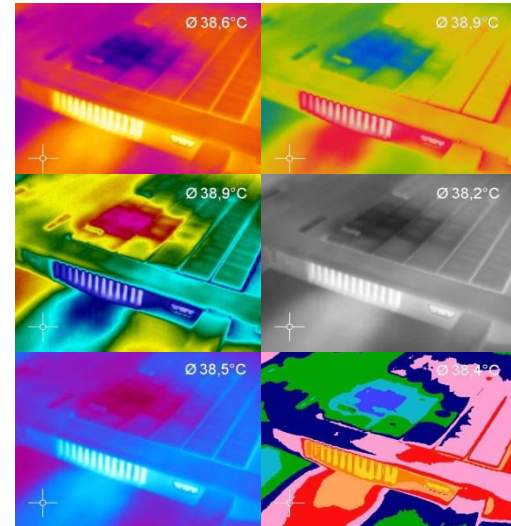


optris® PI160, PI2xx, PI4xx, PI640, PI1M Simatic S7 communication



Operators Manual



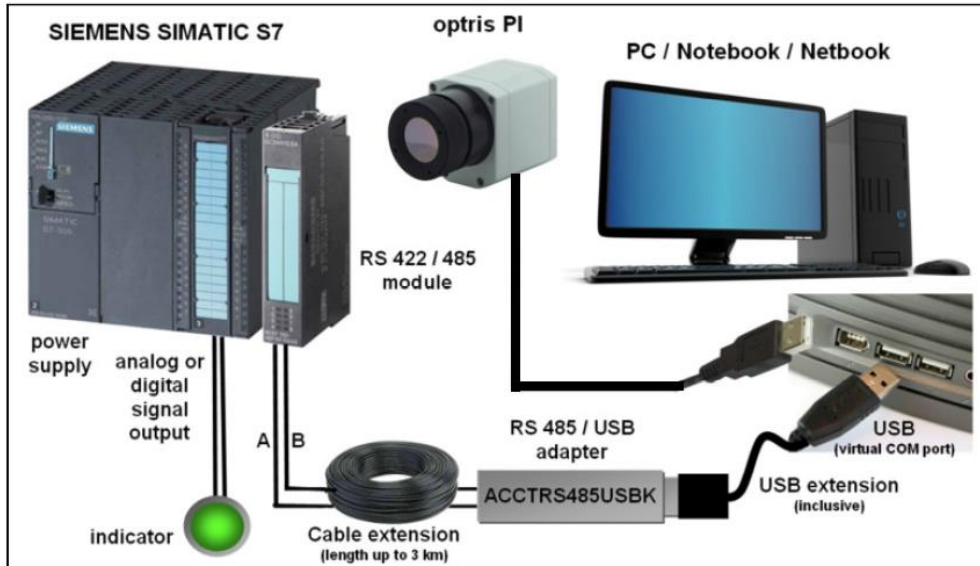
Table of contents

1) Hardware specification	2
2) Create a new project	7
3) Hardware Settings	9
4) Software programming	14
5) Download program to SPS	37

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 1 - Optris PI infrared camera connection to a SPS master system (SIEMENS S7)

Example of the hardware configuration you will need for a RS485 connection between the Optris PI camera and the SIEMENS S7 SPS which includes a RS485 module and also a PROFIBUS module (you can connect a Optris CT/ CTLaser sensor with the PROFIBUS module).

In this manual only the connection with a Optris PI camera is described.

You will need:

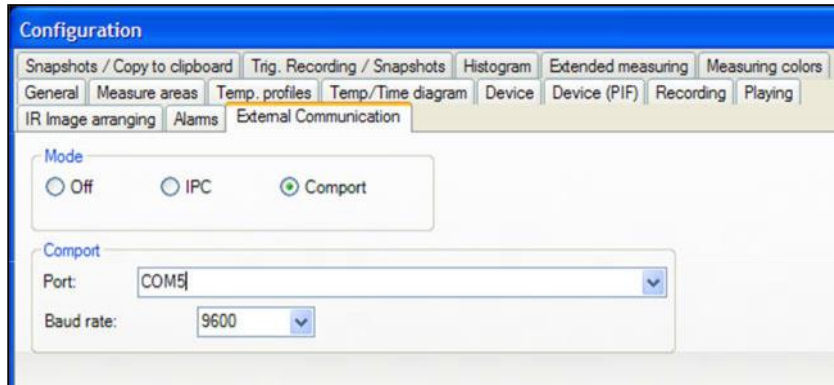
- Optris PI160 / Optris PI450 / Optris PI640 / Optris PI1M or any other Optris PI camera
- RS485 kit (ACCTRS485USBK)
- SIMATIC S7-300, CPU 313C-2 DP
(Processor with a PROFIBUS module)
- SIMATIC S7-300, front module with digital and analog outputs
(To switch actors, for example signal lights)
- SIMATIC S7-300, CP 340
(Communication processor with RS422/485 interface)
- Software STEP 7, in this tutorial V5.5
(To configure the SPS regarding the RS485 module)
- SIMATIC S7 PC Adapter USB
(To connect the SIMATIC S7-300 to your PC)



All implemented commands for the Optris PI are described in the „Serial Communication Description.pdf“, which is included in the Optris PI Connect Software folder.

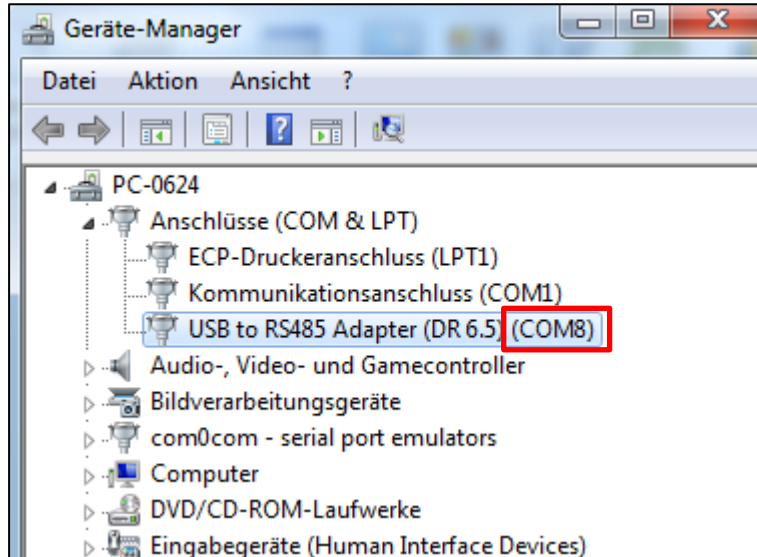
To connect the software PI Connect with the RS485 kit is very easy.

Activate in the software PI Connect at the menu TOOLS – CONFIGURATION - EXTERNAL COMMUNICATION the mode “Comport” and select the Comport regarding your RS485 connection.



Picture 2 - Activate comport communication

You can look up on which Comport your RS485 Kit is assigned in your Windows Device Manager.



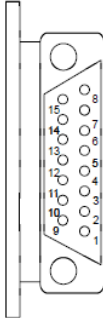
Picture 3 – Assigned COM-Port of your RS485 Kit

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. Picture 4 shows you the PIN allocation of the Siemens SPS.

Pin Allocation

The table below shows the pin allocation for the 15-pin sub D females connector in the front panel of the CP 340-RS 422/485.

Table B-3 Pin Allocation for the 15-Pin Female Connector of the Integrated Interface of the CP 340-RS 422/485

Female Connector on CP340-RS422/485*	Pin	Designation	Input/Output	Meaning
	1	–	–	–
	2	T (A)	Output	Transmitted data (four-wire mode)
	3	–	–	–
	4	R (A)/T (A)	Input Input/Output	Received data (four-wire mode) Received/transmitted data (two-wire mode)
	5	–	–	–
	6	–	–	–
	7	–	–	–
	8	GND	–	Functional ground (isolated)
	9	T (B)	Output	Transmitted data (four-wire mode)
	10	–	–	–
	11	R (B)/T (B)	Input Input/Output	Received data (four-wire mode) Received/transmitted data (two-wire mode)
	12	–	–	–
	13	–	–	–
	14	–	–	–
	15	–	–	–

* View from the front

Picture 4 - PIN Allocation from Siemens CP340 manual (page B-17 / 187)

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 describes the functionality of the program.

Click “Next” to create the project step by step.



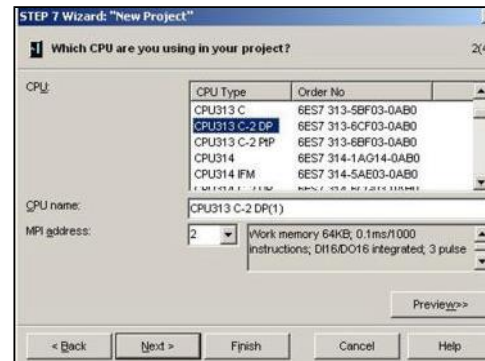
Picture 5 - Create a new Project with the Project Wizard

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.

In this tutorial the **CPU313 C-2 DP** is used.

Choose your CPU and click “Next”.



Picture 6 - Choose your CPU

2. Create a new project

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:

STL: Statement List

LAD: Ladder Logic

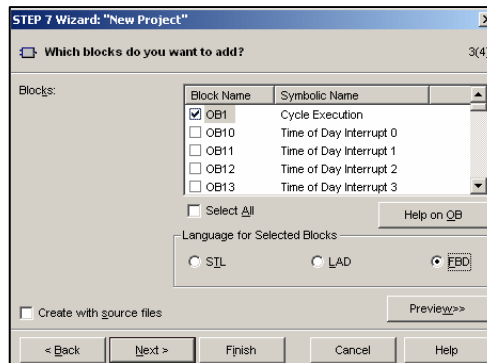
FBD: Function Block diagram

The following project is explained with **FBD** language. Select everything regarding picture 8 and click “Next”.

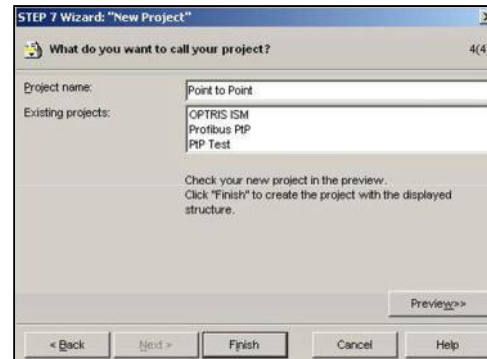
The last picture of the wizard program shows the existing projects and permits to enter the new project name. In this tutorial we will use “Point to Point” as project name.

After you entered a project name you can click “Finish”.

Next you will need to configure your hardware.

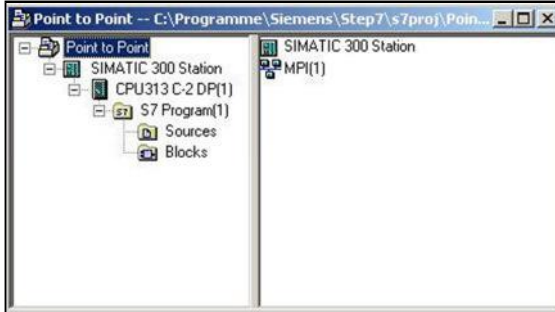


Picture 7 - Select blocks and language



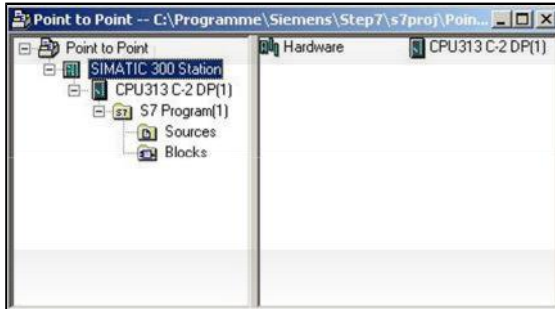
Picture 8 - Name your Project

3) Hardware Settings



Picture 9 - Project Window

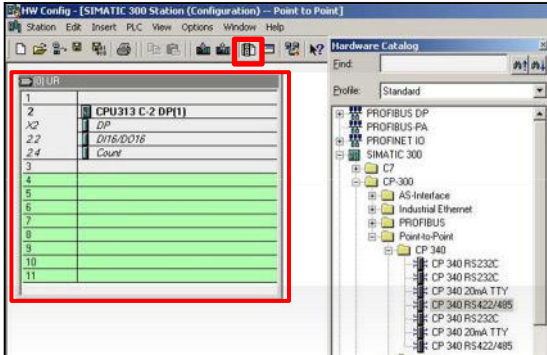
The process tree of this new “Point to Point” S7 program includes the SIMATIC station, the CPU and the SPS program blocks.



Picture 10 - Hardware of your Project

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

To enter the hardware configuration double click on “Hardware”.

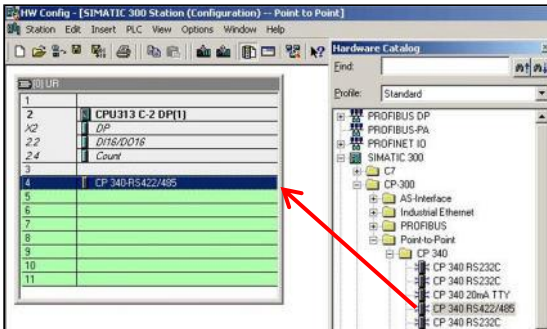


Picture 11 – Add new modules

The hardware overview window (red frame) shows the reserved slots for the CPU (Profibus DP, Digital Input / Digital Output **DI16/DO16** and Counter **Count**).

You need to add a “CP340 RS422/485” module to your Hardware from the “Hardware Catalog”. If you don’t see the catalog yet, open it with the “Catalog” Button (red marker at the top of the window).

Navigate to:
SIMATIC 300 → CP-300 → Point-to-Point → CP340.



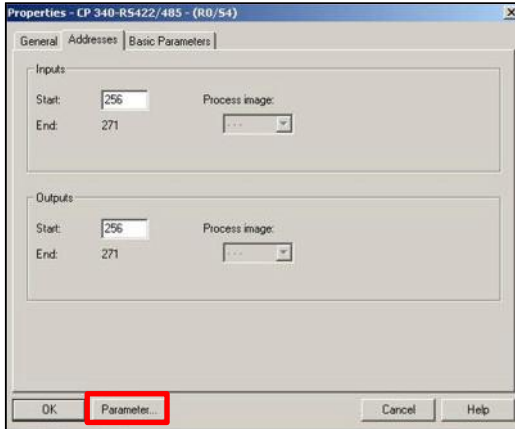
Picture 12 - New communication module added

The CP340 module has three different types of communication protocols (RS232, 20mA TTY, RS422/485).

Once you click on “CP340 RS422/485” some of the Slots of your Hardware on the left are marked green. The green slots indicate the open positions for extra Modules like the “CP340 RS422/485” module you need to add.

For a communication with the PI via USB-RS485 converter (ACCTRS485USBK) the “CP340 RS422/485” module must be dragged and released (Drag & Drop) on a free slot. Here Slot 4 is the first free Slot.

3. Hardware Settings

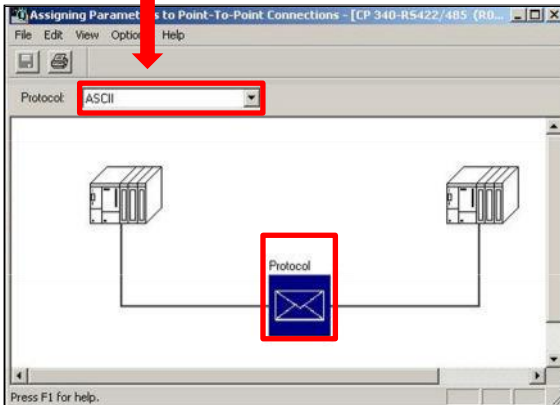


After the “CP340 RS422/485” module is added, double click on it to open its “Properties”.

You need to change some protocol parameters so next click on “Parameter”.



The “Inputs” and the “Outputs” of the CP340 specify the internal address range of the communication data. You may need this Information later in the software programming section.

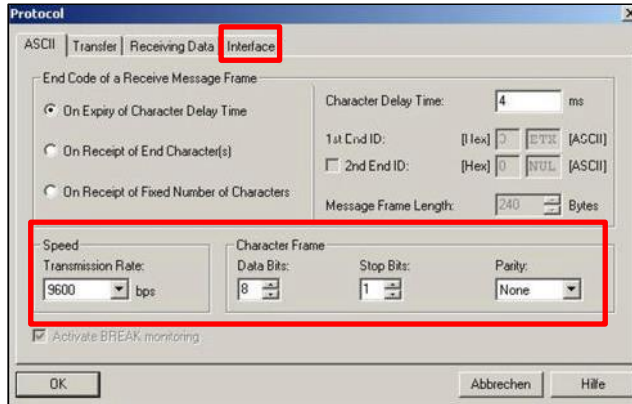


The Optris PI communication operates with an ASCII protocol structure.

In the new opened window change the “Protocol” on the left upper side to “ASCII”.

Next double click on the letter symbol to open the Protocol settings.

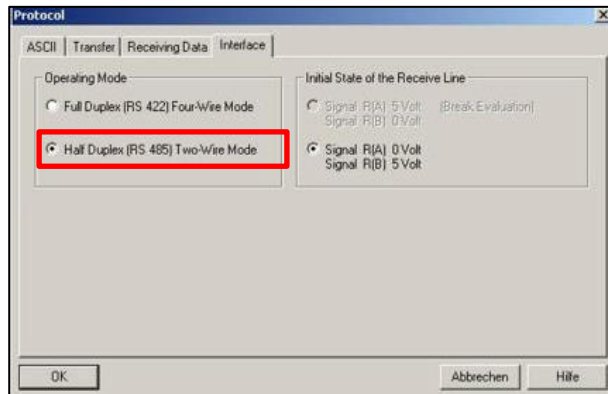
Picture 13 - Properties of the CP340 module



For a correct communication with the PI it is necessary to set:

- Transmission Rate to 9600bps,
- Data Bits to 8,
- Stop Bits to 1 and
- Parity None.

Next click on “Interface”.

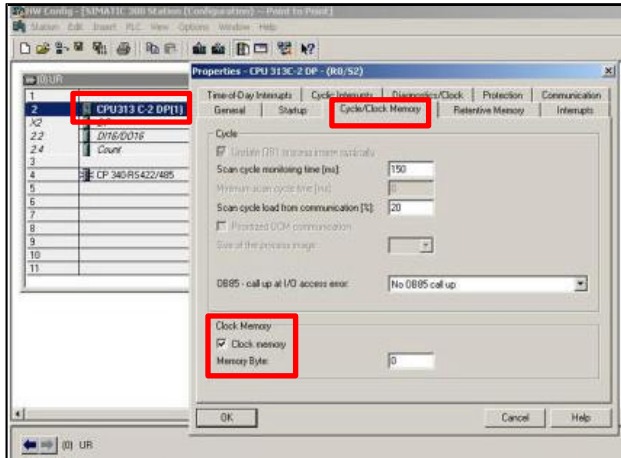


The “Operating Mode” sets the Full Duplex (Four Wire) or the Half Duplex (Two Wire) mode.

The communication with the Optris PI via the ACCTRS485USBK Kit works in a **Half Duplex (RS485)** mode.

After setting the Operating Mode to Half-Duplex you can close the Protocol and Properties windows of the CP340 module (save all changes).

Picture 14 - Protocol properties of the CP340 module



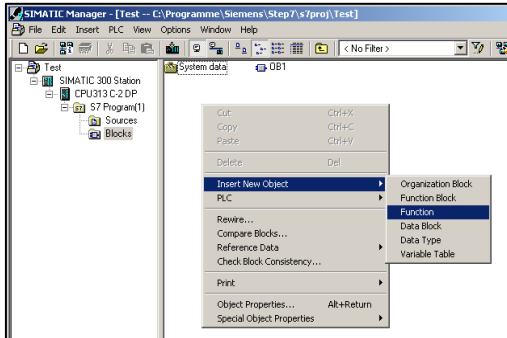
Picture 15 - Properties of the CPU313 C-2 DP

Next double click on your CPU (here CPU313 C-2 DP (1)) to open its “Properties”.

Go to Cycle/Clock Memory and set **Clock Memory** to “ON”. You can leave the Memory Byte at “0”.

Now close the “Properties” window and the “HW Config” window to get back to the S7 main overview window (save the changes).

4) Software programming

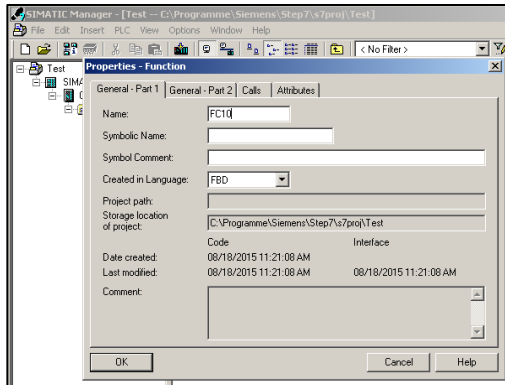


Picture 16 - Create a new Function Block

At the beginning of the programming process it is necessary to enter two different blocks into the system. A Function Block (FC10) and a Data Block (DB10).

The Function Block (FC10) allows the creation of schematic block diagrams.

Go to the Blocks section in the S7 program tree. To create a Function Block right click on a free space on the right side then go to:
Insert New Object → Function



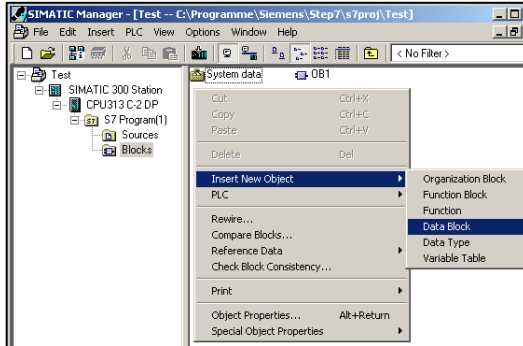
Picture 17 - Function Block Properties

Name: **FC10**

The best way for the beginning is to use the **FBD** (Function Block Diagram) program language.

Enter the name and confirm with “OK”.

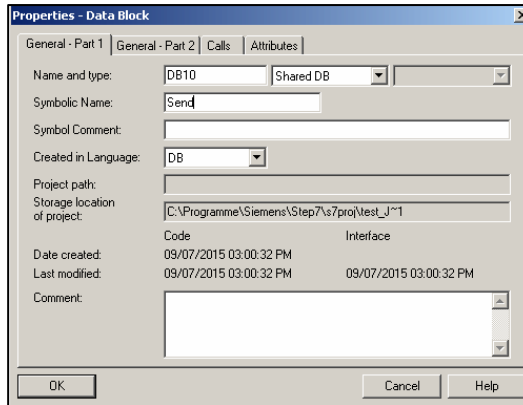
4. Software programming



Picture 18 - Create a new Data Block

Inside the “Data Block” (DB10) it is possible to generate all data structures which are needed for the send, receive and post processing processes.

To create a Function Block right click on a free space:
Insert New Object → Data Block

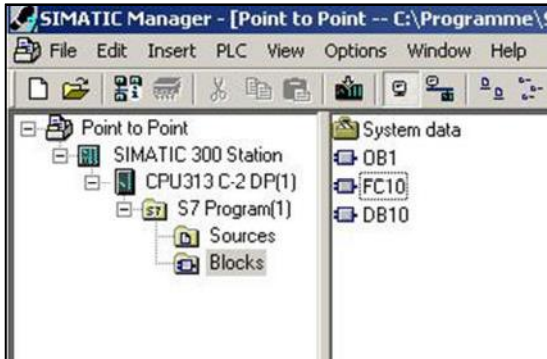


Picture 19 - Data Block Properties

Name: **DB10**
Type: **Shared DB**
Symbolic Name: **Send**

Confirm with “OK”.

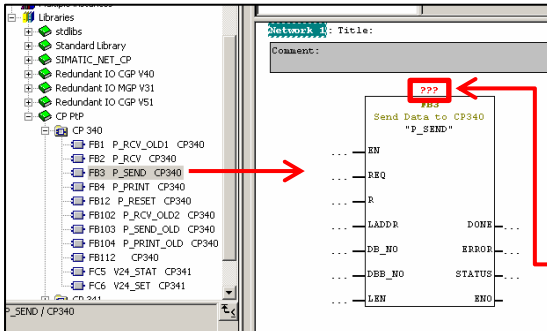
4. Software programming



Picture 20 - Block overview

After the new blocks are added they appear in the “Blocks” overview window.

Next double click on “FC10” to open the function block. Within the block the parameters are saved, which will be send from the SPS to the PI.



Picture 21 – FC10 with Send Data Block FB3

To add the send block FB3 to the created FC10 Block, open in the left program tree:

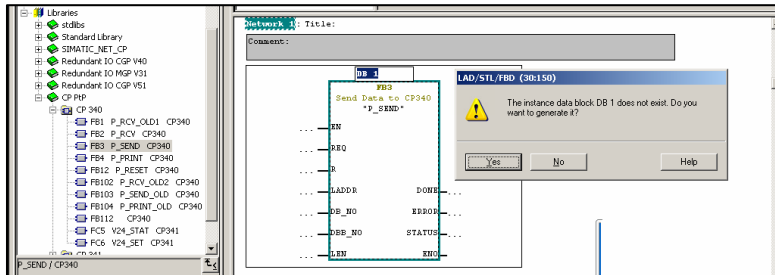
Libraries → CP PtP → CP 340

Then Drag & Drop the Block “FB3 P_SEND CP340” from the program tree on the left to the Network 1 in the FC10 window on the right.

Next enter a Name for the block where the red question mark is located.

Name: **DB 1** (the gap between DB and 1 is important)

4. Software programming

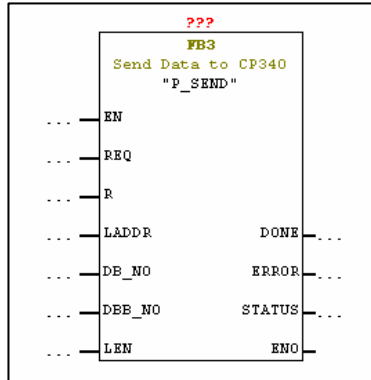


Picture 22 - Create an Instance Data Block

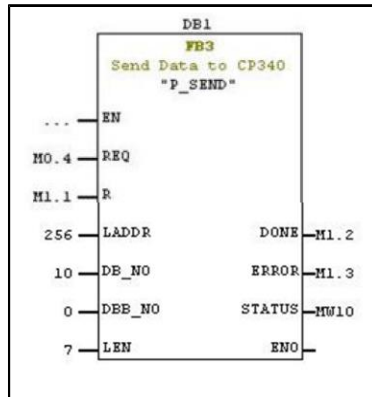
After inserting the name you are asked if you want to generate the data block as an instance block.

Confirm with “YES”.

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 (in the main overview window).
The only setting which must be changed is from “Shared DB” to “Instance DB” (normally this is changed automatically when the block is created).



Picture 23 - Function Block FB3 overview



Picture 24 - Inputs of the FB3 Function Block

The Function Block FB3 needs the following inputs/ outputs:

REQ: Initiates request
 R: Aborts request
 LADDR: The basic address of the CP340 (from picture 14)
 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

The “Input Address” **LADDR** and the “Data Block Number” **DB_NO** 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” is corresponding to the specific “PI Command” which is send.

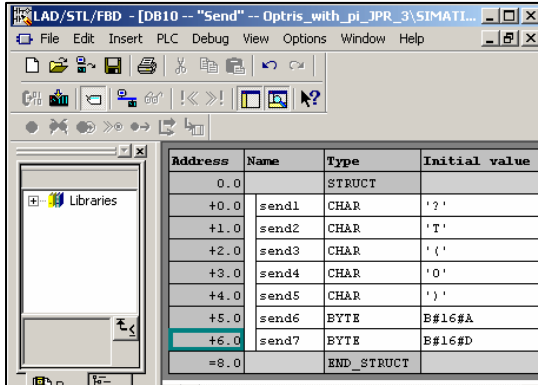
To read the temperature of a measurement area with Index x the command is: **?T(x)** → 5 chars.

Any command must end with CR/LF (0x0D, 0x0A) → 2 chars.

The “DONE”, “ERROR” and “STATUS” values are assign to the M1.2, M1.3 and MW10 parameters. You will create them later.

Enter the Inputs/ Outputs as shown in Picture 25.

Then save and close the FC10 window.



Picture 25 - Input Data for DB10

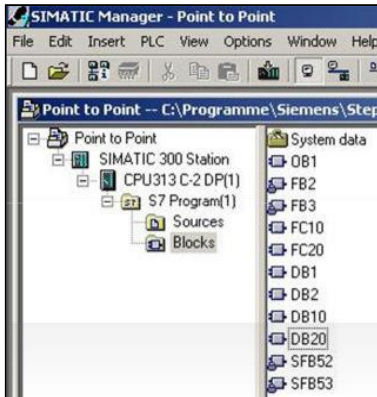
The next step is to enter the “PI Command” inside the DB10. Open DB10 with a double click.

For a better compatibility it is helpful to use a CHAR structure for the request to send and a BYTE structure for the LF/CR command.

Table 1 - Data for DB10

BYTE NUMBER	0	1	2	3	4	5	6
CHARACTER	?	T	(0)	LF	CR
BYTE						B#16#A	B#16#D

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



Picture 26 - Overview with Receive Blocks created

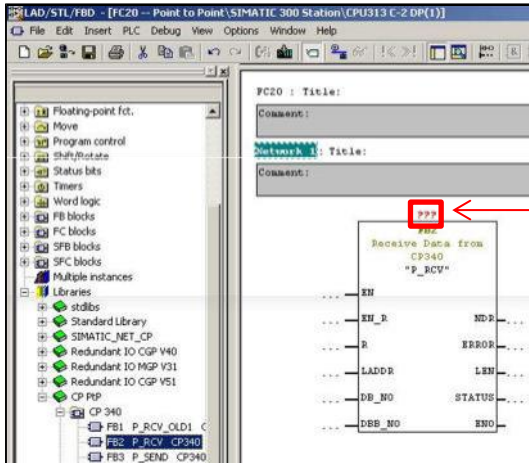
After you entered the data as shown in Picture 26 you can save and close the DB10 window.

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

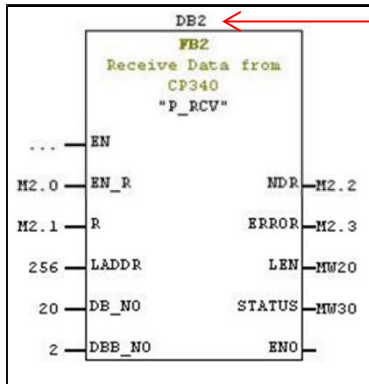
The first steps to implement this process are the same like for the “Send Data” process.

For a better overview it is good to create three blocks named (FC20, DB2 and DB20). The three new Blocks are created like the Blocks before (pictures 17-20).

4. Software programming



Picture 27 - FC20 with receive Block FB2



Picture 28 - Function Block FB2 overview

Create the FC20 Block like the FC10 Block before and add the “FB2 P_RCV CP340” block via Drag and Drop from the left program tree.

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

After you enter “DB 2” as name for the FB2 block you are asked if you want to create the block.

The Booleans for the enable data read and the abort request should be set with marker bytes (M2.0, M2.1).

The basic address LADDR 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).

Enter the Inputs/ Outputs as shown in Picture 29.

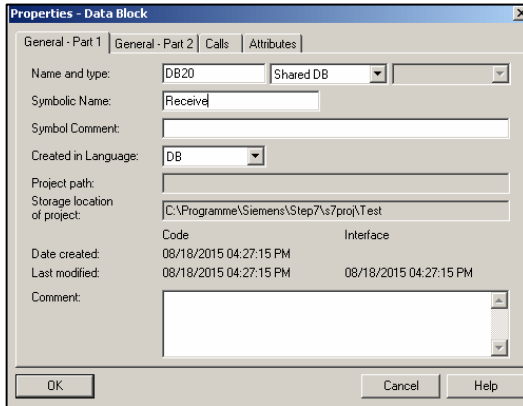
Then save and close the FC20 window.

4. Software programming

Next we create a new Data Block (DB20) the same way we created the Data Block DB10.

Additional to the Name **DB20** we enter a “Symbolic Name” **Receive** to get a better reference within our program later.

Confirm with “OK”.



Picture 29 - Data Block DB20 properties overview

Address	Name	Type	Initial value
0.0		STRUCT	
+0.0	receive	STRING[14]	' '
=16.0		END_STRUCT	

Picture 30 - DB20 Data Input

Next open DB20 with a double click.

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).

Enter the Inputs as shown in Picture 31. Then save and close the DB20 window

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

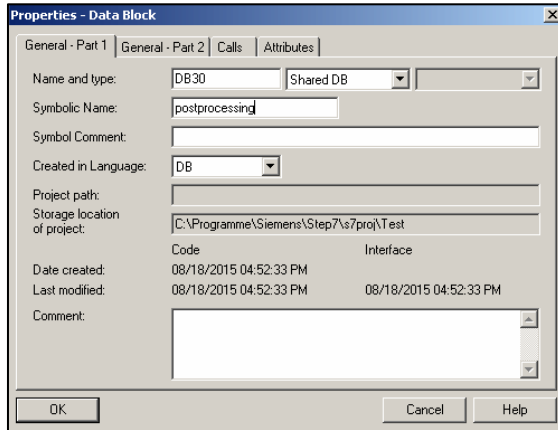
The Optris Pi answers the serial command **?T(0)** with **!T(0)=27.7°C**.

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(0)=**, 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 converting this **277** to a normal integer value.



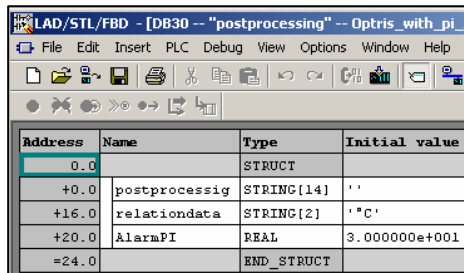
Picture 31 - Data Block DB30 properties overview

To save the intermediate data it is helpful to add another new Data Block, DB30 to the program.

Create a new Data Block **DB30** the same way the Data Blocks DB10 and DB20 were created before.

Also enter a “Symbolic Name”: **postprocessing**.

Confirm with “OK”.



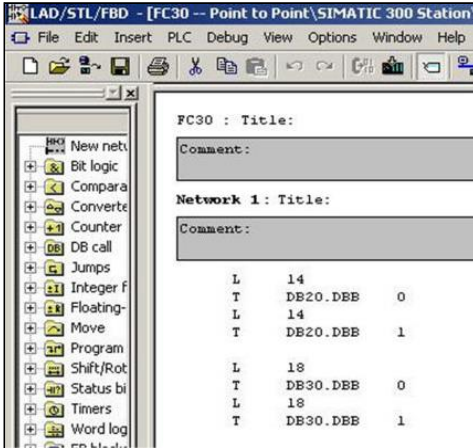
Picture 32 - DB30 Data Input

Next open DB30 with a double click.

**Enter the Inputs as shown in Picture 33.
Then save and close the DB30 window.**

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

4. Software programming



Picture 33 - FC30 Network 1 overview

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

So next create a new function **FC30** the same way the functions FC10 and FC20 were created.

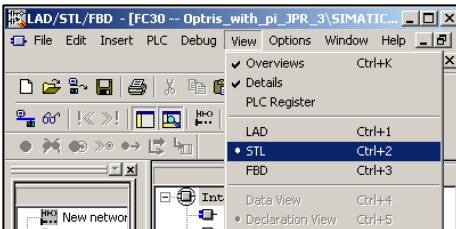
After the FC30 function is created open it with a double click.

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

The maximum length of the strings inside the 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.

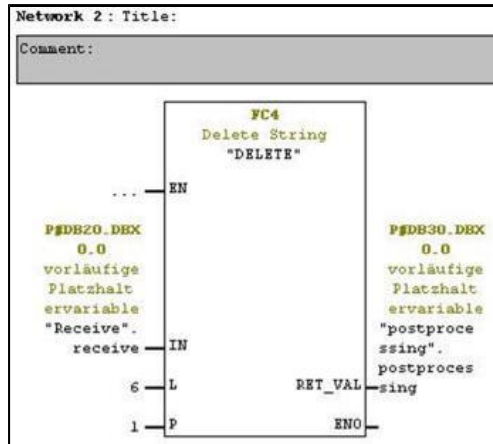
(Byte 0 = maximal length, Byte 1 = real length of the string)



Picture 34 - Change the View

To enter the Data in a text format as shown in picture 34 you need to change the View to STL as shown in picture 35.

After you entered the Data change back the View to FBD.



Picture 35 - Network 2 (FC4) overview

Next create a second network with a right click “Insert Network”.

The second network of the FC30 includes a “DELETE” function **FC4**.

This block deletes the first 6 bytes of the incoming string and saves the post processed string as a new string.

You will find the function “FC4 Delete IEC” in:
Library → Standard Library → IEC Function Blocks

Add the FC4 Block via Drag and Drop to “Network 2” in the FC30 function.

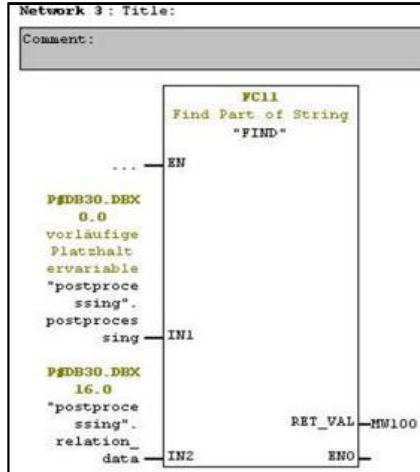
Enter the inputs as shown in picture 36.

The data for IN and RET_VAL can be entered by tipping the first letter, **r** → receive and **p** → postprocessing.

A window opens where you can choose the variable.

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°C
RET_VAL:	27.7°C



Picture 36 - Network 3 (FC11) overview

The next step is to find the position of the “C” character bytes. For that you will need the function FC11. The function FC11 is a “FIND” data block which can be used to find a specific data string inside a string.

Create a new network → **Network 3.**

You find the Block “FC11 FIND IEC” in:
Library → Standard Library → IEC Function Blocks

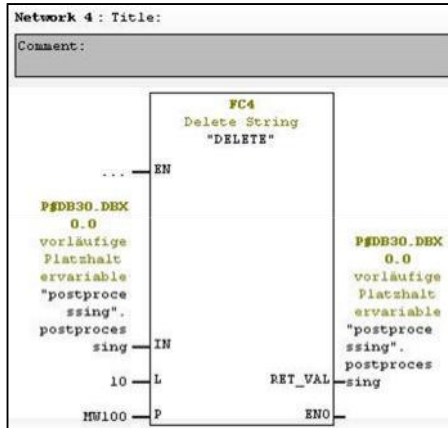
Add the FC11 Block via Drag and Drop to the FC30 Block.

Enter the Inputs as shown in Picture 37.

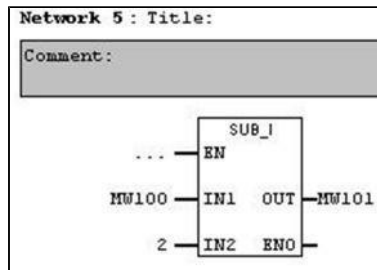
Again you can enter the first letter and choose the wanted variable from the new opened window.

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

IN1:	27.7°C
IN2:	°C
RET_VAL:	5



Picture 37 - Network 4 (FC4) overview



Picture 38 - Network 5 (SUB_I) overview

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

Create a new network → **Network 4**.

Add a new FC4 function (FC4 DELETE IEC) via Drag and Drop to the FC30 function.

From: Library → Standard Library → IEC Function Blocks

Enter the Inputs as shown in picture 38.

IN:	27.7°C
RET_VAL:	27.7

A helpful information is, that the dot . character of every incoming string is 2 bytes in front of the °C character.

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

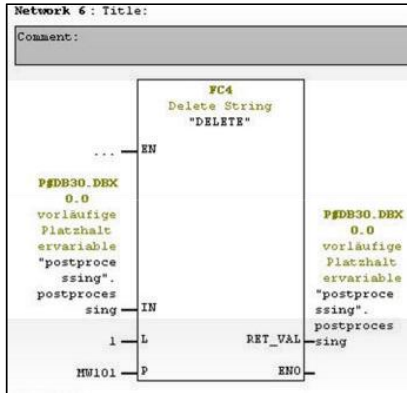
The “SUBTRACT” function only subtract the position in MW100 with 2 and save this value in a new MW101.

Create a new Network → **Network 5**.

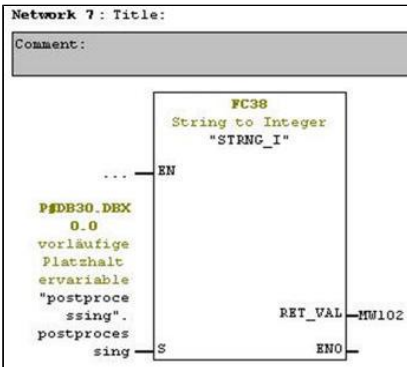
You find the “SUB_I” function under “Integer function”.

Add a “SUB_I” Block via Drag and Drop to the FC30 function.

Enter the Inputs as shown in Picture 39.



Picture 39 - Network 6 (FC4) overview



Picture 40 - Network 7 (FC38) overview

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

Create a new network → **Network 6**.

Add a new FC4 function via Drag and Drop to the FC30 function.
From: Library → Standard Library → IEC Function Blocks

Enter the Inputs as shown in Picture 40.

IN:	27.7
RET_VAL:	277

The “String to Integer” function 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.

Create a new Network → **Network 7**.

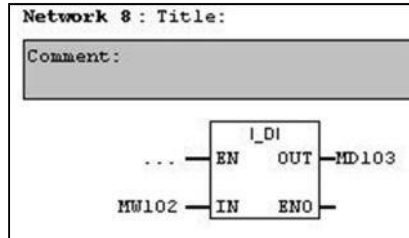
You find the Block “FC38 STRNG_I IEC” in:

Library → Standard Library → IEC Function Blocks

Add a “FC38 STRNG_I IEC” function via Drag and Drop to the FC30 function.

Enter the Inputs as shown in Picture 41.

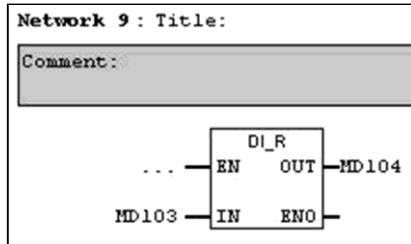
S:	Input string
RET_VAL:	Result integer value



The next networks convert the integer temperature value into a real value.

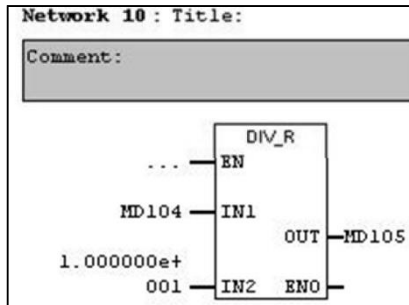
The reason for these steps is a comparison between the temperature value and an alarm value which was entered in DB30. Only real values can have decimal places.

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.



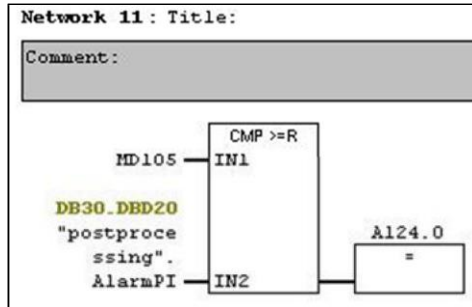
IN:	277
RET_VAL:	27.7

Create three new networks → **Network 8, 9 and 10.**

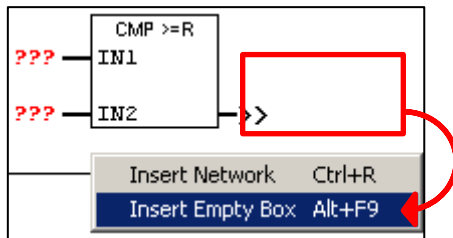
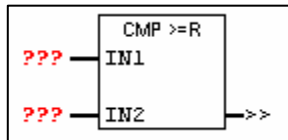
You find the functions "I_DI" and "DI_R" under "Converter" and the "DIV_R" function under "Floating-point fct."

Add the functions to the networks and enter the inputs as shown in picture 42.

Picture 41 - Network 8,9,10
temperature conversion overview



Picture 42 - Network 11 temperature comparison



Picture 43 - Network 11 create output box

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

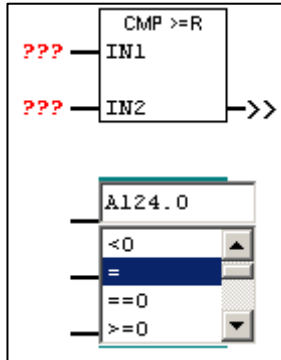
If the incoming temperature is higher than the entered “AlarmPI” value, the analogue output A124.0 is switched on.

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

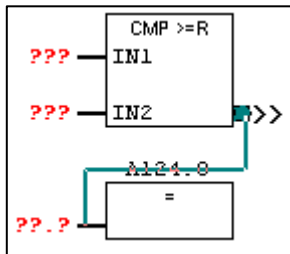
First create a new Network → **Network 11**.

You find the Block “CMP>=R” under “Comparator”. Add a “CMP>=R” function via Drag and Drop to the FC30 function.

Next right click on a free space within Network 11 and select “Insert Empty Box”.



Picture 44 - Network 11
enter output data



Picture 45 - Network 11
connect Blocks

Enter **A124.0** as output and select the **equal sign** from the menu.

A → output in German Mnemonics

If you use the English Mnemonics in Step 7 use “Q” instead of “A”.

124 → Start of the output addresses (can be looked up in the hardware properties of the DI16/DO16 module within the “HW Config” window)

0 → number of the physical output at the module

Connect the “CMP>=R” block with the created box.

To establish a connection click the output on the right of the “CMR>=R” block and hold the left mouse button.

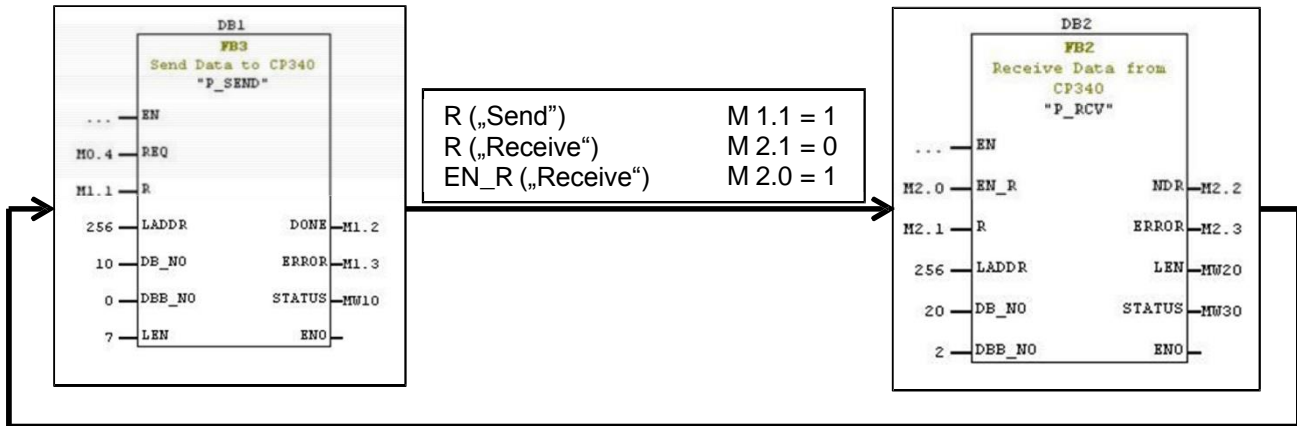
Then drag a connection to the input of the created output box.

The program will print a possible connection when you move the mouse to the right position at the output box while you hold the left mouse button. Release the mouse button to establish a connection.

Enter the inputs as shown in picture 43 and 46.

Then save and close the FC30 window.

4. Software programming



Picture 46 – cycle request overview

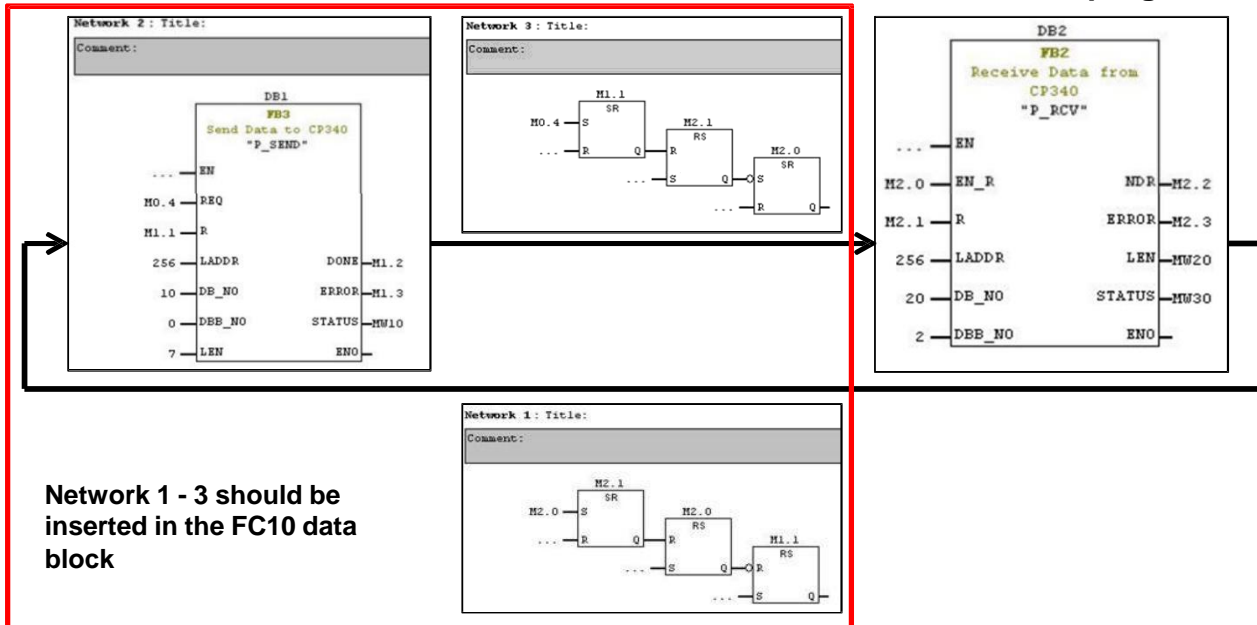
R („Receive“)	M 2.1 = 1
EN_R („Receive“)	M 2.0 = 0
R („Send“)	M 1.1 = 0

The next step 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 process which is needed to implement a cycle process between the send and receive function.

4. Software programming



Picture 47 - cycle request components

This cycle process can be included with „Flip Flop” functions of the Siemens Step 7 program. Network 1 - 3 describe the functional configuration of the FC10 function and the internal shifts of the enable and abort requests.

The DB1 and DB2 blocks were already created at the beginning of this tutorial (pictures 25 and 29 on page 19 and 21).

4. Software programming

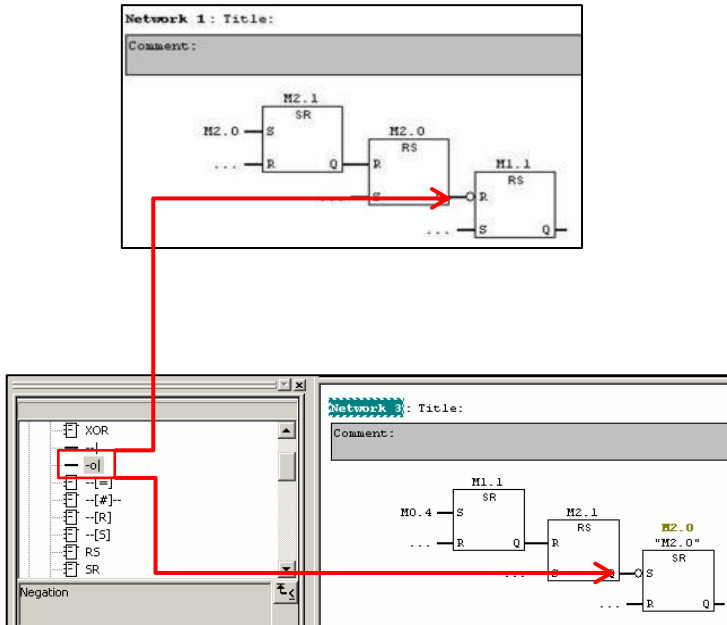
Next open the FC10 block from the main overview window and add two new networks. Then cut the DB1 function and insert it in network 2. Now is one empty network before the network with DB1 and one after it.

Add the “SR” and “RS” blocks from the “Bit logic” folder to network 1 and 3 like shown in picture 49.

Connect the blocks like shown on page 32.

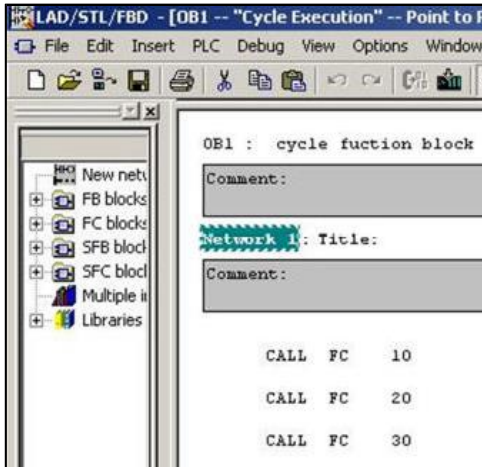
To get the negation between the connected blocks drag and drop the negation from the “Bit logic” folder on the two connections.

**Connect the blocks and enter the input as shown in Picture 49.
Then save and close the FC10 window.**



Picture 48 - Block connection and Negation insertion

4. Software programming



Picture 49 - set up the request cycle

Next it is important to set up the call of every subprogram that was created in every cycle.

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

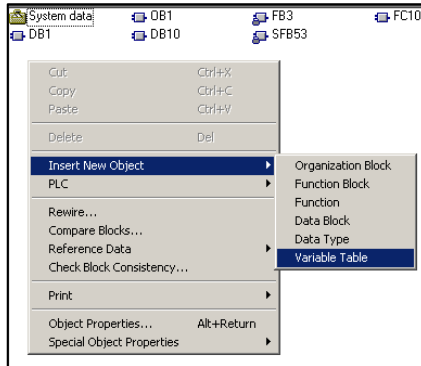
The “SEND” (FC10), “RECEIVE” (FC20) and “POST PROCESSING” (FC30) block should be integrated in this block.

Open the OB1 block from the program overview and add call up lines as shown on picture 50 to the block.

If you can't enter the lines as text change the view to “STL” (as shown in picture 35 on page 25).

Then save and close the FC10 window.

4. Software programming



Picture 50 - create a variable table

Now right click on a free space in the overview window and create a new “Variable Table” under “Insert New Object”.

As symbolic name enter: **temperatures**

After the table is created open it with a double click.

	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		'I'
4	DB10.DBB 2	"Send".send3	CHARACTER		'C'
5	DB10.DBB 3	"Send".send4	CHARACTER		'0'
6	DB10.DBB 4	"Send".send5	CHARACTER		'J'
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					

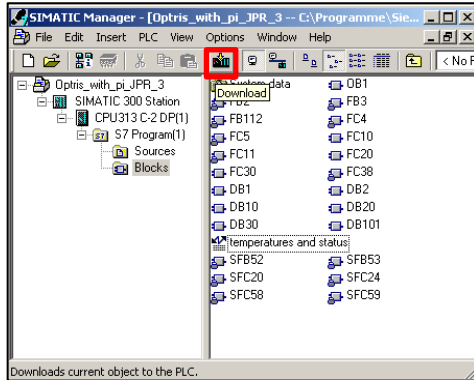
Picture 51 - variable table data

The created 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) without going through all subprograms.

Enter the data as shown in picture 52.

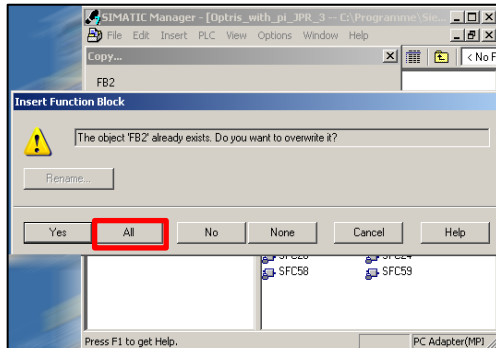
Then save and close the FC30 window.

5) Download program to SPS



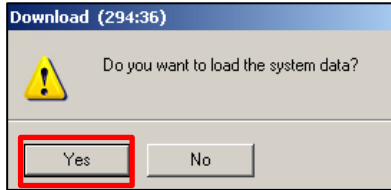
After you connect the SPS to your programming PC via the “SIMATIC S7 PC adapter USB” and start up the SPS you can download your program to the SPS storage. Make sure the SPS is set to “STOP”.

To download your program to the SPS click on the “Download” button on the top of the program overview.



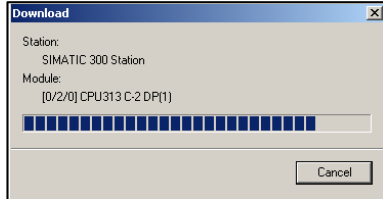
If you are asked to overwrite data on the SPS click “All”.

Picture 52 - download to SPS



Picture 53 - download system download

Also click “Yes” when you are asked to load the system data.



Picture 54 - download process

When the download is finished the SPS is ready. Start the SPS by putting the SPS into the “Run” mode.

Make sure to connect the SPS to the PC on which the Optris PI Software is running and configure the Software to use the right ComPort (Chapter 1).