FAQ • 06/2014

Modbus/TCP with MB_CLIENT and MB_SERVER Instructions

S7-1500 CPU

http://support.automation.siemens.com/WW/view/en/94766380

Contents

1 Introduction

Modbus/TCP communication between two S7-1500 CPUs is presented. The instruction "MB_CLIENT" or "MB_SERVER" is called and parameterized in the user program of the S7-1500 CPU.

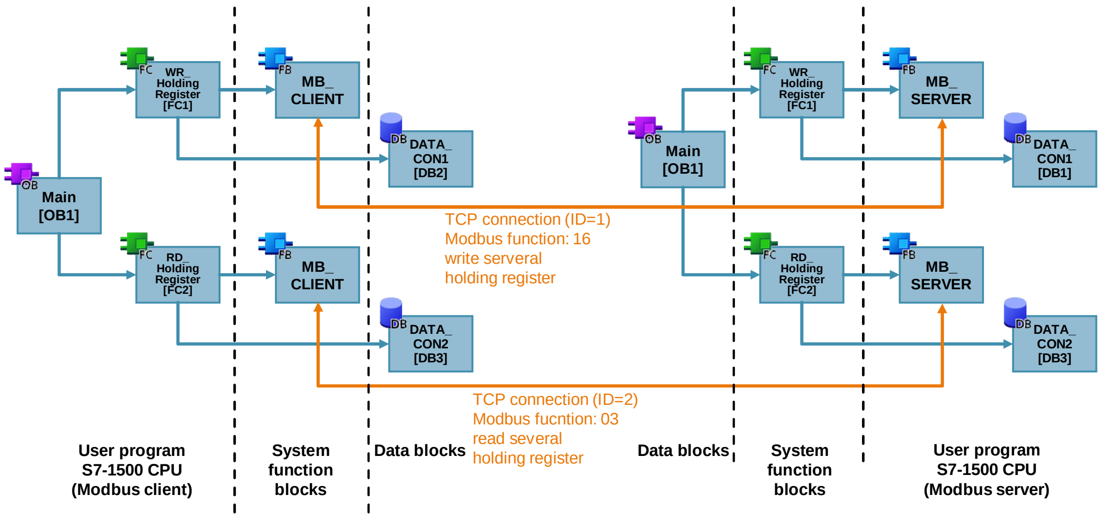
The "MB_CLIENT" instruction communicates as Modbus/TCP client over the PROFINET connection. You use the "MB_CLIENT" instruction to establish a connection between the client and the server, send Modbus requests and receive responses, and control the connection disconnection of the Modbus/TCP client.

The "MB_SERVER" instruction communicates as Modbus/TCP server over the PROFINET connection. The "MB_SERVER" instruction processes connection requests of a Modbus/TCP client, receives and processes Modbus requests and sends response messages.

In this example we demonstrate 2 Modbus functions. A Modbus/TCP connection is established for each Modbus function via a Modbus block pair (MB_CLIENT and MB_SERVER).

Figure 1-1 shows an overview of the demonstrated Modbus functions and the assignment of the block pairs.

Figure 1-1



2 Modbus/TCP Client User Program

In the Modbus/TCP client user program, the "MB_CLIENT" instruction is called for each Modbus/TCP connection with a unique ID and separate instance data block. The call of the "MB_CLIENT" instruction is done each time in a separate function.

Table 2-1

ID	Call of the "MB_CLIENT" instruction	Instance DB of the "MB_CLIENT" instruction	Description
1	FC1 "WR_HoldingRegister"	DB1 "MB_CLIENT_DB"	Write to holding register
2	FC2 "RD_HoldingRegister"	DB4 "MB_CLIENT_DB_1"	Read holding register

2.1 FC1 "WR_HoldingRegister"

The FC1 "WR_HoldingRegister" function calls the "MB_CLIENT" instruction internally to establish the Modbus/TCP connection with ID=1 and write to the holding register of the Modbus/TCP server.

The communication request to write to the holding register is controlled by the variable "DATA_CON1".REQ at the REQ input.

In this example the Modbus/TCP connection with connection number=1 is established to Port 502 of the Modbus/TCP server. The Modbus/TCP server has the IP address 192.168.0.10.
10 data words are written to the holding register of the Modbus/TCP server. For this you set the input parameters MB_MODE, MB_DATA_ADDR and MB_DATA_LEN as follows:

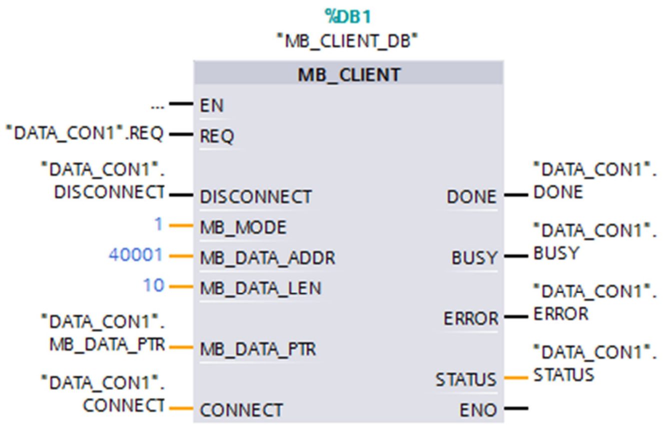
- MB_MODE = 1
- MB_DATA_ADDR = 40001
- MB_DATA_LEN = 10

Note Section 2.4 gives a detailed description of parameters MB_MODE and DATA_ADDR.

2.1 FC1 "WR_HoldingRegister"

Figure 2-1 shows the call and parameters of the "MB_CLIENT" instruction in FC1.

Figure 2-1



Note Section 2.3 gives an overview and description of the input and output parameters of the "MB_CLIENT" instruction.

Send buffer

At input parameter MB_DATA_PTR you specify the buffer for the data to be sent to the Modbus/TCP server. The data to be written to the holding register is stored in DB2 "DATA_CON1" in the MB_DATA_PTR variable.

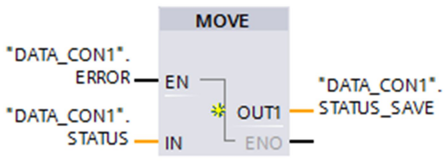
Table 2-2

Variable name	Data type
MB_DATA_PTR	Array [0..9] of Word

Error evaluation

If the "MB_CLIENT" instruction terminates with an error in FC1, the value of the STATUS output parameter is saved in the "STATUS_SAVE" variable of the "Word" data type for error evaluation in DB2 "DATA_CON1".

Figure 2-2



2.2 FC2 "RD_HoldingRegister"

The FC2 "RD_HoldingRegister" function calls the "MB_CLIENT" instruction internally to establish the Modbus/TCP connection with ID=2 and read the holding register.

The communication request to read the holding register is controlled by the variable "DATA_CON2.REQ" at the REQ input.

In this example the Modbus/TCP connection with connection number=2 is established to Port 503 of the Modbus/TCP server. The Modbus/TCP server has the IP address 192.168.0.10.

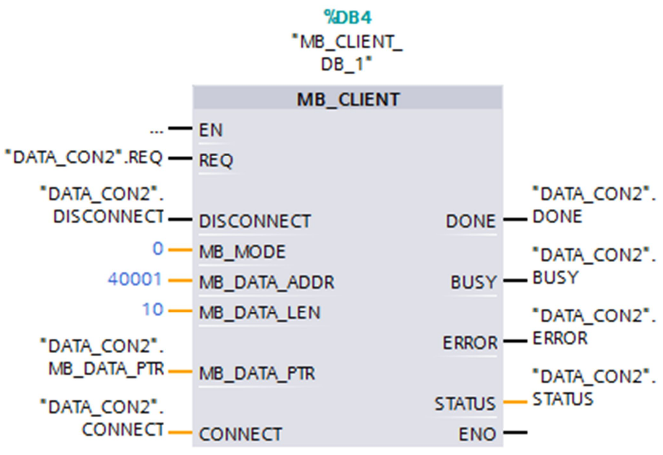
10 data words are read from the holding register. For this you set the input parameters MB_MODE, MB_DATA_ADDR and MB_DATA_LEN as follows:

- MB_MODE = 0
- MB_DATA_ADDR = 40001
- MB_DATA_LEN = 10

Note Section 2.4 gives a detailed description of parameters MB_MODE and MB_DATA_ADDR.

Figure 2-3 shows the call and parameters of the "MB_CLIENT" instruction in FC2.

Figure 2-3



Note Section 2.3 gives an overview and description of the input and output parameters of the "MB_CLIENT" instruction.

Receive buffer

At input parameter MB_DATA_PTR you specify the buffer for the data received from the Modbus/TCP server. The data read from the holding register is stored in DB3 "DATA_CON2" in the "MB_DATA_PTR" variable.

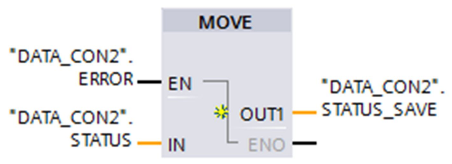
Table 2-3

Variable name	Data type
MB_DATA_PTR	Array [0..9] of Word

Error evaluation

If the "MB_CLIENT" instruction terminates with an error in FC2, the value of the STATUS output parameter is saved in the "STATUS_SAVE" variable of the "Word" data type for error evaluation in DB3 "DATA_CON2".

Figure 2-4



2.3 Input and Output Parameters of the "MB_CLIENT" Instruction

Input parameters

The "MB_CLIENT" has the following input parameters.

Table 2-4

Input parameters	Data type	Description
REQ	BOOLEAN	Communication request with the Modbus/TCP server The REQ parameter is level controlled. This means that the instruction sends communication requests for as long as the input is set (REQ=true). • With the communication request the instance DB is blocked for other clients. • Changes to the input parameters only become effective when the server responds or an error message is issued. • If the REQ parameter is set again when a Modbus request is running, no other transfer is made.

Input parameters	Data type	Description
DISCONNECT	BOOLEAN	You use the parameters to control the establishment of the connection to and disconnection from the Modbus server. • 0: Establish communication connection to the connection partner configured for the CONNECT parameter. • 1: Disconnect communication connection. No other function is executed while the connection is being disconnected. After successful disconnection of the connection the value w#16#0003 is output at the STATUS parameter. If the REQ parameter is set when the connection is established, the Modbus request is sent immediately.
MB_MODE	USINT	Selection of the Modbus request mode (read, write or diagnostics) Section 2.4 gives a detailed description of the MB_MODE parameter.
MB_DATA_ADDR	UDINT	Initial address of the data which the "MB_CLIENT" instruction accesses. Section 2.4 gives a detailed description of the MB_DATA_ADDR parameter.
MB_DATA_LEN	UINT	Data length: number of bits or words for the data access.
MB_DATA_PTR	VARIANT	Pointer to the Modbus data register. The register is a buffer for the data received from the Modbus/TCP server or to be sent to the Modbus/TCP server. The pointer must refer to a global data block (DB) with standard access.
CONNECT	VARIANT	Pointer to the connection description. You can use the following structures (system data types). • TCON_IP_v4: contains all the address parameters needed for establishing a programmed connection. When TCON_IP_v4 is used, the connection is established when the "MB_CLIENT" instruction is called. • TCON_Configured: contains the address parameters of a configured connection. When TCON_Configured is used, an existing connection is used, which was established after loading of the hardware configuration by the CPU. The TCON_IP_v4 structure is used in this example. The structure of TCON_IP_v4 is described in section 2.5.

Output parameters

The "MB_CLIENT" instruction has the following output parameters.

Table 2-5

Output parameters	Data type	Description
DONE	BOOLEAN	The bit at the DONE output parameter is set to "1" as soon as the last job has been executed without error.
BUSY	BOOLEAN	• 0: No Modbus request being processed • 1: Modbus request is being processed
ERROR	BOOLEAN	• 0: No error • 1: Error occurred. The cause of the error is displayed by the STATUS parameter.
STATUS	WORD	Detailed Status information of the instruction.

2.4 MB_MODE and MB_DATA_ADDR Parameters

The "MB_CLIENT" instruction uses the MB_MODE instead of a function code. You use the MB_DATA_ADDR parameter to define the Modbus start address which you wish to access. The combination of the MB_MODE, MB_DATA_ADDR and MB_DATA_LEN parameters defines the function code that is used in the current Modbus message.

Table 2-6 shows the relationship between the input parameters of the "MB_CLIENT" function and the Modbus function.

Table 2-6

MB_MODE	MB_DATA_ADDR	MB_DATA-LEN	Modbus function	Function and data type
0	Initial address: • 40001 to 49999 • 400001 to 465535	Data length (WORD) per call: • 1 to 125 • 1 to 125	03	Read holding register • 0 to 9998 • 0 to 65534
1	Initial address: • 40001 to 49999 • 400001 to 465535	Data length (WORD) per call: • 2 to 123 • 2 to 123	16	Write multiple holding registers • 0 to 9998 • 0 to 65534

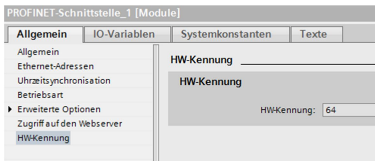
2.5 CONNECT Parameter

Use the following structure for connection description according to TCON_IP_v4 for programmed connections at the CONNECT parameter.

- Make sure that you specify connections only of the TCP type in the TCON_IP_v4 structure.
- The connection must not use the following TCP port numbers: 20, 21, 25, 80, 102, 123, 5001, 34963 and 34964.

2.5 CONNECT Parameter

Table 2-7

Byte	Parameter	Data type	Value	Description
0..1	InterfaceID	HW_ANY	64	Hardware ID of the local interface (value range: 0 to 65535). The hardware ID is to be found in the device configuration of the CPU. Mark the PROFINET interface to display the properties of the PROFINET interface in the inspector window. In the "General" tab you navigate to "HW Identifier" to determine the hardware ID.  PROFINET-Schnittstelle_1 [Module] HW-Kennung: 64 Allgemein IO-Variablen Systemkonstanten Texte Allgemein Ethernet-Adressen Uhrzeitsynchronisation Betriebsart Erweiterte Optionen Zugriff auf den Webserver HW-Kennung HW-Kennung HW-Kennung
2..3	ID	CONN_OUC	1: For the Modbus function "Write holding register" 2: For the Modbus function "Read holding register"	Reference to this connection (value range: 1 to 4095). The parameter uniquely identifies a connection in the CPU. Each single instance of the "MB_CLIENT" instruction must use a unique ID.
4	ConnectionType	BYTE	11	Connection type Select 11 (decimal) for TCP. Other connection types are not permissible.
5	ActiveEstablished	BOOLEAN	TRUE	ID for the type of connection setup. The TCP/Modbus client actively establishes the connection. Select TRUE for active establishment of the connection.

6..9	RemoteAddress	ARRAY[1..4] of BYTE	addr[1]=192
			addr[2]=168
			addr[3]=0
			addr[4]=10
				IP address of the connection partner (Modbus/TCP server). In this example: 192.168.0.10

10..11	RemotePort	UINT	502: for ID=1 503: for ID=2	Port number of the remote connection partner. Use the IP port number of the server to which the client establishes a connection and communicates over the TCP/IP protocol.
12..13	LocalPort	UINT	0	Port number of the local connection partner. • Port numbers: 1 to 49151 Any port "0"

3 Modbus/TCP Server User Program

In the Modbus/TCP server user program, the "MB_SERVER" instruction is called for each Modbus/TCP connection with a unique ID and unique instance data block. The call of the "MB_SERVER" instruction is done each time in a separate function.

Table 3-1

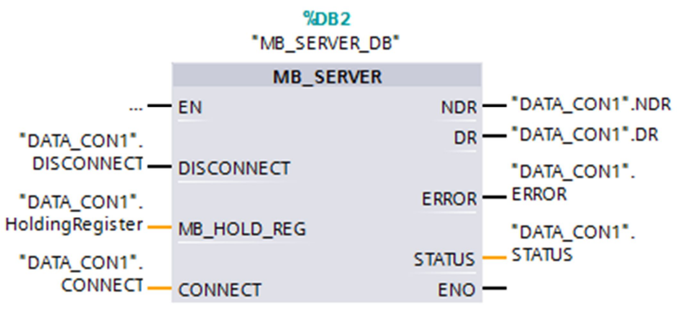
ID	Call of the "MB_SERVER" instruction	Instance DB of the "MB_SERVER" instruction	Description
1	FC1 "WR_HoldingRegister"	DB2 "MB_SERVER_DB"	Write to holding register
2	FC2 "RD_HoldingRegister"	DB4 "MB_SERVER_DB_1"	Read holding register

3.1 FC1 "WR_HoldingRegister"

The FC1 "WR_HoldingRegister" function calls the "MB_SERVER" instruction internally to process the connection request to write to the holding register. The connection request is made over the Modbus/TCP connection with ID=1 and Port 502.

Figure 3-1 shows the call and parameters of the "MB_SERVER" instruction in FC1.

Figure 3-1



Note Section 3.3 gives an overview and description of the input and output parameters of the "MB_SERVER" instruction.

MB_HOLD_REG parameter

The MB_HOLD_REG parameter is a pointer to a data buffer for storing the data that is read from the Modbus/TCP server or is to be written to the Modbus/TCP server. You can use a global data block or a marker as memory area.

The table below shows how the Modbus addresses are mapped to the holding register for the Modbus function 16 (write multiple WORDs).

Table 3-2

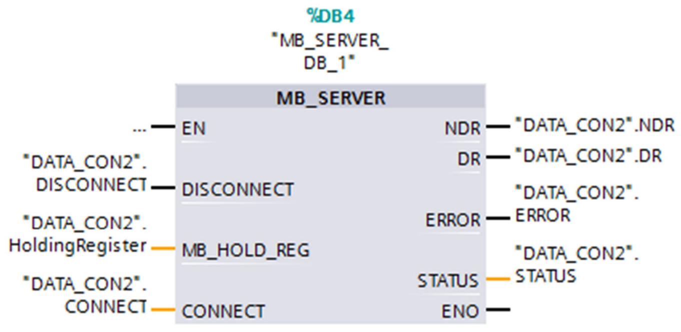
Modbus address	Symbolic name
40001	"DATA_CON1".HoldingRegister[100]
40002	"DATA_CON1".HoldingRegister[101]
40003	"DATA_CON1".HoldingRegister[102]
40004	"DATA_CON1".HoldingRegister[103]
40005	"DATA_CON1".HoldingRegister[104]
40006	"DATA_CON1".HoldingRegister[105]
40007	"DATA_CON1".HoldingRegister[106]
40008	"DATA_CON1".HoldingRegister[107]
40009	"DATA_CON1".HoldingRegister[108]
40010	"DATA_CON1".HoldingRegister[109]

3.2 FC2 "RD_HoldingRegister"

The FC2 "RD_HoldingRegister" function calls the "MB_SERVER" instruction internally to process the connection request to read the holding register. The connection request is made over the Modbus/TCP connection with ID=2 and Port 503.

Figure 3-2 shows the call and parameters of the "MB_SERVER" instruction in FC2.

Figure 3-2



Note Section 3.3 gives an overview and description of the input and output parameters of the "MB_SERVER" instruction.

MB_HOLD_REG parameter

The MB_HOLD_REG parameter is a pointer to a data buffer for storing the data that is read from the Modbus/TCP server or is to be written to the Modbus/TCP server. You can use a global data block or a marker as memory area.

The table below shows how the Modbus addresses are mapped to the holding register for the Modbus function 3 (read WORD).

Table 3-3

Modbus address	Symbolic name
40001	"DATA_CON2".HoldingRegister[100]
40002	"DATA_CON2".HoldingRegister[101]
40003	"DATA_CON2".HoldingRegister[102]
40004	"DATA_CON2".HoldingRegister[103]
40005	"DATA_CON2".HoldingRegister[104]
40006	"DATA_CON2".HoldingRegister[105]
40007	"DATA_CON2".HoldingRegister[106]
40008	"DATA_CON2".HoldingRegister[107]
40009	"DATA_CON2".HoldingRegister[108]
40010	"DATA_CON2".HoldingRegister[109]

3.3 Input and Output Parameters of the "MB_SERVER" Instruction

Input parameters

The "MB_SERVER" has the following input parameters.

Table 3-4

Input parameters	Data type	Description
DISCONNECT	BOOLEAN	The "MB_SERVER" instruction enters into a passive connection with a partner module. The server reacts to a connection request from the IP address that is specified in the "TCON_IP_v4" structure at the CONNECT input parameter. You use this parameter to control when the connection request is accepted. • 0: If there is no communication connection, a passive connection is established. • 1: Initialization of connection disconnection. If the input is set, no other processes are executed. After successful disconnection of the connection the value w#16#0003 is output at the STATUS output parameter.
MB_HOLD_REG	VARIANT	Pointer to the Modbus holding register of the "MB_SERVER" instruction. The holding register contains the values which a Modbus client is allowed to access over the Modbus functions 3 (read), 6 and 16 (write). Use a global data block (DB) with optimized access as holding register or the memory area of the markers.

Input parameters	Data type	Description
CONNECT	VARIANT	Pointer to the connection description. You can use the following structures (system data types). • TCON_IP_v4: contains all the address parameters needed for establishing a programmed connection. When TCON_IP_v4 is used, the connection is established when the "MB_SERVER" instruction is called. • TCON_Configured: contains the address parameters of a configured connection. When TCON_Configured is used, the connection is established after loading of the hardware configuration by the CPU. The TCON_IP_v4 structure is used in this example. The structure of TCON_IP_v4 is described in section 3.4.

Output parameters

The "MB_SERVER" instruction has the following output parameters.

Table 3-5

Output parameters	Data type	Description
NDR	BOOLEAN	"New Data Ready" • 0: No new data • 1: New data written by the Modbus/TCP client
DR	BOOLEAN	"Data Read" • 0: No data read • 1: Data read by the Modbus/TCP client
BUSY	BOOLEAN	• 0: No Modbus request is being processed • 1: Modbus request is being processed
ERROR	BOOLEAN	• No error • Error occurred. The cause of the error is displayed by the STATUS parameter.
STATUS	WORD	Detailed Status information of the instruction.

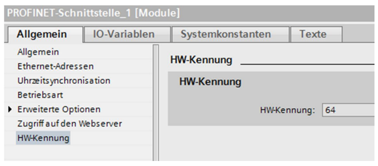
3.4 CONNECT Parameter

Use the following structure for connection description according to TCON_IP_v4 for programmed connections at the CONNECT parameter.

- Make sure that you specify connections only of the TCP type in the TCON_IP_v4 structure.
- The connection must not use the following TCP port numbers: 20, 21, 25, 80, 102, 123, 5001, 34963 and 34964.

3.4 CONNECT Parameter

Table 3-6

Byte	Parameter	Data type	Value	Description
0..1	InterfaceID	HW_ANY	64	Hardware ID of the local interface (value range: 0 to 65535). The hardware ID is to be found in the device configuration of the CPU. Mark the PROFINET interface to display the properties of the PROFINET interface in the inspector window. In the "General" tab you navigate to "HW Identifier" to determine the hardware ID.  PROFINET-Schnittstelle_1 [Module] Allgemein IO-Variablen Systemkonstanten Texte Allgemein Ethernet-Adressen Uhrzeitsynchronisation Betriebsart Erweiterte Optionen Zugriff auf den Webserver HW-Kennung HW-Kennung HW-Kennung HW-Kennung: 64
2..3	ID	CONN_OUC	1: For the Modbus function "Write holding register" 2: For the Modbus function "Read holding register"	Reference to this connection (value range: 1 to 4095). The parameter uniquely identifies a connection in the CPU. Each single instance of the "MB_SERVER" instruction must use a unique ID.
4	ConnectionType	BYTE	11	Connection type Select 11 (decimal) for TCP. Other connection types are not permissible.
5	ActiveEstablished	BOOLEAN	FALSE	ID for the type of connection setup. The TCP/Modbus server participates passively in establishing the connection. Select FALSE for passive establishment of the connection.

6..9	RemoteAddress	ARRAY[1..4] of BYTE	addr[1]=192
			addr[2]=168
			addr[3]=0
			addr[1]=192
				IP address of the connection partner. In this example: 192.168.0.1.
				If the "MB_SERVER" instruction is to accept connection requests from any connection partner use the IP address 0.0.0.0.

10..11	RemotePort	UINT	0	Port number of the remote connection partner (value range 1 to 49151). If the "MB_SERVER" instruction is to accept connection requests from any remote connection partner, use the "0" as the port number.

3.4 CONNECT Parameter

Byte	Parameter	Data type	Value	Description
12..13	LocalPort	UINT	502: for ID=1 503: for ID=2	Port number of the local connection partner (value range 1 to 49151). The number of the IP port defines which IP port is monitored for connection requests of the Modbus client. The following TCP port numbers must not be used for the passive connection of the "MB_SERVER" instruction: 20, 21, 25, 80, 102, 123, 5001, 34963 and 34964.