














Gantner Instruments Lab View e.gate driver manual

1. Table of Content

1. Table of Content.....	1
2. About This Manual.....	2
3. Driver Installation.....	2
4. Driver Description.....	3
Brief protocol description.....	3
Types of data	3
5. VI Overview.....	4
6. General steps for Communication	5
7. Detailed VI Description	5
 e.gate Initialize.vi	5
 e.gate ASCII Read Value.vi	6
 e.gate ASCII Write Value.vi	6
 e.gate ASCII Read All Variables.vi	7
 e.gate HighSpeedPort Online Communication.vi	7
 e.gate HighspeedPort Read Buffer.vi	9
 e.gate Cleanup.vi	11
 e.gate Get File Via HighSpeed Protocol.vi	11
 e.gate RealTimeClock.vi	13
 e.gate ReadeStateInfo.vi.....	13
 e.gate Decode UDBF.vi	15
 e.gate HighSpeedPort Diagnostics.vi	16
8. Examples	18
e.gate Example - ASCII Read Value.vi.....	18
e.gate Example – ASCII Write Value.vi	19
e.gate Example - HighSpeedPort Get File.vi	19
e.gate Example - HighSpeedPort Online Communication.vi.....	20
e.gate Example - HighSpeedPort Read Buffer.vi.....	20
e.gate Example - Real Time Clock.vi	21
e.gate Exemple - Device Information.vi	22
e.gate Example – Decode UDBF.vi	22
e.gate Example – HighSpeedPort Read Diagnostics.vi.....	23

9. Core VI Overview	24
10. e.gate Type Descriptions	25
 e.gate Channel Data	25
e.gate Configuration	26
e.gate Time	27
e.gate Diagnostic Interface Data	27
e.gate Diagnostic Transport Data	27
e.gate Diagnostic Variable Data	27
e.gate Protocol Type	27
11. Error Codes.....	28

2. About This Manual

This document describes how to use Gantner-Instruments e.gate driver for data transmission between Gantner-Instruments e.series controller and a Lab View Application.

Parts of this manual are also available as online help in Lab View.

After selecting a VI in Block Diagram, "Strg + H" will pop up a short description including also a link to the detailed description of this manual.

3. Driver Installation

Gantner-Instruments e.gate driver can be chosen directly out of the Lab View instruments palette:
Therefore, the directory "Gantner Instruments" from the .zip file has to be copied to "C:\Program Files\National Instruments\LabView 8.5\instr.lib"

Now, after Lab View restart, Gantner-Instruments e.gate driver should be part of the Lab View instruments palette and can be found at "Instrument I/O -> Instr Drivers" in Block Diagram.

4. Driver Description

Gantner-Instruments e.series controller delivers several possibilities for data transfer over Ethernet. Detailed descriptions about getting access to Gantner-Instruments e.series controller's can be found in the manual "e.bloxx System Guide".

Brief protocol description

This driver enables data transfer over ASCII TCP/IP or HighSpeedPort TCP/IP protocol. The protocol has to be defined at the beginning and can not be mixed afterwards.

- **ASCII TCP/IP**

This is a configurable and easy to access TCP/IP interface of Gantner-Instruments e.series controllers on Port 10000. The communication is done via ASCII commands.

- **HighSpeedPort TCP/IP**

The HighSpeedPort TCP/IP protocol is a fast and fixed interface to Gantner – Instruments e.series controllers. Data reading and writing can be done within one command. Request and response are coded binary.

Types of data

Following types of data can be handled by this driver.

- **Online Data**

This is the synchronised data frame of the controller when the request is recognized by the controller. Online Data can be read and written with some VI's for ASCII and for HighSpeedPort protocol. There are VI's to read/write single channels, groups of channels or all **channels**.

- **Buffer Data**

Gantner-Instruments e.series controller store data periodically in a circular buffer, which can be defined in the controller configuration.

Every response delivers a block of data frames over all channels. The time difference of the frames is depending on the system synchronisation clock and a configurable frequency divider of the buffer.

Buffer data is just available, if the protocol type "HighSpeedPort Buffer" is chosen at the beginning of the communication.

- **State Information**














Deliver information about actually opened ports, operating states and errors. This information's are available for ASCII and for HighSpeedPort protocol.

- **Real Time Clock**

With ASCII and HighSpeedPort protocol, the system clock of Gantner-Instruments e.series controller can be read and written.








5. VI Overview

Overview of top Level VI's which can be used directly without decoding and without protocol knowledge.

Icon	Name	Description
	VI Tree	Gives a visual overview of the most important VI's.
	Initialize	Initializes communication and decoding information.
	Cleanup	Closes TCP/IP connection at the end of data transfer.
	ReadStateInfo	Reads state information from controller.
	RealTimeClock	Reads and writes the Real Time Clock of the controller.
	ASCII Read Value	Reads single channels via ASCII TCP/IP protocol with specified data type.
	ASCII Write Value	Writes single Channels via ASCII TCP/IP protocol.
	ASCII Read All Variables	Reads all channels via ASCII TCP/IP protocol.
	HighSpeedPort Online Communication	Reads and writes channels via HighspeedPort TCP/IP protocol.
	HighSpeedPort Read Buffer	Reads buffer data via HighspeedPort TCP/IP protocol.
	HighSpeedPort Get File	Reads files via HighspeedPort TCP/IP protocol from controller.
	Decode UDBF	Decodes UDBF files or streams.
	HighSpeedPort Diagnostics	Reads interface diagnostic information.

6. General steps for Communication

For every application which should communicate over Ethernet with Gantner-Instruments controllers, following general steps have to be done:

	    	
Initialisation <ul style="list-style-type: none"> Essential communication parameters (IP, Port,...) Open TCP/IP connection Read and decode configuration to be used for data decoding Filling config cluster 	Data Handling <ul style="list-style-type: none"> Read and decode data periodical or singular Using config cluster Parameters for online controlling Interpret/analyze data 	Cleanup <ul style="list-style-type: none"> Close TCP/IP connection Indicate error information

7. Detailed VI Description

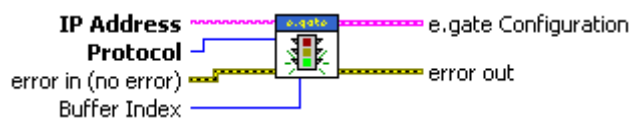








e.gate Initialize.vi

This VI initializes the TCP/IP Communication and reads and decodes Controller information to be used during data transmission.

It can be used for HighspeedPort TCP/IP and ASCII Port connection.

The e.gate Configuration output is used by every VI of the e.gate Driver.

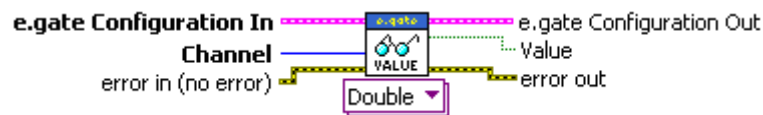


IP Address 	Defines the IP address of the controller used for TCP/IP connection.
Protocol 	Defines the data transfer protocol (ASCII TCP/IP; HighspeedPortTCP/IP online; HighspeedPortTCP/IP buffer)
Buffer Index 	Chooses the buffer of the controller
error in (no error) 	Error input
 e.gate Configuration Out	Delivers the Configuration cluster with decoding information.
 error out	Error output



e.gate ASCII Read Value.vi

This VI is used to read single channels via ASCII Port to a defined data type which can be selected with the polymorphic selector.

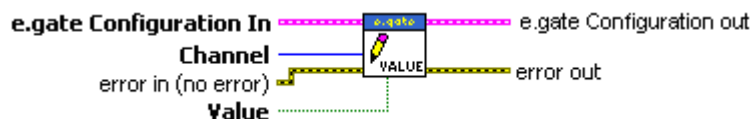


e.gate Configuration	Includes configuration data for data reading and decoding.
Channel	Defines the channel to be read from.
error in (no error)	Error input
e.gate Configuration Out	Includes configuration data for data reading and decoding. This VI will not modify anything in this cluster.
Value	Result depends on chosen data type (polymorphic selector).
error out	Error output



e.gate ASCII Write Value.vi

This VI is used to write single channels via ASCII port.








e.gate Configuration	Includes configuration data for data reading and decoding.
Channel	Defines the channel to be written.
error in (no error)	Error input
Value	Defines the value to be written to the selected channel. Every data type is supported.
e.gate Configuration Out	Includes configuration data for data reading and decoding. This VI will not modify anything in this cluster.
error out	Error output



e.gate ASCII Read All Variables.vi

This VI is used to receive the whole controller data frame via ASCII port.
The result is a double array which contains every channel as double value.



e.gate Configuration 	Includes configuration data for data reading and decoding.
error in (no error) 	Error input
 e.gate Configuration Out	Includes configuration data for data reading and decoding. This VI will not modify anything in this cluster.
 Array	The whole controller data frame stored in a double array.
 error out	Error output



e.gate HighSpeedPort Online Communication.vi

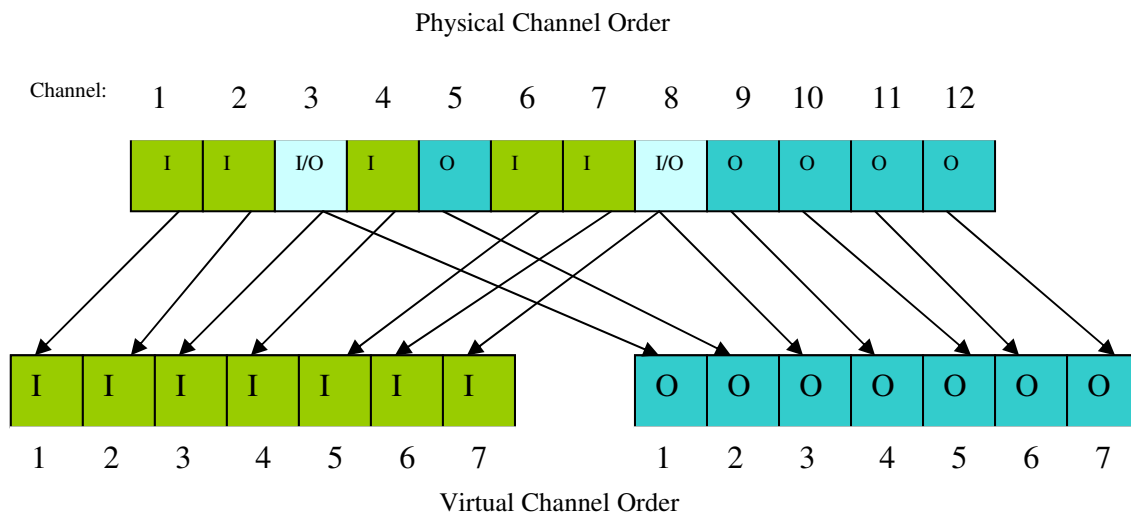
Performs HighSpeed Port TCP/IP online data transmission between Controller and PC and handles a request and the respective response within one command.

“e.gate Initialize.vi” has to be run first which delivers the Configuration Cluster.

There are several options to define which and how much data has to be read and/or written.

Every channel is stored in physical order in the channel data array of the e.gate configuration cluster independent of the data direction.

But the HighSpeedPort Online protocol differs between input and output data and that's why this driver splits the physical channel order into input and output data frame.



“ReadIndex” and “ReadCount” will access only channels in the input data frame.

“WriteIndex” and “WriteCount” will access only channels in the output data frame.

e.g.:

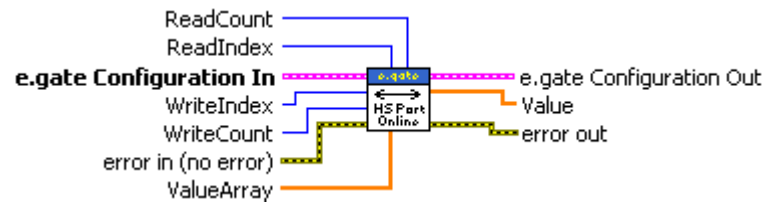
ReadIndex: 4
ReadCount: 3











Output Array:

Channel 4	Channel 6	Channel 7
-----------	-----------	-----------

Values for data writing have to be delivered in a double array.

Output Values are delivered in a double array according to "ReadIndex" and "ReadCount".



e.gate Configuration 	Includes configuration data for data reading and decoding.
ReadIndex 	"ReadIndex" defines the first channel to be read.
ReadCount 	"ReadCount" defines the number of channels to be handled.
WriteIndex 	"WriteIndex" defines the first channel to be written.
WriteCount 	"WriteCount" defines the numbers of channels to be written
ValueArray 	"ValueArray" delivers the values to be written.
error in (no error) 	Error input
 e.gate Configuration Out	Includes configuration data for data reading and decoding. This VI will not modify anything in this cluster.
 Data Out	"Data Out" delivers one frame of double values in a 1D – array.
 error out	Error output

Example for one Request – Response:

Input:

ReadIndex = 0
ReadCount = 5
WriteIndex=2
WriteCount=3
Value Array=[0,5;2;8]

Output

Data Out = [V0;V2;0,5;2;8]

E.g.: If "WriteCount" was "0", no writing would be done.

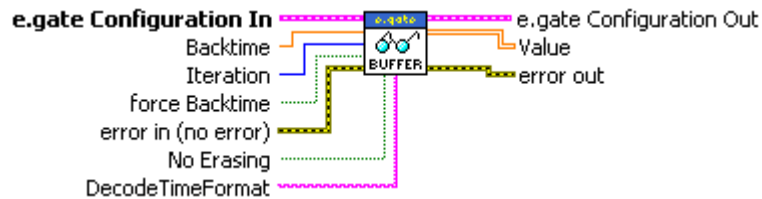




e.gate HighspeedPort Read Buffer.vi









Reads the Circular Buffer of Gantner – Instruments e.series Controller via HighspeedPort TCP/IP protocol.

e.gate Initialize.vi has to be run first which delivers the Configuration Cluster.

For every request, the VI delivers a 2-dimensional double array with a cell for every channel and a row for every sampled frame according to the input parameters which defines buffer handling.



e.gate Configuration		Includes configuration data for data reading and decoding.																																																																																																
DecodeTimeFormat		<p>Defines the Time/Date Information to be added in the output array.</p> <p>The Time Format String can contain following commands in arbitrary order and number suffixed without any space or other separating chars than “%”.</p> <table><tr><td>%TS</td><td>...</td><td>Timestamp</td></tr><tr><td>%OL</td><td>...</td><td>OLE time</td></tr><tr><td>%YYYY</td><td>...</td><td>Year</td></tr><tr><td>%MM</td><td>...</td><td>Month</td></tr><tr><td>&DD</td><td>...</td><td>Day</td></tr><tr><td>%hh</td><td>...</td><td>Hour</td></tr><tr><td>%mm</td><td>...</td><td>Minute</td></tr><tr><td>%ss</td><td>...</td><td>Second</td></tr><tr><td>%ps</td><td>...</td><td>Part of second</td></tr><tr><td>%DJ</td><td>...</td><td>Day of year</td></tr><tr><td>%DW</td><td>...</td><td>Day of week</td></tr><tr><td>%ST</td><td>...</td><td>Is summer time</td></tr></table> <p>e.g.: %YYYY%MM%DD%hh%mm%ss%ps</p> <p>Every part of the Format String will effect an additionally cell with the respective date/time information as double value.</p> <p>DecodeTimeFormat</p> <div><p>%YYYY%MM%DD%ps</p></div> <p>Table</p> <table><tr><th>Year</th><th>Month</th><th>Day</th><th>< Second</th><th>Variable_2</th><th>AI_8</th></tr><tr><td>2008,000000</td><td>8,000000</td><td>20,000000</td><td>0,586923</td><td>39680,726562</td><td>10,0000</td></tr><tr><td>2008,000000</td><td>8,000000</td><td>20,000000</td><td>0,591923</td><td>39680,726562</td><td>10,0000</td></tr><tr><td>2008,000000</td><td>8,000000</td><td>20,000000</td><td>0,596923</td><td>39680,726562</td><td>10,0000</td></tr><tr><td>2008,000000</td><td>8,000000</td><td>20,000000</td><td>0,601923</td><td>39680,726562</td><td>10,0000</td></tr><tr><td>2008,000000</td><td>8,000000</td><td>20,000000</td><td>0,606924</td><td>39680,726562</td><td>10,0000</td></tr><tr><td>2008,000000</td><td>8,000000</td><td>20,000000</td><td>0,611923</td><td>39680,726562</td><td>10,0000</td></tr><tr><td>2008,000000</td><td>8,000000</td><td>20,000000</td><td>0,616923</td><td>39680,726562</td><td>10,0000</td></tr><tr><td>2008,000000</td><td>8,000000</td><td>20,000000</td><td>0,621923</td><td>39680,726562</td><td>10,0000</td></tr><tr><td>2008,000000</td><td>8,000000</td><td>20,000000</td><td>0,626923</td><td>39680,726562</td><td>10,0000</td></tr></table> <p>Effect of DecodeTimeFormat in output array.</p>	%TS	...	Timestamp	%OL	...	OLE time	%YYYY	...	Year	%MM	...	Month	&DD	...	Day	%hh	...	Hour	%mm	...	Minute	%ss	...	Second	%ps	...	Part of second	%DJ	...	Day of year	%DW	...	Day of week	%ST	...	Is summer time	Year	Month	Day	< Second	Variable_2	AI_8	2008,000000	8,000000	20,000000	0,586923	39680,726562	10,0000	2008,000000	8,000000	20,000000	0,591923	39680,726562	10,0000	2008,000000	8,000000	20,000000	0,596923	39680,726562	10,0000	2008,000000	8,000000	20,000000	0,601923	39680,726562	10,0000	2008,000000	8,000000	20,000000	0,606924	39680,726562	10,0000	2008,000000	8,000000	20,000000	0,611923	39680,726562	10,0000	2008,000000	8,000000	20,000000	0,616923	39680,726562	10,0000	2008,000000	8,000000	20,000000	0,621923	39680,726562	10,0000	2008,000000	8,000000	20,000000	0,626923	39680,726562	10,0000
%TS	...	Timestamp																																																																																																
%OL	...	OLE time																																																																																																
%YYYY	...	Year																																																																																																
%MM	...	Month																																																																																																
&DD	...	Day																																																																																																
%hh	...	Hour																																																																																																
%mm	...	Minute																																																																																																
%ss	...	Second																																																																																																
%ps	...	Part of second																																																																																																
%DJ	...	Day of year																																																																																																
%DW	...	Day of week																																																																																																
%ST	...	Is summer time																																																																																																
Year	Month	Day	< Second	Variable_2	AI_8																																																																																													
2008,000000	8,000000	20,000000	0,586923	39680,726562	10,0000																																																																																													
2008,000000	8,000000	20,000000	0,591923	39680,726562	10,0000																																																																																													
2008,000000	8,000000	20,000000	0,596923	39680,726562	10,0000																																																																																													
2008,000000	8,000000	20,000000	0,601923	39680,726562	10,0000																																																																																													
2008,000000	8,000000	20,000000	0,606924	39680,726562	10,0000																																																																																													
2008,000000	8,000000	20,000000	0,611923	39680,726562	10,0000																																																																																													
2008,000000	8,000000	20,000000	0,616923	39680,726562	10,0000																																																																																													
2008,000000	8,000000	20,000000	0,621923	39680,726562	10,0000																																																																																													
2008,000000	8,000000	20,000000	0,626923	39680,726562	10,0000																																																																																													

Backtime 	<p>At starting data transfer, mostly the circular buffer is filled with old values depending on the buffer size and the fill rate.</p> <p>The “BackTime” defines if and how much data should be deleted before data transfer.</p> <p>Negative data e.g.: -1The buffer data, excluding one second of most actual data will be deleted before data transfer.</p> <p>0The buffer will be completely erased before data Transfer (first request).</p> <p>Positive data and “NaN”No data will be deleted and the response will contain the whole buffer data.</p>
Force Backtime 	<p>Normally, if the “Iteration input” is wired, the VI detects the first cycle and disables the “BackTime” afterwards, so that it is ensured that all data will be read.</p> <p>Although there is the possibility to use “BackTime” later with the “Force Backtime” Parameter.</p>
No Erasing 	<p>If “No Erasing” is “1”, the “read out data” remain in buffer until overrun. Nevertheless, when “Backtime” is used, the “data before Backtime” will be deleted.</p>
Iteration 	<p>Connected with a loop counter, the VI will automatically disable “Backtime” after first request. (excluding if “Force Backtime” is used)</p>
error in (no error) 	Error input
 e.gate Configuration Out	Includes configuration data for data reading and decoding. This VI will not modify anything in this cluster.
 Data Out	2D - double array with a cell for every channel and a row for every sampled frame.
 error out	Error output

ATTENTION: Don't use “Force Backtime” with older Controller Firmware then V1.15!

Example:

Backtime = -5
Force Backtime = 1
No Erasing = 1

Every Response will deliver all values since 5 seconds.
Even at lower sampling (reading from controller) then 5 seconds.






e.gate Cleanup.vi

This VI closes the open TCP/IP connection.

e.gate Configuration



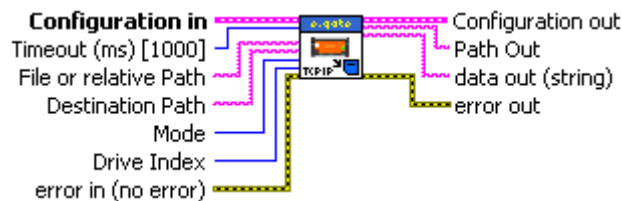
e.gate Configuration 	Includes configuration data for data reading and decoding, also connection id.
error in (no error) 	Error input
 error out	Error output














e.gate Get File Via HighSpeed Protocol.vi

This VI is used to read files and file related data from Gantner-Instruments e.series controllers.

It's a polymorphic VI with parameters like in the following image.
With the polymorphic selector, "Mode" can be chosen.



e.gate Configuration 	Includes configuration data for data reading and decoding, also connection id.
Timeout (1000) 	Defines the Timeout for special reading method.
Mode 	<p>With "Mode", the type of information to be read from controller can be defined</p> <ul style="list-style-type: none"> ReadDirectory <p>Data Out will deliver an ASCII string including the name of files from controller which are chosen with "Drive Index".</p> <p>Drive frame contains (all in ASCII-format): "DR:"\t"<DriveName>"\t"<DriveIndex>"\t"<TotalSize>"\t"<FreeSize>"\t"<UsedSize>"\t"<BadSize>"\n</p> <p>File frame contains (all in ASCII-format): "FI:"\t"<FileName>"\t"<FileSize>"\t"<DateTime>"\t"<AttrStr>"\n (AttrStr: B0=READONLY / B1=HIDDEN / B2=SYSTEM / B3=VOLUME / B4=DIR / B5=ARC)</p> <p>Full requested block delimiter is \r\n</p> <p>If an item is "-1", this is invalid. Maybe this drive is not installed or parameter is not available there !!!</p>

	<p>e.g. ReadDirectory of DriveIndex=0: \00\00\00 DR:\t[FLASH_APPLICATION]:\t0\t7077888\t4878336\t2199552\t0\n FI:\tecpcu01.bin\t1764336\t2008-03-11\t10:51:40\t0020\n FI:\tecpcu01.ini\t495\t1980-01-01\t12:00:00\t0020\n FI:\tconfig.sys\t219\t1980-01-01\t12:00:00\t0020\n FI:\tfpga.bit\t403014\t1980-01-01\t12:00:00\t0020\n FI:\tftp_lic.cnf\t538\t2008-02-26\t15:43:42\t0020\n FI:\tmaster.gcf\t10750\t2008-02-26\t15:43:50\t0020\n FI:\tself.ini\t78\t2008-03-11 09:27:36\t0020\n\r\n</p> <ul style="list-style-type: none"> ReadFileSize <p>Data Out will be a string with the size of a chosen file in ASCII-Format.</p> <ul style="list-style-type: none"> ReadFileData <p>Data Out will contain the content of a chosen file. (" e.gate Get Summary Via Highspeed Protocol can be taken as example).</p>
Drive Index 	<p>Is only used if "Mode=ReadDirectory" and defines which file types will be found. It is an enumerator with following possibilities:</p> <ul style="list-style-type: none"> All Drives Flash – Application Files Flash – Data USB Storage – Data Virtual State Virtual Circlebuffer Virtual Onlinebuffer Virtual Archive Virtual Logger
File or relative Path 	Filename or relative path to be read from controller if chosen ReadFileSize or ReadFileData.
Destination Path 	If destination path != " " and "ReadFileData" is chosen., the VI tries to save the file to Destination Path.
error in (no error) 	Error output
 e.gate Configuration	Includes configuration data for data reading and decoding. This VI will not modify anything in this cluster.
 data out (string)	Delivers data depending on "Mode" and "DriveIndex"
 Path Out	Delivers the Path of the copied file if "ReadFileData" and "Destination Path" are used.
 error out	Error output

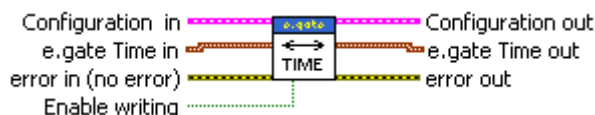


e.gate RealTimeClock.vi

Transfers date and time information between controller and pc to read out the system time or synchronize the system time with the pc.

“e.gate Initialize.vi” has to be run first which delivers the configuration cluster.

Every date/time segment for reading and writing is handled as a double value.



e.gate Configuration	Includes configuration data for data reading and decoding.
Date/Time in	Every date/time segment has to be referred as a double value.
Enable writing	Since reading and writing is done within the same command, it's important to have a possibility to block writing ("Enable writing"=0).
error in (no error)	Error input
e.gate Configuration Out	Includes configuration data for data reading and decoding. This VI will not modify anything in this cluster.
Date/Time out	Every date/time segment will be delivered as a double value.
error out	Error output

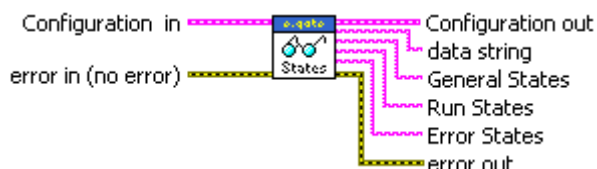


e.gate ReadeStateInfo.vi






Reads actual states via Highspeedport TCP/IP protocol from controller.

“e.gate Initialize.vi” has to be run first which delivers the Configuration Cluster.

The data is available as binary coded string and as ASCII strings.



e.gate Configuration	Includes configuration data for data reading and decoding.
error in (no error)	Error input
e.gate Configuration Out	Includes configuration data for data reading and decoding. This VI will not modify anything in this cluster.

 data string	<p>Binary data which are coded as follows:</p> <p>Byte 0-3 : General States</p> <p>B0 ... InitActive B1 ... MeasRunInActive B2 ... ConfigurationModeActive B3 ... ConfigurationStable B4 ... ForceNoHealthCheckActive</p> <p>Byte 4-7 : Run States</p> <p>B0 ... HostConfigBusRS485Active B1 ... HostConfigBusRS232Active B2 ... HostFTPActive B3 ... Reserved B4 ... HostFieldbusActive B5 ... HostDataPortActive B6 ... HostDistributorPortActive B7 ... HostHighspeedPortTCPIPAActive B8 ... HostHighspeedPortUDPActive B9 ... HostPacKernelActive B10 ... HostTransparentPortActive B11 ... HostFTPClientActive B12 ... HostMailClientActive B13 ... HostWebServerActive B14 ... MassStorageActionActive</p> <p>Byte 8-11: Error States</p> <p>B0 ... ConfigFilesError B1 ... VariableError B2 ... VariableAccessInstableError B3 ... ReducedPerformanceError B4 ... PacKernelOperationDeniedError B5 ... FieldbusConfigurationError B6 ... DistributorSyncError B7 ... SocketOverloadedError B8 ... ExtensionBoardError B9 ... ClientConnectionError B10 ... PacKernelNotSynchedError B11 ... DataFLASHFileSystemError B12 ... DataFLASHUnitCombinedError B13 ... FtpClientUnitCombinedError B14 ... MailClientUnitCombinedError B15 ... MailServerUnitCombinedError B16 ... USBHostUnitCombinedError B17 ... ExternalClockSignalMissingError B18 ... ExternalSyncSignalMissingError</p>
 General States	Delivers General States as an ASCII string.
 Run States	Delivers Run States as an ASCII string
 Error States	Delivers Error States as an ASCII string
 error out	Error output

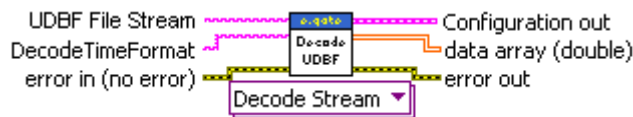


e.gate Decode UDBF.vi

This VI decodes UDBF coded files from local drive or UDBF coded streams and can be used without initialisation.

The Data is decoded to a 2D array of double values also including configurable information about date and time.

e.gate Decode UDBF.vi can be used in following two different ways:



The “Decode Stream” option enables decoding of UDBF flash file data which is written from controller and not saved to local drive (See e.gate Example Decode UDBF.vi).



With “Decode File” option, the VI uses the path of a local stored UDBF coded file for decoding data to a 2D double array.

UDBF Stream	Is a UDBF coded stream of binary data including header and data. The VI e.gate HighSpeedPort ReadFile delivers this stream if data should not be stored on a local drive (See e.gate Example Decode UDBF.vi).
UDBF File Path	Is the path to a UDBF coded File which includes header and data. (e.g.: e.reader stored flash file)
DecodeTimeFormat	See “e.gate HighSpeedPort Read Buffer.vi” for a detailed description of “DecodeTimeFormat”.
error in (no error)	Error Input
e.gate Configuration Out	Includes configuration data for decoding.
data array (double)	2D - double array with a cell for every channel and a row for every sampled frame. Also including date/time information at the beginning of every row like defined with “Decode Time Format”.
error out	Error out



e.gate HighSpeedPort Diagnostics.vi

This VI reads and decodes interface diagnostic data via HighSpeedPort.

Diagnostic data's are:

- Cycle counts for interfaces or slave modules.
- Error states
- Error counts
- Time since last request to an interface or module
-

Diagnostic data can be distinguished by themes which are similar to the physical setup.

There are internal and external interfaces present:

- Internal interfaces are virtual and system data's.
- External Interfaces are the UART's, Field Busses like PB, Can Open or EtherCAT.

To enable selection of special diagnostic data types, it contains 3 polymorphic VI's which can be chosen with the polymorphic selector.

Option "All" will deliver diagnostic information from internal and external interfaces.

Option "Internal" will deliver only information about system and virtual channels.





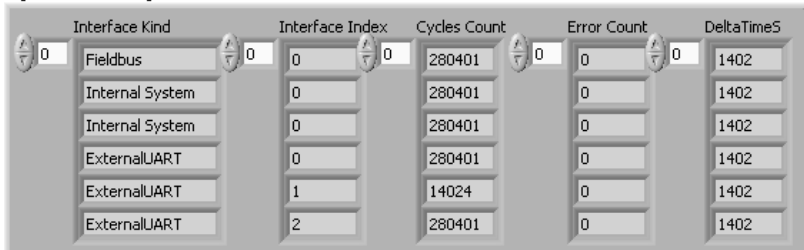

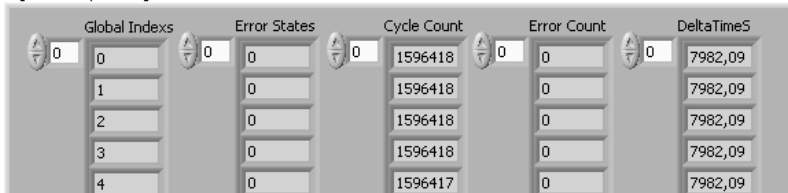
Option "External" will deliver information about UART's and Field busses.






The VI delivers output data as clusters containing arrays of interface data.

Diagnostic information are split up in themes:

- Interface diagnostics are information from the physical view. (Internal, ExternalUART, External Field Busses).
- Transport diagnostics distinguishes between slave modules and internal interface groups. (System Channels, Virtual Channels, e.bloxx modules,.).
- Variable diagnostics delivers information about every single channel in the data frame and doesn't distinguish between virtual and physical channels.

e.gate Configuration 	Includes configuration data for data reading and decoding.
error in (no error) 	Error input
 e.gate Configuration Out	Includes configuration data for data reading and decoding. This VI will not modify anything in this cluster.
 e.gate Interface Diagnostic Data	<p>Is a cluster that contains arrays with diagnostic information for every interface:</p> <p>e.gate Interface Diagnostic Data</p>  <p>Cluster on the front panel with polymorphic option ="All".</p> <p>The polymorphic selector defines the interfaces to be read.</p> <ul style="list-style-type: none"> • "Interface Kind" indicates if it is a field bus, a UART or internal interfaces. • "Interface Index" delivers an index for the interface. E.g.: UART index 0-3 • "Cycles Count" indicates the number of cycles for this interface since last request. • "Error Count" indicates the number of errors on this interface since last request. • "DeltaTimeS" indicates the number of seconds from the last request to now.
 e.gate Transport Diagnostic Data	<p>Is a cluster that contains diagnostic information for groups of internal interfaces and slave modules:</p> <p>e.gate Transport Diagnostic Data</p>  <p>Cluster on the front panel with polymorphic option="All".</p> <p>If polymorphic option="All", the first two transport entries are "System Channels" and "Virtual Channels" followed by the slave modules on the UART's.</p> <ul style="list-style-type: none"> • "Global Index" is the continuous index of every transport entry. • "Error States" is a bit set containing error codes coded as follows: Bit0...Transfer Error • "Cycle Count" is the number of cycles at this module/channel group since last request. • "Error Count" is the number of errors since last request. • "DataTimeS" indicates the number of seconds from the last request to now.

 e.gate Variable Diagnostic Data	<p>Is a cluster that contains diagnostic information for every channel in the data frame (if polymorphic option="All", otherwise only physical or virtual channels)</p> <p>e.gate Variable Diagnostic Data</p>  <ul style="list-style-type: none"> • "Global Index" is the continuous index of every channel. • "Error States" is a bit set containing error codes coded as follows: <ul style="list-style-type: none"> B0...ConversionCalculation B1...RangeMin B2...RangeMax
 error out	Error out

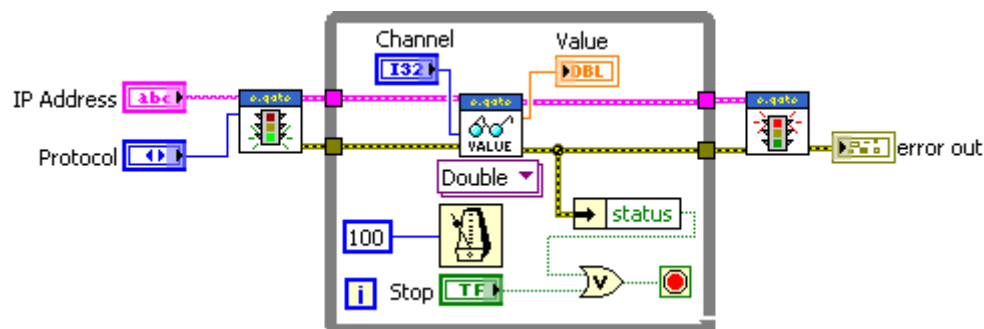
8. Examples

Following example VI's demonstrate the usage of Gantner Instruments Lab View driver VI's for various data transfer possibilities.

All example VI's can be found in the Lab View driver menu for Gantner-Instruments e.gate.

e.gate Example - ASCII Read Value.vi

This Example demonstrates how to read channels over ASCII port, using e.gate ASCII Read Value.vi.



E.gate Initialize.vi uses the correct IP Address of the controller and Protocol Type "ASCII" to open the ASCII TCP/IP Connection.

The TCP/IP connection is identified with a connection ID (RefNum), an essential part of the configuration cluster which has to be connected to every driver VI.

In a loop, the defined channel value is being read with e.gate Read ASCII Value and written to the Value Indicator.

The cycle time of the loop is set with a wait timer.

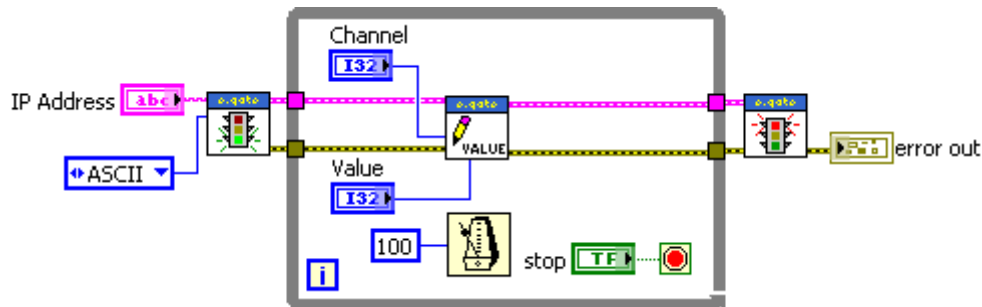
The error cluster is connected with every driver VI to guarantee an assorted data flow.

In this example, the loop will end if an error occurs during data reading or if the stop button is pressed.

At the end, e.gate Cleanup closes the TCP/IP Connection and the error cluster is indicated.

e.gate Example – ASCII Write Value.vi

This Example demonstrates how to write channels over ASCII port, using “e.gate ASCII WriteValue.vi”.

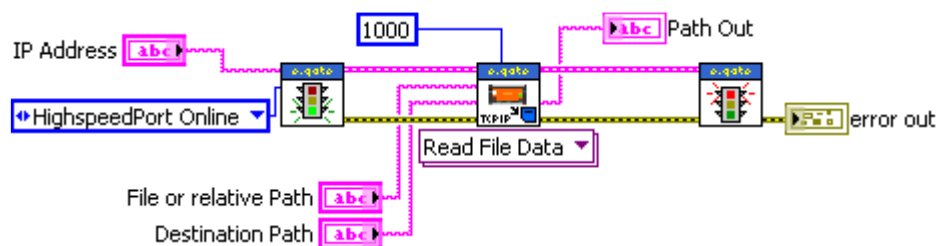


This example is the writing equivalent to e.gate ASCII Read Value.

The channel and its value to be written must be set to write values to a specified channel.

e.gate Example - HighSpeedPort Get File.vi

This Example demonstrates how to read files and file related information from Gantner-Instruments e.series controllers via HighSpeedPort TCP/IP, using e.gate HighSpeedPort Get File.vi

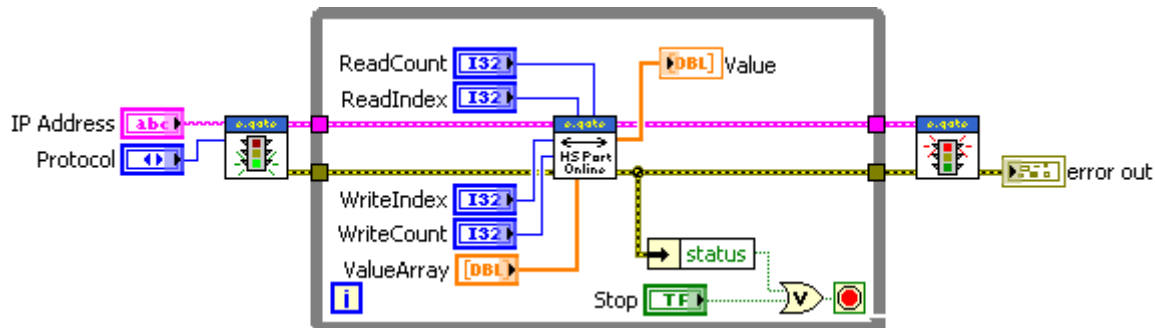


This example only works, if HighSpeedPort Protocol is chosen for the initialisation.

If the option “Read File Data” is chosen for “e.gate HighSpeedPort Get File.vi”, it will save the controller file that is chosen with “File or relative Path” to the destination path.

e.gate Example - HighSpeedPort Online Communication.vi

This Example demonstrates how to use e.gate HighSpeedPort Online Communication.vi for online data reading and writing.

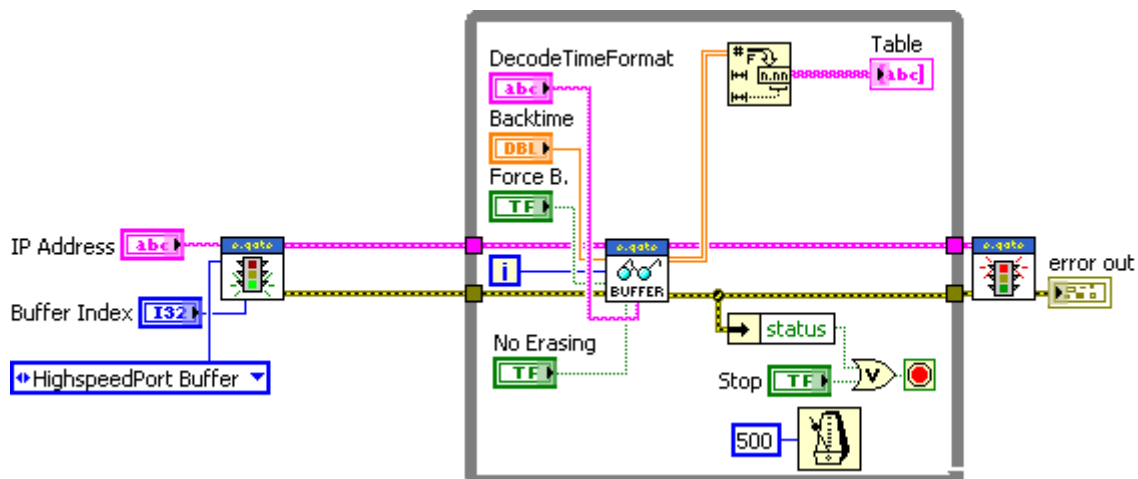


The protocol has to be HighSpeedPort Online that decoding data are read out of the controller configuration file #summary.sta.

For every cycle the data frame of the controller is read and written depending on the input parameters (see detailed description of “e.gate HighSpeedPort Online Communication.vi”). The output values are delivered in a 1D array of decoded double values, which represents the chosen data frame.

e.gate Example - HighSpeedPort Read Buffer.vi

This Example demonstrates how to read and handle Circular buffer data using e.gate HighSpeedPort Read Buffer.vi



The protocol has to be HighSpeedPort Buffer that decoding data is read as a binary stream of state and header data.

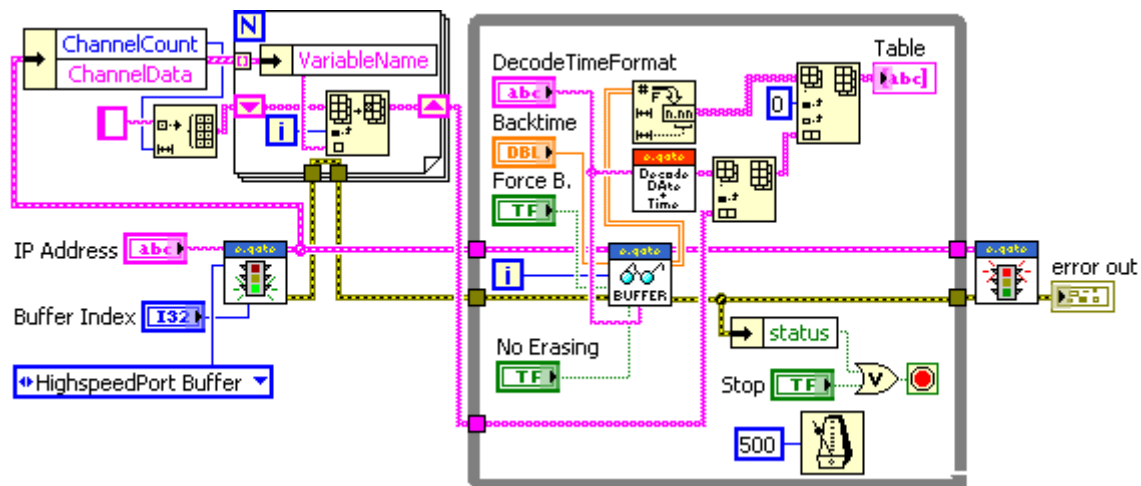
The buffer index has to be set before initializing, because the header data which describes this buffer is decoded during initialization.

The loop iteration counter is connected with the VI, that the first access can be recognized to control and ensure successive data reading.

The “DecodeTimeFormat” can be changed online to modify output date and time information.

See detailed description of “e.gate HighSpeedPort Read Buffer” how to use and combine input parameters.

Additional functionality to add channel names to the table.



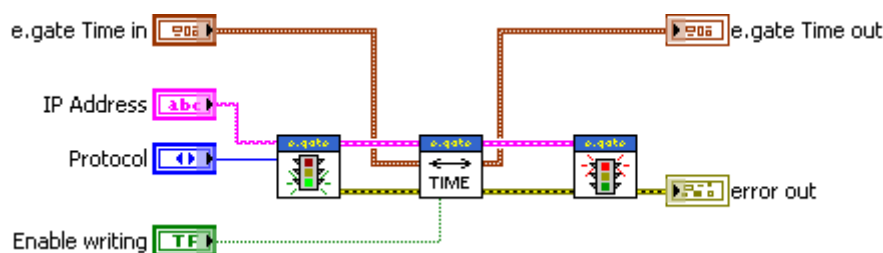
The first example only delivers the data frame without channel information to the output table. This modification adds channel names and date/time descriptions to the top of the table.

Channel Count and Channel Data are extracted from the configuration cluster to fill the first line of the array with variable names before starting the loop.

In the loop, Date/Time descriptions are added to the first line for every cycle (online modification of "TimeFormatString"). Then the data from the VI is appended to the following lines of the table.

e.gate Example - Real Time Clock.vi

This example demonstrates the handling of the Real Time Clock over ASCII and HighSpeedPort using e.gate RealTimeClock.vi



The VI "e.gate Real Time Clock" detects the connection protocol and uses the right time/date handling routines automatically.

This enables universal usage for ASCII and HighSpeedPort.

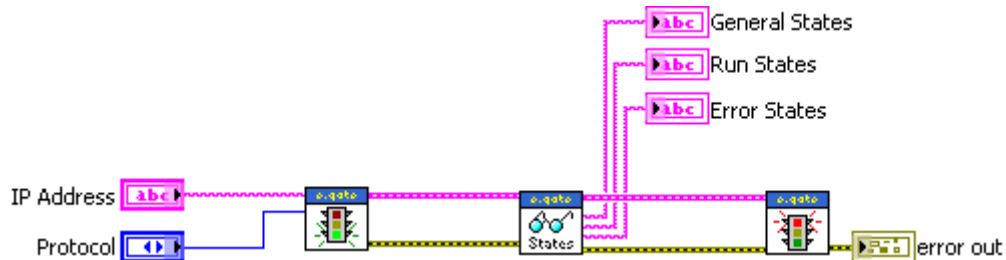
The only difference in data is the absence of milliseconds for ASCII connection.

The Date/Time elements are handled in a cluster of double values to maximize clarity.

"Enable writing" is used to control if "e.gate Time in" (e.g.: the pc time) is written to the controller.

e.gate Exemple - Device Information.vi

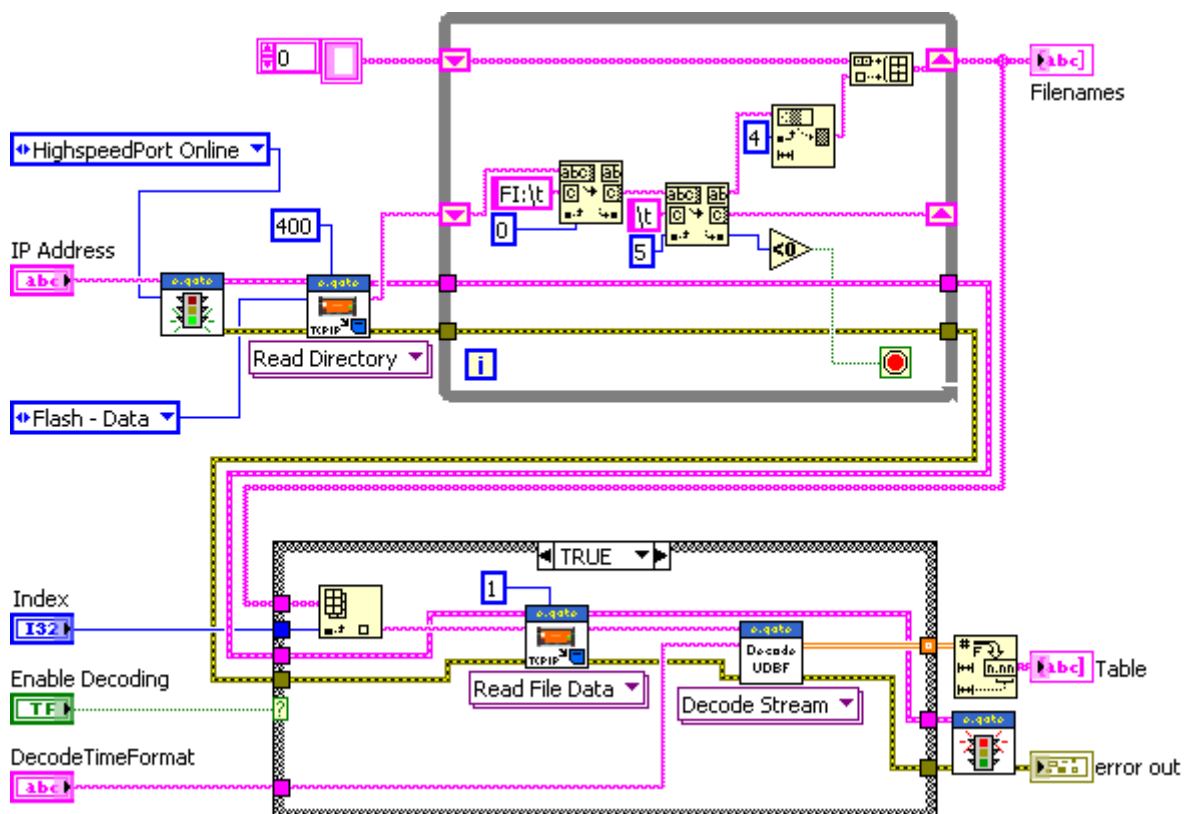
This example demonstrates how to read device information via ASCII and HighSpeedPort using “e.gate ReadStateInfo.vi”.



The VI “e.gate Device Information” detects the connection protocol and uses the right decoding routines automatically.
This enables universal usage for ASCII and HighSpeedPort.

e.gate Example – Decode UDBF.vi

This example demonstrates how to decode UDBF data (e.g.: e.reader flash files) using e.gate Decode UDBF.vi



The data handling of this VI can be split up into two parts:

- **Reading the directory**

The VI “HighSpeedPort Get File” can be used to read out the filenames on the controller.

The drive index “Flash – Data” defines that only flash data files will be indicated in a string (see detailed description of “e.gate HighSpeedPort Get File” how the output string is built up).

The filenames are decoded and written to an array.

- **Reading and decoding the file**

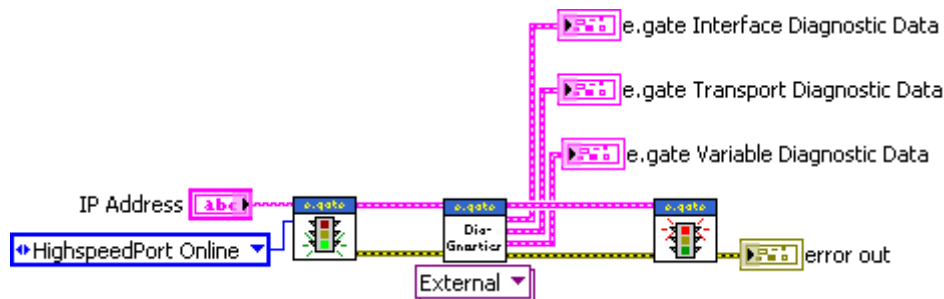
Depending on the index and if decoding is enabled, the file is also read with the VI “e.gate HighSpeedPort Get File” but with option “Read File Data”.

Because there is no destination path defined, the file isn’t written to the hard drive but the binary data is delivered as a string which is decoded by the VI “e.gate Decode UDBF”

The result of the decoding will be the same than reading out the buffer of a controller, also date/time information can be added with “DecodeTimeFormat”.

e.gate Example – HighSpeedPort Read Diagnostics.vi

This example demonstrates how to read diagnostic information from Gantner – Instruments controller.











9. Core VI Overview

These VI's are basic parts of all Top Level VI's.

For special applications that are not able to build up with User Top Level VI's, these VI's can be used.

Also some of them have special decoding functions which can be very useful.

Icon	Name	Description
	ASCII Error Handler	Analyzes the ASCII query response if it is "ACK" or throws an error if "NACK".
	ASCII Query	Completes and transmits ASCII Commands between controller and pc.
	ASCII Write Value - CORE	Uses "ASCII Error Handler" and "ASCII Query" to write data to the controller.
	ASCII Read Value - CORE	This is the basic VI to read single channels without data type dependences.
	Decode Summary File	This VI decodes the file #summary.sta and stores the information in the configuration cluster.
	HighSpeedPort Build Online Command	This VI codes logical positions and values in the data frame into its byte representations.
	HighSpeedPort Check Command Supported	This VI checks, if new HighSpeedPort Buffer commands are supported by the controller.
	HighSpeedPort Code Data Type	This VI is used to code a numeric value to a binary array depending on the channel configuration it should be written to. Also some other information's are delivered
	HighSpeedPort Decode Buffer Config Header Data	This VI is used to decode Buffer Config Header Data from a binary array to the "Configuration" cluster.
	HighSpeedPort Decode Buffer Variable Header Data	This VI is used to decode Buffer Variable Header Data to an array of "Channel Data" clusters.
	HighSpeedPort Decode Buffer	This VI is used to decode a binary stream of UDBF coded data depending on the decoding information of the header.
	HighSpeedPort Decode Data Type	This VI is used to decode a binary array into a numeric variable depending on decoding information out of the configuration cluster.
	HighSpeedPort Decode TimeStamp	This VI is used to decode the timestamp depending on time information from buffer header. An output data frame gets prefilled with date/time information depending on the "TimeFormat"string".
	HighSpeedPort Get Summary	This VI is used to receive and store the file #summary.sta from the controller.
	HighSpeedPort Online Communication - CORE	This VI handles the HighSpeed Online Communication.

	HighSpeedPort Query - Base	This VI is used to code every HighSpeedPort command into a binary stream.
	HighSpeedPort Query	This VI is responsible for the HighSpeed TCP/IP data transmission depending on some special conditions.
	HighSpeedPort Read Buffer - CORE	This VI is used to read the circular buffer to a binary array.
	HighSpeedPort Read Buffer Header	This VI is used to read the circular buffer header to a binary array.
	HighSpeedPort Read Buffer State	This VI is used to read the circular buffer state information to a binary array.
	Time Format Header	This VI is used to decode the "TimeFormatString" to an array of its real names.
	HighSpeedPort Decode Diagnostic Data	This VI is used to decode a binary array into diagnostic information for modules and variables.
	HighSpeedPort Decode Interface Diagnostic	This VI is used to decode a binary array into diagnostic information for interfaces.

10. e.gate Type Descriptions

Gantner-Instruments e.gate Lab View Driver uses clusters to concentrate data to user defined data types for better structuring and clarity.



e.gate Channel Data

This cluster contains all for data decoding relevant channel information.

Name	Type	Description
VariableName	Text String	Name of the channel as text.
DataDirection	Text String	Data direction of the channel as text.
DataType	Text String	Data type of the channel as text.
DataTypeIndex	Int16	Data type of the channel coded as index.
Precision	Int16	Precision for integer channels.
FieldLength	Int16	The maximum field length of this value.
ByteCount	Int32	The number of bytes in the binary data frame.
Format	Text String	The format of this value as text including field length and precision.
Unit	Text String	The unit of this channel as text.
InputAccessOffset	Int32	The offset of this channel in the binary input data frame.
OutputAccessOffset	Int32	The offset of this channel in the binary output data frame.
InputAccessIndex	Int32	The input index of this channel.
OutputAccessIndex	Int32	The output index of this channel.

This information's are necessary to decode values from the binary stream which is delivered by the HighSpeedPort protocol.



e.gate Configuration

This cluster contains all for decoding relevant configuration and channel information.

Name	Type	Description
Connection ID	Refnum	This is the identifier of the TCP/IP connection to be used in every VI which uses TCP/IP sockets.
IPAddress	Text String	The correct IP address of the controller.
BufferIndex	Int32	The index of the buffer to be read from controller.
State Message	Text String	State information as text.
State Code	Byte	State information coded as number.
HeaderSize	Int32	The size of the buffer header.
BufferSize	Int32	The buffer size at reading buffer state information.
IsBigEndian	Int32	Indicates if binary data have Intel or Motorola format.
Version	Int32	Is used to detect which information can be decoded.
ChannelCount	Int32	The number of channels.
FrameLength	Int32	The number of bytes of one data frame.
StartTime	Double	The absolute time of the first stored frame in buffer.
dActTimeToSecondFactor	Double	Is used to convert the actual time into seconds after "StartTime" depending on the type of the timestamp.
StartTimeToDayFactor	Double	Is used to convert the StartTime into OLE time format. (Days after 01.01.1900)
BufferFillRate	Double	Indicates how much data is stored to the buffer (/sec).
SampleRate	Double	This is the configured sample rate of the controller.
ChannelData	Array-e.gate Channel Data	This is an array with the "e.gate Channel Data" of every channel in the data frame.
DecodeTimeFormat	Text String	This string defines the date/time information for the output array with HighSpeed Buffer transfer.
SummaryPath	TextString	Stores the path to the #summary.sta file.
WithChecksum	Int32	Indicates if binary data have a checksum appended.
dActTimeDataType	Int32	This index represents the data type of the timestamp.
MaxWriteCount	Int32	Is used to avoid errors with wrong indexes.
MaxReadCount	Int32	
TempOffsetRead	Int32	Is used to find the right decoding information of this byte index for the output data frame.
TempLengthRead	Int32	
Protocol	Protocol Type	Stores the protocol type which is used in every VI.
IsSupported	Int32	Indicates if the controller supports new commands.
InputAccessList	Array - Int32	References from the input index to the main index of every input channel.
OutputAccessList	Array - Int32	References from the output index to the main index of every channel.

e.gate Time

This cluster is used to concentrate time information for better clarity.

Name	Type	Description
Year	Double	Year as double value.
Month	Double	Month as double value.
Day	Double	Day as double value.
Hour	Double	Hour as double value.
Minute	Double	Minute as double value.
Second	Double	Second as double value.
Millisecond	Double	Millisecond as double value.

e.gate Diagnostic Interface Data

This cluster contains arrays of Interface Diagnostic data.

Name	Type	Description
Interface Kind	Text String	The interface type
Interface Index	Int16	The interface index if there are comparable interfaces.
Cycles Count	Int32	The number of cycles for this interface since last request.
Error Count	Int32	The number of errors for this interface since last request.
DeltaTimeS	Double	The time in seconds since last request.

e.gate Diagnostic Transport Data

This cluster contains arrays of module and/or internal channel type diagnostic data.

Name	Type	Description
Global Index	Int16	The continuous index for every entity.
Error States	Int16	Error states coded as bit set (see detailed VI description).
Cycles Count	Int32	The number of cycles for this entity since last request.
Error Count	Int32	The number of errors for this entity since last request.
DeltaTimeS	Double	The time in seconds since last request.

e.gate Diagnostic Variable Data

This cluster contains arrays of channel diagnostic data.

Name	Type	Description
Global Index	Int16	The continuous index for every channel
Error States	Int16	Error states coded as bit set (see detailed VI description).

e.gate Protocol Type

e.gate Protocol Type is an enumerator to store the protocol type.

11. Error Codes

The Range for Gantner – Instruments Lab View driver error codes is 6000-7000.

Following error codes indicate the validity of HighSpeedPort commands and will be delivered, if there is an error, at any HighSpeedPort data transfer.

To check where the error is thrown, every VI has special error codes for the same error.

- **command error**

If the binary request frame of HighSpeedPort contains an unknown command.

- **decoding error**

If the command is correct but the request frame is not valid for this command.

- **response not defined**

No valid return code received from controller.

All other error codes in the specified range indicate logical errors if the command was ok, but the result is not acceptable.

Code	Source	Description
6000	e.gate HighSpeedPort Get File – Read FileData.vi	file not existing
6001	e.gate HighSpeedPort Get File – Read FileData.vi	command error
6002	e.gate HighSpeedPort Get File – Read FileData.vi	decoding error
6003	e.gate HighSpeedPort Get File – Read FileData.vi	response not defined
6004	e.gate HighSpeedPort Get File – Read Directory.vi	command error
6005	e.gate HighSpeedPort Get File – Read Directory.vi	decoding error
6006	e.gate HighSpeedPort Get File – Read Directory.vi	response not defined
6007	e.gate HighSpeedPort Get File – Read FileSize.vi	command error
6008	e.gate HighSpeedPort Get File – Read FileSize.vi	decoding error
6009	e.gate HighSpeedPort Get File – Read FileSize.vi	response not defined
6010	e.gate HighSpeedPort Read Buffer State.vi	command error
6011	e.gate HighSpeedPort Read Buffer State.vi	decoding error
6012	e.gate HighSpeedPort Read Buffer State.vi	response not defined
6020	e.gate HighSpeedPort Read Buffer Header.vi	wrong number of bytes received
6030	e.gate ASCII Read All Variables.vi	wrong protocol type
6040	e.gate ASCII Error Handler.vi	response “NACK”
6050	e.gate ASCII Read Value – CORE.vi	wrong protocol type
6060	e.gate ASCII Write Value – CORE.vi	wrong protocol type
6070	e.gate HighSpeedPortDiagnostics – Full.vi	command error
6071	e.gate HighSpeedPortDiagnostics – Full.vi	decoding error
6072	e.gate HighSpeedPortDiagnostics – Full.vi	response not defined
6073	e.gate HighSpeedPortDiagnostics – Internal.vi	command error
6074	e.gate HighSpeedPortDiagnostics – Internal.vi	decoding error
6075	e.gate HighSpeedPortDiagnostics – Internal.vi	response not defined
6076	e.gate HighSpeedPortDiagnostics – External.vi	command error
6077	e.gate HighSpeedPortDiagnostics – External.vi	decoding error
6078	e.gate HighSpeedPortDiagnostics – External.vi	response not defined
6080	e.gate HighSpeedPort Online Communication – CORE.vi	command error
6081	e.gate HighSpeedPort Online Communication – CORE.vi	decoding error
6082	e.gate HighSpeedPort Online Communication – CORE.vi	response not defined
6090	e.gate HighSpeedPort Read Buffer.vi	command error
6091	e.gate HighSpeedPort Read Buffer.vi	decoding error
6092	e.gate HighSpeedPort Read Buffer.vi	response not defined
6093	e.gate HighSpeedPort Read Buffer.vi	received byte count !=ToTransferSize
6100	e.gate ReadStateInfo.vi	command error
6101	e.gate ReadStateInfo.vi	decoding error
6102	e.gate ReadStateInfo.vi	response not defined
6110	e.gate RealTimeClock.vi	command error
6111	e.gate RealTimeClock.vi	decoding error
6112	e.gate RealTimeClock.vi	response not defined
6120	e.gate HighSpeedPort Query – BASE.vi	Protocol not supported

