

# MODBUS TCP

## EM-MODBUS-GATEWAY-IFS with Phoenix Contact controllers



Application note  
106033\_en\_00

© PHOENIX CONTACT 2014-04-09

## 1 Description

This application note describes how to implement and start up the EM-MODBUS-GATEWAY-IFS on controllers from Phoenix Contact based on an example of project creation with PC Worx.

## 2 System requirements

### 2.1 Software requirements

In order to use Modbus TCP, you require the AUTOMATIONWORX Software Suite Version 1.70, Service Pack 1, or later and the IFS-CONF SUITE-INTERFACE.

### 2.2 Hardware requirements for PC Worx

Please refer to the PC Worx quick start guide UM QS EN PC WORX for hardware requirements.



Make sure you always use the latest documentation. It can be downloaded at [phoenixcontact.net/products](http://phoenixcontact.net/products).



This application note is valid for all products listed on the following page.

### 3 Ordering data

#### Products

Description	Type	Order No.	Pcs. / Pkt.
Inline controller with Ethernet interface for coupling to other controllers and systems, with programming options according to IEC 61131-3, complete with connector and labeling field.	ILC 131 ETH	2700973	1
Inline controller with Ethernet interface for coupling to other controllers and systems, with programming options according to IEC 61131-3, complete with connector and labeling field.	ILC 151 ETH	2700974	1
Inline controller with Ethernet interface for coupling to other controllers and systems, with programming options according to IEC 61131-3, complete with connector and labeling field.	ILC 171 ETH	2700975	1
Inline controller with Ethernet interface for coupling to other controllers and systems, with programming options according to IEC 61131-3, complete with connector and labeling field.	ILC 191 ETH 2TX	2700976	1
Inline controller with Ethernet interface for coupling to other controllers and systems, with programming options according to IEC 61131-3, complete with connector and labeling field.	ILC 131 ETH/XC	2701034	1
Inline controller with Ethernet interface for coupling to other controllers and systems, with programming options according to IEC 61131-3, complete with connector and labeling field.	ILC 151 ETH/XC	2701141	1

#### Accessories

Description	Type	Order No.	Pcs. / Pkt.
The EM-MBUS-GATEWAY-IFS Modbus/TCP module connects up to 32 motor management modules (EMM-...IFS) or other INTERFACE system devices together via the TBUS DIN rail connector.	EM-MODBUS-GATEWAY-IFS	2901528	1
Inline connector set for Inline bus coupler with connected I/Os	IL BKDIO-PLSET	2878599	1
Connecting cable for connecting the Inline controller to a PC (RS-232 cable)	PRG CAB MINI DIN	2730611	1
Program and configuration memory, plug-in, 256 Mbyte	SD FLASH 256MB	2988120	1
Program and configuration memory, plug-in, 256 Mbyte with license key for function block libraries, e.g., for: SNMP, SQL, wireless, motion functions, etc.	SD FLASH 256MB APPLIC A	2988816	1
Program and configuration memory, plug-in, 2 Gbyte	SD FLASH 2 GB	2988162	1
Program and configuration memory, plug-in, 2 Gbyte with license key for function block libraries, e.g., for: SNMP, SQL, wireless, motion functions, etc.	SD FLASH 2GB APPLIC A	2701190	1
QUINT POWER power supply units	See latest Phoenix Contact INTERFACE catalog		

#### Documentation

Description	Type	Order No.	Pcs. / Pkt.
"Installing and operating the ILC 131 ETH, ILC 151 ETH, ILC 171 ETH, ILC 191 ETH 2TX, ILC 131 ETH/XC and ILC 151 ETH/XC Inline controllers" user manual	UM EN ILC 1X1	-	1
"PC Worx" quick start guide	UM QS EN PC WORX	-	1
"EM-...-GATEWAY-IFS Quick Start" application note	AH EN EM-xxx-GATEWAY-IFS	-	1

## 4 Modbus TCP

Modbus is a communication protocol used to exchange process data between a client and a server in an Ethernet network. There are three different operating modes for data transmission: Modbus ASCII, Modbus RTU and Modbus TCP.

In Modbus TCP mode, the TCP protocol (Transmission Control Protocol) is used for data transmission. The Modbus protocol data to be transmitted is embedded in the TCP protocol. A TCP/IP connection must be established between the client and the server prior to data transmission. In general, the connection is established automatically. The established TCP/IP connection between client and server remains permanently active during cyclic communication. For acyclic communication, however, the TCP/IP connection can be disconnected once the data has been transmitted and then reestablished if there is a communication request. By default, the TCP port 502 reserved for Modbus is used for communication.

The client initiates communication between the client and the server. The client sends a request in the form of a command code (and data, if required) to the server. After successful receipt of the request, the server sends a corresponding response to the client, including the requested data and status information or an error message. The data may contain bit or word information.

The internal data organization (memory addresses, etc.) varies depending on the device and manufacturer. Please refer to the documentation of the corresponding device for more information.

Modbus provides various commands for read and write access to digital inputs and outputs and to registers for client/server communication.

The following table shows the supported Modbus function codes:


Modbus function codes			
Code No.	Function code	Description	Method
FC1	Read Coils	Read several internal bits or digital outputs	Bit-by-bit/word-by-word
FC2	Read Discrete Inputs	Read several digital inputs	Bit-by-bit/word-by-word
FC3	Read Holding Register	Read several internal registers or output registers	Word-by-word
FC4	Read Input Register	Read several input registers	Word-by-word
FC15	Write Multiple Coils	Write several internal bits or digital outputs	Bit-by-bit/word-by-word
FC16	Write Multiple Registers	Write several internal registers or output registers	Word-by-word
FC23	Read/Write Multiple Register	Read and write several internal registers or output registers simultaneously	Word-by-word

## 5 Example of a project with Modbus

The following project consists of the ILC 171 ETH (controller) and the EM-MODBUS-GATEWAY-IFS.

### 5.1 Sequence for creating the Modbus project

The complete sequence for creating the Modbus project in PC Worx is shown in Figure 1.



For more detailed information on creating a project, please refer to the UM QS EN PC WORX quick start guide or the PC Worx online help.

When implementing the project, most of the tasks can be performed offline (without a connection to the Modbus system).

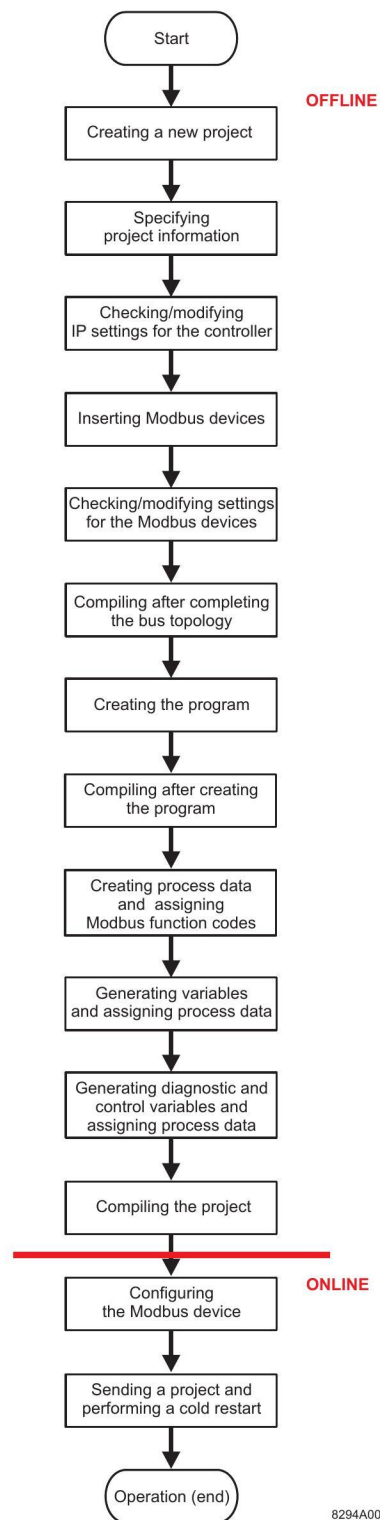


Figure 1 Sequence for creating the Modbus project



The sequence described here for creating a project applies to projects based on a cyclic communication between the controller and the Modbus device. Section 6 “Acyclic communication using the MB\_ASYNC\_RW function block” provides detailed information on acyclic communication on a general basis.

**5.2 Creating a new project**

- Select the “New Project...” command from the “File” menu.
- Select the controller and confirm with “OK”.
- Select the “File, Save Project As / Zip Project As...” command.
- Enter a unique and meaningful project name and save the project.

**5.3 Specifying project information**

- Switch to the bus configuration workspace.
- Adapt the project information to your project.

**5.4 Checking/modifying IP settings for the controller**

The IP settings for the controller are made when the project is created.

Adapt these settings, if necessary.

- Switch to the bus configuration workspace.
- Select the controller node.
- In the “Device Details” window, switch to the “IP Settings” tab.
- Check the IP settings and modify them, if necessary.
- Assign an IP address, if it has still not been assigned. For detailed information on assigning the IP address, please refer to the UM QS EN PC WORX quick start guide.



The IP address that is assigned here for the controller is also implemented as the IP address for the communication path via TCP/IP.

**5.5 Inserting a Modbus device**

- Make sure you are in the bus configuration workspace. Insert the module as Generic Modbus Device below the MODBUS\_CLT node.
- If the device catalog is hidden, show it by selecting the “View, Device Catalog” menu.
- Open the “Phoenix Contact, Generic, Device” device catalog.

- Select the “Generic Modbus Device”.

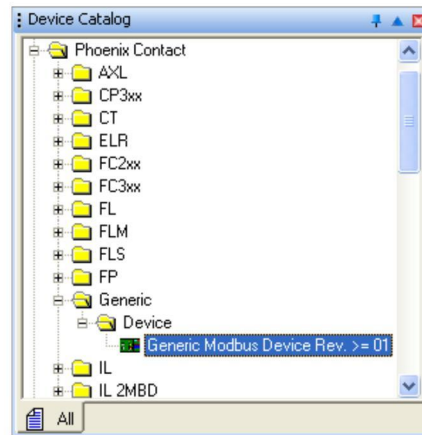


Figure 2 Selecting the Modbus device

- Hold down the left mouse button and move the Modbus device in the “Bus structure” window to the right of the MODBUS\_CLT icon until the “Insert in the lower level” icon appears.
- Move all other Modbus devices to below the preceding Modbus device until the “Insert at the same level” icon appears.

Figure 3 shows the bus configuration with the inserted Modbus device.

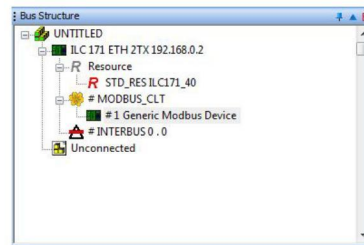


Figure 3 Generic Modbus Device inserted



The “Blind out device” or “Deactivate Bus” options can be selected via the context menu of the device. To make the device visible again or to activate the bus, re-select the above-mentioned settings from the context menu.

### 5.6 Modifying the settings for the Modbus device

After having added devices to the bus configuration, default values are set for each Modbus device. The settings can be modified via the “Modbus-Settings” tab.

- Make sure you are in the bus configuration workspace.
- In the “Bus structure” window, select the Generic Modbus Device.
- Under “Device Details”, select the “Modbus-Settings” tab.
- Modify the Modbus settings depending on your requirements.

Name	Value
Vendor	Phoenix Contact
Designation	Generic Modbus Device
Device ID	0x0002
Functional description	
Device type	Device
Device family	Generic
Order number	
Revision: HW / Master FW (/COP FW)	01
Station Name	Generic_Modbus_16
Device Name	
Module Equipment ID	
MAC Address	00-A0-45-42-DB-21
IP Address	192.168.0.3
Subnetmask	255.255.255.0
Default Gateway	
Port	502
Protocol	TCP
Swap Bytes	No
Consecutive Number	1
Connection timeout / UDP timeout	5000 ms
Reconnection interval	60000 ms
Trigger rate	500 ms
Node ID	16

Figure 4 Modbus settings of the EM-MODBUS-GATEWAY-IFS

The Modbus settings comprise:

#### Station name

This name is the unique identification for the Modbus device in the network. It must be known to the Modbus device before it can be used in the network.

#### MAC address

The MAC address is used to provide worldwide unique identification for each network device. Enter the MAC address of the Modbus device. It is printed on the respective device. In the case of Phoenix Contact devices, it starts with “00.a0.45.”.

#### IP address

The IP address allows the Modbus device to be accessed during operation. PC Worx selects the address from the area that is set on the project node.



If the area for the IP addresses is later modified in the project node, you will also have to adapt the addresses of the Modbus device accordingly.

#### Subnet mask

The subnet mask that was specified on the project node is assigned to each Modbus device. It can be modified specifically for each individual device.

#### Connection timeout

This value specifies the minimum time required to identify an interruption.

#### Reconnect interval

When the connection is interrupted (connection timeout) and the set time interval elapsed, an attempt is made to establish a new connection.

#### Trigger rate

The trigger rate specifies the time period during which data is exchanged with the server. The smallest trigger rate of all configured Modbus devices determines the bus cycle time.

### 5.7 Compiling after completing the bus topology

- Select the “Build, Make” command.

### 5.8 Creating the program

- Create the program.

To program the example program, proceed as described in the UM QS EN PC WORX quick start guide.

### 5.9 Compiling after creating the program

- Select the “Build, Make” command.

### 5.10 Creating process data and assigning Modbus function codes

Define specific process data for read and write access to digital inputs and outputs as well as registers and assign the data the corresponding Modbus function code.

- Make sure you are in the bus configuration workspace.
- In the “Bus structure” window, select the Generic Modbus Device.
- Under “Device Details”, select the “Modbus Register Editor” tab.
- Specify a unique and meaningful name for the process data item in the “Name” field.
- Select the desired function code”, see Table “Modbus function codes” on page 3.  
Use the function code FC03 to read and FC15 to write 16-bit words.
- Select the desired data type.
- Enter the number of bits or registers to be read or written.
- For the Modbus device, enter the memory area of the process data item as the “Address” for which the selected function code should be used.  
The memory area corresponds to the address that has been defined in the process data configuration of the IFS CONF software, refer also to the application note “EM-...-Gateway-IFS Quick Start”.  
In the example, a 16-bit word should be read from an internal register. The FC03 Modbus function code is used here. The value “29760” is set as the address, since the memory area for the process data item used to read an internal register word-by-word has the value “29760” for the EM-MODBUS-GATEWAY-IFS module.
- The “Data Direction” indicates whether the function accesses a digital input/an internal register or a digital output/output register. The data direction depends on the selected function code and cannot be modified manually.

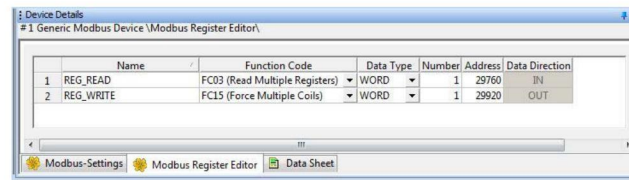


Figure 5 Creating process data and assigning Modbus function codes

### 5.11 Generating variables and assigning process data

Process data and variables are assigned in the process data assignment workspace.

- Switch to the process data assignment workspace to assign the variables to the process data.
- Select the Modbus device in the top right window. The standard configuration is then displayed in the top left window, “Symbols/Variables”.
- In the top left window, “Symbols/Variables”, select the standard resource.
- In the top right window, select the device for which you would like to link the process data to variables (in Figure 6: Generic Modbus Device; in the example: EM-MODBUS-GATEWAY-IFS).
- Select the process data item to be linked.
- Variables are created when the program is created. Using drag & drop, link the selected variable to one of the variables on the left-hand side.  
If you would like to link further process data but no corresponding variables have been created yet, select “Create Variable” in the context menu.

The created variable is displayed in the bottom left window.

- Repeat this procedure for all inputs to be evaluated and for all outputs to be controlled.

The result of the process data assignment is shown in Figure 6.

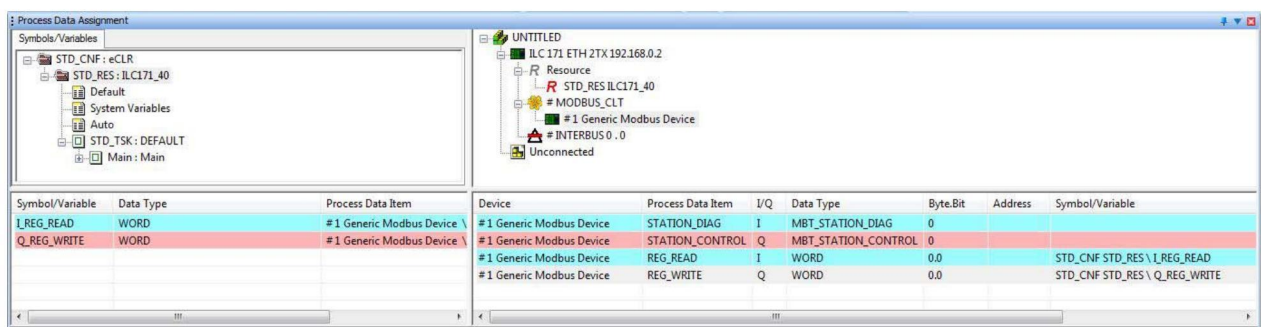


Figure 6 All used process data assigned to variables

**5.12 Generating diagnostic and control variables and assigning process data**



In the IEC programming workspace, the diagnostic and control structure is declared in the project tree window under “Data Types, sys\_flag\_types”.

PC Worx provides a diagnostic and control structure for each Modbus device allowing the connection status, connection statistics and connection interruptions to be read. To use the diagnostic and control structure, always create a diagnostic and a control variable.

- Switch to the IEC programming workspace.

- In the project tree window, double-click on “Global Variables”.

The global variables of the standard resource are displayed.

- Enter a new variable via the context menu which should be used as a control variable.
- Select the MBT\_STATION\_CONTROL type for the control variable.
- Enter a new variable via the context menu which should be used as a diagnostic variable.
- Select the MBT\_STATION\_DIAG type for the diagnostic variable.

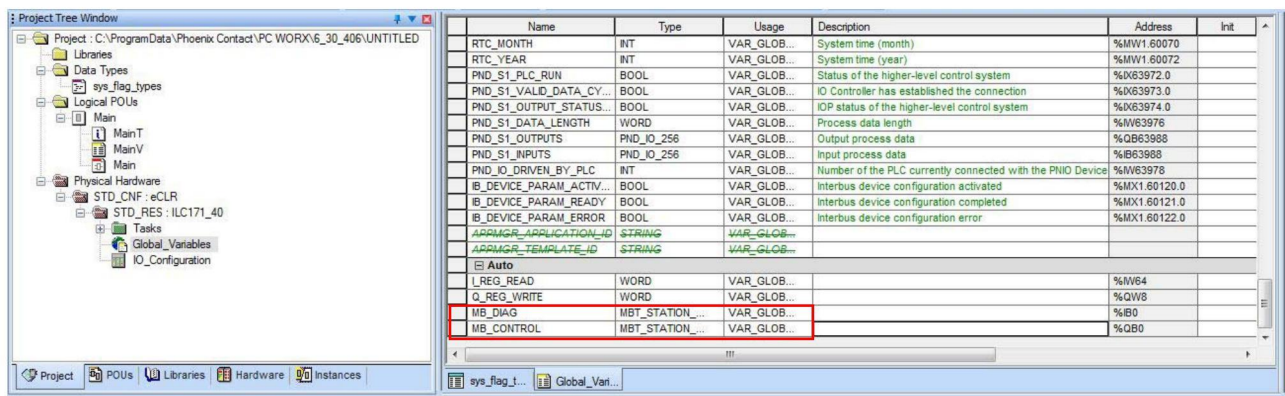


Figure 7 Generating diagnostic and control variables

- Switch to the process data assignment workspace to assign the process data to the control and diagnostic variables, as described in Section 5.11 on page 7.

The result of the process data assignment is shown in Figure 8.



Figure 8 Process data assigned to the control and diagnostic variables



### 5.13 Compiling a project

- Select the “Build, Make” command.

### 5.14 Configuring the Modbus device

Perform all of the required configurations for the Modbus device (e.g., setting the IP address and process data watchdog).

For information on how to configure and start up the device, please refer to the application note “EM-...-Gateway-IFS Quick Start”.

Ensure that the configuration setting "PC-WORX mode" has been activated via the IFS CONF software.



Byte order of process data words:

The EM-MODBUS-GATEWAY-IFS saves data words in Big Endian format (Motorola), which means that the high byte is saved first. The ILC saves data words in Little Endian format (Intel), which means that the low byte is saved first.

Please consider this when addressing the outputs (here RegWrite).

### 5.15 Sending a project and performing a cold restart

- Open the “Project control” dialog box.
- Activate the “Include Bootproject” checkbox in the “Project” area.
- Click on “Download” in the area on the left.

## 6 Acyclic communication using the MB\_ASYNC\_RW function block

The MB\_ASYNC\_RW function block enables acyclic communication between the controller and the Modbus device.

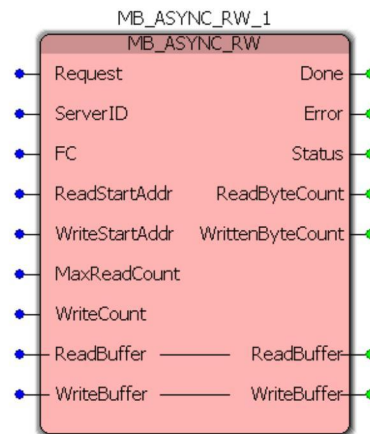







Figure 9 MB\_ASYNC\_RW function block

Input and output parameters of the function block		
Name	Data type	Description
Request	BOOL	The input parameters are checked and the function block is activated with a positive edge at this input. After the function code has been executed successfully, the function block is deactivated and can only be re-activated by a new positive edge.
ServerID	BYTE	Server ID of the Modbus device in the bus configuration.  <div style="border: 1px solid black; padding: 5px;"> <p><b>i Recommended:</b> Use the server ID of the previously created and linked diagnostic variable (see Section 6.2 “Using the server ID of the diagnostic variable”). Alternatively, the consecutive device number which is displayed in the “Device Details” window can be used (see Figure 4).</p> <p><b>Please note:</b> Under some circumstances, changes in the bus structure may lead to modified consecutive device numbers. This may cause errors when creating the consecutive number as a server ID on the function block.</p> </div>
FC	BYTE	Modbus function code (see Table “Modbus function codes” on page 3)
ReadStartAddr	WORD	Start address of the memory from which data should be read out.
WriteStartAddr	WORD	Start address from which data is to be written to the memory.
MaxReadCount	INT	Number of bits or registers to be read.
WriteCount	INT	Number of bits or registers to be written.
Done	BOOL	TRUE: The block has been executed successfully and acyclic communication has taken place. FALSE: The block is still being executed or has not been executed.
Error	BOOL	TRUE: An error has occurred. The “Status” output provides details. FALSE: No error has occurred.  <div style="border: 1px solid black; padding: 5px;"> <p><b>i</b> “Error” indicates an error as long as the “Request” input is active.</p> </div>

Input and output parameters of the function block		
Name	Data type	Description
Status	DWORD	In the event of an error (Error = TRUE), the “Status” output contains an error code. The possible error codes are shown in Table “Error codes of the Status output” on page 11.
ReadByteCount	INT	Number of read bits/registers
WrittenByteCount	INT	Number of written bits/registers
ReadBuffer	ARRAY OF BYTE, ARRAY OF WORD	Buffer (250 bytes, maximum), in which the read bits/registers are stored (depending on the function code used).  <div style="border: 1px solid black; padding: 2px; display: inline-block;">  To define the size of the buffer, see Section 6.3 “Specifying the size of the ReadBuffer/WriteBuffer”.                 </div>
WriteBuffer	ARRAY OF BYTE, ARRAY OF WORD	Buffer (250 bytes, maximum) for the written bits/registers (depending on the function code used).  <div style="border: 1px solid black; padding: 2px; display: inline-block;">  To define the size of the buffer, see Section 6.3 “Specifying the size of the ReadBuffer/WriteBuffer”.                 </div>

**6.1 Error codes of the Status output (Error = TRUE)**

Error codes of the Status output	
Value	Meaning
0x0101 0000	Unsupported/unknown Modbus function
0x0102 0000	The value for MaxReadCount is outside the permissible range.
0x0103 0000	The value for WriteCount is outside the permissible range.
0x0104 0000	Wrong data type for ReadBuffer. Permissible data types are ARRAY OF BYTE or ARRAY OF WORD.
0x0105 0000	The elements of the ReadBuffer array are not of the type WORD or BYTE.
0x0106 0000	Wrong data type for WriteBuffer. Permissible data types are ARRAY OF BYTE or ARRAY OF WORD.
0x0107 0000	The elements of the WriteBuffer array are not of the type WORD or BYTE.
0x0108 0000	ReadBuffer is too small. Reduce the number of elements for MaxReadCount or increase the ReadBuffer size.
0x0109 0000	WriteBuffer is too small. Reduce the number of elements for WriteCount or increase the WriteBuffer size.
0x010A 0000	ReadBuffer is too large.  <div style="border: 1px solid black; padding: 2px; display: inline-block;">  To define the size of the buffer, see Section 6.3 “Specifying the size of the ReadBuffer/WriteBuffer”.                 </div>
0x010B 0000	WriteBuffer is too large.  <div style="border: 1px solid black; padding: 2px; display: inline-block;">  To define the size of the buffer, see Section 6.3 “Specifying the size of the ReadBuffer/WriteBuffer”.                 </div>
0x0201 0000	Unknown server ID (not configured)
0x0202 xxxx	Modbus protocol error code  <div style="border: 1px solid black; padding: 2px; display: inline-block;">  For detailed information on the error codes of the Modbus TCP protocol, refer to the “MODBUS APPLICATION PROTOCOL SPECIFICATION” document. Current document versions can be found at <a href="http://www.modbus-ida.org">www.modbus-ida.org</a>.                 </div>

Error codes of the Status output	
Value	Meaning
0x0203 0016	Timeout when receiving the response from the Modbus device.
0x0203 0019	The connection was terminated by the Modbus device.
0x0203 001F	The request was not sent. The Modbus device is not accessible.
0x0301 0000	Timeout when receiving the response from the Modbus stack. The Modbus device is not accessible.

6.2 Using the server ID of the diagnostic variable

**i** More detailed information on functional block diagram (FBD) programming can be found in the UM QS EN PC WORX quick start guide.

- Double-click on the “ServerID” input parameter of the function block to specify the variable properties.
- In the “Variable Properties” window, select the name of the previously created diagnostic variable (here: “MB\_DIAG”).

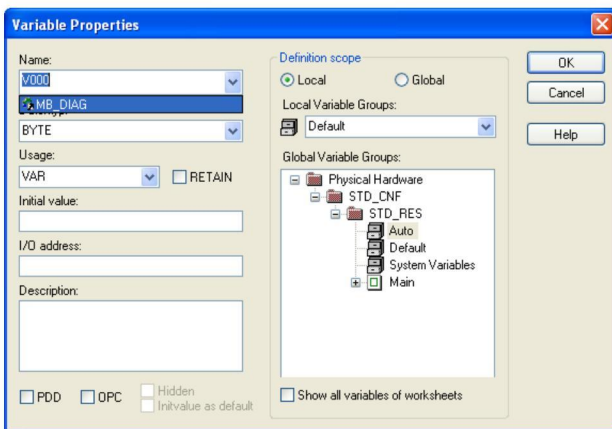


Figure 10 Creating the MB\_DIAG.ServerID variable (1)

- Put a period after the selected name and select the “ServerID” entry from the appearing list.

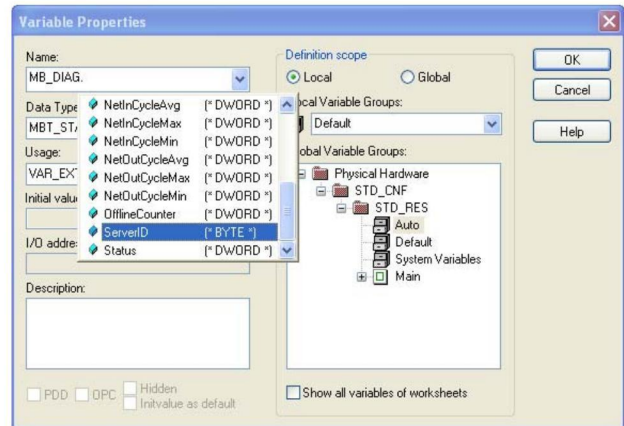


Figure 11 Creating the MB\_DIAG.ServerID variable (2)

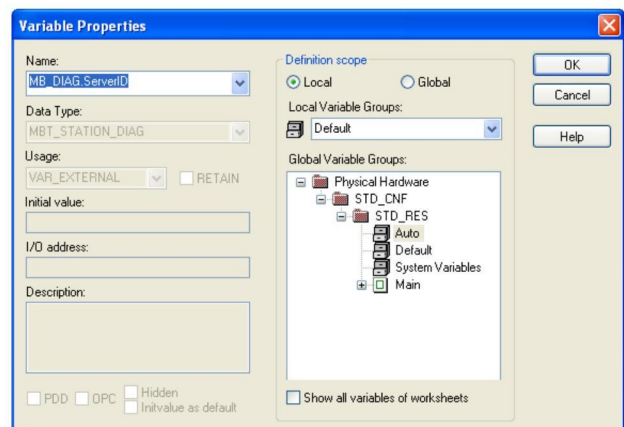


Figure 12 Creating the MB\_DIAG.ServerID variable (3)

- Confirm your entries with “OK”.

The server ID of the “MB\_DIAG” diagnostic variable has now been assigned to the “ServerID” input parameter of the function block (see Figure 13).

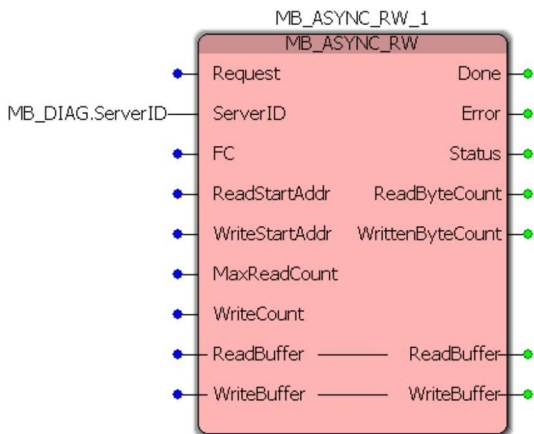


Figure 13 MB\_DIAG.ServerID variable as “ServerID” in the function block

### 6.3 Specifying the size of the ReadBuffer/WriteBuffer

The size of the “ReadBuffer” and “WriteBuffer” parameters can be specified individually by defining the corresponding data types.

**i** For ReadBuffer and WriteBuffer, the maximum size is 250 bytes each.

- Double-click on “sys\_flag\_types” in the project tree window.
- Define the desired data types and their sizes as shown in Figure 14.

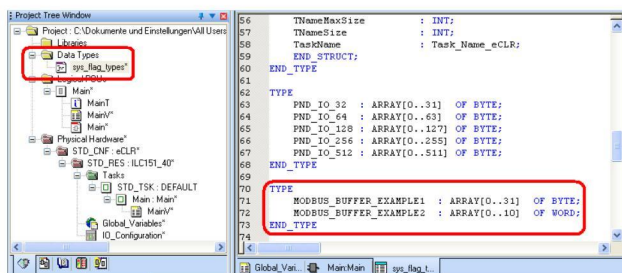


Figure 14 Creating data types

- Once you have defined the data types, select the “Build, Make” command.

The data types can be used in the variable worksheet after compiling. In Figure 15, the previously created “MODBUS\_BUFFER\_EXAMPLE1” data type is used for the “ReadBuffer” input/output parameter.

- Double-click on the “ReadBuffer” input parameter in the MB\_ASYNC\_RW\_1 function block.
- In the “Variable Properties” window, enter a name for the variable (in the example: “Read\_Buffer\_1”).
- In the “Data Type” list, select the previously created data type you wish to use (in the example: “MODBUS\_BUFFER\_EXAMPLE1”).

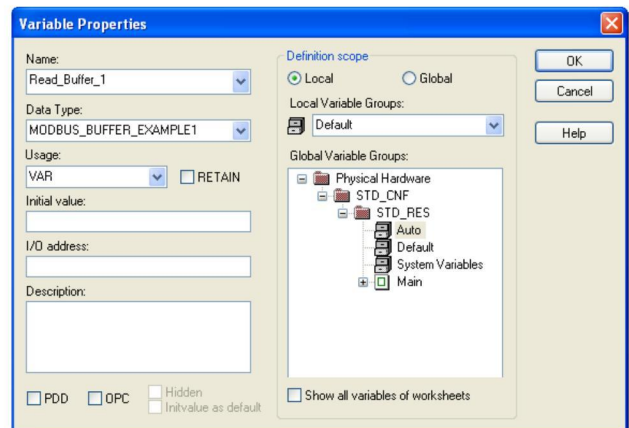


Figure 15 Creating a variable

- Confirm your entries with “OK”.

The “Read\_Buffer\_1” variable has now been assigned to the “ReadBuffer” input/output parameter.

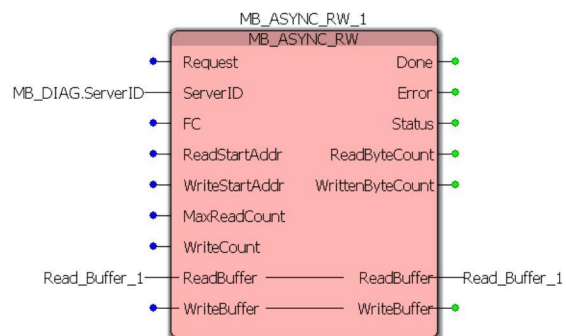


Figure 16 “Read\_Buffer\_1” variable as “ReadBuffer” in the function block

**i** For further information on user-defined data types and their use, please refer to the PC Worx online help.