zenon driver manual

MATHDR32

v.7.20

Contents

1. Welcome to COPA-DATA help

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 (mailto:documentation@copadata.com).

PROJECT SUPPORT

You can receive support for any real project you may have from our Support Team, who you can contact via email at support@copadata.com (mailto: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 (mailto:sales@copadata.com).

2. MATHDR32

The 7.20 mathematics driver is used for defining formulas and for calculating them by using data from other process drivers.

> **Information**
>
> *The mathematics driver does not use any I/Os.*

If a counter variable of the mathematics driver is saved in an archive where saves are carried out in the event of a value change (spontaneous), this can lead to the saving of very large amounts of data.

Workaround: Assign the mathematics variable a simulation variable and save the simulation variable in the archive.
Attention: The simulation variable must be of the `INT` data type (not `REAL`).

Information

The value of the mathematic driver is saved when Runtime is ended normally. To ensure that values are saved if limits or rema conditions where the counter runs become inactive:

Carry out the ***Save AML and CEL ring buffer*** *(main.chm::/11253.htm) or the* ***Save remanent data*** *(main.chm::/25971.htm) function. In addition, you can also still save the values cyclically (every hour, for example).*

3. MATHDR32 - Data sheet

General:	
Driver file name	MATHDR32.exe
Driver name	Driver for Mathematics variabl
PLC types	-
PLC manufacturer	zenon system driver; Internal driver;

Driver supports:	
Protocol	unknown;
Addressing: Address-based	-
Addressing: Name-based	x
Spontaneous communication	-
Polling communication	x
Online browsing	-
Offline browsing	-
Real-time capable	-
Blockwrite	-
Modem capable	-
Serial logging	-
RDA numerical	-
RDA String	-

Requirements:	
Hardware PC	-
Software PC	-
Hardware PLC	-
Software PLC	-
Requires v-dll	-

Platforms:	
Operating systems	Windows CE 6.0, Embedded Compact 7; Windows 7, 8, 8.1 Server 2008R2, Server 2012, Server 2012R2;
CE platforms	x86; ARM;

4. Driver history

Date	Driver version	Change
07.07.08	400	Created driver documentation

DRIVER VERSIONING

The versioning of the drivers was changed with zenon 7.10. There is a cross-version build number as of this version. This is the number in the 4th position of the file version,
For example: `7.10.0.4228` means: The driver is for version `7.10` service pack `0`, and has the build number `4228`.

Expansions or error rectifications will be incorporated into a build in the future and are then available form the next consecutive build number.

Example

A driver extension was implemented in build `4228`. The driver that you are using is build number `8322`. Because the build number of your driver is higher than the build number of the extension, the extension is included. The version number of the driver (the first three digits of the file version) do not have any significance in relation to this. The drivers are version-agnostic

5. Configuration

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

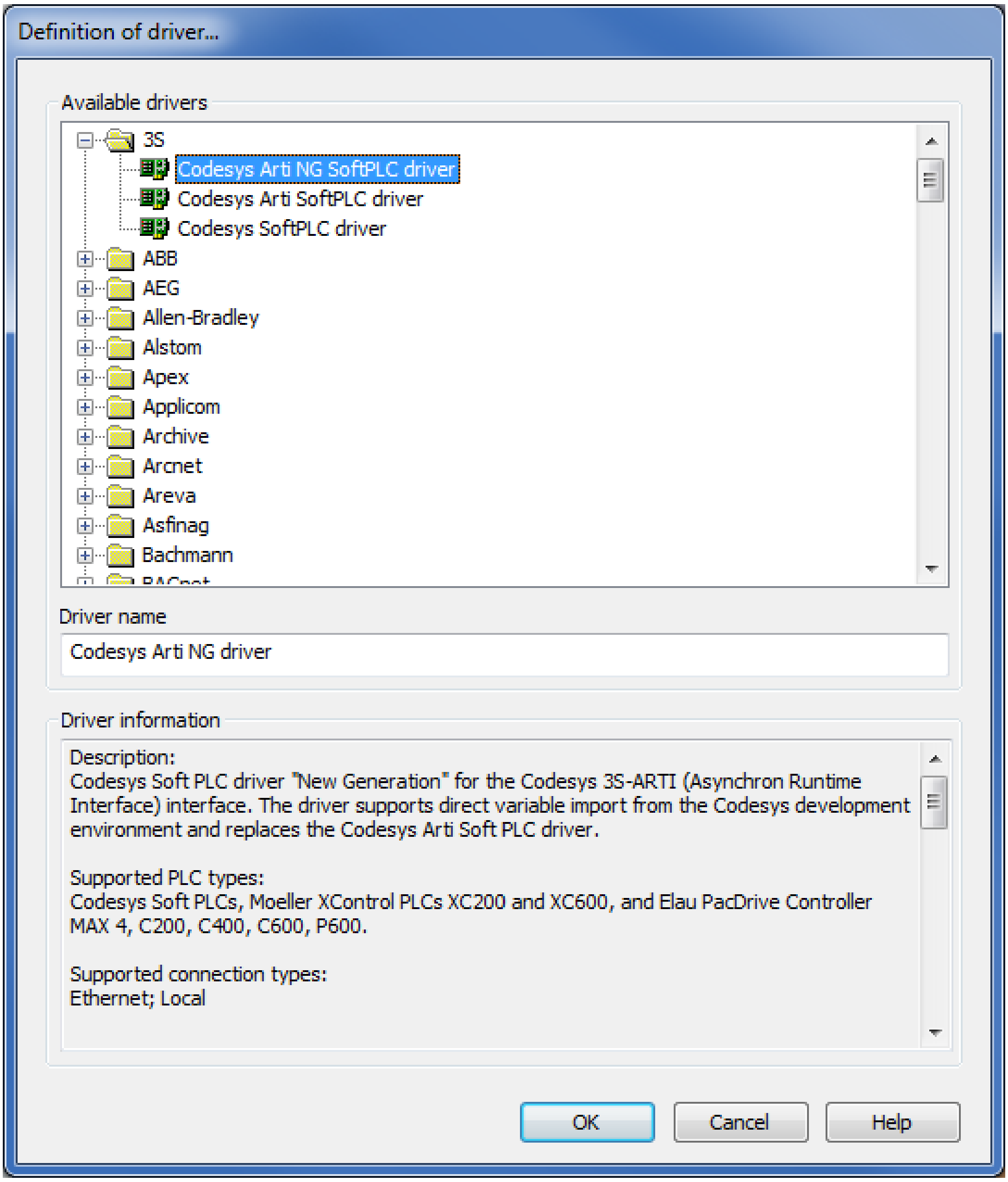
Information

Find out more about further settings for zenon variables in the chapter Variables (main.chm::/15247.htm) of the online manual.

5.1 Creating a driver

In order to create a new driver:

1. Right-click on `Driver` in the Project Manage and select `Driver new` in the context menu.
2. In the following dialog the control system offers a list of all available drivers.



3. Select the desired driver and give it a name:
   - The driver name has to be unique, i.e. if one and the same driver is to be used several times in one project, a new name has to be given each time.
   - 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 with OK. In the following dialog the single configurations of the drivers are defined.

Only the respective required drivers need to be loaded for a project. Later loading of an additional driver is possible without problems.

Information

For new projects and for existing projects which are converted to version 6.21 or higher, the following drivers are created automatically:

- Internal
- MathDr32
- SysDrv.

6. Creating variables

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

6.1 Creating variables in the Editor

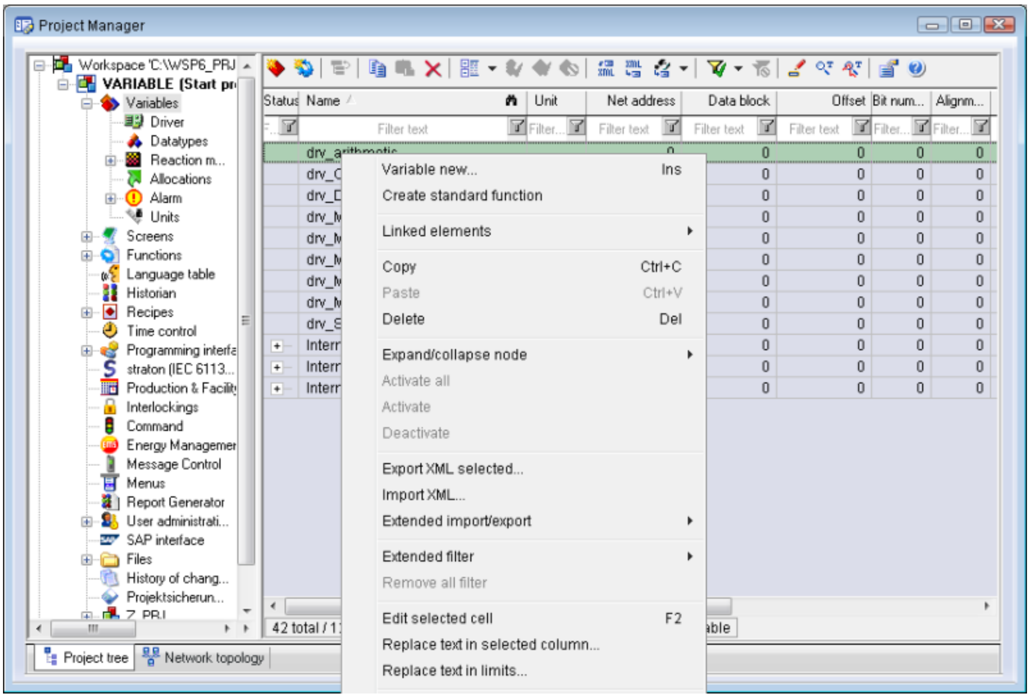
Variables can be created:

- as simple variables
- in arrays (main.chm::/15262.htm)
- as structure variables (main.chm::/15278.htm)

VARIABLE DIALOG

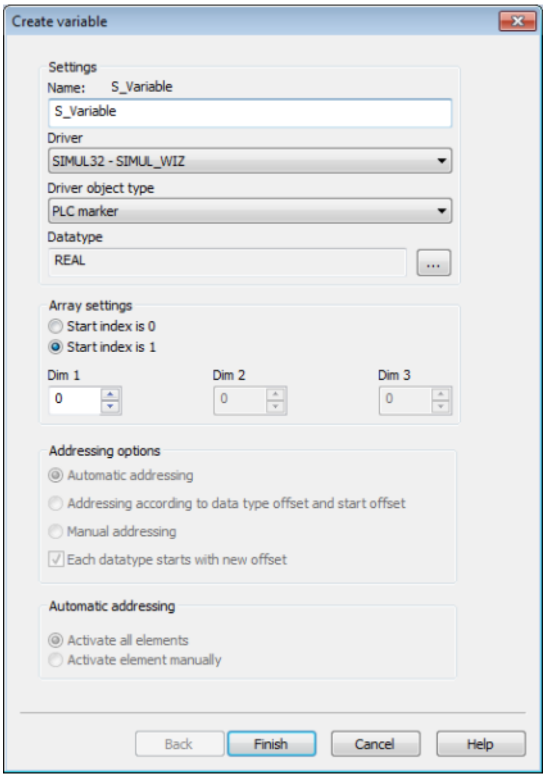
To create a new variable, regardless of which type:

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



2. The dialog for configuring variables is opened
3. configure the variable

4. The settings that are possible depends on the type of variables



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 Zeichen **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: For some drivers, the addressing is possible over the property **Symbolic address**, as well.
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 (Main.chm::/Intern.chm::/Intern.htm)) is automatically loaded.
Driver object type (cti.chm::/28685.htm)	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 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.

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

MATHEMATICS VARIABLES

The mathematics driver is used for defining formulas (linkage type of arithmetics, trigonometry, etc.) and calculating them by using the data from other process drivers. From the point of view of zenon, the Mathematics driver is a "normal" process driver. Variables are defined in the Editor (mathematics variable). The calculated values of the Mathematics variables are provided in the online operation. In order to define and use Mathematics variables, the Mathematics driver must be loaded first.

A formula must be defined for each mathematical variable of the Mathematics driver. Each formula must have at least one source variable. Any previously defined variable (even previously defined Mathematics variables) can be used as a source variable. The link is made with the signal resolution (technical values) of the variables.

Attention

You can configure a maximum of 4096 Mathematics variables with formulas. If you use more formulas, only the first 4096 will be executed during Runtime. Any further formulas will be ignored.

You can sum up a maximum of 4096 numerical constants among all Mathematics formulas. Any further constants will be assumed as 0 during Runtime.

If you require a larger number of variables/constants, we recommend the use of zenon Logic.

Hint: *zenon Logic is already included in zenon. In case zenon Logic is not included in your license, please contact the distributor that is responsible for you.*

Information

Up to 99 variables can be linked in one formula. X01 to X99. The length of the formula must not exceed 4096 characters.

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 TYPES AND SUPPORTED IEC DATA TYPES FOR PROCESS VARIABLES IN ZENON

Driver object type	Channel type	Read / Write	Supported data types	Comment
Formula	17	R	BOOL, LREAL	

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

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.

MAPPING OF THE DATA TYPES FROM THE PLC TO ZENON DATA TYPES

Control	zenon	Data type
BOOL	BOOL	8
LREAL	LREAL	5

Data type: The property **Data type** is the internal numerical name 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 (main.chm::/13028.htm) manual in the Variables (main.chm::/13045.htm) section.

6.4.1 XML import

For the import/export of variables the following is true:

- The import/export must not be started from the global project.
- The start takes place via:
  - Context menu of variables or data typ in the project tree
  - or context menu of a variable or a data type
  - or symbol in the symbol bar variables

Attention

When importing/overwriting an existing data type, all variables based on the existing data type are changed.

Example:

There is a data type XYZ derived from the type `INT`*with variables based on this data type. The XML file to be imported also contains a data type with the name XYZ but derived from type* `STRING`*. If this data type is imported, the existing data type is overwritten and the type of all variables based on it is adjusted. I.e. the variables are now no longer* `INT` *variables, but* `STRING` *variables.*

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 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 export 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 there 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

Description	Type	Field size	Comment
KANALNAME	Char	128	Variable name. The length can be limited using the MAX_LAENGE entry in **project.ini**.
KANAL_R	C	128	The original name of a variable that is to be replaced by the new name entered under "VARIABLENNAME" (field/column must be entered manually). The length can be limited using the MAX_LAENGE entry in **project.ini**.
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 **project.ini**.
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	Bus address
BAUSTEIN	N	3	Datablock address (only for variables from the data area of the PLC)
ADDRESS	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 Recipe Group Manager
LES_SCHR	R	1	Write-Read-Authorization

			0: Not allowed to set value. 1: Allowed to set value.
MIT_ZEIT	R	1	time stamp in zenon 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
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 DEFINITION

Limit definition for limit values 1 to 4, and status 1 bis 4:

Description	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 (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
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 text
A_DELAY1	N	10	Time delay
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.

6.5 Driver variables

The driver kit implements a number of driver variables. These are divided into:

- Information
- Configuration
- Statistics and
- Error message

The definitions of the variables defined in the driver kit are available in the import file **drvvar.dbf** (on the CD in the directory: `CD_Drive:/Predefined/Variables`) and can be imported from there.

Note: Variable names must be unique in zenon. If driver variables are to be imported from **drvvar.dbf** again, the variables that were imported beforehand must be renamed.

Information

Not every driver supports all driver variants.

For example:

- Variables for modem information are only supported by modem-compatible drivers
- Driver variables for the polling cycle only for pure polling drivers
- Connection-related information such as ErrorMSG only 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
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 status of Runtime for driver simulation.

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
ComPort	UINT	9	Number of the serial interface.
Baud rate	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.
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.
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

Updating: The driver update time is `500 ms`.

Attention

If a counter variable of the mathematics driver is saved in an archive for which `saving is carried out in the event of a value change` (spontaneous), this can lead to very large amounts of data being saved.

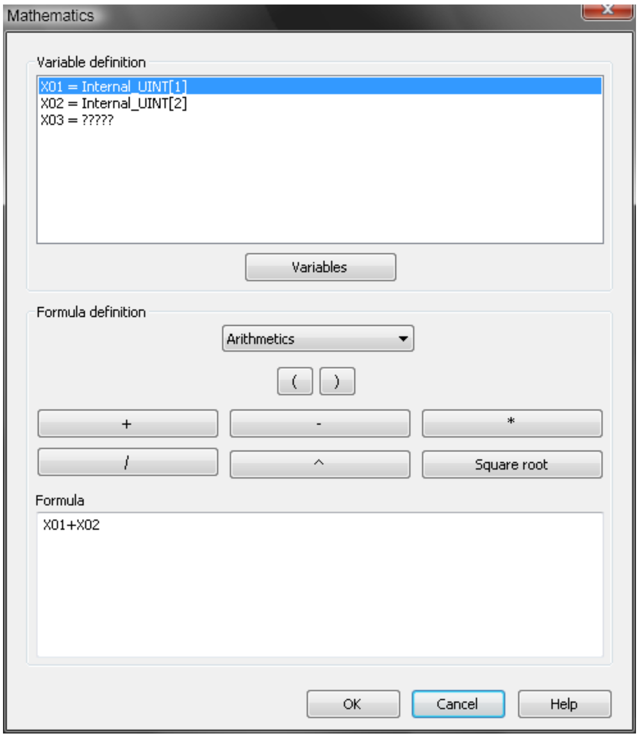
Workaround: Assign the mathematics variable a simulation variable and save the simulation variable in the archive.
Attention: The simulation variable must be of the `INT` data type (not `REAL`).

The driver supports the following functions:

7.1 Formulas

To enter formulas

- there must be a variable of the type MATHDR32 Mathematics driver
- open the node **Value calculation** in the properties of the variable
- click into the empty field next to the property **Formula**
- the dialog for entering formulas opens



Formulas can be entered as Float formulas (on page 36) or as Bit formulas (on page 36):

- Float formulas:
  - Direct input in the section **Formulas**

    or
  - Select a variable in the window **Variable definition** and select a formula type by clicking on the corresponding buttons for functions in the section **Formula definition**.
- Bit formulas:
  Direct input in the section **Formulas** after selecting **Boolean algebra**. If the variable was created as a binary variable, you will only be able to use **Boolean algebra**.

> **Information**
>
> *Up to 99 variables can be linked in one formula. X01 to X99. The length of the formula must not exceed 4096 characters.*

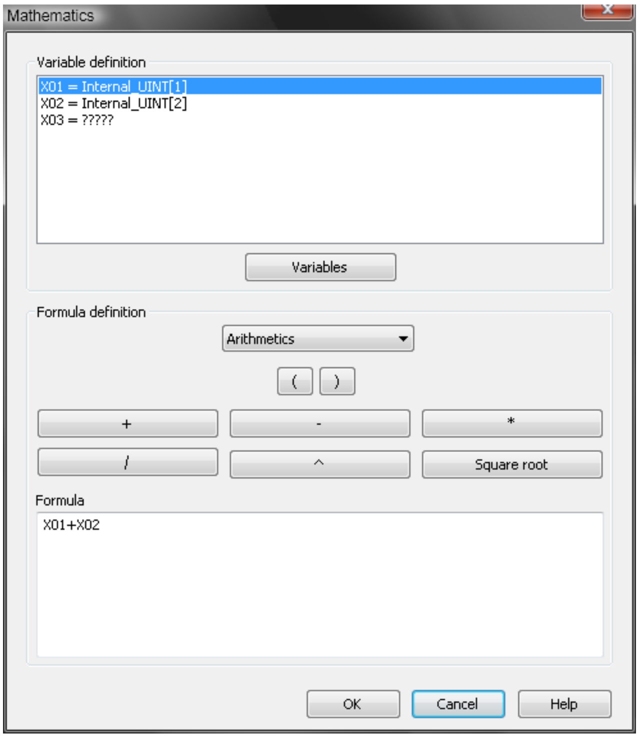
Note: When selecting a function, the necessary parentheses are created automatically and the cursor is placed between the parentheses.

Attention

Avoid recursive calls, such as a `statistic`*-type mathematic variable, which counts itself in* **`X01`**. *Recursive calls can lead to the whole system becoming unstable.*

7.1.1 Assign variable

Source variables for formula input are displayed in the field **`Variable definition`** .



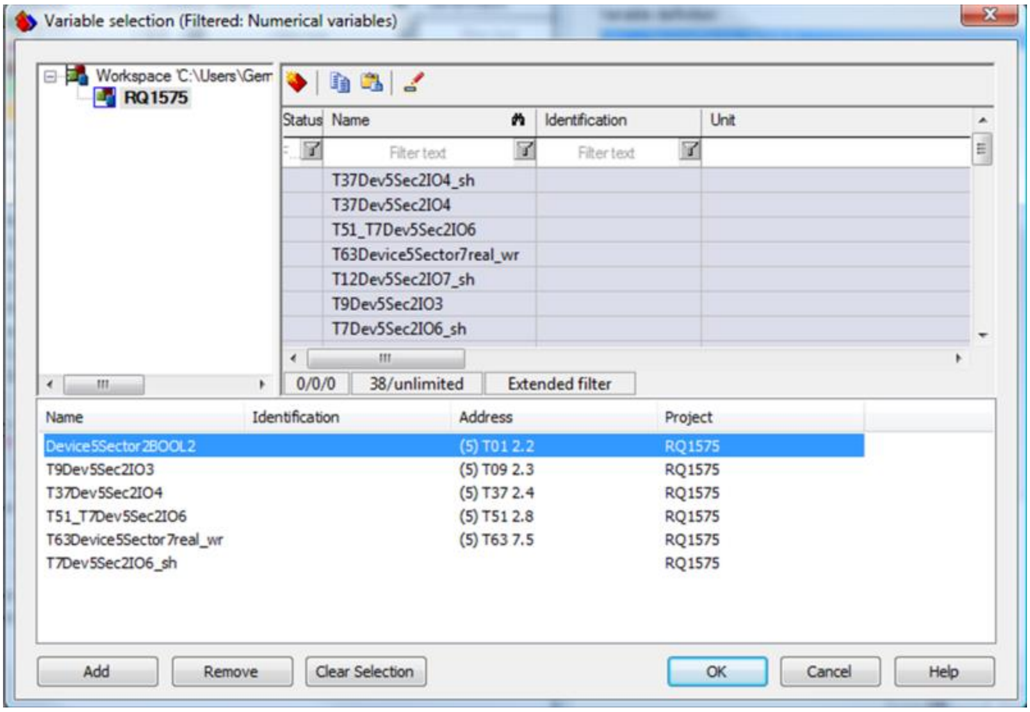
To add a new variable, change an existing variable or to delete a variable, open the variable selection:

1. Click on the button Variables

   or

   double-click on the last consecutive number with a question sign as the definition set in the field **`Variable definition`**

2. the dialog for selecting variables will be opened
3. Add new variables or delete existing ones from the list


Variable selection (Filtered: Numerical variables)
Workspace 'C:\Users\Gern
RQ1575
Status
Name
Identification
Unit
Filter text
Filter text
T37Dev5Sec2IO4_sh
T37Dev5Sec2IO4
T51_T7Dev5Sec2IO6
T63Device5Sector7real_wr
T12Dev5Sec2IO7_sh
T9Dev5Sec2IO3
T7Dev5Sec2IO6_sh
0/0/0
38/unlimited
Extended filter
Name
Identification
Address
Project
Device5Sector2BOOL2
(5) T01 2.2
RQ1575
T9Dev5Sec2IO3
(5) T09 2.3
RQ1575
T37Dev5Sec2IO4
(5) T37 2.4
RQ1575
T51_T7Dev5Sec2IO6
(5) T51 2.8
RQ1575
T63Device5Sector7real_wr
(5) T63 7.5
RQ1575
T7Dev5Sec2IO6_sh
RQ1575
Add
Remove
Clear Selection
OK
Cancel
Help

Element	Description
Project tree	Definition of the project from which the variable shall be selected.
Selection window	Selection of the variables: ▶ Double click the selected variable in order to add it to the variable list. ▶ You can move the selected variable to the variable list via Drag&Drop ▶ Select the desired variable. With the help of Ctrl and/or Shift multi-selection is possible. By clicking **Add** the selected variables are added to the variable list.
Variable list	Lists all selected variables.
Add	Adds the currently selected variable of the selection window to the variable list.
Delete	Removes the variables which are selected in the variable list from the list.
No selection	Depending on the element: ▶ the dialog is canceled ▶ certain links such as lot variables in archiving can be released

7.1.2 Link types

For entering formulas, you can use different functions from the following mathematical areas:

- **Arithmetics** (combinable with other areas)
- **Trigonometry** (combinable with other areas)
- **Boolean algebra** (combinable with other areas)
- **Statistics** (not combinable)
- **Cross calculation** (not combinable)
- **Data reduction** (not combinable)
- **Comparison** (combinable with other areas)
- **Counted measurand processing** (only SICAM 230)

Range	Expression	Function
Arithmetics	Phases	
	`+`	Addition
	`-`	Subtraction
	`*`	Multiplication
	`/`	Division
	`^`	Exponential calculation
	`SQRT`	Square root
	`()`	Brackets
Trigonometry	Trigonometric functions: Value is regarded as radian (0..x*PI)	
	`sin`	sine
	`cos`	cosine
	`tan`	tangent
	`sinh`	hyperbolic sine
	`cosh`	hyperbolic cosine
	`tanh`	hyperbolic tangent
	`()`	Brackets
Boolean algebra	Logical links: Variables will only be checked for the logical value '0'; if the value does not equal '0', it will be considered as '1'. As opposed to Bit formulas, the technical range of the Mathematics variables can be modified by a stretch factor (not equal '0' or '1').	
	`AND`	logical 'AND'
	`OR`	logical 'OR'
	`XOR`	logical 'EXCLUSIVE OR'
	`NOT`	Negation
	`()`	Brackets
Statistics	The value is only available at the end of the calculation period (number of events).	

	GltndMW	Moving average for changes
	GltndStdabw	moving deviation of changes
	Counters	counter function
	()	Brackets
Cross calculation		
	Quer MW	Calculation of the average of several channels at the same time
	()	Brackets
Data reduction		
	TMittelung	Average over time and events
	Sum	Sum over time and events
	Max	Maximum over time and events
	Min	Minimum over time and events
	Dif	Difference over time
	()	Brackets
Comparison		
	<	less than
	>	greater than
	<=	Less then or equal
	>=	Greater than or equal
	<>	less or greater
	=	Equal
	()	Brackets
Counting value processing	For linkages with time or event reference (statistics, data reduction) it is necessary to make additional entries. You can make them when ending the link.	Only for SICAM 230.

7.1.3 Bit formulas

Bit formulas are suitable for formula results with LReal mathematic variables that only have a logical low or high state. In contrast to float formulas with Boolean processsing (on page 38), the raw value is already predefined (0,1).

Entering Bit formulas:

- mapping the variable name
- optional: inputs for technical operation, limits and update behavior.
- Allocating formulas

Only Boolean operators are available for calculations (see chapter Boolean algebra (on page 38).)

The mathematics driver also allows to define values (e.g. multi-bit) bitwise via Boolean formulas. To do this, the bit pattern of the variables to be linked has to be entered decimally in the bit formula after the operator (for example **Bit0** and **Bit1** ->3).

Syntax	Description
(X01)NAND(X02)	NOT AND
(X01)NOR(X02)	NOT OR
(X01&3)	**Bit0=**1 and **Bit1=**1 **AND** are linked to the value of the variables X01.
(X01|3)	**Bit0=**1 and **Bit1=**1 **OR** are linked to the value of the variables X01.
(X01#3)	**Bit0=**1 and **Bit1=**1 **XOR** are linked to the value of the variables X01.

EXAMPLES

Formula	Values (binary)	Result	= numerical	= Bool
for: X01=9 (1001b)				
X01|5	1001|101	**1101b**	13	true
X01&5	1001&101	**0001b**	1	true
X01#5	1001#101	**1100b**	12	true
for X01=4 (0100b)				
X01&2	0100&010	**0100b**	0	false

7.1.4 Float formula

The Mathematics interpreter in zenon provides functions from the following areas for float formulas:

- Arithmetics (on page 37)
- Trigonometry (on page 37)
- Boolean algebra (on page 38)
- Statistics (on page 39)
- Cross calculation (on page 44)
- Data reduction (on page 45)
- Comparison (on page 56)
- Counting value processing (on page 56)

as well as some other float formulas (on page 56) that must be entered directly.

Arithmetics

Select **Arithmetics** from the drop-down list in the formula definition section. The following basic functions are available:

Expression	Function
+	Addition
-	Subtraction
*	Multiplication
/	Division
^	Exponential calculation
SQRT	Square root
()	Brackets

Arithmetic functions can be combined with:

- Trigonometry (on page 37)
- Boolean algebra (on page 38)
- Comparison (on page 56)

Trigonometry

Select **Trigonometry** from the drop-down list in the **Formula definition** section. The following trigonometric functions are available; the value will be handled as a radian (0...x*PI):

Expression	Function
`sin`	sine
`cos`	cosine
`tan`	tangent
`sinh`	hyperbolic sine
`cosh`	hyperbolic cosine
`tanh`	hyperbolic tangent
`()`	Brackets

Trigonometry functions can be combined with:

- Arithmetics (on page 37)
- Boolean algebra (on page 38)
- Comparison (on page 56)

Boolean algebra

Select **`Boolean algebra`** from the drop-down list in the **`Formula definition`**section. The following expressions are available:

Expression	Function
`AND`	logical 'AND'
`OR`	logical 'OR'
`XOR`	logical 'EXCLUSIVE OR'
`NOT`	Negation
`()`	Brackets

For logical calculations, the variables will only be checked for the logical value '0'. If the value is not '0', it will be assumed as '1'. As opposed to Bit formulas (on page 36), the technical range of the Mathematics variables can be modified by a stretch factor (not equal '0' or '1').

Boolean alegba can be combined with:

- Arithmetics (on page 37)
- Trigonometry (on page 37)
- Comparison (on page 56)

Statistics

Select **Statistics** from the drop-down list in the **Formula definition** section. The following basic functions are available:

Expression	Function
`GltndMW`	Moving average for changes (on page 39)
`GltndStdabw`	floating standard deviation (on page 40)
`Counters`	counter function (on page 41)
`()`	Brackets

The value is only available at the end of the calculation period (number of events).

Statistical functions cannot be combined with any further functions.

Information

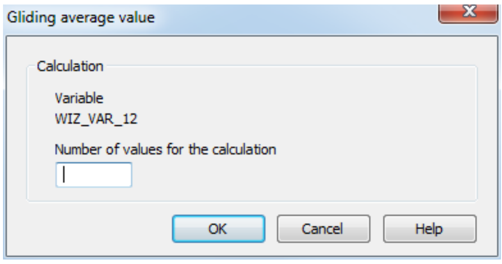
The value of the mathematic driver is saved when Runtime is ended normally. To ensure that values are saved if limits or rema conditions where the counter runs become inactive:

*Carry out the **Save AML and CEL ring buffer** (**main.chm::/11253.htm**) or the **Save remanent data** (**main.chm::/25971.htm**) function. In addition, you can also still save the values cyclically (every hour, for example).*

Floating average

The standard deviation over a defined number of values is calculated.

A dialog to input the number of values is opened when the function is confirmed:



Parameters	Description
`Calculation`	Configuration of the calculation for the given value.
Number of values for calculation	Number of values for which the calculation is carried out. After this number has been reached, the calculation is carried out and and moved by one with further value.
`OK`	Accepts input and closes dialog as well as formula input.
`Cancel`	Discards input, closes dialog and returns to formula input.
`Help`	Opens online help.

Example

`10` values are set as the requirement.

Procedure in Runtime:

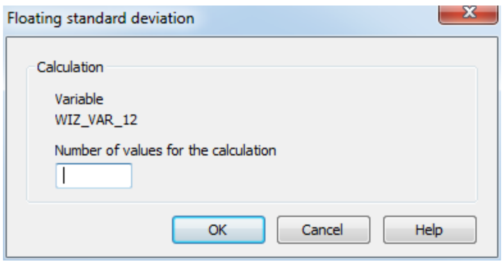
- Starting with the initial value, `10` values are waited for and then the average is calculated from this.
- After the first 10 values, the first value from the calculation is removed and the new 11th value is used for the calculation, and so on. Value included in the calculation etc.

The area for calculation therefore also moves with the growing number of the values.

Floating standard deviation

The standard deviation over a defined number of values. The first output of the value occurs when all changes are present.

A dialog to input the number of values is opened when the function is confirmed:



Parameters	Description
Calculation	Configuration of the calculation for the given value.
Number of values for calculation	Number of values for which the calculation is carried out. After this number has been reached, the calculation is carried out and and moved by one with further value.
OK	Accepts input and closes dialog as well as formula input.
Cancel	Discards input, closes dialog and returns to formula input.
Help	Opens online help.

Example

10 values are set as the requirement.

Procedure in Runtime:

- Starting with the initial value, 10 values are waited for and then the standard deviation is calculated from this.
- After the first 10 values, the first value from the calculation is removed and the new 11th value is used for the calculation, and so on. Value included in the calculation etc.

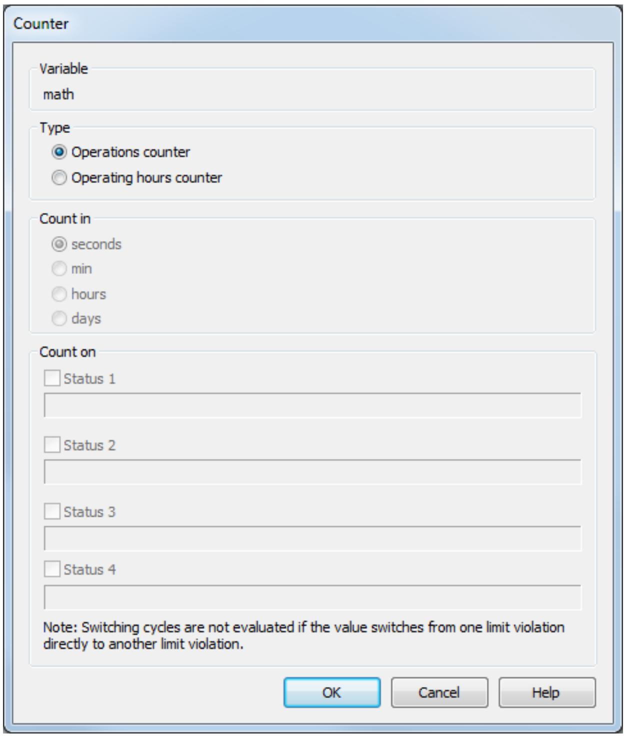
The area for calculation therefore also moves with the growing number of the values.

Counter

The counter function allows you to increase a counter variable every time the Rema status or the limit of a source variable is reached. With this, the operating hours of equipment can be counted, for example, and the signal for required maintenance work can be given with the counter variables.

CONFIGURING THE COUNTER

By clicking on OK in the statistics dialog, the dialog to configure the counter type and the states or limits to be counted is opened:



Parameters	Description
Variable	*Displays the variables to be configured.*

TYPE

Parameters	Description
Type	*Counter type. Select between* ***Operations counter*** ***Hours counter***
Operations counter	*Counts if a state or limit has been reached after a delay time that may have been configured has expired.* ***Note:*** *Switching cycles are not evaluated if the value switches from one limit violation directly to another limit violation. For example, breaches of limit values of binary variables cannot be evaluated with the **switching cycle counter**.*
Hours counter	*The operating hours counter begins counting when:* - the last value is `1` and communication changes to `invalid` or - Runtime is closed *The counter will stop as soon as the communication is restored and the value is `0`.*

COUNT IN

Parameters	Description
Count in	*Activation of the states in which counting is to take place.* *If the source variable is linked to a reaction matrix, the states 1 - 4 match the counter numbers of the Rema.* *If the source variable was not linked to a reaction matrix, the possible states correspond to the limits.*
`OK`	*Accepts input and closes dialog.*
`Cancel`	*Discards entries and closes the dialog.*
`Help`	*Opens online help.*

COUNTER-WISE

The counter only counts new breaches of the sum of all configured limits, but not breaches of individual values. That means:

- The counter value is not increased if:
  - A limit has already been breached and
  - The variable value changes so that a limit is breached again
- The counter value is not increased if:
  - The variable value changes within a limit and
  - The **Treat each value change as a new limit violation** property is activated
- The counter value is only increased if:
  - None of the 4 states was previously breached
  - At least one of the 4 states after the update was breached

Cross calculation

The cross calculation allows the calculation of mean values over several variables.

Select `Cross calculation` from the drop-down list in the `Formula definition` section. The following basic functions are available:

Expression	Function
`Quer MW`	Calculation of the average of several channels at the same time
`()`	Brackets

Cross calculations cannot be combined with any further functions.

Data reduction

Select **`Data reduction`** from the drop-down list in the **`Formula definition`** section. The following basic functions are available:

Expression	Function
`TMittelung`	Average over time and events (on page 45)
`Sum`	Sum over time and events (on page 47)
`Max`	Maximum over time and events (on page 50)
`Min`	Minimum over time and events (on page 52)
`Dif`	Difference over time (on page 54)
`()`	Brackets

Data reduction cannot be combined with any further functions.

Time average

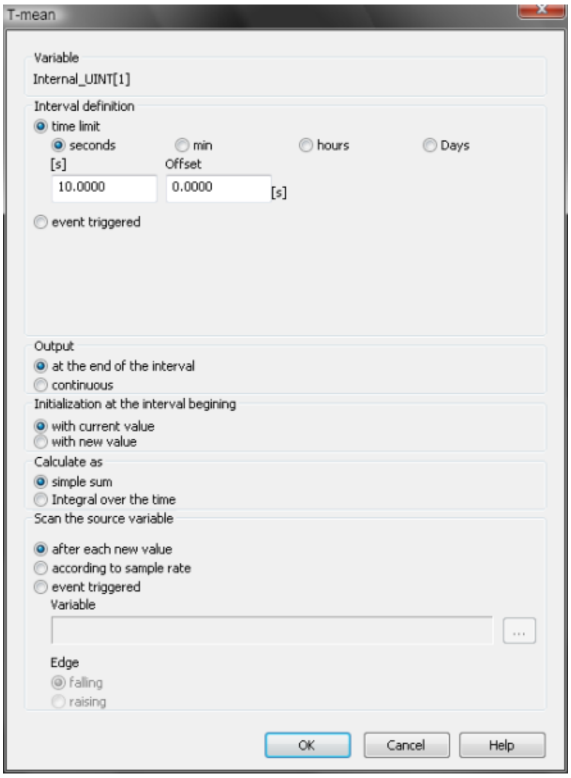
The data reduction **`Time average`** allows the calculation of the average of a variable. The possible average calculations are:

- Time interval:
  Averaging over time interval (seconds, minutes, hours, days)
- Event interval:
  The average is calculated when a change in an event variable occurs

Further parameters in the configuration are:

- Output: The result is provided at the end of the interval or continuously
- The value is initialized at the beginning of the interval (existing or new value)
- Type of calculation: Stipulation of the calculation type (sum, integral via the time)
- Scanning type: Type of scanning of the source Variable (spontaneous, after update parameterization, event-controlled)

Press the button OK to open a dialog where you can enter the parameters for calculating the average:



Variable

Display of the event variable

Interval definition

Parameters	Description
time limit	Calculation of average over time.
event-triggered	Average is calculated when an event variable changes.
Time	Entry of the numeric value for the corresponding time (real-time synchronous, i.e. entry 15 means 15, 30, 45, 0 etc.).
Seconds, minutes, hours, days	Selection of the time interval.
Offset	Delay for start and end of the average calculation.

Output

Parameters	Description
At the end of the interval	The value is only provided at the end of the interval; until then, only the previous value is available.
continuous	The value is updated during each scan of the source variable.

`Initialization at the interval begining`

Parameters	Description
with current value	Value is based on previous value.
with new value	Value must be re-calculated.

`Calculate as`

Parameters	Description
simple sum	Addition of the values.
Integral over time	Integration of the values over the time interval

`Scan the source variable`

Parameters	Description
after each new value	on change changes of the source variable are immediately considered for the calculation
According to scanning rate	The source values are only used in the defined update rate for the calculation (button **`Update is fixed to 0,5 sec.`**).
event-triggered	Source value is used for the calculation when a change of edge occurs in the configured binary event variable.
Button ... for event variable	Active if projecting was set to `event-controlled.` Selection of the binary event variable.
Edge	Active if projecting was set to `event-controlled.` Determination of the respective edge change for the acceptance of the source value (rising, falling).

Summation

The data reduction **`Summation`** allows to calculate the sum of a variable.

Possible summations are:

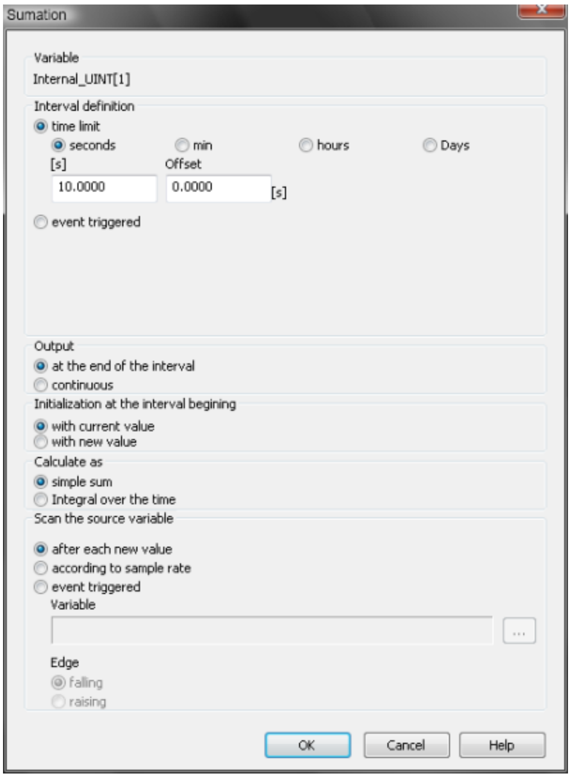
- Time interval:
  Averaging over time interval (seconds, minutes, hours, days)

- Event interval:
  The sum is formed when a change to an event variable occurs

Further parameters in the configuration are:

- Output: The result is provided at the end of the interval or continuously
- The value is initialized at the beginning of the interval (existing or new value)
- Type of calculation: Stipulation of the calculation type (sum, integral via the time)
- Scanning type: Type of scanning of the source Variable (spontaneous, after update parameterization, event-controlled)

Press the button OK to open a dialog where you can enter the parameters for calculating the sum:



Variable
Display of the event variable

Interval definition

Parameters	Description
time limit	Minimum over time.
event-triggered	Minimum is calculated when an event variable changes.
Time	Entry of the numeric value for the corresponding time (real-time synchronous, i.e. entry 15 means 15, 30, 45, 0 etc.).
Seconds, minutes, hours, days	Selection of the time interval.
Offset	Delay for start and end of the average calculation.

Output

Parameters	Description
At the end of the interval	The value is only provided at the end of the interval; until then, only the previous value is available.
continuous	The value is updated during each scan of the source variable.

Initialization at the interval begining

Parameters	Description
with current value	Value is based on previous value.
with new value	Value must be re-calculated.

Calculate as

Parameters	Description
simple sum	Addition of the values.
Integral over time	Integration of the values over the time interval

Scan the source variable

Parameters	Description
after each new value	on change changes of the source variable are immediately considered for the calculation
According to scanning rate	The source values are only used in the defined update rate for the calculation (button **Update is fixed to 0,5 sec.**).
event-triggered	Source value is used for the calculation when a change of edge occurs in the configured binary event variable.
Button ... for event variable	Active if projecting was set to event-controlled. Selection of the binary event variable.
Edge	Active if projecting was set to event-controlled. Determination of the respective edge change for the acceptance of the source value (rising, falling).

Maximum calculation

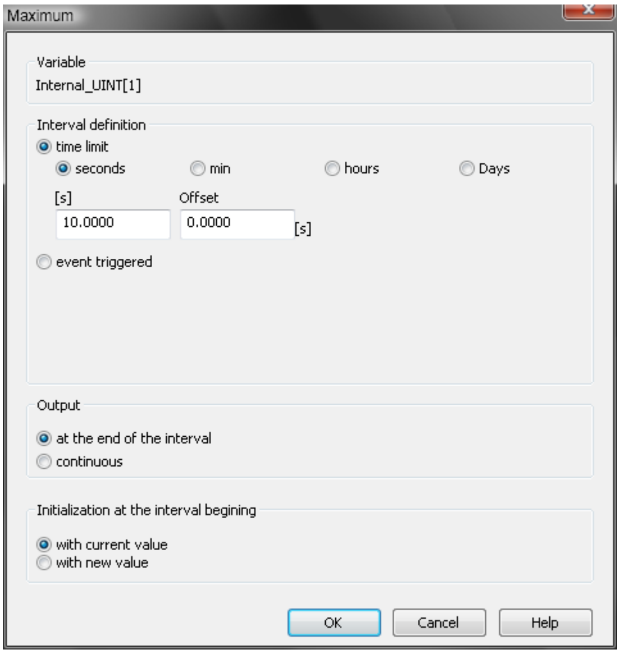
The data reduction of maximum offers the possibility to determine the maximum of a variable within the interval. Possible settings are:

- Time interval:
  Maximum within a time interval (seconds, minutes, hours, days)
- Event interval:
  The maximum is formed when a change in an event variable occurs

Further parameters in the configuration are:

- Output: The result is provided at the end of the interval or continuously
- The value is initialized at the beginning of the interval (existing or new value)

Press the button OK to open a dialog where you can enter the parameters for calculating the maximum:



`Variable`: Display of the event variable

INTERVAL DEFINITION

Parameters	Description
Time limited	Maximum over time.
event-triggered	Maximum is calculated when an event variable changes.
Time	Entry of the numeric value for the corresponding time (real-time synchronous, i.e. entry 15 means 15, 30, 45, 0 etc.).
Seconds, minutes, hours, days	Selection of the time interval.
Offset	Delay for start and end of the average calculation.

OUTPUT

Parameters	Description
At the end of the interval	The value is only provided at the end of the interval; until then, only the previous value is available.
continuous	The value is updated during each scan of the source variable.

INITIALIZATION AT THE INTERVAL BEGINING

Parameters	Description
with current value	Value is based on previous value.
with new value	Value must be re-calculated.

Information

The synchronization time is determined based on the time interval.

Example: If you configure the time interval to be 15 minutes, the calculation will be performed every 15 minutes, starting with the full hour.

Minimum calculation

The data reduction of minimum offers the possibility to determine the minimum of a variable within the interval. Possible settings are:

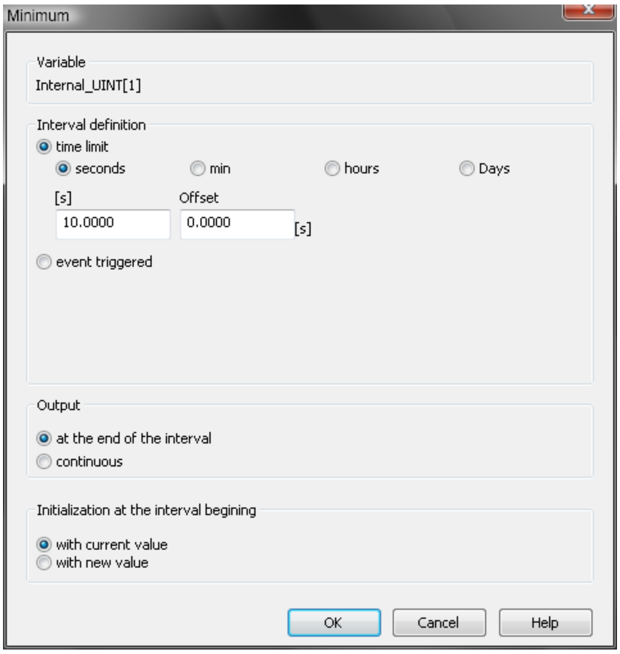
- Time interval: Minimum within a time interval (seconds, minutes, hours, days)

- Event interval: The minimum is formed when a change in an event variable occurs

Further parameters in the configuration are:

- Output: The result is provided at the end of the interval or continuously
- The value is initialized at the beginning of the interval (existing or new value)

Press the button OK to open a dialog where you can enter the parameters for calculating the minimum:



Variable: Display of the event variable

INTERVAL DEFINITION

Parameters	Description
time limit	Minimum over time.
event-triggered	Minimum is calculated when an event variable changes.
Time	Entry of the numeric value for the corresponding time (real-time synchronous, i.e. entry 15 means 15, 30, 45, 0 etc.).
Seconds, minutes, hours, days	Selection of the time interval.
Offset	Delay for start and end of the average calculation.

OUTPUT

Parameters	Description
At the end of the interval	The value is only provided at the end of the interval; until then, only the previous value is available.
continuous	The value is updated during each scan of the source variable.

INITIALIZATION AT THE INTERVAL BEGINING

Parameters	Description
with current value	Value is based on previous value.
with new value	Value must be re-calculated.

Information

The synchronization time is determined based on the time interval.

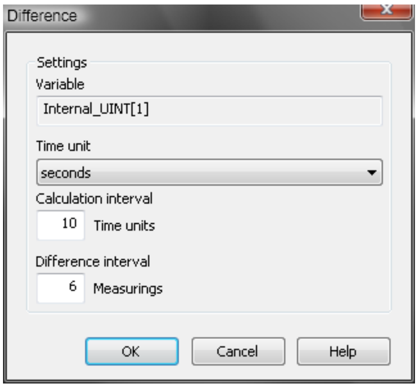
Example: If you configure the time interval to be 15 minutes, the calculation will be performed every 15 minutes, starting with the full hour.

Difference over time

The function can be used e.g. for dynamically monitoring the consumption of liquids or for checking containers for leakages (based on fill level values).

The variable that was inserted into the function is monitored and its values are stored internally with the configurable measurement distance (in time units). If a specific difference is reached (number of measurements), the function will display the first result (R(n)-R(0)), which will then be updated after each measurement. The result of the function is an element of the set of real numbers.

Press the button OK to open a dialog where you can enter the parameters for calculating differences:



Variable

Display of the event variable

Time unit

Dropdown list for selecting the time unit (seconds, minutes, hours, days)

Calculation interval

Parameters	Description
Time units	Number of units

Difference interval

Parameters	Description
Measurings	Number of measurements

Information

The calculation of the difference over time is not absolute but relative, starting with the Runtime start.

Comparison

Select `Comparison` from the drop-down list in the formula definition section. The following operators are available:

Expression	Function
<	less than
>	greater than
<=	Less then or equal
>=	Greater than or equal
<>	less or greater
=	Equal
( )	Brackets

Comparison operators can be combined with:

- Arithmetics (on page 37)
- Trigonometry (on page 37)
- Boolean algebra (on page 38)

Counting value processing

For linkages with time or event reference (statistics, data reduction) it is necessary to make additional entries. You can make them when ending the link.

Attention

This function is only available for SICAM 230 and requires a corresponding license. Find out more in the help pages of SICAM 230.

Additional float formulas

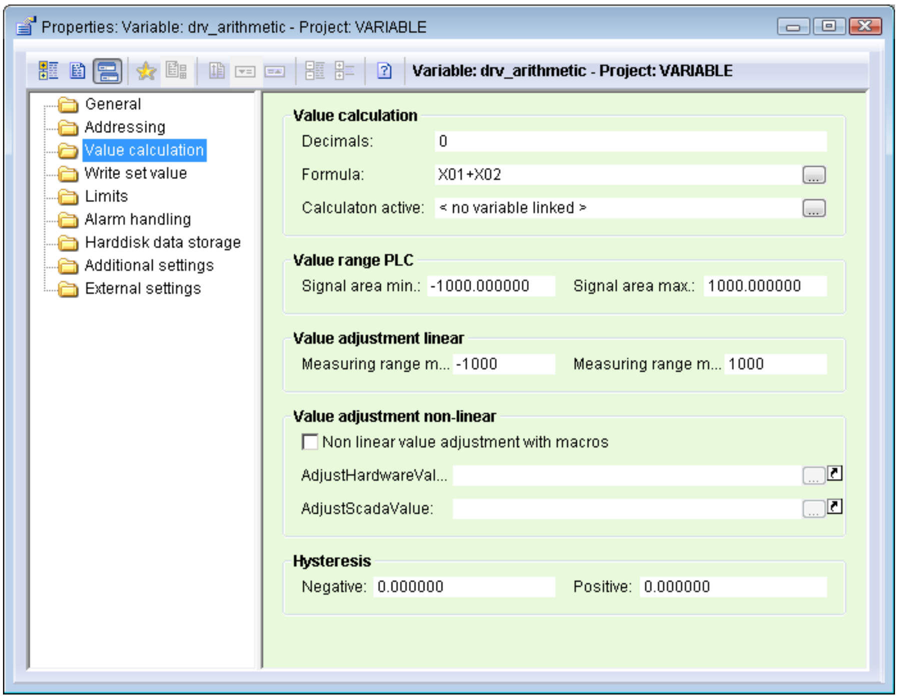
You can use further float formulas that cannot be reached via buttons. You have to enter them directly in the area `Formulas`:

Syntax	Description	Notation
ABS(X01)	Absolute value	|X01|
EXP(X01)	Exponent	$e^{(X01)}$
LN(X01)	Natural logarithm	ln(X01)
LOG(X01)	Logarithm	log(X01)
PI*(X01)	Constant Pi	PI*(X01)
SQR(X01)	Square calculation	$(X01)^{2}$
SQRT(X01)	Square root calculation	$(X01)^{1/2}$

7.2 Event-triggered calculation of Mathematics variables

A Mathematics variable can be assigned to a variable, whose value triggers a calculation. In order to assign a variable for the activation of the calculation:

1. navigate to the **Value calculation** node in properties
2. in the property **Calculaton active**, click on the field with the variable name or on the button ...
3. the dialog for assigning variables opens now
4. select the variable you want to assign



FUNCTIONALITY

- Mathematics variables without an activation variable register all source variables, interval variables and scan variables when the Runtime starts.
- Start of the calculation:
  If a Mathematics variable is linked to an activation variable, the calculation starts as soon as the activation variable has a value of `<> 0`.
  - If the activation variable changes to `activated`, all source variables, interval variables and scan variables will be requested.
  - If the activation variable changes back to `deactivated`, they will be signed off again.
- The activation has priority over the calculation.

  This shows when the activation variable serves as a source variable for the Mathematics variable. If the activation variable changes to deactivated, the result will stay at the same value as it was at the time when the calculation was activated.
- The data type `String` is allowed if the value can be converted to a number.

  If the conversion fails (empty, no number), this will result in the status `deactivated`.
- Status of the Mathematics variables:
  - `deactivated: S_AUS` (Bit 20, switched off).
    The current value of the Mathematics variable will be used as the value. If no value exists yet, the replacement value of the Mathematics variable will be used. Mathematics variables

retain the status S_AUS until there is a calculation result, e. g. calculation type *Counter -> Switching cycle counter with status*.

- for the calculation types Min and Max, with an interval variable but without an activation variable, until the edge of the interval variable is received: S_IBIT (Bit 18, invalid)
- for the calculation types Min and Max, with an interval variable and an activation variable, until the edge of the interval variable is received: S_AUS

- For Mathematics variables that have an interval variable or a scan variable, edge recognition works only if calculation is activated.

  If the interval variable or the scan variable is already active when the activation variable changes from deactivated to activated, this corresponds to an edge of the interval variable / the scan variable.

- If you write to a Mathematics variable, whose calculation is deactivated:
  - the written value will not be sent and is therefore not visible.
  - The written value serves as the initial value (current value) of the calculation, as soon as the calculation is activated. **However:**
  - The write command will be sent to the Standby Server, independent of the activation variable.

7.2.1 Activation and deactivation of the calculation

ACTIVATION

Calculation will be activated if the following conditions apply for the activation variable after a value change:

- The activation variable has a value
- The value changes from 0 to <> 0 and one of the following states is active:
  - S_GA General interrogation Bit 16
  - S_SPONTAN Spontaneous Bit 17
  - S_REVISION Revision Bit 9
  - S_EW_KENNUNG Replacement value Bit 27

This allows you to trigger a calculation by switching to a replacement value.

DEACTIVATION

The calculation will be deactivated if the following conditions apply after a value change of the activation value:

- The variable does not have a valid value yet.
- The value is `0` or none of the states `S_GA`, `S_SPONTAN`, `S_REVISION` or `S_EW_KENNUNG` is active.

7.2.2 Calculation types with special behavior

COUNTER: COUNT WITH STATE

If the caluclation for **`Count with state`** is deactivated, neither the receiving nor the clearing of the state will be recognized.

This means: If the activation variable is `deactivated`, the state is considered as `not violated`.

For example:

1. Status received
2. Activation variable changes to `activated`
3. *Violation is recognized -> Counter gets a value and counts*
4. Activation variable `deactivated`
5. Status cleared
6. Status received
7. *Activation variable activated -> no counting*
8. Status cleared
9. *Status received -> counting*
10. Activation variable `deactivated`
11. Status cleared
12. Activation variable `activated`
13. *Status received -> counting*

7.2.3 Error messages

Error message	Reason	Solution
Source variable `<VariablenProjekt Name>/<Variablen-ID>` for mathematics missing. Mathematics variable `<Projektname>/<MaV Name>` disabled!	The source variable of the mentioned project is missing. The Mathematics variable is not calculated.	Change configuration or start project.
Advise for source variable `<VariablenProjekt Name>/<Variable Name>` failed. Mathematics variable `<Projektname>/<MaV Name>` disabled	Data point request for the variable of the mentioned project has failed. The Mathematics variable is not calculated.	Increase system resources, restart Runtime.

7.3 Importing Mathematics variables

When importing Mathematics variables (via XML import), make sure that all variables used in the formulas already exist. If you import variables of different drivers, we recommend to perform the import a second time. This makes sure that recently imported variables are also linked to the Mathematics driver formulas.

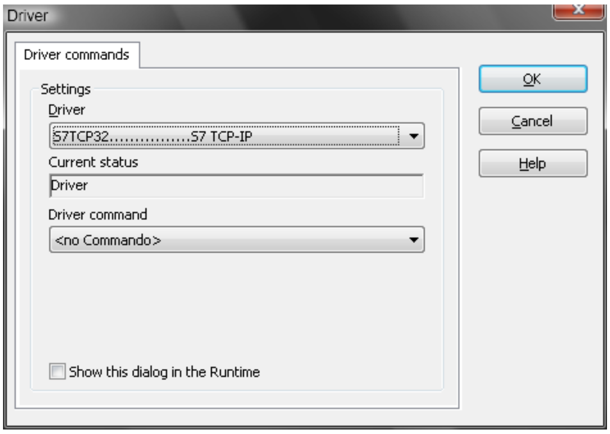
> **Attention**
>
> Network: On a Standard Server defined as data server, the Mathematics variables are not displayed if it is upgraded to be a server.

8. Driver commands

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.

Driver commands are used to influence drivers using zenon; start and stop for example.
The engineering is implemented with the help of function `Driver commands`. To do this:

- create a new function
- select *Variables -> Driver commands*
- The dialog for configuration is opened



Parameters	Description
Drivers	Drop-down list with all drivers which are loaded in the project.
Current state	Fixed entry which has no function in the current version.
Driver commands	Drop-down list for the selection of the command.
▸ `Start driver (online mode)`	Driver is reinitialized 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`	Enter driver-specific commands. Opens input field in order to enter a command.
▸ `Activate driver write set value`	Write set value to a driver is allowed.
▸ `Deactivate driver write set value`	Write set value to a driver is prohibited.
▸ `Establish connection 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)
Show this dialog in the Runtime	The dialog is shown in Runtime so that changes can be made.

DRIVER COMMANDS IN THE NETWORK

If the computer, on which the **`driver command`** function is executed, is part of the zenon network, additional actions are carried out. A special network command is sent from the computer to the project server, which 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 (main.chm::/12464.htm) program that was also installed with zenon. You can find it under *Start/All programs/zenon/Tools 7.20 -> Diagviewer.*

zenon driver log all errors in the log files. The default folder for the log files is subfolder **LOG** in directory `ProgramData`, example:

`C:\ProgramData\COPA-DATA\LOG`. Log files are text files with a special structure.

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 currently created entries live
- customize the logging settings
- change the folder in which the log files are saved

Note:

1. In Windows CE even errors are not logged per default due to performance reasons.
2. The Diagnosis Viewer displays all entries in UTC (coordinated world time) and not in local time.
3. 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.
4. If you only use **Error logging**, the problem description is in column **Error text**. For other diagnosis level the description is in column **General text**.
5. For communication problems many drivers also log error numbers which the PLC assigns to them. They are displayed in **Error text** and/or **Error code** and/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.
6. 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**.

You can find further information on the Diagnosis Viewer in the Diagnose Viewer (main.chm::/12464.htm) chapter.

9.2 Log server

All messages and logs of the driver are sent to the Log Server. The messages can be displayed with the Diagnosis Viewer (main.chm::/12464.htm). (Older versions of zenon use an "Error text file".

By default, the messages of the type 'Error' for the module driver (DRV) will be logged. If you want extended logs to be created, you have to configure this accordingly in the client settings of the Diagnosis Viewer in the Runtime: *Settings – Client configuration, List of parameters, ' Configuration of the message level* '; confirm with **Accept** .

9.3 Check list

- Analysis with the **Diagnosis Viewer** (on page 64):
  -> Which messages are displayed?