Sending and Receiving SMS Messages via serial CPs and the MD720-3 GPRS/GSM Modem

SIMATIC S7-300/400/ET 200S, SINAUT MD 720-3

Application Description • February 2013

Applications & Tools

Answers for industry.

SIEMENS

Industry Automation and Drive Technologies Service & Support Portal

This document is taken from the Service Portal of Siemens AG, Industry Automation and Drive Technologies. The following link takes you directly to the download page of this document.

http://support.automation.siemens.com/WW/view/en/25545680

Caution:
The functions and solutions described in this document are restricted to the realization of the automation task. Please note that networking your plant with other plant components, the company network or the internet, requires appropriate protection measures within the framework of Industrial Security. For further information, please refer to the ID number 50203404.

http://support.automation.siemens.com/WW/view/en/50203404.

If you have any questions about this document, please contact us at the following e-mail address:

online-support.automation@siemens.com

You can also browse this subject in our Technical Forum at the Service & Support Portal. Add your questions, suggestions and problems and discuss them with our large forum community:

http://www.siemens.de/forum-applications

SIEMENS

SIMATIC

Sending and Receiving SMS Messages with MD720-3

Warranty and Liability

Note

The application examples are not binding and do not claim to be complete regarding the circuits shown, equipping and any eventuality. The application examples do not represent customer-specific solutions. They are only intended to provide support for typical applications. You are responsible for ensuring that the described products are correctly used. These application examples do not relieve you of the responsibility of safely and professionally using, installing, operating and servicing equipment. When using these application examples, you recognize that Siemens cannot be made liable for any damage/claims beyond the liability clause described. We reserve the right to make changes to these application examples at any time without prior notice. If there are any deviations between the recommendations provided in these application examples and other Siemens publications – e.g. Catalogs – then the contents of the other documents have priority.

We do not accept any liability for the information contained in this document.

Any claims against us – based on whatever legal reason – resulting from the use of the examples, information, programs, engineering and performance data etc. described in this application example shall be excluded. Such an exclusion shall not apply in the case of mandatory liability, e.g. under the German Product Liability Act ("Produkthaftungsgesetz"), in case of intent, gross negligence, or injury of life, body or health, guarantee for the quality of a product, fraudulent concealment of a deficiency or breach of a condition which goes to the root of the contract ("wesentliche Vertragspflichten"). However, claims arising from a breach of a condition which goes to the root of the contract shall be limited to the foreseeable damage which is intrinsic to the contract, unless caused by intent or gross negligence or based on mandatory liability for injury of life, body or health. The above provisions do not imply a change in the burden of proof to your detriment.

Table of Contents

1 Problem

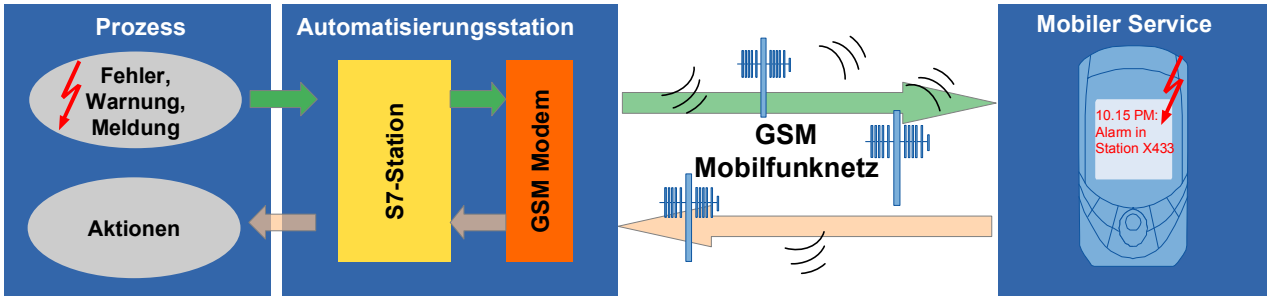
Introduction

In this application, we will show you how you can implement a simple system for wireless signaling and switching based on SMS messaging.

1.1 Overview of the automation problem

The figure below provides an overview of the automation problem.

Figure 1-1



1.2 Description of the automation problem

The aim of this solution is to realize the following scenarios:

An S7-300 CPU is to

- send an SMS message to a configurable recipient
- send an SMS message with an attached coefficient (e.g., an analog process value) to a configurable recipient
- receive an SMS message from a mobile GSM device, evaluate it and initiate a control action
- receive a remote query of a value via SMS.

As an alternative to the S7-300 CPU, an S7-400 CPU or a CPU of the ET 200S distributed I/O system can also be used.

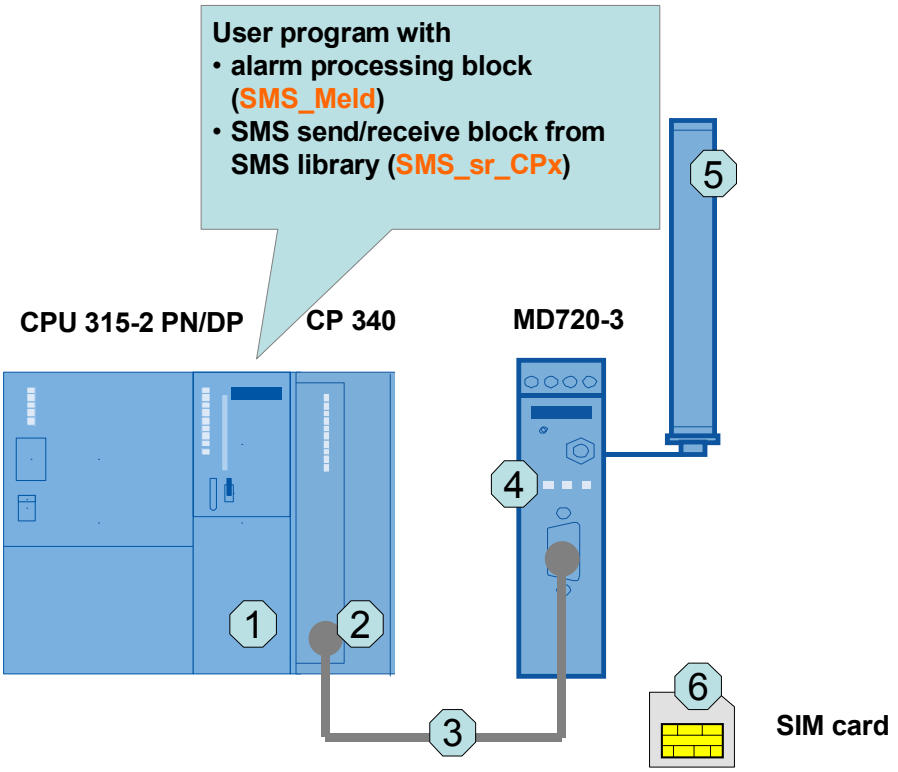
2 Solution

2.1 Overview of the overall solution

Diagrammatic representation

The figure below shows the most important components of the solution with an S7-300 CPU (alternatively, the solution can be implemented with an S7-400 or a CPU of the ET 200S distributed I/O system with SI module – see chapter 4.1):

Figure 2-1



Components included

Table 2-1 Hardware components

No.	Component	Description
1	S7 CPU	An S7-300, S7-400 or an ET 200S station
2	Serial communications processor	Depending on the S7 CPU, the respective CP is connected.
3	Serial cable	CP and GSM modem are connected via a serial cable.
4	GSM modem	MD720-3
5	Quad band antenna	ANT794-4MR
6	SIM card	From the relevant service provider

Table 2-2 Software components

Block	Function	Comment
FB “SMS_Meld”	• Generation of an SMS message according to a predefined logic • Send/receive management with SMS library block • Evaluation of a receive SMS message	Individual user block in SCL
FB “SMS_sr_CPxxx”	Coordinated data exchange between S7 CPU, serial CP and MD720-3	Universal SMS library block in SCL

Scope

This application does not include the basics of

- GSM wireless communications. For more information, refer to document \12\ in the appendix.
- the LAD/ FBD/ STL/ SCL programming languages.

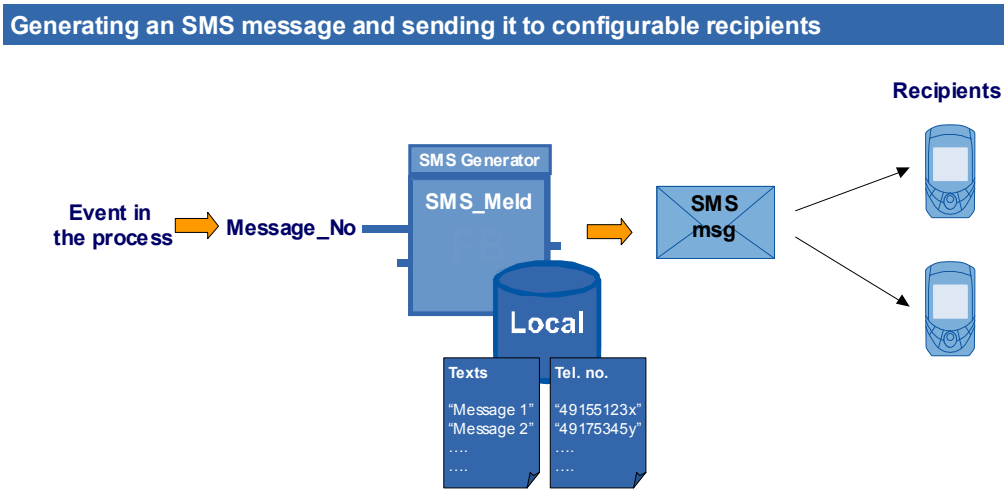
Basic knowledge of these topics is required.

2.2 Description of the core functionality

In this example, the required scenarios are implemented with a user block programmed in SCL (**FB "SMS_MeId"**) and with the aid of a universal SMS library block (**FB "SMS_sr_CPxxx"**).

2.2.1 Scenario: "Generating and sending an SMS message"

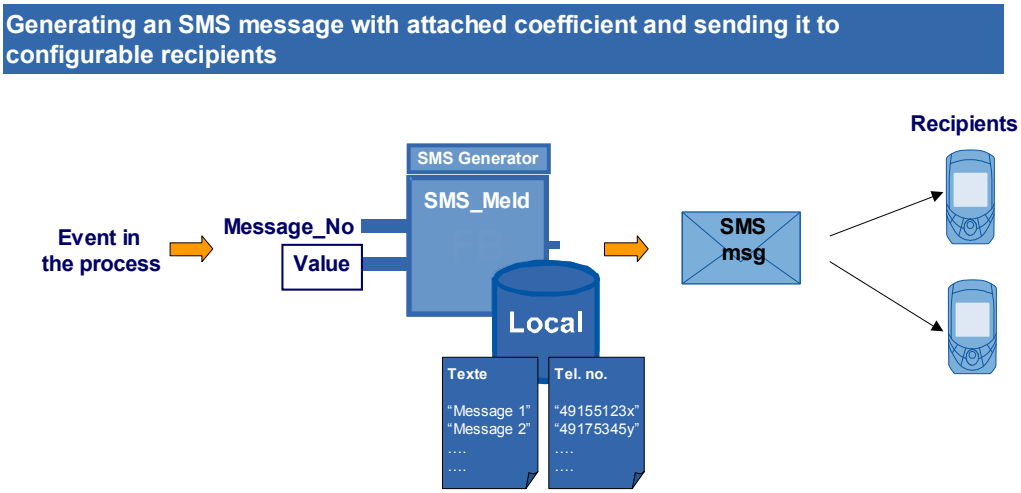
Figure 2-2



When the S7 CPU (e.g., via I/O sensors) detects an event in the process, this event will be assigned an individually defined message number. In the **FB "SMS_MeId"** user block, a text and a recipient are assigned to the message number. The **FB "SMS_MeId"** block coordinates the sending to this recipient.

2.2.2 Scenario: “Sending an SMS message with attached coefficient”

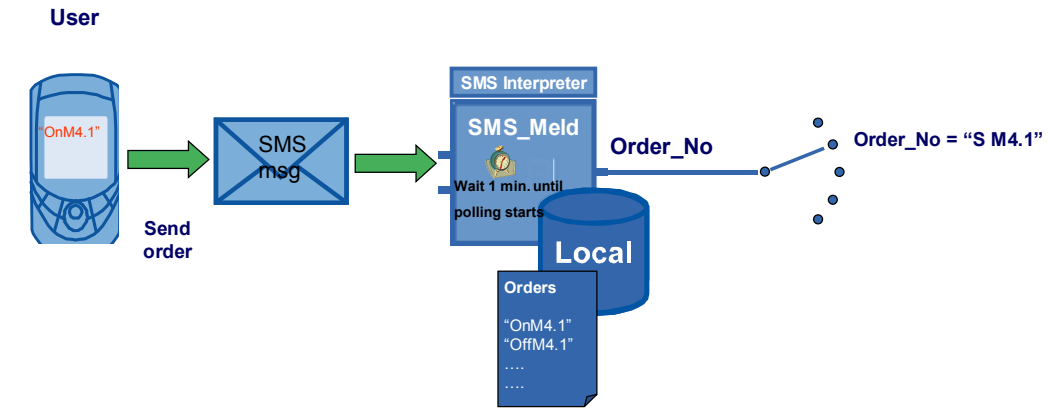
Figure 2-3



When the S7 CPU (for example, via I/O sensors) detects an event in the process, this event will be assigned an individually defined message number and any coefficient (e.g., an analog process value). In the **FB “SMS_MeId”** user block, a text and a recipient are assigned to the message number. The block coordinates the sending of the text with the coefficient to the recipient.

2.2.3 Scenario: “Receiving and evaluating an SMS order”

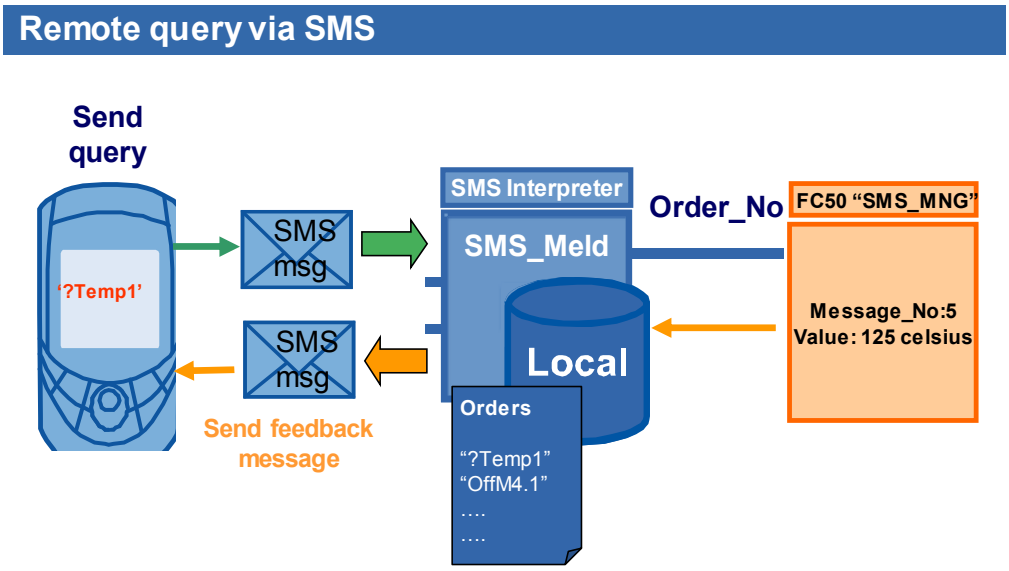
Figure 2-4



When a user sends a defined string to the controller via SMS, the **FB “SMS_MeId”** user block outputs a permanently assigned order number (Order_No) once it has found this order in its configurable order list. The user can then individually use this order number in his/her program to control the process.

2.2.4 Scenario: “Receiving a remote query of a value”

Figure 2-5



In this scenario, the user wants to query a specific value from the controller. To do so, he/she sends a specific command string to the controller by SMS.

The controller replies with the aid of the **FB “SMS_MeId”** user block and the **FC “SMS_MNG”** user function.

FB “SMS_MeId” interprets the receive text of the SMS message and assigns an **Order_No** to the command string. This number is transferred to **FC “SMS_MNG”**, which transfers the desired value and the appropriate **Message_No** back to the **FB “SMS_MeId”** block.
An SMS message is then sent back to the user.

Advantages of these solutions

The solutions presented here offer you the following advantages:

- The code of the **FB “SMS_MeId”** block can be used as a basis and adjusted to your individual signaling or command scenarios (for example, any SMS recipients, any text messages, any command strings, etc.).
- The **FB “SMS_sr_CPxxx”** SMS block for sending and receiving SMS messages used in the solutions is available in a separate library for each serial communications processor (CP 340, CP 341, CP 441-2, ET 200S 1SI).
- The SMS block from the library can be used for your own solutions even without the **FB “SMS_MeId”** block. For more information, please refer to the library description for this block (see Internet link \1\).

2.3 Hardware and software components used

The application was created with the following components:

2.3.1 Hardware for the SIMATIC station

Necessary hardware for the S7-300 station

The following components are necessary if you want to set up the example with an S7-300 station.

Table 2-3

Component	Qty.	MLFB/order number	Note
PS307 5A	1	6ES7307-1EA00-0AA0	
CPU 315-2 PN/DP	1	6ES7315-2EH14-0AB0	
Micro Memory Card	1	6ES7953-8LF11-0AA0	
CP 341	1	6ES7341-1AH01-0AE0	Alternatively, the CP 340 can also be used (6ES7340-1AH02-0AE0)
MPI connecting cable	1	6ES7901-0BF00-0AA0	For loading the CPU

Alternative hardware for the S7-400 solution

The following components are necessary if you want to set up the example with an S7-400 station.

Table 2-4

Component	Qty.	Order number	Note
PS407 10A	1	6ES7407-0KA02-0AA0	
CPU 414-2	1	6ES7414-2XG04-0AB0	
Memory Card	1	6ES7952-1AP00-0AA0	
CP 441-2	1	6ES7441-2AA04-0AE0	
MPI connecting cable	1	6ES7901-0BF00-0AA0	
RS232 interface module	1	6ES7963-1AA00-0AA0	

Alternative hardware for the ET200S solution

The following components are necessary if you want to set up the example with an ET 200S station with SI module.

Table 2-5

Component	Qty.	Order number	Note
PS307 5A	1	6ES7307-1EA00-0AA0	
CPU 315-2 PN/DP	1	6ES7315-2EH14-0AB0	
Micro Memory Card	1	6ES7953-8LF11-0AA0	
IM151-3 PN STANDARD interface module (alternatively IM151-8 PN/DP CPU)	1	6ES7151-3AA20-0AB0 alternatively 6ES7151-3AB01-0AB0	
Industrial Ethernet Twisted Pair cable	2	6XV1 850-2GH60	
PM-E 24VDC	1	6ES7138-4CA01-0AA0	
Bus connector	1	6ES7972-0BA12-0XA0	
ET 200S 1SI 3964/ASCII	1	6ES7138-4DF01-0AB0	
ET 200S, TERM. MOD. TM-E15S24-01	1	6ES7193-4CB20-0AA0	
ET 200S, TERM. MOD. TM-P15S23-A0	1	6ES7193-4CD20-0AA0	

2.3.2 GSM components

Table 2-6

Component	Qty.	Order number	Note
MD720-3	1	6NH9720-3AA00	
GSM antenna	1	6NH9860-1AA00	
SIM card	2		
Serial 9-pin cable	1		

2.3.3 Software components

Table 2-7

Component	Qty.	Order number	Note
STEP 7 V5.5	1	6ES7810-4CC08-0YA5	
S7-SCL V5.3+SP5	1	6ES7811-1CC05-0YA5	
S7-PTP_PARAM	1	This CD is included in the delivery of the serial CPs and contains entries for the STEP 7 hardware configuration and manuals.	

Sample files and projects

The following table contains all files and projects that are used in this example.

Table 2-8

Component	Note
SMS_SR_Library.zip	This zip file contains the library blocks.
SMS_Example.zip	This zip file contains the user program.
25545680_Application_SMS_MD720_DOKU_V2_1_en.pdf	This document.
25545680_Library_SMS_MD720_DOKU_V2_0_en.pdf	Library description.

2.4 Performance data

This chapter provides you with an overview of the performance data of the components and the performance of the application.

Communications processors

This application was tested on the following serial CPs of the SIMATIC S7-300/400/ ET 200S series:

Table 2-9

Module	Driver	Possible baud rates in combination with MD720-3
CP 340 –RS232C	ASCII driver	2400,4800,9600 bps
CP 341 –RS232C	ASCII driver	1200,2400,4800,9600,19200,38400,57600 bps
CP 441-2	RS232 module (ASCII driver)	1200,2400,4800,9600,19200,38400,57600 bps
ET 200S 1SI/ ASCII	ASCII driver	1200,2400,4800,9600,19200,38400,57600 bps

MD720-3

Table 2-10

	Property	Performance data
X1 interface	Standard connection	RS232, jack; D-SUB 9-pin
	Default transmission rate	9600 bps
	Control via	AT commands
	Possible transmission rate	• 1200, 2400, 4800, 9600 • 19200, 38400, 57600 baud
Radio interface	GSM module	GPRS / CSD / quad band
	GPRS	• Up to 2 uplinks • Up to 4 downlinks (max. 5 slots)
	Transmitter power	• GSM 850 MHz (max. 2W) • GSM 900 MHz (max. 2W) • DCS 1800 MHz (max. 1W) • PCS 1900 MHz (max. 1W)

Application software

The following table shows the performance data of the **FB "SMS_MeId"** user block.

Table 2-11

Criterion	Performance data	Note
Transmission rate with ASCII driver	9600 bps	Can be changed in the block.
Number of message texts	20	Can be changed in the block.
Number of telephone numbers	5	Can be changed in the block.
Maximum text length	30 characters	Can be changed in the block.
Maximum order length	10 characters	Can be changed in the block.
Coefficient length	6 characters	A variable of the INTEGER type is used as a coefficient. In the user program, this variable is converted as a STRING.

Measured data/empirical values

The following table shows typical key data and reflects the test environment for the development of this application example. It provides only a rough basis.

Table 2-12

Criterion	Empirical value	Remark
Request time of sending an SMS message	10 sec	Average time from send trigger command to end of send operation.
Number of different stations	Unlimited	
Modem initialization duration	25 sec	Average time from initialization trigger command to end of initialization.

3 Functional Mechanisms of this Application

Introduction

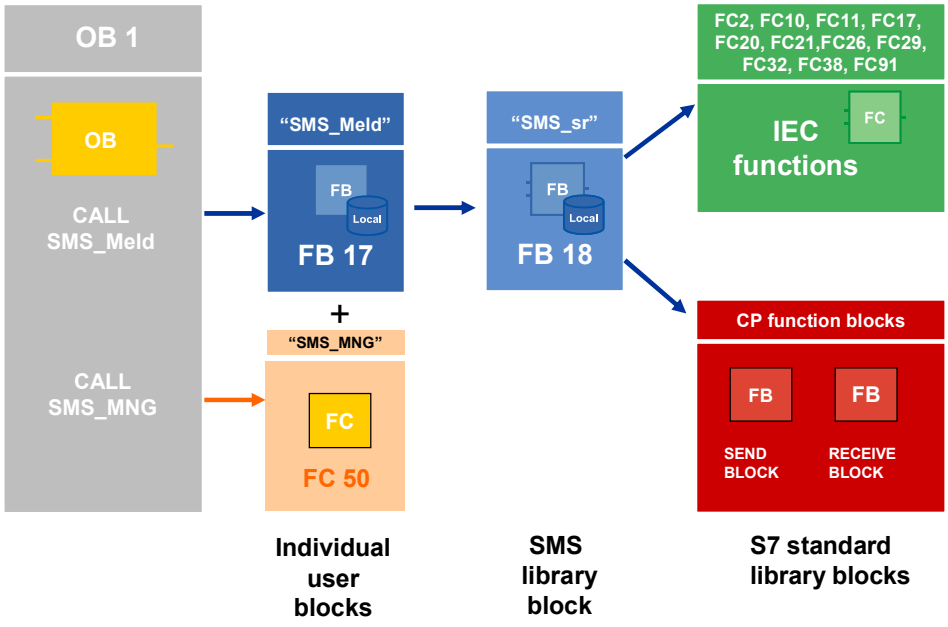
The following sections provide detailed explanations of the functionalities of the different scenarios

- Generating and sending an SMS message
- Sending an SMS message with attached coefficient
- Receiving and evaluating SMS orders and
- Receiving a remote query of a value.

3.1 Program overview

The figure below shows the program structure of the entire STEP 7 project.

Figure 3-1



Call of SMS_Meld in OB1

The following figure and table show the call interface of the core user block **FB "SMS_Meld"** in OB1.

Figure 3-2

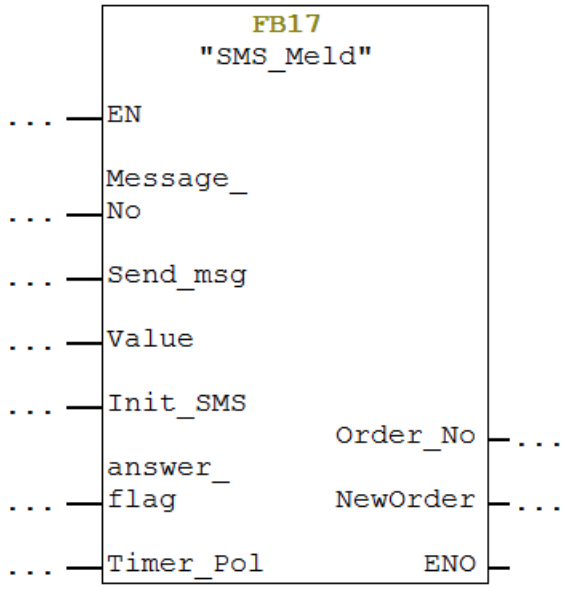


Table 3-1

Symbol	Data type	Explanation
Message_No	INT	Contains the number of the SMS message to be generated
Send_msg	BOOL	Send trigger command
Value	INT	Optional coefficient
Init_SMS	BOOL	Initialization trigger command
answer_flag	BOOL	sets telephonnumber to the one last SMS was received from
Order_No	INT	Output order number
NewOrder	BOOL	Set to TRUE when a new SMS message is received.
Timer_Pol	TIME	Once the timer has elapsed, polling of the received SMS messages starts.

The **FB "SMS_Meld"** function block is called cyclically in OB1 and then internally calls the **"SMS_sr"** function block.

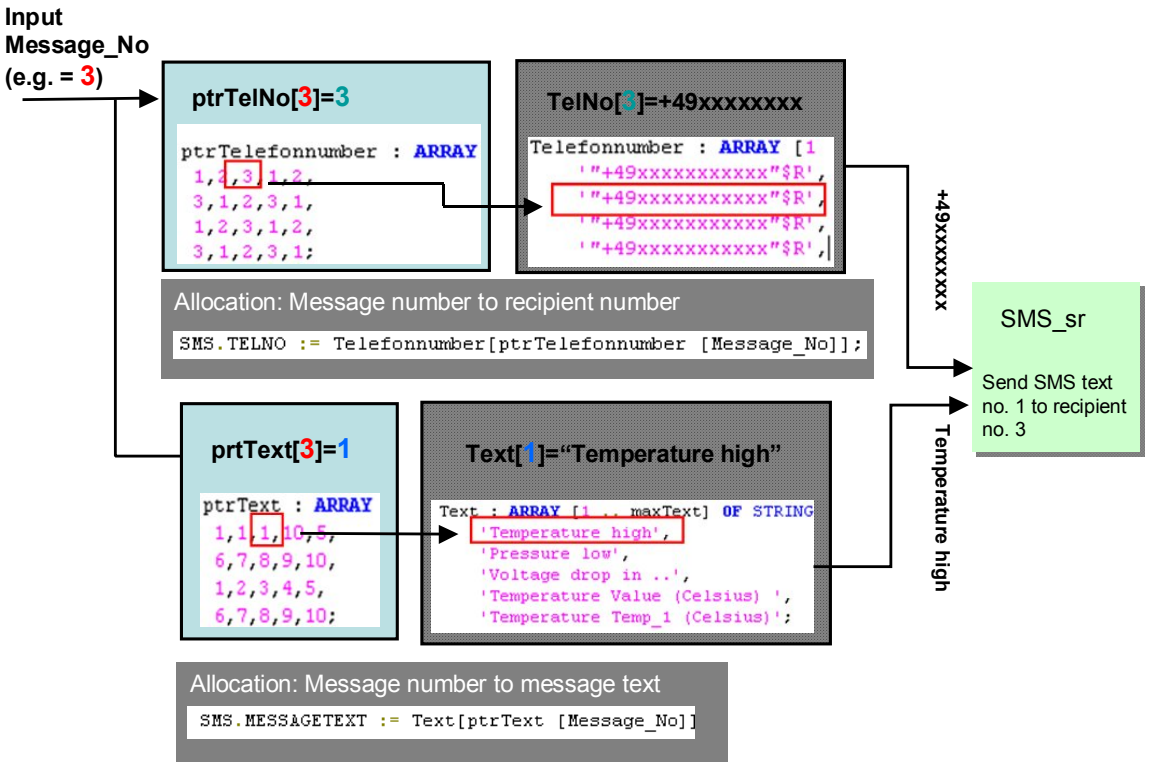
3.2 Scenario: “Generating and sending an SMS message”

Via an internal data array, the **FB “SMS_MeId”** function block generates a configurable text message using the **Message_No** input variable and sends it to a recipient that is assigned to this variable.

3.2.1 Diagram for the “Generating and sending an SMS message” scenario

The diagrammatic representation below shows how **FB “SMS_MeId”** generates a text message from a message number.

Figure 3-3



The **Message_No** variable contains the message number that is necessary to generate an SMS message. Using this variable, the associated telephone number and message text are determined from a total of four arrays.

Two of these lists are linked to one another:

- The ptrTelefonnumber and Telefonnumber lists are used to determine the telephone number.
- The ptrText and Text lists are used to determine the message text.

Depending on the value of the **Message_No** variable, the number is now determined from the **ptrTelefonnumber** and **ptrText** lists, which then defines the position in the list in **Telefonnumber** and **Text**.
In this way, the information necessary for the SMS message is determined (telephone number and message text).

By reconfiguring the indices and contents of the arrays, you can create any allocation and individually design the SMS message.

3.2.2 Program details for the “Generating and sending an SMS message” scenario

In this section, we show you the most important code fragments of this function from the documented source code of this example.

Variables for SMS allocation

Figure 3-4

```
45        //Alloccation Table of Messagenumbers to Telefonnumber
46       ptrText : ARRAY [1 .. maxMessage] OF INT :=
47        1,1,1,10,5,
48        6,7,8,9,10,
49        1,2,3,4,5,
50        6,7,8,9,10;
51        //Alloccation Table of Messagenumbers to Messagetext
52       ptrTelefonnumber : ARRAY [1 .. maxMessage] OF INT :=
53        1,2,3,1,2,
54        3,1,2,3,1,
55        1,2,3,1,2,
56        3,1,2,3,1;
57       //Messagetext
58       Text : ARRAY [1 .. maxText] OF STRING  [30] :=
59           'Temperature high',
60           'Pressure low',
61           'Voltage drop in ..',
62           'Temperature Value (Celsius) ',   // Exampletext with Input Value
63           'Temperature Temp_1 (Celsius)';
64
65       // Telefonnumbern
66       Telefonnumber : ARRAY [1 .. maxTel_nr] OF STRING  [24] :=
67           '"+49xxxxxxxxxxx"$R',
68           '"+49xxxxxxxxxxx"$R',
69           '"+49xxxxxxxxxxx"$R',
70           '"+49xxxxxxxxxxx"$R',
71           '"+49xxxxxxxxxxx"$R';
```

General variables

Figure 3-5

```
VAR
var_send : BOOL;
Pin_Code : STRING[10] := 'xxxx$R';//Pin of Sim-Card in Modem
Service_No:STRING[24]:='"+49xxxxxxxx"$R';
loop : INT;//loop counter
ValueStr : STRING[6] := ''; // Value converted to String
SMS : SMS_sr_CP_340;//instancing SMS Block
```

Variants	SMS Block/ SMS-Library
CP 340	SMS_sr_CP_340
CP 341	SMS_sr_CP_340
CP 441-2	SMS_sr_CP_441
ET 200S	SMS_sr_ET_200S

Message assignment code fragment

The following code lines illustrate how Message_No is assigned to SMS text and telephone number in SCL.

Figure 3-6

```
//Search Messagetext
SMS.MESSAGETEXT := Text[ptrText [Message_No]];


//Search Telefonnumber
SMS.TELNO := Telefonnumber[ptrTelefonnumber [Message_No]];
```

Call of SMS_sr_CPxxx

To send the SMS message, the **FB “SMS_MeId”** function block calls the **FB “SMS_sr_CPxxx”** library block.

The figure below shows the call of the **FB “SMS_sr_CPxxx”** function block.

Figure 3-7

```
SMS (CP_ADR := 256, //CP Address
  PIN_CODE := Pin_Code,//Pin Code
  SMS_SCA:=Service_No,//Service Center Number
  SMS_SEND:= Send_msg,
  INIT:=Init_SMS,
  TIMER_POL:=Timer_Pol);
```

3.3 Scenario: “Sending an SMS message with attached coefficient”

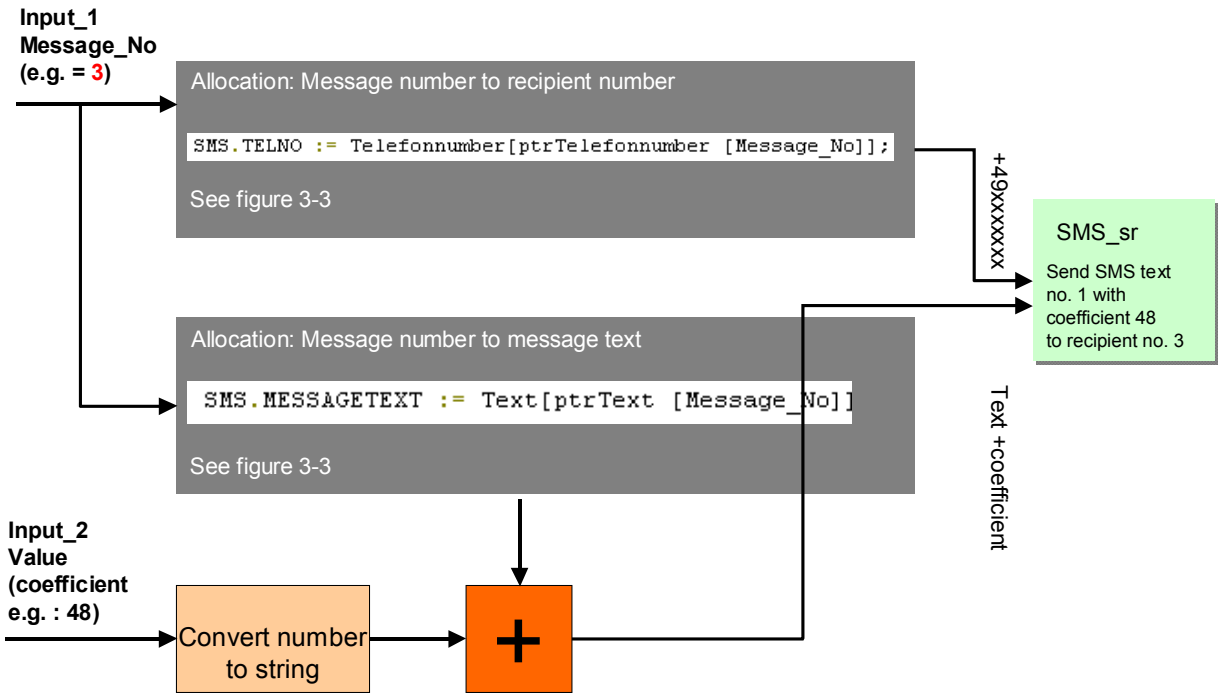
This scenario works like the “Generating and sending an SMS message” scenario. The only difference is that the message text can be sent with an attached coefficient.

The coefficient is stored in the **“Value”** variable (type:INT) and, using the **“ValueStr”** variable, must first be converted to the STRING data type for further processing.

3.3.1 Diagram for the “Sending an SMS message with attached coefficient” scenario

The diagrammatic representation below shows how **FB “SMS_MeId”** generates a text message from a message number and sends it to the recipient with an attached coefficient.

Figure 3-8



3.3.2 Program details for the “Sending an SMS message with attached coefficient” scenario

In this section, we show you the most important code fragments of this function from the documented source code of this example.

Converting the “Value” variable from number to text

Figure 3-9

```
IF Value <> 0 THEN
   ValueStr:= I_STRNG(I :=Value  // IN: INT
                      ); // STRING
```

Message assignment code fragment

The following code lines illustrate how to attach a coefficient to a message text in SCL.

Figure 3-10

```
SMS.MESSAGETEXT :=CONCAT(IN1 := Text[ptrText [Message_No]],IN2:= ' : '); // IN: STRING

SMS.MESSAGETEXT :=CONCAT(IN1 := SMS.MESSAGETEXT,IN2:=ValueStr ); // STRING
```

3.4 Scenario: “Receiving and evaluating SMS orders”

The **FB “SMS_MeId”** function block checks all incoming SMS messages for special “keywords”. When checking the received text with the keyword is positive, the respective order number will be output on the “**Order-No**” parameter.

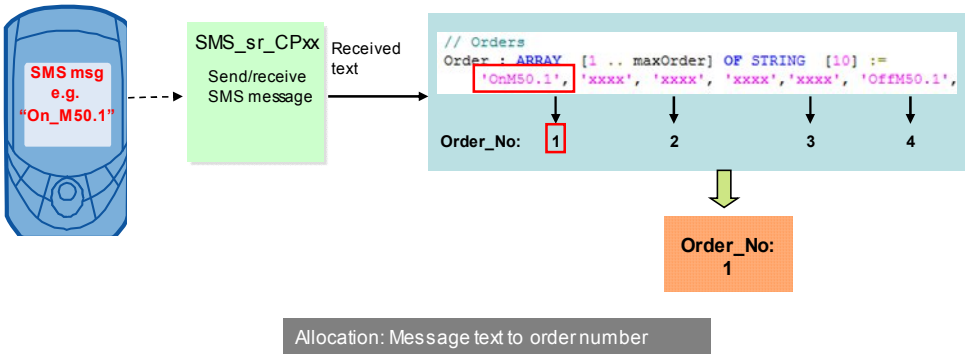
The receive buffer of the GSM modem is checked at regular intervals. The time value is defined by the value of the **Timer_Polling** parameter (in this configuration: 1 minute). After **receiving** a new message, the information contained in the message is stored in the receive data area.

NOTE For sending SMS to the modem it is recommended to use only alphabetic characters and figures. This is in order to avoid display and recognition errors of the STEP 7 application program.

3.4.1 Diagram for the “Evaluating SMS orders” scenario

The diagrammatic representation below shows how **FB “SMS_MeId”** receives and evaluates an incoming SMS message. The keywords are stored in the **Order** array.

Figure 3-11



3.4.2 Program details for the “Evaluating SMS orders” scenario

Variables for order allocation

The following code fragment shows the “keywords” for which an SMS message is checked in this example. You can enter any special “keywords” in the **FB “SMS_MeId”** SCL source.

Figure 3-12

```
// Orders
Order : ARRAY  [1 .. maxOrder] OF STRING  [10] :=
    'OnM50.1', 'xxxx', 'xxxx', 'xxxx','xxxx',
    'OffM50.1', '?Temp1'  ;
```

The **FB “SMS_MeId”** function block again uses the **FB “SMS_sr_CPxxx”** function block from the SMS library.

Order assignment code fragment

The following section of the documented source code shows how the incoming SMS message is checked for special “keywords” in a search loop.

Figure 3-13

```
(* ----------------------------------Check incoming SMS-----------------------------
   In this paragraph the incoming SMS  is checked, if it contains a defined order.
  ---------------------------------------------------------------------------------

   IF (SMS.SMS_NDR)THEN
     FOR loop := 1 TO maxOrder DO
      //analyse Order
       IF SMS.RECV_MESSAGE=Order[loop] THEN
         Order_No := loop;
        EXIT;
       ELSE
         Order_No :=0 ;
       END_IF;
     END_FOR;
```

3.5 Scenario: “Receiving a remote query of a value”

Using the **FB “SMS_MeId”** block, SMS messages for querying a value (e.g., the analog value of a temperature) can be interpreted. An automatic feedback message is sent with the aid of the **FC “SMS_MNG”** function.

NOTE For sending SMS to the modem it is recommended to use only alphabetic characters and figures. This is in order to avoid display and recognition errors of the STEP 7 application program.

3.5.1 Diagram for the “Receiving a remote query of a value” scenario

The diagrammatic representation below shows how **FC “SMS_MNG”** sends an answer to a query of a value back to the sender.

Figure 3-14

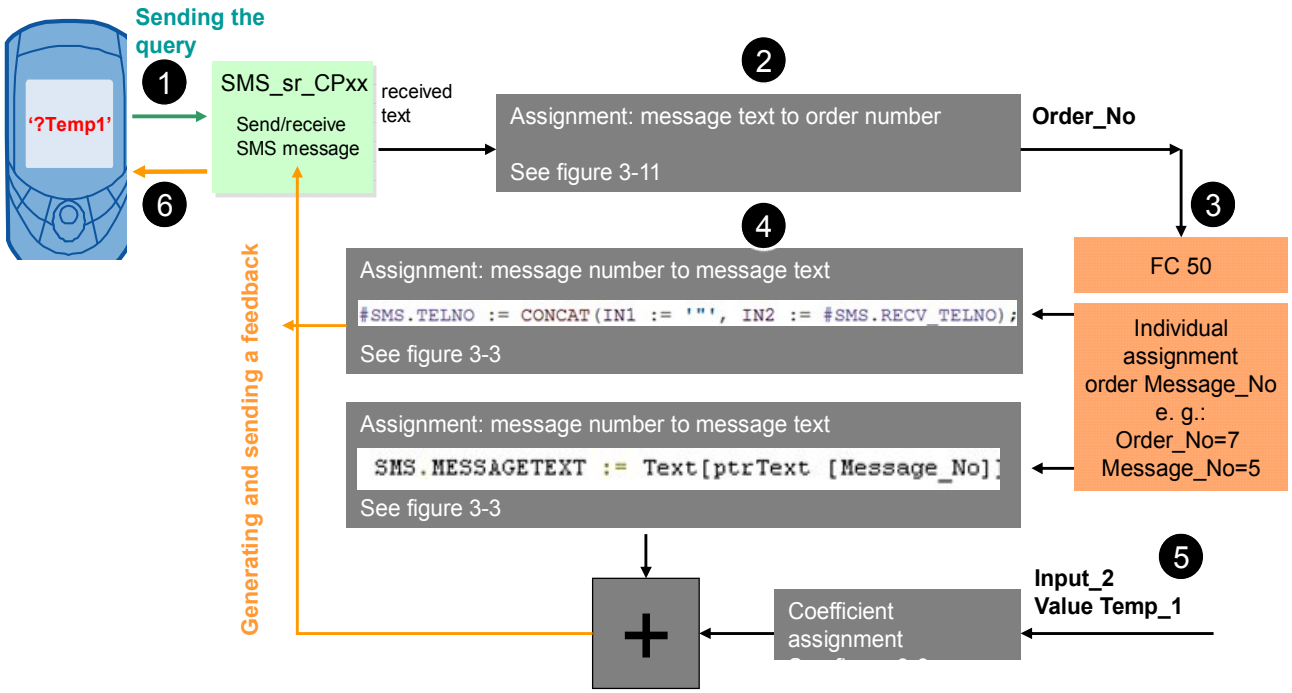


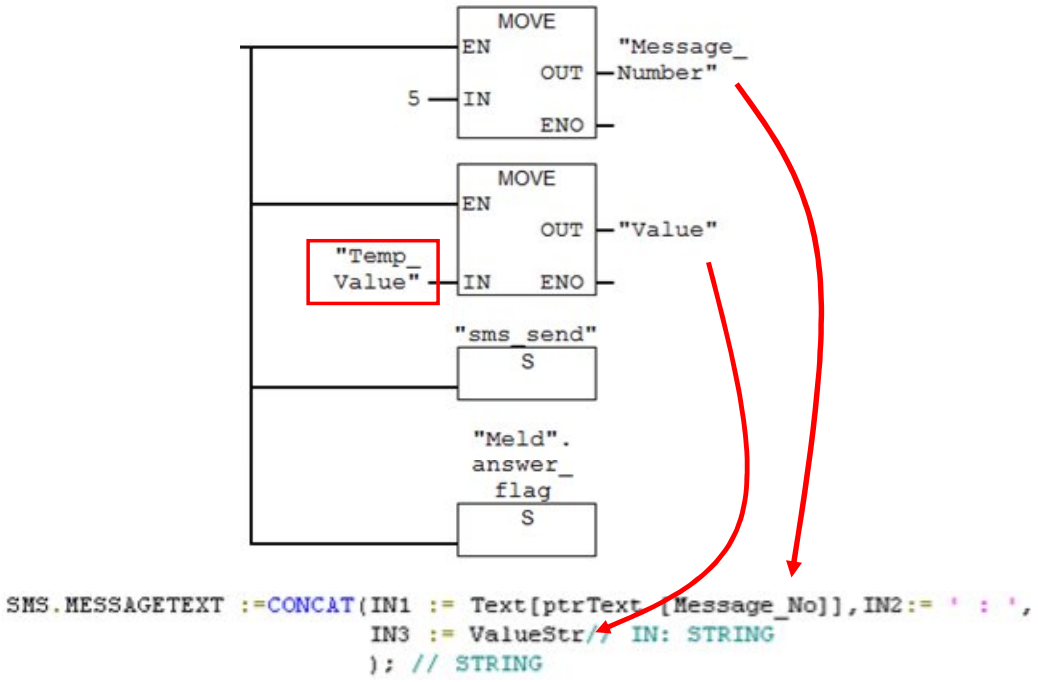
Table 3-2

No.	Description
1.	A query is sent to the controller.
2.	The received text is evaluated and assigned to an order number.
3.	With the aid of FC 50, the order number is assigned to a message number.
4.	Message text is generated from this message number, recipient number is extracted from received SMS.
5.	The coefficient is attached to the message text.
6.	The feedback message is generated and sent to the user.

3.5.2 Program details for the “Receiving a remote query of a value” scenario

The figure below shows the functionality of **FC “SMS_MNG”**.
To implement this scenario, the block is called in OB1.

Figure 3-15



Message_No and Value are then used in the SCL source to generate the feedback message.

4 Installation

4.1 Hardware configuration of the S7 station

Overview

This application example includes the following S7 project variants:

Table 4-1

Station	Variants/program name
S7-300 station	**CP_340** HW Config and S7 program for variant with CP 340
	CP_341 HW Config and S7 program for variant with CP 341
	ET 200S _1SI HW Config and S7 program for variant with ET 200S SI module
S7-400 station	**CP_441-2** HW Config and S7 program for variant with CP 441-2

NOTICE

The following applies to all project variants:
Before you switch on the power supply, complete and check the configuration!

4.1.1 S7-300 station with CP 340

The figure below shows the hardware configuration of the project variant with CP340.

Figure 4-1

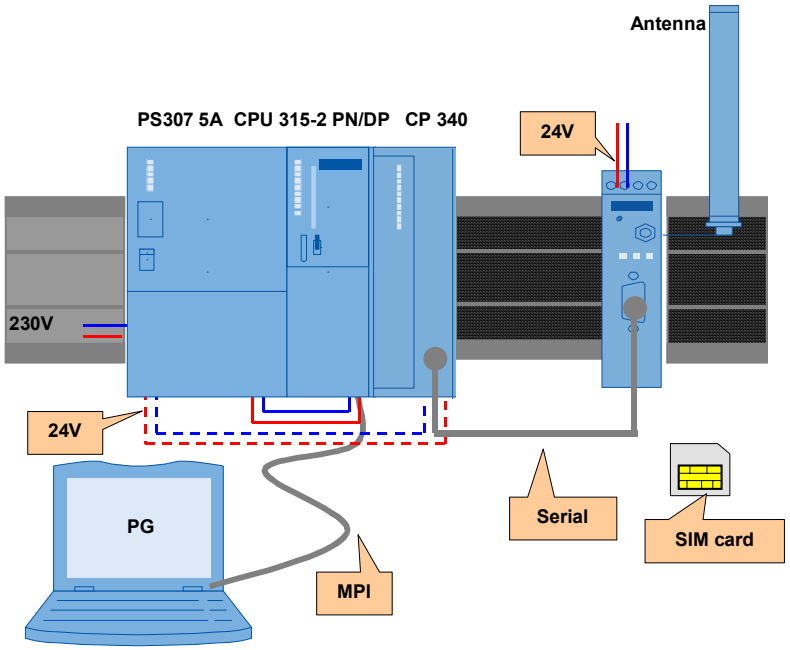


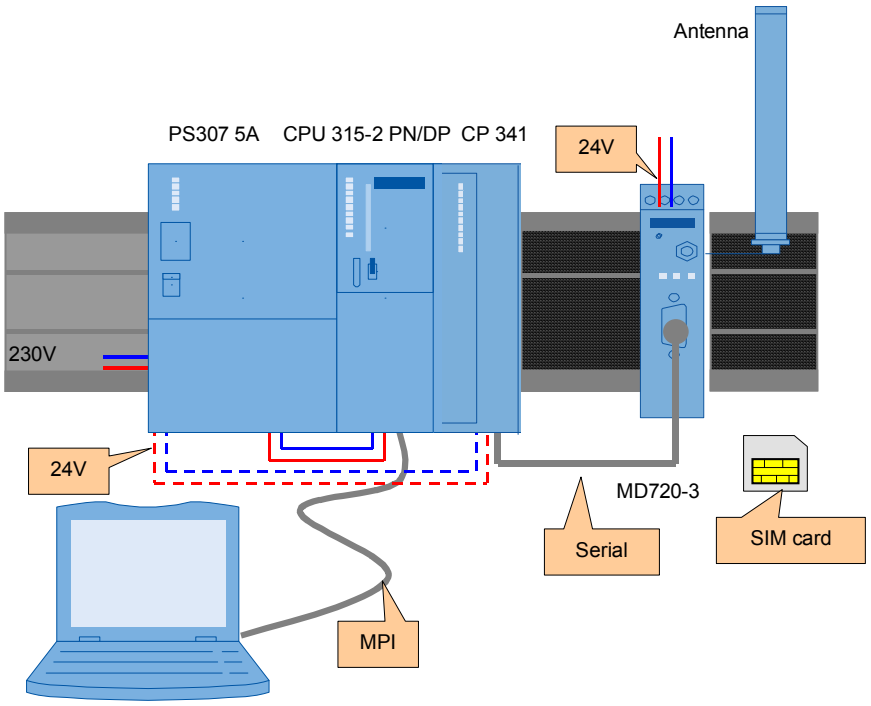
Table 4-2

No.	Action	Remark
1.	Attach the individual modules to a suitable rack.	List of components Table 2-3
2.	Use a backplane bus connector to connect CPU and CP.	
3.	Connect all respective components to a 24 V direct current source (PS307). Connect the PS307 to the electricity-supply system (230 V AC).	Ensure that the polarity is correct.
4.	Connect the MPI of the engineering PG to the MPI of the S7 CPU.	
5.	Connect the serial cable to the CP.	

4.1.2 S7-300 station with CP 341

The figure below shows the hardware configuration of the project variant with CP341.

Figure 4-2



Configure the S7 station/CP341 as shown in Table 4-2.

4.1.3 S7-300 station with ET 200S with 1SI module

The figure below shows the configuration of the S7-300/ET 200S station.

Figure 4-3

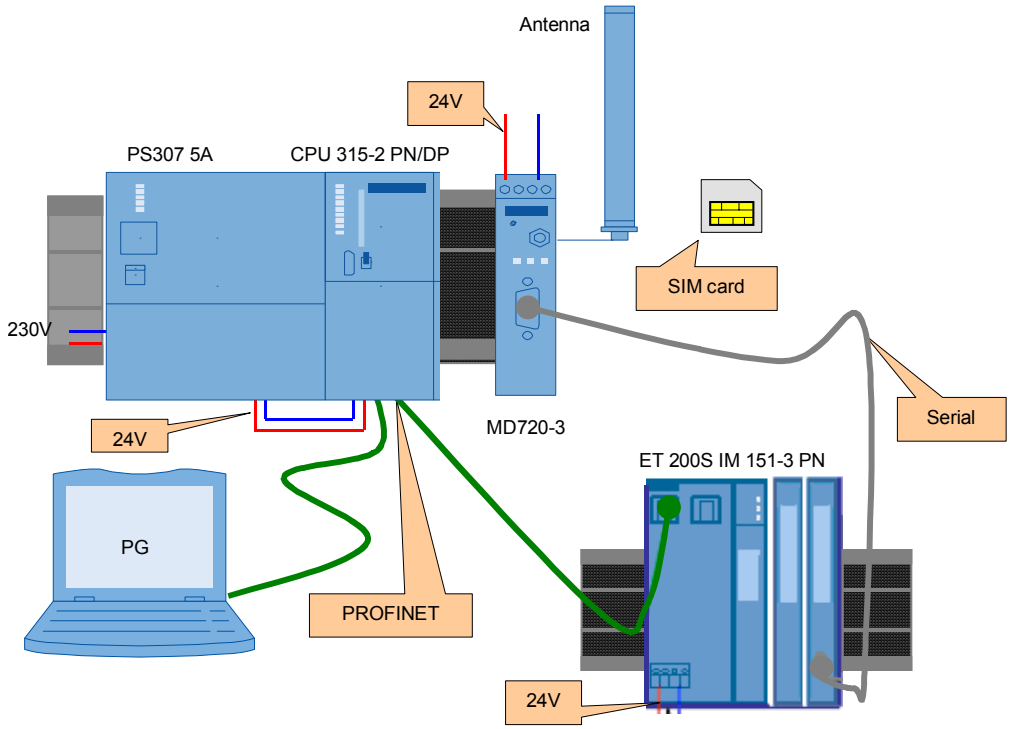
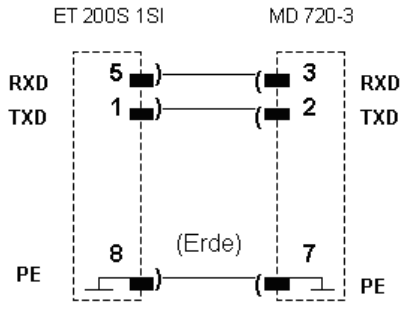


Table 4-3

No.	Action	Remark
1.	Attach the individual modules to a suitable rack.	List of components Table 2-5
2.	Connect the PS307 to the electricity-supply system. (230 V AC)	Ensure that the polarity is correct.
3.	Connect the following Profinet interface: • Engineering PG to the S7 CPU. • S7 CPU to the ET 200S.	
4.	Connect the serial cable to the 1SI module. Connect the other side to the MD 720-3.	

4.1.4 S7-400 station with CP 441-2

The figure below shows the hardware configuration of the project variant with CP441-2.

Figure 4-4

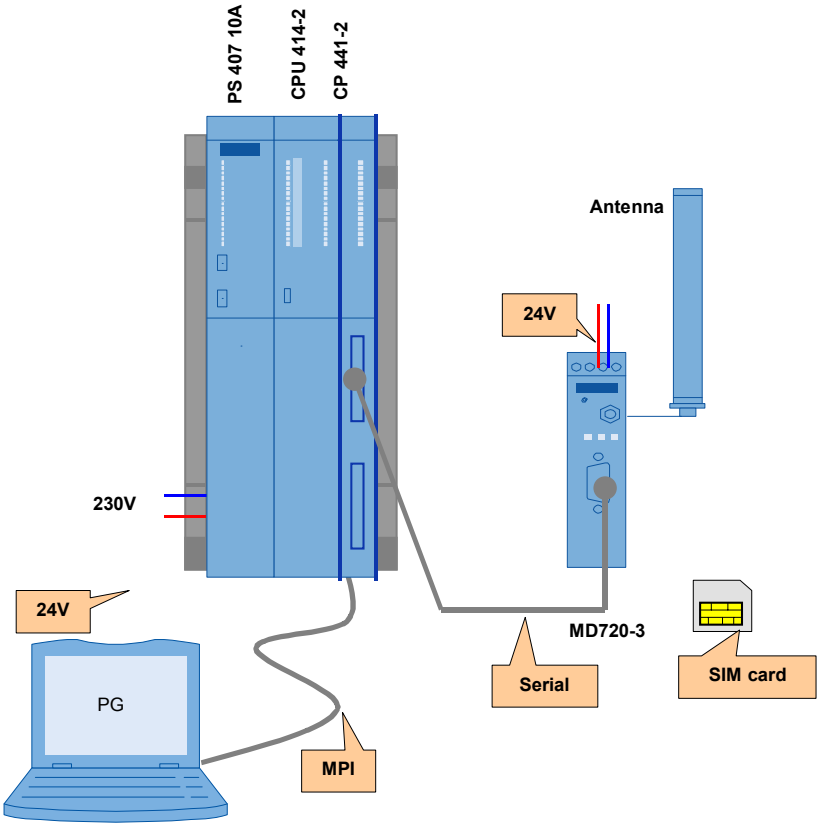


Table 4-4

No.	Action	Remark
1.	Attach the individual modules to a suitable rack.	List of components Table 2-5
2.	Connect the PS407 to the electricity-supply system. (230 V AC).	Ensure that the polarity is correct.
3.	Connect the MPI of the engineering PG to the MPI of the S7 CPU.	
4.	Connect the serial cable to the CP.	

4.2 MD720-3 hardware configuration

Table 4-5

No.	Action	Remark
1.	Open the casing of the MD720-3 and insert the SIM card. Close the MD720-3.	Follow the MD720-3 manual.
2.	Connect the antenna to the respective socket.	
3.	Connect the MD720-3 to a 24 V direct current source.	You can also use the PS307 or the PS407 of the S7 station.
4.	Connect the serial cable to the interface of the MD720-3.	

Note Always follow the installation guidelines for the installation of all components.

4.3 Software installation

The engineering station is used as a configuration computer for the S7 station.

Table 4-6

No.	Action	Remark
1.	Install STEP 7 V5.5.	Follow the instructions of the installation program.
2.	Install the configuration package for the point-to-point communication.	This CD is included in the delivery of the serial CPs and contains entries for the STEP 7 hardware configuration and manuals.
3.	Install S7-SCL V5.3+SP5.	Follow the instructions of the installation program.

1.

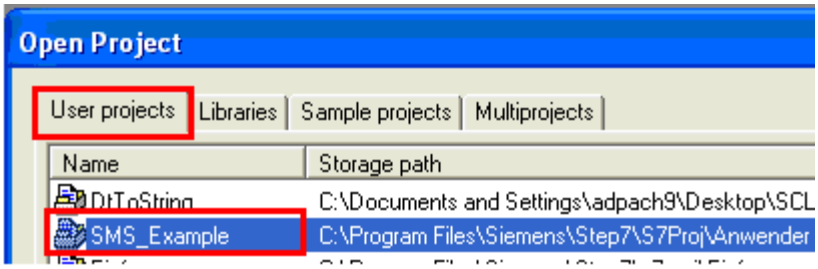
5 Commissioning of the Application

5.1 General preparations

5.1.1 Installing the application software

The following table lists the steps necessary to install the sample code.

Table 5-1

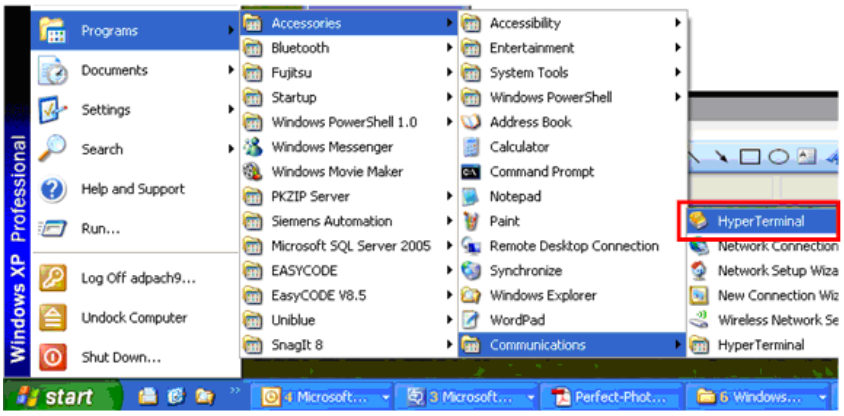
Step	Procedure
1.	The project is available on the HTML page from which you downloaded this document. Save the **"SMS_Example.zip"** project to your hard drive.
2.	Open the **SIMATIC MANAGER** and retrieve the STEP 7 project. "File > Retrieve..."
3.	The project is now available in User projects. 

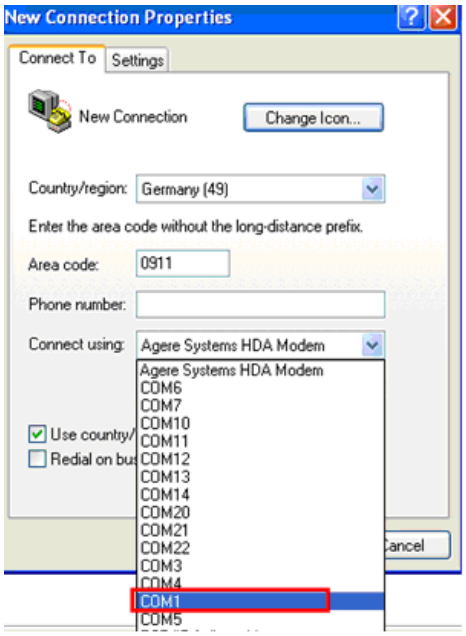
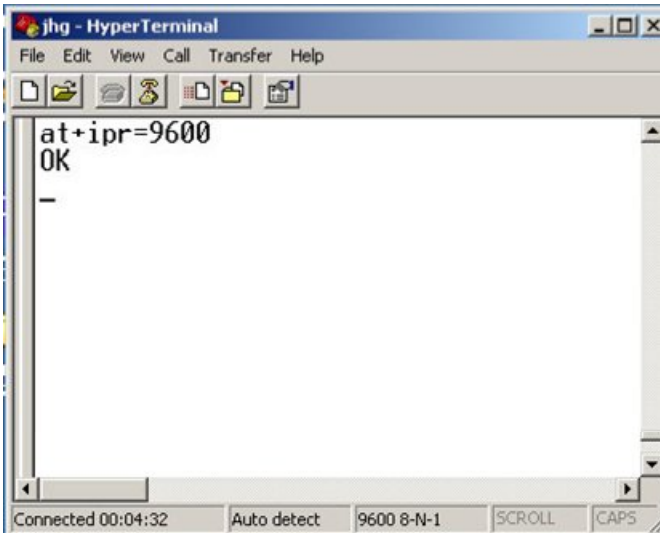
5.1.2 Setting the baud rate of the MD720-3

In this project, the baud rate of the CPs is always set to 9600 bps. CP and MD720-3 must support the same baud rate.

The rate on the MD720-3 is changed using a terminal program.

Table 5-2

Step	Procedure
1.	Connect a PC to the serial interface of the MD720-3.
2.	Start a terminal program, for example HyperTerminal. 

Step	Procedure
3.	Select the respective COM interface. 
4.	Set character format and baud rate to the same values as the serial interface of the MD720-3. The factory settings of the MD720-3 are as follows: Baud rate: 19200 bps Character format: 8N1. The baud rate is changed via AT command: **AT+IPR=<baud rate>**. Enter this command in the terminal program and press the return key. 

.

Note

The modem is only accessed by AT commands when it is in the command phase. If this is not the case, the modem must be reset to factory settings.

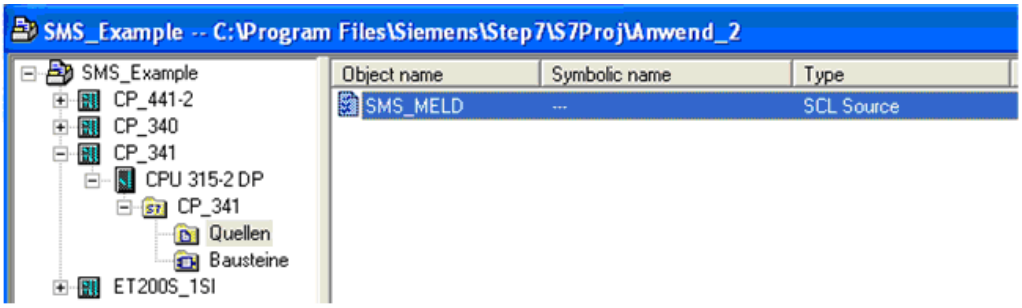
5.1.3 Setting the modem parameters

Before you can send and receive SMS messages, you have to store the following information for the initialization of the MD720-3 modem:

- PIN of the SIM card in the modem.
- Number of the short message service center (see library description "25545680_Library_SMS_MD720_DOKU_V2.0_e.pdf", table 4-1 "SMS_SCA").

In addition, the text to be sent and the telephone number of the recipient must be entered.

Table 5-3

No.	Procedure
1.	Open the **FB "SMS_MeId"** SCL source of the **SMS_Example** project. 

2.	For the **Pin_code** parameter, enter the PIN of the SIM card you have inserted into the MD720-3 and for the **Service_No** parameter, enter the short message service center number.
	`VAR`
	`Pin_Code : STRING[10] := 'xxxx$R'; //Pin of Sim-Card in Modem`
	`Service_No:STRING[24]:='"+49xxxxxxxxxx"$R';`
	Make sure that you replace only the telephone number (+49xxxxxxxxxx), the "$R" control character must **NOT** be deleted!

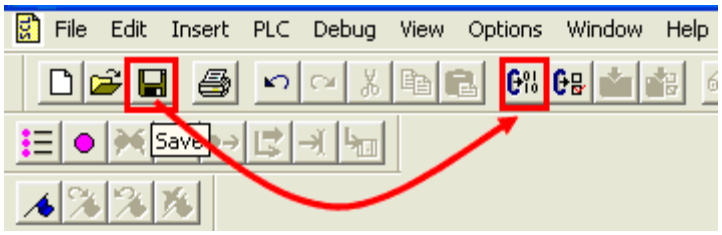
3.	In the **Text** array, you can enter the individual message texts that are to be sent for the assigned Message_No (to the array index).
	`//Messagetext`
	`Text : ARRAY [1 .. maxText] OF STRING [30] :=`
	`'Temperature high',`
	`'Pressure low',`
	`'Voltage drop in ..',`

No.	Procedure

4.	In the **Telefonnumber** array, enter the possible recipient numbers to which the SMS messages are to be sent.
	`// Telefonnumbern`
	`Telefonnumber : ARRAY [1 .. maxTel_nr] OF STRING [24] :=`
	`'"+49xxxxxxxxxx"$R',`
	`'"+49xxxxxxxxxx"$R',`
	`'"+49xxxxxxxxxx"$R',`
	`'"+49xxxxxxxxxx"$R',`
	`'"+49xxxxxxxxxx"$R';`
	Make sure that you replace only the telephone number (+49xxxxxxxxxx). The “$R” control character must **NOT** be deleted!

5.	For the **cp_adr** parameter, enter the logical address of the CP.
	`VAR`
	`var_send : BOOL;`
	`Pin_Code : STRING[6] := 'xxxx$R';//Pin of Sim-Card in Modem`
	`Service_No:STRING[24]:='"+49xxxxxxxxxx"$R';`
	`loop : INT;//loop counter`
	`ValueStr : STRING[6] := ''; // Value converted to String`
	`SMS : SMS_sr_CP_340;//instancing SMS Block`
	`cp_adr:INT:= 256;`
	`END_VAR`

6.	In the **ptrText** and **ptrTelefonnumber** allocation arrays, set the values so that any message is sent to any recipient. In the example: Message_No := 1
	`//Alloccation Table of Messagenumbers to Telefonnumber`
	`ptrText : ARRAY [1 .. maxMessage] OF INT :=`
	`1,1,1,10,5,`
	`6,7,8,9,10,`
	`1,2,3,4,5,`
	`6,7,8,9,10;`
	`SMS.MESSAGETEXT := Text[ptrText [Message_No]]`
	`//Alloccation Table of Messagenumbers to Messagetext`
	`ptrTelefonnumber : ARRAY [1 .. maxMessage] OF INT :=`
	`1,2,3,1,2,`
	`3,1,2,3,1,`
	`1,2,3,1,2,`
	`3,1,2,3,1;`
	`SMS.TELNO := Telefonnumber[ptrTelefonnumber [Message_No]];`

7.	Save and compile the SCL source. 

5.1.4 Configuring the serial CPs

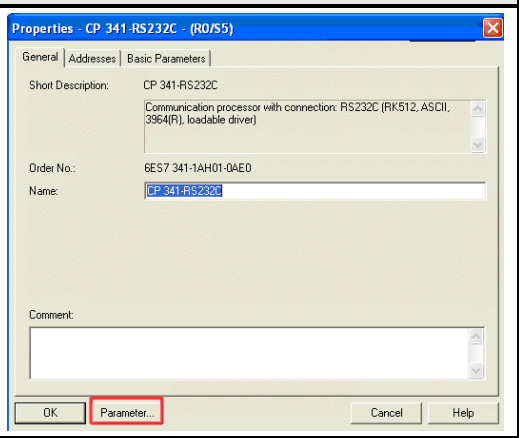
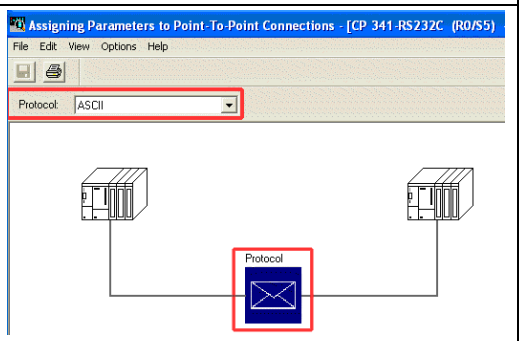
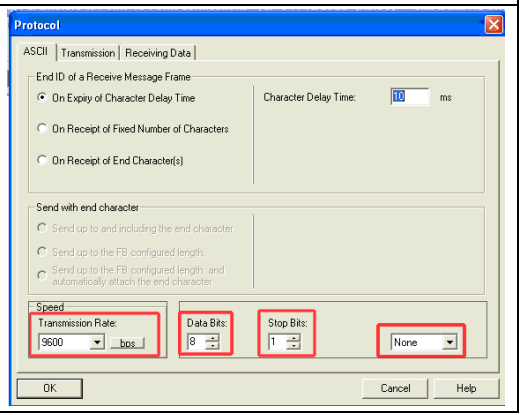
Configuration for CP 340 and CP 341

In this example, the serial CP uses the ASCII driver for the interface. This allows sending or receiving AT commands and mere text code to or from a communications partner via the interface.

The properties of the ASCII driver are to be as follows:

- Transmission Rate: 9600 bps
- Data Bits: 8
- Stop Bits: 1
- No parity check (none)

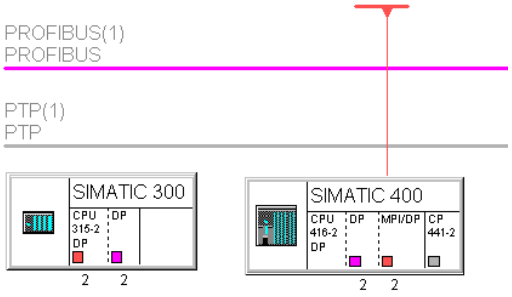
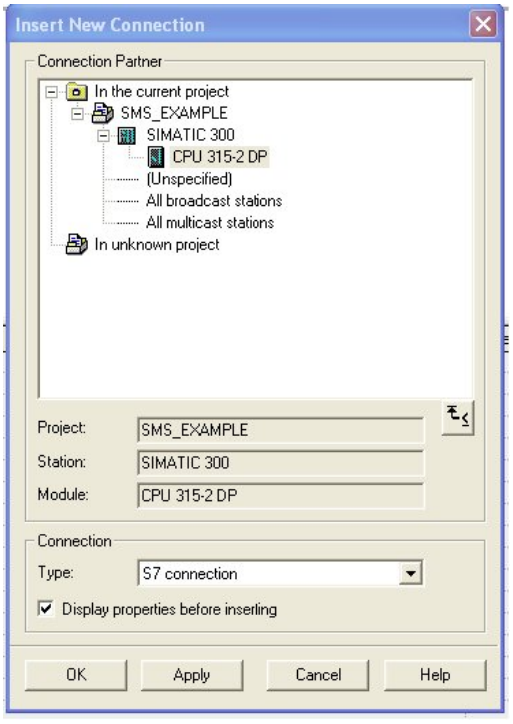
Table 5-4 Configuration of CP 340/ 341/ 441-2

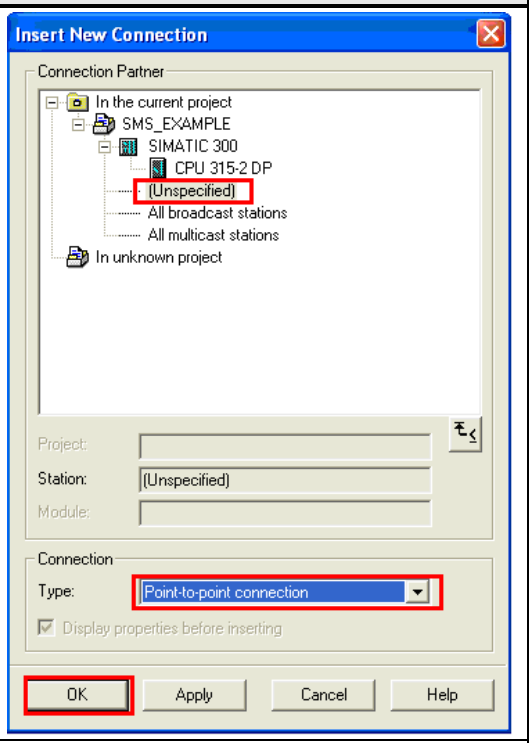
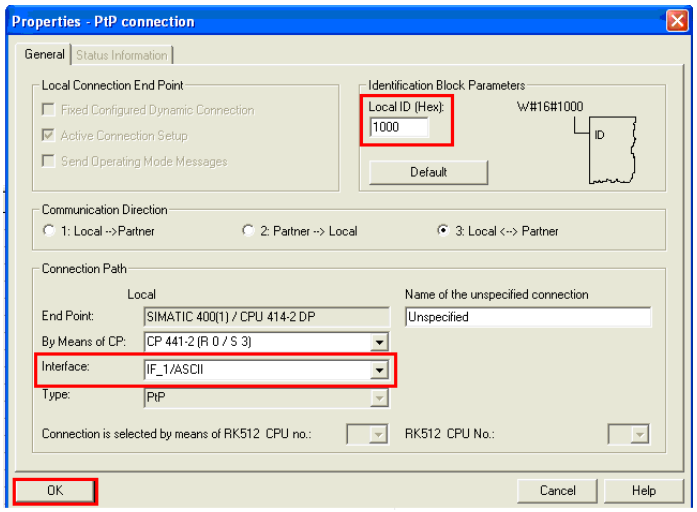
No.	Action	Remark
1.	In the hardware configuration, double-click on the CP. The **Properties** dialog box of the module opens. In this dialog box, select the **Parameter…** button.	
2.	In this dialog box, you can define the protocol of the CP. In the drop-down list, select **ASCII**. Then double-click on the **Protocol** envelope to define the properties for the selected protocol.	
3.	The properties of the ASCII driver are to be as follows: • **Transmission Rate**: 9600 bps • **Data Bits**: 8 • **Stop Bits**: 1 • **No** parity check (**none**)	

No.	Action	Remark
4.	Confirm all dialog boxes with OK. Then recompile the hardware configuration. “Station > Save and Compile”	

Parameterization and configuration for CP 441-2

Table 5-5

No.	Action	Remark
1.	CP_441-2 has the **RS232 module** at **interface 1** and **no module** at **interface 2**. If the interface assignment of your CP differs from the one described above, HW Config must be changed accordingly.	Communication with a serial CP requires that a connection be configured in **NetPro**. For a configuring guide for the CP, please refer to Table 5-4 Configuration of CP 340/ 341/ 441-2.
2.	Open NetPro. “Options > Configure Network”	
3.	Select the CPU of the S7-400 station. Create a new connection. “Insert > New Connection…”	

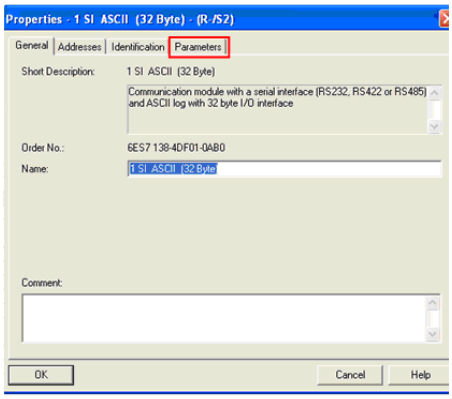
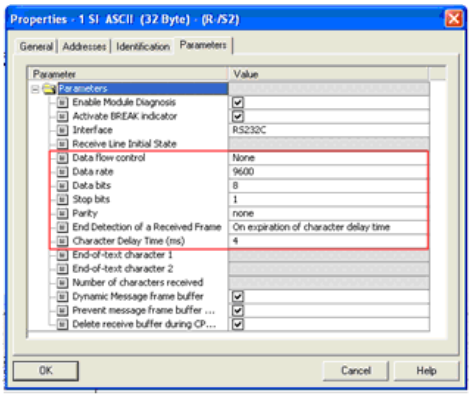
No.	Action	Remark
4.	For Connection Partner, select **Unspecified** and for Connection Type, select **Point-to-point connection**. Close the dialog box with OK.	
5.	In the following dialog box, you can define the properties. In **Local ID**, select **1000**. This ID is used by the BSEND and BRCV. In **Interface**, select your **interface** with the **RS232 module** and the **ASCII driver**. Close the dialog box with OK. 	
6.	Save and recompile NetPro.	

Note If you want to project even further S7-400 CPs, you need to select varying Local IDs respectively (change them via the instance DB: con_id).

Parameterization and configuration for ET 200S 1SI

The following table shows the necessary settings for the ASCII driver of the ET 200S 1SI ASCII serial interface module.

Table 5-6

No.	Action
1.	In the hardware configuration, double-click on 1 SI ASCII. The **Properties** dialog box of the module opens. Click on **Parameters…** 
2.	In this dialog box, you can define the properties for the ASCII driver. • **Transmission Rate**: 9600 bps • **Data Bits**: 8 • **Stop Bits**: 1 • **No** parity check (**none**) 
3.	Confirm all dialog boxes with “OK”. Then recompile the hardware configuration. “Station > Save and Compile”

Note

When parameterizing the ET 200S distributed I/O, you have to assign a Profinet device name to the IM 151-3 PN module.

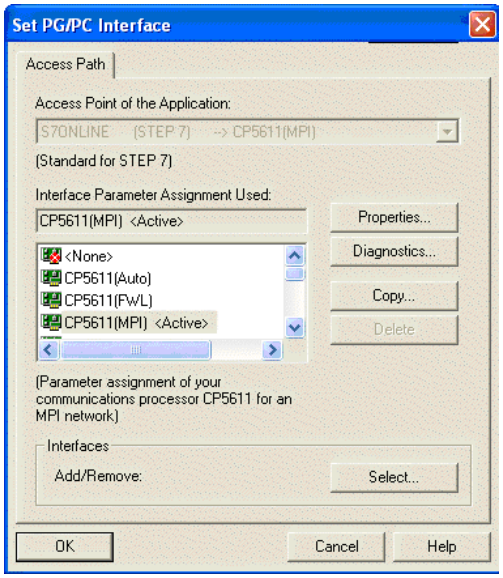
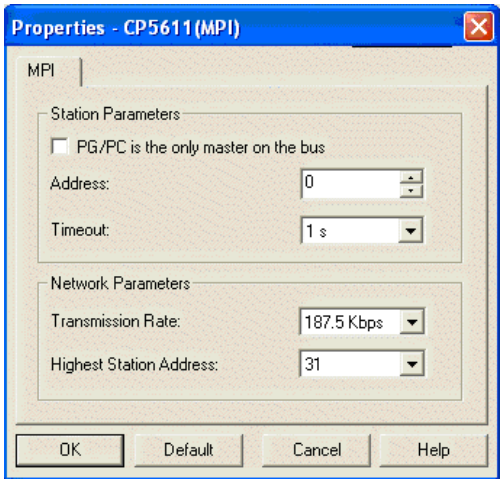
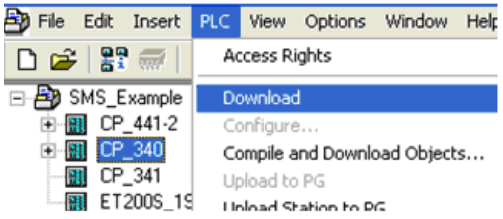
“HW Config > PLC > Ethernet >…”

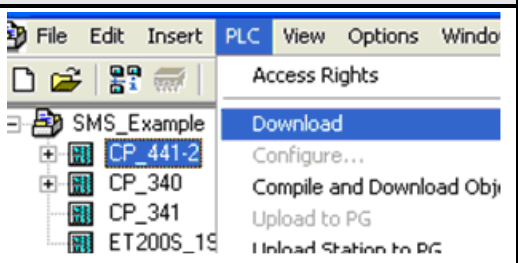
IP address of ET 200 S IM151 3 PN -1: 192.168.255.4

5.2 Downloading the STEP 7 project

Variant for the CP 340/ CP 341/ CP 441-2 communications modules

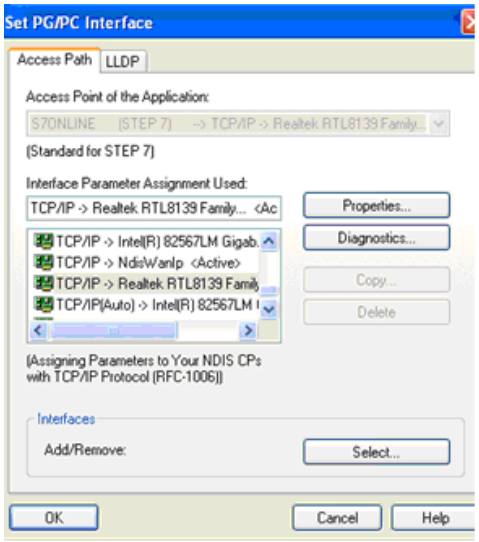
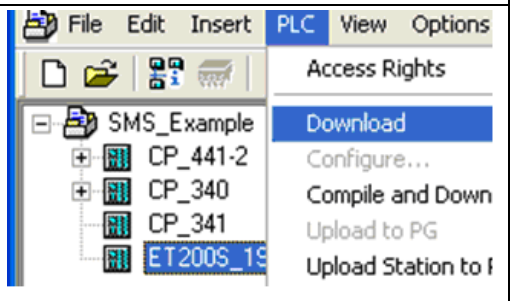
Table 5-7

No.	Action	Remark
1.	In the **SIMATIC MANAGER**, set the PC interface to MPI. "Options > Set PC/PG Interface…"	
2.	Use the **Properties** button to view the MPI properties. If necessary, change your MPI properties as shown in the screen shot. Close all dialog boxes with **OK**.	
3.	If you are using an S7-300 station, select the CP_340 or CP_341 S7 station and download the entire project to your CPU. "PLC > Download"	

No.	Action	Remark
4.	If you are using an S7-400 station, select the CP_441-2 S7 station and download the entire project to your CPU. "PLC > Download" Double-click on the CPU and then click on **Connections**. NetPro opens. Select the **SIMATIC 400** station and download it to the CPU.	

Variant for the ET 200S 1SI communications module

Table 5-8

No.	Action	Remark
1.	In the **SIMATIC MANAGER**, set the PC interface to TCP/IP. "Options > Set PC/PG Interface"	
2.	Select the access path. For the used network card, select TCP/IP. Then click on the "Properties…" button. In the "Properties" dialog box, select the "Assign project-specific IP address" option. Then confirm twice with "OK". ("Configuring the PROFINET interface" manual, see \11\)	
3.	Select the ET 200S_1SI S7 station and download the entire project to your CPU. "PLC > Download"	

6 Operation of the Application

This chapter shows you how to operate the described scenarios of this application.

All necessary variables can be found in the **“VAT_MELD”** variable table.

Note For information on errors that may occur when operating the application, please refer to the library description “25545680_Library_SMS_MD720_DOKU_V2.0_e.pdf”. If the support does not lead to the desired result, it may be useful to download the entire project.

6.1 Sending an SMS message (scenario 1/scenario 2)

Sending an SMS message is controlled via the **“sms_send”** flag.

The prerequisites for generating and sending a new SMS message are as follows:

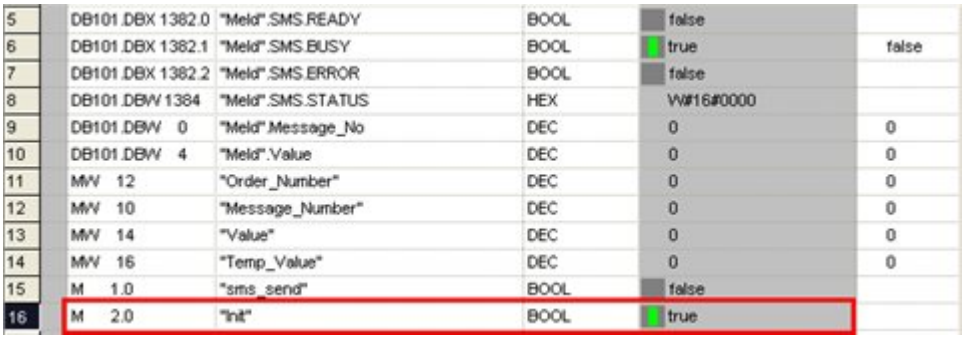
- A value was assigned to the “**Message_No**” variable.
- The **“sms_send”** flag is enabled.
- The modem was initialized.
- No other send operation is being actively processed (**"Meld".SMS.BUSY =true**).

6.1.1 Generating and sending an SMS message

The table below lists instructions for sending an SMS message.

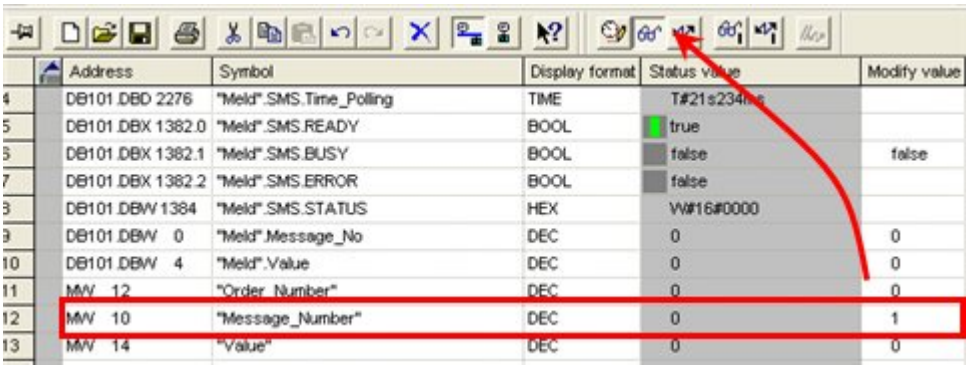
Table 6-1

No.	Procedure
1.	Set the desired parameters as described in Table 5-3. Download the entire project as described in chapter 5.2.
2.	Open OB1 and update the instances “File > Check and Update Accesses” Set within the properties of the Instance-DB the option "Non Retain ". „Object Properties> General Part2> Non Retain“ Save and once again download OB1 and the instances.
3.	Open the “VAT_MELD” variable table and initialize the modem by a positive edge on **“Init”**.



5	DB101.DBX 1382.0	"Meld".SMS.READY	BOOL	false	
6	DB101.DBX 1382.1	"Meld".SMS.BUSY	BOOL	true	false
7	DB101.DBX 1382.2	"Meld".SMS.ERROR	BOOL	false	
8	DB101.DBW 1384	"Meld".SMS.STATUS	HEX	W#16#0000	
9	DB101.DBW 0	"Meld".Message_No	DEC	0	0
10	DB101.DBW 4	"Meld".Value	DEC	0	0
11	MW 12	"Order_Number"	DEC	0	0
12	MW 10	"Message_Number"	DEC	0	0
13	MW 14	"Value"	DEC	0	0
14	MW 16	"Temp_Value"	DEC	0	0
15	M 1.0	"sms_send"	BOOL	false	
16	M 2.0	"Init"	BOOL	true	

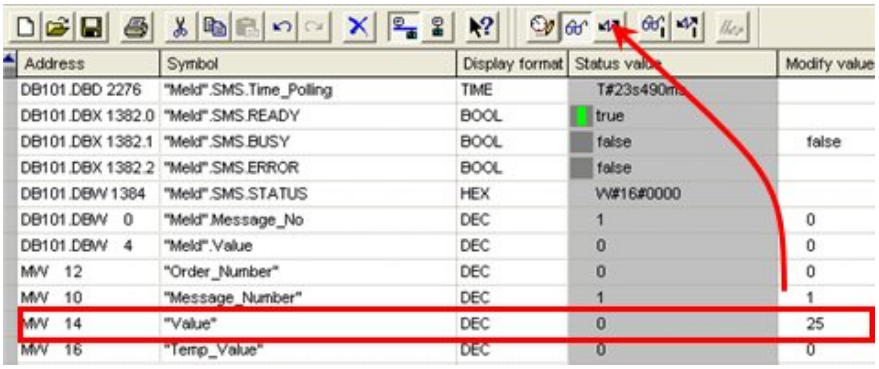
6.1 Sending an SMS message (scenario 1/scenario 2)

No.	Procedure
4.	In the “VAT_MELD” variable table, check whether the **“Meld.SMS.READY”** variable is set. If this is not the case, enable **“Init”** and reinitialize the modem. 
5.	The **“Message_Number”** variable contains the message number that is necessary to generate an SMS message. Use the VAT to modify the value. 
6.	Enable **“sms_send”** to send the message. 
7.	Once the SMS message has been sent (without errors), reset the send trigger command (“sms_send” flag) since the trigger command reacts only to a positive edge.

6.1.2 Sending an SMS message with attached coefficient

The table below lists instructions for sending an SMS message with an attached coefficient.

Table 6-2

No.	Procedure
1.	Generate a new SMS message as described in steps 4, 5 of table 6.1.
2.	In MW 14 **"Value"**, enter any coefficient as an integer value and enable the value. 
3.	Send the new SMS message as described in steps 6, 7 of table 6.1.

Note If you want to send an SMS message without an attached coefficient, enter the value "0" in MW14 "Value". If you do not enter a value, the previous value will be applied.

6.2 Receiving an SMS message (scenario 3/scenario 4)

The prerequisites for receiving a new SMS message are as follows:

- The modem was initialized.
- The polling timer has elapsed.

CAUTION	**Do not send an SMS message containing the key word 'ERROR' or 'OK', since MD720-3 evaluates these words when a faulty behavior occurs during sending from MD720-3 to the communication module.**

NOTE For sending SMS to the modem it is recommended to use only alphabetic characters and figures. This is in order to avoid display and recognition errors of the STEP 7 application program.

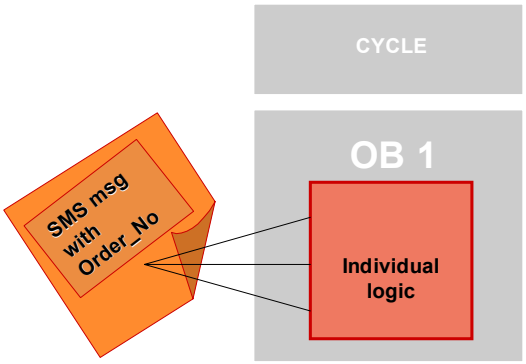
6.2.1 Receiving and evaluating an SMS message

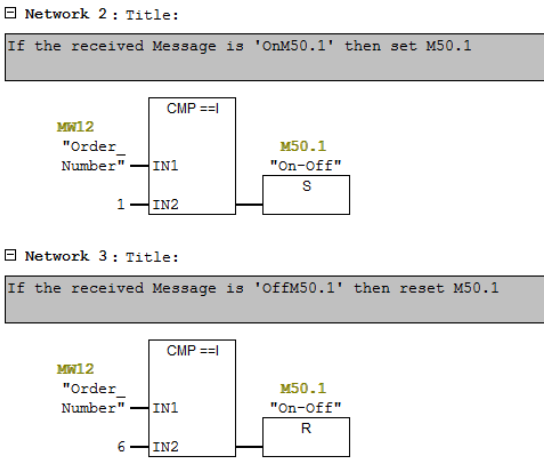
With the aid of the FB17 **FB "SMS_Meld"** function block, an S7-300/400/ET 200S station can receive and evaluate an order SMS.
The received SMS messages are polled every minute (default setting). You can change the timer as desired.
Once the timer has elapsed, all SMS messages are read out of the memory in the modem and evaluated.

Table 6-3

No.	Action	Remark

1.	Write an SMS message with an command of Figure 3-12 and send it to the MD720-3.
	Pay attention to upper and lower case capitalization.
		`Order : ARRAY [1 .. maxOrder] OF STRING [10] :`
		`'OnM50.1', 'xxxx', 'xxxx', 'xxxx','xxxx', 'O`
		Order_No: 1 2 3 4 ...

2.	All SMS messages received by the modem are evaluated. If one of the "keywords" is detected in the receive text, the assigned Order_No (e.g., no. 1 for 'OnM50.1') is output on the respective parameter of FB SMS_Meld and can then be used for individual further processing.	

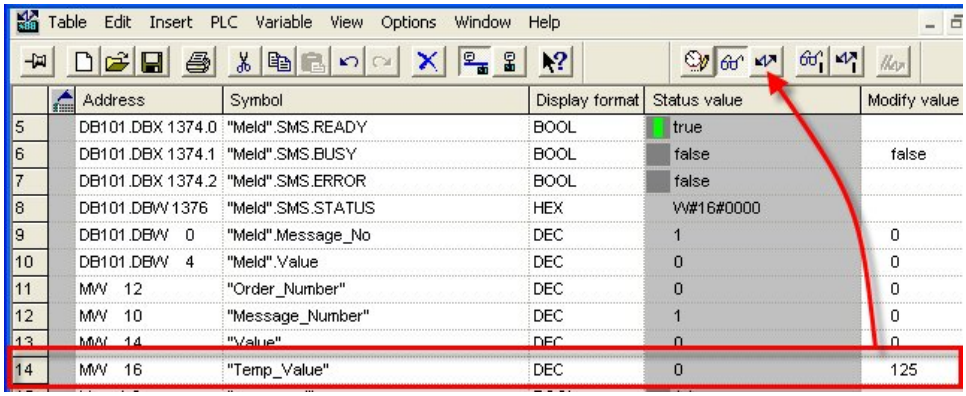
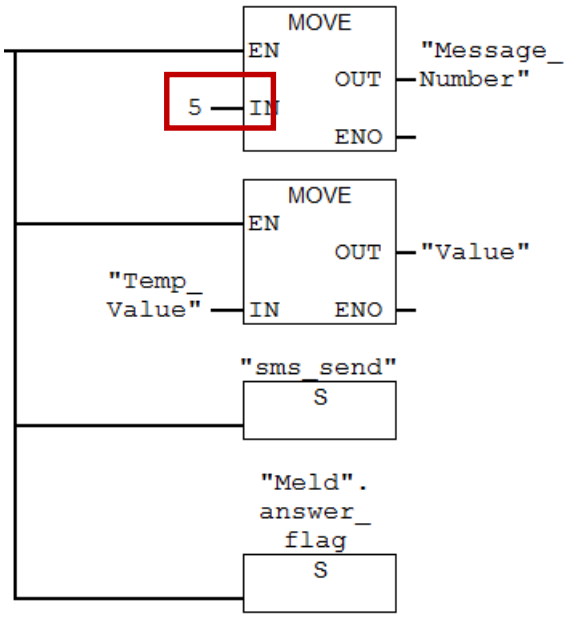
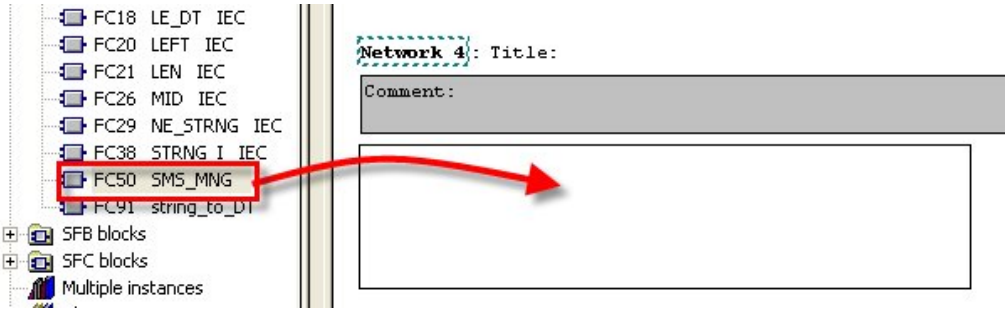
No.	Action	Remark
3.	An individual logic for the Order_No parameter may look as follows: Set and reset flag. • If the Order_No of the received message, is, for example, '1' 'OnM50.1', the M50.1 flag will be set. • If the Order_No of the received message, is, for example, '6' 'OffM50.1', the M50.1 flag will be reset.	

6.2.2 Remote query of a value

In this scenario, the **FB "SMS_MeId"** function block and the **FC "SMS_MNG"** function are used to receive and evaluate a query SMS message and to send a feedback message with the desired value back to the user.

6.2 Receiving an SMS message (scenario 3/scenario 4)

Table 6-4

No.	Action
1.	First open the **"VAT_MELD"** variable table and enable MW16 with the current temperature value. 
2.	Open the FC50 **"SMS_MNG"** function and specify Message_No: 5.  Save and download the function.
3.	Call the **FC "SMS_MNG"** function in OB1.  Save and once again download the organization block.
4.	Make sure that the controller is ready to receive or send an SMS message. See step 4 of table 6.1.

No.	Action

5.	Write an SMS message which consists of ”?Temp1” and send it to the S7 station.
	`// Orders`
	`Order : ARRAY [1 .. maxOrder] OF STRING [10] :=`
	`'OnM50.1', 'xxxx', 'xxxx', 'xxxx','xxxx',`
	`'OffM50.1', '?Temp1' ;`

6.	Once the message is received, the receive text is evaluated. If the SMS message has an ‘Order’ as content, the assigned Order_No (e.g., no. 7 for ‘?Temp1’) will be output on the parameter of **FB “SMS_MeId”** and saved in the flag word (MW12) in OB1. A feedback message with message text (Message_No: 5) and temperature value (MW16) will be automatically sent back to the sender of the query.

Limitation

If the modem receives more than one SMS requesting data in a polling cycle, the PLC application program will only answer, at maximum, one SMS.

To answer more than one SMS after one cycle it is necessary to extend the programm with a buffer for received SMS, etc.

7 Further Information

RS 232 sniffer

Communications processor (CP) and MD720-3 are directly connected to one another via a serial cable (RS232 cable). Communication is full duplex, i.e. data can be simultaneously sent and received.

For this 1:1 communication, it is not possible to track the data traffic between the terminal units using a normal serial cable.

Especially for error analysis and data recording, it is desirable to listen to the RS232 communication. Therefore, a special cable must be manufactured.

The RS232 cable is extended by two additional ports, with each port tapping one communication direction (send/receive).**RS 232 pin assignment**

The following table shows the pin assignment for the RS232 interface:

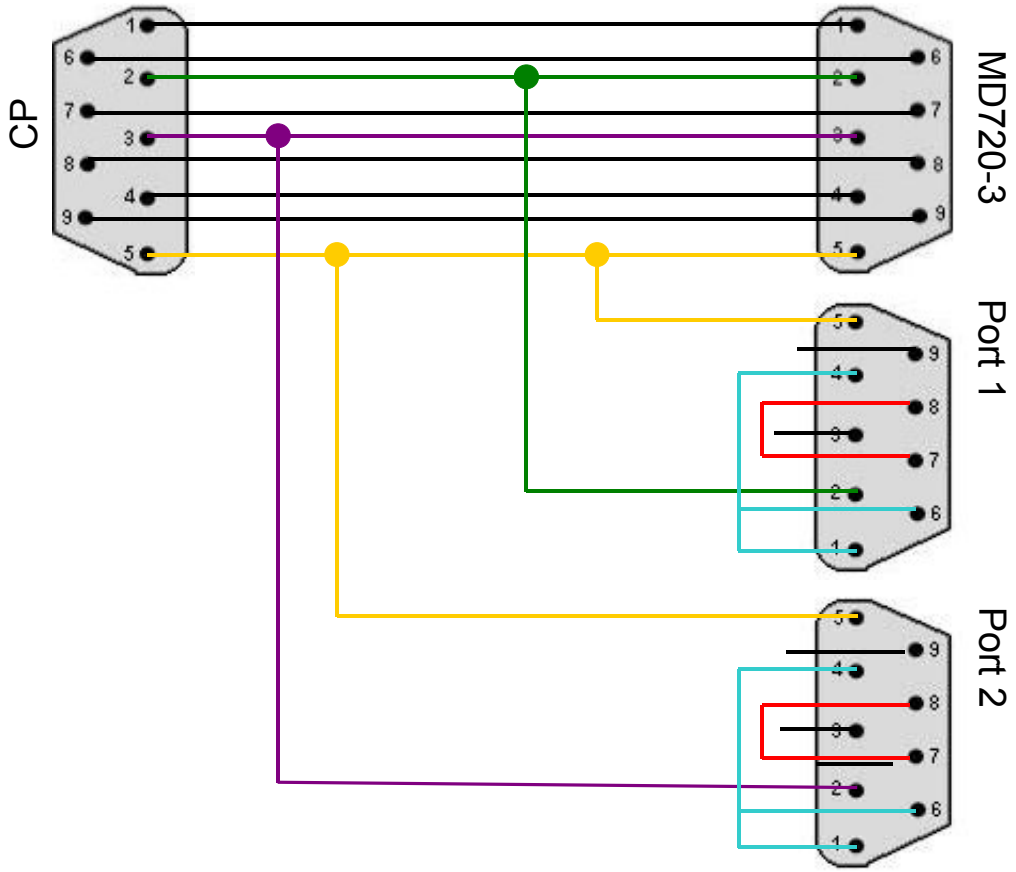
Table 7-1

Pin	Name	Meaning
1	DCD	(Data) Carrier Detect; MD720-3 signals to CP that it detects data on the line.
2	RxD	Receive Data; line for receive data.
3	TxD	Transmit Data; line for send data.
4	DTR	Data Terminal Ready; via this line, CP signals to MD720-3 that it is ready.
5	Gnd	Signal ground
6	DSR	Dataset Ready: Via this line, MD720-3 signals to CP that it is ready.
7	RTS	Request to Send; send request: MD720-3 can send data.
8	CTS	Clear to Send; MD720-3 can receive data.
9	RI	MD720-3 informs CP of an incoming call.

Connection diagram

The figure below shows the connection diagram for a full duplex RS232 sniffer. Port 1 taps the data of the receive line and port 2 taps the data of the send line.

Figure 7-1



If your PC has two serial interfaces, you can display the data traffic in both directions on a screen using a terminal program, e.g. HyperTerminal. A separate terminal must be opened for each data direction.

However, special RS232 sniffer software products are already available, which allow combining the data traffic of several ports on one screen. It is thus no longer necessary to switch between terminal windows.

8 References

Table 8-1

	Topic	Link
\1\	Reference to the document	http://support.automation.siemens.com/WW/view/en/25545680
\2\	Siemens Industry Customer Support	https://support.automation.siemens.com
\3\	MD720-3 manual	5http://support.automation.siemens.com/WW/view/en/23117745
\4\	CP 340 manual	http://support.automation.siemens.com/WW/view/en/1137332
\5\	First steps with the CP 340	http://support.automation.siemens.com/WW/view/en/12108826
\6\	CP 341 manual	http://support.automation.siemens.com/WW/view/en/1117397
\7\	First steps with the CP 341	http://support.automation.siemens.com/WW/view/en/1188622
\8\	CP 441-2 manual	http://support.automation.siemens.com/WW/view/en/1137419
\9\	First steps with the CP 441-2	http://support.automation.siemens.com/WW/view/en/1188835
\10\	SIMATIC ET 200S manual	http://support.automation.siemens.com/WW/view/en/9260793
\11\	PROFINET S7-300 CPU manual	http://support.automation.siemens.com/WW/view/en/48080216
\12\	WAN access methods	http://support.automation.siemens.com/WW/view/en/26662448

9 History

Table 9-1

Version	Date	Modification
V2.2	11/2014	Adaptions receive-sms scenarios
V2.1	02/2013	The screenshot in table 5-3, point 2 has been changed
V2.1	10/2011	The screenshot in table 5-3, point 5 has been changed
V2.0	07/2011	Total revision of V1.0
V1.0	16/07/2007	First edition