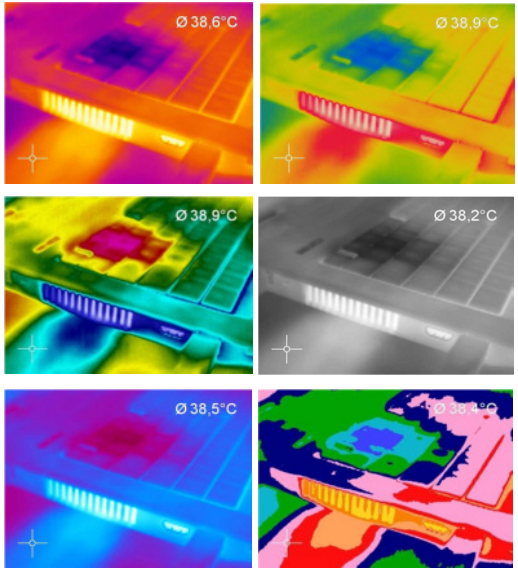
optris® PI160, PI200, PI400, PI450
Simatic S7 communication



Manual

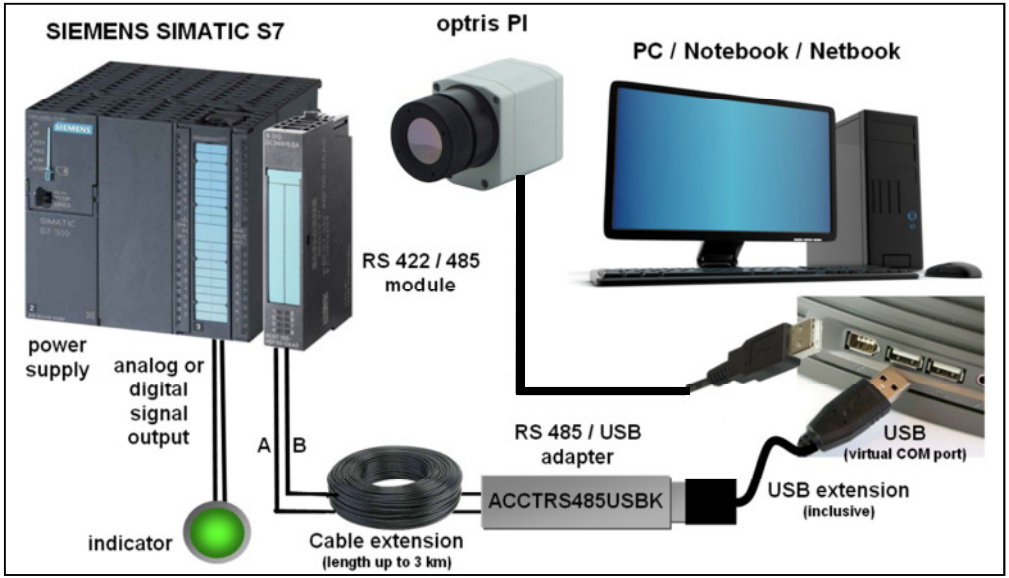
optris
infrared thermometers

Table of contents

1. Hardware specification

Regarding long distances (up to 3 km) between the infrared camera Optris PI and a SPS master system there is now an easy solution available.

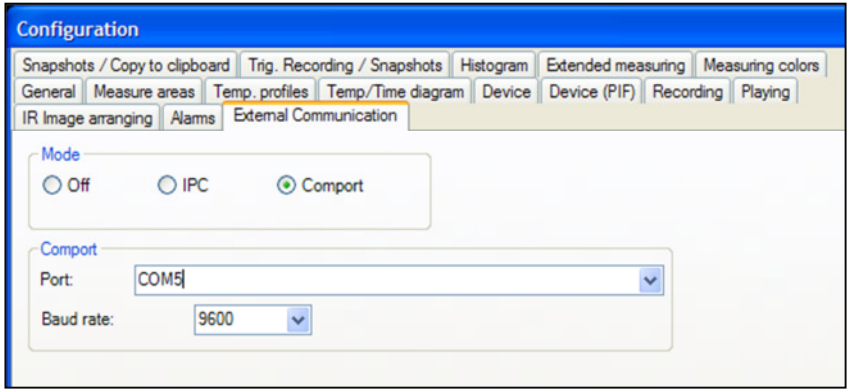
With the Optris RS485 Kit (product code: ACCTRS485USBK) you can connect the USB port of your PC with a RS485 module of a SPS master system



Picture: Optris PI infrared camera connection to a SPS master system (SIEMENS S7)

With the software CD which is included at the RS485 kit you have the possibility to create a virtual COM port at your PC.

To connect the software PI Connect with the RS485 kit is very easy.
Activate in the software PI Connect at the menu TOOLS – CONFIGURATION - EXTERNAL COMMUNICATION the mode “Comport” and select the COM port regarding your RS485 connection.



On the other site of the RS485 kit you have an A and B cable connected to the RS485 module of the SIEMENS S7 SPS.

Example of the hardware configuration which shows you a RS485 connection
between the optris PI camera to the SIEMENS S7 SPS which includes a PROFIBUS module and a RS485 module:

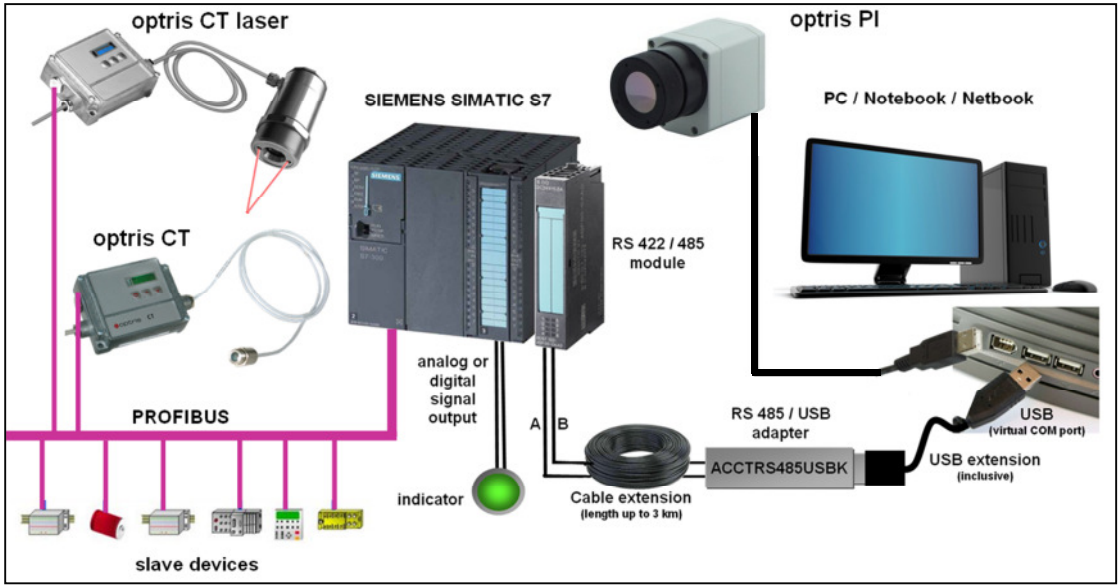
- optris PI 160 / optris PI200 / optris PI400 / optris PI450
- RS485 kit (ACCTRS485USBK)

- SIMATIC S7-300, CPU 313C-2 DP
  (processor with a PROFIBUS module)
- SIMATIC S7-300, front module with digital and analog outputs
  (switch actors, for example signal lights)

- SIMATIC S7-300, CP 340
  (communication processor with RS422/485 interface)

- Software STEP 7
  (to configure the SPS regarding the RS485 and the PROFIBUS module)

With the optris PROFIBUS kit you will get a GSD file to install each CT / CTlaser sensors at the SPS.



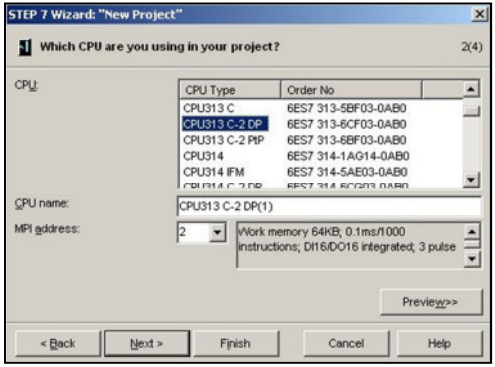
Picture**:** Optris PI infrared camera connection to a SPS master system (SIEMENS S7) included a PROFIBUS network with slave devices

All implemented commands for the Optris PI are described in the „Serial Communication Description.doc“ which is included on the “PI Connect” Software.

2. Create a new project

The best way to implement a new project in STEP 7 is to use the internal wizard. Step 1 of the wizard program only describe the functionality of the program. “Next” creates the project step by step. To create a project according to the preview it is necessary to activate “Finish”.

In Step 2 the user can choose the right CPU corresponding to his project. All other hardware settings are automatically set, based on this CPU.
If there are more than one CPU in one project the name of this specific CPU can be set.
The Multi Point Interface (MPI) address set the communication address between the CPU and the STEP 7 software computer.

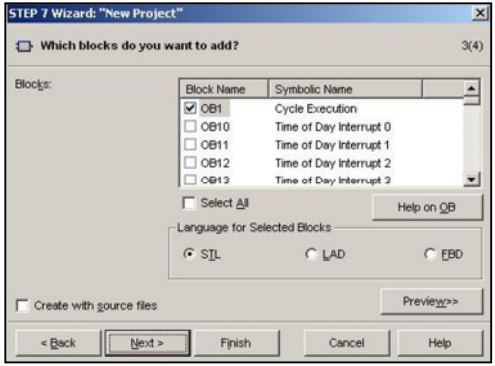


The communication between the PI and the Siemens S7 system is a cycle execution process. To add this communication process the “OB1” for “Cycle Execution” should be selected.

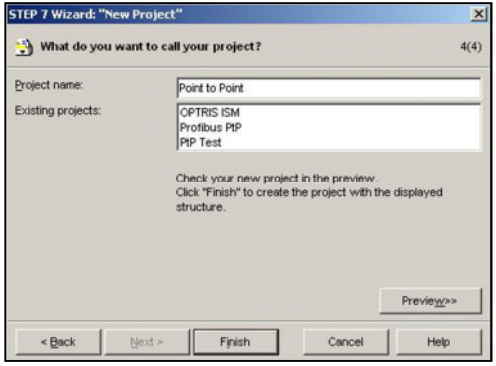
The language options for the selected blocks are

LAD: Ladder Logic
FBD: Function Block diagram
STL: Statement List

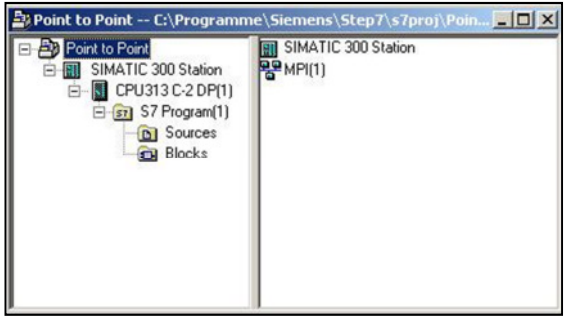
The following project is explained with LAD language.



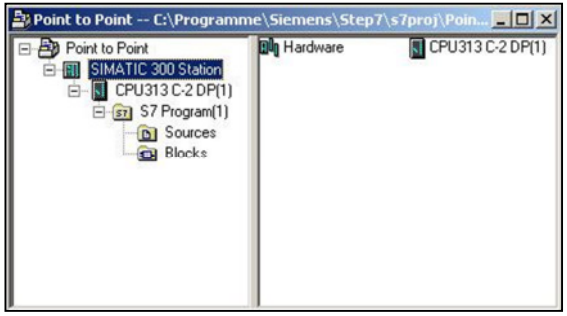
The last picture of the wizard program shows the existing projects and permits to enter the new project name.



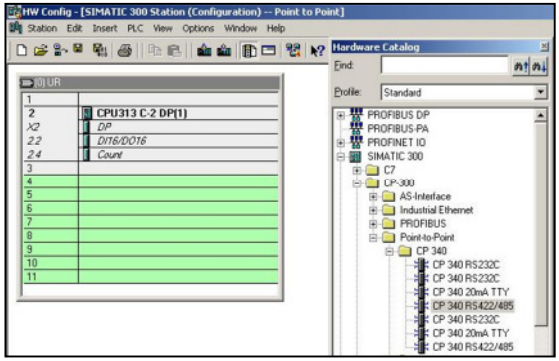
3. Hardware Settings



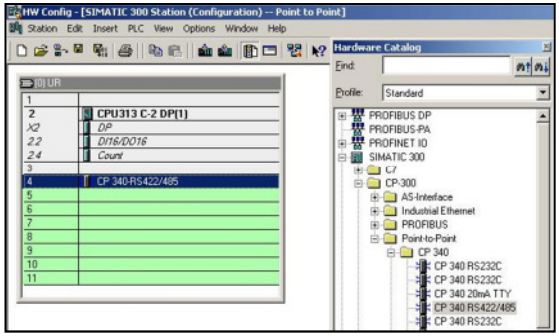
The process tree of this new “Point to Point” program includes the SIMATIC Station, the CPU and the S7 program.



The Hardware configurations of the system and the selected CPU are part of the SIMATIC 300 Station. In the Hardware configuration menu the user has the possibility to add a “Point to Point” module manually.

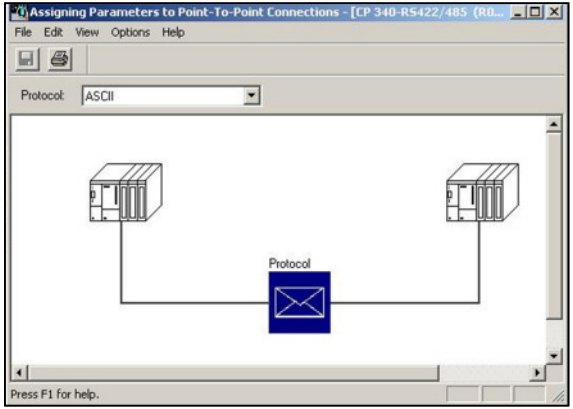


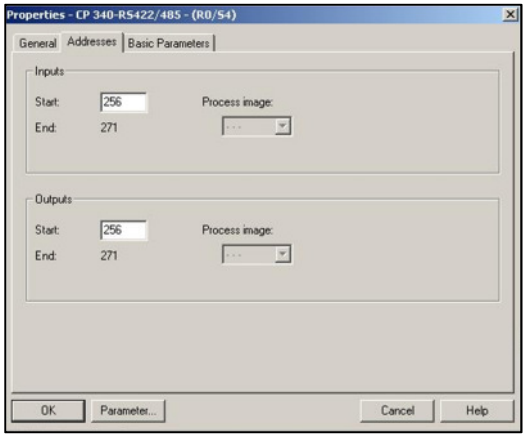
The Hardware configuration window shows the reserved slots for the CPU (Profibus DP, Digital Input / Digital Output DI16/DO16 and Counter).
The green slots indicate the open positions for extra Modules like "Point to Point" communication.



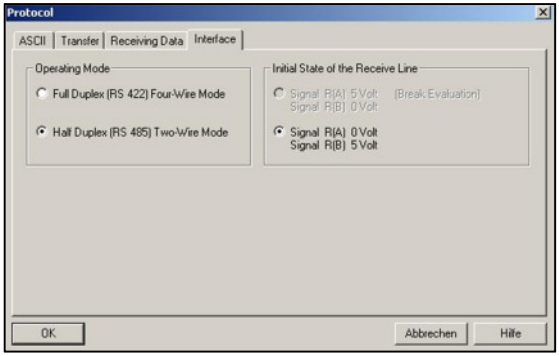
The CP340 module has three different types of communication protocols (RS232, 20mA TTY, RS 422/485). For a communication with the PI via USB-RS485 converter (ACCTRS485USBK) the CP340 RS422/485 module must be dragged and released on a free slot.

A double click on the reserved CP340 slot leads to configuration menu of the CP340 module.

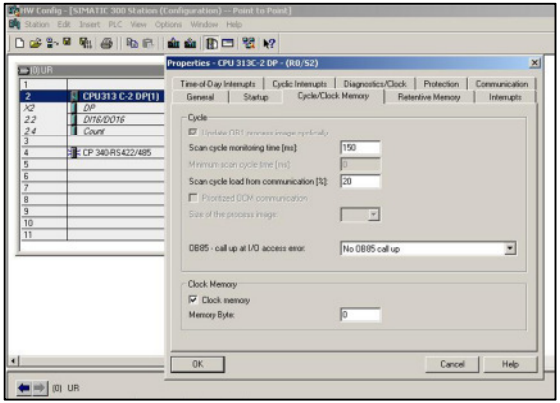




The “Input” and the “Output” Addresses of the CP340 specify the internal address range of the communication data.
The optrisPI communication operates with an ASCII protocol structure.
The settings for the CP340 protocol structure can be set inside the “Parameter” menu.
A double click on the “Protocol” envelope guides to the protocol settings.
For a correct communication with the PI it is necessary to set the baudrate to 9600bps, 8 Data Bits, 1 Stop Bits and Parity None.



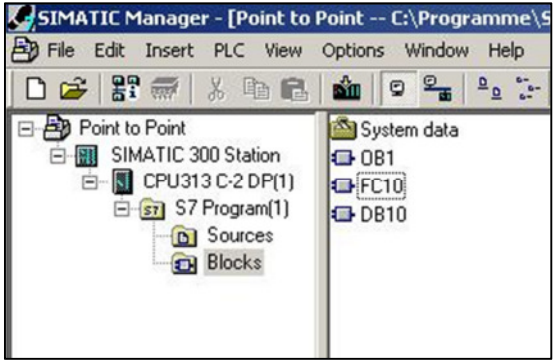
The “Operation Mode” defines the Full Duplex (Four Wire) or the Half Duplex (Two Wire) mode.
The communication with the optrisPI via ACCTRS485USBK works in a Half Duplex (RS485) mode.



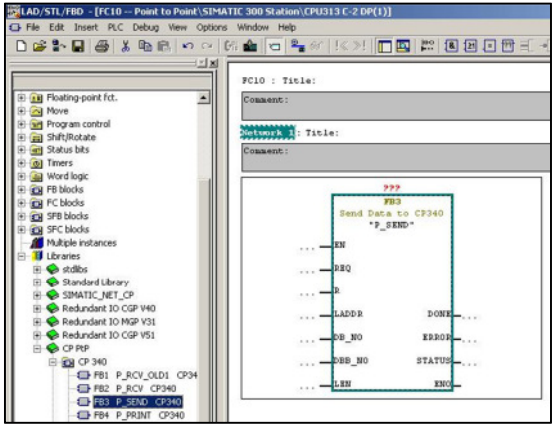
The implemented communication process needs a clock memory bit to start every “sending” cycle.
In the factory default settings of the Siemens S7 300 is this option deactivated.

Clock Memory Bit	Mx.7	Mx.6	Mx.5	Mx.4	Mx.3	Mx.2	Mx.1	Mx.0
Frequency in Hz	0,5	0,625	1	1,25	2	2,5	5	10
Periodic time in seconds	2	1,6	1	0,8	0,5	0,4	0,2	0,1

4. Software programming

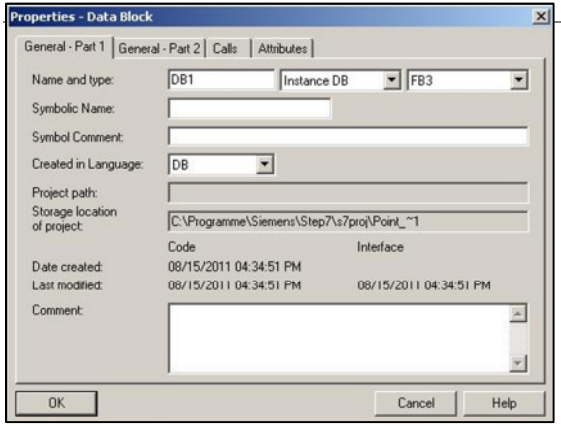


At the beginning of the programming process it is necessary to enter two different blocks into the system.
The “Function Block” (FC10) allows the creating of function schematic block diagrams.
Inside the “Data Block” (DB10) it is possible to generate all data structures which are needed for the “Send”, “Receive” and “Post Processing” processes.
The best way for the beginning is to use the FDB (Function Block diagram) program language.

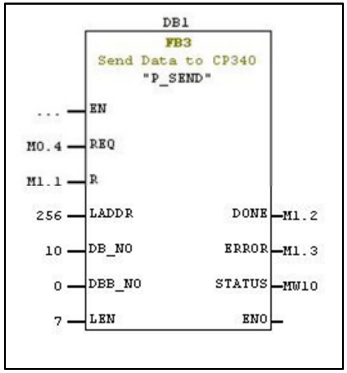


For the “Send Data” process via the CP340 module, special “Function Blocks” (FB) are included inside the Siemens S7 Software.

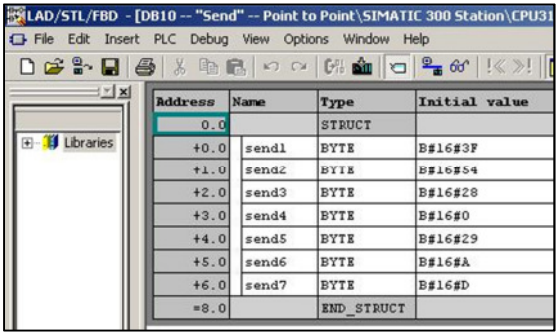
REQ:	Initiates request
R:	Aborts request
LADDR:	The basic address of the CP340
DB_NO:	Data block number
DBB_NO:	Data byte number
LEN:	Data Length
DONE:	Request completed without errors
ERROR:	Request completed with errors
STATUS:	Error specification



Each of this FB's for the data communication needs an "Instance Data Block" to save the internal settings.
This Instance Data Blocks are integrated in the system like normal Data Blocks.
The only setting which must be changed is from "Shared DB" to "Instance DB".
The FB3 (Send Data to CP340) from the FC10 Block is automatically set inside the properties.
All other setting can be let like shown in the picture.



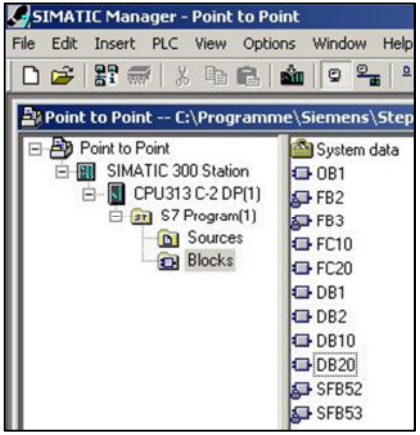
With the Clock Byte M0.4 the frequency of the initiates request is 1,25Hz.
The "Input Address" and the "Data Block Number" are the same we entered in the previous settings.
The first byte of the "Data String" should be the byte 0.
The length of the "Data String" corresponding to the specific "PI Command".
To read the temperature of a measurement area with Index x is the command ?T(x).
Any command must end with CR/LF (0x0D, 0x0A).
The "DONE", "ERROR" and "STATUS" values are assign to the M1.2, M1.3 and MW10 parameters.



The next step is to enter the “PI Command” inside the DB10.
For a better survey it is helpful to use a single Byte structure instead of an ARRAY or STRING structure.

BYTE NUMBER	0	1	2	3	4	5	6
CHARACTER	?	T	(	0	)	LF	CR
ASCII HEX	3F	54	28	00	29	0A	0D

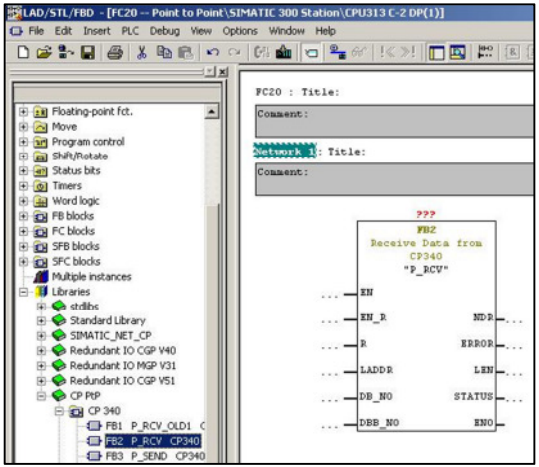
The initial value describes the start value if the system is set from STOP to RUN mode.



After the three blocks (FC10, DB1 and DB10) for the “Send Data” process are inserted in the system, every “SIMATIC System” needs a special block for the “Receive Data” process.

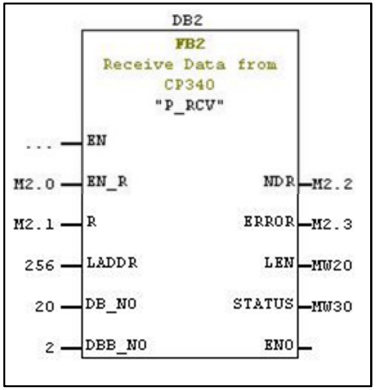
The first steps to implement this process are the same like for the “Send Data” process.

For a better overview it is good to create three blocks named (FC20, DB2 and DB20).

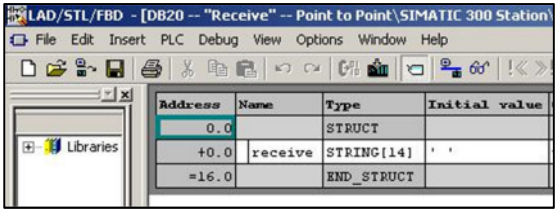


For the “Receive Data” process a special “Function Blocks” (FB) are included again inside the Siemens S7 Software.

EN_R:	Enable data read
R:	Aborts request
LADDR:	The basic address of the CP340
DB_NO:	Data block number
DBB_NO:	Data byte number
NDR:	Request completed without errors
ERROR:	Request completed with errors
LEN:	Length of message received
STATUS:	Error specification



The Booleans for the enable data read and the abort request should be set with marker bytes (M2.0, M2.1).
The basic address of the CP340 is the same like for the sending process.
The data block number specifies the data block to save the incoming data (DB20).
The Data byte number defines the byte position of the incoming string which should be the first saved byte.
The byte 0 and 1 of the incoming string are uninteresting (Byte 0 = maximal length, Byte 1 = real length of the string).

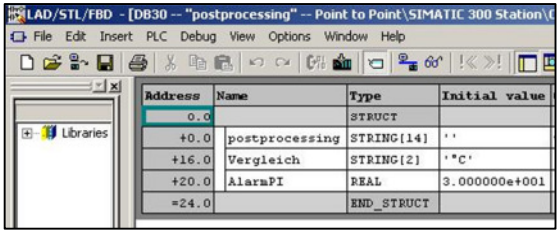


The data block DB20 saves the incoming data. The S7 system needs the information how much space the internal CPU should reserve for these data.
This information can be entered as a “String” value with a maximum account of 14 Bytes in the “Type” column.
The 16 bytes as shown in the example are the result of the 14 bytes incoming data and the two information bytes for every string (Byte 0 = maximal length, Byte 1 = real length of the string).

This data string isn’t only made up of the measured object temperature. The following example shows that this string only contains character bytes.

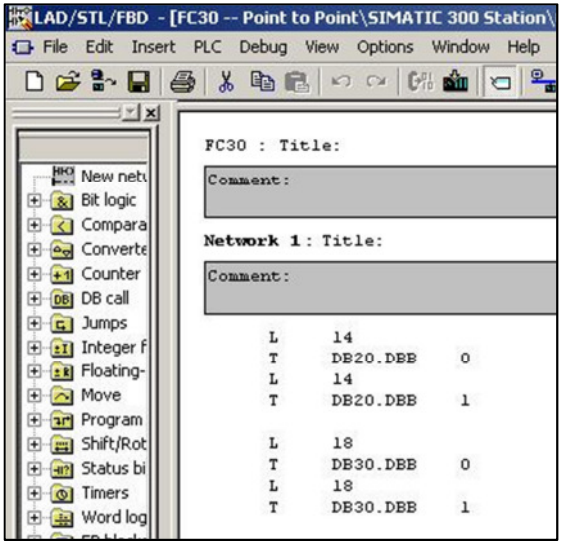
The optris Pi answers of the serial command **?T(x)** with **!T(x)=27.7°C**.

To get the temperature value out of this string, a post processing is required.
The solution of this problem is to delete the first 6 characters (**!T(x)=**), the last two characters (**°C**) and the dot (**.**) between the values.
The result of this example process is only the **277** as three character bytes.
A “String to Integer” data block allows to convert this **277** to a normal integer value.



To save the intermediate data it is helpful to enter a new data block DB30 in the program.

The “postprocessing” string is to save the processed data.
The “relation_data” string is used to find the (**°C**) character bytes.
The “AlarmPI” REAL value defines the alarm threshold of the PI temperature measurement.

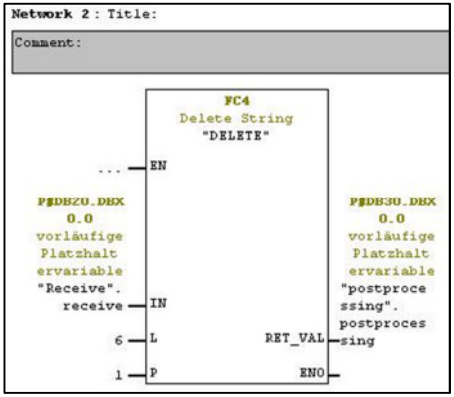


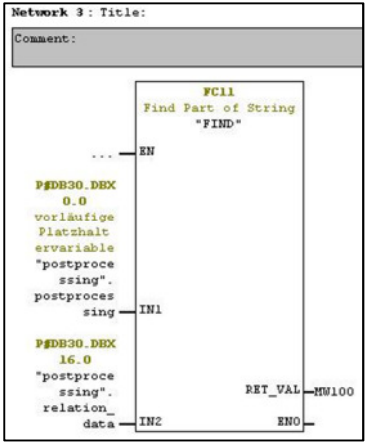
The post processing will be implemented in a new function block FC30.

The two entered data blocks DB20 and DB30 must be initialized before the first usage.

The maximum length of the strings inside our data blocks is 14 bytes (DB20) and 18 bytes (DB30).

These both values should be loaded in the system and transferred to the two first bytes of DB20 and DB30.
(Byte 0 = maximal length, Byte 1 = real length of the string)





The second network of the FC30 includes a “DELETE” function block FC4.
This block deletes the first 6 bytes of the incoming string and saves the post processed string in a new string.

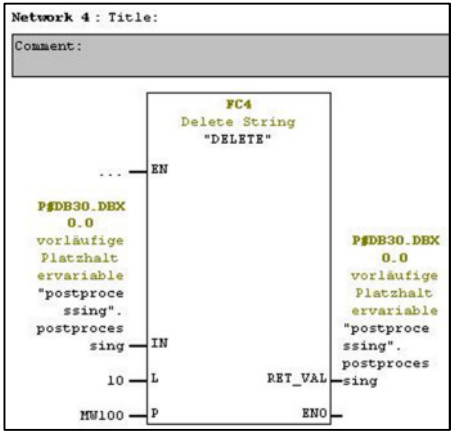
IN:	STRING variable to be deleted in
L:	Number of characters to be deleted
P:	Position of first character to be deleted
RET_VAL:	Result string

IN:	**!T(x)=27.7℃**
RET_VAL:	**27.7℃**

The next step is to find the position of the (**℃**) character bytes.
The function block FC11 is a “FIND” data block which can be used to find specific data string inside a string.

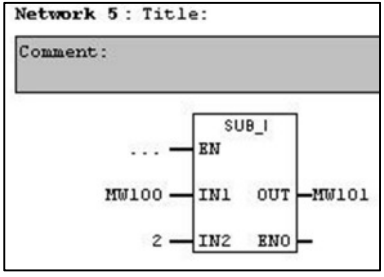
IN1:	STRING variable to be searched in
IN2:	STRING variable to be found
RET_VAL:	Position of the string found

IN1:	**27.7℃**
IN2:	**℃**
RET_VAL:	**5**



The next “DELETE” function block delete all character from the position which was found of the “FIND” function block.

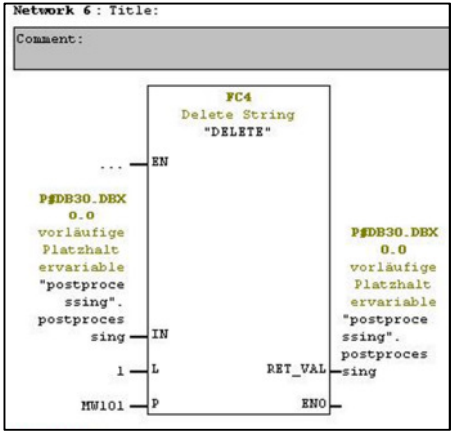
IN: **27.7℃**
RET_VAL: **27.7**



A helpful information is, that the dot (**.**) character of every incoming string is 2 bytes in front of the (**℃**) character.

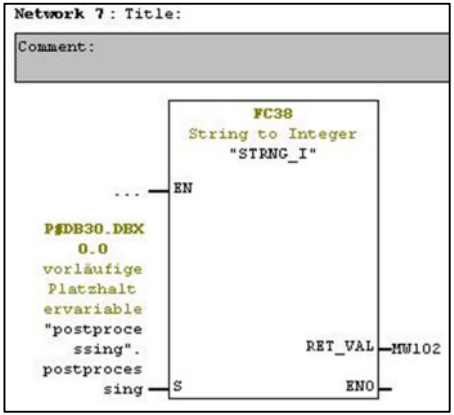
So the position of this character is two less than the position which was found from the “FIND” function block.

The “SUBTRACT” function only subtract the position in MW100 with 2 and save this value in a new MW101.



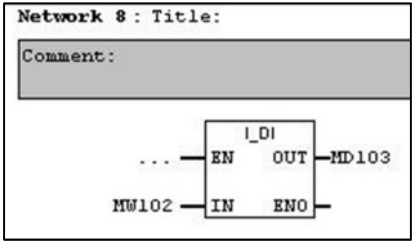
The dot (**.**) character can be deleted because of this new information about the dot (**.**) position saved in MW101.

IN: **27.7**
RET_VAL: **277**



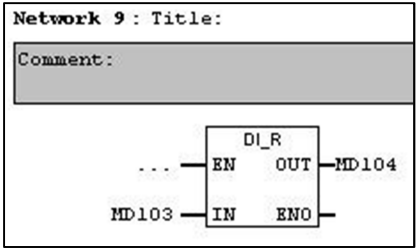
The “String to Integer” function block allows a transformation of a string into an integer value.
The only requirement is that the string is not equal to zero and not greater than 6 character bytes.
The first character of this string can be a sign (+ or -) or a number.

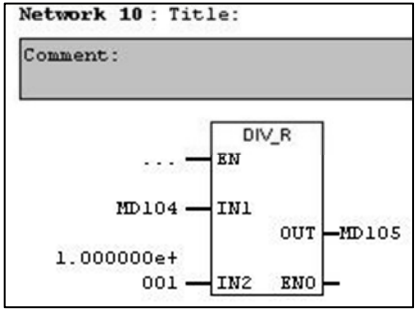
S: Input string
RET_VAL: Result



The next networks convert this integer temperature value into a real value.
The reason for these steps is a comparison between the temperature value and an alarm value which was entered in DB30.

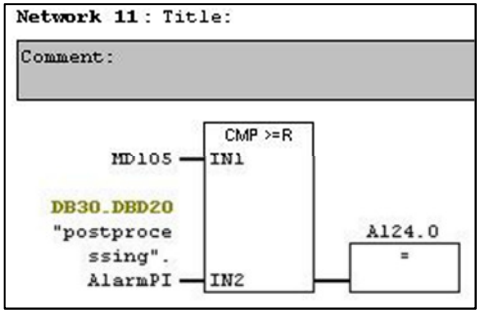
IN: Value to be converted
OUT: Result





This calculated real value doesn't show the right temperature at the moment.
The missing decimal place can be recovered with a division of this integer value by 10.

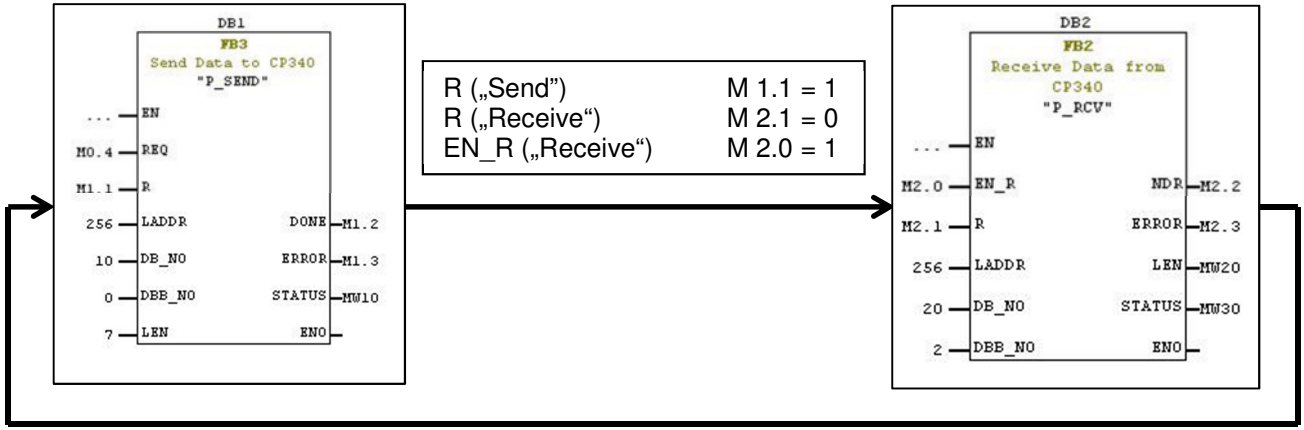
IN: **277**
RET_VAL: **27.7**



Network 11 shows a comparison between the saved calculated real value of the temperature and the saved “AlarmPI” value of DB30.

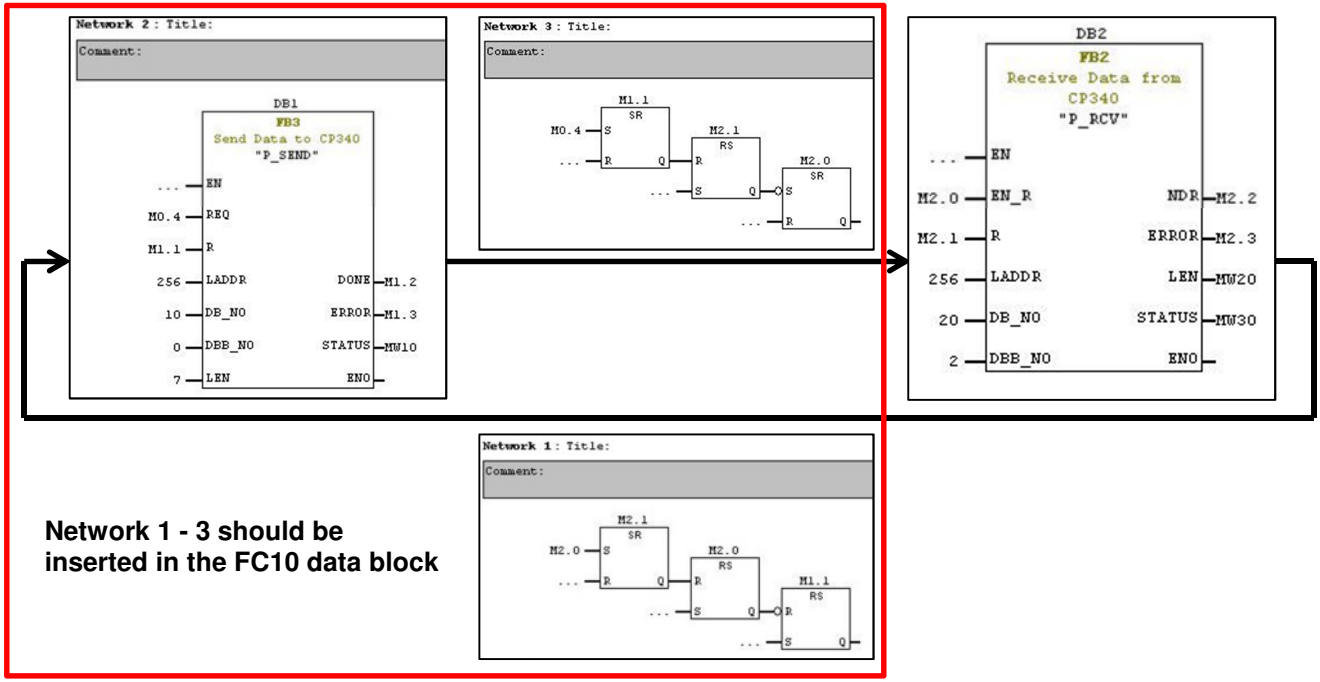
If the incoming temperature is higher than the entered “AlarmPI” value, the analogue output A124.0 is switched on.

This output signal can control LED’s, motors or other electronic devices to react of an incoming alarm.

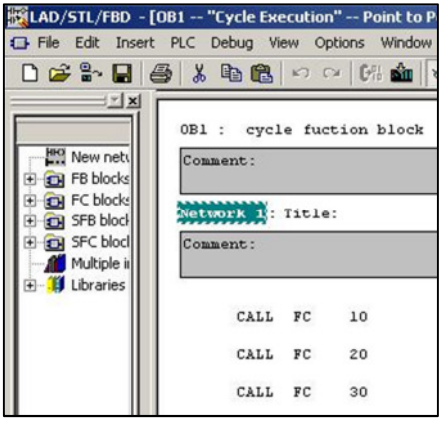


R („Receive")	M 2.1 = 1
EN_R („Receive")	M 2.0 = 0
R („Send")	M 1.1 = 0

The intention is to get a cycle measurement of the object temperature. These two blocks („Send" and „Receive") are normally used to send one command and receive one answer.
To achieve a cycle request for the temperature it is important to set and reset the marker bytes for the enable and abort requests.
The diagram shows the set and reset processes which are needed to implement a cycle process between the sending and receive data block.



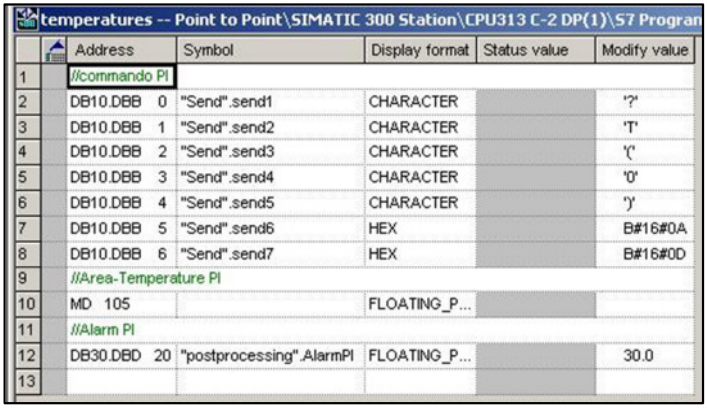
These setting processes can be included with „Flip Flop" data blocks of the Siemens Step 7 program. Network 1 - 3 describe the functional configuration of the FC10 data block and the internal shifts of the enable and abort requests.



The cycle function block OB1 defines the process structure of every single cycle.

The “SEND” (FC10), “RECEIVE” (FC20) and “POST PROCESSING” (FC30) block should be integrated in this block.

The integrated variable table (“temperatures”) can be used to check the calculated temperature (MD105), to set a new alarm value (DB30.DBD) or to change the number of the measured area (DB10.DBB 3).



	Address	Symbol	Display format	Status value	Modify value
1	//commando PI				
2	DB10.DBB 0	"Send".send1	CHARACTER		'?'
3	DB10.DBB 1	"Send".send2	CHARACTER		'T'
4	DB10.DBB 2	"Send".send3	CHARACTER		'('
5	DB10.DBB 3	"Send".send4	CHARACTER		'0'
6	DB10.DBB 4	"Send".send5	CHARACTER		')'
7	DB10.DBB 5	"Send".send6	HEX		B#16#0A
8	DB10.DBB 6	"Send".send7	HEX		B#16#0D
9	//Area-Temperature PI				
10	MD 105		FLOATING_P...		
11	//Alarm PI				
12	DB30.DBD 20	"postprocessing".AlarmPI	FLOATING_P...		30.0
13					