COSYS Communication Driver

Driver for TCP/IP and Serial Communication
Using PLC Handler CodeSys Library

Contents

Introduction

The COSYS 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 COSYS driver to enable communication with these devices. The information in this document is organized as follows:

- **Introduction**: Provides an overview of the COSYS driver documentation.
- **General Information**: Provides information needed to identify all the required components (hardware and software) used to implement communication between Studio and the COSYS driver.
- **Installing the Driver**: Explains how to install the COSYS driver.
- **Configuring the Driver**: Explains how to configure the COSYS 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 COSYS 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 COSYS driver and the COSYS 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**:

  Any CodeSys runtime compatible with the PLHandler library, such as
  - Codesys SP WIN V3 Control
  - Codesys SP PLCWinNT V2.4;
  - EATON - XC-CPU202
  - Schneider Electric –M241/M251 PLCs
  - WAGO PLC running CodeSys (CPU 750-841)
- **Programmer Software**: Codesys V3.x or Codesys V2.x

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**: Codesys Runtime Software;
- **Specific PC Board**: None;

Driver Characteristics

The COSYS driver is composed of the following files:

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

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

You can use the COSYS 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 5.

The COSYS 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 **Cosys30.project**
- **Cable**: Ethernet Cable

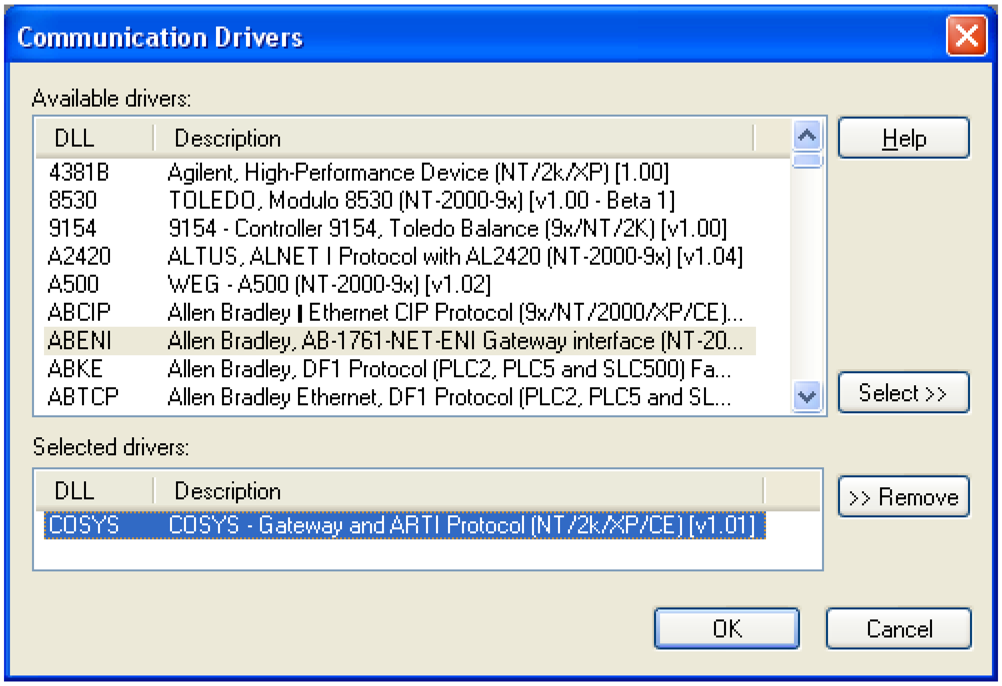
Driver Version	Studio Version	Operating System (Development)	Operating System (Target)	Equipment
3.3	8.1 + SP2	Windows 8	▪ Windows 7/8 ▪ Windows CE 7.0 ▪ Windows Embedded Standard 7.0	**Equipment**: Runtime: - Codesys SP WIN V3.5.8 Control - Codesys SP PLCWinNT V2.4.4.0 - Wago CPU 750-841 - CodeSys v3.2 for Kep France running on WinCE 5.0 ARMV4i - EATON XC-CPU202 Codesys 3.5

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 **COSYS** driver from the *Available Drivers* list, and then click the **Select** button:



Available drivers:

DLL	Description
4381B	Agilent, High-Performance Device (NT/2k/XP) [1.00]
8530	TOLEDO, Modulo 8530 (NT-2000-9x) [v1.00 - Beta 1]
9154	9154 - Controller 9154, Toledo Balance (9x/NT/2K) [v1.00]
A2420	ALTUS, ALNET I Protocol with AL2420 (NT-2000-9x) [v1.04]
A500	WEG - A500 (NT-2000-9x) [v1.02]
ABCIP	Allen Bradley I Ethernet CIP Protocol (9x/NT/2000/XP/CE)...
ABENI	Allen Bradley, AB-1761-NET-ENI Gateway interface (NT-20...
ABKE	Allen Bradley, DF1 Protocol (PLC2, PLC5 and SLC500) Fa...
ABTCP	Allen Bradley Ethernet, DF1 Protocol (PLC2, PLC5 and SL...

Selected drivers:

DLL	Description
COSYS	COSYS - Gateway and ARTI Protocol (NT/2k/XP/CE) [v1.01]

Communication Drivers Dialog

5. When the **COSYS** 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 COSYS driver, you must configure the driver. Configuring the COSYS driver is done in three parts:

- Setting your CodeSys 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*.

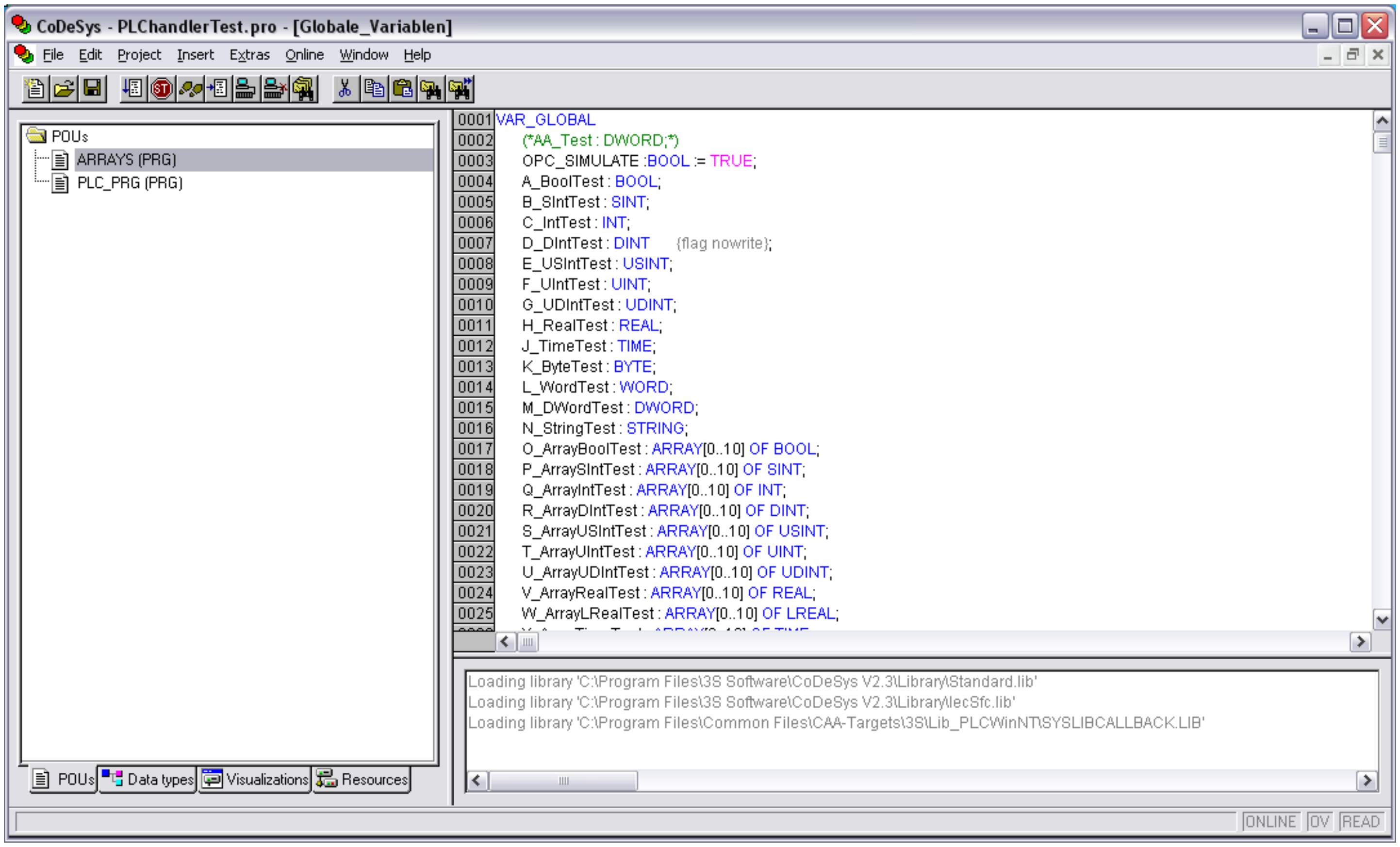
Configuring CodeSys projects to communicate with External Applications

Each version of Codesys is configured in a particular way. So this section will be divided in two parts, CodeSys 2.X and CodeSys 3.X.

1. CodeSys 2.X

In order to communicate with the Variables configured in a CodeSys 2.x Application, you must configure it to generate and download the Symbol Files.

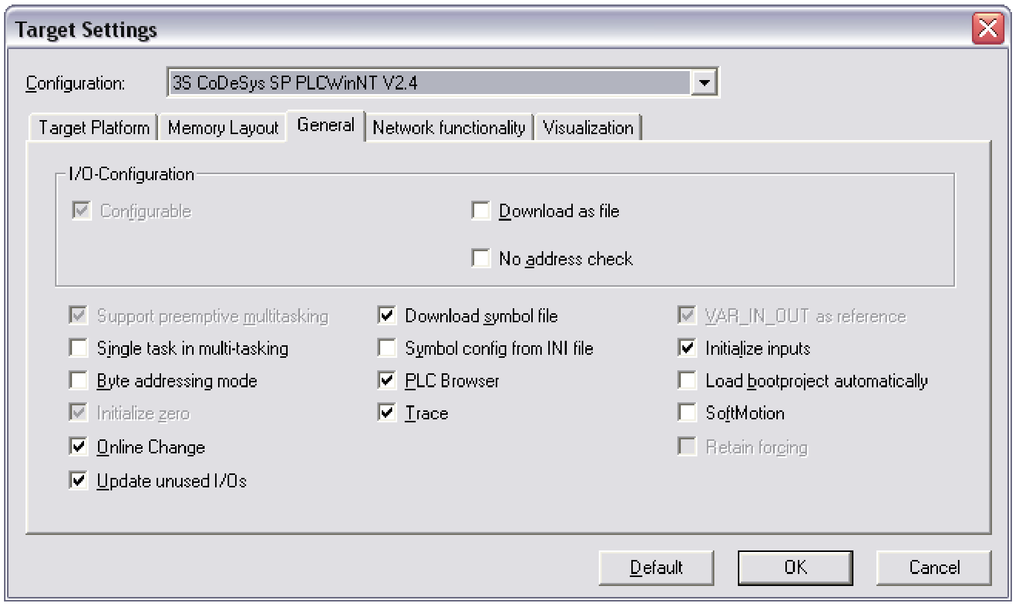
1.1 Open your CodeSys project.


CoDeSys - PLChandlerTest.pro - [Globale_Variablen]
File Edit Project Insert Extras Online Window Help
POUs
ARRAYS (PRG)
PLC_PRG (PRG)
POUs
Data types
Visualizations
Resources
ONLINE
OV
READ

```
VAR_GLOBAL
    (*AA_Test : DWORD;*)
    OPC_SIMULATE :BOOL := TRUE;
    A_BoolTest : BOOL;
    B_SIntTest : SINT;
    C_IntTest : INT;
    D_DIntTest : DINT      {flag nowrite};
    E_USIntTest : USINT;
    F_UIntTest : UINT;
    G_UDIntTest : UDINT;
    H_RealTest : REAL;
    J_TimeTest : TIME;
    K_ByteTest : BYTE;
    L_WordTest : WORD;
    M_DWordTest : DWORD;
    N_StringTest : STRING;
    O_ArrayBoolTest : ARRAY[0..10] OF BOOL;
    P_ArraySIntTest : ARRAY[0..10] OF SINT;
    Q_ArrayIntTest : ARRAY[0..10] OF INT;
    R_ArrayDIntTest : ARRAY[0..10] OF DINT;
    S_ArrayUSIntTest : ARRAY[0..10] OF USINT;
    T_ArrayUIntTest : ARRAY[0..10] OF UINT;
    U_ArrayUDIntTest : ARRAY[0..10] OF UDINT;
    V_ArrayRealTest : ARRAY[0..10] OF REAL;
    W_ArrayLRealTest : ARRAY[0..10] OF LREAL;
```

```
Loading library 'C:\Program Files\3S Software\CoDeSys V2.3\Library\Standard.lib'
Loading library 'C:\Program Files\3S Software\CoDeSys V2.3\Library\IecSfc.lib'
Loading library 'C:\Program Files\Common Files\CAA-Targets\3S\Lib_PLCWinNT\SYSLIBCALLBACK.LIB'
```

1.2 In the Resources tab go to ***Target Settings*** and open the *General Tab* of the *Target Settings* dialog.



Configuration: 3S CoDeSys SP PLCWinNT V2.4

I/O-Configuration
- [✓] Configurable
- [ ] Download as file
- [ ] No address check

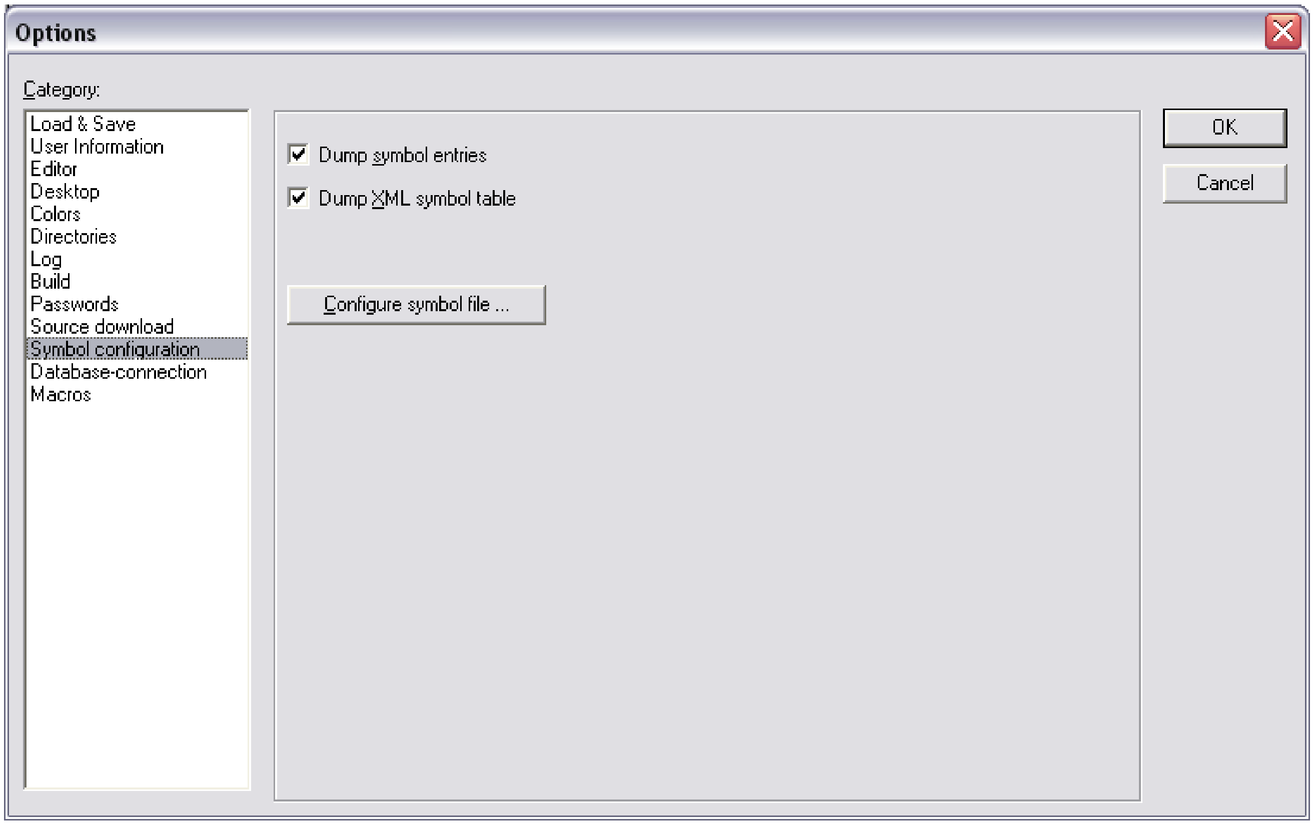
- [✓] Support preemptive multitasking
- [ ] Single task in multi-tasking
- [ ] Byte addressing mode
- [✓] Initialize zero
- [✓] Online Change
- [✓] Update unused I/Os
- [✓] Download symbol file
- [ ] Symbol config from INI file
- [✓] PLC Browser
- [✓] Trace
- [✓] VAR_IN_OUT as reference
- [✓] Initialize inputs
- [ ] Load bootproject automatically
- [ ] SoftMotion
- [ ] Retain forcing

Make sure that the checkbox **Download Symbol File** is marked.

If this check box was not checked in your project make sure that you download the new version (option checked) into the CodeSys Runtime.

1.3 Click **OK**, and then Double Click the option **Workspace**.

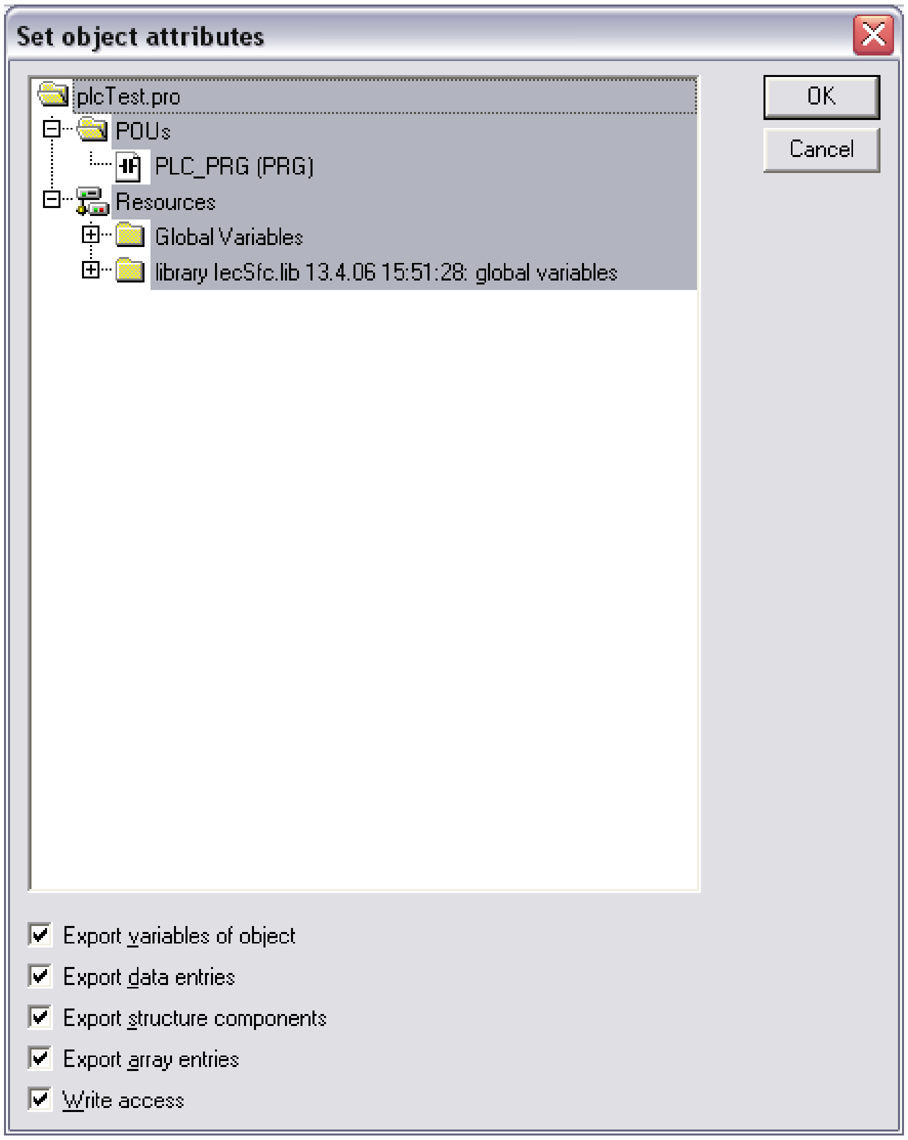
1.4 Select **Symbol Configuration** from the Category List and check all options available, then click ***Configure Symbol File***



☑ Dump symbol entries

☑ Dump XML symbol table

Check all the available options again and selected all the variables you want to be able to be read and write from an external application. Then click ok.



plcTest.pro
- POUs
  - PLC_PRG (PRG)
- Resources
  - Global Variables
  - library IecSfc.lib 13.4.06 15:51:28: global variables

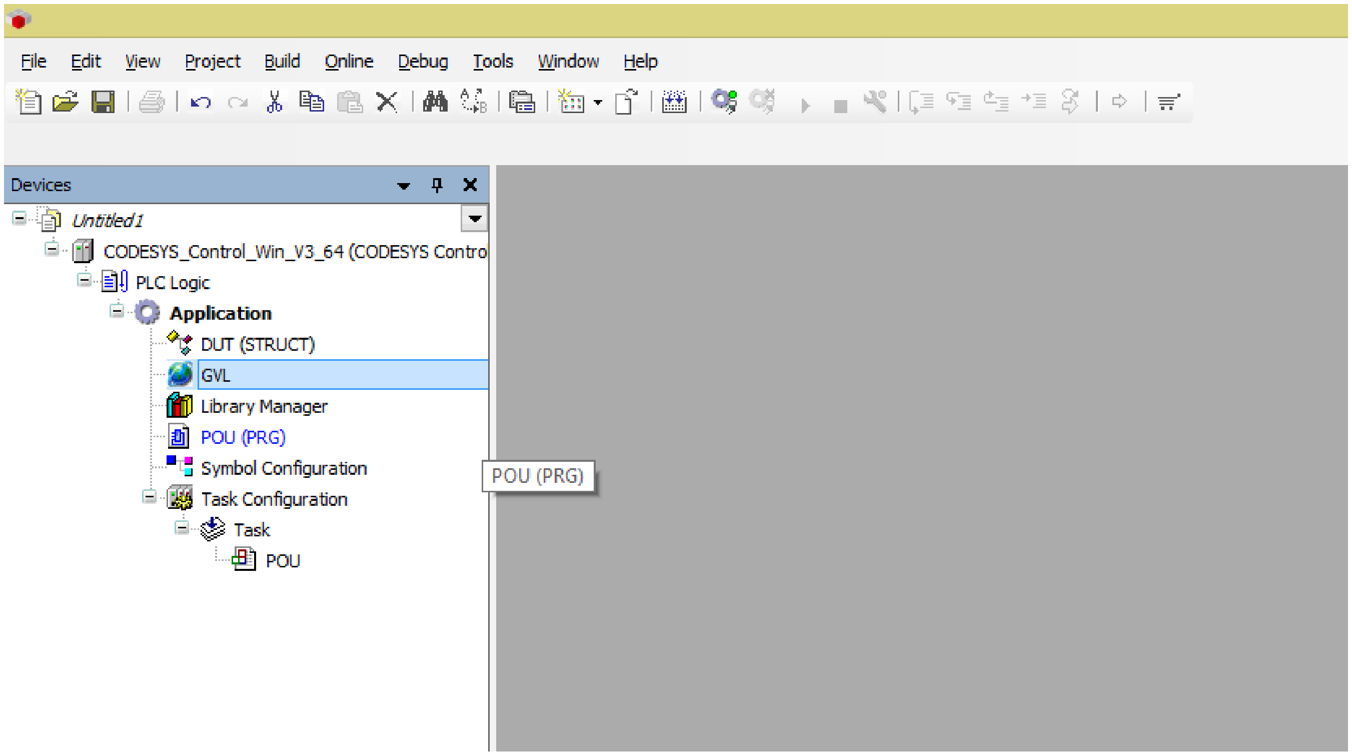
- [✓] Export variables of object
- [✓] Export data entries
- [✓] Export structure components
- [✓] Export array entries
- [✓] Write access

1.5 Go to the menu **Project->Clean All** followed by **Project->Rebuild All**
The next time that you connect with the Runtime system, you will need to download the application again, and it will include the symbol files.

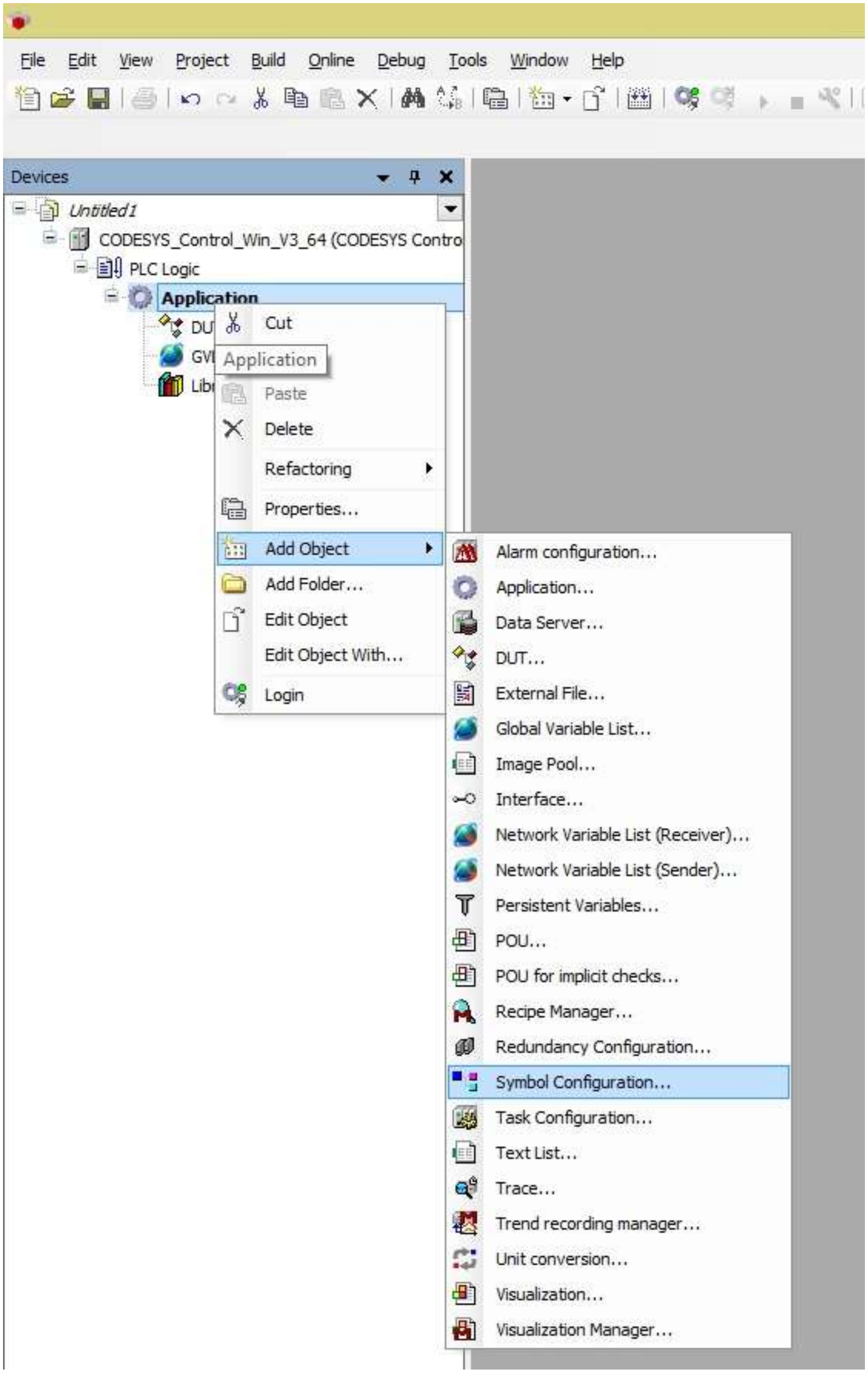
2. CodeSys 3.X

In order to communicate with a runtime CodeSys application developed using the versions 3.x, you must be sure that you properly configure a **Symbol Configuration** object in your project.

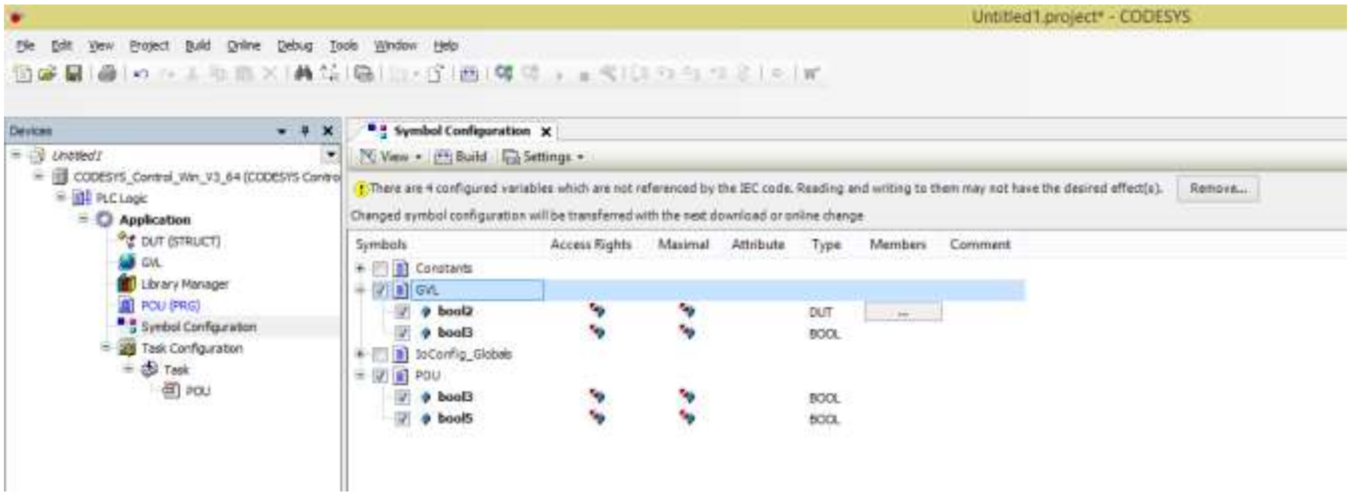
2.1 Open your CodeSys 3.X project.



2.2 Right-click on the **Application** name and select *Add Object.* Select ***Symbol Configuration.***


File
Edit
View
Project
Build
Online
Debug
Tools
Window
Help
Devices
Untitled1
CODESYS_Control_Win_V3_64 (CODESYS Contro
PLC Logic
Application
Cut
Application
Paste
Delete
Refactoring
Properties...
Add Object
Add Folder...
Edit Object
Edit Object With...
Login
Alarm configuration...
Application...
Data Server...
DUT...
External File...
Global Variable List...
Image Pool...
Interface...
Network Variable List (Receiver)...
Network Variable List (Sender)...
Persistent Variables...
POU...
POU for implicit checks...
Recipe Manager...
Redundancy Configuration...
Symbol Configuration...
Task Configuration...
Text List...
Trace...
Trend recording manager...
Unit conversion...
Visualization...
Visualization Manager...

Open the **Symbol configuration** and add the variables you want to communicate with. You need to move them from the tree-view on the left to the **Selected Variables** window on the right.



There are 4 configured variables which are not referenced by the IEC code. Reading and writing to them may not have the desired effect(s).

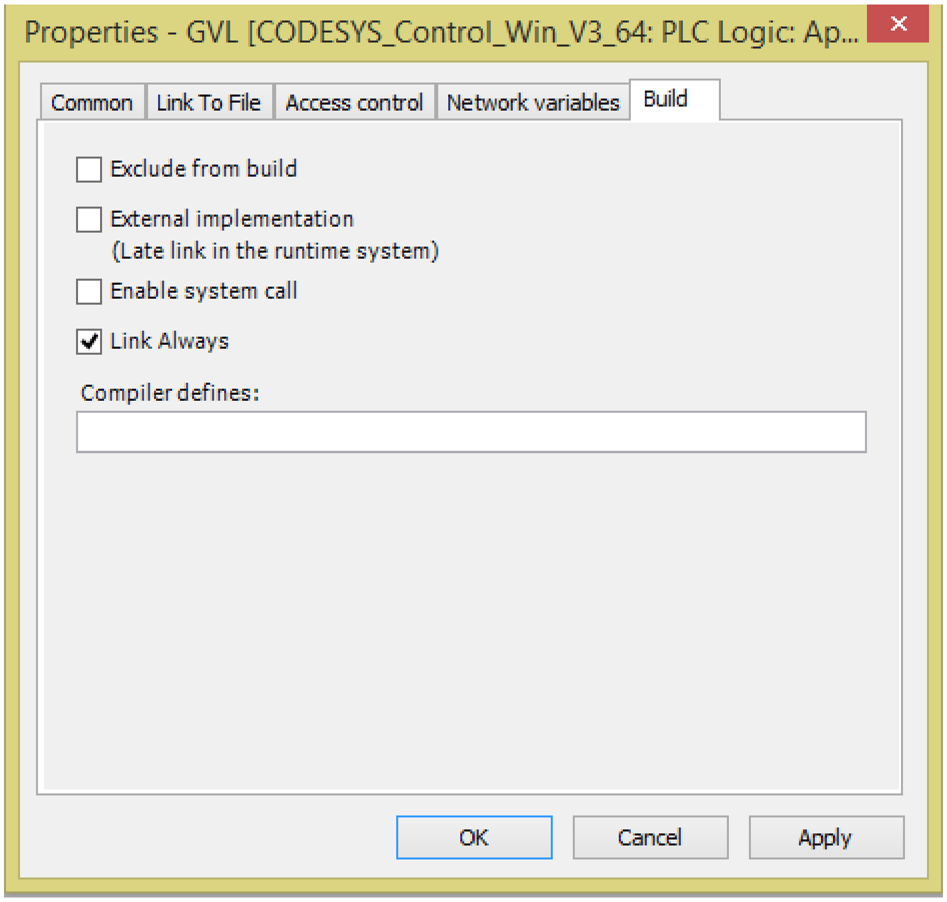
Changed symbol configuration will be transferred with the next download or online change

Symbols	Access Rights	Maximal	Attribute	Type	Members	Comment
Constants						
GVL						
bool2				DUT	...	
bool3				BOOL		
IoConfig_Globals						
POU						
bool3				BOOL		
bool5				BOOL		

If you do not see your variables on the **Available Variables** window, you need to check the following:

- For **Local Variables** (POU variables) the POU containing them must be called in a **Task**.
  - To configure a **Task**, add a **Task Configuration** object to the **Application**
  - Add a Task
  - Add the POU to the Task

- For **Global Variables**: Open the properties of Global Variables list and in the Build tab select an appropriate method to link the file.


Properties - GVL [CODESYS_Control_Win_V3_64: PLC Logic: Ap...

- [ ] Exclude from build
- [ ] External implementation (Late link in the runtime system)
- [ ] Enable system call
- [✓] Link Always

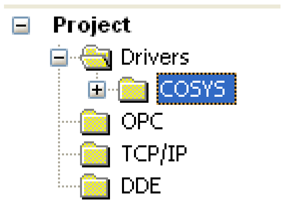
Compiler defines:

Once you configure that, go to the menu ***Build->Rebuild <Application>***
Now, when you go **Online,** you can download the program and the symbol files will be sent to the runtime system

Setting the Communication Parameters

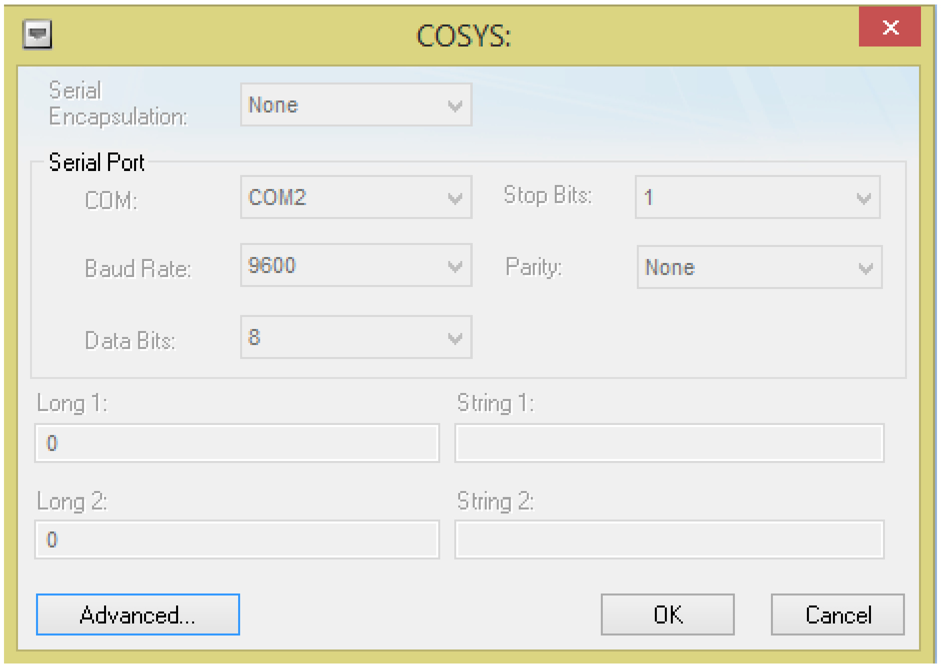
Use the following steps to configure the communication parameters, which are valid for all *Driver* worksheets configured in the system:

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 *COSYS* subfolder and when the pop-up menu displays, select the **Settings** option:



Select Settings from the Pop-Up Menu

The *COSYS: Communication Parameters* dialog displays:



Serial Encapsulation: None

Serial Port

COM: COM2

Stop Bits: 1

Baud Rate: 9600

Parity: None

Data Bits: 8

Long 1: 0

String 1:

Long 2: 0

String 2:

Communication Parameters Dialog

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

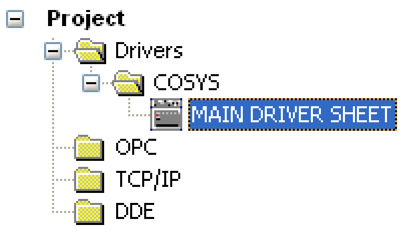
> **Notes:**
> - Do not change any of the other *Advanced* parameters at this time. You can consult the Studio *Technical Reference Manual* for information about configuring these parameters for future reference.

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.

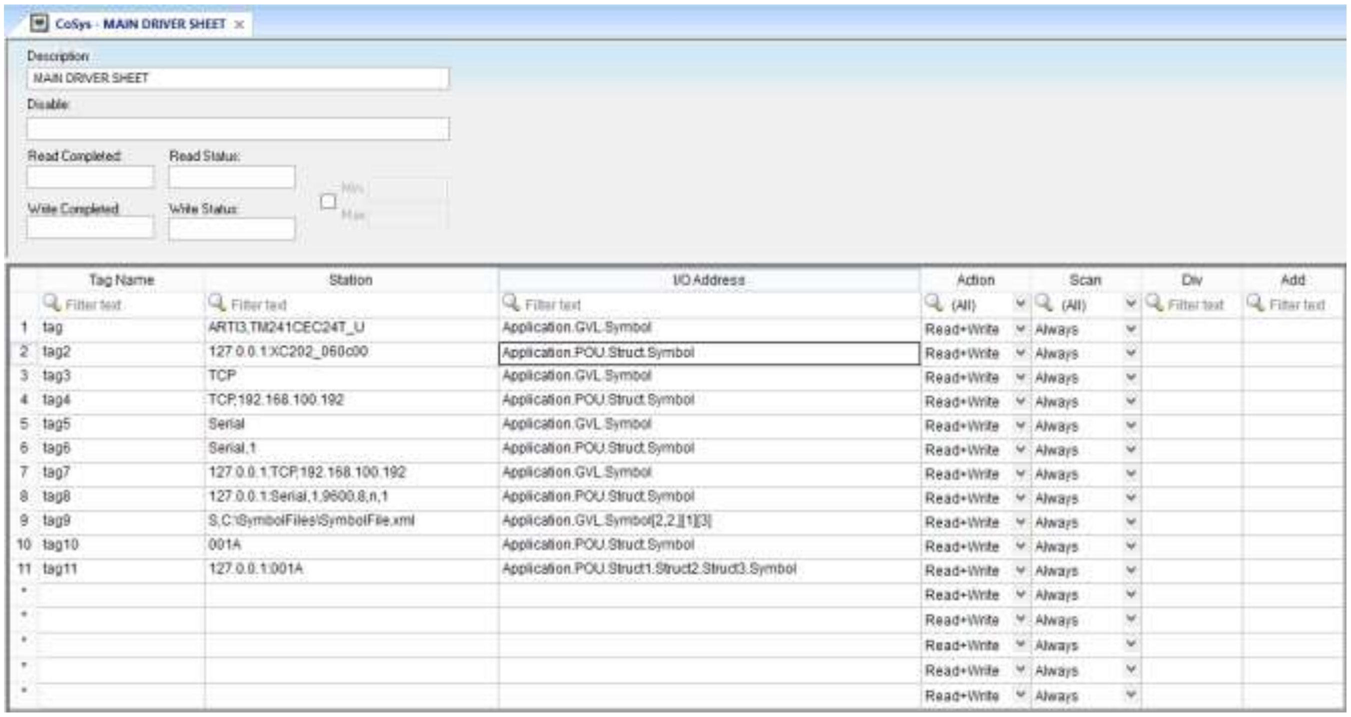
Configuring the MAIN DRIVER SHEET

When you add the COSYS driver to your application, Studio automatically adds a *MAIN DRIVER SHEET* to the driver folder, as shown in the following figure:



Main Driver Sheet

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



Description: MAIN DRIVER SHEET
Disable:
Read Completed:
Read Status:
Write Completed:
Write Status:
Min:
Max:

	Tag Name	Station	I/O Address	Action	Scan	Div	Add
	Filter text	Filter text	Filter text	(All)	(All)	Filter text	Filter text
1	tag	ARTI3,TM241CEC24T_U	Application.GVL.Symbol	Read+Write	Always		
2	tag2	127.0.0.1:XC202_060c00	Application.POU.Struct.Symbol	Read+Write	Always		
3	tag3	TCP	Application.GVL.Symbol	Read+Write	Always		
4	tag4	TCP,192.168.100.192	Application.POU.Struct.Symbol	Read+Write	Always		
5	tag5	Serial	Application.GVL.Symbol	Read+Write	Always		
6	tag6	Serial,1	Application.POU.Struct.Symbol	Read+Write	Always		
7	tag7	127.0.0.1:TCP,192.168.100.192	Application.GVL.Symbol	Read+Write	Always		
8	tag8	127.0.0.1:Serial,1,9600,8,n,1	Application.POU.Struct.Symbol	Read+Write	Always		
9	tag9	S,C:\SymbolFiles\SymbolFile.xml	Application.GVL.Symbol[2,2][1][3]	Read+Write	Always		
10	tag10	001A	Application.POU.Struct.Symbol	Read+Write	Always		
11	tag11	127.0.0.1:001A	Application.POU.Struct1.Struct2.Struct3.Symbol	Read+Write	Always		
*				Read+Write	Always		
*				Read+Write	Always		
*				Read+Write	Always		
*				Read+Write	Always		
*				Read+Write	Always		

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 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 CodeSys 3.x 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 Codesys runtime V3.x, with the Device Name MY_PLC_NAME, using the ARTI3 protocol.

Connecting to CodeSys 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

Station	Description
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

Connecting to CodeSys 2.x Runtime using TCP/IP Connection:

`[optGatewayIPAddress:]TCP,<RuntimeIPAddress>[:optTCPPort][,optProtocol]`

- ***`[optGatewayIPAddress:]`*** If you are going to connect with the runtime through a CodeSys Gateway, type here its IP Address. This parameter is optional and if omitted, the driver will connect directly to the CodeSys Runtime using ARTI over TCP/IP.
- ***`<RuntimeIPAddress>`*** IP Address where the CodeSys Runtime is running. If this parameter is omitted, the driver will assume the *localhost* IP address: 127.0.0.1
- ***`[:optTCPPort]`*** TCP/IP Port number. This is an optional parameter and, if omitted, the driver will use the port number **1200**
- ***`[,optProtocol]`*** Communication protocol to be used for connection with the Runtime. The valid options are **L4 (for *Level 4*)** or **L2 (for *Level 2*)**. This is an optional parameter and, if omitted, the driver will use **L4.**

Note:
Please notice some of the parameters are separated by *commas* and others by *colon*.

Examples:

Station	Description
192.168.1.10:TCP,192.168.1.50	Connects to the Runtime in the IP address 198.168.1.50 on port 1200 via gateway with IP address 192.168.1.10 and using protocol L4
192.168.1.10:TCP,192.168.1.50:1201,L2	Connects to Runtime in the IP address 198.168.1.50 on port 1201 via gateway with IP address 192.168.1.10 and using protocol L2
TCP, ,L2	Connects to Runtime using ARTI on the *localhost (IP 127.0.0.1)* on port 1200 and using protocol L2
TCP,192.168.0.50	Connects to Runtime using ARTI with IP address 192.168.0.50 on port 1200 and using protocol L4

Attention:
The next connection types were implemented but not fully tested. Before deploying applications using these connection types it is strongly recommended that you contact your technical support.

Connecting to CodeSys 2.x Runtime using Serial Connection:

`[optGatewayIPAddress:]SERIAL[,optCOMPort][,<optBaudRate]`

- ***`[optGatewayIPAddress:]`*** Configure here the IP Address of the gateway if you are using a gateway to connect to the Runtime. If this parameter is not specified the driver will connect directly to the Runtime using ARTI over serial.
- ***`[,optCOMPort]`*** Serial Port Number. If this parameter is not specified, the driver will use the serial port number specified in the driver settings.
- ***`[,<optBaudRate]`*** Communication Baud Rate. If this parameter is not specified, the driver will use the value specified in the driver settings.

Examples:

Station	Description
192.168.1.10:SERIAL,1	Connects to Runtime using COM1 via gateway with IP address 192.168.1.10 and the Baud Rate specified in the driver settings
SERIAL	Connects to Runtime using ARTI with COM Port and Baud Rate specified in the driver settings

Connecting to CodeSys 2.x or 3.x Runtime using settings from a PLCHandler.ini file (see PLC Handler documentation for details):

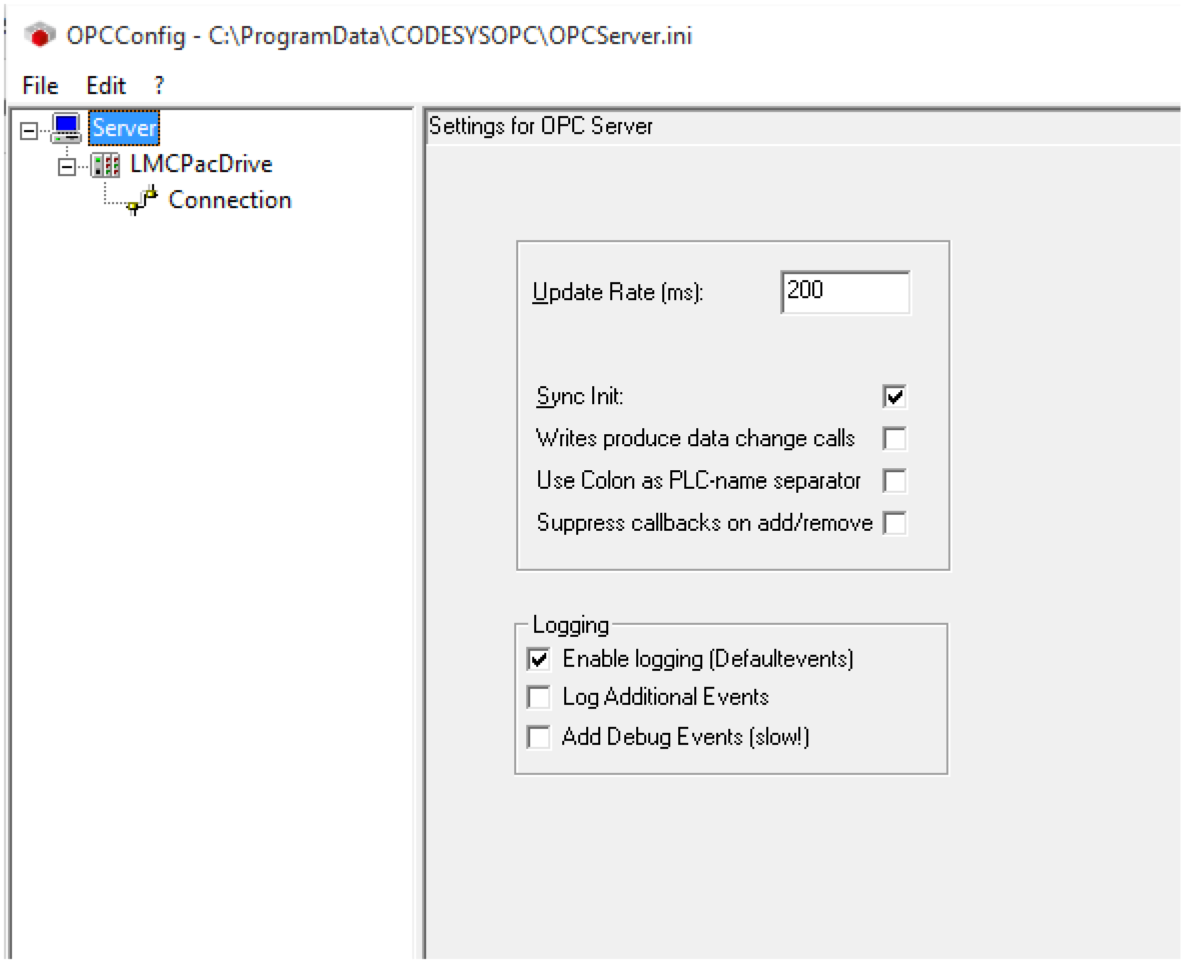
`CFILE,<Configuration File>,<Configuration ID>`

- **`<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: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.
For example:



Settings for OPC Server

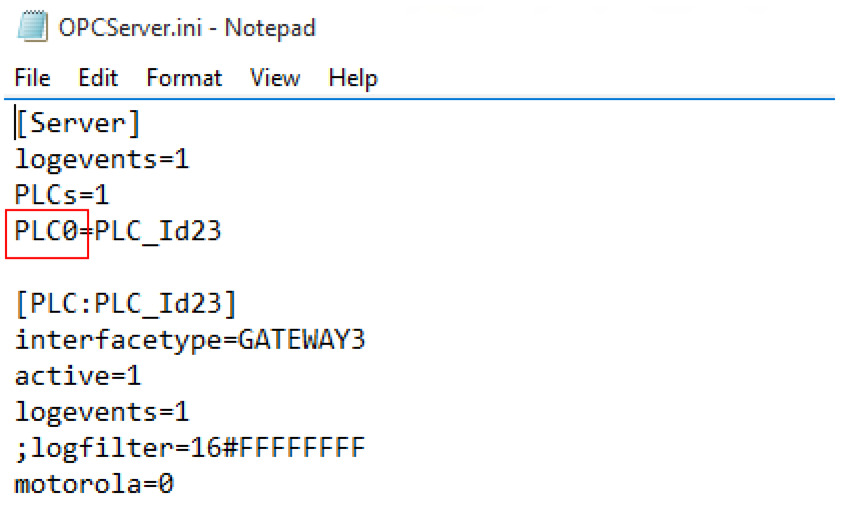
Update Rate (ms): 200

Sync Init: [✓]
Writes produce data change calls [ ]
Use Colon as PLC-name separator [ ]
Suppress callbacks on add/remove [ ]

Logging
[✓] Enable logging (Defaultevents)
[ ] Log Additional Events
[ ] Add Debug Events (slow!)

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:



```
[Server]
logevents=1
PLCs=1
PLC0=PLC_Id23

[PLC:PLC_Id23]
interfacetype=GATEWAY3
active=1
logevents=1
;logfilter=16#FFFFFFFF
motorola=0
```

✍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`

Connecting to CodeSys 2.x or 3.x Runtime using simulation mode:

`SIM, <SymbolFile>:`

- `<SymbolFile>`: CodeSys symbol file exported using the steps on section "Configuring CodeSys projects to communicate with External Applications"

- **I/O Address:** This field is configured with name of the variable used into CodeSys Runtime

- **For version 2.x:**

    - For **Global Variables**, the syntax is **`.<Variable Name>`** **(**Please notice the *dot* before the variable name**)**

      .intPosition

      .bHMIStart

      .Timer1.StartTime

    - For **Local Variables,** the syntax is **`<POU Name>.<Variable Name>.`** For example:

      PLC_PRG.intPosition1

      PLC_PRG.Timer2[1,3,0].StartTime

- **For version 3.x**, the syntax is `<Application Name>.<Object Name>.<Variable Name>`

  For example:

  Application.GVL.bHMIStart

  Application.PLC_PRG.bCycleCompleted

  Application.PLC_PRG.Nested4.struct2.struct2.struct2.struct1.wWord

  Application.GVL.DINT_Array_3Dim_2[1,-5,0]

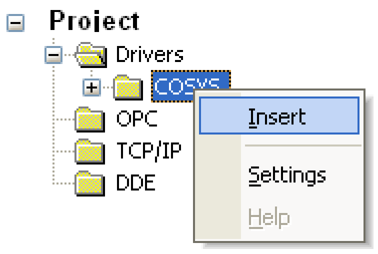
  Application.GVL.struct[9][5].Bool

  Application.GVL.Dint_Array_negative_index[-5]

Configuring the *STANDARD DRIVER SHEET*

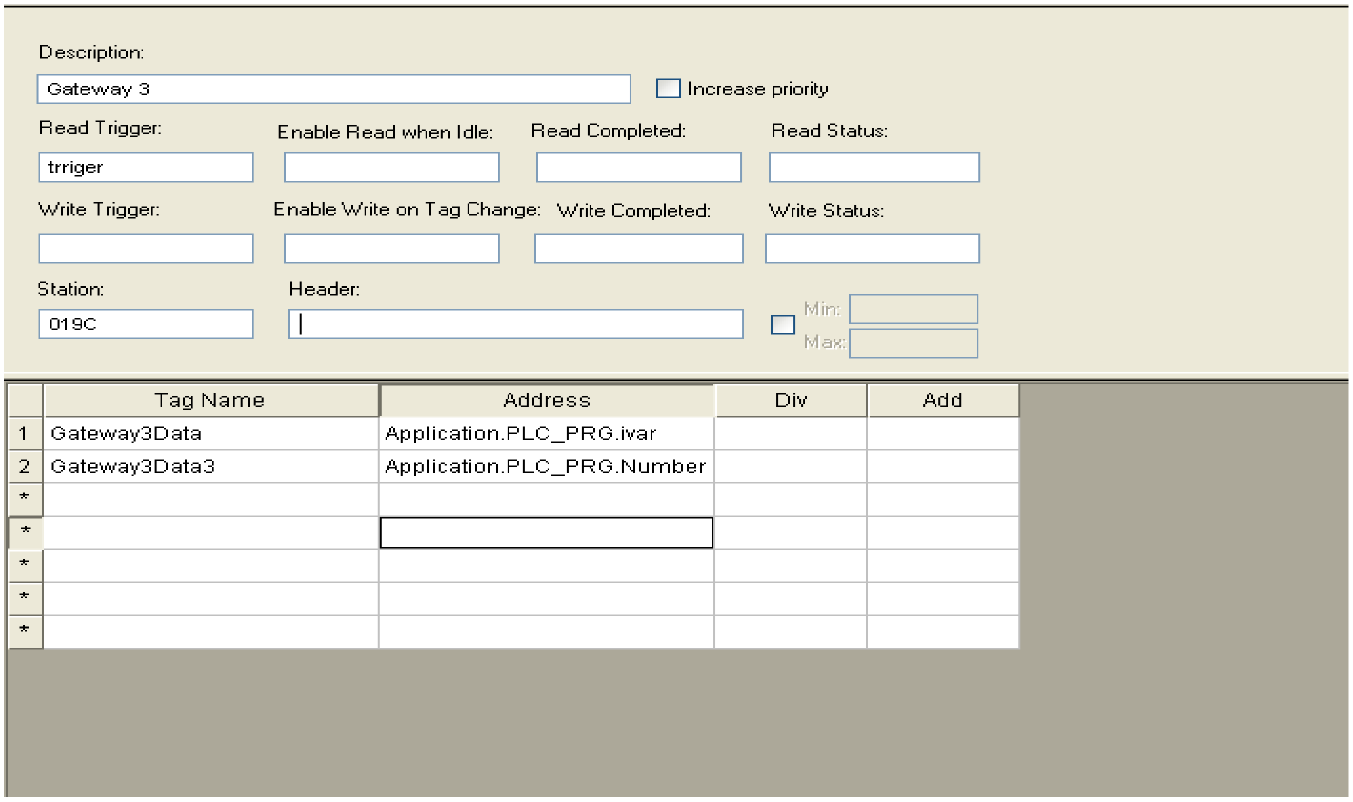
To insert a new Standard Driver Worksheet:

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



Inserting a New Worksheet

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



Description: Gateway 3
Increase priority: ☐

Read Trigger:	Enable Read when Idle:	Read Completed:	Read Status:
trriger			

Write Trigger:	Enable Write on Tag Change:	Write Completed:	Write Status:

Station: 019C
Header:
☐ Min: Max:

	Tag Name	Address	Div	Add
1	Gateway3Data	Application.PLC_PRG.ivar		
2	Gateway3Data3	Application.PLC_PRG.Number		
*				
*				
*				
*				
*				

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 CodeSys 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

Connecting to CodeSys 2.x Runtime using TCP/IP Connection:

`[optGatewayIPAddress:]TCP,<RuntimeIPAddress>[:optTCPPort][,optProtocol]`

- ***`[optGatewayIPAddress:]`*** If you are going to connect with the runtime through a CodeSys Gateway, type here its IP Address. This parameter is optional and if omitted, the driver will connect directly to the CodeSys Runtime using ARTI over TCP/IP.
- ***`<RuntimeIPAddress>`*** IP Address where the CodeSys Runtime is running. If this parameter is omitted, the driver will assume the *localhost* IP address: 127.0.0.1
- ***`[:optTCPPort]`*** TCP/IP Port number. This is an optional parameter and, if omitted, the driver will use the port number **1200**
- ***`[,optProtocol]`*** Communication protocol to be used for connection with the Runtime. The valid options are **L4 (for *Level 4*)** or **L2 (for *Level 2*)**. This is an optional parameter and, if omitted, the driver will use **L4.**

> **Note:**
> Please notice some of the parameters are separated by *commas* and others by *colon*.

Examples:

Station	Description
192.168.1.10:TCP,192.168.1.50	Connects to the Runtime in the IP address 198.168.1.50 on port 1200 via gateway with IP address 192.168.1.10 and using protocol L4
192.168.1.10:TCP,192.168.1.50:1201,L2	Connects to Runtime in the IP address 198.168.1.50 on port 1201 via gateway with IP address 192.168.1.10 and using protocol L2
TCP, ,L2	Connects to Runtime using ARTI on the *localhost (IP 127.0.0.1)* on port 1200 and using protocol L2
TCP,192.168.0.50	Connects to Runtime using ARTI with IP address 192.168.0.50 on port 1200 and using protocol L4

➲ **Attention:**
The next connection types were implemented but not fully tested. Before deploying applications using these connection types it is strongly recommended that you contact your technical support.

<u>Connecting to CodeSys 2.x Runtime using Serial Connection:</u>

[optGatewayIPAddress:]SERIAL[,optCOMPort][,<optBaudRate]

- ***[optGatewayIPAddress:]*** Configure here the IP Address of the gateway if you are using a gateway to connect to the Runtime. If this parameter is not specified the driver will connect directly to the Runtime using ARTI over serial.
- ***[,optCOMPort]*** Serial Port Number. If this parameter is not specified, the driver will use the serial port number specified in the driver settings.
- ***[,<optBaudRate]*** Communication Baud Rate. If this parameter is not specified, the driver will use the value specified in the driver settings.

Examples:

Station	Description
192.168.1.10:SERIAL,1	Connects to Runtime using COM1 via gateway with IP address 192.168.1.10 and the Baud Rate specified in the driver settings
SERIAL	Connects to Runtime using ARTI with COM Port and Baud Rate specified in the driver settings

Connecting to CodeSys 2.x or 3.x Runtime using settings from a PLCHandler.ini file (see PLC Handler documentation for details):

`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: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.

Connecting to CodeSys 2.x or 3.x Runtime using simulation mode:

`SIM, <SymbolFile>:`

`<SymbolFile>`: CodeSys symbol file exported using the steps on section "Configuring CodeSys projects to communicate with External Applications"

- **Header:** This field is not used in this particular driver
- **Address**: This field is configured with name of the variable used into CodeSys Runtime

- **For version 2.x**:

  - For **Global Variables**, the syntax is `.<Variable Name>` (Please notice the *dot* before the variable name)

    .intPosition

    .bHMIStart

    .Timer1.StartTime

  - For **Local Variables**, the syntax is `<POU Name>.<Variable Name>`. For example:

    PLC_PRG.intPosition1

    PLC_PRG.Timer2[1,3,0].StartTime

- **For version 3.x,** the syntax is **`<Application Name>.<Object Name>.<Variable Name>`**

For example**:**

```
Application.GVL.bHMIStart

Application.PLC_PRG.bCycleCompleted

Application.PLC_PRG.Nested4.struct2.struct2.struct2.struct1.wWord

Application.GVL.DINT_Array_3Dim_2[1,-5,0]

Application.GVL.struct[9][5].Bool

Application.GVL.Dint_Array_negative_index[-5]
```

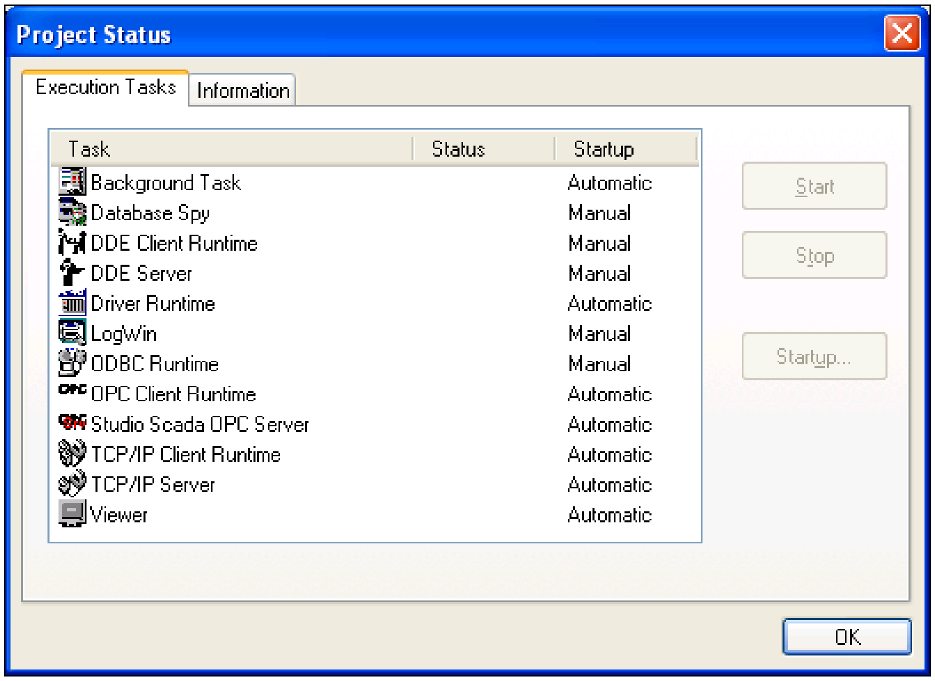
Executing the Driver

After adding the COSYS 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:



Task	Status	Startup
Background Task		Automatic
Database Spy		Manual
DDE Client Runtime		Manual
DDE Server		Manual
Driver Runtime		Automatic
LogWin		Manual
ODBC Runtime		Manual
OPC Client Runtime		Automatic
Studio Scada OPC Server		Automatic
TCP/IP Client Runtime		Automatic
TCP/IP Server		Automatic
Viewer		Automatic

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 COSYS 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. - Added support for Array or Arrays
N	2.9	Oct/28/2016	- Changed Boolean variables to accept any non-zero value as TRUE.
O	3.0	Apr/14/2017	-Fixed issue of Boolean variables not reading values correctly.
P	3.1	July/24/2017	-Improved symbol loading frequency.
Q	3.2	Mar/16/2018	- Fixed issue when using write trigger on SDS
R	3.3	Dec/6/2018	- Fixed potential buffer overruns in the driver. - Fixed memory leaks in the driver - Changed the max block size of the driver - Fixed error message text in the driver when it fails to communicate. - Fixed the logging of invalid symbols in the communication messages - Fixed issues when the driver is not able to reconnect with the PLC upon disconnection.