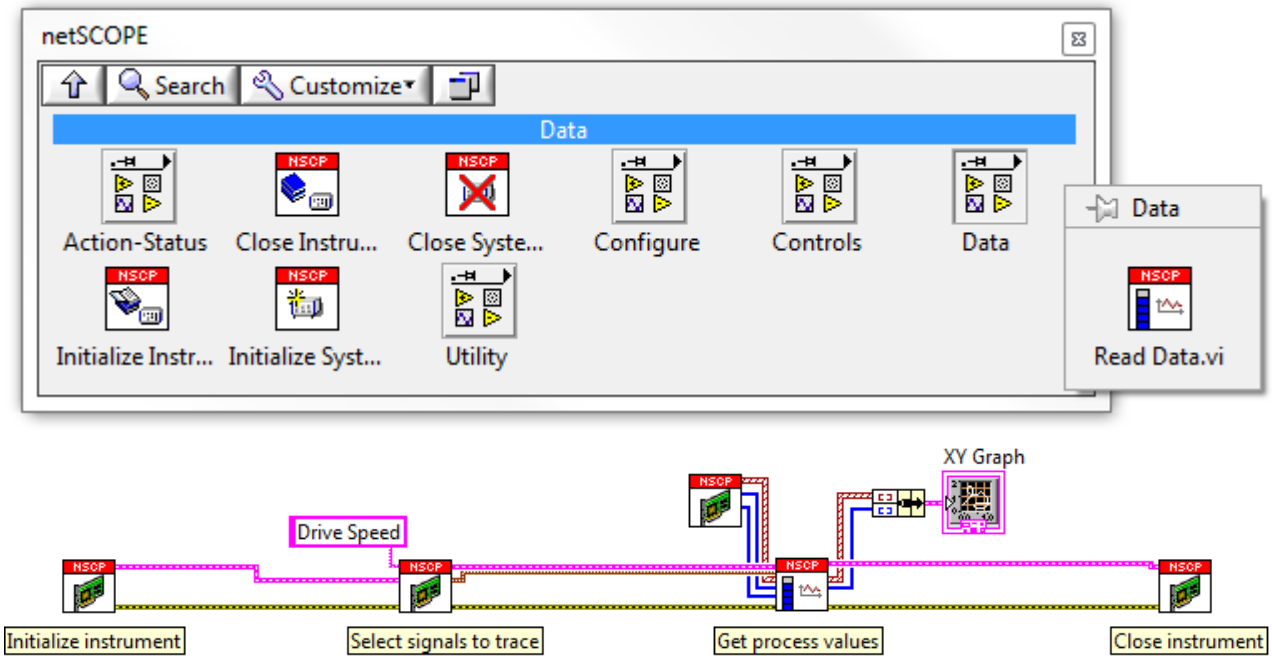
Operating Instruction Manual

netSCOPE

Instrument Driver for LabVIEW



Hilscher Gesellschaft für Systemautomation mbH

www.hilscher.com

DOC131005OI01EN | Revision 1 | English | 2013-11 | In Development | Internal

Table of Contents

1 Introduction

1.1 About this Manual

This manual provides to you descriptions about the netSCOPE instrument driver in LabVIEW.

For the netSCOPE data processing in LabVIEW you only need to perform a view programming steps.

1.1.1 Online Help

The netSCOPE VIs in LabVIEW contains an integrated online help facility.

- To open the online help, click on **Help** or press **F1**.

1.1.2 List of Revisions

Index	Date	Version	Component	Chapter	Revision
01	13-11-12	netSCOPE for LabVIEW Instrument Driver	1.0.x.x	All	Created

Table 1: List of Revisions

1.1.3 Conventions in this Manual

Notes, operation instructions and results of operation steps are marked as follows:

Notes

Important: <important note>

Note: <note>

<note, where to find further information>

Operation Instructions

1. <instruction>
2. <instruction>

or

➢ <instruction>

Results

⇘ <result>

1.2 Legal Notes

1.2.1 Copyright

1.2.2 Important Notes

The user manual, accompanying texts and the documentation were created for the use of the products by qualified experts, however, errors cannot be ruled out. For this reason, no guarantee can be made and neither juristic responsibility for erroneous information nor any liability can be assumed. Descriptions, accompanying texts and documentation included in the user manual do not present a guarantee nor any information about proper use as stipulated in the contract or a warranted feature. It cannot be ruled out that the user manual, the accompanying texts and the documentation do not correspond exactly to the described features, standards or other data of the delivered product. No warranty or guarantee regarding the correctness or accuracy of the information is assumed.

We reserve the right to change our products and their specification as well as related user manuals, accompanying texts and documentation at all times and without advance notice, without obligation to report the change. Changes will be included in future manuals and do not constitute any obligations. There is no entitlement to revisions of delivered documents. The manual delivered with the product applies.

Hilscher Gesellschaft für Systemautomation mbH is not liable under any circumstances for direct, indirect, incidental or follow-on damage or loss of earnings resulting from the use of the information contained in this publication.

1.2.3 Exclusion of Liability

The software was produced and tested with utmost care by Hilscher Gesellschaft für Systemautomation mbH and is made available as is. No warranty can be assumed for the performance and flawlessness of the software for all usage conditions and cases and for the results produced when utilized by the user. Liability for any damages that may result from the use of the hardware or software or related documents, is limited to cases of intent or grossly negligent violation of significant contractual obligations. Indemnity claims for the violation of significant contractual obligations are limited to damages that are foreseeable and typical for this type of contract.

It is strictly prohibited to use the software in the following areas:

- for military purposes or in weapon systems;
- for the design, construction, maintenance or operation of nuclear facilities;
- in air traffic control systems, air traffic or air traffic communication systems;
- in life support systems;
- in systems in which failures in the software could lead to personal injury or injuries leading to death.

We inform you that the software was not developed for use in dangerous environments requiring fail-proof control mechanisms. Use of the software in such an environment occurs at your own risk. No liability is assumed for damages or losses due to unauthorized use.

1.2.4 Warranty

Although the hardware and software was developed with utmost care and tested intensively, Hilscher Gesellschaft für Systemautomation mbH does not guarantee its suitability for any purpose not confirmed in writing. It cannot be guaranteed that the hardware and software will meet your requirements, that the use of the software operates without interruption and that the software is free of errors. No guarantee is made regarding infringements, violations of patents, rights of ownership or the freedom from interference by third parties. No additional guarantees or assurances are made regarding marketability, freedom of defect of title, integration or usability for certain purposes unless they are required in accordance with the law and cannot be limited. Warranty claims are limited to the right to claim rectification.

1.2.5 Export Regulations

The delivered product (including the technical data) is subject to export or import laws as well as the associated regulations of different counters, in particular those of Germany and the USA. The software may not be exported to countries where this is prohibited by the United States Export Administration Act and its additional provisions. You are obligated to comply with the regulations at your personal responsibility. We wish to inform you that you may require permission from state authorities to export, re-export or import the product.

1.2.6 Registered Trademarks

Windows® XP, Windows® Vista, Windows® 7 and Windows® 8 are registered trademarks of Microsoft Corporation.

EtherCAT® is a registered trademark of Beckhoff Automation GmbH, Verl, Germany, formerly Elektro Beckhoff GmbH.

LabVIEW is a graphical programming system from National Instruments.

All other mentioned trademarks are property of their respective legal owners.

2 Overview

2.1 About netSCOPE for LabVIEW

netSCOPE uses LabVIEW as software frontend. The netSCOPE device is delivered with the LabVIEW instrument driver interface.

The netSCOPE device gets process data from the automation network and provides the process data to LabVIEW. Users can program their application in LabVIEW. In LabVIEW

1. first, the netSCOPE data recording card is initialized,
2. then the signals to be detected are parameterized,
3. then, the process values can be recorded and processed in LabVIEW.
4. After the measurement is complete, the netSCOPE data recording card is closed.

Use Cases

- Machine condition monitoring / visualization

The netSCOPE device acquires process data

The user implements condition monitoring tasks and visualization in LabVIEW.

- Process documentation

netSCOPE device acquires process data

User implements documentation tasks and database connection in LabVIEW.

Generic Variable Definition

Note: If possible avoid to use specific variable definitions. Instead use generic variable definitions. This allows you to reuse the variable definitions for other systems.

2.2 netSCOPE System Data Flow

The netSCOPE for LabVIEW instrument driver supports process data recording with multiple netSCOPE data acquisition cards at the same time. The process data captured from the network is stored in an individual ring buffer of the PC. Depending on the user configuration, the ring buffer is being created either in the main memory (RAM) or on the hard drive (HDD). The ring buffer data is then being converted to be displayed in LabVIEW using the "Get Data.vi" function. In LabVIEW the acquisition data can be shown in a diagram or histogram, for example.

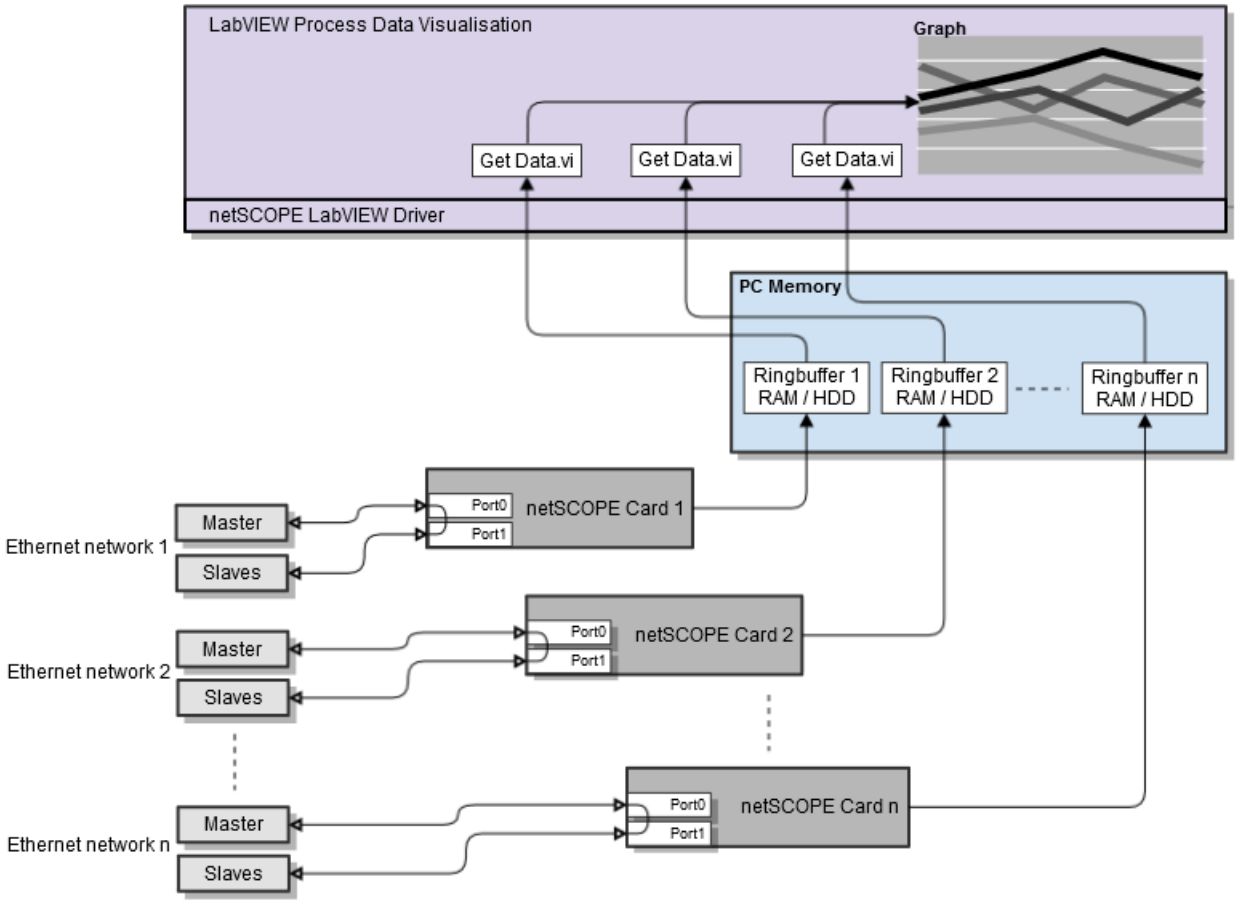


Figure 1: netSCOPE System Data Flow

3 Instrument Driver for LabVIEW

3.1 Opening LabVIEW, netSCOPE.lvlib and VI

➢ Open **LabVIEW**.

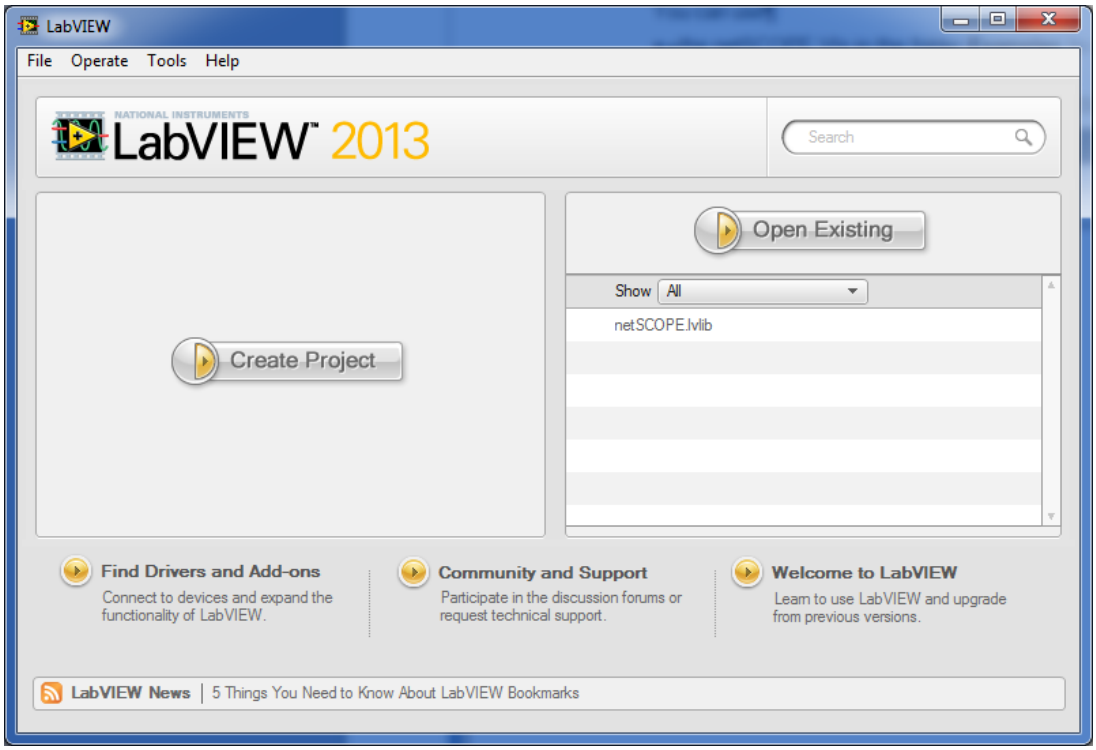


Figure 2: LabVIEW Start Screen

➢ Select **netSCOPE.lvlib**.

➢ The **netSCOPE.lvlib on Main Application Instance / Items** window is displayed.

➢ Select the **Items** tab.

➢ Select **netSCOPE.lvlib**.

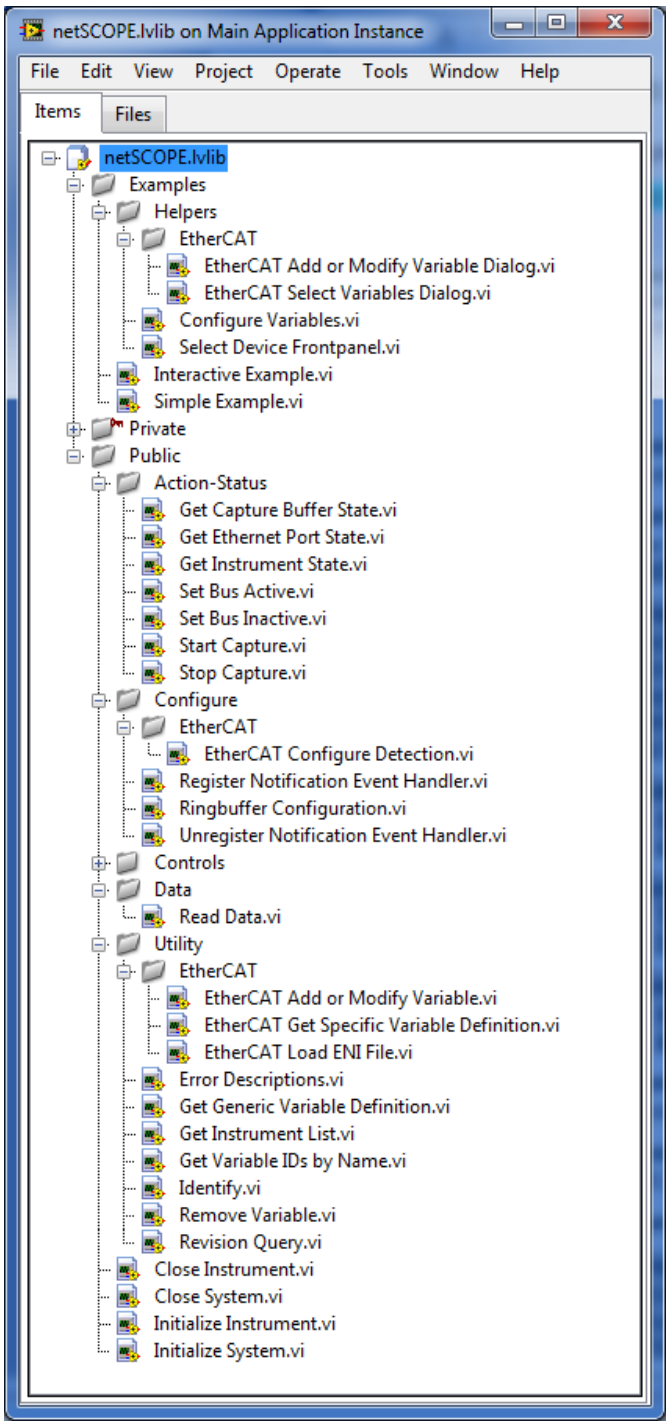


Figure 3: LabVIEW netSCOPE.lvlib on Main Application Instance / Items Pane

➢ Double click to the **VI** you need (e. g. **Interactive Example.vi**).

You can use

- the netSCOPE VIs in the folder **Examples** to understand how to create the netSCOPE programming in LabVIEW.
- The VIs in the folder **Public** to create your netSCOPE programming.

➢ The **Front Panel** view of the corresponding VI is opened (e. g. **Interactive Example.vi**, see section *netSCOPE.lvlib:Interactive Example.vi* on page 13).

3.2 Examples

3.2.1 netSCOPE.lvlib:Interactive Example.vi

The **netSCOPE.lvlib:Interactive Example.vi** example shows how to import variables from ENI file and how to add a variable manually or to edit a value and respectively how to visualize the resulting data.

Figure 4: netSCOPE.lvlib:Interactive Example.vi

3.2.1.1 Open Front Panel, Select Interface, Select Device Frontpanel

In the LabVIEW **netSCOPE.lvlib on Main Application Instance / Items** pane:

- Select the **Items** tab > **netSCOPE.lvlib** > **Examples**.
- Double click to **Interactive Example.vi**.
- The **Front Panel** view of the **netSCOPE.lvlib:Interactive Example.vi** is opened.

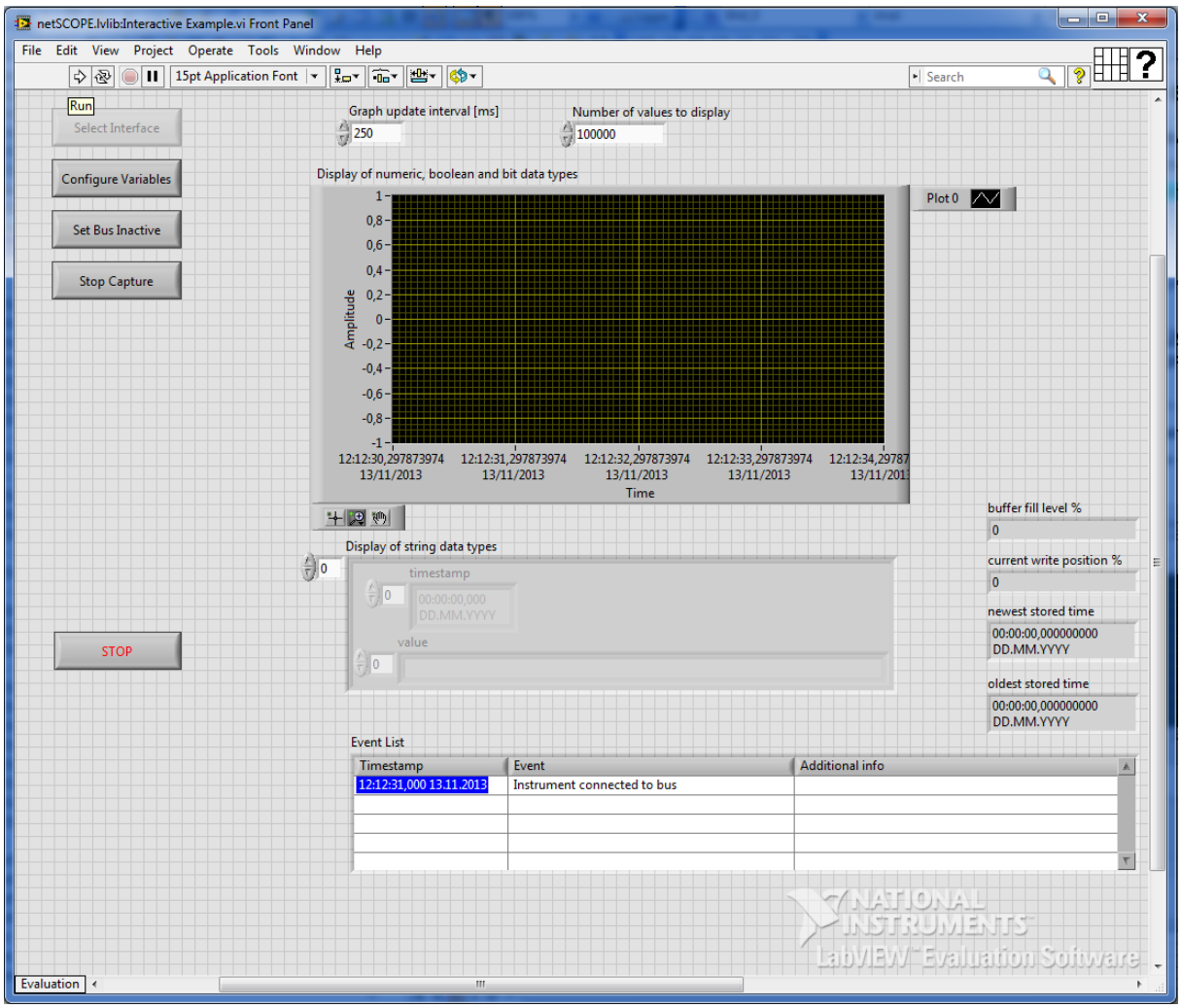


Figure 5: netSCOPE.lvlib:Interactive Example.vi - Front Panel

- Click **Run**.

➢ **Select Interface** is enabled.

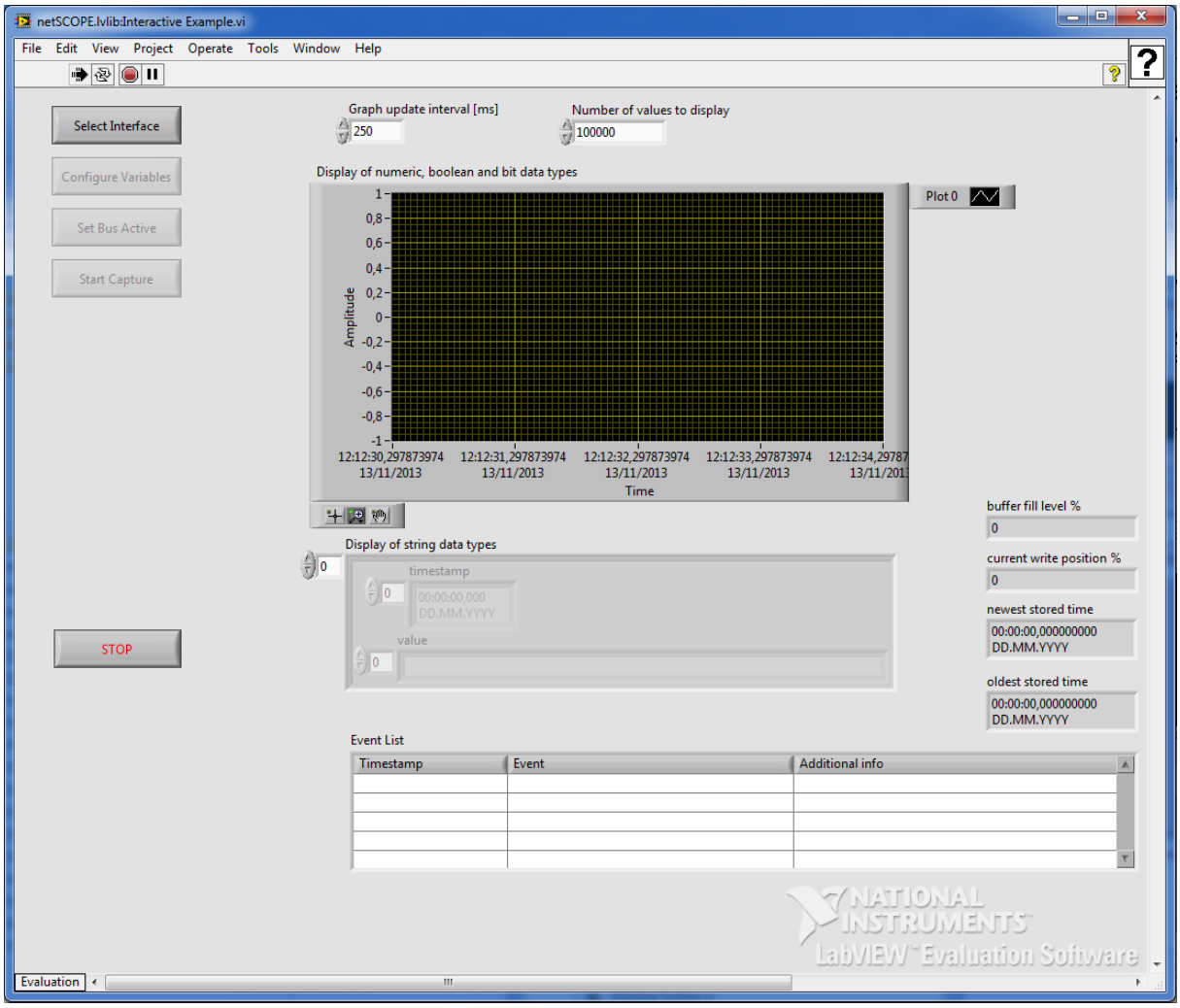


Figure 6: netSCOPE.lvlib:Interactive Example.vi - Front Panel

➢ Click on **Select Interface**.

➢ The **Select Device Frontpanel** pane is opened:

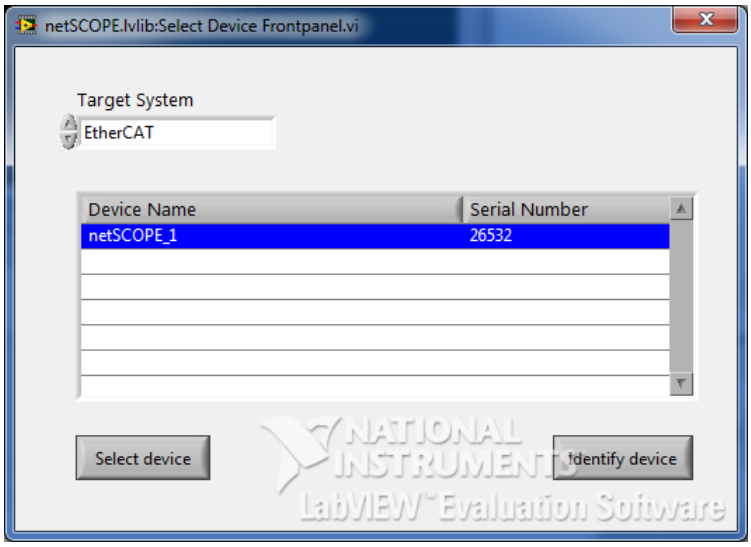


Figure 7: netSCOPE.lvlib: Interactive Example.vi - Select Device Frontpanel.vi

In the **Select Device Frontpanel** pane:

- Select the **Target System**: “EtherCAT”.
- Click **Identify device**, to identify your device (optionally).
- The STA0 and the STA1 LED at the netSCOPE data acquisition card blink green for approx 10 sec.
- Click **Select device** and select your device.

➧ The **Select Device Frontpanel** pane is closed.

3.2.1.2 Configure Variables

➧ **Configure Variables** and **Set Bus Active** are enabled.

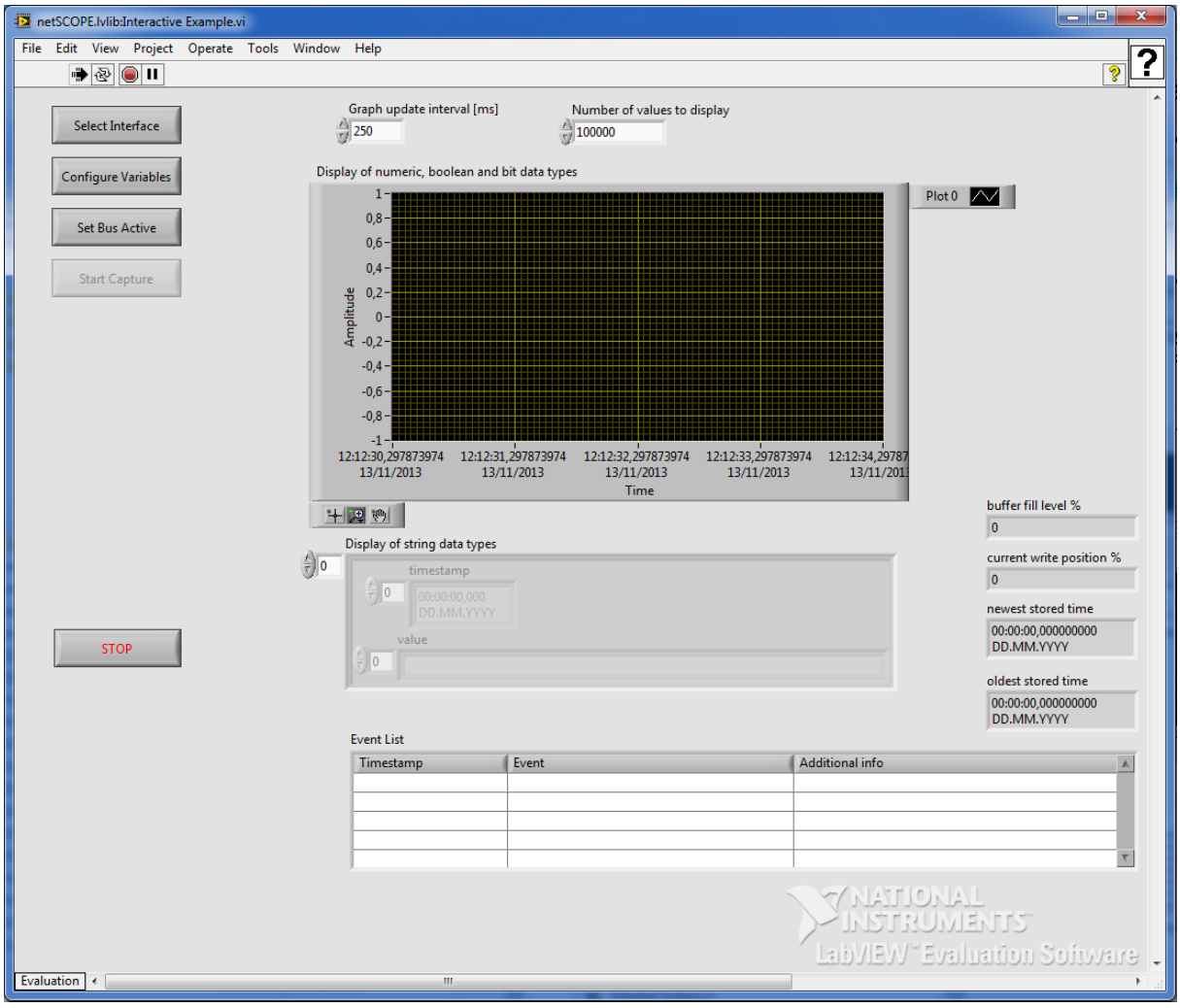


Figure 8: netSCOPE.lvlib:Interactive Example.vi - Front Panel

- Click on **Configure Variables**.

➧ The **netSCOPE.lvlib: EtherCAT Select Variables Dialog.vi** pane is opened.

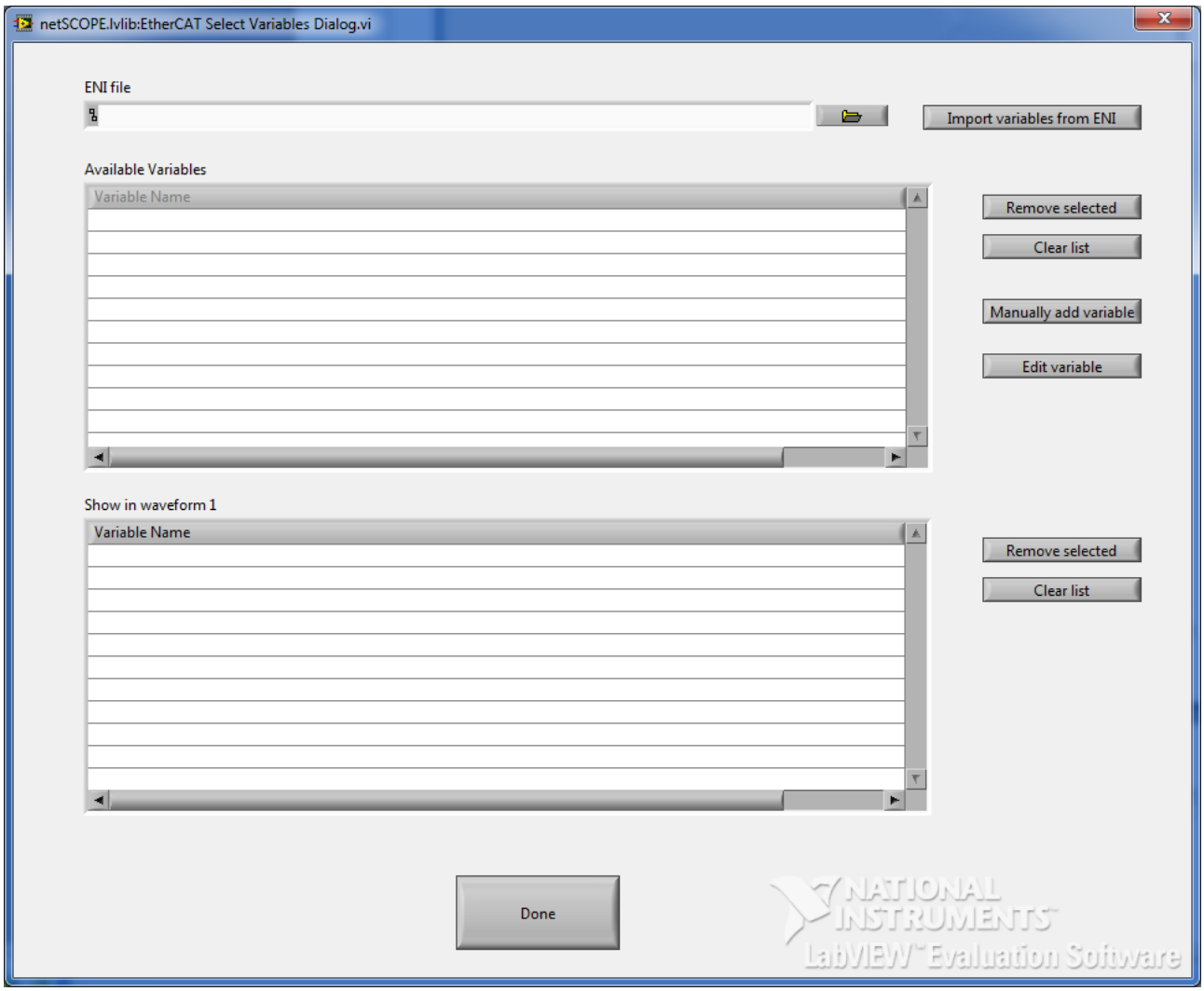


Figure 9: netSCOPE.lvlib: EtherCAT Select Variables Dialog.vi

- Select  .
- Select the required ENI file (*.xml).
- Select **Import variables from ENI**.
- The imported variables are listed in the **Available Variables** table.

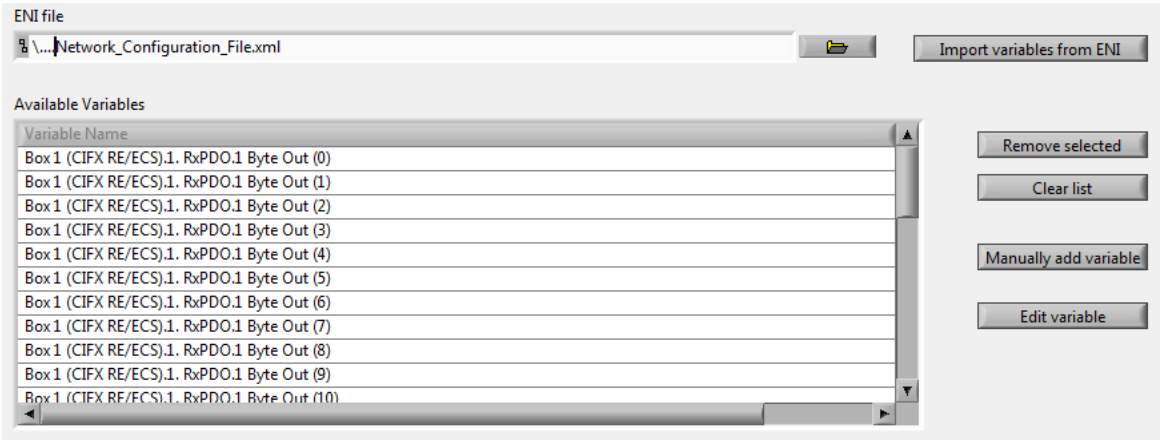


Figure 10: netSCOPE.lvlib: EtherCAT Select Variables Dialog.vi – Available Variables

Under **Available Variables** you can remove a variable, clear all variables, add a variable manually or edit a variable.

Remove selected:

To remove a variable from the **Available Variables** list:

- Select the variable to be removed.
- Click on **Remove selected**.

Clear list:

To clear the total **Available Variables** list:

- Click on **Clear list**.

Manually add variable:

To add a variable manually to the **Available Variables** list:

- Click on **Manually add variable**.

➪ The **netSCOPE.lvlib: Add or Modify Variable Dialog.vi** pane is displayed.

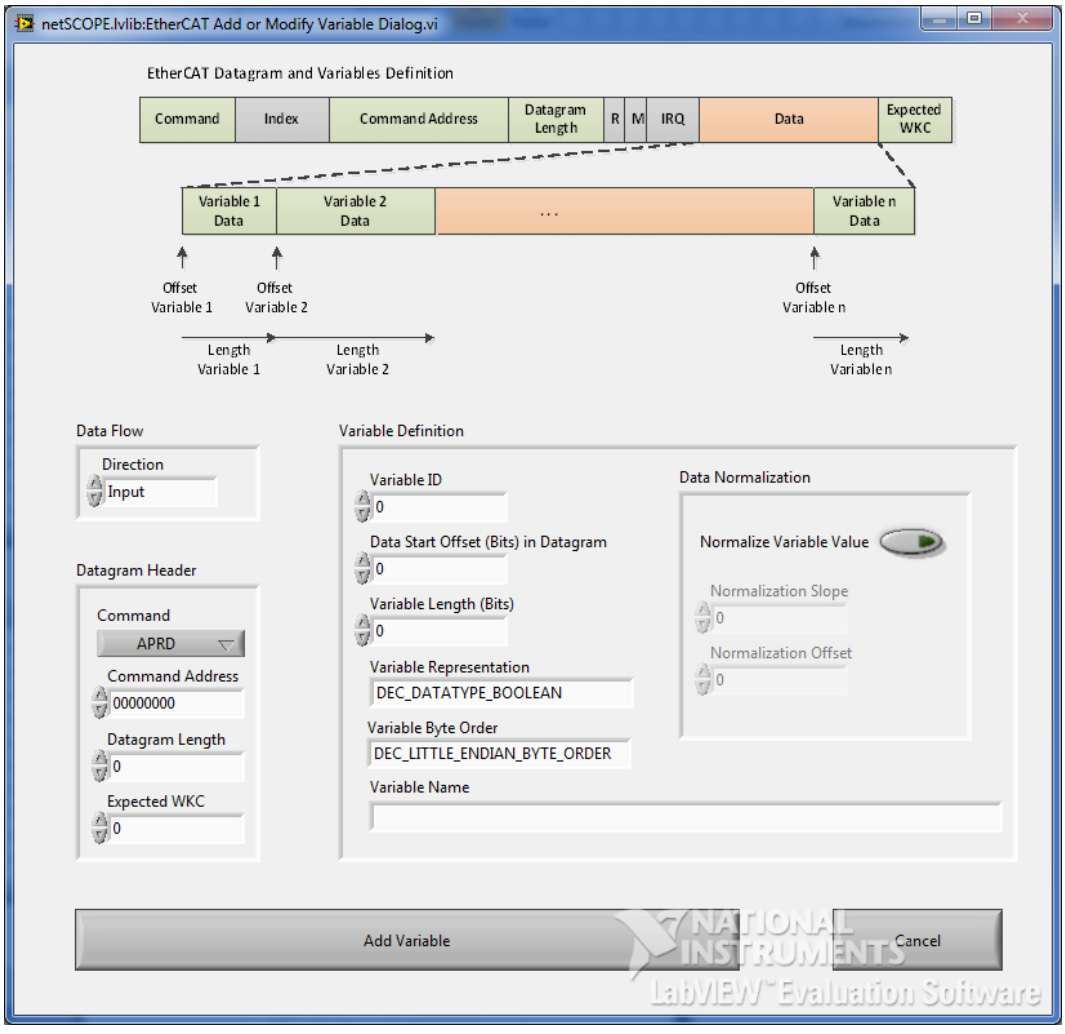


Figure 11: netSCOPE.lvlib: Add or Modify Variable Dialog.vi

- Enter the single variable definition values as described in the table *Supported Data Types in EtherCAT* on page 75.

➢ Click on **Add Variable** (below the entry fields).

➮ The variable definition values for the new variable are stored and the **netSCOPE.lvlib: Add or Modify Variable Dialog.vi** pane is closed.

Edit variable:

To edit a variable given in the **Available Variables** list:

➢ Click on **Edit variable**.

➮ The **netSCOPE.lvlib: Add or Modify Variable Dialog.vi** pane is displayed showing the variable definition values of the selected variable.

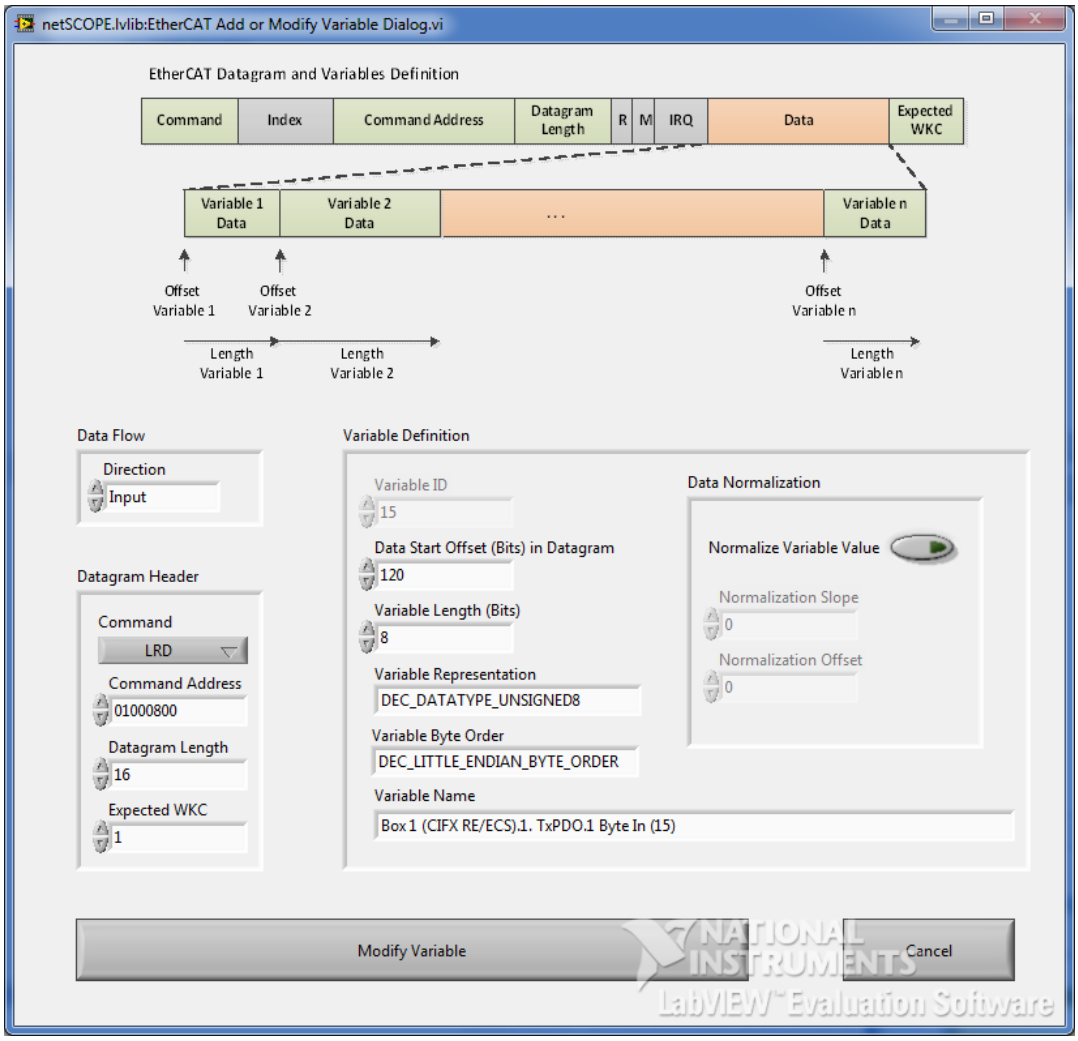


Figure 12: netSCOPE.lvlib: Add or Modify Variable Dialog.vi

The single variable definition values as described in the table *Supported Data Types in EtherCAT* on page 75.

➢ Edit or change the values.

➢ Click on **Modify Variable** (below the entry fields).

➮ The variable definition values are changes and the **netSCOPE.lvlib: Add or Modify Variable Dialog.vi** pane is closed.

3.2.1.3 Show in waveform

- In the **netSCOPE.lvlib: EtherCAT Select Variables Dialog.vi** pane put a variable from the **Available Variable** list by drag & drop to the **Show in waveform1** list.

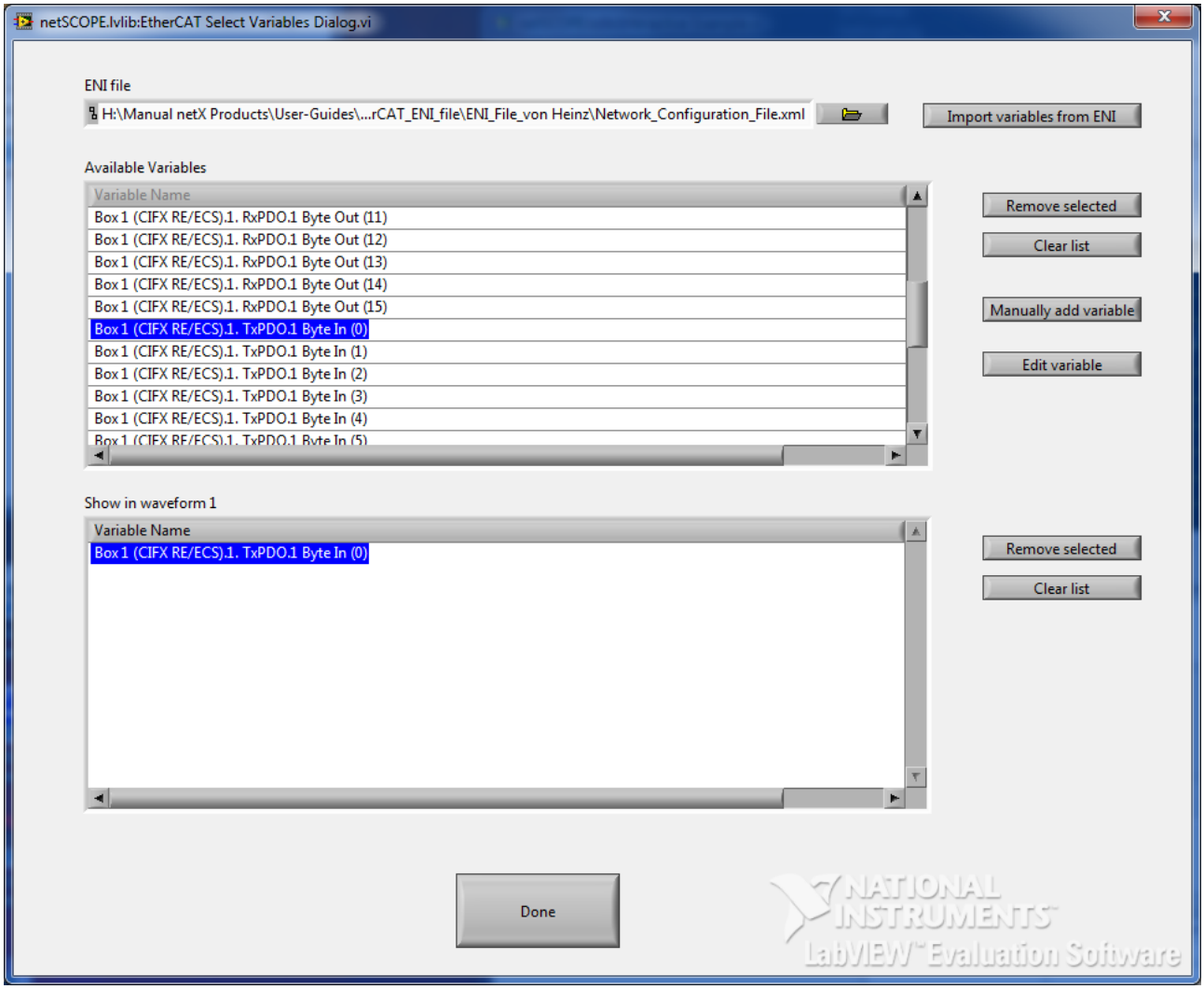


Figure 13: netSCOPE.lvlib: EtherCAT Select Variables Dialog.vi

Under **Show in waveform1** you can remove a variable and clear all variables.

Remove selected:

To remove a variable from the **Show in waveform1** list:

- Select the variable to be removed.
- Click on **Remove selected**.

Clear list:

To clear the total **Show in waveform1** list:

- Click on **Clear list**.

Done

➢ Click on **Done**.

➢ The **netSCOPE.lvlib: EtherCAT Select Variables Dialog.vi** pane is closed.

➢ The newly defined variables are saved.

3.2.1.4 Set Bus Active / Set Bus Inactive

➢ Select **Set Bus Active**.

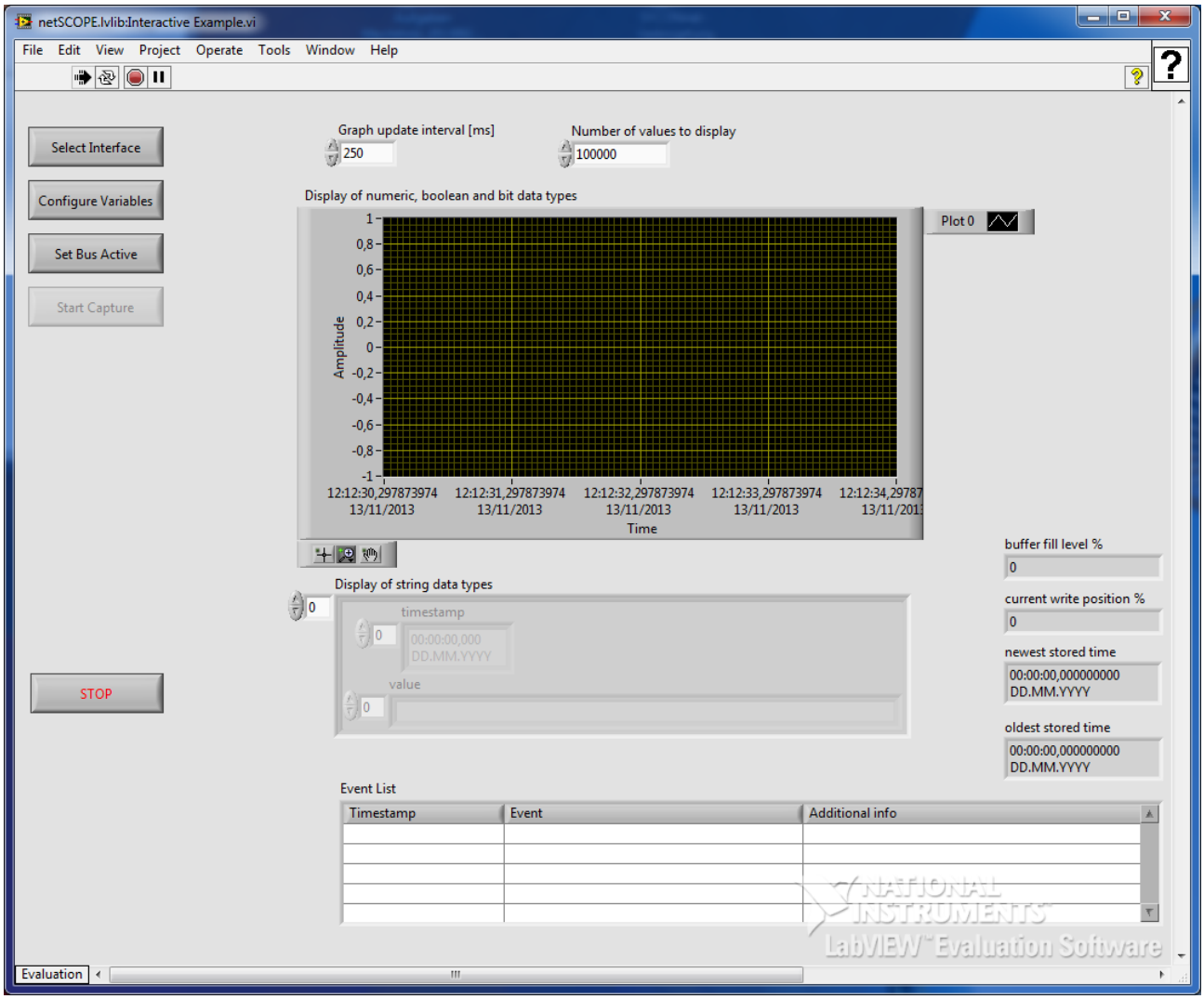


Figure 14: netSCOPE.lvlib:Interactive Example.vi - Front Panel

➢ Via **Set Bus Active** the netSCOPE data acquisition card is started

➢ **Set Bus Active** changes to **Set Bus Inactive**.

➢ The netSCOPE data acquisition card is activated on the bus and ready for data capturing. The measurement and data capturing are not yet started.

3.2.1.5 Start Capture

➢ Select **Start Capture**.

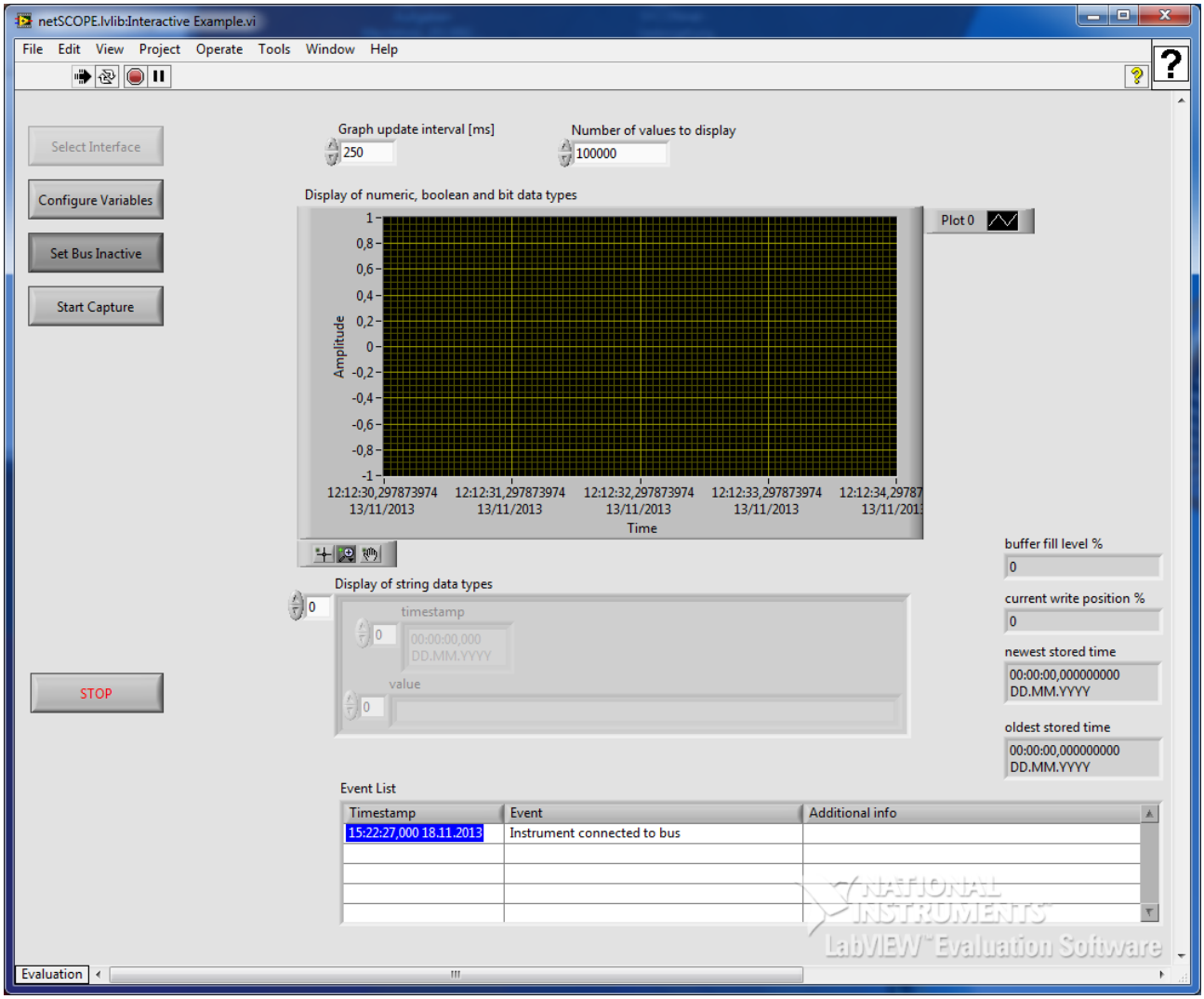


Figure 15: netSCOPE.lvlib:Interactive Example.vi - Front Panel

- Via **Start Capture** the measurement and data capturing are started.
- **Start Capture** changes to **Stop Capture**.
- In the **Event List** (below) any possible notification events (states or error states) are listed. See section *Notification Events* on page 56.

➢ The display shows the measured and captured data, which are transferred from the Slave to the Master. The history of the variable gets visible. Any values transferred at the bus get visible (inputs, outputs, default values, counter, sinus signals etc.).

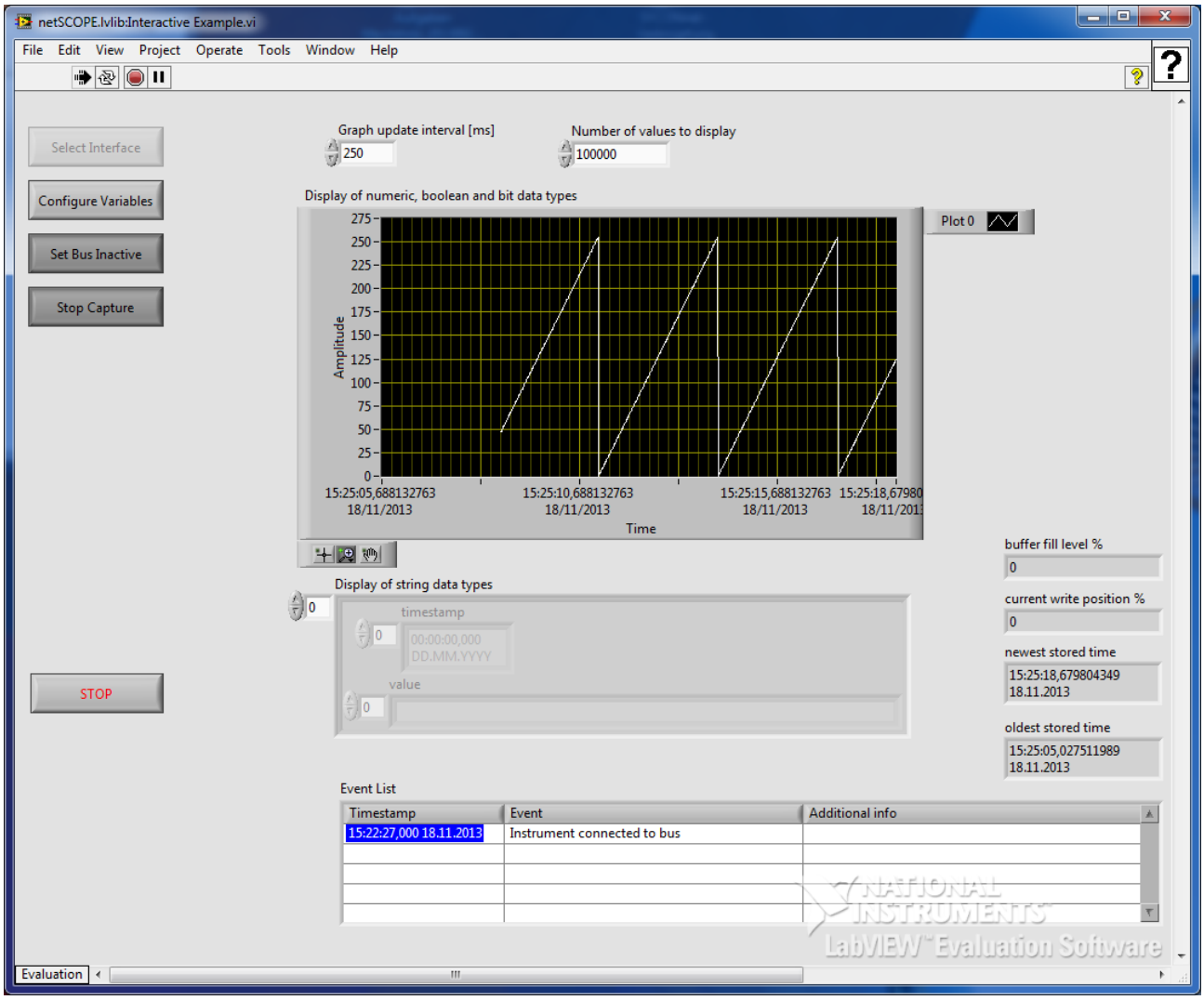


Figure 16: netSCOPE.lvlib:Interactive Example.vi - Front Panel (Example: 4 Bytes in cyclic)

3.2.1.6 Stop Capture, Set Bus inactive, STOP

➢ To stop the capturing process click on **Stop Capture**.

➢ To set the Bus inactive, click on **Set Bus Inactive**.

➢ To stop the netSCOPE.lvlib:Interactive Example.vi click on **STOP**.

Note: When STOP has been selected, for another measuring and capturing cycle the ENI file must bee loaded newly.

Important: Do not use the LabVIEW's **Abort Execution** to stop the capturing and measuring process. Instead of this, use **Stop Capture**, **Set Bus inactive** and **STOP**.

3.2.2 netSCOPE.lvlib:Simple Example.vi

The **netSCOPE.lvlib:Simple Example.vi** shows the minimal programming effort which is needed to acquire a single process data signal from a netSCOPE device.

Figure 17: netSCOPE.lvlib:Simple Example.vi

3.2.2.1 Open Front Panel

In the LabVIEW **netSCOPE.lvlib on Main Application Instance / Items** pane:

1. Open Frontpanel

➢ Select the **Items** tab > **netSCOPE.lvlib** > **Examples**.

➢ Double click to **Simple Example.vi**.

➢ The **Front Panel** view of the **netSCOPE.lvlib:Simple Example.vi** is opened.

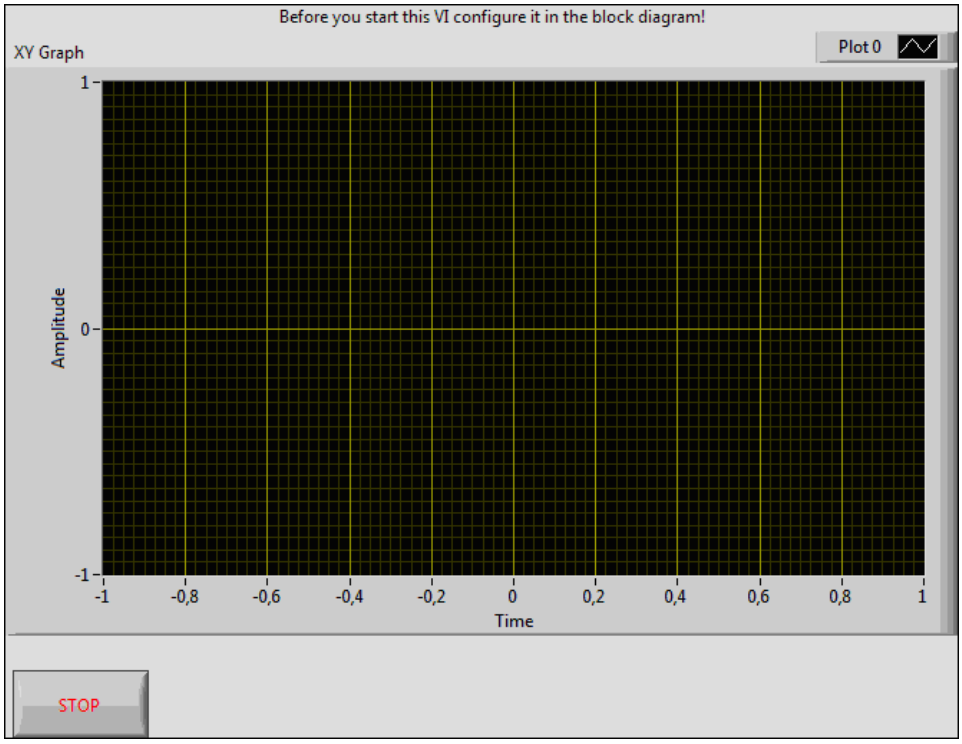


Figure 18: netSCOPE.lvlib:Simple Example.vi - Front Panel

2. Open Block Diagram.

➢ Double click to the **netSCOPE.lvlib:Simple Example.vi Front Panel.**

➢ The **netSCOPE.lvlib:Simple Example.vi Block Diagram** is opened (see figure *netSCOPE.lvlib:Simple Example.vi* on page 23).

3. Enter or change data manually.

➢ Under **My variable name:** Enter the name of the variable to display (regexp).

- Under **Change to path for ENI file:** Select the path to the ENI file to be loaded.
- Under **Change to data type the variable has:** Manually select the data type of the variable which shall be displayed.
- Under **Number of values to display:** Enter or change the number of samples which can be viewed in the graph at the same time.

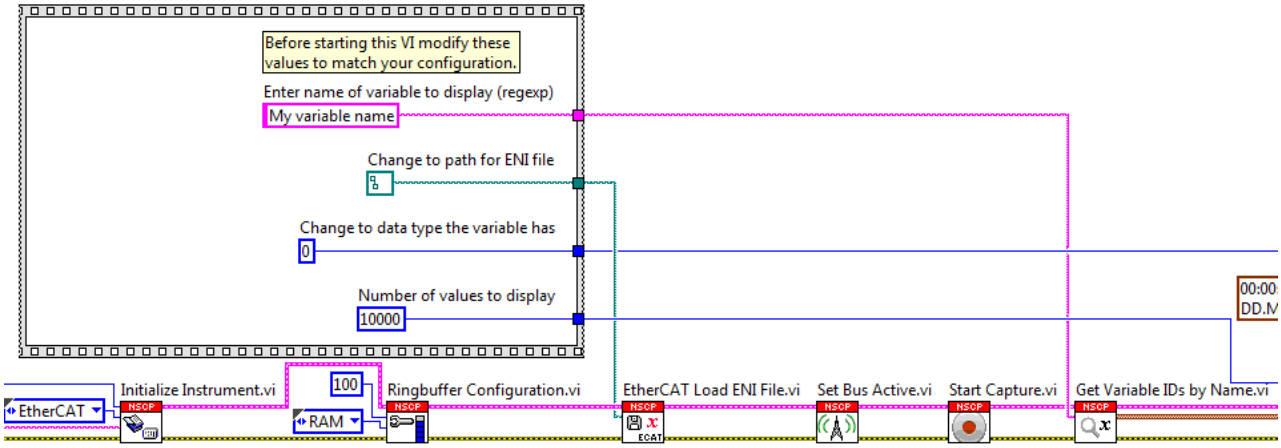


Figure 19: netSCOPE.lvlib:Simple Example.vi Block Diagram - Slope for Manual Data Input

4. Start Visualization

- Change to the **netSCOPE.lvlib:Simple Example.vi Front Panel**.
- Click to Run.
- The values of the variable are displayed in the XY graph over Time diagram.

5. Stop Visualization

- To stop the visualization click on **STOP**.

3.2.2.2 netSCOPE.lvlib:Simple Example.vi Block Diagram

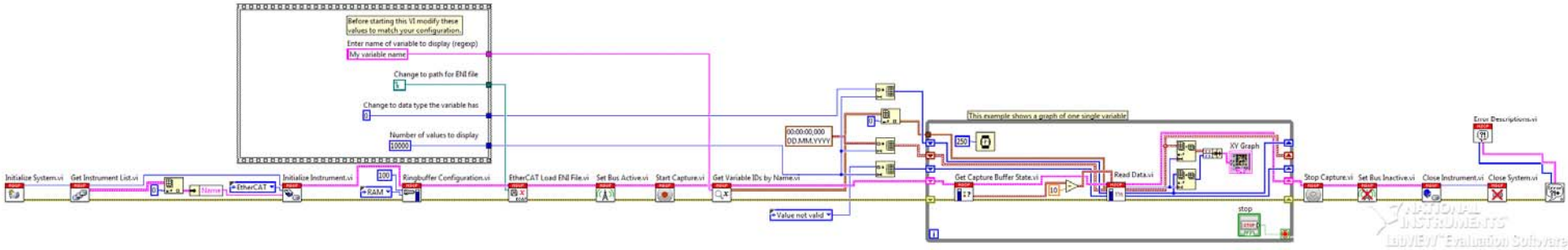


Figure 20: netSCOPE.lvlib:Simple Example.vi Block Diagram

The **netSCOPE.lvlib:Simple Example.vi Block Diagram** (see figure *netSCOPE.lvlib:Simple Example.vi Block Diagram* on page 25) shows the VIs required to visualize the values of a certain variable and how they are connected to each other in the block diagram from the left side to the right side:

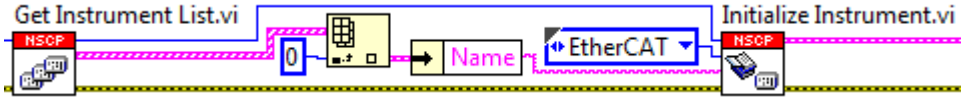
- **Initialize System.vi**:

This driver VI initializes the netSCOPE system. This is the first VI to be called before any other netSCOPE VI is useable. For details see section *netSCOPE.lvlib:Initialize System.vi* on page 86.

- **Get Instrument List.vi**:

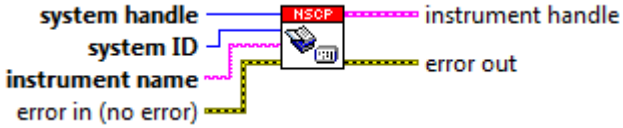
This driver VI returns a list of all instruments of the system. For details see section *netSCOPE.lvlib:Get Instrument List.vi* on page 77.

- Connection to the device:



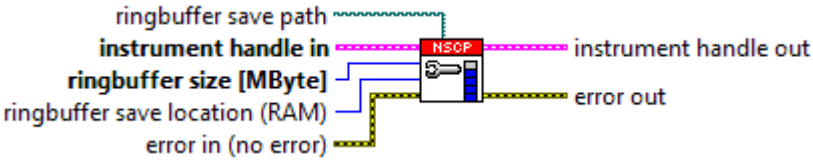
The first netSCOPE device is selected.

- **Initialize Instrument.vi:**



This VI initializes one instrument specified by its name. This VI must be called once before using any instrument specific VIs. For details see section *netSCOPE.lvlib:Initialize Instrument.vi* on page 85.

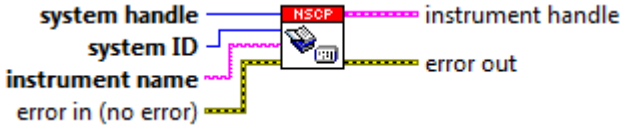
- **Ringbuffer Configuration.vi**



Configures the ringbuffer storage size in Megabytes and location.
- RAM storage location does not need a save path.
- HDD storage location needs a save path to be specified.
Note, that HDD storage is most likely less performant than RAM storage. For details see section *netSCOPE.lvlib:Ringbuffer Configuration.vi* on page 57.

- Manual Data Input:
  My variable name: To enter the name of variable to be displayed (regexp)
  Change to path for ENI file: The path to the ENI file to be loaded must be selected. The ENI file contains all variables and its values.
  Change to data type the variable has: To manually select the data type of the variable which shall be displayed.
  Number of values to display: Allows to enter or change the number of samples which can be viewed in the graph at the same time.

- **EtherCAT Load ENI File.vi:**



  This EtherCAT specific VI loads all variables from the given ENI file. For details see section *netSCOPE.lvlib:EtherCAT Load ENI File.vi* on page 72.

- **Set Bus Active.vi:**

  Activates the physical connection to the communication bus or network. This is a prerequisite before calling the Start Capture VI. For details see section *netSCOPE.lvlib:Set Bus Active.vi* on page 49.

- **Start Capture.vi:**

  Starts the capture task for process data values. This requires the bus to be activated via the Set Bus Active VI. For details see section *Start Capture* on page 21.

- **Get Variable IDs by Name.vi:**

  Returns a list of variables IDs for all variables which's name matches the given regular expression. For details see section *netSCOPE.lvlib:Get Variable IDs by Name.vi* on page 79.

- Data Visualization:
  The variable is visualized.

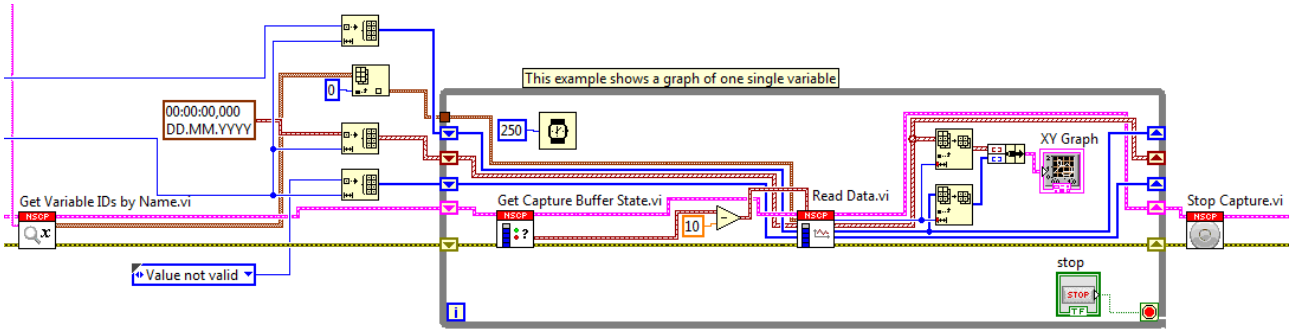
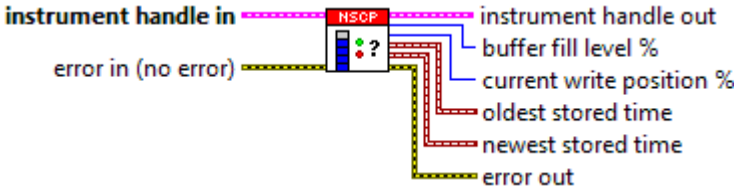


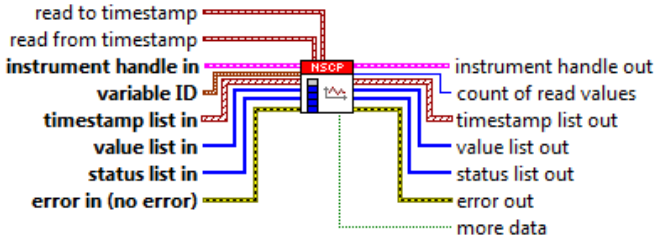
Figure 21: netSCOPE.lvlib:Simple Example.vi Block Diagram - Loop for Data Visualization

- **netSCOPE.lvlib:Get Capture Buffer State.vi:**



Gets the current state of the capture ring buffer. For details see section *netSCOPE.lvlib:Get Capture Buffer State.vi* on page 44.

- **Read Data.vi:**



Reads a variables value from the capture data ring buffer.
Reading is limited to the time span given, from time must always be specified.
The maximum amount of data that is read out is implicitly specified by the input array size. All input arrays (timestamp list, value list, status list) must have the same size. The value list contains elements which must be preinitialized with the LabVIEW data type and its expected size.
The amount of actually read values is returned by “count of read values” if this value is smaller than the array size, the rest of the arrays elements do not contain correct data and must be ignored. The VI does not resize the arrays automatically. For details see section *netSCOPE.lvlib:Read Data.vi* on page 60.

- **Stop Capture.vi:**

Stops the capture task for process data values.
After stopping no new data will be stored in the capture ring buffer, but yet captured data is still available. For details see section *netSCOPE.lvlib:Stop Capture.vi* on page 52.

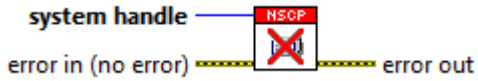
- **Set Bus Inactive.vi:**

Deactivates the physical connection to the communication bus or network.
If a capture is running is must be stopped via the Stop Capture VI first. For details see section *netSCOPE.lvlib:Set Bus Inactive.vi* on page 50.

- **Close Instrument.vi:**

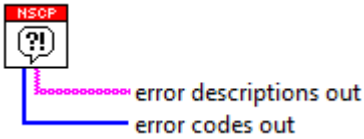
Closes an instrument and returns the system handle the instrument belongs to. This will discard all configurations and captured ring buffer data for this instrument. The Instrument will not be accessible anymore unless it is reopened via the Initialize Instrument VI. For details see section *netSCOPE.lvlib:Close Instrument.vi* on page 83.

- **Close System.vi:**



Closes a system. All instruments belonging to this system will be closed automatically, all captured ringbuffer data in this system will be discarded. For details see section *netSCOPE.lvlib:Close System.vi* on page 84.

- **Error Description.vi:**



This VI returns all netSCOPE specific error codes and descriptions. Useful to be connected to the General Error Handler VIs [user-defined codes] and [user-defined descriptions] inputs. For details see section *netSCOPE.lvlib:Close System.vi* on page 84.

3.3 Examples - Helpers

3.3.1 netSCOPE.lvlib:Select Device Frontpanel.vi

The **netSCOPE.lvlib:Select Device Frontpanel.vi** example represents a subfunction of the **netSCOPE.lvlib:Interactive Example.vi** (see section *Open Front Panel, Select Interface, Select Device Frontpanel* on page 13) and includes the subfuntions **Select the Target System, Identify device** and **Select device**.

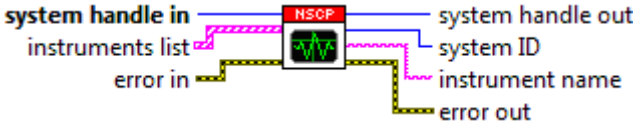


Figure 22: netSCOPE.lvlib:Select Device Frontpanel.vi

system handle in Valid system handle generated by “Initialize System.vi” (see section *netSCOPE.lvlib:Initialize System.vi* page 86).

instrument list A list of available instruments found on the system. The instrument list is created by “Get Instrument List.vi” (see section *netSCOPE.lvlib:Get Instrument List.vi* page 77).

error in (no error) describes error conditions that occur before this VI runs. The default input of this cluster is no error. If an error already occurred, this VI returns the value of **error in** in **error out.** The VI runs normally only if no incoming error exists. Otherwise, the VI passes the **error in** value to **error out.** The **error in** cluster contains the following parameters:

- **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a negative error code. If **status** is FALSE, **code** is 0 or a warning code.
- **source** identifies where an error occurred. The source string includes the name of the VI that produced the error, what inputs are in error, and how to eliminate the error.

system handle out Valid system handle generated by “Initialize System.vi” (see section *netSCOPE.lvlib:Initialize System.vi p*age 86).

system ID Selected target system identifier.

instrument name Name of the selected instrument (for example “netSCOPE”).

error out contains error information. If **error in** indicates that an error occurred before this VI or function ran, **error out** contains the same error information. Otherwise, it describes the error status that this VI or function produces.

- **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.

code is the error or warning code. The default is 0. If **status** is TRUE, **code** is a negative error code. If **status** is FALSE, **code** is 0 or a warning code.

source identifies where an error occurred. The source string includes the name of the VI that produced the error, what inputs are in error, and how to eliminate the error.

3.3.1.1 Select Device Frontpanel

In the LabVIEW **netSCOPE.lvlib on Main Application Instance / Items** pane:

- Select the **Items** tab > **netSCOPE.lvlib** > **Examples** > **Helpers**.
- Double click to **netSCOPE.lvlib:Select Device Frontpanel.vi**.

The **Front Panel** view of the **netSCOPE.lvlib:Select Device Frontpanel.vi** is opened.

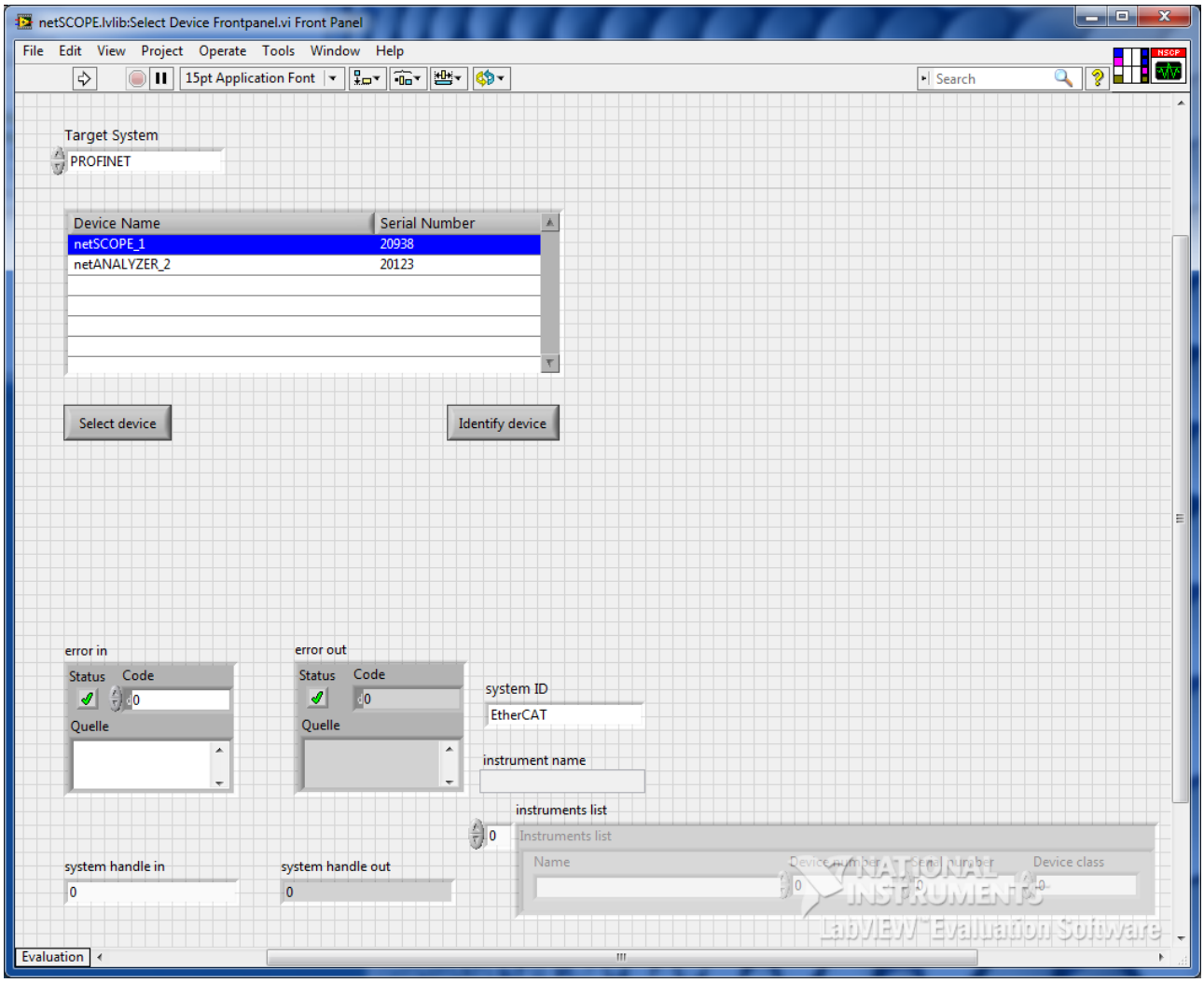


Figure 23: netSCOPE.lvlib: Select Device Frontpanel.vi - Front Panel

- Click **Run**.

The **netSCOPE.lvlib:Select Device Frontpanel.vi** is in **Run** mode:

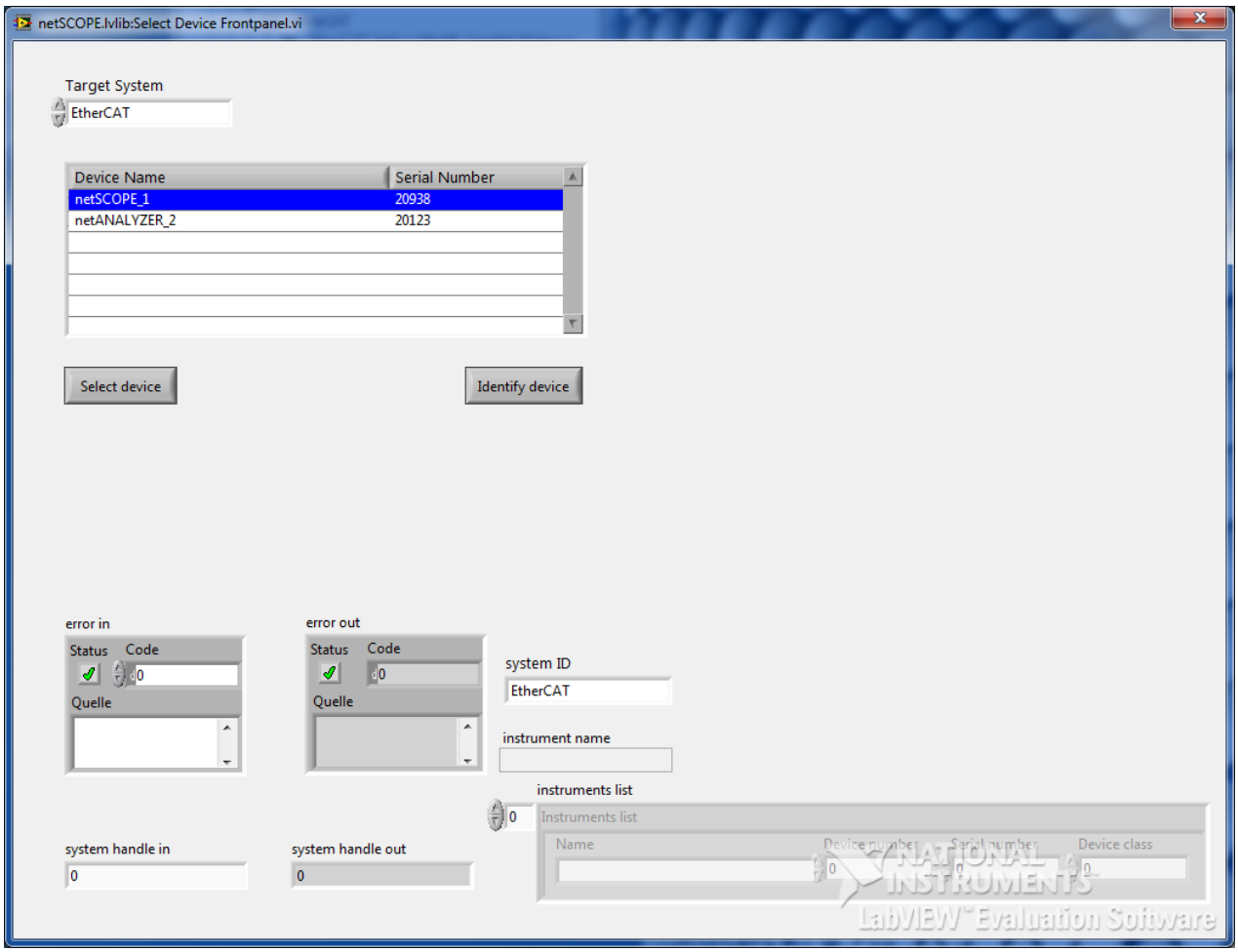


Figure 24: netSCOPE.lvlib:Select Device Frontpanel.vi - Front Panel

In the **netSCOPE.lvlib:Select Device Frontpanel.vi** pane:

- Select the **Target System**: “EtherCAT”.
- Click **Identify device**, to identify your device (optionally).
- The STA0 and the STA1 LED at the netSCOPE data acquisition card blink green for approx 10 sec.
- Click **Select device** and select your device.
- The **netSCOPE.lvlib:Select Device Frontpanel.vi** is in **Stop** mode:

3.4 Examples - Helpers - EtherCAT

3.4.1 netSCOPE.lvlib:EtherCAT Add or Modify Variable Dialog.vi

- Adds or modifies the EtherCAT-specific definition of the given variable.

- EtherCAT-specific VI.

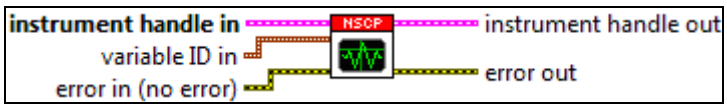


Figure 25: netSCOPE.lvlib:EtherCAT Add or Modify Variable Dialog.vi

instrument handle in identifies a particular instrument session.

variable ID Identifier of the existing variable that should be modified.

error in (no error) describes error conditions that occur before this VI runs. The default input of this cluster is no error. If an error already occurred, this VI returns the value of **error in** in **error out**. The VI runs normally only if no incoming error exists. Otherwise, the VI passes the **error in** value to **error out**. The **error in** cluster contains the following parameters:

- **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a negative error code. If **status** is FALSE, **code** is 0 or a warning code.
- **source** identifies where an error occurred. The source string includes the name of the VI that produced the error, what inputs are in error, and how to eliminate the error.

instrument handle out has the same value as the **instrument handle**.

error out contains error information. If **error in** indicates that an error occurred before this VI or function ran, **error out** contains the same error information. Otherwise, it describes the error status that this VI or function produces.

- **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a negative error code. If **status** is FALSE, **code** is 0 or a warning code.
- **source** identifies where an error occurred. The source string includes the name of the VI that produced the error, what inputs are in error, and how to eliminate the error.

3.4.1.1 Open Front Panel, configure Variable

In the LabVIEW **netSCOPE.lvlib on Main Application Instance / Items** pane:

- Select the **Items** tab > **netSCOPE.lvlib** > **Examples** > **Helpers** > **EtherCAT**.
- Double click to **netSCOPE.lvlib:EtherCAT Add or Modify Variable Dialog.vi**.
- The **Front Panel** view of the **netSCOPE.lvlib: Add or Modify Variable Dialog.vi** is displayed.
- Click **Run**.
- **Add Variable** is enabled.

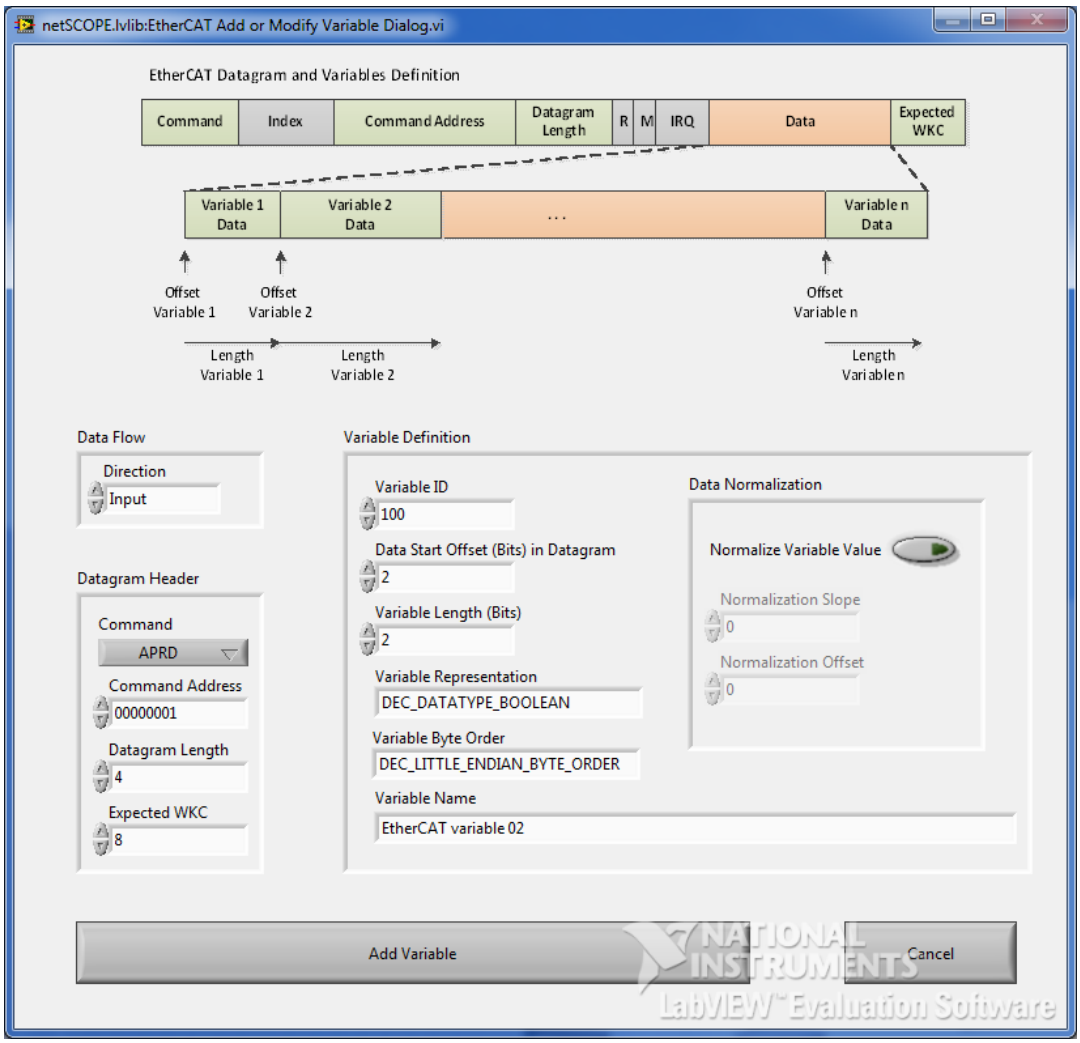


Figure 26: netSCOPE.lvlib:EtherCAT Add or Modify Variable Dialog.vi – Front Panel

- Enter the single variable definition values as described in the table *Supported Data Types in EtherCAT* on page 75.

<table>
<tr><th>Parameter</th><th>Description</th></tr>
<tr><td colspan="2">Data Flow Area</td></tr>
<tr><td>Direction</td><td>Indicates the signal direction and can either have the value "input" or "output".</td></tr>
<tr><td colspan="2">Datagram Header Area</td></tr>
<tr><td>Command</td><td>This selection specifies the EtherCAT command executed in the EtherCAT datagram. Corresponds to the EtherCAT command specified in the Command field of the EtherCAT datagram.
The following EtherCAT commands are defined in the EtherCAT specification:
<table>
<tr><th>Code</th><th>Command</th></tr>
<tr><td>APRD</td><td>Auto increment physical read</td></tr>
<tr><td>APWR</td><td>Auto increment physical write</td></tr>
<tr><td>APRW</td><td>Auto increment physical read write</td></tr>
<tr><td>FPRD</td><td>Configured address physical read</td></tr>
<tr><td>FPWR</td><td>Configured address physical write</td></tr>
<tr><td>FPRW</td><td>Configured address physical read write</td></tr>
<tr><td>BRD</td><td>Broadcast read</td></tr>
<tr><td>BWR</td><td>Broadcast write</td></tr>
<tr><td>BRW</td><td>Broadcast read write</td></tr>
<tr><td>LRD</td><td>Logical read</td></tr>
<tr><td>LWR</td><td>Logical write</td></tr>
<tr><td>LRW</td><td>Logical read write</td></tr>
<tr><td>ARMW</td><td>Auto increment physical read multiple write</td></tr>
<tr><td>FRMW</td><td>Configured address physical read multiple write</td></tr>
</table></td></tr>
<tr><td>Command Address</td><td>This value is specified as a hexadecimal address. Corresponds to the address specified in the Command field of the EtherCAT datagram address.
The allowed value range extends from 0x0 to 0xFFFFFFFF.</td></tr>
<tr><td>Datagram Length</td><td>Length of the datagram (expressed as the number of bits in the datagram)
Corresponds to the length specified in the "Datagram Length" field of the EtherCAT datagram.</td></tr>
<tr><td>Expected WKC</td><td>Expected value of the working counter. Corresponds to the value specified in the field "Expected WKC" of the EtherCAT datagram.
The allowed value range extends from 0 to 65535.</td></tr>
<tr><td colspan="2">Variable Definition Area</td></tr>
<tr><td>Variable ID</td><td>ID that uniquely identifies the variable.
Note: You must not use the same variable ID twice otherwise the error message is displayed “Duplicate Variable ID, please select another ID!”</td></tr>
<tr><td>Data Start Offset (Bits) in Datagram</td><td>This value indicates the offset of the variable currently to be defined relative to the beginning of the data field (data) in the EtherCAT datagram. It is expressed as the number of bits counted from the memory location of the first bit of the first variable of the data field.
If the variable currently to be defined is the first in the data field, the value is 0.</td></tr>
<tr><td>Variable Length (Bits)</td><td>This value specifies the length of the variable currently to be defined specified as number of the bits.</td></tr>
<tr><td>Variable Representation</td><td>This value specifies the data type of the variable currently to be defined.
The following data types are supported in EtherCAT:
<table>
<tr><th>Data Type</th><th>Description</th><th>Number of Bits</th><th>Range of Value</th></tr>
<tr><td>BOOLBIT</td><td>‘0’: FALSE
‘1’: TRUE</td><td>1</td><td></td></tr>
<tr><td>BIT8</td><td></td><td>8</td><td></td></tr>
<tr><td>SINT</td><td>Short integer</td><td>8</td><td>-128 … 127</td></tr>
<tr><td>INT</td><td>Integer</td><td>16</td><td>-32768 … 32767</td></tr>
</table></td></tr>
</table>

Parameter	Description
	<table><tr><td>INT24</td><td></td><td>24</td><td></td></tr><tr><td>DINT</td><td>Double integer</td><td>32</td><td>-231 … +231-1</td></tr><tr><td>INT40</td><td></td><td>40</td><td></td></tr><tr><td>INT48</td><td></td><td>48</td><td></td></tr><tr><td>INT56</td><td></td><td>56</td><td></td></tr><tr><td>LINT</td><td>Long integer</td><td>64</td><td></td></tr><tr><td>USINT</td><td>Unsigned short integer</td><td>8</td><td>0 … 255</td></tr><tr><td>UINT</td><td>Unsigned integer/Word</td><td>16</td><td>0 …65535</td></tr><tr><td>UINT24</td><td></td><td>24</td><td></td></tr><tr><td>UDINT</td><td>Unsigned double integer</td><td>32</td><td>0 … +232-1</td></tr><tr><td>UINT40</td><td></td><td>40</td><td></td></tr><tr><td>UINT48</td><td></td><td>48</td><td></td></tr><tr><td>UINT56</td><td></td><td>56</td><td></td></tr><tr><td>ULINT</td><td>Unsigned long integer</td><td>64</td><td>0 … +264-1</td></tr><tr><td>REAL</td><td>Floating point</td><td>32</td><td></td></tr><tr><td>LREAL</td><td>Long Float</td><td>64</td><td></td></tr><tr><td>VISIBLE_STRING</td><td>Visible string (1 octet per character)</td><td>8*n</td><td></td></tr><tr><td>OCTET_STRING</td><td>Sequence of octets</td><td>8*(n+1)</td><td></td></tr><tr><td>UNICODE_STRING</td><td>Sequence of UNIT</td><td>16*(n+1)</td><td></td></tr></table>
Variable Byte Order	This value indicates the byte order used in the internal representation of the variable currently to be defined. Possible values are: DEC_LITTLE_ENDIAN_BYTE_ORDER (Intel format, which means: the most significant byte comes first, the less significant comes last). DEC_BIG_ENDIAN_BYTE_ORDER (Motorola format, which means: the less significant byte comes first, the most significant byte comes last).
Variable Name	This value indicates the full name of the variable currently to be defined. **Note:** You must enter a variable name otherwise the error message is displayed “No variable name specified”.
Data Normalization Area	
Normalization Factor	The data can be normalized if necessary by multiplication by a normalization factor and adding a normalization offset. In this field, the normalization factor can be specified.
Normalization Offset	The data can be normalized if necessary by multiplication by a normalization factor and adding a normalization offset. In this field, the normalization offset can be specified.

Table 2: netSCOPE.lvlib: Add or Modify Variable Dialog.vi – Example EtherCAT

- Click on **Add Variable** (below the entry fields).

The variable definition values for the new variable are stored and the **netSCOPE.lvlib: Add or Modify Variable Dialog.vi** pane is closed.

3.4.2 netSCOPE.lvlib: EtherCAT Select Variables Dialog.vi

The **netSCOPE.lvlib:** EtherCAT **Select Variables Dialog.vi** example represents a subfunction of the **netSCOPE.lvlib:Interactive Example.vi** (see section *Open Front Panel, Select Interface, Select Device Frontpanel* on page 13) and includes the subfuntions **Import variables from ENI, Manually add variable**, **Edit variable** and **Show in waveform1**.

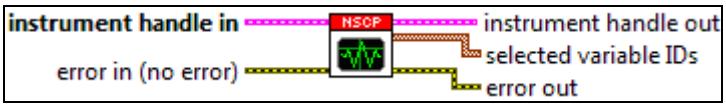


Figure 27: netSCOPE.lvlib: EtherCAT Select Variables Dialog.vi

instrument handle in identifies a particular instrument session.

error in (no error) describes error conditions that occur before this VI runs. The default input of this cluster is no error. If an error already occurred, this VI returns the value of **error in** in **error out.** The VI runs normally only if no incoming error exists. Otherwise, the VI passes the **error in** value to **error out.** The **error in** cluster contains the following parameters:

- **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a negative error code. If **status** is FALSE, **code** is 0 or a warning code.
- **source** identifies where an error occurred. The source string includes the name of the VI that produced the error, what inputs are in error, and how to eliminate the error.

instrument handle out has the same value as the **instrument handle.**

selected variable IDs An array of variable identifiers of all selected variables.

error out contains error information. If **error in** indicates that an error occurred before this VI or function ran, **error out** contains the same error information. Otherwise, it describes the error status that this VI or function produces.

- **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a negative error code. If **status** is FALSE, **code** is 0 or a warning code.
- **source** identifies where an error occurred. The source string includes the name of the VI that produced the error, what inputs are in error, and how to eliminate the error.

3.4.2.1 Open Front Panel, select Variables

In the LabVIEW **netSCOPE.lvlib on Main Application Instance / Items** pane:

- Select the **Items** tab > **netSCOPE.lvlib** > **Examples** > **Helpers** > **EtherCAT**.
- Double click to **netSCOPE.lvlib: EtherCAT Select Variables Dialog.vi**.
- The **Front Panel** view of the **netSCOPE.lvlib: EtherCAT Select Variables Dialog.vi** is displayed.
- Click **Run**.
- The **netSCOPE.lvlib: EtherCAT Select Variables Dialog.vi** is in **Run** mode:

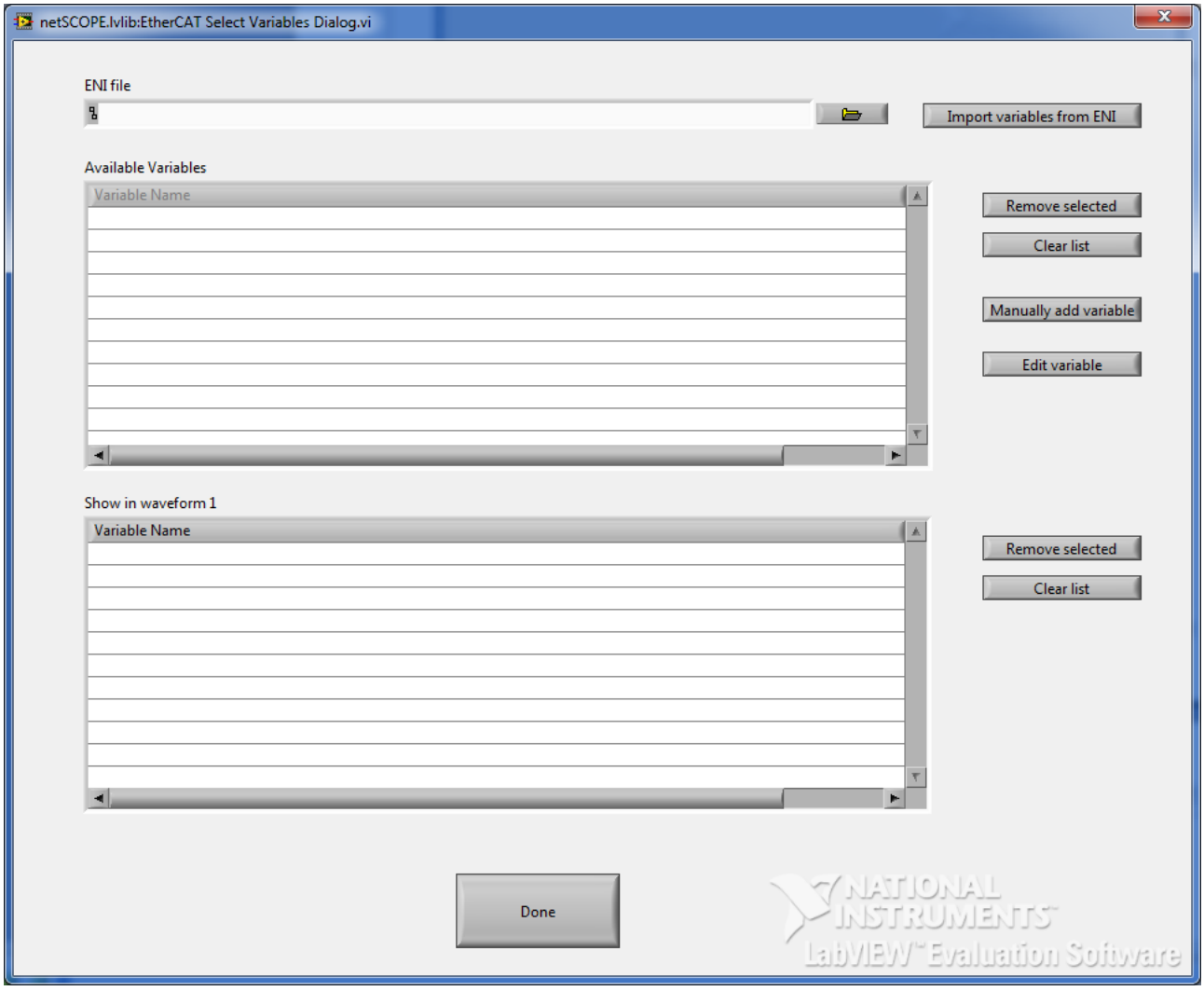


Figure 28: netSCOPE.lvlib: EtherCAT Select Variables Dialog.vi

- Select  .
- Select the required ENI file (*.xml).
- Select **Import variables from ENI**.
- The imported variables are listed in the **Available Variables** table.

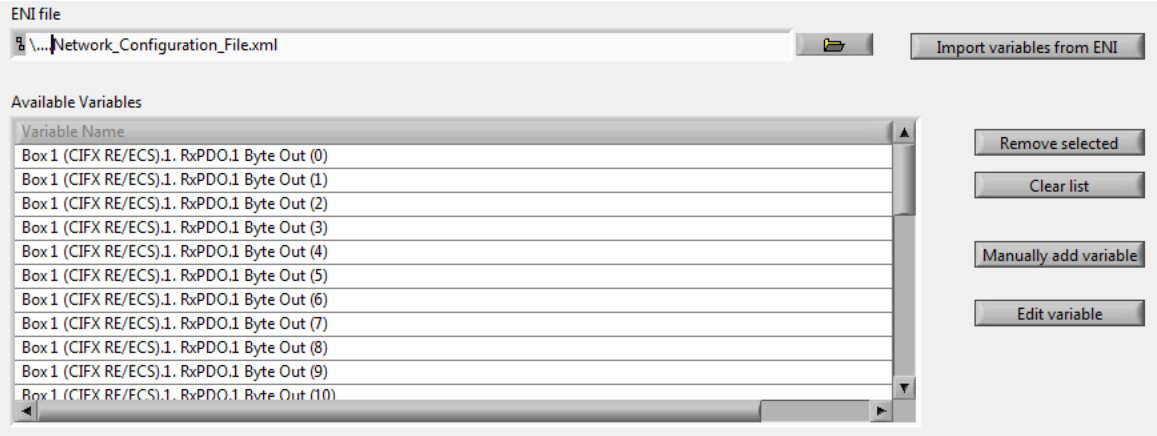


Figure 29: netSCOPE.lvlib: EtherCAT Select Variables Dialog.vi – Available Variables

Under **Available Variables** you can remove a variable, clear all variables, add a variable manually or edit a variable.

Remove selected:

To remove a variable from the **Available Variables** list:

- Select the variable to be removed.
- Click on **Remove selected**.

Clear list:

To clear the total **Available Variables** list:

- Click on **Clear list**.

Manually add variable:

To add a variable manually to the **Available Variables** list:

- Click on **Manually add variable**.
- The **netSCOPE.lvlib: Add or Modify Variable Dialog.vi** pane is displayed.

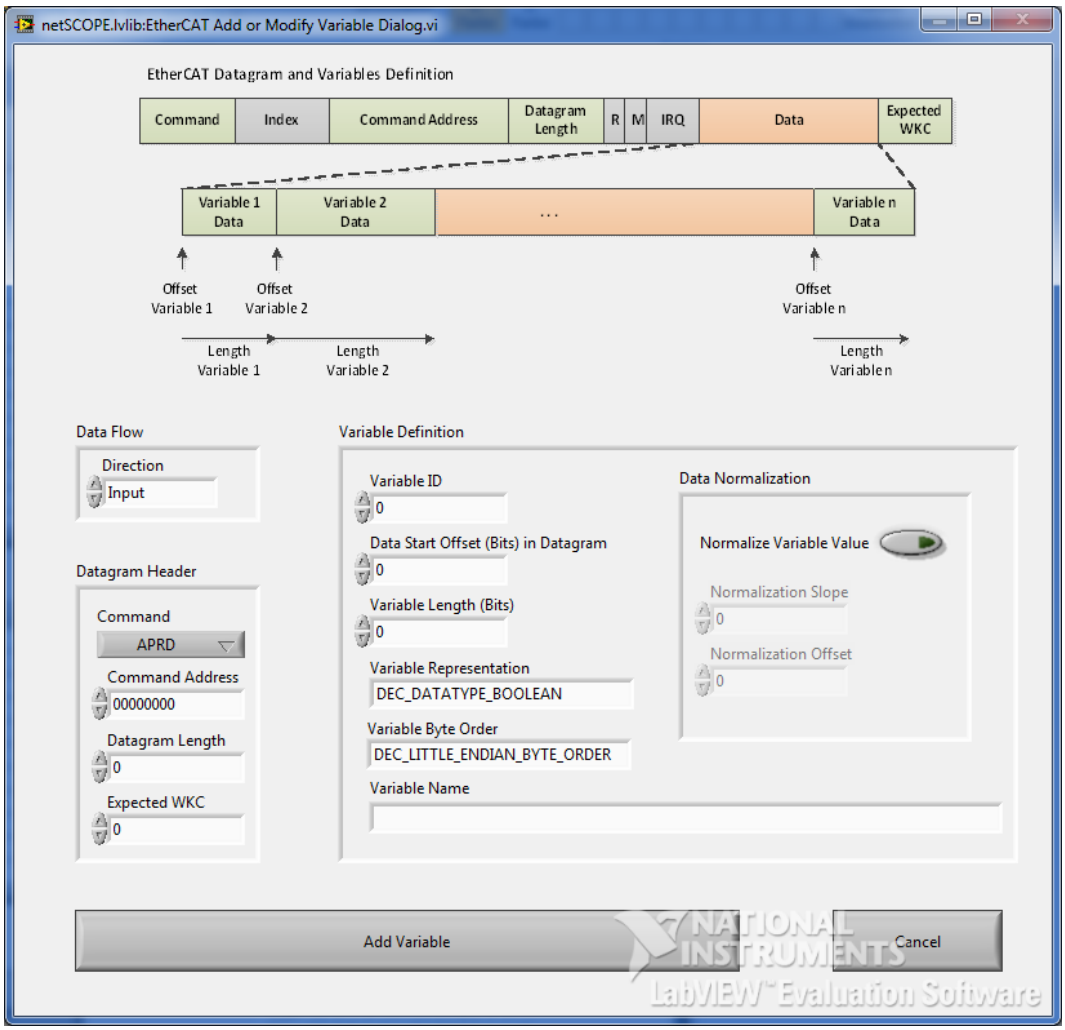


Figure 30: netSCOPE.lvlib: Add or Modify Variable Dialog.vi

- Enter the single variable definition values as described in the table *Supported Data Types in EtherCAT* on page 75.
- Click on **Add Variable** (below the entry fields).

The variable definition values for the new variable are stored and the **netSCOPE.lvlib: Add or Modify Variable Dialog.vi** pane is closed.

Edit variable:

To edit a variable given in the **Available Variables** list:

- Click on **Edit variable**.

The **netSCOPE.lvlib: Add or Modify Variable Dialog.vi** pane is displayed showing the variable definition values of the selected variable.

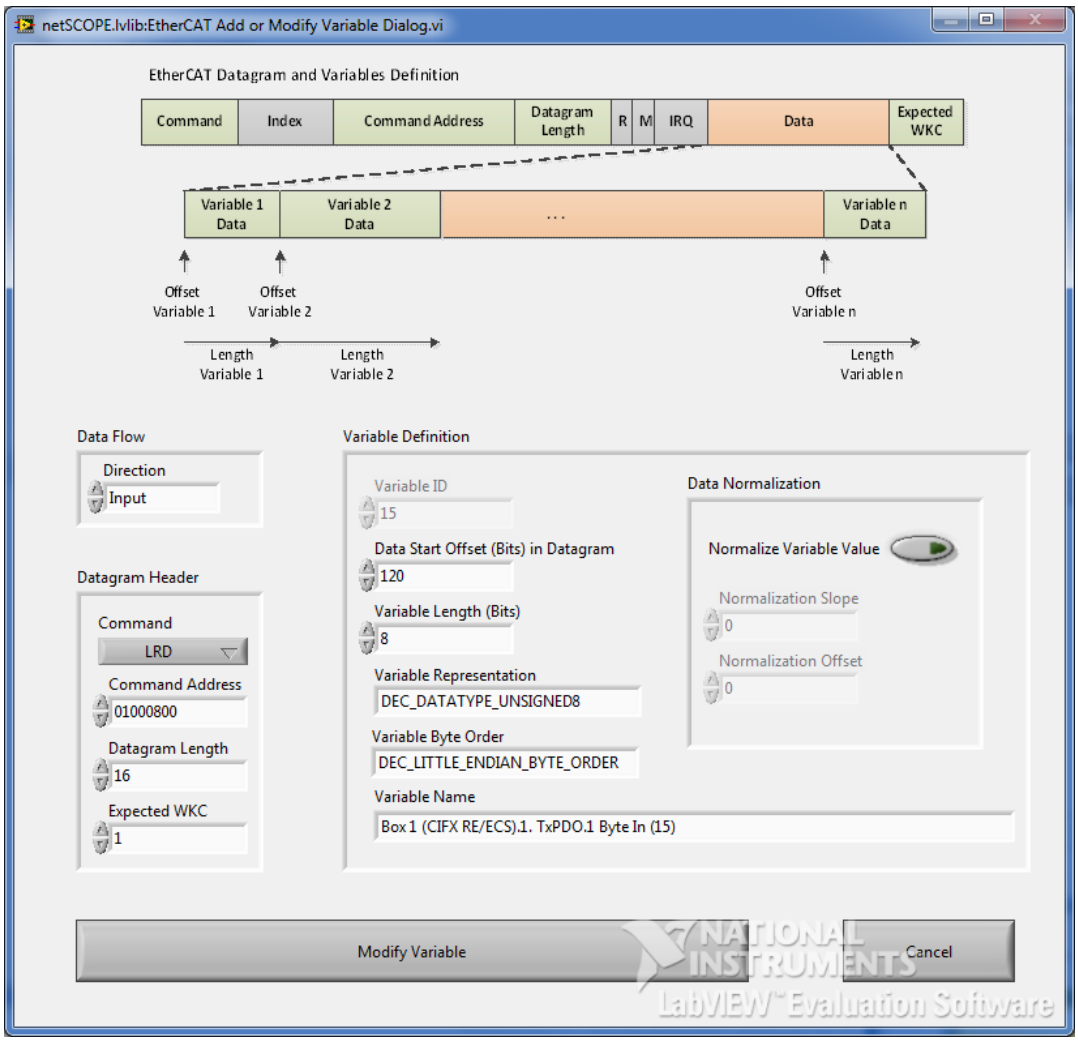


Figure 31: netSCOPE.lvlib: Add or Modify Variable Dialog.vi

The single variable definition values as described in the table *Supported Data Types in EtherCAT* on page 75.

- Edit or change the values.
- Click on **Modify Variable** (below the entry fields).

The variable definition values are changes and the **netSCOPE.lvlib: Add or Modify Variable Dialog.vi** pane is closed.

3.4.2.2 Show in waveform

- In the **netSCOPE.lvlib: EtherCAT Select Variables Dialog.vi** pane put a variable from the **Available Variable** list by drag & drop to the **Show in waveform1** list.

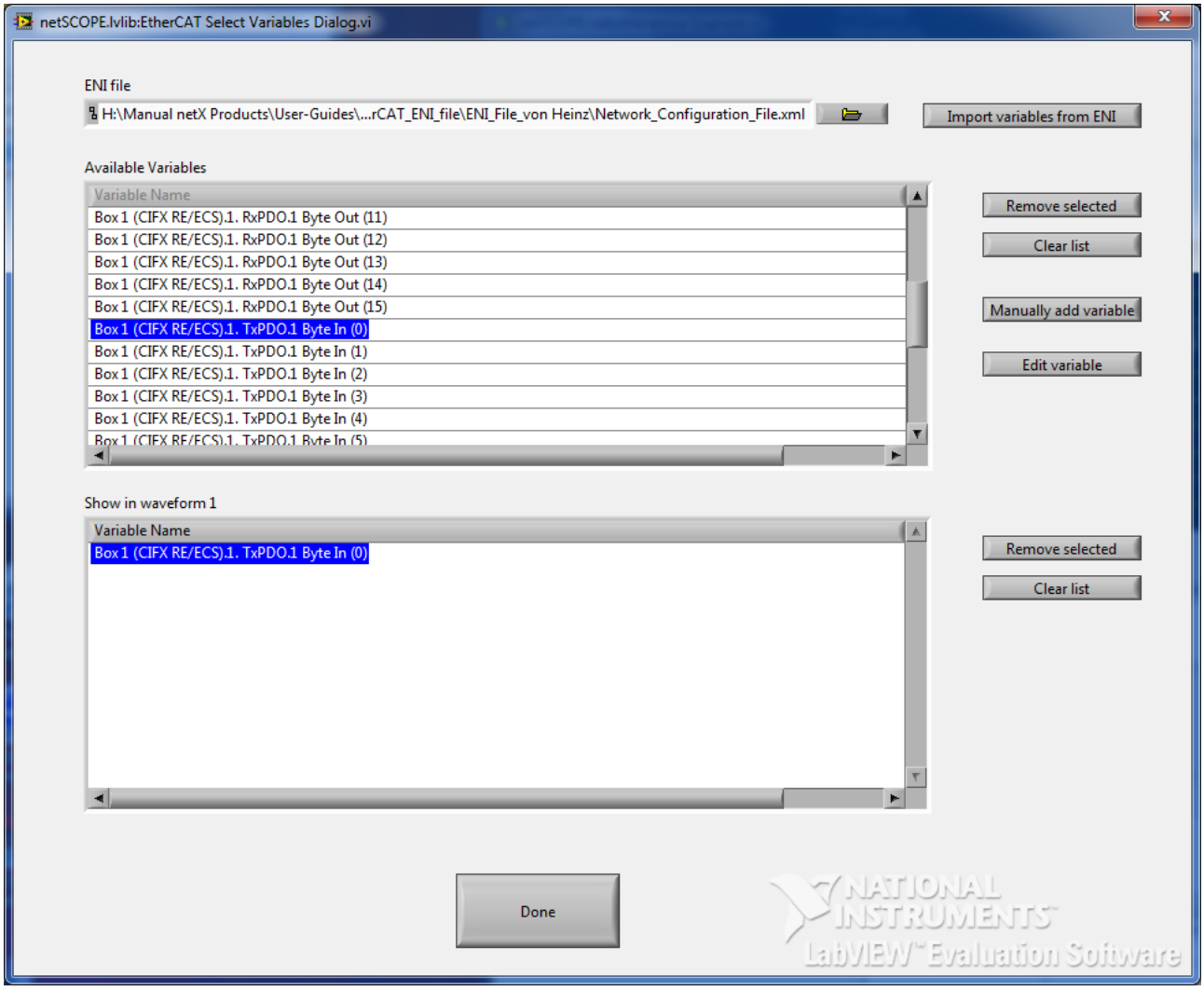


Figure 32: netSCOPE.lvlib: EtherCAT Select Variables Dialog.vi

Under **Show in waveform1** you can remove a variable and clear all variables.

Remove selected:

To remove a variable from the **Show in waveform1** list:

- Select the variable to be removed.
- Click on **Remove selected**.

Clear list:

To clear the total **Show in waveform1** list:

- Click on **Clear list**.

Done

➢ Click on **Done**.

➢ The **netSCOPE.lvlib: EtherCAT Select Variables Dialog.vi** pane is closed.

➢ The newly defined variables are saved.

3.4.2.3 Controls and Indicators in the Frontpanel

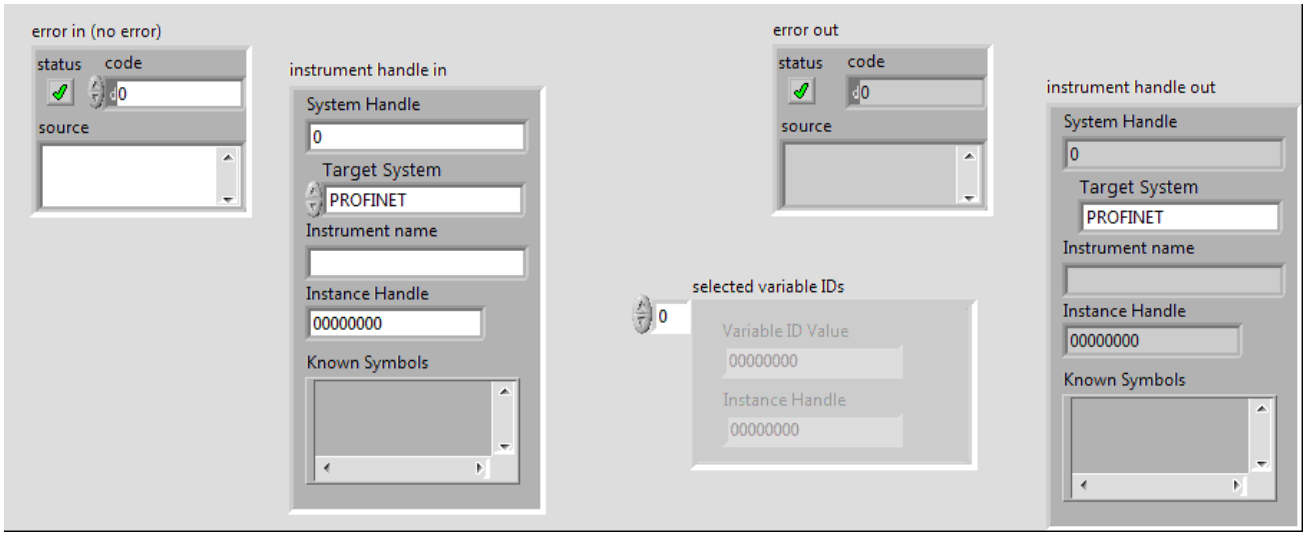


Figure 33: netSCOPE.lvlib: EtherCAT Select Variables Dialog.vi – Controls and Indicators

3.5 Public - Action Status

3.5.1 netSCOPE.lvlib:Get Capture Buffer State.vi

- Gets the current state of the capture ring buffer.

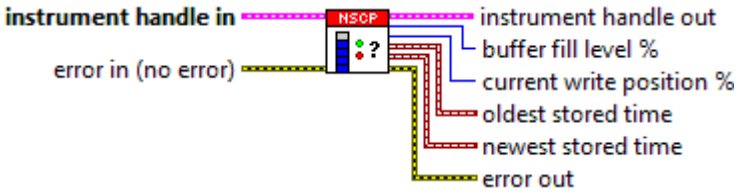


Figure 34: netSCOPE.lvlib:Get Capture Buffer State.vi

instrument handle in identifies a particular instrument session.

error in (no error) describes error conditions that occur before this VI runs. The default input of this cluster is no error. If an error already occurred, this VI returns the value of **error in** in **error out**. The VI runs normally only if no incoming error exists. Otherwise, the VI passes the **error in** value to **error out**. The **error in** cluster contains the following parameters:

- **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a negative error code. If **status** is FALSE, **code** is 0 or a warning code.
- **source** identifies where an error occurred. The source string includes the name of the VI that produced the error, what inputs are in error, and how to eliminate the error.

instrument handle out has the same value as the **instrument handle**.

buffer fill level % Current ring buffer fill level in percent.

current write position % Current write position of the ring buffer in percent.

oldest stored time Current write position of the ring buffer in percent.

newest stored time Time stamp of the oldest captured and stored datagram in the ring buffer.

error out contains error information. If **error in** indicates that an error occurred before this VI or function ran, **error out** contains the same error information. Otherwise, it describes the error status that this VI or function produces.

- **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a negative error code. If **status** is FALSE, **code** is 0 or a warning code.

source identifies where an error occurred. The source string includes the name of the VI that produced the error, what inputs are in error, and how to eliminate the error.

3.5.2 netSCOPE.lvlib:Get Ethernet Port State.vi

- Gets the state of the Ethernet capture ports of the instrument.

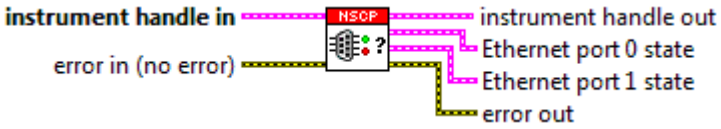


Figure 35: netSCOPE.lvlib:Get Ethernet Port State.vi

instrument handle in identifies a particular instrument session.

error in (no error) describes error conditions that occur before this VI runs. The default input of this cluster is no error. If an error already occurred, this VI returns the value of **error in** in **error out**. The VI runs normally only if no incoming error exists. Otherwise, the VI passes the **error in** value to **error out**. The **error in** cluster contains the following parameters:

status is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.

code is the error or warning code. The default is 0. If **status** is TRUE, **code** is a negative error code. If **status** is FALSE, **code** is 0 or a warning code.

source identifies where an error occurred. The source string includes the name of the VI that produced the error, what inputs are in error, and how to eliminate the error.

instrument handle out has the same value as the **instrument handle**.

Ethernet port 0 state Current state information of the netSCOPE instrument on the port 0.

Ethernet link up Current link state of this port.

FALSE: link down

TRUE: link up

Correct frames Total number of successfully received Ethernet frames.

RX_ER errors Total number of faulty received Ethernet frames.

Alignment errors Number of frames with alignment errors (1 additional nibble received)

FCS errors Number of frames with a bad FCS (including short frames with a bad FCS).

Frame length errors Number of frames with invalid Ethernet frame length.

SFD errors Number of Ethernet frames with a SFD (Start of frame delimiter) errors.

Preamble length errors Number of frames with invalid length of preamble.


Average bus load Bus load on this port in percent.


Ethernet speed Current Ethernet speed 10MBit or 100MBit.


Ethernet port 1 state Current state information of the netSCOPE instrument on the port 1.


Ethernet link up Current link state of this port. 0: link down

1: link up


Correct frames Total number of successfully received Ethernet frames.


RX_ER errors Total number of faulty received Ethernet frames.


Alignment errors Number of frames with alignment errors (1 additional nibble received)


FCS errors Number of frames with a bad FCS (including short frames with a bad FCS).


Frame length errors Number of frames with invalid Ethernet frame length.


SFD errors Number of Ethernet frames with a SFD (Start of frame delimiter) errors.


Preamble length errors Number of frames with invalid length of preamble.


Average bus load Bus load on this port in percent.


Ethernet speed Current Ethernet speed 10MBit or 100MBit.


error out contains error information. If **error in** indicates that an error occurred before this VI or function ran, **error out** contains the same error information. Otherwise, it describes the error status that this VI or function produces.


status is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.


code is the error or warning code. The default is 0. If **status** is TRUE, **code** is a negative error code. If **status** is FALSE, **code** is 0 or a warning code.


source identifies where an error occurred. The source string includes the name of the VI that produced the error, what inputs are in error, and how to eliminate the error.

3.5.3 netSCOPE.lvlib:Get Instrument State.vi

- Gets the current state of the instrument.

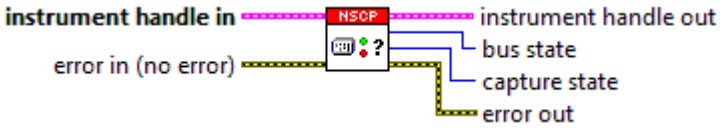


Figure 36: netSCOPE.lvlib:Get Instrument State.vi

instrument handle in identifies a particular instrument session.

instrument handle out has the same value as the **instrument handle.**

bus state Current bus state of the instrument.
- Instrument is connected to the bus and ready to capture the data.
- Instrument is disconnected from the bus (the capturing of the data is no longer possible).
- Instrument is connected to the bus but stopped because an internal instrument error occurred (the capturing of data is not longer possible).

capture state Current capture state of the instrument.
- Data capturing is started.
- Data capturing is stopped.

error out contains error information. If **error in** indicates that an error occurred before this VI or function ran, **error out** contains the same error information. Otherwise, it describes the error status that this VI or function produces.

status is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.

code is the error or warning code. The default is 0. If **status** is TRUE, **code** is a negative error code. If **status** is FALSE, **code** is 0 or a warning code.

source identifies where an error occurred. The source string includes the name of the VI that produced the error, what inputs are in error, and how to eliminate the error.

3.5.4 netSCOPE.lvlib:Set Bus Active.vi

- Activates the physical connection to the communication bus or network.

- This is a prerequisite before calling the Start Capture VI.

Figure 37: netSCOPE.lvlib:Set Bus Active.vi

instrument handle in identifies a particular instrument session.

instrument handle out has the same value as the **instrument handle**.

error out contains error information. If **error in** indicates that an error occurred before this VI or function ran, **error out** contains the same error information. Otherwise, it describes the error status that this VI or function produces.

status is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.

code is the error or warning code. The default is 0. If **status** is TRUE, **code** is a negative error code. If **status** is FALSE, **code** is 0 or a warning code.

source identifies where an error occurred. The source string includes the name of the VI that produced the error, what inputs are in error, and how to eliminate the error.

3.5.5 netSCOPE.lvlib:Set Bus Inactive.vi

- Deactivates the physical connection to the communication bus or network.

- If a capture is running it must be stopped via the Stop Capture VI first.

Figure 38: netSCOPE.lvlib:Set Bus Inactive.vi

instrument handle in identifies a particular instrument session.

instrument handle out has the same value as the **instrument handle**.

error out contains error information. If **error in** indicates that an error occurred before this VI or function ran, **error out** contains the same error information. Otherwise, it describes the error status that this VI or function produces.

status is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.

code is the error or warning code. The default is 0. If **status** is TRUE, **code** is a negative error code. If **status** is FALSE, **code** is 0 or a warning code.

source identifies where an error occurred. The source string includes the name of the VI that produced the error, what inputs are in error, and how to eliminate the error.

3.5.6 netSCOPE.lvlib:Start Capture.vi

- Starts the capture task for process data values.

- This requires the bus to be activated via the Set Bus Active VI.

Figure 39: netSCOPE.lvlib:Start Capture.vi

instrument handle in identifies a particular instrument session.

instrument handle out has the same value as the **instrument handle**.

error out contains error information. If **error in** indicates that an error occurred before this VI or function ran, **error out** contains the same error information. Otherwise, it describes the error status that this VI or function produces.

status is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.

code is the error or warning code. The default is 0. If **status** is TRUE, **code** is a negative error code. If **status** is FALSE, **code** is 0 or a warning code.

source identifies where an error occurred. The source string includes the name of the VI that produced the error, what inputs are in error, and how to eliminate the error.

3.5.7 netSCOPE.lvlib:Stop Capture.vi

- Stops the capture task for process data values.

- After stopping no new data will be stored in the capture ring buffer, but yet captured data is still available.

Figure 40: netSCOPE.lvlib:Stop Capture.vi

instrument handle in identifies a particular instrument session.

instrument handle out has the same value as the **instrument handle**.

error out contains error information. If **error in** indicates that an error occurred before this VI or function ran, **error out** contains the same error information. Otherwise, it describes the error status that this VI or function produces.

- **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a negative error code. If **status** is FALSE, **code** is 0 or a warning code.
- **source** identifies where an error occurred. The source string includes the name of the VI that produced the error, what inputs are in error, and how to eliminate the error.

3.6 Public - Configure - EtherCAT

3.6.1 netSCOPE.lvlib:EtherCAT Configure Detection.vi

- Configures how the EtherCAT direction detection for input / output data works.

- It can be set either to automatic detection or to a user specified fixed configuration.

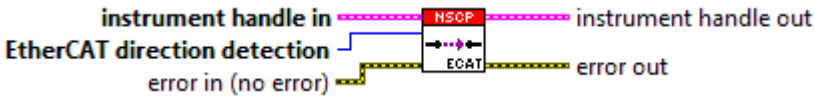


Figure 41: netSCOPE.lvlib:EtherCAT Configure Detection.vi

instrument handle in identifies a particular instrument session.

EtherCAT direction detection Specified EtherCAT direction of input/output data.
- Port 0 input / Port 1 output
- Port 0 output / Port 1 input
- Automatic

error in (no error) describes error conditions that occur before this VI runs. The default input of this cluster is no error. If an error already occurred, this VI returns the value of **error in** in **error out**. The VI runs normally only if no incoming error exists. Otherwise, the VI passes the **error in** value to **error out**. The **error in** cluster contains the following parameters:

status is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.

code is the error or warning code. The default is 0. If **status** is TRUE, **code** is a negative error code. If **status** is FALSE, **code** is 0 or a warning code.

source identifies where an error occurred. The source string includes the name of the VI that produced the error, what inputs are in error, and how to eliminate the error.

instrument handle out has the same value as the **instrument handle**.

error out contains error information. If **error in** indicates that an error occurred before this VI or function ran, **error out** contains the same error information. Otherwise, it describes the error status that this VI or function produces.

status is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.

code is the error or warning code. The default is 0. If **status** is TRUE, **code** is a negative error code. If **status** is FALSE, **code** is 0 or a warning code.

source identifies where an error occurred. The source string includes the name of the VI that produced the error, what inputs are in error, and how to eliminate the error.

3.7 Public – Configure

3.7.1 netSCOPE.lvlib:Register Notification Event Handler.vi

- Registers an user event which will be issued every time a status or error notification is generated by an instrument (see section *Notification Events* on page 56).

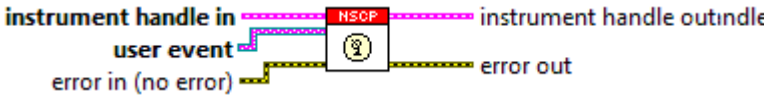


Figure 42: netSCOPE.lvlib:Register Notification Event Handler.vi

instrument handle in identifies a particular instrument session.

user event User event handle that allows receiving the notification events from the backend. This user event is generated by the LabVIEW specific “Create User Event.vi”.

notification event entry Notification event structure. This structure should be specified when creating the user.

Time Stamp Timestamp of the notification event.

Event Notification event identifier.

Additional Information Additional information notification event dependent.

error in (no error) describes error conditions that occur before this VI runs. The default input of this cluster is no error. If an error already occurred, this VI returns the value of **error in** in **error out.** The VI runs normally only if no incoming error exists. Otherwise, the VI passes the **error in** value to **error out.** The **error in** cluster contains the following parameters:

status is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.

code is the error or warning code. The default is 0. If **status** is TRUE, **code** is a negative error code. If **status** is FALSE, **code** is 0 or a warning code.

source identifies where an error occurred. The source string includes the name of the VI that produced the error, what inputs are in error, and how to eliminate the error.

instrument handle out has the same value as the **instrument handle**.

error out contains error information. If **error in** indicates that an error occurred before this VI or function ran, **error out** contains the same error information. Otherwise, it describes the error status that this VI or function produces.

status is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.

code is the error or warning code. The default is 0. If **status** is TRUE, **code** is a negative error code. If **status** is FALSE, **code** is 0 or a warning code.

source identifies where an error occurred. The source string includes the name of the VI that produced the error, what inputs are in error, and how to eliminate the error.

3.7.1.1 Notification Events

Notification Event	Additional Information	Description
Instrument connected to bus	----------	The instrument was successfully connected on the bus and is ready to capture of data.
Instrument connected to bus but stopped by error	Instrument Error: 0xXXXXXXX	The capture of data was automatically stopped because an internal instrument error occurred. Use "Set Bus Inactive.vi" to disconnect the instrument from the bus.
Instrument disconnected from bus	----------	The instrument was successfully disconnected from bus. Capturing of data is not longer possible.
Instrument access failed	Instrument Error: 0xXXXXXXX	Access to internal instrument functionality failed. The instrument specific error code is transmitted in "Additional Information" field. Detailed description of error codes see in the netSCOPE documentation.
License for this product is not activated	----------	The license for this product is not activated. The capturing of data was automatically stopped.
Ringbuffer out of memory	----------	Internal backend module "Ringbuffer" reports no free system memory.
Ecat out of memory	----------	Internal backend module "EtherCAT Decoder" reports
Incompatible data type	----------	The data type of read data is incompatible to defined data types in LabVIEW. The execution of "Read Data.vi" is broken.

Table 3: Important Error Codes, possible Causes and Troubleshooting

3.7.2 netSCOPE.lvlib:Ringbuffer Configuration.vi

- Configures the ringbuffer storage size in Megabytes and location.

- RAM storage location does not need a save path.

- HDD storage location needs a save path to be specified.

Note, that HDD storage is most likely less performant than RAM storage.

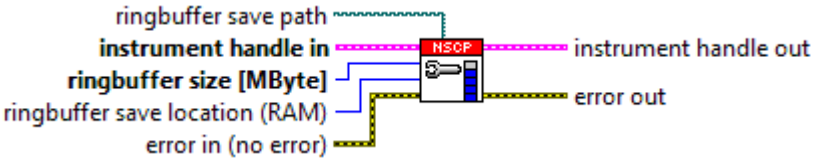


Figure 43: netSCOPE.lvlib:Ringbuffer Configuration.vi

ringbuffer save path Path of the ring buffer location on the HDD.

instrument handle in identifies a particular instrument session.

ringbuffer size [MByte] Size of the ring buffer.

ringbuffer save location (RAM) Specifies where the ring buffer should be created RAM / HDD.

error in (no error) describes error conditions that occur before this VI runs. The default input of this cluster is no error. If an error already occurred, this VI returns the value of **error in** in **error out**. The VI runs normally only if no incoming error exists. Otherwise, the VI passes the **error in** value to **error out**. The **error in** cluster contains the following parameters:

- **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a negative error code. If **status** is FALSE, **code** is 0 or a warning code.
- **source** identifies where an error occurred. The source string includes the name of the VI that produced the error, what inputs are in error, and how to eliminate the error.

instrument handle out has the same value as the **instrument handle**.

error out contains error information. If **error in** indicates that an error occurred before this VI or function ran, **error out** contains the same error information. Otherwise, it describes the error status that this VI or function produces.

- **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a negative error code. If **status** is FALSE, **code** is 0 or a warning code.

source identifies where an error occurred. The source string includes the name of the VI that produced the error, what inputs are in error, and how to eliminate the error.

3.7.3 netSCOPE.lvlib:Unregister Notification Event Handler.vi

- Unregisters the instruments notification user event handler. No more user events will be issued.

Figure 44: netSCOPE.lvlib:Unregister Notification Event Handler.vi

instrument handle in identifies a particular instrument session.

instrument handle out has the same value as the **instrument handle**.

error out contains error information. If **error in** indicates that an error occurred before this VI or function ran, **error out** contains the same error information. Otherwise, it describes the error status that this VI or function produces.

- **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a negative error code. If **status** is FALSE, **code** is 0 or a warning code.
- **source** identifies where an error occurred. The source string includes the name of the VI that produced the error, what inputs are in error, and how to eliminate the error.

3.8 Public - Data

3.8.1 netSCOPE.lvlib:Read Data.vi

- Reads a variables values from the capture data ring buffer.

- Reading is limited to the time span given, ‘read from timestamp’ must always be specified.

- The maximum amount of data that is read out is implicitly specified by the input arrays size.

- All input arrays (timestamp list, value list, status list) must have the same size.

- The value list contains elements which must be preinitialized with the LabVIEW data type and its expected size.

- The amount of actually read values is returned by "count of read values", if this value is smaller than the array size, the rest of the arrays elements do not contain correct data and must be ignored. The VI does not resize the arrays automatically.

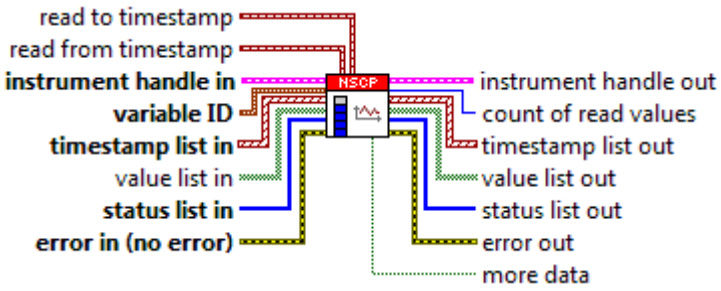


Figure 45: netSCOPE.lvlib:Read Data.vi

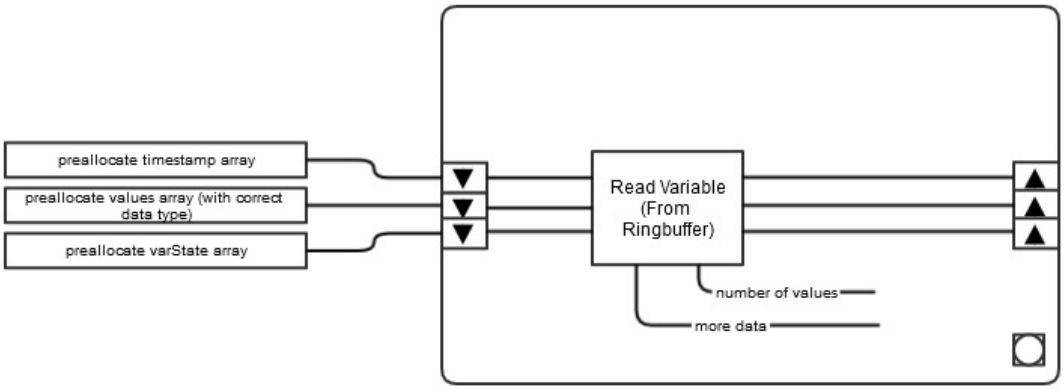


Figure 46: Variable Data Management

read to timestamp Timestamp where the reading process should be aborted.

read from timestamp Timestamp where the reading process should be started.

instrument handle in identifies a particular instrument session.

variable ID Identifier of the variable that should be read from the ring buffer.

Variable ID Value Value of the variable ID.

Instance Handle Internal driver information.

timestamp list in Timestamp array with pre-initialized size.

value list in Value array with pre-initialized size. The data type of this array should match the expected value data type.

status list in State array with pre-initialized size.

error in (no error) describes error conditions that occur before this VI runs. The default input of this cluster is no error. If an error already occurred, this VI returns the value of **error in** in **error out**. The VI runs normally only if no incoming error exists. Otherwise, the VI passes the **error in** value to **error out**. The **error in** cluster contains the following parameters:

- **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a negative error code. If **status** is FALSE, **code** is 0 or a warning code.
- **source** identifies where an error occurred. The source string includes the name of the VI that produced the error, what inputs are in error, and how to eliminate the error.

instrument handle out has the same value as the **instrument handle**.

count of read values Counter that indicates how many values were read from the ringbuffer (relevant for timestamp list out, value list out and status list out arrays).

timestamp list out Array of read timestamps.

value list out Array of read values.

status list out Array of read value states.

error out contains error information. If **error in** indicates that an error occurred before this VI or function ran, **error out** contains the same error information. Otherwise, it describes the error status that this VI or function produces.

- **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a negative error code. If **status** is FALSE, **code** is 0 or a warning code.
- **source** identifies where an error occurred. The source string includes the name of the VI that produced the error, what inputs are in error, and how to eliminate the error.

more data Indicates if reading values has been completed in the specified time span (reading from timestamp - reading to timestamp). - True: Reading the data has not been completed because the size of pre-initialized arrays (timestamp list in, value list in and status list in)

was not sufficient.
- False: Reading the data has been completed.

3.9 Public - Utility - EtherCAT

3.9.1 netSCOPE.lvlib:EtherCAT Add or Modify Variable.vi

- Adds or modifies the EtherCAT-specific definition of the given variable.
- EtherCAT-specific VI.

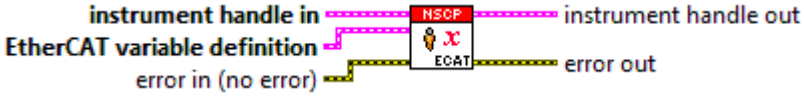


Figure 47: netSCOPE.lvlib:EtherCAT Add or Modify Variable.vi

instrument handle in identifies a particular instrument session.

instrument handle out has the same value as the **instrument handle.**

EtherCAT variable definition Cluster of specific EtherCAT variable definition.

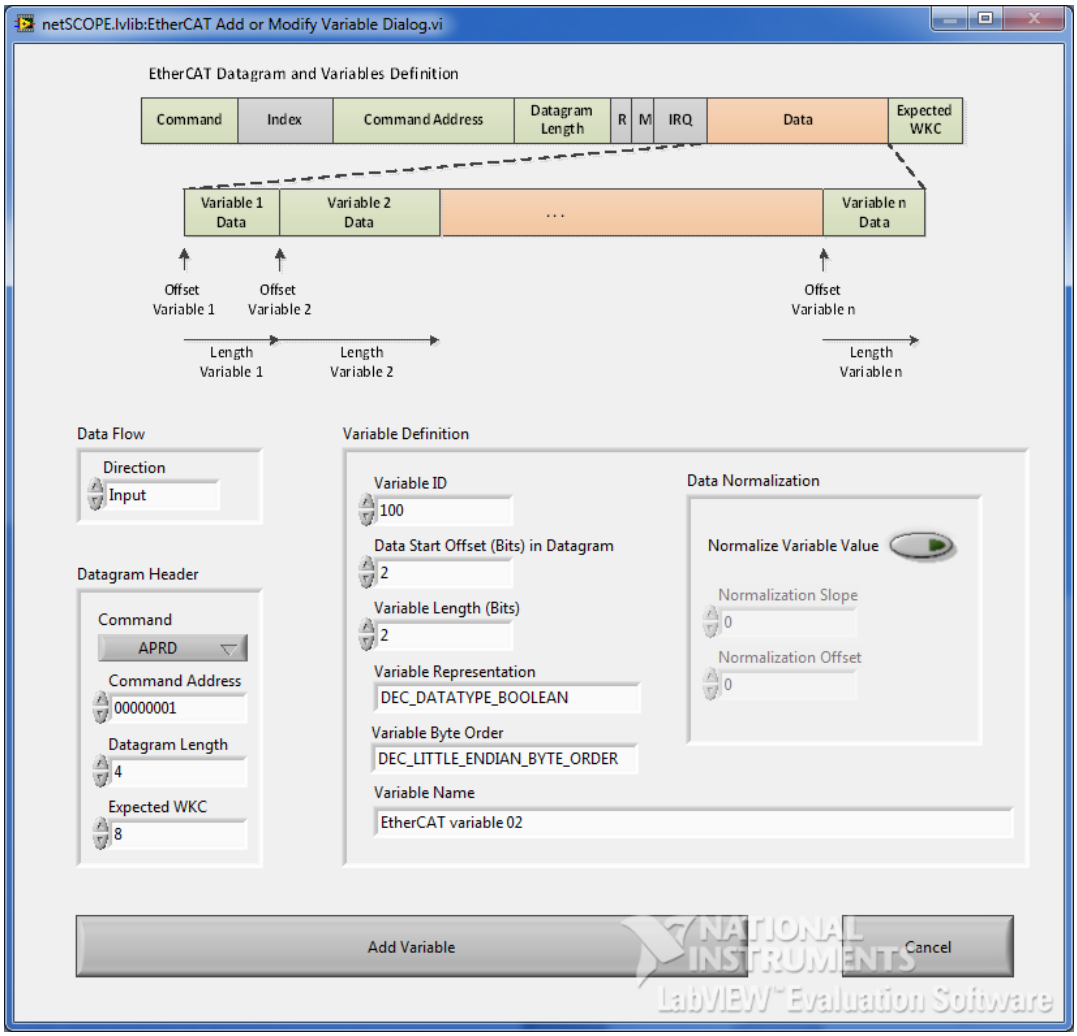


Figure 48: EtherCAT Datagram and Variable Definition

Command (Datagram Header Area) This selection specifies the EtherCAT command executed in the EtherCAT datagram. Corresponds to the EtherCAT command specified in the Command field of the EtherCAT datagram.

The following EtherCAT commands are defined in the EtherCAT specification:

Code	Command
APRD	Auto increment physical read
APWR	Auto increment physical write
APRW	Auto increment physical read write
FPRD	Configured address physical read
FPWR	Configured address physical write
FPRW	Configured address physical read write
BRD	Broadcast read
BWR	Broadcast write
BRW	Broadcast read write
LRD	Logical read
LWR	Logical write
LRW	Logical read write
ARMW	Auto increment physical read multiple write
FRMW	Configured address physical read multiple write

Table 4: EtherCAT Commands

Command Address (Datagram Header Area) This value is specified as a hexadecimal address. Corresponds to the address specified in the Command field of the EtherCAT datagram address.
The allowed value range extends from 0x0 to 0xFFFFFFFF.

Working Counter (Datagram Header Area) Expected value of the working counter. Corresponds to the value specified in the field "Expected WKC" of the EtherCAT datagram.
The allowed value range extends from 0 to 65535.

Command Length (Datagram Header Area) Length of the datagram (expressed as the number of bits in the datagram). Corresponds to the length specified in the "Datagram Length" field of the EtherCAT datagram.

Variable Bit-Address (Variable Definition Area - Data Start Offset (Bits) in Datagram) This value indicates the offset of the variable currently to be defined relative to the beginning of the data field (data) in the EtherCAT datagram. It is expressed as the number of bits counted from the memory location of the first bit of the first variable of the data field.
If the variable currently to be defined is the first in the data field, the value is 0.

Generic Variable Definition Cluster of generic variable definition.

Variable Data Type (Variable Definition Area) This value specifies the data type of the variable currently to be defined. The following data types are supported in EtherCAT:

Data Type	Description	Number of Bits	Range of Value
BOOLBIT	'0': FALSE '1': TRUE	1	
BIT8		8	
SINT	Short integer	8	-128 … 127

Data Type	Description	Number of Bits	Range of Value
INT	Integer	16	-32768 … 32767
INT24		24	
DINT	Double integer	32	-231 … +231-1
INT40		40	
INT48		48	
INT56		56	
LINT	Long integer	64	
USINT	Unsigned short integer	8	0 … 255
UINT	Unsigned integer/Word	16	0 …65535
UINT24		24	
UDINT	Unsigned double integer	32	0 … +232-1
UINT40		40	
UINT48		48	
UINT56		56	
ULINT	Unsigned long integer	64	0 … +264-1
REAL	Floating point	32	
LREAL	Long Float	64	
VISIBLE_STRING	Visible string (1 octet per character)	8*n	
OCTET_STRING	Sequence of octets	8*(n+1)	
UNICODE_STRING	Sequence of UNIT	16*(n+1)	

Table 5: Supported Data Types in EtherCAT

Variable Direction (Data Flow Area) Indicates the signal direction and can either have the value "input" or "output".

Variable Byte Order (Variable Definition Area) This value indicates the byte order used in the internal representation of the variable currently to be defined. Possible values are:
DEC_LITTLE_ENDIAN_BYTE_ORDER
(Intel format, which means: the most significant byte comes first, the less significant comes last).
DEC_BIG_ENDIAN_BYTE_ORDER
(Motorola format, which means: the less significant byte comes first, the most significant byte comes last).

Name (Variable Definition Area) This value indicates the full name of the variable currently to be defined.

Normalization Slope (Data Normalization Area) The data can be normalized if necessary by multiplication by a normalization factor and adding a normalization offset.
In this field, the normalization factor can be specified.

Normalization Offset (Data Normalization Area) The data can be normalized if necessary by multiplication by a normalization factor and adding a normalization offset.
In this field, the normalization offset can be specified.

Variable ID (Variable Definition Area) ID that uniquely identifies the variable.

Variable ID Value Value of the variable ID.

Instance Handle Internal driver information.

Scaling active 'Normalization Slope' and 'Normalization Offset' and are only considered when 'Scaling active' is TRUE.

Variable Bit-Length (Variable Definition Area) This value specifies the length of the variable currently to be defined specified as number of the bits.

error in (no error) describes error conditions that occur before this VI runs. The default input of this cluster is no error. If an error already occurred, this VI returns the value of **error in** in **error out**. The VI runs normally only if no incoming error exists. Otherwise, the VI passes the **error in** value to **error out**. The **error in** cluster contains the following parameters:

status is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.

code is the error or warning code. The default is 0. If **status** is TRUE, **code** is a negative error code. If **status** is FALSE, **code** is 0 or a warning code.

source identifies where an error occurred. The source string includes the name of the VI that produced the error, what inputs are in error, and how to eliminate the error.

instrument handle out has the same value as the **instrument handle**.

error out contains error information. If **error in** indicates that an error occurred before this VI or function ran, **error out** contains the same error information. Otherwise, it describes the error status that this VI or function produces.

status is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.

code is the error or warning code. The default is 0. If **status** is TRUE, **code** is a negative error code. If **status** is FALSE, **code** is 0 or a warning code.

source identifies where an error occurred. The source string includes the name of the VI that produced the error, what inputs are in error, and how to eliminate the error.

3.9.2 netSCOPE.lvlib:EtherCAT Get Specific Variable Definition.vi

- Gets the EtherCAT-specific definition of a variable.

- EtherCAT-specific VI.

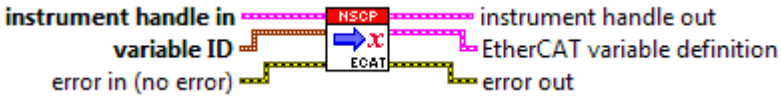


Figure 49: netSCOPE.lvlib:EtherCAT Get Specific Variable Definition.vi

instrument handle in identifies a particular instrument session.

variable ID Variable identifier.

Variable ID Value Value of the variable ID.

Instance Handle Internal driver information.

error in (no error) describes error conditions that occur before this VI runs. The default input of this cluster is no error. If an error already occurred, this VI returns the value of **error in** in **error out**. The VI runs normally only if no incoming error exists. Otherwise, the VI passes the **error in** value to **error out**. The **error in** cluster contains the following parameters:

status is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.

code is the error or warning code. The default is 0. If **status** is TRUE, **code** is a negative error code. If **status** is FALSE, **code** is 0 or a warning code.

source identifies where an error occurred. The source string includes the name of the VI that produced the error, what inputs are in error, and how to eliminate the error.

EtherCAT variable definition Cluster of specific EtherCAT variable definition.

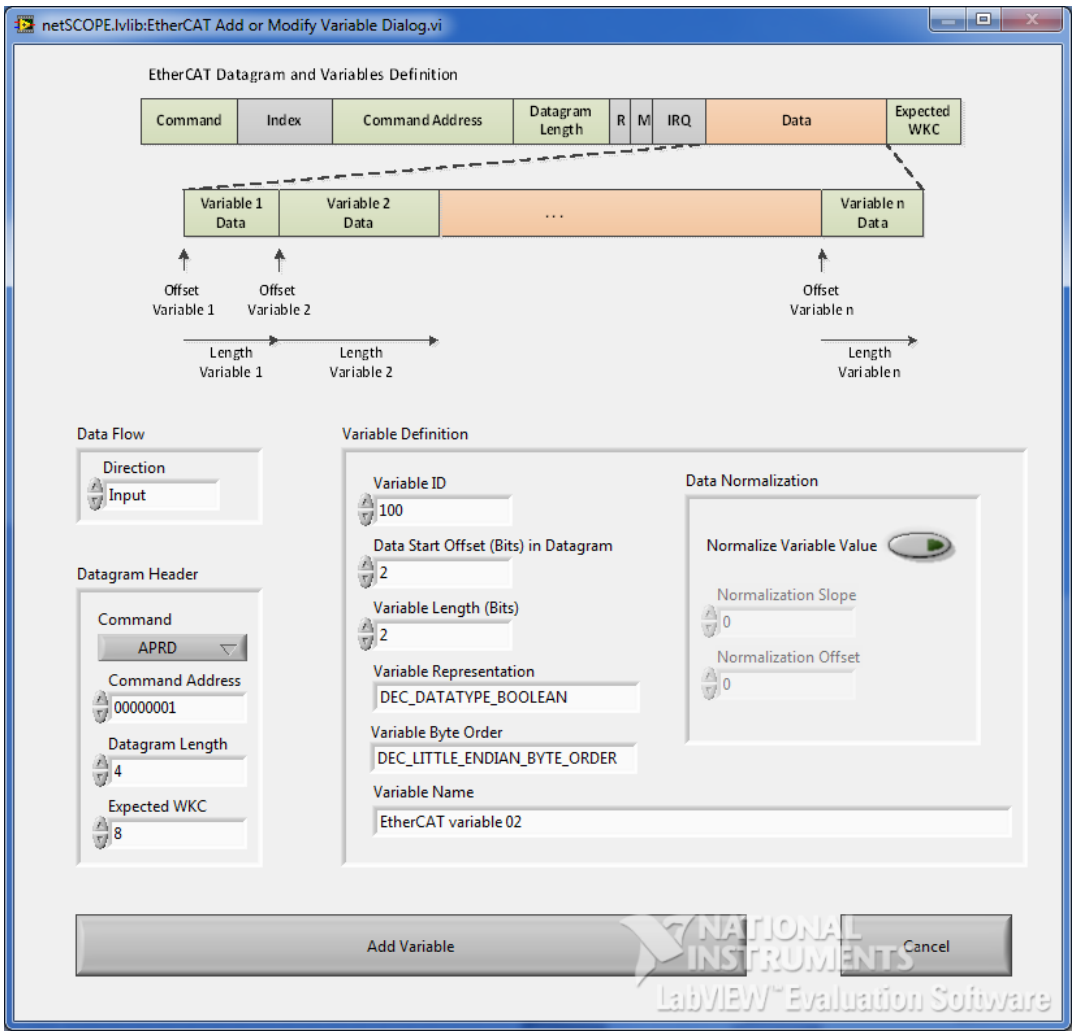


Figure 50: EtherCAT Datagram and Variable Definition

U32 **Command** (Datagram Header Area) This selection specifies the EtherCAT command executed in the EtherCAT datagram. Corresponds to the EtherCAT command specified in the Command field of the EtherCAT datagram.
The following EtherCAT commands are defined in the EtherCAT specification:

Code	Command
APRD	Auto increment physical read
APWR	Auto increment physical write
APRW	Auto increment physical read write
FPRD	Configured address physical read
FPWR	Configured address physical write
FPRW	Configured address physical read write
BRD	Broadcast read
BWR	Broadcast write
BRW	Broadcast read write
LRD	Logical read
LWR	Logical write
LRW	Logical read write
ARMW	Auto increment physical read multiple write

Code	Command
FRMW	Configured address physical read multiple write

Table 6: EtherCAT Commands

Command Address (Datagram Header Area) This value is specified as a hexadecimal address. Corresponds to the address specified in the Command field of the EtherCAT datagram address.
The allowed value range extends from 0x0 to 0xFFFFFFFF.

Working Counter (Datagram Header Area) Expected value of the working counter. Corresponds to the value specified in the field "Expected WKC" of the EtherCAT datagram.
The allowed value range extends from 0 to 65535.

Command Length (Datagram Header Area) Length of the datagram (expressed as the number of bits in the datagram). Corresponds to the length specified in the "Datagram Length" field of the EtherCAT datagram.

Variable Bit-Address (Variable Definition Area - Data Start Offset (Bits) in Datagram) This value indicates the offset of the variable currently to be defined relative to the beginning of the data field (data) in the EtherCAT datagram. It is expressed as the number of bits counted from the memory location of the first bit of the first variable of the data field.
If the variable currently to be defined is the first in the data field, the value is 0.

Generic Variable Definition Cluster of generic variable definition.

Variable Data Type (Variable Definition Area) This value specifies the data type of the variable currently to be defined. The following data types are supported in EtherCAT:

Data Type	Description	Number of Bits	Range of Value
BOOLBIT	'0': FALSE '1': TRUE	1	
BIT8		8	
SINT	Short integer	8	-128 … 127
INT	Integer	16	-32768 … 32767
INT24		24	
DINT	Double integer	32	-231 … +231-1
INT40		40	
INT48		48	
INT56		56	
LINT	Long integer	64	
USINT	Unsigned short integer	8	0 … 255
UINT	Unsigned integer/Word	16	0 …65535
UINT24		24	
UDINT	Unsigned double integer	32	0 … +232-1
UINT40		40	
UINT48		48	
UINT56		56	
ULINT	Unsigned long integer	64	0 … +264-1

Data Type	Description	Number of Bits	Range of Value
REAL	Floating point	32	
LREAL	Long Float	64	
VISIBLE_STRING	Visible string (1 octet per character)	8*n	
OCTET_STRING	Sequence of octets	8*(n+1)	
UNICODE_STRING	Sequence of UNIT	16*(n+1)	

Table 7: Supported Data Types in EtherCAT

Variable Direction (Data Flow Area) Indicates the signal direction and can either have the value "input" or "output".

Variable Byte Order (Variable Definition Area) This value indicates the byte order used in the internal representation of the variable currently to be defined. Possible values are:
DEC_LITTLE_ENDIAN_BYTE_ORDER
(Intel format, which means: the most significant byte comes first, the less significant comes last).
DEC_BIG_ENDIAN_BYTE_ORDER
(Motorola format, which means: the less significant byte comes first, the most significant byte comes last).

Name (Variable Definition Area) This value indicates the full name of the variable currently to be defined.

Normalization Slope (Data Normalization Area) The data can be normalized if necessary by multiplication by a normalization factor and adding a normalization offset.
In this field, the normalization factor can be specified.

Normalization Offset (Data Normalization Area) The data can be normalized if necessary by multiplication by a normalization factor and adding a normalization offset.
In this field, the normalization offset can be specified.

Variable ID (Variable Definition Area) ID that uniquely identifies the variable.

Variable ID Value Value of the variable ID.

Instance Handle Internal driver information.

Scaling active 'Normalization Slope' and 'Normalization Offset' and are only considered when 'Scaling active' is TRUE.

Variable Bit-Length (Variable Definition Area) This value specifies the length of the variable currently to be defined specified as number of the bits.

error out contains error information. If **error in** indicates that an error occurred before this VI or function ran, **error out** contains the same error information. Otherwise, it describes the error status that this VI or function produces.

status is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.

code is the error or warning code. The default is 0. If **status** is TRUE, **code** is a negative error code. If **status** is FALSE, **code** is 0 or a warning code.

source identifies where an error occurred. The source string includes the name of the VI that produced the error, what inputs are in error, and how to eliminate the error.

3.9.3 netSCOPE.lvlib:EtherCAT Load ENI File.vi

- Loads all variables from the given ENI file.

- EtherCAT-specific VI.

Figure 51: netSCOPE.lvlib:EtherCAT Load ENI File.vi

instrument handle in identifies a particular instrument session.

ENI file path Path to the EtherCAT specific ENI file.

error in (no error) describes error conditions that occur before this VI runs. The default input of this cluster is no error. If an error already occurred, this VI returns the value of **error in** in **error out**. The VI runs normally only if no incoming error exists. Otherwise, the VI passes the **error in** value to **error out**. The **error in** cluster contains the following parameters:

- **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a negative error code. If **status** is FALSE, **code** is 0 or a warning code.
- **source** identifies where an error occurred. The source string includes the name of the VI that produced the error, what inputs are in error, and how to eliminate the error.

instrument handle out has the same value as the **instrument handle**.

error out contains error information. If **error in** indicates that an error occurred before this VI or function ran, **error out** contains the same error information. Otherwise, it describes the error status that this VI or function produces.

- **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a negative error code. If **status** is FALSE, **code** is 0 or a warning code.
- **source** identifies where an error occurred. The source string includes the name of the VI that produced the error, what inputs are in error, and how to eliminate the error.

3.10 Public - Utility

3.10.1 netSCOPE.lvlib:Error Descriptions.vi

- This VI returns all netSCOPE-specific error codes and descriptions.

- Useful to be connected to the General Error Handler VIs [user-defined codes] and [user-defined descriptions] inputs.

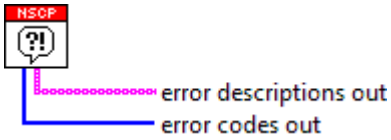


Figure 52: netSCOPE.lvlib:Error Descriptions.vi

error descriptions out Error code.

error codes out Error short description.

3.10.2 netSCOPE.lvlib:Get Generic Variable Definition.vi

- Gets the generic, system-independent definition of a variable.

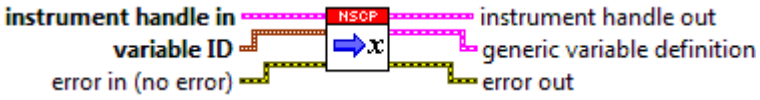


Figure 53: netSCOPE.lvlib:Get Generic Variable Definition.vi

instrument handle in identifies a particular instrument session.

variable ID Variable identifier.

Variable ID Value Value of the variable ID.

Instance Handle Internal driver information.

error in (no error) describes error conditions that occur before this VI runs. The default input of this cluster is no error. If an error already occurred, this VI returns the value of **error in** in **error out**. The VI runs normally only if no incoming error exists. Otherwise, the VI passes the **error in** value to **error out**. The **error in** cluster contains the following parameters:

status is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.

code is the error or warning code. The default is 0. If **status** is TRUE, **code** is a negative error code. If **status** is FALSE, **code** is 0 or a warning code.

source identifies where an error occurred. The source string includes the name of the VI that produced the error, what inputs are in error, and how to eliminate the error.

instrument handle out has the same value as the **instrument handle**.

generic variable definition Cluster of generic variable definition.

Variable Data Type (Variable Definition Area) This value specifies the data type of the variable currently to be defined. The following data types are supported in EtherCAT:

Data Type	Description	Number of Bits	Range of Value
BOOLBIT	'0': FALSE '1': TRUE	1	
BIT8		8	
SINT	Short integer	8	-128 … 127
INT	Integer	16	-32768 … 32767
INT24		24	
DINT	Double integer	32	-231 … +231-1
INT40		40	
INT48		48	
INT56		56	
LINT	Long integer	64	
USINT	Unsigned short integer	8	0 … 255

Data Type	Description	Number of Bits	Range of Value
UINT	Unsigned integer/Word	16	0 …65535
UINT24		24	
UDINT	Unsigned double integer	32	0 … +232-1
UINT40		40	
UINT48		48	
UINT56		56	
ULINT	Unsigned long integer	64	0 … +264-1
REAL	Floating point	32	
LREAL	Long Float	64	
VISIBLE_STRING	Visible string (1 octet per character)	8*n	
OCTET_STRING	Sequence of octets	8*(n+1)	
UNICODE_STRING	Sequence of UNIT	16*(n+1)	

Table 8: Supported Data Types in EtherCAT

Variable Direction (Data Flow Area) Indicates the signal direction and can either have the value "input" or "output".

Variable Byte Order (Variable Definition Area) This value indicates the byte order used in the internal representation of the variable currently to be defined. Possible values are:
DEC_LITTLE_ENDIAN_BYTE_ORDER
(Intel format, which means: the most significant byte comes first, the less significant comes last).
DEC_BIG_ENDIAN_BYTE_ORDER
(Motorola format, which means: the less significant byte comes first, the most significant byte comes last).

Name (Variable Definition Area) This value indicates the full name of the variable currently to be defined.

Normalization Slope (Data Normalization Area) The data can be normalized if necessary by multiplication by a normalization factor and adding a normalization offset.
In this field, the normalization factor can be specified.

Normalization Offset (Data Normalization Area) The data can be normalized if necessary by multiplication by a normalization factor and adding a normalization offset.
In this field, the normalization offset can be specified.

Variable ID (Variable Definition Area) ID that uniquely identifies the variable.

Variable ID Value Value of the variable ID.

Instance Handle Internal driver information.

Scaling active 'Normalization Slope' and 'Normalization Offset' and are only considered when 'Scaling active' is TRUE.

error out contains error information. If **error in** indicates that an error occurred before this VI or function ran, **error out** contains the same error information. Otherwise, it describes the error status that this VI or function produces.

status is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.

code is the error or warning code. The default is 0. If **status** is TRUE, **code** is a negative error code. If **status** is FALSE, **code** is 0 or a warning code.

source identifies where an error occurred. The source string includes the name of the VI that produced the error, what inputs are in error, and how to eliminate the error.

3.10.3 netSCOPE.lvlib:Get Instrument List.vi

- Returns a list of all instruments of the system.

Figure 54: netSCOPE.lvlib:Get Instrument List.vi

system handle in Valid system handle generated by “Initialize System.vi” (see section *netSCOPE.lvlib:Initialize System.vi* page 86).

error in (no error) describes error conditions that occur before this VI runs. The default input of this cluster is no error. If an error already occurred, this VI returns the value of **error in** in **error out**. The VI runs normally only if no incoming error exists. Otherwise, the VI passes the **error in** value to **error out**. The **error in** cluster contains the following parameters:

- **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a negative error code. If **status** is FALSE, **code** is 0 or a warning code.
- **source** identifies where an error occurred. The source string includes the name of the VI that produced the error, what inputs are in error, and how to eliminate the error.

system handle out Valid system handle generated by “Initialize System.vi” (see section *netSCOPE.lvlib:Initialize System.vi* page 86).

instruments list A list of available instruments found on the system.

- **instruments list** Structure with device information.
  - **Instruments list** Device number of the instrument.
    - **Device number**
      - 7330100 NSCP-C100-RE\50
      - 7330101 NSCP-C100-RE\50E
      - 7330102 NSCP-C100-RE\70E
      - 7330103 NSCP-C100-RE\80
      - 7330105 NSCP-C100-RE\90E
    - **Serial number** Serial number of the instrument.
    - **Device Class** Device class of the instrument.
    - **Name** Instrument name.

error out contains error information. If **error in** indicates that an error occurred before this VI or function ran, **error out** contains the same error information. Otherwise, it describes the error status that this VI or function produces.

- **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.

code is the error or warning code. The default is 0. If **status** is TRUE, **code** is a negative error code. If **status** is FALSE, **code** is 0 or a warning code.

source identifies where an error occurred. The source string includes the name of the VI that produced the error, what inputs are in error, and how to eliminate the error.

3.10.4 netSCOPE.lvlib:Get Variable IDs by Name.vi

- Returns a list of variables IDs for all variables which's name matches the given regular expression.

Figure 55: netSCOPE.lvlib:Get Variable IDs by Name.vi

instrument handle in identifies a particular instrument session.

regular expression (.*)Variable name.

error in (no error) describes error conditions that occur before this VI runs. The default input of this cluster is no error. If an error already occurred, this VI returns the value of **error in** in **error out**. The VI runs normally only if no incoming error exists. Otherwise, the VI passes the **error in** value to **error out**. The **error in** cluster contains the following parameters:

- **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a negative error code. If **status** is FALSE, **code** is 0 or a warning code.
- **source** identifies where an error occurred. The source string includes the name of the VI that produced the error, what inputs are in error, and how to eliminate the error.

instrument handle out has the same value as the **instrument handle.**

variable ID list An array of variable identifiers that match the given variable name.

- **variable ID** Identifier of the variable that should be read from the ring buffer.
  - **Variable ID Value** Value of the variable ID.
  - **Instance Handle** Internal driver information.

error out contains error information. If **error in** indicates that an error occurred before this VI or function ran, **error out** contains the same error information. Otherwise, it describes the error status that this VI or function produces.

- **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a negative error code. If **status** is FALSE, **code** is 0 or a warning code.
- **source** identifies where an error occurred. The source string includes the name of the VI that produced the error, what inputs are in error, and how to eliminate the error.

3.10.5 netSCOPE.lvlib:Identify.vi

- Blinks the given instruments LEDs for identification.

Figure 56: netSCOPE.lvlib:Identify.vi

system handle in Valid system handle generated by “Initialize System.vi” (see section *netSCOPE.lvlib:Initialize System.vi* page 86).

instrument name Instrument name (for example „netSCOPE0“). The instrument name can be extracted from the instrument list generated by “Get Instrument List.vi” (see section *netSCOPE.lvlib:Get Instrument List.vi* page 77.

error in (no error) describes error conditions that occur before this VI runs. The default input of this cluster is no error. If an error already occurred, this VI returns the value of **error in** in **error out**. The VI runs normally only if no incoming error exists. Otherwise, the VI passes the **error in** value to **error out**. The **error in** cluster contains the following parameters:

- **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a negative error code. If **status** is FALSE, **code** is 0 or a warning code.
- **source** identifies where an error occurred. The source string includes the name of the VI that produced the error, what inputs are in error, and how to eliminate the error.

system handle out Valid system handle generated by “Initialize System.vi” (see section *netSCOPE.lvlib:Initialize System.vi* page 86).

error out contains error information. If **error in** indicates that an error occurred before this VI or function ran, **error out** contains the same error information. Otherwise, it describes the error status that this VI or function produces.

- **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a negative error code. If **status** is FALSE, **code** is 0 or a warning code.
- **source** identifies where an error occurred. The source string includes the name of the VI that produced the error, what inputs are in error, and how to eliminate the error.

3.10.6 netSCOPE.lvlib:Remove Variable.vi

- Removes a variable definition from the list of known variables.

- If the variable is removed its values may not be read out anymore by the Get Data VI.

Figure 57: netSCOPE.lvlib:Remove Variable.vi

instrument handle in identifies a particular instrument session.

variable ID Identifier of the variable that should be removed from the known variables list.

Variable ID Value Value of the variable ID.

Instance Handle Internal driver information.

error in (no error) describes error conditions that occur before this VI runs. The default input of this cluster is no error. If an error already occurred, this VI returns the value of **error in** in **error out**. The VI runs normally only if no incoming error exists. Otherwise, the VI passes the **error in** value to **error out**. The **error in** cluster contains the following parameters:

status is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.

code is the error or warning code. The default is 0. If **status** is TRUE, **code** is a negative error code. If **status** is FALSE, **code** is 0 or a warning code.

source identifies where an error occurred. The source string includes the name of the VI that produced the error, what inputs are in error, and how to eliminate the error.

instrument handle out has the same value as the **instrument handle**.

error out contains error information. If **error in** indicates that an error occurred before this VI or function ran, **error out** contains the same error information. Otherwise, it describes the error status that this VI or function produces.

status is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.

code is the error or warning code. The default is 0. If **status** is TRUE, **code** is a negative error code. If **status** is FALSE, **code** is 0 or a warning code.

source identifies where an error occurred. The source string includes the name of the VI that produced the error, what inputs are in error, and how to eliminate the error.

3.10.7 netSCOPE.lvlib:Revision Query.vi

- Queries version information of all netSCOPE software and hardware modules.

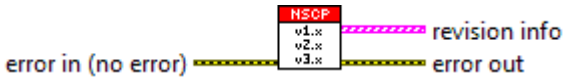


Figure 58: netSCOPE.lvlib:Revision Query.vi

error in (no error) describes error conditions that occur before this VI runs. The default input of this cluster is no error. If an error already occurred, this VI returns the value of **error in** in **error out.** The VI runs normally only if no incoming error exists. Otherwise, the VI passes the **error in** value to **error out.** The **error in** cluster contains the following parameters:

- **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a negative error code. If **status** is FALSE, **code** is 0 or a warning code.
- **source** identifies where an error occurred. The source string includes the name of the VI that produced the error, what inputs are in error, and how to eliminate the error.

revision info Includes version information of individual system components (driver version, ringbuffer version, ….).

- **Revision Info Entry** Structure with version information.
  - **Component Name** Name of the component.
  - **Component Details** Version information.

error out contains error information. If **error in** indicates that an error occurred before this VI or function ran, **error out** contains the same error information. Otherwise, it describes the error status that this VI or function produces.

- **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a negative error code. If **status** is FALSE, **code** is 0 or a warning code.
- **source** identifies where an error occurred. The source string includes the name of the VI that produced the error, what inputs are in error, and how to eliminate the error.

3.11 Public

3.11.1 netSCOPE.lvlib:Close Instrument.vi

- Closes an instrument and returns the system handle the instrument belongs to.

- This will discard all configuration and captured ring buffer data for this instrument.

- The Instrument will not be accessible anymore unless it is reopened via the Initialize Instrument VI.

Figure 59: netSCOPE.lvlib:Close Instrument.vi

instrument handle in identifies a particular instrument session.

system handle in Valid system handle generated by “Initialize System.vi” (see section *netSCOPE.lvlib:Initialize System.vi* page 86).

error out contains error information. If **error in** indicates that an error occurred before this VI or function ran, **error out** contains the same error information. Otherwise, it describes the error status that this VI or function produces.

status is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.

code is the error or warning code. The default is 0. If **status** is TRUE, **code** is a negative error code. If **status** is FALSE, **code** is 0 or a warning code.

source identifies where an error occurred. The source string includes the name of the VI that produced the error, what inputs are in error, and how to eliminate the error.

3.11.2 netSCOPE.lvlib:Close System.vi

- Closes a system.

- All instruments belonging to this system will be closed automatically, all captured ringbuffer data in this system will be discarded.

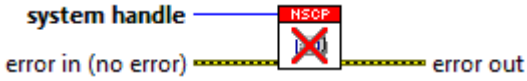


Figure 60: netSCOPE.lvlib:Close System.vi

system handle in Valid system handle generated by “Initialize System.vi” (see section *netSCOPE.lvlib:Initialize System.vi* page 86).

error in (no error) describes error conditions that occur before this VI runs. The default input of this cluster is no error. If an error already occurred, this VI returns the value of **error in** in **error out**. The VI runs normally only if no incoming error exists. Otherwise, the VI passes the **error in** value to **error out**. The **error in** cluster contains the following parameters:

- **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a negative error code. If **status** is FALSE, **code** is 0 or a warning code.
- **source** identifies where an error occurred. The source string includes the name of the VI that produced the error, what inputs are in error, and how to eliminate the error.

error out contains error information. If **error in** indicates that an error occurred before this VI or function ran, **error out** contains the same error information. Otherwise, it describes the error status that this VI or function produces.

- **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a negative error code. If **status** is FALSE, **code** is 0 or a warning code.
- **source** identifies where an error occurred. The source string includes the name of the VI that produced the error, what inputs are in error, and how to eliminate the error.

3.11.3 netSCOPE.lvlib:Initialize Instrument.vi

- Initialize one instrument specified by its name.

- This VI must be called once before using any instrument specific VIs.

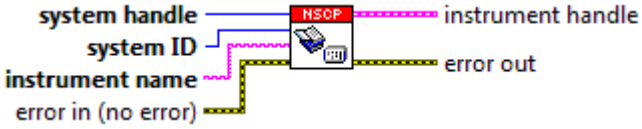


Figure 61: netSCOPE.lvlib:Initialize Instrument.vi

system handle in Valid system handle generated by “Initialize System.vi” (see section *netSCOPE.lvlib:Initialize System.vi* page 86).

system ID Target system identifier.

instrument name Name of the instrument that should be initialized.

error in (no error) describes error conditions that occur before this VI runs. The default input of this cluster is no error. If an error already occurred, this VI returns the value of **error in** in **error out**. The VI runs normally only if no incoming error exists. Otherwise, the VI passes the **error in** value to **error out**. The **error in** cluster contains the following parameters:

- **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a negative error code. If **status** is FALSE, **code** is 0 or a warning code.
- **source** identifies where an error occurred. The source string includes the name of the VI that produced the error, what inputs are in error, and how to eliminate the error.

instrument handle out has the same value as the **instrument handle**.

error out contains error information. If **error in** indicates that an error occurred before this VI or function ran, **error out** contains the same error information. Otherwise, it describes the error status that this VI or function produces.

- **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
- **code** is the error or warning code. The default is 0. If **status** is TRUE, **code** is a negative error code. If **status** is FALSE, **code** is 0 or a warning code.
- **source** identifies where an error occurred. The source string includes the name of the VI that produced the error, what inputs are in error, and how to eliminate the error.

3.11.4 netSCOPE.lvlib:Initialize System.vi

- Initialized the netSCOPE system.

- This is the first VI to be called before any other netSCOPE VI is useable.

Figure 62: netSCOPE.lvlib:Initialize System.vi

error in (no error) describes error conditions that occur before this VI runs. The default input of this cluster is no error. If an error already occurred, this VI returns the value of **error in** in **error out**. The VI runs normally only if no incoming error exists. Otherwise, the VI passes the **error in** value to **error out**. The **error in** cluster contains the following parameters:

status is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.

code is the error or warning code. The default is 0. If **status** is TRUE, **code** is a negative error code. If **status** is FALSE, **code** is 0 or a warning code.

source identifies where an error occurred. The source string includes the name of the VI that produced the error, what inputs are in error, and how to eliminate the error.

system handle in Valid system handle generated by “Initialize System.vi” (see section *netSCOPE.lvlib:Initialize System.vi* page 86).

error out contains error information. If **error in** indicates that an error occurred before this VI or function ran, **error out** contains the same error information. Otherwise, it describes the error status that this VI or function produces.

status is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.

code is the error or warning code. The default is 0. If **status** is TRUE, **code** is a negative error code. If **status** is FALSE, **code** is 0 or a warning code.

source identifies where an error occurred. The source string includes the name of the VI that produced the error, what inputs are in error, and how to eliminate the error.

4 Error Codes

4.1 Overview Error Codes

Error Codes		Type	Range
LabVIEW Errors		Errors	5000 … 5305
netANALYZER / netSCOPE Windows Device Driver Errors [1]	Generic Errors	Warnings	0x00000000 … 0x80200009
	Toolkit Errors	Warnings	0x80210001 … 0x8021000E
	Driver Errors	Warnings	0x80220001 … 0x80220012
Capturing Errors		Errors	0x00000000 … 0xC0770000

Table 9: Overview Error Codes and Ranges

4.2 LabVIEW Errors Description

Value	Error Code (Definition)	Short Description
5000	Duplicate variable ID	During manual configuration of variable was assigned a duplicate variable ID.
5001	Instrument not opened	No instrument initialized ("Open Instrument.vi" wasn't executed or during the execution failed).
5002	Instance mismatch	Wrong instrument instance.
5003	Target system not supported	The selected target system isn't supported of netSCOPE.
5004	Error loading backend DLLs	Loading of internal DLL components is failed.
5005	Error loading file	Loading of ENI file is failed. Invalid path was specified or imported file has incorrect format.
5006	Variable ID not found	The variable with the specified ID wasn't found in variable list.
5007	Array sizes don't match	The size of timestamp-, data- and valid-array don't match.
5008	Incompatible data type	The variable data type don't match with defined LabVIEW data types (see "Variable Data Type.ctl").
5009	Out of memory	No more free system memory available.
5010	Unknown interface command	Unknown backend interface command.
5011	Invalid instance handle	Invalid instance handle = 0x00000000. ("Open Instrument.vi" wasn't executed or during the execution failed).
5012	System not initialized	System not initialized ("Initialize System.vi" wasn't executed or during the execution failed).
5014	Invalid parameter	Invalid parameter in the calling VI.
5100	Instrument access failed	Access to internal instrument functionality is failed. It triggers a notification event which contains a detailed description of this error.
5101	Instrument invalid parameter	Wrong configuration parameter. (internal instrument error)
5102	Instrument not found	Instrument with passed name wasn't found. (internal instrument error)
5103	Instrument IOCTL failed	General error at sending of IOCTL. (internal instrument error)
5200	Ringbuffer failed	Internal ringbuffer module error. (internal instrument error)
5201	Ringbuffer get time invalid	Invalid read data time span, from time is greater as to time. (internal ringbuffer error)
5202	Ringbuffer invalid parameter	Access to EtherCAT-Decoder failed because of an invalid transfer parameter. (internal ringbuffer error)
5203	Ringbuffer out of memory	No more free system memory available. (internal ringbuffer error)

Value	Error Code (Definition)	Short Description
5204	Ringbuffer invalid var group ID	Read data failed (invalid variable group ID).
5205	Ringbuffer thread creating failed	Internal initialization of ringbuffer failed. Data capturing not possible. (internal ringbuffer error)
5206	Ringbuffer not stopped	Configuration of ringbuffer not possible because it wasn't stopped.
5300	Ecat invalid parameter	Access to EtherCAT-Decoder failed because of an invalid transfer parameter. (internal EtherCAT error)
5301	Ecat out of memory	No more free system memory available. (internal EtherCAT error)
5302	Ecat load ringbuffer failed	Load of internal ringbuffer module failed. (internal EtherCAT error)
5303	Ecat wrong ringbuffer version	Incorrect version number of loaded ringbuffer module. (internal EtherCAT error)
5304	Ecat ringbuffer not loaded	Access to ringbuffer module not possible, because module wasn't loaded. (internal EtherCAT error)
5305	Ecat invalid backend handle	Access to EtherCAT-Decoder failed because of an invalid handle. (internal EtherCAT error)

Table 10: LabVIEW Errors Description

4.3 Generic Errors

Value	Error Code (Definition)	Description	Possible Causes	Troubleshooting
0x80200003	NETANA_OUT_OF_MEMORY	Out of memory	The available storage capacity of central memory is full.	Upgrade the storage capacity of the central memory.
				Close all other open applications on the PC.

Table 11: Generic Errors Description

4.4 Toolkit Errors

Value	Error Code (Definition)	Description
0x80210001	NETANA_TKIT_INITIALIZATION_FAILED	Toolkit initialization failed
0x80210002	NETANA_DMABUFFER_CREATION_FAILED	Creation of DMA buffers failed
0x80210003	NETANA_HWRESET_ERROR	Error during hardware reset of device
0x80210004	NETANA_CHIP_NOT_SUPPORTED	Chip type is not supported by toolkit
0x80210005	NETANA_DOWNLOAD_FAILED	Download of Bootloader / Firmware failed
0x80210006	NETANA_FW_START_FAILED	Error starting firmware
0x80210007	NETANA_DEV_MAILBOX_FULL	Device mailbox is full
0x80210008	NETANA_DEV_NOT_READY	Device not ready
0x80210009	NETANA_DEV_MAILBOX_TOO_SHORT	Mailbox is too short for packet
0x8021000A	NETANA_DEV_GET_NO_PACKET	No packet available
0x8021000B	NETANA_BUFFER_TOO_SHORT	Given buffer is too short

Value	Error Code (Definition)	Description
0x8021000C	NETANA_TRANSFER_TIMEOUT	Transfer timed out
0x8021000D	NETANA_IRQEVENT_CREATION_FAILED	Error creating interrupt events
0x8021000E	NETANA_IRQLOCK_CREATION_FAILED	Error creating internal IRQ locks

Table 12: Toolkit Errors Description

4.5 Driver Errors

Value	Error Code (Definition)	Description	Possible Causes	Troubleshooting
0x80220002	NETANA_DRIVER_NOT_RUNNING	netANALYZER / netSCOPE Windows Device Driver is not running	The netANALYZER / netSCOPE Windows Device Driver is not installed.	Install the netANALYZER / netSCOPE Windows Device Driver.
			The netANALYZER / netSCOPE Windows Device Driver is installed, but the netANALYZER hardware is not installed in the PC or not connected.	The netANALYZER hardware installed in the PC and connect.
			The netSCOPE device is disabled in the device manager.	Enable the netSCOPE device in Device Manager.
0x80220003	NETANA_DEVICE_NOT_FOUND	Device with the given name does not exist	The netSCOPE device was removed from the PC during operation of the netSCOPE software.	Update the netSCOPE Software device list.
0x80220004	NETANA_DEVICE_STILL_OPEN	Device is still in use by another application	The netSCOPE device was already opened in another application.	Close the netSCOPE device in the other application or select another device.

Table 13: Toolkit Errors Description

4.6 Capturing Errors

Value	Error Code (Definition)	Description	Possible Causes	Troubleshooting
0xC0660004	NETANA_ CAPTURE_ ERROR_NO_ DMACHANNEL	No free DMA channel available. Probably host is too slow	The data load of the capturing is too high.	Check whether the hard disk of the PC is fast enough to save the captured data. The theoretical maximum load is 50 MB/s. Reduce the load of the data to be captured.
0xC0660005	NETANA_ CAPTURE_ ERROR_URX_ OVERFLOW	XC buffer overflow (URX overflow)	Occurs because a non IEEE802.3 conform traffic is captured (e.g. too short frames, too small IFG).	Record only IEEE802.3-compliant message traffic.
0xC066000B	NETANA_ CAPTURE_ER ROR_NO_HOS TBUFFER	No free DMA buffer available.	Host is too slow to handle data efficiently.	Check whether the hard disk of the PC is fast enough to save the captured data. The theoretical maximum load is 50 MB/s. Reduce the load of the data to be captured.
0xC066000C	NETANA_ CAPTURE_ ERROR_NO_IN TRAMBUFFER	Internal capture buffer overflow	No free INTRAM Firmware is out of memory resources and is unable to buffer more data. This may also be caused by a slow file system or a slow application	Check whether the hard disk of the PC is fast enough to save the captured data. The theoretical maximum load is 50 MB/s. Reduce the load of the data to be captured.
0xC066000D	NETANA_ CAPTURE_ ERROR_FIFO_ FULL	Firmware is out of FIFO resources and is unable to buffer more data.	This may also be caused by a slow file system or a slow application	Optimize your application or use a faster PC.
0xC0770000	NETANA_ CAPTURE_ ERROR_ DRIVER_FILE_ FULL	End of capture file reached. Driver has stopped capturing.	The error is triggered when the ringbuffer mode is not activated and the end of capture file is reached.	No error

Table 14: Capturing Errors Description

5 Appendix

5.1 References

[1] Driver Manual netANALYZER API, Windows XP/Vista/7/8, V1.x

5.2 List of Figures

5.3 List of Tables

5.4 Glossary

ENI

EtherCAT **N**etwork **I**nformation

The EtherCAT Network Information (ENI) Specification describes the structure of ENI files using XML schemas.

EtherCAT

A communication system for industrial Ethernet designed and developed by Beckhoff Automation GmbH.

LabVIEW

Laboratory **V**irtual **I**nstrumentation **E**ngineering **W**orkbench

LabVIEW is a graphical programming system from National Instruments. It is the leading graphical programming language for measurement and automation applications.

netSCOPE

Hilscher's netSCOPE is a tool to capture network traffic from Real-Time Ethernet systems and to display data content for analysis purposes.

VI

Virtual Instrument

LabVIEW programs/subroutines are called virtual instruments (VIs).

5.5 Contacts

Headquarters

Germany
Hilscher Gesellschaft für
Systemautomation mbH
Rheinstrasse 15
65795 Hattersheim
Phone: +49 (0) 6190 9907-0
Fax: +49 (0) 6190 9907-50
E-Mail: info@hilscher.com

Support
Phone: +49 (0) 6190 9907-99
E-Mail: de.support@hilscher.com

Subsidiaries

China
Hilscher Systemautomation (Shanghai) Co. Ltd.
200010 Shanghai
Phone: +86 (0) 21-6355-5161
E-Mail: info@hilscher.cn

Support
Phone: +86 (0) 21-6355-5161
E-Mail: cn.support@hilscher.com

France
Hilscher France S.a.r.l.
69500 Bron
Phone: +33 (0) 4 72 37 98 40
E-Mail: info@hilscher.fr

Support
Phone: +33 (0) 4 72 37 98 40
E-Mail: fr.support@hilscher.com

India
Hilscher India Pvt. Ltd.
New Delhi - 110 065
Phone: +91 11 26915430
E-Mail: info@hilscher.in

Italy
Hilscher Italia S.r.l.
20090 Vimodrone (MI)
Phone: +39 02 25007068
E-Mail: info@hilscher.it

Support
Phone: +39 02 25007068
E-Mail: it.support@hilscher.com

Japan
Hilscher Japan KK
Tokyo, 160-0022
Phone: +81 (0) 3-5362-0521
E-Mail: info@hilscher.jp

Support
Phone: +81 (0) 3-5362-0521
E-Mail: jp.support@hilscher.com

Korea
Hilscher Korea Inc.
Seongnam, Gyeonggi, 463-400
Phone: +82 (0) 31-789-3715
E-Mail: info@hilscher.kr

Switzerland
Hilscher Swiss GmbH
4500 Solothurn
Phone: +41 (0) 32 623 6633
E-Mail: info@hilscher.ch

Support
Phone: +49 (0) 6190 9907-99
E-Mail: ch.support@hilscher.com

USA
Hilscher North America, Inc.
Lisle, IL 60532
Phone: +1 630-505-5301
E-Mail: info@hilscher.us

Support
Phone: +1 630-505-5301
E-Mail: us.support@hilscher.com