FAQ • 09/2014

MODBUS/TCP with the MB_CLIENT and MB_SERVER Instructions

S7-1500 CPU and S7-1200 CPU

http://support.automation.siemens.com/WW/view/en/102020340

Contents

1 Introduction

Modbus/TCP communication between S7-1500 CPUs and S7-1200 CPUs is presented.

The instructions "MB_CLIENT" and "MB_SERVER" are called and parameterized in the user program of the S7-1200 CPU and the S7-1500 CPU.

The "MB_CLIENT" instruction communicates as Modbus/TCP client over the PROFINET connection of the CPU. You do not need any additional hardware to use the instruction. You use the "MB_CLIENT" instruction to establish a connection between the client and the server, send requests and receive responses, and control disconnection of the connection.

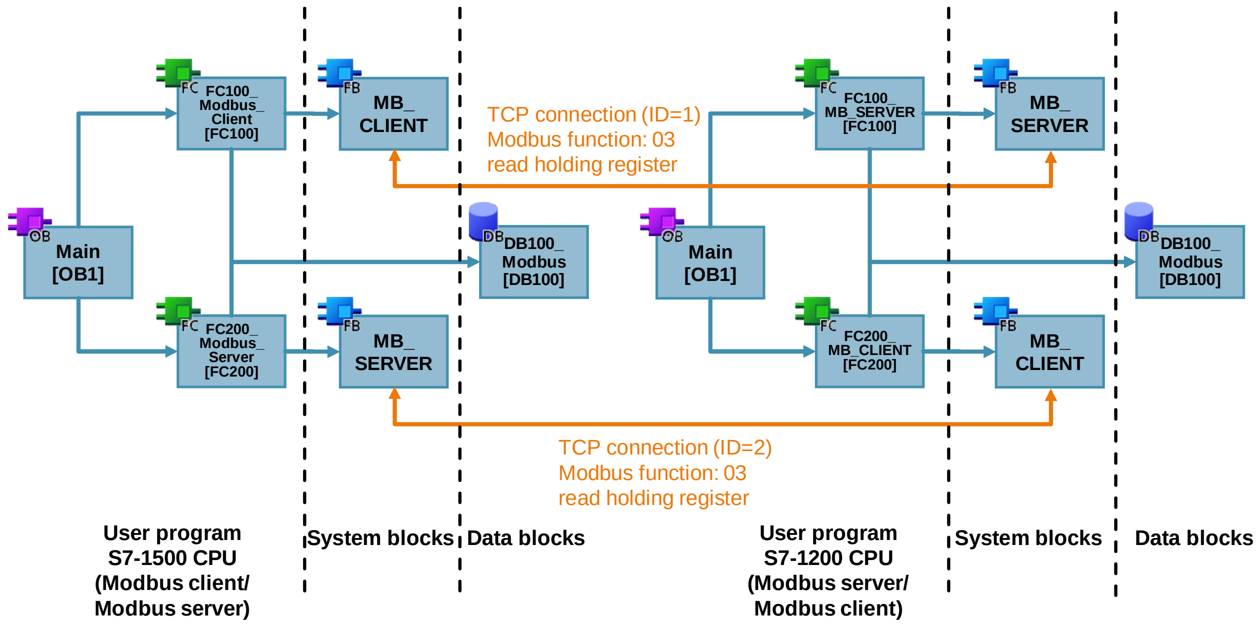
The "MB_SERVER" instruction communicates as Modbus/TCP server over the PROFINET connection of the CPU. You do not need any additional hardware to use the instruction. The "MB_SERVER" instruction processes connection requests of a Modbus/TCP client, receives requests from Modbus functions and sends response messages.

In this example a Modbus function is demonstrated via two Modbus/TCP connections.
The S7-1500 CPU establishes the first connection as Modbus TCP client. The S7-1200 CPU is Modbus TCP server.
The S7-1200 CPU establishes the second connection as Modbus TCP client. The S7-1500 CPU is Modbus TCP server.

The Modbus/TCP connections are established each via a Modbus block pair (MB_CLIENT and MB_SERVER).

Figure 1-1 shows an overview of the demonstrated Modbus function and Modbus/TCP connections as well as the assignment of the block pairs.

Figure 1-1



2 User Program of the S7-1500 CPU

In the user program of the S7-1500 CPU, the "MB_CLIENT" and "MB_SERVER" instructions are called for each Modbus/TCP connection with a unique ID and separate instance data block. The "MB_CLIENT" and "MB_SERVER" instructions are called each time in a separate function.

As Modbus TCP client the S7-1500 CPU establishes the connection to the Modbus TCP server (S7-1200 CPU) and sends the request to read the holding register.

Table 2-1

ID	Call of the "MB_CLIENT" instruction	Instance DB of the "MB_CLIENT" instruction	Description
1	FC100 "FC100_Modbus_Client"	DB3 "iDB_Modbus_Client"	Read holding register

As Modbus TCP server the S7-1500 CPU processes the connection request of the Modbus TCP client (S7-1200 CPU) and receives the request to read the holding register.

Table 2-2

ID	Call of the "MB_SERVER" instruction	Instance DB of the "MB_SERVER" instruction	Description
2	FC200 "FC200_Modbus_Server"	DB4 "iDB_Modbus_Server"	Read holding register

2.1 S7-1500 CPU: Modbus TCP Client

2.1.1 FC100 "FC100_Modbus_Client"

The FC100 "FC100_Modbus_Client" function calls the "MB_CLIENT" instruction internally to establish the Modbus/TCP connection with ID=1 and read the holding register of the Modbus/TCP server.

The communication request to read the holding register is controlled by two variables:

- "DB100_Modbus".INO_MB_CLIENT.REQ at the input parameter REQ
- "DB100_Modbus".INO_MB_CLIENT.EN at the input parameter EN

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 192.168.0.12.

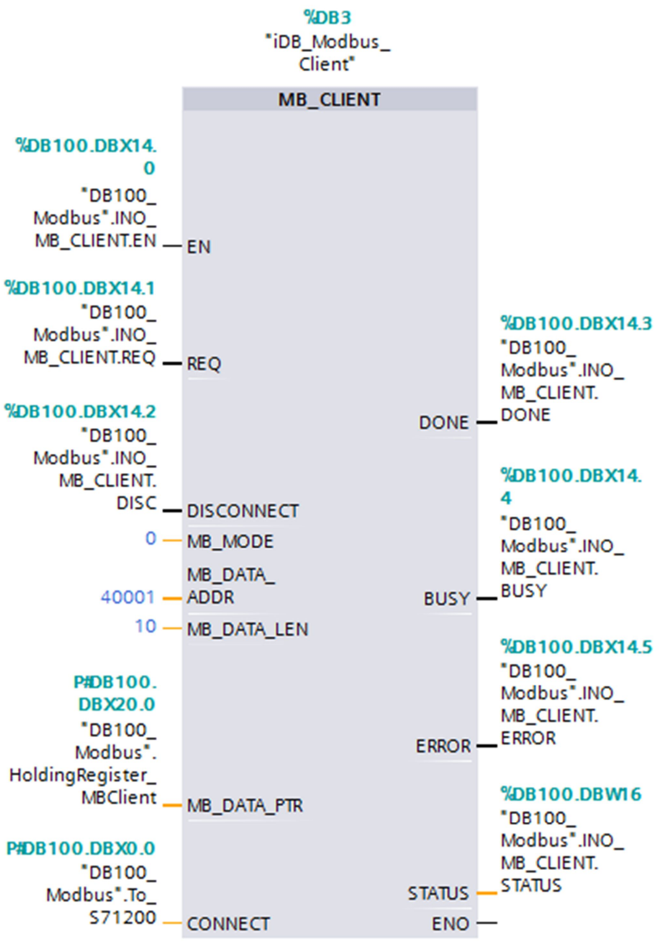
10 data words are read from the holding register. For this you set the input parameters MB_MODE, MB_DATA_ADDR and MB_DATA_LEN as follows:

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

Note Section 2.1.3 gives a detailed description of the parameters MB_MODE and MB_ADDR.

Figure 2-1 shows the call and parameters of the "MB_CLIENT" instruction in FC100.

Figure 2-1



Note

Section 2.1.2 gives an overview and description of the input and output parameters of the "MB_CLIENT" instruction.

2.1.2 Input and Output Parameters of the "MB_CLIENT" Instruction

Input parameters

The "MB_CLIENT" has the following input parameters.

Table 2-3

Input parameters	Data type	Description
REQ	BOOL	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.
DISCONNECT	BOOL	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 2.1.3 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 2.1.3 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.

Input parameters	Data type	Description
MB_DATA_PTR	VARIANT	Pointer to the Modbus data register. The register is a buffer for the data received from or to be sent to the Modbus/TCP server. The pointer can refer to a: • global data block (DB) with standard access • memory area of markers • DB with optimized access In this example the pointer refers to DB100 "DB100_Modbus" (see section 2.1.5).
CONNECT	VARIANT	Pointer to the structure of 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 sections 2.1.4 and 2.3.

Output parameters

The "MB_CLIENT" instruction has the following output parameters.

Table 2-4

Output parameters	Data type	Description
DONE	BOOL	The bit at the DONE output parameter is set to "1" as soon as the last job has been executed without error.
BUSY	BOOL	• 0: No Modbus request is being processed • 1: Modbus request is being processed
ERROR	BOOL	• 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.

2.1.3 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 2-5 shows the relationship between the input parameters of the "MB_CLIENT" function and the Modbus function.

Table 2-5

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

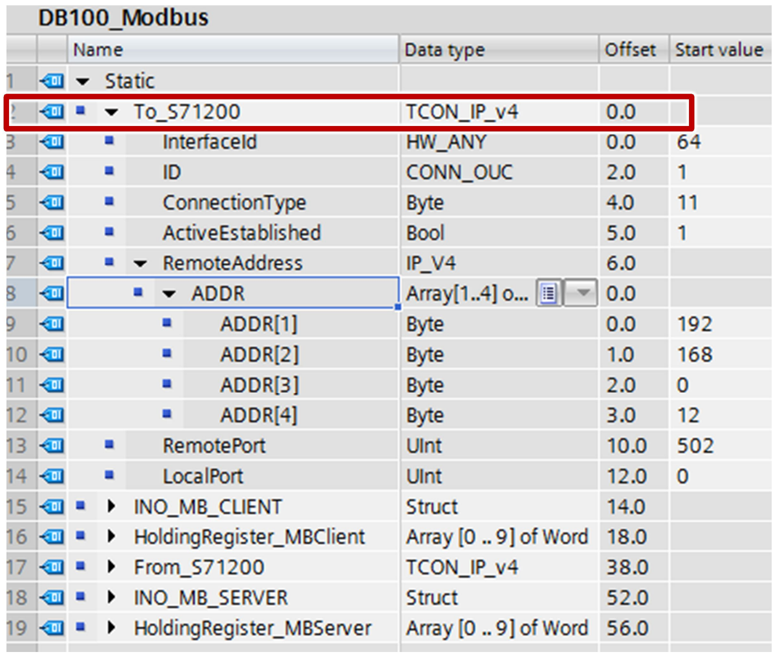
2.1.4 Data Structure at the CONNECT Parameter of the "MB_CLIENT" Instruction

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.

The figure below shows the structure of TCON_IP_v4 with the name "To_S71200". You specify this structure at the CONNECT parameter of the "MB_CLIENT" instruction. A description of the parameters of the "TCON_IP_v4" structure is available in Table 2-10.

Figure 2-2



DB100_Modbus

	Name	Data type	Offset	Start value
1	Static			
2	To_S71200	TCON_IP_v4	0.0	
3	InterfaceId	HW_ANY	0.0	64
4	ID	CONN_OUC	2.0	1
5	ConnectionType	Byte	4.0	11
6	ActiveEstablished	Bool	5.0	1
7	RemoteAddress	IP_V4	6.0	
8	ADDR	Array[1..4] o...	0.0	
9	ADDR[1]	Byte	0.0	192
10	ADDR[2]	Byte	1.0	168
11	ADDR[3]	Byte	2.0	0
12	ADDR[4]	Byte	3.0	12
13	RemotePort	UInt	10.0	502
14	LocalPort	UInt	12.0	0
15	INO_MB_CLIENT	Struct	14.0	
16	HoldingRegister_MBClient	Array [0 .. 9] of Word	18.0	
17	From_S71200	TCON_IP_v4	38.0	
18	INO_MB_SERVER	Struct	52.0	
19	HoldingRegister_MBServer	Array [0 .. 9] of Word	56.0	

2.1.5 Parameter MB_DATA_PTR

At 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 DB100 "DB100_Modbus" starting with address 18.

Table 2-6

Variable name	Data type	Address in DB100
HoldingRegister_MBClient	Array [0..9] of Word	18.0

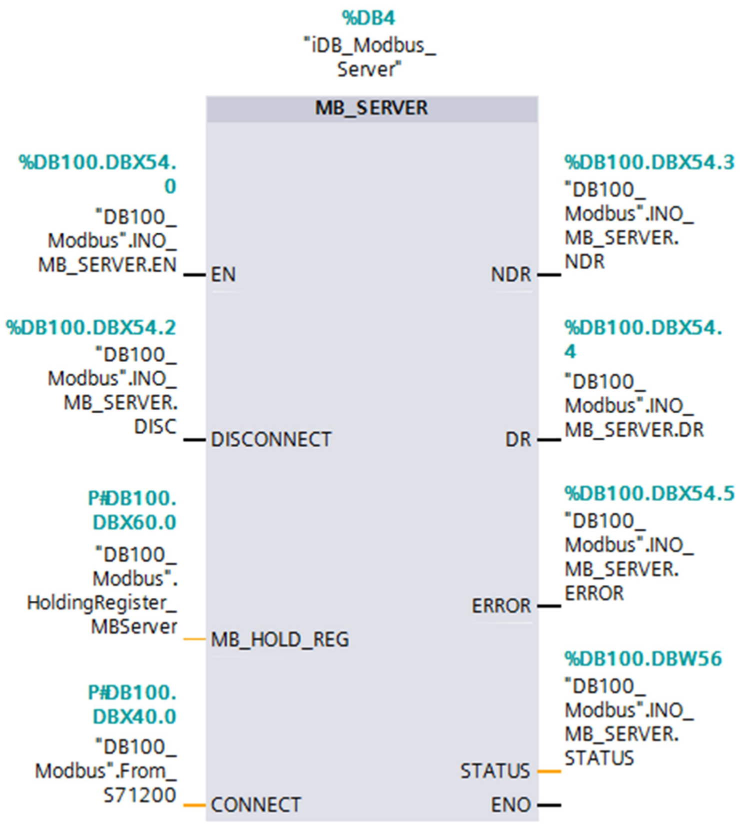
2.2 S7-1500 CPU: Modbus TCP Server

2.2.1 FC200 "FC200_Modbus_Server"

The FC200 "FC200_Modbus_Server" 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 2-3 shows the call and parameters of the "MB_SERVER" instruction in FC200.

Figure 2-3


%DB4
"iDB_Modbus_Server"
MB_SERVER
%DB100.DBX54.0
"DB100_Modbus".INO_MB_SERVER.EN
EN
%DB100.DBX54.2
"DB100_Modbus".INO_MB_SERVER.DISC
DISCONNECT
P#DB100.DBX60.0
"DB100_Modbus".HoldingRegister_MBServer
MB_HOLD_REG
P#DB100.DBX40.0
"DB100_Modbus".From_S71200
CONNECT
NDR
%DB100.DBX54.3
"DB100_Modbus".INO_MB_SERVER.NDR
DR
%DB100.DBX54.4
"DB100_Modbus".INO_MB_SERVER.DR
ERROR
%DB100.DBX54.5
"DB100_Modbus".INO_MB_SERVER.ERROR
STATUS
%DB100.DBW56
"DB100_Modbus".INO_MB_SERVER.STATUS
ENO

Note

Section 2.2.2 gives an overview and description of the input and output parameters of the "MB_SERVER" instruction.

2.2.2 Input and Output Parameters of the "MB_SERVER" Instruction

Input parameters

The "MB_SERVER" has the following input parameters.

Table 2-7

Input parameters	Data type	Description
DISCONNECT	BOOL	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 one of the following data area as holding register: • global data block (DB) with optimized access • DB with standard access • memory area of the markers Table 2-9 shows how the Modbus addresses are mapped to the holding register for the Modbus function 3 (read WORD).
CONNECT	VARIANT	Pointer to the structure of 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 sections 2.2.4 and 2.3.

Output parameters

The "MB_SERVER" instruction has the following output parameters.

Table 2-8

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

2.2.3 MB_HOLD_REG parameter

The data read from the Modbus TCP server or written to the Modbus TCP server is stored in the data block DB100 "DB100_Modbus".

Table 2-9 shows how the Modbus addresses are mapped to the holding register for the Modbus function 3 (read WORD).

Table 2-9

Modbus address	Absolute address	Symbolic name
40001	DB100.DBW56	"DB100_Modbus".HoldingRegister_MBServer[0]
40002	DB100.DBW58	"DB100_Modbus".HoldingRegister_MBServer[1]
40003	DB100.DBW60	"DB100_Modbus".HoldingRegister_MBServer[2]
40004	DB100.DBW62	"DB100_Modbus".HoldingRegister_MBServer[3]
40005	DB100.DBW64	"DB100_Modbus".HoldingRegister_MBServer[4]
40006	DB100.DBW66	"DB100_Modbus".HoldingRegister_MBServer[5]
40007	DB100.DBW68	"DB100_Modbus".HoldingRegister_MBServer[6]
40008	DB100.DBW70	"DB100_Modbus".HoldingRegister_MBServer[7]
40009	DB100.DBW72	"DB100_Modbus".HoldingRegister_MBServer[8]
40010	DB100.DBW74	"DB100_Modbus".HoldingRegister_MBServer[9]

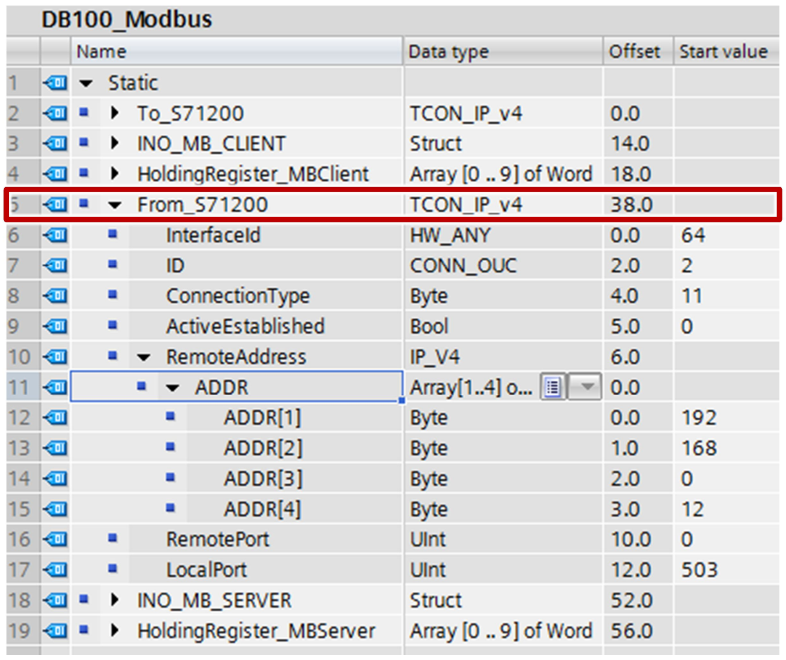
2.2.4 Data Structure at the CONNECT Parameter of the "MB_SERVER" Instruction

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.

The figure below shows the structure of TCON_IP_v4 with the name "From_S71200". You specify this structure at the CONNECT parameter of the "MB_SERVER" instruction. A description of the parameters of the "TCON_IP_v4" structure is available in Table 2-10.

Figure 2-4



DB100_Modbus

	Name	Data type	Offset	Start value
1	Static			
2	To_S71200	TCON_IP_v4	0.0	
3	INO_MB_CLIENT	Struct	14.0	
4	HoldingRegister_MBClient	Array [0 .. 9] of Word	18.0	
5	From_S71200	TCON_IP_v4	38.0	
6	InterfaceId	HW_ANY	0.0	64
7	ID	CONN_OUC	2.0	2
8	ConnectionType	Byte	4.0	11
9	ActiveEstablished	Bool	5.0	0
10	RemoteAddress	IP_V4	6.0	
11	ADDR	Array[1..4] o...	0.0	
12	ADDR[1]	Byte	0.0	192
13	ADDR[2]	Byte	1.0	168
14	ADDR[3]	Byte	2.0	0
15	ADDR[4]	Byte	3.0	12
16	RemotePort	UInt	10.0	0
17	LocalPort	UInt	12.0	503
18	INO_MB_SERVER	Struct	52.0	
19	HoldingRegister_MBServer	Array [0 .. 9] of Word	56.0	

2.3 Structure of "TCON_IP_v4"

Table 2-10 describes the parameters of the "TCON_IP_v4" structure.

Table 2-10

Byte	Parameter	Data type	Description
0..1	InterfaceID	HW_ANY	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.
2..3	ID	CONN_OUC	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" and "MB_SERVER" instructions must use a unique ID.

Byte	Parameter	Data type	Description
4	ConnectionType	BYTE	Connection type Select 11 (decimal) for TCP. Other connection types are not permissible.
5	ActiveEstablished	BOOL	ID for how the connection is established. True: connection established actively False: connection established passively
6..9	RemoteAddress	ARRAY[1..4] of BYTE	IP address of the connection partner.
10..11	RemotePort	UINT	Port number of the remote connection partner (value range: 1 to 49151). • MB_CLIENT: Use the IP port number of the server to which the client establishes a connection and communicates over the TCP/IP protocol. • MB_SERVER: Use the IP port number of the client from which the connection request is to be accepted. If the "MB_SERVER" instruction is to accept connection requests from any remote connection partner, use "0" as the port number.
12..13	LocalPort	UINT	Port number of the local connection partner (value range: 1 to 49151). • MB_CLIENT: - Port numbers: 1 to 49151 - Any port: 0 • MB_SERVER: The number of the IP port defines which IP port is monitored for connection requests of the Modbus client.

3 User Program of the S7-1200 CPU

In the user program of the S7-1200 CPU, the "MB_CLIENT" and "MB_SERVER" instructions are called for each Modbus/TCP connection with a unique ID and separate instance data block. The "MB_CLIENT" and "MB_SERVER" instructions are called each time in a separate function.

As Modbus TCP server the S7-1200 CPU processes the connection request of the Modbus TCP client (S7-1500 CPU) and receives the request to read the holding register.

Table 3-1

ID	Call of the "MB_SERVER" instruction	Instance DB of the "MB_SERVER" instruction	Description
1	FC100 "FC100_MB_SERVER"	DB2 "iDB_MB_SERVER"	Read holding register

As Modbus TCP client the S7-1200 CPU establishes the connection to the Modbus TCP server (S7-1500 CPU) and sends the request to read the holding register.

Table 3-2

ID	Call of the "MB_CLIENT" instruction	Instance DB of the "MB_CLIENT" instruction	Description
2	FC200 "FC200_MB_CLIENT"	DB3 "iDB_MB_CLIENT"	Read holding register

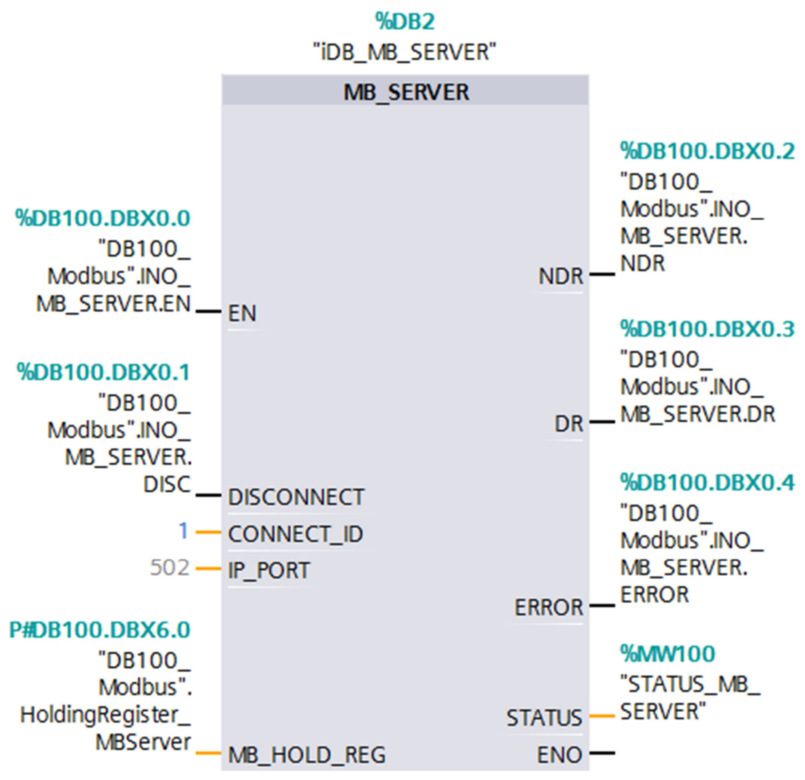
3.1 S7-1200: Modbus TCP Server

3.1.1 FC100 "FC100_MB_SERVER"

The FC200 "FC200_MB_SERVER" 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=1 and Port 502.

Figure 3-1 shows the call and parameters of the "MB_SERVER" instruction in FC100.

Figure 3-1



Note

Section 3.1.2 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 or is to be written to the Modbus 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).

3.1.2 Input and Output Parameters of the "MB_SERVER" Instruction

Input parameters

The "MB_SERVER" has the following input parameters.

Table 3-3

Input parameters	Data type	Description
DISCONNECT	BOOL	The "MB_SERVER" instruction enters into a passive connection with a partner module; this means that the server reacts to a TCP connection request from any requesting IP address. You use this parameter to control when a 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 7003 is output at the STATUS parameter.
CONNECT_ID	UINT	Unique ID to identify the connection. A unique connection ID must be assigned to each instance of the "MB_SERVER" instruction.
IP_PORT	UINT	Start value 502 The number of the IP port defines which IP port is monitored for connection requests of the Modbus TCP client.
MB_HOLD_REG	VARIANT	Pointer to the Modbus holding register of the "MB_SERVER" instruction. Use a global data block with standard access as holding register. The holding register contains the values which a Modbus client is allowed to access over the Modbus functions 3 (read), 6 (write) and 16 (write). Table 3-5 shows how the Modbus addresses are mapped to the holding register for the Modbus function 3 (read WORD).

Output parameters

The "MB_SERVER" instruction has the following output parameters.

Table 3-4

Output parameters	Data type	Description
NDR	BOOL	New Data Ready • 0: No new data. • 1: New data written by the Modbus client
DR	BOOL	Data Read • 0: No new data read. • 1: Data read by the Modbus client
ERROR		If an error occurs while the "MB_SERVER" instruction is being called, the output at the ERROR parameter is set to "1". Detailed information about the cause of error is displayed at the STATUS parameter.

Output parameters	Data type	Description
STATUS	WORD	Detailed Status information of the instruction.

3.1.3 MB_HOLD_REG parameter

The data read from the Modbus TCP server or written to the Modbus TCP server is stored in the data block DB100 "DB100_Modbus".

Table 3-5 shows how the Modbus addresses are mapped to the holding register for the Modbus function 3 (read WORD).

Table 3-5

Modbus address	Absolute address	Symbolic name
40001	DB100.DBW4	"DB100_Modbus".HoldingRegister_MBServer[0]
40002	DB100.DBW6	"DB100_Modbus".HoldingRegister_MBServer[1]
40003	DB100.DBW8	"DB100_Modbus".HoldingRegister_MBServer[2]
40004	DB100.DBW10	"DB100_Modbus".HoldingRegister_MBServer[3]
40005	DB100.DBW12	"DB100_Modbus".HoldingRegister_MBServer[4]
40006	DB100.DBW14	"DB100_Modbus".HoldingRegister_MBServer[5]
40007	DB100.DBW16	"DB100_Modbus".HoldingRegister_MBServer[6]
40008	DB100.DBW18	"DB100_Modbus".HoldingRegister_MBServer[7]
40009	DB100.DBW20	"DB100_Modbus".HoldingRegister_MBServer[8]
40010	DB100.DBW22	"DB100_Modbus".HoldingRegister_MBServer[9]

3.2 S7-1200: Modbus TCP Client

3.2.1 FC200 "FC200_MB_CLIENT"

The FC200 "FC200_MB_CLIENT" 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 two variables:

- "DB100_Modbus".INO_MB_CLIENT.REQ at the input parameter REQ
- "DB100_Modbus".INO_MB_CLIENT.EN at the input parameter EN

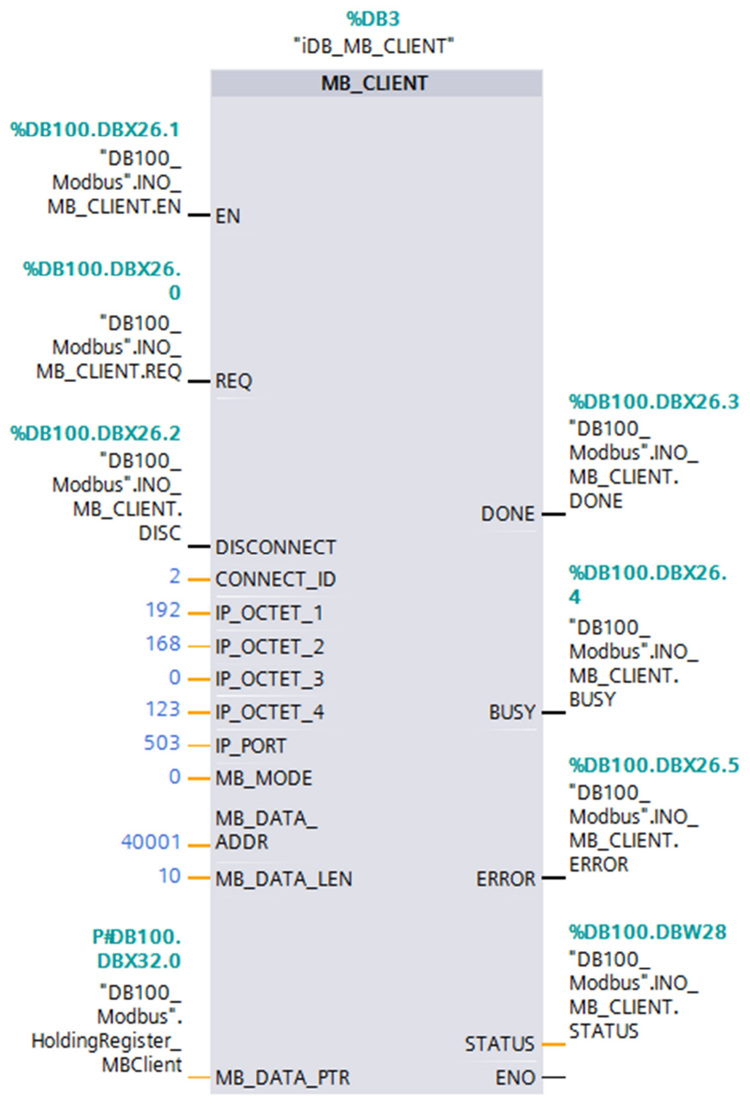
In this example the Modbus/TCP connection with connection number=2 is established to Port 503 of the Modbus/TCP server. The Modbus/TCP server has the IP address 192.168.0.123.

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

Figure 3-2 shows the call and parameters of the "MB_CLIENT" instruction in FC200.

Figure 3-2


%DB3
"iDB_MB_CLIENT"
MB_CLIENT
%DB100.DBX26.1 "DB100_Modbus".INO_MB_CLIENT.EN — EN
%DB100.DBX26.0 "DB100_Modbus".INO_MB_CLIENT.REQ — REQ
%DB100.DBX26.2 "DB100_Modbus".INO_MB_CLIENT.DISC — DISCONNECT
2 — CONNECT_ID
192 — IP_OCTET_1
168 — IP_OCTET_2
0 — IP_OCTET_3
123 — IP_OCTET_4
503 — IP_PORT
0 — MB_MODE
40001 — MB_DATA_ADDR
10 — MB_DATA_LEN
P#DB100.DBX32.0 "DB100_Modbus".HoldingRegister_MBClient — MB_DATA_PTR
DONE — %DB100.DBX26.3 "DB100_Modbus".INO_MB_CLIENT.DONE
BUSY — %DB100.DBX26.4 "DB100_Modbus".INO_MB_CLIENT.BUSY
ERROR — %DB100.DBX26.5 "DB100_Modbus".INO_MB_CLIENT.ERROR
STATUS — %DB100.DBW28 "DB100_Modbus".INO_MB_CLIENT.STATUS
ENO

Note Section 3.2.2 gives an overview and description of the input and output parameters of the "MB_CLIENT" instruction.

3.2.2 Input and Output Parameters of the "MB_CLIENT" Instruction

Input parameters

The "MB_CLIENT" instruction has the following output parameters.

Table 3-6

Input parameters	Data type	Description
REQ	BOOL	Communication request with the Modbus/TCP server on a rising edge. **Note** 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 request is running, no other transfer is made.
DISCONNECT	BOOL	You use the parameters to control the establishment of the connection to and disconnection from the Modbus TCP server. • 0: Establish communication connection to the specified IP address and port number. • 1: Disconnect communication connection. No other function is executed while the connection is being disconnected. After successful disconnection of the connection the value 7003 is output at the STATUS parameter. If the REQ parameter is set when the connection is established, the request is sent immediately.
CONNECT_ID	UINT	Unique ID to identify the connection. A unique connection ID must be assigned to each instance of the "MB_CLIENT" instruction.
IP_OCTET_1	USINT	First octet of the IP address of the Modbus/TCP server.
IP_OCTET_2	USINT	Second octet of the IP address of the Modbus/TCP server.
IP_OCTET_3	USINT	Third octet of the IP address of the Modbus/TCP server.
IP_OCTET_4	USINT	Fourth octet of the IP address of the Modbus/TCP server.
IP_PORT	UINT	IP port number of the server to which the client establishes a connection and communicates over the TCP/IP protocol. (standard value: 502)
MB_MODE	USINT	Selection of the request mode Section gives a detailed description of the MB_MODE parameter.

Input parameters	Data type	Description
MB_DATA_ADDR	UDINT	Initial address of the data which the "MB_CLIENT" instruction accesses. Section _ 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 or to be sent to the Modbus/TCP server. The pointer must refer to a global data block with standard access. In this example the pointer refers to DB100 "DB100_Modbus" (see section 3.2.3).

Output parameters

The "MB_CLIENT" instruction has the following output parameters.

Table 3-7

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

3.2.3 Parameter MB_DATA_PTR

At 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 DB100 "DB100_Modbus" starting with address 28.

Table 3-8

Variable name	Data type	Address in DB100
HoldingRegister_MBClient	Array [0..9] of Word	28.0