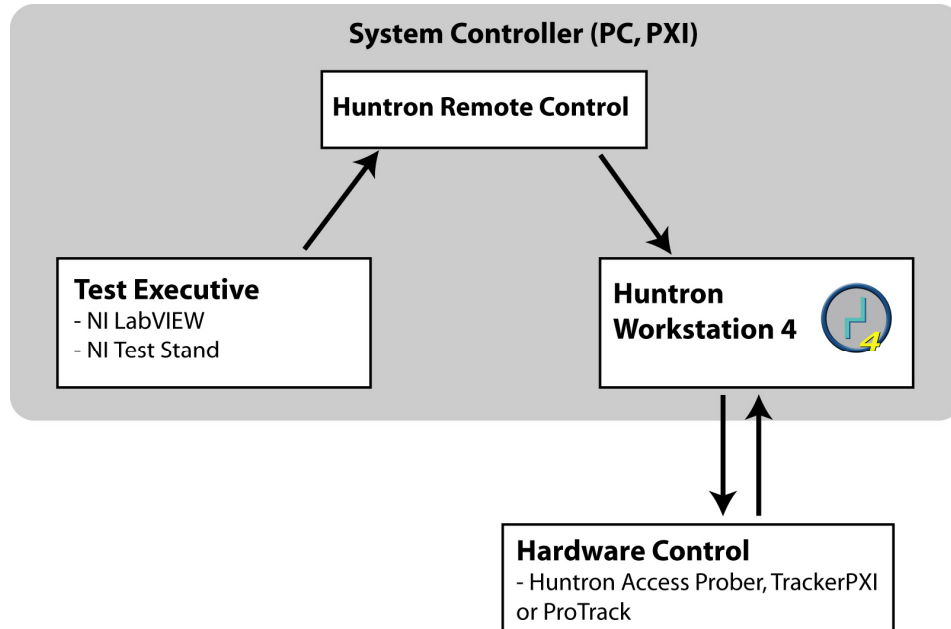
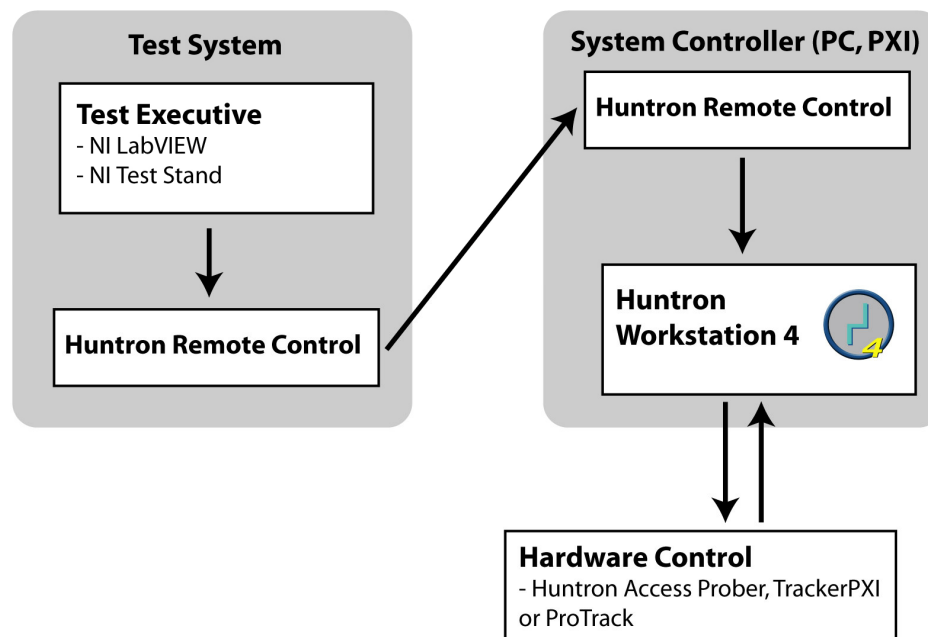


Huntron Workstation Remote Control

The Huntron Workstation optional Remote Control feature allows control of the software from other programs. Its main purpose is to allow scans of sequences and components using a Tracker with a Scanner or a Prober. Tests are created and verified in Huntron Workstation and then “controlled” by other programs. Test results can be produced in an ASCII file or generated PDF files. In the future information could be provided to allow accessing test information from the Huntron Workstation MDB file.



Remote Control through a single Controller



Remote Control through a Test Executive on a separate controller

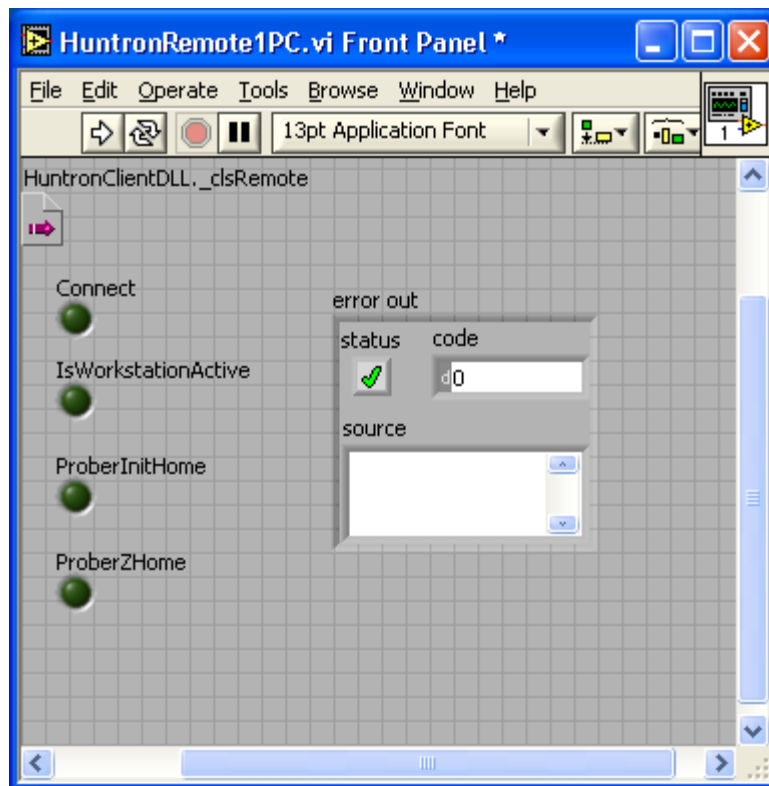
Other features include controlling a Prober to put the pin down on a pin to take a measurement with another tester, capturing a camera image over a selected pin and retrieving scan and reference signature data for the selected pin.

The process of adding Huntron Tracker and Prober capabilities to other programs and testers using drivers requires a lot of programming by the customer. Huntron Workstation Remote Control makes this a lot easier by providing most of the functionality needed with minimal programming.

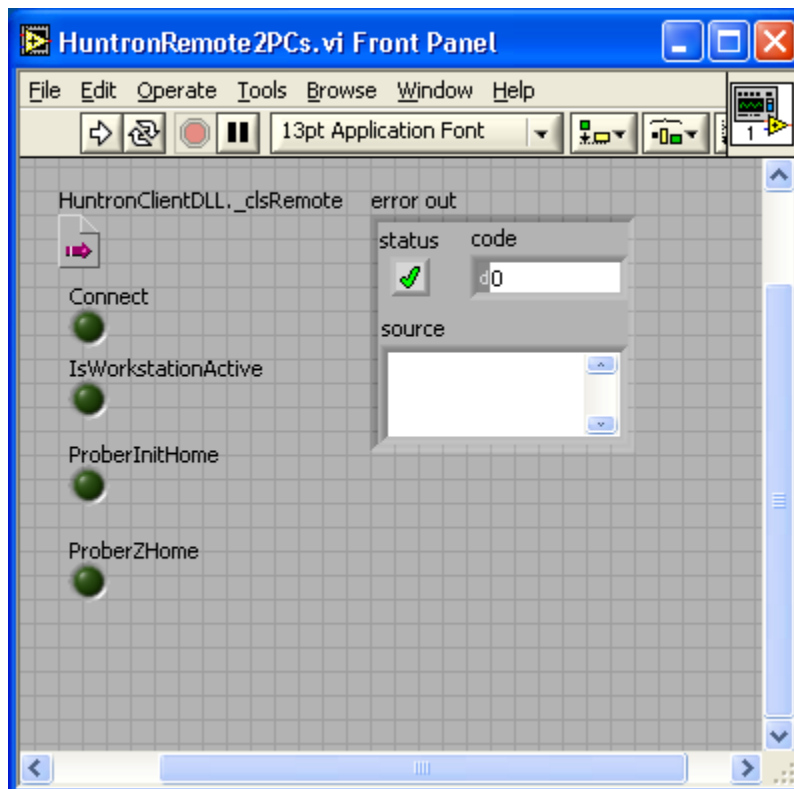
Remote Control is used by programs talking to the Huntron Client VB.NET DLL (HuntrulClient.dll). The DLL exposes functions that are called by the programs to control Huntron Workstation.

Below are screens captures of the sample programs and a detailed listed of the functions available.

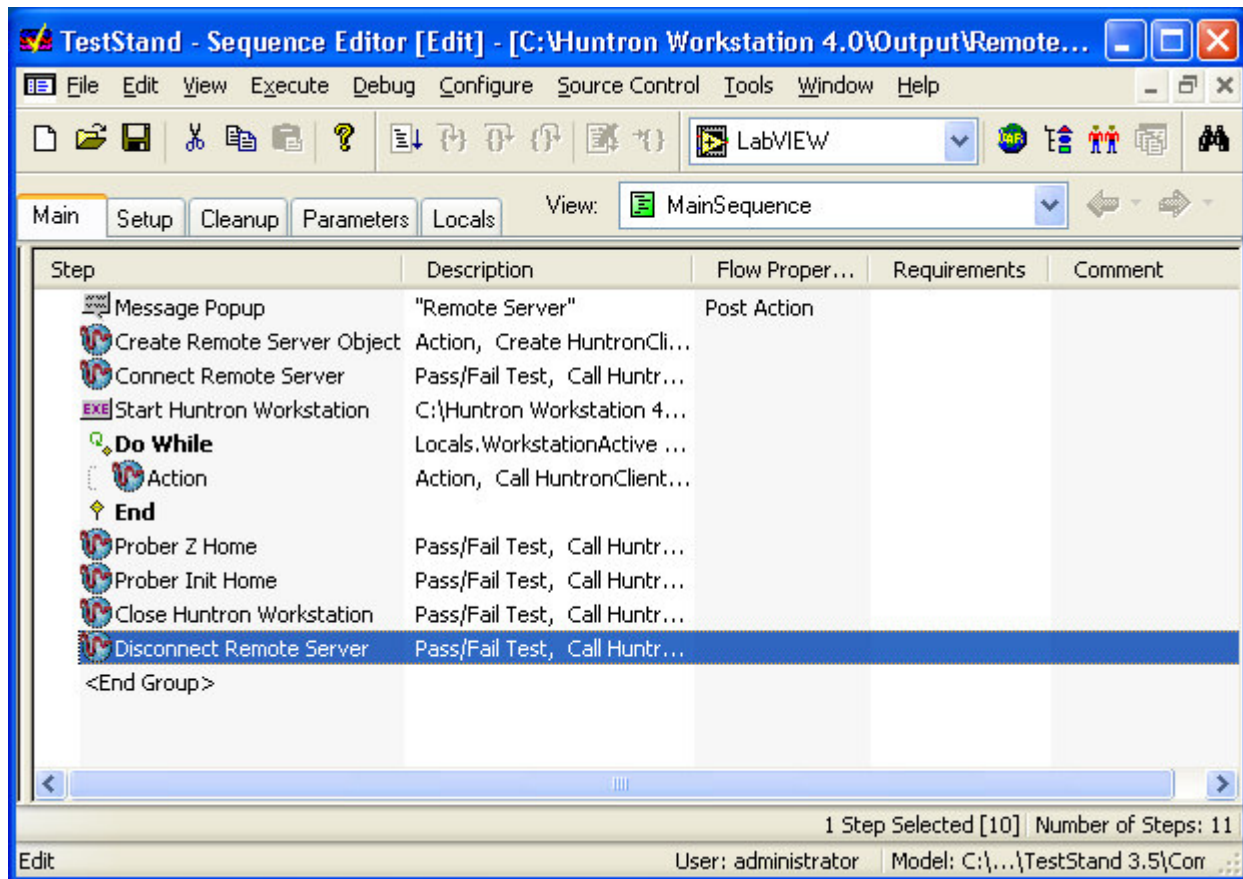
Huntron Remote Control Sample VB.Net Program (HuntronRemoteApp.exe)



Huntron Remote One PC Sample LabVIEW Program (HuntronRemote1PC.vi)



Huntron Remote Two PC Sample LabVIEW Program (HuntronRemote1PC.vi)



Huntron Remote Sample TestStand Program (RemoteControl.seq)

Huntron Client DLL Function List:

Function Connect(ByVal sIPAddress As String) As Boolean

Connects to the Huntron Remote Server. To connect to the server on the same PC, sIPAddress should be set to "localhost". To connect to the server on another PC, sIPAddress should be set to the IP address of that PC. The Huntron Remote server runs on the PC where Huntron Workstation is running. Returns true if the function succeeds.

Function Disconnect() As Boolean

Disconnects from the Huntron Remote. Server Returns true if the function succeeds.

Function OpenFile(ByVal MDBPath As String) As Boolean

Loads a Huntron Workstation created MDB board database file into Huntron Workstation. Set MDBPath to the full path name of the MDB file (i.e. "C:\Documents and Settings\username\My Documents\Huntron\Boards\Test.mdb". Returns true if the function succeeds.

Function SelectSequence(ByVal SequenceName As String) As Boolean

Selects a sequence of the loaded board. SequenceName should match one of the sequence names on the Sequences grid in Huntron Workstation. Returns true if the function succeeds.

Function SelecComponentNet(ByVal ComponentNetName As String) As Boolean

Selects a Component or Net of the selected sequence. ComponentNetName should match one of the Component or Net names on the Component/Net grid in Huntron Workstation. Returns true if the function succeeds.

Function SelectPin(ByVal PinNumber As String) As Boolean

Selects a Pin of the selected component or net. PinNumber should match one of the Pin numbers on the Pin grid in Huntron Workstation. Returns true if the function succeeds.

Function ScanSequence(ByVal SerialNumber As String, ByVal Operator As String, ByRef iFailed As Integer) As Boolean

Scans the selected sequence in Huntron Workstation. SerialNumber should be set to the serial number of the board being scanned. Operator should be set the name of the operator running the test. iFailed will be set to the number of components that failed during the scan. Returns true if the function succeeds.

Function ScanComponent(ByVal SerialNumber As String, ByVal Operator As String, ByRef iFailed As Integer) As Boolean

Scans the selected Component/Net in Huntron Workstation. SerialNumber should be set to the serial number of the board being scanned. Operator should be set the name of the operator running the test. iFailed will be set to the number of components that failed during the scan. Returns true if the function succeeds.

Function ScanList(ByVal SerialNumber As String, ByVal Operator As String, ByVal ListPath As String, ByRef iFailed As Integer) As Boolean

Scans the selected sequence using the provided scan list in Huntron Workstation. SerialNumber should be set to the serial number of the board being scanned. Operator should be set the name of the operator running the test. ListPath should be set to the full path of the scan list file to be used (i.e. "C:\Documents and Settings\username\My Documents\Huntron\Lists\Test.lst"). iFailed will be set to the number of components that failed during the scan. Returns true if the function succeeds.

Function ProberAlign() As Boolean

Uses the Huntron Workstation Auto Align feature to align the board in the Prober for scanning. Make sure that Auto Align performs successfully in Huntron Workstation before using it through Remote Control. Returns true if the function succeeds.

Function ProberPinDown() As Boolean

Moves the Prober probe tip down on to the selected pin. PinNumber should match one of the Pin numbers on the Pin grid in Huntron Workstation. Returns true if the function succeeds.

Function ProberZHome() As Boolean

Moves the Prober probe tip all the way up. Returns true if the function succeeds.

Function ProberInitHome() As Boolean

Moves the Prober probe tip all the way up and to the back right corner. Returns true if the function succeeds.

Function CameraPicture(ByVal Filepath As String, ByVal FileType As String) As Boolean

Captures an image from the Prober camera at the location of the probe tip. FilePath should be the full file path of the location for the image file. FileType should be set to "BMP", "JPEG" or "PNG". Anything else defaults to "BMP". The FilePath filename extension should match the FileType (.bmp for "BMP", .jpg for "JPEG" and .png for "PNG"). Returns true if the function succeeds.

Function CloseWorkstation() As Boolean

Closes Huntron Workstation. This should be done before disconnecting. Returns true if the function succeeds.

Function IsWorkstationActive() As Boolean

Checks to see if Huntron Workstation in running and available. Returns true if the function succeeds.

Function GetPinSignatures(ByRef baSigData() As Byte, ByRef iRanges As Integer) As Boolean

Gets the signature data for all the ranges of the currently selected pin from the last scan. baSigData returns an array of bytes. Its length is equal to the value of iRanges * 200. iRanges is set to the number of ranges that there are signatures for in baSigData. The first 100 bytes of the signature data is the horizontal waveform and the second hundred bytes are the vertical waveform. To display a signature, plot the corresponding horizontal and vertical bytes in XY. Returns true if the function succeeds.

Function GetReferencePinSignatures(ByRef baSigData() As Byte, ByRef iRanges As Integer) As Boolean

Gets the signature data for the reference signatures for all the ranges of the currently selected pin from the last scan. baSigData returns an array of bytes. Its length is equal to the value of iRanges * 200. iRanges is set to the number of ranges that there are signatures for in baSigData. The first 100 bytes of the signature data is the horizontal waveform and the second hundred bytes are the vertical waveform. To display a signature, plot the corresponding horizontal and vertical bytes in XY. Returns true if the function succeeds.

Function ScanPin(ByVal SerialNumber As String, ByRef iFailed As Integer, ByRef baSigData() as Byte, ByRef iRanges as Integer, ByRef baRefSigData() as Byte) As Boolean

Scans the selected Pin in Huntron Workstation. SerialNumber should be set to the serial number of the board being scanned. iFailed will be set to the number of components that failed during the scan. iRanges will be set to the number of ranges that there are signatures for in baSigData. A scan is not created in Huntron Workstation. Returns true if the function succeeds.

Function GetExternalPinSignatures(ByRef daWaveforms() As Double, ByRef iaWaveformXSizes() As Integer, ByRef iaWaveformYSizes() As Integer, ByRef daWaveformXMin() As Double, ByRef daWaveformXMax() As Double, ByRef daWaveformYMin() As Double, ByRef daWaveformYMax() As Double, ByRef baWaveformOvertimes() As Boolean, ByRef daReadings() As Double, ByRef daReadings2() As Double, ByRef daReadings3() As Double, ByRef msalimages() As System.IO.MemoryStream, ByRef iRanges As Integer) As Boolean

Gets the External Hardware reading, waveform and image data for all the ranges of the currently selected pin from the last scan. daWaveforms returns an array of doubles. Its length is equal to the value of iaWaveformXSizes + iaWaveformYSizes for each range. daWaveformXMin, daWaveformXMax, daWaveformYMin, daWaveformYMax are the min and max data values for each range. daReadings, daReadings2() and daReadings3 are the reading values for each range. msalimages are the image data for each range. iRanges is set to the number of ranges. This function is used for NFS remote control.

Function GetExternalReferencePinSignatures(ByRef daWaveforms() As Double, ByRef iaWaveformXSizes() As Integer, ByRef iaWaveformYSizes() As Integer, ByRef daWaveformXMin() As Double, ByRef daWaveformXMax() As Double, ByRef daWaveformYMin() As Double, ByRef daWaveformYMax() As Double, ByRef baWaveformOvertimes() As Boolean, ByRef daReadings() As Double, ByRef daReadings2() As Double, ByRef daReadings3() As Double, ByRef msalimages() As System.IO.MemoryStream, ByRef iRanges As Integer) As Boolean

Gets the External Hardware reading, waveform and image data for all the ranges of the currently selected pin from the last scan. daWaveforms returns an array of doubles. Its length is equal to the value of iaWaveformXSizes + iaWaveformYSizes for each range. daWaveformXMin, daWaveformXMax, daWaveformYMin, daWaveformYMax are the min and max data values for each range. daReadings, daReadings2() and daReadings3 are the reading values for each range. msalimages are the image data for each range. iRanges is set to the number of ranges. This function is used for NFS remote control.