zenon driver manual

OMR_FINS

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. OMR_FINS

3. OMR_FINS - Data sheet

General:	
Driver file name	OMR_FINS.exe
Driver name	Omron FINS
PLC types	OMRON SYSMAC
PLC manufacturer	Omron;

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

Requirements:	
Hardware PC	RS 232 serial interface, cable type: SYSMAC WAY; Standard network card
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	1300	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. Requirements

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

5.1 PC

HARDWARE

Serial interface RS232 or standard network card (TCP/IP)

SOFTWARE

Copy the driver file OMR_FINS.exe into the current program directory (unless it is already there) and enter it into the file TREIBER_EN.XML with the tool driverinfo.exe.

Information

Windows CE is not supported at the moment.

6. Configuration

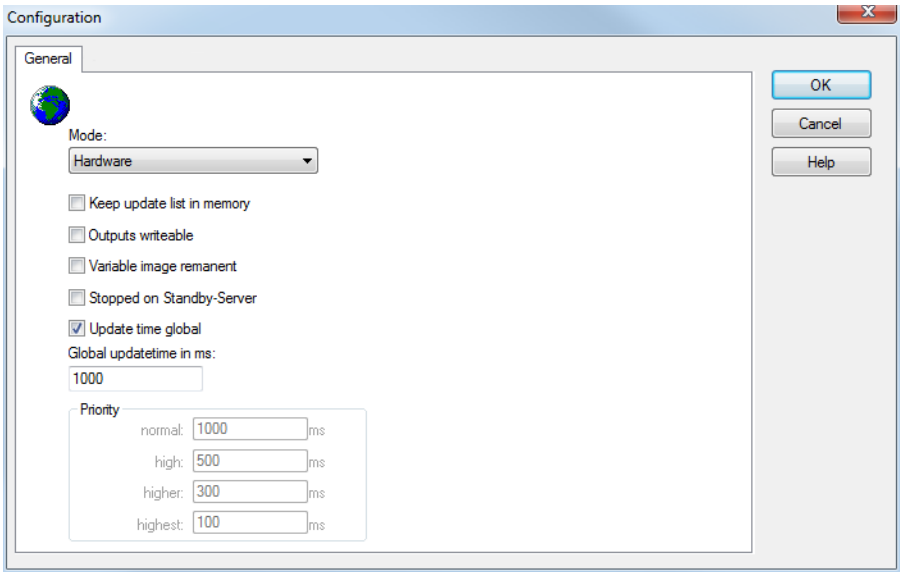
In this chapter you will learn how to use the driver in a project and which settings you can change.

Information

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

6.1 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.



Mode: Hardware

□ Keep update list in memory

□ Outputs writeable

□ Variable image remanent

□ Stopped on Standby-Server

☑ Update time global

Global updatetime in ms: 1000

Priority

normal:	1000	ms
high:	500	ms
higher:	300	ms
highest:	100	ms

Parameters	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 N communication is established to the PLC. 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 a zenon Logic Runtime which is integrated in the driver. For details see chapter Driver simulation (main.chm::/25206.htm).
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 mode hardware if: - one of the states S_MERKER_1(0) up to S_MERKER8(7), REVISION(9), AUS(20) or ERSATZWERT(27) is active The variable image is always saved if: - the variable is of the object type **Driver variable** - the driver runs in simulation mode. (not programmed simulation) The following states are not restored at the start of the Runtime: - 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 on 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** (**statusverarbeitung.chm::/24150.htm**) but an empty value. This prevents that at the upgrade to the Server irrelevant values are created in the AML, CEL and Historian. **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	Active: The set **Global update time** in ms 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.
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 allocation to the variables takes place separately in the settings of the variable properties. The communication of the individual variables are graduated in respect of importance or necessary topicality using the priorities.

	Thus the communication load is distributed better. **Attention:** Priority classes are not supported by each driver For example, drivers that communicate spontaneously do not support it.

CLOSE DIALOG

Parameters	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.

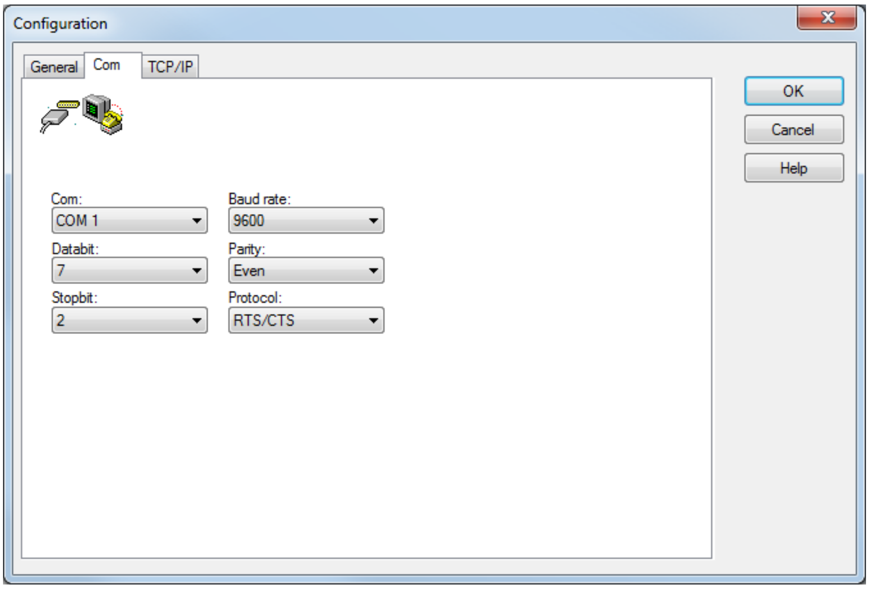
UPDATE TIME FOR CYCLICAL DRIVERS

The following applies for cyclical drivers:

For **Set value**, **Advising** of variables and **Requests**, a read cycle is immediately triggered for all drivers - regardless of the set update time. This ensures that the value is immediately available for visualization after writing. Update times can therefore be shorter than pre-set for cyclical drivers.

6.2 COM

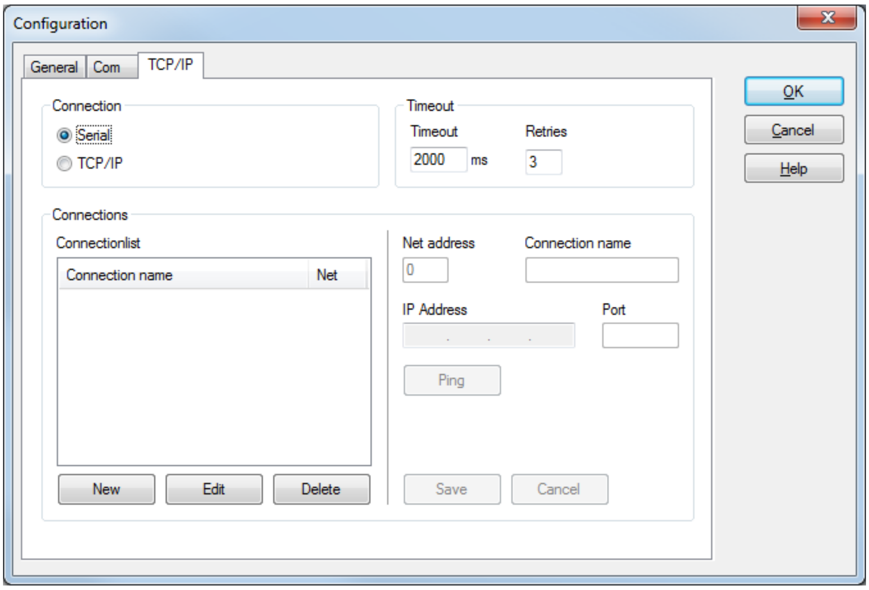
Configuration of the communication parameters for the serial interface



Com: COM 1
Baud rate: 9600
Databit: 7
Parity: Even
Stopbit: 2
Protocol: RTS/CTS

Parameters	Description
Com	Selection of the serial interface, to which the PLC is connected.
Baud rate	Baud rate of the connection. Default: `9600`
Data bit	Data word size in Bit: Default: `7`
Parity	Settings for the parity of the connection Default: `Even`
Stop bit	Number of stopbits for the connection Default: `2`
Protocol	Protocol of the connection. Default: `RTS/CTS`

6.3 TCP/IP



Configuration

General | Com | TCP/IP

Connection
- ● Serial
- ○ TCP/IP

Timeout
- Timeout: 2000 ms
- Retries: 3

Connections

Connectionlist

Connection name	Net

Net address: 0

Connection name:

IP Address: . . .

Port:

CONNECTION

Parameters	Description
Connection	Selection of connection type.
Serial	Active: Serial connection is used.
TCP/IP	Active: TCP/IP connection is used.

TIMEOUT

Parameters	Description
Timeout	Options for Error response time.
Timeout	Waiting time for establishing a connection in milliseconds.
Repetitions	Number of retries if establishing a connection is not successful.

CONNECTIONS

Parameters	Description
Connections	Settings of the connections.
List of connections	List of defined connections to PLCs.
Net address	Corresponds to the **Net address** property in variable configuration.
Connection name	Freely definable name.
IP address	Address of the PLC.
Port	Port address of PLC. You can find details in the manual of your PLC.
Ping	Sends a ping to the IP address that is configured for this connection. Allows the connection to the device to be tested. If the ping is concluded with a negative response, check the IP address and check to see if the device is online.
New	Establishes a new connection.
Edit	Opens highlighted connection for editing.
Delete	Deletes highlighted connection from the list.
Save	Accepts all changes for edited connection and closes editing option.
Cancel	Discards all changes for edited connection and closes editing option.
OK	Accept changes in the dialog and close dialog. Only available if no connection is in the "edit" state.
Cancel	Discards all changes and closes the dialog.
Help	Opens online help.

Information

Maximum number of connections: 256 (0-255).

7. Addressing

VARIABLE ADDRESSING VIA

Address

The variables are allocated to the memory area of the PLC by their offset.

Information

The addressing of the PLC is carried out as WORD. For uneven string length always one additional character is read or written.

8. 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.

8.1 Driver objects

DRIVER OBJECT TYPES AND SUPPORTED IEC DATA TYPES FOR PROCESS VARIABLES IN THE CONTROL SYSTEM

Driver object types	Channel type	Supported data types (DataType)	Read	Write	Comment
IR Area	64	BOOL, INT, UINT	Y	Y	
HR Area	65	BOOL, INT, DINT, REAL, UINT, UDINT, STRING	Y	Y	
AR Area	66	BOOL, INT, UINT	Y	Y	
TIM/CNT	68	BOOL	Y	Y	
EM Area	73	BOOL, INT, DINT, REAL, UINT, UDINT, STRING	Y	Y	
DM Area	70	BOOL, INT, DINT, REAL, UINT, UDINT, STRING	Y	Y	
DM Area BCD	71	INT, DINT, UINT, UDINT	Y	Y	
CIO Area	71	BOOL, INT, DINT, REAL UINT, UDINT, STRING	Y	Y	
WORK Area	72	BOOL, INT, DINT, REAL, UINT, UDINT, STRING	Y	Y	

9. 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.

EXAMPLES FOR ALL POSSIBLE IEC DATA TYPES

SPS	zenon
I16	INT
I32	DINT
U16	UINT
U32	UDINT
REAL	REAL
Boolean	BOOL

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.

10. Driver-specific functions

The driver supports the following functions:

INI ENTRIES

ZENON6.INI

Blockwrite

With an entry in the zenon.ini, block write can be activated. To activate it the entry BLOCKWRITE=1 has to be set in the section [OMR_FINS].

PROJECT.INI

Serial logging

To activate this option a project.ini entry has to be added in the section [RS232LOG] and the entry LOGCOMx=0 or 1 has to be set below. X has to be replaced by the number of the desired interface.

0 switches logging off, 1 switches logging on. If logging is switched on, a file called LOG_COMxxx.txt is generated in the driver folder. X is replaced by the number of the defined interface.

Examples for the activation of logging for COM 1:

[RS232LOG]

LOGCOM1=1

> **Information**
>
> *We recommend activating logging only for a short time, when problems occur. Active logging needs considerable computer performance. Additionally the log file needs a lot of memory within short time.*

ERROR FILE

The driver supports the common error files; with zenon 6.20 or higher, central logging is possible.

EXTENDED ERROR FILE

The driver does not support extended logging.

SERIAL LOGGING

To activate this option, you must add section [RS232LOG] to the project.ini: Here

the entry LOGCOMx=0 or 1 has to be set. X is replaced by the number of the defined interface. With o

0 switches logging off, 1 switches logging on. If logging is switched on, a file called LOG_COMxxx.txt is generated in the driver folder. X is replaced by the number of the defined interface.

Examples for the activation of logging for COM 1:

[RS232LOG]

LOGCOM1=1

We recommend activating logging only for a short time, when problems occur. Active logging needs considerable computer performance. Additionally the log file needs a lot of memory within short time.

BLOCKWRITE

With an entry in the zenon.ini, block write can be activated. To activate it the entry BLOCKWRITE=1 has to be set in the section [OMR_FINS].

REDUNDANCY

Yes

RDA

-

REAL TIME STAMPING

No

BROWSING

Control system data type not supported by driver.

ERROR TIMEOUT

No settings possible.

ACCESS METHODS

POLLING

Cyclical query of the variable, driver.

SPONTANEOUS

-

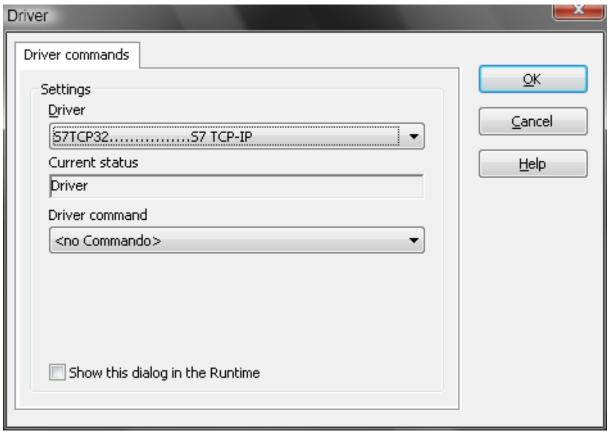
11. Driver commands

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



Driver

Driver commands

Settings

Driver: S7TCP32...............S7 TCP-IP

Current status: Driver

Driver command: <no Commando>

□ Show this dialog in the Runtime

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.

12. Test

TESTED WITH THE FOLLOWING HARDWARE AND SOFTWARE

omron SYSMAC CJ1M /CPU11 (Programmable Controller)

TESTING ENVIRONMENT

Tested with serial interface (serial cable, no null modem cable, cable type: SYSMAC WAY), Parameter: 9600, 7, 2, even, no and TCP/IP connection with default port 9600. The IP-Address of the PLC can be set via Web-interface eg: . http://[IP-Address of the PLC\\/0 or via the DM memory area with serial connection (Offset: m = D30000 + (100 x unit number) + 98; in these 4 Bytes, the IP-Addresse is entered, which will be used by the PLC).

For details, please check the Omron Documentation 'W441E101_Ethernet+CPUs_Operation_Manual.pdf'.

LIMITATIONS

In our test, wrting on HR and AR was impossible. With the driver object type IR neither write nor read was possible.

13. Error analysis

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

13.1 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.

13.2 Error numbers

In case of communication problems, an entry in the error log file of the driver is generated, in which the error cause is specified with a number.

Information

Refer to the FINS documentation section 5-1-3 „End Codes" for an explanation of the error numbers. Here all error numbers are documented.

13.3 Check list

- If the PLC is connected correctly (serial connection cable – no null modem cable with serial connection) and the connection has been configured correct 9600,7,2,even,no)?
- Is the IP address of the PLC configured correctly (with connection via TCP/IP)? Can the PLC be contacted via the Web interface or with e.g. ping in the driver configuration/Conn. TCP/IP?
- Did you analyze the error file (up to zenon version 6.01) or the central log file (version zenon 6.20 and higher)?