zenon
by COPA-DATA

zenon driver manual

BACnetNG

v.8.20

Contents

1 Welcome to COPA-DATA help

ZENON VIDEO TUTORIALS

You can find practical examples for project configuration with zenon in our YouTube channel (https://www.copadata.com/tutorial_menu). The tutorials are grouped according to topics and give an initial insight into working with different zenon modules. All tutorials are available in English.

GENERAL HELP

If you cannot find any information you require in this help chapter or can think of anything that you would like added, please send an email to documentation@copadata.com.

PROJECT SUPPORT

You can receive support for any real project you may have from our customer service team, which you can contact via email at support@copadata.com.

LICENSES AND MODULES

If you find that you need other modules or licenses, our staff will be happy to help you. Email sales@copadata.com.

2 BACnetNG

BACNET/IP CLIENT DRIVER

The BACnet protocol was defined by ASHRAE (American Society of Heating, Refrigeration and Air-Conditioning Engineers, Inc.) and is described extensively in the ASHRAE standard 135 (year) + Appendix A – L "A Data Communication Protocol for Building Automation and Control Networks". BACnet/IP is specified in appendix J and describes the BACnet communication via IP/UDP telegrams.

This driver allows communication between one or more devices that support BACnet (*BACnet-Devices*) and zenon Runtime via BACnet/IP. This requires that the connected BACnet devices run as servers. Only client functionality is implemented in the driver.

The BACnet protocol defines objects and object properties. The driver makes it possible to read and write one or several properties of an object. In principal both polling reading and spontaneous communication using COV (change-of-value) subscription is supported. Many devices support spontaneous communication only for the **PRESENT-VALUE** and **STATUS-FLAGS** properties:

3 BACnetNG - data sheet

General:	
Driver file name	BACnetNG.exe
Driver name	BACnet Driver Next Generation
PLC types	All controllers with BACnet/IP support
PLC manufacturer	ABB; Siemens; Kieback + Peter; BACnet; SE Elektronic; Trend

Driver supports:	
Protocol	BACnet/IP
Addressing: Address-based	Name based
Addressing: Name-based	--
Spontaneous communication	X
Polling communication	X
Online browsing	X
Offline browsing	--
Real-time capable	--
Blockwrite	--
Modem capable	--
RDA numerical	--

Driver supports:	
RDA String	--
Hysteresis	X
extended API	X
Supports status bit **WR-SUC**	X
alternative IP address	--

Requirements:	
Hardware PC	Standard network card
Software PC	--
Hardware PLC	--
Software PLC	--
Requires v-dll	X

Platforms:	
Operating systems	Windows 10; Windows 7; Windows 8; Windows 8.1; Windows Server 2008 R2; Windows Server 2012; Windows Server 2012 R2; Windows Server 2016

4 Requirements

This chapter contains information on the requirements that are necessary for use of this driver.

4.1 PC

For the BACnet/IP communication an IP network connection is needed which supports the UDP protocol.

If the driver is started several times, a separate network connection (network card) is necessary for each instance of the driver.

4.2 PLC

PLCs or BACnet Devices must also support the BACnet protocol BACnet/IP. If a PLC is not connected to the zenon Runtime via BACnet/IP, a corresponding BACnet router must be used.

MINIMUM REQUIREMENT FOR BACNET INTEROPERABILITY BUILDING BLOCKS :

Data sharing	Device & Network Management
DS-RP-B	DM-DDB-B
DS-RPM-B	DM-DOB-B

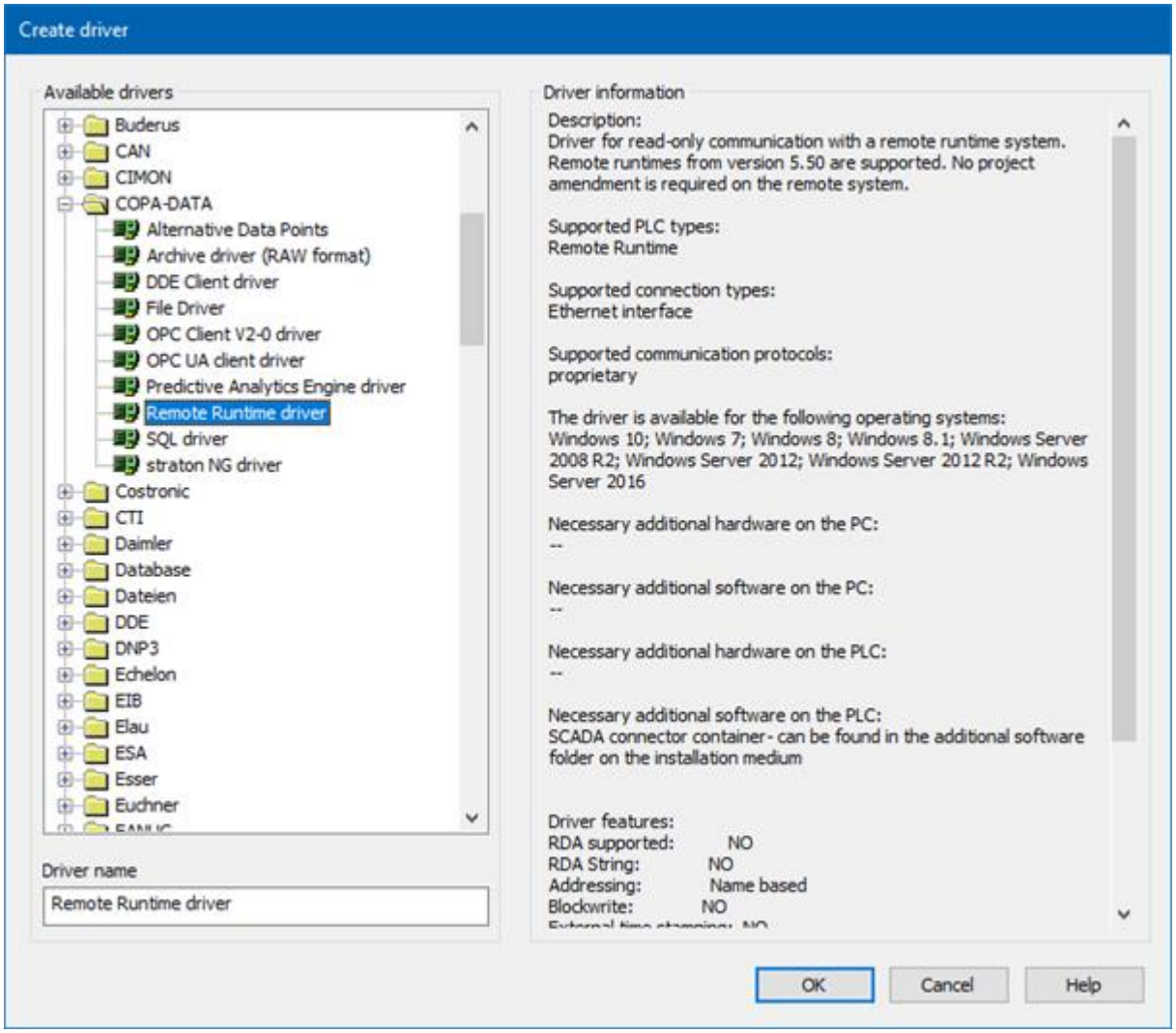
5 Configuration

In this chapter you will learn how to use the driver in a project and which settings you can change.

> **Information**
>
> Find out more about further settings for zenon variables in the chapter Variables of the online manual.

5.1 Creating a driver

In the **Create driver** dialog, you create a list of the new drivers that you want to create.



Parameter	Description
Available drivers	List of all available drivers. The display is in a tree structure: *[+]* expands the folder structure and shows the drivers contained therein. [-] reduces the folder structure Default: *No selection*
Driver name	Unique **Identification** of the driver. Default: *empty* The input field is pre-filled with the pre-defined

Parameter	Description
	Identification after selecting a driver from the list of available drivers.
Driver information	Further information on the selected driver. Default: *empty* The information on the selected driver is shown in this area after selecting a driver.

CLOSE DIALOG

Option	Description
OK	Accepts all settings and opens the driver configuration dialog of the selected driver.
Cancel	Discards all changes and closes the dialog.
Help	Opens online help.

> **Information**
>
> The content of this dialog is saved in the file called Treiber_[Language].xml. You can find this file in the following folder:
> *C:\ProgramData\COPA-DATA\zenon[version number]*.

CREATE NEW DRIVER

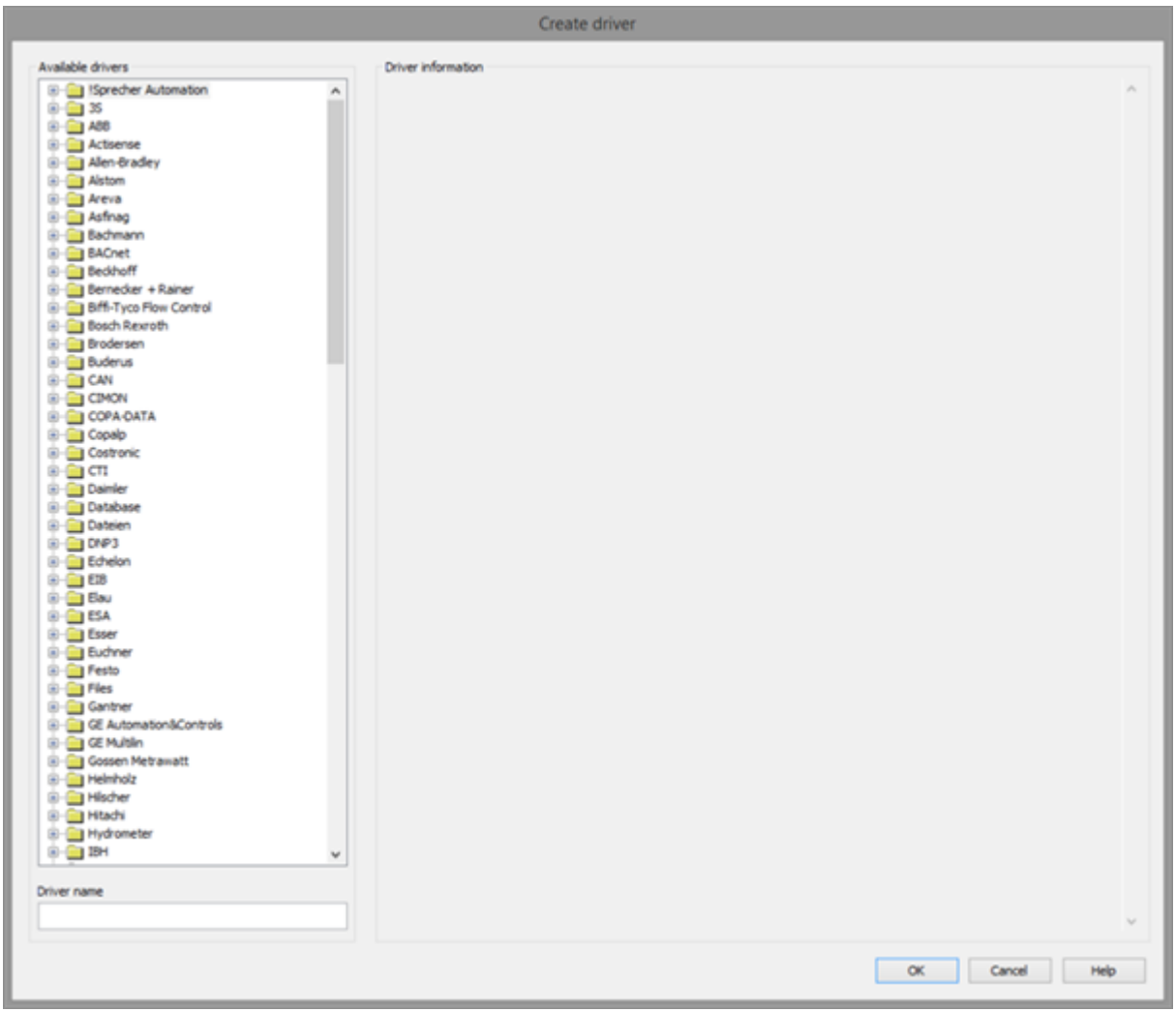
In order to create a new driver:

1. Right-click on **Driver** in the Project Manager and select **New driver** in the context menu.

   Optional: Select the **New driver** button from the toolbar of the detail view of the **Variables**.The Create driver dialog is opened.

   The **Create simple data type** dialog is opened.

2. The dialog offers a list of all available drivers.



3. Select the desired driver and name it in the **Driver name** input field.
   This input field corresponds to the **Identification** property. The name of the selected driver is automatically inserted into this input field by default.
   The following is applicable for the **Driver name**:

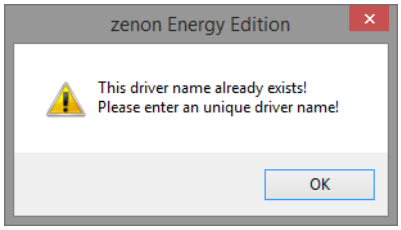
   - The **Driver name** must be unique.
     If a driver is used more than once in a project, a new name has to be given each time. This is evaluated by clicking on the **OK** button. If the driver is already present in the project, this is shown with a warning dialog.
   - The **Driver name** is part of the file name.
     Therefore it may only contain characters which are supported by the operating system. Invalid characters are replaced by an underscore (_).
   - **Attention:** This name cannot be changed later on.

4. Confirm the dialog by clicking on the **OK** button.
   The configuration dialog for the selected driver is opened.

Note: The language of driver names cannot be switched. They are always shown in the language in which they have been created, regardless of the language of the Editor. This also applies to driver object types.

DRIVER NAME DIALOG ALREADY EXISTS

If there is already a driver in the project, this is shown in a dialog. The warning dialog is closed by clicking on the **OK** button. The driver can be named correctly.



ZENON PROJECT

The following drivers are created automatically for newly-created projects:

- **Intern**
- **MathDr32**
- **SysDrv**

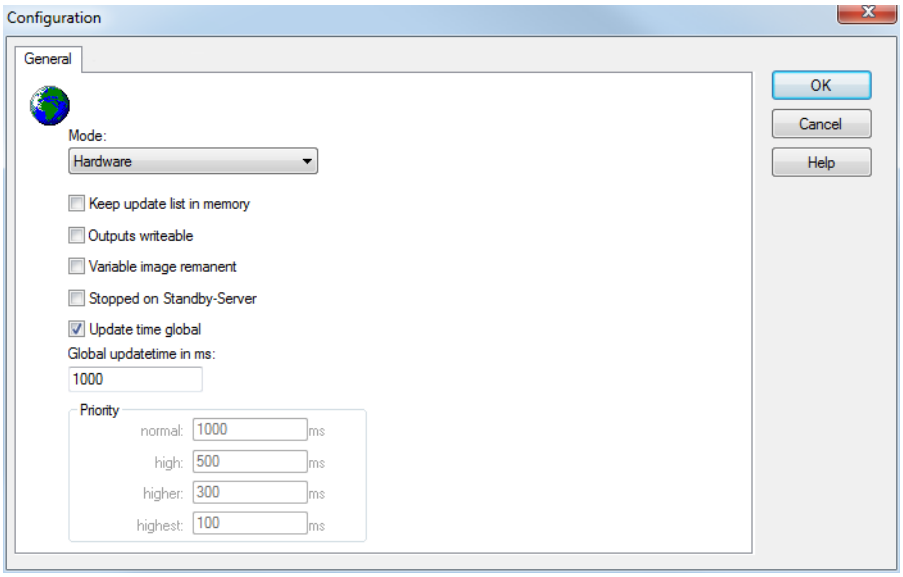
> **Information**
>
> Only the required drivers need to be present in a zenon project. Drivers can be added at a later time if required.

5.2 Settings in the driver dialog

You can change the following settings of the driver:

5.2.1 General

The configuration dialog is opened when a driver is created. In order to be able to open the dialog later for editing, double click on the driver in the list or click on the **Configuration** property.



Option	Description
Mode	Allows to switch between hardware mode and simulation mode ▸ *Hardware*: A connection to the control is established. ▸ Simulation - static: No communication between to the control is established, the values are simulated by the driver. In this modus the values remain constant or the variables keep the values which were set by zenon Logic. Each variable has its own memory area. E.g. two variables of the type marker with offset 79 can have different values in the Runtime and do not influence each other. Exception: The simulator driver. ▸ *Simulation - counting*: No communication between to the control is established, the values are simulated by the driver. In this modus the driver increments the values within a value range automatically. ▸ *Simulation - programmed*: No communication is established to the PLC. The

Option	Description
	values are calculated by a freely programmable simulation project. The simulation project is created with the help of the zenon Logic Workbench and runs in a zenon Logic Runtime which is integrated in the driver. For details see chapter Driver simulation.
Keep update list in the memory	Variables which were requested once are still requested from the control even if they are currently not needed. This has the advantage that e.g. multiple screen switches after the screen was opened for the first time are executed faster because the variables need not be requested again. The disadvantage is a higher load for the communication to the control.
Output can be written	▸ *Active*: Outputs can be written. ▸ *Inactive*: Writing of outputs is prevented. **Note**: Not available for every driver.
Variable image remanent	This option saves and restores the current value, time stamp and the states of a data point. Fundamental requirement: The variable must have a valid value and time stamp. The variable image is saved in hardware mode if one of these statuses is active: ▸ User status *M1* (*0*) to *M8* (*7*) ▸ *REVISION(9)* ▸ *AUS(20)* ▸ *ERSATZWERT(27)* The variable image is always saved if: ▸ the variable is of the **Communication details** object type ▸ the driver runs in simulation mode. (not programmed simulation) The following states are not restored at the start of the Runtime:

Option	Description
	▸ *SELECT(8)* ▸ *WR-ACK(40)* ▸ *WR-SUC(41)* The mode **Simulation - programmed** at the driver start is not a criterion in order to restore the remanent variable image.
Stop on Standby Server	Setting for redundancy at drivers which allow only one communication connection. For this the driver is stopped at the Standby Server and only started at the upgrade. **Attention:** If this option is active, the gapless archiving is no longer guaranteed. ▸ *Active*: Sets the driver at the not-process-leading Server automatically in a stop-like state. In contrast to stopping via driver command, the variable does not receive status **switched off** but an empty value. This prevents that at the upgrade to the Server irrelevant values are created in the AML, CEL and Historian. Default: *inactive* **Note:** Not available if the CE terminal serves as a data server. You can find further information in the zenon Operator manual in the CE terminal as a data server chapter.
Global Update time	Setting for the global update times in milliseconds: ▸ *Active*: The set **Global update time** is used for all variables in the project. The priority set at the variables is not used. ▸ *Inactive*: The set priorities are used for the individual variables. **Exceptions:** Spontaneous drivers ignore this option. They generally use the shortest possible update time. For details, see the **Spontaneous driver update time** section.

Option	Description
Priority	The polling times for the individual priority classes are set here. All variables with the according priority are polled in the set time. The variables are allocated separately in the settings of the variable properties. The communication of the individual variables can be graded according to importance or required topicality using the priority classes. Thus the communication load is distributed better. **Attention:** Priority classes are not supported by each driver, e.g. spontaneously communicating zenon drivers.

CLOSE DIALOG

Option	Description
OK	Applies all changes in all tabs and closes the dialog.
Cancel	Discards all changes in all tabs and closes the dialog.
Help	Opens online help.

UPDATE TIME FOR SPONTANEOUS DRIVERS

With spontaneous drivers, for **Set value**, **advising** of variables and **Requests**, a read cycle is triggered immediately - regardless of the set update time. This ensures that the value is immediately available for visualization after writing. The update time is generally 100 ms.

Spontaneous drivers are **ArchDrv**, **BiffiDCM**, **BrTcp32**, **DNP3**, **Esser32**, **FipDrv32**, **FpcDrv32**, **IEC850**, **IEC870**, **IEC870_103**, **Otis**, **RTK9000**, **S7DCOS**, **SAIA_Slave**, **STRATON32** and **Trend32**.

5.2.1.1 Handling update times for BACnetNG

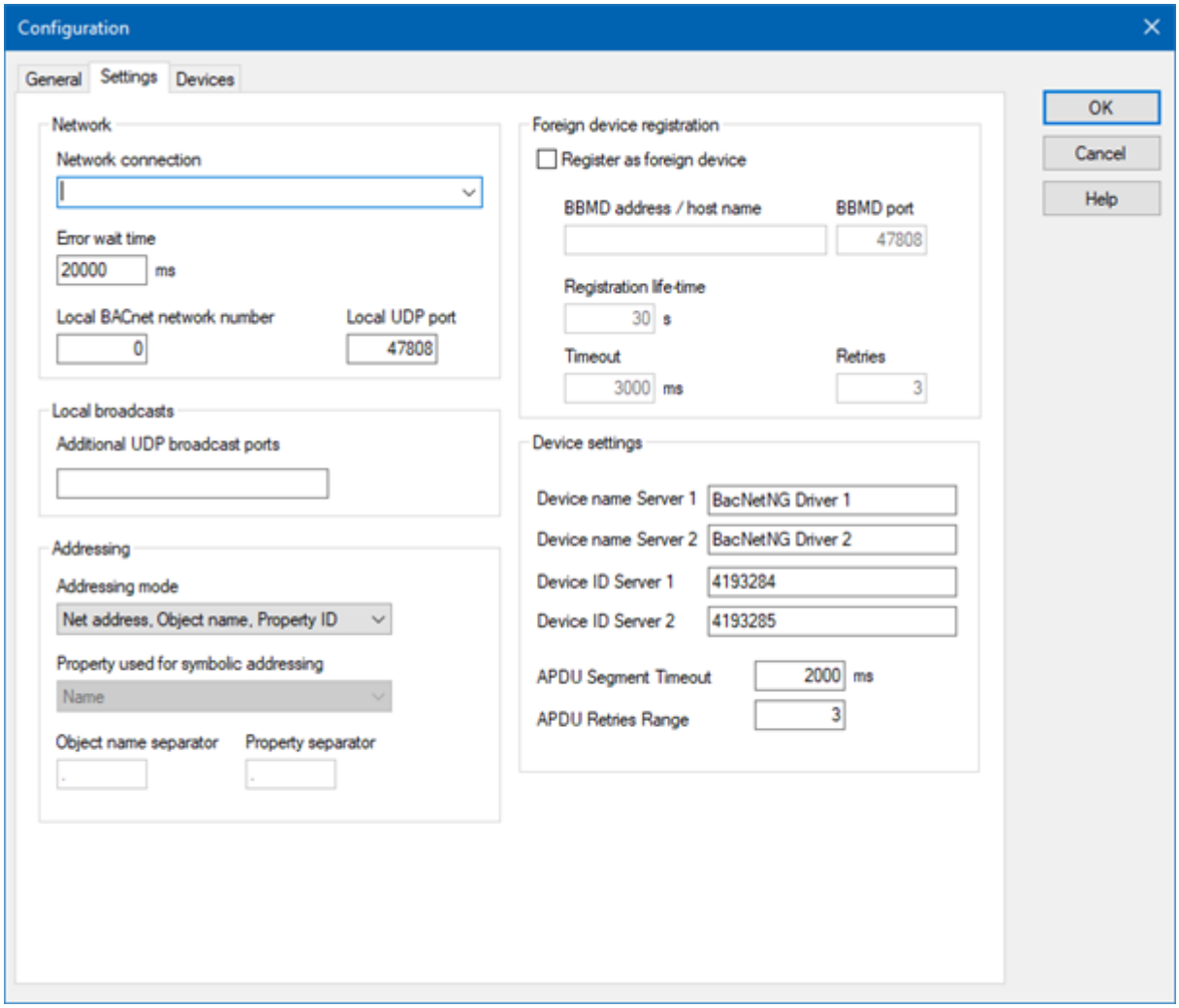
The **BACnetNG driver** uses option **Priority** in order to differentiate between the types of communication (*spontaneous/polling*).

- Variables with **Priority** *Normal* are communicated spontaneously via the *Change Of Value* (*DS-COV-A*) functionality.
- For all priorities except priority *Normal* the set update interval equals the polling cycle.
- The *Normal* time must be set to a higher value than all other values. However this time is not used.

For more information refer to chapter Access methods (on page 132) in this manual.

5.2.2 Settings

The settings configured in this tab are valid for all configured BACnet devices.



NETWORK

Network configurations for communication with the BACnet PLC.

Property	Description
Network connection	Name of the network connection or IP address which should be used for the BACnet/IP communication. Selection from drop-down list: The name matches the name which is displayed under **Control Panel -> Network connections**.

Property	Description
	Configuration and behavior: ▸ The drop-down list contains the names of all connections available at the system. As an alternative you can enter any name. Instead of a name, an IP address can also be entered manually. ▸ If you do not enter a name, the first network adapter of the system is used. **Note:** With systems with several network cards, always enter the connection name. ▸ If there is no adapter with a name that corresponds to the **Network connection** setting, the IP addresses of all adapters are searched. If a match is found, this address is used.
Error wait time	Error wait time in milliseconds. Default: *20000 ms*
Local BACnet network number	BACnet network number of the local BACnet network. Default: *1*
Local UDP port	Local UDP port which should be used for the communication Default: *47808*

LOCAL BROADCASTS

Property	Description
Additional UDP broadcast ports	In addition to the BACnet/IP port standard (*0xBAC0* or *47808*) BACnet/IP broadcasts can be directed to further UDP ports. Example: *47809; 47810* **Note:** Only available if **Register as foreign device** has been activated.

ADDRESSING

Settings for the exchange of data between driver and PLC.

Property	Description
Adressing mode	Type of addressing zenon variables. Select from drop-down list: ▸ *Net address, Object name, Property ID* ▸ *Net address, Object ID, Property ID* ▸ *Symbolic, Property ID* ▸ *Symbolic* Default: *Net address, Object name, Property ID* For more information about the different types of addressing refer to chapter Addressing (on page 31) in this manual.
Property used for addressing	Selection of addressing from drop-down list: ▸ *Name*: Corresponds to the **Name** variable property in the zenon Editor. ▸ *Identification*: Corresponds to the **Identification** variable property in the zenon Editor. ▸ *Symbolic address*: Corresponds to the **Symbolic address** variable property in the zenon Editor. Default:*Name* **Note:** This option is only available if the **Adressing mode** is configured with *Symbolic, Property ID* or *Symbolic*.
Object name separator	Entry of the separator for variable name, identification or symbolic address, in order to separate the device name from the object name. Default: Period (.) **Note**: The characters **@** and **#** are not permitted as separators. If one of the two characters is used, communication is not possible.
Property separator	Entry of the separator in variable name, identification or symbolic address, in order to separate the object name

Property	Description
	from the property name. Default: Period (.) **Note**: The characters **@** and **#** are not permitted as separators. If one of the two characters is used, communication is not possible.

FOREIGN DEVICE REGISTRATION

Settings for communication with a *BACnet Broadcast Management Device* for communication with a BACnet network in a different subnetwork.

Note: The **Register as foreign device** option must first be activated to configure the section.

Parameter	Description
Register as foreign device	Checkbox for the activation of communication to a BBMD: ▸ *active*: The BACnetNG driver registers itself on the configured BBMD (BACnet Broadcast Management Device) as a foreign device. ▸ *Inactive*: The driver does not communicate to a BBMD Default: *inactive*
BBMD address/host name	Host name or IP address of the BBMD to which the *foreign device* registration of the driver is sent. **Attention:** If a host name is used here, this can lead to problems with the current BACnet communication in the event of invalid naming.
BBMD port	Port for communication to the BACnet Broadcast Management Device. Default:*47808*
Registration life-time	Repetition rate in seconds for cyclical registration of the driver on the foreign device. Default:*30 s*
Timeout	Time limitation for registration in milliseconds. If no confirmation for registration is received within the

Parameter	Description
	configured time, a repetition is sent. Default:*3000 ms*
Retries	Repetition of the registration if no confirmation has been received within the configured timeout time. If the number of repetitions has been completed, this is considered an error. **Note:** After an error, a new registration is only attempted once the error wait time configured in the **Error wait time** option has expired. Default:*3*

DEVICE SETTING

Configuring the driver as **BACnet device**. The configurations at the zenon **BACnetNG driver** must be unique in the BACnet network (name and ID). For more information about the supported BACnet services and BACnet device object properties refer to chapter "Device object of the driver (on page 133)".

Parameter	Description
Device name Server 1	Name of the *Device Object* in the driver on the primary server or the standalone Runtime. Is to allow the *Device Object* to be identified uniquely in the network. Entry of an alphanumeric value. Only ANSI characters are permitted. At least one character must be entered. Default:*BacNetNG Driver 1*
Device name Server 2	Name of the *Device object* in the driver on the standby server. Is to allow the device object to be identified uniquely in the network. Entry of an alphanumeric value. Only ANSI characters are permitted. At least one character must be entered. Default:*BacNetNG Driver 2*
Device ID Server 1	*BACnetObjectIdentifier* of the driver on the primary server or the standalone zenon Runtime. Numeric value in order to identify the *Device Object* uniquely in the network. Default:*4193284*

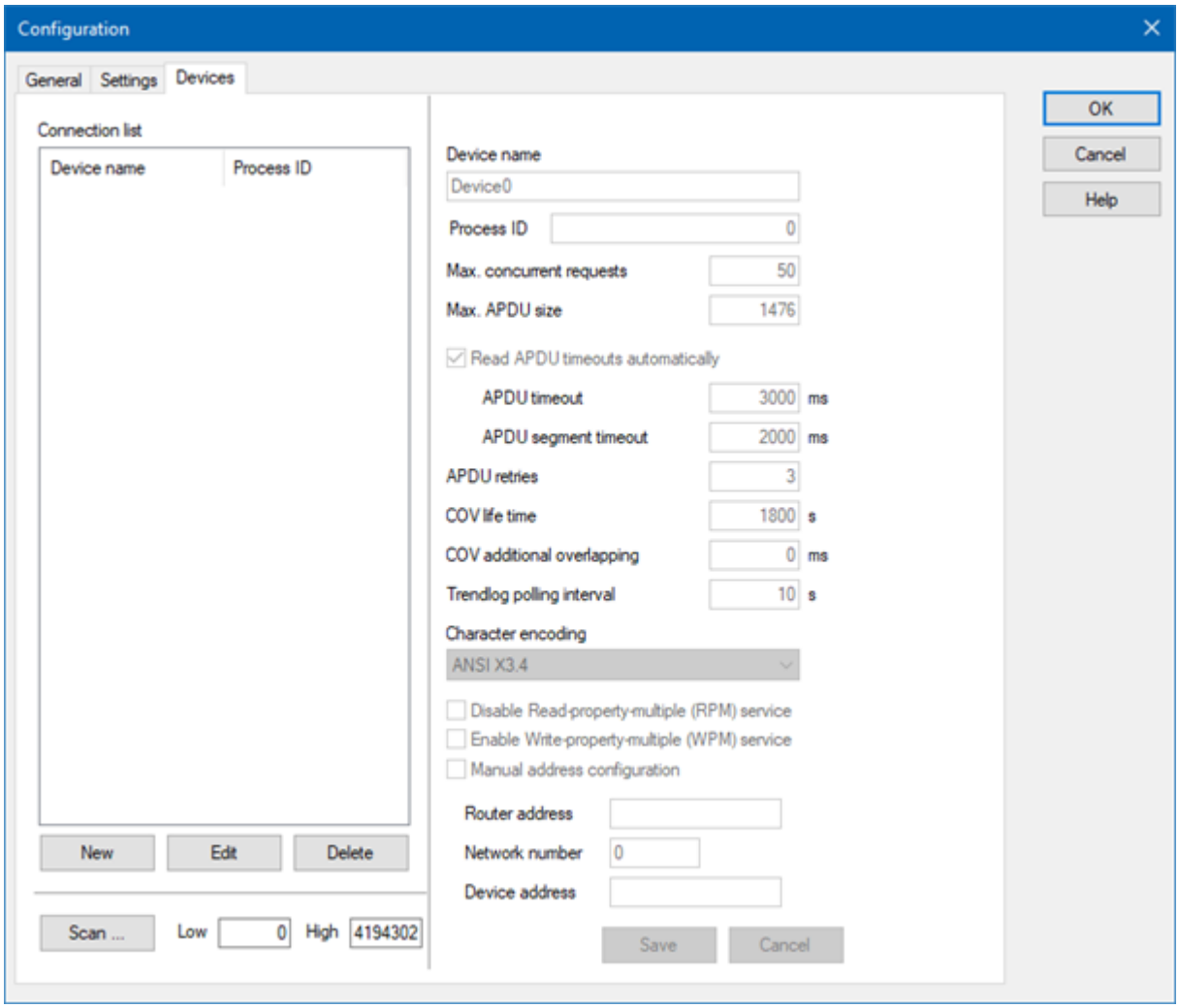
Parameter	Description
Device ID Server 2	*BACnetObjectIdentifier* of the driver on the standby server. Numeric value in order to identify the *Device Object* uniquely in the network. Default:*4193285*
APDU Segment Timeout	Time in milliseconds between the recurrences of APDU segments if the driver responds as a **BACnet device**. Default: *2000 ms*
APDU Retries Range	Maximum number of recurrences for APDUs that are requested by a **BACnet Device**. This is applicable for APDUs that need a confirmation but for which no confirmation has yet been received. Default: *3*

CLOSE DIALOG

Option	Description
OK	Applies all changes in all tabs and closes the dialog.
Cancel	Discards all changes in all tabs and closes the dialog.
Help	Opens online help.

5.2.3 Devices

LIST OF BACNET DEVICES AND DEVICE-SPECIFIC SETTINGS



Parameter	Description
Connection list	List of all configured devices.
New	Inserts a new device in the list.
Delete	Deletes the selected device from the list.
Edit	Edit an existing device.
Scan ...	Searches the network for *BACnet Devices* using their BACnet ID. Found devices are listed in the **Connection list**. If no device is found, a corresponding dialog is displayed.
Low	Setting the lower scan area. Set the highest ID of a *BACnet Devices*

Parameter	Description
	which can be found during a scan. Default:*0*
High	Setting the higher scan area. Set the highest ID of a *BACnet Device* which can be found during a scan. Default:*4194302*

DEVICE-SPECIFIC SETTINGS

Parameter	Description
Device name	Name of the BACnet device (name of the *device object*).
Process ID	Unique identification for the assignment of incoming *COV-Notifications* or *Event-Notifications*. Incoming *COV-Notifications* and *Event-Notifications* are assigned to a configured device using the value configured in this option. Default:*1* In doing so, the following applies: ▸ The **Process ID** must be unique per connection. ▸ This ID is automatically used for *COV-Subscriptions*. **Note:** For **Event-Notifications**, this process ID in the *Notification-Class Objekt* in the **BACnet Device** must correspond to the configuration configured in the driver.
Max. concurrent requests	Maximum number of concurrently-pending requests. ▸ Minimum: *1* ▸ Maximum: *256* Default: *50* The setting of the **Max. concurrent requests** influences the InvokeID of the readProperty(Multiple) requests used. The InvokeID *1 - 255* is used. If *256* **Max. concurrent requests** are given, the InvokeID *0* is also used. Some devices do not accept InvokeID *0*. If a device sends more requests than have been set, new buffers are reserved until the theoretical maximum of 256 is reached. **Note:** The setting should be amended to the number of connected devices for better performance.

Parameter	Description
	Note: If more than 128 simultaneous requests have been configured, it cannot be ensured that duplicates will be reliably detected. It is therefore absolutely recommended that a value less than 128 is used.
Max. APDU size	Maximum transferable size of an APDU or an APDU segment. Default: *1476*
Read APDU timeouts automatically	*Active*: The APDU timeout settings are read by the device and the timeouts set for the device applies.
APDU timeout [ms]	Timeout for the acknowledgment of a request. Default: *3000*
APDU segment timeout [ms]	Timeout for the acknowledgment of one or several segments of a request. Default: *2000*
APDU retries	Request retries in case of a timeout. Default: *3*
COV life time [s]	Life time of COV subscriptions. Default: *1800 s* **Note:** More information on the behavior of COV can be found in the **Access procedures (spontaneous and polled reading)** (on page 132) chapter in this manual.
COV additional overlapping [ms]	Additional parameter to determine when the renewed subscription is sent. Prevents, with a delayed BACnet reaction, the loss of sent values through a timeout. Default: *0* Formula: **(APDU_timeout***(**APDU_retries+***1*))**- COV_additional_overlapping**
Trendlog poll interval	Interval for the polling query of **Trendlog** driver object type variables. Default: 10
Character encoding	Character set coding used for communication with STRING variables. Select from drop-down list: ▸ UTF-8

Parameter	Description
	▸ ISO 10646 (UCS-4) ▸ ISO 10646 (UCS-2) ▸ ISO 8859-1
Disable Read-property-multiple (RPM) service	Behavior for receiving values of BACnet Device. ▸ *activated:* For connection buildup and for reading values BACnet service *Read-Property* is used exclusively. ▸ *inactive:* All values are read using BACnet service *Read-Property-Multiple*. If the *Read-Property-Multiple* is answered by *Reject – Service unsupported* during connection buildup, the **BACnetNG driver** repeats the request with BACnet service *Read-Property* and then uses for this BACnet Device only the service *Read-Property*. In this case a renewed *Read-Property-Multiple* is tried after a restart or after reloading the zenon Runtime. Default: *inactive* **Attention:** If *Read-Property Multiple* is not supported (**Disable RPM** option activated or the BACnet service is not supported by the BACnet device), no BACnet status-flags property is read and mapped to the zenon status bits for the *present-value*. Thus the variable always has zenon status SPONTANEOUS regardless of the BACnet status-flags.
Enable Write-property-multipe (WPM) service	Behavior for writing values to the PLC ▸ *activated:* For writing BACnet service *Write-Property-Multiple* is used exclusively. At that it is tried that if possible all *Write-Property-Requests* in the write queue are pooled together in a *Write-Property-Multiple Request*. If the maximum APDU size is reached, a new *Write-Property-Multiple-Request* is sent. **Example:** When writing recipes, the single *Write-Property-Requests* are pooled together in a *Write-Property-Multiple-Request*. ▸ *inactive:* For writing BACnet service *Write-Property* is used exclusively. Default: *inactive*

Parameter	Description
Manual address configuration	*Active*: The address is not determined automatically via broadcast and the device name but can be entered manually via **Router address**, **Network number** and **Device address**.
Router address	BACnet MAC address of the router. It consists of the IP address of the router and the UDP port; e.g. in the following format *192.168.0.5:47808*. **Note:** not active if the **Manual address configuration** property is inactive.
Network number	BACnet network number of the device.
Device address	BACnet MAC address of the device. For BACnet/IP devices, it consists of the IP address and the UDP port (see **Router address**), for example *192.168.0.5:47808*. For all other BACnet devices it consists of a byte sequence in hexadecimal format separated by *:* (e.g. *06:0A:67:EE*).
Save	Saves the configuration for the selected *Device*.
Cancel	Discards the changes of the configuration for the selected *Device*.

CLOSE DIALOG

Options	Description
OK	Applies settings and closes the dialog.
Cancel	Discards all changes and closes the dialog.
Help	Opens online help.

6 Creating variables

This is how you can create variables in the zenon Editor:

6.1 Creating variables in the Editor

Variables can be created:

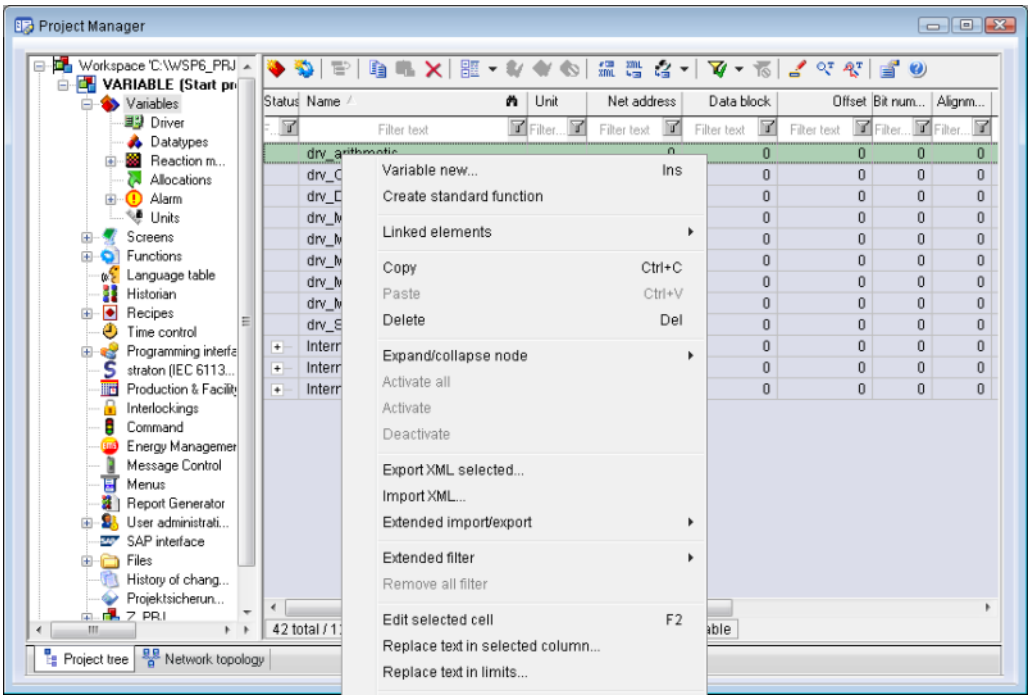
- as simple variables

- in arrays
- as structure variables

VARIABLE DIALOG

To create a new variable, regardless of which type:

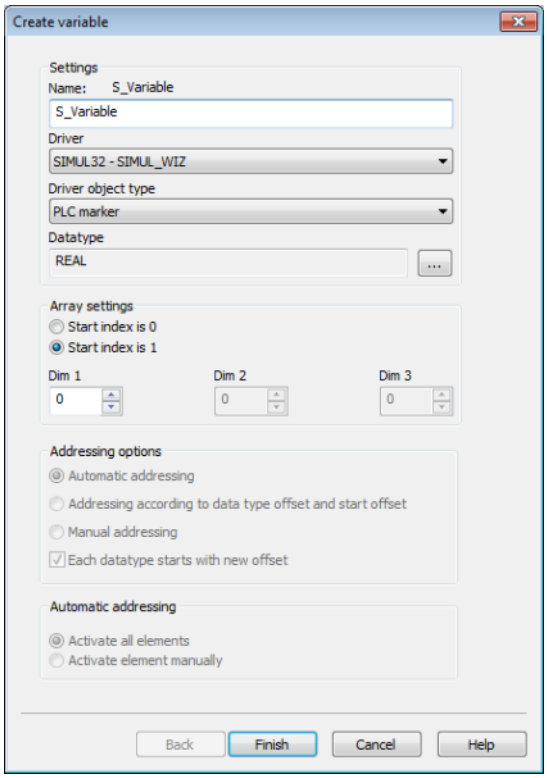
1. Select the **New variable** command in the **Variables** node in the context menu



   The dialog for configuring variables is opened

2. Configure the variable
3. The settings that are possible depend on the type of variables

CREATE VARIABLE DIALOG



Property	Description
Name	Distinct name of the variable. If a variable with the same name already exists in the project, no additional variable can be created with this name. Maximum length: 128 characters **Attention:** the characters **#** and **@** are not permitted in variable names. If non-permitted characters are used, creation of variables cannot be completed and the **Finish** button remains inactive. **Note:** Some drivers also allow addressing using the **Symbolic address** property.
Driver	Select the desired driver from the drop-down list. **Note:** If no driver has been opened in the project, the driver for internal variables (**Intern.exe**) is automatically loaded.
Driver Object Type	Select the appropriate driver object type from the drop-down list.
Data Type	Select the desired data type. Click on the ... button to open the selection dialog.
Array settings	Expanded settings for array variables. You can find details in the

Property	Description
	Arrays chapter.
Addressing options	Expanded settings for arrays and structure variables. You can find details in the respective section.
Automatic element activation	Expanded settings for arrays and structure variables. You can find details in the respective section.

SYMBOLIC ADDRESS

The **Symbolic address** property can be used for addressing as an alternative to the **Name** or **Identification** of the variables. Selection is made in the driver dialog; configuration is carried out in the variable property. When importing variables of supported drivers, the property is entered automatically.

Maximum length: *1024* characters.

The following drivers support the **Symbolic address**:

- **3S_V3**
- **AzureDrv**
- **BACnetNG**
- **IEC850**
- **KabaDPServer**
- **OPCUA32**
- **Phoenix32**
- **POZYTON**
- **RemoteRT**
- **S7TIA**
- **SEL**
- **SnmpNg32**
- **PA_Drv**
- **EUROMAP63**

INHERITANCE FROM DATA TYPE

Measuring range, **Signal range** and **Set value** are always:

- derived from the datatype
- Automatically adapted if the data type is changed

Note for signal range: If a change is made to a data type that does not support the set **signal range**, the **signal range** is amended automatically. For example, for a change from **INT** to **SINT**, the **signal range** is changed to *127*. The amendment is also carried out if the **signal range** was not inherited from the data type. In this case, the **measuring range** must be adapted manually.

6.2 Addressing

Group/Property	Description
General	Property group for general settings.
Name	Name of the variable consists of device name, object name and property name. Devices, object name and property name are separated by a period (.). Each separator can be configured. If a single item of an array should be addressed, the offset must be stated as postfix in brackets. (for example: **MyServer.AnalogValue7<priority-array[8]**) **Attention:** For every zenon project the name must be unambiguous. The device object of a devices is also addressed in accordance with the scheme mentioned above. In this case, the device name is repeated as the object name (for example **MyServer.MyServer<object-name**).
Identification	Can be used as an alternative for addressing variables if it is set in the driver configuration. The format for name and identification is identical.
Addressing	
Net address	Is used if at driver configuration in the **Settings** tab option **Adressing mode** is projected with *Net address*, *Object name*, *Property ID* or *Net address*, *Object ID*, *Property ID*.
Data block	not used for this driver
Offset	not used for this driver
Alignment	not used for this driver
Bit number	not used for this driver
String length	Only available for String variables. Maximum number of characters that the variable can take.
Driver connection/Driver	Object type of the variables. Depending on the driver used, is selected when the variable is created and can be changed here.

Group/Property	Description
Object Type	
Driver connection/Data Type	Data type of the variable. Is selected during the creation of the variable; the type can be changed here. **Attention:** If you change the data type later, all other properties of the variable must be checked and adjusted, if necessary.

Group/Property	Description
BACnet data type	BACnet data type of the property. With the help of the data type you can define how a numeric variable or a string variable are mapped to a BACnet property. For a description of the possible data types refer to chapter "Data types supported by the driver for BACnet object properties (on page 36)".
BACnet object instance ID	Is used if at driver configuration in the **Settings** tab option **Adressing mode** is projected with *Net address*, *Object ID*, *Property ID*.
BACnet object name	Is used if at driver configuration in the **Settings** tab option **Adressing mode** is projected with *Net address*, *Object name*, *Property ID*.
BACnet object type	For user-specific properties or when you do not want to state the property in the name or in the identification, you can set the BACnet Object ID there.
BACnet property ID	For user-specific properties or when you do not want to state the property in the name or in the identification, you can set the BACnet Property ID there. **Attention:** Only if you set *-1* as Property ID, the property is determined via the name or identification.
BACnet trend-log variable type	Selection which *BACnet Trendlog Object properties* should be displayed for the variable. ▸ *Log status* ▸ *Time chnge* ▸ *Value* Can only be used for variables of **driver object type** *Trendlog*.
BACnet string length	Only available if you select *BITSTRING* as **BACnet data type**. Defines the bit length of the bit string.
BACnet write priority	Write priority. Valid write priorities are *1 - 16*. If you enter *0*, the variable is written without a write priority. Possible entries: *0 ... 16* Default: *0* The priority is set in the priority-array object property in the corresponding bit. To reset the priority of a written value, i.e. to reset the bit in the

Group/Property	Description
	priority-array, you need an additional variable. The variable must be a *String* data type, address the same object and have the same **BACnet property ID** and **BACnet data type** '*NULL*'. You set an empty string to this variable (with write set value function, for example). The set **BACnet write priority** is reset; the previously-set set value with this priority is discarded.
Priority	Read priority / read method The BACnet driver evaluates this property in order to distinguish the read method. Normal: Spontaneous reading via COV subscriptions **Attention:** Only for **PRESENT-VALUE** and **STATUS-FLAGS** for stating the update cycle in accordance with the driver configuration. Possible values: ▸ increased ▸ high ▸ highest
Symbolic address	The **Symbolic address** property can be used for addressing as an alternative to the **Name** or **Identification** of the variables. Selection is made in the driver dialog; configuration is carried out in the variable property. When importing variables of supported drivers, the property is entered automatically. Maximum length: *1024* characters.

6.2.1 Addressing mode

The type of addressing is configured in the driver's configuration dialog on the **Settings** tab in the **Adressing Mode** option.

NET ADDRESS, OBJECT ID, PROPERTY ID – DEFAULT

- Addressing via variable properties, **Net address**, **BACnetObjectName** and **BACnet property ID**/**BACnet event state**/**BACnetTrendVarType**.
- Assignment to the connection via the variable property.**Net address**. The configured network address must correspond to the connection configuration (on page 23) option **Process ID**.

- Object assignment using the variable property **BACnetObjectType** and **BACnetObjectInstanceID**.
- *Property ID/Event state/Trend-log value type* from the corresponding variable properties **BACnet property ID**/**BACnet event state**/**BACnetTrendVarType**.
- The array index is determined by the variable name (if the name ends with [x] and x is a numeric string).
- Behavior in zenon Runtime:
  The **Object ID**is not read because it has already been configured.

NET ADDRESS, OBJECT NAME, PROPERTY ID

- Addressing via variable properties, **Net address**, **BACnetObjectName** and **BACnet property ID**/**BACnet event state**/**BACnetTrendVarType**.
- Assignment to the connection using the variable properties**Net address**. The configured network address must correspond to the connection configuration (on page 23) option **Process ID**.
- Object name is read from the variable property **BACnetObjectName**.
- *Property ID/Event state/Trend-log value type* from the corresponding variable property **BACnet property ID**/**BACnet event state**/**BACnetTrendVarType**.
- The array index is determined by the variable name (if the name ends with [x] and x is a numeric string).
- Behavior in zenon Runtime
  The **Object ID** is read when the connection is established.

SYMBOLIC, PROPERTY ID

Corresponds to the behavior when the **Do not read property/event state from address string** option, which was used up to Version 8.00, has been activated.

- Using Object name separator.
- Assignment to the connection via the device name in the address.
- Assignment to the Object via the object name in the address.
- Symbolic addressing via the set variable property:
  - **Identification**
  - **Symbolic address**
- The addressing of array elements is not supported.
- Trend log value type is determined from the address.

- Behavior in zenon Runtime
  The **Object ID** is read when the connection is established.

SYMBOLIC

Corresponds to the behavior when the **Do not read property/event state from address string** option, which was used up to Version 8.00, has been deactivated.

- Use of Object name separator and Property separator.
- Symbolic addressing via the set variable property:
  - **Name**
  - **Identification**
  - **Symbolic address**
- Assignment to the connection via the device name in the address.
- Assignment to the Object via the object name in the address.
- Property ID/event state is determined from the address if the variable property **BACnet property ID** contains the **BACnet event state** value *-1*.
- The array index is determined by the address.
- Trend log value type is determined from the address.
- Behavior in zenon Runtime
  The **Object ID** is read when the connection is established.

6.2.2 Addressing

§§Trenner

BACnet data type	Description	Compatible IEC types	String coding example
NULL	Data type without a value	*String*	[0]
BOOLEAN	Boolean	*BOOL, String*	TRUE
UNSIGNED	Positive integer	*UDINT, String*	8
SIGNED	Integer	*DINT, String*	-1
REAL	Float	*REAL, String*	7.9

BACnet data type	Description	Compatible IEC types	String coding example
DOUBLE	Float	*LREAL*, *String*	8.0
OCTETSTRING	Bytes sequence	*String*	65A8B900
CHARACTERSTRING	String	*String*	abcd
BITSTRING	Bit string	*String*, *UDINT*(if size < 32)	01110
ENUMERATED	Enumeration	*UDINT*, *String*	8
DATE	Date	*UDINT*, *String*	109.12.1.7
TIME	Time	*UDINT*, *String*	17:00:00:000
OBJECTIDENTIFIER	Object ID	*String*	0008 0000001
[Any]	Any value	*String*	[2] 78
[DateTime]	Date (DATE) followed by time (TIME)	*String*	{[10] 109.12.1.7},{[11] 17:00:00:000}
[TimeStamp]	BACnetTimeStamp Contains either: *0: TIME* *1: Unsigned* *2: [DateTime]*	*String*	{<2> {[10] 109.12.1.7},{[11] 17:00:00:000}}
[PriorityArray]	Array with 16 elements of the same data type as the *PRESENT-VALUE* property or NULL	*String*	{[0]},{[4] 5.0},{[0]}, ...
[Recipient]	BACnetRecipient	*String*	
[DeviceObjectReference]	BACnetDeviceObjectReference	*String*	
[DeviceObjectPropertyReference]	BACnetDeviceObjectPropertyReference	*String*	

BACnet data type	Description	Compatible IEC types	String coding example
[ObjectPropertyReference]	BACnetObjectProperyReference	*String*	
[SetpointReference]	BACnetSetPointReference	*String*	
[ActionList]	BACnetActionList	*String*	
[EventParamater]	BACnetEventParameter	*String*	
[DateRange]	Start date (DATE) followed by end date (DATE)	*String*	{109.12.1.255},{109.12.2.255}
[DailySchedule]	BACnetDailySchedule List of: TIME followed by [Any]	*String*	
[SpecialEvent]	BACnetSpecialEvent	*String*	{<0> {(2) 1.1.1}},{<2> {[11] 17:00:00:000},{[1] TRUE}},{(3) 1}
[VTSession]	BACnetVTSession	*String*	
[SessionKey]	BACnetSessionKey	*String*	
[CalendarEntry]	BACnetCalendarEntry Contains either: *0: DATE* *1: [DateRange]* *2: [WeekNDay]*	*String*	(2) 255.255.1
[AddressBinding]	BACnetAddressBinding	*String*	
[COVSubscription]	BACnetCOVSubscription	*String*	
[ReadAccessSpec	BACnetReadAccessS	*String*	

BACnet data type	Description	Compatible IEC types	String coding example
ification]	pecification		
[ReadAccessResult]	BACnetReadAccessResult	*String*	
[Destination]	BACnetDestination	*String*	
[LogRecord]	BACnetLogRecord	*String*	
[WeekNDay]	BACnetWeekNDay	*String*	3.3.1
BITSTRING (StatusFlags)	Bit string with length 4	*String*	0010
BITSTRING (EventTransitionBits)	Bit string with length 3	*String*	111
BITSTRING (ServicesSupported)	Bit string with length 40	*String*	
[Raw]	Data should always be coded as bytes sequence.	*String*	%10% 6D071603

6.2.3 Addressing

ACCUMULATOR(23)

PropertyName	PropertyDatatype	PropertyConformanceCode
object-identifier(75)	*[ObjectIdentifier]*	Read
object-name(77)	*[CharacterString]*	Read
object-type(79)	*ENUMERATED*	Read
present-value(85)	*[Unsigned]*	Read
description(28)	*[CharacterString]*	Optional
device-type(31)	*[CharacterString]*	Optional

PropertyName	PropertyDatatype	PropertyConformanceCode
status-flags(111)	*BITSTRING*	Read
event-state(36)	*ENUMERATED*	Read
reliability(103)	*ENUMERATED*	Optional
out-of-service(81)	*[BOOLEAN]*	Read
scale(187)	*[Scale]*	Read
units(117)	*ENUMERATED*	Read
prescale(185)	*[Prescale]*	Optional
max-pres-value(65)	*[Unsigned]*	Read
value-change-time(192)	*[DateTime]*	Optional
value-before-change(190)	*[Unsigned]*	Optional
value-set(191)	*[Unsigned]*	Optional
logging-record(184)	*[AccumulatorRecord]*	Optional
logging-object(183)	*[ObjectIdentifier]*	Optional
pulse-rate(186)	*[Unsigned]*	Optional
high-limit(45)	*[Unsigned]*	Optional
low-limit(59)	*[Unsigned]*	Optional
limit-monitoring-interval(182)	*[Unsigned]*	Optional
notification-class(17)	*[Unsigned]*	Optional
time-delay(113)	*[Unsigned]*	Optional
limit-enable(52)	*BITSTRING*	Optional
event-enable(35)	*BITSTRING*	Optional
acked-transitions(0)	*BITSTRING*	Optional
notify-type(72)	*ENUMERATED*	Optional
event-time-stamps(130)	*[TimeStamp], Array[3]*	Optional
event-message-texts(351)	*[CharacterString], Array[3]*	Optional

PropertyName	PropertyDatatype	PropertyConformanceCode
event-message-texts-config(352)	*[CharacterString], Array[3]*	Optional
event-detection-enable(353)	*[BOOLEAN]*	Optional
event-algorithm-inhibit-ref(355)	*[ObjectPropertyReference]*	Optional
event-algorithm-inhibit(354)	*[BOOLEAN]*	Optional
time-delay-normal(356)	*[Unsigned]*	Optional
reliability-evaluation-inhibit(357)	*[BOOLEAN]*	Optional
property-list(371)	*ENUMERATED, Array[N]*	Read
profile-name(168)	*[CharacterString]*	Optional

ANALOGINPUT(0)

PropertyName	PropertyDatatype	PropertyConformanceCode
object-name(77)	*[ObjectIdentifier]*	Read
OBJECT-TYPE(79)	*[CharacterString]*	Read
present-value(85)	*ENUMERATED*	Read
description(28)	*[CharacterString]*	Read
device-type(31)	*[CharacterString]*	Optional
status-flags(111)	*[CharacterString]*	Optional
event-state(36)	*BITSTRING*	Read
reliability(103)	*ENUMERATED*	Read
out-of-service(81)	*ENUMERATED*	Optional
update-interval(118)	*[BOOLEAN]*	Read
units(117)	*[Unsigned]*	Optional

PropertyName	PropertyDatatype	PropertyConformanceCode
min-pres-value(69)	*ENUMERATED*	Read
max-pres-value(65)	*[REAL]*	Optional
resolution(106)	*[REAL]*	Optional
cov-increment(22)	*[REAL]*	Optional
time-delay(113)	*[REAL]*	Optional
notification-class(17)	*[Unsigned]*	Optional
high-limit(45)	*[Unsigned]*	Optional
low-limit(59)	*[REAL]*	Optional
deadband(25)	*[REAL]*	Optional
limit-enable(52)	*[REAL]*	Optional
event-enable(35)	*BITSTRING*	Optional
acked-transitions(0)	*BITSTRING*	Optional
notify-type(72)	*BITSTRING*	Optional
event-time-stamps(130)	*ENUMERATED*	Optional
event-message-texts(351)	*[TimeStamp], Array[3]*	Optional
event-message-texts-config(352)	*[CharacterString], Array[3]*	Optional
event-detection-enable(353)	*[CharacterString], Array[3]*	Optional
event-algorithm-inhibit-ref(355)	*[BOOLEAN]*	Optional
event-algorithm-inhibit(354)	*[ObjectPropertyReference]*	Optional
time-delay-normal(356)	*[BOOLEAN]*	Optional
reliability-evaluation-inhibit(357)	*[Unsigned]*	Optional
property-list(371)	*[BOOLEAN]*	Optional

PropertyName	PropertyDatatype	PropertyConformanceCode
profile-name(168)	*ENUMERATED, Array[N]*	Read
object-identifier(75)	*[CharacterString]*	Optional

ANALOGOUTPUT(1)

PropertyName	PropertyDatatype	PropertyConformanceCode
object-identifier(75)	*[ObjectIdentifier]*	Read
object-name(77)	*[CharacterString]*	Read
object-type(79)	*ENUMERATED*	Read
present-value(85)	*[REAL]*	Write
description(28)	*[CharacterString]*	Optional
device-type(31)	*[CharacterString]*	Optional
status-flags(111)	*BITSTRING*	Read
event-state(36)	*ENUMERATED*	Read
reliability(103)	*ENUMERATED*	Optional
out-of-service(81)	*[BOOLEAN]*	Read
units(117)	*ENUMERATED*	Read
min-pres-value(69)	*[REAL]*	Optional
max-pres-value(65)	*[REAL]*	Optional
resolution(106)	*[REAL]*	Optional
priority-array(87)	*[PriorityArray]*	Read
relinquish-default(104)	*[REAL]*	Read
cov-increment(22)	*[REAL]*	Optional
time-delay(113)	*[Unsigned]*	Optional
notification-class(17)	*[Unsigned]*	Optional
high-limit(45)	*[REAL]*	Optional

PropertyName	PropertyDatatype	PropertyConformanceCode
low-limit(59)	*[REAL]*	Optional
deadband(25)	*[REAL]*	Optional
limit-enable(52)	*BITSTRING*	Optional
event-enable(35)	*BITSTRING*	Optional
acked-transitions(0)	*BITSTRING*	Optional
notify-type(72)	*ENUMERATED*	Optional
event-time-stamps(130)	*[TimeStamp], Array[3]*	Optional
event-message-texts(351)	*[CharacterString], Array[3]*	Optional
event-message-texts-config(352)	*[CharacterString], Array[3]*	Optional
event-detection-enable(353)	*[BOOLEAN]*	Optional
event-algorithm-inhibit-ref(355)	*[ObjectPropertyReference]*	Optional
event-algorithm-inhibit(354)	*[BOOLEAN]*	Optional
time-delay-normal(356)	*[Unsigned]*	Optional
reliability-evaluation-inhibit(357)	*[BOOLEAN]*	Optional
property-list(371)	*ENUMERATED, Array[N]*	Read
profile-name(168)	*[CharacterString]*	Optional

ANALOGVALUE(2)

PropertyName	PropertyDatatype	PropertyConformanceCode
object-identifier(75)	*[ObjectIdentifier]*	Read
object-name(77)	*[CharacterString]*	Read
object-type(79)	*ENUMERATED*	Read

PropertyName	PropertyDatatype	PropertyConformanceCode
present-value(85)	*[REAL]*	Read
description(28)	*[CharacterString]*	Optional
status-flags(111)	*BITSTRING*	Read
event-state(36)	*ENUMERATED*	Read
reliability(103)	*ENUMERATED*	Optional
out-of-service(81)	*[BOOLEAN]*	Read
units(117)	*ENUMERATED*	Read
priority-array(87)	*[PriorityArray]*	Optional
relinquish-default(104)	*[REAL]*	Optional
cov-increment(22)	*[REAL]*	Optional
time-delay(113)	*[Unsigned]*	Optional
notification-class(17)	*[Unsigned]*	Optional
high-limit(45)	*[REAL]*	Optional
low-limit(59)	*[REAL]*	Optional
deadband(25)	*[REAL]*	Optional
limit-enable(52)	*BITSTRING*	Optional
event-enable(35)	*BITSTRING*	Optional
acked-transitions(0)	*BITSTRING*	Optional
notify-type(72)	*ENUMERATED*	Optional
event-time-stamps(130)	*[TimeStamp], Array[3]*	Optional
event-message-texts(351)	*[CharacterString], Array[3]*	Optional
event-message-texts-config(352)	*[CharacterString], Array[3]*	Optional
event-detection-enable(353)	*[BOOLEAN]*	Optional
event-algorithm-inhibit-re	*[ObjectPropertyReference]*	Optional

PropertyName	PropertyDatatype	PropertyConformanceCode
f(355)		
event-algorithm-inhibit(354)	*[BOOLEAN]*	Optional
time-delay-normal(356)	*[Unsigned]*	Optional
reliability-evaluation-inhibit(357)	*[BOOLEAN]*	Optional
min-pres-value(69)	*[REAL]*	Optional
max-pres-value(65)	*[REAL]*	Optional
resolution(106)	*[REAL]*	Optional
property-list(371)	*ENUMERATED, Array[N]*	Read
profile-name(168)	*[CharacterString]*	Optional

AVERAGING(18)

PropertyName	PropertyDatatype	PropertyConformanceCode
object-identifier(75)	*[ObjectIdentifier]*	Read
object-name(77)	*[CharacterString]*	Read
object-type(79)	*ENUMERATED*	Read
minimum-value(136)	*[REAL]*	Read
minimum-value-timestamp(150)	*[DateTime]*	Optional
average-value(125)	*[REAL]*	Read
variance-value(151)	*[REAL]*	Optional
maximum-value(135)	*[REAL]*	Read
maximum-value-timestamp(149)	*[DateTime]*	Optional
description(28)	*[CharacterString]*	Optional
attempted-samples(124)	*[Unsigned]*	Write

PropertyName	PropertyDatatype	PropertyConformanceCode
valid-samples(146)	*[Unsigned]*	Read
object-property-reference (78)	*[DeviceObjectPropertyReference]*	Read
window-interval(147)	*[Unsigned]*	Write
window-samples(148)	*[Unsigned]*	Write
property-list(371)	*ENUMERATED, Array[N]*	Read
profile-name(168)	*[CharacterString]*	Optional

BINARYINPUT(3)

PropertyName	PropertyDatatype	PropertyConformanceCode
object-identifier(75)	*[ObjectIdentifier]*	Read
object-name(77)	*[CharacterString]*	Read
object-type(79)	*ENUMERATED*	Read
present-value(85)	*ENUMERATED*	Read
description(28)	*[CharacterString]*	Optional
device-type(31)	*[CharacterString]*	Optional
status-flags(111)	*BITSTRING*	Read
event-state(36)	*ENUMERATED*	Read
reliability(103)	*ENUMERATED*	Optional
out-of-service(81)	*[BOOLEAN]*	Read
polarity(84)	*ENUMERATED*	Read
inactive-text(46)	*[CharacterString]*	Optional
active-text(4)	*[CharacterString]*	Optional
change-of-state-time(16)	*[DateTime]*	Optional
change-of-state-count(15)	*[Unsigned]*	Optional
time-of-state-count-reset(	*[DateTime]*	Optional

PropertyName	PropertyDatatype	PropertyConformanceCode
115)		
elapsed-active-time(33)	*UNSIGNED*	Optional
time-of-active-time-reset(114)	*[DateTime]*	Optional
time-delay(113)	*[Unsigned]*	Optional
notification-class(17)	*[Unsigned]*	Optional
alarm-value(6)	*ENUMERATED*	Optional
event-enable(35)	*BITSTRING*	Optional
acked-transitions(0)	*BITSTRING*	Optional
notify-type(72)	*ENUMERATED*	Optional
event-time-stamps(130)	*[TimeStamp], Array[3]*	Optional
event-message-texts(351)	*[CharacterString], Array[3]*	Optional
event-message-texts-config(352)	*[CharacterString], Array[3]*	Optional
event-detection-enable(353)	*[BOOLEAN]*	Optional
event-algorithm-inhibit-ref(355)	*[ObjectPropertyReference]*	Optional
event-algorithm-inhibit(354)	*[BOOLEAN]*	Optional
time-delay-normal(356)	*[Unsigned]*	Optional
reliability-evaluation-inhibit(357)	*[BOOLEAN]*	Optional
property-list(371)	*ENUMERATED, Array[N]*	Read
profile-name(168)	*[CharacterString]*	Optional

BINARYOUTPUT(4)

PropertyName	PropertyDatatype	PropertyConformanceCode
object-identifier(75)	*[ObjectIdentifier]*	Read
object-name(77)	*[CharacterString]*	Read
object-type(79)	*ENUMERATED*	Read
present-value(85)	*ENUMERATED*	Write
description(28)	*[CharacterString]*	Optional
device-type(31)	*[CharacterString]*	Optional
status-flags(111)	*BITSTRING*	Read
event-state(36)	*ENUMERATED*	Read
reliability(103)	*ENUMERATED*	Optional
out-of-service(81)	*[BOOLEAN]*	Read
polarity(84)	*ENUMERATED*	Read
inactive-text(46)	*[CharacterString]*	Optional
active-text(4)	*[CharacterString]*	Optional
change-of-state-time(16)	*[DateTime]*	Optional
change-of-state-count(15)	*[Unsigned]*	Optional
time-of-state-count-reset(115)	*[DateTime]*	Optional
elapsed-active-time(33)	*UNSIGNED*	Optional
time-of-active-time-reset(114)	*[DateTime]*	Optional
minimum-off-time(66)	*UNSIGNED*	Optional
minimum-on-time(67)	*UNSIGNED*	Optional
priority-array(87)	*[PriorityArray]*	Read
relinquish-default(104)	*ENUMERATED*	Read
time-delay(113)	*[Unsigned]*	Optional

PropertyName	PropertyDatatype	PropertyConformanceCode
notification-class(17)	*[Unsigned]*	Optional
feedback-value(40)	*ENUMERATED*	Optional
event-enable(35)	*BITSTRING*	Optional
acked-transitions(0)	*BITSTRING*	Optional
notify-type(72)	*ENUMERATED*	Optional
event-time-stamps(130)	*[TimeStamp], Array[3]*	Optional
event-message-texts(351)	*[CharacterString], Array[3]*	Optional
event-message-texts-config(352)	*[CharacterString], Array[3]*	Optional
event-detection-enable(353)	*[BOOLEAN]*	Optional
event-algorithm-inhibit-ref(355)	*[ObjectPropertyReference]*	Optional
event-algorithm-inhibit(354)	*[BOOLEAN]*	Optional
time-delay-normal(356)	*[Unsigned]*	Optional
reliability-evaluation-inhibit(357)	*[BOOLEAN]*	Optional

BINARYVALUE(5)

PropertyName	PropertyDatatype	PropertyConformanceCode
object-identifier(75)	*[ObjectIdentifier]*	Read
object-name(77)	*[CharacterString]*	Read
object-type(79)	*ENUMERATED*	Read
present-value(85)	*ENUMERATED*	Read
description(28)	*[CharacterString]*	Optional
status-flags(111)	*BITSTRING*	Read

PropertyName	PropertyDatatype	PropertyConformanceCode
event-state(36)	*ENUMERATED*	Read
reliability(103)	*ENUMERATED*	Optional
out-of-service(81)	*[BOOLEAN]*	Read
inactive-text(46)	*[CharacterString]*	Optional
active-text(4)	*[CharacterString]*	Optional
change-of-state-time(16)	*[DateTime]*	Optional
change-of-state-count(15)	*UNSIGNED*	Optional
time-of-state-count-reset(115)	*[DateTime]*	Optional
elapsed-active-time(33)	*UNSIGNED*	Optional
time-of-active-time-reset(114)	*[DateTime]*	Optional
minimum-off-time(66)	*UNSIGNED*	Optional
minimum-on-time(67)	*UNSIGNED*	Optional
priority-array(87)	*[PriorityArray]*	Optional
relinquish-default(104)	*ENUMERATED*	Optional
time-delay(113)	*[Unsigned]*	Optional
notification-class(17)	*[Unsigned]*	Optional
alarm-value(6)	*ENUMERATED*	Optional
event-enable(35)	*BITSTRING*	Optional
acked-transitions(0)	*BITSTRING*	Optional
notify-type(72)	*ENUMERATED*	Optional
event-time-stamps(130)	*[TimeStamp], Array[3]*	Optional
event-message-texts(351)	*[CharacterString], Array[3]*	Optional
event-message-texts-config(352)	*[CharacterString], Array[3]*	Optional

PropertyName	PropertyDatatype	PropertyConformanceCode
event-detection-enable(353)	*[BOOLEAN]*	Optional
event-algorithm-inhibit-ref(355)	*[ObjectPropertyReference]*	Optional
event-algorithm-inhibit(354)	*[BOOLEAN]*	Optional
time-delay-normal(356)	*[Unsigned]*	Optional
reliability-evaluation-inhibit(357)	*[BOOLEAN]*	Optional
property-list(371)	*ENUMERATED, Array[N]*	Read
profile-name(168)	*[CharacterString]*	Optional

CALENDAR(6)

PropertyName	PropertyDatatype	PropertyConformanceCode
object-identifier(75)	*[ObjectIdentifier]*	Read
object-name(77)	*[CharacterString]*	Read
object-type(79)	*ENUMERATED*	Read
description(28)	*[CharacterString]*	Optional
present-value(85)	*[BOOLEAN]*	Read
date-list(23)	*[CalendarEntry], List*	Read
property-list(371)	*ENUMERATED, Array[N]*	Read
profile-name(168)	*[CharacterString]*	Optional

COMMAND(7)

PropertyName	PropertyDatatype	PropertyConformanceCode
object-identifier(75)	*[ObjectIdentifier]*	Read
object-name(77)	*[CharacterString]*	Read

PropertyName	PropertyDatatype	PropertyConformanceCode
object-type(79)	*ENUMERATED*	Read
description(28)	*[CharacterString]*	Optional
present-value(85)	*[Unsigned]*	Write
in-process(47)	*[BOOLEAN]*	Read
all-writes-successful(9)	*[BOOLEAN]*	Read
action(2)	*[ActionList], Array[N]*	Read
action-text(3)	*[CharacterString], Array[N]*	Optional
property-list(371)	*ENUMERATED, Array[N]*	Read
profile-name(168)	*[CharacterString]*	Optional

DEVICE(8)

PropertyName	PropertyDatatype	PropertyConformanceCode
object-identifier(75)	*[ObjectIdentifier]*	Read
object-name(77)	*[CharacterString]*	Read
object-type(79)	*ENUMERATED*	Read
system-status(112)	*ENUMERATED*	Read
vendor-name(121)	*[CharacterString]*	Read
vendor-identifier(120)	*UNSIGNED*	Read
model-name(70)	*[CharacterString]*	Read
firmware-revision(44)	*[CharacterString]*	Read
application-software-version(12)	*[CharacterString]*	Read
location(58)	*[CharacterString]*	Optional
description(28)	*[CharacterString]*	Optional
protocol-version(98)	*[Unsigned]*	Read
protocol-revision(139)	*[Unsigned]*	Read

PropertyName	PropertyDatatype	PropertyConformanceCode
protocol-services-supported(97)	*BITSTRING*	Read
protocol-object-types-supported(96)	*BITSTRING*	Read
object-list(76)	*[ObjectIdentifier], Array[N]*	Read
structured-object-list(209)	*[ObjectIdentifier], Array[N]*	Optional
max-apdu-length-accepted(62)	*[Unsigned]*	Read
segmentation-supported(107)	*ENUMERATED*	Read
max-segments-accepted(167)	*[Unsigned]*	Optional
vt-classes-supported(122)	*ENUMERATED, List*	Optional
active-vt-sessions(5)	*[VTSession], List*	Optional
local-time(57)	*[Time]*	Optional
local-date(56)	*[Date]*	Optional
utc-offset(119)	*[INTEGER]*	Optional
daylight-savings-status(24)	*[BOOLEAN]*	Optional
apdu-segment-timeout(10)	*[Unsigned]*	Optional
apdu-timeout(11)	*[Unsigned]*	Read
number-of-apdu-retries(73)	*[Unsigned]*	Read
time-synchronization-recipients(116)	*[Recipient], List*	Optional
max-master(64)	*[Unsigned]*	Optional
max-info-frames(63)	*[Unsigned]*	Optional
device-address-binding(3	*[AddressBinding], List*	Read

PropertyName	PropertyDatatype	PropertyConformanceCode
0)		
database-revision(155)	*[Unsigned]*	Read
configuration-files(154)	*[ObjectIdentifier], Array[N]*	Optional
last-restore-time(157)	*[TimeStamp]*	Optional
backup-failure-timeout(153)	*UNSIGNED*	Optional
backup-preparation-time(339)	*UNSIGNED*	Optional
restore-preparation-time(341)	*UNSIGNED*	Optional
restore-completion-time(340)	*UNSIGNED*	Optional
backup-and-restore-state(338)	*ENUMERATED*	Optional
active-cov-subscriptions(152)	*[COVSubscription], List*	Optional
slave-proxy-enable(172)	*[BOOLEAN], Array[N]*	Optional
manual-slave-address-binding(170)	*[AddressBinding], List*	Optional
auto-slave-discovery(169)	*[BOOLEAN], Array[N]*	Optional
slave-address-binding(171)	*[AddressBinding], List*	Optional
last-restart-reason(196)	*ENUMERATED*	Optional
time-of-device-restart(203)	*[TimeStamp]*	Optional
restart-notification-recipients(202)	*[Recipient], List*	Optional
utc-time-synchronization-recipients(206)	*[Recipient], List*	Optional
time-synchronization-interval(204)	*[Unsigned]*	Optional

PropertyName	PropertyDatatype	PropertyConformanceCode
align-intervals(193)	*[BOOLEAN]*	Optional
interval-offset(195)	*[Unsigned]*	Optional
serial-number(372)	*[CharacterString]*	Optional
property-list(371)	*ENUMERATED, Array[N]*	Read
profile-name(168)	*[CharacterString]*	Optional

EVENTENROLLMENT(9)

PropertyName	PropertyDatatype	PropertyConformanceCode
object-identifier(75)	*[ObjectIdentifier]*	Read
object-name(77)	*[CharacterString]*	Read
object-type(79)	*ENUMERATED*	Read
description(28)	*[CharacterString]*	Optional
event-type(37)	*ENUMERATED*	Read
notify-type(72)	*ENUMERATED*	Read
event-parameters(83)	*[EventParameter]*	Read
object-property-reference (78)	*[DeviceObjectPropertyReference]*	Read
event-state(36)	*ENUMERATED*	Read
event-enable(35)	*BITSTRING*	Read
acked-transitions(0)	*BITSTRING*	Read
notification-class(17)	*[Unsigned]*	Read
event-time-stamps(130)	*[TimeStamp], Array[3]*	Read
event-message-texts(351)	*[CharacterString], Array[3]*	Optional
event-message-texts-config(352)	*[CharacterString], Array[3]*	Optional
event-detection-enable(35	*[BOOLEAN]*	Read

PropertyName	PropertyDatatype	PropertyConformanceCode
3)		
event-algorithm-inhibit-ref(355)	*[ObjectPropertyReference]*	Optional
event-algorithm-inhibit(354)	*[BOOLEAN]*	Optional
time-delay-normal(356)	*[Unsigned]*	Optional
status-flags(111)	*BITSTRING*	Read
reliability(103)	*ENUMERATED*	Read
fault-type(359)	*ENUMERATED*	Optional
fault-parameters(358)	*[FaultParameter]*	Optional
reliability-evaluation-inhibit(357)	*[BOOLEAN]*	Optional
property-list(371)	*ENUMERATED, Array[N]*	Read
profile-name(168)	*[CharacterString]*	Optional

FILE(10)

PropertyName	PropertyDatatype	PropertyConformanceCode
object-identifier(75)	*[ObjectIdentifier]*	Read
object-name(77)	*[CharacterString]*	Read
object-type(79)	*ENUMERATED*	Read
description(28)	*[CharacterString]*	Optional
file-type(43)	*[CharacterString]*	Read
file-size(42)	*[Unsigned]*	Read
modification-date(71)	*[DateTime]*	Read
archive(13)	*[BOOLEAN]*	Write
read-only(99)	*[BOOLEAN]*	Read
file-access-method(41)	*ENUMERATED*	Read

PropertyName	PropertyDatatype	PropertyConformanceCode
record-count(141)	*[Unsigned]*	Optional
property-list(371)	*ENUMERATED, Array[N]*	Read
profile-name(168)	*[CharacterString]*	Optional

GROUP(11)

PropertyName	PropertyDatatype	PropertyConformanceCode
object-identifier(75)	*[ObjectIdentifier]*	Read
object-name(77)	*[CharacterString]*	Read
object-type(79)	*ENUMERATED*	Read
description(28)	*[CharacterString]*	Optional
list-of-group-members(53)	*[ReadAccessSpecification], List*	Read
present-value(85)	*[ReadAccessResult], List*	Read
property-list(371)	*ENUMERATED, Array[N]*	Read
profile-name(168)	*[CharacterString]*	Optional

LIFESAFETYPOINT(21)

PropertyName	PropertyDatatype	PropertyConformanceCode
object-identifier(75)	*[ObjectIdentifier]*	Read
object-name(77)	*[CharacterString]*	Read
object-type(79)	*ENUMERATED*	Read
present-value(85)	*ENUMERATED*	Read
tracking-value(164)	*ENUMERATED*	Read
description(28)	*[CharacterString]*	Optional
device-type(31)	*[CharacterString]*	Optional
status-flags(111)	*BITSTRING*	Read

PropertyName	PropertyDatatype	PropertyConformanceCode
event-state(36)	*ENUMERATED*	Read
reliability(103)	*ENUMERATED*	Read
out-of-service(81)	*[BOOLEAN]*	Read
mode(160)	*ENUMERATED*	Write
accepted-modes(175)	*ENUMERATED, List*	Read
time-delay(113)	*[Unsigned]*	Optional
notification-class(17)	*[Unsigned]*	Optional
life-safety-alarm-values(166)	*ENUMERATED, List*	Optional
alarm-values(7)	*ENUMERATED, List*	Optional
fault-values(39)	*ENUMERATED, List*	Optional
event-enable(35)	*BITSTRING*	Optional
acked-transitions(0)	*BITSTRING*	Optional
notify-type(72)	*ENUMERATED*	Optional
event-time-stamps(130)	*[TimeStamp], Array[3]*	Optional
event-message-texts(351)	*[CharacterString], Array[3]*	Optional
event-message-texts-config(352)	*[CharacterString], Array[3]*	Optional
event-detection-enable(353)	*[BOOLEAN]*	Optional
event-algorithm-inhibit-ref(355)	*[ObjectPropertyReference]*	Optional
event-algorithm-inhibit(354)	*[BOOLEAN]*	Optional
time-delay-normal(356)	*[Unsigned]*	Optional
reliability-evaluation-inhibit(357)	*[BOOLEAN]*	Optional

PropertyName	PropertyDatatype	PropertyConformanceCode
silenced(163)	*ENUMERATED*	Read
operation-expected(161)	*ENUMERATED*	Read
maintenance-required(158)	*ENUMERATED*	Optional
setting(162)	*UNSIGNED*	Optional
direct-reading(156)	*[REAL]*	Optional
units(117)	*ENUMERATED*	Optional
member-of(159)	*[DeviceObjectReference], List*	Optional
property-list(371)	*ENUMERATED, Array[N]*	Read
profile-name(168)	*[CharacterString]*	Optional

LIFESAFETYZONE(22)

PropertyName	PropertyDatatype	PropertyConformanceCode
object-identifier(75)	*[ObjectIdentifier]*	Read
object-name(77)	*[CharacterString]*	Read
object-type(79)	*ENUMERATED*	Read
present-value(85)	*ENUMERATED*	Read
tracking-value(164)	*ENUMERATED*	Read
description(28)	*[CharacterString]*	Optional
device-type(31)	*[CharacterString]*	Optional
status-flags(111)	*BITSTRING*	Read
event-state(36)	*ENUMERATED*	Read
reliability(103)	*ENUMERATED*	Read
out-of-service(81)	*[BOOLEAN]*	Read
mode(160)	*ENUMERATED*	Write
accepted-modes(175)	*ENUMERATED, List*	Read

PropertyName	PropertyDatatype	PropertyConformanceCode
time-delay(113)	*[Unsigned]*	Optional
notification-class(17)	*[Unsigned]*	Optional
life-safety-alarm-values(166)	*ENUMERATED, List*	Optional
alarm-values(7)	*ENUMERATED, List*	Optional
fault-values(39)	*ENUMERATED, List*	Optional
event-enable(35)	*BITSTRING*	Optional
acked-transitions(0)	*BITSTRING*	Optional
notify-type(72)	*ENUMERATED*	Optional
event-time-stamps(130)	*[TimeStamp], Array[3]*	Optional
event-message-texts(351)	*[CharacterString], Array[3]*	Optional
event-message-texts-config(352)	*[CharacterString], Array[3]*	Optional
event-detection-enable(353)	*[BOOLEAN]*	Optional
event-algorithm-inhibit-ref(355)	*[ObjectPropertyReference]*	Optional
event-algorithm-inhibit(354)	*[BOOLEAN]*	Optional
time-delay-normal(356)	*[Unsigned]*	Optional
reliability-evaluation-inhibit(357)	*[BOOLEAN]*	Optional
silenced(163)	*ENUMERATED*	Read
operation-expected(161)	*ENUMERATED*	Read
maintenance-required(158)	*[BOOLEAN]*	Optional
zone-members(165)	*[DeviceObjectReference], List*	Read
member-of(159)	*[DeviceObjectReference], List*	Optional

PropertyName	PropertyDatatype	PropertyConformanceCode
property-list(371)	*ENUMERATED, Array[N]*	Read
profile-name(168)	*[CharacterString]*	Optional

LOOP(12)

PropertyName	PropertyDatatype	PropertyConformanceCode
object-identifier(75)	*[ObjectIdentifier]*	Read
object-name(77)	*[CharacterString]*	Read
object-type(79)	*ENUMERATED*	Read
present-value(85)	*[REAL]*	Read
description(28)	*[CharacterString]*	Optional
status-flags(111)	*BITSTRING*	Read
event-state(36)	*ENUMERATED*	Read
reliability(103)	*ENUMERATED*	Optional
out-of-service(81)	*[BOOLEAN]*	Read
update-interval(118)	*[Unsigned]*	Optional
output-units(82)	*ENUMERATED*	Read
manipulated-variable-reference(60)	*[ObjectPropertyReference]*	Read
controlled-variable-reference(19)	*[ObjectPropertyReference]*	Read
controlled-variable-value(21)	*[REAL]*	Read
controlled-variable-units(20)	*ENUMERATED*	Read
setpoint-reference(109)	*[SetpointReference]*	Read
setpoint(108)	*[REAL]*	Read
action(2)	*ENUMERATED*	Read

PropertyName	PropertyDatatype	PropertyConformanceCode
proportional-constant(93)	*[REAL]*	Optional
proportional-constant-units(94)	*ENUMERATED*	Optional
integral-constant(49)	*[REAL]*	Optional
integral-constant-units(50)	*ENUMERATED*	Optional
derivative-constant(26)	*[REAL]*	Optional
derivative-constant-units(27)	*ENUMERATED*	Optional
bias(14)	*[REAL]*	Optional
maximum-output(61)	*[REAL]*	Optional
minimum-output(68)	*[REAL]*	Optional
priority-for-writing(88)	*[Unsigned]*	Read
cov-increment(22)	*[REAL]*	Optional
time-delay(113)	*[Unsigned]*	Optional
notification-class(17)	*[Unsigned]*	Optional
error-limit(34)	*[REAL]*	Optional
deadband(25)	*[REAL]*	Optional
event-enable(35)	*BITSTRING*	Optional
acked-transitions(0)	*BITSTRING*	Optional
notify-type(72)	*ENUMERATED*	Optional
event-time-stamps(130)	*[TimeStamp], Array[3]*	Optional
event-message-texts(351)	*[CharacterString], Array[3]*	Optional
event-message-texts-config(352)	*[CharacterString], Array[3]*	Optional
event-detection-enable(353)	*[BOOLEAN]*	Optional

PropertyName	PropertyDatatype	PropertyConformanceCode
event-algorithm-inhibit-ref(355)	*[ObjectPropertyReference]*	Optional
event-algorithm-inhibit(354)	*[BOOLEAN]*	Optional
time-delay-normal(356)	*[Unsigned]*	Optional
reliability-evaluation-inhibit(357)	*[BOOLEAN]*	Optional
property-list(371)	*ENUMERATED, Array[N]*	Read
profile-name(168)	*[CharacterString]*	Optional

MULTISTATEINPUT(13)

PropertyName	PropertyDatatype	PropertyConformanceCode
object-identifier(75)	*[ObjectIdentifier]*	Read
object-name(77)	*[CharacterString]*	Read
object-type(79)	*ENUMERATED*	Read
present-value(85)	*[Unsigned]*	Read
description(28)	*[CharacterString]*	Optional
device-type(31)	*[CharacterString]*	Optional
status-flags(111)	*BITSTRING*	Read
event-state(36)	*ENUMERATED*	Read
reliability(103)	*ENUMERATED*	Optional
out-of-service(81)	*[BOOLEAN]*	Read
number-of-states(74)	*[Unsigned]*	Read
state-text(110)	*[CharacterString], Array[N]*	Optional
time-delay(113)	*[Unsigned]*	Optional
notification-class(17)	*[Unsigned]*	Optional

PropertyName	PropertyDatatype	PropertyConformanceCode
alarm-values(7)	*[Unsigned], List*	Optional
fault-values(39)	*[Unsigned], List*	Optional
event-enable(35)	*BITSTRING*	Optional
acked-transitions(0)	*BITSTRING*	Optional
notify-type(72)	*ENUMERATED*	Optional
event-time-stamps(130)	*[TimeStamp], Array[3]*	Optional
event-message-texts(351)	*[CharacterString], Array[3]*	Optional
event-message-texts-config(352)	*[CharacterString], Array[3]*	Optional
event-detection-enable(353)	*[BOOLEAN]*	Optional
event-algorithm-inhibit-ref(355)	*[ObjectPropertyReference]*	Optional
event-algorithm-inhibit(354)	*[BOOLEAN]*	Optional
time-delay-normal(356)	*[Unsigned]*	Optional
reliability-evaluation-inhibit(357)	*[BOOLEAN]*	Optional
property-list(371)	*ENUMERATED, Array[N]*	Read
profile-name(168)	*[CharacterString]*	Optional

MULTISTATEOUTPUT(14)

PropertyName	PropertyDatatype	PropertyConformanceCode
object-identifier(75)	*[ObjectIdentifier]*	Read
object-name(77)	*[CharacterString]*	Read
object-type(79)	*ENUMERATED*	Read
present-value(85)	*[Unsigned]*	Write

PropertyName	PropertyDatatype	PropertyConformanceCode
description(28)	*[CharacterString]*	Optional
device-type(31)	*[CharacterString]*	Optional
status-flags(111)	*BITSTRING*	Read
event-state(36)	*ENUMERATED*	Read
reliability(103)	*ENUMERATED*	Optional
out-of-service(81)	*[BOOLEAN]*	Read
number-of-states(74)	*[Unsigned]*	Read
state-text(110)	*[CharacterString], Array[N]*	Optional
priority-array(87)	*[PriorityArray]*	Read
relinquish-default(104)	*[Unsigned]*	Read
time-delay(113)	*[Unsigned]*	Optional
notification-class(17)	*[Unsigned]*	Optional
feedback-value(40)	*[Unsigned]*	Optional
event-enable(35)	*BITSTRING*	Optional
acked-transitions(0)	*BITSTRING*	Optional
notify-type(72)	*ENUMERATED*	Optional
event-time-stamps(130)	*[TimeStamp], Array[3]*	Optional
event-message-texts(351)	*[CharacterString], Array[3]*	Optional
event-message-texts-config(352)	*[CharacterString], Array[3]*	Optional
event-detection-enable(353)	*[BOOLEAN]*	Optional
event-algorithm-inhibit-ref(355)	*[ObjectPropertyReference]*	Optional
event-algorithm-inhibit(354)	*[BOOLEAN]*	Optional
time-delay-normal(356)	*[Unsigned]*	Optional

PropertyName	PropertyDatatype	PropertyConformanceCode
reliability-evaluation-inhibit(357)	*[BOOLEAN]*	Optional
property-list(371)	*ENUMERATED, Array[N]*	Read
profile-name(168)	*[CharacterString]*	Optional

MULTISTATEVALUE(19)

PropertyName	PropertyDatatype	PropertyConformanceCode
object-identifier(75)	*[ObjectIdentifier]*	Read
object-name(77)	*[CharacterString]*	Read
object-type(79)	*ENUMERATED*	Read
present-value(85)	*[Unsigned]*	Read
description(28)	*[CharacterString]*	Optional
status-flags(111)	*BITSTRING*	Read
event-state(36)	*ENUMERATED*	Read
reliability(103)	*ENUMERATED*	Optional
out-of-service(81)	*[BOOLEAN]*	Read
number-of-states(74)	*[Unsigned]*	Read
state-text(110)	*[CharacterString], Array[N]*	Optional
priority-array(87)	*[PriorityArray]*	Optional
relinquish-default(104)	*[Unsigned]*	Optional
time-delay(113)	*[Unsigned]*	Optional
notification-class(17)	*[Unsigned]*	Optional
alarm-values(7)	*[Unsigned], List*	Optional
fault-values(39)	*[Unsigned], List*	Optional
event-enable(35)	*BITSTRING*	Optional
acked-transitions(0)	*BITSTRING*	Optional

PropertyName	PropertyDatatype	PropertyConformanceCode
notify-type(72)	*ENUMERATED*	Optional
event-time-stamps(130)	*[TimeStamp], Array[3]*	Optional
event-message-texts(351)	*[CharacterString], Array[3]*	Optional
event-message-texts-config(352)	*[CharacterString], Array[3]*	Optional
event-detection-enable(353)	*[BOOLEAN]*	Optional
event-algorithm-inhibit-ref(355)	*[ObjectPropertyReference]*	Optional
event-algorithm-inhibit(354)	*[BOOLEAN]*	Optional
time-delay-normal(356)	*[Unsigned]*	Optional
reliability-evaluation-inhibit(357)	*[BOOLEAN]*	Optional
property-list(371)	*ENUMERATED, Array[N]*	Read
profile-name(168)	*[CharacterString]*	Optional

NOTIFICATIONCLASS(15)

PropertyName	PropertyDatatype	PropertyConformanceCode
object-identifier(75)	*[ObjectIdentifier]*	Read
object-name(77)	*[CharacterString]*	Read
object-type(79)	*ENUMERATED*	Read
description(28)	*[CharacterString]*	Optional
notification-class(17)	*[Unsigned]*	Read
priority(86)	*[Unsigned], Array[3]*	Read
ack-required(1)	*BITSTRING*	Read
recipient-list(102)	*[Destination], List*	Read

PropertyName	PropertyDatatype	PropertyConformanceCode
property-list(371)	*ENUMERATED, Array[N]*	Read
profile-name(168)	*[CharacterString]*	Optional

PROGRAM(16)

PropertyName	PropertyDatatype	PropertyConformanceCode
OBJECT-IDENTIFIER(75)	*[ObjectIdentifier]*	Read
object-name(77)	*[CharacterString]*	Read
object-type(79)	*ENUMERATED*	Read
program-state(92)	*ENUMERATED*	Read
program-change(90)	*ENUMERATED*	Write
reason-for-halt(100)	*ENUMERATED*	Optional
description-of-halt(29)	*[CharacterString]*	Optional
program-location(91)	*[CharacterString]*	Optional
description(28)	*[CharacterString]*	Optional
instance-of(48)	*[CharacterString]*	Optional
status-flags(111)	*BITSTRING*	Read
reliability(103)	*ENUMERATED*	Optional
out-of-service(81)	*[BOOLEAN]*	Read
event-detection-enable(353)	*[BOOLEAN]*	Optional
notification-class(17)	*[Unsigned]*	Optional
event-enable(35)	*BITSTRING*	Optional
event-state(36)	*ENUMERATED*	Optional
acked-transitions(0)	*BITSTRING*	Optional
notify-type(72)	*ENUMERATED*	Optional

PropertyName	PropertyDatatype	PropertyConformanceCode
event-time-stamps(130)	*[TimeStamp], Array[3]*	Optional
event-message-texts(351)	*[CharacterString], Array[3]*	Optional
event-message-texts-config(352)	*[CharacterString], Array[3]*	Optional
reliability-evaluation-inhibit(357)	*[BOOLEAN]*	Optional
property-list(371)	*ENUMERATED, Array[N]*	Read
profile-name(168)	*[CharacterString]*	Optional

PULSECONVERTER(24)

PropertyName	PropertyDatatype	PropertyConformanceCode
object-identifier(75)	*[ObjectIdentifier]*	Read
object-name(77)	*[CharacterString]*	Read
object-type(79)	*ENUMERATED*	Read
description(28)	*[CharacterString]*	Optional
present-value(85)	*[REAL]*	Read
input-reference(181)	*[ObjectPropertyReference]*	Optional
status-flags(111)	*BITSTRING*	Read
event-state(36)	*ENUMERATED*	Read
reliability(103)	*ENUMERATED*	Optional
out-of-service(81)	*[BOOLEAN]*	Read
units(117)	*ENUMERATED*	Read
scale-factor(188)	*[REAL]*	Read
adjust-value(176)	*[REAL]*	Write
count(177)	*[Unsigned]*	Read
update-time(189)	*[DateTime]*	Read

PropertyName	PropertyDatatype	PropertyConformanceCode
count-change-time(179)	*[DateTime]*	Read
count-before-change(178)	*[Unsigned]*	Read
cov-increment(22)	*[REAL]*	Optional
cov-period(180)	*[Unsigned]*	Optional
notification-class(17)	*[Unsigned]*	Optional
time-delay(113)	*[Unsigned]*	Optional
high-limit(45)	*[REAL]*	Optional
low-limit(59)	*[REAL]*	Optional
deadband(25)	*[REAL]*	Optional
limit-enable(52)	*BITSTRING*	Optional
event-enable(35)	*BITSTRING*	Optional
acked-transitions(0)	*BITSTRING*	Optional
notify-type(72)	*ENUMERATED*	Optional
event-time-stamps(130)	*[TimeStamp], Array[3]*	Optional
event-message-texts(351)	*[CharacterString], Array[3]*	Optional
event-message-texts-config(352)	*[CharacterString], Array[3]*	Optional
event-detection-enable(353)	*[BOOLEAN]*	Optional
event-algorithm-inhibit-ref(355)	*[ObjectPropertyReference]*	Optional
event-algorithm-inhibit(354)	*[BOOLEAN]*	Optional
time-delay-normal(356)	*[Unsigned]*	Optional
reliability-evaluation-inhibit(357)	*[BOOLEAN]*	Optional
property-list(371)	*ENUMERATED, Array[N]*	Read

PropertyName	PropertyDatatype	PropertyConformanceCode
profile-name(168)	*[CharacterString]*	Optional

SCHEDULE(17)

PropertyName	PropertyDatatype	PropertyConformanceCode
object-identifier(75)	*[ObjectIdentifier]*	Read
object-name(77)	*[CharacterString]*	Read
object-type(79)	*ENUMERATED*	Read
present-value(85)	*[Any]*	Read
description(28)	*[CharacterString]*	Optional
effective-period(32)	*[DateRange]*	Read
weekly-schedule(123)	*[DailySchedule], Array[7]*	Optional
exception-schedule(38)	*[SpecialEvent], Array[N]*	Optional
schedule-default(174)	*[Any]*	Read
list-of-object-property-references(54)	*[DeviceObjectPropertyReference], List*	Read
priority-for-writing(88)	*[Unsigned]*	Read
status-flags(111)	*BITSTRING*	Read
reliability(103)	*ENUMERATED*	Read
out-of-service(81)	*[BOOLEAN]*	Read
event-detection-enable(353)	*[BOOLEAN]*	Optional
notification-class(17)	*[Unsigned]*	Optional
event-enable(35)	*BITSTRING*	Optional
event-state(36)	*ENUMERATED*	Optional
acked-transitions(0)	*BITSTRING*	Optional
notify-type(72)	*ENUMERATED*	Optional

PropertyName	PropertyDatatype	PropertyConformanceCode
event-time-stamps(130)	*[TimeStamp], Array[3]*	Optional
event-message-texts(351)	*[CharacterString], Array[3]*	Optional
event-message-texts-config(352)	*[CharacterString], Array[3]*	Optional
reliability-evaluation-inhibit(357)	*[BOOLEAN]*	Optional
property-list(371)	*ENUMERATED, Array[N]*	Read
profile-name(168)	*[CharacterString]*	Optional

TRENDLOG(20)

PropertyName	PropertyDatatype	PropertyConformanceCode
object-identifier(75)	*[ObjectIdentifier]*	Read
object-name(77)	*[CharacterString]*	Read
object-type(79)	*ENUMERATED*	Read
description(28)	*[CharacterString]*	Optional
enable(133)	*[BOOLEAN]*	Write
start-time(142)	*[DateTime]*	Optional
stop-time(143)	*[DateTime]*	Optional
log-device-object-property(132)	*[DeviceObjectPropertyReference]*	Optional
log-interval(134)	*[Unsigned]*	Optional
cov-resubscription-interval(128)	*[Unsigned]*	Optional
client-cov-increment(127)	*[ClientCOV]*	Optional
stop-when-full(144)	*[BOOLEAN]*	Read
buffer-size(126)	*UNSIGNED*	Read
log-buffer(131)	*[LogRecord], List*	Read

PropertyName	PropertyDatatype	PropertyConformanceCode
record-count(141)	*UNSIGNED*	Write
total-record-count(145)	*UNSIGNED*	Read
logging-type(197)	*ENUMERATED*	Read
align-intervals(193)	*[BOOLEAN]*	Optional
interval-offset(195)	*[Unsigned]*	Optional
trigger(205)	*[BOOLEAN]*	Optional
status-flags(111)	*BITSTRING*	Read
reliability(103)	*ENUMERATED*	Optional
notification-threshold(137)	*UNSIGNED*	Optional
records-since-notification(140)	*UNSIGNED*	Optional
last-notify-record(173)	*UNSIGNED*	Optional
event-state(36)	*ENUMERATED*	Read
notification-class(17)	*[Unsigned]*	Optional
event-enable(35)	*BITSTRING*	Optional
acked-transitions(0)	*BITSTRING*	Optional
notify-type(72)	*ENUMERATED*	Optional
event-time-stamps(130)	*[TimeStamp], Array[3]*	Optional
event-message-texts(351)	*[CharacterString], Array[3]*	Optional
event-message-texts-config(352)	*[CharacterString], Array[3]*	Optional
event-detection-enable(353)	*[BOOLEAN]*	Optional
event-algorithm-inhibit-ref(355)	*[ObjectPropertyReference]*	Optional
event-algorithm-inhibit(354)	*[BOOLEAN]*	Optional

PropertyName	PropertyDatatype	PropertyConformanceCode
reliability-evaluation-inhibit(357)	*[BOOLEAN]*	Optional
property-list(371)	*ENUMERATED, Array[N]*	Read
profile-name(168)	*[CharacterString]*	Optional

ACCESSDOOR(30)

PropertyName	PropertyDatatype	PropertyConformanceCode
object-identifier(75)	*[ObjectIdentifier]*	Read
object-name(77)	*[CharacterString]*	Read
object-type(79)	*ENUMERATED*	Read
present-value(85)	*ENUMERATED*	Write
description(28)	*[CharacterString]*	Optional
status-flags(111)	*BITSTRING*	Read
event-state(36)	*ENUMERATED*	Read
reliability(103)	*ENUMERATED*	Read
out-of-service(81)	*[BOOLEAN]*	Read
priority-array(87)	*[PriorityArray]*	Read
relinquish-default(104)	*ENUMERATED*	Read
door-status(231)	*ENUMERATED*	Optional
lock-status(233)	*ENUMERATED*	Optional
secured-status(235)	*ENUMERATED*	Optional
door-members(228)	*[DeviceObjectReference], Array[N]*	Optional
door-pulse-time(230)	*[Unsigned]*	Read
door-extended-pulse-time(227)	*[Unsigned]*	Read
door-unlock-delay-time(2	*[Unsigned]*	Optional

PropertyName	PropertyDatatype	PropertyConformanceCode
32)		
door-open-too-long-time(229)	*[Unsigned]*	Read
door-alarm-state(226)	*ENUMERATED*	Optional
masked-alarm-values(234)	*ENUMERATED, List*	Optional
maintenance-required(158)	*ENUMERATED*	Optional
time-delay(113)	*[Unsigned]*	Optional
notification-class(17)	*[Unsigned]*	Optional
alarm-values(7)	*ENUMERATED, List*	Optional
fault-values(39)	*ENUMERATED, List*	Optional
event-enable(35)	*BITSTRING*	Optional
acked-transitions(0)	*BITSTRING*	Optional
notify-type(72)	*ENUMERATED*	Optional
event-time-stamps(130)	*[TimeStamp], Array[3]*	Optional
event-message-texts(351)	*[CharacterString], Array[3]*	Optional
event-message-texts-config(352)	*[CharacterString], Array[3]*	Optional
event-detection-enable(353)	*[BOOLEAN]*	Optional
event-algorithm-inhibit-ref(355)	*[ObjectPropertyReference]*	Optional
event-algorithm-inhibit(354)	*[BOOLEAN]*	Optional
time-delay-normal(356)	*[Unsigned]*	Optional

EVENTLOG(25)

PropertyName	PropertyDatatype	PropertyConformanceCode
object-identifier(75)	*[ObjectIdentifier]*	Read
object-name(77)	*[CharacterString]*	Read
object-type(79)	*ENUMERATED*	Read
property-list(371)	*ENUMERATED, Array[N]*	Read
description(28)	*[CharacterString]*	Optional
status-flags(111)	*BITSTRING*	Read
event-state(36)	*ENUMERATED*	Read
reliability(103)	*ENUMERATED*	Optional
enable(133)	*[BOOLEAN]*	Write
start-time(142)	*[DateTime]*	Optional
stop-time(143)	*[DateTime]*	Optional
stop-when-full(144)	*[BOOLEAN]*	Read
buffer-size(126)	*UNSIGNED*	Read
log-buffer(131)	*[EventLogRecord], List*	Read
record-count(141)	*UNSIGNED*	Write
total-record-count(145)	*UNSIGNED*	Read
notification-threshold(137)	*UNSIGNED*	Optional
records-since-notification(140)	*UNSIGNED*	Optional
last-notify-record(173)	*UNSIGNED*	Optional
notification-class(17)	*[Unsigned]*	Optional
event-enable(35)	*BITSTRING*	Optional
acked-transitions(0)	*BITSTRING*	Optional
notify-type(72)	*ENUMERATED*	Optional
event-time-stamps(130)	*[TimeStamp], Array[3]*	Optional

PropertyName	PropertyDatatype	PropertyConformanceCode
event-message-texts(351)	*[CharacterString], Array[3]*	Optional
event-message-texts-config(352)	*[CharacterString], Array[3]*	Optional
event-detection-enable(353)	*[BOOLEAN]*	Optional
event-algorithm-inhibit-ref(355)	*[ObjectPropertyReference]*	Optional
event-algorithm-inhibit(354)	*[BOOLEAN]*	Optional
reliability-evaluation-inhibit(357)	*[BOOLEAN]*	Optional
profile-name(168)	*[CharacterString]*	Optional

LOADCONTROL(28)

PropertyName	PropertyDatatype	PropertyConformanceCode
object-identifier(75)	*[ObjectIdentifier]*	Read
object-name(77)	*[CharacterString]*	Read
object-type(79)	*ENUMERATED*	Read
description(28)	*[CharacterString]*	Optional
present-value(85)	*ENUMERATED*	Read
state-description(222)	*[CharacterString]*	Optional
status-flags(111)	*BITSTRING*	Read
event-state(36)	*ENUMERATED*	Read
reliability(103)	*ENUMERATED*	Optional
requested-shed-level(218)	*[ShedLevel]*	Write
start-time(142)	*[DateTime]*	Write
shed-duration(219)	*[Unsigned]*	Write

PropertyName	PropertyDatatype	PropertyConformanceCode
duty-window(213)	*[Unsigned]*	Write
enable(133)	*[BOOLEAN]*	Write
full-duty-baseline(215)	*[REAL]*	Optional
expected-shed-level(214)	*[ShedLevel]*	Read
actual-shed-level(212)	*[ShedLevel]*	Read
shed-levels(221)	*[Unsigned], Array[N]*	Write
shed-level-descriptions(220)	*[CharacterString], Array[N]*	Read
notification-class(17)	*[Unsigned]*	Optional
time-delay(113)	*[Unsigned]*	Optional
event-enable(35)	*BITSTRING*	Optional
acked-transitions(0)	*BITSTRING*	Optional
notify-type(72)	*ENUMERATED*	Optional
event-time-stamps(130)	*[TimeStamp], Array[3]*	Optional
event-message-texts(351)	*[CharacterString], Array[3]*	Optional
event-message-texts-config(352)	*[CharacterString], Array[3]*	Optional
event-detection-enable(353)	*[BOOLEAN]*	Optional
event-algorithm-inhibit-ref(355)	*[ObjectPropertyReference]*	Optional
event-algorithm-inhibit(354)	*[BOOLEAN]*	Optional
time-delay-normal(356)	*[Unsigned]*	Optional
reliability-evaluation-inhibit(357)	*[BOOLEAN]*	Optional
property-list(371)	*ENUMERATED, Array[N]*	Read

PropertyName	PropertyDatatype	PropertyConformanceCode
profile-name(168)	*[CharacterString]*	Optional

STRUCTUREDVIEW(29)

PropertyName	PropertyDatatype	PropertyConformanceCode
object-identifier(75)	*[ObjectIdentifier]*	Read
object-name(77)	*[CharacterString]*	Read
object-type(79)	*ENUMERATED*	Read
description(28)	*[CharacterString]*	Optional
node-type(208)	*ENUMERATED*	Read
node-subtype(207)	*[CharacterString]*	Optional
subordinate-list(211)	*[DeviceObjectReference], Array[N]*	Read
subordinate-annotations(210)	*[CharacterString], Array[N]*	Optional
property-list(371)	*ENUMERATED, Array[N]*	Read
profile-name(168)	*[CharacterString]*	Optional

TRENDLOGMULTIPLE(27)

PropertyName	PropertyDatatype	PropertyConformanceCode
object-identifier(75)	*[ObjectIdentifier]*	Read
object-name(77)	*[CharacterString]*	Read
object-type(79)	*ENUMERATED*	Read
description(28)	*[CharacterString]*	Optional
status-flags(111)	*BITSTRING*	Read
event-state(36)	*ENUMERATED*	Read
reliability(103)	*ENUMERATED*	Optional
enable(133)	*[BOOLEAN]*	Write

PropertyName	PropertyDatatype	PropertyConformanceCode
start-time(142)	*[DateTime]*	Optional
stop-time(143)	*[DateTime]*	Optional
log-device-object-property(132)	*[DeviceObjectPropertyReference], Array[N]*	Read
logging-type(197)	*ENUMERATED*	Read
log-interval(134)	*[Unsigned]*	Read
align-intervals(193)	*[BOOLEAN]*	Optional
interval-offset(195)	*[Unsigned]*	Optional
trigger(205)	*[BOOLEAN]*	Optional
stop-when-full(144)	*[BOOLEAN]*	Read
buffer-size(126)	*UNSIGNED*	Read
log-buffer(131)	*[LogMultipleRecord], List*	Read
record-count(141)	*UNSIGNED*	Write
total-record-count(145)	*UNSIGNED*	Read
notification-threshold(137)	*UNSIGNED*	Optional
records-since-notification(140)	*UNSIGNED*	Optional
last-notify-record(173)	*UNSIGNED*	Optional
notification-class(17)	*[Unsigned]*	Optional
event-enable(35)	*BITSTRING*	Optional
acked-transitions(0)	*BITSTRING*	Optional
notify-type(72)	*ENUMERATED*	Optional
event-time-stamps(130)	*[TimeStamp], Array[3]*	Optional
event-message-texts(351)	*[CharacterString], Array[3]*	Optional
event-message-texts-config(352)	*[CharacterString], Array[3]*	Optional

PropertyName	PropertyDatatype	PropertyConformanceCode
event-detection-enable(353)	*[BOOLEAN]*	Optional
event-algorithm-inhibit-ref(355)	*[ObjectPropertyReference]*	Optional
event-algorithm-inhibit(354)	*[BOOLEAN]*	Optional
reliability-evaluation-inhibit(357)	*[BOOLEAN]*	Optional
property-list(371)	*ENUMERATED, Array[N]*	Read
profile-name(168)	*[CharacterString]*	Optional

ACCESSPOINT(33)

PropertyName	PropertyDatatype	PropertyConformanceCode
object-identifier(75)	*[ObjectIdentifier]*	Read
object-name(77)	*[CharacterString]*	Read
object-type(79)	*ENUMERATED*	Read
description(28)	*[CharacterString]*	Optional
status-flags(111)	*BITSTRING*	Read
event-state(36)	*ENUMERATED*	Read
reliability(103)	*ENUMERATED*	Read
out-of-service(81)	*[BOOLEAN]*	Read
authentication-status(260)	*ENUMERATED*	Read
active-authentication-policy(255)	*[Unsigned]*	Read
number-of-authentication-policies(289)	*[Unsigned]*	Read
authentication-policy-list(258)	*[AuthenticationPolicy], Array[N]*	Optional

PropertyName	PropertyDatatype	PropertyConformanceCode
authentication-policy-names(259)	*[CharacterString], Array[N]*	Optional
authorization-mode(261)	*ENUMERATED*	Read
verification-time(326)	*[Unsigned]*	Optional
lockout(282)	*[BOOLEAN]*	Optional
lockout-relinquish-time(283)	*[Unsigned]*	Optional
failed-attempts(273)	*[Unsigned]*	Optional
failed-attempt-events(272)	*ENUMERATED, List*	Optional
max-failed-attempts(285)	*[Unsigned]*	Optional
failed-attempts-time(274)	*[Unsigned]*	Optional
threat-level(307)	*UNSIGNED*	Optional
occupancy-upper-limit-enforced(298)	*[BOOLEAN]*	Optional
occupancy-lower-limit-enforced(295)	*[BOOLEAN]*	Optional
occupancy-count-adjust(291)	*[BOOLEAN]*	Optional
accompaniment-time(253)	*[Unsigned]*	Optional
access-event(247)	*ENUMERATED*	Read
access-event-tag(322)	*[Unsigned]*	Read
access-event-time(250)	*[TimeStamp]*	Read
access-event-credential(249)	*[DeviceObjectReference]*	Read
access-event-authentication-factor(248)	*[AuthenticationFactor]*	Optional
access-doors(246)	*[DeviceObjectReference], Array[N]*	Read

PropertyName	PropertyDatatype	PropertyConformanceCode
priority-for-writing(88)	*[Unsigned]*	Read
muster-point(287)	*[BOOLEAN]*	Optional
zone-to(321)	*[DeviceObjectReference]*	Optional
zone-from(320)	*[DeviceObjectReference]*	Optional
notification-class(17)	*[Unsigned]*	Optional
transaction-notification-class(309)	*[Unsigned]*	Optional
access-alarm-events(245)	*ENUMERATED, List*	Optional
access-transaction-events(251)	*ENUMERATED, List*	Optional
event-enable(35)	*BITSTRING*	Optional
acked-transitions(0)	*BITSTRING*	Optional
notify-type(72)	*ENUMERATED*	Optional
event-time-stamps(130)	*[TimeStamp], Array[3]*	Optional
event-message-texts(351)	*[CharacterString], Array[3]*	Optional
event-message-texts-config(352)	*[CharacterString], Array[3]*	Optional
event-detection-enable(353)	*[BOOLEAN]*	Optional
event-algorithm-inhibit-ref(355)	*[ObjectPropertyReference]*	Optional
event-algorithm-inhibit(354)	*[BOOLEAN]*	Optional
reliability-evaluation-inhibit(357)	*[BOOLEAN]*	Optional
property-list(371)	*ENUMERATED, Array[N]*	Read
profile-name(168)	*[CharacterString]*	Optional

ACCESSZONE(36)

PropertyName	PropertyDatatype	PropertyConformanceCode
object-identifier(75)	*[ObjectIdentifier]*	Read
object-name(77)	*[CharacterString]*	Read
object-type(79)	*ENUMERATED*	Read
description(28)	*[CharacterString]*	Optional
global-identifier(323)	*UNSIGNED*	Write
occupancy-state(296)	*ENUMERATED*	Read
status-flags(111)	*BITSTRING*	Read
event-state(36)	*ENUMERATED*	Read
reliability(103)	*ENUMERATED*	Read
out-of-service(81)	*[BOOLEAN]*	Read
occupancy-count(290)	*[Unsigned]*	Optional
occupancy-count-enable(292)	*[BOOLEAN]*	Optional
adjust-value(176)	*[INTEGER]*	Optional
occupancy-upper-limit(297)	*[Unsigned]*	Optional
occupancy-lower-limit(294)	*[Unsigned]*	Optional
credentials-in-zone(266)	*[DeviceObjectReference], List*	Optional
last-credential-added(277)	*[DeviceObjectReference]*	Optional
last-credential-added-time(278)	*[DateTime]*	Optional
last-credential-removed(279)	*[DeviceObjectReference]*	Optional
last-credential-removed-time(280)	*[DateTime]*	Optional
passback-mode(300)	*ENUMERATED*	Optional

PropertyName	PropertyDatatype	PropertyConformanceCode
passback-timeout(301)	*[Unsigned]*	Optional
entry-points(268)	*[DeviceObjectReference], List*	Read
exit-points(269)	*[DeviceObjectReference], List*	Read
time-delay(113)	*[Unsigned]*	Optional
notification-class(17)	*[Unsigned]*	Optional
alarm-values(7)	*ENUMERATED, List*	Optional
event-enable(35)	*BITSTRING*	Optional
acked-transitions(0)	*BITSTRING*	Optional
notify-type(72)	*ENUMERATED*	Optional
event-time-stamps(130)	*[TimeStamp], Array[3]*	Optional
event-message-texts(351)	*[CharacterString], Array[3]*	Optional
event-message-texts-config(352)	*[CharacterString], Array[3]*	Optional
event-detection-enable(353)	*[BOOLEAN]*	Optional
event-algorithm-inhibit-ref(355)	*[ObjectPropertyReference]*	Optional
event-algorithm-inhibit(354)	*[BOOLEAN]*	Optional
time-delay-normal(356)	*[Unsigned]*	Optional
reliability-evaluation-inhibit(357)	*[BOOLEAN]*	Optional
property-list(371)	*ENUMERATED, Array[N]*	Read
profile-name(168)	*[CharacterString]*	Optional

ACCESSUSER(35)

PropertyName	PropertyDatatype	PropertyConformanceCode
object-identifier(75)	*[ObjectIdentifier]*	Read
object-name(77)	*[CharacterString]*	Read
object-type(79)	*ENUMERATED*	Read
description(28)	*[CharacterString]*	Optional
global-identifier(323)	*UNSIGNED*	Write
status-flags(111)	*BITSTRING*	Read
reliability(103)	*ENUMERATED*	Read
user-type(318)	*ENUMERATED*	Read
user-name(317)	*[CharacterString]*	Optional
user-external-identifier(310)	*[CharacterString]*	Optional
user-information-reference(311)	*[CharacterString]*	Optional
members(286)	*[DeviceObjectReference], List*	Optional
member-of(159)	*[DeviceObjectReference], List*	Optional
credentials(265)	*[DeviceObjectReference], List*	Read
reliability-evaluation-inhibit(357)	*[BOOLEAN]*	Optional
property-list(371)	*ENUMERATED, Array[N]*	Read
profile-name(168)	*[CharacterString]*	Optional

ACCESSRIGHTS(34)

PropertyName	PropertyDatatype	PropertyConformanceCode
object-identifier(75)	*[ObjectIdentifier]*	Read
object-name(77)	*[CharacterString]*	Read
object-type(79)	*ENUMERATED*	Read

PropertyName	PropertyDatatype	PropertyConformanceCode
description(28)	*[CharacterString]*	Optional
global-identifier(323)	*UNSIGNED*	Write
status-flags(111)	*BITSTRING*	Read
reliability(103)	*ENUMERATED*	Read
enable(133)	*[BOOLEAN]*	Read
negative-access-rules(288)	*[AccessRule], Array[N]*	Read
positive-access-rules(302)	*[AccessRule], Array[N]*	Read
accompaniment(252)	*[DeviceObjectReference]*	Optional
reliability-evaluation-inhibit(357)	*[BOOLEAN]*	Optional
property-list(371)	*ENUMERATED, Array[N]*	Read
profile-name(168)	*[CharacterString]*	Optional

ACCESSCREDENTIAL(32)

PropertyName	PropertyDatatype	PropertyConformanceCode
object-identifier(75)	*[ObjectIdentifier]*	Read
object-name(77)	*[CharacterString]*	Read
object-type(79)	*ENUMERATED*	Read
description(28)	*[CharacterString]*	Optional
global-identifier(323)	*UNSIGNED*	Write
status-flags(111)	*BITSTRING*	Read
reliability(103)	*ENUMERATED*	Read
credential-status(264)	*ENUMERATED*	Read
reason-for-disable(303)	*ENUMERATED, List*	Read
authentication-factors(257)	*[CredentialAuthenticationFactor], Array[N]*	Read

PropertyName	PropertyDatatype	PropertyConformanceCode
activation-time(254)	*[DateTime]*	Read
expiry-time(270)	*[DateTime]*	Read
credential-disable(263)	*ENUMERATED*	Read
days-remaining(267)	*[INTEGER]*	Optional
uses-remaining(319)	*[INTEGER]*	Optional
absentee-limit(244)	*[Unsigned]*	Optional
belongs-to(262)	*[DeviceObjectReference]*	Optional
assigned-access-rights(256)	*[AssignedAccessRights], Array[N]*	Read
last-access-point(276)	*[DeviceObjectReference]*	Optional
last-access-event(275)	*ENUMERATED*	Optional
last-use-time(281)	*[DateTime]*	Optional
trace-flag(308)	*[BOOLEAN]*	Optional
threat-authority(306)	*UNSIGNED*	Optional
extended-time-enable(271)	*[BOOLEAN]*	Optional
authorization-exemptions(364)	*ENUMERATED, List*	Optional
reliability-evaluation-inhibit(357)	*[BOOLEAN]*	Optional
property-list(371)	*ENUMERATED, Array[N]*	Read
profile-name(168)	*[CharacterString]*	Optional

CREDENTIALDATAINPUT(37)

PropertyName	PropertyDatatype	PropertyConformanceCode
object-identifier(75)	*[ObjectIdentifier]*	Read
object-name(77)	*[CharacterString]*	Read

PropertyName	PropertyDatatype	PropertyConformanceCode
object-type(79)	*ENUMERATED*	Read
present-value(85)	*[AuthenticationFactor]*	Read
description(28)	*[CharacterString]*	Optional
status-flags(111)	*BITSTRING*	Read
reliability(103)	*ENUMERATED*	Read
out-of-service(81)	*[BOOLEAN]*	Read
supported-formats(304)	*[AuthenticationFactorFormat], Array[N]*	Read
supported-format-classes(305)	*[Unsigned], Array[N]*	Optional
update-time(189)	*[TimeStamp]*	Read
event-detection-enable(353)	*[BOOLEAN]*	Optional
notification-class(17)	*[Unsigned]*	Optional
event-enable(35)	*BITSTRING*	Optional
event-state(36)	*ENUMERATED*	Optional
acked-transitions(0)	*BITSTRING*	Optional
notify-type(72)	*ENUMERATED*	Optional
event-time-stamps(130)	*[TimeStamp], Array[3]*	Optional
event-message-texts(351)	*[CharacterString], Array[3]*	Optional
event-message-texts-config(352)	*[CharacterString], Array[3]*	Optional
reliability-evaluation-inhibit(357)	*[BOOLEAN]*	Optional
property-list(371)	*ENUMERATED, Array[N]*	Read
profile-name(168)	*[CharacterString]*	Optional

CHARACTERSTRINGVALUE(40)

PropertyName	PropertyDatatype	PropertyConformanceCode
object-identifier(75)	*[ObjectIdentifier]*	Read
object-name(77)	*[CharacterString]*	Read
object-type(79)	*ENUMERATED*	Read
description(28)	*[CharacterString]*	Optional
present-value(85)	*[CharacterString]*	Read
status-flags(111)	*BITSTRING*	Read
event-state(36)	*ENUMERATED*	Optional
reliability(103)	*ENUMERATED*	Optional
out-of-service(81)	*[BOOLEAN]*	Optional
priority-array(87)	*[PriorityArray]*	Optional
relinquish-default(104)	*[CharacterString]*	Optional
time-delay(113)	*[Unsigned]*	Optional
notification-class(17)	*[Unsigned]*	Optional
alarm-values(7)	*[OptionalCharacterString], Array[N]*	Optional
fault-values(39)	*[OptionalCharacterString], Array[N]*	Optional
event-enable(35)	*BITSTRING*	Optional
acked-transitions(0)	*BITSTRING*	Optional
notify-type(72)	*ENUMERATED*	Optional
event-time-stamps(130)	*[TimeStamp], Array[3]*	Optional
event-message-texts(351)	*[CharacterString], Array[3]*	Optional
event-message-texts-config(352)	*[CharacterString], Array[3]*	Optional
event-detection-enable(353)	*[BOOLEAN]*	Optional
event-algorithm-inhibit-ref(355)	*[ObjectPropertyReference]*	Optional

PropertyName	PropertyDatatype	PropertyConformanceCode
event-algorithm-inhibit(354)	*[BOOLEAN]*	Optional
time-delay-normal(356)	*[Unsigned]*	Optional
reliability-evaluation-inhibit(357)	*[BOOLEAN]*	Optional
property-list(371)	*ENUMERATED, Array[N]*	Read
profile-name(168)	*[CharacterString]*	Optional

DATETIMEVALUE(44)

PropertyName	PropertyDatatype	PropertyConformanceCode
object-identifier(75)	*[ObjectIdentifier]*	Read
object-name(77)	*[CharacterString]*	Read
object-type(79)	*ENUMERATED*	Read
description(28)	*[CharacterString]*	Optional
present-value(85)	*[DateTime]*	Read
status-flags(111)	*BITSTRING*	Read
event-state(36)	*ENUMERATED*	Optional
reliability(103)	*ENUMERATED*	Optional
out-of-service(81)	*[BOOLEAN]*	Optional
priority-array(87)	*[PriorityArray]*	Optional
relinquish-default(104)	*[DateTime]*	Optional
is-utc(344)	*[BOOLEAN]*	Optional
reliability-evaluation-inhibit(357)	*[BOOLEAN]*	Optional
property-list(371)	*ENUMERATED, Array[N]*	Read
profile-name(168)	*[CharacterString]*	Optional

LARGEANALOGVALUE(46)

PropertyName	PropertyDatatype	PropertyConformanceCode
object-identifier(75)	*[ObjectIdentifier]*	Read
object-name(77)	*[CharacterString]*	Read
object-type(79)	*ENUMERATED*	Read
description(28)	*[CharacterString]*	Optional
present-value(85)	*[Double]*	Read
status-flags(111)	*BITSTRING*	Read
event-state(36)	*ENUMERATED*	Optional
reliability(103)	*ENUMERATED*	Optional
out-of-service(81)	*[BOOLEAN]*	Optional
units(117)	*ENUMERATED*	Read
priority-array(87)	*[PriorityArray]*	Optional
relinquish-default(104)	*[Double]*	Optional
cov-increment(22)	*[Double]*	Optional
time-delay(113)	*[Unsigned]*	Optional
notification-class(17)	*[Unsigned]*	Optional
high-limit(45)	*[Double]*	Optional
low-limit(59)	*[Double]*	Optional
deadband(25)	*[Double]*	Optional
limit-enable(52)	*BITSTRING*	Optional
event-enable(35)	*BITSTRING*	Optional
acked-transitions(0)	*BITSTRING*	Optional
notify-type(72)	*ENUMERATED*	Optional
event-time-stamps(130)	*[TimeStamp], Array[3]*	Optional
event-message-texts(351)	*[CharacterString], Array[3]*	Optional

PropertyName	PropertyDatatype	PropertyConformanceCode
event-message-texts-config(352)	*[CharacterString], Array[3]*	Optional
event-detection-enable(353)	*[BOOLEAN]*	Optional
event-algorithm-inhibit-ref(355)	*[ObjectPropertyReference]*	Optional
event-algorithm-inhibit(354)	*[BOOLEAN]*	Optional
time-delay-normal(356)	*[Unsigned]*	Optional
reliability-evaluation-inhibit(357)	*[BOOLEAN]*	Optional
min-pres-value(69)	*[Double]*	Optional
max-pres-value(65)	*[Double]*	Optional
resolution(106)	*[Double]*	Optional
property-list(371)	*ENUMERATED, Array[N]*	Read
profile-name(168)	*[CharacterString]*	Optional

BITSTRINGVALUE(39)

PropertyName	PropertyDatatype	PropertyConformanceCode
object-identifier(75)	*[ObjectIdentifier]*	Read
object-name(77)	*[CharacterString]*	Read
object-type(79)	*ENUMERATED*	Read
description(28)	*[CharacterString]*	Optional
present-value(85)	*[BITSTRING]*	Read
bit-text(343)	*[CharacterString], Array[N]*	Optional
status-flags(111)	*BITSTRING*	Read
event-state(36)	*ENUMERATED*	Optional

PropertyName	PropertyDatatype	PropertyConformanceCode
reliability(103)	*ENUMERATED*	Optional
out-of-service(81)	*[BOOLEAN]*	Optional
priority-array(87)	*[PriorityArray]*	Optional
relinquish-default(104)	*[BITSTRING]*	Optional
time-delay(113)	*[Unsigned]*	Optional
notification-class(17)	*[Unsigned]*	Optional
alarm-values(7)	*[BITSTRING], Array[N]*	Optional
bit-mask(342)	*[BITSTRING]*	Optional
event-enable(35)	*BITSTRING*	Optional
acked-transitions(0)	*BITSTRING*	Optional
notify-type(72)	*ENUMERATED*	Optional
event-time-stamps(130)	*[TimeStamp], Array[3]*	Optional
event-message-texts(351)	*[CharacterString], Array[3]*	Optional
event-message-texts-config(352)	*[CharacterString], Array[3]*	Optional
event-detection-enable(353)	*[BOOLEAN]*	Optional
event-algorithm-inhibit-ref(355)	*[ObjectPropertyReference]*	Optional
event-algorithm-inhibit(354)	*[BOOLEAN]*	Optional
time-delay-normal(356)	*[Unsigned]*	Optional
reliability-evaluation-inhibit(357)	*[BOOLEAN]*	Optional
property-list(371)	*ENUMERATED, Array[N]*	Read
profile-name(168)	*[CharacterString]*	Optional

OCTETSTRINGVALUE(47)

PropertyName	PropertyDatatype	PropertyConformanceCode
object-identifier(75)	*[ObjectIdentifier]*	Read
object-name(77)	*[CharacterString]*	Read
object-type(79)	*ENUMERATED*	Read
description(28)	*[CharacterString]*	Optional
present-value(85)	*[OCTETSTRING]*	Read
status-flags(111)	*BITSTRING*	Read
event-state(36)	*ENUMERATED*	Optional
reliability(103)	*ENUMERATED*	Optional
out-of-service(81)	*[BOOLEAN]*	Optional
priority-array(87)	*[PriorityArray]*	Optional
relinquish-default(104)	*[OCTETSTRING]*	Optional
reliability-evaluation-inhibit(357)	*[BOOLEAN]*	Optional
property-list(371)	*ENUMERATED, Array[N]*	Read
profile-name(168)	*[CharacterString]*	Optional

TIMEVALUE(50)

PropertyName	PropertyDatatype	PropertyConformanceCode
object-identifier(75)	*[ObjectIdentifier]*	Read
object-name(77)	*[CharacterString]*	Read
object-type(79)	*ENUMERATED*	Read
description(28)	*[CharacterString]*	Optional
present-value(85)	*[Time]*	Read
status-flags(111)	*BITSTRING*	Read
event-state(36)	*ENUMERATED*	Optional

PropertyName	PropertyDatatype	PropertyConformanceCode
reliability(103)	*ENUMERATED*	Optional
out-of-service(81)	*[BOOLEAN]*	Optional
priority-array(87)	*[PriorityArray]*	Optional
relinquish-default(104)	*[Time]*	Optional
reliability-evaluation-inhibit(357)	*[BOOLEAN]*	Optional
property-list(371)	*ENUMERATED, Array[N]*	Read
profile-name(168)	*[CharacterString]*	Optional
INTEGERVALUE(45)		
object-identifier(75)	*[ObjectIdentifier]*	Read
object-name(77)	*[CharacterString]*	Read
object-type(79)	*ENUMERATED*	Read
description(28)	*[CharacterString]*	Optional
present-value(85)	*[INTEGER]*	Read
status-flags(111)	*BITSTRING*	Read
event-state(36)	*ENUMERATED*	Optional
reliability(103)	*ENUMERATED*	Optional
out-of-service(81)	*[BOOLEAN]*	Optional
units(117)	*ENUMERATED*	Read
priority-array(87)	*[PriorityArray]*	Optional
relinquish-default(104)	*[INTEGER]*	Optional
cov-increment(22)	*[Unsigned]*	Optional
time-delay(113)	*[Unsigned]*	Optional
notification-class(17)	*[Unsigned]*	Optional
high-limit(45)	*[INTEGER]*	Optional
low-limit(59)	*[INTEGER]*	Optional

PropertyName	PropertyDatatype	PropertyConformanceCode
deadband(25)	*[Unsigned]*	Optional
limit-enable(52)	*BITSTRING*	Optional
event-enable(35)	*BITSTRING*	Optional
acked-transitions(0)	*BITSTRING*	Optional
notify-type(72)	*ENUMERATED*	Optional
event-time-stamps(130)	*[TimeStamp], Array[3]*	Optional
event-message-texts(351)	*[CharacterString], Array[3]*	Optional
event-message-texts-config(352)	*[CharacterString], Array[3]*	Optional
event-detection-enable(353)	*[BOOLEAN]*	Optional
event-algorithm-inhibit-ref(355)	*[ObjectPropertyReference]*	Optional
event-algorithm-inhibit(354)	*[BOOLEAN]*	Optional
time-delay-normal(356)	*[Unsigned]*	Optional
reliability-evaluation-inhibit(357)	*[BOOLEAN]*	Optional
min-pres-value(69)	*[INTEGER]*	Optional
max-pres-value(65)	*[INTEGER]*	Optional
resolution(106)	*[INTEGER]*	Optional
property-list(371)	*ENUMERATED, Array[N]*	Read
profile-name(168)	*[CharacterString]*	Optional

POSITIVEINTEGERVALUE(48)

PropertyName	PropertyDatatype	PropertyConformanceCode
object-identifier(75)	*[ObjectIdentifier]*	Read

PropertyName	PropertyDatatype	PropertyConformanceCode
object-name(77)	*[CharacterString]*	Read
object-type(79)	*ENUMERATED*	Read
description(28)	*[CharacterString]*	Optional
present-value(85)	*[Unsigned]*	Read
status-flags(111)	*BITSTRING*	Read
event-state(36)	*ENUMERATED*	Optional
reliability(103)	*ENUMERATED*	Optional
out-of-service(81)	*[BOOLEAN]*	Optional
units(117)	*ENUMERATED*	Read
priority-array(87)	*[PriorityArray]*	Optional
relinquish-default(104)	*[Unsigned]*	Optional
cov-increment(22)	*[Unsigned]*	Optional
time-delay(113)	*[Unsigned]*	Optional
notification-class(17)	*[Unsigned]*	Optional
high-limit(45)	*[Unsigned]*	Optional
low-limit(59)	*[Unsigned]*	Optional
deadband(25)	*[Unsigned]*	Optional
limit-enable(52)	*BITSTRING*	Optional
event-enable(35)	*BITSTRING*	Optional
acked-transitions(0)	*BITSTRING*	Optional
notify-type(72)	*ENUMERATED*	Optional
event-time-stamps(130)	*[TimeStamp], Array[3]*	Optional
event-message-texts(351)	*[CharacterString], Array[3]*	Optional
event-message-texts-config(352)	*[CharacterString], Array[3]*	Optional
event-detection-enable(35	*[BOOLEAN]*	Optional

PropertyName	PropertyDatatype	PropertyConformanceCode
3)		
event-algorithm-inhibit-ref(355)	*[ObjectPropertyReference]*	Optional
event-algorithm-inhibit(354)	*[BOOLEAN]*	Optional
time-delay-normal(356)	*[Unsigned]*	Optional
reliability-evaluation-inhibit(357)	*[BOOLEAN]*	Optional
min-pres-value(69)	*[Unsigned]*	Optional
max-pres-value(65)	*[Unsigned]*	Optional
resolution(106)	*[Unsigned]*	Optional
property-list(371)	*ENUMERATED, Array[N]*	Read
profile-name(168)	*[CharacterString]*	Optional

DATEVALUE(42)

PropertyName	PropertyDatatype	PropertyConformanceCode
object-identifier(75)	*[ObjectIdentifier]*	Read
object-name(77)	*[CharacterString]*	Read
object-type(79)	*ENUMERATED*	Read
description(28)	*[CharacterString]*	Optional
present-value(85)	*[Date]*	Read
status-flags(111)	*BITSTRING*	Read
event-state(36)	*ENUMERATED*	Optional
reliability(103)	*ENUMERATED*	Optional
out-of-service(81)	*[BOOLEAN]*	Optional
priority-array(87)	*[PriorityArray]*	Optional
relinquish-default(104)	*[Date]*	Optional

PropertyName	PropertyDatatype	PropertyConformanceCode
reliability-evaluation-inhibit(357)	*[BOOLEAN]*	Optional
property-list(371)	*ENUMERATED, Array[N]*	Read
profile-name(168)	*[CharacterString]*	Optional

DATETIMEPATTERNVALUE(43)

PropertyName	PropertyDatatype	PropertyConformanceCode
object-identifier(75)	*[ObjectIdentifier]*	Read
object-name(77)	*[CharacterString]*	Read
object-type(79)	*ENUMERATED*	Read
description(28)	*[CharacterString]*	Optional
present-value(85)	*[DateTime]*	Read
status-flags(111)	*BITSTRING*	Read
event-state(36)	*ENUMERATED*	Optional
reliability(103)	*ENUMERATED*	Optional
out-of-service(81)	*[BOOLEAN]*	Optional
is-utc(344)	*[BOOLEAN]*	Optional
priority-array(87)	*[PriorityArray]*	Optional
relinquish-default(104)	*[DateTime]*	Optional
reliability-evaluation-inhibit(357)	*[BOOLEAN]*	Optional
property-list(371)	*ENUMERATED, Array[N]*	Read
profile-name(168)	*[CharacterString]*	Optional

TIMEPATTERNVALUE(49)

PropertyName	PropertyDatatype	PropertyConformanceCode
object-identifier(75)	*[ObjectIdentifier]*	Read
object-name(77)	*[CharacterString]*	Read
object-type(79)	*ENUMERATED*	Read
description(28)	*[CharacterString]*	Optional
present-value(85)	*[Time]*	Read
status-flags(111)	*BITSTRING*	Read
event-state(36)	*ENUMERATED*	Optional
reliability(103)	*ENUMERATED*	Optional
out-of-service(81)	*[BOOLEAN]*	Optional
priority-array(87)	*[PriorityArray]*	Optional
relinquish-default(104)	*[Time]*	Optional
reliability-evaluation-inhibit(357)	*[BOOLEAN]*	Optional
property-list(371)	*ENUMERATED, Array[N]*	Read
profile-name(168)	*[CharacterString]*	Optional

DATEPATTERNVALUE(41)

PropertyName	PropertyDatatype	PropertyConformanceCode
object-identifier(75)	*[ObjectIdentifier]*	Read
object-name(77)	*[CharacterString]*	Read
object-type(79)	*ENUMERATED*	Read
description(28)	*[CharacterString]*	Optional
present-value(85)	*[Date]*	Read
status-flags(111)	*BITSTRING*	Read
event-state(36)	*ENUMERATED*	Optional

PropertyName	PropertyDatatype	PropertyConformanceCode
reliability(103)	*ENUMERATED*	Optional
out-of-service(81)	*[BOOLEAN]*	Optional
priority-array(87)	*[PriorityArray]*	Optional
relinquish-default(104)	*[Date]*	Optional
reliability-evaluation-inhibit(357)	*[BOOLEAN]*	Optional
property-list(371)	*ENUMERATED, Array[N]*	Read
profile-name(168)	*[CharacterString]*	Optional

NETWORKSECURITY(38)

PropertyName	PropertyDatatype	PropertyConformanceCode
object-identifier(75)	*[ObjectIdentifier]*	Read
object-name(77)	*[CharacterString]*	Read
object-type(79)	*ENUMERATED*	Read
description(28)	*[CharacterString]*	Optional
base-device-security-policy(327)	*ENUMERATED*	Write
network-access-security-policies(332)	*[NetworkSecurityPolicy], Array[N]*	Write
security-time-window(335)	*[Unsigned]*	Write
packet-reorder-time(333)	*[Unsigned]*	Write
distribution-key-revision(328)	*UNSIGNED*	Read
key-sets(330)	*[SecurityKeySet], Array[2]*	Read
last-key-server(331)	*[AddressBinding]*	Write
security-pdu-timeout(334)	*UNSIGNED*	Write

PropertyName	PropertyDatatype	PropertyConformanceCode
update-key-set-timeout(337)	*UNSIGNED*	Read
supported-security-algorithms(336)	*UNSIGNED, List*	Read
do-not-hide(329)	*[BOOLEAN]*	Write
property-list(371)	*ENUMERATED, Array[N]*	Read
profile-name(168)	*[CharacterString]*	Optional

GLOBALGROUP(26)

PropertyName	PropertyDatatype	PropertyConformanceCode
object-identifier(75)	*[ObjectIdentifier]*	Read
object-name(77)	*[CharacterString]*	Read
object-type(79)	*ENUMERATED*	Read
description(28)	*[CharacterString]*	Optional
group-members(345)	*[DeviceObjectPropertyReference], Array[N]*	Read
group-member-names(346)	*[CharacterString], Array[N]*	Optional
present-value(85)	*[PropertyAccessResult], Array[N]*	Read
status-flags(111)	*BITSTRING*	Read
event-state(36)	*ENUMERATED*	Read
member-status-flags(347)	*BITSTRING*	Read
reliability(103)	*ENUMERATED*	Optional
out-of-service(81)	*[BOOLEAN]*	Read
update-interval(118)	*[Unsigned]*	Optional
requested-update-interval(348)	*[Unsigned]*	Optional

PropertyName	PropertyDatatype	PropertyConformanceCode
cov-resubscription-interval(128)	*[Unsigned]*	Optional
client-cov-increment(127)	*[ClientCOV]*	Optional
time-delay(113)	*[Unsigned]*	Optional
notification-class(17)	*[Unsigned]*	Optional
event-enable(35)	*BITSTRING*	Optional
acked-transitions(0)	*BITSTRING*	Optional
notify-type(72)	*ENUMERATED*	Optional
event-time-stamps(130)	*[TimeStamp], Array[3]*	Optional
event-message-texts(351)	*[CharacterString], Array[3]*	Optional
event-message-texts-config(352)	*[CharacterString], Array[3]*	Optional
event-detection-enable(353)	*[BOOLEAN]*	Optional
event-algorithm-inhibit-ref(355)	*[ObjectPropertyReference]*	Optional
event-algorithm-inhibit(354)	*[BOOLEAN]*	Optional
time-delay-normal(356)	*[Unsigned]*	Optional
covu-period(349)	*[Unsigned]*	Optional
covu-recipients(350)	*[Recipient], List*	Optional
reliability-evaluation-inhibit(357)	*[BOOLEAN]*	Optional
property-list(371)	*ENUMERATED, Array[N]*	Read
profile-name(168)	*[CharacterString]*	Optional

NOTIFICATIONFORWARDER(51)

PropertyName	PropertyDatatype	PropertyConformanceCode
object-identifier(75)	*[ObjectIdentifier]*	Read
object-name(77)	*[CharacterString]*	Read
object-type(79)	*ENUMERATED*	Read
description(28)	*[CharacterString]*	Optional
status-flags(111)	*BITSTRING*	Read
reliability(103)	*ENUMERATED*	Read
out-of-service(81)	*[BOOLEAN]*	Read
recipient-list(102)	*[Destination], List*	Read
subscribed-recipients(362)	*[EventNotificationSubscription], List*	Write
process-identifier-filter(361)	*[ProcessIdSelection]*	Read
port-filter(363)	*[PortPermission], Array[N]*	Optional
local-forwarding-only(360)	*[BOOLEAN]*	Read
reliability-evaluation-inhibit(357)	*[BOOLEAN]*	Optional
property-list(371)	*ENUMERATED, Array[N]*	Read
profile-name(168)	*[CharacterString]*	Optional

ALERTENROLLMENT(52)

PropertyName	PropertyDatatype	PropertyConformanceCode
object-identifier(75)	*[ObjectIdentifier]*	Read
object-name(77)	*[CharacterString]*	Read
object-type(79)	*ENUMERATED*	Read
description(28)	*[CharacterString]*	Optional
present-value(85)	*[ObjectIdentifier]*	Read

PropertyName	PropertyDatatype	PropertyConformanceCode
event-state(36)	*ENUMERATED*	Read
event-detection-enable(353)	*[BOOLEAN]*	Read
notification-class(17)	*[Unsigned]*	Read
event-enable(35)	*BITSTRING*	Read
acked-transitions(0)	*BITSTRING*	Read
notify-type(72)	*ENUMERATED*	Read
event-time-stamps(130)	*[TimeStamp], Array[3]*	Read
event-message-texts(351)	*[CharacterString], Array[3]*	Optional
event-message-texts-config(352)	*[CharacterString], Array[3]*	Optional
event-algorithm-inhibit-ref(355)	*[ObjectPropertyReference]*	Optional
event-algorithm-inhibit(354)	*[BOOLEAN]*	Optional
property-list(371)	*ENUMERATED, Array[N]*	Read
profile-name(168)	*[CharacterString]*	Optional

CHANNEL(53)

PropertyName	PropertyDatatype	PropertyConformanceCode
object-identifier(75)	*[ObjectIdentifier]*	Read
object-name(77)	*[CharacterString]*	Read
object-type(79)	*ENUMERATED*	Read
description(28)	*[CharacterString]*	Optional
present-value(85)	*[ChannelValue]*	Write
last-priority(369)	*[Unsigned]*	Read
write-status(370)	*ENUMERATED*	Read

PropertyName	PropertyDatatype	PropertyConformanceCode
status-flags(111)	*BITSTRING*	Read
reliability(103)	*ENUMERATED*	Optional
out-of-service(81)	*[BOOLEAN]*	Read
list-of-object-property-references(54)	*[DeviceObjectPropertyReference], Array[N]*	Write
execution-delay(368)	*[Unsigned], Array[N]*	Optional
allow-group-delay-inhibit(365)	*[BOOLEAN]*	Optional
channel-number(366)	*UNSIGNED*	Write
control-groups(367)	*UNSIGNED, Array[N]*	Write
event-detection-enable(353)	*[BOOLEAN]*	Optional
notification-class(17)	*[Unsigned]*	Optional
event-enable(35)	*BITSTRING*	Optional
event-state(36)	*ENUMERATED*	Optional
acked-transitions(0)	*BITSTRING*	Optional
notify-type(72)	*ENUMERATED*	Optional
event-time-stamps(130)	*[TimeStamp], Array[3]*	Optional
event-message-texts(351)	*[CharacterString], Array[3]*	Optional
event-message-texts-config(352)	*[CharacterString], Array[3]*	Optional
reliability-evaluation-inhibit(357)	*[BOOLEAN]*	Optional
property-list(371)	*ENUMERATED, Array[N]*	Read
profile-name(168)	*[CharacterString]*	Optional

LIGHTINGOUTPUT(54)

PropertyName	PropertyDatatype	PropertyConformanceCode
object-identifier(75)	*[ObjectIdentifier]*	Read
object-name(77)	*[CharacterString]*	Read
object-type(79)	*ENUMERATED*	Read
present-value(85)	*[REAL]*	Write
tracking-value(164)	*[REAL]*	Read
lighting-command(380)	*[LightingCommand]*	Write
in-progress(378)	*ENUMERATED*	Read
description(28)	*[CharacterString]*	Optional
status-flags(111)	*BITSTRING*	Read
reliability(103)	*ENUMERATED*	Optional
out-of-service(81)	*[BOOLEAN]*	Read
blink-warn-enable(373)	*[BOOLEAN]*	Read
egress-time(377)	*[Unsigned]*	Read
egress-active(386)	*[BOOLEAN]*	Read
default-fade-time(374)	*[Unsigned]*	Read
default-ramp-rate(375)	*[REAL]*	Read
default-step-increment(376)	*[REAL]*	Read
transition(385)	*ENUMERATED*	Optional
feedback-value(40)	*[REAL]*	Optional
priority-array(87)	*[PriorityArray]*	Read
relinquish-default(104)	*[REAL]*	Read
power(384)	*[REAL]*	Optional
instantaneous-power(379)	*[REAL]*	Optional
min-actual-value(383)	*[REAL]*	Optional

PropertyName	PropertyDatatype	PropertyConformanceCode
max-actual-value(382)	*[REAL]*	Optional
lighting-command-default-priority(381)	*[Unsigned]*	Read
cov-increment(22)	*[REAL]*	Optional
reliability-evaluation-inhibit(357)	*[BOOLEAN]*	Optional
property-list(371)	*ENUMERATED, Array[N]*	Read
profile-name(168)	*[CharacterString]*	Optional

6.3 Driver objects and datatypes

Driver objects are areas available in the PLC, such as markers, data blocks etc. Here you can find out which driver objects are provided by the driver and which IEC data types can be assigned to the respective driver objects.

6.3.1 Driver objects

The following object types are available in this driver:

Driver Object Type	Channel type	Read	Write	Supported data types	Description
Command status	11	X	--	*UDINT*	State of whether command was executed successfully: *0*: Successful *1*: Command is executed *2*: Errors
Device event information	10	X	--	*STRING*	Summary of all open alarms of a device
Event	9	X	X	*BOOL, STRING*	Receipt of BACnet event notifications. Addressing: **<Device name>.<Object**

Driver Object Type	Channel type	Read	Write	Supported data types	Description
					name>.Event.<Event state>.bool or: **<Device name>.<Object name>.Event.<Event state>.string**
Property	8	X	X	*BOOL, DINT, INT, LREAL, REAL, SINT, STRING, UDINT, UINT, USINT*	Corresponds to PLC marker
Driver variable	35	X	X	*BOOL, DINT, INT, REAL, SINT, STRING, UDINT, UINT, USINT*	Variables for the statistical analysis of communication. Find out more in the chapter about the Driver variables (on page 126)
Trendlog	64	X	--	*BOOL, DINT, INT, LREAL, SINT, STRING, UDINT, UINT, USINT*	Trendlog Variable For this reason the variable property **Harddisk data storage active** for trendlog variables is always active and grayed out.in the zenon Editor. Also the variable property **Recording type** is pre-configured with *Postsorted values (RDA)* and grayed out. **Attention:** The creation of multiple Trendlog variables with the same address is not supported. Model for Variable names: ▸ *<Device name>.<Object*

Driver Object Type	Channel type	Read	Write	Supported data types	Description
					name>.LogStatus ▸ *<Device name>.<Object name>.TimeChange* ▸ *<Device name>.<Object name>.Value bzw.* ▸ *<Device name>.<Object name>.Value[n]*
Trendlog - internal total record count	65	X	X	*UDINT*	Trendlog variable for configuration of the Trendlog redundancy in zenon. For nond-redundancy operation this driver object type is not used. You can find detailed information on this in the Communication details (Driver variables) (on page 126) chapter. Addressing: *<Device name>.<Object name>.internal_total_record_count*

Key:

X: supported

--: not supported

EVENT

AE-N-A ALARM AND EVENT NOTIFICATION-A:

Receipt of *BACnet EventNotifications* (BACnet alarms and events). A *BACnet EventNotification* is sent by BACnet objects. Prerequisite: Linking to a *Notification Class Object*.

The **Event** driver object type can be used to evaluate these events in zenon.

Addressing:

<Device name>.<Object name>.Event.<Event state>.bool
or

<Device name>.<Object name>.Event.<Event state>.string

Note:

- Whether (**.**) or **#** is used as a separator depends on the settings in the **Property Separator** property in the **Options** tab in the driver configuration (on page 17).
- Exception: The device name is always separated from the object name by a period (.).
- The last element of the name (**bool** or **string in the example)** can be issued freely by the user.

<EVENT STATE> VALUES

<Event state> can have the following values:

Event state (text)	Event state (numerical)
Normal	0
Fault	1
Off-normal	2
High-limit	3
Low-limit	4
Life-safety-alarm	5

The event state can also be set as a number using the **BACnet event state** property. In this case, the **event state** can be replaced in the name by any desired string.

Requirements: The process ID in the notification class (ID for the recipient of the events in the recipient address) must be set to *0*.

PROCEDURE

- Each time notification of the object with the given **Event states** is received, the associated BOOL variable is first set to *false* and then to *true*.
- If a notification of an object with different **Event states** is received, the variable is set to *false*. As a result of this, it is also possible to edit events without changing the **Event states**.
- If **Event state** is active, the whole received event notification of the string variables is assigned.
- If an **Event state** that is not the same as the configured one is received, the variable is set to an empty string.
- The BOOL variables are initialized when the driver is started and the string variables are set to empty string.

The format of the string variables is similar to the property format.

FORMAT FROM BACNET SPECIFICATION

SEQUENCE {

processIdentifier [0] Unsigned32,

initiatingDeviceIdentifier [1] BACnetObjectIdentifier,

eventObjectIdentifier [2] BACnetObjectIdentifier,

timeStamp [3] BACnetTimeStamp,

notificationClass [4] Unsigned,

priority [5] Unsigned8,

eventType [6] BACnetEventType,

messageText [7] CharacterString OPTIONAL,

notifyType [8] BACnetNotifyType,

ackRequired [9] BOOLEAN OPTIONAL,

fromState [10] BACnetEventState OPTIONAL,

toState [11] BACnetEventState,

eventValues [12] BACnetNotificationParameters OPTIONAL

EXAMPLE

{{0} 0},{{1} 0008 0000010},{{2} 0002 0000000},{<3> {<2> {[10] 110.08.09.1},{[11] 12:03:52.018}}},{{4} 1},{{5} 128},{{6} 5},{{8} 1},{{9} FALSE},{{10} 0},{{11} 4},{<12> {<5> {{0} 0.000000},{{1} 1000},{{2} 0.000000},{{3} 100.000000}}}}

AE-ACK-A ALARM AND EVENT ACK-A

Acknowledge: The event is set to *acknowledged* by means of the event variable (AcknowledgeAlarm Telegram). In doing so, the last event received with this **event state** is *acknowledged*.

AE-INFO-A ALARM AND EVENT INFORMATION-A

Read a summary of all open alarms for one device. This summary can be read using an **Event info** object string variable.

FORMAT FROM BACNET SPECIFICATION

SEQUENCE OF SEQUENCE {

objectIdentifier [0] BACnetObjectIdentifier,

eventState [1] BACnetEventState,

acknowledgedTransitions [2] BACnetEventTransitionBits,

eventTimeStamps [3] SEQUENCE SIZE (3) OF BACnetTimeStamp,

notifyType [4] BACnetNotifyType,

eventEnable [5] BACnetEventTransitionBits,

eventPriorities [6] SEQUENCE SIZE (3) OF Unsigned

}

EXAMPLE

{(0) 0005 0000001},{(1) 2},{(2) 111},{<3> {<2> {[10] 110.07.08.4},{[11] 15:50:28:048}},{<2> {[10] 255.255.255.255},{[11] 255:255:255:255}},{<2> {[10] 110.07.08.4},{[11] 15:50:20:047}}},{(4) 0},{(5) 000},{<6> {[2] 128},{[2] 128},{[2] 128}},{(0) 0002 0000001},{(1) 4},{(2) 111},{<3> {<2> {[10] 255.255.255.255},{[11] 255:255:255:255}},{<2> {[10] 255.255.255.255},{[11] 255:255:255:255}},{<2> {[10] 255.255.255.255},{[11] 255:255:255:255}}},{(4) 0},{(5) 010},{<6> {[2] 128},{[2] 128},{[2] 128}},{(0) 0002 0000000},{(1) 4},{(2) 101},{<3> {<2> {[10] 110.08.09.1},{[11] 12:03:52:018}},{<2> {[10] 110.07.29.4},{[11] 15:41:08:088}},{<2> {[10] 110.07.30.5},{[11] 07:16:12:082}}},{(4) 1},{(5) 111},{<6> {[2] 128},{[2] 128},{[2] 128}

CHANNEL TYPE

The term **Kanaltyp** is the internal numerical name of the driver object type. It is also used for the extended DBF import/export of the variables.

"**Kanaltyp**" is used for advanced CSV import/export of variables in the "**HWObjectType**" column.

6.3.2 Mapping of the data types

All variables in zenon are derived from IEC data types. The following table compares the IEC datatypes with the datatypes of the PLC.

Control	zenon	Data type
Boolean	BOOL	8
-	USINT	9
-	SINT	10
-	UINT	2
-	INT	1
Unsigned, Bit-String, Date, Time, Enumerated	UDINT	4
Integer	DINT	3
-	ULINT	27
-	LINT	26
Real	REAL	5
Double	LREAL	6
Character string, NULL, Boolean, Unsigned, Integer, Real, Double, Bit-String, Octet string, Enumerated, Date, Time, ObjectID and all constructed or combined types	STRING	12
-	WSTRING	21
-	DATE	18
-	TIME	17
-	DATE_AND_TIME	20
-	TOD (Time of Day)	19

DATA TYPE

The term **data type** is the internal numerical identification of the data type. It is also used for the extended DBF import/export of the variables.

6.4 Creating variables by importing

Variables can also be imported by importing them. The XML and DBF import is available for every driver.

> **Information**
>
> You can find details on the import and export of variables in the Import-Export manual in the Variables section.

6.4.1 XML import

During XML import of variables or data types, these are first assigned to a driver and then analyzed. Before import, the user decides whether and how the respective element (variable or data type) is to be imported:

- *Import*:
  The element is imported as a new element.
- *Overwrite*:
  The element is imported and overwrites a pre-existing element.
- *Do not import*:
  The element is not imported.

Note: The actions and their durations are shown in a progress bar during import. The import of variables is described in the following documentation. Data types are imported along the same lines.

REQUIREMENTS

The following conditions are applicable during import:

- **Backward compatibility**

  At the XML import/export there is no backward compatibility. Data from older zenon versions can be taken over. The handover of data from newer to older versions is not supported.
- **Consistency**

The XML file to be imported has to be consistent. There is no plausibility check on importing the file. If there are errors in the import file, this can lead to undesirable effects in the project.

Particular attention must be paid to this, primarily if not all properties exist in the XML file and these are then filled with default values. E.g.: A binary variable has a limit value of *300*.

- **Structure data types**

  Structure data types must have the same number of structure elements.
  Example: A structure data type in the project has 3 structure elements. A data type with the same name in the XML file has 4 structure elements. Then none of the variables based on this data type in the file are imported into the project.

> **Hint**
>
> You can find further information on XML import in the **Import - Export** manual, in the **XML import** chapter.

6.4.2 DBF Import/Export

Data can be exported to and imported from dBase.

> **Information**
>
> Import and Export via CSV or dBase supported; no driver specific variable settings, such as formulas. Use export/import via XML for this.

IMPORT DBF FILE

To start the import:

1. right-click on the variable list.
2. In the drop-down list of **Extended export/import...** select the **Import dBase** command.
3. Follow the instructions of the import assistant.

The format of the file is described in the chapter File structure.

Information

Note:

- Driver object type and data type must be amended to the target driver in the DBF file in order for variables to be imported.
- dBase does not support structures or arrays (complex variables) at import.

EXPORT DBF FILE

To start the export:

1. right-click on the variable list.
2. In the drop-down list of **Extended export/import...** select the **Export dBase...** command .
3. Follow the instructions of the import assistant.

Attention

DBF files:

- must correspond to the 8.3 DOS format for filenames (8 alphanumeric characters for name, 3 character suffix, no spaces)
- must not have dots (.) in the path name.
  e.g. the path *C:\users\John.Smith\test.dbf* is invalid.
  Valid: *C:\users\JohnSmith\test.dbf*
- must be stored close to the root directory in order to fulfill the limit for file name length including path: maximum 255 characters

The format of the file is described in the chapter File structure.

Information

dBase does not support structures or arrays (complex variables) at export.

FILE STRUCTURE OF THE DBASE EXPORT FILE

The dBaseIV file must have the following structure and contents for variable import and export:

Attention

dBase does not support structures or arrays (complex variables) at export.

DBF files must:

- conform with their name to the 8.3 DOS format (8 alphanumeric characters for name, 3 characters for extension, no space)
- Be stored close to the root directory (Root)

STRUCTURE

Identification	Type	Field size	Comment
KANALNAME	Char	128	Variable name. The length can be limited using the **MAX_LAENGE** entry in the **project.ini** file.
KANAL_R	C	128	The original name of a variable that is to be replaced by the new name entered under "VARIABLENNAME" (variable name) (field/column must be entered manually). The length can be limited using the **MAX_LAENGE** entry in the **project.ini** file.
KANAL_D	Log	1	The variable is deleted with the *1* entry (field/column has to be created by hand).
TAGNR	C	128	Identification. The length can be limited using the **MAX_LAENGE** entry in the **project.ini** file.
EINHEIT	C	11	Technical unit
DATENART	C	3	Data type (e.g. bit, byte, word, ...) corresponds to the data type.
KANALTYP	C	3	Memory area in the PLC (e.g. marker area, data area, ...) corresponds to the driver object type.
HWKANAL	Num	3	Net address
BAUSTEIN	N	3	Datablock address (only for variables from the data area

Identification	Type	Field size	Comment
			of the PLC)
ADRESSE	N	5	Offset
BITADR	N	2	For bit variables: bit address For byte variables: 0=lower, 8=higher byte For string variables: Length of string (max. 63 characters)
ARRAYSIZE	N	16	Number of variables in the array for index variables ATTENTION: Only the first variable is fully available. All others are only available for VBA or the Recipegroup Manager
LES_SCHR	L	1	Write-Read-Authorization 0: Not allowed to set value. 1: Allowed to set value.
MIT_ZEIT	R	1	time stamp in zenon (only if supported by the driver)
OBJEKT	N	2	Driver-specific ID number of the primitive object comprises TREIBER-OBJEKTTYP and DATENTYP
SIGMIN	Float	16	Non-linearized signal - minimum (signal resolution)
SIGMAX	F	16	Non-linearized signal - maximum (signal resolution)
ANZMIN	F	16	Technical value - minimum (measuring range)
ANZMAX	F	16	Technical value - maximum (measuring range)
ANZKOMMA	N	1	Number of decimal places for the display of the values (measuring range)
UPDATERATE	F	19	Update rate for mathematics variables (in sec, one decimal possible) not used for all other variables
MEMTIEFE	N	7	Only for compatibility reasons
HDRATE	F	19	HD update rate for historical values (in sec, one decimal possible)
HDTIEFE	N	7	HD entry depth for historical values (number)
NACHSORT	R	1	HD data as postsorted values

Identification	Type	Field size	Comment
DRRATE	F	19	Updating to the output (for zenon DDE server, in [s], one decimal possible)
HYST_PLUS	F	16	Positive hysteresis, from measuring range
HYST_MINUS	F	16	Negative hysteresis, from measuring range
PRIOR	N	16	Priority of the variable
REAMATRIZE	C	32	Allocated reaction matrix
ERSATZWERT	F	16	Substitute value, from measuring range
SOLLMIN	F	16	Minimum for set value actions, from measuring range
SOLLMAX	F	16	Maximum for set value actions, from measuring range
VOMSTANDBY	R	1	Get value from standby server; the value of the variable is not requested from the server but from the Standby Server in redundant networks
RESOURCE	C	128	Resources label. Free string for export and display in lists. The length can be limited using the MAX_LAENGE entry in **project.ini**.
ADJWVBA	R	1	Non-linear value adaption: *0*: Non-linear value adaption is used *1*: Non-linear value adaption is not used
ADJZENON	C	128	Linked VBA macro for reading the variable value for non-linear value adjustment.
ADJWVBA	C	128	ed VBA macro for writing the variable value for non-linear value adjustment.
ZWREMA	N	16	Linked counter REMA.
MAXGRAD	N	16	Gradient overflow for counter REMA.

Attention

When importing, the driver object type and data type must be amended to the target driver in the DBF file in order for variables to be imported.

LIMIT VALUE DEFINITION

Limit definition for limit values *1* to *4*, or status *1* to *4*:

Identification	Type	Field size	Comment
AKTIV1	R	1	Limit value active (per limit value available)
GRENZWERT1	F	20	technical value or ID number of a linked variable for a dynamic limit value (see VARIABLEx) (if VARIABLEx is *1* and here it is *-1*, the existing variable linkage is not overwritten)
SCHWWERT1	F	16	Threshold value for limit value
HYSTERESE1	F	14	Is not used
BLINKEN1	R	1	Set blink attribute
BTB1	R	1	Logging in CEL
ALARM1	R	1	Alarm
DRUCKEN1	R	1	Printer output (for CEL or Alarm)
QUITTIER1	R	1	Must be acknowledged
LOESCHE1	R	1	Must be deleted
VARIABLE1	R	1	Dyn. limit value linking the limit is defined by an absolute value (see field GRENZWERTx).
FUNC1	R	1	Functions linking
ASK_FUNC1	R	1	Execution via Alarm Message List
FUNC_NR1	N	10	ID number of the linked function (if "-1" is entered here, the existing function is not overwritten during import)
A_GRUPPE1	N	10	Alarm/Event Group
A_KLASSE1	N	10	Alarm/Event Class
MIN_MAX1	C	3	Minimum, Maximum
FARBE1	N	10	Color as Windows coding
GRENZTXT1	C	66	Limit value text
A_DELAY1	N	10	Time delay

Identification	Type	Field size	Comment
INVISIBLE1	R	1	Invisible

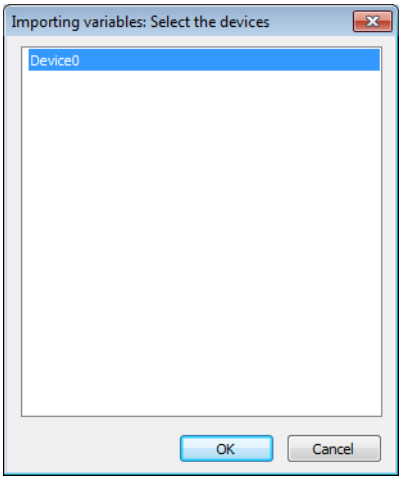
Expressions in the column "Comment" refer to the expressions used in the dialog boxes for the definition of variables. For more information, see chapter Variable definition.

6.4.3 Online import

You can automatically create variables for the properties of all objects of one or several devices by means of online import. The devices must be created in the drive configuration (on page 23) to do this.

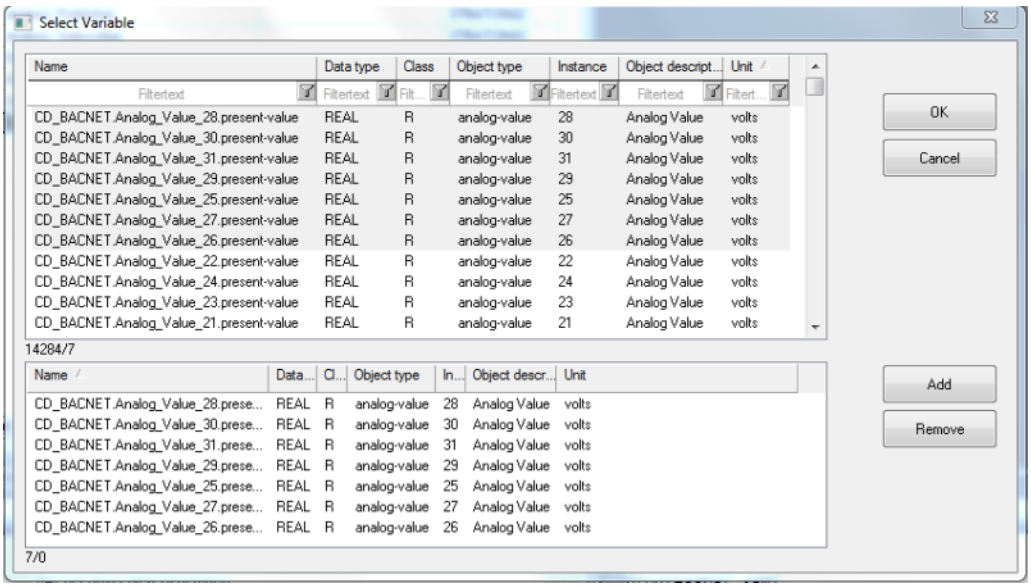
To import data online:

1. Right-click on the driver
2. Select **Import variables from the driver ...**
   The dialog to select the device is opened.



3. Select the desired device.
4. Click on the **OK** button.

5. The dialog for variable import is opened



6. Select the desired values

   Filtering according to the following factors is possible:

   - **Name**
   - **Data type**
   - **Class**
   - **Object type**
   - **Instance**
   - **Object description**
   - **Unit**

7. The **Add** button is used to accept selected elements into the import list
   You remove elements from the import list with **Remove**
8. Confirm the selection by clicking on **OK**
9. The variables are imported

UNITS

Units are read off during import for:

- Analog Value
- Analog Input
- Analog Output
- Loop Object

The **Unit ID** is converted into a readable string. This string is saved as a variable unit. Units contain the prefix @, in order to allow language switching. Units correspond to the BACnet Standard 135-2001.

Supported unit groups:

- Area
- Currency
- Electrical
- Energy
- Enthalpy
- Entropy
- Frequency
- Humidity
- Length
- Light
- Mass
- Mass Flow
- Power
- Pressure
- Temperature
- Time
- Velocity
- Volume
- Volumetric Flow

6.5 Communication details (Driver variables)

The driver kit implements a number of driver variables. This variables are part of the driver object type *Communication details*. These are divided into:

- Information
- Configuration
- Statistics and
- Error message

The definitions of the variables implemented in the driver kit are available in the import file **DRVVAR.DBF** and can be imported from there.
Path to file: *%ProgramData%\COPA-DATA\zenon<Versionsnummer>\PredefinedVariables*

Note: Variable names must be unique in zenon. If driver variables of the driver object type *Communication details* are to be imported from **DRVVAR.DBF** again, the variables that were imported beforehand must be renamed.

Information

Not every driver supports all driver variables of the driver object type *Communication details*.

For example:

- Variables for modem information are only supported by modem-compatible drivers.
- Driver variables for the polling cycle are only available for pure polling drivers.
- Connection-related information such as **ErrorMSG** is only supported for drivers that only edit one connection at a a time.

INFORMATION

Name from import	Type	Offset	Description
MainVersion	*UINT*	0	Main version number of the driver.
SubVersion	*UINT*	1	Sub version number of the driver.
BuildVersion	*UINT*	29	Build version number of the driver.
RTMajor	*UINT*	49	zenon main version number
RTMinor	*UINT*	50	zenon sub version number
RTSp	*UINT*	51	zenon Service Pack number
RTBuild	*UINT*	52	zenon build number
LineStateIdle	*BOOL*	24.0	TRUE, if the modem connection is idle
LineStateOffering	*BOOL*	24.1	TRUE, if a call is received
LineStateAccepted	*BOOL*	24.2	The call is accepted
LineStateDialtone	*BOOL*	24.3	Dialtone recognized
LineStateDialing	*BOOL*	24.4	Dialing active

Name from import	Type	Offset	Description
LineStateRingBack	*BOOL*	24.5	While establishing the connection
LineStateBusy	*BOOL*	24.6	Target station is busy
LineStateSpecialInfo	*BOOL*	24.7	Special status information received
LineStateConnected	*BOOL*	24.8	Connection established
LineStateProceeding	*BOOL*	24.9	Dialing completed
LineStateOnHold	*BOOL*	24.10	Connection in hold
LineStateConferenced	*BOOL*	24.11	Connection in conference mode.
LineStateOnHoldPendConf	*BOOL*	24.12	Connection in hold for conference
LineStateOnHoldPendTransfer	*BOOL*	24.13	Connection in hold for transfer
LineStateDisconnected	*BOOL*	24.14	Connection terminated.
LineStateUnknow	*BOOL*	24.15	Connection status unknown
ModemStatus	*UDINT*	24	Current modem status
TreiberStop	*BOOL*	28	Driver stopped For *driver stop*, the variable has the value *TRUE* and an **OFF** bit. After the driver has started, the variable has the value *FALSE* and no **OFF** bit.
SimulRTState	*UDINT*	60	Informs the state of Runtime for driver simulation.
ConnectionStates	*STRING*	61	Internal connection status of the driver to the PLC. Connection statuses: ▸ *0:* Connection OK ▸ *1:* Connection failure ▸ *2:* Connection simulated Formating: **<Net address>:<Connection status>;...;...;** A connection is only known after a variable

Name from import	Type	Offset	Description
			has first signed in. In order for a connection to be contained in a string, a variable of this connection must be signed in once. The status of a connection is only updated if a variable of the connection is signed in. Otherwise there is no communication with the corresponding controller.

CONFIGURATION

Name from import	Type	Offset	Description
ReconnectInRead	*BOOL*	27	If TRUE, the modem is automatically reconnected for reading
ApplyCom	*BOOL*	36	Apply changes in the settings of the serial interface. Writing to this variable immediately results in the method SrvDrvVarApplyCom being called (which currently has no further function).
ApplyModem	*BOOL*	37	Apply changes in the settings of the modem. Writing this variable immediately calls the method SrvDrvVarApplyModem. This closes the current connection and opens a new one according to the settings **PhoneNumberSet** and **ModemHwAdrSet**.
PhoneNumberSet	*STRING*	38	Telephone number, that should be used
ModemHwAdrSet	*DINT*	39	Hardware address for the telephone number
GlobalUpdate	*UDINT*	3	Update time in milliseconds (ms).
BGlobalUpdaten	*BOOL*	4	TRUE, if update time is global
TreiberSimul	*BOOL*	5	TRUE, if driver in sin simulation mode
TreiberProzab	*BOOL*	6	TRUE, if the variables update list should be kept in the memory
ModemActive	*BOOL*	7	TRUE, if the modem is active for the driver
Device	*STRING*	8	Name of the serial interface or name of the modem

Name from import	Type	Offset	Description
ComPort	*UINT*	9	Number of the serial interface.
Baudrate	*UDINT*	10	Baud rate of the serial interface.
Parity	*SINT*	11	Parity of the serial interface
ByteSize	*USINT*	14	Number of bits per character of the serial interface Value = *0* if the driver cannot establish any serial connection.
StopBit	*USINT*	13	Number of stop bits of the serial interface.
Autoconnect	*BOOL*	16	TRUE, if the modem connection should be established automatically for reading/writing
PhoneNumber	*STRING*	17	Current telephone number
ModemHwAdr	*DINT*	21	Hardware address of current telephone number
RxIdleTime	*UINT*	18	Modem is disconnected, if no data transfer occurs for this time in seconds (s)
WriteTimeout	*UDINT*	19	Maximum write duration for a modem connection in milliseconds (ms).
RingCountSet	*UDINT*	20	Number of ringing tones before a call is accepted
ReCallIdleTime	*UINT*	53	Waiting time between calls in seconds (s).
ConnectTimeout	*UINT*	54	Time in seconds (s) to establish a connection.

STATISTICS

Name from import	Type	Offset	Description
MaxWriteTime	*UDINT*	31	The longest time in milliseconds (ms) that is required for writing.
MinWriteTime	*UDINT*	32	The shortest time in milliseconds (ms) that is required for writing.
MaxBlkReadTime	*UDINT*	40	Longest time in milliseconds (ms) that is required to read a data block.

Name from import	Type	Offset	Description
MinBlkReadTime	*UDINT*	41	Shortest time in milliseconds (ms) that is required to read a data block.
WriteErrorCount	*UDINT*	33	Number of writing errors
ReadSucceedCount	*UDINT*	35	Number of successful reading attempts
MaxCycleTime	*UDINT*	22	Longest time in milliseconds (ms) required to read all requested data.
MinCycleTime	*UDINT*	23	Shortest time in milliseconds (ms) required to read all requested data.
WriteCount	*UDINT*	26	Number of writing attempts
ReadErrorCount	*UDINT*	34	Number of reading errors
MaxUpdateTimeNormal	*UDINT*	56	Time since the last update of the priority group **Normal** in milliseconds (ms).
MaxUpdateTimeHigher	*UDINT*	57	Time since the last update of the priority group **Higher** in milliseconds (ms).
MaxUpdateTimeHigh	*UDINT*	58	Time since the last update of the priority group **High** in milliseconds (ms).
MaxUpdateTimeHighest	*UDINT*	59	Time since the last update of the priority group **Highest** in milliseconds (ms).
PokeFinish	*BOOL*	55	Goes to *1* for a query, if all current pokes were executed

ERROR MESSAGE

Name from import	Type	Offset	Description
ErrorTimeDW	*UDINT*	2	Time (in seconds since 1.1.1970), when the last error occurred.
ErrorTimeS	*STRING*	2	Time (in seconds since 1.1.1970), when the last error occurred.
RdErrPrimObj	*UDINT*	42	Number of the PrimObject, when the last reading error occurred.
RdErrStationsName	*STRING*	43	Name of the station, when the last reading error occurred.

Name from import	Type	Offset	Description
RdErrBlockCount	*UINT*	44	Number of blocks to read when the last reading error occurred.
RdErrHwAdresse	*DINT*	45	Hardware address when the last reading error occurred.
RdErrDatablockNo	*UDINT*	46	Block number when the last reading error occurred.
RdErrMarkerNo	*UDINT*	47	Marker number when the last reading error occurred.
RdErrSize	*UDINT*	48	Block size when the last reading error occurred.
DrvError	*USINT*	25	Error message as number
DrvErrorMsg	*STRING*	30	Error message as text
ErrorFile	*STRING*	15	Name of error log file

7 Driver-specific functions

The driver supports the following functions:

- Driver commands (on page 143)
- Access method (spontaneous or polling reading) (on page 132)
- Mapping the BACnet status flags to the status bits of a variable (on page 135)
- Mapping of BACnet data types to string variables (on page 135)
- Device Management (on page 139)

7.1 Access method (spontaneous or polling reading)

Depending on the priority that has been set (property in the address setting), the variable is either read spontaneously via COV subscriptions (priority normal) or polled (all other priorities). For all priorities except *Normal* the set update interval equals the polling cycle.

Attention

Normally only the **PRESENT-VALUE** and **STATUS FLAGS** properties can be read via COV subscription, i.e. spontaneously. As a COV-Subscription always refers to only one object and not a property, the creation of a variable which cannot be read spontaneously leads to a log entry or a **INVALID-bit**. In this case, the value of the variable remains empty.

COV SUBSCRIPTIONS

The following applies to change of value subscriptions on the BACnet device:

- COV subscriptions are unadvised if variables are inactive (e.g. screen switch) and when the driver is closed.
- After a variable is written, the variable will not be advised (subscribed) again.
- A shadow value is saved in the variable for COV subscriptions. Thus, the variable does not need to be advised (subscribed) and read again if the variable is switched, for example, from a substitute value to a spontaneous value
- If a variable is configured as COV but is not contained in the notification, the I bit is set for the variable and a corresponding LOG message is created
- If no notification is received after a subscription, the I bit is set for the variable and a corresponding LOG message is created. (configured **APDU Timeout** * (configured **APDU Retries** +1))

LOG MESSAGES

Parameters	Description
Device '%s': Property (%d) not contained in COV notification for object '%s'.	A variable is configured as a COV, but not part of a notification.
Device '%s': No initial COV notification received for object '%s'.	After notification (subscription), no notification has been received.

7.2 Device Object of the driver

The **BACnetNG driver** is a BACnet standard compliant device from zenon 8.00.
Supported **BACnet Services**

- *DM-DDB-B* (Who-Is)
- *DM-DOB-B* (Who-HAS)

- *DS-RP-B* (Read-Property)
- *DS-RPM-B* (Read-Property multiple)

> **Information**
>
> You can find detailed information on this in the PICS (on page 154) chapter.

The following **BACnet Device-Object Properties** can be read by other **BACnet Devices**:

- *OBJECT_IDENTIFIER,*
- *OBJECT_NAME,*
- *OBJECT_TYPE,*
- *SYSTEM_STATUS,*
- *VENDOR_NAME,*
- *VENDOR_IDENTIFIER,*
- *MODEL_NAME,*
- *FIRMWARE_REVISION,*
- *APPLICATION_SOFTWARE_VERSION,*
- *PROTOCOL_VERSION,*
- *PROTOCOL_REVISION,*
- *PROTOCOL_SERVICES_SUPPORTED,*
- *PROTOCOL_OBJECT_TYPES_SUPPORTED,*
- *OBJECT_LIST,*
- *MAX_APDU_LENGTH_ACCEPTED,*
- *SEGMENTATION_SUPPORTED,*
- *MAX_SEGMENTS_ACCEPTED,*
- *APDU_SEGMENT_TIMEOUT,*
- *APDU_TIMEOUT,*
- *NUMBER_OF_APDU_RETRIES,*
- *DEVICE_ADDRESS_BINDING,*
- *DATABASE_REVISION,*
- *PROPERTY_LIST*

7.3 Mapping the BACnet status flags to the status bits of a zenon variable.

At reading the **PRESENT-VALUE** property the **STATUS-FLAG** is always also read. If the property **PRESENT-VALUE** is read via a variable, the property **STATUS-FLAGS** is mapped to the status bits of the variable as follows:

BACnet STATUS-FLAG	zenon Status bit
In alarm	-
Fault	INVALID
Overriden	INVALID
Out of service	SB-Bit

Attention

Mapping the status flag requires that the device used for communication supports BACnet service Read Property Multiple (DS-RPM-A).

7.4 Mapping of BACnet data types to string variables

The data of a BACnet property are transferred in **BACnet tags**. One property can contain one or more tags. A tag consists of a **Tag class**, **Tag number** and a value.

For the application tag class the data type comes off tag number of the encoded value unambiguously.

For tags with class *context -specific* the data type is additionally depended on the property and the data type. This means for decoding/encoding the value it is necessary that you have knowledge about the property and the object or the data type of the property.

Properties which can consist of an application tag can also be mapped to primitive types. All other properties can only be mapped to string variables.

CODING OF THE VALUES

Data Type	Tag number	Description
NULL	0	-
Boolean	1	FALSE (0): "FALSE" TRUE (1): "TRUE"
Unsigned	2	Unsigned decimal value.
Integer	3	Signed decimal value.
Real	4	Float.
Double	5	Float.
Bit string	8	Bit sequence (sequence of "1" respectively "0").
Octet string	6	Byte sequence (sequence of double-digit hexadecimal numbers).
Character string	7	Character sequence; If the string contains other characters beside the character sequence - e.g. Tag number - the character sequence is put inside quotation marks.
Enumerated	9	Unsigned decimal value.
Date	10	*YYY.mm.dd.tt* *YYY*: Year since 1900 *mm*: Month *dd*: Day *tt*: weekday) Value *255* in one of the fields means: *not specified*
Time	11	*hh:mm:ss:MMM* *hh*: Hour *mm*: Minute *ss*: Second *MMM*: 100/second Value *255* in one of the fields means: *not specified*
ObjectID	12	*xxxx yyyyy*

Data Type	Tag number	Description
		xxxx: Object type *yyyyy*: Object instance number
WeekNDay	context specific	*mm.ww.tt* *mm*: Month *ww*: Week of the month *tt*: Weekday Value *255* in one of the fields means: *not specified*

LISTS OR ARRAYS OF SIMPLE DATA TYPES

List or arrays of Boolean, Unsigned, Integer, Real, Double, Bit string, Octet string, Character string, Enumerated, Date, Time or ObjectID are displayed as sequence of the coded values. At this every value is put between face brackets '{' and '}' and the single values are separated by comma (,).

COMPLEX DATA TYPES

For complex data types (i.e. for context specific tags and all types which consist of more than one tag but are no lists or arrays of simple data types) the tag number is put in the sting beside the value of the tag. In this case a tag always consists of tag number and value. If the BACnet data type "[Raw]" was not selected in the variable configuration and the data type can be determined (possible for all known properties), the value is coded as shown in the table above. Otherwise the value is coded as a byte sequence (sequence of double-digit hexadecimal numbers).

Format of the tag number:

Tag class	Data type unknown or set type "[Raw]"	Data type known - coding of the value by means of the data type
Tag class:Application	Tag number within '%' and '%'	Tag number within brackets '[and ']'
Tag class:Context specific	Tag number within '#' and '#'	Tag number within parenthesis '(' and ')'

Tags which are referred to as constructed tags are tags which contain a tag sequence are put in faced brackets '{' and '}' and consist of the tag number followed by the enclosed tags which are separated by comma (,). The tag number is put between '<' and '>'.

Attention: For the correct decoding/encoding you must set the respective BACnet data type in the address setting of the variable.

FORMATTING EXAMPLE

NULL:

```
[0]
```

Boolean, BOOLEAN:

```
TRUE
FALSE
[1] TRUE
[1] FALSE
```

Date, DATE:

```
107.11.25.7
[10] 107.11.25.7
%10% 6D071603
```

Time, TIME:

```
5:00:00 PM:000
[11] 5:00:00 PM:000
```

Objekt ID, OBJECTIDENTIFIER:

```
0017 0000000
[12] 0017 0000000
```

Bitstring, BITSTRING:

```
010
[8] 010
```

Byte-String, OCTETSTRING:

```
6D071603
[5] 6D071603
```

Character sting, CHARACTERSTRING:

```
BACnet
[7] "BACnet"
```

Date, [CalendarEntry] with [WeekNDay]:

```
(2) 255.255.1
```

[DateRange] or date list/array:

```
{107.11.25.7},{109.12.28.255}
```

Constructed type ([SpecialEvent])

```
{<0> {(2) 1.1.1}},{<2> {[11] 17:00:00:000},{[1] TRUE}},{(3) 1}
```

7.5 Device Management

DRIVER COMMANDS

A command can be sent to a selected BacnetNG driver using the zenon function **Driver commands**. To do this, link the **driver command** *driver-specific command* to one of the following options as a text:

- DM-DCC-A Device Communication Control-A
- DM-RD-A Reinitialize Device-A
- DM-TS-A Time Synchronisation-A
- DM-UTC-A UTC Time Synchronisation-A

DM-DCC-A DEVICE COMMUNICATION CONTROL-A

Switches BACnet communication on or off.

- Syntax: **DeviceCommunicarionControl("<Device name>", <Communication On(1)/Off (0)>, <Time in seconds communication should remain off>,"<Password>")**

- Example: **DeviceCommunicationControl("BACnet_Device", 0, 5, "secret")**

DM-RD-A REINITIALIZE DEVICE-A

Reinitializes the device (reset).

- Syntax: **ReinitializeDevice("<Device name>", <start mode (<Warm> resp. <Cold>)>)**
- Example: **ReinitializeDevice("BACnet_Device", Cold)**

DM-TS-A TIME SYNCHRONIZATION-A

Sends a time synchronization broadcast with local time.

- Syntax: **TimeSynchronisation**

DM-UTC-A UTC TIME SYNCHRONIZATION-A

Sends a time synchronization broadcast with UTC time.

- Syntax: **UTCTimeSynchronisation**

COMMUNICATION TO FOREIGN DEVICE

Communication to a different sub network is, in accordance with the BACnet standard, implemented by means of a foreign device. The communication parameters for this are configured in the driver configuration dialog in the **Settings** (on page 17) tab in the *Foreign device registration* area.

If the **Register as foreign device** option is active in the **Settings** (on page 17) configuration dialog, registration as a foreign device is initially carried out on the BACnet Broadcast Management Device once the driver has started. If this registration fails (timeout and repetitions have expired, name cannot be resolved, send failure, for example), the status of all variables is set to I-bit (*invalid*). Another registration is only attempted after an error wait time.

After successful registration, the connection to all configured devices is established and BACnet communication starts. During the complete runtime environment, registration is renewed cyclically after expiry of the configured **Registration lifetime**.

7.6 Import

For the zenon variable that is created for **PRESENT-VALUE**, the following are set if present:

- **Min/Max value** property
- Decimal points according to the properties **min-/max-pres-value** and **resolution** of the BACnet object

ONLINE IMPORT FOR TRENDLOG VARIABLES

For Trendlog and *Trendlog Multiple* objects, (from zenon 7.60), *Property* variables are offered.

For Trendlog and *Trendlog Multiple* objects Trendlog variables are created.

RDA varibles of the driver object type **Trendlog**:
Note: The character . (dot) represents the seperators configured in the driver configuration (Tab Settings, property **Object name separator**)

- *<Device name>.<Object name>.LogStatus*
- *<Device name>.<Object name>.TimeChange*
- *<Device name>.<Object name>.Value*
  This is applicable for *Trendlog*
- *<Device name>.<Object name>.Value[n]*
  This is applicable for *Trendlog multiple*

Variables for the driver object type **Treiberobjekttyp Trendlog - Internal total record count** for the synchronisation of the sequence number for Redundancy:

- *<Device name>.<Object name>.internal_total_record_count*

7.7 Trendlog and trendlog multiple variables

The Bacnet property *Total_record count* is polled cyclically. This polling rate is configured in the driver configuration in the **Devices** tab in the **Trendlog poll interval** property.

Attention

If a **Property** driver object type variable has been created and signed into the driver (advised), the *Total_record_count* is read with the poling rate set for this variable. In this case, this property is ignored.

A LOG message is created for lost *Records*.

Information

The buffer for received Trendlogs is limited to 100 MB. As a result, historical *Records* can get lost if the LOG exceeds size of 100 MB.

LOG STATUS VALUES FOR TRENDLOG AND TRENDLOG MULTIPLE

LOG_DISABLED	BUFFER_PURGED	LOG_INTERRUPTED	Log status Value
FALSE	*FALSE*	*FALSE*	*0*
TRUE	*FALSE*	*FALSE*	*1*
FALSE	*TRUE*	*FALSE*	*2*
TRUE	*TRUE*	*FALSE*	*3*
FALSE	*FALSE*	*TRUE*	*4*
TRUE	*FALSE*	*TRUE*	*5*
FALSE	*TRUE*	*TRUE*	*6*
TRUE	*TRUE*	*TRUE*	*7*

7.7.1 Redundancy

ENGINEERING

In redundancy operation, for each *Trendlog*, a **Trendlog - internal total record count** driver object type variable must have been created. In addition, a self-allocation must have been configured for this variable.

> **Attention**
>
> Direct self-allocations are ignored by Runtime. It is therefore necessary to implement the self-allocation by means of an additional internal variable.
>
> Note the configuration example in the process

As a result of this allocation and the remanent image, the sequence number of the last-read *Record* is transferred to the Standby Server. After a switch, the (new) primary server starts by reading the *Trendlogs* with the last-transferred sequence number.

REDUNDANCY SWITCHING - READING TRENDLOGS

The following is applicable for the reading of a Trendlog variable after a server switch:

- The reading of the Trendlogs only starts when all configured variables of the *Trendlog* driver object type are registered with the driver (*advised*). All Trendlog variables must therefore be assigned an archive.
- A **Trendlog - internal total record count** driver object type variable must also be registered with the driver (*advised*) before reading of the Trendlog is started.

DATA SOURCE

Trendlogs are only read by the server and not by the Standby Server.

This means that, when configuring the variable property, **Read from Standby Server only** must not be activated!

8 Driver command function

The zenon **Driver commands** function is to influence drivers using zenon.
You can do the following with a driver command:

- Start
- Stop
- Shift a certain driver mode
- Instigate certain actions

Note: This chapter describes standard functions that are valid for most zenon drivers.
Not all functions described here are available for every driver. For example, a driver that does not, according to the data sheet, support a modem connection also does not have any modem functions.

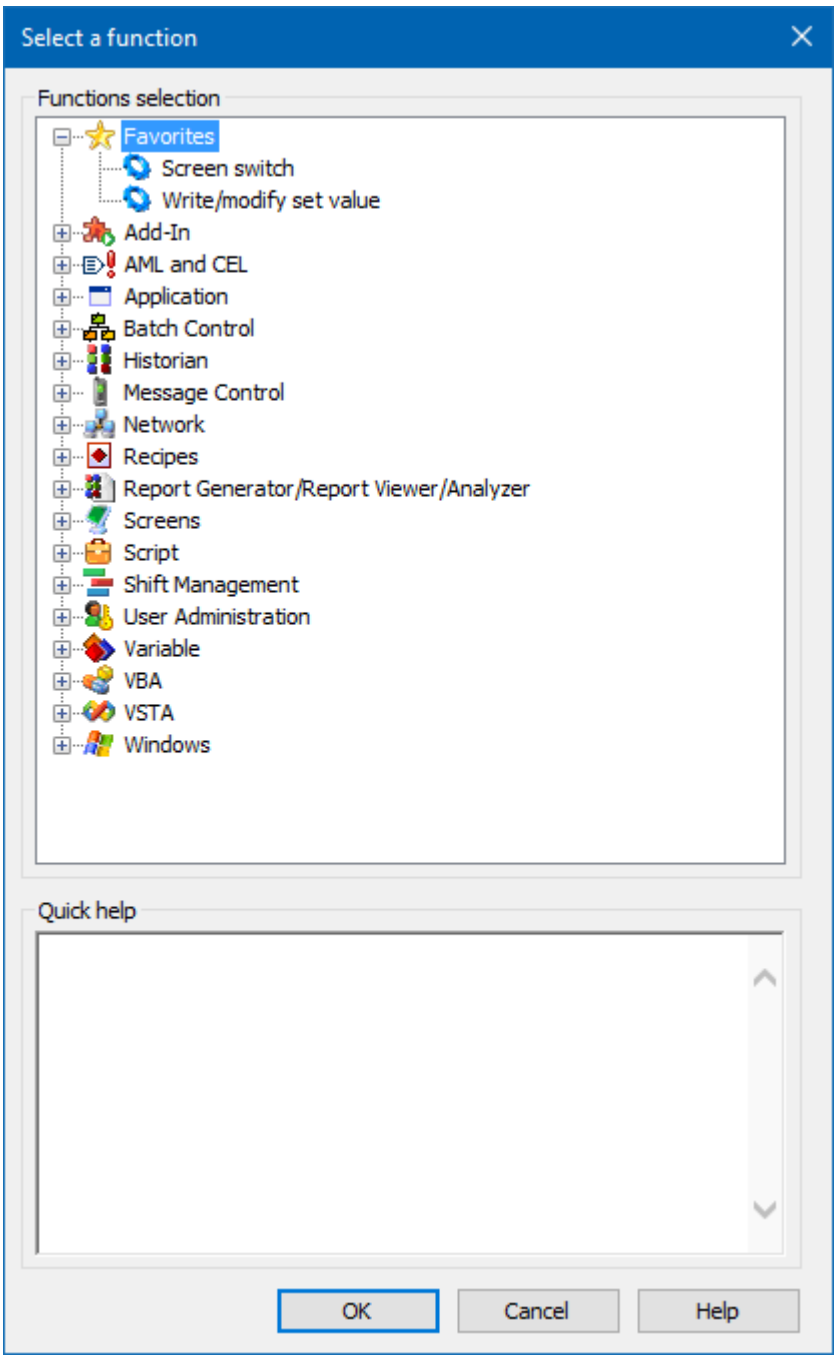
> **Attention**
>
> The zenon **Driver commands** function is not identical to driver commands that can be executed in the Runtime with Energy drivers!

CONFIGURATION OF THE FUNCTION

Configuration is carried out using the **Driver commands** function.
To configure the function:

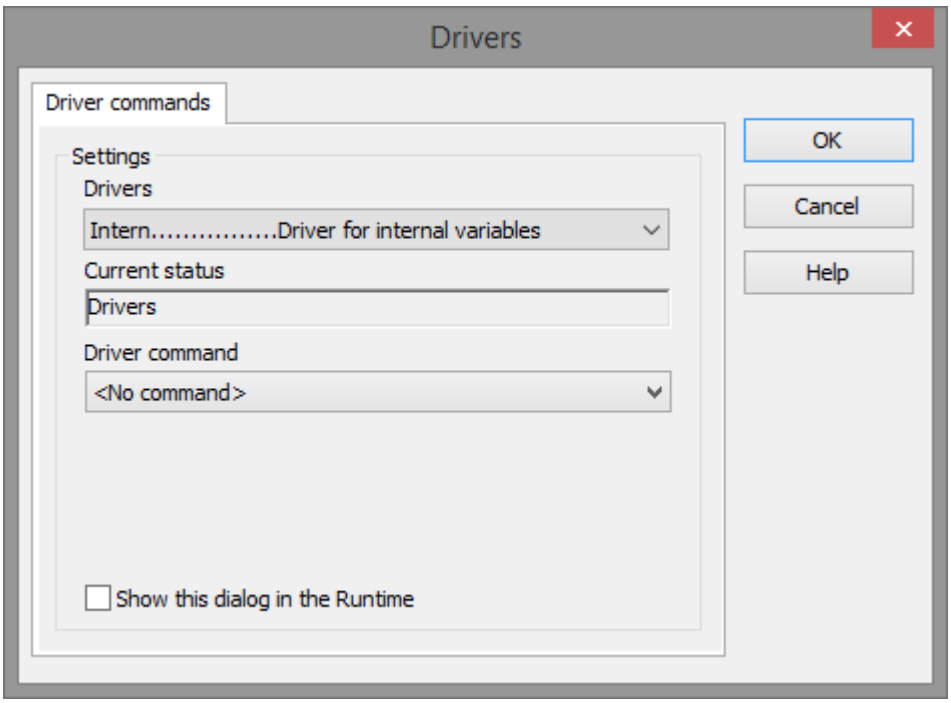
1. Create a new function in the zenon Editor.

The dialog for selecting a function is opened



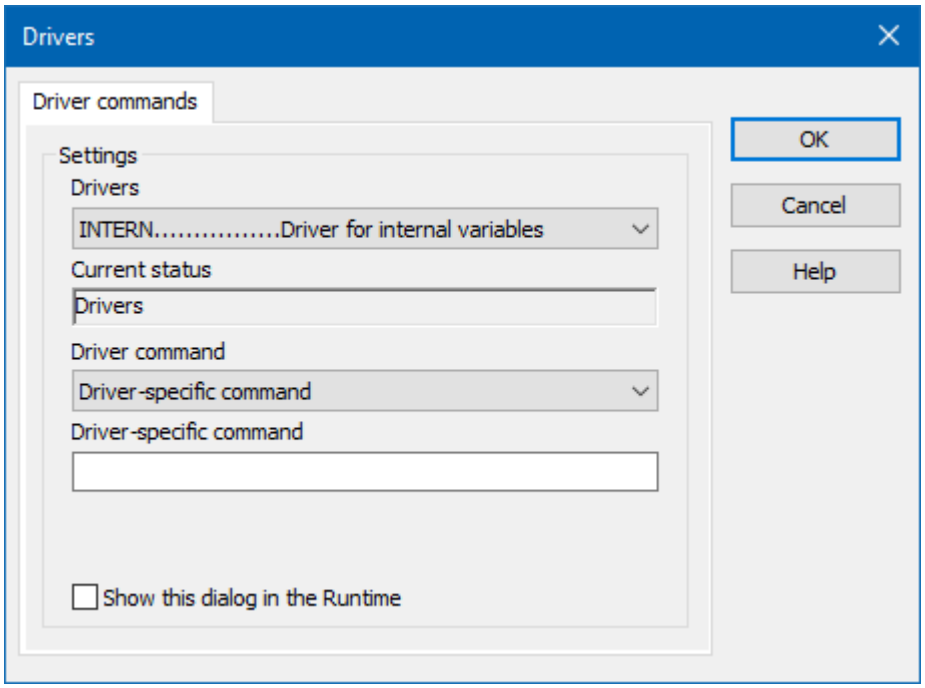
2. Navigate to the node **Variable.**
3. Select the **Driver commands** entry.

The dialog for configuration is opened



4. Select the desired driver and the required command.
5. Close the dialog by clicking on **OK** and ensure that the function is executed in the Runtime. Heed the notices in the **Driver command function in the network** section.

DRIVER COMMAND DIALOG



Option	Description
Driver	Selection of the driver from the drop-down list. It contains all drivers loaded in the project.
Current condition	Fixed entry that is set by the system. no function in the current version.
Driver command	no function in the current version. For details on the configurable driver commands, see the **available driver commands** section.
Driver-specific command	Entry of a command specific to the selected driver. **Note:** Only available if, for the **driver command** option, the *driver-specific command* has been selected.
Show this dialog in the Runtime	Configuration of whether the configuration can be changed in the Runtime: ▸ *Active*: This dialog is opened in the Runtime before executing the function. The configuration can thus still be changed in the Runtime before execution. ▸ *Inactive*: The Editor configuration is applied in the Runtime when executing the function. Default: *inactive*

CLOSE DIALOG

Options	Description
OK	Applies settings and closes the dialog.
Cancel	Discards all changes and closes the dialog.
Help	Opens online help.

AVAILABLE DRIVER COMMANDS

These driver commands are available - depending on the selected driver:

Driver command	Description
No command	No command is sent. A command that already exists can thus be removed from a configured function.

Driver command	Description
Start driver (online mode)	Driver is reinitialized and started. **Note:** If the driver has already been started, it must be stopped. Only then can the driver be re-initialized and started.
Stop driver (offline mode)	Driver is stopped. No new data is accepted. **Note:** If the driver is in offline mode, all variables that were created for this driver receive the status *switched off* (*OFF*; Bit *20*).
Driver in simulation mode	Driver is set into simulation mode. The values of all variables of the driver are simulated by the driver. No values from the connected hardware (e.g. PLC, bus system, ...) are displayed.
Driver in hardware mode	Driver is set into hardware mode. For the variables of the driver the values from the connected hardware (e.g. PLC, bus system, ...) are displayed.
Driver-specific command	Entry of a driver-specific command. Opens input field in order to enter a command.
Driver - activate set setpoint value	Write set value to a driver is possible.
Driver - deactivate set setpoint value	Write set value to a driver is prohibited.
Establish connecton with modem	Establish connection (for modem drivers) Opens the input fields for the hardware address and for the telephone number.
Disconnect from modem	Terminate connection (for modem drivers)
Driver in counting simulation mode	Driver is set into counting simulation mode. All values are initialized with *0* and incremented in the set update time by *1* each time up to the maximum value and then start at *0* again.
Driver in static simulation mode	No communication to the controller is established. All values are initialized with *0*.
Driver in programmed simulation mode	The values are calculated by a freely-programmable simulation project. The simulation project is created with the help of the zenon Logic Workbench and runs in the zenon Logic Runtime.

DRIVER COMMAND FUNCTION IN THE NETWORK

If the computer on which the **Driver commands** function is executed is part of the zenon network, further actions are also carried out:

- A special network command is sent from the computer to the project server.
  It then executes the desired action on its driver.
- In addition, the Server sends the same driver command to the project standby.
  The standby also carries out the action on its driver.

This makes sure that Server and Standby are synchronized. This only works if the Server and the Standby both have a working and independent connection to the hardware.

9 Error analysis

Should there be communication problems, this chapter will assist you in finding out the error.

9.1 Analysis tool

All zenon modules such as Editor, Runtime, drivers, etc. write messages to a joint log file. To display them correctly and clearly, use the Diagnosis Viewer program that was also installed with zenon. You can find it under **Start/All programs/zenon/Tools 8.20 -> Diagviewer.**

zenon driver log all errors in the LOG files.LOG files are text files with a special structure. The default folder for the LOG files is subfolder **LOG** in the folder **ProgramData**. For example:

%ProgramData%\COPA-DATA\LOG.

Attention: With the default settings, a driver only logs error information. With the Diagnosis Viewer you can enhance the diagnosis level for most of the drivers to "Debug" and "Deep Debug". With this the driver also logs all other important tasks and events.

In the Diagnosis Viewer you can also:

- Follow newly-created entries in real time
- customize the logging settings
- change the folder in which the LOG files are saved

Note:

1. The Diagnosis Viewer displays all entries in UTC (coordinated world time) and not in local time.

2. The Diagnosis Viewer does not display all columns of a LOG file per default. To display more columns activate property **Add all columns with entry** in the context menu of the column header.
3. If you only use **Error-Logging**, the problem description is in the column **Error text**. For other diagnosis level the description is in the column **General text**.
4. For communication problems many drivers also log error numbers which the PLC assigns to them. They are displayed in **Error text** or **Error code** or **Driver error parameter** (**1** and **2**). Hints on the meaning of error codes can be found in the driver documentation and the protocol/PLC description.
5. At the end of your test set back the diagnosis level from **Debug** or **Deep Debug**. At **Debug** and **Deep Debug** there are a great deal of data for logging which are saved to the hard drive and which can influence your system performance. They are still logged even after you close the Diagnosis Viewer.

> **Attention**
>
> In Windows CE errors are not logged per default due to performance reasons.

You can find further information on the Diagnosis Viewer in the Diagnose Viewer manual.

9.2 Driver monitoring

Runtime monitors the availability of the driver by means of a watchdog. If a driver is no longer available, the *INVALID* status bit is also set for all checked-in variables.

Possible causes for a triggering of the watchdog:

- The driver process is no longer running.

  Check whether the driver EXE file is still running in the Task Manager.
- Operating system is busy with processes that have a higher priority.

  Check the configuration of your system to see whether there is sufficient memory and CPU power. In this case, the driver only resets the *INVALID* status bit if there is a value change on the connected party. Static values retain the *INVALID* status bit until the next time the Runtime or the driver is started.

CONFIGURATION OF WATCHDOG

For the monitoring of communication in the Runtime, the connection to the driver is checked in a fixed, prescribed time period of 60 seconds. This process is repeated several times. If, within 5 attempts (= within 5 minutes), no valid connection to the driver is detected, the *INVALID* bit is set for the checked-in (*advised*) variables. In addition, the *INVALID* bit is also set when new variables are advised. The *INVALID* bit will no longer be reset.

Corresponding LOG entries are created for this.

LOG ENTRY

An error message is logged in the LOG when the watchdog is triggered:

Parameter	Description
Communication with driver:<drvExe>/<drvDesc>(id:<drvId>) timed out. No communication for <time> ms.	No communication with driver within the given time. ▸ *<time>*: Time (in milliseconds) ▸ *<drvDesc>*: Driver name ▸ *<drvExe>*: Driver EXE name ▸ *<drvId>*: Driver ID in the zenon project
Communication with %s timed out. Invalid-Bit will be set.	Communication to the *%s* driver could not be established after 5 attempts within 60 seconds. The *INVALID* bit is set for the variable.
Communication with %s timed out. Timeout happened %d times	Communication to the *%s* driver could not be established after *%d* times within 60 seconds.

9.3 BACnet Error codes

Description of BACnet error codes which are displayed in log messages.

REJECT REASON

0	Undefined (other)
1	Buffer overflow (buffer-overflow)
2	Inconsistent parameter (inconsistent-parameters)
3	Invalid parameter data type (invalid-parameter-data-type)
4	Invalid tag (invalid-tag)
5	Missing parameter (missing-required-parameter)
6	Parameter not within the allowed range of value (parameter-out-of-range)

0	Undefined (other)
7	Too many arguments (too-many-arguments)
8	Invalid value for enumeration (undefined-enumeration)
9	Unknown service (unrecognized-service)

ABORT REASON

0	Undefined (other)
1	Buffer overflow (buffer-overflow)
2	Invalid APDU at this time (invalid-apdu-in-this-state)
3	Preempted by task with higher priority (preempted-by-higher-priority-task)
4	Segmentation is not supported (segmentation-not-supported)

ERROR

ERROR CLASS

0	Device
1	Object
2	Property
3	Resource
4	Security
5	Services
6	VT

ERROR CODE

0	Undefined (other)
1	Authentication failed (authentication-failed)
2	Configuration in progress (configuration-in-progress)

0	Undefined (other)
3	Device busy (device-busy)
4	Dynamic creation not possible (dynamic-creation-not-supported)
5	File access denied (file-access-denied)
6	Incompatible security levels (incompatible-security-levels)
7	Inconsistent parameter (inconsistent-parameters)
8	Inconsistent selection criterion (inconsistent-selection-criterion)
9	Invalid data type (invalid-data-type)
10	Invalid file access method (invalid-file-access-method)
11	Invalid file start position (invalid-file-start-position)
12	Invalid operator name (invalid-operator-name)
13	Invalid parameter data type (invalid-parameter-data-type)
14	Invalid time stamp (invalid-time-stamp)
15	Generation of key failed (key-generation-error)
16	Missing parameter (missing-required-parameter)
17	No objects of specific types (no-objects-of-specified-type)
18	No space for object (no-space-for-object)
19	No space for adding a list element (no-space-to-add-list-element)
20	No space for writing a property (no-space-to-write-property)
21	No VT session available (no-vt-sessions-available)
22	Property is not a list (property-is-not-a-list)
23	Object deletion no permitted (object-deletion-not-permitted)
24	Object ID already exists (object-identifier-already-exists)
25	Operational problem (operational-problem)
26	Password failure (password-failure)
27	Read access denied (read-access-denied)
28	Security not supported (security-not-supported)

0	Undefined (other)
29	Service request denied (service-request-denied)
30	Timeout (timeout)
31	Unknown object (unknown-object)
32	Unknown property (unknown-property)
33	-
34	Unknown VT class (unknown-vt-class)
35	Unknown VT session (unknown-vt-class)
36	Unsupported object type (unsupported-object-type)
37	Value not within the allowed range (value-out-of-range)
38	VT session already closed (vt-session-already-closed)
39	Failure while closing VT session (vt-session-termination-failure)
40	Write access denied (write-access-denied)
41	Character set not supported (character-set-not-supported)
42	Invalid array index (invalid-array-index)
43	COV subscription failed (cov-subscription-failed)
44	No COV property (not-cov-property)
45	Optional functionality not supported (optional-functionality-not-supported)
46	Invalid configuration data (invalid-configuration-data)

9.4 Check list

Checks after communication errors:

- Is the PLC connected to the power supply?
- Are the participants available in the **IP** network?
- Can the PLC be reached via the **Ping** command?
- Was the device name set correctly in both the driver dialog and at the variable?
- Does the property separator of the variable match the set separator in the driver dialog?

- Did you use the right object type for the variable?
- When communicating over COV subscriptions: Can the selected property by read over COV?
- Analysis with the Diagnosis Viewer: Which messages are displayed?

10 PICS - Protocol Implementation Conformance Statement

Date:	Feb 12, 2018
Vendor Name:	Ing. Punzenberger COPA-DATA GmbH
Product Name:	BACnet NG driver -Driver for process control system (HMI/SCADA)
Product Model Number:	n.a.
Applications Software Version:	8.00
Firmware Revision:	n.a.
BACnet protocol Revision:	1.14

PRODUCT DESCRIPTION:

The **BACnet NG driver** enables the SCADA runtime to use data sharing, scheduling, trend and alarming services of BACnet/IP capable devices.

BACNET STANDARDIZED DEVICE PROFILE (ANNEX L):

[x]	BACnet Operator Workstation (B-OWS)
[ ]	BACnet Building Controller (B-BC)
[ ]	BACnet Advanced Application Controller (B-AAC)
[ ]	BACnet Application Specific Controller (B-ASC)
[ ]	BACnet Smart Sensor (B-SS)

[x]	BACnet Operator Workstation (B-OWS)
[ ]	BACnet Smart Actuator (B-SA)

LIST OF ALL BACNET INTEROPERABILITY BUILDING BLOCKS SUPPORTED (ANNEX K):

DATA SHARING

[x]	Data Sharing – Read Property-A	DS-RP-A
[x]	Data Sharing – Read Property-B	DS-RP-B
[x]	Data Sharing – Read Property Multiple-A	DS-RPM-A
[x]	Data Sharing – Read Property Multiple-B	DS-RPM-B
[x]	Data Sharing – Write Property-A	DS-WP-A
[ ]	Data Sharing – Write Property-B	DS-WP-B
[x]	Data Sharing – Write Property Multiple-A	DS-WPM-A
[ ]	Data Sharing – Write Property Multiple-B	DS-WPM-B
[x]	Data Sharing – Change of Value -A	DS-COV-A
[ ]	Data Sharing – Change of Value -B	DS-COV-B
[ ]	Data Sharing – Change of Value Property -A	DS-COVP-A
[ ]	Data Sharing – Change of Value Property -B	DS-COVP-B
[x]	Data Sharing – Change of Value-Unsolicited-A	DS-COVU-A
[ ]	Data Sharing – Change of Value-Unsolicited-B	DS-COVU-B
[x]	Data Sharing – View-A	DS-V-A
[x]	Data Sharing – Advanced View-A	DS-AV-A
[x]	Data Sharing – Modify-A	DS-M-A

[x]	Data Sharing – Read Property-A	DS-RP-A
[x]	Data Sharing – Advanced Modify-A	DS-AM-A

ALARM AND EVENT MANAGEMENT

[x]	Alarm and Event – Notification-A	AE-N-A
[ ]	Alarm and Event – Notification Internal-B	AE-N-I-B
[ ]	Alarm and Event – Notification External-B	AE-N-E-B
[x]	Alarm and Event – ACK-A	AE-ACK-A
[ ]	Alarm and Event – ACK-B	AE-ACK-B
[ ]	Alarm and Event – Alarm Summary-B	AE-ASUM-B
[ ]	Alarm and Event – Enrollment Summary-B	AE-ESUM-B
[ ]	Alarm and Event – Information-B	AE-INFO-B
[ ]	Alarm and Event – Life Safety-A	AE-LS-A
[ ]	Alarm and Event – Life Safety-B	AE-LS-B
[x]	Alarm and Event – View Notifications-A	AE-VN-A
[x]	Alarm and Event – Advanced View Notifications-A	AE-AVN-A
[x]	Alarm and Event – View Modify-A	AE-VM-A
[ ]	Alarm and Event – Advanced View and Modify-A	AE-AVM-A
[x]	Alarm and Event – Alarm Summary View-A	AE-AS-A
[ ]	Alarm and Event – Event Log View-A	AE-ELV-A
[ ]	Alarm and Event – Event Log View and Modify-A	AE-ELVM-A

[x]	Alarm and Event – Notification-A	AE-N-A
[ ]	Alarm and Event – Event Log Internal-B	AE-EL-I-B
[ ]	Alarm and Event – Event Log External-B	AE-EL-E-B
[ ]	Alarm and Event – Alarm Summary-A	AE-ASUM-A
[ ]	Alarm and Event – Enrollment Summary-A	AE-ESUM-A
[x]	Alarm and Event – Information-A	AE-INFO-A

SCHEDULING

[ ]	Scheduling – Internal-B	SCHED-I-B
[ ]	Scheduling – External-B	SCHED-E-B
[ ]	Scheduling – Advanced View Modify-A	SCHED-AVM-A
[x]	Scheduling – View Modify-A	SCHED-VM-A
[x]	Scheduling – Weekly Schedule-A	SCHED-WS-A
[ ]	Scheduling – Weekly Schedule Internal-B	SCHED-WS-I-B
[ ]	Scheduling – Readable-B	SCHED-R-B
[ ]	Scheduling – A (Deprecated BIBB)	SCHED-A

TRENDING

[ ]	Trending – Viewing and Modifying Trends-A	T-VMT-A
[ ]	Trending – Viewing and Modifying Internal-B	T-VMT-I-B
[	Trending – Viewing and Modifying External-B	T-VMT-E-B

[ ]	Trending – Viewing and Modifying Trends-A	T-VMT-A
]		
[ ]	Trending – Viewing and Modifying Multiple Values-A	T-VMMV-A
[ ]	Trending – Viewing and Modifying Multiple Values Internal-B	T-VMMV-I-B
[ ]	Trending – Viewing and Modifying Multiple Values External -B	T-VMMV-E-B
[ ]	Trending – Automated Multiple Value Retrieval-A	T-AMVR-A
[ ]	Trending – Automated Multiple Value Retrieval-B	T-AMVR-B
[x]	Trending – View-A	T-V-A
[ ]	Trending – Advanced View and Modify-A	T-AVM-A
[ ]	Trending – Archival-A	T-A-A
[ ]	Trending – Automated Trend Retrieval-A	T-ATR-A
[ ]	Trending – Automated Trend Retrieval-B	T-ATR-B
[ ]	Trending – Viewing and Modifying Trends-A (Deprecated BIBB)	T-VMT-A
[ ]	Trending – Viewing and Modifying Multiple Values-A (Deprecated BIBB)	T-VMMV-A

DEVICE MANAGEMENT

[x]	Device Management – Dynamic Device Binding-A	DM-DDB-A
[x]	Device Management – Dynamic Device Binding-B	DM-DDB-B
[x]	Device Management – Dynamic Object Binding-A	DM-DOB-A
[x]	Device Management – Dynamic Object Binding-B	DM-DOB-B

[x]	Device Management – Dynamic Device Binding-A	DM-DDB-A
[x]	Device Management – Device Communication Control-A	DM-DCC-A
[ ]	Device Management – Device Communication Control-B	DM-DCC-B
[ ]	Device Management – Text Message-A	DM-TM-A
[ ]	Device Management – Text Message-B	DM-TM-B
[x]	Device Management – Time Synchronization-A	DM-TS-A
[ ]	Device Management – Time Synchronization-B	DM-TS-B
[x]	Device Management – UTC Time Synchronization-A	DM-UTC-A
[ ]	Device Management – UTC Time Synchronization-B	DM-UTC-B
[x]	Device Management – Reinitialize Device-A	DM-RD-A
[ ]	Device Management – Reinitialize Device-B	DM-RD-B
[ ]	Device Management – Backup and Restore-A	DM-BR-A
[ ]	Device Management – Backup and Restore-B	DM-BR-B
[ ]	Device Management – Restart-A	DM-R-A
[ ]	Device Management – Restart-B	DM-R-B
[ ]	Device Management – List Manipulation-A	DM-LM-A
[ ]	Device Management – List Manipulation-B	DM-LM-B
[ ]	Device Management – Object Creation and Deletion-A	DM-OCD-A
[	Device Management – Object Creation and Deletion-B	DM-OCD-B

[x]	Device Management – Dynamic Device Binding-A	DM-DDB-A
]		
[ ]	Device Management – Virtual Terminal-A	DM-VT-A
[ ]	Device Management – Virtual Terminal-B	DM-VT-B
[ ]	Device Management – Automatic Network Mapping-A	DM-ANM-A
[ ]	Device Management – Automatic Device Mapping-A	DM-ADM-A
[ ]	Device Management – Automatic Time Synchronization-A	DM-ATS-A
[x]	Device Management – Manual Time Synchronization-A	DM-MTS-A
[ ]	Device Management – Private Transfer-A	DM-PT-A
[ ]	Device Management – Private Transfer-B	DM-PT-B

STANDARD OBJECT TYPES SUPPORTED:

The driver does not impose any restrictions on properties and object accessed from a BACnet device. Among vendor specific object types the listed standard object types are supported by accessing the addressable properties from other BACnet device's objects:

Object type	Object type supported	Dynamically creatable and deletable	Addressable properties	Writeable properties
Analog Input	+	-	all	all
Analog Output	+	-	all	all
Analog Value	+	-	all	all
Averaging	+	-	all	all
Binary Input	+	-	all	all
Binary Output	+	-	all	all

Object type	Object type supported	Dynamically creatable and deletable	Addressable properties	Writeable properties
Binary Value	+	-	all	all
Calendar	+	-	all	all
Command	+	-	all	all
Device	+	-	all	all
Event Enrollment	+	-	all	all
File	+	-	all	all
Group	+	-	all	all
Life Safety Point	+	-	all	all
Life Safety Zone	+	-	all	all
Loop	+	-	all	all
Multi-state Input	+	-	all	all
Multi-state Output	+	-	all	all
Multi-state Value	+	-	all	all
Notification Class	+	-	all	all
Program	+	-	all	all
Schedule	+	-	all	all
Trend Log	+	-	all	all
Accumulator	+	-	all	all
Pulse Converter	+	-	all	all
EventLog	+	-	all	all
Global Group	+	-	all	all
TrendLog Multiple	+	-	all	all
Load Control	+	-	all	all
Structured View	+	-	all	all

Object type	Object type supported	Dynamically creatable and deletable	Addressable properties	Writeable properties
Network Security	+	-	all	all
BitString Value	+	-	all	all
CharacterString Value	+	-	all	all
DatePattern Value	+	-	all	all
Date Value	+	-	all	all
DateTime Pattern Value	+	-	all	all
DateTime Value	+	-	all	all
Integer Value	+	-	all	all
Large Analog Value	+	-	all	all
OctetString Value	+	-	all	all
Positive Integer Value	+	-	all	all
TimePattern Value	+	-	all	all
Time Value	+	-	all	all
Notification Forwarder	+	-	all	all
Alert Enrollment	+	-	all	all
Channel	+	-	all	all
Lighting Output	+	-	all	all

SEGMENTATION CAPABILITY:

X	Segmented requests supported	Window Size: any
X	Segmented responses supported	Window Size: any

DATA LINK LAYER

X	BACnet/IP, (Annex J)
X	BACnet/IP, (Annex J), Foreign Device
	ISO 8802-2, Ethernet (Clause 7)
	ASTM 878.1, 2.5Mb. ARCNET (Clause 8)
	ASTM 878.1, RS485 ARCNET (Clause 8), baud rate(s): _____
	MS/TP master (Clause 9), baud rate(s): _____
	MS/TP slave (Clause 9), baud rate(s): _____
	Point-To-Point, EIA 232 (Clause 10), baud rate(s): _____
	Point-To-Point, modem (Clause 10), baud rate(s): _____
	LonTalk, (Clause 11), medium: _____
	Other _____

DEVICE ADDRESS BINDING:

Is static device binding supported? (This is currently necessary for two-way communication with MS/TP slaves and certain other devices.)

	Yes
X	No

CHARACTER SETS SUPPORTED

X	UTF-8
	IBM /Microsoft DBCS
X	**ISO 8859-1**
X	**ISO 10646 (ICS-4)**
X	**ISO 10646 (UCS2)**
	JIS C 6226

11 Communication with the PLC

1. If the address cannot be configured manually, a who-has broadcast which includes the name of the control is used to detect the object ID of the control and its address.

   If the address can be configured manually, a who-has unicast with the name of the control to the stated address is used to detect the object ID of the control.

2. Via ReadPropertyMultiple the communication parameter of the control are read (maximum APDU size, support for segmentation, maximum number of segments, APDU timeout and APDU segment timeout). If the automatic configuration of the timeouts is active, the read timeouts are applied for the further communication.

3. When browsing, the individual items of the object list properties of the device object are read using ReadProperty and ReadPropertyMultiple.

4. For the object of all created variables the object ID is detected via the object name with the help of unicast who-has request.

   At this only as many who-has request are allowed as the number of maximum simultaneous requests.

5. The COV subscriptions are renewed cyclically or the polled properties are read via ReadProperty and ReadPropertyMultiple (PRESENT-VALUE with STATUS-FLAGS).

6. Alarms: GetAlarmInfo; reading of the events by means of ReadPropertyMultiple, reading of the initial time stamp and states

7. Receiving the event notifications

8. In the event of an error, all open requests are aborted and the set error waiting time is waited. It is then restarted with 1).