Systeme Helmholz®
compatible with you

SAS 341 – Modbus Driver

Serial Interface Module for S7-300
with Modbus RTU Master/Slave Driver

800-341-MOD01

Manual

Edition 1 / 02.08.2011 HW1 & FW1.16 and higher

Order number of manual: 900-341-MOD01/en

Systeme Helmholz GmbH ● Hannberger Weg 2 ● D-91091 Grossenseebach ● Germany
Phone: +49 9135 7380-0 ● Fax: +49 9135 7380-110 ● E-mail: info@helmholz.de ● Internet: www.helmholz.com

Systeme Helmholz GmbH

Hannberger Weg 2, 91091 Grossenseebach, Germany

Note:

We have checked the content of this manual for conformity with the hardware and software described. Nevertheless, because deviations cannot be ruled out, we cannot accept any liability for complete conformity. The information in this manual is regularly updated. When using purchased products, please heed the latest version of the manual, which can be viewed in the Internet at www.helmholz.com, from where it can also be downloaded.

Our customers are important to us. We are always glad to receive suggestions for improvement and ideas.

Revision history of this document:

Edition	Date	Revision
1	2.8.2011	1st version

Contents

1 Safety Information

Please observe the safety information given for your own and other people's safety. The safety information indicates possible hazards and provides information about how you can avoid hazardous situations.

The following symbols are used in this manual.

Caution, indicates hazards and sources of error

Gives information

Hazard, general or specific

*Danger of **electric shock***

1.1 General

The SAS 341 module is only used as part of a complete system.

The operator of a machine system is responsible for observing all safety and accident prevention regulations applicable to the application in question.

During configuration, safety and accident prevention rules specific to the application must be observed.

Emergency OFF facilities according to EN 60204 / IEC 204 must remain active in all modes of the machine system. The system must not enter an undefined restart.

Faults occurring in the machine system that can cause damage to property or injury to persons must be prevented by additional external equipment. Such equipment must also ensure entry into a safe state in the event of a fault. Such equipment includes electromechanical safety buttons, mechanical interlocks, etc. (see EN 954-1, risk assessment).

Never execute or initiate safety-related functions using the operator terminal.

1.2 Restriction of access

Only authorized persons must have access to the modules!

The modules are open equipment and must only be installed in electrical equipment rooms, cabinets, or housings. Access to the electrical equipment rooms, barriers, or housings must only be possible using a tool or key and only permitted to personnel having received instruction or authorization. See also Section **Fehler! Verweisquelle konnte nicht gefunden werden.**.

1.3 Information for the user

This manual is addressed to anyone wishing to configure or install the SAS 341 module.

It is intended for use as a programming manual and reference work by the configuring engineer. It provides the installing technician with all the necessary data.

The SAS 341 module is exclusively for use in an S7-300 programmable controller from Siemens. For that reason, the configuring engineer, user, and installing technician must observe the standards, safety and accident prevention rules applicable in the particular application. The operator of the automation system is responsible for observing these rules.

1.4 Use as intended

The SAS 341 module must only be used as a communication system as described in the manual.

1.5 Avoiding use not as intended!

Safety-related functions must not be controlled using the SAS 341 module alone.

2 Installation and Mounting

The SAS 341 module must be installed according to VDE 0100/ IEC 364. Because it is an “OPEN type” module, you must install it in a (switching) cabinet. Ambient temperature: 0 ºC – 60 ºC.

Before you start installation work, all system components must be disconnected from their power source.

*Danger of **electric shock**!*

During installation, application-specific safety and accident prevention rules must be observed.

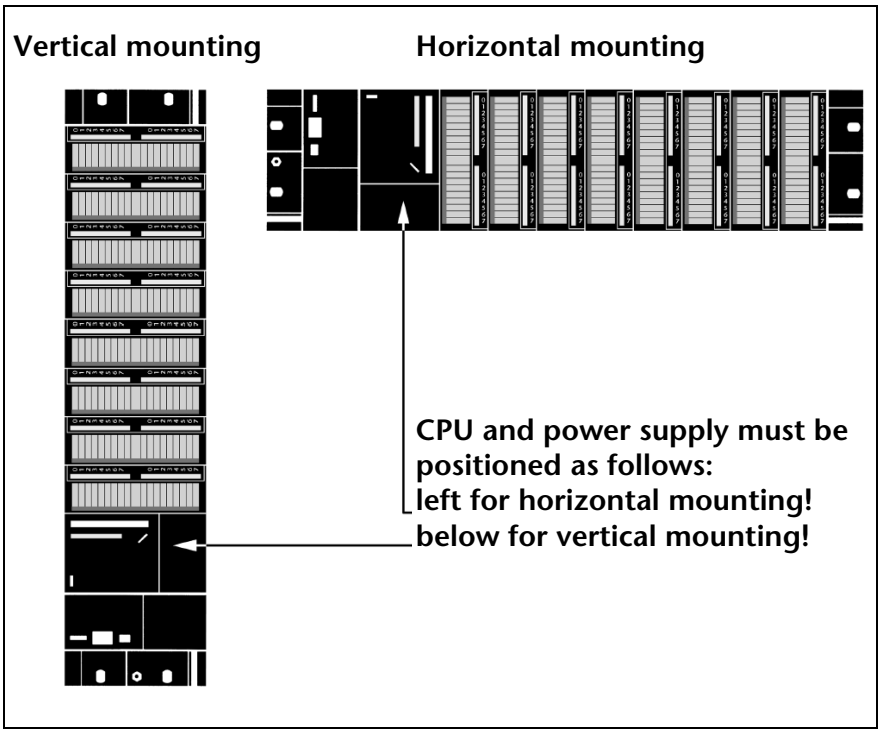
2.1 Vertical and horizontal mounting

The modules can be mounted either vertically or horizontally.

Permissible ambient temperature:

for vertical mounting: from 0 to 40 ºC

for horizontal mounting: from 0 to 60 ºC



2.2 Minimum clearance

Minimum clearances must be observed because

it ensures cooling of the SAS 341 modules

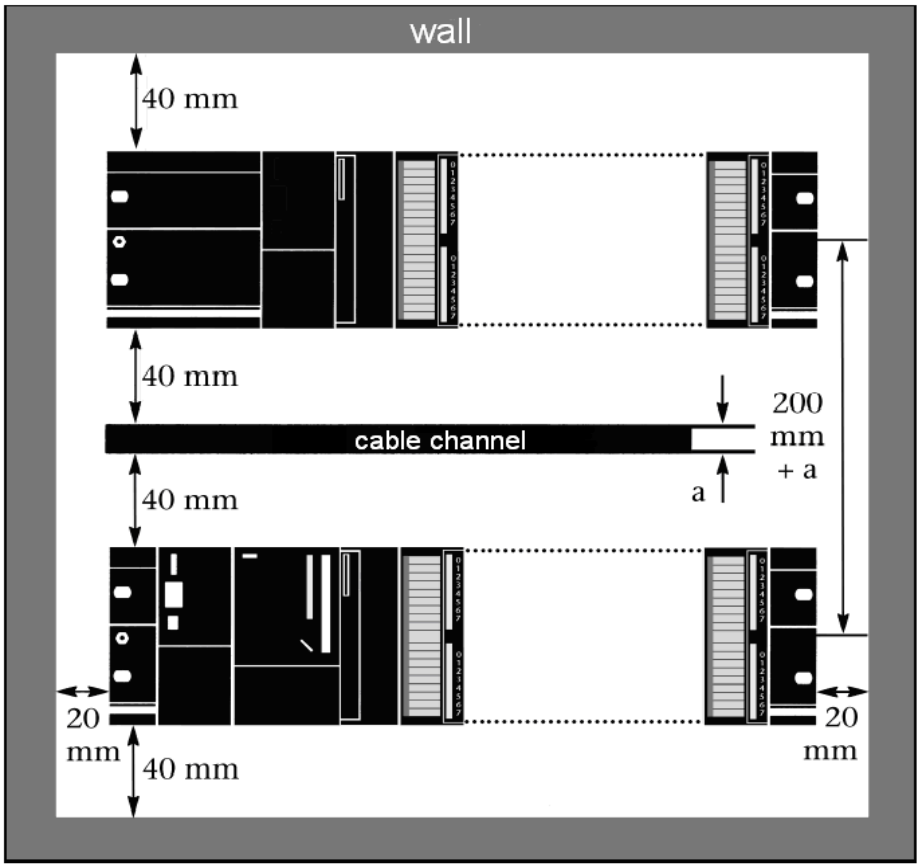
it provides space to insert and remove modules

it provides space to route cables

it increases the mounting height of the module rack to 185 mm, although the minimum spacing of 40 mm must still be observed

The following diagram shows the minimum spacing between the module racks and between these and any adjacent cabinet walls, equipment, cable ducts, etc. for S7-300s mounted in several module racks.

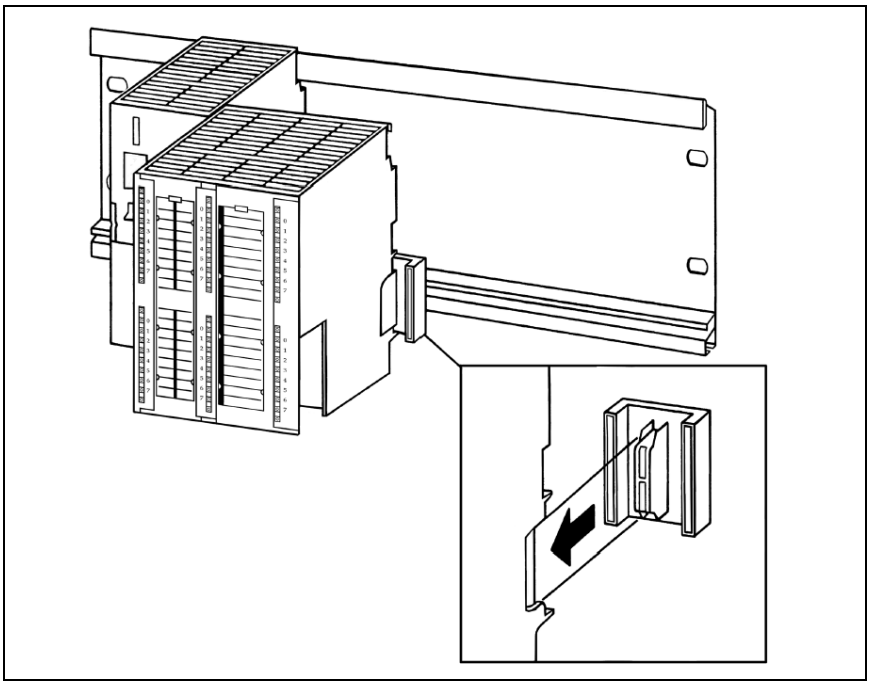
Non-observance of the minimum distances can destroy the module at high ambient temperatures!



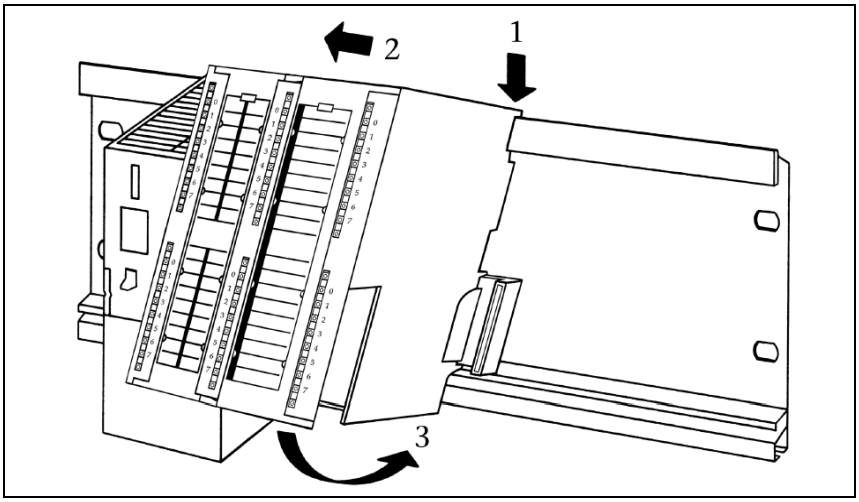
2.3 Mounting of the module on the DIN rail

A bus connector is included with each signal module but not with the CPU. When connecting the bus connector, always start with the CPU.

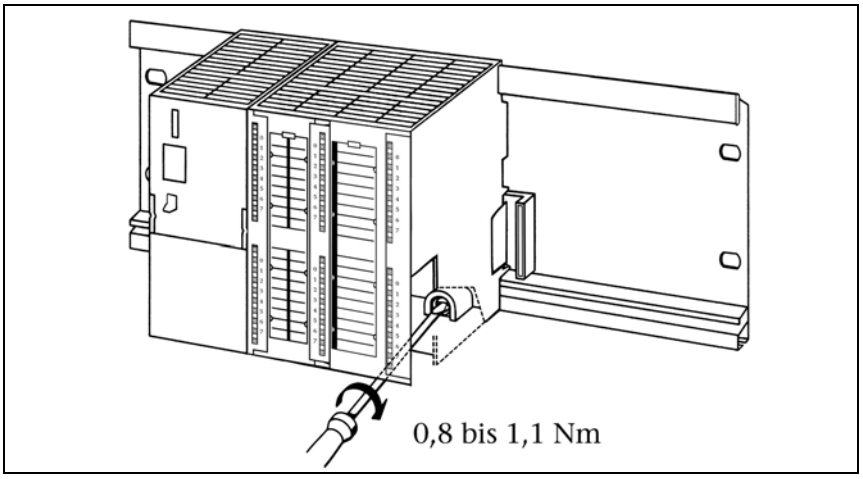
Take the bus connector off the last module and insert it into the CPU. Do not plug a bus connector into the last module of the tier.



Hook on the modules (1), slide them up to the left module, and click them downward (3).



Screw the modules on with a torque of 0.8 to 1.1 Nm.



3 Overview of the System

3.1 Possible uses

The SAS 341 is a communications module for use in Simatic S7-300 systems. The SAS 341 facilitates connection of other manufacturers' serial devices such as barcode scanners, operating terminals, serial printers, PCs or PLSs to the PLC and supports the ASCII, 3964R and RK512 protocols.

Serial devices can be connected via RS232 (V.24), TTY (20 mA) or RS422/RS485. The 9 pin Sub D connector (15 pin with RS422/ RS485) with the standardised layout is for connecting partner's devices.

Parameterisation of the module is performed in the PLC's hardware configurator. The accompanying handling blocks enable a simple and flexible connection to the PLC.

3.2 Modbus Driver

The "Modbus Master/Slave" driver add-on facilitates communication with Modbus RTU capable devices. Using this driver the SAS 341 can work as either a Modbus RTU Master or Modbus RTU Slave.

The driver can be used with a SAS 341-1 with RS232 interface (700-341-1AH02) or with a SAS 341-1 with RS485 interface (700-341-1CH02). Point to point connections can be set up using the RS232 interface and using the RS485 interface, up to 32 users can be addressed in 2-wire half duplex mode.

The driver does not work in SAS 340 or SAS 341 with 2 interfaces. Firmware version (minimum) 1.16 is required for the SAS 341.

When communicating with remote systems the Modbus RTU function codes 01, 02, 03, 04, 05, 06, 07, 08, 11, 12, 15 and 16 are supported.

Data transfer to and from the S7 CPU is handled block-wise via the accompanying function blocks.

3.3 Inserting the driver MMC

The SAS 341 must be powered down when inserting the driver MMC (800-341-MOD01). The Modbus driver is operable as soon as the power is switched on.

The driver MMC must remain permanently in the SAS 341 module.

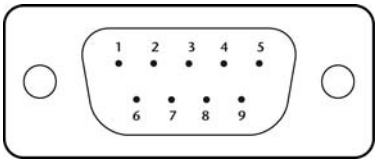
3.4 Interfaces

The SubD connector, located behind the front flap of the SAS 341 module, is either a 9 pin plug for RS232 or a 15 pin socket for RS485. The pin assignments are compatible with the Siemens CP 341 module.

In an SAS 341 with RS232 interface there is also a USB connector which, when the card is being used as Modbus Master/Slave, is not operable.

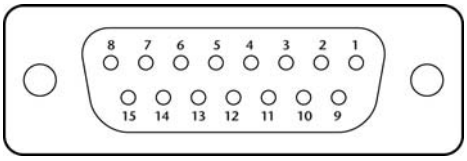
A mini-USB connector is provided for service personnel (update or diagnostic).

3.4.1 SUB D connector RS232 (700-341-1AH02 / -2AH02)



Pin	Designation		Direction	Description
1	DCD	Data Carrier Detect	Input	Carrier signal (modem)
2	RxD	Receive Data	Input	Empfangfleitung
3	TxD	Transmit Data	Output	Transmit line
4	DTR	Data Terminal Ready	Output	ON = SAS is ready
5	GND	Signal ground	-	Zero reference level
6	DSR	Data Set Ready	Input	Communication partner ready?
7	RTS	Request to send	Output	ON = SAS ready to transmit, OFF = nothing to transmit
8	CTS	Clear to send	Input	Communication partner ready to receive?
9	RI	Ring indicator	Input	Ring tone (modem)

3.4.2 SUB D socket RS422/RS485 (700-341-1CH02 / -2CH02)



Pin	Designation	Direction	Description
1	-	-	
2	T (A)	Output	Transmit data (four-wire operation)
3	-	-	
4	R (A) / T (A)	Input / Input/Output	Receive data (four-wire operation) Receive/Transmit data (two-wire operation)
5	-	-	
6	-	-	
7	-	-	
8	GND	-	
9	T (B)	Output	Transmit data (four-wire operation)
10	-	-	
11	R (B) / T B)	Input / Input/Output	Receive data (four-wire operation) Receive/Transmit data (two-wire operation)
12	-	-	
13	-	-	
14	-	-	
15	-	-	

3.5 LED indicators

The LEDs on the front face of the module provide information about the operating status of the SAS 341.

LED "SF'' (orange):
System error due to invalid parameterisation. The message can also be generated by an incorrectly inserted driver MMC.

LED "BF" (red):
This LED indicates an error on the serial interface (e.g. parity, framing, overflow).

LED „RX“ (green):
Receiver active: indicates the reception of characters on the serial interface.

LED „TX“ (orange):
Sender active: indicates the sending of characters on the serial interface.

LED "CPU" (orange):
Data transmission to the PLC active: indicates the transfer of data or commands on the backplane bus (between S7 CPU and module).

LED "ON" (green):
Indicates that the module is being supplied with the correct voltage and the operating system is running.

If the PLC is in Stop, this LED will blink.

4 Modbus Protocol

4.1 Introduction

Modbus is a master/slave protocol and only the master can initiate communications. Each slave has a unique address (1-255). The master can exchange data with an individual slave and can also broadcast data to all slaves.

Using the Modbus protocol, data can be exchanged on either a bit-oriented or register-oriented (16 bit) basis. Data exchange is controlled by function codes.

4.2 Telegram layout Request

Address	Function code	Data	CRC check
Byte	Byte	n bytes	2 bytes

Slave address: 1-255, addresses a specific slave on the bus. Broadcast is represented by sending to address 0. Broadcasts are only allowed for function code 05, 06, 15 and 16. Slaves do not send response telegrams in response to a broadcast.

Function code: Function codes define the meaning of the telegram, for this reason the layout of the telegrams is pre-defined.

Function code	Function according to Modbus	Function in the PLC (at a slave)
01	Read coil status	Read 1…2040 bits F, Q, T, C
02	Read input status	Read 1…2040 bits F, I
03	Read output registers	Read 1…127 words from DB
04	Read input registers	Read 1…127 words from DB
05	Force single coil	Set 1 bit F, O
06	Preset Single Register	Write 1 word in DB
07	Read Exception Status	Read event bits
08	Loopback Test	Retransmit received data
11	Fetch communication event counter	*Not supported by S7 Slave*
12	Fetch communication event log	*Not supported by S7 Slave*
15	Force multiple coils	Write 1…2040 Bits F, Q
16	Preset multiple registers	Write 1…127 words in DB

Data: the specific information and user data is located in byte 2 onwards, dependent on the function code.

CRC check: in order to secure against transmission errors, every telegram is terminated with a 2 byte CRC-16 error check code.

End of telegram marks: the telegram is complete when 3.5x character delay time has expired.

4.3 Telegram layout Slave response

Address	Functioncode	Data	CRC-Check
Byte	Byte	n Bytes	2 Byte

The address and function code are reflected back, the data will be dependent on the function code.

If the Slave detects an error in the request telegram, it answers with an exception response.

Address	Functioncode	Exception code	CRC-Check
Byte	Byte + 80_{hex}	1 Byte	2 Byte

Exception code	Meaning according to Modbus Spec.	Cause
1	Illegal function	Function code is not supported
2	Illegal data address	Destination address is not authorised
3	Illegal data value	Data value not allowed
4	Failure in associated device	Failure in slave, slave not ready
5	Acknowledge	Function is being executed
6	Busy, rejected message	Slave is not ready to receive
7	Negative acknowledgement	Function can not be carried out

If a Slave responds with an exception telegram, this is displayed in the FB 5 – MODB_MAST_RCV (see chapter 7.3.2) as error code 0xE6x (x= exception code).

4.4 Explanation of the function codes

4.4.1 Function code 01 – Read coil status

This function is for reading individual bits from the Slave. If the Slave is an S7 then flags, outputs, times and counters can be read, dependent on the bit start address.

Request telegram Function 01:

Byte	Name	Type	Example	Note
0	Address	Byte	B#16#09	
1	Function code	Byte	B#16#01	
2+3	Bit start address	Word	W#16#0020	not checked when sending
4+5	Bit count	Int	24	1-2040

Response telegram from slave:

Byte	Name	Type	Example	Note
0	Address	Byte	B#16#09	
1	Function code	Byte	B#16#01	
2	Byte counter	Byte	B#16#03	
3	Data byte 1	Byte	B#16#12	
4	Data byte 2	Byte	B#16#34	
5	Data byte 3	Byte	B#16#56	

Data in RCV destination area of PLC

The data is transferred word-wise into the RCV-area and then a byte swap carried out. With odd byte counts a 00h is introduced.

Byte	Name	Type	Example	Note
+0	Data byte 1	Byte	B#16#34	
+1	Data byte 2	Byte	B#16#12	
+2	Data byte 3	Byte	B#16#00	*Fill byte*
+3	Data byte 4	Byte	B#16#56	

4.4.2 Function code 02 – Read input status

This function is for reading individual bits from the Slave. If the Slave is an S7 then flags or inputs can be read, dependent on the bit start address.

Request telegram Function 02:

Byte	Name	Type	Example	Note
0	Address	Byte	B#16#09	
1	Function code	Byte	B#16#02	
2+3	Bit start address	Word	W#16#0120	not checked when sending
4+5	Bit count	Int	16	1-2040

Response telegram from slave:

Byte	Name	Type	Example	Note
0	Address	Byte	B#16#09	
1	Function code	Byte	B#16#02	
2	Byte counter	Byte	B#16#03	
3	Data byte 1	Byte	B#16#12	
4	Data byte 2	Byte	B#16#34	
5	Data byte 3	Byte	B#16#56	

Data in RCV destination area of PLC:

Byte	Name	Type	Example	Note
+0	Data byte 1	Byte	B#16#34	
+1	Data byte 2	Byte	B#16#12	
+2	Data byte 3	Byte	B#16#00	*Fill byte*
+3	Data byte 4	Byte	B#16#56	

The data is transferred word-wise into the RCV-area and then a byte swap carried out. With odd byte counts a 00h is introduced.

4.4.3 Function code 03 – Read output registers

This function is for reading individual registers from the Slave. If the Slave is an S7 then, words can be read from a DB.

Request telegram Function 03:

Byte	Name	Type	Example	Note
0	Address	Byte	B#16#09	
1	Function code	Byte	B#16#03	
2+3	Register start address	Word	W#16#0020	not checked when sending
4+5	Register count	Int	2	1-127

Response telegram from slave:

Byte	Name	Type	Example	Note
0	Address	Byte	B#16#09	
1	Function code	Byte	B#16#03	
2	Byte counter	Byte	B#16#04	
3	Register 1 high	Byte	B#16#12	
4	Register 1 low	Byte	B#16#34	
5	Register 2 high	Byte	B#16#56	
6	Register 2 low	Byte	B#16#78	

Data in RCV destination area of PLC

Byte	Name	Type	Example	Note
+0	Register 1	Word	W#16#1234	
+2	Register 2	Word	W#16#5678	

4.4.4 Function code 04 – Read input registers

This function is for reading individual registers from the Slave. If the Slave is an S7 then words can be read from a DB.

Request telegram Function 04:

Byte	Name	Type	Example	Note
0	Address	Byte	B#16#09	
1	Function code	Byte	B#16#04	
2+3	Register start address	Word	W#16#0020	not checked when sending
4+5	Register count	Int	2	1-127

Response telegram from slave:

Byte	Name	Type	Example	Note
0	Address	Byte	B#16#09	
1	Function code	Byte	B#16#04	
2	Byte counter	Byte	B#16#04	
3	Register 1 high	Byte	B#16#AB	
4	Register 1 low	Byte	B#16#CD	
5	Register 2 high	Byte	B#16#01	
6	Register 2 low	Byte	B#16#23	

Data in RCV destination area of PLC:

Byte	Name	Type	Example	Note
+0	Register 1	Word	W#16#ABCD	
+2	Register 2	Word	W#16#0123	

4.4.5 Function code 05 – Force single coil

This function is for setting or clearing an individual bit. If the Slave is an S7 then a bit in flags, inputs, timers or counters can be set or cleared.

Request telegram Function 05:

Byte	Name	Type	Example	Note
0	Address	Byte	B#16#09	
1	Function code	Byte	B#16#05	
2+3	Bit address	Word	W#16#0017	not checked when sending
4+5	Bit status	WORD	W#16#FF00	FF00h = set bit; 0000h = delete bit

Response telegram from slave:

Byte	Name	Type	Example	Note
0	Address	Byte	B#16#09	
1	Function code	Byte	B#16#04	
2+3	Bit address	Word	W#16#0017	Echo
4+5	Bit status	WORD	W#16#FF00	FF00h = Bit set; 0000h = Bit deleted

No data is stored in the RCV area of the PLC.

4.4.6 Function code 06 – Preset single register

This function is for writing to an individual register. If the Slave is an S7 then a word can be written in a DB.

Request telegram Function 06:

Byte	Name	Type	Example	Note
0	Address	Byte	B#16#09	
1	Function code	Byte	B#16#06	
2+3	Register address	Word	W#16#0140	not checked when sending
4+5	Register value	WORD	W#16#1234	

Response telegram from slave:

Byte	Name	Type	Example	Note
0	Address	Byte	B#16#09	
1	Function code	Byte	B#16#06	
2+3	Register address	Word	W#16#0140	Echo
4+5	Register value	WORD	W#16#1234	Set value

No data is stored in the RCV area of the PLC.

4.4.7 Function code 07 – Read exception status

This function enables the 8 event bits to be read.

Request telegram Function 07:

Byte	Name	Type	Example	Note
0	Address	Byte	B#16#09	
1	Function code	Byte	B#16#07	

Antworttelegramm vom Slave:

Byte	Name	Type	Example	Note
0	Address	Byte	B#16#09	
1	Function code	Byte	B#16#07	
2	Data	Byte	B#16#3E	

Daten im RCV-Zielbereich der SPS:

Byte	Name	Type	Example	Note
+0	Exception Status	Byte	B#16#3E	
+1		Byte	B#16#xx	is not altered

4.4.8 Function code 08 – Loop back diagnostic test

This function is for checking the communication link.

Request telegram Function 08:

Byte	Name	Type	Example	Note
0	Address	Byte	B#16#09	
1	Function code	Byte	B#16#08	
2+3	Diag code	Word	W#16#0000	Only diag code 0000h allowed
4+5	Test value	Word	W#16#A55A	

Antworttelegramm vom Slave:

Byte	Name	Type	Example	Note
0	Address	Byte	B#16#09	
1	Function code	Byte	B#16#08	
2+3	Diag code	Word	W#16#0000	
4+5	Test value	Word	W#16#A55A	

No data is stored in the RCV area of the PLC.

4.4.9 Function code 11 – Fetch communications event counter

This function is for reading the status words and event counters.

Request telegram Function 11:

Byte	Name	Type	Example	Note
0	Address	Byte	B#16#09	
1	Function code	Byte	B#16#0B	

Response telegram from slave:

Byte	Name	Type	Example	Note
0	Address	Byte	B#16#09	
1	Function code	Byte	B#16#0B	
2+3	Status word	Word	W#16#ABCD	
4+5	Event counter	Word	W#16#0123	

Data in RCV destination area of PLC:

Byte	Name	Type	Example	Note
+0	Status word high	Byte	B#16#AB	
+1	Status word low	Byte	B#16#CD	
+2	Event counter high	Byte	B#16#01	
+3	Event counter low	Byte	B#16#23	

4.4.10 Function code 12 – Fetch communications event log

This function is for reading the status words, event counters, the message counters and up to 64 event bytes.

Request telegram Function 12:

Byte	Name	Type	Example	Note
0	Address	Byte	B#16#09	
1	Function code	Byte	B#16#0C	

Response telegram from slave:

Byte	Name	Type	Example	Note
0	Address	Byte	B#16#09	
1	Function code	Byte	B#16#0B	
2	Byte counter	Byte	B#16#04	
3+4	Status-Word	Word	W#16#ABCD	
5+6	Event-Counter	Word	W#16#0123	
7+8	Message-Counter	Word	W#16#0102	
9	Event-Byte 1	Byte		
…				
73	Event-Byte 64	Byte		

Data in RCV destination area of PLC:

Byte	Name	Type	Example	Note
+0	Status-Word High	Byte	B#16#AB	
+1	Status-Word Low	Byte	B#16#CD	
+2	Event-Counter High	Byte	B#16#01	
+3	Event-Counter Low	Byte	B#16#23	
+4	Message-Counter High	Byte	B#16#01	
+5	Message-Counter Low	Byte	B#16#23	
+6	Event-Byte 1	Byte		
…	…			
+69	Event-Byte 64	Byte		

4.4.11 Function code 15 – Force Multiple Coils

This function is for setting bits in the slave. If the Slave is an S7 then, dependent on the bit start address, flags, outputs, times and counters can be written.

The bytes of the send data are reversed byte-wise before being sent. The byte counter is generated automatically and must not be contained in the send data.

Request telegram Function 15:

Byte	Name	Type	Example	Note
0	Address	Byte	B#16#09	
1	Function code	Byte	B#16#0F	
2+3	Bit start address	Word	W#16#0020	not checked when sending
4+5	Bit count	Int	12	1-2040
	Byte counter	*Byte*	*2*	*Is calculated automatically. The byte counter is not contained in the send data of the PLC.*
	Coil status	Word	W#16#1230	

The byte counter is generated automatically and included in the send data. It must not be contained in the send data.

Response telegram from slave:

Byte	Name	Type	Example	Note
0	Address	Byte	B#16#09	
1	Function code	Byte	B#16#0F	
2+3	Bit start address	Word	W#16#0020	
4+5	Bit count	Int	12	

No data is stored in the RCV area of the PLC.

4.4.12 Function code 16 – Preset multiple registers

This function is for setting words in the slave. If the Slave is an S7 then words in a DB can be written.

The byte counter is generated automatically and must not be included in the send data.

Request telegram Function 16:

Byte	Name	Type	Example	Note
0	Address	Byte	B#16#09	
1	Function code	Byte	B#16#10	
2+3	Register start	Word	W#16#0002	not checked when sending
4+5	Register count	Int	2	1-127
	Byte counter	*Byte*	*4*	*Is calculated automatically. The byte counter is not contained in the send data of the PLC.*
	Register 1	Word	W#16#1230	
	Register 2	Word	W#16#1230	

The byte counter is generated automatically and included in the send data. It must not be contained in the send data.

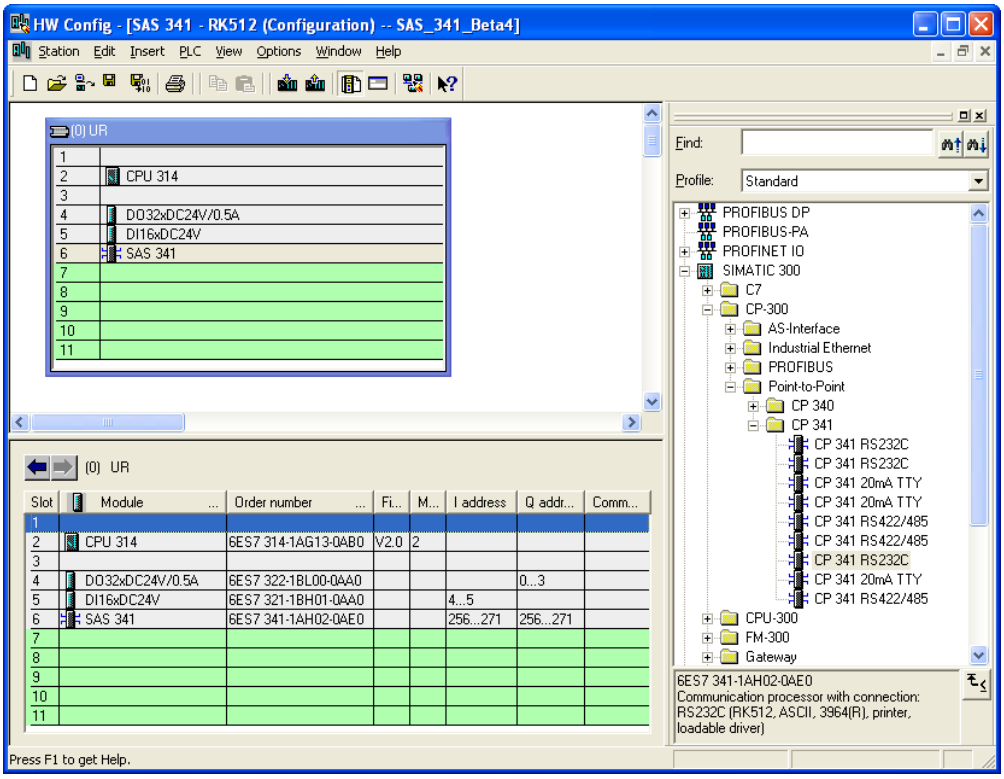
Response telegram from slave:

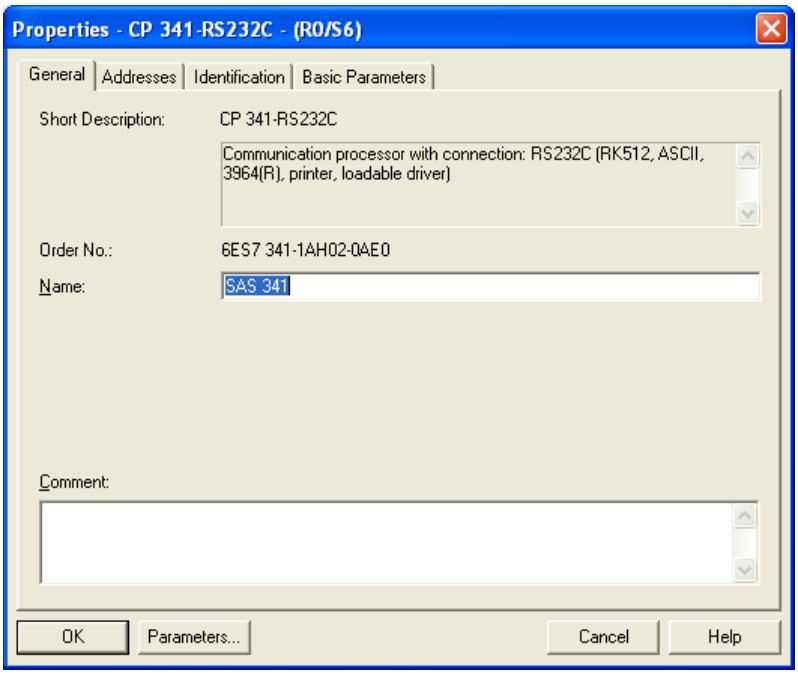
Byte	Name	Type	Example	Note
0	Adresse	Byte	B#16#09	
1	Funktionscode	Byte	B#16#10	
2+3	Register-Start	Word	W#16#0002	
4+5	Register-Anzahl	Int	2	

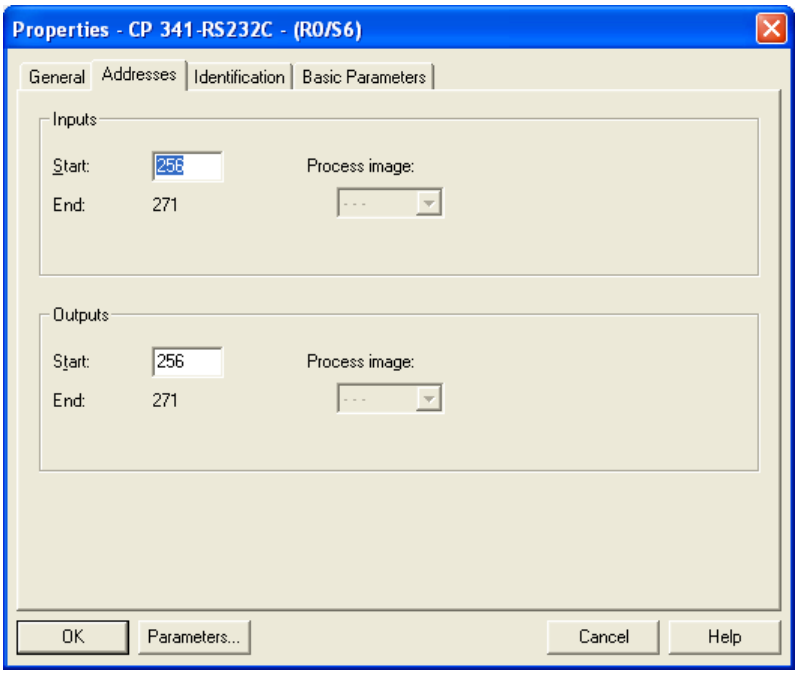
No data is stored in the RCV area of the PLC.

5 Configuring the module

The SAS 341 module is configured in the programming software of the PLC as a CP 341 communication module (6ES7-341-1AH02, -1BH02, -1CH02). Use the same CP modules for the SAS 341-2.





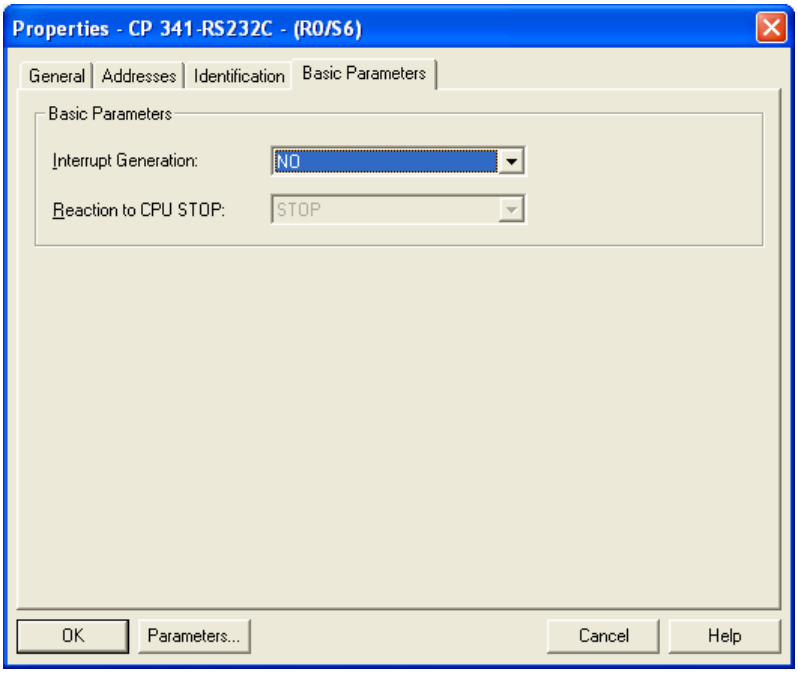


Only the input addresses are used in the data handling blocks; the output addresses have no relevance to the function.

Access to the input I/Os can only be performed with the I/O direct access commands: L PIB, L PIW.

In the case of the CPU 318, the I/O addresses must be outside the cyclic process image.

The settings on the “Identification” and “Basic Parameters” tab cards have no meaning as these functions are not supported.



6 Parameterisation of the Module

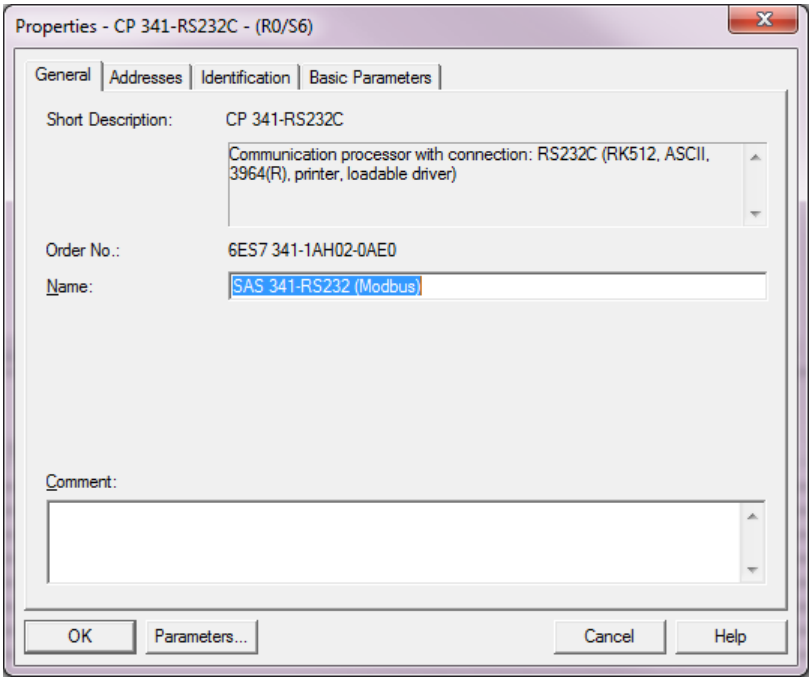
6.1 Installing the parameterisation software

For setting further parameters of the procedure the parameter interface "CP341: Parameterise Point-to-Point Communication" (PtP Param V5.1 from SP10) in Step 7 (from V5.3) is required which must be installed afterwards.

The software can be downloaded from Siemens:

http://support.automation.siemens.com/WW/view/de/27013524

If the software is installed the "Properties" dialog "Parameter ..." button will be active and the module can then be parameterised.



Another parameter software module is required for Modbus parameterisation:

Modbus Master (from Version 3.1 + SP6):

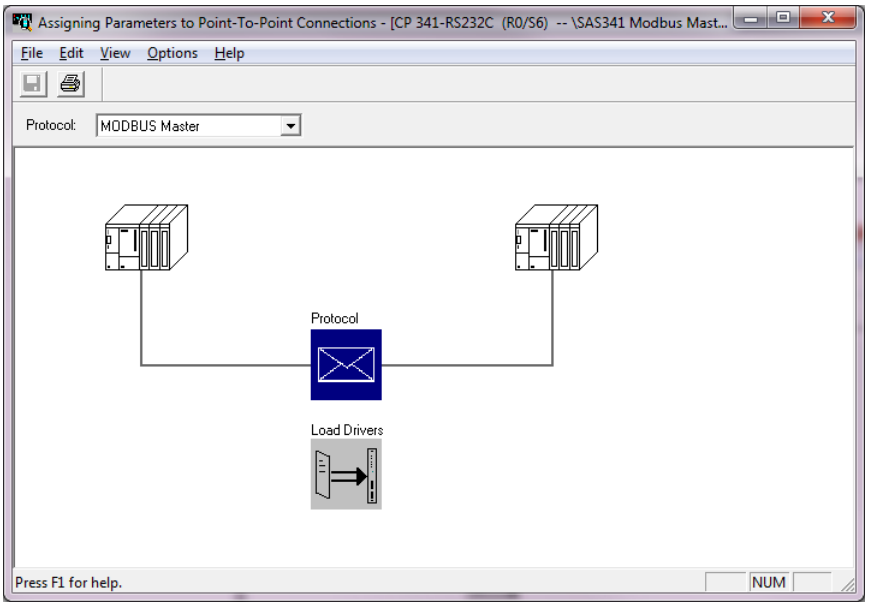
http://support.automation.siemens.com/WW/view/de/27774018

Modbus-Slave (from Version 3.1 + SP7):

http://support.automation.siemens.com/WW/view/de/27774276

The corresponding software must be downloaded and installed. Both can also be installed (Master / Slave).

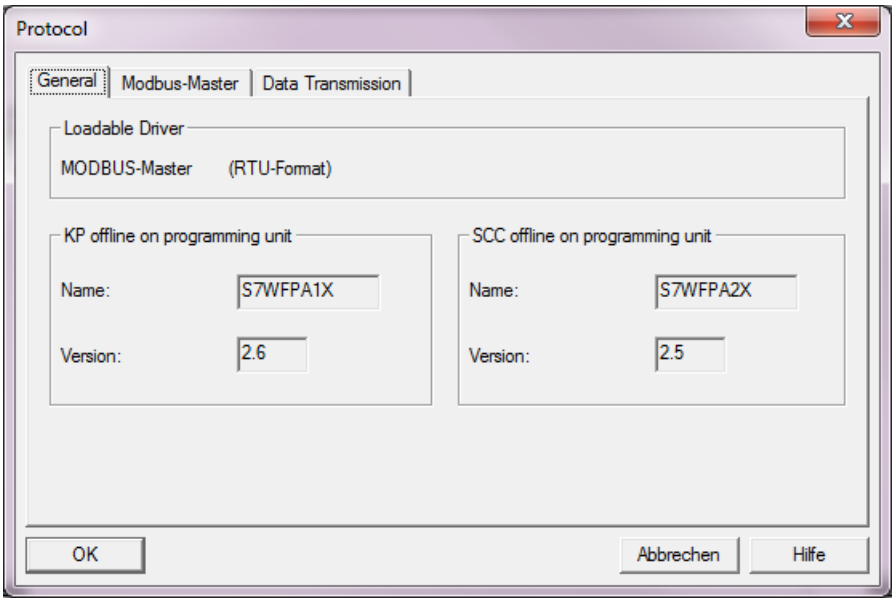
On pressing the "Parameter…“ button the standalone parameterisation GUI is called up.

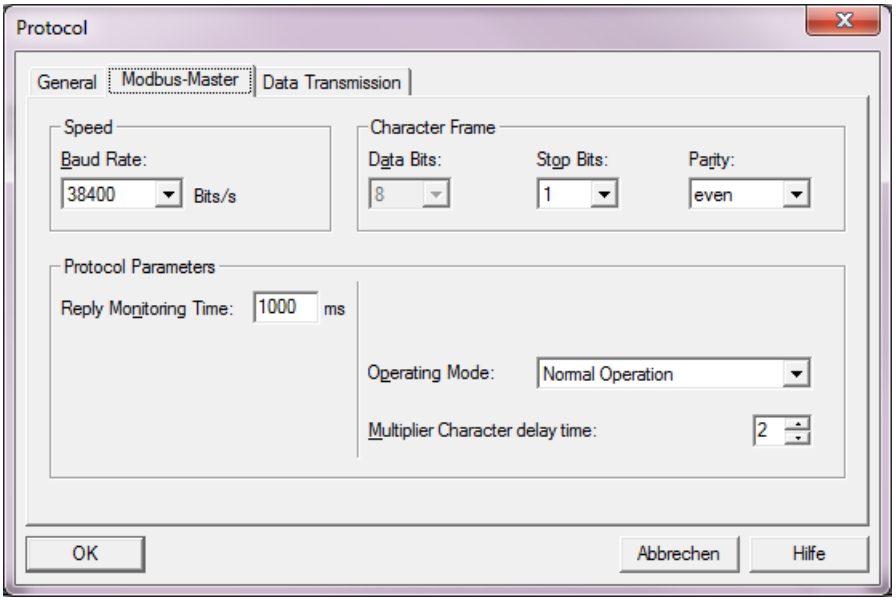


The "Load Driver" function is not relevant for the SAS 341; both Modbus drivers (Master & Slave) are automatically released by the Modbus driver MMC.

6.2 Modbus Master

The first dialog tab "General" is only for information. The "KP Offline on the PG" and "SCC Offline on the PG" are not relevant for the SAS 341.





Speed/character framing: define the transfer speed (300baud to 115Kbaud) and the character framing (stop bits/parity).

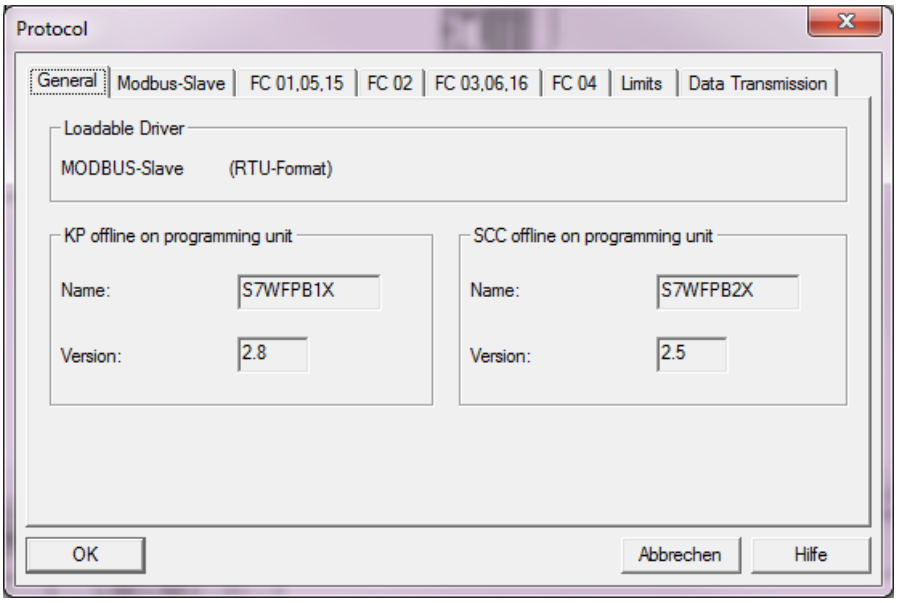
Response monitoring time (5 to 65500 ms): maximum waiting time between sending a request telegram and receiving the response telegram from a Modbus slave.

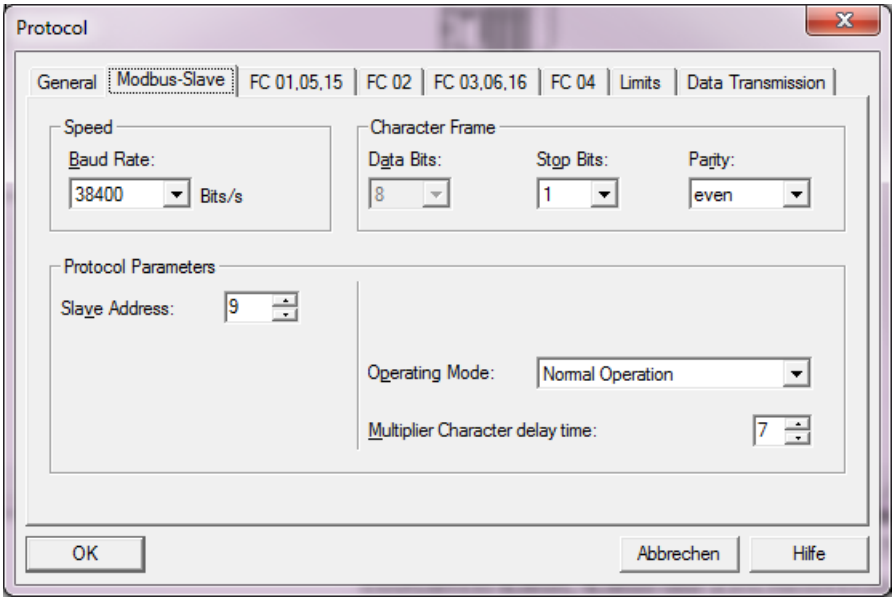
Operating Mode:
Normal Operation = transfer errors result in error messages.
Interferrence Suppression = transfer errors are ignored.

Multiplier Character delay time (1 to 10): if a Slave cannot comply with the timing requirements of the Modbus specification the character delay time can be multiplied by this multiplication factor.

6.3 Modbus Slave

The first dialog tab "General" is only for information. The "KP Offline on the PG" and "SCC Offline on the PG" are not relevant for the SAS 341.



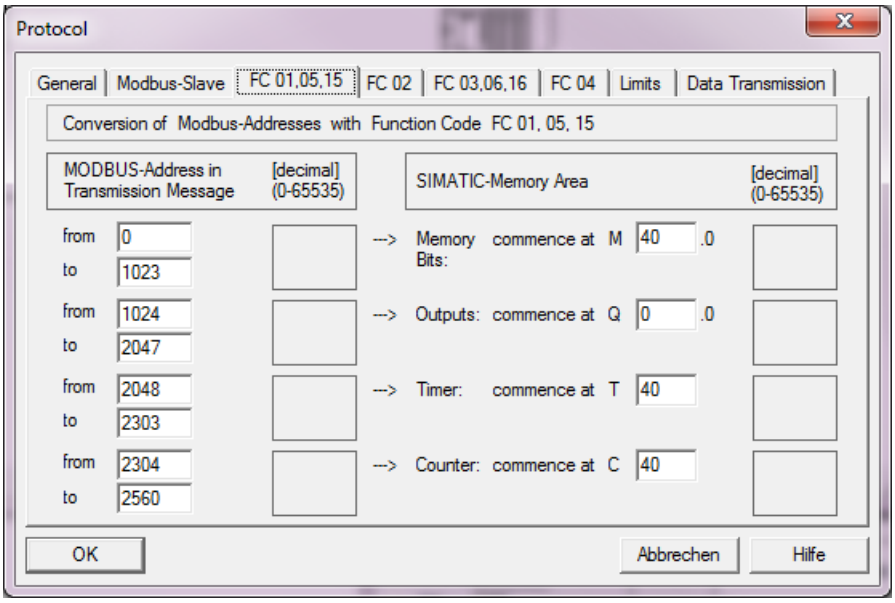


Speed/character framing: define the transfer speed (300baud to 115Kbaud) and the character framing (stop bits/parity).

Slave Address (1 to 255): address of the Modbus user

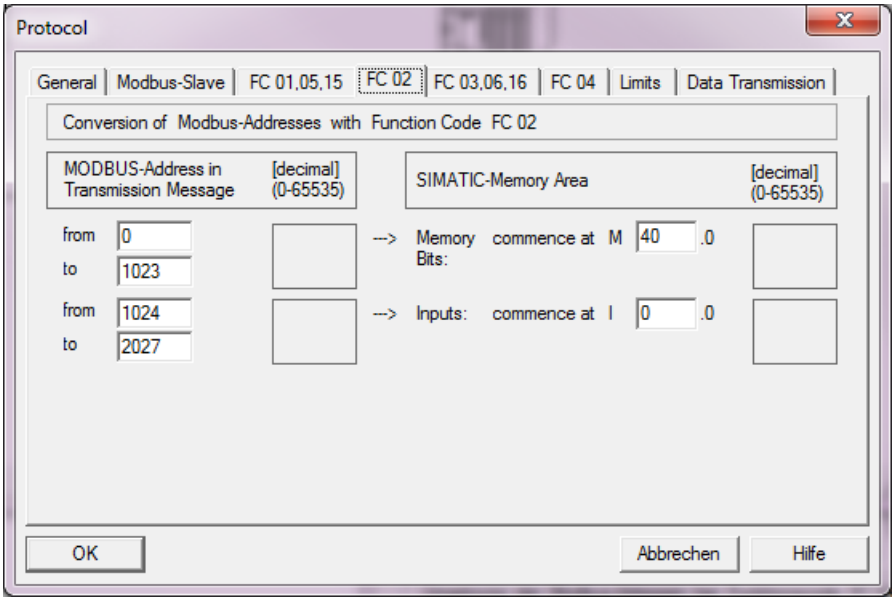
Operating Mode:
Normal Operation = transfer errors result in error messages.
Interferrence Suppression = transfer errors are ignored.

Multiplier Character delay time (1 to 10): if a Slave cannot comply with the timing requirements of the Modbus specification the character delay time can be multiplied by a factor.



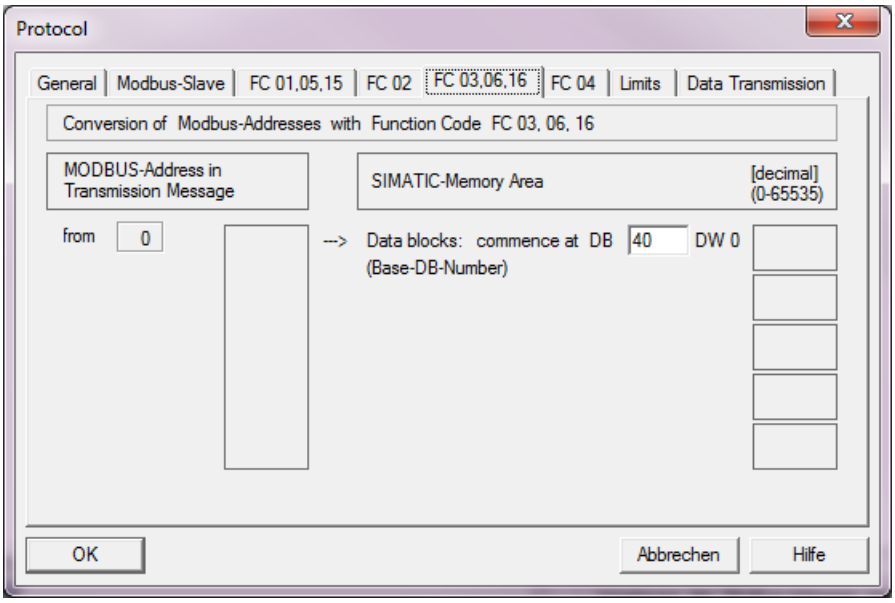
For function codes 1, 5 and 15 (Read Coil Status, Force Single Coil, Force Multiple Coils) the corresponding coil addresses (bit addresses) can be mapped onto flags, outputs, times and counters.

The individual areas must not overlap but holes are allowed. If 0 is entered in "from" and "to" then no mappings are created.

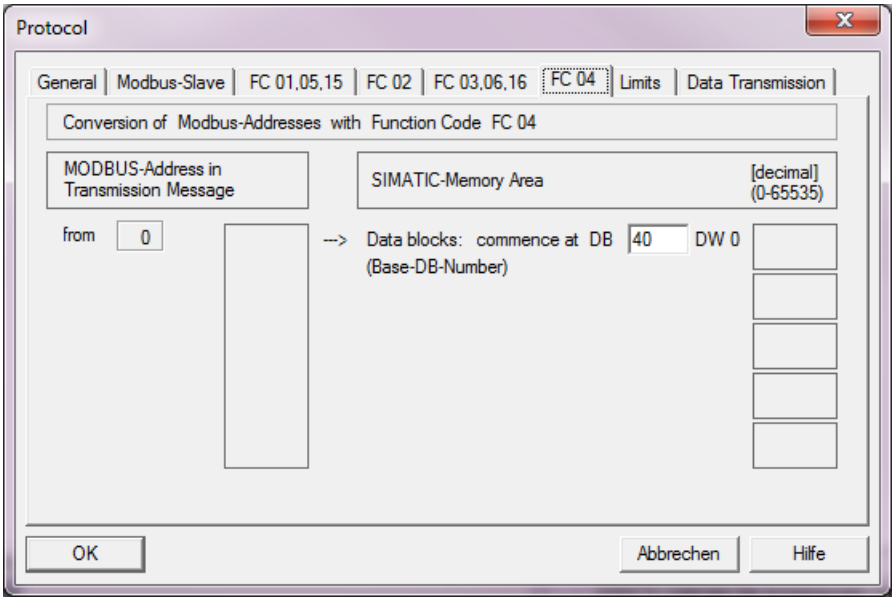


For function code 2 (Read Input Status) input addresses can be mapped onto flags or inputs.

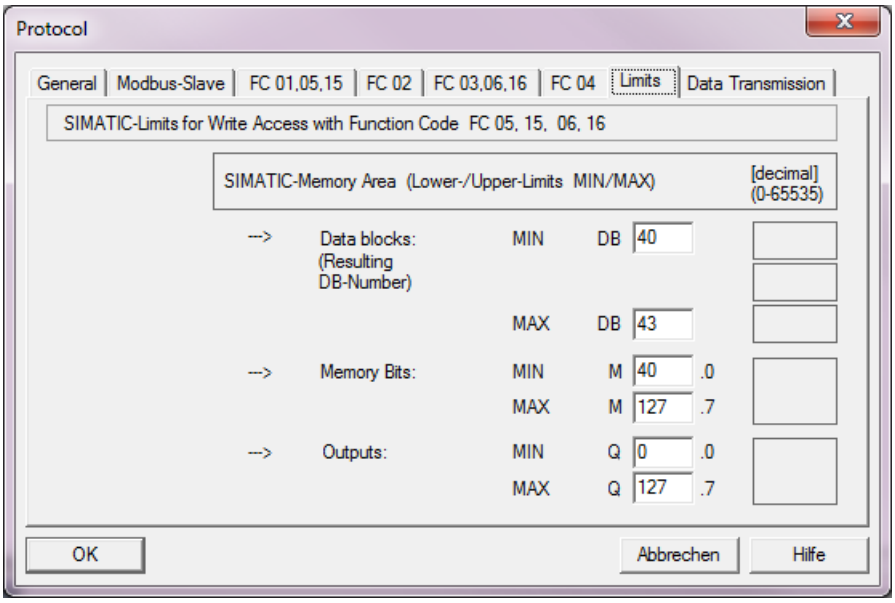
The individual areas must not overlap but holes are allowed. If 0 is entered in "from" and "to" then no mappings are created.



For the function codes 3, 6 and 16 (Read Holding Registers, Preset Single Register, Preset Multiple Registers) register memory can be mapped to data modules. The first register 0 maps onto the defined DB in the DBW 0. It is accessed word-wise. From register address 512, 1024 etc. the following DB is accessed.



For the function code 4 (Read Input Registers) register memory can be mapped to data modules. The first register 0 maps onto the defined DB in the DBW 0. It is accessed word-wise. From register address 512, 1024 etc. the following DB is accessed.



The function codes that perform writes (5, 15, 6, 16) can have boundaries defined in order to prevent accidental write accesses outside the required areas.

7 Programming in the PLC

7.1 Overview

The SAS 341 module is programmed in the PLC by means of the data handling blocks contained in the software package.

In terms of their functions and call parameters, the data handling blocks are based on the data handling blocks of the Siemens CP341. There are separate data handling blocks for Modbus Master and Modbus Slave.

The following blocks are available for communication as a Modbus Master.

FB 4 **MODB_MAST_SND** Send Modbus Requests

FB 5 **MODB_MAST_RCV** Receive Modbus Requests

The following blocks are available for communication as a Modbus Slave.

FB 80 **MODBUS_SLAVE** Process Modbus Requests as Slave.

7.2 Peripheral data in the PLC

The SAS 341 module occupies 16 bytes in the input and in the output peripheral area of the PLC. The contents of the output area are not used.

The contents of the input area can be employed in the application by the user for information purposes.

Byte	Meaning
0	Module status in general, group error display
1	Status signal channel 1
2	FIFO status channel 1
3	Error bits channel 1
4	Active protocol channel 1
5	Status signal channel 2
6	FIFO status channel 2
7	Error bits channel 2
8	Active protocol channel 2
9	*Reserved*
10	*Reserved*
11…15	*Used internally*

The input profile can only be accessed with the peripheral direct access commands: L PIB, L PIW.

Bytes 5, 6 and 7 only hold useful data in SAS modules with 2 interfaces (SAS 340-2/341-2) and are not relevant in connection with the use of Modbus drivers.

7.2.1 Byte 0: module status

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Always 1 = module is present	0 = SAS 340 1 = SAS 341	0 = 1 Port 1 = 2 Ports	0	0	Collection error channel 2	Collection error channel 1	Module parameterised and running

Bit 0: The SAS 341 module is ready for operation.

Bit 1: Collection error channel 1

Bit 5: Number of channels (0 = 1 channel)

Bit 6: Type identifier of module (1 = SAS 341)

Bit 7: Bit is always set for recognition of the module, providing the SAS 341 has started-up correctly.

7.2.2 Byte 1: Status signal channel 1

Bit 7	Bit 6	Bit 5	Bit 4
0	BREAK recognised (In)	RI Ring Indicator (In)	DCD Data Carrier Detect (In)

Bit 3	Bit 2	Bit 1	Bit 0
DTR (Out)	DSR (Out)	CTS (In)	RTS (Out)

7.2.3 Byte 2: FIFO status Bits channel 1

Bit 7	Bit 6	Bit 5	Bit 4
0	0	Send-FIFO half full	Send-FIFOs empty

Bit 3	Bit 2	Bit 1	Bit 0
0	0	Receive-FIFO half full	Receive-FIFOs empty

7.2.4 Byte 3: error bits channel 1

Bit 7	Bit 6	Bit 5	Bit 4
protocol CRC error	0	Send FIFO overflow	Receive FIFOs overflow

Bit 3	Bit 2	Bit 1	Bit 0
0	Interface overflow	Parity error	Framing error

7.2.5 Byte 4: active protocol channel 1

0x30 = ASCII

0x31 = 3964R

0x20 = RK512

0x40 = Modbus RTU

7.3 Modbus Master Data Handling Blocks

7.3.1 FB 4 MODB_MAST_SND

Modbus requests are sent using the function block FB 4 – MODB_MAST_SND. The completed telegram must previously have been made available in a data area (e.g. a data block). The function should be called repeatedly.

Parameter	Direction	Type	Function
Req	IN	BOOL	Trigger request
R	IN	BOOL	Reset request processing
LADDR	IN	INT	Base address of the SAS 341
Src	IN	ANY	Pointer to data area for send telegram
Len	IN	INT	Length of telegram (in bytes)
Busy	OUT	BOOL	Request processing in progress
Done	OUT	BOOL	Request completed without error
Error	OUT	BOOL	Request completed with error
Status	OUT	INT	Error number with *Error* (see chap. 7.3.3)

The data of the Modbus telegram must first be constructed in the send area, see chapter 4.4.

Exception: The CRC and the byte count are generated by the Modbus driver.

The FB 4 block transfers a Modbus request to the SAS 341 module. If the request could be correctly transferred, **Done** is set for 1 cycle.

The SAS 341 sends the request and waits for the answer from the Modbus Slave. All responses, with and without data and also errors (e.g. Slave does not answer) are processed and displayed by the FB 5 – MODB_MAST_RCV.

A new request should first be sent after a response to the FB 5 – MODB_MAST_RCV has been received (positive or error).

If a broadcast is sent, there will be no response telegram transferred across the bus.

Exception: No responses are generated in response to broadcasts (address 0). After a broadcast has been sent (Bit **Done** set) the next request can be started.

Code example for FB 4:

```
      CALL  FB     4 , DB4
       Req   :=M10.0
       R     :=M10.1
       LADDR :=256
       Src   :=P#DB20.DBX0.0 BYTE 262
       Len   :=MW12
       Busy  :=M10.5
       Done  :=M10.6
       Error :=M10.7
       Status:=MW14

      AN    M     10.7                 // Error ?
      JC    SX
      L     MW    14
      T     MW    16                   // copy Status
      JU    Next

SX:   AN    M     10.6                 // Job Done?
      JC    Next
      R     M     10.0
      L     0
      T     MW    16                   // Clear Status

Next: …
```

7.3.2 FB 5 MODB_MAST_RCV

The FB 5 – MODB_MAST_RCV function receives the response telegram from the slave and makes the data available in the receive area (Pointer **Dest**) if the request contained a data request. The function should be called repeatedly.

Parameter	Richtung	Typ	Funktion
EN_R	IN	BOOL	Read release for data
R	IN	BOOL	Reset the receive function
LADDR	IN	INT	Base address of the SAS 341
Dest	IN	ANY	Pointer to data area for received data
Done	OUT	BOOL	Request completed without error
NDR	OUT	BOOL	New data is available
Error	OUT	BOOL	Request completed with error
Len	OUT	INT	Length of the data (in bytes)
Status	OUT	INT	Error number with *Error* (see chap. 7.3.3)

If a broadcast is sent, there will be no response telegram transferred across the bus. The FB 5 does not display anything!

All responses to the request that was sent with the FB 4 MODB_MAST_SND are processed here. It could be responses without data (**Done** is set for one cycle) and responses with data (**Done**, **NDR** and **Len** are set for one cycle). Error responses are also processed (**Error** and **Status** are set for one cycle).

The function codes that result in data being made available in the Receive area (Pointer **Dest**) are described in the protocol description in chapter 4.4 (→ Data in RCV destination area of PLC“).

Code example for FB 5:

```
RCV:  CALL  FB     5 , DB5
       EN_R  :=M11.0
       R     :=M11.1
       LADDR :=256
       Dest  :=P#DB21.DBX0.0 BYTE 262
       Done  :=M11.5
       NDR   :=M11.6
       Error :=M11.7
       Len   :=MW20
       Status:=MW24

      AN    M     11.7                 // Error occurred?
      JC    RDon
      L     MW    24
      T     MW    26                   // copy errorcode
      JU    End

RDon: AN    M     11.5                 // Job done?
      JC    End

      AN    M     11.6                 // Data received?
      JC    End
      L     MW    20                   // copy length
      T     MW    22
      …                                // process data

End:  …
```

7.3.3 Error numbers Modbus Master driver

Errors in the processing of the functional components are generally indicated by an **Error** bit being set. The cause of the error is indicated in the **Status** output operands

0502 Module not available or not ready
0708 Handshake timeout expired (CTS or XON)
07F0 SRC buffer is smaller than LEN
07F4 SAS 341 send folder still occupied
0806 Character delay time expired
080A Buffer overrun
0830 Response monitoring timer expired
08F0 Incorrect identifier received from SAS (internal error)
08F2 Receive buffer DEST not large enough

0E4x Error in SEND request

0E40 Parameter LEN < 2 bytes
0E41 Parameter LEN < 6 bytes (dependent on function code)
0E42 Function code not allowed / incorrect
0E47 LEN does not agree with Bit count or Register count (with function codes 15 & 16)

0E5x Error in response telegram

0E50 Wrong slave address in response telegram
0E51 Wrong function code in response telegram
0E57 CRC error (or telegram length < 4)

0E6x Error response from Slave / x = Exception code

0E61 Exceptioncode 1: Illegal function
0E62 Exceptioncode 2: Illegal data address
0E63 Exceptioncode 3: Illegal data value
0E64 Exceptioncode 4: Failure in associated device
0E65 Exceptioncode 5: Acknowledge
0E66 Exceptioncode 6: Busy, rejected message
0E67 Exceptioncode 7: Negative acknowledgement

1Exx System error

1E0E Error is RETVAL from SFC58, SFC 59 or SFC 20.The value RETVAL is in the master data module of the addressed FB.

7.4 Modbus Slave Data Handling Blocks

7.4.1 FB 80 MODBUS_SLAVE

The FB 80 – MODBUS_SLAVE function handles the protocol processing of Modbus telegrams received in the SAS 341. The FB 80 uses memory areas defined in the hardware configuration (see chapter 6.3) in order to read or write data, dependent on function code.

Parameter	Richtung	Typ	Funktion
LADDR	IN	INT	Base address of the SAS 341
START_TIMER	IN	T	Timer for start-up monitoring
START_TIME	IN	S5T	Time for start-up monitoring
CP_START_OK	OUT	BOOL	TRUE: start-up carried out without error
CP_START_ERROR	OUT	BOOL	Start-up with error (monitoring time)
CP_NDR	OUT	BOOL	New data was received
RCV_ERROR	OUT	BOOL	Error while getting telegram from the SAS
SND_ERROR	OUT	BOOL	Error while sending telegram to SAS
RCV_STATUS	OUT	WORD	Error number when getting telegram
SND_STATUS	OUT	WORD	Error number when sending telegram
CP_START	IN_OUT	BOOL	Start FB initialisation
MB_ERROR_NR	IN_OUT	WORD	Error number (see 7.4.2)
MB_ERROR_INFO	IN_OUT	WORD	Additional info. for error

The FB 80 should be called repeatedly in the PLC program.

Code example for FB 80:

```
CALL  FB    80 , DB80
 LADDR         :=256
 START_TIMER   :=T10
 START_TIME    :=S5T#3S
 CP_START_OK   :=M10.0
 CP_START_ERROR:=M10.1
 CP_NDR        :=M10.5
 RCV_ERROR     :=M10.2
 SND_ERROR     :=M10.3
 RCV_STATUS    :=MW12
 SND_STATUS    :=MW14
 CP_START      :=M10.4
 MB_ERROR_NR   :=MW16
 MB_ERROR_INFO :=MW18
```

7.4.2 Error numbers Modbus Slave

MB_ERROR_NR & MB_ERROR_INFO:

0 = error free

1 = SFC51 error; RETVAL in MB_ERROR_INFO

2 = timeout initialisation

3 = SFC102 error; RETVAL in MB_ERROR_INFO

12 = register count = 0

13 = max. register count exceeded

14 = storage area flags exceeded

15 = storage area output exceeded

16 = storage area times exceeded

17 = storage area counters exceeded

18 = storage area input exceeded

19 = storage area DB exceeded

20 = storage area DB not available

40 = invalid Modbus function code received

41 = invalid Diagnostic Code received during function code 8

50 = Register access outside the parameterised area.

51 = Times and Counter: register numbers not divisible by 16

52 = Times and Counters: Start register error

60 = Flags: Register access outside the parameterised area

61 = Output: Register access outside the parameterised area

62 = Input: Register access outside the parameterised area

63 = Times: Register access outside the parameterised area

64 = Counters: Register access outside the parameterised area

70 = Access to flags outside boundaries

71 = Access to output outside boundaries

72 = Access to DB outside boundaries

73 = Write access to Times and Counters not allowed

RCV_STATUS & SND_STATUS:

0502	Module not available or not ready
07F0	SRC buffer is smaller than LEN
07F4	SAS 341 send folder still occupied
08F0	Incorrect identifier received from SAS (internal error)
08F2	Receive buffer not large enough
1E0E	Error is RETVAL from SFC58, SFC 59. The value RETVAL is in the master data module of the addressed FB.

8 Appendix

8.1 Technical data

Order number	SAS 341-1, 1xRS232+USB	700-341-1AH02
	SAS 341-1, 1xRS422/485	700-341-1CH02
Dimensions	116 x 40 x 125 mm (LxWxH)	
Weight	Approx. 200g	

Communications interface

Type:	RS 232 (V.24), SubD 9pol. male, RS422/485 (X27), SubD 15pol. male USB 1.1 *(only for 700-341-1AH02 / -2AH02)*
Transmission rate:	300 baud to 115 Kbaud (TTY max. 19200 baud)
Protocols:	ASCII 3964R RK512 Modbus RTU Master/Slave *(with Driver-MMC)*

Service interface (update/diagnostics)

Type:	USB 1.1
Transmission rate:	Fullspeed 12Mbps
Connector:	Mini USB

Power supply

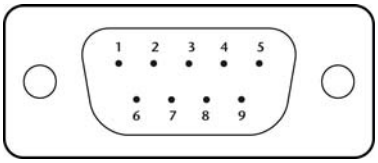
Voltage:	+5V DC via backplane bus
Current consumption:	160mA (typ.) / 190mA (max.)

Special features

Quality assurance:	According to ISO 9001:2008
Maintenance:	Maintenance-free (no battery, rechargeable or non-rechargeable)

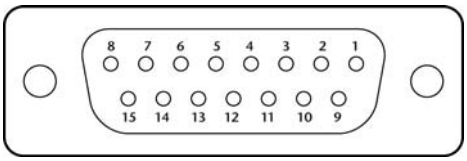
8.2 Pin assignment

8.2.1 SUB D connector RS232 (700-341-1AH02 / -2AH02)



Pin	Designation		Direction	Description
1	DCD	Data Carrier Detect	Input	Carrier signal (modem)
2	RxD	Receive Data	Input	Empfangfleitung
3	TxD	Transmit Data	Output	Transmit line
4	DTR	Data Terminal Ready	Output	ON = SAS is ready
5	GND	Signal ground	-	Zero reference level
6	DSR	Data Set Ready	Input	Communication partner ready?
7	RTS	Request to send	Output	ON = SAS ready to transmit, OFF = nothing to transmit
8	CTS	Clear to send	Input	Communication partner ready to receive?
9	RI	Ring indicator	Input	Ring tone (modem)

8.2.2 SUB D socket RS422/RS485 (700-341-1CH02 / -2CH02)



Pin	Designation	Direction	Description
1	-	-	
2	T (A)	Output	Transmit data (four-wire operation)
3	-	-	
4	R (A) / T (A)	Input / Input/Output	Receive data (four-wire operation) Receive/Transmit data (two-wire operation)
5	-	-	
6	-	-	
7	-	-	
8	GND	-	
9	T (B)	Output	Transmit data (four-wire operation)
10	-	-	
11	R (B) / T B)	Input / Input/Output	Receive data (four-wire operation) Receive/Transmit data (two-wire operation)
12	-	-	
13	-	-	
14	-	-	
15	-	-	

Notes