Data Transfer from the SIPLUS CMS2000 TCP/IP Data Interface to an S7-300CP

SIPLUS CMS2000

Application Example May 2012

Service & Support

Answers for industry.

SIEMENS

Warranty and Liability

This article originates from the Service & Support Portal of Siemens AG, Sector Industry, Industry Automation and Drive Technologies. The respective terms of use are applicable (www.siemens.com/nutzungsbedingungen).

This document can be directly downloaded via the following link:

http://support.automation.siemens.com/WW/view/en/63326108

Note

The application examples are non-binding and do not claim to be complete in terms of configuration and equipment or to consider any other contingencies. The application examples do not represent any customer-specific solutions, but are only intended to provide support with typical tasks. The reader shall be solely responsible for assurance of the described products' proper operation. These application examples do not exempt the reader from his obligation to due diligence with regard to application, installation, operation and maintenance. By utilizing these application examples, the reader acknowledges that Siemens cannot be held liable for any damage beyond the liability provisions described herein. We reserve the right to implement changes in these application examples at any time and without prior notice. In case of deviations between the suggestions relating to these application examples and other Siemens publications, e.g. catalogs, the contents of such other documentation shall take priority.

We do not assume any warranty for the information contained in this document.

Our liability for any damage caused by the utilization of examples, notes, programs, configuration and performance data, etc., described in this application example shall be excluded, irrespective of the legal ground, unless liability is legally imposed, e.g. in accordance with the German product liability act ("Produkthaftungsgesetz"), in cases of intent, gross negligence, injury to life, body or health, assumption of a product's guarantee of quality, fraudulent concealment of defects or violation of fundamental contractual obligations. However, claims for damages on the grounds of violation of fundamental contractual obligations shall be limited to the contract-typical, foreseeable damage, except for cases of intent or gross negligence or where liability is legally imposed due to injury to life, body or health. This does not imply a change in the burden of proof to your detriment.

Any transfer or reproduction of these application examples or excerpts thereof shall not be permitted, unless expressly approved by Siemens Industry Sector.

Table of Contents

Warranty and Liability 3

1 Introduction 6

1.1 Application Target 6
1.2 Prerequisites 6
1.3 Differentiation 6

2 SIPLUS CMS2000 7

2.1 General Information 7
2.2 Configuration 7

3 S7-300 Project 13

3.1 General Information 13
3.2 NetPro Configuration 13
3.3 Generation of Function Blocks 15
3.4 Function of Blocks 17
3.5 Integration in the Program 18

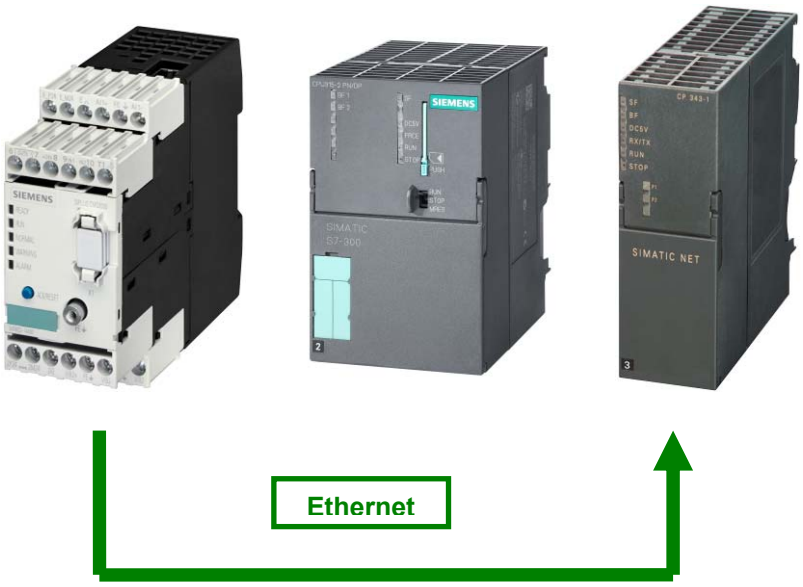
4 Annex 19

4.1 References 19

5 History 20

1 Introduction

1.1 Application Target



Drawing 1: Schematic representation of the application example

The target of this application example lies in the demonstration of the cyclic transfer of current data from a SIPLUS CMS2000 system to an S7-300 controller via Ethernet telegram.

1.2 Prerequisites

For SIPLUS CMS 2000:

- Firefox V3.6.x

For Step 7 V5.4:

- Knowledge of SIMATIC Step 7 Basic
- System requirements
  - The conditions of SIMATIC Step 7 Basic (V5.4 SP5) are applicable

1.3 Differentiation

This application example is not readily suitable for process monitoring, but merely describes the data transfer. The data's processing, such as release of an alarm or generation of a warning message, has to be carried out by the user.

2 SIPLUS CMS2000

2.1 General Information

Layout

Communication with the S7-300 controller is realized via TCP/IP. The SIPLUS CMS2000 system features a data interface which can be used with minimum expenditures.

Connection is realized via RJ45 connectors, i.e. conventional twisted pair wiring. Configuration (IP, netmask,…) is subject to the same conditions as all Ethernet networks.

Application

The SIPLUS CMS2000 system offers two different telegram types. The compact telegram contains diagnostics data only, while the extended telegram additionally contains 64 measured values and information on the calculated spectra. Configuration of SIPLUS CMS 2000 is explained below.

2.2 Configuration

To configure SIPLUS CMS2000, a PC is connected to the device via an Ethernet interface. The device now tries to reach a DHCP server in the connected network. If none can be found, its default configuration IP is 192.168.1.160. The device can then be reached via the Mozilla Firefox V3.6.x web browser under this IP.

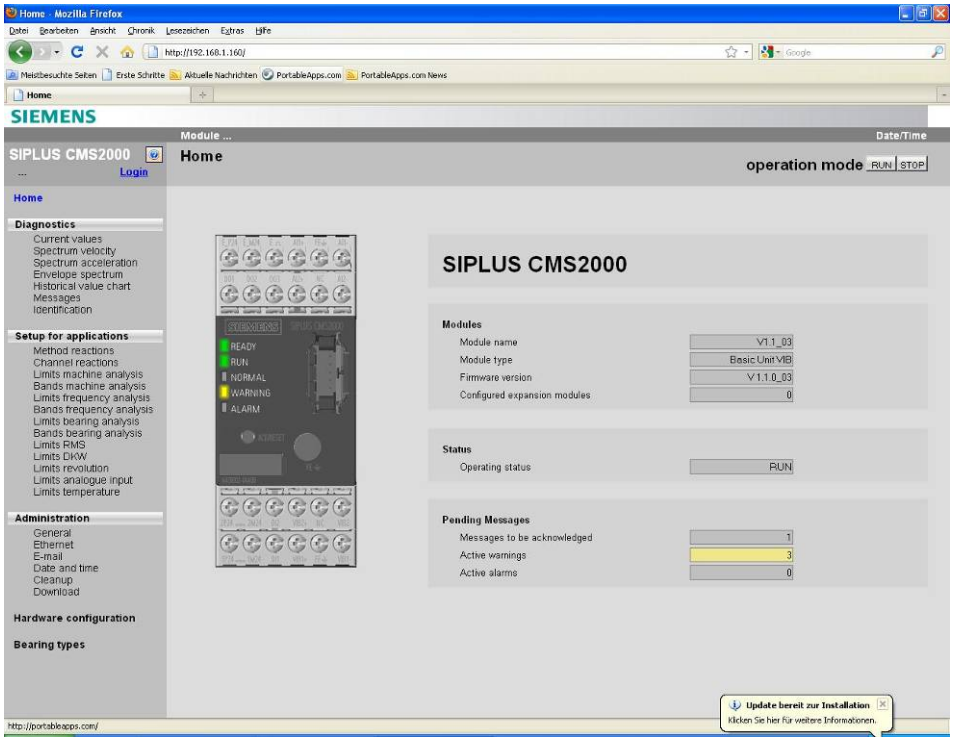


Figure1: SIPLUS CMS 2000 home menu

The user has to log in prior to making any settings. In the default settings, the user name is "admin" and the password "0000".

The "Ethernet" input mask in which the network settings can be edited can be accessed via the "Administration" sub-menu.

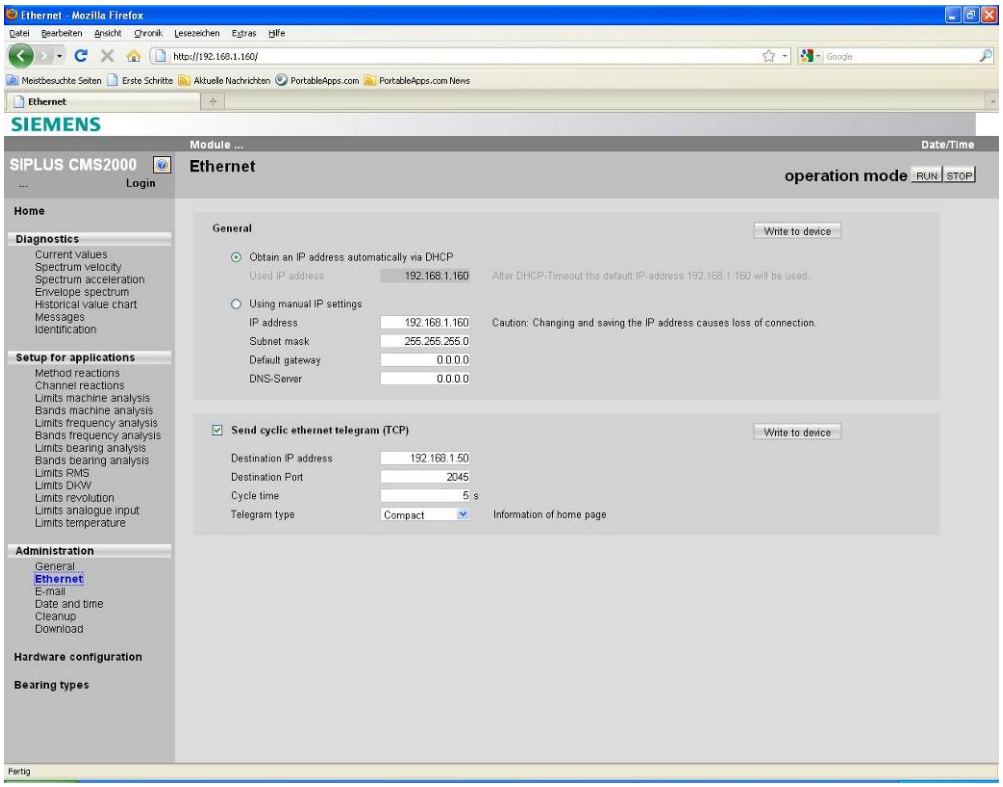


Figure 2: SIPLUS CMS2000 network menu

All settings important for this application example are made in this menu. If "Send cyclic ethernet telegram (TCP)" is ticked, the SIPLUS CMS2000 module tries to send a TCP telegram to the target socket specified below. The desired periodicity has to be set under "Cycle time" and starts at 2 seconds. The IP of the target socket is the CP of S7-300, the port has to be the same which is set later (see 3.2 NetPro Configuration).

Next, the telegram type has to be specified. A differentiation is made between telegrams which are to contain the diagnostics data of SIPLUS CMS2000 only and telegrams which are to additionally transfer measured values (RMS = root mean square, DKW = diagnostics characteristic values) and information on the calculated spectra.

The information from this part are also contained in the manual of the SIPLUS CMS2000 system.

Length [bytes]	Designation	Data type	Description
1	Operating state	Unsigned char	Operating state of the SIPLUS CMS2000 device: 3 = Fault state ("Not ready") 4 = Stop ("Ready") 5 = Measuring 6 = Teaching 7 = Run-inhibit 8 = Run Note: Telegram transfer is effected in all operating states listed above. Temporary operating states such as start-up or shutdown are not specified here as no telegrams are transferred in these states.
1	State LED/DO "Normal" (green)	Unsigned char	Current state of the LED and the digital output "Normal" (green): 0 = OFF 1 = ON
1	State LED/DO "Warning" (yellow)	Unsigned char	Current state of the LED and the digital output "Warning" (yellow): 0 = OFF 1 = ON 2 = Slow flashing (0.5 Hz) 3 = Fast flashing (2 Hz)
1	State LED/DO "Alarm" (red)	Unsigned char	Current state of the LED and the digital output "Alarm" (red): 0 = OFF 1 = ON 2 = Slow flashing (0.5 Hz) 3 = Fast flashing (2 Hz)
1	Active process alarms	Unsigned char	Number of currently active process alarms
1	Active system alarms	Unsigned char	Number of currently active system alarms
1	Active process warnings	Unsigned char	Number of currently active process warnings
1	Active system warnings	Unsigned char	Number of currently active system warnings
1	Process messages to be acknowledged	Unsigned char	Number of process messages to be acknowledged which have not yet been acknowledged
1	System messages to be acknowledged	Unsigned char	Number of system messages to be acknowledged which have not yet been acknowledged
6	Reserve	Unsigned char[6]	Reserve fields for future use

Table 1: Use data structure with the compact telegram

Length [bytes]	Designation	Data type	Description
16	Use data of the compact telegram	Structure	Data structure according to Table 1
256	Measured values	Float[64]	Field with 64 measured values / characteristic values in IEEE 32-bit floating point format; indexation see below *(hereinafter referred to as "measured values")*
128	Measured value state	Unsigned short[64]	Field with 1 status word (16 bit) for each measured value; indexation see blow; assignment as follows: Bit 0: 0 = Measured value not configured 1 = Measured value configured Bit 1: 0 = Signal not detected or incorrectly detected 1 = Signal correctly detected Bit 2: 0 = Measured value not calculated / not valid 1 = Measured value calculated and valid Bit 3: 0 = "not green" 1 = "**green**": Measured value is monitored, no threshold limit violation Bit 4: 0 = "not yellow" 1 = "**yellow**": Measured value is monitored, warning threshold violated Bit 5: 0 = "not red" 1 = "**red**": Measured value is monitored, alarm threshold violated Bit 6: 0 = No warning acknowledgement required 1 = Warning has to be acknowledged for this measured value Bit 7: 0 = No alarm acknowledgement required 1 = Alarm has to be acknowledged for this measured value Bit 8-15: *Reserved*

Indexation of measured values or measured value state information in the above fields:

Index	Measured value	Index	Measured value	Index	Measured value	Index	Measured value
0	REV	16	*Reserved*	32	*Reserved*	48	*Reserved*
1	OPR_HOURS	17	*Reserved*	33	*Reserved*	49	*Reserved*
2	AI 1	18	VIB 1: RMS	34	*Reserved*	50	*Reserved*
3	AI 2	19	*Reserved*	35	*Reserved*	51	*Reserved*
4	TEMP 1.1	20	*Reserved*	36	VIB 1: DKW	52	*Reserved*
5	TEMP 1.2	21	*Reserved*	37	*Reserved*	53	*Reserved*
6	TEMP 1.3	22	*Reserved*	38	*Reserved*	54	*Reserved*
7	TEMP 2.1	23	*Reserved*	39	*Reserved*	55	*Reserved*
8	TEMP 2.2	24	*Reserved*	40	*Reserved*	56	*Reserved*
9	TEMP 2.3	25	*Reserved*	41	*Reserved*	57	*Reserved*
10	*Reserved*	26	*Reserved*	42	*Reserved*	58	*Reserved*
11	*Reserved*	27	VIB 2: RMS	43	*Reserved*	59	*Reserved*
12	*Reserved*	28	*Reserved*	44	*Reserved*	60	*Reserved*
13	*Reserved*	29	*Reserved*	45	VIB 2: DKW	61	*Reserved*
14	*Reserved*	30	*Reserved*	46	*Reserved*	62	*Reserved*
15	*Reserved*	31	*Reserved*	47	*Reserved*	63	*Reserved*

<table>
<tr><th>Length [bytes]</th><th>Designation</th><th>Data type</th><th>Description</th></tr>
<tr><td>128</td><td>Spectrum state</td><td>Unsigned short[64]</td><td>Field with 1 status word (16 bit) for each spectrum
v(f) / a(f) / env(f); indexation see blow; assignment as follows:
Bit 0: 0 = Spectrum not configured
1 = Spectrum configured
Bit 1: 0 = Vibration signal not or incorrectly detected
1 = Vibration signal correctly detected
Bit 2: 0 = Spectrum not calculated / not valid
1 = Spectrum calculated and valid
Bit 3: 0 = "not green"
1 = "green": Spectrum is monitored, no threshold value violation
Bit 4: 0 = "not yellow"
1 = "yellow": Spectrum is monitored, warning threshold violated
Bit 5: 0 = "not red"
1 = "red": Spectrum is monitored, alarm threshold violated
Bit 6: 0 = No warning acknowledgement required
1 = Warning has to be acknowledged for this spectrum
Bit 7: 0 = No alarm acknowledgement required
1 = Alarm has to be acknowledged for this spectrum

High-byte bit assignment depends on the spectrum type:

The following applies to env(f):
Bit 8-11: Reserved
Bit 12: 0 = Inner bearing ring unaffected
1 = W/A threshold for inner bearing ring violated
Bit 13: 0 = Outer bearing ring unaffected
1 = W/A threshold for outer bearing ring violated
Bit 14: 0 = Bearing cage unaffected
1 = W/A threshold for bearing cage violated
Bit 15: 0 = Bearing rolling element unaffected
1 = W/A threshold for bearing rolling element violated

The following applies to v(f) and a(f):
Bit 8-15: Reserved</td></tr>
<tr><td></td><td colspan="3">Indexation of the spectrum state information in the above field:
<table>
<tr><th>Index</th><th>Measured value</th><th>Index</th><th>Measured value</th><th>Index</th><th>Measured value</th><th>Index</th><th>Measured value</th></tr>
<tr><td>0</td><td>VIB 1: v(f)</td><td>16</td><td>Reserved</td><td>32</td><td>Reserved</td><td>48</td><td>Reserved</td></tr>
<tr><td>1</td><td>Reserved</td><td>17</td><td>Reserved</td><td>33</td><td>Reserved</td><td>49</td><td>Reserved</td></tr>
<tr><td>2</td><td>Reserved</td><td>18</td><td>VIB 1: a(f)</td><td>34</td><td>Reserved</td><td>50</td><td>Reserved</td></tr>
<tr><td>3</td><td>Reserved</td><td>19</td><td>Reserved</td><td>35</td><td>Reserved</td><td>51</td><td>Reserved</td></tr>
<tr><td>4</td><td>Reserved</td><td>20</td><td>Reserved</td><td>36</td><td>VIB 1: env(f)</td><td>52</td><td>Reserved</td></tr>
<tr><td>5</td><td>Reserved</td><td>21</td><td>Reserved</td><td>37</td><td>Reserved</td><td>53</td><td>Reserved</td></tr>
<tr><td>6</td><td>Reserved</td><td>22</td><td>Reserved</td><td>38</td><td>Reserved</td><td>54</td><td>Reserved</td></tr>
<tr><td>7</td><td>Reserved</td><td>23</td><td>Reserved</td><td>39</td><td>Reserved</td><td>55</td><td>Reserved</td></tr>
<tr><td>8</td><td>Reserved</td><td>24</td><td>Reserved</td><td>40</td><td>Reserved</td><td>56</td><td>Reserved</td></tr>
<tr><td>9</td><td>VIB 2: v(f)</td><td>25</td><td>Reserved</td><td>41</td><td>Reserved</td><td>57</td><td>Reserved</td></tr>
<tr><td>10</td><td>Reserved</td><td>26</td><td>Reserved</td><td>42</td><td>Reserved</td><td>58</td><td>Reserved</td></tr>
<tr><td>11</td><td>Reserved</td><td>27</td><td>VIB 2: a(f)</td><td>43</td><td>Reserved</td><td>59</td><td>Reserved</td></tr>
<tr><td>12</td><td>Reserved</td><td>28</td><td>Reserved</td><td>44</td><td>Reserved</td><td>60</td><td>Reserved</td></tr>
<tr><td>13</td><td>Reserved</td><td>29</td><td>Reserved</td><td>45</td><td>VIB 2: env(f)</td><td>61</td><td>Reserved</td></tr>
</table></td></tr>
</table>

Length [bytes]	Designation		Data type		Description			
	14	*Reserved*	30	*Reserved*	46	*Reserved*	62	*Reserved*
	15	*Reserved*	31	*Reserved*	47	*Reserved*	63	*Reserved*

Table 2: Structure of the extended telegram

Tables 1 and 2 show how the SIPLUS CMS2000 device transfers the corresponding data. Their extraction from the TCP telegram is up to the S7-300 controller. The program can determine the type of sent telegram from the data index and length contained in the compact and extended telegram.

3 S7-300 Project

3.1 General Information

In this example, communication is realized via a CP343-1 module. The S7-300 has to contain such module which is connected to the same network as SIPLUS CMS 2000.

Caution: The S7-300 CPU organizes its memory in the **BigEndian format**. In contrast, the SIPLUS CMS2000 system uses the **LittleEndian format**. The supplied blocks output all data in the **BigEndian format**.

3.2 NetPro Configuration

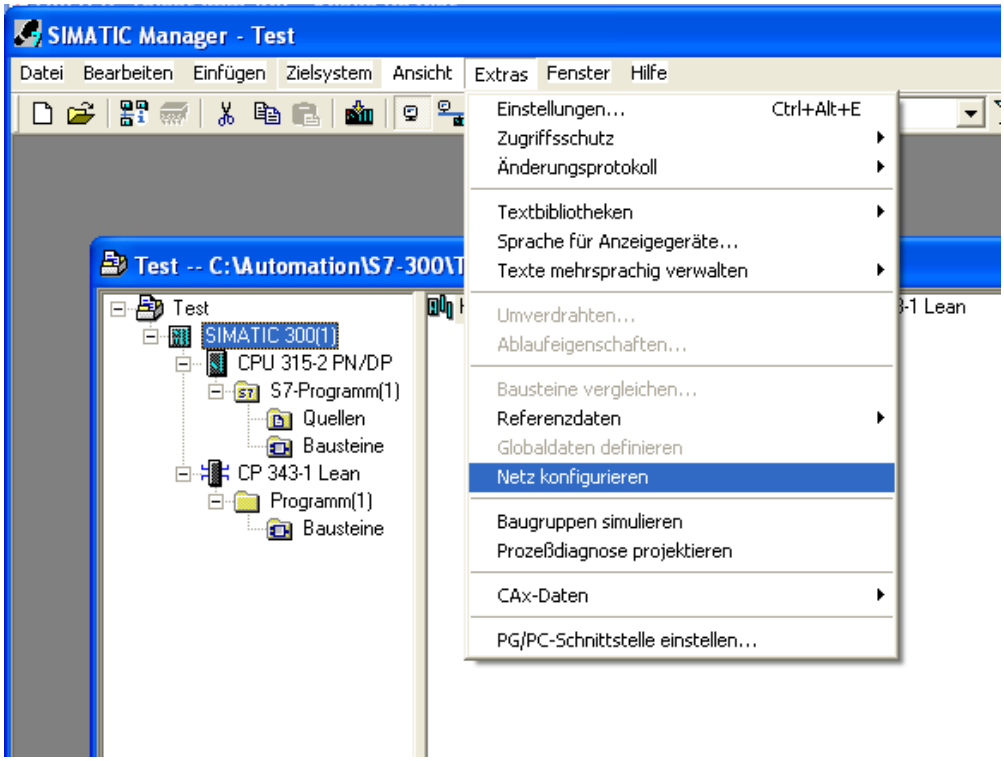


Figure 3: Simatic Manager

The menu for network configuration is called up in the Simatic Manager (Fig. 3). For configuration, the CPU is selected and the first line in the connection table is selected via a double-click, after which the window "Add new connection" opens (Fig. 4).

TCP is selected as connection type and confirmed with "OK". The window which opens next contains a "General" tab for specification of the connection's name (Fig. 5). As the connection is passive, this option is not ticked.

Under the "Addresses" tab, the port number via which the S7-300 is to be reached by the SIPLUS CMS2000 system is assigned (Fig. 6). In theory, all permissible port numbers can be selected. However, the number has to correspond to the number assigned during the configuration of SIPLUS CMS2000.

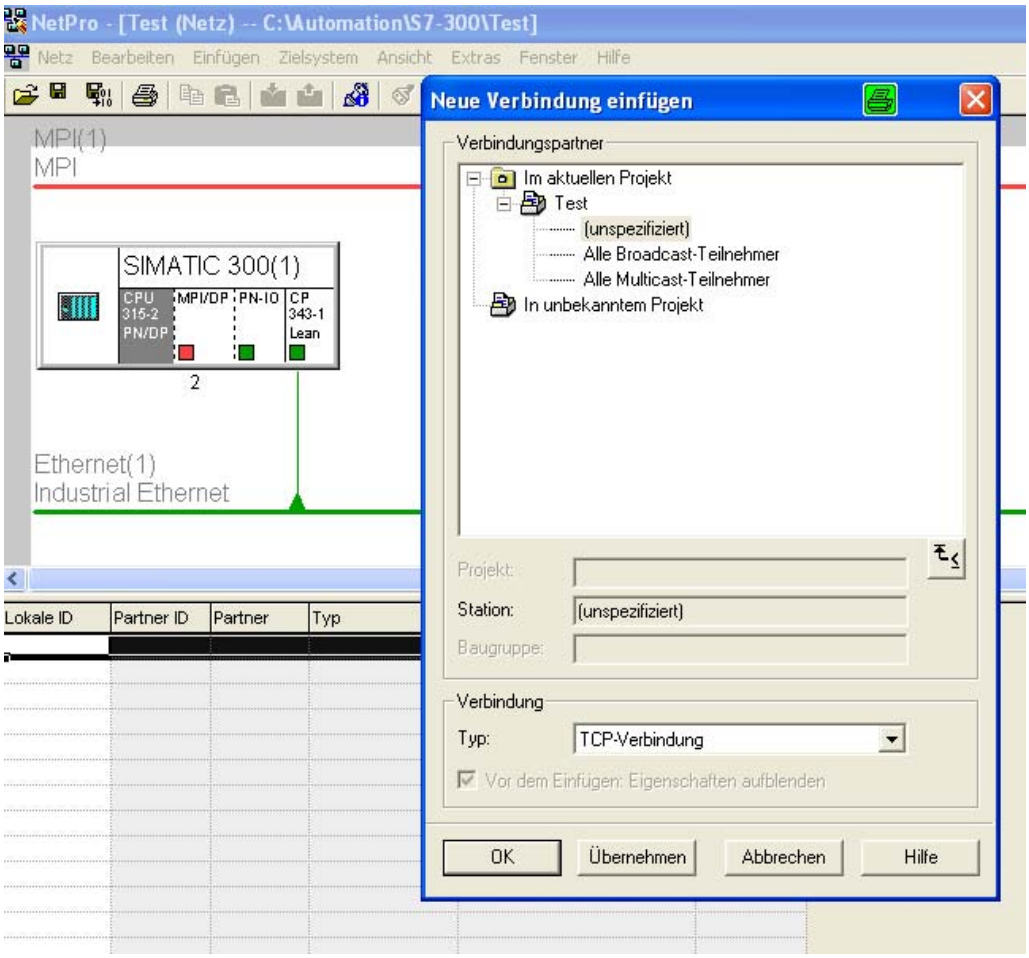


Figure 4: Creation of new connection in NetPro

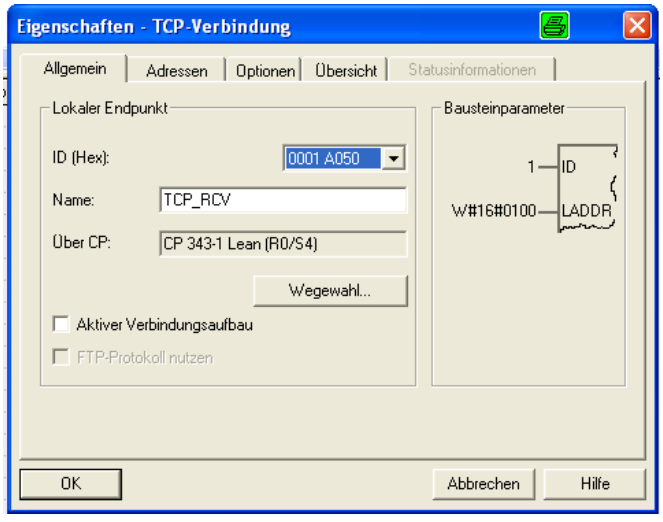


Figure 5: Properties of the new TCP connection

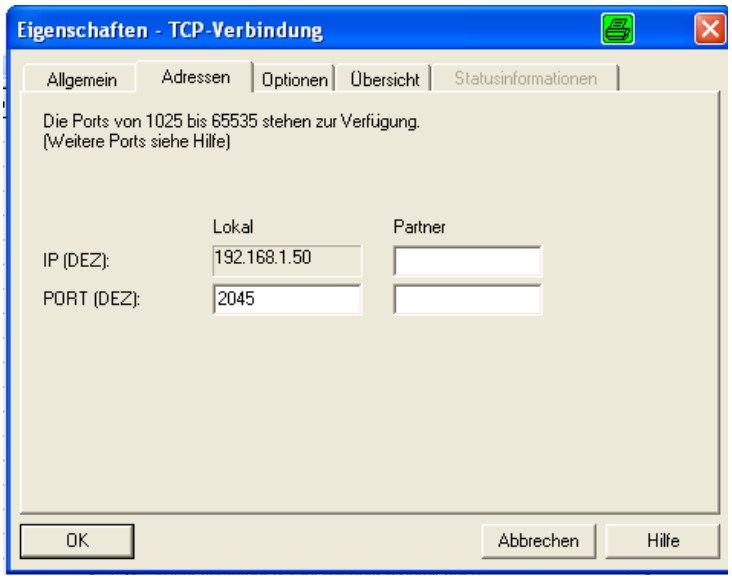


Figure 6: Port number assignment

NetPro configuration is completed with the establishment of this connection. NetPro ID and LADDR are later required as parameters for the S7 program's function blocks.

3.3 Generation of Function Blocks

The supplied library S7300_CMS2000TCP comprises all function blocks required for reception.

In the Simatic Manager, select File->Open. In the window which opens, select the tab "Library" and select S7300_CMS2000TCP. Open the program contained in the library and open the folder "Blocks".

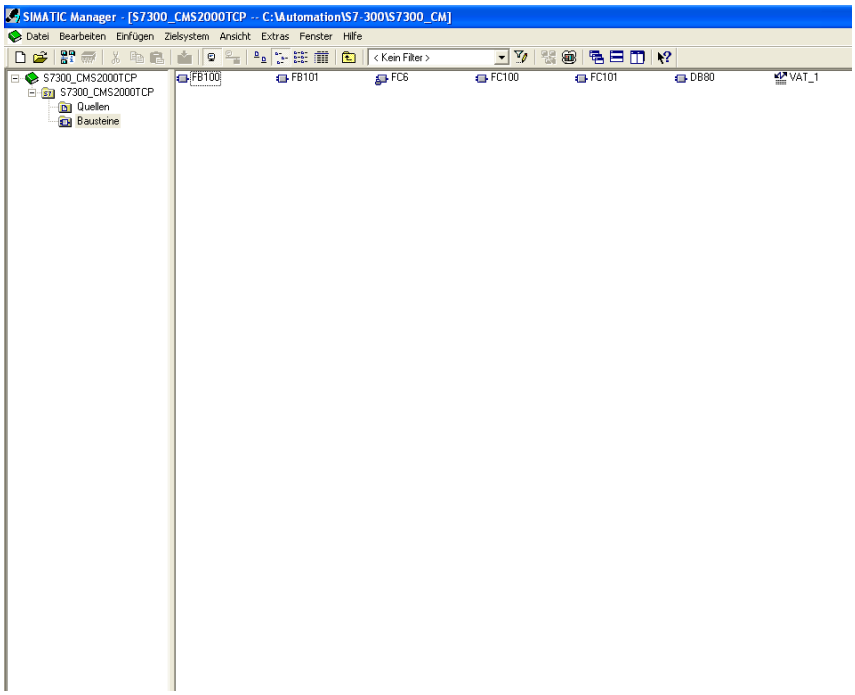


Figure 7: Library contents

Now, add FB100, FB101, FC6, FC100, FC101 under blocks. Please ensure that no FB or FC exist which have the same number as the blocks to be added. The data block DB80 and the variable table VAT1 can be used for test purposes. DB80 contains variables for all relevant signals of a vibration sensor. The variable table contains the addresses of these variables.

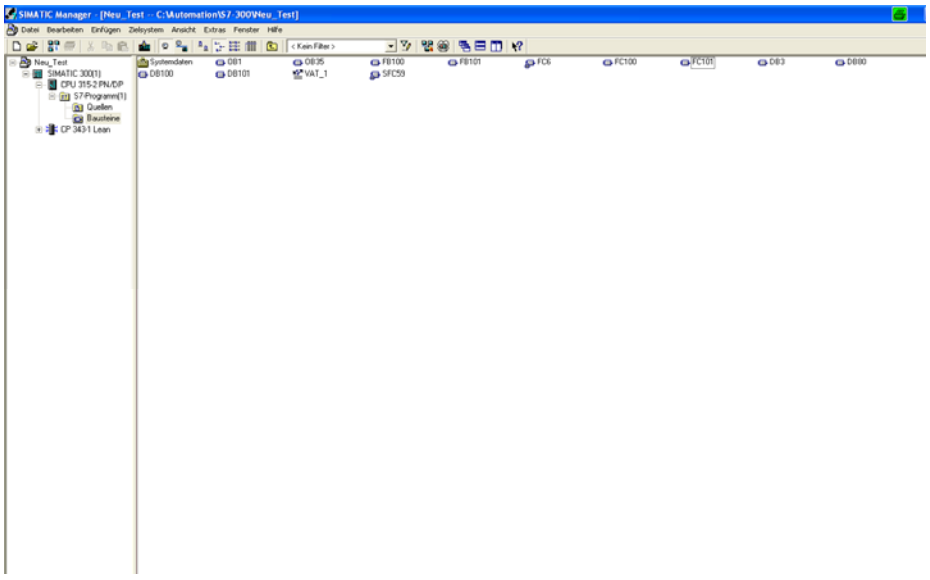


Figure 8: Project with all required blocks

3.4 Function of Blocks

```
CALL  "RCV-TCPProxy_EXTD" , DB101                                   FB101
 NetProID           :=1
 NetProLADDR        :=W#16#100
 Timeout            :=T1
 Wrong_TYPE         :="Test_Global_DB".Wrong_Type                   DB80.DBX48.0
 Timestamp          :=
 Length             :=
 Index              :="Test_Global_DB".Index                        DB80.DBW50
 EDelimiter         :=
 NewDataRcv         :=#NewDt
 Tel_Error          :=
 Tel_Timeout        :="Test_Global_DB".Tel_Timeout                  DB80.DBX68.0
 Tel_Counter        :=
 Op_status          :="Test_Global_DB".OP_Status                    DB80.DBB52
 LED1_GR            :="Test_Global_DB".LED1_GR                      DB80.DBB59
 LED2_YE            :="Test_Global_DB".LED2_YE                      DB80.DBB60
 LED3_RE            :="Test_Global_DB".LED3_RE                      DB80.DBB61
 Act_Proc_Alerts    :="Test_Global_DB".Act_Proc_Alerts              DB80.DBB53
 Act_Sys_Alerts     :="Test_Global_DB".Act_Sys_Alerts               DB80.DBB54
 Act_Proc_Warnings  :="Test_Global_DB".Act_Proc_Warnings            DB80.DBB55
 Act_Sys_Warnings   :="Test_Global_DB".Act_Sys_Warnings             DB80.DBB56
 T_Q_Proc_Messages  :="Test_Global_DB".T_Q_Proc_Messages            DB80.DBB57
 T_Q_Sys_Messages   :="Test_Global_DB".T_Q_Sys_Messages             DB80.DBB58
 CPCT_SPARE10       :=
 CPCT_SPARE11       :=
 CPCT_SPARE12       :=
 CPCT_SPARE13       :=
 CPCT_SPARE14       :=
 CPCT_SPARE15       :=
 DEV                :="Test_Global_DB".DEV                          DB80.DBD0
```

Figure 9: Integration of the FB

FB100

Processing of the compact telegram's data is carried out by FB100. The input parameters are: NetPro ID, CP module address and timeout.

The ID set under 3.2 is used as NetPro ID. The required module address (NetProLADDR) is shown in the same menu (Fig. 5). As shown in the figure, it is to be transferred as hex value (e.g. W#16#100).

In this example, a system timer of S7 was used as timer parameter. This timer is used in the FB to check whether a further telegram is received within 5 seconds. If this is not the case, the output bit "Tel_Timeout" is set. Through use of the external timer, FB100 can also be operated outside OB1 (watchdog interrupt OB, or similar).

All data of the compact telegram as well as the bits "NewDataRcv" and "Tel_Error" are present on the output interface of FB100. "Tel_Error" checks whether the telegram's structure is correct and "NewDataRcv" outputs a positive edge with a new telegram. The integer output variable "Tel_Counter" is incremented with each received telegram.

At the start, FB100 calls up FC6 (AG_RCV), a standard function which opens the port selected under 3.2 and waits for an active connection parameter. After completion of the data transfer, the data are present in a prepared buffer. The data size of the DB is selected in a way which ensures that it can both accommodate compact as well as extended telegrams.

This is to avoid buffer overflows in case of incorrect parameterization/programming. The output bit "Wrong_TYPE" of FB100 indicates whether the correct telegram type was received. In case of fault, it is HIGH. In this case, further processing of FB100 is cancelled.

FB101: "RCV-TCPProxy_EXTD"

FB101 serves the reception of the extended telegram. Yet, it also contains an internal FB100 so to speak as the compact telegram is contained in the extended telegram. The input parameters are the same as with this block.

Also all outputs of FB100 are contained in those of FB101. In addition, it contains all data from the extended telegram. This block queries whether the complete telegram was received and sets the bit "Wrong_TYPE" to HIGH if it receives the compact telegram instead. Also FB101 cancels any further processing when the "Wrong_TYPE" bit is set.

FC100: "LDW_To_BREAL"

FC100 is only used in FB101 for conversion of a 32 bit floating point number received in LittleEndian format into BigEndian format and return transfer as REAL value.

FC101: "DW_TO_H_L_WORD"

Also FC101 is only used in FB101. This function divides the DWORD present on the input into a "High" and "Low" WORD. They are then present as output words.

FC6: "AG_RCV"

NetPro block from the "SIMATIC_NET_CP" library. It is both used in FB100 and FB101 and is responsible for data reception via the Ethernet or TCP/IP protocol.

3.5 Integration in the Program

First, the telegram type to be received has to be specified. Accordingly, either FB100 (compact) or FB101 (extended) has to be called up in an operation block. As S7-300 takes over the passive part with this connection, FB100/FB101 can run in OB1 (see Fig. 9).

4 Annex

4.1 References

This list is by no means complete and merely reflects a selection of suitable literature.

Table 4-1

	Subject field	Title
[1]	SIPLUS CMS 2000	SIPLUS CMS - User Manual http://support.automation.siemens.com/WW/view/de/40862689

5 History

Table 5-1

Version	Date	Modification
2012-05	05/30/2012	First edition