VIPA SPEED7

OPL-LIB | SW90xS0MA | Manual

HB00 | OPL-LIB | SW90xS0MA | GB | Rev. 14-15

VIPA
A YASKAWA COMPANY

VIPA GmbH
Ohmstr. 4
91074 Herzogenaurach
Telephone: +49 9132 744-0
Fax: +49 9132 744-1864
email: info@vipa.de
Internet: www.vipa.com

SW90xS0MA_000_OPL-LIB,1,GB - © 2014

Table of contents

1 General **4**
1.1 Copyright © VIPA GmbH 4
1.2 About this manual 5
2 Embedding the VIPA library **7**
2.1 Deployment in the Siemens SIMATIC Manager 7
2.2 Deployment in the Siemens TIA Portal 7
3 Function Blocks **9**
3.1 FB 70 - TCP_MB_CLIENT - Modbus/TCP client 9
3.1.1 Example 11
3.2 FB 71 - TCP_MB_SERVER - Modbus/TCP server 12
3.2.1 Example 15
3.3 FB 72 - RTU_MB_MASTER - Modbus RTU master 16
3.3.1 Example 18
3.4 FB 73 - RTU_MB_SLAVE - Modbus RTU slave 19
3.4.1 Example 23

1 General

1.1 Copyright © VIPA GmbH

All Rights Reserved

Tel.: +49 9132 744 -0

Fax.: +49 9132 744-1864

EMail: info@vipa.de

http://www.vipa.com

Every effort has been made to ensure that the information contained in this document was complete and accurate at the time of publishing. Nevertheless, the authors retain the right to modify the information.

This customer document describes all the hardware units and functions known at the present time. Descriptions may be included for units which are not present at the customer site. The exact scope of delivery is described in the respective purchase contract.

CE Conformity Declaration

Hereby, VIPA GmbH declares that the products and systems are in compliance with the essential requirements and other relevant provisions. Conformity is indicated by the CE marking affixed to the product.

Conformity Information

For more information regarding CE marking and Declaration of Conformity (DoC), please contact your local VIPA customer service organization.

Trademarks

VIPA, SLIO, System 100V, System 200V, System 300V, System 300S, System 400V, System 500S and Commander Compact are registered trademarks of VIPA Gesellschaft für Visualisierung und Prozessautomatisierung mbH.

SPEED7 is a registered trademark of profichip GmbH.

SIMATIC, STEP, SINEC, TIA Portal, S7-300 and S7-400 are registered trademarks of Siemens AG.

Microsoft and Windows are registered trademarks of Microsoft Inc., USA.

Portable Document Format (PDF) and Postscript are registered trademarks of Adobe Systems, Inc.

All other trademarks, logos and service or product marks specified herein are owned by their respective companies.

Information product support

Contact your local VIPA Customer Service Organization representative if you wish to report errors or questions regarding the contents of this document. If you are unable to locate a customer service centre, contact VIPA as follows:

VIPA GmbH, Ohmstraße 4, 91074 Herzogenaurach, Germany

Telefax: +49 9132 744-1204

EMail: documentation@vipa.de

Technical support

Contact your local VIPA Customer Service Organization representative if you encounter problems with the product or have questions regarding the product. If you are unable to locate a customer service centre, contact VIPA as follows:

VIPA GmbH, Ohmstraße 4, 91074 Herzogenaurach, Germany

Tel.: +49 9132 744-1150 (Hotline)

EMail: support@vipa.de

1.2 About this manual

Objective and contents

This manual contains the description of the block library for Modbus RTU and TCP and their installation in the Siemens SIMATIC Manager and TIA Portal:

- SW90AS0MA: VIPA Modbus block library for Siemens SIMATIC Manager
- SW90BS0MA: VIPA Modbus block library for Siemens TIA Portal

Target audience

The manual is targeted at users who have a background in automation technology.

Structure of the manual

The manual consists of chapters. Every chapter provides a self-contained description of a specific topic.

Guide to the document

The following guides are available in the manual:

- An overall table of contents at the beginning of the manual
- References with page numbers

Availability

The manual is available in:

- printed form, on paper
- in electronic form as PDF-file (Adobe Acrobat Reader)

Icons Headings

Important passages in the text are highlighted by following icons and headings:

DANGER!
Immediate or likely danger. Personal injury is possible.

CAUTION!
Damages to property is likely if these warnings are not heeded.

Supplementary information and useful tips.

2 Embedding the VIPA library

Overview

The VIPA specific blocks can be found in the service area of www.vipa.com as library download file at Downloads > VIPA LIB. The libraries are available as packed zip file:

- FX000024_Vxxx.zip: for Siemens SIMATIC Manager
- FX000025_Vxxx.zip: for Siemens TIA Portal

As soon as you want to use VIPA specific blocks you have to import them into your project.

2.1 Deployment in the Siemens SIMATIC Manager

Installing the library

1. Start your un-zip application with a double click on the file FX000024_Vxxx.zip and copy the file VIPA.ZIP to your work directory. It is not necessary to extract this file, too.
2. To retrieve your library start the Siemens SIMATIC Manager.
3. Open the dialog window for archive selection via *'File ➔ Retrieve'*.
4. Navigate to your work directory.
5. Choose VIPA.ZIP and click at [Open].
6. Select a destination folder where the blocks are to be stored.
7. [OK] starts the extraction.
   ⇨ The blocks of the library are now available.

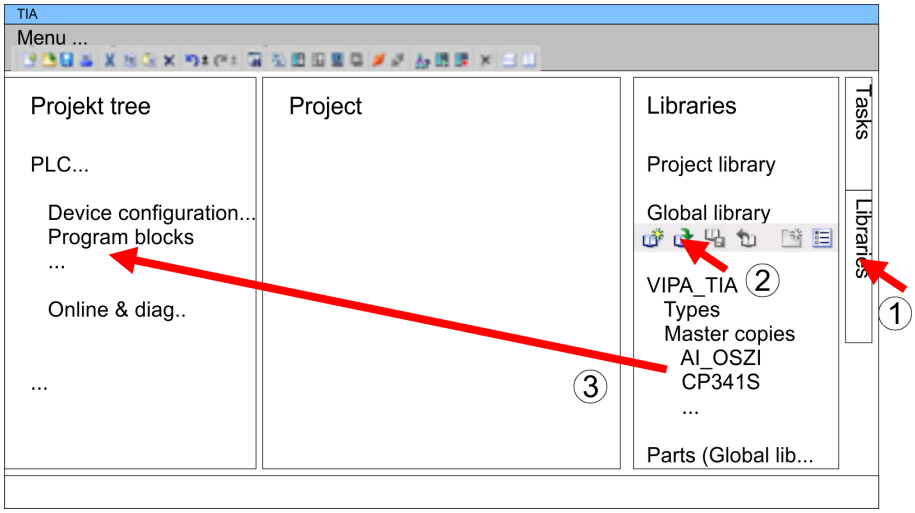
Using the library

1. Open the library after the extraction.
2. Open your project and copy the necessary blocks from the library into the directory "blocks" of your project.
   ⇨ Now you have access to the VIPA specific blocks via your user application.

2.2 Deployment in the Siemens TIA Portal

Open library and transfer blocks to project

1. Start your un-zip application with a double click on the file FX000025_V... .zip and copy all the files and folders in a work directory for the Siemens TIA Portal.
2. Start the Siemens TIA Portal with your project.
3. Select the *'Project view'*.
4. Choose *'Libraries'* from the task cards on the right side.
5. Click at *'Global libraries'*.
6. Click at *'Open global libraries'*.
7. Navigate to your directory and load the file MODBUS.al11.



8. Copy the necessary blocks from the library into the *'Program blocks'* of the *'Project tree'* of your project.

⇨ Now you have access to the blocks via your user application.

3 Function Blocks

In the following the blocks FB 70 ... 73 are described for their use standard blocks are required. More information about the standard blocks may be found in the manual operation list (HB00_OPL_SP7) of your CPU.

3.1 FB 70 - TCP_MB_CLIENT - Modbus/TCP client

Description This function allows the operation of an Ethernet interface as Modbus/TCP client.

Call parameter

Name	Declaration	Type	Description
REQ	IN	BOOL	Start job with edge 0-1.
ID	IN	WORD	ID from TCON.
MB_FUNCTION	IN	BYTE	Modbus: *Function code*.
MB_DATA_ADDR	IN	WORD	Modbus: Start address or *sub function code*.
MB_DATA_LEN	IN	INT	Modbus: Number of register/bits.
MB_DATA_PTR	IN	ANY	Modbus: Data buffer (only flag area or data block of data type byte allowed).
DONE *	OUT	BOOL	Job finished without error.
BUSY	OUT	BOOL	Job is running.
ERROR *	OUT	BOOL	Job is ready with error - parameter STATUS has error information.
STATUS *	OUT	WORD	Extended status and error information.

*) Parameter is available until the next call of the FB

Parameter in instance DB

Name	Declaration	Type	Description
PROTOCOL_TIMEOUT	STAT	INT	Blocking time before an active job can be cancelled by the user. Default: 3s
RCV_TIMEOUT	STAT	INT	Monitoring time for a job. Default: 2s
MB_TRANS_ID	STAT	WORD	Modbus: Start value for the transaction identifier. Default: 1
MB_UNIT_ID	STAT	BYTE	Modbus: Device identification. Default: 255

The following must be observed:

- The *call parameters* must be specified with the block call. Besides the *call parameters* all parameters are located in the instance DB.
- The communication link must be previously initialized via FB 65 (TCON).
- FB 63 (TSEND) and FB 64 (TRCV) are required for the use of the block.
- During a job processing the instance DB is blocked for other clients.
- During job processing changes to the input parameters are not evaluated.
- With the following conditions a job processing is completed or cancelled:
  - DONE = 1 job without error
  - ERROR = 1 job with error
  - Expiration of RCV_TIMEOUT
  - REQ = FALSE after expiration of PROTOCOL_TIMEOUT
- REQ is reset before DONE or ERROR is set or PROTOCOL_TIMEOUT has expired, STATUS 8200h is reported. Here the current job is still processed.

Status and error indication

The function block reports via STATUS the following status and error information.

STATUS	DONE	BUSY	ERROR	Description
0000h	1	0	0	Operation executed without error.
7000h	0	0	0	No connection established or communication error (TCON).
7004h	0	0	0	Connection established and monitored. No job active.
7005h	0	1	0	Data are sent.
7006h	0	1	0	Data are received.
8380h	0	0	1	Received Modbus frame does not have the correct format or has an invalid length.
8381h	0	0	1	Server returns *Exception code 01h*.
8382h	0	0	1	Server returns *Exception code 03h* or wrong start address.
8383h	0	0	1	Server returns *Exception code 02h*.
8384h	0	0	1	Server returns *Exception code 04h*.
8386h	0	0	1	Server returns wrong *Function code*.
8387h	0	0	1	Connection ID (TCON) does not match the instance or server returns wrong protocol ID.
8388h	0	0	1	Server returns wrong value or wrong quantity.
80C8h	0	0	1	No answer of the server during the duration (RCV_TIMEOUT).
8188h	0	0	1	MB_FUNCTION not valid.

STATUS	DONE	BUSY	ERROR	Description
8189h	0	0	1	MB_DATA_ADDR not valid.
818Ah	0	0	1	MB_DATA_LEN not valid.
818Bh	0	0	1	MB_DATA_PTR not valid.
818Ch	0	0	1	BLOCKED_PROC_TIMEOUT or RCV_TIMEOUT not valid.
818Dh	0	0	1	Server returns wrong transaction ID.
8200h	0	0	1	Another Modbus request is processed at the time via the port (PROTOCOL_TIMEOUT).

3.1.1 Example

Task

With *Function code 03h*, starting from address 2000, 100 register are to be read from a Modbus/TCP server and stored in flag area starting from MB200. Errors are to be stored.

OB1

```
CALL  FB    65 , DB65
       REQ:=M100.0
       ID:=1
       DONE:=M100.1
       BUSY:=
       ERROR=
       STATUS:=
       CONNECT:=DB1,

U    M100.1
R    M100.0

CALL  FB    70 , DB70
       REQ:=M101.0
       ID:=1
       MB_FUNCTION :=3
       MB_DATA_ADDR:=2000
       MB_DATA_LEN :=100
       MB_DATA_PTR :=P#M200.0 BYTE 200
       DONE:=M101.1
       BUSY:=
       ERROR:=M101.2
       STATUS:=MW102

L    MW102
UN    M101.2
SPB    ERR
T    MW104
ERR:    NOP 0

U    M101.1
R    M101.0
```

OB1 - Description

1. Calling of FB 65 (TCON) to establish a communication connection with the partner station.
2. Calling the handling block of the Modbus/TCP client with the correct parameters.
3. There is no connection to the partner station and MW102 returns 7000h.

4. Set M100.0 in the CPU to TRUE.
   ⇨ If M100.0 is automatically reset, the connection to the partner station is established and MW102 returns 7004h.
5. Set M101.0 in the CPU to TRUE.
   ⇨ The Modbus request is sent and it is waited for a response.

   If M101.0 is automatically reset, the job was finished without errors and the read data are stored in the CPU starting from bit memory byte 200. MW102 returns 7004h and indicates waiting for a new job.

   If M101.0 is not automatically reset and MW104 returns non-zero, an error has occurred. The cause of error can be read by the code of MW104 (e.g. MW104 = 8382h when the start address 2000 in the server is not available). MW102 returns 7004h and indicates waiting for a new job.

3.2 FB 71 - TCP_MB_SERVER - Modbus/TCP server

Description This function allows the operation of an Ethernet interface as Modbus/TCP server.

Call parameter

Name	Declaration	Type	Description
ENABLE	IN	BOOL	Activation/Deactivation Modbus server.
MB_DATA_PTR	IN	ANY	Modbus: Data buffer (only flag area or data block of type Byte allowed).
ID	IN	WORD	ID from TCON.
NDR *	OUT	BOOL	New data were written by the Modbus client.
DR *	OUT	BOOL	Data were read by the Modbus client.
ERROR *	OUT	BOOL	Job is ready with error - parameter STATUS has error information.
STATUS *	OUT	WORD	Extended status and error information.

*) Parameter is available until the next call of the FB

Parameter in instance DB

Name	Declaration	Type	Description
REQUEST_COUNT	STAT	WORD	Counter for each received frame.
MESSAGE_COUNT	STAT	WORD	Counter for each valid Modbus request.
XMT_RCV_COUNT	STAT	WORD	Counter for each received frame, which contains no valid Modbus request.
EXCEPTION_COUNT	STAT	WORD	Counter for each negatively acknowledged Modbus request.

Name	Declaration	Type	Description
SUCCESS_COUNT	STAT	WORD	Counter for each positively acknowledged Modbus request.
FC1_ADDR_OUTPUT_START	STAT	WORD	Modbus *Function code 01h* start register for Q0.0 Default: 0
FC1_ADDR_OUTPUT_END	STAT	WORD	Modbus *Function code 01h* end register for Qx.y Default: 19999
FC1_ADDR_MEMORY_START	STAT	WORD	Modbus *Function code 01h* start register for M0.0 Default: 20000
FC1_ADDR_MEMORY_END	STAT	WORD	Modbus *Function code 01h* end register for Mx.y Default: 39999
FC2_ADDR_INPUT_START	STAT	WORD	Modbus *Function code 02h* start register for I0.0 Default: 0
FC2_ADDR_INPUT_END	STAT	WORD	Modbus *Function code 02h* end register for Ix.y Default: 19999
FC2_ADDR_MEMORY_START	STAT	WORD	Modbus *Function code 02h* start register for M0.0 Default: 20000
FC2_ADDR_MEMORY_END	STAT	WORD	Modbus *Function code 02h* end register for Mx.y Default: 39999
FC4_ADDR_INPUT_START	STAT	WORD	Modbus *Function code 04h* start register for IW0 Default: 0
FC4_ADDR_INPUT_END	STAT	WORD	Modbus *Function code 04h* end register for IWx Default: 19999
FC4_ADDR_MEMORY_START	STAT	WORD	Modbus *Function code 04h* start register for MW0 Default: 20000
FC4_ADDR_MEMORY_END	STAT	WORD	Modbus *Function code 04h* end register for MWx Default: 39999
FC5_ADDR_OUTPUT_START	STAT	WORD	Modbus *Function code 05h* start register for Q0.0 Default: 0

Name	Declaration	Type	Description
FC5_ADDR_OUTPUT_END	STAT	WORD	Modbus *Function code 05h* end register for Qx.y Default: 19999
FC5_ADDR_MEMORY_START	STAT	WORD	Modbus *Function code 05h* start register for M0.0 Default: 20000
FC5_ADDR_MEMORY_END	STAT	WORD	Modbus *Function code 05h* end register for Mx.y Default: 39999
FC15_ADDR_OUTPUT_START	STAT	WORD	Modbus *Function code 0Fh* start register for Q0.0 Default: 0
FC15_ADDR_OUTPUT_END	STAT	WORD	Modbus *Function code 0Fh* end register for Qx.y Default: 19999
FC15_ADDR_MEMORY_START	STAT	WORD	Modbus *Function code 0Fh* start register for Q0.0 Default: 20000
FC15_ADDR_MEMORY_END	STAT	WORD	Modbus *Function code 0Fh* end register for Qx.y Default: 39999

The following must be observed:

- The *call parameters* must be specified with the block call. Besides the *call parameters* all parameters are located in the instance DB.
- The communication link must be previously initialized via FB 65 (TCON).
- FB 63 (TSEND) and FB 64 (TRCV) are required for the use of the block.
- The INPUT/OUTPUT Modbus addresses of a *Function code* must be located in front of the MEMORY Modbus address and thus always be lower.
- Within a *Function code* no Modbus address may be defined multiple times - also not 0!
- The server can only process one job simultaneously. New Modbus requests during job processing are ignored and not answered.

Status and error indication

The function block reports via STATUS the following status and error information.

STATUS	NDR	DR	ERROR	Description
0000h	0 or 1 *		0	Operation executed without error.
7000h	0	0	0	No connection established or communication error (TCON).
7005h	0	0	0	Data are sent.

STATUS	NDR	DR	ERROR	Description
7006h	0	0	0	Data are received.
8380h	0	0	1	Received Modbus frame does not have the correct format or bytes are missing.
8381h	0	0	1	*Exception code 01h*, *Function code* is not supported.
8382h	0	0	1	*Exception code 03h*, data length or data value are not valid.
8383h	0	0	1	*Exception code 02h*, invalid start address or address range.
8384h	0	0	1	*Exception code 04h*, area length error when accessing inputs, outputs or bit memories.
8387h	0	0	1	Connection ID (TCON) does not match the instance or client returns wrong protocol ID.
8187h	0	0	1	MB_DATA_PTR not valid.

*) Error free Modbus job with *Function code 05h, 06h, 0Fh* or *10h* returns NDR=1 and DR=0. Error free Modbus job with *Function code 01h, 02h, 03h, 04h* return DR=1 and NDR=0.

3.2.1 Example

Task

The CPU provides 100 byte data in the flag area starting from MB200 for a Modbus client via the Modbus register 0...49. Data can be read from the Modbus client via *Function code 03h* and written with *Function code 06h, 10h*. The CPU output Q1.0 is to be controlled by a Modbus client via *Function code 05h* and the start address 5008. Errors are to be stored.

OB1

```
CALL  FB    65 , DB65
       REQ:=M100.0
       ID:=1
       DONE:=M100.1
       BUSY:=
       ERROR=
       STATUS:=
       CONNECT:=DB1,

U    M100.1
R    M100.0

L    5000
T    DB71.DBW   52

CALL  FB    71 , DB71
       ENABLE     :=M101.0
       MB_DATA_PTR:= P#M200.0 BYTE 100
       ID:=1
       NDR:=M101.1
       DR:= M101.2
       ERROR:= M101.3
       STATUS:=MW102

L    MW102
UN    M101.3
SPB    ERR
T    MW104
ERR:    NOP 0
```

```
U     M101.1
S     M106.1
U     M101.2
S     M106.2
```

OB1 - Description

1. Call of FB 65 (TCON) to establish a communication connection with the partner station.
2. Calling the handling block of the Modbus/TCP server with the correct parameters.
3. There is no connection to the partner station and MW102 returns 7000h.
4. Set M100.0 in the CPU to TRUE.
   ⇨ If M100.0 is automatically reset, the connection to the partner station is established and MW102 returns 7006h.
5. The Modbus start register in the process image, which can be reached by *Function code 05h*, may be changed in the example by the parameter FC5_ADDR_OUTPUT_START (word 52 in the instance data block).
6. Set M101.0 in the CPU to TRUE.
   ⇨ The Modbus server now works.
7. The client sends a Modbus request with *Function code 03h* start address 10 and quantity 30.
   ⇨ The server responds with 60 byte starting from MB220. DR is set for one CPU cycle and thus M102.2 is set to 1.
8. The client sends a Modbus request with *Function code 05h* start address 5008 and the value FF00h.
   ⇨ The server acknowledges the request and writes "1" to the output Q1.0. NDR is set for one CPU cycle and thus M102.1 is set to 1.
9. The client sends a Modbus request with *Function code 03h* start address 50 (does not exist) and quantity 1.
   ⇨ The server responds with *Exception code 02h* an sets ERROR/STATUS for one CPU cycle. MW104 returns 8383h

3.3 FB 72 - RTU_MB_MASTER - Modbus RTU master

Description

This function block allows the operation of the internal serial RS485 interface of a CPU from VIPA or a System SLIO CP 040 as Modbus RTU master.

Call parameter

Name	Declaration	Type	Description
REQ	IN	BOOL	Start job with edge 0-1.
HARDWARE	IN	BYTE	1 = System SLIO CP 040 / 2 = VIPA SPEED7 CPU
LADDR	IN	INT	Logical address of the System SLIO CP 040 (parameter is ignored with the VIPA SPEED7 CPU).

Name	Declaration	Type	Description
MB_UNIT_ID	IN	BYTE	Modbus: Device identification = Address of the slave (0 ... 247).
MB_FUNCTION	IN	BYTE	Modbus: *Function code*.
MB_DATA_ADDR	IN	WORD	Modbus: Start address or *Sub function code*.
MB_DATA_LEN	IN	INT	Modbus: Number of register/bits.
MB_DATA_PTR	IN	ANY	Modbus: Data buffer (only flag area or data block of type Byte allowed).
DONE *	OUT	BOOL	Job finished without error.
BUSY	OUT	BOOL	Job is running.
ERROR *	OUT	BOOL	Job is ready with error - parameter STATUS has error information.
STATUS *	OUT	WORD	Extended status and error information.

*) Parameter is available until the next call of the FB

Parameter in instance DB

Name	Declaration	Type	Description
INIT	STAT	BOOL	With an edge 0-1 an synchronous reset is established at the System SLIO CP 040. After a successful reset the bit automatically reset.

The following must be observed:

- The *call parameters* must be specified with the block call. Besides the *call parameters* all parameters are located in the instance DB.
- The interface to be used must be configured before:
  - VIPA System SLIO CP 040: Configuration as “Modbus master RTU” with 60 byte IO-Size in the hardware configuration.
  - Internal serial RS485 interface of a VIPA CPU: Configuration via SFC 216 (SER_CFG) with protocol “Modbus master RTU”.
- FB 60 (SEND) and FB 61 (RECEIVE) are required for the use of the block, even if the internal serial RS485 interface of a CPU from VIPA is used.
- During job processing changes to the input parameters are not evaluated.
- Broadcast request via MB_UNIT_ID = 0 are only accepted for writing functions.
- With the following conditions a job processing is completed or cancelled:
  - DONE = 1 job without error
  - ERROR = 1 job with error
  - Expiration of time-out (parameterization at the interface)

Status and error indication

The function block reports via STATUS the following status and error information.

STATUS	DONE	BUSY	ERROR	Description
0000h	1	0	0	Operation executed without error.
7000h	0	0	0	No connection established or communication error.
7004h	0	0	0	Connection established and monitored. No job active.
7005h	0	1	0	Data are sent.
7006h	0	1	0	Data are received.
8381h	0	0	1	Server returns *Exception code 01h*.
8382h	0	0	1	Server returns *Exception code 03h* or wrong start address.
8383h	0	0	1	Server returns *Exception code 02h*.
8384h	0	0	1	Server returns *Exception code 04h*.
8386h	0	0	1	Server returns wrong *Function code*.
8388h	0	0	1	Server returns wrong value or quantity.
80C8h	0	0	1	No answer of the server during the defined duration (time-out parameterizable via interface).
8188h	0	0	1	MB_FUNCTION not valid.
8189h	0	0	1	MB_DATA_ADDR not valid.
818Ah	0	0	1	MB_DATA_LEN not valid.
818Bh	0	0	1	MB_DATA_PTR not valid.
8201h	0	0	1	HARDWARE not valid.
8202h	0	0	1	MB_UNIT_ID not valid.

3.3.1 Example

Task

With *Function code 03h*, starting from address 2000, 100 register are to be read from a Modbus RTU slave with address 99 and stored in flag area starting from MB200. Errors are to be stored. The Modbus RTU master is realized via the internal serial RS485 interface of a VIPA CPU.

OB100

```
CALL  SFC  216
      Protocol :=B#16#5
      Parameter :=DB10
      Baudrate:=B#16#9
      CharLen:=B#16#3
      Parity:=B#16#2
      StopBits:=B#16#1
      FlowControl:=B#16#1
      RetVal:=MW100
```

OB100 - Description

1. Calling of the SFC 216 (SER_CFG) to configure the internal serial interface of the CPU from VIPA.
2. Protocol: “Modbus Master RTU”, 9600 baud, 8 data bit, 1 stop bit, even parity, no flow control.
3. DB10 has a variable of type WORD with a Modbus time-out (value in ms).

OB1

```
CALL  FB    72 , DB72
       REQ:= M101.0
       HARDWARE:=16#02
       LADDR:=
       MB_UNIT_ID:=16#63
       MB_FUNCTION:=3
       MB_DATA_ADDR:=2000
       MB_DATA_LEN:=100
       MB_DATA_PTR:= P#M200.0 BYTE 200
       DONE:= M101.1
       BUSY:=
       ERROR:= M101.2
       STATUS:=MW102

L    MW102
UN    M101.2
SPB    ERR
T    MW104
ERR:    NOP 0

U    M101.1
R    M101.0
```

OB1 - Description

1. Calling the handling block of the Modbus RTU master with the correct parameters.
2. If the interface was correctly initialized in the OB100, the master can be used and MW102 returns 7004h.
3. Set M101.0 in the CPU to TRUE.

   ⇨ The Modbus request is sent and it is waited for a response.

   If M101.0 is automatically reset, the job was finished without errors and the read data are stored in the CPU starting from bit memory byte 200. MW102 returns 7004h and indicates waiting for a new job.

   If M101.0 is not automatically reset and MW104 returns non-zero, an error has occurred. The cause of error can be read by the code of MW104 (e.g. MW104 = 8382h when the start address 2000 in the server is not available). MW102 returns 7004h and indicates waiting for a new job.

3.4 FB 73 - RTU_MB_SLAVE - Modbus RTU slave

Description

This function block allows the operation of the internal serial RS485 interface of a CPU from VIPA or a System SLIO CP 040 as Modbus RTU slave.

Call parameter

Name	Declaration	Type	Description
ENABLE	IN	BOOL	Activation/Deactivation Modbus server.
HARDWARE	IN	BYTE	1 = System SLIO CP 040 / 2 = VIPA SPEED7 CPU
LADDR	IN	INT	Logical address of the System SLIO CP 040 (parameter is ignored with the VIPA SPEED7 CPU).
MB_UNIT_ID	IN	BYTE	Modbus: Device identification = own address (1 ... 247).
MB_DATA_PTR	IN	ANY	Modbus: Data buffer (only flag area or data block of type Byte allowed).
NDR *	OUT	BOOL	New data were written by the Modbus client.
DR *	OUT	BOOL	Data were read by the Modbus client.
ERROR *	OUT	BOOL	Job is ready with error - parameter STATUS has error information.
STATUS *	OUT	WORD	Extended status and error information.
*) Parameter is available until the next call of the FB			

Parameter in instance DB

Name	Declaration	Type	Description
INIT	STAT	BOOL	With an edge 0-1 an synchronous reset is established at the System SLIO CP 040.
REQUEST_COUNT	STAT	WORD	Counter for each received frame.
MESSAGE_COUNT	STAT	WORD	Counter for each valid Modbus request.
BROADCAST_COUNT	STAT	WORD	Counter for each valid Modbus broadcast request.
EXCEPTION_COUNT	STAT	WORD	Counter for each negatively acknowledged Modbus request.
SUCCESS_COUNT	STAT	WORD	Counter for each positively acknowledged Modbus request.
BAD_CRC_COUNT	STAT	WORD	Counter for each valid Modbus request with CRC error.
FC1_ADDR_OUTPUT_START	STAT	WORD	Modbus *Function code 01h* start register for Q0.0 Default: 0
FC1_ADDR_OUTPUT_END	STAT	WORD	Modbus *Function code 01h* end register for Qx.y Default: 19999

Name	Declaration	Type	Description
FC1_ADDR_MEMORY_START	STAT	WORD	Modbus *Function code 01h* start register for M0.0 Default: 20000
FC1_ADDR_MEMORY_END	STAT	WORD	Modbus *Function code 01h* end register for Mx.y Default: 39999
FC2_ADDR_INPUT_START	STAT	WORD	Modbus *Function code 02h* start register for I0.0 Default: 0
FC2_ADDR_INPUT_END	STAT	WORD	Modbus *Function code 02h* end register for Ix.y Default: 19999
FC2_ADDR_MEMORY_START	STAT	WORD	Modbus *Function code 02h* start register for M0.0 Default: 20000
FC2_ADDR_MEMORY_END	STAT	WORD	Modbus *Function code 02h* end register for Mx.y Default: 39999
FC4_ADDR_INPUT_START	STAT	WORD	Modbus *Function code 04h* start register for IW0 Default: 0
FC4_ADDR_INPUT_END	STAT	WORD	Modbus *Function code 04h* end register for IWx Default: 19999
FC4_ADDR_MEMORY_START	STAT	WORD	Modbus *Function code 04h* start register for MW0 Default: 20000
FC4_ADDR_MEMORY_END	STAT	WORD	Modbus *function-Code 04 h* end register for MW0 Default: 39999
FC5_ADDR_OUTPUT_START	STAT	WORD	Modbus *Function code 05h* start register for Q0.0 Default: 0
FC5_ADDR_OUTPUT_END	STAT	WORD	Modbus *Function code 05h* end register for Qx.y Default: 19999
FC5_ADDR_MEMORY_START	STAT	WORD	Modbus *Function code 05h* start register for M0.0 Default: 20000
FC5_ADDR_MEMORY_END	STAT	WORD	Modbus *Function code 05h* end register for Mx.y Default: 39999

Name	Declaration	Type	Description
FC15_ADDR_OUTPUT_START	STAT	WORD	Modbus *Function code 0Fh* start register for Q0.0 Default: 0
FC15_ADDR_OUTPUT_END	STAT	WORD	Modbus *Function code 0Fh* end register for Qx.y Default: 19999
FC15_ADDR_MEMORY_START	STAT	WORD	Modbus *Function code 0Fh* start register for M0.0 Default: 20000
FC15_ADDR_MEMORY_END	STAT	WORD	Modbus *Function code 0Fh* end register for Mx.y Default: 39999

The following must be observed:

- The *call parameters* must be specified with the block call. Besides the *call parameters* all parameters are located in the instance DB.
- The interface to be used must be configured before:
  - VIPA System SLIO CP 040: Configuration as ASCII module with 60 byte IO-Size in the hardware configuration.
  - internal serial RS485 interface of a VIPA CPU: Configuration via SFC 216 (SER_CFG) with protocol “ASCII”.
- FB 60 (SEND) and FB 61 (RECEIVE) are required for the use of the block, even if the internal serial RS485 interface of a CPU from VIPA is used.
- Broadcast request via MB_UNIT_ID = 0 are only accepted for writing functions.
- The INPUT/OUTPUT Modbus addresses of a *Function code* must be located in front of the MEMORY Modbus address and thus always be lower.
- Within a *Function code* no Modbus address may be defined multiple times - also not 0!
- The slave can only process one job simultaneously. New Modbus requests during job processing are ignored and not answered.

Status and error indication

The function block reports via STATUS the following status and error information.

STATUS	NDR	DR	ERROR	Description
0000h	0 or 1 *		0	Operation executed without error.
7000h	0	0	0	No connection established or communication error.
7005h	0	0	0	Data are sent.
7006h	0	0	0	Data are received.
8380h	0	0	1	CRC error
8381h	0	0	1	*Exception code 01h*, *Function code* is not supported.

STATUS	NDR	DR	ERROR	Description
8382h	0	0	1	*Exception code 03h*, data length or data value are not valid.
8383h	0	0	1	*Exception code 02h*, invalid start address or address range.
8384h	0	0	1	*Exception code 04h*, area length error when accessing inputs, outputs or bit memories.
8187h	0	0	1	MB_DATA_PTR not valid.
8201h	0	0	1	HARDWARE not valid.
8202h	0	0	1	MB_UNIT_ID not valid.
8203 h	0	0	1	

*) Error free Modbus job with *Function code 05h, 06h, 0Fh* or *10h* returns NDR=1 and DR=0. Error free Modbus job with *Function code 01h, 02h, 03h, 04h* return DR=1 and NDR=0.

3.4.1 Example

Task

The CPU provides 100 byte data in the flag area starting from MB200 for a Modbus master via the Modbus register 0 ... 49. Data can be read by the Modbus master via *Function code 03h* and written with *Function code 06h, 10h*. The CPU output Q1.0 is to be controlled by a Modbus master via *Function code 05h* and the start address 5008. Errors are to be stored. The Modbus RTU slave with the address 99 is realized via the internal serial RS485 interface of a VIPA CPU.

OB100

```
CALL  SFC  216
      Protocol :=B#16#1
      Parameter :=DB10
      Baudrate:=B#16#9
      CharLen:=B#16#3
      Parity:=B#16#2
      StopBits:=B#16#1
      FlowControl:=B#16#1
      RetVal:=MW100
```

OB100 - Description

1. Calling of the SFC 216 (SER_CFG) to configure the internal serial interface of the CPU from VIPA.
2. Protocol: “ASCII”, 9600 baud, 8 data bit, 1 stop bit, even parity, no flow control.
3. DB10 has a variable of type WORD and must be passed as “Dummy”.

OB1

```
L    5000
T    DB73.DBW   58

CALL  FB    73 , DB73
       ENABLE:=M101.0
       HARDWARE:=B#16#2
       LADDR :=
       MB_UNIT_ID:=B#16#63
       MB_DATA_PTR:=P#M200.0 BYTE 100
```

```
          NDR:= M101.1
          DR:= M101.2
          ERROR:= M101.3
          STATUS:=MW102

L     MW102
UN     M101.3
SPB     ERR
T     MW104
ERR:     NOP 0

U     M101.1
S     M106.1
U     M101.2
S     M106.2
```

OB1 - Description

1. Calling the handling block of the Modbus/TCP server with the correct parameters.
2. If the interface was correctly initialized in the OB100, the slave can be used and MW102 returns 7006h.
3. The Modbus start register in the process image, which can be reached by *Function code 05h*, may be changed in the example by the parameter FC5_ADDR_OUTPUT_START (word 58 in the instance data block).
4. Set M101.0 in the CPU to TRUE.
   ⇨ The Modbus slave now works.
5. The master sends a Modbus request with *Function code 03h* start address 10 and quantity 30.
   ⇨ The slave responds with 60 byte starting from MB220. DR is set for one CPU cycle and thus M102.2 is set to "1".
6. The master sends a Modbus request with *Function code 05h* start address 5008 and the value FF00h.
   ⇨ The salve acknowledges the request and writes "1" to the output Q1.0. NDR is set for one CPU cycle and thus M102.1 is set to "1".
7. The master sends a Modbus request with *Function code 03h* start address 50 (does not exist!) and quantity 1.
   ⇨ The server responds with *Exception code 02h* and sets ERROR/STATUS for one CPU cycle. MW104 returns 8383h