<div style="background:green;color:white">**SoMachine Communication Driver**</div>

Driver for TCP/IP and Serial Communication
to SoMachine devices

Contents

Introduction

The SOMAC driver enables communication between the Studio system and Soft PLC using the Codesys Interface PLC over TCP/IP and Serial via ARTI or Gateway, according to the specifications discussed in this document.

This document was designed to help you install, configure and execute the SOMAC driver to enable communication with these devices. The information in this document is organized as follows:

- **Introduction**: Provides an overview of the SOMAC driver documentation.
- **General Information**: Provides information needed to identify all the required components (hardware and software) used to implement communication between Studio and the SOMAC driver.
- **Installing the Driver**: Explains how to install the SOMAC driver.
- **Configuring the Driver**: Explains how to configure the SOMAC driver.
- **Executing the Driver**: Explains how to execute the driver to verify that you installed and configured the driver correctly.
- **Troubleshooting**: Lists the most common error codes for this protocol and explains how to fix these errors.
- **Sample Application**: Explains how to use a sample application to test the SOMAC driver configuration.
- **Revision History**: Provides a log of all modifications made to the driver and the documentation.

✎ **Notes:**
- This document assumes that you have read the "Development Environment" chapter in the Studio *Technical Reference Manual*.
- This document also assumes that you are familiar with the Windows environment.
  If you are unfamiliar with Windows, we suggest using the **Help** feature (available from the Windows desktop **Start** menu) as you work through this guide.

General Information

This chapter explains how to identify all the hardware and software components used to implement communication between the Studio SOMAC driver and the SOMAC Runtime.

The information is organized into the following sections:

- Device Characteristics
- Link Characteristics
- Driver Characteristics
- Conformance Testing

Device Characteristics

To establish communication, you must use devices with the following specifications:

- **Manufacturer**: 3S Software;
- **Compatible PLC Runtime**:
  - Modicon M241 PLC
  - Modicon M251
  - PacDrive 3 LMC Eco / Pro / Pro 2
- **Programmer Software**: SoMachine

Link Characteristics

To establish communication, you must use links with the following specifications:

- **Device Communication Port**: Ethernet Port or Serial Port;
- **Physical Protocol**: Ethernet/TCP/IP or Serial RS-232, RS-485;
- **Logic Protocol**: ARTI or Gateway;
- **Device Runtime Software**: SoMachine
- **Specific PC Board**: None;

Driver Characteristics

The SOMAC driver is composed of the following files:

- `SOMAC.INI`: Internal driver file. *You must not modify this file.*
- `SOMAC.MSG`: Internal driver file containing error messages for each error code. *You must not modify this file.*
- `SOMAC.PDF`: Document providing detailed information about the SOMAC driver.
- `SOMAC.DLL`: Compiled driver.

✎ **Notes:**
- All of the preceding files are installed in the `/DRV` subdirectory of the Studio installation directory.

You can use the SOMAC driver on the following operating systems:

- Windows 7/8
- Windows CE ARMV4i, ARMV4 and x86

For a list of the operating systems used for conformance testing, see "Conformance Testing" on page 4.

The SOMAC driver supports the following flair registers:

Register Type	Write	Read	Bit	Byte	Integer	Float	DWord
NAME	•	•	•	•	•	•	•

> ➲ **Attention:**
> All the data types are supported, but this version of the driver does not allow you to access parts of a specific variable. For instance, if you have an integer variable you cannot read or set only one single bit. If this is necessary, you need to create a Boolean variable inside the CodeSys Runtime and use it to set/read only that bit number.

Conformance Testing

The following hardware/software was used for conformance testing:

- **Driver Configuration**: PLC Program **SOMAC30.project**
- **Cable**: Ethernet Cable

Driver Version	Studio Version	Operating System (Development)	Operating System (Target)	Equipment
3.2	8.1	Windows 8	▪ Windows 7/8 ▪ Windows CE 7.0	**Equipment**: Runtime: • - Modicon M241 PLC • Modicon M251 • PacDrive 3 LMC Eco / Pro / Pro 2

Installing the Driver

When you install Studio version 6.1 or higher, all of the communication drivers are installed automatically. You must select the driver that is appropriate for the application you are using.

Perform the following steps to select the driver from within the application:

1. Open Studio from the **Start** menu.
2. From the Studio main menu bar, select **File** → **Open Project** to open your application.
3. Select **Insert** → **Driver** from the main menu bar to open the *Communication Drivers* dialog.
4. Select the **SOMAC** driver from the *Available Drivers* list, and then click the **Select** button:
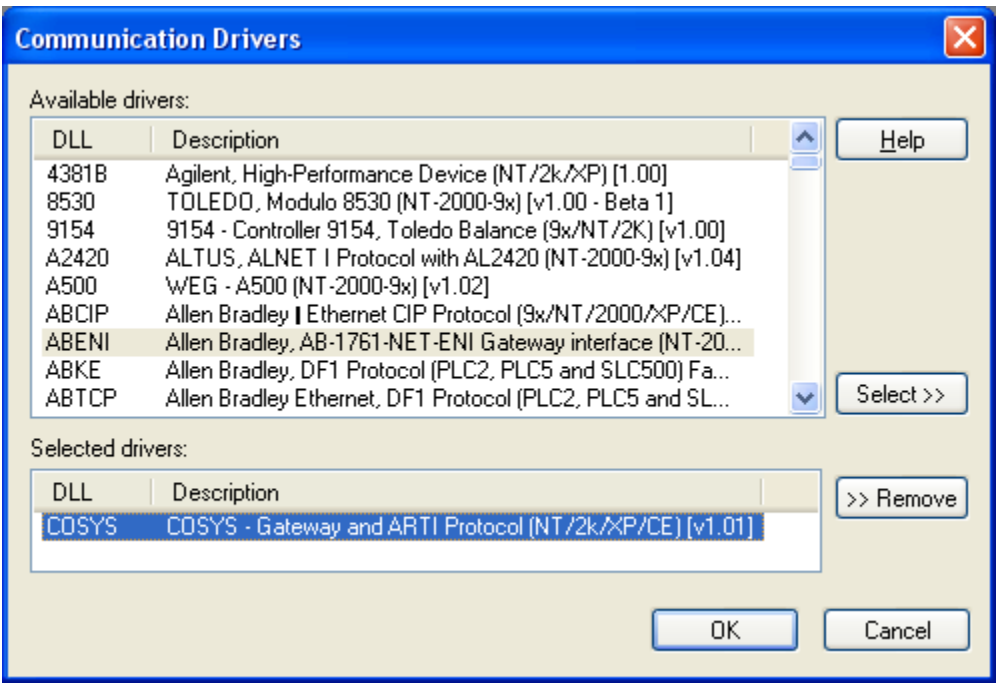
Communication Drivers Dialog

5. When the **SOMAC** driver displays in the **Selected Drivers** list, click the **OK** button to close the dialog.

> ➲ **Attention:**
> For safety reasons, you must use special precautions when installing the physical hardware. Consult the hardware manufacturer's documentation for specific instructions in this area.

Configuring the Driver

After opening Studio and selecting the SOMAC driver, you must configure the driver. Configuring the SOMAC driver is done in three parts:

- Setting your SoMachine project to be able to communicate with external applications.
- Specifying communication parameters
- Defining tags and controls in the *MAIN* and *STANDARD DRIVER SHEET*s (or Communication tables)

    Worksheets are divided into two sections, a *Header* and a *Body*. The fields contained in these two sections are standard for all communications drivers — except the **Station**, **Header** and **Address** fields, which are driver-specific. This document explains how to configure the **Station**, **Header** and **Address** fields only.
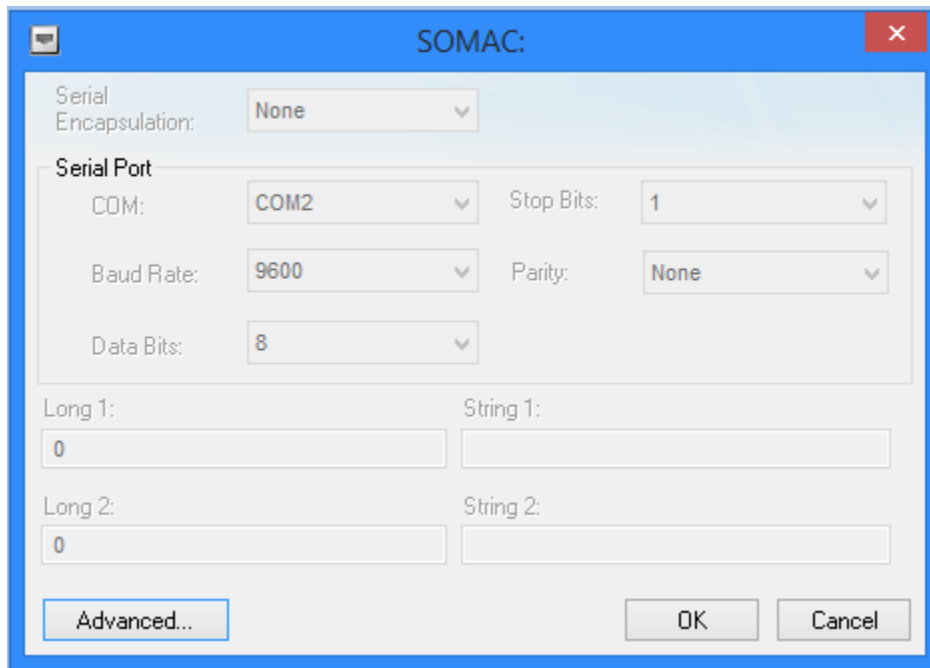
> ✍ **Note:**
> For a detailed description of the Studio *MAIN* and *STANDARD DRIVER SHEET*s, and information about configuring the standard fields, review the product's *Technical Reference Manual*.

Setting the Communication Parameters

Use the following steps to configure the communication parameters, which are valid for all *Driver* worksheets configured in the system:

1. From the Studio development environment, select the **Comm** tab located below the *Workspace* pane*.*
2. Click on the *Drivers* folder in the *Workspace* pane to expand the folder.
3. Right-click on the *SOMAC* subfolder and when the pop-up menu displays, select the **Settings** option:

   The *SOMAC: Communication Parameters* dialog displays:

Communication Parameters Dialog

4. Click the **Advanced** button on the *Communication Parameters* dialog to open the *Advanced Settings* dialog and configure the necessary settings.

✎ **Notes:**

 - Do not change any of the other *Advanced* parameters at this time. You can consult the Studio *Technical Reference Manual* for information about configuring these parameters for future reference.

Configuring the Driver Worksheets

This section explains how to configure the *MAIN* and *STANDARD DRIVER SHEETs* (or Communication tables) to associate application tags with the device addresses. You can configure multiple *Driver* worksheets — each of which is divided into a *Header* section and *Body* section.

Configuring the MAIN DRIVER SHEET

When you add the SOMAC driver to your application, Studio automatically adds a *MAIN DRIVER SHEET* to the driver folder:

You can use this worksheet (similar to the following picture) to associate Studio tags to addresses in the PLC:

Main Driver Sheet

✎ **Note:**

Most of the *MAIN DRIVER SHEET* parameters are standard for all drivers, and are not discussed in this document. Instructions for configuring these standard parameters are provided in the Studio *Technical Reference Manual*.

Use the following information to configure the **Station** and **Address** parameters specific to this driver:

 - **Station:** Type a set of parameters that is used to connect with SoMachine Runtime, see the format below:

> ✎ **Note:**
>
> In the next listed settings, **[Parameter]** (between brackets) means that the parameter is optional and **<Parameter>** indicates that the parameter is mandatory.

Connecting to Runtime using ARTI3:

ARTI3, <Runtime Address/Device Name>

- ☐ **<Runtime Address/Device Name>:** Runtime Device Hexadecimal Address or Device Name specified. Please notice the Device Name is case sensitive.

Examples:

Station	Description
ARTI3,MY_PLC_NAME	Connects to the SoMachine runtime V3.x, with the Device Name MY_PLC_NAME, using the ARTI3 protocol.

Connecting to SoMachine 3.x Runtime using Gateway:

[optGatewayIPAddress:]<Runtime Address/Device Name> [:optPortNumber]

- ☐ **[optGatewayIPAddress:]** Type here the IP Address of the gateway which will give you access to the Runtime. This parameter is option and if omitted, the driver will connect using the IP address 127.0.0.1.
- ☐ **<Runtime Address/Device Name>:** Runtime Device Hexadecimal Address or Device Name specified. Please notice the Device Name is case sensitive.
- ☐ **[:optPortNumber >:** TCP/IP Port number to establish communication with the runtime. If not specified, 1217 will be used

Examples:

Station	Description
192.168.1.10:0A56	Connects to Runtime 0A56 via gateway at IP address 192.168.1.10 on port 1217
127.0.0.1:0A56	Connects to Runtime 0A56 via gateway at IP address 127.0.0.1 on port 1217
0A56:1480	Connects to Runtime 0A56 via gateway at IP address 127.0.0.1 on port 1480
MY-DEVICE-10:1480	Connects to Runtime with name MY-DEVICE-10 via gateway at IP address 127.0.0.1 on port 1480.
192.168.1.10:0A56:1480	Connects to Runtime 0A56 via gateway at IP address 192.168.1.10 on port 1480

- The syntax is `<Application Name>.<Object Name>.<Variable Name>`

  For example:

  Application.GVL.bHMIStart

  Application.PLC_PRG.bCycleCompleted

### *Connecting to Runtime using settings from a OPCServer.ini file:*
`CFILE,<Configuration File>,<Configuration ID>`

   `<Configuration File>,<:` You can specify a path relative to the application path.

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

Examples:

Station	Description
CFILE,.\A\File.ini,PLC:PLC_Id23	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. For example:
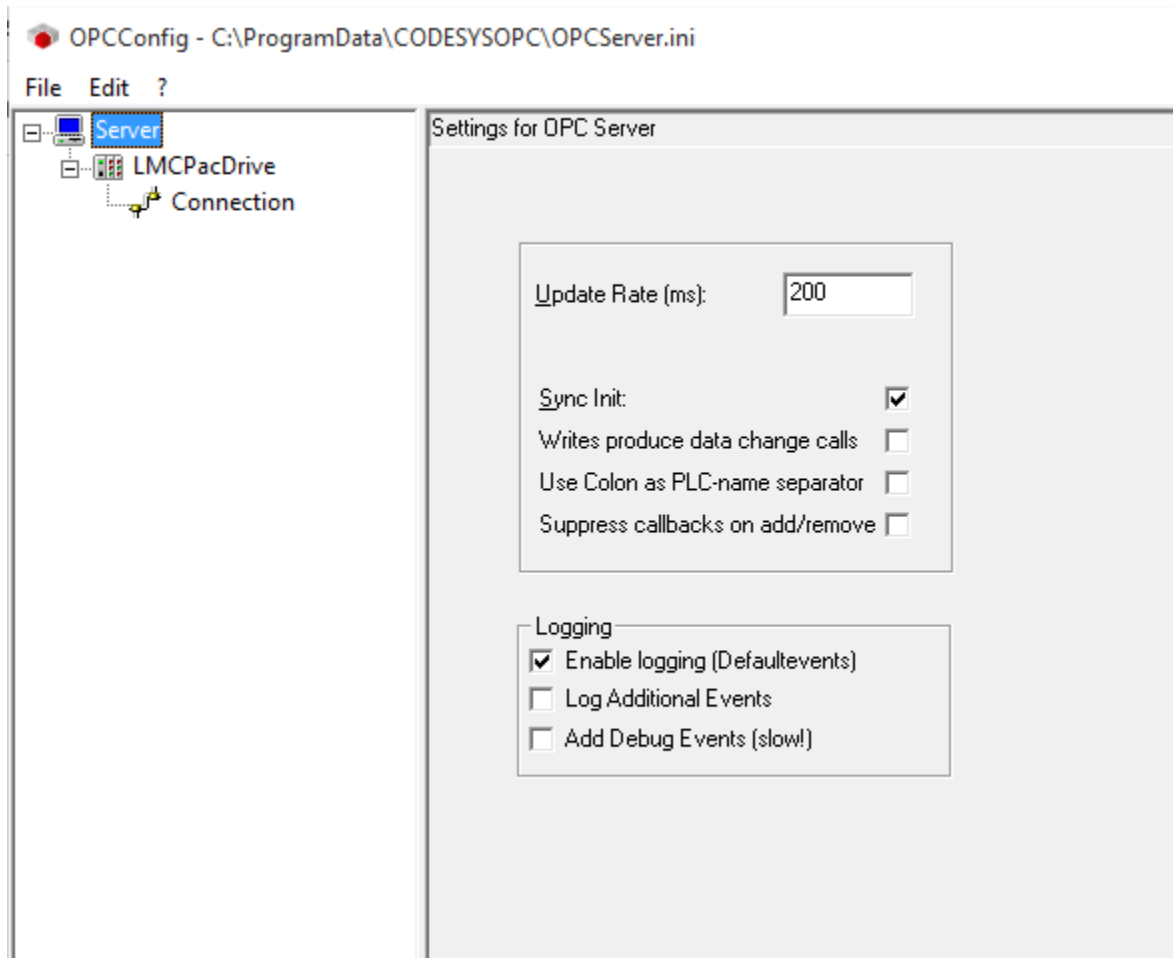
✎ From the file menu click Save As and choose a location to save the file in. This is the first parameter in the example of station `CFILE,C:\Config\OPCServer.ini,0`

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`
✎Note the Parameter under [PLC:PLC_Id] field is the PLC name which pertains to the controller you want to communicate with. This corresponds to `PLC:PLC_Id23` in the example station : `CFILE,C:\Config\OPCServer.ini,PLC:PLC_Id23`

Configuring the *STANDARD DRIVER SHEET*

To insert a new Standard Driver Worksheet:

1. In the *Comm* tab, open the *Drivers* folder and locate the *SOMAC* subfolder.
2. Right-click on the *SOMAC* subfolder, and then select **Insert** from the pop-up menu:

A new SOMAC driver worksheet is inserted into the *SOMAC* subfolder, and the worksheet is open for configuration:

Standard Driver Sheet

In general, all parameters on the *Driver* worksheet (except the **Station**, **Header** and **Address** fields) are standard for all communication drivers, but they will not be discussed in this document. For detailed information about configuring the standard parameters, consult the *Studio Technical Reference Manual*.

3. Configure the **Station** and **Header** fields as follows:...

▪ **Station:** Type a set of parameters that is used to connect with CodeSys Runtime, see the format below:

☞ **Note:**

In the next listed settings, **[Parameter]** (between brackets) means that the parameter is optional and **<Parameter>** indicates that the parameter is mandatory.

Connecting to SoMachine 3.x Runtime:

`[optGatewayIPAddress:]<Runtime Address/Device Name>`

- ☐ **`[optGatewayIPAddress:]`** Type here the IP Address of the gateway which will give you access to the Runtime. This parameter is option and if omitted, the driver will connect using the IP address 127.0.0.1.
- ☐ **`<Runtime Address/Device Name>`**: Runtime Device Hexadecimal Address or device name is specified..

Examples:

Station	Description
192.168.1.10:0A56	Connects to Runtime 0A56 via gateway at IP address 192.168.1.10
127.0.0.1:0A56	Connects to Runtime 0A56 via gateway at IP address 127.0.0.1
127.0.0.1:MY-DEVICE-10	Connects to Runtime with name MY-DEVICE-10 via gateway at IP address 127.0.0.1

- The syntax is **`<Application Name>.<Object Name>.<Variable Name>`**

  For example**:**

    Application.GVL.bHMIStart

    Application.PLC_PRG.bCycleCompleted

Executing the Driver

After adding the SOMAC driver to a project, Studio sets the project to execute the driver automatically when you start the Runtime environment.

To verify that the *Driver Runtime* task is enabled and will start correctly, perform the following steps:

1. Select **Project → Status** from the main menu bar.

    The *Project Status* dialog displays:
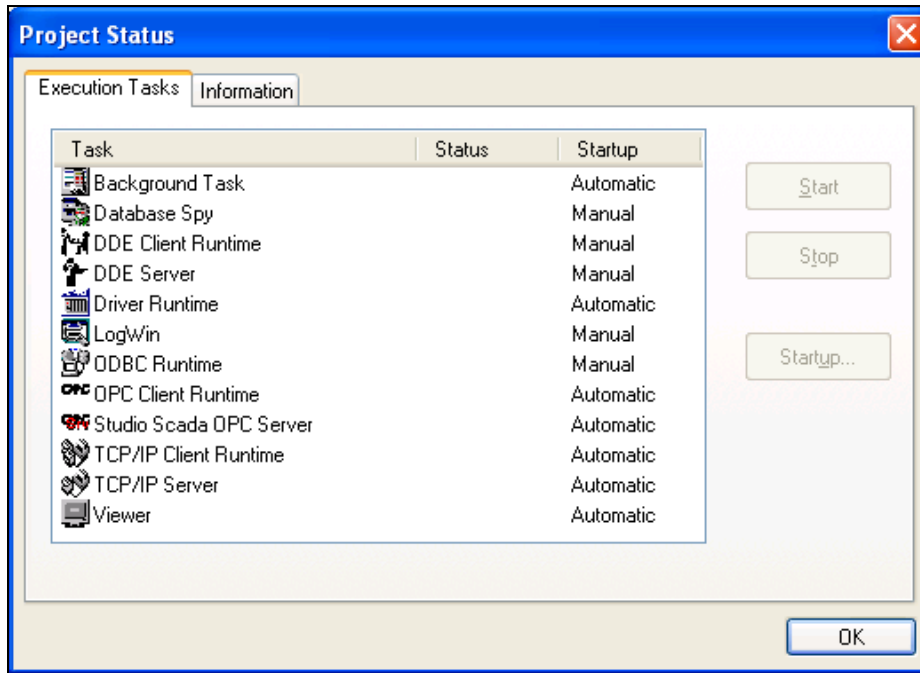
Project Status Dialog

2. Verify that the *Driver Runtime* task is set to **Automatic**.
    –   If the setting is correct, click **OK** to close the dialog.
    –   If the **Driver Runtime** task is set to **Manual**, select the **Driver Runtime** line. When the **Startup** button becomes active, click the button to toggle the *Startup* mode to **Automatic**.
3. Click **OK** to close the *Project Status* dialog.
4. Start the application to run the driver

Troubleshooting

If the SOMAC driver fails to communicate with the device, the tag you configured for the **Read Status** or **Write Status** fields will receive an error code. Use this error code and the following table to identify what kind of failure occurred.

Error Code	Description	Possible Causes	Procedure to Solve
0	Ok	▪ No errors	▪
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 HEX format for CodeSys 3.x ▪ Check if the PLC is running and if you can ping it
2	Login to PLC has failed	▪ Some runtime systems only allow a log-in of one application.	▪ If there is another program connected to the PLC, such as CodeSys programming software, you would need to disconnect it (Log off) and then you should be able to start communicating using the Driver
3	No cyclic list has been found	▪ invalid list or no list variables to read	▪ Internal Driver Error related to the ::CycDefineVarList and ::CycEnterVarAccess functions
4	PLCHandler is inactive	▪ PLCHandler instance isn't 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 Runtime system	▪ Create the Symbol Configuration accordingly
6	The defined communication interface is not valid or not supported	▪ The interface isn't 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 ocurred 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 PLCHanlder 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	▪ Iinternal Driver error related to ::GetAllItems, ::GetItem and ::CycEnterVarAccess functions
10	Communication interface not resp. Incorrectly installed (e.g. Gateway Dlls not available)	▪ The interface can't start successfully (missing interface dependent Dll's). 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

Error Code	Description	Possible Causes	Procedure to Solve
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 transfering any runtime system 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() call)	▪ The driver tried to reconnect to a PLC that is already connected	▪ Internal driver error related to the 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 driver error related to the 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 driver error related to the function ::Connect
17	Asynchronous operation	▪ Asynchronous operation (e. g. cyclic read of variables) has not yet finished	▪ Internal PLCHandler error, should never happen on this driver. Contact technical support if this error happens.
18	ActiveX Error	▪ Internal error, contact technical support	▪ 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, can not 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
24	PLC failure	▪ Communication to the PLC was successful, but the PLC has returned a bad result	▪ Troubleshoot driver settings. Restart driver communication. Contact technical support if required.
50	Invalid Type	▪ Results returned by the PLCHandler or	▪ Contact technical support

Error Code	Description	Possible Causes	Procedure to Solve
		specified by the driver are invalid	
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 ▪ Internal 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 GetLastError call returned zero after a read or write failure	▪ Contact technical support

---

> ⇨ **Tip:**
>
> You can verify communication status using the Studio development environment *Output* window (*LogWin* module). To establish an event log for **Field Read Commands** and **Field Write Commands,** right-click in the *Output* window. When the pop-up menu displays, select the option to set the log events. If you are testing a Windows CE target, you can use the Remote LogWin of Studio (**Tools** → **Remote LogWin**) to get the log events from the target unit remotely.

To test communication with Studio, we recommend using the sample application provided rather than your new application.

If you must contact us for technical support, please have the following information available:

- **Operating System** (type and version): To find this information, select **Tools** → **System Information**.
- **Studio Version**: To find this information, select **Help** → **About.**
- **Driver Version**: To find this information, read the full description of the driver on the *Communication Drivers* dialog.
- **Communication Log**: Displays in the Studio *Output* window (or *LogWin* window) when the driver is running. Be sure to enable the **Field Read Commands** and **Field Write Commands** for the *LogWin* window.
- **Device Model** and **Boards**: Consult the hardware manufacturer's documentation for this information.

Revision History

Doc. Revision	Driver Version	Date	Description of changes
A	1.1	June/01/2009	First driver version
B	1.2	Aug/2009	Modified the PLCHandler version used in the driver
C	2.0	Feb/2011	- Modified the PLCHandler version used in the driver - Created option to specify the port number - Updated the driver to improve performance. - Added support for simultaneous connections
D	2.1	Aug/2011	Fixed a memory leak in the CE version
E	2.1	Sep/2011	Updated Conformance Testing.
F	2.2	Feb/2012	- Improved driver performance - Updated PLC Handler version - Improved error reporting
G	2.2	May/2012	- Added DLL requirement on documentation
H	2.3	Aug/2013	- Fixed memory leak
I	2.4	Feb/2014	- Modified the driver to use a static link with the PLCHandler
J	2.5	Sep/2014	- Fixed timestamp issues - Modified the driver so that connection can be established using the device name. - Modified to support ARTI3 for communication
K	2.6	Oct/2014	- Upgraded to PLCHandler 3.5.5 to fix issue with reconnection and to properly run on WinCE5 ARM processors
L	2.7	Nov/2015	- Changed station field to accept device names containing spaces
M	2.8	Sep/15/2016	- Upgraded to PLC Handler 3.5.8 - Improved support for 64 bit integers.
N	2.9	Oct/28/2016	- Changed boolean variables to accept any non-zero value as TRUE.
O	3.0	Apr/14/2017	- Fixed a bug for boolean variables in main driver sheets
P	3.1	July/24/2017	- Improved symbol loading frequency.
Q	3.2	Mar/16/2018	-Fixed issue with using write trigger on standard driver sheets