

# Siemens TCP/IP Unsolicited Ethernet Driver

© 2017 PTC Inc. All Rights Reserved.

# Table of Contents

<b>Siemens TCP/IP Unsolicited Ethernet Driver</b>	<b>1</b>
<b>Table of Contents</b>	<b>2</b>
Siemens TCP/IP Unsolicited Ethernet Driver	3
Overview	3
<b>Setup</b>	<b>4</b>
Channel Properties - General	4
Channel Properties - Ethernet Communications	5
Channel Properties - Write Optimizations	5
Channel Properties - Advanced	6
Channel Properties - Communications Properties	7
Device Properties - General	7
Device Properties - Scan Mode	9
Device Properties - CPU Settings	9
Master Device Configuration	10
<b>Data Types Description</b>	<b>11</b>
<b>Address Descriptions</b>	<b>12</b>
<b>Event Log Messages</b>	<b>17</b>
Failure to start unsolicited communications.   Port number = <number>.	17
<b>Appendix: Configuring Connections Using the SIMATIC Manager</b>	<b>18</b>
Step One: Creating a New Project	18
Step Two: Configuring the Master and PC Station	22
Step Three: Connecting the Master and the Slave Driver	34
Step Four: Inserting Function Blocks	41
Step Five: Creating the DB3 Data Block	47
Step Six: Inserting PUT FB	49
Step Seven: Downloading to the PLC	53
<b>Index</b>	<b>61</b>

## Siemens TCP/IP Unsolicited Ethernet Driver

---

Help version 1.030

### CONTENTS

#### [Overview](#)

What is the Siemens TCP/IP Unsolicited Ethernet Driver?

#### [Setup](#)

How do I configure a device for use with this driver?

#### [Data Types Description](#)

What data types does this driver support?

#### [Address Descriptions](#)

How do I address a data location on a Siemens TCP/IP Ethernet device?

#### [Event Log Messages](#)

What messages does the driver produce?

### Overview

---

The Siemens TCP/IP Unsolicited Ethernet Driver provides a reliable way to connect Siemens TCP/IP Slave Ethernet devices to client applications; including HMI, SCADA, Historian, MES, ERP, and countless custom applications. This driver actd as a simulated Siemens PLC. It is intended for simulation of Siemens S7-300.

## Setup

The Siemens TCP/IP Unsolicited Ethernet Driver supports one channel and up to 256 virtual devices.

## Supported Protocols

S7 Messaging on Industrial Ethernet (ISO 8073 Class 0) over TCP/IP. This is defined in RFC1006.

## Supported Commands

FB14-GET (S7-300)  
 FB15-PUT (S7-300)  
 SFB14-GET (S7-400)  
 SFB15-PUT (S7-400)

## Libraries

This driver requires a standard Ethernet card. No special libraries or hardware are needed.

**Note:** To communicate with this driver, devices require specialized ladder programming.

## Channel Properties

### Device Properties

### Master Device Configuration

### Appendix: Configuring Connections Using the SIMATIC Manager

## Channel Properties - General

This server supports the use of simultaneous multiple communications drivers. Each protocol or driver used in a server project is called a channel. A server project may consist of many channels with the same communications driver or with unique communications drivers. A channel acts as the basic building block of an OPC link. This group is used to specify general channel properties, such as the identification attributes and operating mode.

Property Groups	<input checked="" type="checkbox"/> <b>Identification</b>	
General	Name	
Write Optimizations	Description	
Advanced	Driver	
	<input checked="" type="checkbox"/> <b>Diagnostics</b>	
	Diagnostics Capture	Disable

## Identification

**Name:** User-defined identity of this channel. In each server project, each channel name must be unique. Although names can be up to 256 characters, some client applications have a limited display window when browsing the OPC server's tag space. The channel name is part of the OPC browser information.  
 For information on reserved characters, refer to "How To... Properly Name a Channel, Device, Tag, and Tag Group" in the server help.

**Description:** User-defined information about this channel.

Many of these properties, including Description, have an associated system tag.

**Driver:** Selected protocol / driver for this channel. This property specifies the device driver that was selected during channel creation. It is a disabled setting in the channel properties.

● **Note:** With the server's online full-time operation, these properties can be changed at any time. This includes changing the channel name to prevent clients from registering data with the server. If a client has already acquired an item from the server before the channel name is changed, the items are unaffected. If, after the channel name has been changed, the client application releases the item and attempts to re-acquire using the old channel name, the item is not accepted. With this in mind, changes to the properties should not be made once a large client application has been developed. Utilize the User Manager to prevent operators from changing properties and restrict access rights to server features.

## Diagnostics

**Diagnostics Capture:** When enabled, this option makes the channel's diagnostic information available to OPC applications. Because the server's diagnostic features require a minimal amount of overhead processing, it is recommended that they be utilized when needed and disabled when not. The default is disabled.

● **Note:** This property is disabled if the driver does not support diagnostics.

● For more information, refer to "Communication Diagnostics" in the server help.

## Channel Properties - Ethernet Communications

Ethernet Communication can be used to communicate with devices.

Property Groups	Ethernet Settings	
General	Network Adapter	Default
<b>Ethernet Communications</b>		
Write Optimizations		
Advanced		

### Ethernet Settings

**Network Adapter:** Specify the network adapter to bind. When Default is selected, the operating system selects the default adapter.

## Channel Properties - Write Optimizations

As with any OPC server, writing data to the device may be the application's most important aspect. The server intends to ensure that the data written from the client application gets to the device on time. Given this goal, the server provides optimization properties that can be used to meet specific needs or improve application responsiveness.

Property Groups	Write Optimizations	
General	Optimization Method	Write Only Latest Value for All Tags
<b>Write Optimizations</b>	Duty Cycle	10

### Write Optimizations

**Optimization Method:** controls how write data is passed to the underlying communications driver. The options are:

- **Write All Values for All Tags:** This option forces the server to attempt to write every value to the controller. In this mode, the server continues to gather write requests and add them to the server's

internal write queue. The server processes the write queue and attempts to empty it by writing data to the device as quickly as possible. This mode ensures that everything written from the client applications is sent to the target device. This mode should be selected if the write operation order or the write item's content must uniquely be seen at the target device.

- **Write Only Latest Value for Non-Boolean Tags:** Many consecutive writes to the same value can accumulate in the write queue due to the time required to actually send the data to the device. If the server updates a write value that has already been placed in the write queue, far fewer writes are needed to reach the same final output value. In this way, no extra writes accumulate in the server's queue. When the user stops moving the slide switch, the value in the device is at the correct value at virtually the same time. As the mode states, any value that is not a Boolean value is updated in the server's internal write queue and sent to the device at the next possible opportunity. This can greatly improve the application performance.

● **Note:** This option does not attempt to optimize writes to Boolean values. It allows users to optimize the operation of HMI data without causing problems with Boolean operations, such as a momentary push button.

- **Write Only Latest Value for All Tags:** This option takes the theory behind the second optimization mode and applies it to all tags. It is especially useful if the application only needs to send the latest value to the device. This mode optimizes all writes by updating the tags currently in the write queue before they are sent. This is the default mode.

**Duty Cycle:** is used to control the ratio of write to read operations. The ratio is always based on one read for every one to ten writes. The duty cycle is set to ten by default, meaning that ten writes occur for each read operation. Although the application is performing a large number of continuous writes, it must be ensured that read data is still given time to process. A setting of one results in one read operation for every write operation. If there are no write operations to perform, reads are processed continuously. This allows optimization for applications with continuous writes versus a more balanced back and forth data flow.

● **Note:** It is recommended that the application be characterized for compatibility with the write optimization enhancements before being used in a production environment.

## Channel Properties - Advanced

This group is used to specify advanced channel properties. Not all drivers support all properties; so the Advanced group does not appear for those devices.

Property Groups	[-] <b>Non-Normalized Float Handling</b>	
General	Floating-Point Values	Replace with Zero
Write Optimizations	[-] <b>Inter-Device Delay</b>	
<b>Advanced</b>	Inter-Device Delay (ms)	0

**Non-Normalized Float Handling:** Non-normalized float handling allows users to specify how a driver handles non-normalized IEEE-754 floating point data. A non-normalized value is defined as Infinity, Not-a-Number (NaN), or as a Denormalized Number. The default is Replace with Zero. Drivers that have native float handling may default to Unmodified. Descriptions of the options are as follows:

- **Replace with Zero:** This option allows a driver to replace non-normalized IEEE-754 floating point values with zero before being transferred to clients.
- **Unmodified:** This option allows a driver to transfer IEEE-754 denormalized, normalized, non-number, and infinity values to clients without any conversion or changes.

● **Note:** This property is disabled if the driver does not support floating point values or if it only supports the option that is displayed. According to the channel's float normalization setting, only real-time driver tags (such as values and arrays) are subject to float normalization. For example, EFM data is not affected by this setting.

● For more information on the floating point values, refer to "How To ... Work with Non-Normalized Floating Point Values" in the server help.

**Inter-Device Delay:** Specify the amount of time the communications channel waits to send new requests to the next device after data is received from the current device on the same channel. Zero (0) disables the delay.

● **Note:** This property is not available for all drivers, models, and dependent settings.

## Channel Properties - Communications Properties

Property Groups	<b>Communication Properties</b>	
General	Port Number	102
Ethernet Communications		
Write Optimizations		
Advanced		
<b>Communication Properties</b>		

**Port Number:** Specify the port number on which the driver listens. Devices must be configured to connect to this port: messages sent to all other ports are ignored by the driver. The valid range is 0 to 65535. The default setting is IE TCP/IP: 102 (TSAP).

● **Note:** Non-standard values may be required by routing and firewall issues.

## Device Properties - General

A device represents a single target on a communications channel. If the driver supports multiple controllers, users must enter a device ID for each controller.

Property Groups	<b>Identification</b>	
<b>General</b>	Name	
Scan Mode	Description	
Timing	Channel Assignment	
Auto-Demotion	Driver	
Redundancy	Model	
	ID Format	Decimal
	ID	2
	<b>Operating Mode</b>	
	Data Collection	Enable
	Simulated	No

### Identification

**Name:** This property specifies the name of the device. It is a logical user-defined name that can be up to 256 characters long, and may be used on multiple channels.

● **Note:** Although descriptive names are generally a good idea, some OPC client applications may have a limited display window when browsing the OPC server's tag space. The device name and channel name become part of the browse tree information as well. Within an OPC client, the combination of channel name and device name would appear as "ChannelName.DeviceName".

● For more information, refer to *"How To... Properly Name a Channel, Device, Tag, and Tag Group"* in server help.

**Description:** User-defined information about this device.

● Many of these properties, including Description, have an associated system tag.

**Channel Assignment:** User-defined name of the channel to which this device currently belongs.

**Driver:** Selected protocol driver for this device.

**Model:** This property specifies the specific type of device that is associated with this ID. The contents of the drop-down menu depends on the type of communications driver being used. Models that are not supported by a driver are disabled. If the communications driver supports multiple device models, the model selection can only be changed when there are no client applications connected to the device.

● **Note:** If the communication driver supports multiple models, users should try to match the model selection to the physical device. If the device is not represented in the drop-down menu, select a model that conforms closest to the target device. Some drivers support a model selection called "Open," which allows users to communicate without knowing the specific details of the target device. For more information, refer to the driver help documentation.

**ID:** This property specifies the device's driver-specific station or node. The type of ID entered depends on the communications driver being used. For many communication drivers, the ID is a numeric value. Drivers that support a Numeric ID provide users with the option to enter a numeric value whose format can be changed to suit the needs of the application or the characteristics of the selected communications driver. The ID format can be Decimal, Octal, and Hexadecimal.

● **Note:** If the driver is Ethernet-based or supports an unconventional station or node name, the device's TCP/IP address may be used as the device ID. TCP/IP addresses consist of four values that are separated by periods, with each value in the range of 0 to 255. Some device IDs are string based. There may be additional properties to configure within the ID field, depending on the driver. For more information, refer to the driver's help documentation.

## Operating Mode

**Data Collection:** This property controls the device's active state. Although device communications are enabled by default, this property can be used to disable a physical device. Communications are not attempted when a device is disabled. From a client standpoint, the data is marked as invalid and write operations are not accepted. This property can be changed at any time through this property or the device system tags.

**Simulated:** This option places the device into Simulation Mode. In this mode, the driver does not attempt to communicate with the physical device, but the server continues to return valid OPC data. Simulated stops physical communications with the device, but allows OPC data to be returned to the OPC client as valid data. While in Simulation Mode, the server treats all device data as reflective: whatever is written to the simulated device is read back and each OPC item is treated individually. The item's memory map is based on the group



Update Rate. The data is not saved if the server removes the item (such as when the server is reinitialized). The default is No.

#### Notes:

1. This System tag (\_Simulated) is read only and cannot be written to for runtime protection. The System tag allows this property to be monitored from the client.
2. In Simulation mode, the item's memory map is based on client update rate(s) (Group Update Rate for OPC clients or Scan Rate for native and DDE interfaces). This means that two clients that reference the same item with different update rates return different data.

Simulation Mode is for test and simulation purposes only. It should never be used in a production environment.

## Device Properties - Scan Mode

The Scan Mode specifies the subscribed-client requested scan rate for tags that require device communications. Synchronous and asynchronous device reads and writes are processed as soon as possible; unaffected by the Scan Mode properties.

Property Groups	<b>Scan Mode</b>	
General	Scan Mode	Respect Client-Specified Scan Rate ▼
<b>Scan Mode</b>	Initial Updates from Cache	Disable

**Scan Mode:** specifies how tags in the device are scanned for updates sent to subscribed clients. Descriptions of the options are:

- **Respect Client-Specified Scan Rate:** This mode uses the scan rate requested by the client.
- **Request Data No Faster than Scan Rate:** This mode specifies the maximum scan rate to be used. The valid range is 10 to 99999990 milliseconds. The default is 1000 milliseconds.
  - **Note:** When the server has an active client and items for the device and the scan rate value is increased, the changes take effect immediately. When the scan rate value is decreased, the changes do not take effect until all client applications have been disconnected.
- **Request All Data at Scan Rate:** This mode forces tags to be scanned at the specified rate for subscribed clients. The valid range is 10 to 99999990 milliseconds. The default is 1000 milliseconds.
- **Do Not Scan, Demand Poll Only:** This mode does not periodically poll tags that belong to the device nor perform a read to get an item's initial value once it becomes active. It is the client's responsibility to poll for updates, either by writing to the \_DemandPoll tag or by issuing explicit device reads for individual items. *For more information, refer to "Device Demand Poll" in server help.*
- **Respect Tag-Specified Scan Rate:** This mode forces static tags to be scanned at the rate specified in their static configuration tag properties. Dynamic tags are scanned at the client-specified scan rate.

**Initial Updates from Cache:** When enabled, this option allows the server to provide the first updates for newly activated tag references from stored (cached) data. Cache updates can only be provided when the new item reference shares the same address, scan rate, data type, client access, and scaling properties. A device read is used for the initial update for the first client reference only. The default is disabled; any time a client activates a tag reference the server attempts to read the initial value from the device.

## Device Properties - CPU Settings

Property Groups	<b>CPU Settings</b>	
General	Rack Number	0
Scan Mode	CPU Slot	0
<b>CPU Settings</b>		

**Rack Number:** This property specifies the number of the rack in which the simulated CPU of interest resides. The valid range is 0 to 7. Devices must have unique rack and slot values. The default setting is 0.

**CPU Slot:** This property specifies the number of the slot in which the simulated CPU of interest resides. The valid range is 0 to 31. Devices must have unique rack and slot values. The default setting is 0.

## Master Device Configuration

Siemens PLCs must be programmed to issue read and write commands to the driver and to handle returned data. *For more information, refer to the Siemens PLC programming documentation. For information on preparing the Master Device and the unsolicited driver for communications, refer to [Configuring Connections Using the SIMATIC Manager](#).*

Messages must be sent to the IP address of the selected Ethernet adapter of the host computer running the unsolicited driver. To do so, update the channel properties.

• *For more information concerning the port number configured for the simulated device, refer to [Communication Properties](#).*

## Data Types Description

Data Type	Description
Boolean	Single bit
Byte	Unsigned 8-bit value
Char	Signed 8-bit value
Word	Unsigned 16-bit value  bit 0 is the low bit bit 15 is the high bit
Short	Signed 16-bit value  bit 0 is the low bit bit 14 is the high bit bit 15 is the sign bit
DWord	Unsigned 32-bit value  bit 0 is the low bit bit 31 is the high bit
Long	Signed 32-bit value  bit 0 is the low bit bit 30 is the high bit bit 31 is the sign bit
BCD	Two byte packed BCD  Value range is 0-9999. Behavior is undefined for values beyond this range.
LBCD	Four byte packed BCD  Value range is 0-99999999. Behavior is undefined for values beyond this range.
Float	32-bit floating point value.  The driver interprets two consecutive registers as a floating point value by making the second register the high word and the first register the low word.
String	NULL-terminated ASCII string

## Address Descriptions

The following information applies to both S7-300 and S7-400 models. The default data types for dynamically defined tags are shown in **bold**.

Address Type	Range	Type	Access
Discrete Inputs	I0.b-I4095.b* .b is Bit Number 0-7	<b>Boolean</b>	Read/Write
	IB0-IB4095	<b>Byte</b> , Char, String**	Read/Write
	IW0-IW4094	<b>Word</b> , Short, BCD	Read/Write
	IW:KT0-IW:KT4094	DWord, <b>Long</b>	Read/Write
	IW:KC0-IW:KC4094	<b>Word</b> , Short	Read/Write
	ID0-ID4092	<b>DWord</b> , Long, LBCD, Float	Read/Write
Discrete Inputs   <b>Note:</b> I and E access the same memory area.	E0.b-E4095.b* .b is Bit Number 0-7	<b>Boolean</b>	Read/Write
	EB0-EB4095**	<b>Byte</b> , Char, String**	Read/Write
	EW0-EW4094	<b>Word</b> , Short, BCD	Read/Write
	EW:KT0-EW:KT4094	DWord, <b>Long</b>	Read/Write
	EW:KC0-EW:KC4094	<b>Word</b> , Short	Read/Write
	ED0-ED4092	<b>DWord</b> , Long, LBCD, Float	Read/Write
Discrete Outputs	Q0.b-Q4095.b* .b is Bit Number 0-7	<b>Boolean</b>	Read/Write
	QB0-QB4095	<b>Byte</b> , Char, String**	Read/Write
	QW0-QW4094	<b>Word</b> , Short, BCD	Read/Write
	QW:KT0-QW:KT4094	DWord, <b>Long</b>	Read/Write
	QW:KC0-QW:KC4094	<b>Word</b> , Short	Read/Write
	QD0-QD4092	<b>DWord</b> , Long, LBCD, Float	Read/Write
Discrete Outputs	A0.b- A4095.b*	<b>Boolean</b>	Read/Write

Address Type	Range	Type	Access
 <b>Note:</b> Q and A access the same memory area.	.b is Bit Number 0-7		
	AB0-AB4095	<b>Byte</b> , Char, String**	Read/Write
	AW0-AW4094	<b>Word</b> , Short, BCD	
	AW:KT0-AW:KT4094	DWord, <b>Long</b>	Read/Write
	AW:KC0-AW:KC4094	<b>Word</b> , Short	Read/Write
	AD0-AD4092	<b>DWord</b> , Long, LBCD, Float	Read/Write Read/Write
Internal Memory	F0.b-F4095.b*	<b>Boolean</b>	Read/Write
	.b is Bit Number 0-7		
	FB0-FB4095	<b>Byte</b> , Char, String**	Read/Write
	FW0-FW4094	<b>Word</b> , Short, BCD	
	FW:KT0-FW:KT4094	DWord, <b>Long</b>	Read/Write
	FW:KC0-FW:KC4094	<b>Word</b> , Short	Read/Write
	FD0-FD4092	<b>DWord</b> , Long, LBCD, Float	Read/Write Read/Write
Internal Memory	M0.b-M4095.b*	<b>Boolean</b>	Read/Write
	.b is Bit Number 0-7		
	MB0-MB4095	<b>Byte</b> , Char, String**	Read/Write
	MW0-MW4094	<b>Word</b> , Short, BCD	Read/Write
	MW:KT0-MW:KT4094	DWord, <b>Long</b>	Read/Write
 <b>Note:</b> F and M access the same memory area.	MW:KC0-MW:KC4094	<b>Word</b> , Short	
	MD0- MD4092	<b>DWord</b> , Long, LBCD, Float	Read/Write Read/Write
Data Block Boolean	DB1-N:KM0.b-KM4094.b* 1-N is Block Number .b is Bit Number 0-15  <i>Alternates</i>	<b>Boolean</b>	Read/Write

Address Type	Range	Type	Access
	DB1DBX0.b- DBNDBX4094.b* 1-N is Block Number .b is Bit Number 0-15  DB1D0.b-DBND4094.b* 1-N is Block Number .b is Bit Number 0-15	<b>Boolean</b>  <b>Boolean</b>	Read/Write  Read/Write
Data Block Left Byte	DB1-N:KL0-KL4095 1-N is Block Number  <i>Alternates</i>  DB1DBB0- DBNDBB4095 1-N is Block Number  DB1DL0-DBNDL4095 1-N is Block Number	<b>Byte, Char, String**</b>  <b>Byte, Char, String**</b>  <b>Byte, Char, String**</b>	Read/Write  Read/Write  Read/Write
Data Block Right Byte	DB1-N:KR0-KR4094 1-N is Block Number  <i>Alternates</i>  DB1DR0-DBNDR4094 1-N is Block Number	<b>Byte, Char, String**</b>  <b>Byte, Char, String**</b>	Read/Write  Read/Write
Data Block Unsigned Word	DB1-N:KH0-KH4094 1-N is Block Number	<b>Word, Short, BCD</b>	Read/Write
Data Block Signed Word	DB1-N:KF0-KF4094 1-N is Block Number  <i>Alternates</i>  DB1DBW0- DBNDBW4094 1-N is Block Number  DB1DW0-DBNDW4094 1-N is Block Number	Word, <b>Short</b> , BCD  Word, <b>Short</b> , BCD  Word, <b>Short</b> , BCD	Read/Write  Read/Write  Read/Write
Data Block Signed Long	DB1-N:KD0-KD4092 1-N is Block Number  <i>Alternates</i>  DB1DBD0- DB1DBD4092 1-N is Block Number	DWord, <b>Long</b> , LBCD, Float  DWord, <b>Long</b> , LBCD, Float	Read/Write  Read/Write

Address Type	Range	Type	Access
	DB1DD0-DB1DD4092 1-N is Block Number	DWord, <b>Long</b> , LBCD, Float	Read/Write
Data Block Float	DB1-N:KG0-KG4092 1-N is Block Number	<b>Float</b>	Read/Write
Data Block BCD	DB1-N:BCD0-BCD4094 1-N is Block Number	<b>Word</b> , Short	Read/Write
Data Block S5 Timer as DB	DB1-N:KT0-KT4094 1-N is Block Number	DWord, <b>Long</b>	Read/Write
Data Block S5 Counter as DB	DB1-N:KC0-KC4094 1-N is Block Number	<b>Word</b> , Short	Read/Write
Data Block String	DB1:S0.n-DB1:S4095.n* .n is string length. 0<n<=218.	<b>String</b>	Read/Write

\*These memory types/subtypes do not support arrays.

\*\*Byte memory types (MB) support strings. The syntax for strings is <address>.<length> where 0<length<=218.

#### Notes:

1. All offsets for memory types I, Q, and F represent a byte starting location within the specified memory type.
2. Use caution when modifying Word, Short, DWord, and Long types. For I, Q and F, each address starts at a byte offset within the device. Therefore, Words FW0 and FW1 overlap at byte 1. Writing to FW0 modifies the value held in FW1. Similarly, DWord, and Long types can also overlap. It is recommended that these memory types be used so that overlapping does not occur. For example, when using DWords, FD0, FD4, FD8 and so on should be used in order to prevent overlapping bytes.

## Arrays

All memory types/subtypes support arrays (excepting those discussed above). The valid syntax for declaring an array is described below. If no rows are specified, row count of 1 is assumed.

```
<address>[rows][cols]
<address>.rows.cols
<address>,rows,cols
<address>_rows_cols
```

For Word, Short, BCD and "KT" arrays, the base address+(rows\*cols\*2) cannot exceed 4096. The elements of the array are words, and are located on a word boundary. For example, IW0[4] would return IW0, IW2, IW4 and IW6. "KT" subtypes fall into the 16-bit category because the data stored in the PLC is contained within a Word.

For Float, DWord, Long and Long BCD arrays (excluding "KT" subtypes), the base address+(rows\*cols\*4) cannot exceed 4096. Keep in mind that the elements of the array are DWords, located on a DWord boundary. For example, ID0[4] returns ID0, ID4, ID8 and ID12.

For all arrays, the total number of bytes being requested cannot exceed the internal block size of 218 bytes.

### KL vs. KR vs. DBB

KL and KR determine whether the left byte or right byte of the data block word is returned.

Value	8	9	A	B	C
Byte	0	1	2	3	4

#### Example 1

DB1:KH0=0x89

DB1:KL0=0x8

DB1:KR0=0x9

DB1:DBB0=0x8

#### Example 2

DB1:KH1=0x9A

DB1:KL1=0x9

DB1:KR1=0xA

DB1:DBB1=0x9

### Examples

- To access bit 3 of Internal Memory F20, declare an address as follows: F20.3
- To access Data Block 5 as word memory at byte 30, declare an address as follows: DB5:KH30
- To access Data Block 2 byte 20 and bit 7, declare an address as follows: DB2:KM20.7
- To access Data Block 1 as left byte memory at byte 10, declare an address as follows: DB1:KL10
- To access Internal Memory F20 as a DWord, declare an address as follows: FD20
- To access Input Memory I10 as a Word, declare an address as follows: IW10



## Event Log Messages

The following information concerns messages posted to the Event Log pane in the main user interface. Consult the server help on filtering and sorting the Event Log detail view. Server help contains many common messages, so should also be searched. Generally, the type of message (informational, warning) and troubleshooting information is provided whenever possible.

---

### **Failure to start unsolicited communications. | Port number = <number>.**

#### **Error Type:**

Error

#### **Possible Cause:**

1. The driver was not able to create a listen socket for unsolicited communications. Another application may be using the port specified.
2. There may be low system resources.

#### **Possible Solution:**

1. Use network monitor software to see if another application is using the port. If so, shut down the conflicting application and restart the OPC Server. If the conflicting application is free to pick any available port, make sure the server is always started first so it can claim the required port. If both the PLC programming software and this driver must use the same port, they may not be able to be used simultaneously.
2. Verify there are adequate system resources or release resources from other processes.

#### **See Also:**

Channel Setup

# Appendix: Configuring Connections Using the SIMATIC Manager

Connections are configured using the SIMATIC Manager software. The following topics provide information on configuring the Siemens TCP/IP Unsolicited Ethernet Driver to run in unsolicited mode, and demonstrate a basic setup using the S7-300 PLC as the active partner and the driver as the passive partner.

● **Note:** The Siemens TCP/IP Unsolicited Ethernet Driver can configure 256 devices, each with an associated slot/rack. When the active partner (master) communicates with the passive partner (unsolicited driver), it directs its requests to a specific device in the unsolicited driver. Multiple remote partners can talk to the same device.

To jump to a specific section, select a link from the list below.

[Step One: Creating a New Project](#)

[Step Two: Configuring the Master and PC Station](#)

[Step Three: Connecting the Master and the Slave Driver](#)

[Step Four: Inserting Function Blocks](#)

[Step Five: Creating the DB3 Data Block](#)

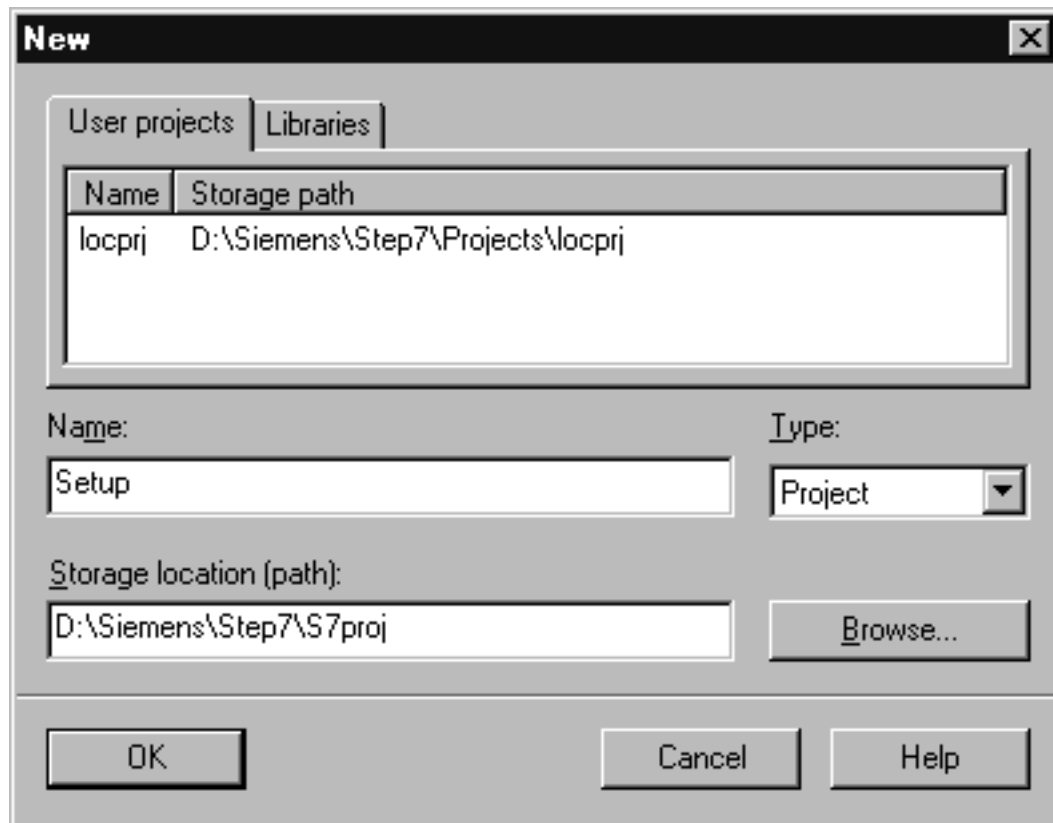
[Step Six: Inserting PUT FB](#)

[Step Seven: Downloading to the PLC](#)

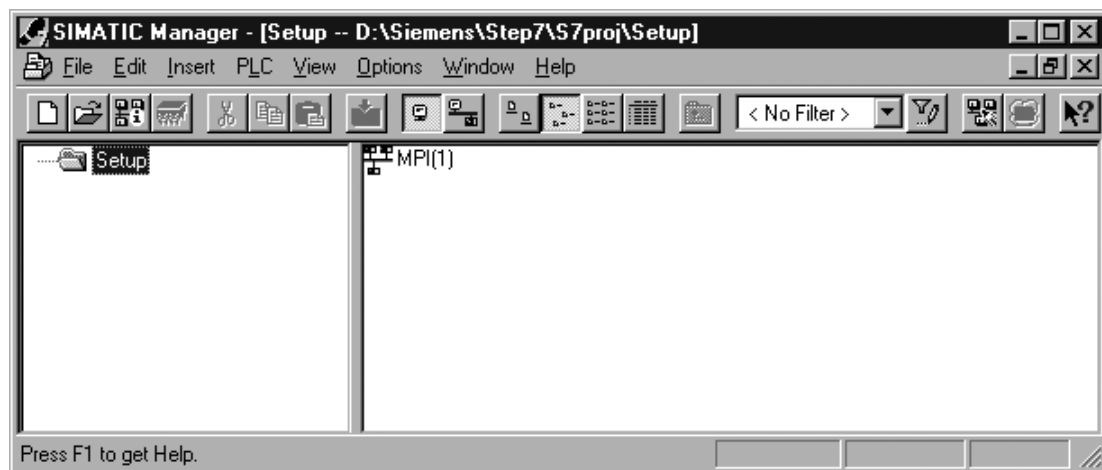
## Step One: Creating a New Project

---

1. To start, open the SIMATIC Manager software and then create a new project. In this example, the project being used is "Setup".

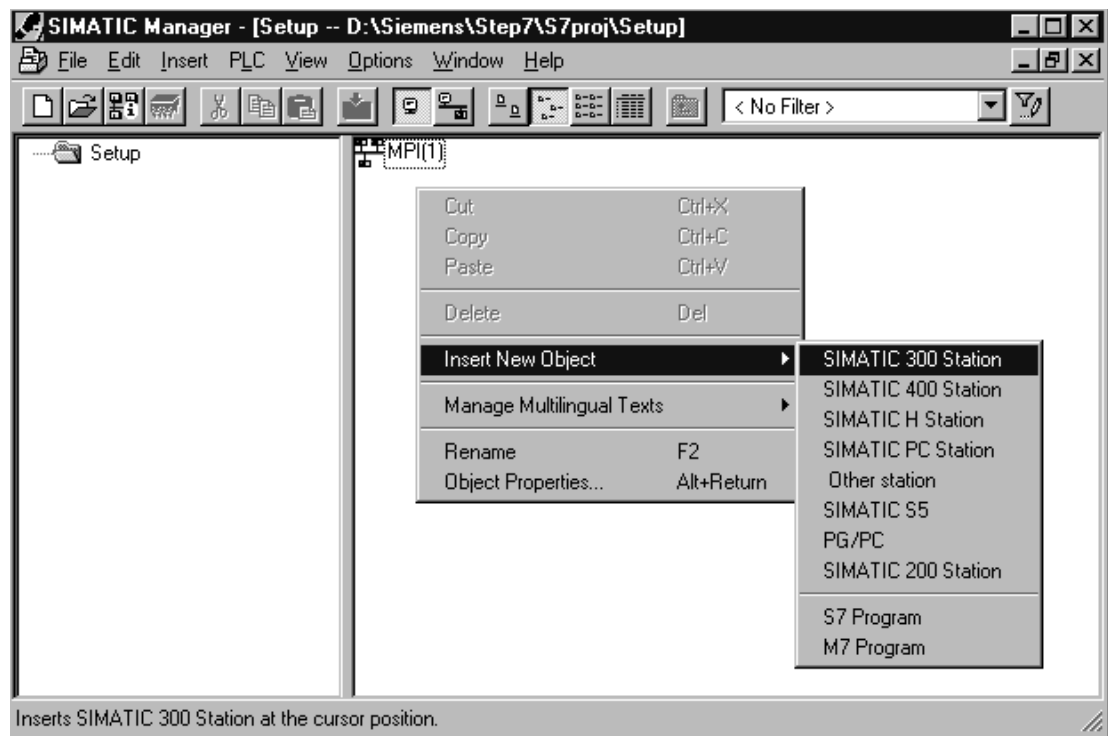


● **Note:** The project's main window should appear as shown below.

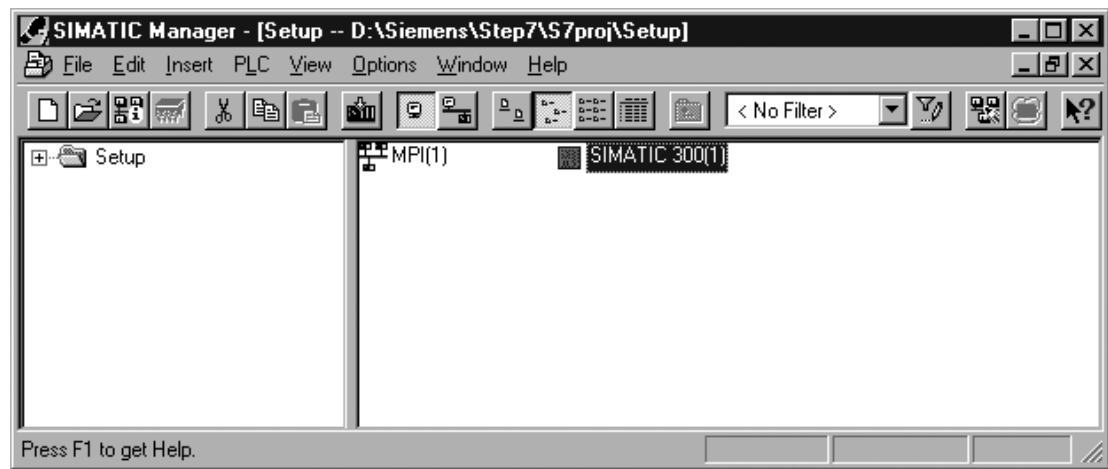


2. Next, create the Master and PC Station. To do so, right-click in the right pane of the window and then select **Insert New Object | SIMATIC 300 Station**.

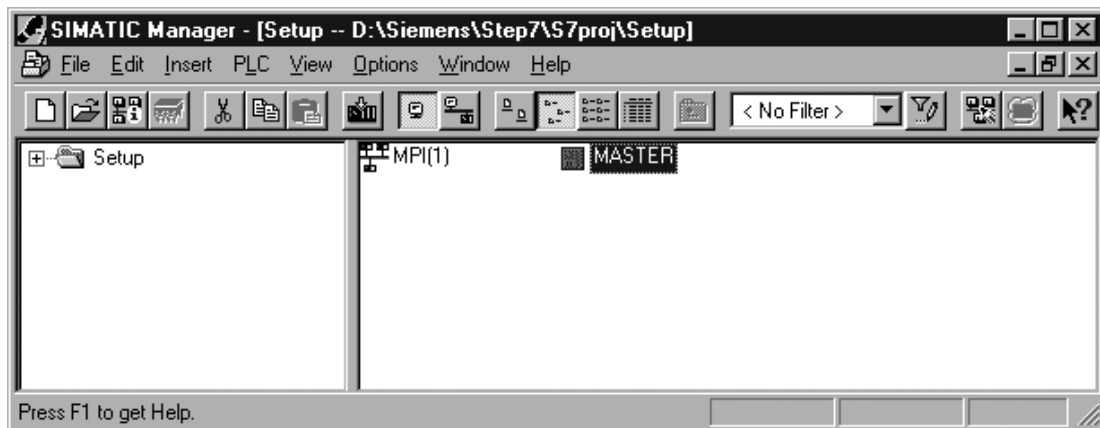
● **Note:** The Master unit is the active partner or the image of the actual PC. The PC Station is the PC on which the SIMATIC Manager software is running.



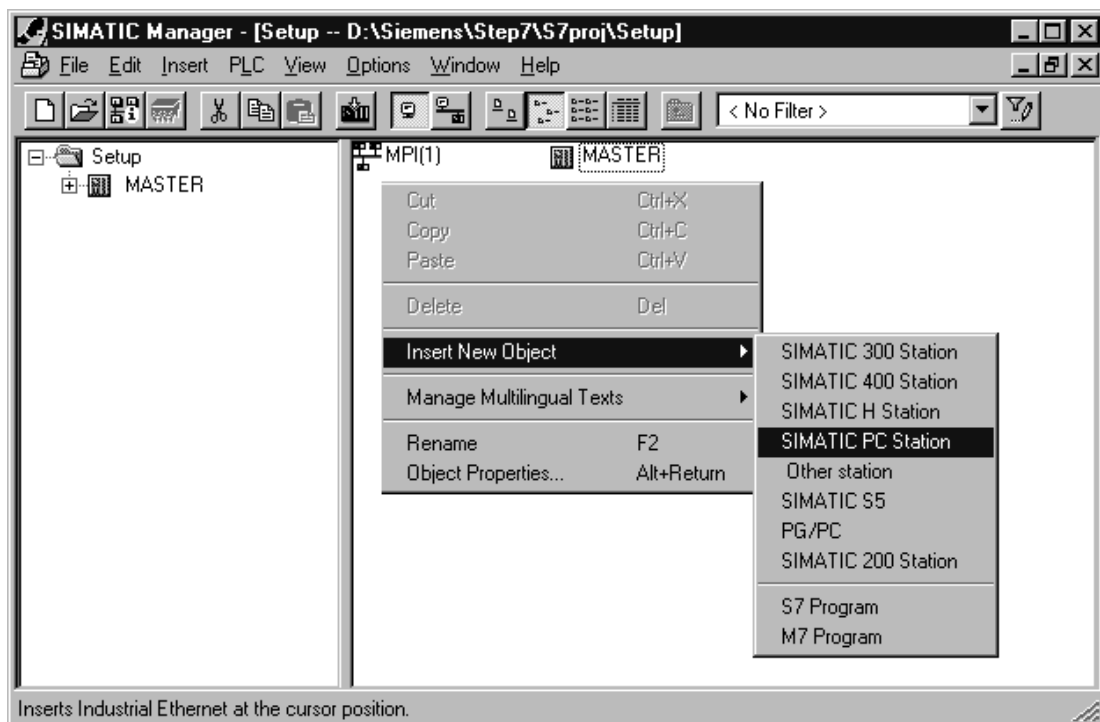
● **Note:** The SIMATIC 300 station should appear as shown below.



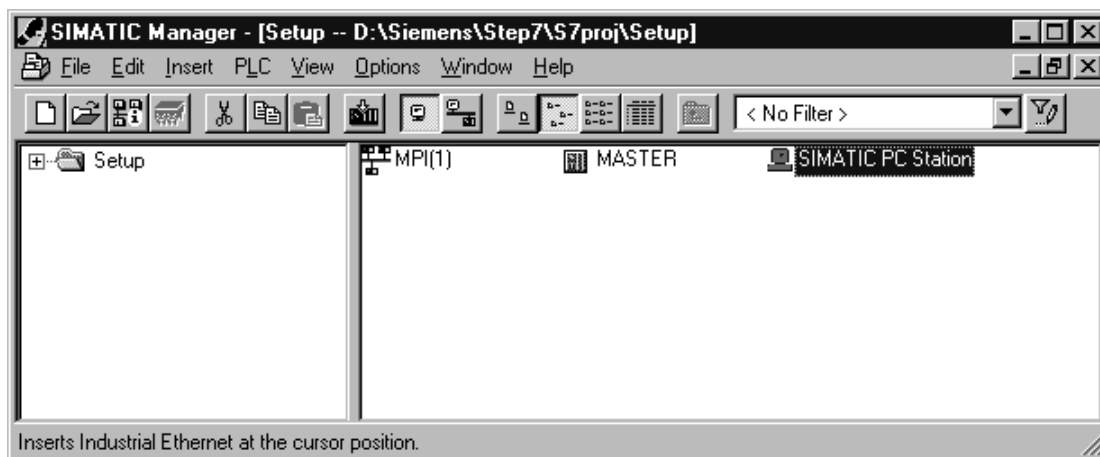
3. Name the new station "MASTER," because it represents the communication's active partner.



4. Next, right-click in the right pane of the window and then select **Insert New Object | SIMATIC PC Station**.



● **Note:** The SIMATIC PC Station should appear as shown below.

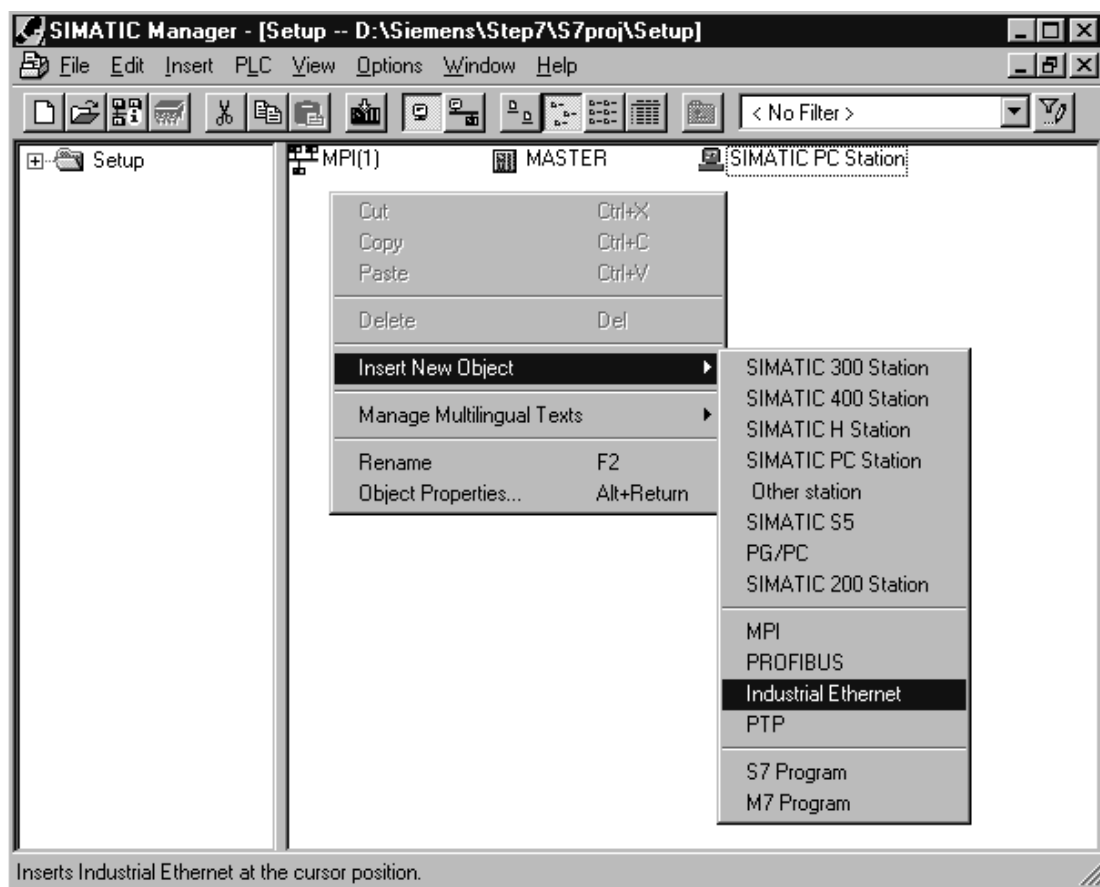


For more information, refer to [Step Two: Configuring the Master and PC Station](#).

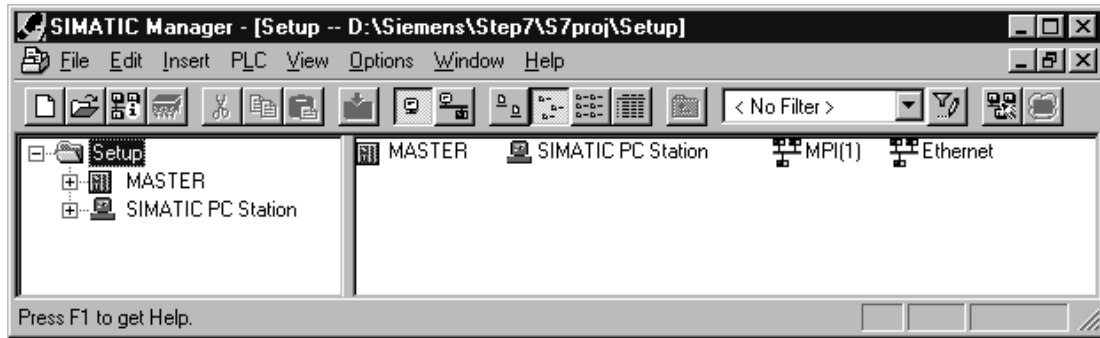
## Step Two: Configuring the Master and PC Station

Industrial Ethernet (IE) is the protocol used for communication.

1. To start, right-click in the right pane of the SIMATIC Manager window. Then, select **Insert New Object | Industrial Ethernet**.



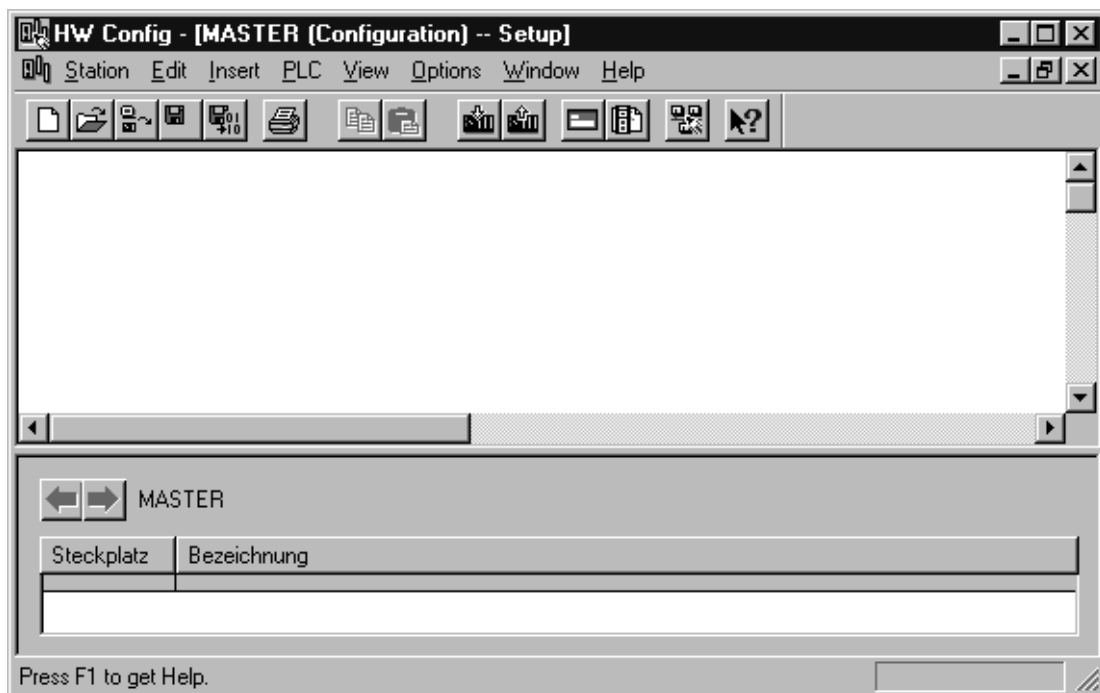
**Note:** The main window should now display an Ethernet icon.



- Next, select the MASTER icon in the left pane of the window. Then, double-click on **Hardware**.

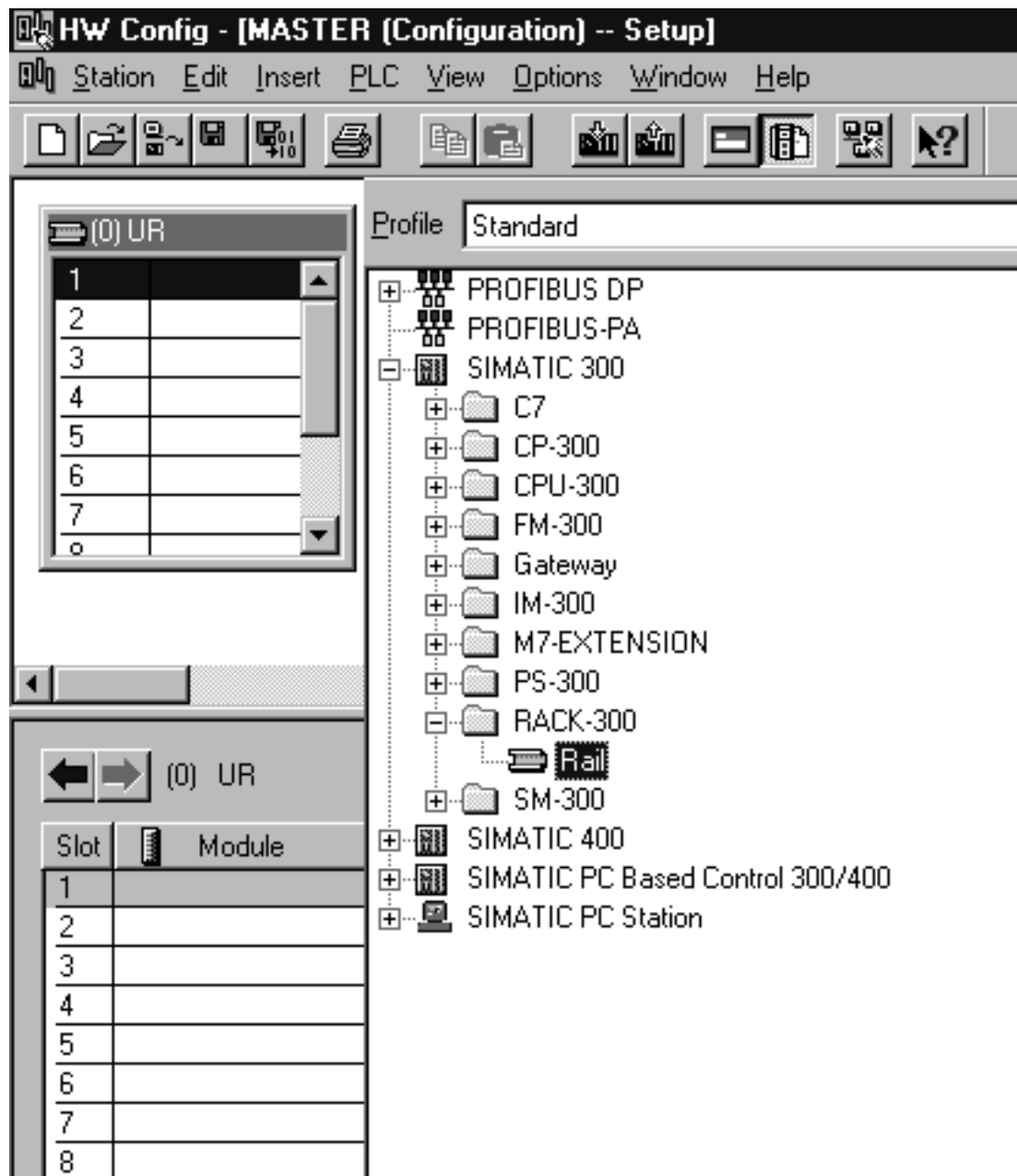


● **Note:** The HW Config window should appear as shown below.



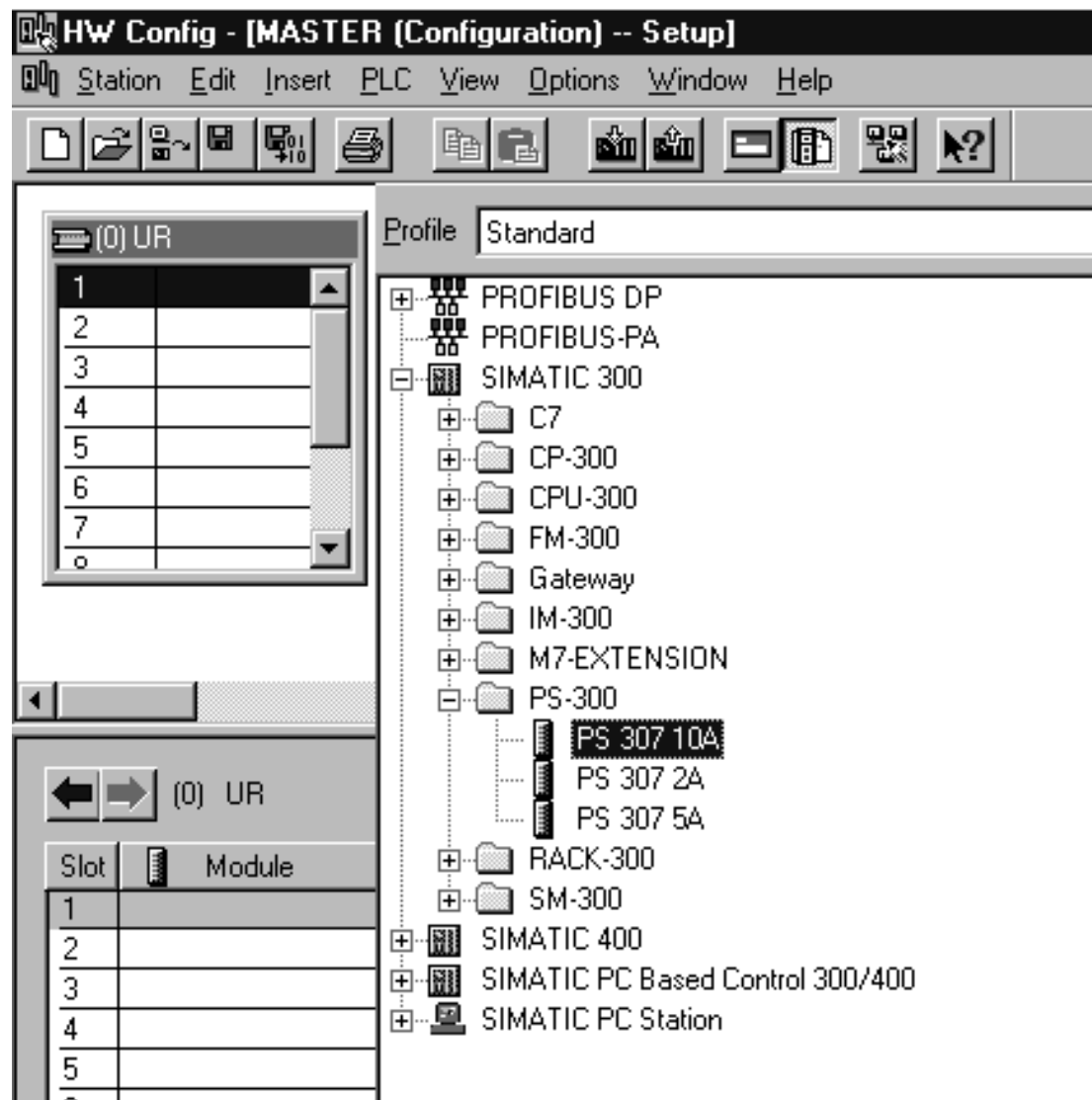
- Next, open the **View** tab and select **Catalog**. Then, expand the **SIMATIC 300** menu and the **Rack 300** menu.

4. To insert the racks, double-click on **Rail**.

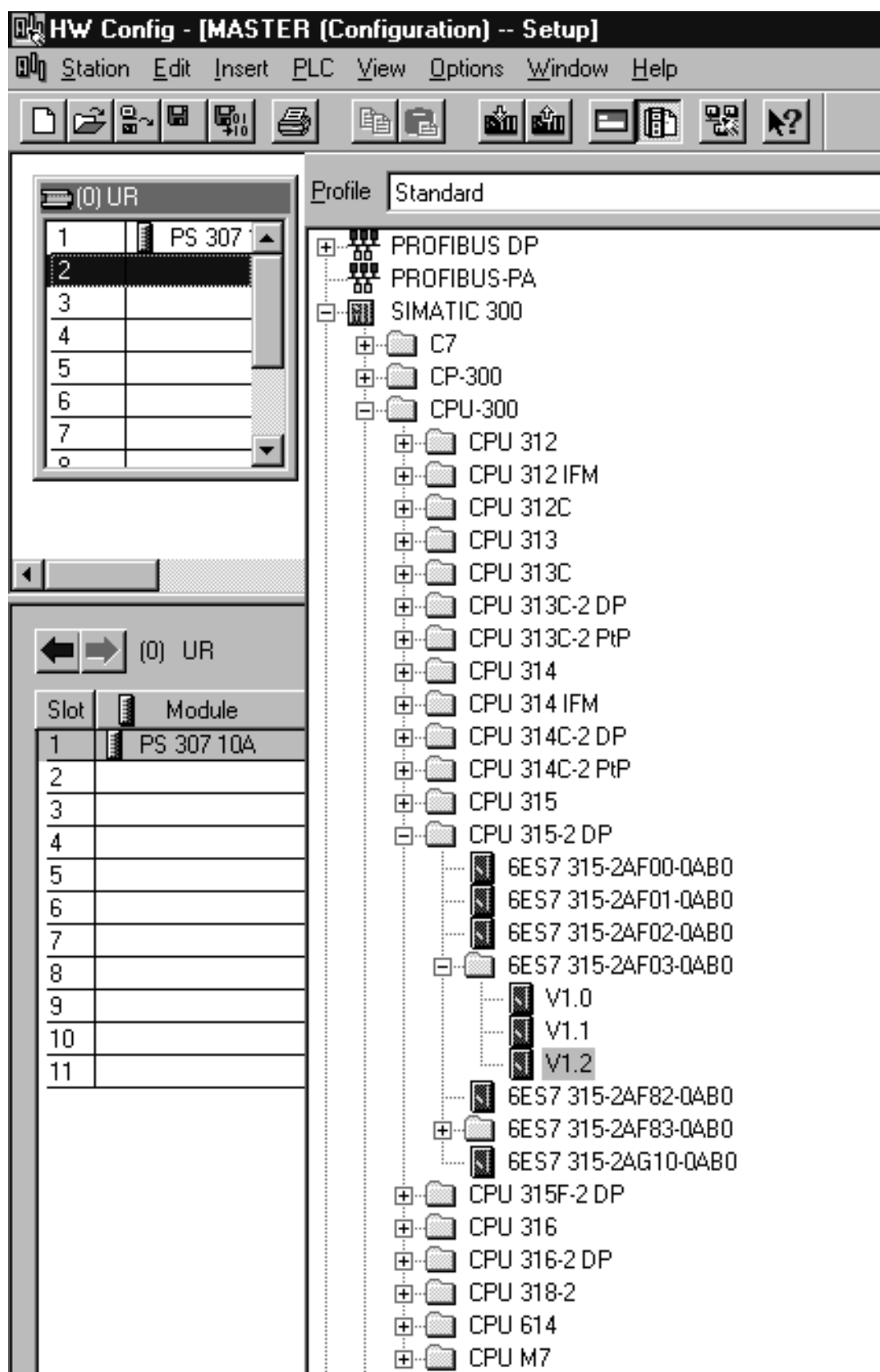


5. Next, expand the **PS 300** menu. Double-click on **PS 307 10A** or any other suitable option to insert the power supply into slot 1.

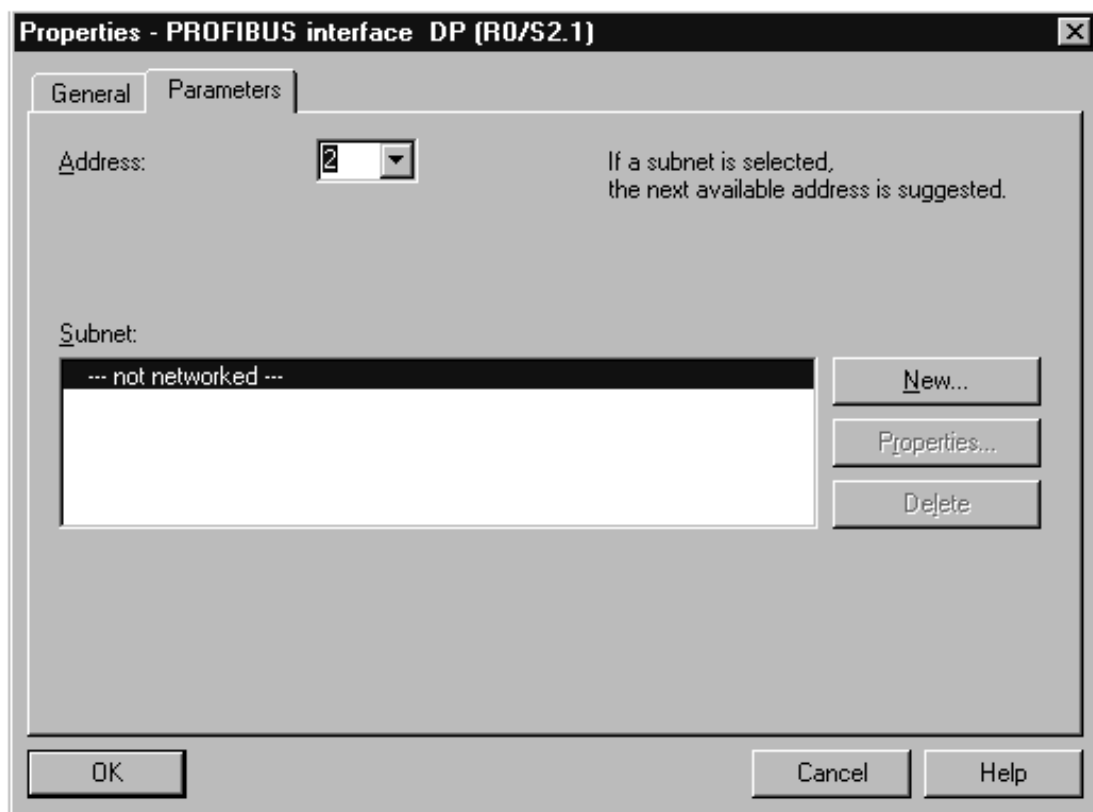




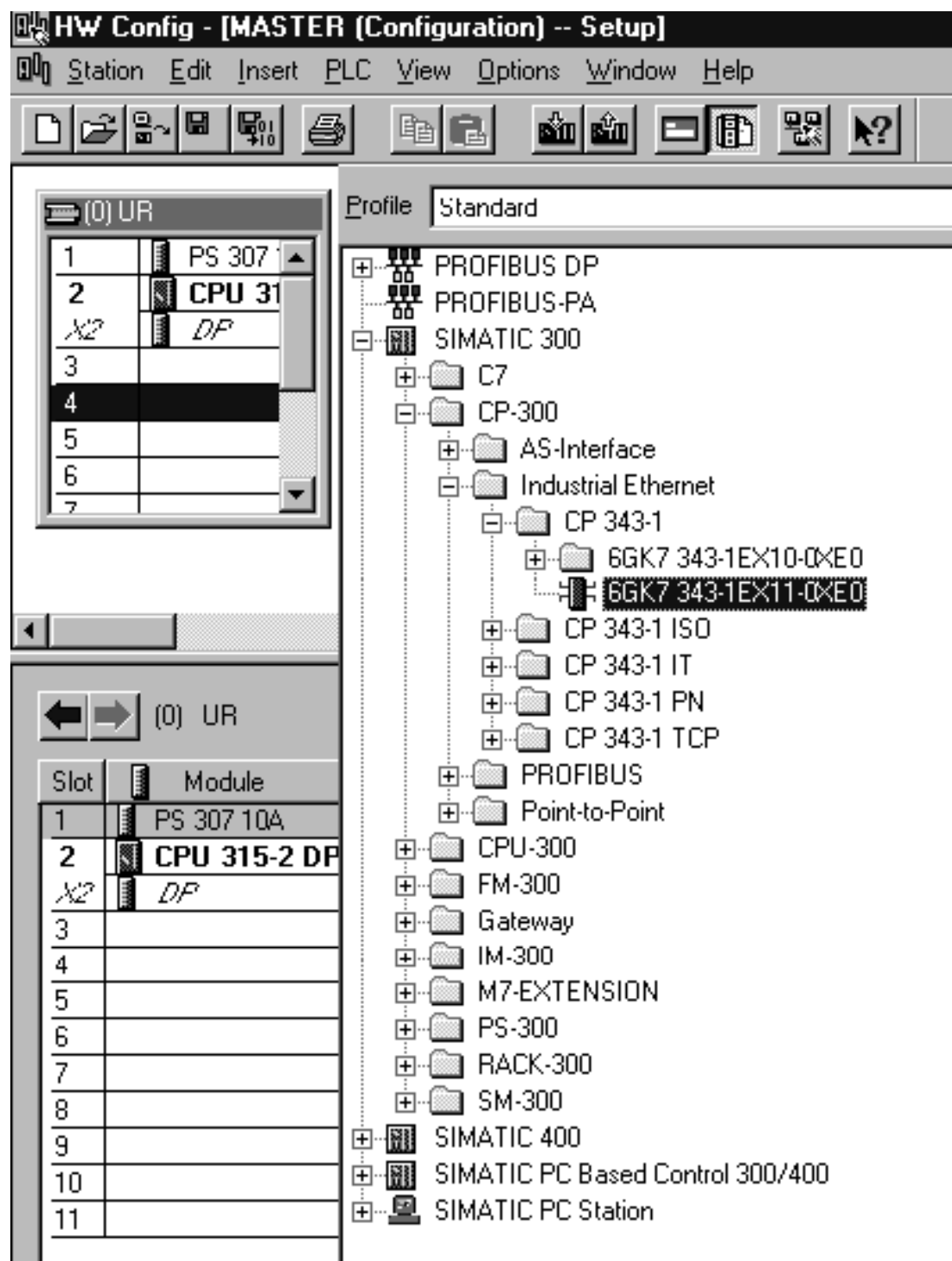
6. To insert the CPU, expand both the **CPU 300** menu and the **CPU 315-2 DP** menu. Then, double-click on the CPU that matches the hardware.



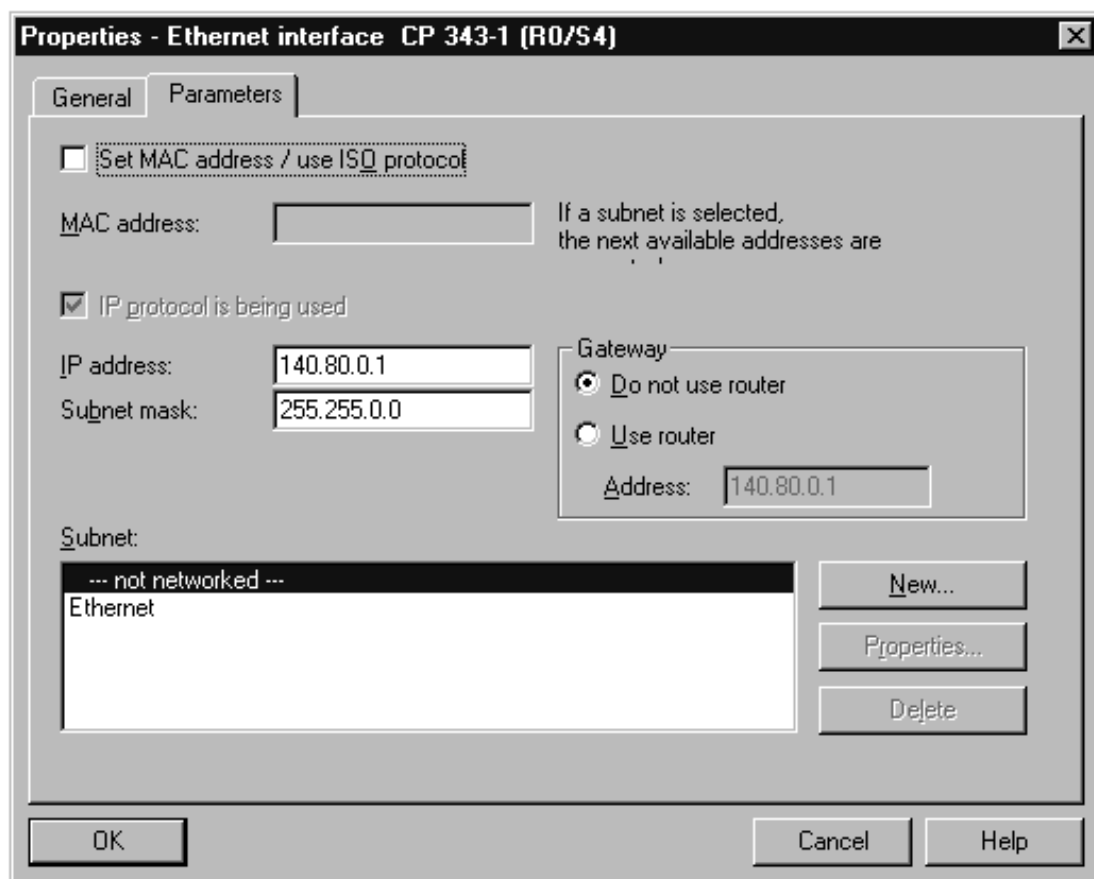
7. To insert the CPU into slot 2, click **OK**.



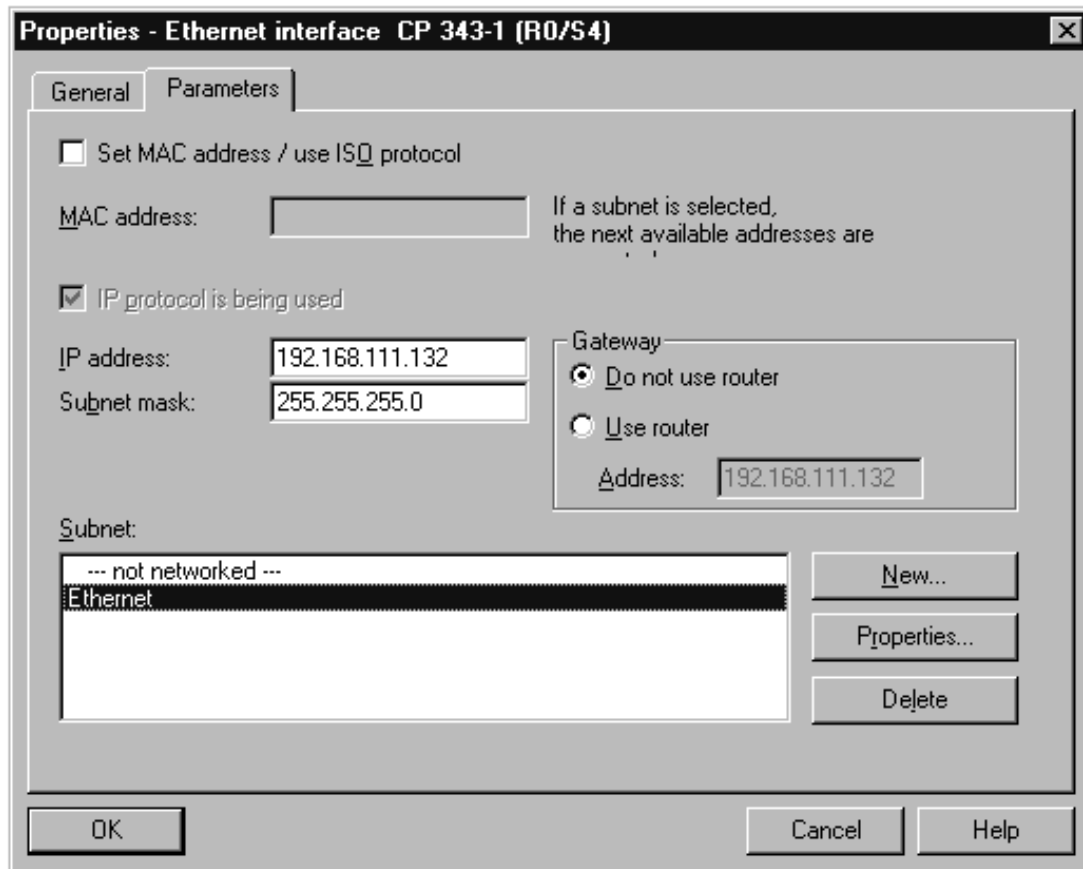
8. To insert the CP, leave slot 3 empty and then click on slot 4 in the racks.
9. Next, expand both the **CP 300** menu and the **Industrial Ethernet** menu. Then, double-click on the CP that matches the hardware.



● **Note:** The window should appear as shown below.

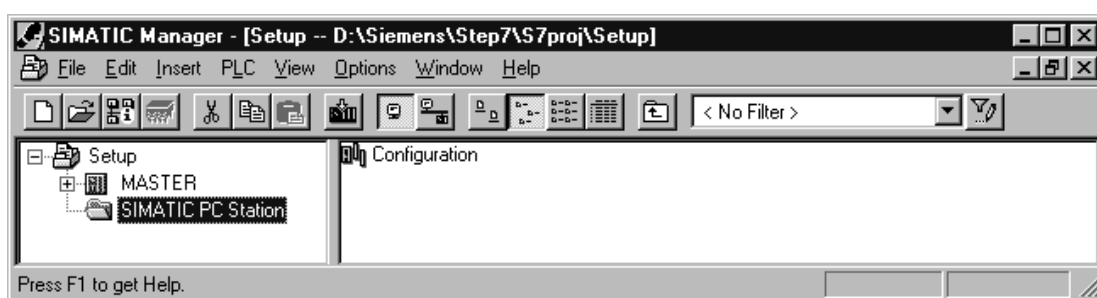


10. Next, enter the PLC's IP address and subnet mask. Then, select **Ethernet** from the subnet box.
11. Click **OK** to configure the Master.

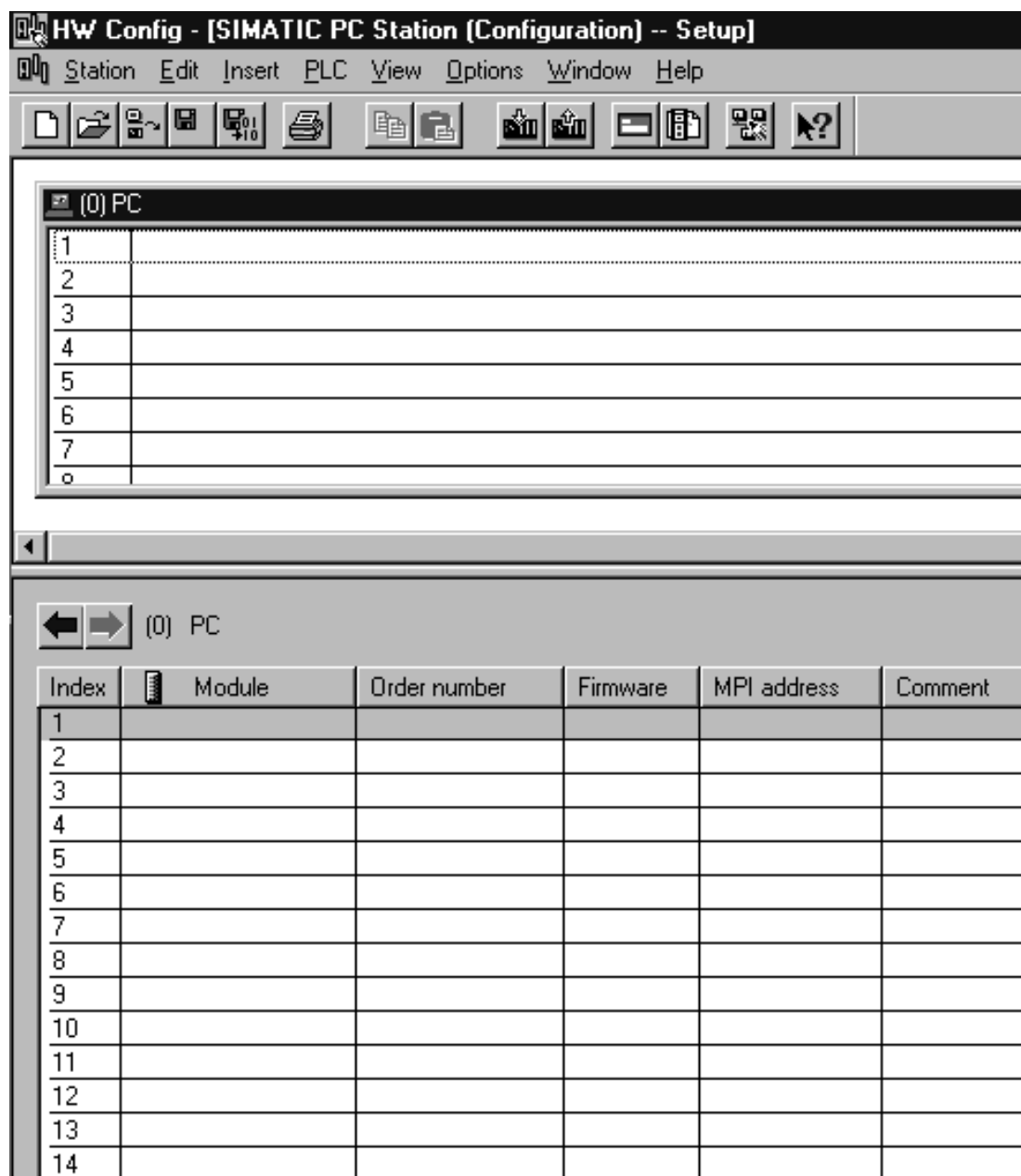


● **Note:** Once finished, open the **View** tab and then select **Catalog** to hide the catalog window.

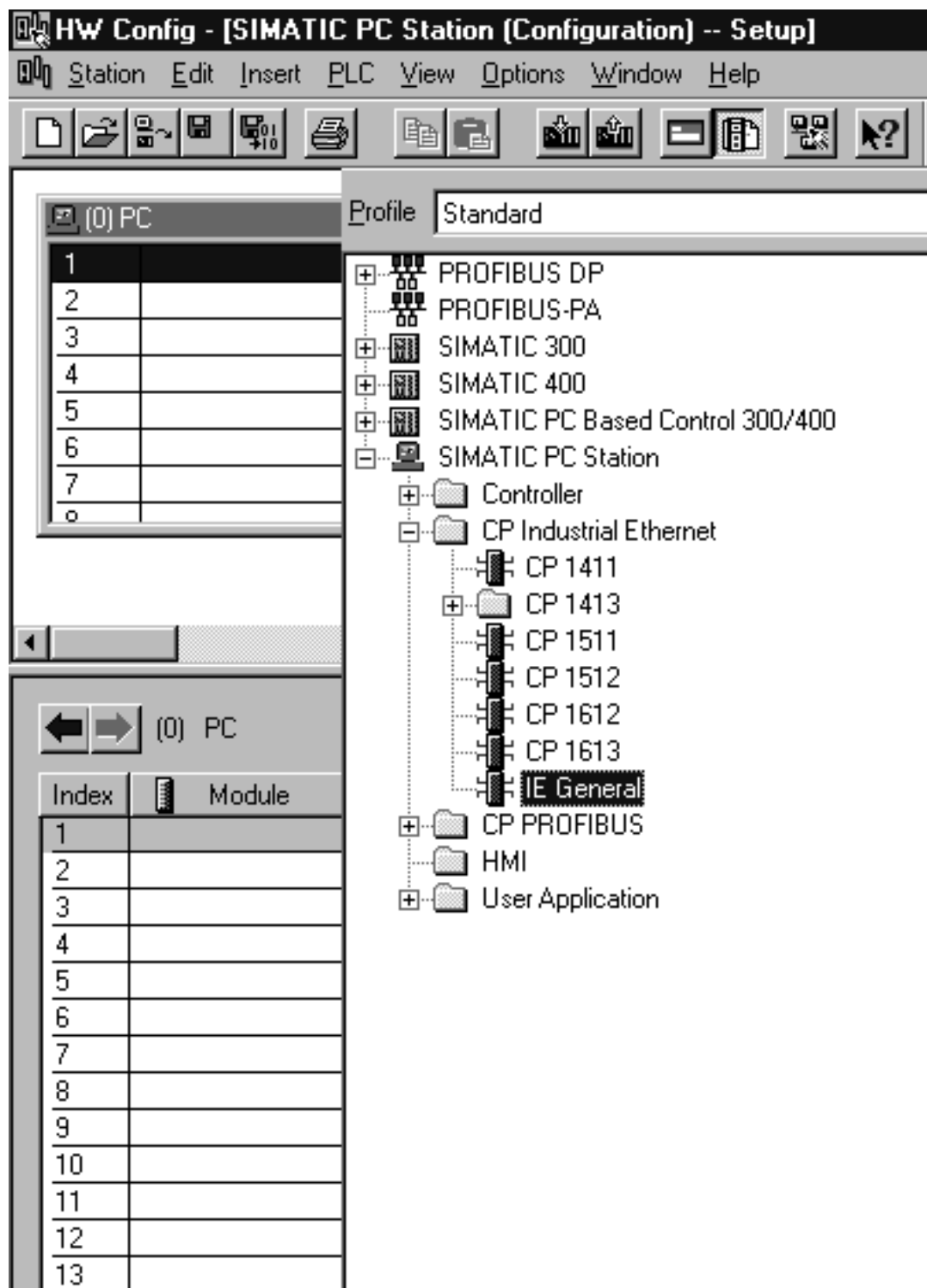
12. Save and exit the HW Configuration window.
13. To configure the PC station, click on the SIMATIC PC Station in the left pane of the SIMATIC Manager window. Then, double-click on **Configuration**.



14. Next, click on the **View** tab and select **Catalog**.

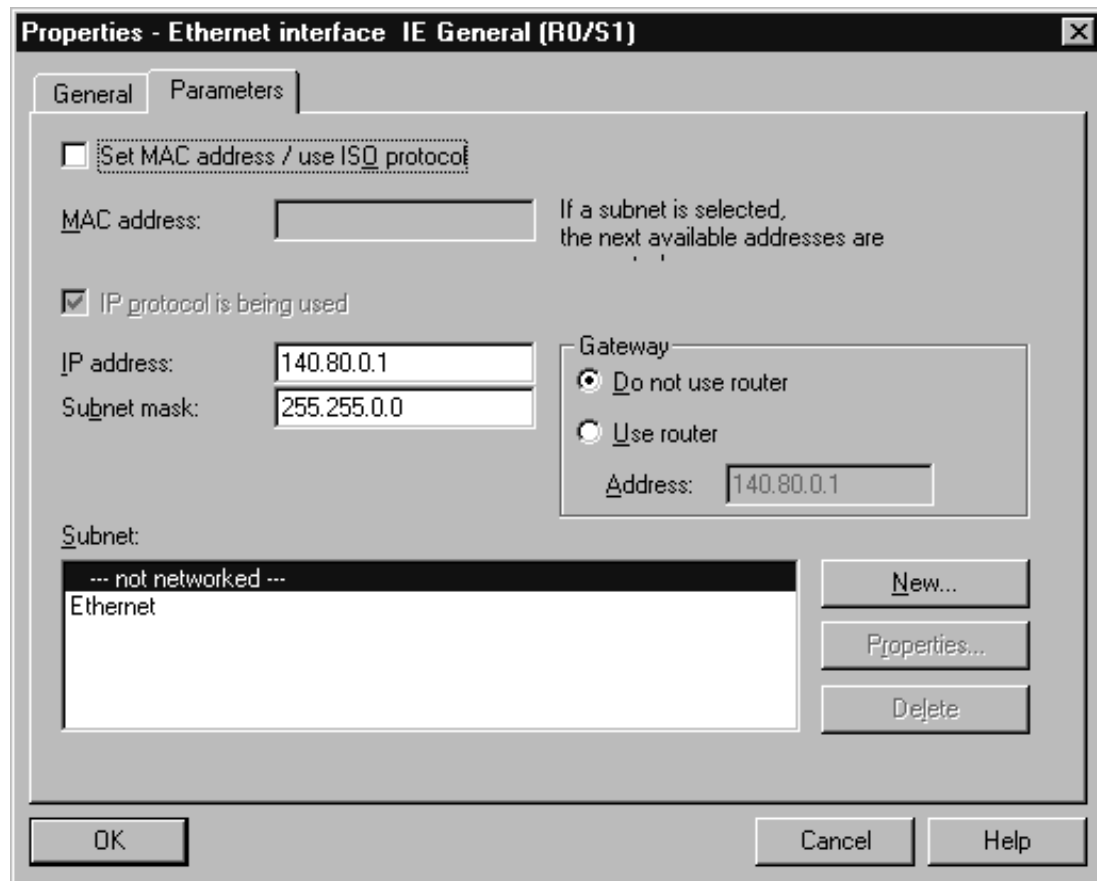


15. Expand both the **SIMATIC PC Station** menu and the **CP Industrial Ethernet** menu. Then, double-click on **IE General** or any other suitable option.

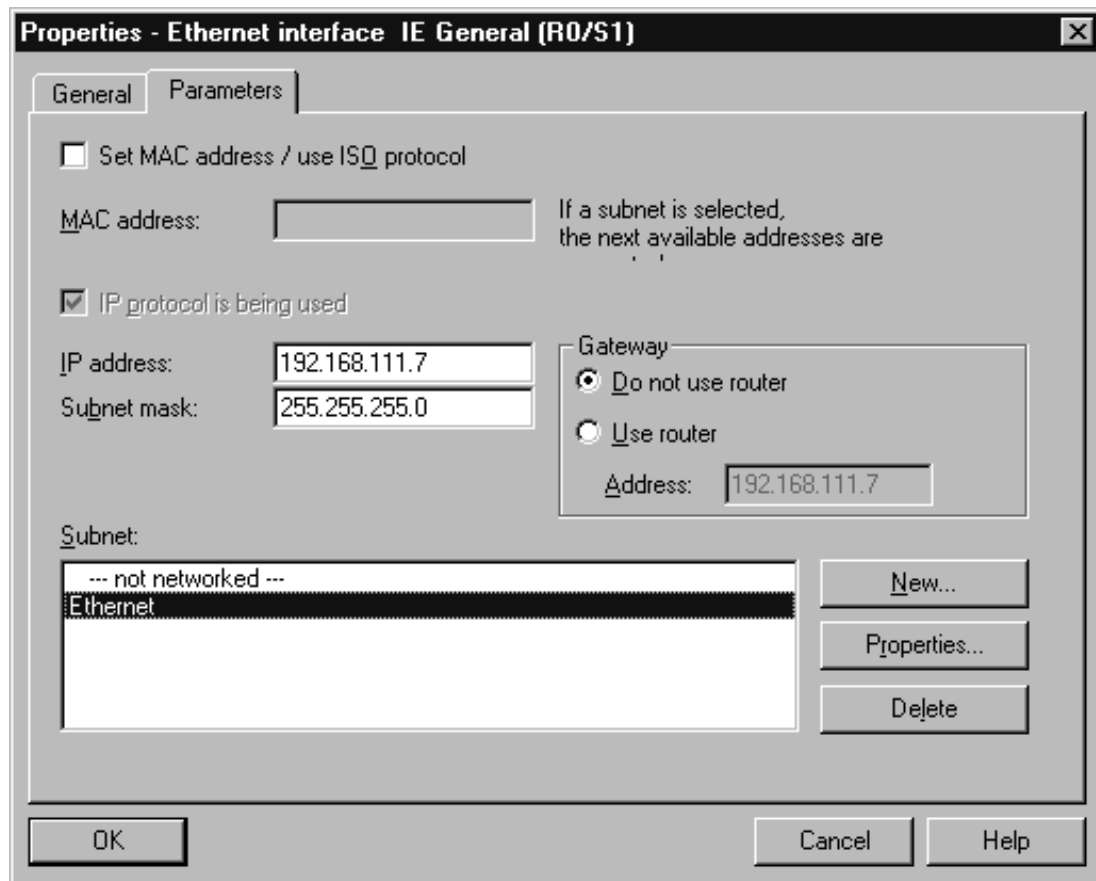


● **Note:** The window should appear as shown below.





16. Enter the IP address of the PC running the SIMATIC Manager software, in addition to the correct subnet mask.
17. Next, select **Ethernet** from the subnet box. Then, click **OK** to configure the PC station.



● **Note:** Once finished, open the **View** tab and then select **Catalog** to hide the catalog window.

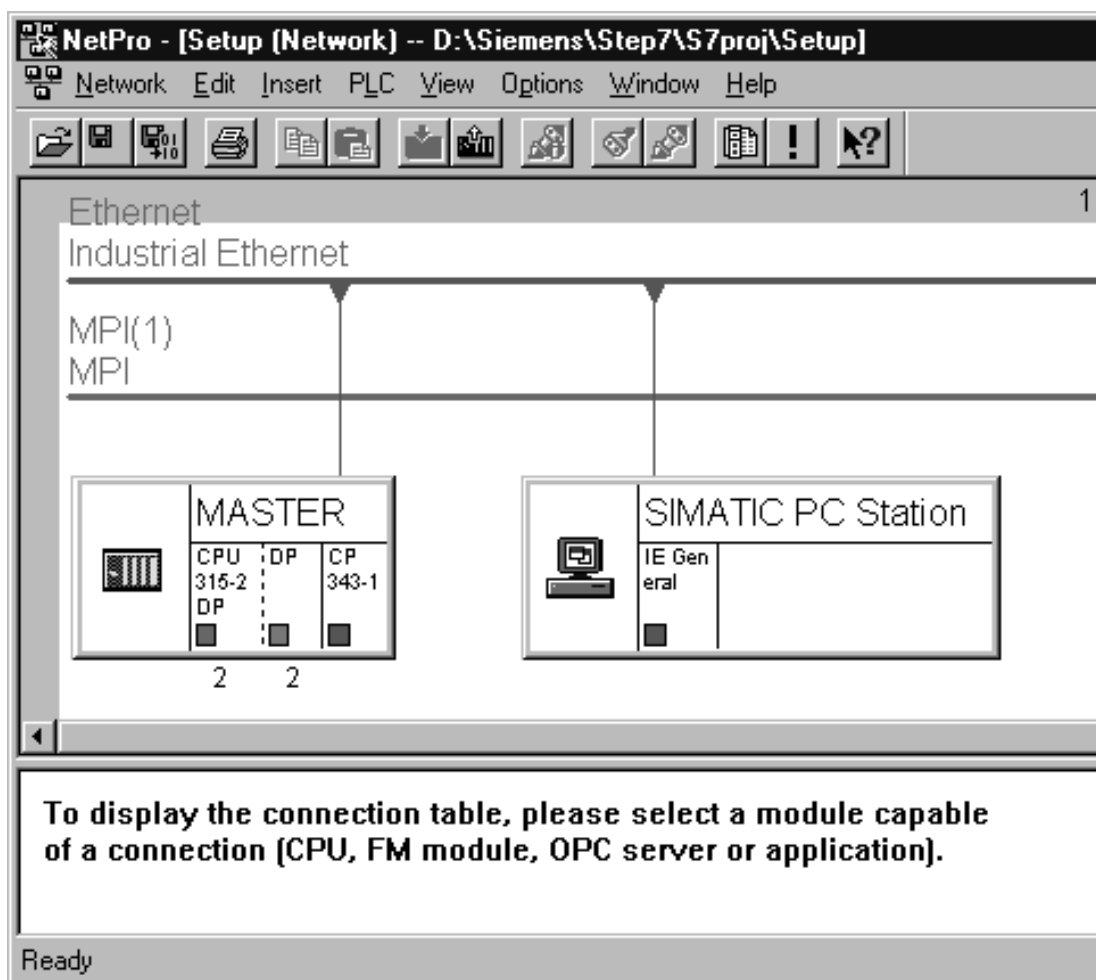
18. Save and exit the HW Configuration window.

● For more information, refer to [Step Three: Connecting the Master and the Slave Driver](#).

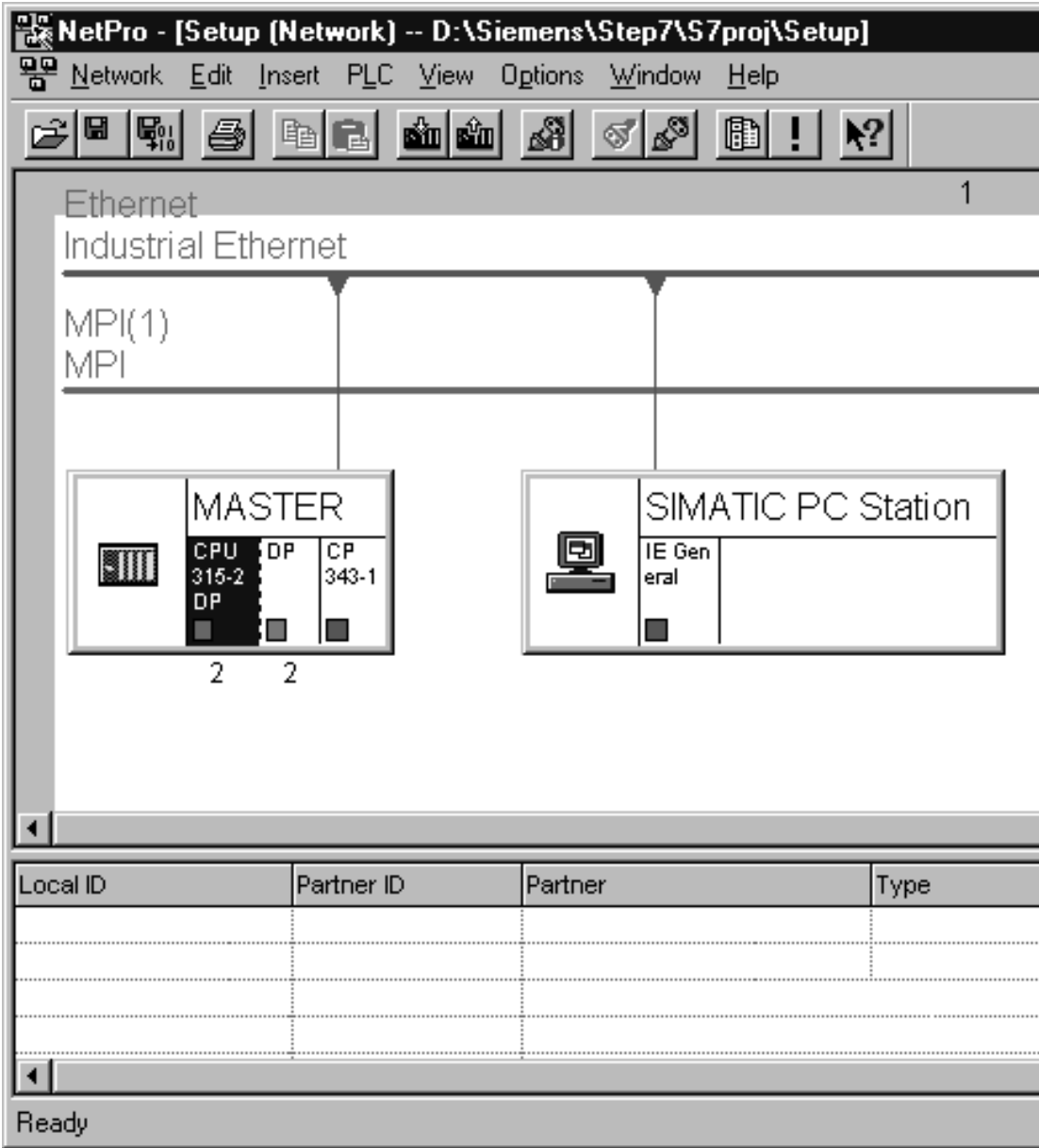
### Step Three: Connecting the Master and the Slave Driver

Once the Master and the PC Station have been successfully configured, the Master and the Slave Driver must be connected.

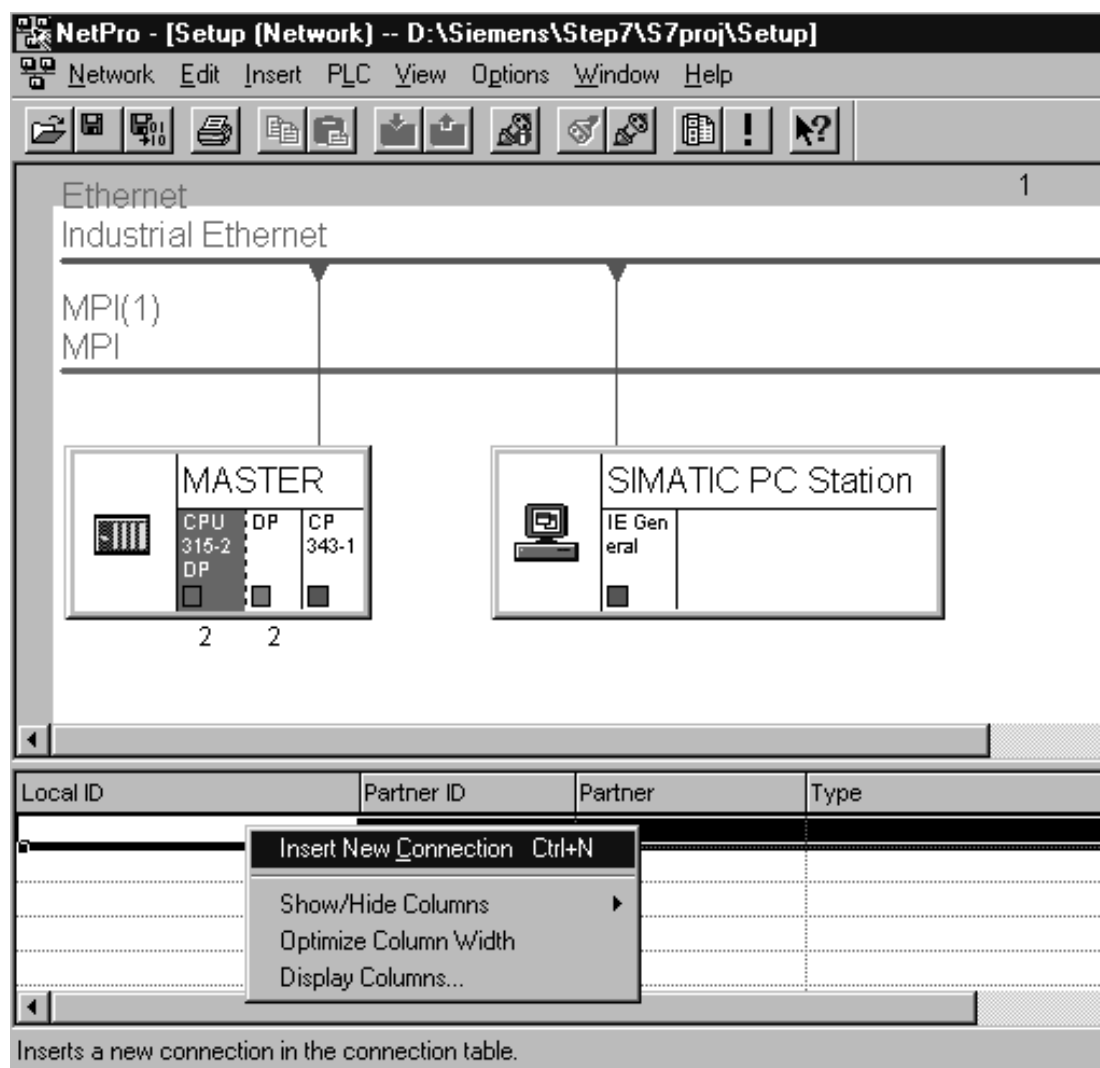
1. To start, open the **Options** tab in the SIMATIC Manager window and then select **Configure Network**.



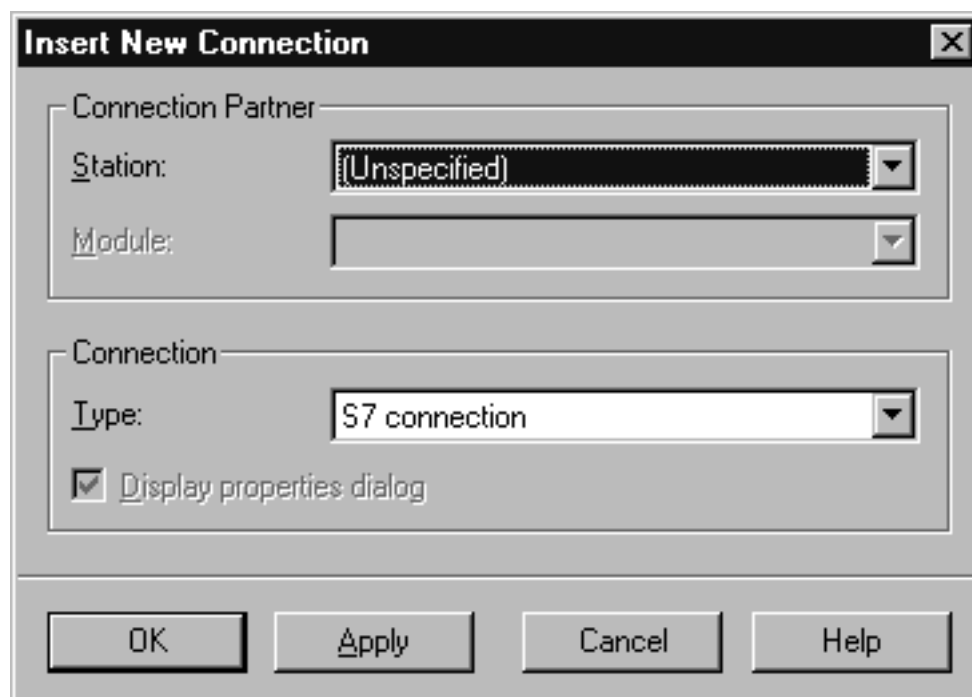
- Next, click on the Master's **CPU 315-2 DP** block. A series of rows should be displayed in the lower half of the window.



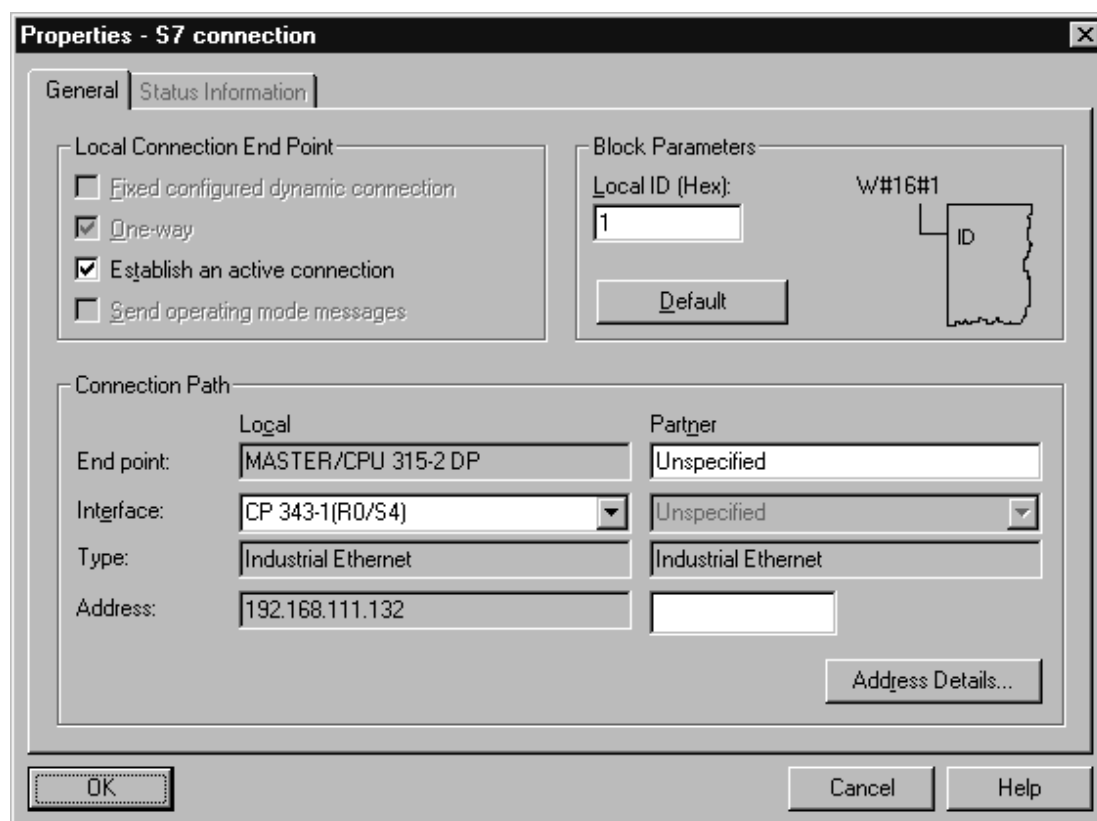
3. Right-click on the first row and select **Insert New Connection**.



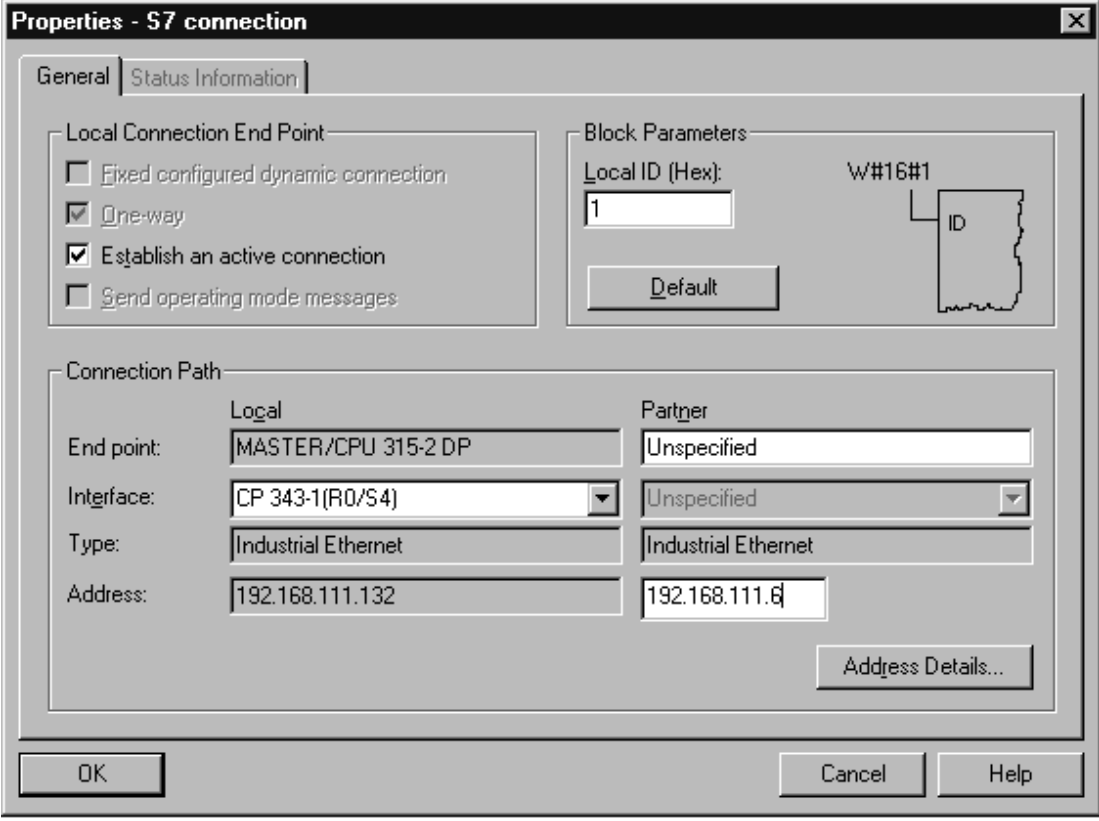
4. Then, click **OK**.



● **Note:** The window should appear as shown below.



5. Next, enter the IP address of the machine on which the Siemens TCP/IP Unsolicited Ethernet Driver runs.



**Properties - S7 connection**

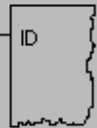
General | Status Information

**Local Connection End Point**

- ☐ Fixed configured dynamic connection
- ☒ One-way
- ☒ Establish an active connection
- ☐ Send operating mode messages

**Block Parameters**

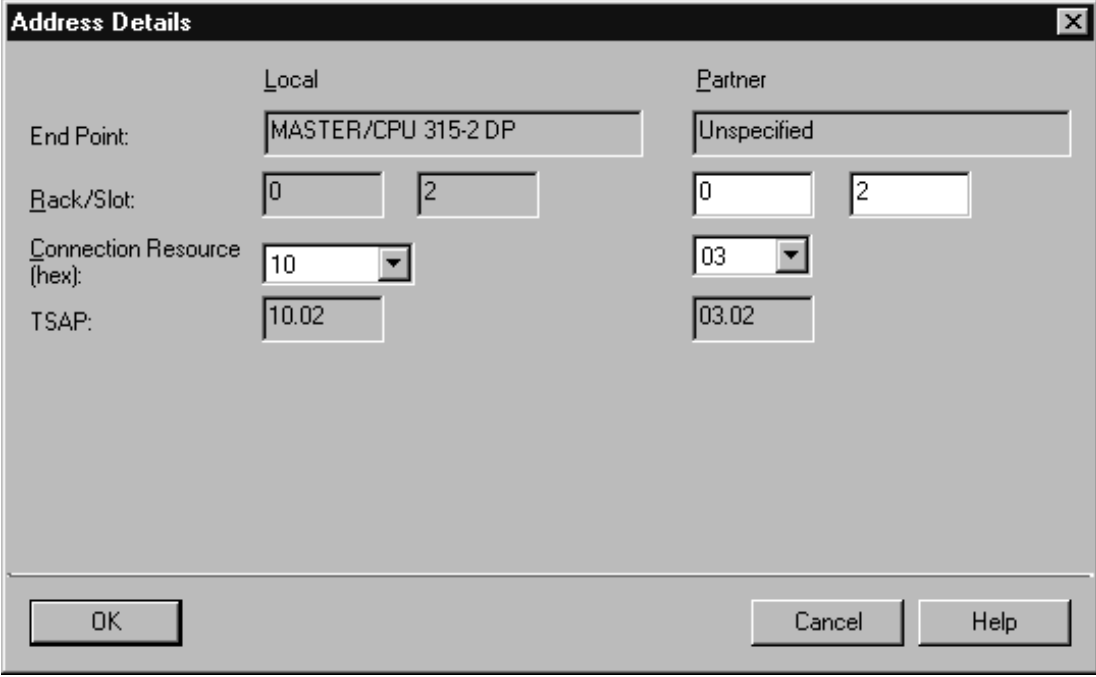
Local ID (Hex):  W#16#1



**Connection Path**

	Local	Partner
End point:	<input type="text" value="MASTER/CPU 315-2 DP"/>	<input type="text" value="Unspecified"/>
Interface:	<input type="text" value="CP 343-1(R0/S4)"/>	<input type="text" value="Unspecified"/>
Type:	<input type="text" value="Industrial Ethernet"/>	<input type="text" value="Industrial Ethernet"/>
Address:	<input type="text" value="192.168.111.132"/>	<input type="text" value="192.168.111.6"/>

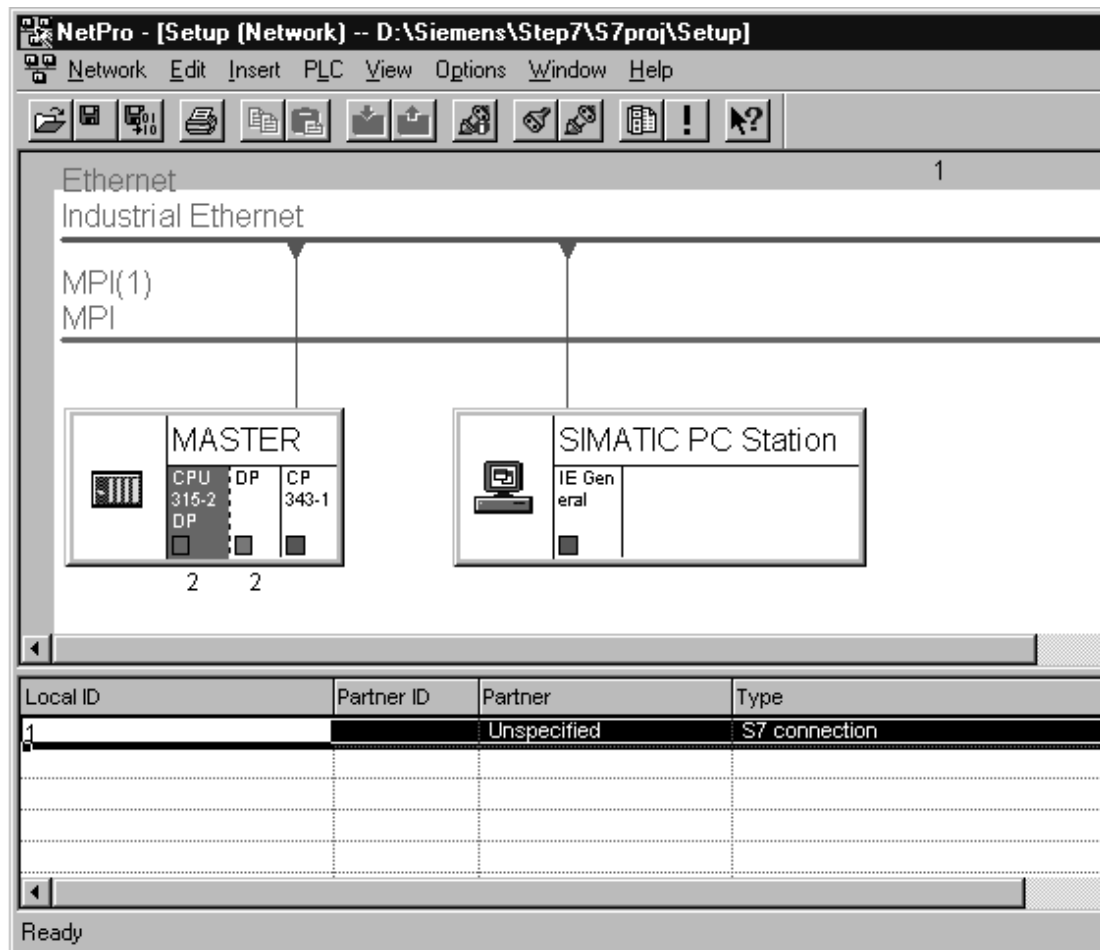
- Next, click **Address Details** and enter the rack/slot values of the device in the unsolicited driver with which the Master should communicate.



**Address Details**

	Local	Partner
End Point:	<input type="text" value="MASTER/CPU 315-2 DP"/>	<input type="text" value="Unspecified"/>
Rack/Slot:	<input type="text" value="0"/> <input type="text" value="2"/>	<input type="text" value="0"/> <input type="text" value="2"/>
Connection Resource (hex):	<input type="text" value="10"/>	<input type="text" value="03"/>
TSAP:	<input type="text" value="10.02"/>	<input type="text" value="03.02"/>

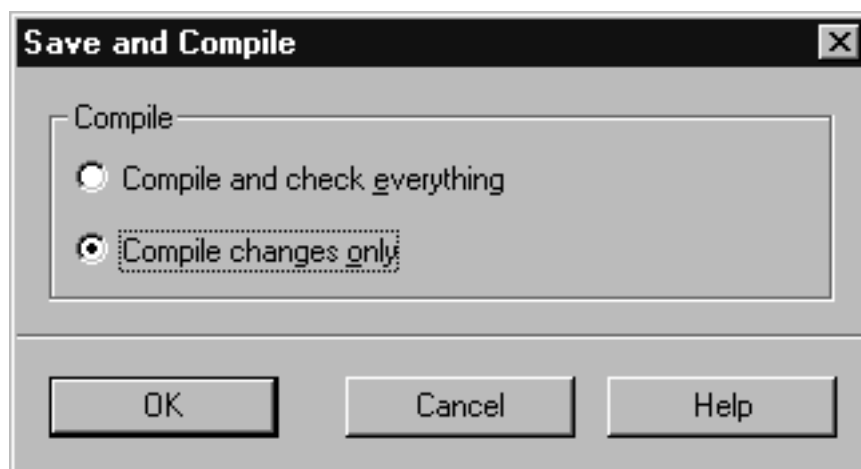
- Once finished, click **OK** twice to successfully connect the master and slave drivers. The master uses these settings to communicate with the destination device at rack 0 and slot 2.



● **Note:** The Local ID number (=1) identifies the connection between the two partners. This number is used later when creating function blocks for reading and writing data.

8. To finish, save and compile the data by opening the **Network** tab and selecting **Save and Compile**. Then, click **OK**.

● **Note:** There should be no errors on compilation.



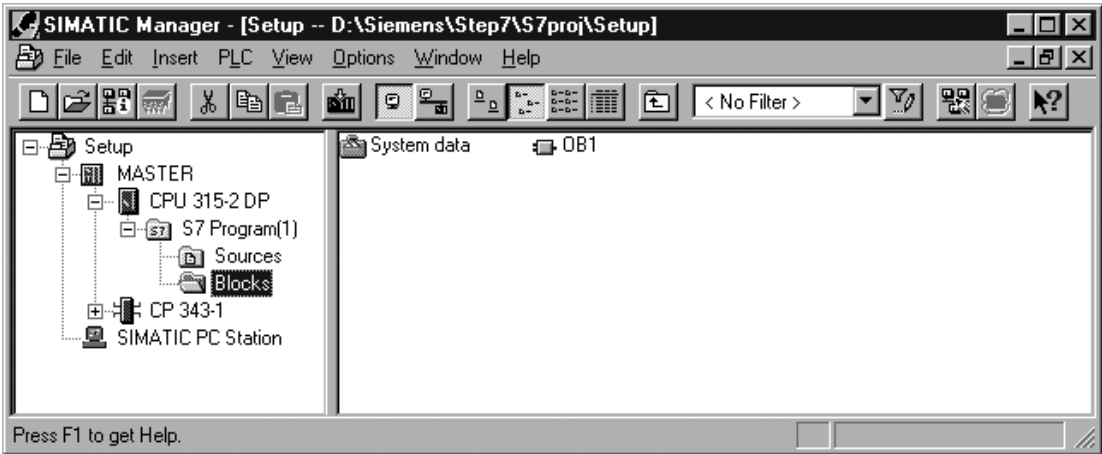
● For more information, refer to [Step Four: Inserting Function Blocks](#).



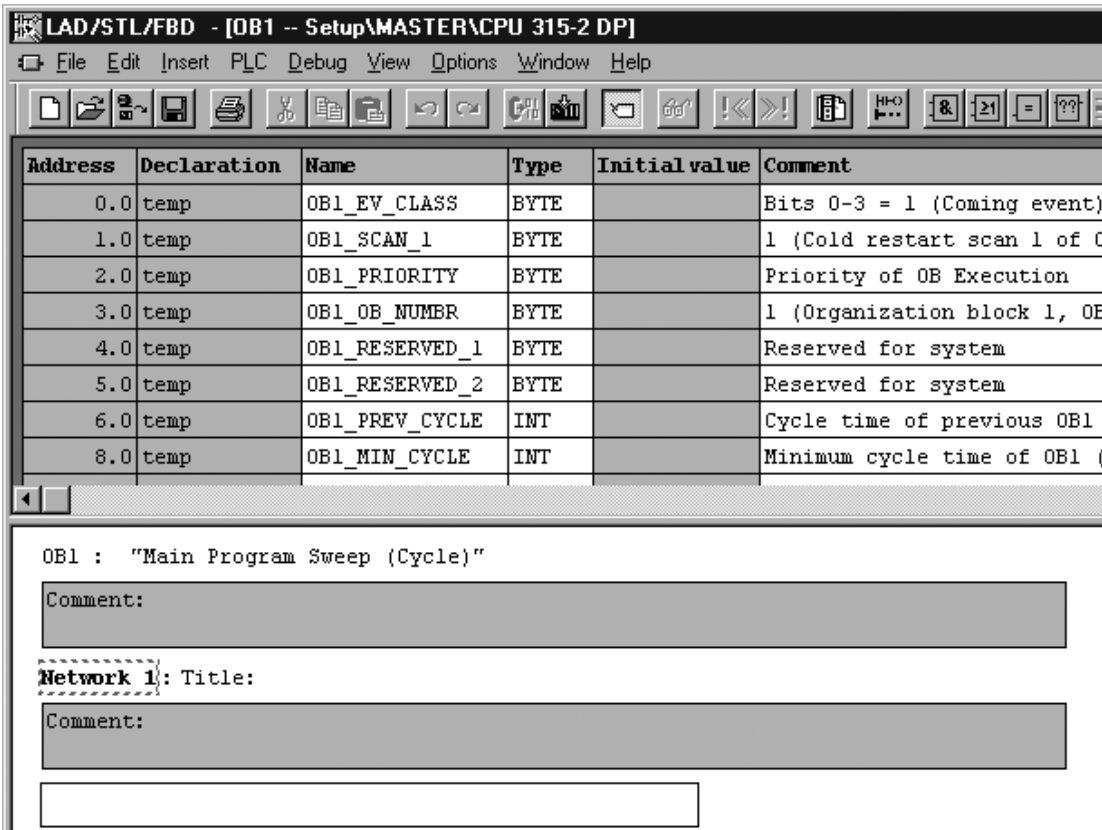
Step Four: Inserting Function Blocks

Once the master has been configured and connected with the unsolicited driver, it must also be prepared to generate requests for the unsolicited partner. This is done by creating function blocks, which can be used to read data from or write data to an unsolicited driver. The function block (FB) used for reading data in this example is FB14 (GET). The function block (FB) for writing data is FB15 (PUT).

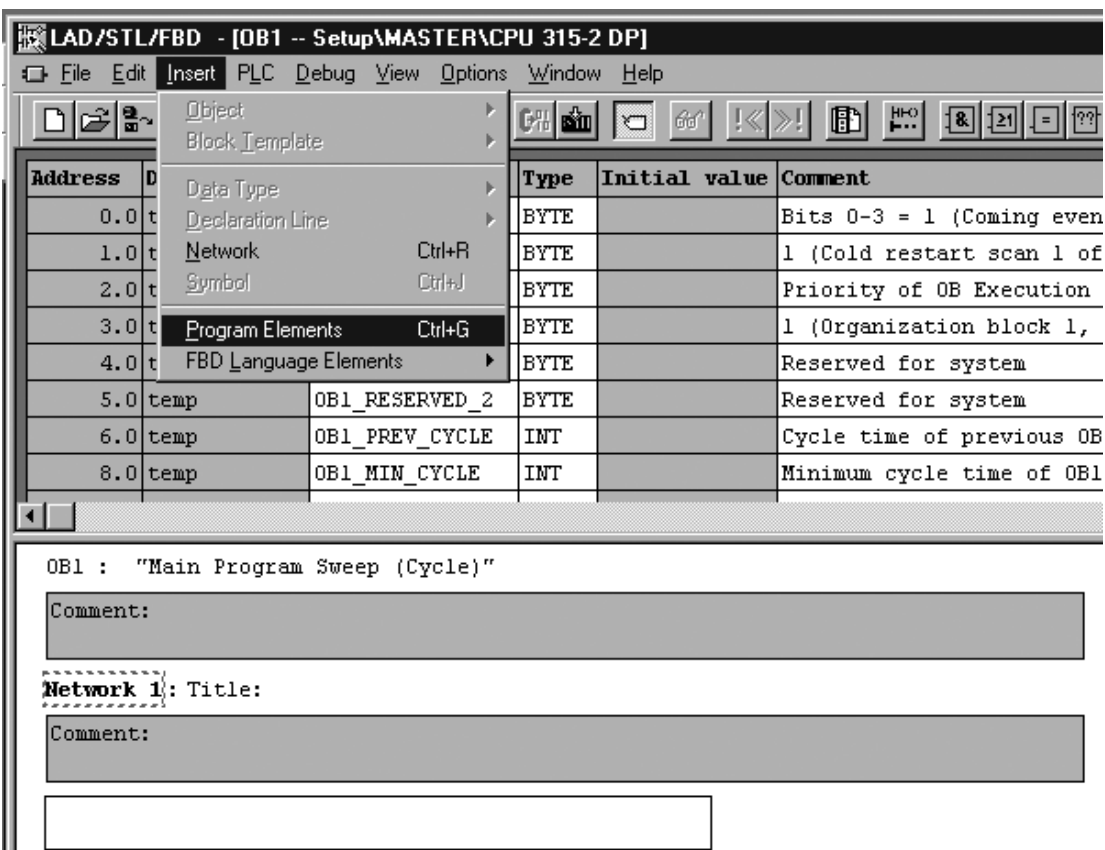
- 1. To start, expand the Master menu, the **CPU 315-2 DP** menu, and the **S7 Program[1]** menu.
- 2. Next, double-click on **Blocks** and **OB1**.



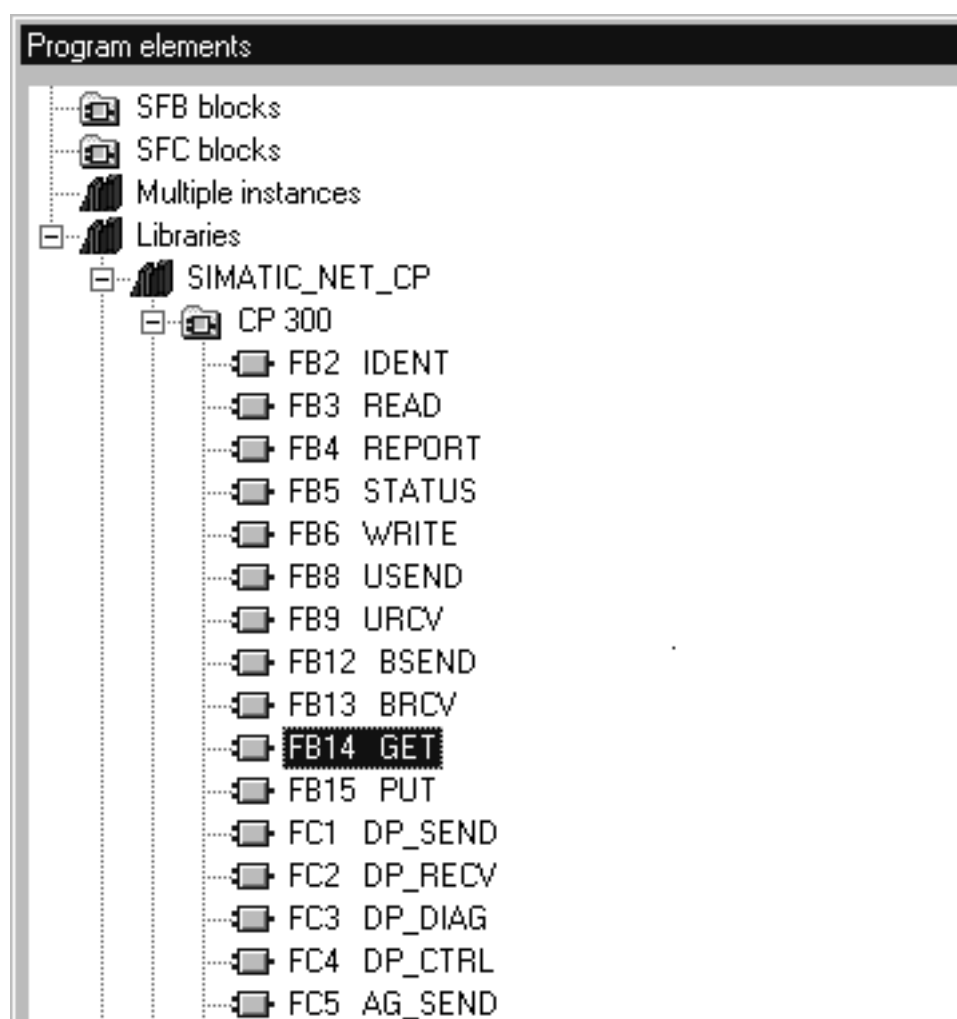
- 3. LAD, STL, or FBD can be used to create function blocks. In this example, FBD is used. In the LAD/STL/FBD window, click on the **Insert** menu.



4. Then, click **Program Elements**.



5. Next, expand the **Libraries**, **SIMATIC\_NET\_CP**, and **CP 300** menu. Then, double-click on **FB14 GET** to insert a function block to read data.



6. Close the **Program Elements** window. "FB14" should be inserted as shown below.

The screenshot shows the Siemens STEP 7 LAD/STL/FBD editor for OB1. The top window displays a table of variables:

Address	Declaration	Name	Type	Initial value	Comment
0.0	temp	OB1_EV_CLASS	BYTE		Bits 0-3 = 1 (Comi
1.0	temp	OB1_SCAN_1	BYTE		1 (Cold restart sc
2.0	temp	OB1_PRIORITY	BYTE		Priority of OB Exe
3.0	temp	OB1_OB_NUMBR	BYTE		1 (Organization bl
4.0	temp	OB1_RESERVED_1	BYTE		Reserved for syste
5.0	temp	OB1_RESERVED_2	BYTE		Reserved for syste
6.0	temp	OB1_PREV_CYCLE	INT		Cycle time of prev
8.0	temp	OB1_MIN_CYCLE	INT		Minimum cycle time

The main editor area shows the OB1 program titled "Main Program Sweep (Cycle)". It includes a comment field and a network titled "Network 1: Title:". Below the network, there is a function block diagram for the "GET" function block. The diagram shows the following connections:

- EN (Enable) connected to ...
- REQ (Request) connected to ...
- ID (Identify) connected to ...
- ADDR\_1 (Address 1) connected to ...
- RD\_1 (Read Data 1) connected to ...
- NDR (Not Done Read) output to ...
- ERROR (Error) output to ...
- STATUS (Status) output to ...
- ENO (Enable Not Output) output to ...

At the top of the function block diagram, there are three red question marks (???) indicating that the data block (DB) has not yet been assigned.

Below the diagram, the "Symbol information:" section shows the following details:

Symbol	Function	Description
FB14	GET	Read Data From a Remote CPU

7. Next, associate a data block (DB) with the function block (FB). To do so, click above the FB where there are three red question marks.

8. Enter the name of a data block. In this example, it is "DB2".

The screenshot shows the Siemens LAD/STL/FBD editor window titled "LAD/STL/FBD - [OB1 -- Setup\MASTER\CPU 315-2 DP]". The menu bar includes File, Edit, Insert, PLC, Debug, View, Options, Window, and Help. The toolbar contains various icons for editing and viewing. Below the toolbar is a table with the following data:

Address	Declaration	Name	Type	Initial value	Comment
0.0	temp	OB1_EV_CLASS	BYTE		Bits 0-3 = 1 (Coming event), Bits 4-7 = 1 (Even)
1.0	temp	OB1_SCAN_1	BYTE		1 (Cold restart scan 1 of OB 1), 3 (Scan 2-n o
2.0	temp	OB1_PRIORITY	BYTE		Priority of OB Execution
3.0	temp	OB1_OB_NUMBER	BYTE		1 (Organization block 1, OB1)
4.0	temp	OB1_RESERVED_1	BYTE		Reserved for system
5.0	temp	OB1_RESERVED_2	BYTE		Reserved for system
6.0	temp	OB1_PREV_CYCLE	INT		Cycle time of previous OB1 scan (milliseconds)
8.0	temp	OB1_MIN_CYCLE	INT		Minimum cycle time of OB1 (milliseconds)

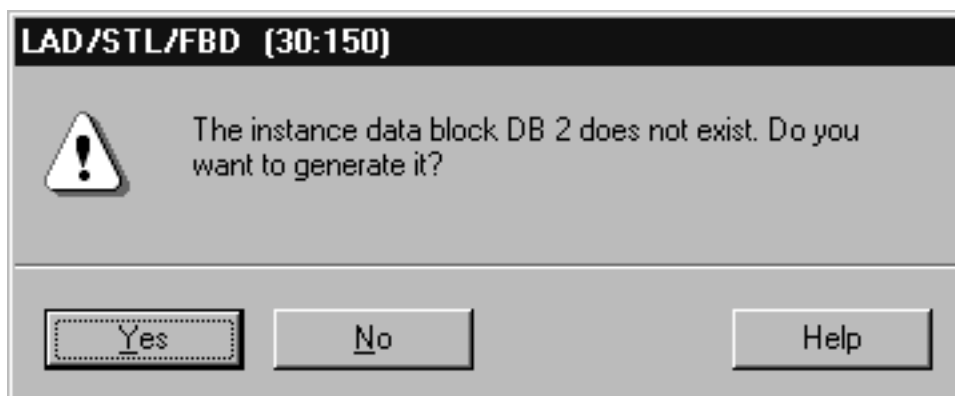
Below the table, the OB1 declaration is shown: "OB1 : 'Main Program Sweep (Cycle)'" with a comment field. Then, "Network 1" is defined with a title and comment field. The main network area shows a function block call for "DB2" using the "GET" instruction. The inputs and outputs are:

- Inputs: EN, REQ, ID, ADDR\_1, RD\_1
- Outputs: NDR, ERROR, STATUS, ENO

At the bottom, the "Symbol information" section shows:

Symbol	Function	Description
FB14	GET	Read Data From a Remote CPU

9. Next, click **Yes** to create the data block.



10. Next, fill in the other details as appropriate for the fields in the function block. Users should consider the following:

- "ADDR\_1" is the address on the destination device in the unsolicited driver.
- "RD\_1" is the address local to the PLC.
- The value at the remote address specified by "ADDR\_1" is written (GET) to the local address specified by "RD\_1".
- Enter the Local ID number that was generated when setting up the connection between the Master and the Slave Driver in the **ID** field. In this example, the Local ID number is 1.

☀ The number of bytes in both the "ADDR\_1" and "RD\_1" fields should be same for the unsolicited driver to respond correctly. Otherwise, an error occurs.

LAD/STL/FBD - [OB1 -- Setup\MASTER\CPU 315-2 DP]

File Edit Insert PLC Debug View Options Window Help

Address	Declaration	Name	Type	Initial value	Comment
0.0	temp	OB1_EV_CLASS	BYTE		Bits 0-3 = 1 (Coming event), Bits 4-7 = 1 (Event cl
1.0	temp	OB1_SCAN_1	BYTE		1 (Cold restart scan 1 of OB 1), 3 (Scan 2-n of OB
2.0	temp	OB1_PRIORITY	BYTE		Priority of OB Execution
3.0	temp	OB1_OB_NUMBR	BYTE		1 (Organization block 1, OB1)
4.0	temp	OB1_RESERVED_1	BYTE		Reserved for system
5.0	temp	OB1_RESERVED_2	BYTE		Reserved for system
6.0	temp	OB1_PREV_CYCLE	INT		Cycle time of previous OB1 scan (milliseconds)
8.0	temp	OB1_MIN_CYCLE	INT		Minimum cycle time of OB1 (milliseconds)
10.0	temp	OB1_MAX_CYCLE	INT		Maximum cycle time of OB1 (milliseconds)
12.0	temp	OB1_DATE_TIME	DATE AND TIME		Date and time OB1 started

OB1 : "Main Program Sweep (Cycle)"

Comment:

Network 1: Title:

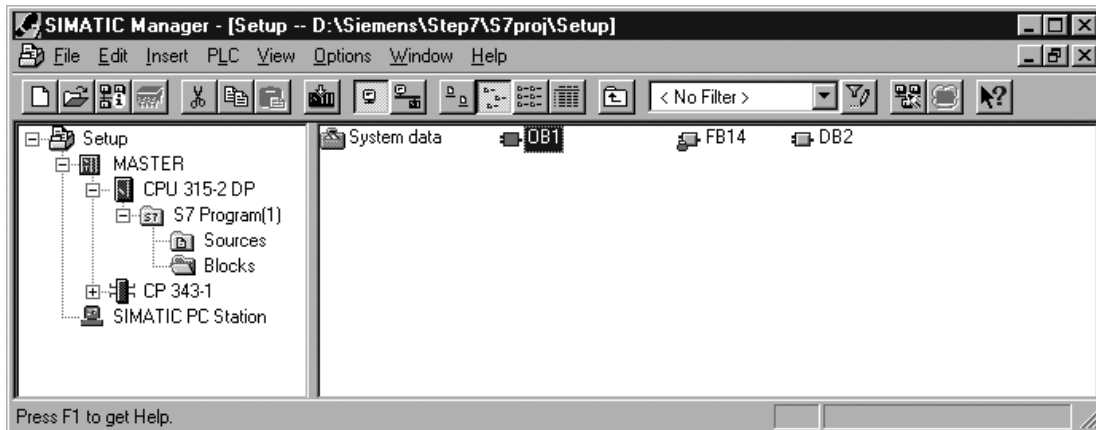
Comment:

Symbol information:

FB14	GET	Read Data From a Remote CPU
------	-----	-----------------------------

☀ **Note:** Now that the GET function block has been created successfully, users must remember that the block gets executed/triggered only on a rising edge (REQ).

11. To finish, click **Save** and then close the **LAD/STL/FBD** window.

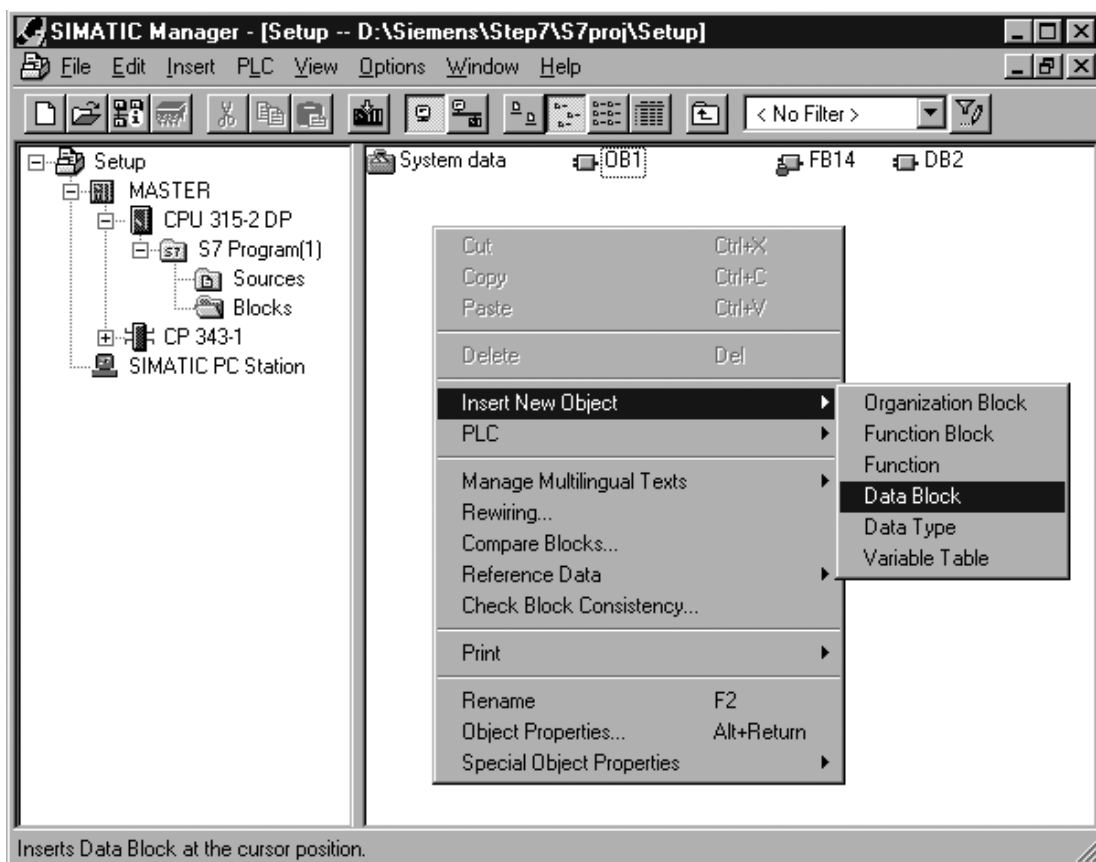


For more information, refer to [Step Five: Creating the DB3 Data Block](#).

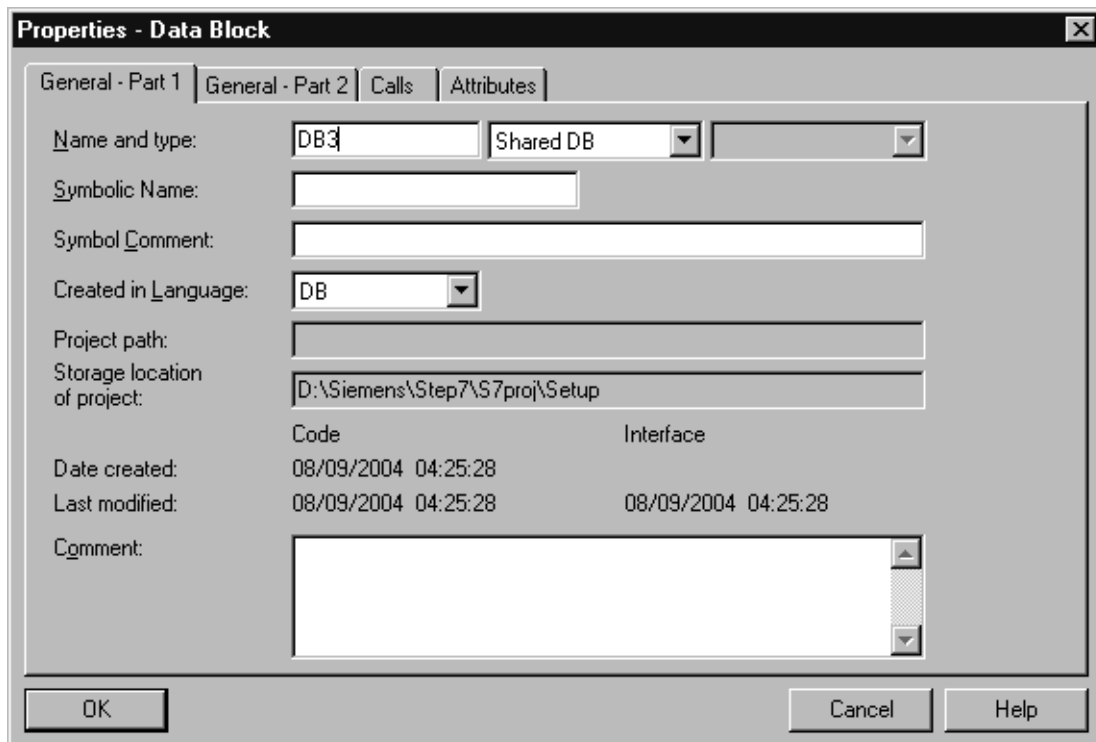
## Step Five: Creating the DB3 Data Block

While configuring GET FB, the data block "DB3" was used for the "RD\_1" field. This is the data block that will store read values.

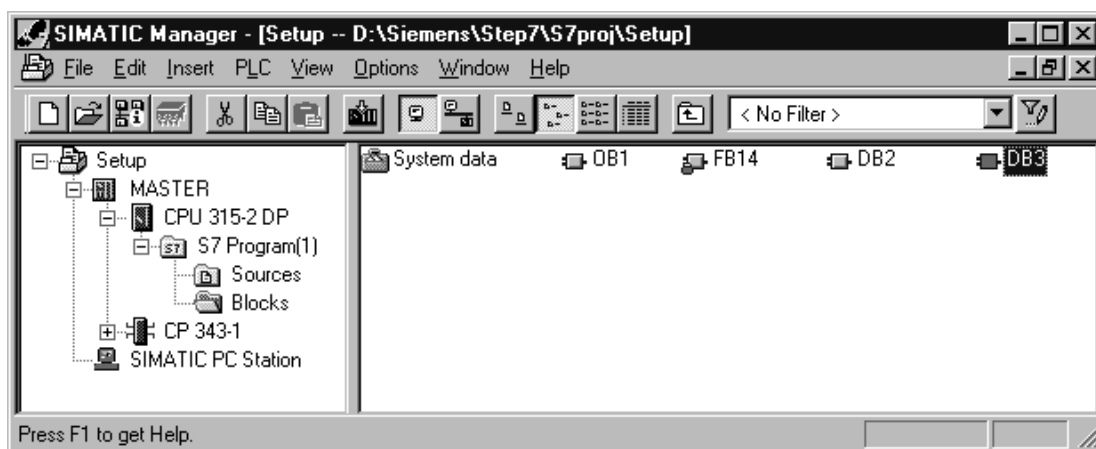
1. To start, right-click in the right pane of the SIMATIC Manager window and then select **Insert New Object | Data Block**.



2. Next, change the name to "DB3."

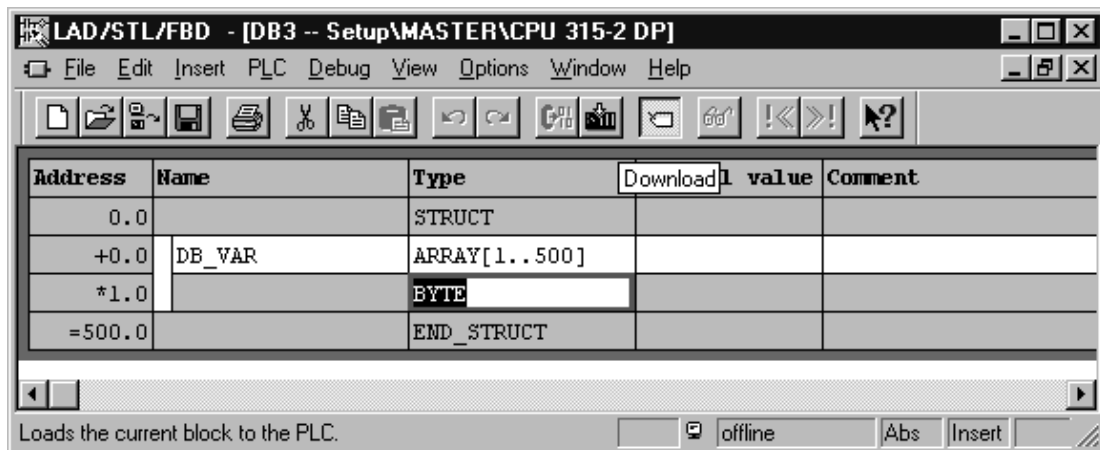


● **Note:** The window should appear as shown below.



3. Next, double-click on **DB3**. In order to assign some memory to the data block, users can make changes similar to those shown in the window below. Although the array size in this example was chosen arbitrarily, values should be specified to fit a particular need.



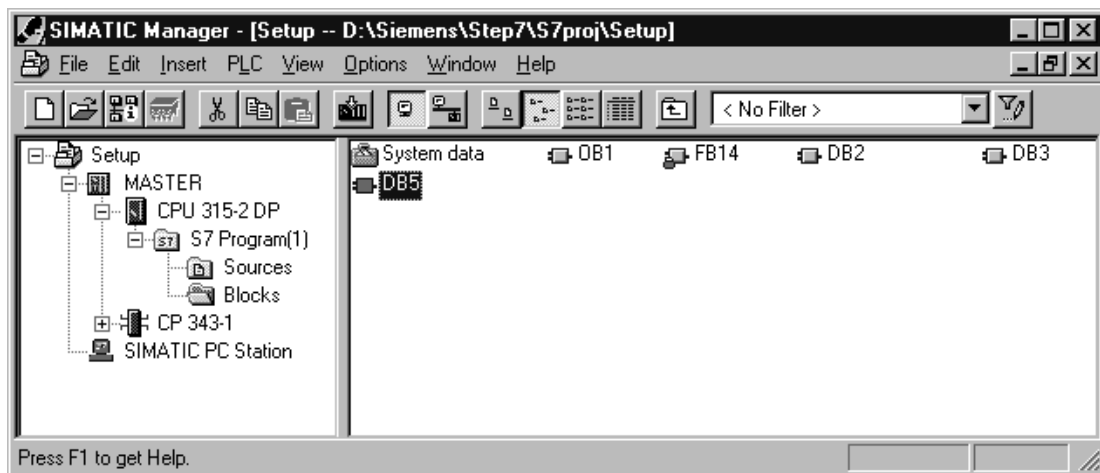


- Once finished, save and close the **LAD/STL/FBD** window.

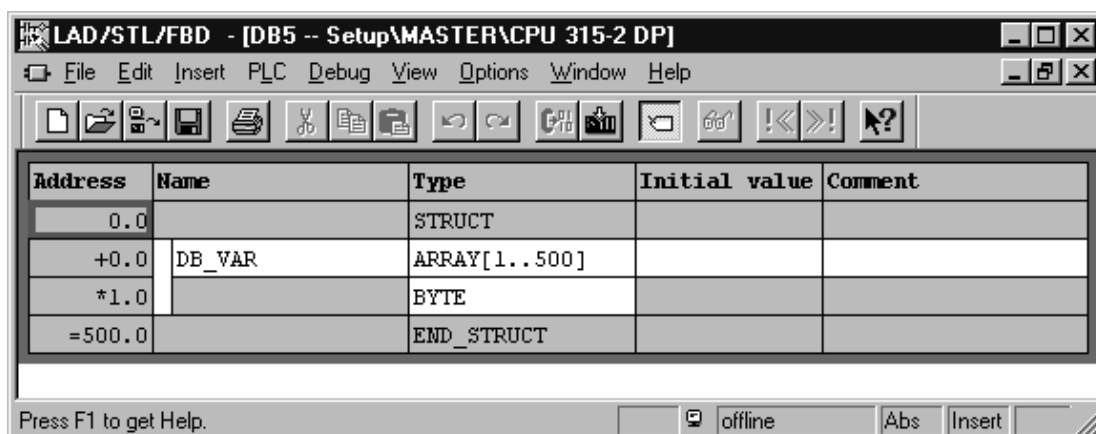
For more information, refer to [Step Six: Inserting PUT FB](#).

## Step Six: Inserting PUT FB

- Next, create a separate data block for the PUT FB, which holds the data that is written to the remote partner. To insert this new data block, follow the steps in [Step Five: Creating the DB3 Data Block](#) but name it "DB5."



- Double-click on **DB5**, and then specify a memory size. Although the array size in this example was chosen arbitrarily, the values should be specified to fit a particular need.



3. To insert the PUT FB, double-click on **OB1** in the SIMATIC Manager window. In **LAD/STL/FBD**, right-click in the blank space below **GET FB**.

- 4. Next, click **Insert Network** and then select the blank space below.

LAD/STL/FBD - [OB1 -- Setup\MASTER\CPU 315-2 DP]

File Edit Insert PLC Debug View Options Window Help

Address	Declaration	Name	Type	Initial value	Comment
0.0	temp	OB1_EV_CLASS	BYTE		Bits 0-3 = 1 (Coming event), Bits 4-7 =
1.0	temp	OB1_SCAN_1	BYTE		1 (Cold restart scan 1 of OB 1), 3 (Scan
2.0	temp	OB1_PRIORITY	BYTE		Priority of OB Execution
3.0	temp	OB1_OB_NUMBR	BYTE		1 (Organization block 1, OB1)
4.0	temp	OB1_RESERVED_1	BYTE		Reserved for system
5.0	temp	OB1_RESERVED_2	BYTE		Reserved for system
6.0	temp	OB1_PREV_CYCLE	INT		Cycle time of previous OB1 scan (millisec
8.0	temp	OB1_MIN_CYCLE	INT		Minimum cycle time of OB1 (milliseconds
10.0	temp	OB1_MAX_CYCLE	INT		Maximum cycle time of OB1 (milliseconds
12.0	temp	OB1_DATE_TIME	DATE_AND_TIME		Date and time OB1 started

DB2

GET

M0.0 — EN

M0.1 — REQ

W#16#1 — ID

P#I 0.0 BY — ADDR\_1

TE 10 —

P#DB3.DBX0 — RD\_1

.0 BYTE 10 —

NDR — M0.2

ERROR — M0.3

STATUS — MW1

ENO —

Symbol information:

FB14	GET	Read Data From a Remote CPU
------	-----	-----------------------------

Network 2: Title:

Comment:

- 5. Next, click **Insert | Program Elements**. Then, expand the **Libraries, SIMATIC\_NET\_CP**, and **CP 300** menus.
- 6. To insert a function block to write data, double-click on **FB15 PUT**. Then, close the **Program Elements** window.

LAD/STL/FBD - [OB1 -- Setup\MASTER\CPU 315-2 DP]

File Edit Insert PLC Debug View Options Window Help

Address	Declaration	Name	Type	Initial value	Comment
0.0	temp	OB1_EV_CLASS	BYTE		Bits 0-3 = 1 (Coming event), Bits 4-7 =
1.0	temp	OB1_SCAN_1	BYTE		1 (Cold restart scan 1 of OB 1), 3 (Scan
2.0	temp	OB1_PRIORITY	BYTE		Priority of OB Execution
3.0	temp	OB1_OB_NUMBR	BYTE		1 (Organization block 1, OB1)
4.0	temp	OB1_RESERVED_1	BYTE		Reserved for system
5.0	temp	OB1_RESERVED_2	BYTE		Reserved for system
6.0	temp	OB1_PREV_CYCLE	INT		Cycle time of previous OB1 scan (millis
8.0	temp	OB1_MIN_CYCLE	INT		Minimum cycle time of OB1 (milliseconds
10.0	temp	OB1_MAX_CYCLE	INT		Maximum cycle time of OB1 (milliseconds
12.0	temp	OB1_DATE_TIME	DATE_AND_TIME		Date and time OB1 started

.0 BYTE 10 — RD\_1 — ENO —

**Symbol information:**

Symbol	Function	Description
FB14	GET	Read Data From a Remote CPU

**Network 2:** Title:

Comment:

- Next, associate a data block (DB) with the function block (FB). To do so, click above the FB where there are three red question marks. Then, specify a name. In this example, "DB4" is used.

**Note:** A window prompt requests confirmation of data block creation. Click **Yes**.

- Fill in the other details as appropriate. Users should consider the following:

- "ADDR\_1" address is on the destination device in the unsolicited driver.
- "SD\_1" is the address local to the PLC.
- The value at the local address specified by "SD\_1" is written (PUT) to the remote address specified by "ADDR\_1".
- Enter the Local ID number that was generated when setting up the connection between the Master and the Slave Driver in the **ID** field. In this example, the Local ID number is 1.

**Important:** The number of bytes in both the "ADDR\_1" and "SD\_1" fields should be same in order for the unsolicited driver to respond correctly. Otherwise, an error occurs.

**LAD/STL/FBD - [OB1 -- Setup\MASTER\CPU 315-2 DP]**

File Edit Insert PLC Debug View Options Window Help

Address	Declaration	Name	Type	Initial value	Comment
0.0	temp	OB1_EV_CLASS	BYTE		Bits 0-3 = 1 (Coming event), Bits 4-7 = 1
1.0	temp	OB1_SCAN_1	BYTE		1 (Cold restart scan 1 of OB 1), 3 (Scan 2
2.0	temp	OB1_PRIORITY	BYTE		Priority of OB Execution
3.0	temp	OB1_OB_NUMBER	BYTE		1 (Organization block 1, OB1)
4.0	temp	OB1_RESERVED_1	BYTE		Reserved for system
5.0	temp	OB1_RESERVED_2	BYTE		Reserved for system
6.0	temp	OB1_PREV_CYCLE	INT		Cycle time of previous OB1 scan (milliseconds)
8.0	temp	OB1_MIN_CYCLE	INT		Minimum cycle time of OB1 (milliseconds)
10.0	temp	OB1_MAX_CYCLE	INT		Maximum cycle time of OB1 (milliseconds)
12.0	temp	OB1_DATE_TIME	DATE_AND_TIME		Date and time OB1 started

Comment:

DB4

"PUT"

M0.0 — EN

M0.1 — REQ

W#16#1 — ID

P#Q 0.0 BY — ADDR\_1

TE 10 — SD\_1

P#DB5.DBX0 — SD\_1

DONE — M0.4

ERROR — M0.5

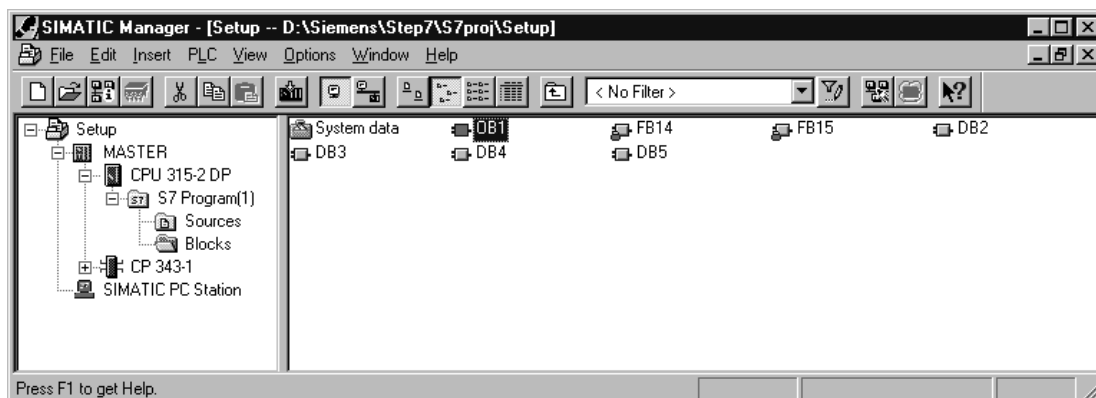
STATUS — MW2

ENO —

**Symbol information:**

FB15	PUT	Write Data to a Remote CPU
------	-----	----------------------------

9. To finish, click **Save** and then close **LAD/STL/FBD**.



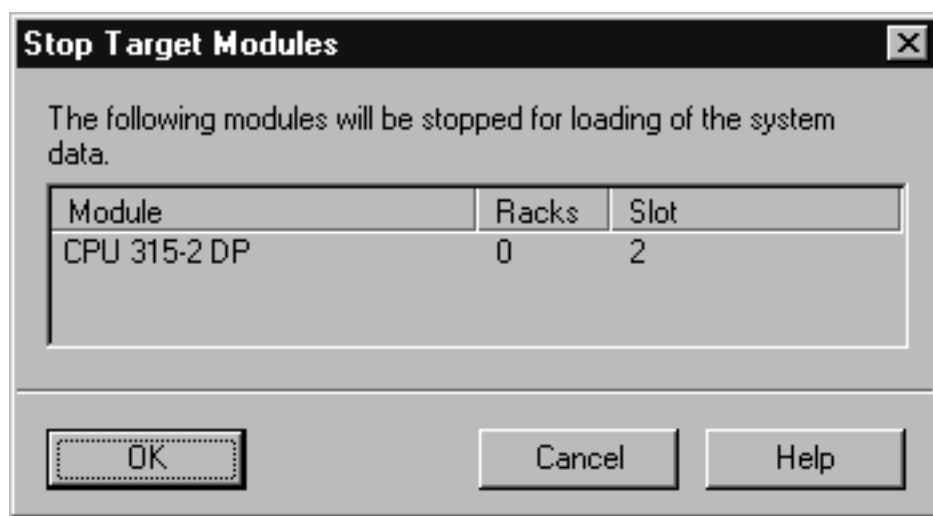
For more information, refer to [Step Seven: Downloading to the PLC](#).

## Step Seven: Downloading to the PLC

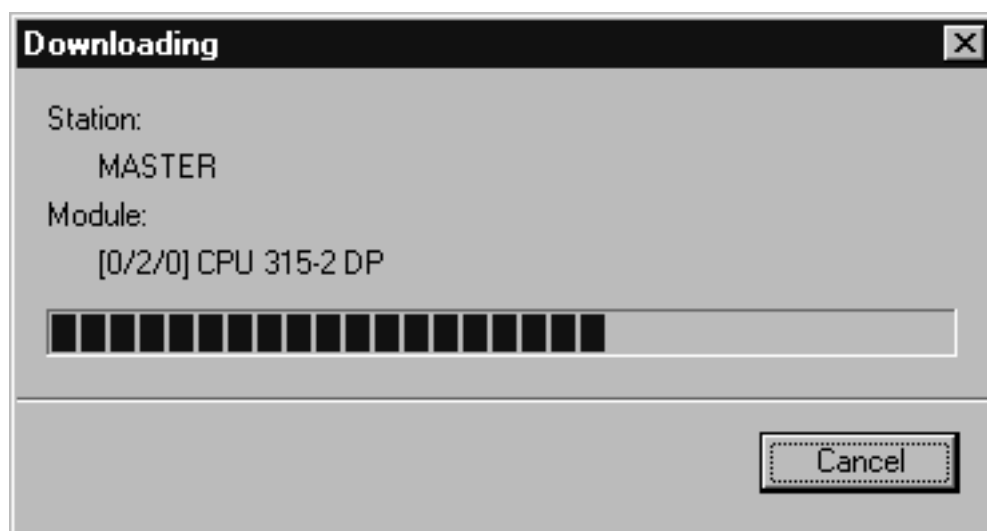
Once the Master has been prepared to generate Read/Write requests for the remote unsolicited partner, the information must be downloaded to the PLC.

1. To start, click **Master** in the left pane of the SIMATIC Manager window. Then, select the **PLC** menu.

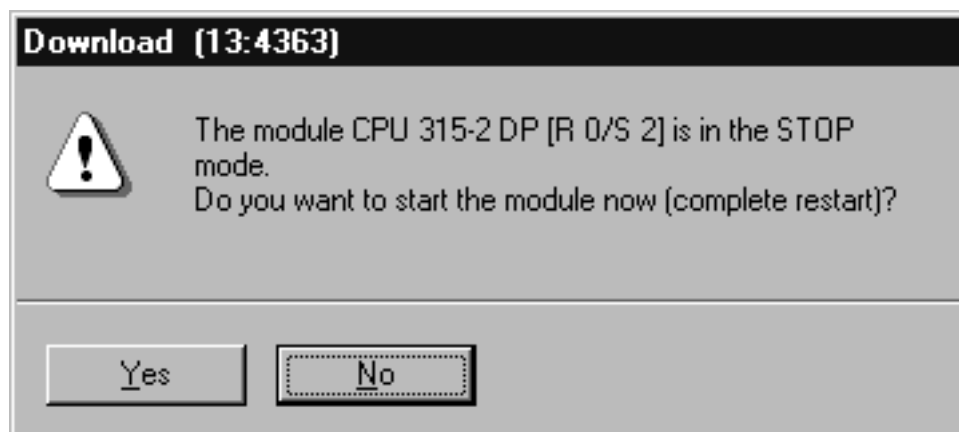
- Next, select **Download** to begin downloading the project to the PLC.



- Click **OK**.

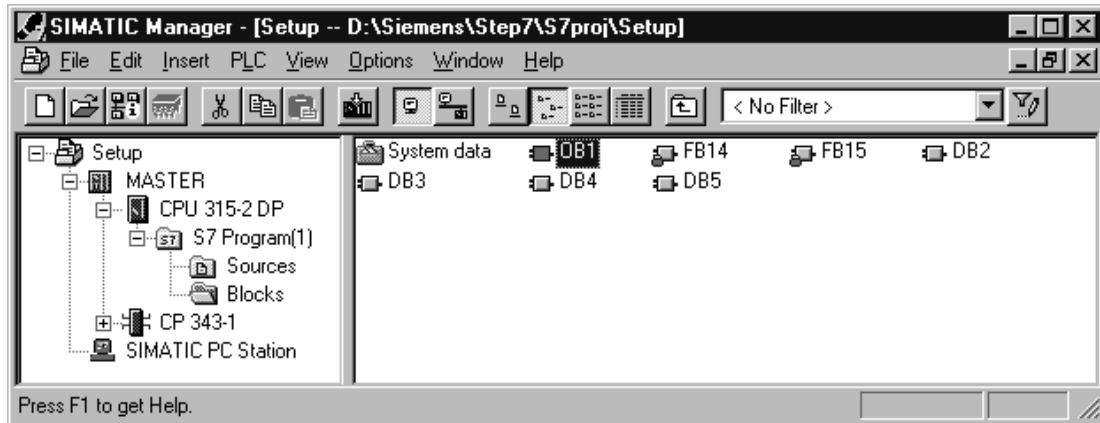


- Then, click **Yes**.

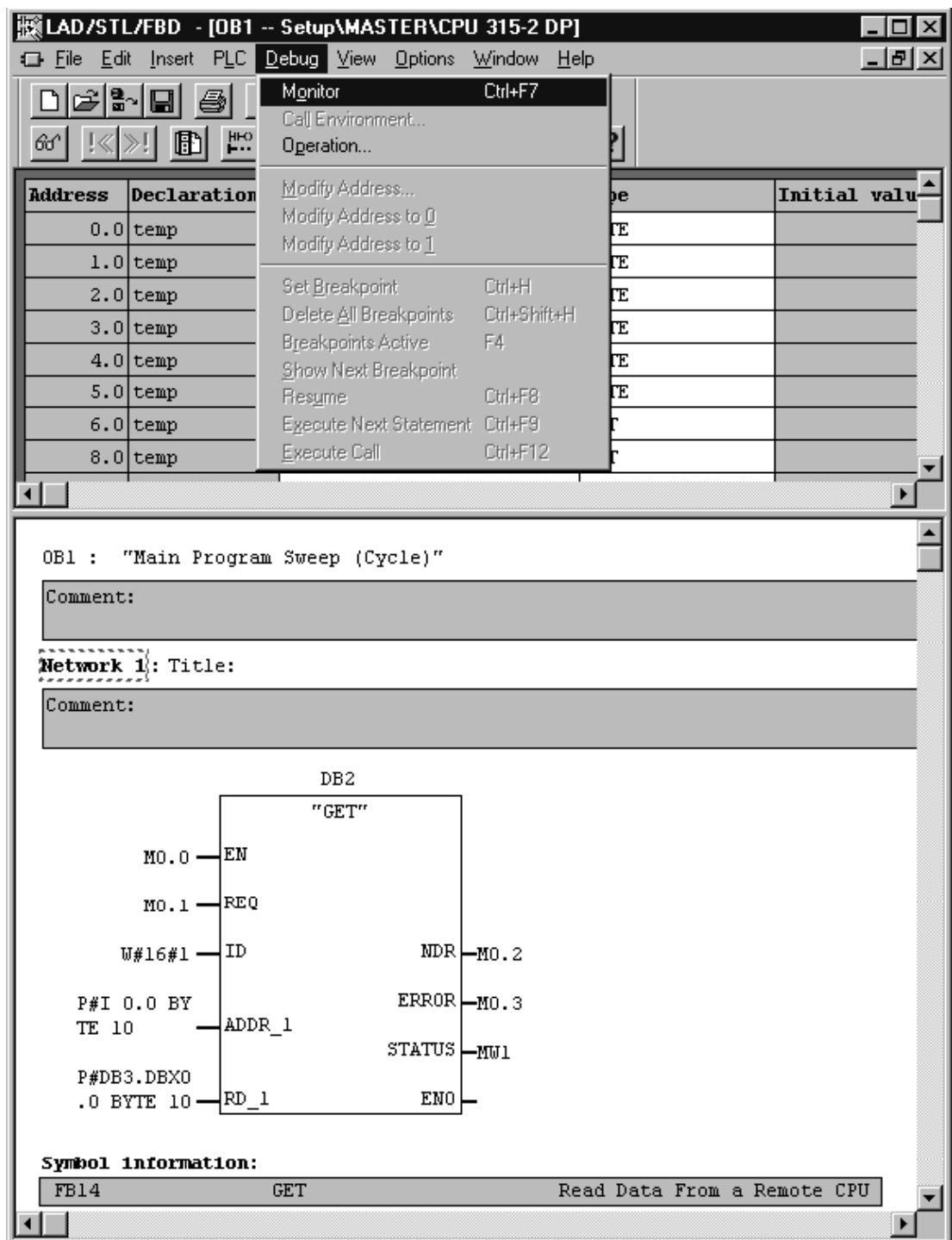


● **Note:** The master must be run in order to trigger the function blocks that generate Read/Write requests.

5. Double-click on **OB1** in the SIMATIC Manager window.



6. In **LAD/STL/FBD**, click **Debug | Monitor**.



● **Note:** LAD/STL/FBD should appear in Online Mode as shown below.



LAD/STL/FBD - [@OB1 -- Setup\MASTER\CPU 315-2 DP] ONLINE

File Edit Insert PLC Debug View Options Window Help

Address	Declaration	Name	Type	Initial value
0.0	temp	OB1_EV_CLASS	BYTE	
1.0	temp	OB1_SCAN_1	BYTE	
2.0	temp	OB1_PRIORITY	BYTE	
3.0	temp	OB1_OB_NUMBR	BYTE	
4.0	temp	OB1_RESERVED_1	BYTE	
5.0	temp	OB1_RESERVED_2	BYTE	
6.0	temp	OB1_PREV_CYCLE	INT	
8.0	temp	OB1_MIN_CYCLE	INT	

OB1 : "Main Program Sweep (Cycle)"

Comment:

Network 1: Title:

Comment:

DB2

"GET"

MO.0 1 EN

MO.1 1 REQ

16#0001 W#16#1 ID

P#I 0.0 BY

TE 10 ADDR\_1

P#DB3.DBX0

.0 BYTE 10 RD\_1

NDR 0 MO.2

ERROR 0 MO.3

STATUS 16#0000 MW1

ENO

Symbol information:

FB14 GET Read Data From a Remote CPU

Press F1 to get Help. RUN Abs

7. To execute **GET/PUT FBs**, change the **REQ** value to 0 and then 1 to indicate the rising edge. To do so, right-click on the **REQ** field and select **Modify to 0** to force a zero to the field.

**LAD/STL/FBD - [@OB1 -- Setup\MASTER\CPU 315-2 DP] ONLINE**

File Edit Insert PLC Debug View Options Window Help

Address	Declaration	Name	Type	Initial value
0.0	temp	OB1_EV_CLASS	BYTE	
1.0	temp	OB1_SCAN_1	BYTE	
2.0	temp	OB1_PRIORITY	BYTE	
3.0	temp	OB1_OB_NUMBR	BYTE	
4.0	temp	OB1_RESERVED_1	BYTE	
5.0	temp	OB1_RESERVED_2	BYTE	
6.0	temp	OB1_PREV_CYCLE	INT	
8.0	temp	OB1_MIN_CYCLE	INT	

OB1 : "Main Program Sweep (Cycle)"

Comment:

**Network 1:** Title:

Comment:

DB2  
"GET"

MO.0 1 EN  
MO.1 1 REQ

16#00  
W#16

P#I 0.0  
TE 10

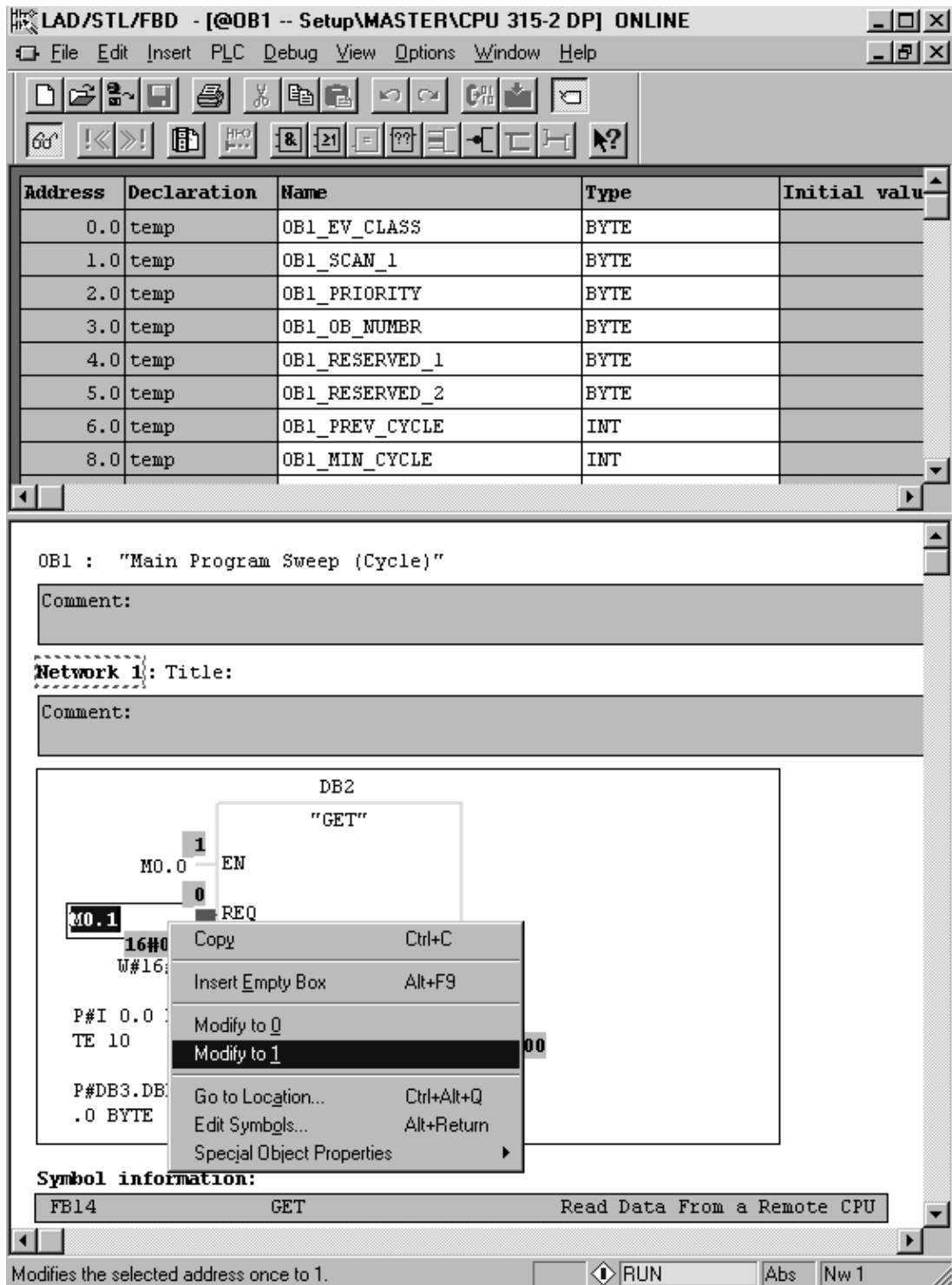
P#DB3.DB  
.0 BYTE

Symbol information

FB14 GET Read Data From a Remote CPU

The statements are not being processed. RUN Abs Nw 1

- Next, right-click on the **REQ** field and select **Modify to 1** to force a value of one to the field.



● **Note:** Both of the FBs must next be configured to respond to the same rising edge in order for the SIMATIC Manager's variables to be locally monitored and modified.

9. In **LAD/STL/FBD**, click on **PLC** and then select **Monitor/Modify Variables**.
10. Enter the variables to be monitored. To view the changes made to this window, execute the function blocks.

● **Note:** Remember that the slot/rack value of the remote device with which the master is

communicating is "rack:0 slot:2". The values can be changed from the NetPro window. Users must make sure that the unsolicited driver on the other end has a device with these values and is running.

# Index

## A

Address Descriptions 12  
Advanced Channel Properties 6  
Arrays 15

## B

BCD 11, 15  
Boolean 11

## C

Channel Assignment 8  
Channel Properties - Ethernet Communications 5  
Channel Properties - General 4  
Channel Properties - Write Optimizations 5  
Communications Properties 7  
CPU Settings 9  
CPU Slot 10

## D

Data Block Boolean 13  
Data Collection 8  
Data Types Description 11  
Description 8  
Device Properties - General 7  
Diagnostics 5  
Discrete Inputs 12  
Discrete Outputs 12  
Do Not Scan, Demand Poll Only 9  
Driver 4, 8  
Duty Cycle 6  
DWord 11

**E**

Ethernet 4

Event Log Messages 17

Examples 16

**F**

Failure to start unsolicited communications. | Port number = <number>. 17

Float 11, 15

**H**

Help Contents 3

**I**

ID 8

IEEE-754 floating point 6

Initial Updates from Cache 9

Internal Memory 13

ISO 8073 Class 0 4

**L**

LBCD 11

Libraries 4

Long 11

**M**

Master Device Configuration 10

Model 8

**N**

Name 7

Network Adapter 5

Non-Normalized Float Handling 6

## O

Optimization Method 5

Overview 3

## P

Port Number 7

Protocols 4

## R

Rack Number 10

Request All Data at Scan Rate 9

Request Data No Faster than Scan Rate 9

Respect Client-Specified Scan Rate 9

Respect Tag-Specified Scan Rate 9

RFC1006 4

## S

S5 Counter 15

S5 Timer 15

Scan Mode 9

Setup 4

Short 11

Siemens S7-300 3

Signed Word 14

SIMATIC Manager 18

Simulated 8

Slave 3

Step Five: Creating the DB3 Data Block 47

Step Four: Inserting Function Blocks 41

Step One: Creating a New Project 18

Step Seven: Downloading to the PLC 53

Step Six: Inserting PUT FB 49

Step Three: Connecting the Master and the Slave Driver 34

Step Two: Configuring the Master and PC Station 22

String 11, 15

Supported Commands 4

## **U**

Unsigned Word 14

## **W**

Word 11

Write All Values for All Tags 5

Write Only Latest Value for All Tags 6

Write Only Latest Value for Non-Boolean Tags 6

Write Optimizations 5