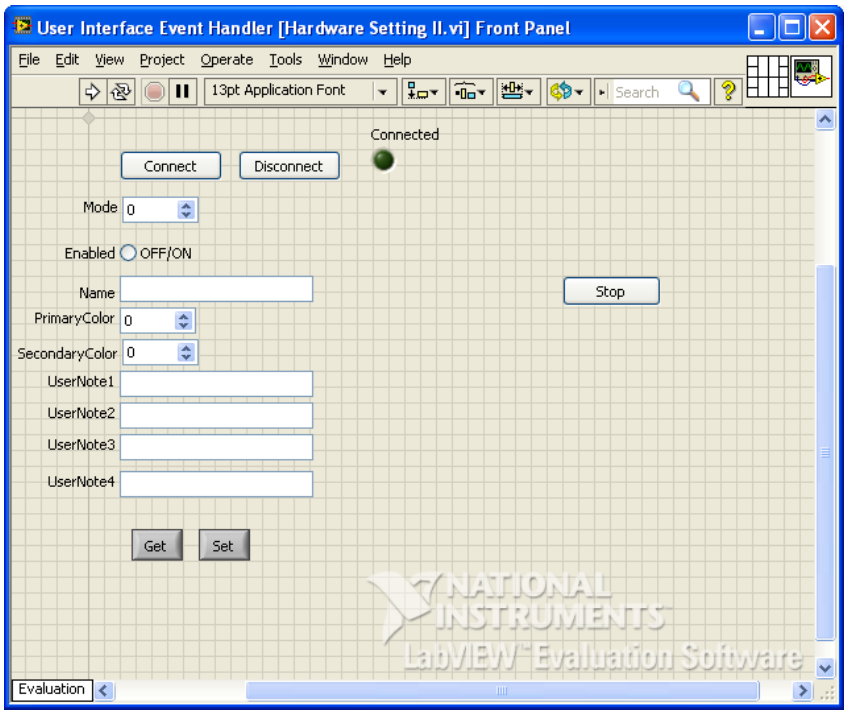
You can find this VI in the examples, its name is **Hardware Setting 1.vi**

In the following example we show you how to get and set some common settings of a channel.
The commom settings are set per channel, therefore the **Get**- and **SetSCCommonSettings** functions have also the input parameters *Mainframe, Recorder* and *Channel.* These parameters are working with 1 based index strings. For example when you want to select the first mainframe use the following string "**1**". The following **LabVIEW** example will use the first channel of the first recorder of the first mainframe. The function for getting the common settings will look like:

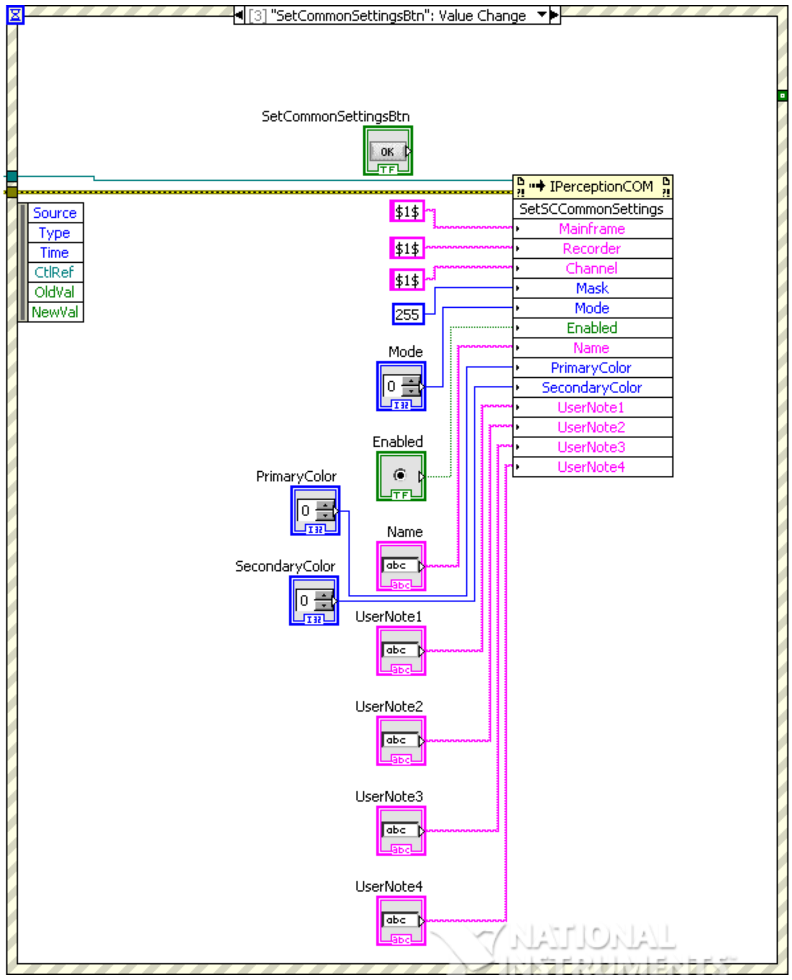
GetCommonSettings("1", "1", "1", Mask, Mode, Enabled, Name, PrimaryColor, SecondaryColor, UserNote1, UserNote2, UserNote3, UserNote4)

We will start by copying the previous created VI called **SimpleAcquisitionControl 1.vi** to **Hardware Setting II.vi**.

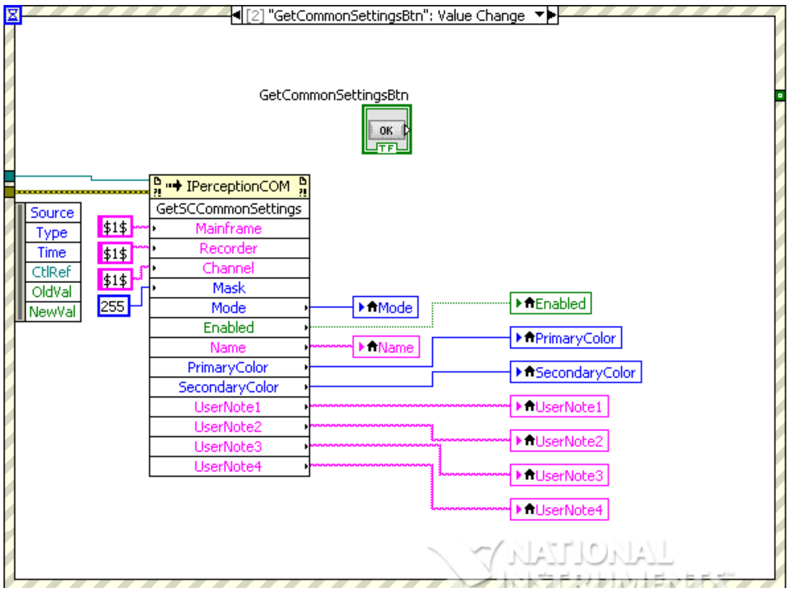
- Modify the user interface to:



- The SetCommonSettingsBtn event looks like:



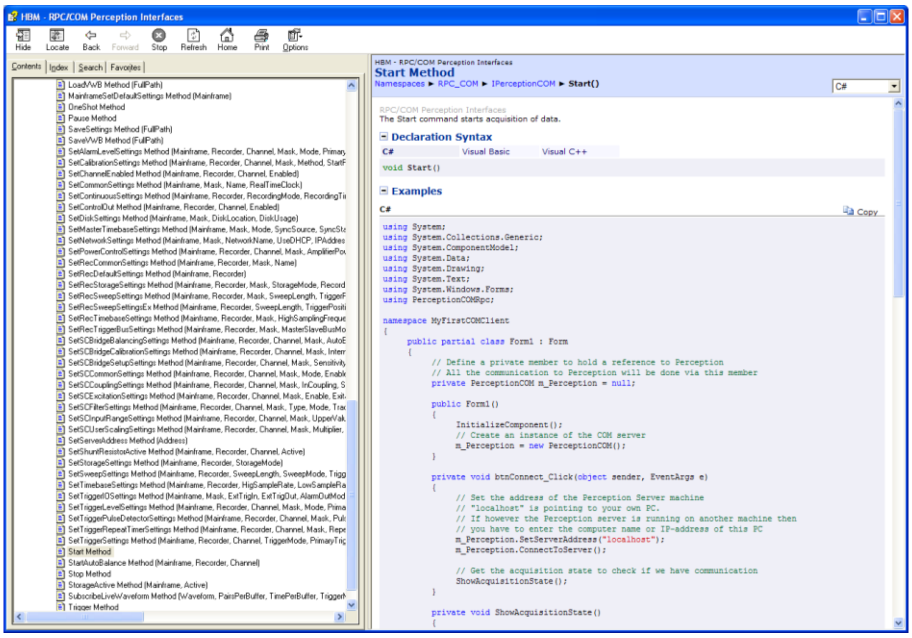
- The getCommonSettingsBtn:



You can find this VI in the examples, its name is **Hardware Setting II.vi**

5 Recommendations

So far we just showed some simple examples to get started, but the RPC/COM interface is much richer than the examples above have shown. To get more insights in this interface we recommend to have a look into the **“Programmers Reference Perception RPC interface”** (I2699_1.0en) and the **Perception COM Help** help file.



HBM - RPC/COM Perception Interfaces

Hide Locate Back Forward Stop Refresh Home Print Options

Contents | Index | Search | Favorites

- LoadVWB Method (FullPath)
- MainframeSetDefaultSettings Method (Mainframe)
- OneShot Method
- Pause Method
- SaveSettings Method (FullPath)
- SaveVWB Method (FullPath)
- SetAlarmLevelSettings Method (Mainframe, Recorder, Channel, Mask, Mode, Primary
- SetCalibrationSettings Method (Mainframe, Recorder, Channel, Mask, Method, StartF
- SetChannelEnabled Method (Mainframe, Recorder, Channel, Enabled)
- SetCommonSettings Method (Mainframe, Mask, Name, RealTimeClock)
- SetContinuousSettings Method (Mainframe, Recorder, RecordingMode, RecordingTi
- SetControlOut Method (Mainframe, Recorder, Channel, Enabled)
- SetDiskSettings Method (Mainframe, Mask, DiskLocation, DiskUsage)
- SetMasterTimebaseSettings Method (Mainframe, Mask, Mode, SyncSource, SyncSta
- SetNetworkSettings Method (Mainframe, Mask, NetworkName, UseDHCP, IPAddres
- SetPowerControlSettings Method (Mainframe, Recorder, Channel, Mask, AmplifierPo
- SetRecCommonSettings Method (Mainframe, Recorder, Mask, Name)
- SetRecDefaultSettings Method (Mainframe, Recorder)
- SetRecStorageSettings Method (Mainframe, Recorder, Mask, StorageMode, Record
- SetRecSweepSettings Method (Mainframe, Recorder, Mask, SweepLength, TriggerF
- SetRecSweepSettingsEx Method (Mainframe, Recorder, SweepLength, TriggerPositi
- SetRecTimebaseSettings Method (Mainframe, Recorder, Mask, HighSamplingFreque
- SetRecTriggerBusSettings Method (Mainframe, Recorder, Mask, MasterSlaveBusMo
- SetSCBridgeBalancingSettings Method (Mainframe, Recorder, Channel, Mask, AutoE
- SetSCBridgeCalibrationSettings Method (Mainframe, Recorder, Channel, Mask, Interr
- SetSCBridgeSetupSettings Method (Mainframe, Recorder, Channel, Mask, Sensitivity
- SetSCCommonSettings Method (Mainframe, Recorder, Channel, Mask, Mode, Enabl
- SetSCCouplingSettings Method (Mainframe, Recorder, Channel, Mask, InCoupling, S
- SetSCExcitationSettings Method (Mainframe, Recorder, Channel, Mask, Enable, Exit
- SetSCFilterSettings Method (Mainframe, Recorder, Channel, Mask, Type, Mode, Trac
- SetSCInputRangeSettings Method (Mainframe, Recorder, Channel, Mask, UpperVal
- SetSCUserScalingSettings Method (Mainframe, Recorder, Channel, Mask, Multiplier,
- SetServerAddress Method (Address)
- SetShuntResistorActive Method (Mainframe, Recorder, Channel, Active)
- SetStorageSettings Method (Mainframe, Recorder, StorageMode)
- SetSweepSettings Method (Mainframe, Recorder, SweepLength, SweepMode, Trigg
- SetTimebaseSettings Method (Mainframe, Recorder, HigSampleRate, LowSampleRa
- SetTriggerIOSettings Method (Mainframe, Mask, ExtTrigIn, ExtTrigOut, AlarmOutMod
- SetTriggerLevelSettings Method (Mainframe, Recorder, Channel, Mask, Mode, Prima
- SetTriggerPulseDetectorSettings Method (Mainframe, Recorder, Channel, Mask, Pul
- SetTriggerRepeatTimerSettings Method (Mainframe, Recorder, Channel, Mask, Repe
- SetTriggerSettings Method (Mainframe, Recorder, Channel, TriggerMode, PrimaryTrig
- Start Method
- StartAutoBalance Method (Mainframe, Recorder, Channel)
- Stop Method
- StorageActive Method (Mainframe, Active)
- SubscribeLiveWaveform Method (Waveform, PairsPerBuffer, TimePerBuffer, TriggerM
- Trigger Method

HBM - RPC/COM Perception Interfaces

Start Method

Namespaces ► RPC_COM ► IPerceptionCOM ► **Start()**

C#

RPC/COM Perception Interfaces

The Start command starts acquisition of data.

Declaration Syntax

C#	Visual Basic	Visual C++

```
void Start()
```

Examples

C# Copy

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Text;
using System.Windows.Forms;
using PerceptionCOMRpc;

namespace MyFirstCOMClient
{
    public partial class Form1 : Form
    {
        // Define a private member to hold a reference to Perception
        // All the communication to Perception will be done via this member
        private PerceptionCOM m_Perception = null;

        public Form1()
        {
            InitializeComponent();
            // Create an instance of the COM server
            m_Perception = new PerceptionCOM();
        }

        private void btnConnect_Click(object sender, EventArgs e)
        {
            // Set the address of the Perception Server machine
            // "localhost" is pointing to your own PC.
            // If however the Perception server is running on another machine then
            // you have to enter the computer name or IP-address of this PC
            m_Perception.SetServerAddress("localhost");
            m_Perception.ConnectToServer();

            // Get the acquisition state to check if we have communication
            ShowAcquisitionState();
        }

        private void ShowAcquisitionState()
        {
```

The install also contains some more sofisticated LabVIEW example which are not described in this manual, but may help you to create your own LabVIEW VI’s.
We also recommand to have a look in the C# example programs.
It is also possible to join a COM/RPC training to understand the interface, this training does not use LabVIEW but C# and Microsoft Visual studio.
HBM also provides programming support, this support can be bought in blocks of 8 hours.

Head Office
HBM
Im Tiefen See 45
64293 Darmstadt
Germany
Tel: +49 6151 8030
Email: info@hbm.com

France
HBM France SAS
46 rue du Champoreux
BP76
91542 Mennecy Cedex
Tél:+33 (0)1 69 90 63 70
Fax: +33 (0) 1 69 90 63 80
Email: info@fr.hbm.com

Germany
HBM Sales Office
Carl-Zeiss-Ring 11-13
85737 Ismaning
Tel: +49 89 92 33 33 0
Email: info@hbm.com

UK
HBM United Kingdom
1 Churchill Court, 58 Station Road
North Harrow, Middlesex, HA2 7SA
Tel: +44 (0) 208 515 6100
Email: info@uk.hbm.com

USA
HBM, Inc.
19 Bartlett Street
Marlborough, MA 01752, USA
Tel : +1 (800) 578-4260
Email: info@usa.hbm.com

PR China
HBM Sales Office
Room 2912, Jing Guang Centre
Beijing, China 100020
Tel: +86 10 6597 4006
Email: hbmchina@hbm.com.cn

measure and predict with confidence

I3290-1.2_en