SIETH Communication Driver

Driver for Ethernet Communication
Between Studio and Siemens Devices

Contents

Introduction

The SIETH driver enables communication between Studio system and the Siemens devices using Ethernet, according to the specifications discussed in this publication.

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

- **Introduction**: Provides an overview of the SIETH driver documentation.
- **General Information**: Provides information needed to identify all the required components (hardware and software) used to implement communication between Studio and the SIETH driver.
- **Installing the Driver**: Explains how to install the SIETH driver.
- **Configuring the Driver**: Explains how to configure the communication 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.
- **Sample Application**: Explains how to use a sample application to test the 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 product’s *Technical Reference Manual*.
- This document also assumes that you are familiar with the Windows XP/7/8 environment. If you are unfamiliar with Windows XP/7/8, 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 SIETH driver and Siemens devices.

The information is organized into the following sections:

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

Device Characteristics

This driver has been tested successfully with the following devices:

- **Manufacturer**: Siemens
- **Compatible Equipment**: Any Siemens S7-200, Siemens S7-300/400 PLC communicating via an Industrial Ethernet interface, S7-1200, S7-1500, and LOGO! in S7-200 mode
- **Siemens PLC Programmer Software**: Step-7, Step7-MicroWin, TIA Portal

Link Characteristics

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

- **Device Communication Port**: Ethernet Port
- **Physical Protocol**: TCP/IP
- **Logic Protocol**: S7 Messaging on Industrial Ethernet
- **Adapters/Converters**: Standard Ethernet card

Driver Characteristics

The SIETH driver is composed of the following files:

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

> **Notes:**
> • All of the preceding files are installed in the **`/DRV`** subdirectory of the Studio installation directory.
> The SIETH driver requires the **`AGLINK.DLL`** into the **`/BIN`**.

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

- Windows XP/7/8 and Servers 2003/2008/2012
- Windows CE

The SIETH driver supports the following registers:

Register Type	Write	Read	Bit	Byte	Word	DWord	Float	String
M (*Flags*)	•	•	•	•	•	•	•	•
SM (Special *Flags*)	•	•	•	•	•	•	•	•
T (*Timers*)	•	•	–	–	•	–	–	•
Z or C (*Counters*)	•	•	–	–	•	–	–	•
E or I (*Inputs*)	–	•	•	•	•	•	•	•
A or Q (*Outputs*)	•	•	•	•	•	•	•	•
DB or V (*Data Blocks*)	•	•	•	•	•	•	•	•

> **Note:**
> The V Register from S7-200 devices can be accessed with the DB1 Register

Conformance Testing

The following hardware/software was used for conformance testing:

- **Driver Configuration**:
  - **PLC Program**: Step 7
  - **Protocol**: S7 Messaging on Industrial Ethernet

Driver Version	Studio Version	Operating System	Equipment
10.13	8.0 + SP1 +P1	Windows XP/7/8 Windows CE 5.0 (x86) and 7.0 (ARMV4I)	S7-315 2DP with CP343 S7-215 with CP-243 S7-1500 (CPU 1511-1 PN). S7-400 (CPU 412-2)

Special Device Configurations

S7-1200 / S7-1500

For the PLCs of these families, you have to enable the communication to remote devices on the SIEMENS TIA-Portal software.

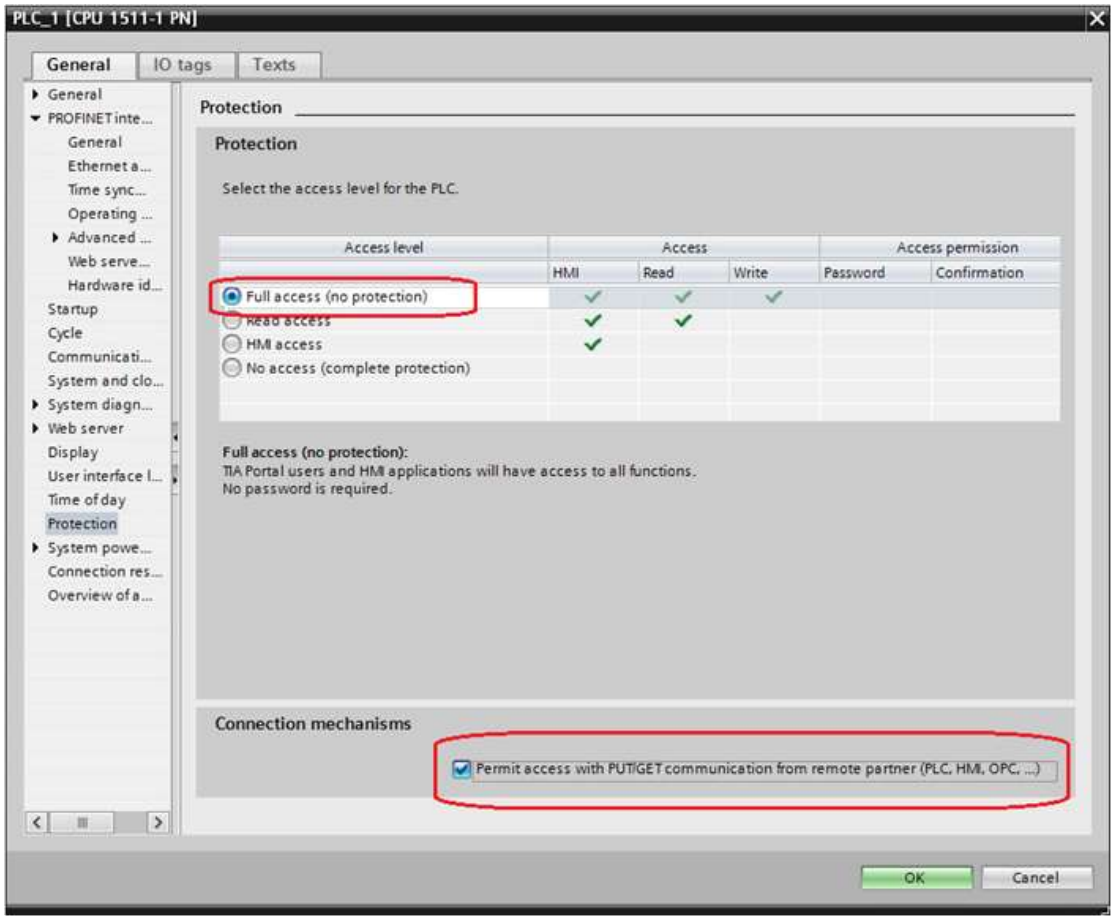
Please follow the steps below to permit access communication from remote partners:

1- After adding the proper CPU to the PLC program, go to the device properties (e.g. right click on the CPU name and pick Properties);

2- Under General tab select Protection at the menu list on the right side;

3- Under Protection make sure you have the access level “Full access (no protection)” selected;

4- At the bottom of this screen, under “Connection mechanisms” enable the option (CHECK the box) “Permit access with PUT/GET communication from remote partner (PLC, HMI, OPC, …)”. See screenshot below;



5- Confirm with OK;

6- Recompile and send the program to the PLC.

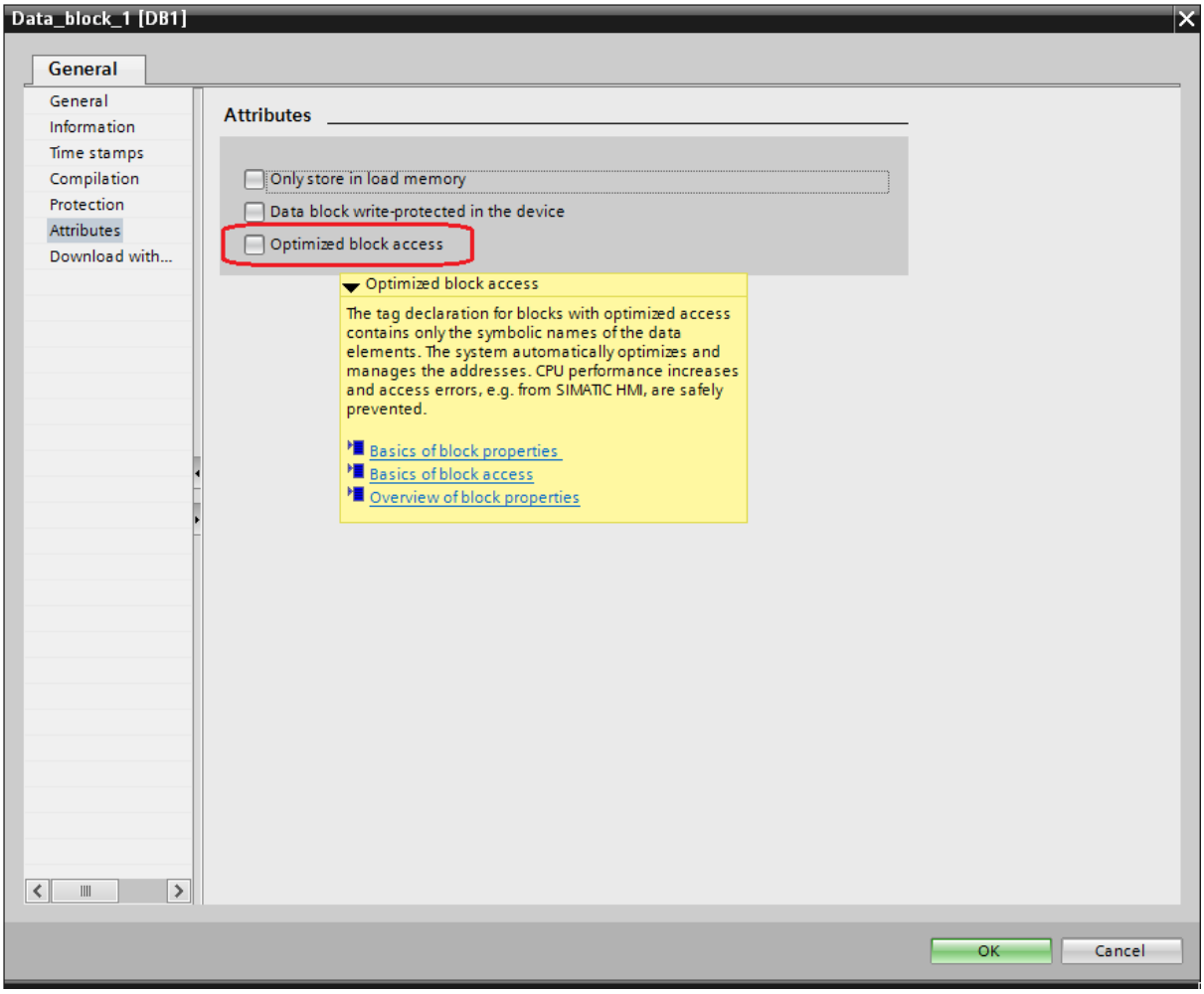
Data Blocks configuration

The Data Blocks (DB's) on SIMATIC TIA Portal V12 are created by default with the option "Optimized block access" enabled. This option provides data only over the variable names (Tags names). Since the InduSoft drivers access the memory offsets, this options must be manually disabled.

Execute the following steps to do so:

1- Go to the Data Block properties dialog (e.g. right click on the desired DB and pick Properties);

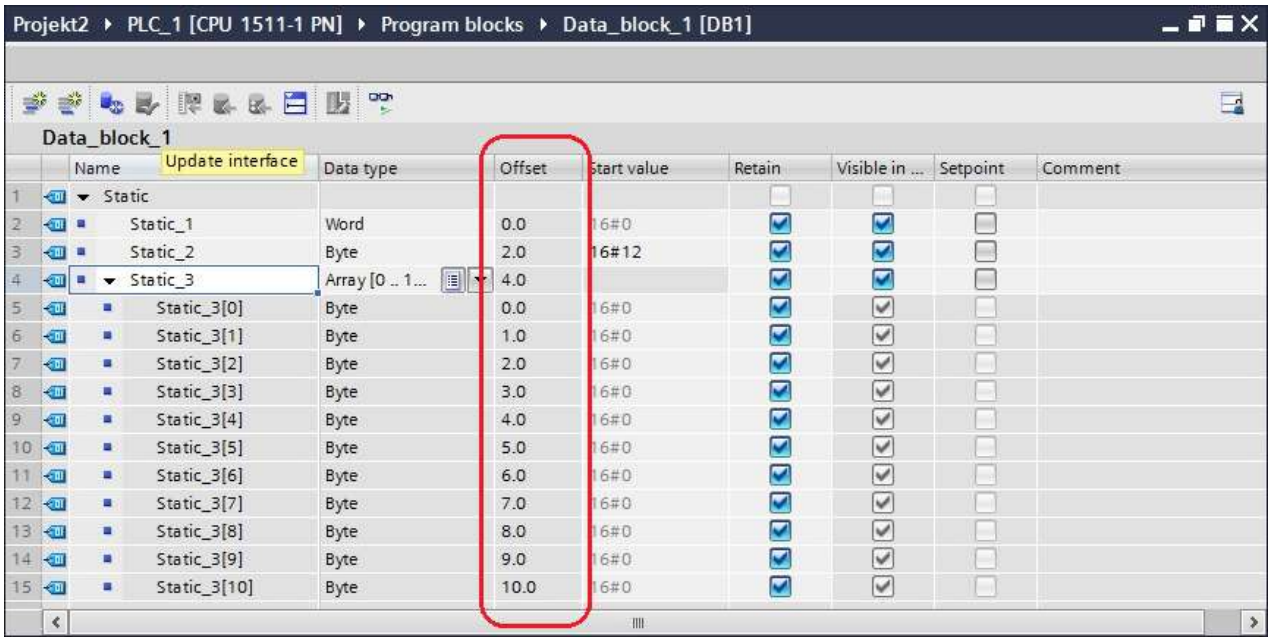
2- On menu Attributes disable the option (UNCHECK the box) "Optimized block access". See screenshot below.



Configuring Optimal Block Access on TIA Portal

3- Confirm with OK.

Opening the Data Block view, the additional column “Offset” must be shown on the table which brings the memory offsets to be used on the InduSoft drivers such as the SIETH.



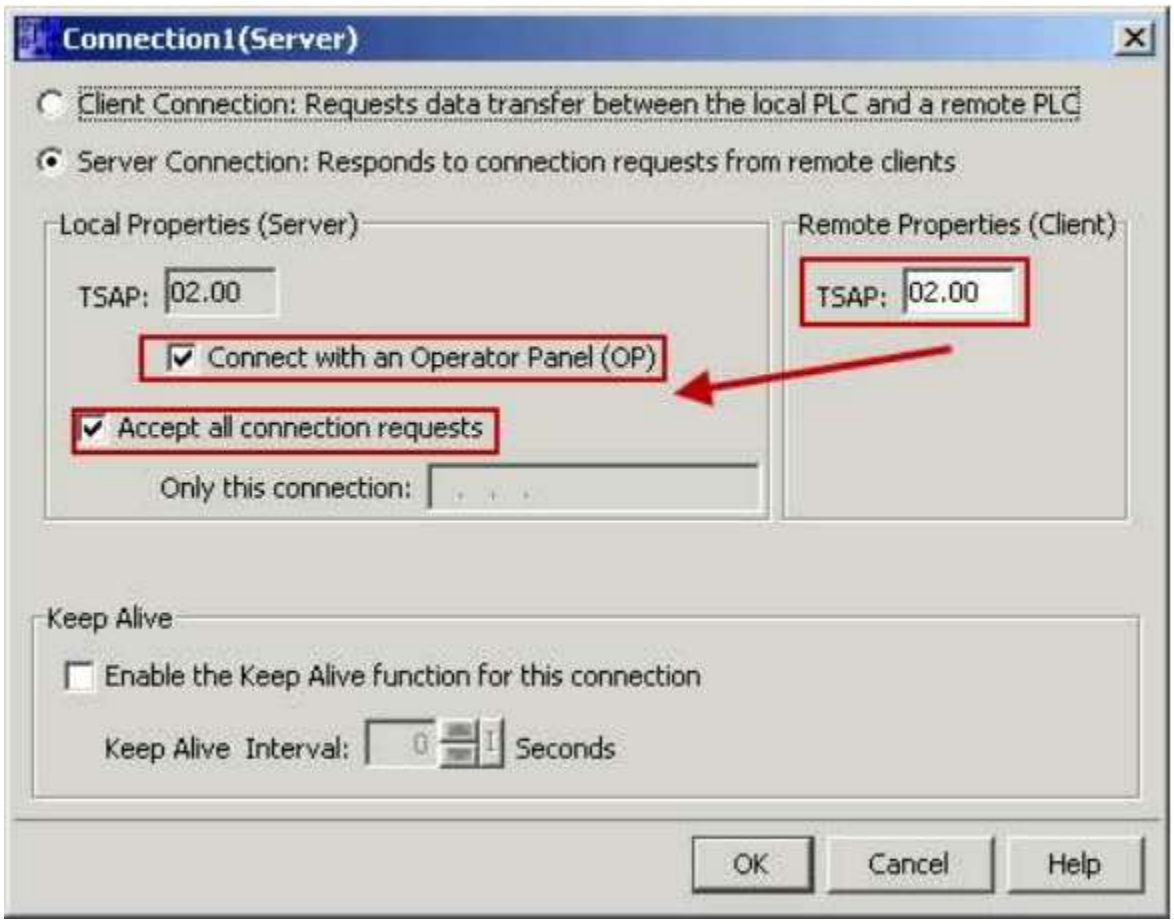
Configuring Offset to Data Blocks on TIA Portal

> **Note:**
> For arrays, the main offset must be added with the element offset – e.g. DB1.DBB6 (Byte 6) refers to the array element Static_3[2], DB1.DBB14 (byte 14) refers to the array element Static_3[10] on the Data Block shown above.

LOGO! (*Acting as S7-200)*

When you have a LOGO! CPU with Ethernet, such as the Logo!12/24 RCE and Logo!230 RCE, you can configure its communication to act as a S7-200 PLC.

The screenshot below shows where to configure the TSAP on the LOGO! Software. Also, be sure to check the indicated boxes.



Configuring TSAP on Logo!

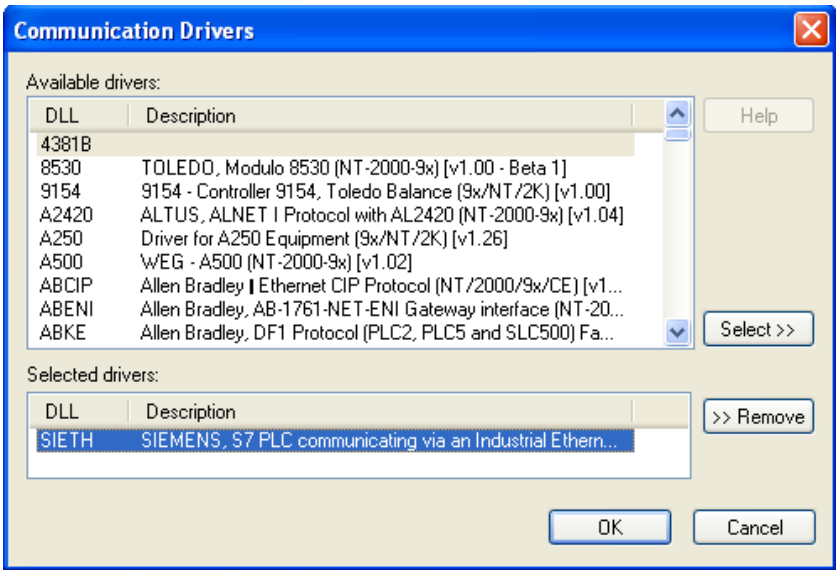
> **Note:**
> Please refer to the Communication Settings and the Station fields on the Driver Sheets on how to properly configure the TSAP parameters.

Installing the Driver

When you install Studio version 5.1 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 **SIETH** driver from the *Available Drivers* list, and then click the **Select** button:



Communication Drivers Dialog

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

> ➲ **Attention:**
> For safety reasons, you must be careful when installing the physical hardware. Consult the hardware manufacturer's documentation for specific installation instructions.

Configuring the Driver

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

- Specifying communication parameters
- Defining communication tags and controls in the Communication tables or *Driver* worksheet

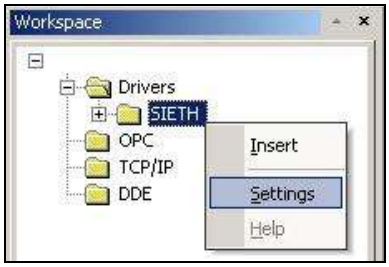
  Worksheets are divided into two sections, a *Header* and a *Body*. The fields contained in these two sections are standard for all communications drivers — except the **Station**, **Header** and **Address** fields, which are driver-specific. This document explains how to configure the **Station**, **Header** and **Address** fields only.

> **Notes:**
> For a detailed description of the Studio *Standard* and *MAIN* Driver Worksheets, and information about configuring the standard fields, review the product's *Technical Reference Manual*.

Setting the Communication Parameters

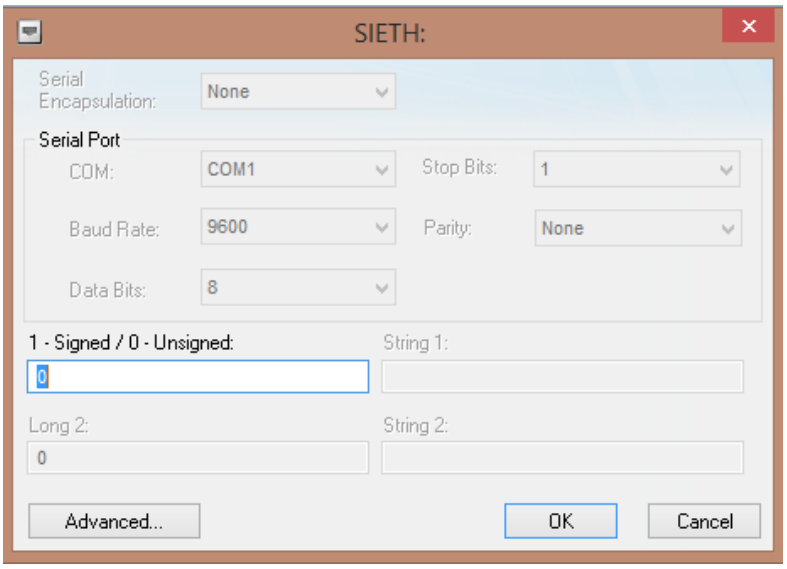
Use the following steps to configure the communication parameters, which are valid for all driver worksheets configured in the system:

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



Select Settings from the Pop-Up Menu

The *SIETH: Communication Parameters* dialog displays:

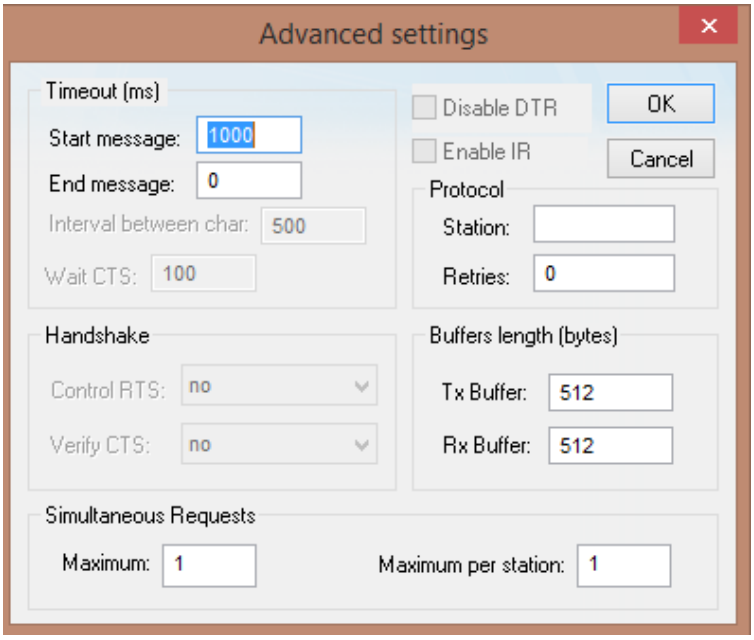


SIETH: Communication Parameters Dialog

4. This driver does not have custom parameters. Specify the parameters as noted in the following table.

Parameters	Default Values	Valid Values	Description
1-Signed **0-Unsigned Value**	0	0 or 1	This value will be default if you do not specify the Unsigned/Signed parameter for each format address.

5. For S7-200 communication (as well as LOGO!), configure the Remote TSAP in the Station field in the advanced settings. If Station field is left empty, the default Remote TSAP of 0x4D57 is used. In the Station of Protocol, we can configure the **Remote TSAP** value.



> **Notes:**
>
> The Remote and Local TSAP are configured in S7-200 using Micro/WIN's Ethernet Wizard. In that wizard, TSAP's are configured in the format `XX.YY`. In SIETH driver, write these values as `XXYY`. The Remote and Local TSAPs are shown in this documentation using the same perspective as Micro/WIN. That is, configure the TSAP's with the same values in both driver and Micro/WIN. For Configuration of Local TSAP refer to configuration of the worksheets.

> **Note:**
>
> Additional communication settings can be accessed in the *Advanced Settings* dialog. To open this dialog, simply click the **Advanced** button in the *Communication Settings* dialog.
>
> Simultaneous connection are supported on this driver for configuring this settings please check the "Simutaneos Requests" field on the advanced settings dialog. The maximum supported simultaneous connections are 32 and 16 per station.

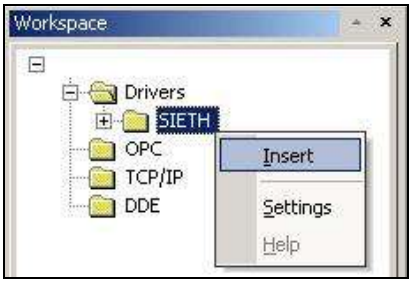
6. Click **OK** to close the dialog.

Configuring the Driver Worksheet

This section explains how to configure a *Standard Driver Worksheet* (or Communication table) to associate application tags with the PLC addresses. You can configure multiple *Driver* worksheets — each of which is divided into a *Header* section and a *Body* section.

Use the following steps to create a new *Standard Driver* worksheet:

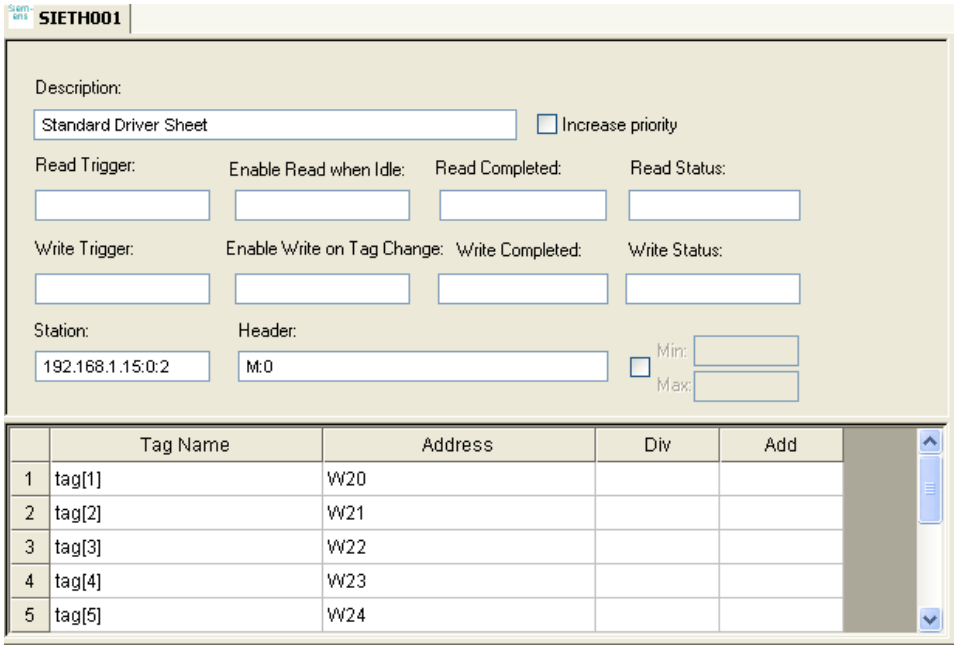
1. From the Studio development environment, select the **Comm** tab, located below the *Workspace* pane.
2. In the *Workspace* pane, expand the *Drivers* folder, and right-click the *SIETH* subfolder.
3. When the pop-up menu displays, select the **Insert** option:



Inserting a New Worksheet

> **Note:**
>
> To optimize communication and ensure better system performance, you must tie the tags in different driver worksheets to the events that trigger communication between each tag group and the period in which each tag group must be read or written. Also, we recommend configuring the communication addresses in sequential blocks to improve performance.

The *SIETH.drv* dialog displays (similar to the following figure):



SIETH Driver Worksheet

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

4. Use the following information to complete the **Station**, **Header**, and **Address** fields on this worksheet:

Station field: Use this field to specify the IP Address, Rack, Slot and Local TSAP.

Syntax:

For **S7-300, S7-400, S7-1200** and **S7-1500**:

<IP Address>:<Rack>:<Slot>[:<Port>]

The user may also prefix the syntax above with ***S7-300:*** or ***S7-400:*** if desired, however this is optional.

Examples

192.168.0.10:0:2

S7-300:192.168.0.10:1:1

192.168.0.10:0:1

> **Note:**
>
> The most common configuration for **S7-300** and **S7-400** is to have the CPU on the Slot number 2 of the Rack 0. In that case, the **Station** would look like: ***<IP Address>:0:2***
>
> As for the S7-1200 and S7-1500, usually the CPU is seen on the Slot number 1 of the Rack 0, which would then cause the **Station** field to like: ***<IP Address>:0:1***

For **S7-200** and connected to the network via a CP243 module, and **LOGO!** use the syntax:

S7-200:<IP Address>[:<Port>]

The port parameter is optional and defaults to 102.

Or:

S7-200:<IP Address>:<Port>[:<Local TSAP>]

The Local TSAP parameter is optional and default to 4D57. To include the Local TSAP, you must include the Port number.

Examples:

S7-200:192.168.0.21

S7-200:192.168.0.21:102

S7-200:192.168.0.21:102:1100

S7-200:192.168.0.21:102:1200

> **Notes**:
> The Remote and Local TSAP are configured in S7-200 using Micro/WIN's Ethernet Wizard. In that wizard, TSAP's are configured in the format XX.YY. In SIETH driver, write these values as XXYY. The Remote and Local TSAPs are shown in this documentation using the same perspective as Micro/WIN. That is, configure the TSAP's with the same values in both driver and Micro/WIN. For Configuration of Remote TSAP refer to configuration of the driver settings.

Header field: Use the information in the following table to define the type of variables that will be read from or written to the device and a reference to the initial address.

These variables must comply with the following syntax:

- For Flags, Special Flags, Timers, Counters, Inputs and Outputs:

<*Type*>:<*optAddressReference*> (for example: **M:1**)

- For Data Blocks:

<*Type*><*TypeGroup*>:<*optAddressReference*> (for example: **DB2:1**) or

<*Type*><*TypeGroup*>.DB :<*optAddressReference*> (for example: **DB2.DB or DB2.DB:1**)

Where:

- ***<Type>*** is the register type. (**M**=Flags, **SM**=Special Flags, **T**=Timers, **Z** or **C**=Counters, **E** or **I**=Inputs, **A** or **Q**=Outputs, and **DB**=Data Blocks)
- ***<TypeGroup>*** is the group number of the configured register type (for Data Block types only).
- ***<optAddressReference>*** (optional) is the initial address reference of the configured group. If it is not configured, it assumes the value 0 (zero). This number *always* refers to the *Byte address number* (see the following table).

> **Note:**
> SIETH driver supports until 8 (eight) different stations, if you need more please contact your Studio technical support representative

SIETH driver supports using hostnames in Station field.

The following table lists all of the valid initial address (reference) values for the SIETH driver:

Header Address	SIETH Address		
Byte Address Number	Byte Address Number	Word Address Number	
Byte 0	Byte 0	W0	
Byte 1	Byte 1		W1
Byte 2	Byte 2	W2	
Byte 3	Byte 3		W3
Byte 4	Byte 4	W4	
Byte 5	Byte 5		W5
Byte 6	Byte 6	W6	
Byte 7	Byte 7		W7
Byte 8	Byte 8	W8	
Byte 9	Byte 9		W9
Byte 10	Byte 10	W10	
Byte 11	Byte 11		

The next table lists all of the data types and address ranges that are valid for the SIETH driver:

Header Field Information			
Data Types	Sample Syntax	Valid Range of Initial Addresses	Comments
Flags	**M:1** or **M**	Varies according to the equipment	Logical Flags
Special Flags	**SM:1** or **SM**	Varies according to the equipment	Special Flags
Timers	**T:2** or **T**	Varies according to the equipment	Timer Values
Counters	**Z:10**, **C:10 or Z,C**	Varies according to the equipment	Counter Values
Inputs	**E:5**, **I:5** or **E,I**	Varies according to the equipment	Physical Input Values
Outputs	**A:8**, **Q:8** or **A,Q**	Varies according to the equipment	Physical Output Values
Data Blocks	**DB2:1,** or **DB2.DB** or **DB2.DB:1**	Varies according to the equipment	Data block values, where: ▪ Number following DB(2) specifies the data block number ▪ Number following the colon specifies the word offset in the data block

Address field: Use the information provided in the following table to associate each tag to its respective device address.

Type the tag from your application database into the **Tag Name** column. This tag will receive values from or send values to an address on the device. The address must comply with the following syntax:

`[Signed/Unsigned]<Format><AddressOffset>.<Bit>` (for example: **`X10.2`**)

`<Format><AddressOffset>.<Len>` (for example: **`ST2.10, SST2.10`**) – String format only

Where:

- **`[Signed/Unsigned]`** (*optional parameter used for integer values only*): If you do not specify this parameter, Studio inserts an integer value based on the parameters you set in the *Communication Parameters* dialog. Valid values are **S** (signed) and **U** (unsigned). Dword does not use **U** (unsigned).
- ***<Format>*** defines how Studio treats the value read or written from/to the device. (**X=**Bit, **B**=Byte or Bit, **W**=Word, **D** or **DW**=Dword, **F**=Float, **ST**=String, **SST**= S7 String format).
  - For Data-Blocks:

  T=Timer Format (S5T), **TS**=Timer in String format (hh:mm:ss:ms), **C**=Counter Format
- ***<AddressOffset>*** is a parameter added to the **AddressReference** parameter (configured in the **Header** field) to compose the group address configured in the **Header** field.
- ***<Bit>*** is the bit number (from 0 – 7) from the **Byte** address. This parameter is optional, and it is supported only when the format is Byte or Bit(**B** or **X**).
- ***<Len>*** is the length to read or to write. It is in bytes. String format only.

> **Note:**
> T, TS and C data formats are most used for Data-Blocks operand, however they can be used for all operands.

Sample Address Configuration		
Address on the Device	**Header Field**	**Address Field**
M (Word 5 = Byte 5 / Byte 6 / String 7, Length 10)	M	W5
	M:5	W0
	M:3	W2
M (Byte 5)	M	B5
	M:5	B0
	M:1	B4
M (Byte 6)	M	6
	M:6	0
	M:3	B3
M (String 7, Length 10)	M	ST7.10
SM (Word 5 = Byte 5 / Byte 6 / String 7, Length 10)	SM	W5
	SM:5	W0
	SM:3	W2
SM (Byte 5)	SM	B5
	SM:5	B0

Sample Address Configuration		
Address on the Device	**Header Field**	**Address Field**
	SM:1	B4
SM (Byte 6)	SM	B6
	SM:6	B0
	SM:3	B3
SM (String 7, Length 10)	SM	ST7.10
	SM:7	ST0.10
	SM:4	ST3.10
T (33)	T	33
	T:30	3
	T33	T0
T(35)	T	TS35 (string format)
	T:35	0
	T31	4
C (3)	C	3
	C:3	C0
	C:2	1
C (4)	C	4
	C:4	0
	C:2	2
DB5 (Word 2 = Byte 2 / Byte 3 / String 4 length 10)	DB5:0	W2
	DB5:2	W0
	DB5:1	W1
DB5 (Byte 2)	DB5.DB	B2
	DB5:2	B0
	DB5:1	B1
DB5 (Byte 3)	DB5.DB:0	B3
	DB5:3	B0
	DB5:2	B1
DB5 (String 4 length 10)	DB5:0	ST4.10
	DB5:4	ST0.10
	DB5:2	ST2.10
DB5 (S7-String Format, length 10)	DB5:0	SST4.10
	DB5:4	SST0.10
	DB5:2	SST2.10
DB5 (Word 7 = Byte 7 / Byte 8 / Double Word 3)	DB5:0	W7
	DB5:7	W0
	DB5:4	W3
DB5 (Byte 7)	DB5:0	B7
	DB5:7	B0
	DB5:4	B3
DB5 (Address 8)	DB5:0	C8 (counter format)
	DB5:8	T0 (timer format)
	DB5:4	TS4 (timer string format)

Sample Address Configuration		
Address on the Device	**Header Field**	**Address Field**
DB5 (Double Word 3)	DB5.DB	D3
	DB5:0	DW3
	DB5.DB:2	DW1
Input (Address 1, bit 4)	I:0 or E:0	X1.4
	I:1 or E:1	X0.4
	I:0 or E:0	B1.4
Input (Byte 1)	I:0 or E:0	B1
	I:1 or E:1	B0
Output(Address 1, bit 4)	Q:0 or A:0	X1.4
	Q:1 or A:1	X0.4
	Q:0 or A:0	B1.4
Output(Byte 1)	Q:0 or A:0	B1
	Q:1 or A:1	B0

➲ **Attention:**

You must not configure a range of addresses in the same worksheet that is greater than the maximum block size (data buffer length) supported by the protocol. The maximum data buffer length for this driver is 1024 bytes in each *Standard Driver* worksheet.

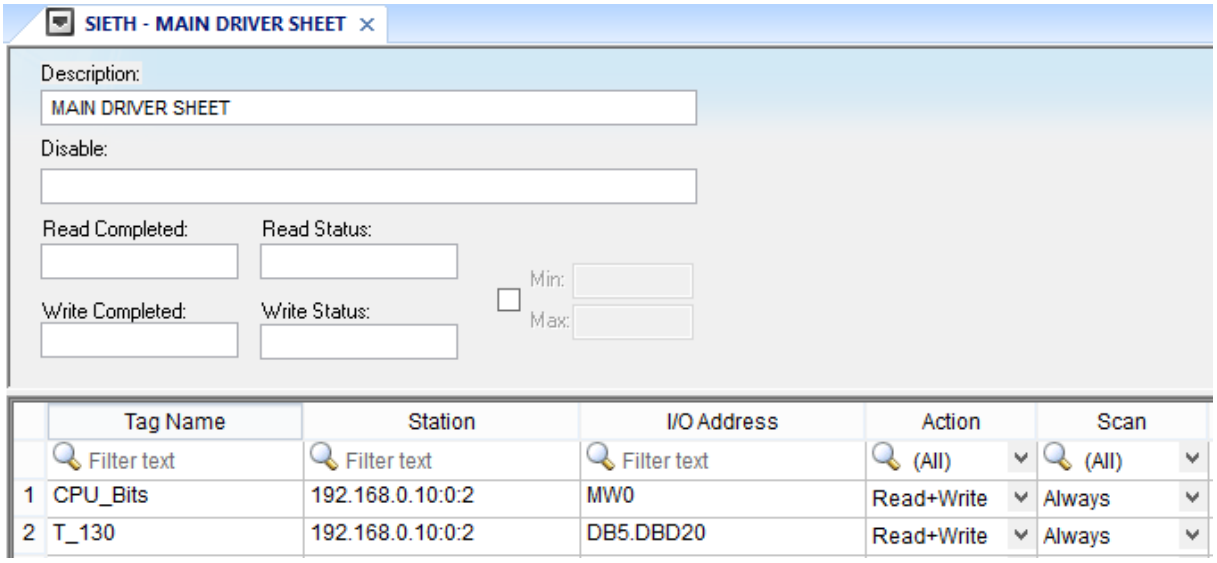
If you are using a **S7-200** the block size restrictions are different than for other families. For timers the limitation is of 12 operands, counters are limited on 21 operands and the other headers are limited on 64 bytes.

Main Driver Sheet (MDS)

When the driver is inserted into the application, the *MAIN DRIVER SHEET* is automatically added to the driver folder.

Main Driver Sheet

The MAIN DRIVER SHEET provides a simple way to associate Studio tags to addresses in the PLC. Most of the MAIN DRIVER SHEET entries are standard for any driver. Refer to the Studio *Technical Reference Manual* about the configuration of the standard fields. The fields that require specific syntax for this driver are described below:



Main Driver Sheet

Station field: Use this field to specify the IP Address, Rack, Slot and Local TSAP.

Syntax:

For **S7-300, S7-400, S7-1200** and **S7:1500**

<IP Address>:<Rack>:<Slot>[:<Port>]

The user may also prefix the syntax above with ***S7-300:*** or ***S7-400:*** if desired, however this is optional.

Examples

192.168.0.10:0:2

S7-300:192.168.0.10:1:1

192.168.0.10:0:1

> **Note:**
>
> The most common configuration for **S7-300** and **S7-400** is to have the CPU on the Slot number 2 of the Rack 0. In that case, the **Station** would look like: ***<IP Address>:0:2***
>
> As for the **S7-1200** and **S7-1500**, usually the CPU is seen on the Slot number 1 of the Rack 0, which would then cause the **Station** field to like: ***<IP Address>:0:1***

If the device is a **S7-200** connected to the network via a CP243 module, or a **LOGO!**, use the syntax:

S7-200:<IP Address>[:<Port>]

The port parameter is optional and defaults to 102. Or:

S7-200:<IP Address>:<Port>[:<Local TSAP>]

The Local TSAP parameter is optional and default to 4D57.

Examples:
192.168.0.10:0:2
S7-300:192.168.0.10:1:1
S7-200:192.168.0.21
S7-200:192.168.0.21:102
S7-200:192.168.0.21:102:1100
S7-200:192.168.0.21:102:1200
S7-200:192.168.0.21:102:1700

> **Notes**:
> The Remote and Local TSAP are configured in S7-200 using Micro/WIN's Ethernet Wizard. In that wizard, TSAP's are configured in the format `XX.YY`. In SIETH driver, write these values as `XXYY`. The Remote and Local TSAPs are shown in this documentation using the same perspective as Micro/WIN. That is, configure the TSAP's with the same values in both driver and Micro/WIN. For Configuration of Remote TSAP refer to configuration of the driver settings.

- **I/O Address:** Address of each register from the PLC. The syntax used in this field is described below:
  - For Flags, Special Flags, Timers, Counters, Inputs and Outputs:

    <*Type*>[Signed/Unsigned]<*Format*><*Address*>.<*Bit*> (for example: **MB10.2**)

    <*Type*><*Format*><*Address*>.<*Len*> (for example: **MST2.10**) – String format only

  - For Data Blocks:

    <*Type*><*TypeGroup*>[Signed/Unsigned]<*Format*><*Address*>.<*Bit*> (for example: **DB2W1**)

    <*Type*><*TypeGroup*>.DB[Signed/Unsigned]<*Format*><*Address*>.<*Bit*> (for example: **DB2.DBB1, DB2.DBUB5, DB2.DBD20**)

    <*Type*><*TypeGroup*><*Format*><*Address*>.<*Len*> (for example: **DB2:ST2.10**) – String format only

Where:

- **[*Signed/Unsigned*]** (*optional*): Parameter used for integer values only. If you do not specify this parameter, Studio uses the *Communication Parameters* settings to configure integers.
  Valid values for this parameter are **S** (*Signed*) or **U** (*Unsigned*). Dword does not use **U** (*Unsigned*).

- ***<Type>*** is the register type. (**M**=Flags, **SM**=Special Flags, **T**=Timers, **Z** or **C**=Counters, **E** or **I**=Inputs, **A** or **Q**=Outputs,
  and **DB**=Data Blocks)

- ***<TypeGroup>*** is the group number of the configured register type (for Data Block types only).

- ***<Address>*** is the device address. This number always refers to the **Byte** address number.

- ***<Format>*** defines how Studio treats the value read or written from/to the device (**X**=Bit**,** **B**=Byte or Bit, **W**=Word, **D** or **DW**=DWord, **F**=Float, **ST**=String). (String format does not support bits)
  - For Data-Blocks:
    T=Timer Format (S5T), **TS**=Timer in String format (hh:mm:ss:ms), **C**=Counter Format

- ***<Bit>*** is the bit number (from 0 – 7) from the **Byte** address. This parameter is optional, and it is supported only when the format is Byte or Bit (**B** or **X**).

- ***<Len>*** is the length to read or to write. It is in bytes. String format only.

> ✎ **Note:**
> SIETH driver supports until 8 (eight) different stations, if you need more please contact your Studio technical support representative

> ✎ **Note:**
> SIETH driver supports using hostnames in Station field.

> ✎ **Note:**
> T, TS and C data formats are most used for Data-Blocks operand, however they can be used for all operands.

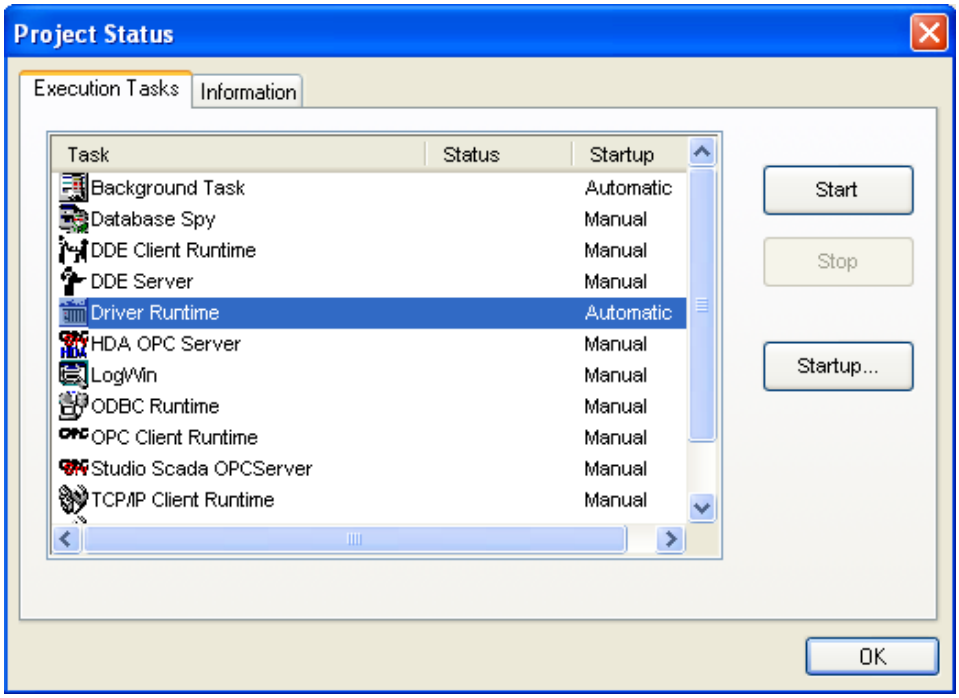
Executing the Driver

After adding the SIETH driver to a project, Studio sets the project to execute the driver automatically when you start the run-time environment.

To verify that the driver run-time 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 SIETH 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
-33	Invalid INI file
-34	Invalid Address
-36	Invalid Data Type
-38	Invalid Station
-39	Invalid Block Size
0	OK
1	Error to initialize driver
2	Maximum number of connections was exceeded.
53	Invalid Message
60	Handshake Error
61	Invalid Connection
62	Invalid Message
244	Function is not supported
245	Internal error, please check
246	Listed job number is invalid
247	At least one parameter for opening the device is invalid
248	No free space in the request queue
249	The necessary class cannot be initialized
250	The necessary memory cannot be allocated
251	Device is not open
252	Device was not found
253	Device is already in use, open, or update is not valid
254	Function is not valid
255	End of program request
256	A parameter was not in the defined range
508	Not all parameters can be changed because the adapter is already initialized
509	The program "**AGLink_Config.EXE**" cannot be started
510	The parameters were changed because of plausibility checks
511	Parameter length is not supported
512	Requested option is not available

Error Code	Description
752	Sending buffer too small for packet
753	Receiving buffer too small for packet
754	Timeout while waiting for DLE after sending the packet
755	Packet to be sent is not correct (length 0 or NULL pointer)
756	After STX arbitrary information was sent instead of DLE
757	After STX NAK was sent instead of DLE
758	Timeout after initialization conflict (both have high priority)
759	Timeout while waiting for DLE after sending STX
760	Initialization conflict
761	Wrong protocol status
762	Checksum error
763	Timeout while waiting for checksum
764	Information after DLE was not DLE or ETX
765	Timeout while waiting for packet information (ZVZ)
766	Timeout while waiting for beginning of packet (QVZ)
767	Wrong information received instead of STX
768	Timeout while waiting for STX at the beginning of the program
1019	Adapter is not initialized
1020	Unknown error message from adapter
1021	Wrong MPI baud rate
1022	The address code is higher than HAS
1023	Requested adapter address already exists
1024	Received packet has wrong content
1260	Type (of data) is not supported
1261	Access to object is not permitted
1262	Invalid address
1263	Context is not supported
1264	PLC sends no data
1265	Function protection level is not sufficient
1266	Context is not supported
1267	Information cannot be determined at the moment
1268	Unknown error message from PLC, please check
1269	Wrong size operands or selected range too large
1270	Wrong operating status of PLC
1271	Error while restarting the PLC
1272	Error while starting the PLC
1273	Wrong PLC operating status

Error Code	Description
1274	Internal error, please check
1275	No data available (for example, missing DB)
1276	Hardware error (for example, nonexistent peripheral equipment)
1277	Number of frame does not fit
1278	PLC was not found
1279	No additional connection possible
1280	No connection to the requested PLC
1523	DSR signal changed to 0 (modem disconnected)
1524	DCD signal changed to 0 (no carrier)
1525	No connection to remote terminal
1526	No modem found at the device
1527	Error during initialization of auto-answer
1528	Error during initialization of dial tone
1529	Error during initialization of selection procedure
1530	Error during initialization sequence 4
1531	Error during initialization sequence 3
1532	Error during initialization sequence 2
1533	Error during initialization sequence 1
1534	Error during basis initialization (AT&FE0V1)
1535	Modem cannot hang up
1536	General modem error
1786	CIF card not logged in on the logical ring (bus)
1787	A resource error exists
1788	Wrong firmware version of CIF card
1789	Wrong hardware version of CIF card
1790	Error in a device driver function
1791	Requested board not found
1792	Requested device driver not found
2038	Close received instead of ReadOK
2039	Timeout while reading IP
2040	Error while reading IP
2041	Close received instead of WriteOK
2042	Timeout while writing IP
2043	Error while writing IP
2044	Close received instead of ConnectOK (for example, wrong rack or slot number)
2045	Timeout while establishing IP connection
2046	Error while establishing IP connection

Error Code	Description
2047	Listed IP address invalid
2048	Socket cannot be opened

⇨ **Tip:**

You can verify communication status using the Studio development environment *Output* window (*LogWin* module). To establish an event log for **Field Read Commands**, **Field Write Commands**, and **Protocol Analyzer**, right-click in the *Output* window. When the pop-up menu displays, select the option to set the log events.

If you are unable to establish communication with the PLC, try to establish communication between the PLC Programming Tool and the PLC.

Try pinging the PLC, or telnet to the default port, which is usually 102.

Quite frequently, communication is not possible because you have a hardware or cable problem, or a PLC configuration error. After successfully establishing communication between the device's Programming Tool and the PLC, you can retest the supervisory driver.

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**.
- **Project Information**: To find this information, select **Help** → **Support Information.**
- **Driver version** and **communication log**: Displays in the Studio *Output* window when the driver is running.
- **Device model** and **boards**: Consult the hardware manufacturer's documentation for this information.

Revision History

Doc. Revision	Driver Version	Author	Date	Description of Changes
A	1.00	Eric Vigiani	15 Jun 04	First version
B	1.01	Eric Vigiani	26 Jun 04	Fixed bug with bit writing and reading
C	1.02	Eric Vigiani	29 Sept 04	Fixed bug in the MDS with DB data type
E	1.04	Fabio Carvalho	19 Oct 05	Fixed bug on mapping functions. Changed from (AGLINK_API) to (WINAPI)
F	1.05	Lourenço Teodoro	28 Mar 06	Fixed problems in function GetBit
G	1.06	Graziane C. Forti	11 Jul 06	Implemented String format Implemented the Unsigned and Signed options
H	1.07	Graziane C. Forti	07 Dec 06	Created Initial ID to Connection (to use more than one driver) Implemented "Initial ID Connection" field in Communication Parameters
I	3.00	Rafael R. Fernandes	26 Feb 08	Data types X and D (equivalent to Bit and DWord) were created. Driver was changed for Colon (:) to be optional in MAIN DRIVER SHEET New syntaxes to Data Blocks operand were added. Included TS data type (Timer in string format) Included Special Flags operand (SM).
J	3.01	Rafael R. Fernandes	24 Apr 08	Fixed problem with multiple stations.
K	10.1	Marcelo Carvalho	07 Jan 09	Updated driver version, no changes in the contents.
L	10.3	Fellipe Peternella	21 Apr 09	Modified the driver to support SST format
M	10.4	Lourenço Teodoro	02 Dec 09	- Modified the driver to create less communication groups and optimize the communication
N	10.5	Fellipe Peternella	01 Apr 10	Fixed problem when communicating with multiple stations
O	10.6	André Körbes	21 Jul 10	- Siemens drivers are allowed to coexist on the same application - Added support to S7-200 ethernet communication via CP243 module

				- Fixed virtual group separation
P	10.7	André Körbes	1 Jul 2011	- Fixes on block size check
Q	10.8	Ajay Anumalla	12 Mar 2013	- Included configuration of Remote and Local TSAP for S7-200. - Fixed issues with subnets - Fixed issues with Bit writing on S7-200 - Fixed address offset issue when above 10000 - Fixed a communication issue with WinAC SoftPLC
R	10.8	Andre Bastos	4 Jun 2013	- Updated documentation to show the configuration for S7-1200/1500 and Logo! No changes in the driver
S	10.8	André Körbes	8 Oct 2013	- Updated documentation of the S7-300 and S7-400 station examples to use Rack 0 and Slot 2.
T	10.9	Paulo Balbino Felipe Andrade	16 Oct 2014	- Added support to Simultaneous Connections - Removed Initial Connection ID field
U	10.9	Andre Bastos	17 Apr 2015	- Updated documentation only. No changes in the driver
V	10.10	Anushree Phanse	10 Jan 2017	- Added support to using hostnames in the station field. - Changed INI file to increase range allowed for DB header
W	10.11	Anushree Phanse	10 Feb 2017	-Fixed issue of driver not running on CE devices after previous release.
X	10.12	Anushree Phanse	23 Aug 2017	- Fixed issue of reading long reals or double floats with poor precision and now they read as 64 bit reals
Y	10.13	Anushree Phanse	23 Aug 2018	- Added support for PLC port in station parameters