AVEVA™ Communication Drivers Pack – Standards – CODESYS Driver

User Guide

Publication date: Tuesday, May 9, 2023

Publication ID: 868863

Contact Information

AVEVA Group Limited
High Cross
Madingley Road
Cambridge
CB3 0HB. UK

https://sw.aveva.com/

For information on how to contact sales and customer training, see https://sw.aveva.com/contact.

For information on how to contact technical support, see https://sw.aveva.com/support.

To access the AVEVA Knowledge and Support center, visit https://softwaresupport.aveva.com.

Contents

AVEVA

Chapter 1

Getting Started with CODESYS Communication Driver

This document describes the CODESYS Communication Driver, and the device and protocol environment in which it works. It includes application-level and bus-level communications protocols, item naming conventions, and Communication Driver features.

Introduction to the CODESYS Communication Driver

The Operations Integration CODESYS Communication Driver (referred to as the Communication Driver through the remainder of this user's guide) is a Microsoft® Windows® application program that acts as a communications protocol server. This Communication Driver is hosted by the OI Server Manager, a Microsoft Management Console (MMC) snap-in, which is part of the Operations Control Management Console (OCMC) suite of utilities.

This Communication Driver allows other Windows application programs access to data in PLCs (also referred to as devices) attached to an Ethernet network. The server can operate in either stand-alone mode or connect with any OPC, DDE, or SuiteLink compliant client application.

This Communication Driver documentation covers only the information you need to configure and run the Communication Driver component. See the documentation that comes with the related components for details on their operation. You can find installation instructions in a help file on the distribution CD. Many high-level functions and user-interface elements of the OI Server Manager are universal to all Communication Drivers, and only the documentation for the OI Server Manager contains descriptions of those universal functions/UI elements. Therefore, reading the documentation for both the MMC and the OI Server Manager is critical to understanding this user's guide. To read the documentation about the MMC and OI Server Manager, click the **Help Topics** on the OCMC **Help** menu.

Note: The shortcut menu items described in this document typically represent only a subset of any actual shortcut menu. Most items in each shortcut menu are standard Windows commands. See the MMC Help for more information about those commands.

Supported Hardware and Software

These are the technical specifications for the CODESYS Communication Driver.

Supported Hardware and Software

This Communication Driver enables TCP/IP Ethernet communication with any computer, controller, or device that hosts a CODESYS runtime, as long as the runtime is compatible with CODESYS PLCHandler 3.5 library. For CODESYS application development, you can use the CODESYS Development System from 3S-Smart Software Solutions. For more information, go to:

Conformance

The following hardware and software was used for conformance testing of this Communication Driver:

Equipment:

- Schneider Electric M241/M251
- CODESYS SP WIN V3.5.1 Control
- CODESYS SP PLCWinNT V2.4.4.0
- Wago CPU 750-841
- EATON XC-CPU202 CODESYS V3.5
- Bosch Rexroth IndraLogic XLC L65 and IndraLogic L20

Chapter 2

Configuring the CODESYS Communication Driver

Each server instance has its own hierarchy of objects, and each object has parameters that you need to configure in order to establish communication between the Communication Driver and individual devices on the network.

You can view a CODESYS server instance's configuration hierarchy under its Configuration node.

This section only describes how to configure object parameters for a CODESYS server instance. For more general information about adding and configuring objects, see "Configuring Your OI Server" in the Communication Drivers Pack Help.

- Adding and Configuring Channel Selector Object
- Adding and Configuring Device Selector Objects
- Configuring the Communication Settings of a Channel
- Device Groups and Device Items

Adding and Configuring Channel Selector Object

The server-specific configuration of the CODESYS Communication Driver hierarchy tree under the OI Server Manager starts at the Channel Selector object. This object lets you set server parameters for communication with agents (devices) in the hierarchy tree.

To add a ChannelSelector connection to your CODESYS hierarchy

1. In the console tree, right-click **Configuration** and then click **Add ChannelSelector Connection**. The **New_ChannelSelector_000** object appears in the hierarchy.
2. Edit the object name to appropriately describe components of your specific hardware environment. If you do not rename the object at this time, a numeric sequencing system is applied. You can rename the hierarchy entry later.

   The **New_ChannelSelector_000 Parameters** view is displayed.

Configuring Advanced Settings

Click this button to open the **Advanced Settings** dialog box, which provides access to additional communication settings such as timeouts, retries, and buffer sizes. You might need to change these settings if the Communication Driver behaves unexpectedly during run time, but the default settings should work for most

network configurations. For more information about these settings, see "Advanced Settings" in Communication Drivers Pack Help.

Adding and Configuring Device Selector Objects

The CODESYS Communication Driver can connect to different Windows agents, PLCs, and other data sources. These connections are modeled in the hierarchy by means of Device Selector objects, each of which models the end-point of the communications path.

From the **ChannelSelector** branch of the Communication Driver hierarchy, create the new **DeviceSelector** object.

To add a Device Selector connection to your CODESYS hierarchy

1. Right-click the **New_ChannelSelector_000** object, and select **Add DeviceSelector Connection**.

   The **New_DeviceSelector_000** object is created.

2. Rename the object as appropriate.
3. **The New_DeviceSelector_000 Parameters** configuration view is displayed.

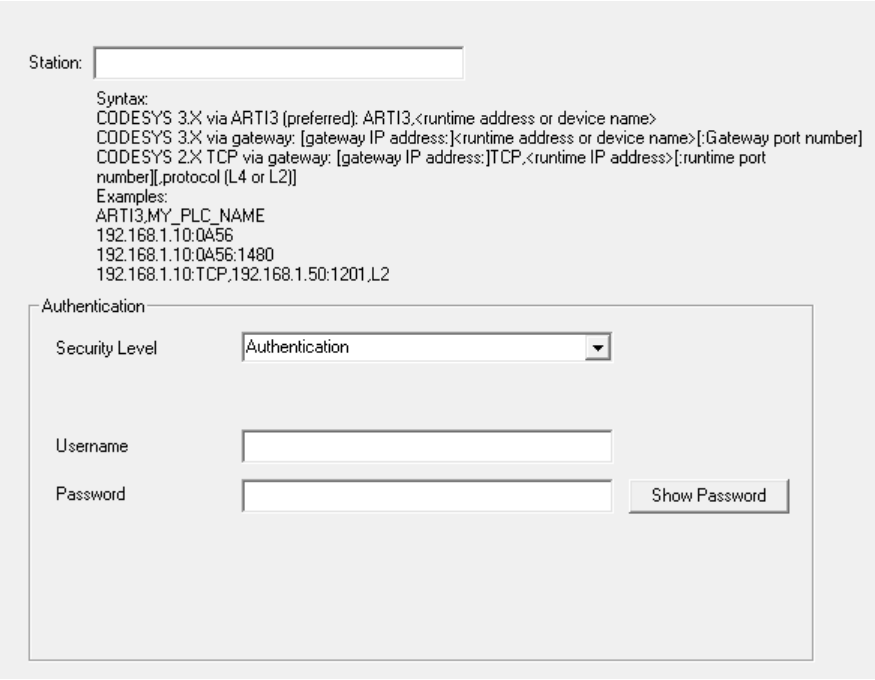
To configure the Device Selector connection

- Configure the **Station**.

  The Station field cannot be empty. The syntax of the Station depends on the ARTI3 or the gateway being used by the Communication Driver. For more information, see Setting Station ID of a Device.

Authenticating the Connection

The **Authentication** section allows you to configure the security settings while connecting to a secure (ARTI3) device.



To configure a secure connection

1. In the **Authentication** section, click the **Security** dropdown and select Authentication.

   If the connection to the data source is unsecure, select **No Security**.
2. In the **Username** field, enter the name of the user that is configured with the secure device.
3. In the **Password** field, enter the password that is configured with the secure device.

Setting the Station ID of a Device

Set the station ID for a selected device so that the Communication Driver can identify and communicate with it on the network.

Syntax

To connect to a CODESYS V3.x device using ARTI3, use the following syntax:

```
ARTI3,<runtime address or device name>
```

To connect to CODESYS V3.x device via Gateway, use the following syntax:

```
[gateway IP address:]<runtime address or device address>[:Gateway port number]
```

To connect to a CODESYS V2.x device using TCP/IP communication via CODESYS Gateway, use the following syntax:

```
[gateway IP address:]TCP,<runtime IP address>[:runtime port number][,protocol (L4 or
L2)]
```

To connect to a CODESYS V2.x device in simulation mode using an exported symbol file, use the following syntax:

```
SIM,<symbol file>
```

The following syntax diagrams show all of the possible options:

For CODESYS V3.x device:

```
{ {  | gateway IP address: }runtime address or device name{  | :gateway port number } |
ARTI3,runtime address or device address }
```

For CODESYS V2.x device:

```
{ {  | gateway IP address: }{ TCP{  | ,{  | runtime IP address{  | :runtime port
number } }{  | ,{ L4 | L2 } } } }
| SIM,symbol file }
```

Where,

gateway IP address: The IP address of the CODESYS Gateway server that is managing communication with the CODESYS device(s). For CODESYS V3.x, if no address is specified, the default is 127.0.0.1 (i.e., localhost). For CODESYS V2.x, if no address is specified, it will connect directly using ARTI.

runtime address or device name: The name (case sensitive) or hexadecimal address of the CODESYS device.

gateway port number: The port number of the CODESYS Gateway server that is managing communication with the CODESYS device(s). If no port is specified, the default is 1217.

runtime IP address: The IP address of the CODESYS device. If no address is specified, the default is 127.0.0.1 (i.e., localhost).

runtime port number: The port number of the CODESYS device. If no port is specified, the default is 1200.

protocol: Protocol to be used to communicate with the CODESYS device. The valid options are L4 (Level 4) and L2 (Level 2). If no protocol is specified, the default is L4.

symbol file: The file path to a symbol file (*.sym or *.xml) that has been exported from the CODESYS programming software. For more information, see the original CODESYS documentation, as well as the appendices at the end of this document.

Connect to CODESYS 2.x or 3.x Runtime using CFILE

To connect to CODESYS 2.x or 3.x Runtime using CFILE

Configure the Station parameter using the following syntax:

```
CFILE,<Configuration File>,<Configuration ID>
```

where,

<Configuration File>,: the path relative to the application path.

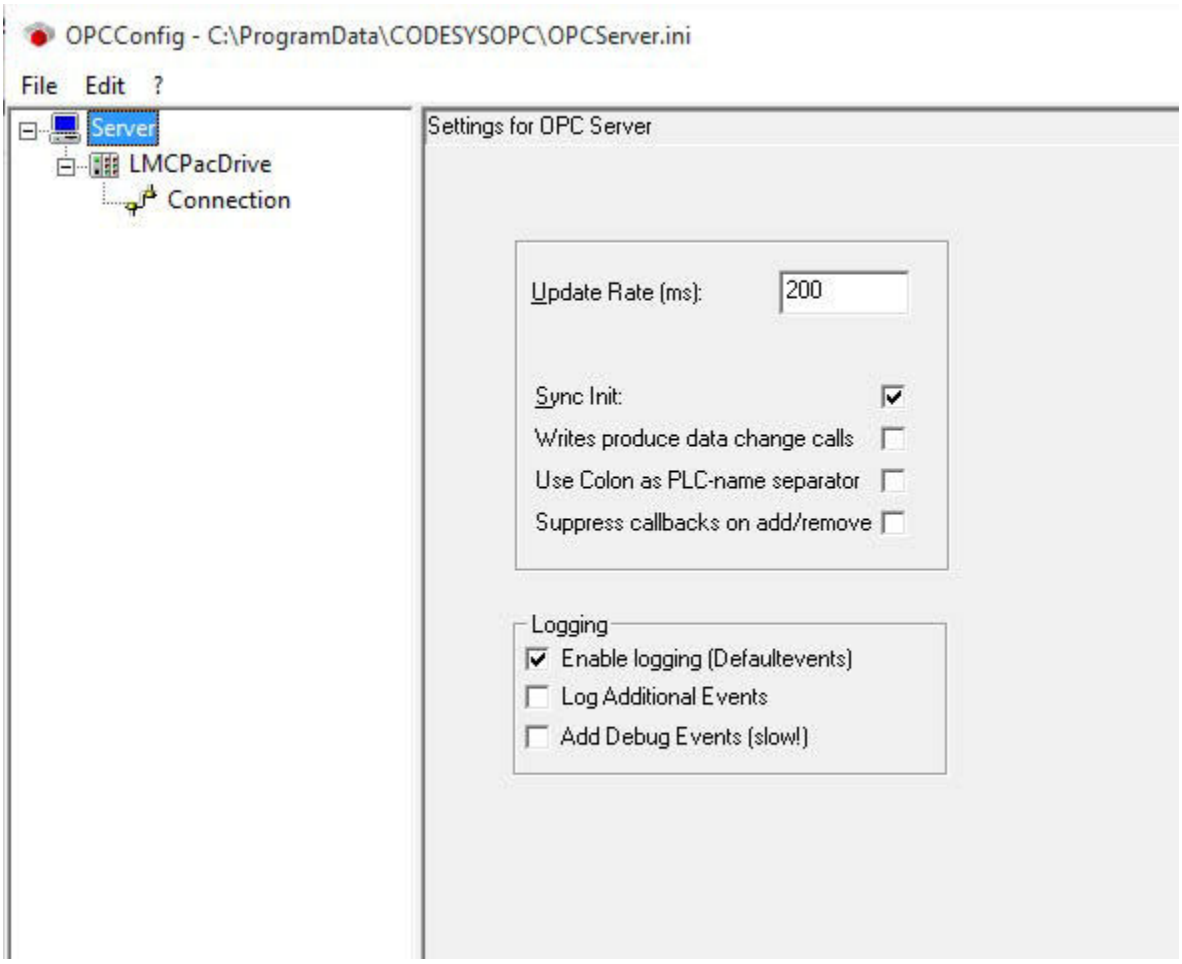
<Configuration ID>: the configuration ID within the configuration file.

Note: The parameters specified between angle brackets < > indicate that the parameter is mandatory, and the parameters between square brackets [ ] indicate that it is optional.

Examples:

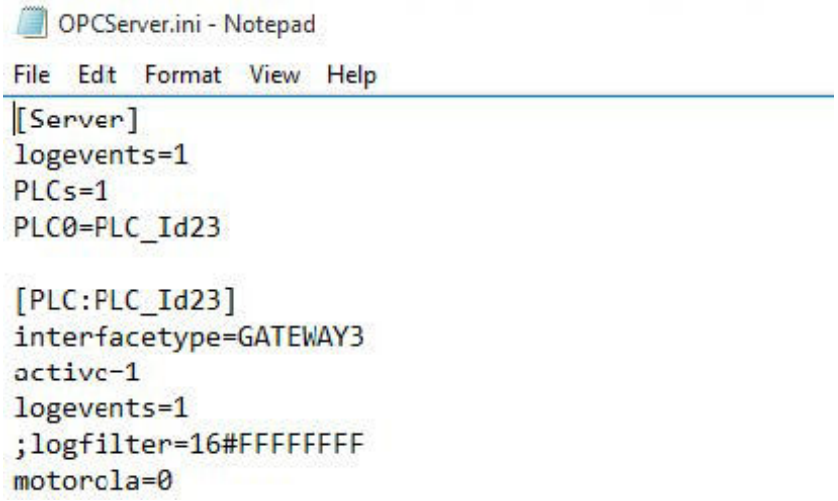
Station	Description
CFILE,.\A\File.ini,PLC:PLCWinNT	Connects to the Runtime using the configuration in the file located in a sub-directory A which is inside the application folder. The connection is established using the configuration in section [PLC:PLCWinNT] inside the file.
CFILE,C:\Config\OPCServer.ini,0	See below for explanation

To find the Configuration file, find the OPCConfigurator for the device that comes with the device software, as shown in the image below.



From the **File** menu click **Save As** and choose a location to save the file in. This is the first parameter in the example of station CFILE,C:\Config\OPCServer.ini,0

The file looks like this on opening in notepad:



Note the Parameter under [Server] for the PLC# which pertains to the controller you want to communicate with. This corresponds to 0 in the example station : CFILE,C:\Config\OPCServer.ini,0

Examples of valid station IDs for CODESYS devices

Example	Description
192.168.1.10:0A56	Connects to a CODESYS V3.x device at hexadecimal address 0A56, via a CODESYS Gateway server running at IP address 192.168.1.10 on default port 1217
127.0.0.1:0A56	Connects to a CODESYS V3.x device at hexadecimal address 0A56, via a CODESYS Gateway server running at localhost on default port 1217
0A56:1480	Connects to a CODESYS V3.x device at hexadecimal address 0A56, via a CODESYS Gateway server running at localhost on port 1480
MY-DEVICE-10:1480	Connects to a CODESYS V3.x device named "MY-DEVICE-10", via a CODESYS Gateway server running at localhost on port 1480
192.168.1.10:0A56:1480	Connects to a CODESYS V3.x device at hexadecimal address 0A56, via a CODESYS Gateway server running at IP address 192.168.1.10 on port 1480
192.168.1.10:TCP,192.168.1.50	Connects to a CODESYS V2.x device at IP address 192.168.1.50 on default port 1200, using L4 protocol, via a CODESYS Gateway server running at IP address 192.168.1.10
192.168.1.10:TCP,192.168.1.50:1201,L2	Connects to a CODESYS V2.x device at IP address 192.168.1.50 on port 1201, using L2 protocol, via a CODESYS Gateway server running at IP address 192.168.1.10
TCP, ,L2	Connects to a CODESYS V2.x device at localhost on default port 1200, using L2 protocol over ARTI
TCP,192.168.0.50	Connects to a CODESYS V2.x device at IP address 192.168.1.50 on default port 1200, using L4 protocol over ARTI.
ARTI3,MY_PLC_NAME	Connects to a CODESYS V3.x device named "MY_PLC_NAME", using ARTI3

SIM,C:\A\File.xml	Connects to a CODESYS V2.x device in simulation mode, as specified in the CODESYS symbol file located at C:\A\File.xml..
CFILE,.\A\File.ini,PLC:PLCWinNT	Connects to the Runtime using the configuration in the file located in a sub-directory A which is inside the application folder. The connection is established using the configuration in section [PLC:PLCWinNT] inside the file.

Device Groups and Device Items

Device Group Definitions

Use the **Device Groups** configuration view, to create, add, delete, and define device groups. You can also configure default update intervals for the objects and edit update intervals in this dialog box. To open the Device Groups dialog box, in the Device Selector configuration editor, click the **Device Groups** tab.

Note: When you select another part of the Communication Driver tree hierarchy, you are prompted to save the modifications to the configuration set.

To create or add device groups

1. Right-click anywhere in the table, and then click **Add**. A device group is added with a default name and update interval.
2. Enter a unique name up to 32 characters long for the device group.

To delete device groups

1. Right-click the device group to be deleted, and then click **Delete**.
2. Read the warning, and then click **Yes**.

To edit device groups

Use the **Edit** option from the **Device Groups** tab only for configuring the Communication Driver's unsolicited message handling.

To configure default update intervals

To configure a default update interval for the object, right-click in the **Device Groups** box and then click **Config Default Update Interval**.

To edit update intervals

To edit the update interval for an object, double-click its value in the **Update Interval** column and make the edits.

or

Right-click its value in the **Update Interval** column and then click **Modify Update** Interval.

The update interval is the frequency, in milliseconds, that the Communication Driver acquires data from the topics associated with that device group.

Different topics can be polled at different rates from a PLC by defining multiple device group names for the same PLC and setting a different update interval for each device group.

Device Item Definitions

The device item name is an is an “alias” or a label for the data in the device. It is an alternative name for the item reference, and ca be used instead of the item reference when you create the client application. Device item configuration is optional, but is strongly recommended.

To create or add device items

1. Right-click anywhere in the table, and then click **Add**.
2. In the **Name** column, type a unique item name. The maximum is 32 characters.
3. In the corresponding line, double-click the **Item Reference** column and enter the correlated item reference for the name you created.

To rename device items

Right-click the device item to be renamed and click **Rename**. Make the changes.

To delete device items

Right-click the device item to be deleted from the list and click **Delete**.

To clear all device items

Right-click in the **Device Items** box and click **Clear All**. All the device items listed are cleared after you confirm their deletion.

NOTE: You can import a .csv file containing your item definitions to help streamline configuration. See "Exporting and Importing CSV Files" in the Communication Drivers Pack Help.

AVEVA

Chapter 3

CODESYS Communication Driver Reference

Item Reference Syntax

Use item references to access data stored in memory registers in connected devices, as well as to access standard system items in the Communication Driver itself.

This section only describes the item reference syntax and options for the CODESYS server. For more general information about item references, see "Managing Device Items" and "Item Reference Descriptions" in the Communication Drivers Pack Help.

Item references in this Communication Driver use the following syntax.

CODESYS V3.x

For all variables in a CODESYS V3.x device, use the following syntax:

```
<application name>.<object name>.<variable name>
```

The following syntax diagram shows all of the possible options:

```
application name.object name.variable name
```

Where,

application name: The name of the CODESYS application.

object name: The name of the program organization unit (POU), global variable list, or other programming object that contains the variable. For example, PLC_PRG.

variable name: The name of the variable.

CODESYS V2.x

For local and global variables in a CODESYS V2.x device, use the following syntax:

```
[object name].<variable name>
```

The following syntax diagram shows all of the possible options:

```
{ object name }.variable name
```

Where,

object name: The name of the program organization unit (POU), global variable list, or other programming object that contains the variable. For example, PLC_PRG. This is required only if it is a local variable.

variable name: The name of the variable.

Note: Even if it is a global variable without an object, you must include the period (.) before the variable name.

This Communication Driver supports all data types, but it cannot access specific parts of a variable. For example, it cannot read from or write to a single bit in an Integer variable. If you need that sort of functionality, modify your application to create more variables of the appropriate data types.

In order for this Communication Driver to communicate with the specified variables, the variables must be included as symbols in the CODESYS application. Use your application development software to make sure the symbols are included. The procedure to do this varies by device manufacturer and CODESYS version, but for an example of how to do it using the standard CODESYS V3.x software, see the appendix at the end of this document.

Examples of Item References

These are examples of valid item references for this Communication Driver. For more information about the referenced addresses, see the manufacturer's documentation for your device.

Version	Local	Global
CODESYS V2.x	PLC_PRG.initPosition1	.initPosition1
	PLC_PRG.bHMIStart	.bHMIStart
	PLC_PRG.Timer2.StartTime	.Timer2.StartTime
CODESYS V3.x	Application.PLC_PRG.initPosition1	Application.GVL.initPosition1
	Application.PLC_PRG.Timer2[1,3,0].StartTime	Application.Global_POU.Timer2[1,3,0].StartTime
	Application.PLC_PRG.DINT_Array_3Dim_2[1,-5,0]	Application.GVL.DINT_Array_3Dim_2[1,-5,0]
	Application.PLC_Local_POU.struct[9][5].Bool	Application.Global1.struct[9][5].Bool
	Application.PLC_PRG.Dint_Array_negative_index[-5].member1	Application.Global_POU.Dint_Array_negative_index[-5].member1

Chapter 4

Troubleshooting the CODESYS Communication Driver

CODESYS Communication Driver Error Codes

The following tables describe the additional error codes that you might receive when poll/poke requests and operations fail.

Code	Description	Possible Causes	Solution
1	PLC not connected	• Lost connection to the PLC due to a hardware failure, such as PLC in error node, or cables issues. • Wrong Station field configuration.	• Check the Station field configuration, confirming that the IP Addresses for the Gateway(if it is used) and the PLC are correct, as well as the PLC ID number in hexadecimal format for CODESYSV3.x. • Check if the PLC is running and if you can ping it.
2	Login to PLC has failed	Some devices only allow a log-in of one application.	If there is another program connected to the PLC, such as CODESYS programming software, you need to disconnect it (i.e.,log off). Then you should be able to communicate with the PLC.
3	No cyclic list has been found	Invalid list or no list variables to read.	Internal error related to the PLCHandler functions CycDefineVarList and CycEnterVarAccess.
4	PLC Handler is inactive	PLCHandler instance is not set active. This error happens when you use the INI file option and it is misconfigured.	Properly configure the INI file and the **Station** field.

5	Loading of the symbols has failed	There is no symbol configuration in the application.	Create the Symbol Configuration accordingly.
6	The defined communication interface is not valid or not supported	The interface is not supported (ARTI, Gateway). This error happens when trying to establish a connection with the PLC.	Check if your CODESYS configuration supports the desired interface (GATEWAY, ARTI, INI file).
7	Communication error occurred during action	• Error while trying to start the communication with the PLC. • Exceeded number of retries to receive a response from the PLC before throwing a COMM_FATAL. Related to the PLC Hanlder PlcConfig Struct.	Check if your PLC is properly configured and reachable.
8	Wrong or erroneous configuration of the PLCHandler	No configuration for this PLCHandler instance (Id unknown). This error happens when trying to establish a connection with the PLC and you are using a INI file that is not properly configured for that PLC instance.	Properly configure the INI file.
9	Invalid parameter	Invalid function parameters(for e.g.NULL).Usually happens when trying to retrieve the Variable Names from the PLC.	Internal error related to the PLCHandler functions GetAllItems, GetItem, and CycEnterVarAccess.
10	Communication interface not resp. Incorrectly installed (e.g., Gateway Dlls not available)	The interface can’t start successfully (missing interface-dependent DLLs). This error happens when trying to establish a connection with the PLC.	If you are using the Gateway, check to see if it properly installed and running.
11	Method not yet supported resp.implemented	Spare error.	Not applicable.
12	Exception occurred during action	An exception occurred in the underlying interface. This error happens when transferring any application service to the PLC.	Internal driver error related to the PLCHandler function SyncSendService.

13	Timeout time exceeded	Time for the answer on a data package from the PLC exceeded. This could be caused by a wrong Station field configuration or the PLC is unreachable.	• Check the **Station** field. • Check if you can have access to the PLC using pinging and testing the TCP/IP ports.
14	PLC already connected (at a further ::Connect function call)	The driver tried to reconnect to a PLC that is already connected.	Internal error related to the PLCHandler function Connect.
15	Reconnect thread already active	Reconnect thread is still active. This error happens when trying to establish a connection with the PLC.	Internal error related to the PLCHandler function Connect.
16	Symbols available offline	Cannot open connection to the PLC but could load the symbol file offline. This error happens when trying to establish a connection with the PLC.	Internal error related to the PLCHandler function Connect.
17	Asynchronous operation	Asynchronous operation (e.g., cyclic read of variables) has not yet finished.	Internal PLCHandler error that should never happen on this driver. Contact technical support if this error occurs.
18	ActiveX error	Internal error.	The communication driver does not use this capability of PLCHandler, if you see this error it is probably a problem with the PLCHandler. Please contact technical support.
19	Target ID mismatch	PLC does not match to the passed target ID specified.	Use the programming software to scan the network and find the correct PLC ID..
20	Object not found	No object found for the required action (e.g., tried to get an element beyond the end of the list).	Contact technical support.
21	Components not loaded	No object found for the required action (e.g.,tried to get an element beyond the end of the list).	Components required to establish communication are missing. Please contact your supplier to receive the additional files.

22	Busy	Last action still in progress, cannot start the required one.	• The driver tried to start a communication task before the previous one was completed. Contact the technical support. • If you are seeing intermittent communication problems because of this issue, please try increasing the number of retries.
23	Disabled	Driver tried to use the log feature but logging is disabled.	Contact technical support.
50	Invalid type	Results returned by the PLCHandler or specified by the driver are invalid.	Contact technical support.
51	Symbols not found	None of the variables specified match the symbols currently present in the PLC.	• Make sure that your symbols are properly added to the controller. • Verify if the name specified in the driver worksheet matches the variable name in the PLC.
52	Initialization error	The operating system does not have enough resources for the driver initialization.	Enable the protocol analyzer and run the driver again to retrieve further details.
53	Memory allocation error	• The driver could not allocate memory. • IInternal programming error in the driver.	• Verify the memory available on your device. • If enough memory is available, contact technical support.
54	Driver is closing	Driver could not be initialized because it is in shutdown process.	Wait for until the driver close and then retry.
55	PLCHandler returned invalid code	PLCHandler function GetLastError returned zero after a read or write failure.	Contact technical support.
0	OK	Communicating without error.	None required.

-15	Timeout waiting for message to start	• Disconnected cables. • PLC is turned off, in stop mode, or in error mode. • Wrong station number. • Wrong parity(for serial communication). • Wrong RTS/CTS configuration(for serial communication).	• Check cable wiring. • Check the PLC mode—it must be RUN. • Wrong station number. • Wrong parity(for serial communication). • Wrong RTS/CTS configuration(for serial communication).

Maximum Simultaneous Requests Settings

When using ARTI3 communication (UDP), if you configure the **Maximum** and **Maximum per station** fields under the **Simultaneous Requests** section with a value greater than 4 for each field, the CODESYS Communication Driver cannot connect to more than 4 devices.

The real maximum simultaneous requests settings cannot exceed 4 since there is a limitation in the CODESYS protocol library PLCHandler.

Each device attempts to create 1 UDP port. If the number of UDP ports created from all devices is greater than the driver's maximum simultaneous request, then it stops creating new UDP ports.

Regardless of how many devices there are, when the maximum simultaneous requests are less than or equal to 4, the driver distributes and shares those ports to all connected devices. Each device has multiple connections to the PLC (based on the maximum simultaneous requests settings), and also depends on the device's unused available channels/connection limit.

If the maximum simultaneous requests are greater than 4, the attempts of the device to open more ports fail, and the Communication Driver logic assigns an exclusive port per device. This is the reason why the driver cannot connect to more than 4 devices as all ports are being held exclusively per device if the maximum simultaneous requests are set to 4.

In conclusion, the CODESYS Communication Driver has the following limitations in functionality:

- Port sharing: The driver opens a new port for each device upto 4 UDP ports and shares them as long as maximum request setting is lesser than 4.
- Exclusive port assignment to each device: The driver does not share the ports assigned to each device with any other device once the maximum number of UDP ports are opened.

To avoid exclusive port assignment to each device, maximum simultaneous requests and maximum simultaneous requests per station should be limited to 4 each under 1 channel.

AVEVA

Chapter 5

Appendix

Configure a CODESYS V3.x application to include symbols

This appendix describes how to configure a CODESYS V3.x application to include symbols, which other programs can use to communicate with the corresponding variables.

By default, a CODESYS V3.x application does not include symbols. You must add a Symbol Configuration object to your project, configure the object to include the variables that you want to communicate with, and then rebuild the application to create the symbol file.

To configure your CODESYS V3.x application to include symbols:

1. Open your project in the CODESYS V3.x application development software.
2. In the project explorer, right-click **Application**, and then on the shortcut menu, click **Add Object**. The **Add Object** dialog box is displayed.
3. From the list of objects, select **Symbol configuration**, and then click **Open**. A new Symbol Configuration object is added to your project, and it is opened for editing.
4. In the **Symbol Configuration** object, add the variables you want to communicate with — move them from the Available variables list on the left to the Selected variables list on the right. If you do not see your variables in the Available variables list, check the following:
   a. For Local Variables (POU variables), the POU containing them must be called in a Task:
      - Add a Task Configuration object to the application.
      - Add a Task to the Task Configuration object.
      - Add the POU to the Task.
   b. For Global Variables, at least one of the variables from the Group must be used in at least one POU that is being called by one Task.
5. Close the **Symbol Configuration** object.
6. On the **Build** menu, click **Rebuild Application**.

Once your CODESYS application is configured to include the selected variables, it will automatically create the symbol file and then send it with the application to the runtime.

Note: The location or file path of the symbol file. You might need to copy it to another location so that it can be accessed by other programs.