IEC 61850 Driver

Filename	IEC61850.dll
Manufacturer	
Devices	Servers (IEDs) compatible with IEC 61850 Standard (ED1 and ED2)
Protocol	IEC 61850 MMS over Ethernet TCP/IP
Version	3.0.55
Last Update	10/06/2025
Platform	Win32
Dependencies	IOKit v2.00
Superblock Readings	No
Level	31298 (31201 with a Power license)

Introduction

The IEC 61850 Driver communicates with protection relays and other devices using the IEC61850 protocol over Ethernet TCP/IP (MMS). This Driver allows:

- Communication with several devices on the same Driver
- Importing Tags from devices or via SCL configuration files
- Support for messages without confirmation of **Reports**, **Buffered** or **Unbuffered**
- Polling of variables not belonging to **Reports**
- Support for quality and timestamp information with a one-millisecond precision
- Collecting oscillography files in **COMTRADE** format

Please check the next documents, provided with this manual, for more information about this Driver's functionality:

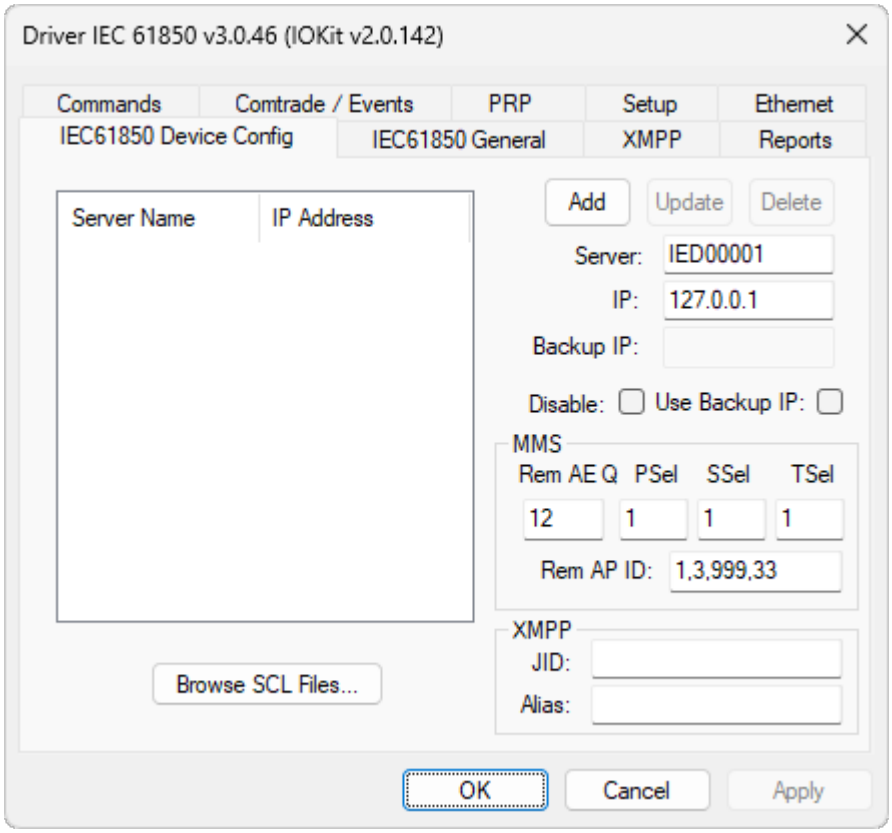
- Elipse IEC 61850 Client Driver PIXIT Ed2.pdf
- Elipse IEC 61850 Client Driver TICS Ed2.pdf
- Elipse IEC 61850 Client Driver PICS Ed2.pdf
- Elipse IEC 61850 Client Driver MICS Ed2.pdf

Configuration

The **[P]** configuration parameters of this Driver are not used. All settings are performed on this Driver's configuration window. The configuration tabs are described on the next topics.

IEC61850 Device Config Tab

This tab allows defining devices (Servers) to establish communication.



IEC61850 Device Config tab

The available options on this tab are described on the next table.

Available options on the IEC61850 Device Config tab

OPTION	DESCRIPTION
Browse SCL Files	Tags from relays can be created online, that is, when users can communicate with relays, or offline by importing SCL files. This option opens a window to select ICD or SCD files to import. After selecting a file, for each description of relay found, an identification is created on the list of servers, and a file with the description of Logical Devices (LD) and Logical Nodes (LN) of a relay is also created, which can be imported to an application by using the **Tag Browser** window. In case there are no SCL files, users can configure each relay directly by using the **Add**, **Update**, and **Delete** options
Server	Informs the name of a device. Used only to identify a device for this Driver, mapping it to an IP address
IP	Informs an IP address of a device. Optionally, users can inform a TCP/IP port, if it is not the default port of MMS protocol, using the format **[0-255].[0-255].[0-255].[0-255]:[0-65535]**, such as 192.168.0.10:102
Backup IP	Informs a backup IP address of a device, if available. Use the same syntax of the **IP** option
PSel (Presentation Selector)	Selection value for the device used by the OSI presentation layer (ISO/IEC 8823). It must be checked, if fixed, or informed in the device's configuration. Its value is usually 1 (one)

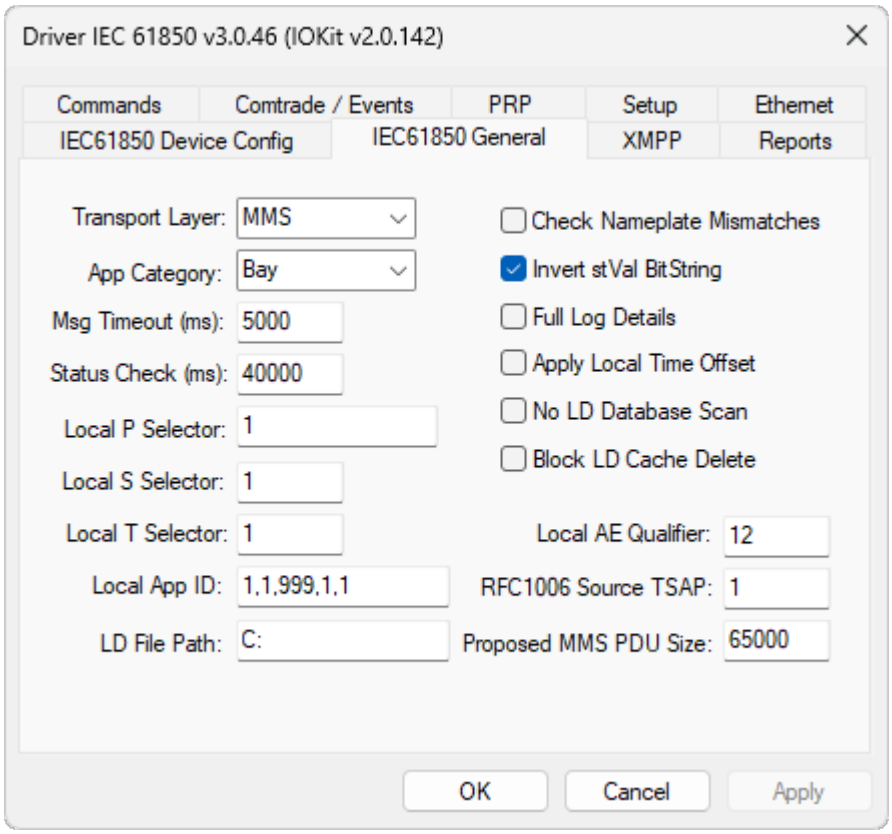
OPTION	DESCRIPTION
SSel (Session Selector)	Selection value for the device used by the OSI session layer (ISO/IEC 8327). It must be checked, if fixed, or informed in the device's configuration. Its values is usually 1 (one)
TSel (Transport Selector)	Selection value for the device used by the OSI transport layer (ISO/IEC 8073). It must be checked, if fixed, or informed in the device's configuration. Its value is usually 1 (one)
Rem AP ID (Remote Application Process Identification)	Identifier used by the OSI link layer OSI (ISO/OSI 8650), in **ASN.1** (*Abstract Syntax Notation 1*) format. Indicates the format of data adopted by the **AARQ** (*Association Request*) function. Its value is usually 1,1,999,1,1 (iso.1.999.1.1)
Rem AE Qual (Remote Application Entity Qualifier)	Identifier used by the communication layer as a formatter. It must be checked, if fixed, or informed in the device's configuration. Its value is usually 12
Disable	Disables this device. Therefore, when starting this Driver, there is no communication with this device
Use Backup IP	Informs whether the backup IP address is used
Revision	Standard Revision for this IED. The available values for this option are **Auto**: This Driver tries to identify automatically, **v1/v2**: Forces revision 1 or 2, or **v2.1**: Forces revision 2.1
XMPP	Function not available in this version

NOTE

The topic **Limitations of this Driver** contains information about the maximum number of IEDs supported by this Driver.

IEC61850 General Tab

Allows defining all other behaviors of this Driver.



IEC61850 General tab

The available options on this tab are described on the next table.

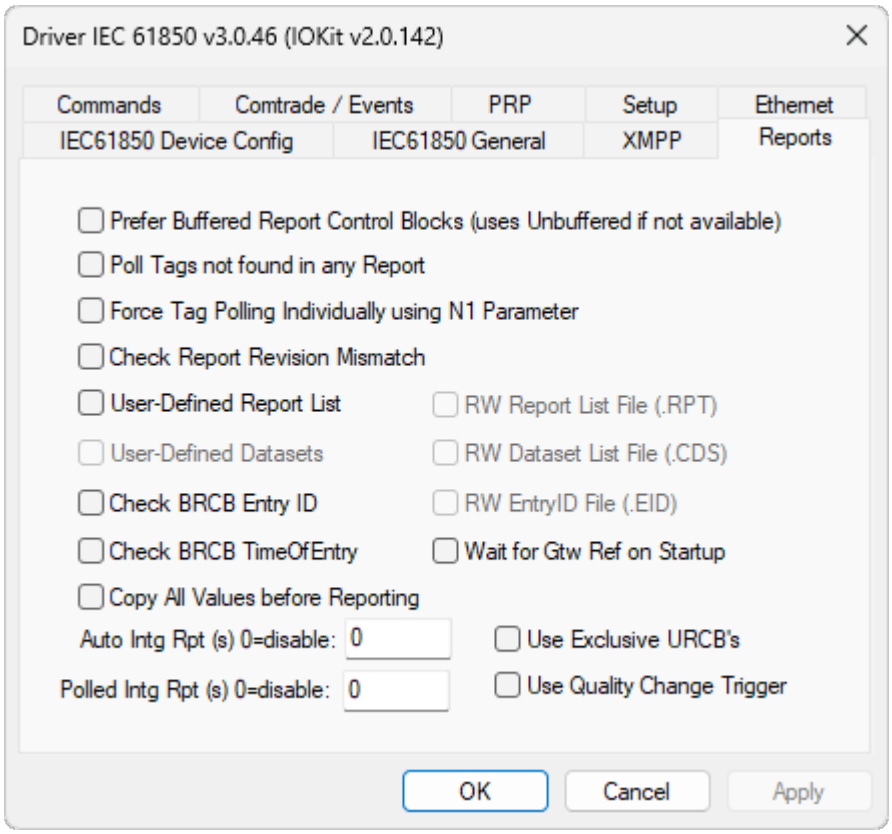
Available options on the IEC61850 General tab

OPTION	DESCRIPTION
Transport Layer	Defines the transport layer between the default **MMS (IEC 61850-8-1)** and **XMPP (IEC 61850-8-2)**
App Category	To send commands, this Driver uses information from this option to fill the **OrCat** (*Origin Category*) property, which specifies the type of application sending a command, for security reasons or for solving conflicts of commands. The available values are **Bay**, **Station**, **Remote**, or **Maintenance**
Msg Timeout (ms)	Time to wait for a response of a command or a full message, which can be formed by several intermediate messages. The time to wait for each byte or intermediate message is defined on the **Setup** tab. For more information, please check topic **Documentation of I/O Interfaces**
Status Check (ms)	Time interval to send status messages, which must be answered by an IED. If there is no response, connection health can be checked, forcing a disconnection and reconnection of an IED. The value of this option must be greater than the value of the **Msg Timeout (ms)** option and less than the disconnection interval configured on the **Setup** tab. For more information, please check topic **Documentation of I/O Interfaces**
Local P Selector	Selection value for this Driver, used by the OSI presentation layer (ISO/IEC 8823)

OPTION	DESCRIPTION
Local S Selector	Selection value for this Driver, used by the OSI session layer (ISO/IEC 8327)
Local T Selector	Selection value for this Driver, used by the OSI transportation layer (ISO/IEC 8073)
Local App ID Local Application Process Identification	Identifier used by the OSI link layer (ISO/OSI 8650), in **ASN.1** (*Abstract Syntax Notation 1*) format. Indicates a data format adopted by the **AARQ** (*Association Request*) function. Its value is usually 1,1,999,1,1 (iso.1.999.1.1)
LD File Path	Default directory where this Driver stores descriptive files for each Logical Device found, to improve the initialization process. In the next initialization, if a file is found on the specified directory, whose name is given by the format **SERVER_LDNAME.LD**, where **SERVER** is the name of the device and **LDNAME** is the name of the Logical Device, then this LD is described based on this file. This Driver provides several ways to detect changes on the database of an IED, to update these files
Check Nameplate Mismatches	Enables checking identification parameters of an IED (nameplate) when starting communication, to compare whether there were any changes. If true, a cache update, or rebuilding LD files, is performed
Invert stVal BitString	Interpretation of *bitstrings* with two or more bits can be changed by this option, by changing the meaning of **Open** or **Closed** statuses in an application. Default behavior is reached by deselecting this option
Full Log Details	Enables inserting in this Driver's log, enabled on the **Setup** tab, detailed information about the notification of events for any Tag. For more information, please check topic **Documentation of I/O Interfaces**
Apply Local Offset to Timestamps	Timestamps adopted by the IEC 61850 standard always refer to **UTC** (*Coordinated Universal Time*) standard. By using this option, users indicate that this Driver must apply a local offset, that is, timezone and daylight saving time, to the timestamp sent by a device
No LD Database Scan	Informs that no Logical Device or Logical Node request must be processed for IEDs. This option can be used when this Driver is used only to transfer files
Block LD Cache Delete	If the initialization process is incomplete, or when any inconsistency is detected between cache files (LD) and the current database of an IED or Server, this Driver deletes and updates cache files. Enable this option to block file removal
Local AE Qual - Local Application Entity Qualifier	Identifier used by the link layer as a formatter. Its value is usually 12
RFC 1006 Source TSAP	This Driver uses RFC 1006 specification as a way to transport ISO packets over TCP. To do so, users must inform the TSAP (*Transport Service Access Point*) used to establish a connection using this protocol. Default value is 1 (one)

OPTION	DESCRIPTION
Proposed MMS PDU Size	When the **Transport Layer** option is configured as **MMS**, this option defines the size of the PDU negotiated with the server. The default value of this option is 65000 bytes

Reports Tab



Reports tab

The available options on this tab are described on the next table.

Available options on the Reports tab

OPTION	DESCRIPTION
Prefer Buffered Report Control Blocks (uses Unbuffered if not available)	IEC 61850 protocol uses **Report** objects to notify client applications about changes on data. Reported data are defined by users in a **DataSet**. Each **Report** can only be linked to a single **DataSet** and a device can have several **Reports** and **DataSets**. Please check section **Prefer Buffered Report Control Blocks** for more information about this option
Poll Tags not Found in any Report	When a Tag is not found in any **Report**, this option allows performing a cyclical reading (polling) of this Tag according to its scan rate. Remember that this method is not the most indicated nor the most efficient, because it may be subject to slow data updates and loss of fast events
Force Tag Polling Individually using N1 Parameter	A polling request for specific variables can be defined with a value different from 0 (zero) in the *N1* parameter if this option is configured

OPTION	DESCRIPTION
Check Report Revision Mismatch	Instructs this Driver to check for the **Reports** version when starting communication. In case these versions are incompatible, an update of cache (LD) files is then performed
User Defined Report List	If users do not want this Driver to perform an automatic search for **Reports**, they can define a list of **Reports** that must be enabled. Please check section **User-Defined Report List** for more information about this option
RW Report List File (.RPT)	Indicates whether the list of **Reports** defined in the previous item must be saved to a file, so that the example script of section **User-Defined Report List** is not needed. Writing to this file can be performed automatically by writing to the Tag or by directly editing that file, which must be on the default directory of cache (LD) files and must have the name of the IED and an .rpt extension. Only one IED is allowed per file. Please check section **Format of RPT Files** for more information
User Defined Datasets	When using a pre-defined list of **Reports**, users can inform whether **DataSets** are declared dynamically by this Driver. Please check section **User-Defined Datasets** for more information about this option. It is advisable to create a configuration script on this Driver's **AfterStart** event
RW DataSet List File (.CDS)	Indicates whether the list of **DataSets** defined in the previous item must be saved to a file, so that the execution of the example script on section **User-Defined Datasets** is not needed. Writing to this file can be performed automatically based on the previous examples or by directly editing that file, which must be on the default directory of cache (LD) files and must have the name of the IED and a .cds extension. Only one IED is allowed per file. For more information, please check section **Format of CDS Files**
Check BRCB Entry ID	By using this option, users inform that when enabling a **Buffered Report** (BRCB), they must configure the *EntryID* parameter that contains a unique identifier for each **Report** message processed. So, this option allows restarting an application, or when there is a switch-over of a redundant application, that the **Report** sends only messages not yet processed. When selecting this option, the application must keep synced the *EntryID* parameters of each **Buffered Report** in use and also perform writings during this Driver's initialization, informing the last received value. For more information, please check topic **Using EntryID**
RW EntryID List File (.EID)	Indicates whether *EntryID* values exposed in the previous item must be saved to a file, so that users do not need to execute scripts to retrieve or inform a field's value during initialization. These files are saved to the default directory of cache (LD) files and must have the name of the IED and an .eid extension. Only one IED is allowed per file. For more information, please check topic **Using EID Files**
Check BRCB TimeOfEntry	This option was deprecated and must be replaced by *EntryIDs*

OPTION	DESCRIPTION
Wait for Gtw Ref on Startup	Enables using the functionality of automatic control of *EntryIDs* for gateway applications. Please check topic **Gateway Configuration**
Copy All Values Before Reporting	If the **DataSet** linked to a **Report** contains objects with quality and timestamp after the corresponding properties, this option can be selected to avoid a notification of value to an application happens without the correct update of quality and timestamp
Auto Integrity Rpt (s)	If the informed value is different from 0 (zero), this value is used when enabling a **Report** as the interval that an unsolicited general interrogation (GI) message is generated by a device
Polled Integrity Report (s)	If the informed value is different from 0 (zero), this value is used to perform a **Report** reading by polling
Default BRCB ResvTms (s)	Default value for the **ResvTms** property, in seconds, used to enable **Buffered Reports**. This value is used in revision 2.1, defining the time interval a **Report** stays reserved for the current client after a connection failure
Use Exclusive URCBs	Defines if, when enabling an **Unbuffered Report** (URCB), this Driver is set to exclusive mode, that is, the **Reserved** property configured to 1 (one), or not, that is, the **Reserved** property configured to 0 (zero). A URCB enabled in exclusive mode cannot be used by other clients
Use Quality Change Trigger	Indicates an option of triggering by change of quality that must be used by default, when users are not using a fixed list of **Reports** or if the **TrgOps** option was not informed

Prefer Buffered Report Control Blocks

Reports can be **Buffered**-type or **Unbuffered**-type. **Buffered** means that all changes on elements of a **DataSet** during a disconnection are stored on buffers, so that a client application receives a notification about all those changes, as long as there is sufficient space in memory on a device and the settings are correct. This type of **Report** is used mainly for SOE (*Sequencing of Events*) data types. **Unbuffered Reports**, on the other hand, only store the last value in that situation. Both **Reports** can be configured to send events spontaneously, at cyclical intervals or to wait for a client application to explicitly request, by polling or general interrogation (GI), data from that **Report**. However, users must notice the following situations:

- If two client applications, such as two **Elipse Power** applications, are connected to a device, only one of those applications can connect to each **Buffered Report**. This is due to the fact that, when sending data, it is deleted from the **Report**'s internal buffer
- Two or more client applications can connect to the same **Unbuffered Report**, in case they are not using the exclusive usage option

This way, the **Prefer Buffered Report Control Blocks** option instructs this Driver, whenever a Tag enters a communication process, that is, enter in advise or in scan, to search among all **Buffered Reports** of a device if the specified Tag belongs to the respective **DataSet** of each **Report**.

If not found, then the **Report** is enabled by this Driver, if not yet, and starts receiving change notifications. Otherwise, this Driver starts to repeat the same search process, now on **Unbuffered Reports** for the same Tag. If found, the **Report** is

enabled, if not yet. If the answer is negative again, the Tag can communicate in polling mode if the **Poll Tags not found in any Report** option is enabled.

If the **Prefer Buffered Report Control Blocks** option is disabled, this Driver repeats the previous procedure, but then directly searching for **Unbuffered Reports** and discarding the search for **Buffered Reports**.

User-Defined Report List

- **TagName**: UserReportList
- **Device**: ServerName
- **Item**: UserDefinedReportList

```
Sub DRV_61850_AfterStart()
'The next script creates an array
'containing the configuration of a Logical Device and its corresponding Report,
'which are activated when writing to the UserReportList Tag.
'Notice that this is a two-position array.
'If users need to activate more Reports,
'configure the size of that array

  Dim arr(1)
  arr(0) = Array("LogicalDeviceName", "LLN0$BR$brcbEV101")
  arr(1) = Array("LogicalDeviceName", "LLN0$BR$brcbEV102")
  Set Cmd = Application.GetObject("DriverName.IEDName.UserReportList")
  Cmd.WriteEx(arr)
End Sub
```

NOTE

When selecting the **User Defined Report List** option, this Driver only finishes the initialization process after receiving a writing to the **UserReportList** Tag.

Format of RPT Files

```
Number_Of_LogicalDevices
LogicalDeviceN; Number_Of_Reports
ReportName; ReportOption1:ReportOption1Value; ReportOptionN:ReportOptionNValue
```

Available options for the RPT file format

PARAMETER	DESCRIPTION
Number_Of_LogicalDevices	Total number of Logical Devices in this file
LogicalDeviceN;Number_Of_Reports	For each Logical Device, insert a name and how many **Reports** are used
ReportName	Right after the name of each Logical Device, follows a list with the **Reports** used and optional fields
ReportOption;ReportValue	The **Optional fields** table contains optional fields allowed in an RPT file

Optional fields

FIELD	DESCRIPTION
DatSet:DataSetName	Name of a **DataSet** that must be linked to the **Report**
TrgOps:Options	Trigger options for the **Report**. The **Options** value corresponds to a mask of six bits with the values **0**: Not used, **1**: Data Change, **2**: Quality Change, **3**: Data Update, **4**: Integrity, and **5**: GI (*General Interrogation*)
IntgPd:Period	Period, in milliseconds, for integrity
BufTm:Value	Timing period to wait for new events before sending, after the notification of the first event to send, in milliseconds
Resv:Value	Use of URCB in exclusive mode. Possible values are **0** (zero) or **1** (one)
ResvTms:Value	Time to wait, in seconds, for a disconnection so that BRCB is reserved for the same client connected previously
PurgeBuf:Options	Deletes the buffer of messages. Possible values are **0**: Not Set (do nothing), **1**: Set Always, or **2**: Set if EntryID Write Fails
RptID:RptName	Identifier of a **Report**
OptFlds:Options	Optional fields for a **Report** message. The **Options** value corresponds to a mask of 10 bits with the values **0**: Reserved, **1**: SequenceNumber*, **2**: ReportTimeStamp*, **3**: ReasonForInclusion*, **4**: DataSetName**, **5**: DataReference, **6**: BufferOverflow*, **7**: EntryID*, **8**: ConfRevision*, and **9**: Segmentation. **NOTE**: Fields marked with * are used by default when the **OptFields** field is not specified. The **DataSetName** field is mandatory and all other parameters are fixed and cannot be changed

Example of an RPT file:

```
1
Device;1
LLN0$BR$BRCB1;ResvTms:1000
```

User-Defined Datasets

- **TagName**: DeclareDSList
- **Device**: ServerName
- **Item**: DeclareClientDSList

Users must inform an array of descriptors, each descriptor containing an array with three Elements, the name of the Logical Device, the name of the **DataSet** (add an @ character at the beginning of this name if volatile), and a unique index for each **DataSet**.

```
Dim arr
arr = Array("LogicalDeviceName", "DataSetName", 1)
Set Cmd = Application.GetObject("DriverName.IEDName.DeclareDSList")
Cmd.WriteEx(arr)
```

- **TagName**: PopulateDSList
- **Device**: ServerName
- **Item**: PopulateClientDSList

Users must inform an array, and each item must contain an array with two Elements, **DataSet**'s index and the name of the LN/DO/DA in the format "LogicalDevice\LN$DO$DA".

```
Dim arr(4)
arr(0) = Array(1, "LDName\GGIO1$ST$stval")
arr(1) = Array(1, "LDName\GGIO2$ST$stval")
arr(2) = Array(1, "LDName\GGIO3$ST$stval")
arr(3) = Array(1, "LDName\GGIO4$ST$stval")
arr(4) = Array(1, "LDName\GGIO5$ST$stval")
Set Cmd = Application.GetObject("DriverName.IEDName.PopulateDSList")
Cmd.WriteEx(arr)
```

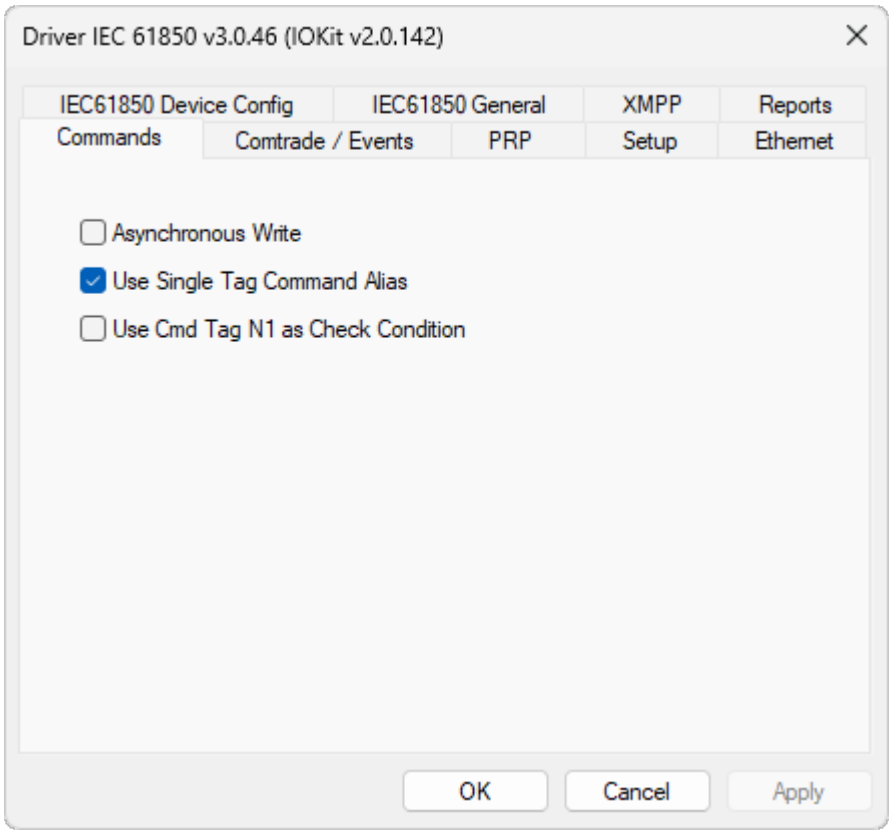
Format of CDS Files

```
Number_Of_LogicalDevices
LogicalDeviceN; Number_Of_DataSets
DataSetName; NumberofMembers
MemberNames1...N
```

Example of a CDS file:

```
1
Device;1
MyDataset1;3
Device/LLN0$DC$NamPlt
Device/LLN0$ST$Mod
Device/LPHD1$DC$PhyNam
```

Commands Tab



Commands tab

The available options on this tab are described on the next table.

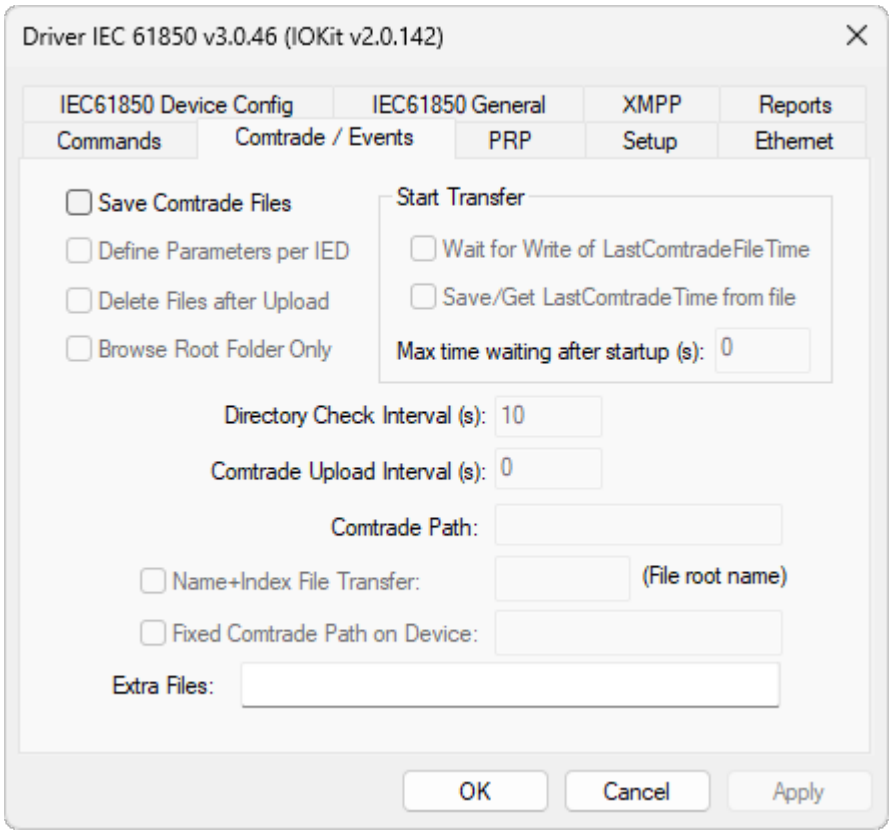
Available options on the Commands tab

OPTION	DESCRIPTION
Asynchronous Write	Allows request commands to return success immediately, without waiting for an answer, as long as its sending was successful, such as when there is an active connection. This behavior aims to facilitate operations involving sending a large number of commands, to increase general speed. However, it does not check for failures during the processing of that command on the device
Use Single Tag Command Alias	Instructs this Driver to create, during the Tag Browsing process, a single Tag to send commands. For more information, please check the next **note**. If this option is not selected, a Block Tag with five Elements is then used. For more information, please check topic **I/O Tags**
Use Cmd Tag N1 as Check Condition	In case of using a simple Tag to send commands, this option informs whether the *N1* parameter of this Tag is used to indicate a security check parameter of a command. The *N1* parameter, in this case, must be a number between 0 (zero) and 3 (three), as the result of a logic **OR** between bits **0**: INTERLOCKING and **1**: SYNCHROCHECK

NOTE

If users are using **Elipse Power** as an OPC server, the command Tag can have its reading enabled, that is, its **AllowRead** property configured to True. This allows this Tag to receive the initial value of the **CtlVal** property of the referred command object, so that an OPC client previously knows the data type. After sending a command, this Driver returns an automatic reading for the same Tag, with the value written in case the operation is successful. This way, the *WriteFeedBackMode* parameter of this Driver, which indicates how a Tag receives a confirmation of the operation by keeping the written value, can be configured as **0**: WaitNextRead, preferably.

Comtrade / Events Tab



Comtrade / Events tab

The available options on this tab are described on the next table.

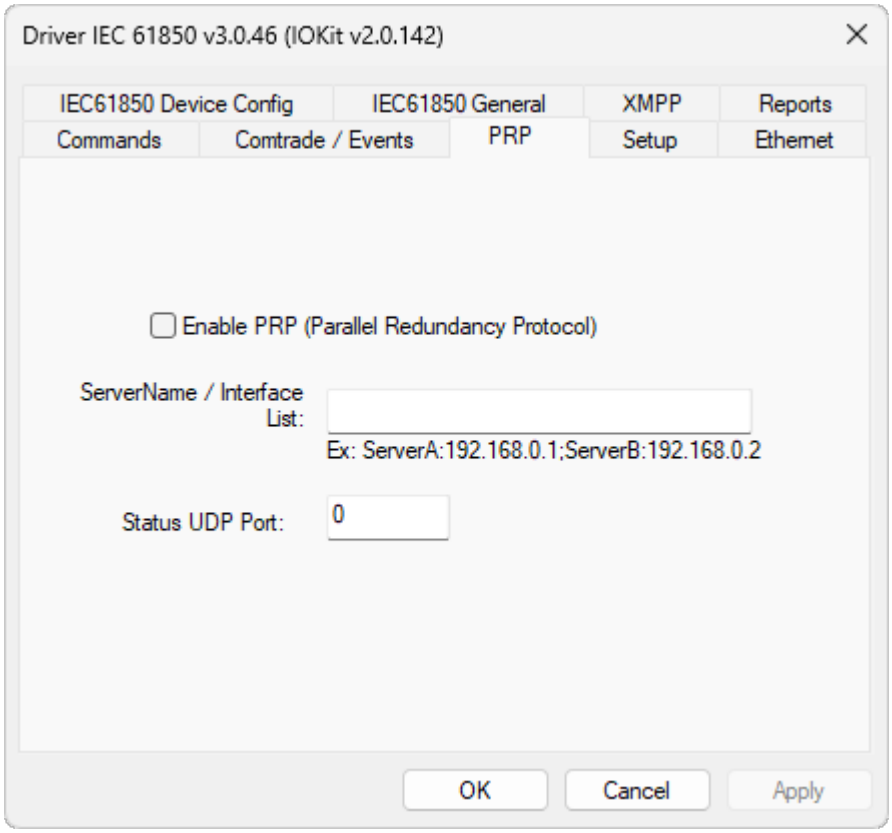
Available options on the Comtrade / Events tab

OPTION	DESCRIPTION
Save Comtrade Files	Indicates whether this Driver must perform a search for files in **COMTRADE** (oscillations) format, saving them on the directory indicated in the **Comtrade Path** option
Define Parameters per IED	Indicates whether the parameters *Save Comtrade Files*, *Delete Files after Upload*, *Browse Root Folder Only*, *Comtrade Path*, *Use Fixed Comtrade Path on Device*, *Name+Index File Transfer*, and *File Root Name* are defined per IED and not globally for the whole Driver. The individual definition can be performed by writing this Driver's internal parameters via **IOKit** library, as explained on topic **Dynamic Configuration**

OPTION	DESCRIPTION
Delete Files After Upload	Instructs this Driver to delete a device's file after successfully finishing the transfer
Browse Root Folder Only	Some relays insert COMTRADE files on a **COMTRADE** folder from the root folder, and other relays insert these files on the root folder, indicating as part of the file name that folder, such as "\COMTRADE\File.cfg". Select this option for devices with that last behavior
Wait for Write of LastComtradeFileTime	With this option selected, this Driver waits for a writing operation with the time of the last transferred COMTRADE file, per IED, so that it can compare this time with the list of existing files on the IED, thus defining which file to transfer
Save/Get Last Comtrade time from File	Gets the time of the last COMTRADE file transferred using a COMTRADE.LST file
Max Time Waiting after Startup	Indicates how many seconds this Driver must wait for the writing of the Tag with the time of the last file, or for the reading of the COMTRADE.LST file. If this time expires, this Driver starts searching for the first available file. If this option is equal to 0 (zero), it waits indefinitely
Comtrade Directory Check Interval	Interval, in seconds, in which this Driver requests a list of files from a device. This Driver compares the date of the last transferred file with the files on the list, and the most recent file or files are transferred
Comtrade Upload Interval (s)	In case of transferring more than one file, this Driver waits the time defined in this option to perform each transfer, to separate a specific time window for this activity. However, receiving events and sending commands are not interrupted while transferring a file
Comtrade Path	Indicates a directory where this Driver saves COMTRADE files
Name+Index File Transfer	Informs the file transfer is performed based on a root name plus an index, instead of comparing dates, which is the default behavior. The root name must be informed in the **File Root Name** option. For example, if an IED generates files in the format Osc1.dat/cfg, Osc2.dat/cfg, and so on, users must select this option and inform the text "Osc" in the **File Root Name** option
Fixed Comtrade Path on Device	Indicates that this Driver must ignore a search for the **COMTRADE** folder on a device, assuming a fixed folder informed in the corresponding field

PRP Tab

This tab allows configuring the operation of this Driver on a PRP (*Parallel Redundancy Protocol*) redundant network.



PRP tab

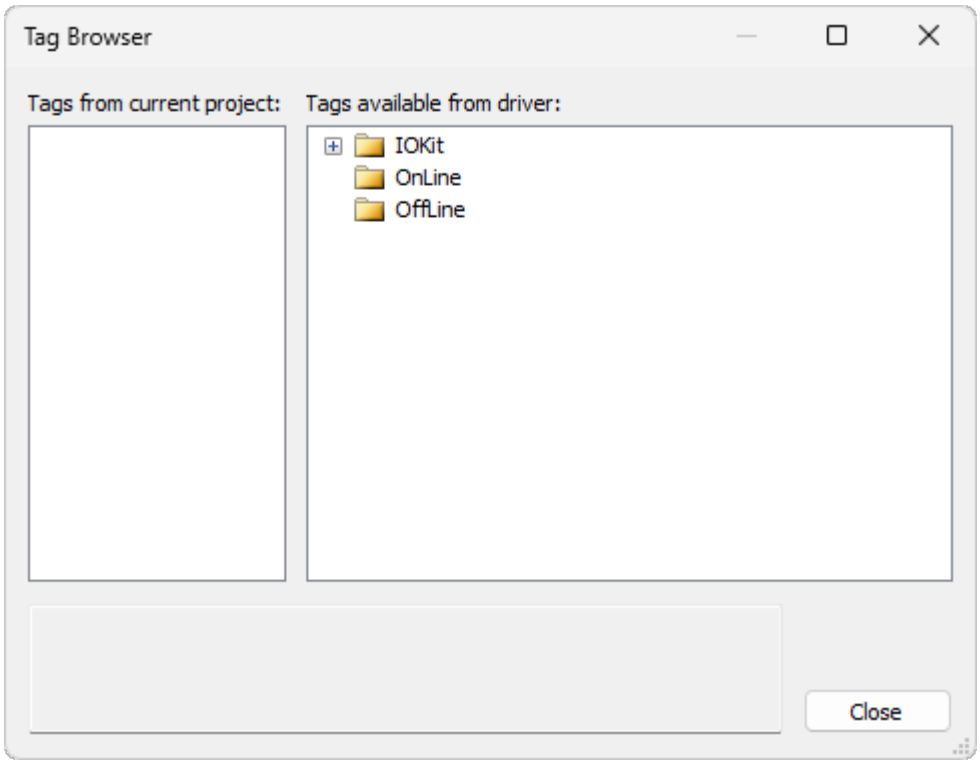
The available options on this tab are explained on the next table. For more information, please check topic **PRP Redundancy**.

Available options on the PRP tab

OPTION	DESCRIPTION
Enable PRP	Indicates whether this Driver must use the PRP Service. This option is used so that this Driver configures the network interface as defined in the **ServerName / Interface List** option, and also starts listening to the UDP status port
ServerName / Interface List	Informs a list of server names (DNS) and IP addresses in the format **ServerA:IPA;ServerB:IPB**, so that the same project file containing this Driver can be used without changes in at least 2 (two) servers. When this Driver starts, it gets the name of the local computer and checks whether there is an IP address on that list that correspondes to this name. If positive, this IP address is used as the IP address of the virtual interface
Status UDP Port	Inform an available local port, which is used by this Driver to receive statistics and logs from the ElipsePRPSvc.exe service. It must be the same port informed in the configuration of the PRP service

Tag Browser

Click **Tag Browser** on this Driver's toolbar in **Elipse Power** to open the Tag Browser window, which allows listing and dragging to this Driver the Tags identified in a device, according to the next figure.



Tag Browser window

The **Tags available from driver** list contains the following folders:

- **IOKit**: Contains standard Tags from **Elipse Software**'s **IOKit** library, allowing to read or write general connection parameters and status, among others
- **Online**: Displays Servers declared on the **IEC61850 Device Config** tab. When clicking a Server, this Driver tries to communicate with the device, creating a new child folder for each Logical Device (LD) found in a device. Click an LD to retrieve all respective Tags
- **Offline**: Displays all Servers and LDs, but clicking an LD displays Tags from LD files created in the import directory, whether from importing SCL files or by executing an application

To use these Tags in an application, drag them from the **Tags available from driver** list to the **Tags from current project** list.

NOTE

This Driver inserts in each Logical Device a **DataSets** folder, in which users can find all **DataSets** of that Logical Device referenced by any **Report**.

About IEC 61850 Standard

The IEC 61850 is an international standard for communication in electrical substation automation systems, widely used in smart grids. It defines a robust, flexible, and interoperable communication model for protection, control, monitoring, and measurement devices in power systems.

Key Features:

- **Standardized Data Model**: Utilizes data objects (Logical Nodes) to represent functions and equipment, such as circuit breakers and transformers, ensuring consistency and interoperability.

- **Ethernet-Based Communication**: Supports protocols like MMS (Manufacturing Message Specification), XMPP (Extensible Messaging and Presence Protocol), GOOSE (Generic Object-Oriented Substation Event), and SV (Sampled Values) for real-time data exchange.
- **SCL Configuration**: Employs the Substation Configuration Language (SCL) to describe device and system configurations, simplifying engineering and maintenance.
- **High Reliability and Scalability**: Designed to meet performance requirements in critical environments, with support for redundancy and efficient communication.

Benefits:

- Interoperability among devices from different manufacturers.
- Reduced configuration and maintenance costs.
- Support for advanced applications, such as automation and real-time monitoring.

The implementation of IEC 61850 in the software enables efficient integration with compatible devices, ensuring reliable and adaptable operation in modern substations.

Transport Layer

The IEC 61850 standard was originally developed for Ethernet-based communication using the MMS protocol, as specified in IEC 61850-8-1, with a focus on local substation networks. In 2018, the IEC 61850-8-2 standard introduced an alternative based on the XMPP (Extensible Messaging and Presence Protocol), primarily aimed at communication outside the local network, expanding its applicability.
Thus, this driver allows the user to choose between these two protocols as the transport layer.
The standard also defines an Ethernet-based data exchange known as GOOSE (Generic Object-Oriented Substation Event), which is not currently implemented in this driver.

Information Structuring

Systems that implement the standard as data servers (typically IEDs or protection relays) organize information based on the following entities:

- **Logical Devices (LD)**: Represent logical devices corresponding to a physical or functional device, such as a bay mapped to an IED (Intelligent Electronic Device). They form the first hierarchical level in an IEC 61850 server, grouping related logical functions to facilitate modeling and management.
- **Logical Nodes (LN)**: Virtual representations of functions or components of a real device, mapped within a Logical Device. For example, a circuit breaker in a bay is represented by a Logical Node with the standard class **XCBR**. A Logical Device typically contains multiple Logical Nodes, each corresponding to a specific function, such as protection, control, or measurement.
- **Data Objects (DO) and Data Attributes (DA)**: Information within a Logical Node is organized into **Data Objects**, which represent specific data sets related to a function. Each Data Object consists of **Data Attributes**, which describe detailed properties, such as values, states, or configurations. The set of Data Attributes in a Data Object follows a defined structure called **CDC (Common Data Class)**, ensuring standardization and interoperability.

- **Functional Constraint (FC)**: Specific services that can be applied to each Data Attribute, forming part of the object's path.

The format used by this Driver for all Tags representing Data Objects or Data Attributes is as follows:

- **Device Parameter**: Server:LD, for example, "AL_07:AL_07PRO"
- **Item Parameter**: LNFCData$DataAttribute, for example, "XCBR1STMod$stVal"

Tag Reference

This section contains information about the configuration of **I/O Tags** and **Internal Tags**.

I/O Tags

The *N* parameters of I/O Tags are not used. Tags are addressed only using the *Device* and *Item* parameters.

- **Device**: Server:LD, configured on the **IEC61850 Device Config** tab
- **Item**: LN$Data$DataAttribute or LNFCData$DataAttribute, according to the next table

Available options for the Item parameter

ITEM	OPERATION	MEANING
LN$Data$DataAttribute or LNFCData$DataAttribute	Reading or Writing	Any Tag to read according to the configuration of **Report** parameters or polling. If this Tag is for writing, it is accepted by this Driver
LNRPurcbXXXX or LNRPbcrbXXXX	--	This Driver does not display the tree corresponding to **Reports**, for simplicity and Tag saving. The whole handling is performed internally. Starting with version 1.0.14, this Driver displays the **TimeofEntry** Tag (reading and writing) only on **Buffered Reports**. Starting with version 1.1.18, also displays the **EntryID** Tag on **Buffered Reports**
LNCOxxxx or LNSPxxxxx	Writing	Command or control block. To simplify the use of commands, this Driver provides, instead of a tree with the control object, a Block Tag with five Elements, described on the **Elements of a command or control block** table. The immediate result of accepting or not a command can be retrieved via script, using a Block Tag's **WriteEx** method on the *wWriteStatus* parameter, or via status block. Please check the next item for more information
LNCOxxxx$Oper, LN$CO$xxxx$TimeActOper, LNSPxxxxxSBO, LNCO$xxxx$SBOw, or LNCOxxxx$Cancel	--	If the **Use Single Tag Command Alias** option is selected, this Driver creates in Tag Browser a single Tag instead of a Block Tag with five Elements. In this case, sending a command is

ITEM	OPERATION	MEANING
		performed according to section **Sending Commands with a Single Tag**
LNCOxxxx or LNSPxxxxx	Reading	Command status block. For each block or command Tag a second block is created, with two Elements, that displays the status of operations. This status can be changed as the result of denying to send a command, such as an unsupported command, as the result of accepting, such as a command accepted or not, or as the result of an effective action, such as opening or closing a switch. The first Element of this Block Tag (**Status**) contains a number code and the second Element (**StatusText**) contains a textual description of that number code. Please check table **Number codes of a command status block** for more information

Elements of a command or control block

ELEMENT	DESCRIPTION
Operation	It must receive the operation to perform in **Text** format. Possible values are **OPERATE**, **SELECT**, **SELECTWITHVALUE**, or **CANCEL**
Value	Value to send, depending on the type of object, in the **CtlVal** property
Time	Time instant, in Visual Basic standard as days since 1900, for use in a **SELECT** command by time and filling the **T** property
Test	Indicates whether a command is a test operation (value equal to one) or not (value equal to zero)
Check	Type of check performed before performing a command. It must be a number between 0 (zero) and 3 (three) as the result of a logic **OR** between bits **0**: INTERLOCKING and **1**: SYNCHROCHECK

Number codes of a command's status block

STATUS	STATUSTEXT
0	Terminated OK
2	Select Accepted
3	Select With Value Accepted
4	Cancel Accepted

STATUS	STATUSTEXT
5	Operate Accepted
50	Terminate Fail
100	AppError:Unknown
101	AppError:NotSupported
102	AppError:BlockedBySwitchingHierarchy
103	AppError:SelectFailed
104	AppError:InvalidPosition
105	AppError:PositionReached
106	AppError:ParameterChangeInExecution
107	AppError:StepLimit
108	AppError:BlockedByMode
109	AppError:BlockedByProcess
110	AppError:BlockedByInterlocking
111	AppError:BlockedBySynchrocheck
112	AppError:CommandAlreadyInExecution
113	AppError:BlockedByHealth
114	AppError:1_Of_N_Control
115	AppError:AbortionByCancel
116	AppError:TimeLimitOver
117	AppError:AbortionByTrip
118	AppError:ObjectNotSelected
200	WriteError:Object_invalidated
201	WriteError:Hardware_fault
202	WriteError:Temporarily_unavailable
203	WriteError:Object_access_denied
204	WriteError:Object_undefined
205	WriteError:Invalid_address
206	WriteError:Type_unsupported
207	WriteError:Type_inconsistent
208	WriteError:Object_attribute_inconsistent
209	WriteError:Object_access_unsupported
210	WriteError:Object_non_existent
211	WriteError:Object_value_invalid
212	WriteError:Error_unknown
213	WriteError:Timeout
214	WriteError:OutOfMemory
215	WriteError:DecodeError

STATUS	STATUSTEXT
216	WriteError:WrongParameters
217	WriteError:CmdnotSupported

Sending Commands with a Single Tag

- Tag's value is the value used in the **CtlVal** property
- Tag's timestamp is used in the **T** property
- The **Test** property has a fixed value of 0 (zero)
- The **Check** property has a fixed value of 3 (three, INTERLOCKING and SYNCHROCHECK), except when the *N1* parameter is enabled in the **Use Cmd Tag N1 as Check Condition** option

This operation is defined in the **Item** property with suffixes **Oper** (*Operate*), **SBO** (*Select Before Operate*), **SBOw** (*Select Before Operate With Value*), or **Cancel**.

The Tag for the **TimeActivatedOperate** operation, with a **TimeActOper** suffix, is available when the LN contains the **OpertTm** property. This operation occurs in a way similar to the **Operate** command, except that the Tag's timestamp is used as the operation's time and a scheduled operation can be canceled using the **Cancel** Tag.

Internal Tags

DEVICE	ITEM	OPERATION	MEANING
ServerName	DeviceConnectionStatus	Read-Only	Informs the device TCP/IP connection status. 0 = not connected 1 = connected
ServerName	ServerStatus	Read-Only	Returns the internal status of this Driver. If the *Device* parameter is only equal to *ServerName*, then this status is relative to the device as a whole. Possible values are **0**: Starting, **1**: Reading LD files, **2**: Retrieving directory (**GetServerDirectory**), **3**: Processing a single LD (*Tag Browsing*), **4**: Retrieving a directory of files, **5**: Processing LDs, **6**: Finishing, **7**: Finished OK, **8**: Finishing with error, **9**: Finished with error, **10**: Reconnecting, or **11**: Finished (waiting)
ServerName:LDName	ServerStatus	Read-Only	If the *Device* parameter is equal to *ServerName:LDName*, then status is relative to the *Logical Device*. Possible values are **0**: Waiting, **1**:

DEVICE	ITEM	OPERATION	MEANING
			Reading LD file, **2**: Retrieving LD directory, **3**: Retrieving LN directory, **4**: Retrieving data definition, **5**: Creating a cache of Tags, **6**: Discovering **Reports**, **7**: Retrieving **DataSets**, **8**: Retrieving the content of **DataSets**, **9**: Retrieving Nameplates, **10**: Retrieving versions of **Reports**, **11**: Checking versions, **12**: Programming **Reports**, **13**: Operation (runtime), **14**: Finishing, or **15**: Finishing with error
ServerName	ComtradeTransferStatus	Read-Only	Returns the activity status of file transfers via COMTRADE. Possible values are **0**: Not connected, **1**: Connected, **4**: Waiting to list files, **5**: Retrieving a list of files, **6**: Files transferred (synced), **7**: Waiting to transfer files, **8**: Transferring file, **101**: Empty list of files in the relay, **102**: Format errors in the reading command, **103**: Error when saving COMTRADE file, or **104**: Error when transferring file
ServerName	LastComtradeFileName	Read-Only	Informs the name of the last transferred COMTRADE file
ServerName	ServerRebuild	Read-Only	Removes cache files (.LD) and restarts communication with a device. The *Device* parameter can have the same name of the IED, with or without a Logical Device
ServerName	UserDefinedReportList	Write-Only	Allows defining a list of user-created **Reports**
ServerName	LastComtradeFileTime	Reading and Writing	Indicates the date of the last collected COMTRADE file. When starting an application and avoiding files already collected to be retrieved again, this Tag must receive a writing with the date of the last existing file. So, It is recommended that an application stores the value of this Tag when quitting and write back this value

DEVICE	ITEM	OPERATION	MEANING
			when starting, or use the **Save/Get Last Comtrade Time from File** option. When the **Name+Index File Transfer** option is set, this Tag indicates the last file index to transfer, and it can also be read or written. It is important to notice that if the IED is completely reset, this index can return to an initial value, such as 1 (one) and, for this Driver to return to transfer these files based on this index, users must write to this Tag the new value. **NOTE**: Files are overwritten if their names are duplicated
ServerName	ComtradeInfo	Read-Only	Returns a Block Tag with three Elements with COMTRADE properties. Values for Elements are **0**: ComtradeTransferStatus, **1**: LastComtradeFileTime, and **2**: LastComtradeFileName
ServerName	BRCBActualList	Read-Only	Returns a Block Tag with four Elements with a list of **Buffered Reports** in use. Values for Elements are **1**: LDName, **2**: ReportName, **3**: TimeOfEntry, and **4**: EntryID
ServerName	URCBActualList	Read-Only	Returns a Block Tag with two Elements with a list of **Unbuffered Reports** in use. Values for Elements are **1**: LDName and **2**: ReportName
ElipseClientInfo	InternalClock	Read-Only	Returns a Block Tag with two Elements with the current time and quality from internal clock. Values for Elements are **0**: Current time (TIME) and **1**: TimeQuality. The meaning of the bits of Element 1 (one) is **7**: Leap Seconds Known, **6**: Not used, **5**: Sync Error, and **4-0**: Precision (in power of 2-y)
-	ServerInitialBuild	Write-Only	Allows changing IEDs parameters before starting communication. Please check section **ServerInitialBuild Item** for more information

DEVICE	ITEM	OPERATION	MEANING
-	GatewayRef	Write-Only	Block Tag with four Elements that allows informing an external gateway reference to a 61850 object, through a writing operation. The elements are **0**: IEDName (name of a 61850 server configured in this Driver), **1**: LDName (name of a Logical Device Name), **2**: Object (name of a 61850 item, such as "LLN0STData$DataAttr"), and **3**: Reference (**String** representing the address of this item in an external Driver, such as "202;34", indicating an object 2 (two), variation 2 (two) and index 34 in a DNP Slave Driver. For more information, please check topic **Gateway Configuration**
-	FinishedGtwRef	Write-Only	Tag that allows informing, through a writing operation, that the operations of adding gateway references ended. For more information, please check topic **Gateway Configuration**
-	GatewayEventOK	Write-Only	Block tag with four Elements that allows informing that an event was confirmed by an external protocol in a gateway application, thus allowing the last **Report** EntryID to be set to avoid event loss or duplication. The elements are **0**: Reference (**String** representing the address of this item in any external Driver, such as "202;34", indicating an object 2 (two) variation 2 (two) and index 34 in a DNP Slave Driver, **1**: Value (value of the received event), **2**: TSValid (indicates whether the timestamp is valid and is being used at the slave protocol), and **3**: Timestamp (in the format of seconds since 1970). For more information, please

DEVICE	ITEM	OPERATION	MEANING
			check topic **Gateway Configuration**
-	DeleteAllDevices	Write-Only	Remove all devices from driver memory and stops their communication. In order to make the devices return to communication, it is possible to write the tag ServerInitialBuild.
ServerName or Empty	GIAllReports	Write-Only	Requests a GI (General Interrogation) of all reports enabled by the driver, for a specific IED (by defining the IED name at the Device property) or for all declared IEDs, if the Device property is empty.
ServerName	WorkOnline	Write-Only	0 = Disconnects an IED 1 = Connects an IED
ServerName	IPSelect	Read-Only	Informs which IP address is in operation, in cases of using main and backup IP addresses
ServerName	IOKitEvent	Read-Only	Returns **IOKit** events for the specified IED, according to the **Read Driver Events** Tag available on topic **Documentation of I/O Interfaces**
ServerName	GetFileDirectory	Read-Only	Informs a list of existing files in the IED after a **GetFileDirectory** (writing) request. If there is no previous request or if there is no data available, returns an empty list
ServerName	FinishedWriteTimeOfEntry	Write-Only	Please check topic **Reports Tab** for more information
ServerName	FinishedWriteEntryID	Write-Only	Please check topic **Using EntryID** for more information
ServerName	IPSwitch	Write-Only	Requests a switch from the main IP address to the backup address or vice versa
ServerName	DeclareClientDSList	Write-Only	Declares a list of **DataSets** defined in the client. Please check topic **Reports Tab** for more information

DEVICE	ITEM	OPERATION	MEANING
ServerName	PopulateClientDSList	Write-Only	Populates the members of each **DataSet** defined in the previous item. Please check topic **Reports Tab** for more information
ServerName	GetFileDirectory	Write-Only	Requests a list of files in the IED, based on a directory informed in Tag's value or on the root directory if no directory is informed
ServerName	GetFile	Write-Only	Requests a transfer of a file with the name informed in the Tag. This file is saved with the same name on the default directory of COMTRADE files
ServerName	DeleteFile	Write-Only	Requests a removal of a file with the name informed in the Tag directly in the IED
ServerName	CompareLogicalDeviceDirectory	Write-Only	Compares the content of a Logical Device (LD) retrieved from a cache file (.LD) with the current content in the IED. If they do not match, a new cache file is generated

ServerName	ReportItemsUnified	Read-Only	A Block containing an overview of all events received by reports of the IED informed at the Device field, with 7 elements:
			Element 0: Logical Device Name
			Element 1: Item Path
			Element 2: Report Path
			Element 3: Dataset Path
			Element 4: EntryID (string)
			Element 5: ReasonCode (DWORD)
			Element 6: Quality, Value and Timestamp (as string)
			
			The reason code use these bits set to inform the transfer reason: (starting from bit 0 as LSB):
			`TrgOpt_DataChange = 1,`
			`TrgOpt_QualityChange = 2,`
			`TrgOpt_DataUpdate = 3,`
			`TrgOpt_Integrity = 4,`

DEVICE	ITEM	OPERATION	MEANING
			`TrgOpt_GI = 5,`
ServerName:LDName	CompareDataDefinition	Write-Only	Compares the content of a Logical Node (LN) retrieved from a cache file (.LD) with the current content in the IED. If they do not match, a new cache file is generated
ServerName:LDName	GetDataValues	Write-Only	Requests a reading of an item, which can be **LN**, **DO**, or **DA**. Values are returned in their respective Tags
ServerName:LDName	SetDataValues	Write-Only	Requests a writing of current values of the informed item, which can be **LN**, **DO**, or **DA**
ServerName:LDName	GetDataSetValues	Write-Only	Requests a reading of all items belonging to a **DataSet**. Values are returned in their respective Tags
ServerName:LDName	GetAllDataValues	Write-Only	Requests a reading of all items belonging to a **Functional Constraint**. Tag's value must be in the format **LN$FC**. Values are returned in their respective Tags
ServerName:LDName	DeleteDataSet	Write-Only	Request the removal of a **DataSet** informed in the Tag

ServerInitialBuild Item

In the next example, supposing there is an **IniBuild** Tag with its *Item* parameter equal to **ServerInitialBuild**, an IED with index 0 (zero) is enabled and the *TSel* parameter is changed.

```
Dim CommArr(1)
CommArr(0) = Array("IEC61850.Device[0].Disable", 0)
CommArr(1) = Array("IEC61850.Device[0].TSel", 1)
Write -1, 0, 0, 3, CommArr
Item("IniBuild").WriteEx(1)
```

Redundancy

The ways this Driver avoids receiving duplicated events during start up are the following:

- By defining the last **EntryID** of each **Report** by script
- By defining the last **EntryID** with automatic generation of a file with an .eid extension

The next topics describe each one of these ways.

Using EntryID

When selecting the **Check BRCB EntryID** option on the **Reports** tab without the option of automatic files (EID), an application must perform the next procedures to work properly:

1. Create in this application, or import via **Tag Browser**, the **EntryID** Tags available in each **Buffered Report** and in use by this Driver. This Tag's value is a **Text**-type and represents a sequence of bytes.

2. If users do not know whether a **Buffered Report** is in use, please check the list of BRCBs, as described on topic **Reports Tab**.

3. Create in this application an Internal Tag for each **EntryID** to sync. This Internal Tag must have its **Retentive** property configured to True. **NOTE**: Users can also perform this procedure using XObject Properties, which can have their **Retentive** property enabled directly.

4. Each Internal Tag, or XObject Property, must receive value changes from the corresponding **EntryID** via script or Link.

5. When starting this application, there must be a script that writes the value of each **EntryID** according to the value of the corresponding Internal Tag, which must be updated according to the value sent by the redundant station. At the end of this process, a special Tag must be written indicating that the process of writing **EntryID** properties ended and this Driver can resume initialization. This Tag must be configured with the next properties.

   - **Name**: FinishedWriteEntryID
   - **Device**: ServerName
   - **Item**: FinishedWriteEntryID

Any writing operation is enough for this process to be accepted.

NOTE

If the **EntryID** check option is configured, this Driver only resumes initialization after writing that Tag.

Using EID Files

When selecting the **R/W EntryID File (.EID)** option on the **Reports** tab, this Driver generates a file for each IED with an .eid extension, containing the last received **EntryIDs** for each **Report** in use. This file is generated in at least two seconds after receiving a Report message and also before shutting down an application. This file is generated on the same directory of cache files (LD).

During this Driver's initialization, the application waits for the availability of this file to proceed, to retrieve the values of the processed **EntryIDs** until this operation is interrupted. In a redundant system, a user-created application must be responsible for syncing this file between main and standby computers.

The next example, which must be executed cyclically, copies **EntryID** files among redundant servers. As this Driver is loaded by the **IOServer** module, which executes on the **System** account, the example code uses the **psExec** (**SysInternals**) application to elevate privileges of a Windows user, by executing a batch (.bat) file that specifies a user and a password.

NOTE

For more information about the **psExec** application, please check *this* article on **Microsoft Learn**.

```
Sub MyTimer_OnPreset()
  Set objWScript = CreateObject("WScript.Network")
  strServer = objWScript.ComputerName
  Set sw = CreateObject("WScript.shell")
  Set fso = CreateObject("Scripting.FileSystemObject")
  Set folder = fso.GetFolder("C:\MyAppDir\MyEntryIdDir")
  'Loads .eid files
  For Each file In folder.Files
    GetAnExtension = fso.GetExtensionName(file.path)
    If GetAnExtension = "EID" then
      'Copies files
      fso.CopyFile file.path, "C:\MyAppDir\MyEntryIdDir\EID\"
    End If
  Next
  If strServer = "Server2" Then
    sw.run "C:\MyAppDir\Bat\psExec.exe -i 0 -u UserAccount -p Domain@password cmd /c C:
\MyAppDir\Bat\toServer1.bat", 0
  End If
  If strServer = "Server1" Then
    sw.run "C:\MyAppDir\Bat\psExec.exe -i 0 -u UserAccount -p Domain@password cmd /c C:
\MyAppDir\Bat\toServer2.bat", 0
  End If
End Sub
```

Content of **toServer1.bat** file:

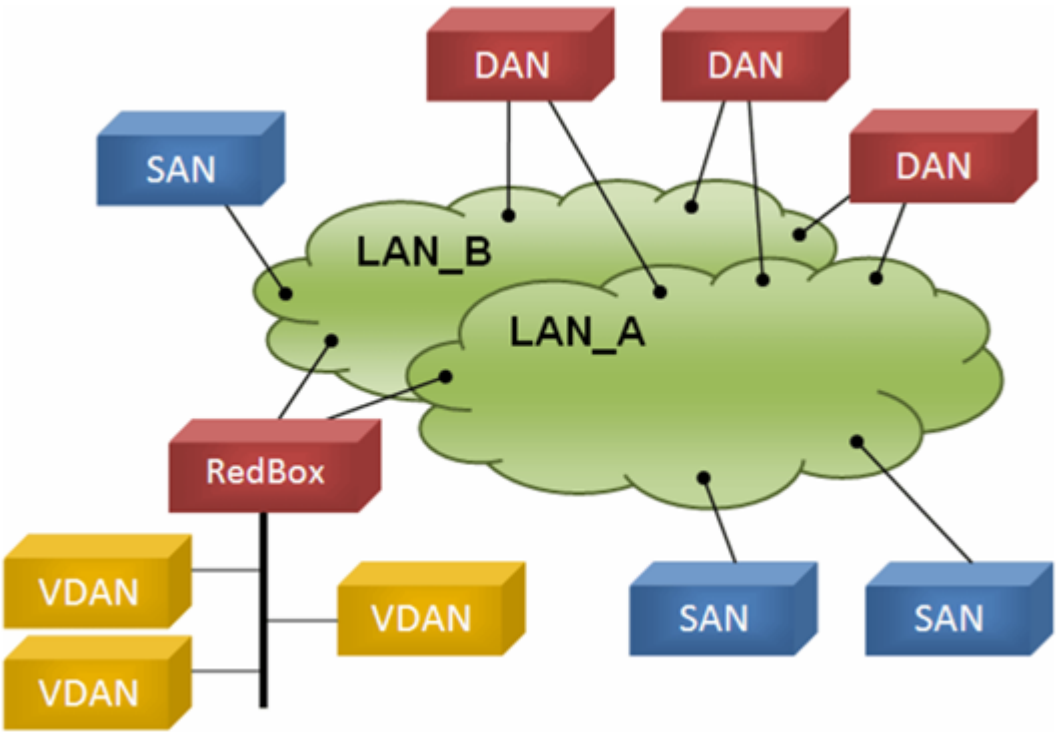
```
copy C:\MyAppDir\EntryId\EID, \\Server1\EntryID
```

Content of **toServer2.bat** file:

```
copy C:\MyAppDir\EntryId\EID, \\Server2\EntryID
```

PRP Redundancy

PRP (*Parallel Redundancy Protocol*) is a protocol standard for Ethernet networks that allows seamless network redundancy against any simple network failure, with transparent redundancy to the application.



Configuration of a PRP network

In PRP protocol, there must be two parallel networks, **LAN_A** and **LAN_B**, in which the preferable and totally redundant adherent devices are called **DANs** (*Dual Attached Nodes*), that is, they have two interfaces, one on each network. Devices with a single interface, called **SAN** (*Single Attached Nodes*), can also take part of the architecture, but do not benefit from redundancy, as well as other devices that do not directly support PRP protocol, called **VDANs** (*Virtual Dual Attached Nodes*), can be connected through a **RedBox** (*Redundancy Box*).

This Driver behaves as a **DAN**, it is enough that the computer has two physical network adapters and one virtual network adapter, installed and configured correctly. The next topics show the steps for this configuration:

- **Installing and Configuring a Traffic Sniffer Library**
- **Installing and Configuring a Virtual Adapter**
- **Configuring Physical Network Adapters**
- **Installing, Configuring, and Executing the ElipsePRPSvc.exe Service**
- **Network Monitoring and Statistics**

Traffic Sniffer Library

The basic behavior of the PRP service offered with this Driver is based on capturing packets from a virtual adapter and transferred to physical adapters **A** and **B**. Likewise, packets received on physical adapters **A** or **B** are transferred back to the virtual adapter.

To perform packet capturing and injection on adapters, users must install the **Npcap** traffic sniffer library on the computer, which is installed with *Wireshark* or can be downloaded on *Nmap* project's website.

To work correctly, users must install the **Npcap** library in compatibility mode with WinPcap API, by selecting this option on the installation dialog box.

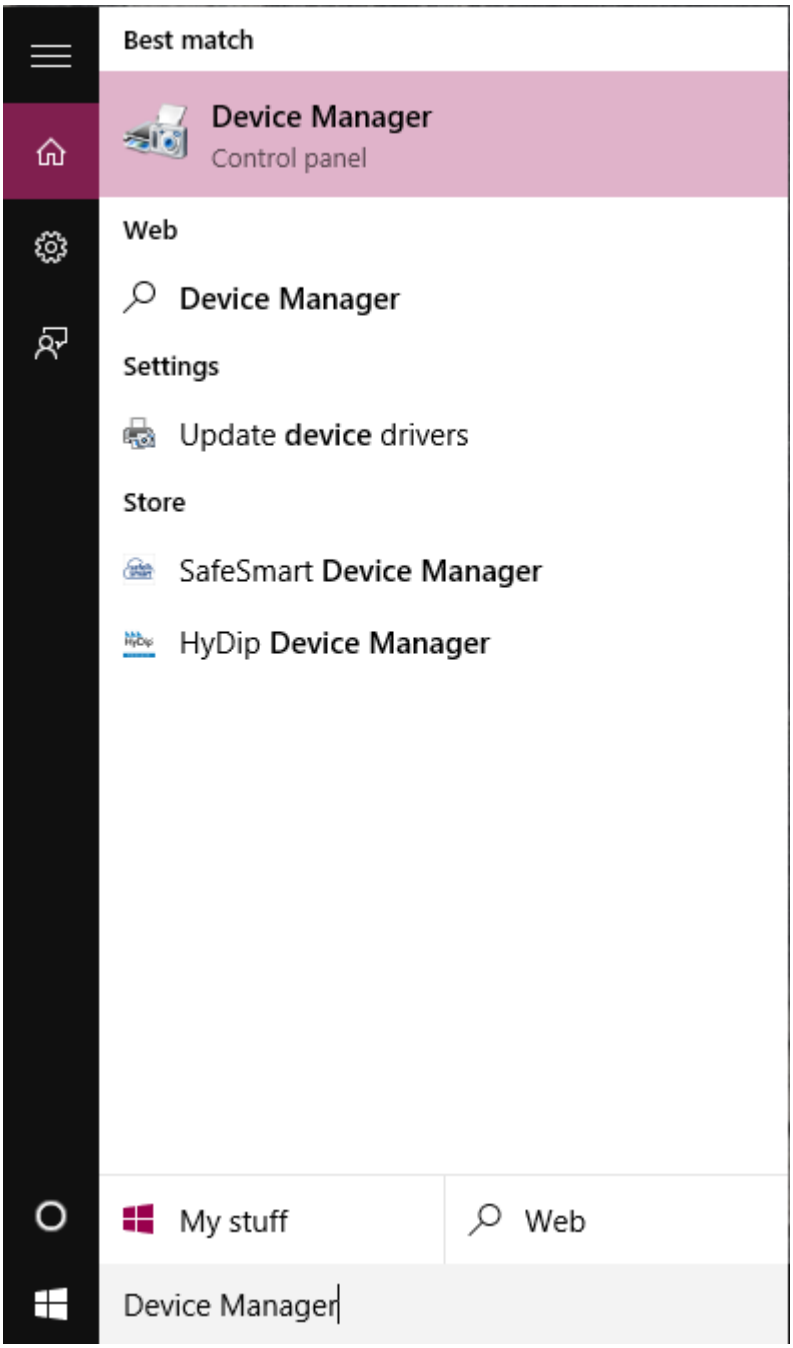
Virtual Adapter

A **Virtual Adapter** is a non-physical network adapter installed on Windows, aiming to provide a unique interface for this Driver, or even other protocols, regardless of which physical network is active, whether **A**, **B**, or both.

Installation

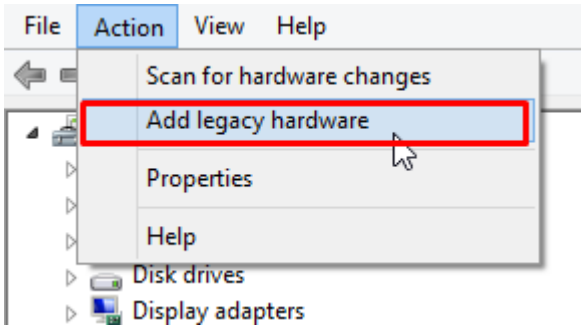
The simplest way to install a virtual network adapter is explained next, for **Windows 10**.

1. On the **Start** menu, select the **Device Manager**.



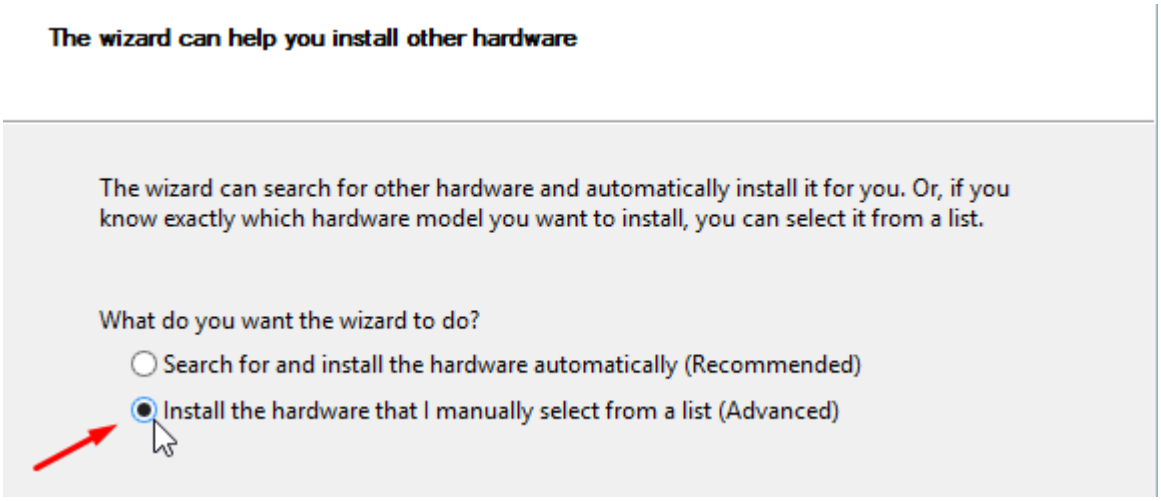
Device Manager

2. Select the **Action - Add legacy hardware** option.



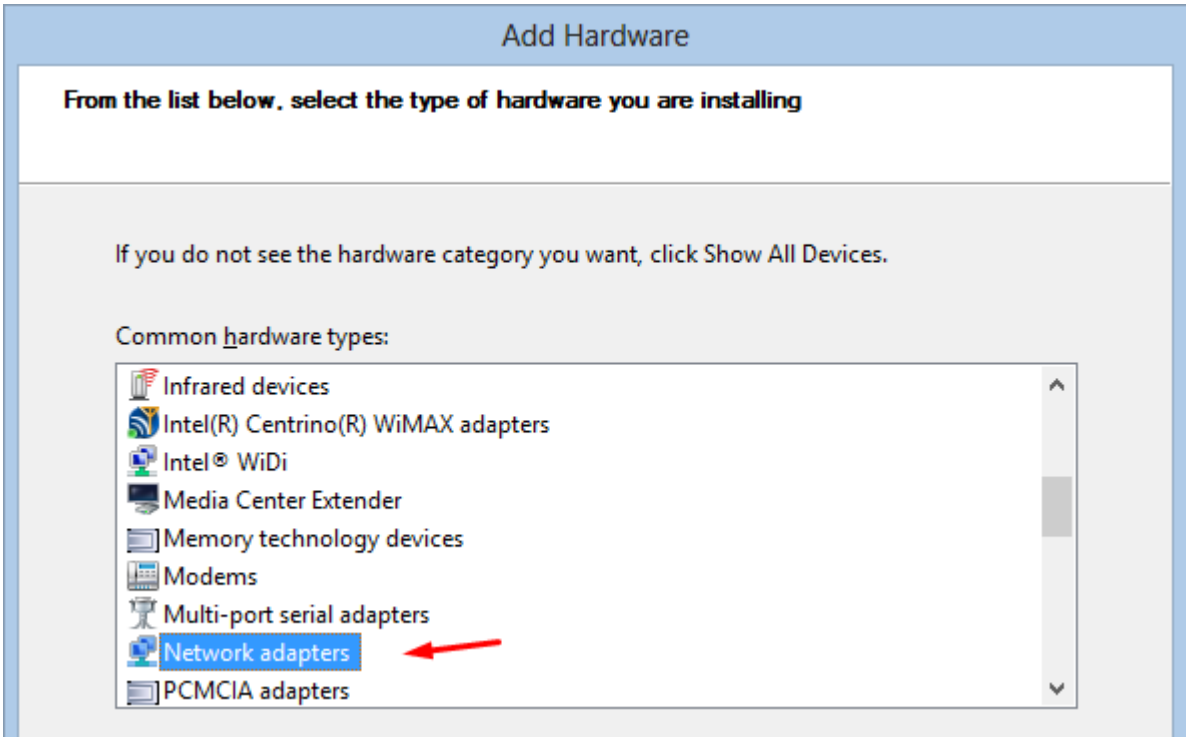
Add legacy hardware option

3. On the Add Hardware window, select the **Install the hardware that I manually select from a list (Advanced)** option and click **Next**.



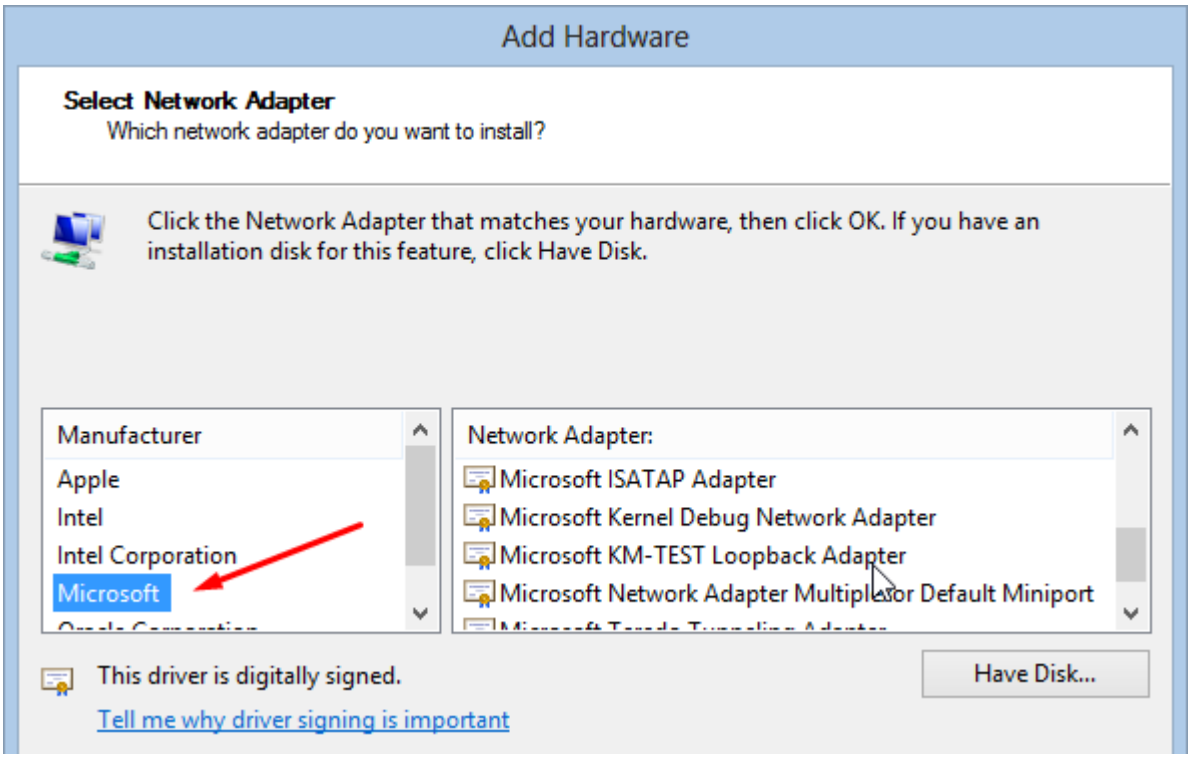
Add Hardware window

4. Select the **Network Adapters** option and click **Next**.



Network adapters

5. Select the **Microsoft** item on the **Manufacturer** list and select the **KM-TEST Loopback** adapter. Click **Next** on the next screens.



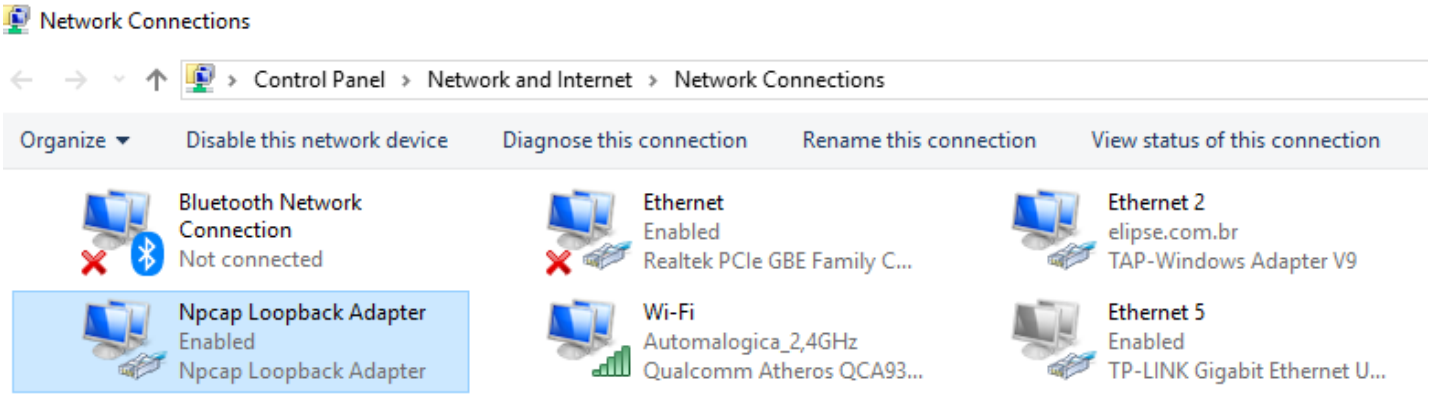
Microsoft network adapters

6. Users can also use the virtual adapter offered during the installation of **Npcap** library, but it is a shortcut to the **KM-Test Loopback** adapter. Users can select this option during the installation of *Npcap* library.

Configuration

Once the virtual adapter is defined, users must enable in the adapter the usage of the **Npcap** sniffer library and define a fixed IP address.

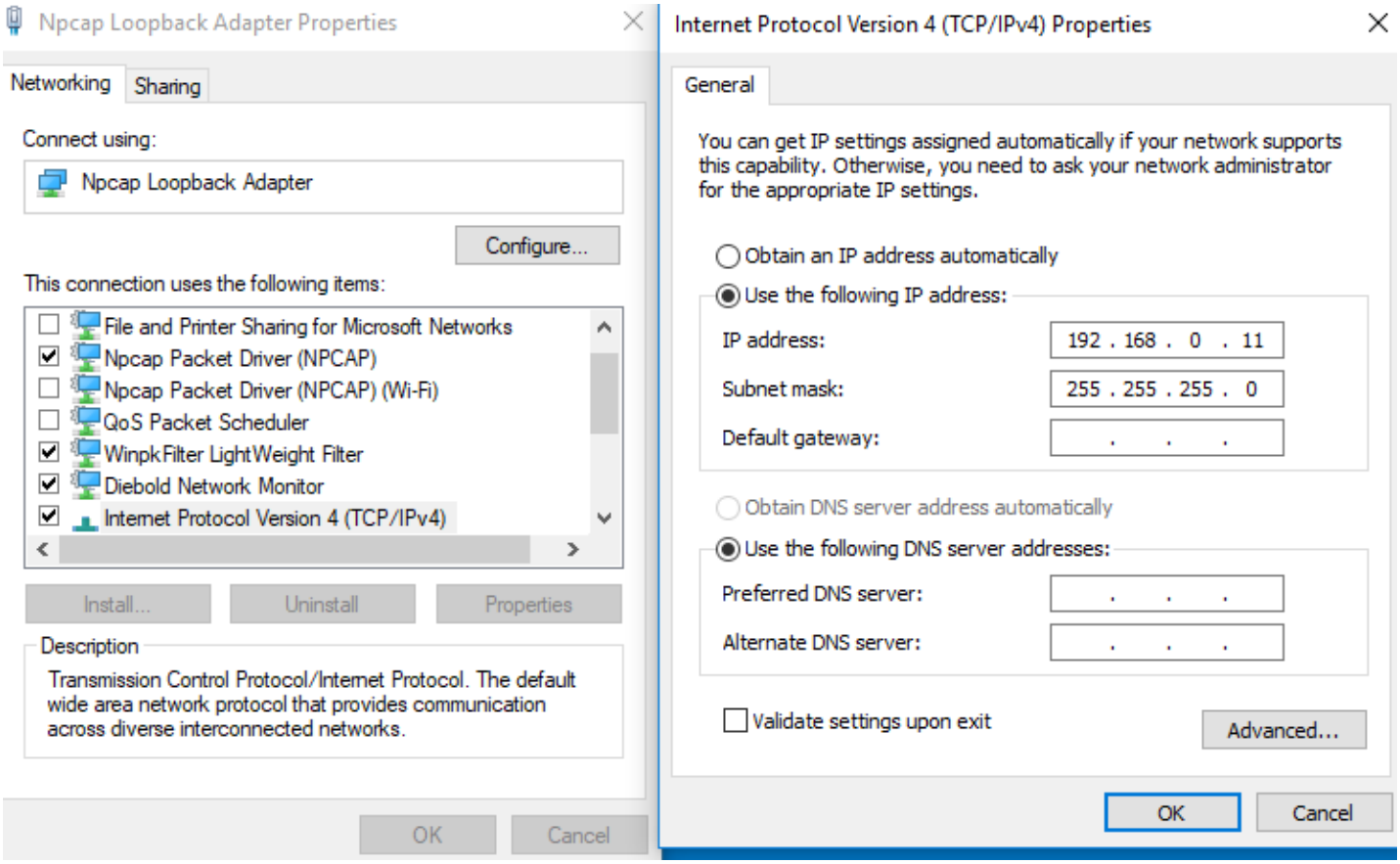
1. To do so, on **Control Panel - Network and Internet - Network Connections**, select the properties of the configured virtual adapter.



Network connections

2. On the properties window, enable the minimum number of options to decrease the network traffic to the 61850 IEDs. The main options are **Npcap Packet Driver** and **TCP/IPv4**.

3. Select the TCP/IP properties and inform a valid and fixed IP address to access the IED network.



Properties of a virtual adapter

4. Click **OK** to apply these changes.

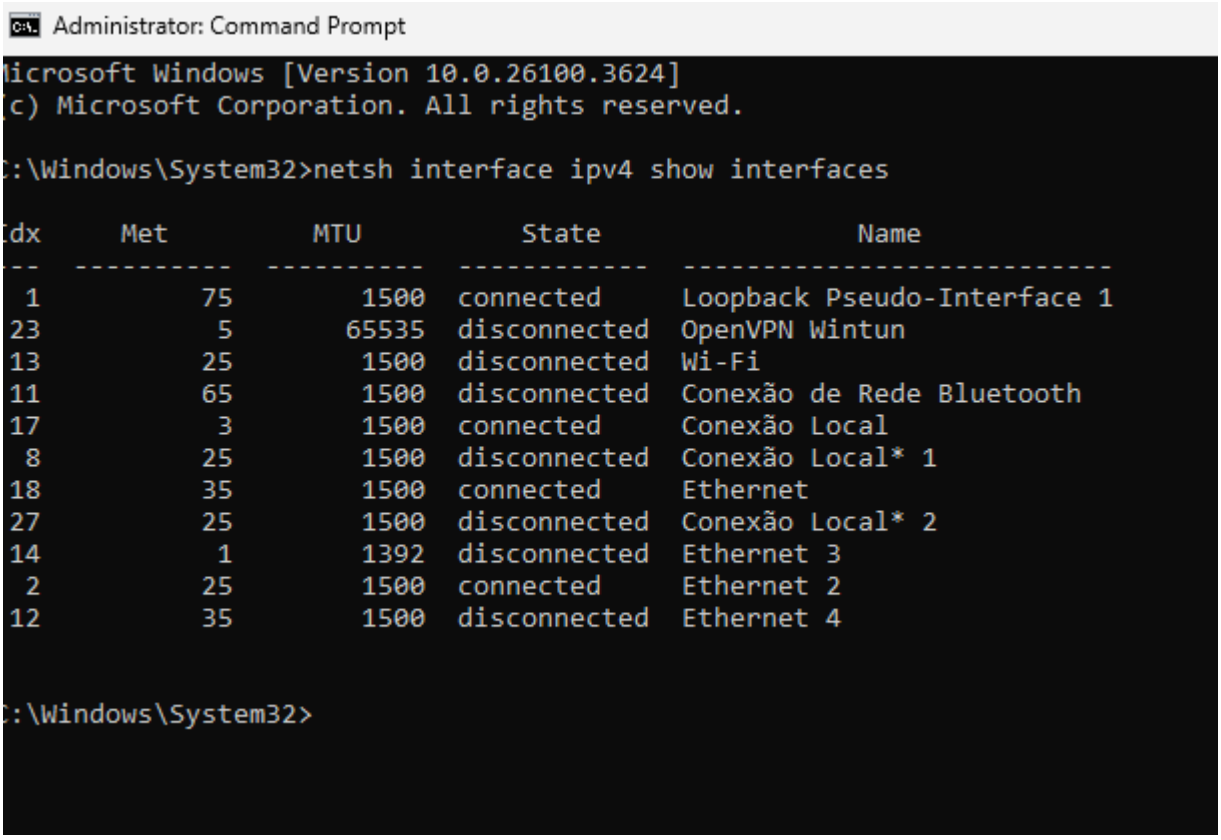
MTU Definition

The virtual interface when installed, defines by default a MTU (Maximum Transmission Unit) pratically unlimited, but the physical interfaces generally use a MTU of 1500 bytes. The PRP Service starting at version 1.0.0.2 sets dynamically the MTU to 1500, so a frame generated at the virtual adapter can be transmitted to the physical adapters normally.

If you´d like to check or adjust the MTU for a particular interface permanently (recommended), follow these steps:

To check the index and MTU of existing interfaces, open a MS-DOS command window in administrator mode and type the following command:

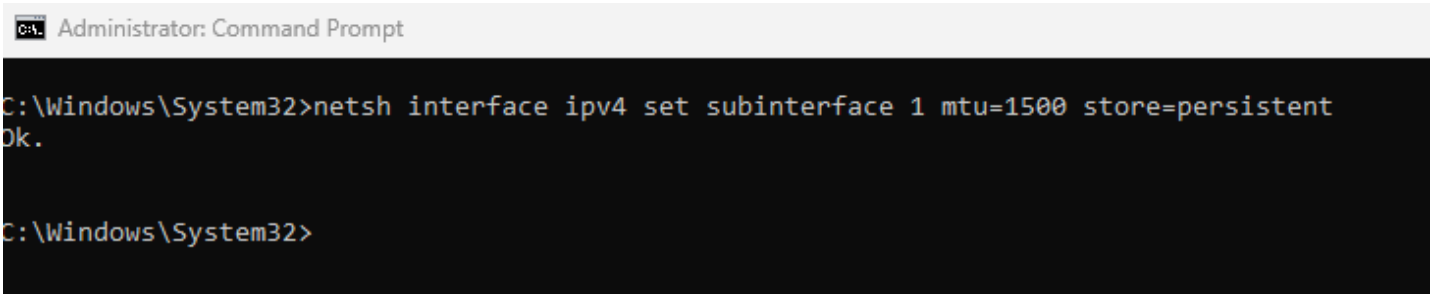
netsh interface ipv4 show interfaces



To change the MTU of an interface, use this command:

netsh interface ipv4 set subinterface Idx mtu=1500 store=persistent

Where Idx is the interface index as shown at the example above.



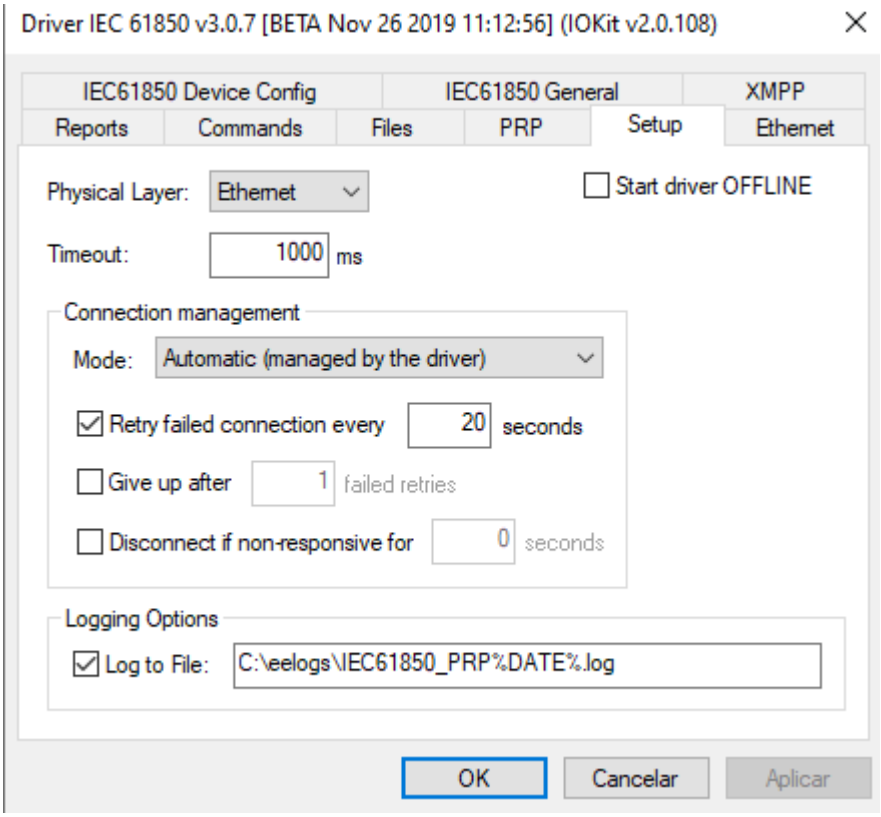
Linking a Virtual Adapter to this Driver

To make sure that 61850 protocol's MMS messages are routed only to the virtual adapter and not to other adapters, users must inform that configuration to this Driver.

The first way to execute this configuration is by manually informing which adapter must be used by this Driver. The second way is by informing in the **ServerName/Interface List** option on the **PRP** tab a list with names of servers that execute this Driver and their respective IP addresses.

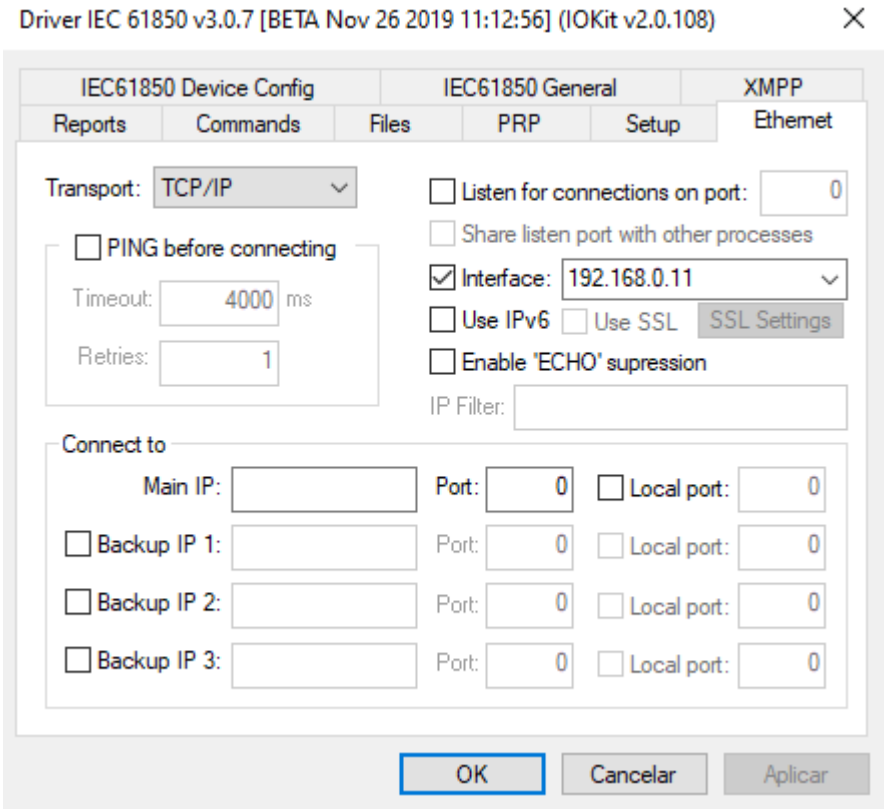
Fixed Definition of a Network Adapter

1. To define a fixed network adapter, open the configuration window of this Driver and, on the **Setup** tab, define the **Physical Layer** option with the value **Ethernet**.



Setup tab

2. On the **Ethernet** tab, select the **Interface** option and, on the list of addresses, select the IP address defined for the virtual adapter. This definition is mandatory for the service to start.



Ethernet tab

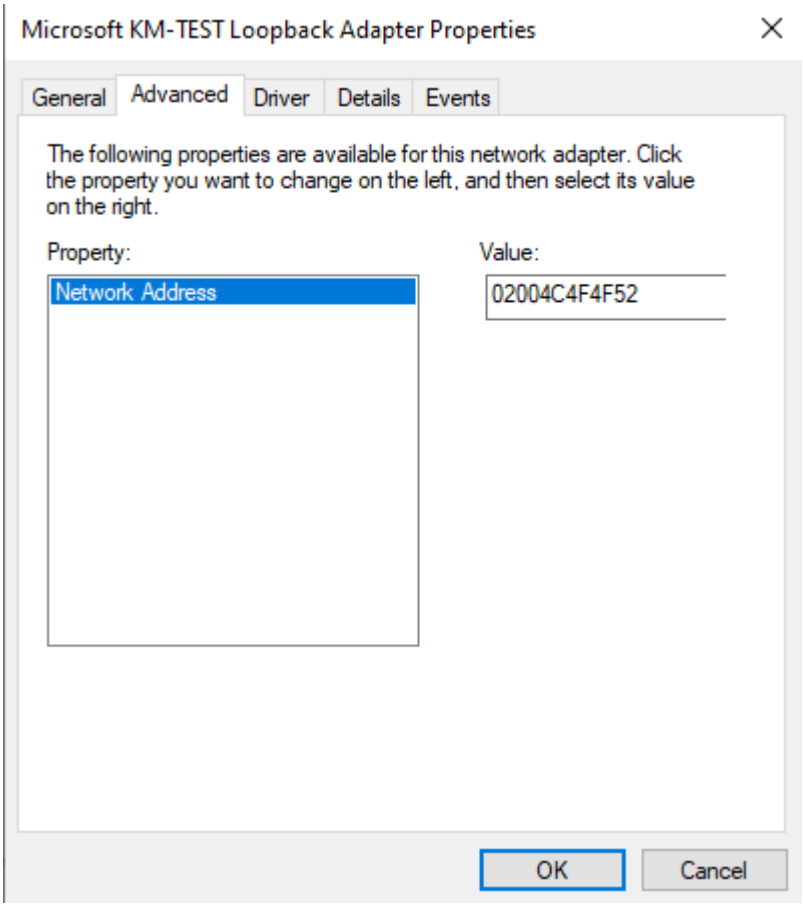
3. Return to the **Setup** tab and configure the **Physical Layer** option with the value **None**. Click **OK** to save this Driver's settings.

Dynamic Definition of a Virtual Adapter

On the **PRP** tab, inform in the **ServerName/Interface List** option a list of server names (DNS) and IP addresses in the format **ServerA:IPA;ServerB:IPB**, so that the same project file containing this Driver can be used without changes in at least two servers. When this Driver starts, it retrieves the name of the local computer and checks whether this list contains an IP address that corresponds to that name. If positive, this IP address is used as the IP address of the virtual interface.

NOTES

- In case of other computers with **Elipse Software**'s **PRP** service on the same network, users must define a different MAC address for each virtual adapter in each computer. This is because the virtual adapter is always installed with a default MAC address, identical among all computers that installed it, such as two redundant **Elipse Power** servers. To change a MAC address, open the configuration window of the network adapter and click **Configure**. Select the **Advanced** tab and, in the **Network Address** option, define a MAC address using a pattern of 12 hexadecimal digits.
- Due to a feature of virtual adapters, when disabling an adapter in Windows during the execution of the **PRP** service and enabling it again, users must restart the service for communication to start working again.



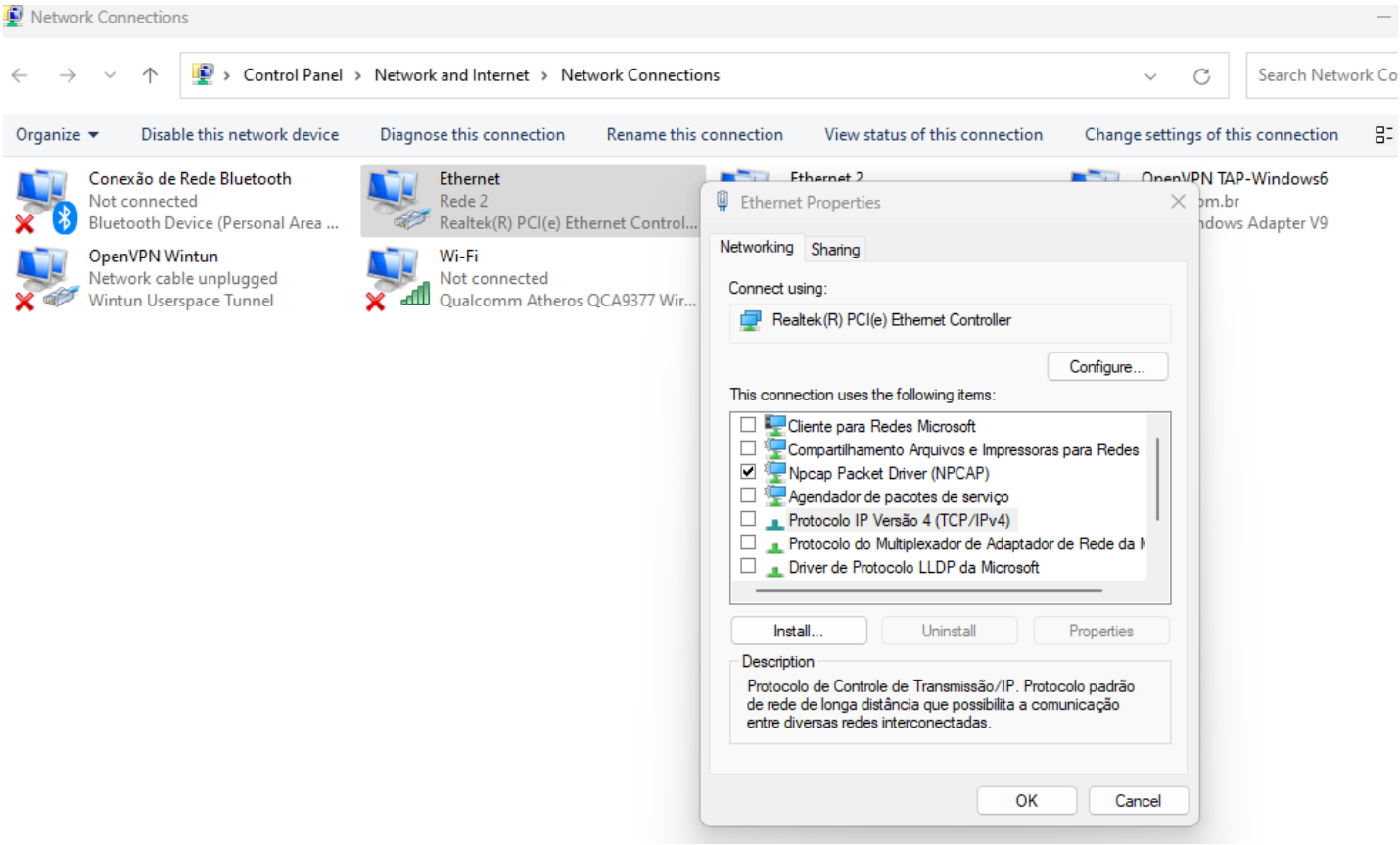
Advanced tab

Physical Adapters

The expected behavior of the **PRP** option is based on two physical network adapters, one connected to network **A** and the other one connected to network **B**. However, the service may start and work temporarily with a single network adapter, as long as it was correctly configured at least once. The configuration of these real network adapters must respect the following rules:

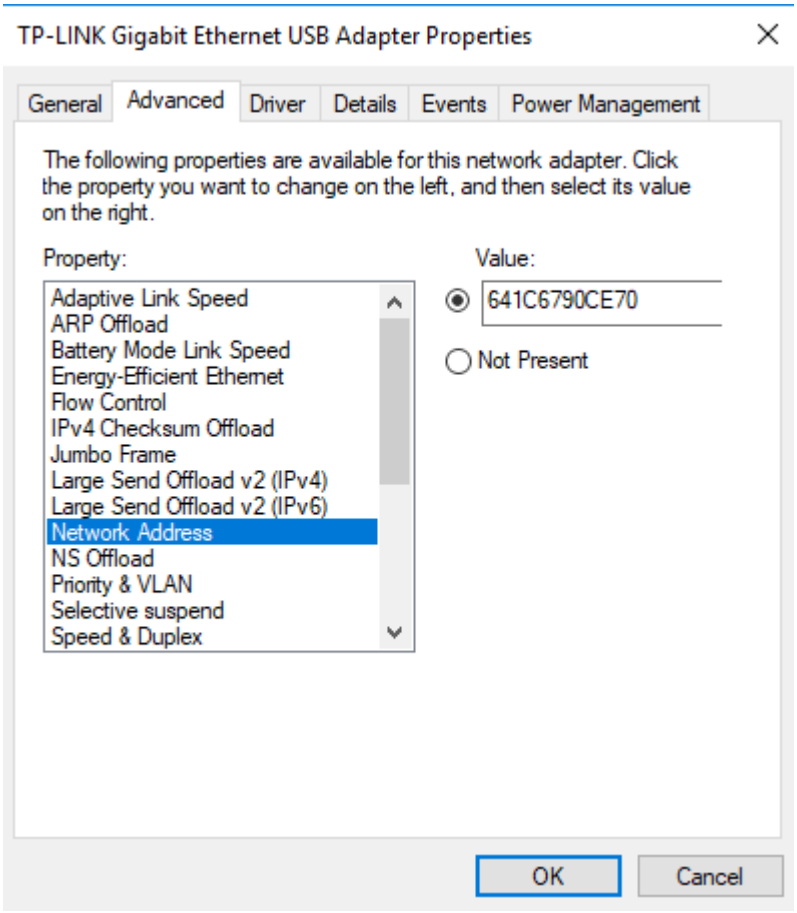
- They cannot have an IP address enabled, as well as other network protocols
- They must have the same MAC address

1. This configuration must be performed on **Control Panel - Network and Internet - Network Connections**, in which all network connections are displayed. Select the properties of the Ethernet adapter that receives network **A** and **disable all network protocols, including TCP/IP**. Leave enabled only the **Npcap** sniffer library and click **OK**.



Properties window of the network adapter

2. Open again the properties window of the network adapter and click **Configure**. Select the **Advanced** tab and, in the **Network Address** option, define a MAC address using a pattern of 12 hexadecimal digits.



Advanced tab

3. Repeat these procedures for the network adapter that receives network **B**.

NOTES

- It is important that both network adapters have the same MAC address. To do so, configure the same MAC address of the first network adapter to the second network adapter or vice versa, or else define a new MAC address for both network adapters.
- To discover the current MAC address of a network adapter, type the **getmac** command on a Windows Command Prompt.

Install, Configure, and Execute a PRP Service

Together with this Driver, starting with version **3.0.42**, the following programs are provided:

- **ElipsePRPSvc.exe**: A service that implements the PRP protocol
- **ElipsePRPTray.exe**: A Windows Tray program that allows configuring and monitoring **ElipsePRPSvc** service

These programs must be copied to a user-defined folder to allow using them on a PRP network. These programs can be located on the same folder of this Driver or on a different location. Users must also copy to that same folder the file et32b.dll, which is found on **Bin** or **Bin32** folder of **Elipse Power** installation.

For this service to work correctly, users must install it on the computer and there also must exist on the same directory an initialization file called ElipsePRPSvc.ini. Both configurations are performed using the **ElipsePRPTray** program.

ElipsePRPTray

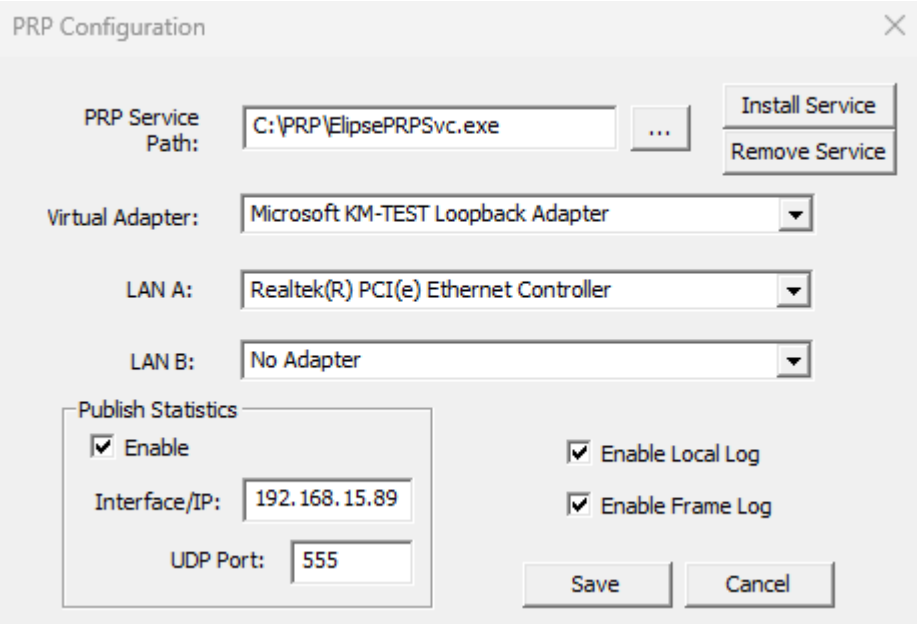
The **ElipsePRPTray** program must be executed as administrator to allow installing and configuring the service. To do so, right-click the program's icon and select the **Run as administrator** option. After opening, this program displays an icon with the following colors:

- **Gray**: Service not installed
- **Red**: Service installed, but stopped
- **Green**: Service executing

Icon of ElipsePRPTray program

When right-clicking this icon, the options on the contextual menu are the following:

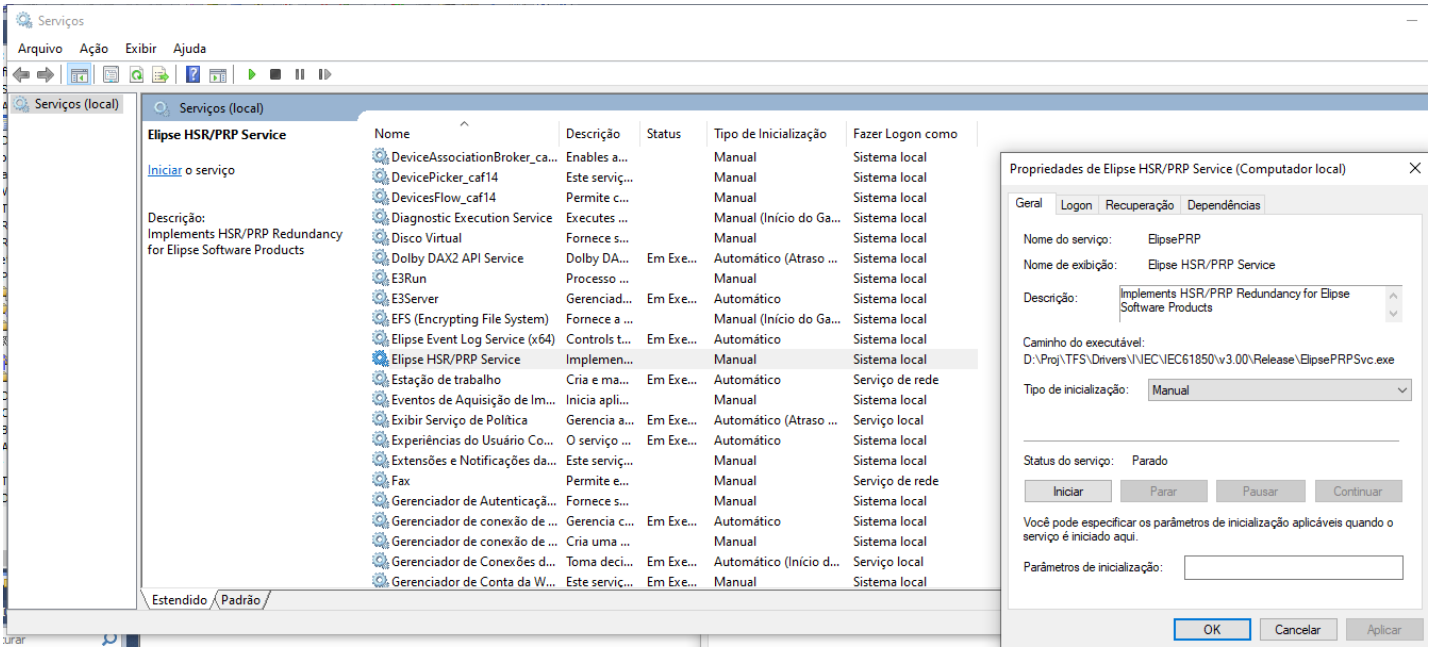
- **About**: Opens a dialog box with information about this product
- **Exit**: Closes this program, without affecting service status
- **Configuration**: Allows installing and configuring the service
- **Run or Stop**: Once installed and configured, users can start or stop the service



Configuration window

Configuration and Installation

1. To install the service, users must inform the folder where the file ElipsePRPSvc.exe was copied. Once defined, click **Install Service**. Analogously, to remove the service, such as when changing the folder, click **Remove Service**.
2. The service is installed with an option for automatic startup, that is, when Windows starts. To change this configuration, open the **Services** program, double-click the **Elipse HSR/PRP Service** service to open the properties window and define the startup type.



Configuration of Elipse PRP service

Service Configuration

After installing the service, users must define the other settings to start.

1. Define, using a list, the network adapters **Virtual**, **LAN A**, and **LAN B**.
2. Define whether this service publishes packet statistics to this Driver, which is performed by sending messages using the UDP/IP protocol. Select an IP address from the local interface and the destination port.
3. Enable service's basic log generation and packet logs.
4. Click **Save** to generate the file ElipsePRPSvc.ini on the installation folder.

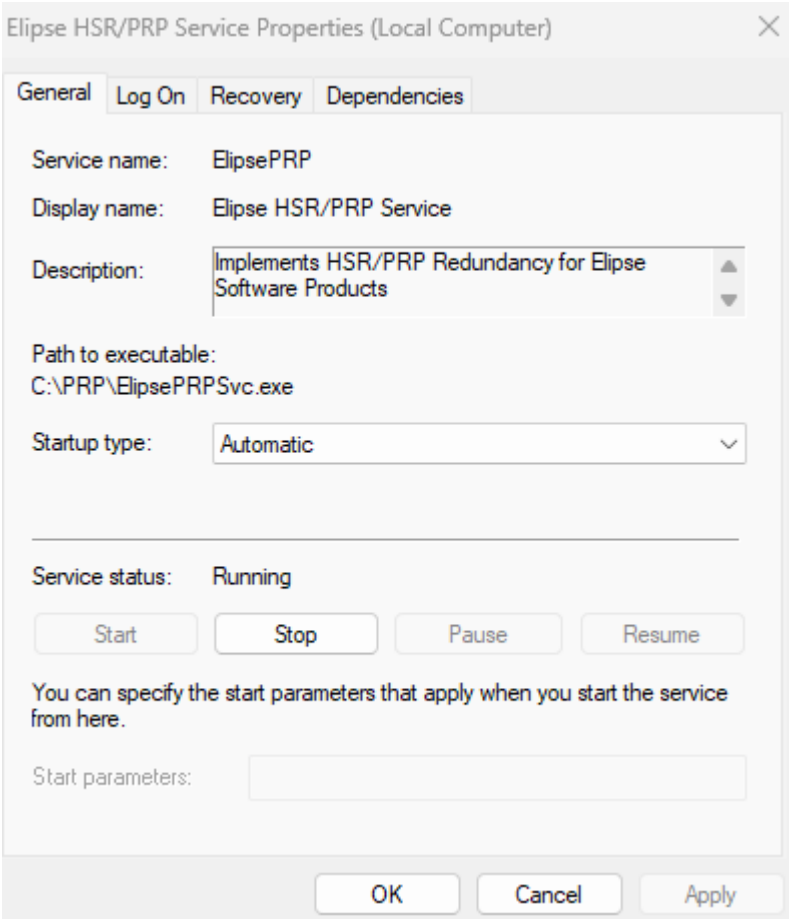
Run or Stop the Service

Once this initialization file is created, users can start the service by selecting the **Run** option. If these settings are correct, the service must start and the icon on the Notification Area must change to green.

Production Environment

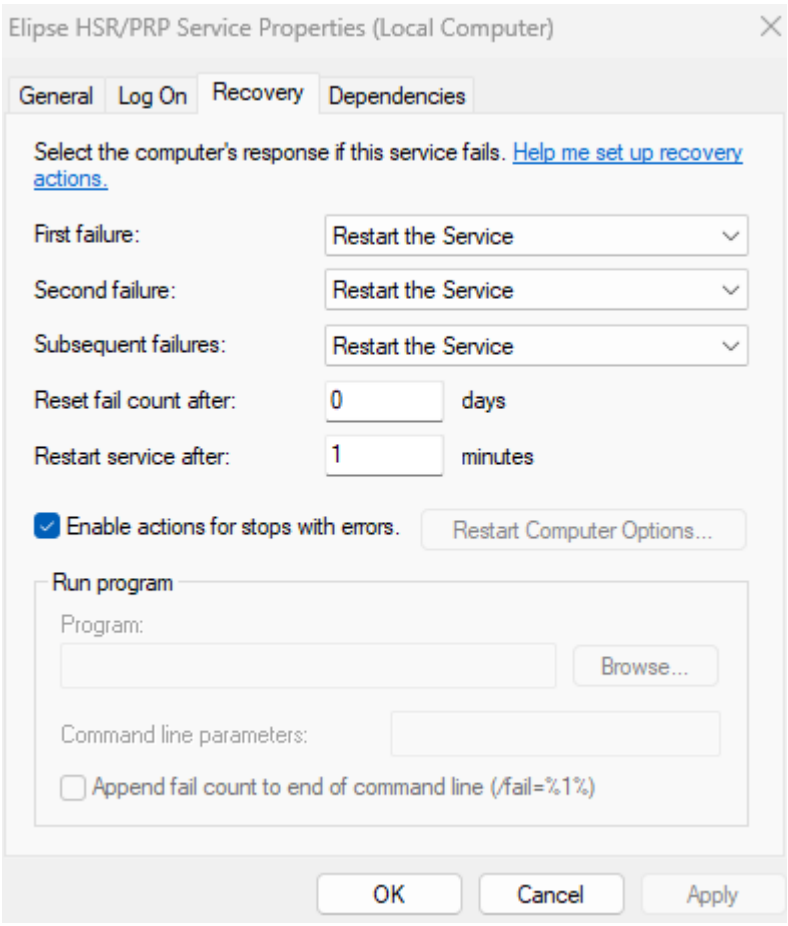
For an installation in a production environment, it is highly recommended that the service is configured with Automatic Startup and also automatically restarted in case of failure. Starting from version 1.0.0.2 of the service, if any of the network interfaces cannot be accessed for more than 1 minute, the service forces an exit with a Windows error code "Adapter Hardware Error" - code 57. It is possible to configure the service to restart automatically as follows:

General Tab: Startup Type: Automatic



Recovery Tab:

- First Failure, Second Failure, Subsequent Failures: Restart the Service
- Check the option "Enable actions for stops with errors"
- Define how many minutes to wait for the restart.



- Disable service logs.

NOTES

- This service's operation is independent from this Driver and from any other **IOKit** application being executed or not.
- To use **ElipsePRPTray** whenever Windows starts, users must add a shortcut to this program on Windows **Start** menu. To do so, right-click this program and select the **Pin to Start** option.
- In case of change the service's folder after installation, users must remove the service from Windows manually. To do so, stop the service and then remove it. Another option can be open a Windows Command Prompt, with an administration profile, and execute the command **sc delete ElipsePRP**.

Monitoring and Statistics

Starting information, such as configuration errors, are registered by the service in a file called ElipsePRPSvc.log generated on the same folder of the service, if this option is enabled.

After starting, and if the monitoring service is enabled, the service sends logs and statistics to the selected UDP port, which are received by this Driver.

Log messages are redirected to this Driver's log with a prefix "PRPService: XXXXXXXX".

Statistical messages are available in the Tags shown on the next figure, which can be created in the application.

PRPStat_LanA_Active	**ElipseClientInfo**	PRPStat_LanA_Active	0	0	0	0		5000	☑
PRPStat_LanB_Active	**ElipseClientInfo**	PRPStat_LanB_Active	0	0	0	0		5000	☑
PRPStat_LanA_TotalSent	**ElipseClientInfo**	PRPStat_LanA_TotalSent	0	0	0	0		5000	☑
PRPStat_LanB_TotalSent	**ElipseClientInfo**	PRPStat_LanB_TotalSent	0	0	0	0		5000	☑
PRPStat_LanA_TotalReceived	**ElipseClientInfo**	PRPStat_LanA_TotalReceived	0	0	0	0		5000	☑
PRPStat_LanB_TotalReceived	**ElipseClientInfo**	PRPStat_LanB_TotalReceived	0	0	0	0		5000	☑

Statistical Tags

The available Tags are described on the next table.

Tags available for statistics

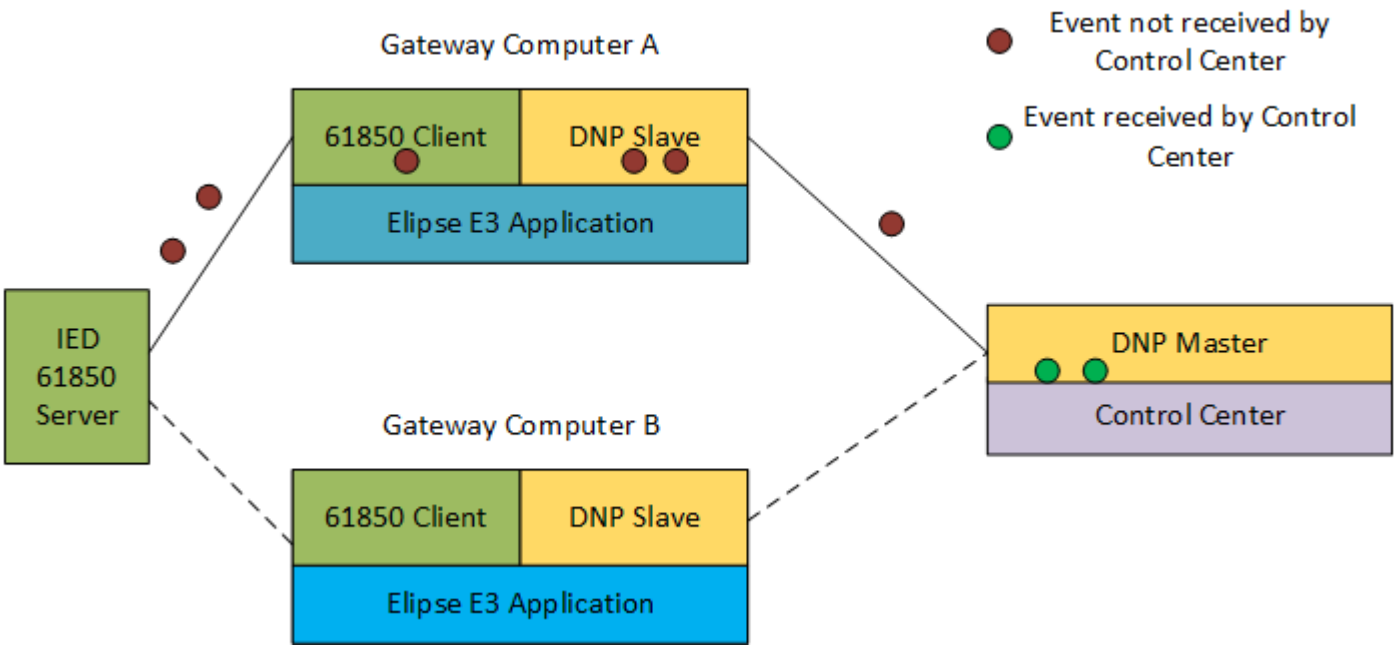
NAME	DESCRIPTION	DEVICE	ITEM
LANA_Active	Indicates whether the **PRP** service is logically active on **LAN A** network adapter. Inverted network adapters, for example, do not allow any of these network adapters to remain active	ElipseClientInfo	PRPStat_LANA_Active
LANB_Active	Indicates whether the **PRP** service is logically active on **LAN B** network adapter. Inverted network adapters, for example, do not allow any of these network adapters to remain active	ElipseClientInfo	PRPStat_LANB_Active
LANA_TotalSent	Indicates the total number of packets sent by the **LAN A** network adapter	ElipseClientInfo	PRPStat_LANA_TotalSent
LANB_TotalSent	Indicates the total number of packets sent by the **LAN B** network adapter	ElipseClientInfo	PRPStat_LANB_TotalSent
LANA_TotalReceived	Indicates the total number of packets received by the **LAN A** network adapter	ElipseClientInfo	PRPStat_LANA_TotalReceived
LANB_TotalReceived	Indicates the total number of packets received by the **LAN B** network adapter	ElipseClientInfo	PRPStat_LANB_TotalReceived

Define a reasonable scan interval for these Tags, to avoid polluting this Driver's log. As a suggestion, a reasonable value can be between 5000 and 10000 ms, that is, between 5 (five) and 10 seconds.

Gateway Configuration

In gateway applications, in which it is necessary to send data from this Driver to other protocols, such as DNP 3.0 or IEC 104, there may have situations, in case of restarting an application or switching to a redundant application, or Hot-Standby, in which there is a loss or duplication of events.

This may occur when an event was received by this Driver and forwarded, but it has not been received yet by the remote master in another protocol, according to the example on the next figure.



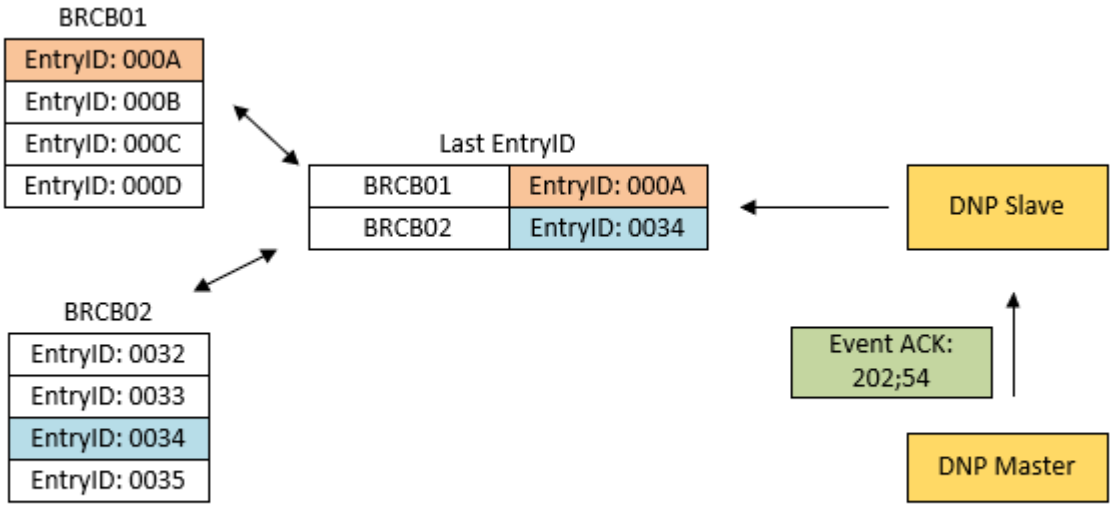
Gateway configuration

The previous image shows that events being transferred and pending of transmission in several gateway components (61850 Server, 61850 Client, DNP Slave, and DNP Master) may exist. This queue of pending events may change its size depending on the configuration of these components and network architecture, among other factors.

When there is a switch-over from **Gateway A** to **Gateway B**, for example, the events pending transmission can be either lost or duplicated, depending on the application's configuration. This occurs because the 61850 protocol allows receiving again **Buffered Report** messages, by programming the **EntryID** field of each **Report**, which is a unique identifier of each message sent.

This Driver already provides a feature to generate files with the last **EntryID** of each **Report**, which is used to enable **Reports** when this Driver starts again. However, these **EntryIDs** are updated based on the moment the messages are received by this Driver, and not by the real reception of data in an external system, in the case of the previous example, the control center.

This way, when enabling the Gateway feature using the **Wait for Gtw Ref on Startup** option, this Driver waits for an **Elipse Power** application to inform a **String** representing the address of a 61850 Tag inside another Driver, such as DNP or IEC 104 Slave. When this other Driver receives the confirmation that an event was delivered to the control center, DNP or IEC 104 Master, it can inform back this Driver about this event, so this Driver can position **EntryID** controls forward so that, in case of a new switching, the flow of events continue from this point on, thus avoiding event loss or repetition.



Management of EntryIDs

The steps to configure this resource are the following:

1. Enable the options **Check BRCB Entry ID**, **R/W Entry ID File (.EID)**, and **Wait for Gtw Ref on Startup**. Also, if the IED uses the same **Reports** for both gateways, users must evaluate using a fixed list of **Reports** in the **User Defined Report List** option, thus avoiding that a redundant application tries to enable unexpected **Reports**.

2. On this Driver, create the **FinishedGtwRef** Tag and the **GatewayRef** and **GatewayEventOK** Block Tags, according to the next figure. These Tags are described on topic **Tag Reference - Internal Tags**.

IEC61850			0	0	0	0				
FinishedGtwRef		FinishedGtwRef	0	0	0	0		1000	☐	☐
GatewayRef		GatewayRef	0	0	0	0	4	1000	☐	☐
IEDName							0			
LDName							1			
Object							2			
Ref							3			
GatewayEventOK	**IED00001**	GatewayEventOK	0	0	0	0	4	1000	☐	☐
Ref							0			
Value							1			
TSValid							2			
Time							3			

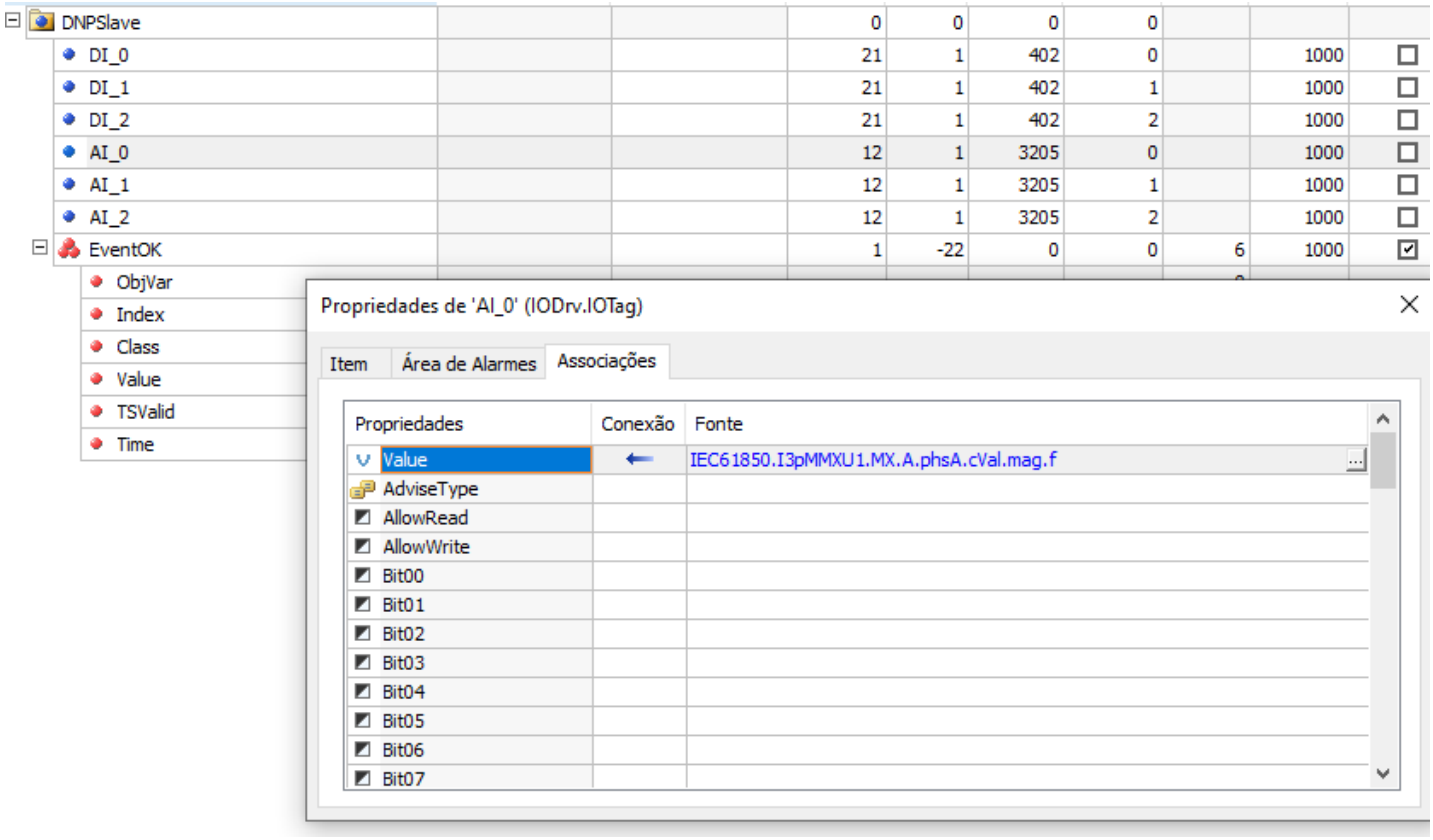
Tags created in this Driver

3. Taking the DNP Slave Driver as an example, create a Block Tag with the *N2* parameter equal to -22, which receives on the **OnRead** event a list with all events confirmed by the Master. Please check DNP Driver's documentation for more information.

DNPSlave			0	0	0	0				
DI_0			21	1	402	0		1000	☐	☑
DI_1			21	1	402	1		1000	☐	☑
DI_2			21	1	402	2		1000	☐	☑
AI_0			12	1	3205	0		1000	☐	☑
AI_1			12	1	3205	1		1000	☐	☑
AI_2			12	1	3205	2		1000	☐	☑
EventOK			1	-22	0	0	6	1000	☑	☐
ObjVar							0			
Index							1			
Class							2			
Value							3			
TSValid							4			
Time							5			

Block Tag on the DNP Slave Driver

4. The Tags on the DNP Slave Driver must receive Links from the Tags of this Driver. The next image shows an example in which the **AI_0** Tag, a 3205 DNP object with index 0 (zero), is fed by a phase A current from this Driver.



Tag properties

5. The application, when starting, must initialize its references, passing a text that represents the DNP address for each Tag of this Driver needed to perform a control of **EntryIDs**. But that writing can only be performed at the moment this Driver has its Logical Devices created, otherwise the writing fails. To solve this problem, the next example shows an Internal Tag that tries to send the configuration every 5 (five) seconds. If that operation is successful, it stops trying. At the end of this process, the **FinishedGtwRef** Tag is written.

6. Also in this example, the reference **String** was retrieved by joining the object and variation from the DNP Tag plus the index. On the DNP Slave Driver, these values are retrieved from the *N3* and *N4* parameters, respectively.

- **Initial Value of the InitGateway Tag**: True (Boolean)
- **WhileNotOK event**: Expression: Value Repeat event every 5000 ms

```
Sub InitGateway_WhileNotOK()
  Set DNPSlave = Application.GetObject("DNPSlave")
  Set Ref = Application.GetObject("IEC61850").Item("GatewayRef")
  bFirst = FALSE
  For Each tag In DNPSlave
    If TypeName(tag) = "IOTag" Then
      If tag.N2 = 1 Then
        Source = tag.Links.Item("Value").Source
        Set s = Application.GetObject(Source)
        arr = Split(s.ParamDevice,":")
        Ref.Item("IEDName").Value = arr(0)
        Ref.Item("LDName").Value = arr(1)
        Ref.Item("Object").Value = s.ParamItem
        Ref.Item("Ref").Value = CStr(tag.N3) + ";" + CStr(tag.N4)
        If Not bFirst Then
          bFirst = Ref.Write(1) 'sync
          If Not bFirst Then Exit Sub
        Else
          Ref.Write 2 'async unconfirmed
        End If
      End If
    End If
  Next
  res = Application.GetObject("IEC61850").Item("FinishedGtwRef").WriteEx(2) 'async unconfirmed
  If res Then Value = FALSE
End Sub
```

7. When the **EventOK** Block Tag is read on the DNP Slave Driver, it means a new event was received by the DNP Master. The next script retrieves information from this event and forwards it to this Driver. In the specific case of a DNP Slave Driver, this Block Tag returns some information about that event, such as object and variation, index, DNP class, value, valid timestamp, and timestamp. All these parameters are sent to this Driver by writing to the **GatewayEventOK** Block Tag, except the DNP class, which is not relevant.

```
Sub EventOK_OnRead()
  Set IECBlock = Application.GetObject("IEC61850.GatewayEventOK")
  IECBlock.Item("Ref").Value = CStr(Item("ObjVar").Value) + ";" + CStr(Item("Index").Value)
  IECBlock.Item("Value").Value = Item("Value").Value
  IECBlock.Item("TSValid").Value = Item("TSValid").Value
  IECBlock.Item("Time").Value = Item("Time").Value
  IECBlock.Write 2 'async
End Sub
```

8. At this moment, this Driver positions the **EntryIDs** at the next pending event. To synchronize the **EntryIDs** with the redundant computer, users must follow the steps indicated on topic **Redundancy - Using EID Files**.

9. To position the **EntryIDs**, this Driver uses the following methodology:

- When a new **Report** is received, checks if any object had a variation in value or quality. If positive, a new event is added to an internal individual list per object.
- When an event is confirmed by an external Driver and received by the **GatewayEventOK** Block Tag, the internal list is then compared with the removed event as follows:
  - If the timestamp is not valid, that is, a Tag on a DNP or IEC 104 Driver is not using a timestamp, all events are removed from the list
  - If the timestamp is valid, events are removed if they are equal or older than the informed event

To avoid unnecessary processing, it is recommended to only add Tag references that cannot have lost or duplicated changes, typically Digital or simple or double state Tags with timestamp. It is also recommended that, in this case, the **Datasets** linked to **Reports** be separated by type of Tag, such as not placing Digital and Analog Tags on the same **Dataset**, to avoid receiving old and unnecessary analog values.

Dynamic Configuration

This Driver allows some configuration parameters to be informed at run time by script, by using a command for writing parameters from the **IOKit** library. These parameters are described on the next table.

Configuration parameters

PROPERTY	PARAMETER	DATA TYPE
Total Device Number	IEC61850.DeviceCount	DWORD
Server	IEC61850.Device[%u].Name	STRING
IP	IEC61850.Device[%u].IP	STRING
Backup IP	IEC61850.Device[%u].IPBackup	STRING
PSEL	IEC61850.Device[%u].PSel	DWORD
SSEL	IEC61850.Device[%u].SSel	DWORD
TSEL	IEC61850.Device[%u].TSel	DWORD
Rem AE Qual	IEC61850.Device[%u].AEQualifier	DWORD
Rem AP ID	IEC61850.Device[%u].AppID	STRING
Disable	IEC61850.Device[%u].Disable	BYTE
Use Backup IP	IEC61850.Device[%u].UseIPBackup	BYTE
Save Comtrade (Per Device)	IEC61850.Device[%u].SaveComtrade	BYTE
Delete Comtrade Files (Per Device)	IEC61850.Device[%u].DeleteComtrade	BYTE
Browse Root Folder Only (Per Device)	IEC61850.Device[%u].BrowseRootFolderOnly	BYTE
Comtrade Path (Per Device)	IEC61850.Device[%u].ComtradePath	STRING
Use Fixed Comtrade Path On Device (Per Device)	IEC61850.Device[%u].UseFixedPathOnDevice	BYTE
Name+Index File Transfer	IEC61850.Device[%u].NameAndIndexFileTransfer	BYTE
File Root Name	IEC61850.Device[%u].FileTransferRootName	STRING
Prefer Buffered Report Control Blocks	IEC61850.UseReports	BYTE
User-Defined Report List	IEC61850.UserReportList	BYTE
Poll Tags not found in any report	IEC61850.PollTags	BYTE
Local P Selector	IEC61850.LocalPSel	DWORD
Local S Selector	IEC61850.LocalSSel	DWORD
Local T Selector	IEC61850.LocalTSel	DWORD
Local AE Qualifier	IEC61850.LocalAEQualifier	DWORD
Local App ID	IEC61850.LocalAppID	STRING
RFC 1006 Source TSAP	IEC61850.SourceTSAP	DWORD
App Category	IEC61850.AppCategory	STRING
Polled Intg Rpt	IEC61850.RptGI	DWORD

PROPERTY	PARAMETER	DATA TYPE
Auto Intg Prt	IEC61850.IntgPd	DWORD
Use Quality change trigger	IEC61850.UseQChgTrgOps	
Conformance Blocks (internal use)	IEC61850.CBB	STRING
Services (internal use)	IEC61850.Services	STRING
LD File Path	IEC61850.LDPath	STRING
Comtrade Path	IEC61850.ComtradePath	STRING
Comtrade Directory on Device (internal use)	IEC61850.ComtradeDir	STRING
Msg Timeout	IEC61850.Timeout	DWORD
Full Log Details	IEC61850.DetailedLog	BYTE
Save Comtrade Files	IEC61850.SaveComtrade	BYTE
Save Comtrade Per IED	IEC61850.SaveComtradePerIED	BYTE
Delete Files after upload	IEC61850.DeleteComtrade	BYTE
Comtrade Directory Check Interval	IEC61850.CheckComtrade	BYTE
Comtrade Upload Interval	IEC61850.UploadComtrade	DWORD
Check Report Time of Entry	IEC61850.RedundantBRCB	BYTE
Status Check	IEC61850.StatusCheck	DWORD
Asynchronous Write	IEC61850.AsyncWrite	BYTE
Use Single Tag Command Alias	IEC61850.SingleTagCmdAlias	BYTE
Apply Local Time Offset to Timestamps	IEC61850.ApplyLocalTime	BYTE
Use Cmd Tag N1 as check condition	IEC61850.SingleTagCmdN1	BYTE
No LD Database Scan	IEC61850.NoLDScan	BYTE
Browse Root Folder Only	IEC61850.ComtradeBrowseRootFolder Only	BYTE
Use Exclusive URCB	IEC61850.ReserveURCB	BYTE

An example of dynamic COMTRADE configuration is described next:

1. This Driver must have all IEDs already declared, and they must have their **Disable** option configured with value 1 (one). This avoids that communication starts with wrong parameters.

2. This Driver must also have its **Define Parameters Per IED** option configured.

3. On the **AfterStart** event, there must have a script that changes the parameters of each IED. These parameters must be written via **IOKit** library. These parameters are informed on the next script, where **%u** must be changed by the index of the IED on this Driver, starting with 0 (zero). These parameters are available on this Driver's log.

```
"IEC61850.Device[%u].SaveComtrade"
"IEC61850.Device[%u].DeleteComtrade"
"IEC61850.Device[%u].BrowseRootFolderOnly"
"IEC61850.Device[%u].ComtradePath"
"IEC61850.Device[%u].UseFixedPathOnDevice"
"IEC61850.Device[%u].NameAndIndexFileTransfer"
"IEC61850.Device[%u].FileTransferRootName"
```

4. Consider that the **ComtradePath** and **FileTransferRootName** parameters are texts, and if their individual value, per IED, is not configured, the default value from the configuration window is considered. The default value for other properties is 0 (zero, False).

5. After writing the parameters, enable each IED by writing the value 0 (zero) in the "IEC61850.Device[%u].Disable" parameter.

6. Start communication with this Driver by writing the value 1 (one) to the special Tag **ServerInitialBuild**. This Tag creates the Hosts that are enabled and that have not been created yet.

```
' Exemple considering an 8-Element Array
' Empty indexes are ignored by this Driver
Dim Arr(8)
Arr(1) = Array("IEC61850.Device[0].SaveComtrade",1) ' This IED retrieves Comtrades
Arr(2) = Array("IEC61850.Device[1].SaveComtrade",0) ' This IED does not retrieve Comtrades
Arr(3) = Array("IEC61850.Device[2].SaveComtrade",1) ' This IED retrieves Comtrades
Arr(4) = Array("IEC61850.Device[2].NameAndIndexFileTransfer",1) ' This IED retrieves Comtrades using
RootName + Index
Arr(5) = Array("IEC61850.Device[0].Disable",0) ' Enables IED
Arr(6) = Array("IEC61850.Device[1].Disable",0) ' Enables IED
Arr(7) = Array("IEC61850.Device[2].Disable",0) ' Enables IED
Driver.Write -1,0,0,3 Arr ' Writes parameters
Driver.Item("ServerInitialBuild").WriteEx 1 ' Considering the existence of this Tag with Item
parameter = ServerInitialBuild,
'When writing 1 (one) it starts communication with all enabled IEDs
```

7. When stopping a Driver, these parameters are not saved, therefore the **AfterStart** event must be kept, so that when starting a Driver, IEDs are correctly configured.

Limitations of this Driver

The IEC 61850 Driver is provided with the following types of licenses by **Elipse Software**:

- **IEC61850.dll**: Allows communication with up to 25 relays (IEDs), to ensure communication performance. For more relays, use a new instance of this Driver
- **Power_IEC61850.dll**: Allows communication with a single relay (IED), using a standard license that allows replacement by another power system Driver. For a full list of Drivers sharing this license, please check with **Elipse Software**

Documentation of I/O Interfaces

This section contains the documentation of I/O Interfaces referring to the **IEC61850** Driver.

Configuration of a Driver

I/O Interface configuration is performed on a Driver's configuration dialog box. To access the configuration of this dialog box in **Elipse E3** in version 1.0, follow these steps:

1. Right-click a Driver object (IODriver).
2. Select the **Properties** item on the contextual menu.
3. Select the **Driver** tab.

4. Click **Other parameters**.

In **Elipse E3** version 2.0 or later, click **Configure driver** on a Driver's toolbar. In **Elipse SCADA**, follow these steps:

1. Open the Organizer.
2. Select a Driver on Organizer's tree.
3. Click **Extras** on the **Driver** tab.

Currently, an I/O Interface allows opening only one connection for each Driver. This means that, if users want to access two serial ports, they must add two Drivers to an application and then configure each one of these Drivers for each serial port.

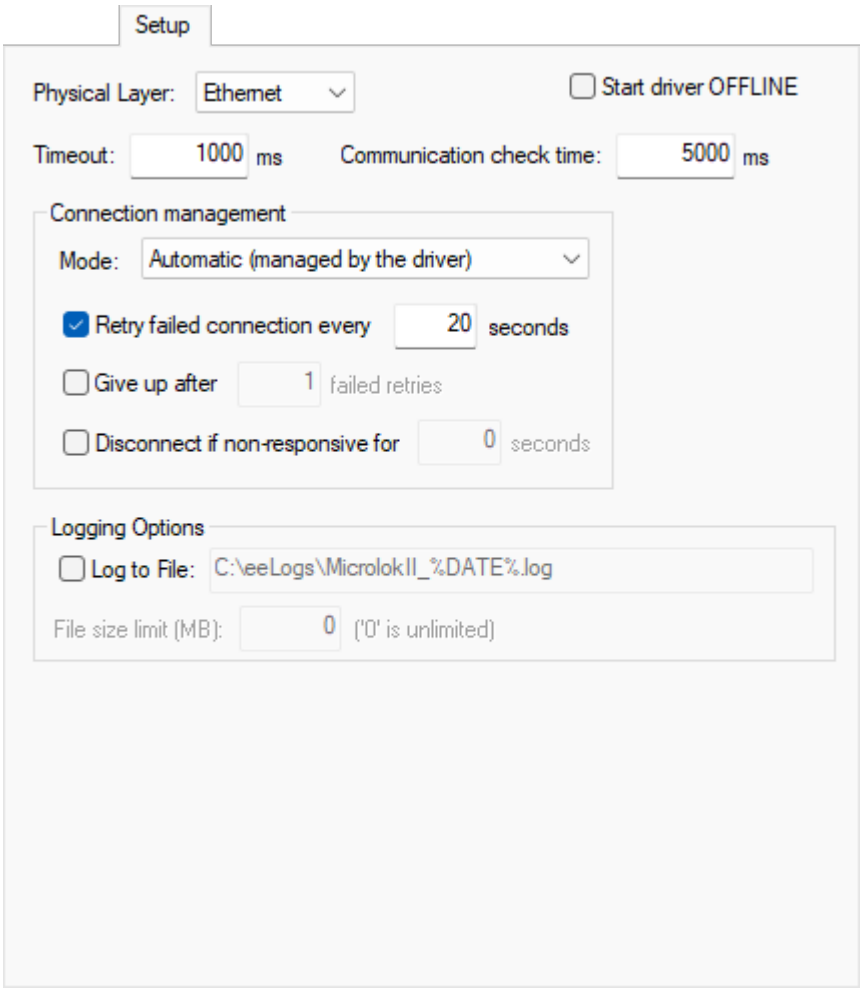
Configuration Dialog Box

The dialog box of I/O Interfaces allows configuring the I/O connection used by a Driver. This dialog box contains the **Setup**, **Serial**, **Ethernet**, **Modem**, and **RAS** tabs, described on the next topics. If a Driver does not implement a specific I/O connection, its corresponding tab is not available for configuration. Some Drivers may contain additional tabs, specific for that Driver, on the configuration dialog box.

Setup Tab

The **Setup** tab contains general configurations of a Driver. This tab is divided into the following groups:

- **General configurations**: Configurations of a Driver's physical layer, time-out, and initialization mode
- **Connection management**: Configurations on how the I/O Interface keeps a connection and which recovery policy is used on failure
- **Logging options**: Controls the generation of log files



Setup tab

General options on the Setup tab

OPTION	DESCRIPTION
Physical Layer	Select the physical layer on a list. Available options are **Serial**, **Ethernet**, **Modem**, and **RAS**. The selected interface must be configured on its specific tab
Timeout	Configure a time-out, in milliseconds, for the physical layer. This is the amount of time an I/O interface waits to receive any byte from the reception's buffer
Communication check time	Set the time, in milliseconds, to define the interval at which communication is considered to be in an inactive state. As long as an I/O Driver receives valid data, its communication state is considered active. However, if during operation an I/O Driver does not receive valid data inside this period of time, the state is considered inactive. The communication state is shown in the **IO.CommunicationStatus** Tag
Start driver OFFLINE	Select this option so that a Driver starts in **Offline** mode or stopped. This means that the I/O interface is not created until this Driver is configured to **Online** mode by using a Tag in an application. This mode enables a dynamic configuration of an I/O interface at run time

Options on the Connection management group

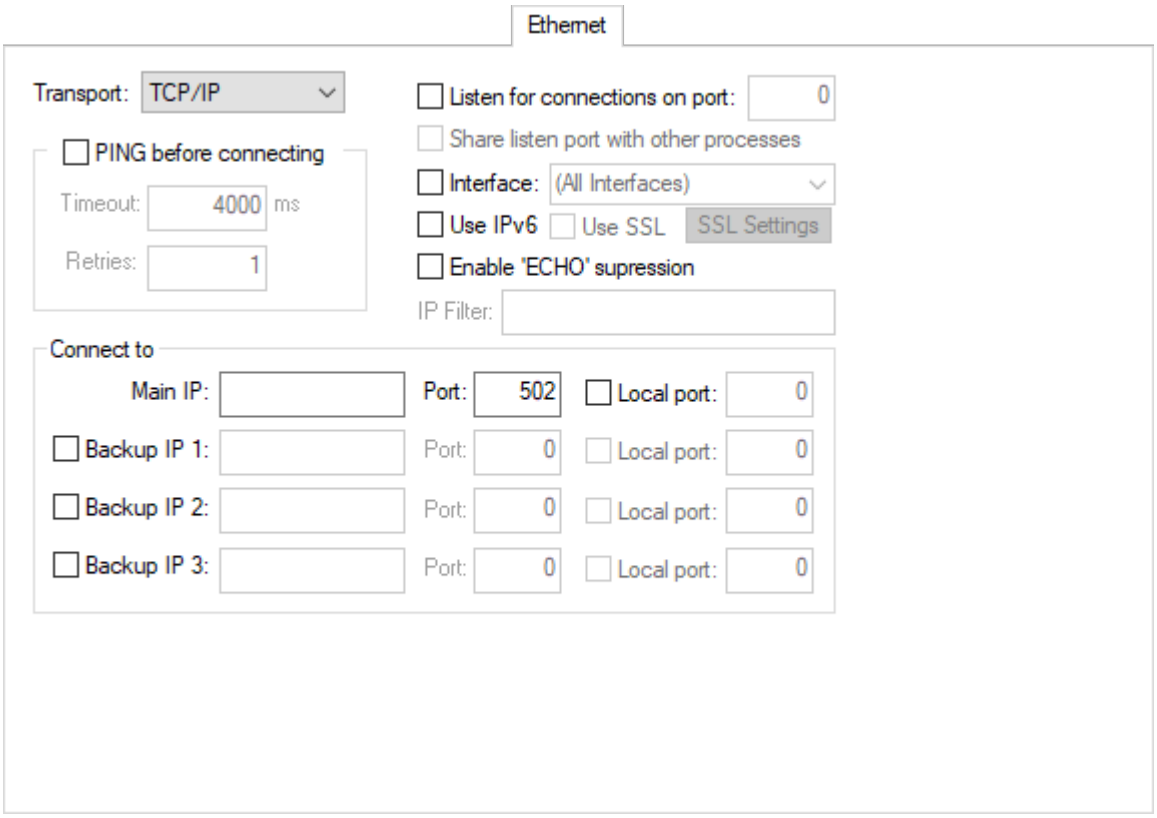
OPTION	DESCRIPTION
Mode	Selects a management mode of a connection. Selecting the **Automatic** option allows a Driver to manage the connection automatically, as specified in the next options. Selecting the **Manual** option allows an application to fully manage a connection
Retry failed connection every ... seconds	Select this option to enable a Driver's connection retry in a certain interval, in seconds. If the **Give up after failed retries** option is not selected, this Driver keeps retrying until a connection is performed, or until the application is stopped
Give up after ... failed retries	Enable this option to define a maximum number of connection retries. When the specified number of consecutive connection retries is reached, a Driver goes to the **Offline** mode, assuming that a hardware problem was detected. If a Driver establishes a successful connection, the number of unsuccessful retries is cleared. If this new connection is lost, then the retry counter starts at zero
Disconnect if non-responsive for ... seconds	Enable this option to force a Driver to disconnect if no byte was received by the I/O interface during the specified time-out, in seconds. This time-out must be greater than the time-out configured in the **Timeout** option

Options on the Logging Options group

OPTION	DESCRIPTION
Log to File	Enable this option and configure the name of a file to write a log. Log files can be large, so use this option for short periods of time, only for testing and debugging purposes. If the **%PROCESS%** macro is used in the log file name, it is replaced by the identifier of the current process. This option is particularly useful when using several instances of the same Driver in **Elipse E3**, thus allowing each instance to generate a separate log file. For example, when configuring this option with value "c:\e3logs\drivers\sim_%PROCESS%.log", it generates a file named c:\e3logs\drivers\sim_00000FDA.log for process **0FDAh**. Users can also use the **%DATE%** macro in the file name. In this case a log file is generated every day, in the format **aaaa_mm_dd**. For example, when configuring this option with value "c:\e3logs\drivers\sim_%DATE%.log", it generates a file named c:\e3logs\drivers\sim_2005_12_31.log in 12/31/2005 and a file named c:\e3logs\drivers\sim_2006_01_01.log in 01/01/2006. Similarly, the **%DATE_HOUR%** macro generates one log file per hour, in the format **aaaa_mm_dd_hh**
File size limit (MB)	Configure the log file size limit, in megabytes. A value equal to 0 (zero) means that there is no size limit for the log file

Ethernet Tab

Use this tab to configure parameters of an **Ethernet** Interface. These parameters, except port configurations, must also be configured for use in the **RAS** Interface.



Ethernet tab

Available options on the Ethernet tab

OPTION	DESCRIPTION
Transport	Select the value **TCP/IP** for a TCP socket (*stream*) or select the value **UDP/IP** to use a UDP socket (*connectionless datagram*)
Listen for connections on port	Use this option to wait for new connections in a specific IP port, common in Slave Drivers. If this option remains unselected, a Driver connects to the address and port specified in the **Connect to** option
Share listen port with other processes	Select this option to share the listening port with other Drivers and processes
Interface	Select the local network interface, identified by its IP address, that a Driver uses to establish and receive connections, or select the value **(All Interfaces)** to allow connection in any network interface
Use IPv6	Select this option to force a Driver to use addresses in **IPv6** format on all Ethernet connections. Leave this option deselected to use the **IPv4** format
Enable 'ECHO' suppression	Enable this option to remove the echo from received data. An echo is a copy of sent data, which can be returned before a reply message
IP Filter	List of restricted or allowed IP addresses from where a Driver accepts connections (*Firewall*). Please check the **IO.Ethernet.IPFilter** property for more information
PING before connecting	Enable this option to execute a **ping** command, that is, to check whether a device can be reached on a network, for a device before trying a socket connection. This is a quick way

OPTION	DESCRIPTION
	of determining a successful connection before trying to open a socket with a device. The time-out of a connection with a socket can be very high. The available options are: • **Timeout**: Specify the number of milliseconds to wait for a reply from a **ping** command. Users must use a **ping** command to check the normal reply time, configuring this option for a value above that average. Usually this value can be configured between 1000 and 4000 milliseconds, that is, between 1 (one) and 4 (four) seconds • **Retries**: Number of retries of a **ping** command, not counting the first attempt. If all attempts fail, then the socket connection is aborted

Available options on the Connect to group

OPTION	DESCRIPTION
Main IP	Type the IP address of a remote device. Users can use an IP address separated by dots, as well as a URL. In case of a URL, a Driver uses the available DNS service to map that URL to an IP address, such as "192.168.0.13" or "Server1"
Port	Type the IP port of a remote device, between 0 (zero) and 65535
Local port	Select this option to use a fixed local IP port when connecting to a remote device
Backup IP 1, 2, and 3	Indicate the IP address, the IP port, and the fixed local IP port of up to 3 (three) backup addresses of a remote device

General Configurations

This section contains information about the configuration of general **I/O Tags** and **Properties** of I/O Interfaces.

I/O Tags

General I/O Interfaces Tags (N2/B2 = 0)

The Tags described next are provided for all supported I/O Interfaces.

IO.CommunicationStatus

Type of Tag	I/O Tag
Type of Access	Reading
N1 Parameter	-1 (minus one)
N2 Parameter	0 (zero)
N3 Parameter	0 (zero)
N4 Parameter	6 (six)
String Configuration	IO.CommunicationStatus

This Tag informs the communication status of a Driver. It indicates how communication works relative to receiving valid data within a time period arbitrated in the configuration. For more information, please check topic **Setup Tab**. Possible values are **0 - Inactive communication**: The Driver did not receive valid data or stopped receiving data after *n* milliseconds, as configured in the properties window, or **1 - Active communication:** The Driver is receiving valid data.

IO.IOKitEvent

Type of Tag	Block Tag
Type of Access	Read-Only
B1 Parameter	-1 (minus one)
B2 Parameter	0 (zero)
B3 Parameter	0 (zero)
B4 Parameter	1 (one)
Size Property	4 (four)
ParamItem Property	IO.IOKitEvent

This Block returns Driver events generated by several sources in I/O Interfaces. The **TimeStamp** property of this Block represents the moment this event occurred. The Block Elements are the following:

- **Element 0**: Type of event. Possible values are **0**: Information, **1**: Warning, or **2**: Error
- **Element 1**: Source of an event. Possible values are **0**: Driver (specific of a Driver), **-1**: IOKit (generic events of I/O Interfaces), **-2**: **Serial** Interface, **-3**: **Modem** Interface, **-4**: **Ethernet** Interface, or **-5**: **RAS** Interface
- **Element 2**: Error number, specific for each source of event
- **Element 3**: Message of an event, a **String** specific for each event

NOTE

A Driver keeps a maximum number of 100 events internally. If additional events are reported, older events are discarded.

IO.PhysicalLayerStatus

Type of Tag	I/O Tag
Type of Access	Read-Only
N1 Parameter	-1 (minus one)
N2 Parameter	0 (zero)
N3 Parameter	0 (zero)
N4 Parameter	2 (two)
String Configuration	IO.PhysicalLayerStatus

This Tag indicates the status of a physical layer. Possible values are the following:

- **0**: Physical layer stopped, that is, a Driver is in **Offline** mode, the physical layer failed when initializing, or exceeded the maximum number of reconnection attempts
- **1**: Physical layer started but not connected, that is, a Driver is in **Online** mode but the physical layer is not connected. If the **Connection management** option is configured with the value **Automatic**, the physical layer can be connecting, disconnecting, or waiting for a reconnection attempt. If the **Connection management** option is configured with the vlaue **Manual**, then the physical layer remains in this status until forced to connect
- **2**: Physical layer connected, that is, the physical layer is ready for use. This **DOES NOT** mean a device is connected, only that the access layer is working

IO.SetConfigurationParameters

Type of Tag	Block Tag
Type of Access	Read-Only
B1 Parameter	-1 (minus one)
B2 Parameter	0 (zero)
B3 Parameter	0 (zero)
B4 Parameter	3 (three)
Size Property	2 (two)
ParamItem Property	IO.SetConfigurationParameters

Use this Tag to change any property of a Driver's configuration dialog box at run time.

This Tag works only while a Driver is in **Offline** mode. To start a Driver in **Offline** mode, select the **Start driver OFFLINE** option on that Driver's configuration dialog box. Users can write to a PLC Tag or to a Block Tag containing the parameters to change. Writing individual Block Elements is not supported, the whole Block must be written at once.

In **Elipse SCADA**, users must use a Block Tag. Every parameter to configure uses two Block Elements. For example, if users want to configure 3 (three) parameters, then the size of the Block must be 6 (six, 3 × 2). The first Element is the property's name, as a **String**, and the second Element is the property's value, according to the next example.

```
// 'Block' must be a Block Tag with automatic reading,
// scan reading, and automatic writings disabled.
// Configure all parameters
Block.element001 = "IO.Type" // Parameter 1
Block.element002 = "Serial"
Block.element003 = "IO.Serial.Port" // Parameter 2
Block.element004 = 1
Block.element005 = "IO.Serial.BaudRate" // Parameter 3
Block.element006 = 19200
// Writes the whole Block
Block.Write()
```

When using **Elipse E3**, the ability to create arrays at run time allows using an I/O Tag as well as a Block Tag. Users can use the **Write** method of a Driver to send the parameters directly to that Driver, without creating a Tag, according to the next example.

```
Dim arr(6)
' Configure all array elements
arr(1) = "IO.Type"
arr(2) = "Serial"
arr(3) = "IO.Serial.Port"
arr(4) = 1
arr(5) = "IO.Serial.BaudRate"
arr(6) = 19200
' There are two methods to send parameters
' Method 1: Using an I/O Tag
tag.WriteEx arr
' Method 2: Without using a Tag
Driver.Write -1, 0, 0, 3, arr
```

A variation of the previous example uses a bidimensional array.

```
Dim arr(10)
' Configure all array elements. Notice the array was resized
' to 10 elements. Empty array elements are ignored by a Driver
arr(1) = Array("IO.Type", "Serial")
arr(2) = Array("IO.Serial.Port", 1)
arr(3) = Array("IO.Serial.BaudRate", 19200)
Driver.Write -1, 0, 0, 3, arr
```

A Driver does not validate parameter names or passed values, therefore be careful when writing parameters and values. The **Write** method fails if the configuration array is incorrectly created. Users can check the log of a Driver or use the *writeStatus* parameter of the **WriteEx** method to find out the exact cause of an error.

```
Dim arr(10), strError
arr(1) = Array("IO.Type", "Serial")
arr(2) = Array("IO.Serial.Port", 1)
arr(3) = Array("IO.Serial.BaudRate", 19200)
If Not Driver.WriteEx -1, 0, 0, 3, arr, , , strError Then
  MsgBox "Failed configuring Driver parameters: " + strError
End If
```

IO.WorkOnline

Type of Tag	I/O Tag
Type of Access	Reading or Writing
N1 Parameter	-1 (minus one)
N2 Parameter	0 (zero)
N3 Parameter	0 (zero)
N4 Parameter	4 (four)
String Configuration	IO.WorkOnline

This Tag informs the current status of a Driver and allows starting or stopping the physical layer. Possible values are the following:

- **0 - Driver Offline**: Physical layer is closed or stopped. This mode allows a dynamic configuration of a Driver's parameters using the **IO.SetConfigurationParameters** Tag
- **1 - Driver Online**: Physical layer is open or executing. While in **Online** mode, the physical layer can be connected or disconnected and its current status can be checked using the **IO.PhysicalLayerStatus** Tag

In the next example, using **Elipse E3**, a Driver is configured to **Offline** mode, its COM port is changed, and then configured to **Online** mode again.

```
'Configure to Offline mode
Driver.Write -1, 0, 0, 4, 0
'Change port to COM2
Driver.Write -1, 0, 0, 3, Array("IO.Serial.Port", 2)
'Configure to Online mode
Driver.Write -1, 0, 0, 4, 1
```

The **Write** method may fail when configuring a Driver to **Online** mode, that is, writing the value 1 (one). In this case, this Driver remains in **Offline** mode. The cause of failure can be:

- Type of physical layer incorrectly configured, probably an invalid value was configured in the **IO.Type** property
- This Driver may have run out of memory
- Physical layer probably did not create its working thread. Search the log file for a message "Failed to create physical layer thread!"
- Physical layer could not start. The cause of this failure depends on the type of physical layer. It can be an invalid serial port number, a failure when starting Windows Sockets, or a failure when starting TAPI (modem), among others. This cause is recorded on the log file

IMPORTANT

Even if the configuration of a Driver to **Online** mode is successful, this does not necessarily mean the physical layer is ready to use, that is, ready to execute input and output operations with an external device. The **IO.PhysicalLayerStatus** Tag must be checked to ensure the physical layer is connected and ready for communication.

Properties

These are general properties of all supported I/O Interfaces.

IO.ConnectionMode

9 Controls the management mode of a Connection. Possible values are **0**: Automatic mode, in which a Driver manages the connection or **1**: Manual mode, in which an application manages the connection.

IO.GiveUpEnable

◩ When configured to True, defines a maximum number of reconnection attempts. If all reconnection attempts fail, a Driver enters the **Offline** mode. When configured to False, a Driver tries until a reconnection is successful.

IO.GiveUpTries

9 Number of reconnection attempts before this one is aborted. For example, if the value of this property is equal to 1 (one), a Driver tries only one reconnection when the connection is lost. If this one fails, this Driver enters the **Offline** mode.

IO.InactivityEnable

◩ Configure to True to enable and to False to disable inactivity detection. The physical layer is disconnected if inactive for a certain period of time. The physical layer is considered inactive only if it is capable of sending data but not capable of receiving it back.

IO.InactivityPeriodSec

9 Number of seconds to check for inactivity. If the physical layer is inactive for this period of time, it is then disconnected.

IO.RecoverEnable

Configure to True to enable a Driver to recover lost connections and to False to leave a Driver in **Offline** mode when a connection is lost.

IO.RecoverPeriodSec

Delay time between two connection attempts, in seconds.

NOTE

The first reconnection is executed immediately after a connection is lost.

IO.StartOffline

Configure to True to start a Driver in **Offline** mode and to False to start a Driver in **Online** mode.

NOTE

It is pointless to change this property at run time, as it can only be changed when a Driver is already in **Offline** mode. To configure a Driver in **Online** mode at run time, write the value 1 (one) to the **IO.WorkOnline** Tag.

IO.TimeoutMs

Defines a time-out for the physical layer, in milliseconds. One second is equal to 1000 milliseconds.

IO.Type

Defines the type of physical interface used by a Driver. Possible values are the following:

- **N or None**: Does not use a physical interface, that is, a Driver must provide a customized interface
- **S or Serial**: Uses a local serial port (COM*n*)
- **M or Modem**: Uses a local modem, internal or external, accessed via TAPI (*Telephony Application Programming Interface*)
- **E or Ethernet**: Uses a TCP/IP or UDP/IP socket
- **R or RAS**: Uses a **RAS** (*Remote Access Server*) Interface. A Driver connects to a RAS device using the **Ethernet** Interface and then sends an **AT** (*dial*) command

Statistical Configuration

This section contains information about the configuration of **I/O Tags** and **Properties** of I/O Interfaces statistics.

I/O Tags

Tags of I/O Interface Statistics (N2/B2 = 0)

The Tags described next display statistics for all I/O Interfaces.

IO.Stats.Partial.BytesRecv

Type of Tag	I/O Tag
Type of Access	Read-Only
N1 Parameter	-1 (minus one)
N2 Parameter	0 (zero)
N3 Parameter	0 (zero)
N4 Parameter	1101
Configuration by String	IO.Stats.Partial.BytesRecv

This Tag returns the number of bytes received in the current connection.

IO.Stats.Partial.BytesSent

Type of Tag	I/O Tag
Type of Access	Read-Only
N1 Parameter	-1 (minus one)
N2 Parameter	0 (zero)
N3 Parameter	0 (zero)
N4 Parameter	1100
Configuration by String	IO.Stats.Partial.BytesSent

This Tag returns the number of bytes sent through the current connection.

IO.Stats.Partial.TimeConnectedSeconds

Type of Tag	I/O Tag
Type of Access	Read-Only
N1 Parameter	-1 (minus one)
N2 Parameter	0 (zero)
N3 Parameter	0 (zero)
N4 Parameter	1102
Configuration by String	IO.Stats.Partial.TimeConnectedSeconds

This Tag returns the number of seconds a Driver is connected in the current connection or 0 (zero) if a Driver is disconnected.

IO.Stats.Partial.TimeDisconnectedSeconds

Type of Tag	I/O Tag
Type of Access	Read-Only
N1 Parameter	-1 (minus one)
N2 Parameter	0 (zero)
N3 Parameter	0 (zero)
N4 Parameter	1103
Configuration by String	IO.Stats.Partial.TimeDisconnectedSeconds

This Tag returns the number of seconds a Driver is disconnected since the last connection ended or 0 (zero) if a Driver is connected.

IO.Stats.Total.BytesRecv

Type of Tag	I/O Tag
Type of Access	Read-Only
N1 Parameter	-1 (minus one)
N2 Parameter	0 (zero)
N3 Parameter	0 (zero)
N4 Parameter	1001
Configuration by String	IO.Stats.Total.BytesRecv

This Tag returns the number of bytes received since a Driver was loaded.

IO.Stats.Total.BytesSent

Type of Tag	I/O Tag
Type of Access	Read-Only
N1 Parameter	-1 (minus one)
N2 Parameter	0 (zero)
N3 Parameter	0 (zero)
N4 Parameter	1000
Configuration by String	IO.Stats.Total.BytesSent

This Tag returns the number of bytes sent since a Driver was loaded.

IO.Stats.Total.ConnectionCount

Type of Tag	I/O Tag
Type of Access	Read-Only
N1 Parameter	-1 (minus one)
N2 Parameter	0 (zero)
N3 Parameter	0 (zero)
N4 Parameter	1004
Configuration by String	IO.Stats.Total.ConnectionCount

This Tag returns the number of connections a Driver already established, successfully, since it was loaded.

IO.Stats.Total.TimeConnectedSeconds

Type of Tag	I/O Tag
Type of Access	Read-Only
N1 Parameter	-1 (minus one)
N2 Parameter	0 (zero)
N3 Parameter	0 (zero)
N4 Parameter	1002
Configuration by String	IO.Stats.Total.TimeConnectedSeconds

This Tag returns the number of seconds a Driver remained connected since it was loaded.

IO.Stats.Total.TimeDisconnectedSeconds

Type of Tag	I/O Tag
Type of Access	Read-Only
N1 Parameter	-1 (minus one)
N2 Parameter	0 (zero)
N3 Parameter	0 (zero)
N4 Parameter	1003
Configuration by String	IO.Stats.Total.TimeDisconnectedSeconds

This Tag returns the number of seconds a Driver remained disconnected since it was loaded.

Properties

Currently, there are no properties defined specifically to display I/O Interface statistics at run time.

Ethernet Interface Configuration

This section contains information about the configuration of **I/O Tags** and **Properties** of an **Ethernet** Interface.

I/O Tags

Tags of an Ethernet Interface (N2/B2 = 4)

The Tags described next allow controlling and identifying an **Ethernet** Interface at run time and they are also valid when the **RAS** Interface is selected.

IMPORTANT
These Tags are available **ONLY** while a Driver is in **Online** mode.

IO.Ethernet.IPSelect

Type of Tag	I/O Tag
Type of Access	Reading or Writing
N1 Parameter	-1 (minus one)
N2 Parameter	0 (zero)
N3 Parameter	4 (four)
N4 Parameter	0 (zero)
String Configuration	IO.Ethernet.IPSelect

Indicates the active IP address. Possible values are **0**: The main IP address is selected, **1**: The first alternative or backup IP address is selected, **2**: The second alternative or backup IP address is selected, or **3**: The third alternative or backup IP address is selected.

If the **Ethernet** or **RAS** Interface is connected, this Tag indicates which one of the four configured IP addresses is in use. If the Interface is disconnected, this Tag indicates which IP address is used first on the next attempt to connect.

During the connection process, if the active IP address is not available, the I/O Interface tries to connect using the other IP address. If the connection with the alternative IP address works, it is configured as the active IP address (automatic switchover).

To force a manual switchover, write values from 0 (zero) to 3 (three) to this Tag. This forces a reconnection with the specified IP address (**0**: Main address or **1, 2, 3**: Alternative address) if a Driver is currently connected. If a Driver is disconnected, this Tag configures the active IP address for the next attempt to connect.

IO.Ethernet.IPSwitch

Type of Tag	I/O Tag
Type of Access	Write-Only
N1 Parameter	-1 (minus one)
N2 Parameter	0 (zero)
N3 Parameter	4 (four)
N4 Parameter	1 (one)
String Configuration	IO.Ethernet.IPSwitch

Any value written to this Tag forces a manual switchover. If the main IP address is active, then the first alternative or backup IP address is activated, and so on for all alternative IP addresses and returning to the main address until a connection is established.

If a Driver is disconnected, this Tag configures the active IP address for the next attempt to connect.

IO.Ethernet.SocketState

Type of Tag	I/O Tag
Type of Access	Read-Only
N1 Parameter	-1 (minus one)
N2 Parameter	0 (zero)
N3 Parameter	4 (four)
N4 Parameter	2 (two)
String Configuration	IO.Ethernet.SocketState

The Value property of this Tag corresponds to socket states as a map of bits:

- **Bit 0**: 0 (zero, not listening) or 1 (one, listening)
- **Bit 1**: 0 (zero, disconnected) or 1 (one, connected)

Properties

These properties control the configuration of an **Ethernet** Interface.

NOTE

The **Ethernet** Interface is also used by the **RAS** Interface.

IO.Ethernet.AcceptConnection

Configure to False if a Driver must not accept external connections, that is, if a Driver behaves as a master, or configure to True to enable the reception of connections, that is, if a Driver behaves as a slave.

IO.Ethernet.BackupEnable[2,3]

Configure to True to enable an alternative or backup IP address. If the reconnection attempt with the main IP address fails, a Driver tries to use an alternative IP address. Configure to False to disable its usage.

IO.Ethernet.BackupIP[2,3]

Alternative or backup IP address of a remote device. Users can use a numerical address, as well as a device's host name, such as "192.168.0.7" or "SERVER2".

IO.Ethernet.BackupLocalPort[2,3]

Local port number to be used when connecting to an alternative IP address of a remote device. Used only if **IO.Ethernet.BackupLocalPortEnable** is equal to True.

IO.Ethernet.BackupLocalPortEnable[2,3]

Configure to True to force the use of a specific local port when connecting to an alternative or backup IP address or configure to False to use any available local port.

IO.Ethernet.BackupPort[2,3]

Port number of an alternative or backup IP address of a remote device, used with the **IO.Ethernet.BackupIP** property.

IO.Ethernet.IPFilter

List with a comma-separated IPv4 or IPv6 addresses, which defines from which addresses a Driver accepts or blocks connections. Users can use asterisks, such as "192.168.*.*", or intervals, such as "192.168.0.41-50", in any part of IP addresses. To block an IP address or a range of IP addresses, use the tilde ("~") character at the beginning of the address, according to the next examples:

- **192.168.0.24**: Accepts only connections from IPv4 address 192.168.0.24
- **192.168.0.41-50**: Accepts connections from IPv4 addresses in the interval between 192.168.0.41 and 192.168.0.50
- **192.168.0.***: Accepts connections from IPv4 addresses in the interval between 192.168.0.0 and 192.168.0.255
- **fe80:3bf:877::*:* (expands to fe80:03bf:0877:0000:0000:0000:*:*)**: Accepts connections from IPv6 addresses in the interval between fe80:03bf:0877:0000:0000:0000:0000:0000 and fe80:03bf:0877:0000:0000:0000:ffff:ffff
- **192.168.0.10, 192.168.0.15, 192.168.0.20**: Accepts connections from IPv4 addresses 192.168.0.10, 192.168.0.15, and 192.168.0.20
- **~192.168.0.95, 192.168.0.***: Accepts connections from IPv4 addresses in the interval between 192.168.0.0 and 192.168.0.255, except the IPv4 address 192.168.0.95

When a Driver receives a connection attempt, the list of filters is scanned sequentially from left to right, searching for a specific authorization or block for the IP address where the connection comes from. If no element on the list corresponds to the IP address, the authorization or block are dictated by the last element of that list:

- If the last element on the list is an authorization, such as "192.168.0.24", then all IP addresses not found on the list are blocked
- If the last element on the list is a block, such as "~192.168.0.24", then all IP addresses not found on the list are authorized

If an IP address appears on more than one filter on the list, the leftmost filter has precedence. For example, in case of "~192.168.0.95, 192.168.0.*", the IP address 192.168.0.95 fits both rules, but the rule that wins is the leftmost one, "~192.168.0.95", and therefore this IP address is blocked.

When **IOKit** blocks a connection, it logs a message "Blocked incoming socket connection from {IP}!".

In case of UDP connections in broadcast listening mode, in which a Driver can receive packets from different IP addresses, blocks or permissions are performed at each packet received. If a packet is received from a blocked IP address, it logs a message "Blocked incoming packet from {IP} (discarding {N} bytes)!".

IO.Ethernet.ListenIP

A IP address of the local network interface that a Driver uses to establish and accept connections. Leave this property empty to establish and accepts connections using any local network interface.

IO.Ethernet.ListenPort

9 Number of the IP port used by a Driver to listen to connections.

IO.Ethernet.MainIP

A IP address of a remote device. Users can use a numerical address, as well as a device's host name, such as "192.168.0.7" or "SERVER2".

IO.Ethernet.MainLocalPort

9 Local port number to use when connecting to the main IP address of a remote device. This value is only used if the **IO.Ethernet.MainLocalPortEnable** property is equal to True.

IO.Ethernet.MainLocalPortEnable

◩ Configure to True to force the use of a specific local port when connecting to the main IP address of a remote device or configure to False to use any available local port.

IO.Ethernet.MainPort

9 Number of the IP port of a remote device, used with the **IO.Ethernet.MainIP** property.

IO.Ethernet.PingEnable

◩ Configure to True to enable sending a **ping** command to the IP address of a remote device, before trying to connect to the socket. This socket's connection time-out cannot be controlled, therefore sending a **ping** command before connecting is a fast way to detect if the connection is going to fail. Configure to False to disable a **ping** command.

IO.Ethernet.PingTimeoutMs

9 Delay time to wait for a response from a **ping** command, in milliseconds.

IO.Ethernet.PingTries

9 Maximum number of attempts of a **ping** command. Minimum value is 1 (one), including the first **ping** command.

IO.Ethernet.ShareListenPort

Configure to True to share a listening port with other Drivers and processes or False to open a listening port in exclusive mode. To successfully share a listening port, all Drivers and processes that use that port must open it in shared mode. When a listening port is shared, each incoming connection is distributed to one of the processes listening. This way, if a Slave Driver only supports one connection at a time, users can use several instances of this Driver listening on the same port, therefore simulating a Driver with support for multiple connections.

IO.Ethernet.SupressEcho

Configure to True to eliminate echoes in communication. An echo is the unwanted reception of an exact copy of all data packets a Driver sent to a device.

IO.Ethernet.Transport

Defines a transport protocol. Possible values are **T or TCP**: Uses the TCP/IP protocol or **U or UDP**: Uses the UDP/IP protocol.

IO.Ethernet.UseIPv6

Configure to True to use IPv6 addresses on all Ethernet connections or configure to False to use IPv4 addresses (default).

Revision History

VERSION	DATE	AUTHOR	COMMENTS
3.0.55	10/06/2025	M. Salvador	• Improvements in XMPP Layer Handling and Log Messages [30839] {implemented in IEC61850 v3.00 Build 23} • Added support for PRP networks through a Windows service and configuration via the driver dialog. [27270] {implemented in IEC61850 v3.00 Build 26} • It is now possible to specify a "none" board for one of the LANs in the PRP configuration. [31538] {implemented in IEC61850 v3.00 Build 26} • Added version information to the PRP service. [31995] {implemented in IEC61850 v3.00 Build 30} • Parameters such as board names, IP address/statistics port are now read from the ElipsePRPSvc.ini file instead of the PRJ; a new property was created with a list of

VERSION	DATE	AUTHOR	COMMENTS
			computer names and IP addresses so the driver can identify the computer and use the corresponding IP interface, allowing the same PRJ to be used in redundant servers without changes—i.e., differences are stored in the ElipsePRPSvc.ini file. [32144] {implemented in IEC61850 v3.00 Build 31} • Improvements in PRP Configuration and Implementation. [34350] {implemented in IEC61850 v3.00 Build 43} • Mapped the TimeQuality property to Tag Quality as follows: Invalid Timestamp: 216 (Non-specific local timestamp, substatus = 6, previously indicated 192); Clock Failure: 220 (Local Override Local Timestamp, substatus = 7); Clock not Synchronized: 224 (custom), substatus = 8. [35905] {implemented in IEC61850 v3.00 Build 45} • It is now possible to download any files from the IED (not only COMTRADE files). [35507] {implemented in IEC61850 v3.00 Build 46} • Added option to identify the standard revision (2.1) and revised the report enabling procedure. [37072] {implemented in IEC61850 v3.00 Build 48} • [BUG] Adapted the process of disabling a report for subsequent dataset deletion to comply with revision 2.1 of the standard. [30486] {fixed in IEC61850 v3.00 Build 18} • [BUG] When enabling a report, the RptID is now compared with the current value to determine if writing is necessary, aiming to

VERSION	DATE	AUTHOR	COMMENTS
			prevent the IED from interpreting it as a configuration change sufficient to clear the event buffer (PurgeBuf). [30721] {fixed in IEC61850 v3.00 Build 19} • [BUG] Fixed the possibility of the COTP connection not respecting the timeout. [30798] {fixed in IEC61850 v3.00 Build 20} • [BUG] Reports are now re-read before the driver attempts to re-enable them after communication is restored, avoiding potential error situations where the value in the driver's memory differs from the IED's (which may have changed during the communication outage). [30820] {fixed in IEC61850 v3.00 Build 20} • [BUG] The driver now correctly re-enables reports that were "locked" as if enabled on the IED for some time. [30858] {fixed in IEC61850 v3.00 Build 22} • [BUG] COMTRADE files containing special characters \ or / are now saved correctly. [30859] {fixed in IEC61850 v3.00 Build 22} • [BUG] If a Dataset is incorrectly configured on the IED without the 't' parameter, the driver now checks if the timestamp was actually read before reporting the value. [31000] {fixed in IEC61850 v3.00 Build 24} • [BUG] Revised the TCP/IP and MMS reconnection procedure after disconnections. [31133] {fixed in IEC61850 v3.00 Build 25} • [BUG] If a polling message is not responded to (timeout),

VERSION	DATE	AUTHOR	COMMENTS
			tags in that message now indicate bad quality. [31537] {fixed in IEC61850 v3.00 Build 26} • [BUG] Fixed the PRP Service. [31717] {fixed in IEC61850 v3.00 Build 28} • [BUG] ComtradeInfo and ComtradeTransfer tags were not correctly showing the status when the IED has no files. [31877] {fixed in IEC61850 v3.00 Build 29} • [BUG] Automatic report enabling could yield inconsistent results due to items with "partial" names in already enabled datasets/reports. [32417] {fixed in IEC61850 v3.00 Build 34} • [BUG] Modified the logging procedure to prevent failures in VMware environments and fixed packet reception filters on physical boards. [32345] {fixed in IEC61850 v3.00 Build 35} • [BUG] The driver now uses the NPCAP SDK instead of WINPCAP. [33489] {fixed in IEC61850 v3.00 Build 38} • [BUG] Adapted the driver to download COMTRADE files if the IED has many files in the root directory, and added support for transfers based on name + index instead of just date comparison. [33741] {fixed in IEC61850 v3.00 Build 38} • [BUG] Fixed the reconnection procedure after communication loss. [33887] {fixed in IEC61850 v3.00 Build 41} • [BUG] Added support for arrays within datasets.

VERSION	DATE	AUTHOR	COMMENTS
			[34243] {fixed in IEC61850 v3.00 Build 42} • [BUG] Fixed behavior when there is no response to a GetFileDirectory after an initial response with MoreFollows=TRUE. [34463] {fixed in IEC61850 v3.00 Build 42} • [BUG] Improved robustness in handling and receiving commands to prevent failures when receiving malformed messages from servers. [35638] {fixed in IEC61850 v3.00 Build 44} • [BUG] After enabling a report, a retry was added to perform GI if the first attempt fails. Additionally, if the user specified writing the DatSet property and it fails, a bug was fixed that caused the driver to write an empty value twice; it should write empty once and then write the correct Dataset value. [36008] {fixed in IEC61850 v3.00 Build 45} • [BUG] Fixed a driver loop that failed to reconnect to an IED that disconnected after an initialization failure caused by sending an untimely Release Request. [36439] {fixed in IEC61850 v3.00 Build 46} • [BUG] Fixed a possible memory leak and cases of unanswered MMS transactions. [36492] {fixed in IEC61850 v3.00 Build 46} • [BUG] Modified the PRP service to use block operations. [36720] {fixed in IEC61850 v3.00 Build 47} • [BUG] Fixed an "unsupported operation" error when opening the configuration dialog of a driver saved with

VERSION	DATE	AUTHOR	COMMENTS
			previous versions. [37224] {occurred in IEC61850 v3.00 Build 47, fixed in IEC61850 v3.00 Build 50} • Created DeviceConnectionStatus, WorkOnline, and DeleteAllDevices tags to monitor and control communication with IEDs. [37238] {} • Created the GIAllReports tag to force a GI of all reports enabled by the driver. [37631] {} • Created the ReportItemsUnified block to receive all report events, per IED. [37633] {} • [BUG] Removed references to Elipse E3 in the driver documentation. [36973] {} • [BUG] Fixed behavior of not correctly handling large frames (approximately larger than 1500 bytes). [37512] {} • [BUG] When attempting to re-enable reports after recovering from a communication loss, the driver now only avoids enabling a report with RptEna=TRUE if ResvTms = 0. [37632] {} • [BUG] [37788] Revised XMPP operation and documentation. {}
3.0.17	21/12/2020	M. Salvador	• Added support for gateways. • Time files for the last COMTRADE.
3.0.9	11/12/2019	M. Salvador	• Support for PRP protocol.
3.0.6	01/09/2019	M. Ludwig	• Driver ported to Visual Studio 2017 (*Case 27095*).
3.0.1	04/06/2019	M. Salvador	• Implemented IEC 61850 ED2 Certification.

VERSION	DATE	AUTHOR	COMMENTS
2.0.23	08/02/2019	M. Salvador	• Implemented an option to wait for a writing to the **LastComtradeFileTime** Tag, configurable on the **Comtrade / Events** tab (*Case 25806*).
2.0.20	19/11/2018	M. Salvador	• Fixed a memory leak when the server does not implement or does not reply to a **GetVariableAccessAttributes** request (*Case 25546*).
2.0.19	17/09/2018	M. Salvador	• Now an IED that does not support using dynamic **DataSets** can be used together with another IEDs that support that operation on the same Driver (*Case 25072*).
2.0.16	16/11/2017	M. Salvador	• Creating a new Driver defines as default the use of a Tag for each command in the **Single Tag Cmd Alias** Tag (*Case 23693*). • Performed corrections when using dynamic **DataSets**, including handling IED reconnection and restart (*Case 23644*).
2.0.3	11/02/2014	M. Salvador	• Implemented KEMA Certification.
1.1.27	03/12/2012	M. Salvador	• Fixed the process of connecting and disconnecting. • Fixed the handling of **Reports** with **BitInclusion** incoherent with the size of **Dataset**. • Fixed the process of connecting and disconnecting, by creating a new thread to check the connection status. • Changed the handling of reading exceptions. • **Reject** message was being handled as **ConfirmedError** in some cases. • Fixed errors when handling lists.

VERSION	DATE	AUTHOR	COMMENTS
			• Handling invalid real numbers (QNAN). • Fixed the verification of connection status. • Fixed the information about the execution status of a command. • Additional options to search for COMTRADE files. • Fixed a possible deadlock between semaphores during a reconnection. • COMTRADE status block. • Support for **EntryID** during the initialization of **Reports**. • Dynamic creation of devices. • Propagating **IOKit** events. • Fixed the collecting of oscillography for Areva and Alstom relays. • Support for **Double** data types in **mms_float**. • **DNSNames** with port specification in IED's IP address. • Fixed the ICD Sel import process, additional protection to avoid writings and polling during a reconnection. • Support for user session requirements in Presentation CPA PPDU. • Fixed a memory leak (VLD). • Fixed a memory leak when disconnected (COTP) and test of SEL semaphore. • Log of timezone when starting this Driver. • Considers *TimeZone Unknown* as valid for local offset calculation. • Changed the re-enabling mode of **Reports** after a reconnection. • LDs cannot be created anymore, except when reading LD files or after executing a

VERSION	DATE	AUTHOR	COMMENTS
			GetServerDirectory command (*Case 11378*). • Removed zeroes to the left of IP addresses (*Case 11379*). • Removing zeroes to the left did not consider using the port after the IP address. • Reprogrammed **Reports** completely after a reconnection. • Created Tags **IPSelect** and **IPSwitch** and removed the **TimeofEntry** from polling, if enabled. • Implemented a thread to check the status of hosts. • Limit of 20 Tags for reading by scan and update of **RptEna** when there are Tags in an uncertain situation. • Problem of wrong random reception of packets on *Session* and *Presentation* layers. • Changed the *Nesting Level Requested* parameter from five to six (INGETEAM). • The **GetVariableAccessAtributes** command is performed one directory below in case of failure (INGETEAM). **EntryID** and **OptFlds** are no longer required in **OptFlds** when a **Report** type is **Unbuffered**. • Fixed a potential failure inside **MMS::SendRequest**, which may occur during a connection error. • Improved performance of status Tags. • A request for TCP/IP disconnection was not performed when an **MMS::Status** request failed or did not cover all situations. • Changed the way to perform a **ServerRebuild**

VERSION	DATE	AUTHOR	COMMENTS
			operation, which is now asynchronous. • Improved system's initialization performance, mainly when reading LD files.
1.0.1	31/03/2010	M. Salvador M. Bihre	• Fixed sending integer numbers to commands in the **ctlNum** property (*Case 11097*). • The *WriteStatus* parameter of writing operations using the **WriteEx** method of **Elipse Power** was not informed in some cases (*Case 11104*). • Added support for downloading oscillography files. • Added support for asynchronous commands. • Added support for individual Tags for commands. • Fixed support for ICD and SCD files. • Original version of this Driver.

Headquarters
Rua Mostardeiro, 322/Cj. 902, 1001 e 1002
90510-002 — Porto Alegre — RS
Phone: (+55 51) 3346-4699
Fax: (+55 51) 3222-6226
E-mail: elipse-rs@elipse.com.br

Branch in Taiwan
9F., No.12, Beiping 2nd St., Sanmin Dist.
807 — Kaohsiung City — Taiwan
Phone: (+886 7) 323-8468
Fax: (+886 7) 323-9656
E-mail: evan@elipse.com.br

Check our website for information about a representative in your country.

www.elipse.com.br

kb.elipse.com.br
forum.elipse.com.br
www.youtube.com/elipsesoftware
elipse@elipse.com.br

Gartner 2014 Cool Vendor

OPC FOUNDATION MEMBER

Gartner, Cool Vendors in Brazil 2014, April 2014.
Gartner does not endorse any vendor, product or service depicted in its research publications, and does not advise technology users to select only those vendors with the highest ratings. Gartner research publications consist of the opinions of Gartner's research organization and should not be construed as statements of fact. Gartner disclaims all warranties, expressed or implied, with respect to this research, including any warranties of merchantability of fitness for a particular purpose.

Microsoft Partner
Gold Independent Software Vendor (ISV)

Copyright © 2008-2025 Elipse Software Ltda. All rights reserved.