SIEMENS

SITRAIN
Training for Industry

SIMATIC S7
S7 1200
Teknoloji ve Haberleşme Kursu

TIA-MICRO3

Name: ______________________________

Course from: ____________ to: ____________

Trainer: ______________________________

Training site: ______________________________

This document was produced for training purposes. SIEMENS assumes no responsibility for its contents. The reproduction, transmission, communication or use exploitation of this document or its contents is not permitted without express written consent authority. Offenders will be liable to damages. Non-compliances with this prohibition make the offender inter alia liable for damages.

SITRAIN courses on the internet: www.siemens.com.tr/sitrain

Course folder Version: V13.1 (for STEP7 V13 SP1)

Contents 1

1. Presentation of the Course Contents and Training Devices

1.1. Objectives

At the end of the chapter the participant will ...

... be familiar with the main course contents

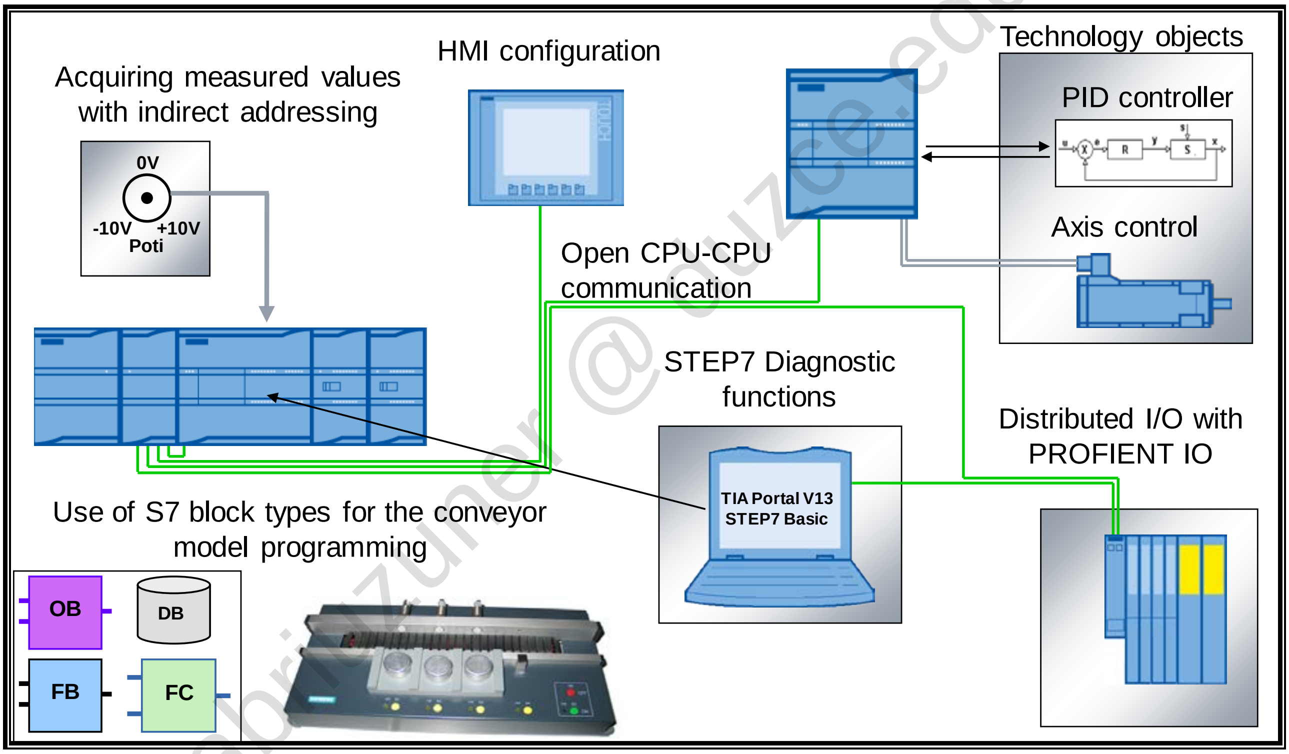
… be familiar with the training devices

… be familiar with the networking of the training devices

Objectives

In this chapter, the main course contents and the training devices are presented.

1.2. Course Contents


Acquiring measured values with indirect addressing
0V
-10V
+10V
Poti
HMI configuration
Technology objects
PID controller
Axis control
Open CPU-CPU communication
STEP7 Diagnostic functions
Distributed I/O with PROFIENT IO
TIA Portal V13 STEP7 Basic
Use of S7 block types for the conveyor model programming
OB
DB
FB
FC

Course Contents

The following topics are dealt with in this course:

- Analog value processing

  Analog value processing is used to convert process values (for example, the measured values (measuring points) of a level sensor) into a "tangible" unit (e.g. m³), in order to then display it on an HMI, for example.

- S7 block types

  In this course, the S7 block types and their use are presented. Differentiation is made between logic (code) and data blocks. Data blocks are structured, user-defined data memory. They can be used for simple data storage or as interface to HMI devices. They can also be used to create complex database structures.

  In the section Logic Blocks (FC, FB, OB), the properties and the fields of application are presented.

- Programming in SCL

  SCL is a further IEC programming language, in addition to LAD and FBD, for S7. Programming in SCL offers decisive advantages in the implementation of complex, mathematical calculations, as well as in the handling of large amounts of data.

- Indirect addressing

  Indirect addressing enables you to dynamically address memory cells within the PLC during program runtime. For example, measured value series can thus be formed by writing each new measured value into a different memory cell. Fundamental mechanisms for indirect addressing are available in LAD and FBD. The complete instruction set is available in SCL.

- Introduction to PROFINET IO

  PROFINET (similar to PROFIBUS) is used to connect distributed I/Os to the CPU. In this course, the fundamental addressing mechanisms and procedures for configuring distributed PROFINET field devices are presented.

- Expanded HMI configuration

  In addition to the basic functions of an HMI device presented in the Basic Course, the alarm message system for the display of discrete and analog alarms as well as the creation of input/output fields and the time-of-day synchronization between HMI device and PLC are dealt with in this course.

- Integrated CPU technology objects

  The S7-1200 offers integrated technology functions for motion control of axes and for PID control loops. The necessary steps for creating a technology object and the commissioning of a PID control loop and stepper motor are part of this course.

- Troubleshooting with STEP7 (TIA Portal)

  As the diagnostic functions of the TIA Portal are crucial for troubleshooting and system analysis, the available online and offline functions for quick and efficient elimination of arising faults are presented in this course.

- Open CPU-CPU Ethernet communication

  For data exchange between controllers, the S7-1200 is equipped with Ethernet communication concepts which are presented in more detail in this course and are also practiced.

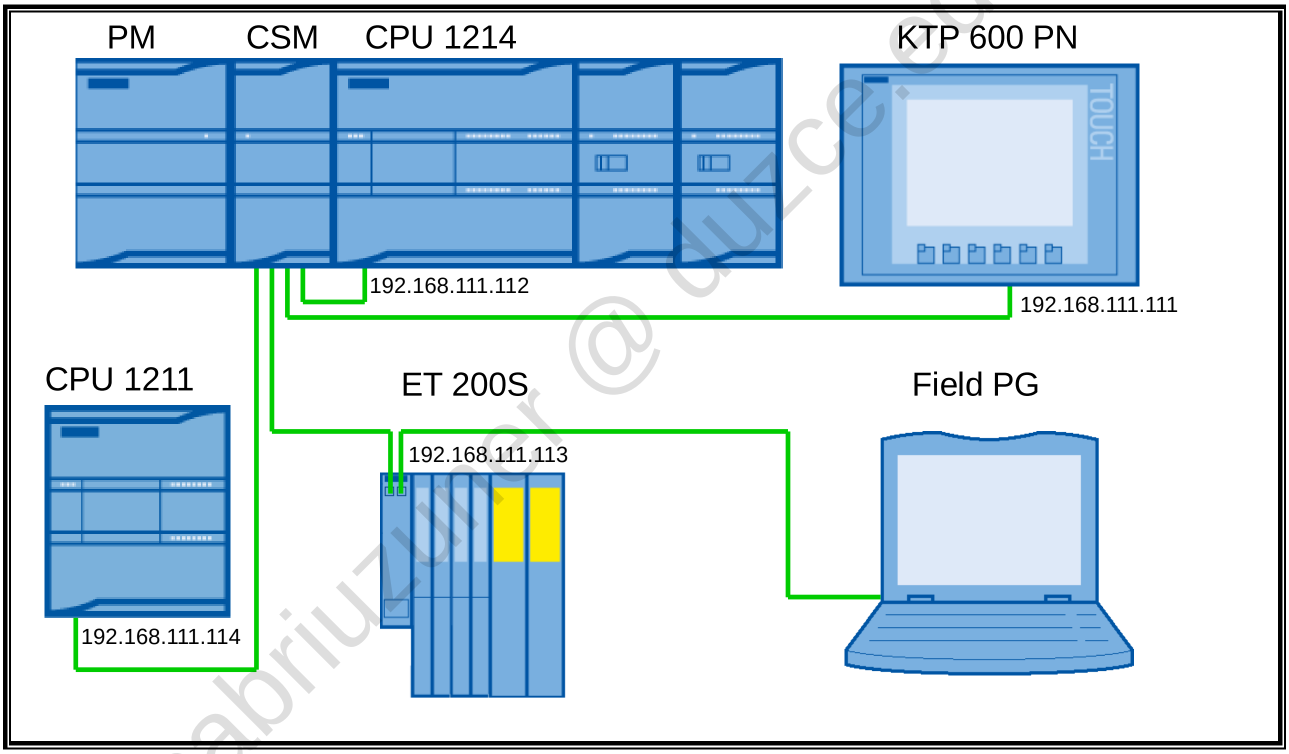
1.3. Training Area with S7-1200

Training Area Setup

The training area for this course contains the following components:

- SIMATIC Field PG
- Training case with S7-1214, touchpanel and simulator
- Training case with S7-1211, stepper motor and control loop
- Training setup ET 200S as distributed I/O
- Conveyor model

1.4. Schematic Diagram Industrial Ethernet/ PROFINET Networking



Networking of the Individual Components

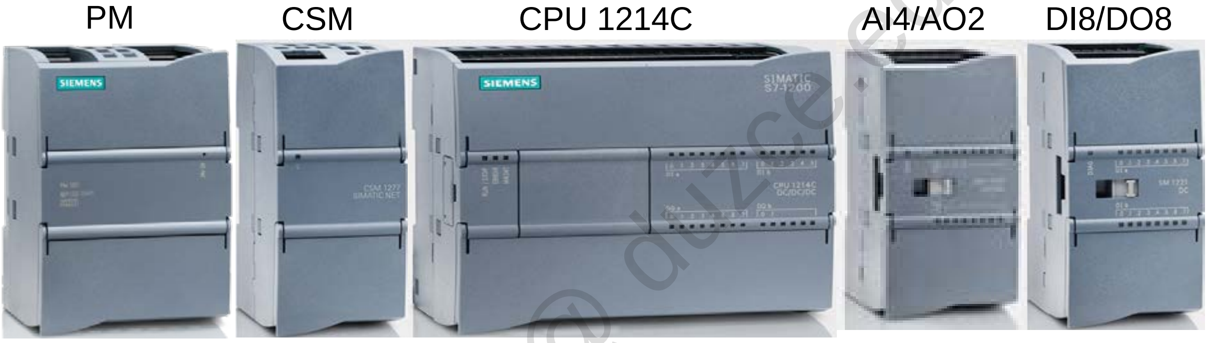
The components of the training area are all networked to one another via (industrial) Ethernet. The RJ45 connection technology is the most widely used and can also be found in the home.

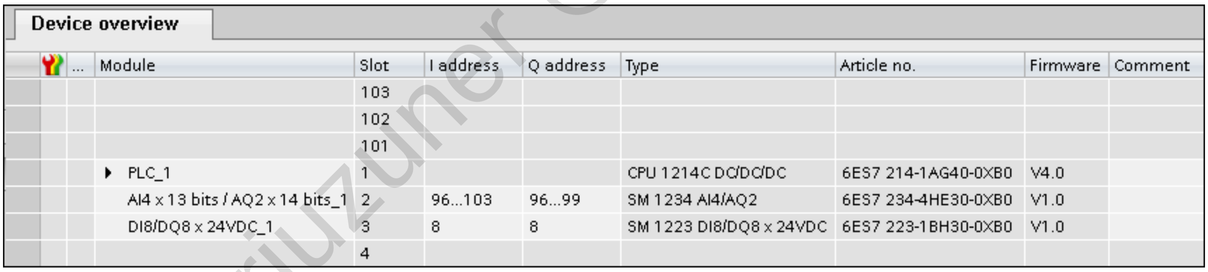
A point-to-point connection always exists between the components. This makes it necessary to use a switch (here in the form of a Compact Switch Module) which is used here as a network distributor.

The (industrial) Ethernet is the basis for the communication between the components. Depending on the type of communication, different protocols are used, for example:

- TCP communication between CPU and HMI and
- PROFINET IO communication between CPU and ET 200S

1.5. Configuration of the S7-1214 Training Device




Device overview

		...	Module	Slot	I address	Q address	Type	Article no.	Firmware	Comment
				103						
				102						
				101						
			▸ PLC_1	1			CPU 1214C DC/DC/DC	6ES7 214-1AG40-0XB0	V4.0	
			AI4 x 13 bits / AQ2 x 14 bits_1	2	96...103	96...99	SM 1234 AI4/AQ2	6ES7 234-4HE30-0XB0	V1.0	
			DI8/DQ8 x 24VDC_1	3	8	8	SM 1223 DI8/DQ8 x 24VDC	6ES7 223-1BH30-0XB0	V1.0	
				4						

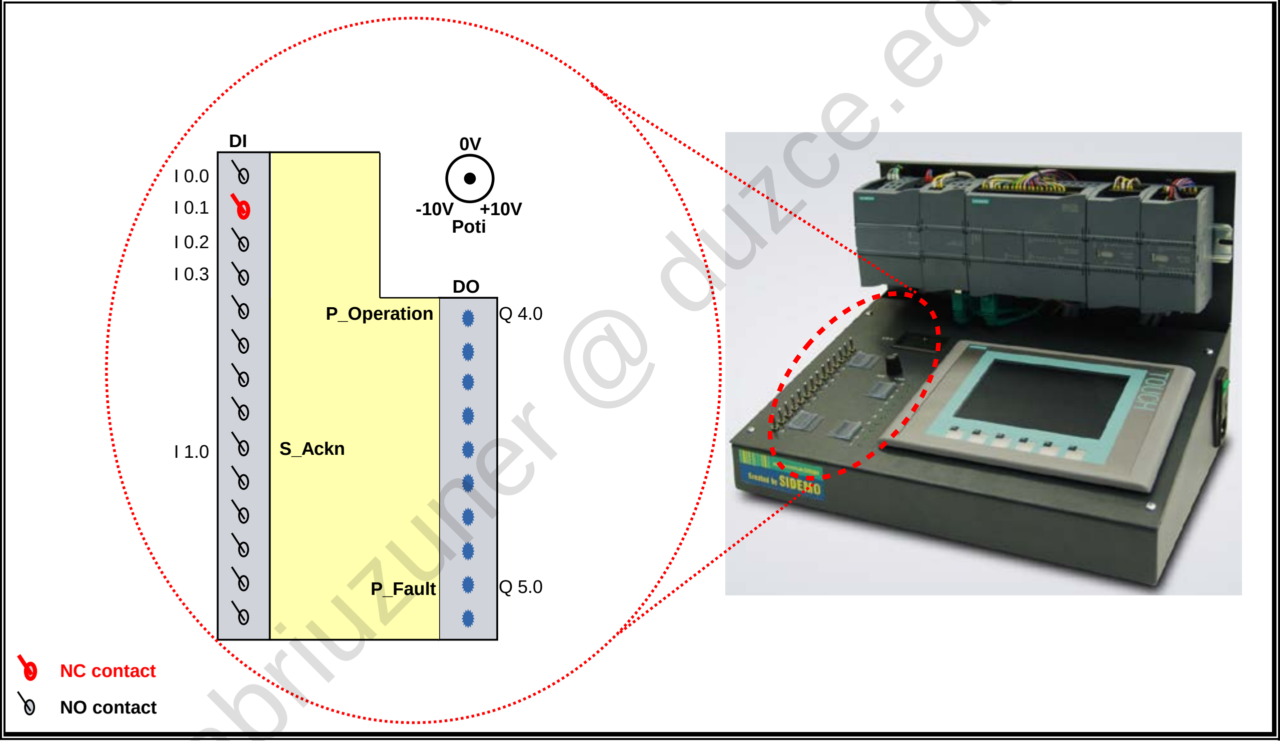
Configuration of the S7-1214 Training Device

The picture shows the central module of the S7-1214 training case. The CPU has two signal modules (SMs) for digital and analog I/Os, as well as a signal board (SB) with an analog output as an expansion. The I/O addresses of the modules shown in the picture are already stored in the start project (→ next chapter) and do not have to be parameterized separately.

Module addresses at a glance:

- CPU 1214
  - DI14 → I 0.0 to I 1.5
  - DO10 → Q4.0 to Q5.1
  - AI2 → IW64, IW66
  - AO1 → QW80
- SM 1234
  - AI4 → IW96, IW98, IW100, IW102
  - AO2 → QW96, QW98
- SM1223
  - DI8 → I 8.0 to I 8.7
  - DO8 → Q8.0 to Q8.7

1.6. The Simulator



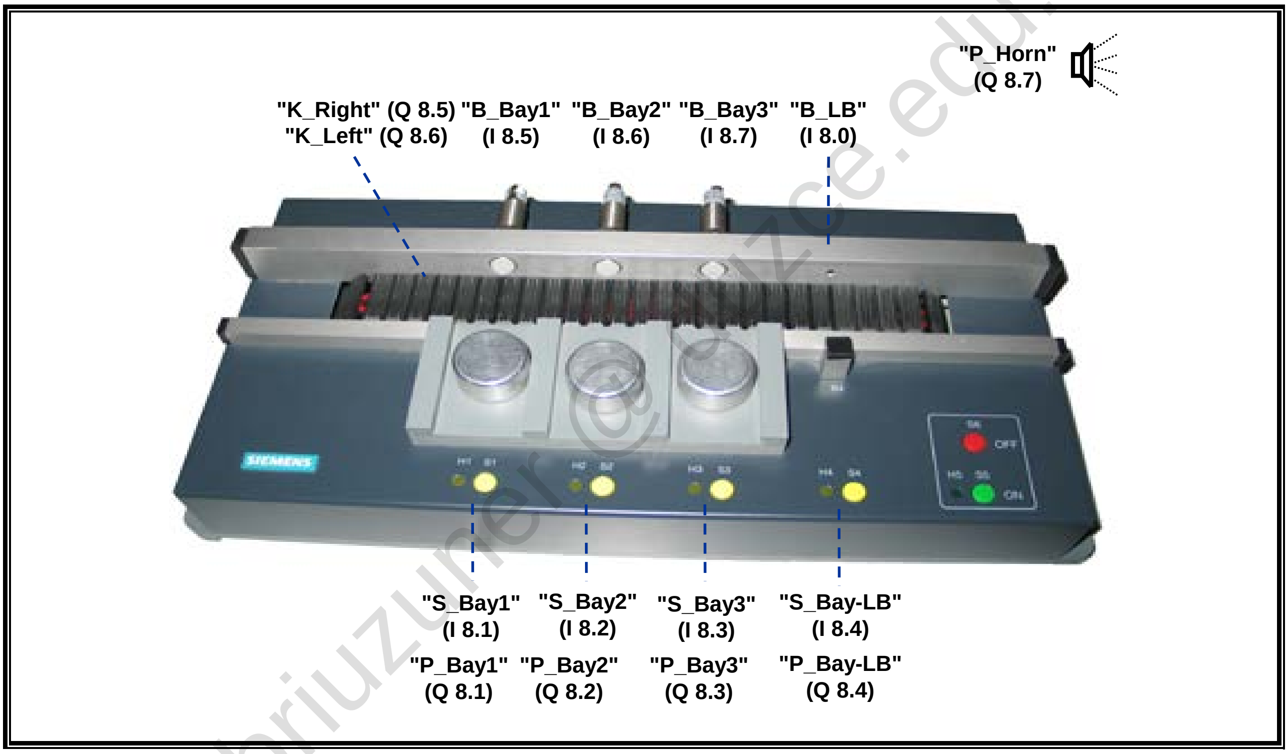
The Simulator

Together with the touchpanel, the simulator is used to operate the system. It consists of the following components:

- 14 switches, whereby the I 0.1 switch is an NC contact
- 10 LEDs
- Rotary potentiometer for setting or simulating analog input signals

The digital signals are connected to the I/Os of the CPU. The analog signal is processed by the SM 1234 analog module.

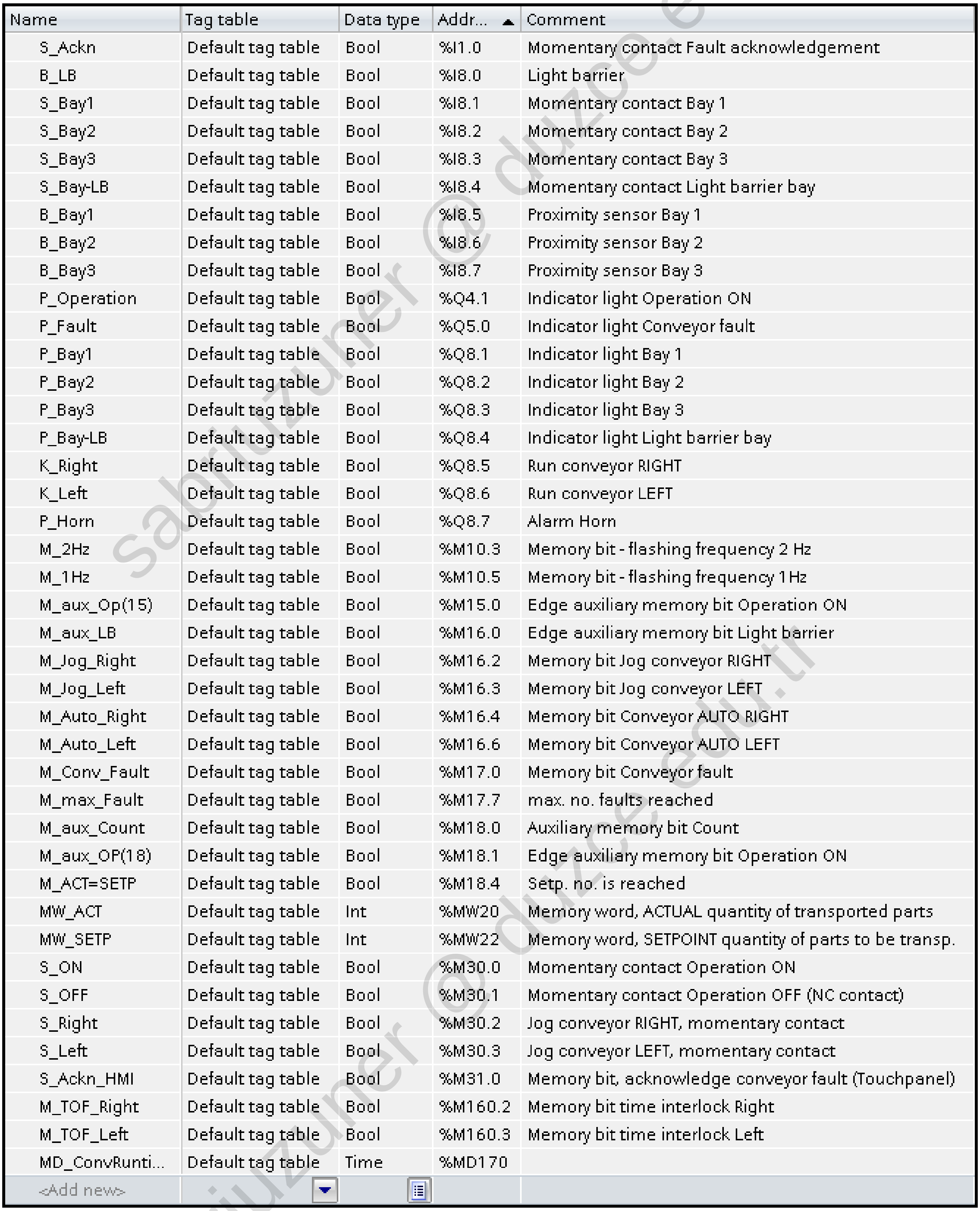
1.7. The Conveyor Model



The Conveyor Model

The picture shows the sensors and actuators of the conveyor model as well as the I/O addresses to which they are wired.

1.8. PLC Tags



Name	Tag table	Data type	Addr... ▲	Comment
S_Ackn	Default tag table	Bool	%I1.0	Momentary contact Fault acknowledgement
B_LB	Default tag table	Bool	%I8.0	Light barrier
S_Bay1	Default tag table	Bool	%I8.1	Momentary contact Bay 1
S_Bay2	Default tag table	Bool	%I8.2	Momentary contact Bay 2
S_Bay3	Default tag table	Bool	%I8.3	Momentary contact Bay 3
S_Bay-LB	Default tag table	Bool	%I8.4	Momentary contact Light barrier bay
B_Bay1	Default tag table	Bool	%I8.5	Proximity sensor Bay 1
B_Bay2	Default tag table	Bool	%I8.6	Proximity sensor Bay 2
B_Bay3	Default tag table	Bool	%I8.7	Proximity sensor Bay 3
P_Operation	Default tag table	Bool	%Q4.1	Indicator light Operation ON
P_Fault	Default tag table	Bool	%Q5.0	Indicator light Conveyor fault
P_Bay1	Default tag table	Bool	%Q8.1	Indicator light Bay 1
P_Bay2	Default tag table	Bool	%Q8.2	Indicator light Bay 2
P_Bay3	Default tag table	Bool	%Q8.3	Indicator light Bay 3
P_Bay-LB	Default tag table	Bool	%Q8.4	Indicator light Light barrier bay
K_Right	Default tag table	Bool	%Q8.5	Run conveyor RIGHT
K_Left	Default tag table	Bool	%Q8.6	Run conveyor LEFT
P_Horn	Default tag table	Bool	%Q8.7	Alarm Horn
M_2Hz	Default tag table	Bool	%M10.3	Memory bit - flashing frequency 2 Hz
M_1Hz	Default tag table	Bool	%M10.5	Memory bit - flashing frequency 1Hz
M_aux_Op(15)	Default tag table	Bool	%M15.0	Edge auxiliary memory bit Operation ON
M_aux_LB	Default tag table	Bool	%M16.0	Edge auxiliary memory bit Light barrier
M_Jog_Right	Default tag table	Bool	%M16.2	Memory bit Jog conveyor RIGHT
M_Jog_Left	Default tag table	Bool	%M16.3	Memory bit Jog conveyor LEFT
M_Auto_Right	Default tag table	Bool	%M16.4	Memory bit Conveyor AUTO RIGHT
M_Auto_Left	Default tag table	Bool	%M16.6	Memory bit Conveyor AUTO LEFT
M_Conv_Fault	Default tag table	Bool	%M17.0	Memory bit Conveyor fault
M_max_Fault	Default tag table	Bool	%M17.7	max. no. faults reached
M_aux_Count	Default tag table	Bool	%M18.0	Auxiliary memory bit Count
M_aux_OP(18)	Default tag table	Bool	%M18.1	Edge auxiliary memory bit Operation ON
M_ACT=SETP	Default tag table	Bool	%M18.4	Setp. no. is reached
MW_ACT	Default tag table	Int	%MW20	Memory word, ACTUAL quantity of transported parts
MW_SETP	Default tag table	Int	%MW22	Memory word, SETPOINT quantity of parts to be transp.
S_ON	Default tag table	Bool	%M30.0	Momentary contact Operation ON
S_OFF	Default tag table	Bool	%M30.1	Momentary contact Operation OFF (NC contact)
S_Right	Default tag table	Bool	%M30.2	Jog conveyor RIGHT, momentary contact
S_Left	Default tag table	Bool	%M30.3	Jog conveyor LEFT, momentary contact
S_Ackn_HMI	Default tag table	Bool	%M31.0	Memory bit, acknowledge conveyor fault (Touchpanel)
M_TOF_Right	Default tag table	Bool	%M160.2	Memory bit time interlock Right
M_TOF_Left	Default tag table	Bool	%M160.3	Memory bit time interlock Left
MD_ConvRunti...	Default tag table	Time	%MD170	
<Add new>				

Contents 2

2. Commissioning the Hardware and Software

2.1. Objectives

At the end of the chapter the participant will ...

... be familiar with the different S7 block types

... be familiar with the principle of "structured programming"

... be familiar with the meaning of the process image tables (PII, PIQ)

... be able to explain the principle of cyclic program execution

… be able to establish an online connection to the controller

... be able to create a hardware station and parameterize it

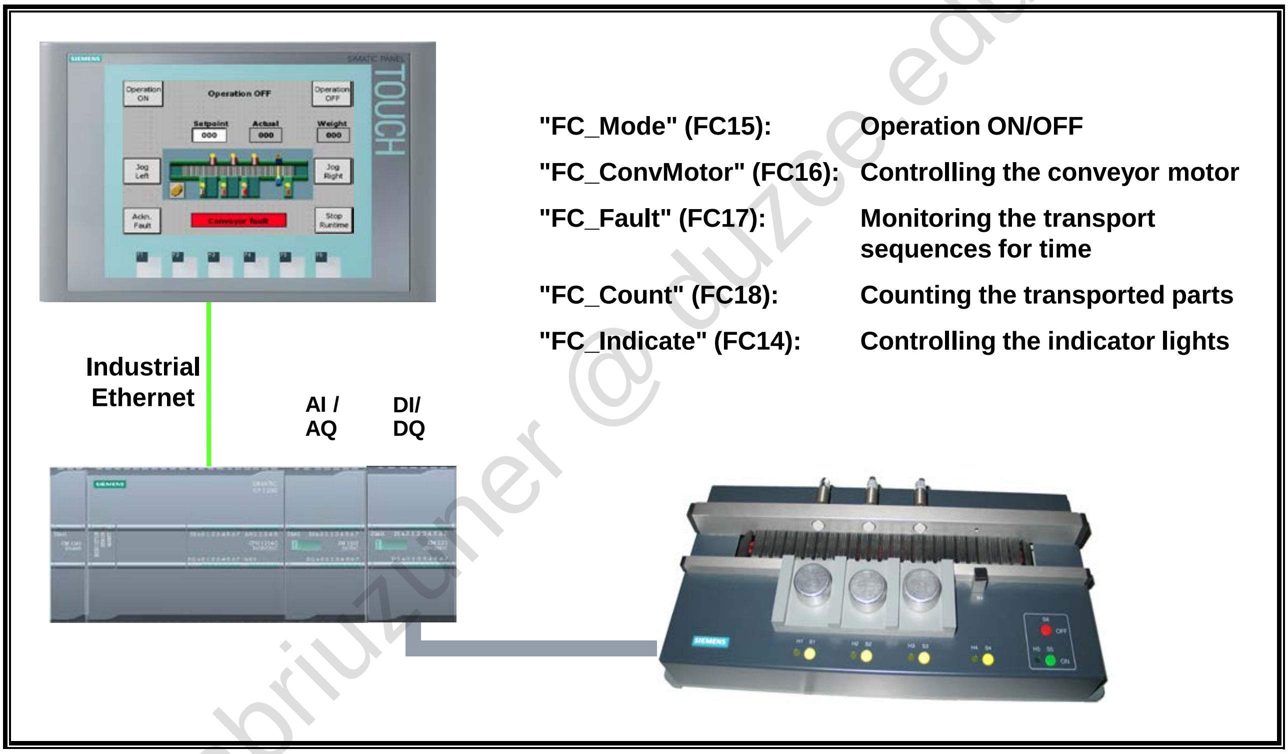
… be able to commission an existing PLC project

… be able to commission a touchpanel

Objectives

Important basics from the TIA-MICRO1 course are first of all repeated in this chapter. Then, an already configured system with described basic functions is commissioned.

2.2. Task Description: The Conveyor Model as Distribution Conveyor



"FC_Mode" (FC15):	**Operation ON/OFF**
"FC_ConvMotor" (FC16):	**Controlling the conveyor motor**
"FC_Fault" (FC17):	**Monitoring the transport sequences for time**
"FC_Count" (FC18):	**Counting the transported parts**
"FC_Indicate" (FC14):	**Controlling the indicator lights**

Task Description

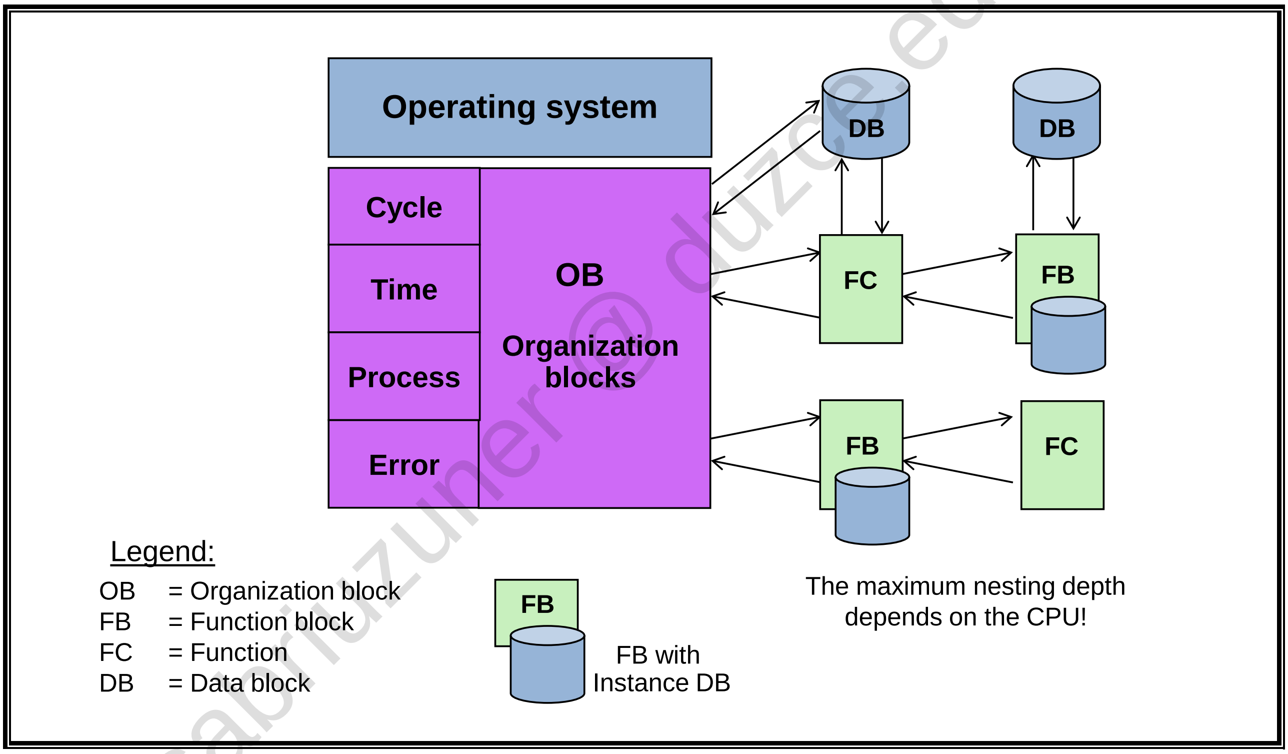
In this chapter, the conveyor model is commissioned as a distribution conveyor. For this, an appropriate exercise is carried out at the end of the chapter.

A CPU and touchpanel are already generated in the initial project. The user program of the CPU is structured as follows:

1. "FC_Mode“ (FC15)
   – Switching on and switching off the operation is implemented in this function.
2. "FC_ConvMotor" (FC16)
   – The control of the conveyor motor for the states Operation ON and Operation OFF is programmed in this function.
3. "FC_Fault" (FC17)
   – In this function, the conveyor is monitored for time. If a transport sequence takes longer than 6 seconds, the conveyor is stopped and an error message is triggered.
4. "FC_Count" (FC18)
   – Counting the already transported parts is programmed in this function. Setpoint and actual quantity are specified or read out via the touchpanel.
5. "FC_Indicate" (FC14)
   – The control of the indicator lights during operation is programmed in this function.

The conveyor model can be completely controlled via the touchpanel. The functions necessary for this are already stored in the HMI project.

2.3. Types of Program Blocks



Blocks

The programmable logic controller provides various types of blocks in which the user program and the related data can be stored. Depending on the requirements of the process, the program can be structured in different blocks. You can use the entire operation set in all blocks (FB, FC and OB).

Organization Blocks (OBs)

Organization blocks (OBs) form the interface between the operating system and the user program. The entire program can be stored in OB1 that is cyclically called by the operating system (linear program) or the program can be divided and stored in several blocks (structured program).

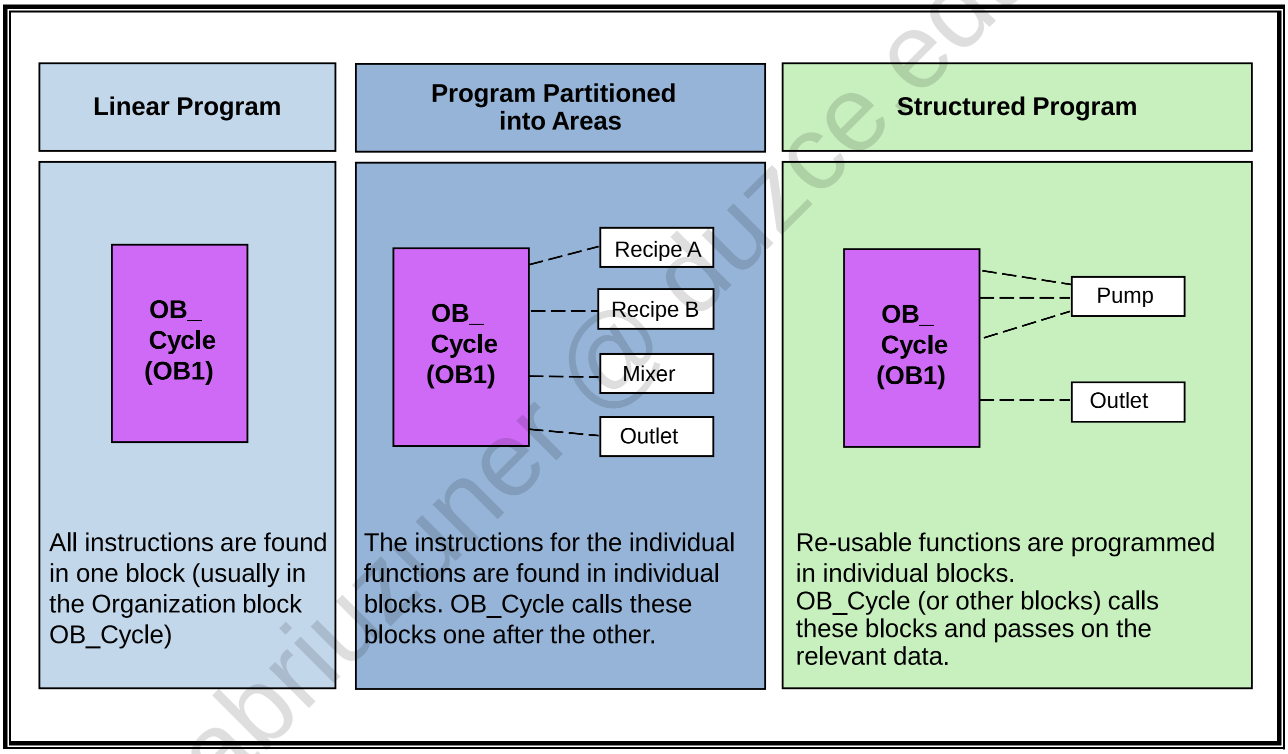
Functions (FCs)

A function (FC) contains a partial functionality of the program. It is possible to program functions as parameter-assignable so that when the function is called it can be assigned parameters. As a result, functions are also suited for programming frequently recurring, complex partial functionalities such as calculations.

Function Blocks (FBs)

Basically, function blocks offer the same possibilities as functions. In addition, function blocks have their own memory area in the form of instance data blocks. As a result, function blocks are suited for programming frequently recurring, complex functionalities such as closed-loop control tasks.

2.4. Possibilities for Program Structuring


Linear Program
Program Partitioned into Areas
Structured Program
OB_ Cycle (OB1)
OB_ Cycle (OB1)
Recipe A
Recipe B
Mixer
Outlet
OB_ Cycle (OB1)
Pump
Outlet
All instructions are found in one block (usually in the Organization block OB_Cycle)
The instructions for the individual functions are found in individual blocks. OB_Cycle calls these blocks one after the other.
Re-usable functions are programmed in individual blocks.
OB_Cycle (or other blocks) calls these blocks and passes on the relevant data.

Linear Programming

You can solve small automation tasks by writing the entire user program linearly in one cycle OB. This is only recommended for simple programs.

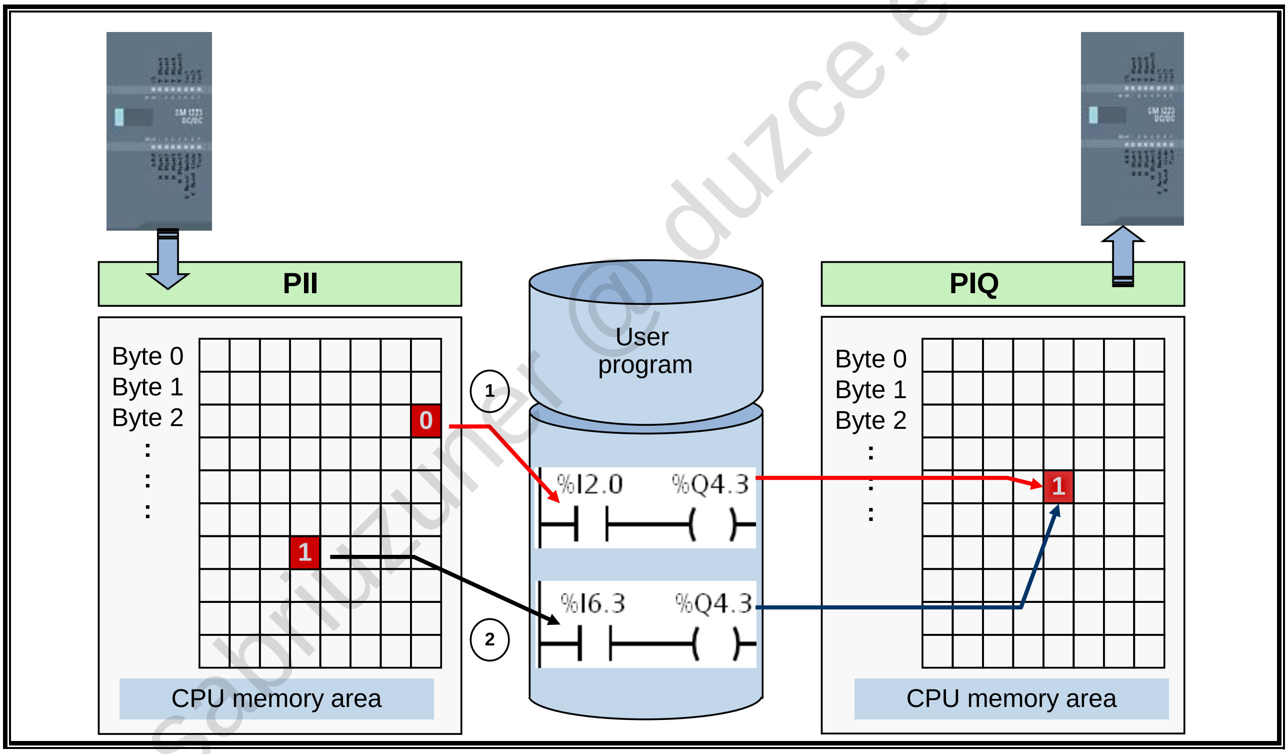
Structured Programming

Complex automation tasks can be implemented and maintained more easily when they are divided into smaller partial tasks which reflect the technological functions of the automation process or if they are to be used repeatedly. In the user program, these partial tasks are represented by appropriate program parts, the blocks. Each block is an independent segment of the user program.

Advantages

1. Extensive programs can be clearly programmed.
2. Individual program parts can be standardized.
3. The program organization is simplified.
4. Changes to the program can be carried out more easily.
5. The program test is simplified because it can be made section by section.
6. Commissioning is made easier.

2.5. Process Images



Process Images

For the storage of all digital input and output states, the CPU has reserved memory areas: the Process-Image Input table (PII) and the Process-Image Output table (PIQ). During program execution, the CPU accesses these memory areas exclusively. It does not access the digital input and output modules directly.

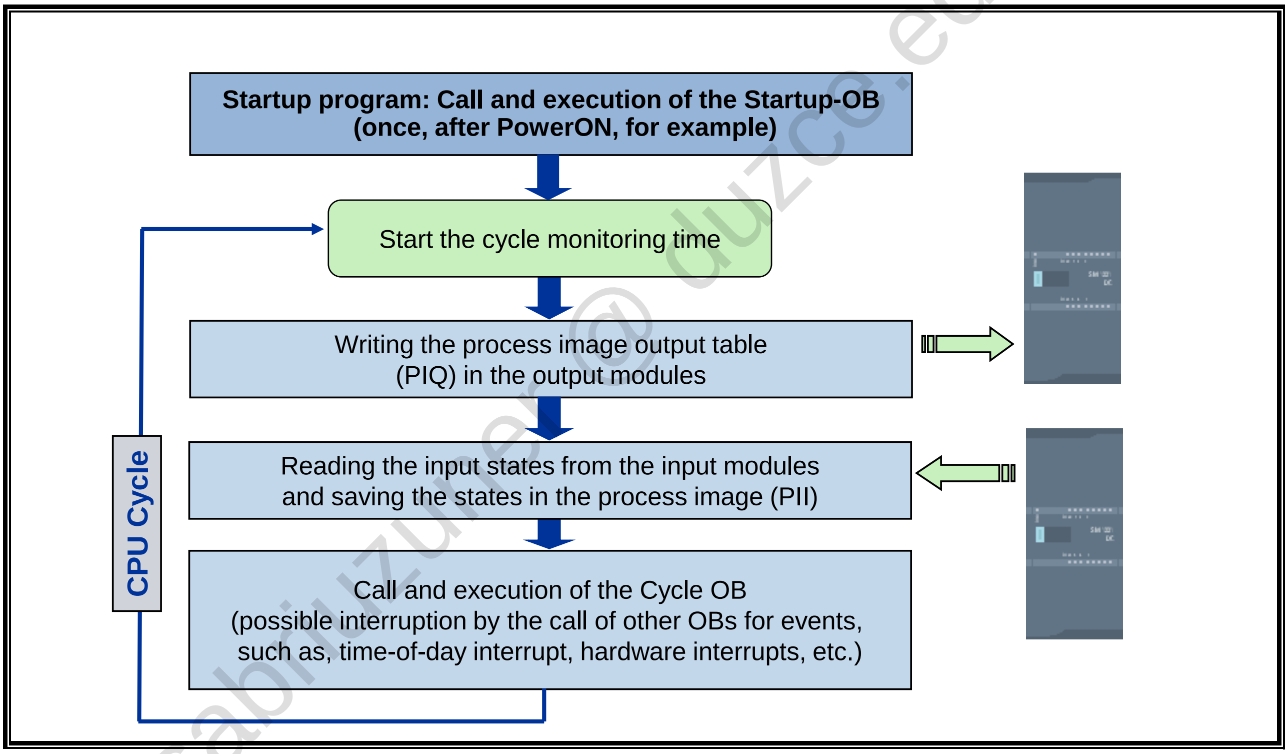
PIQ

The Process-Image Output table (PIQ) is the memory area in which the states of all digital outputs are stored. The PIQ is output to the digital output modules at the beginning of the cycle. Outputs can be assigned as well as queried in the program. If an output is assigned a state in several locations in the program, then only the state that was assigned last is transferred to the particular output module. As a rule, these types of double assignments are programming errors.

PII

The Process-Image Input table (PII) is the memory area in which the states of all digital inputs are stored. After the PIQ is output, the PII is read by the digital input modules. When an input is linked, the state of this input stored in the PII is linked. This state cannot change within a cycle since the PII is only updated or read-in at the beginning of a cycle. This guarantees that when there are multiple queries of the input in one cycle, the same result is always supplied.

2.6. Cyclic Program Execution


Startup program: Call and execution of the Startup-OB
(once, after PowerON, for example)
Start the cycle monitoring time
Writing the process image output table
(PIQ) in the output modules
Reading the input states from the input modules
and saving the states in the process image (PII)
Call and execution of the Cycle OB
(possible interruption by the call of other OBs for events,
such as, time-of-day interrupt, hardware interrupts, etc.)
CPU Cycle

Restart

When you switch on or switch from STOP --> RUN, the CPU carries out a complete restart (execution of all Startup OBs). During restart, the operating system deletes the non-retentive bit memories and resets all stored hardware and diagnostic interrupts.

Cyclic Program Execution

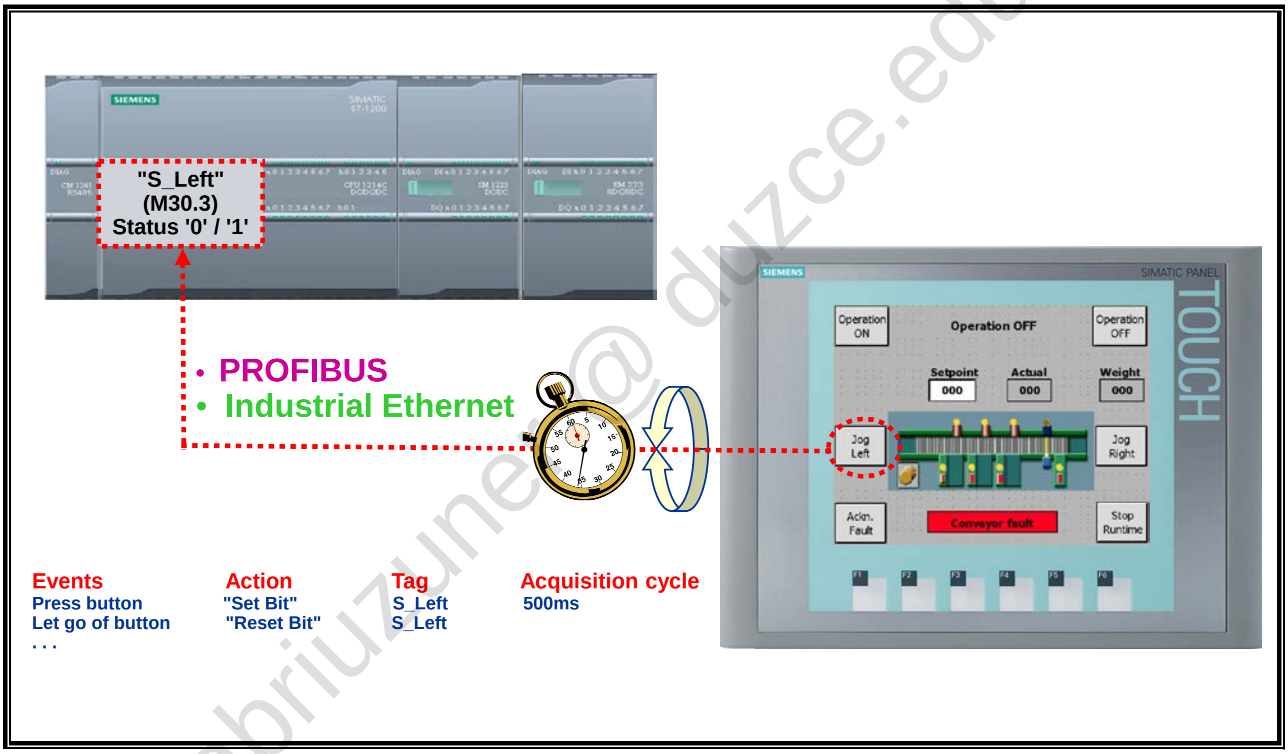
Cyclic program execution occurs in an endless loop. After the execution of a program cycle is completed, the execution of the next cycle occurs automatically. In every program cycle, the CPU carries out the following steps.

7. The CPU starts the cycle monitoring time,
8. The CPU transfers the output states from the process image output table to the digital output modules,
9. The CPU scans the states of the input signals and updates the process image input table,
10. The CPU sequentially processes the instructions of the user program using the process images, not the inputs and outputs of the digital input / output modules,

Cycle Time and Cycle Monitoring Time

The time that the CPU requires for the execution of the complete program cycle, is the cycle time which is monitored for time by the CPU operating system. If the cycle time exceeds the cycle monitoring time defined in the CPU properties, the system requests the call of the “time-error OB”. If it is loaded, it is executed. If the CPU exceeds twice the cycle monitoring time, the CPU goes into the STOP state.

2.7. Data Exchange between Touchpanel and CPU



"S_Left"
(M30.3)
Status '0' / '1'

- PROFIBUS
- Industrial Ethernet

Events	Action	Tag	Acquisition cycle
Press button	"Set Bit"	S_Left	500ms
Let go of button	"Reset Bit"	S_Left	
. . .			

Tags

Data is exchanged between SIMATIC S7 and the HMI system via process tags. For this, tags are created in the configuration of the WinCC system and are then assigned to a data area of the CPU. The HMI system reads out the value of the tags cyclically and displays it, for example, in an output field.

Data Areas

For the configuration of the tags, the following global data areas of the CPU can be used:

1. Data blocks (DB)
2. Bit memories (M)
3. Inputs (I) and outputs (Q)
4. I/O (peripheral) inputs and I/O (peripheral) outputs

HMI systems also recognize local tags without process connection, that is, these tags are exclusively processed internally and also do not reserve any communication resources whatsoever.

Communication

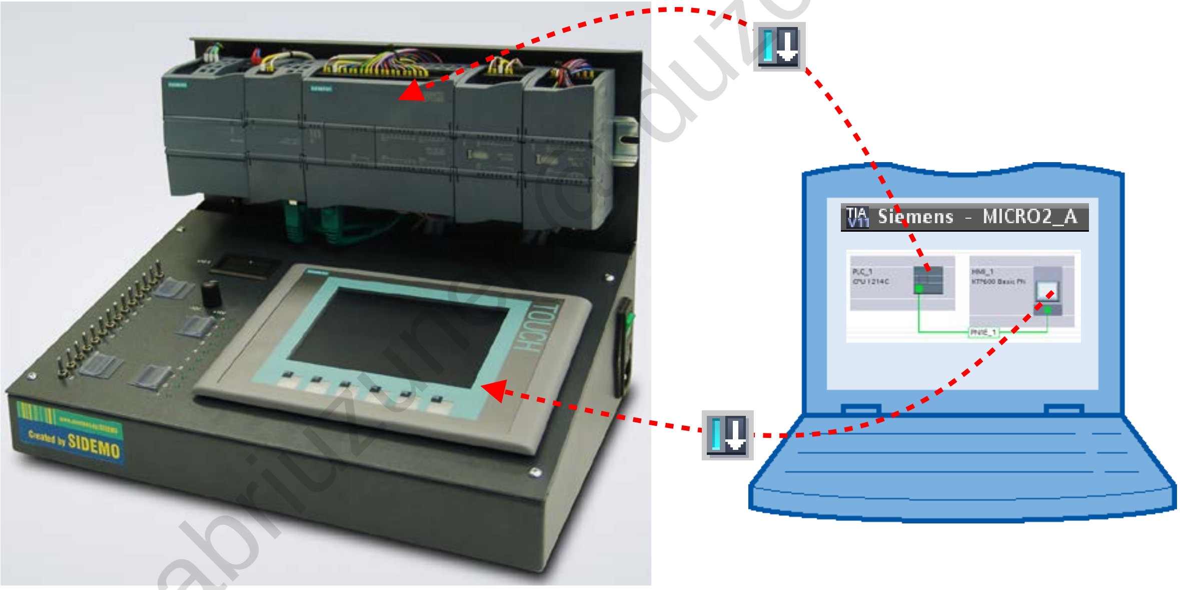
The HMI devices can communicate with the controller via the bus systems MPI, PROFIBUS DP or Industrial Ethernet. The S7 protocol is used for this purpose. Communication is handled by the operating systems of the S7 CPU and the HMI system. No user programming on the S7 is required for this purpose.
An HMI device can exchange data with more than one controller at the same time.

Updating

Data is transferred cyclically between SIMATIC S7 and the HMI system, that is, process tags are read and written cyclically in accordance with the configured refresh (update) time.

2.8. Task Description: Commissioning the Training Case

The required basic project "MICRO2_A" is already located in <Drive>:Archives\TIA_Portal\TIA-MICRO2 on your hard drive and must, if necessary, be adjusted to the existing hardware and loaded into the CPU or the touchpanel.

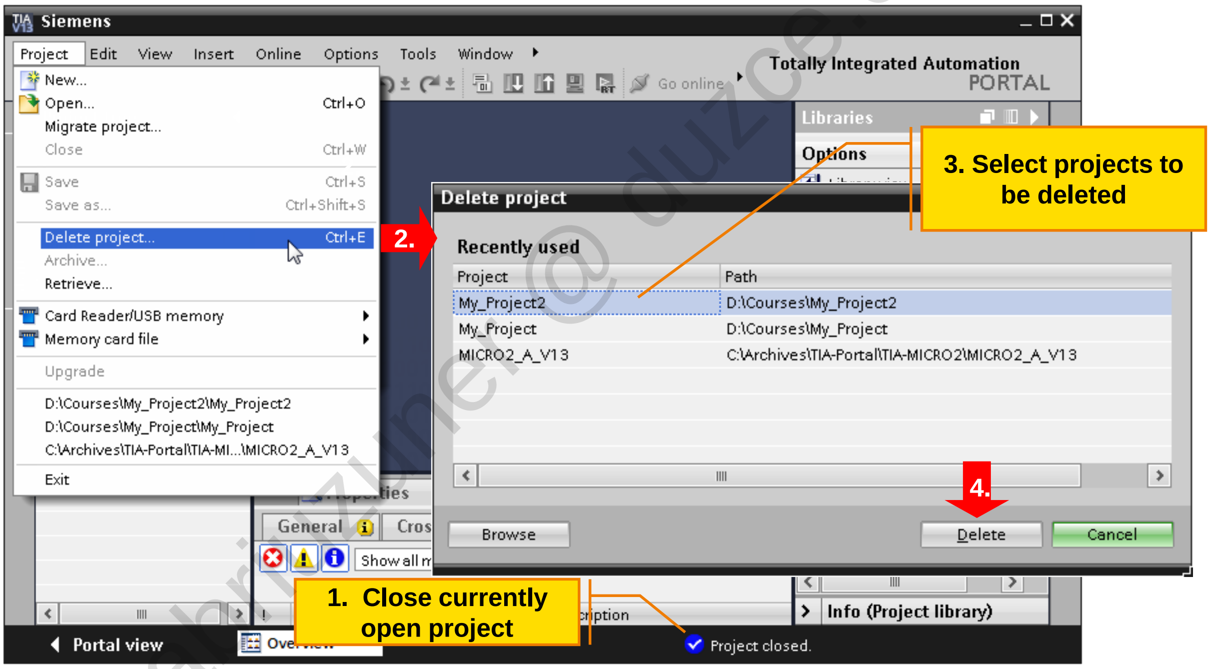


Task Description

The S7-1214 training case with touchpanel is to be commissioned.

☞ The basic project required for this is already located on your hard drive and must, if necessary, be adjusted to the existing hardware and loaded into the CPU or the touchpanel.

2.9. Exercise 1: Deleting Old Projects



Project	Path
My_Project2	D:\Courses\My_Project2
My_Project	D:\Courses\My_Project
MICRO2_A_V13	C:\Archives\TIA-Portal\TIA-MICRO2\MICRO2_A_V13

3. Select projects to be deleted

1. Close currently open project

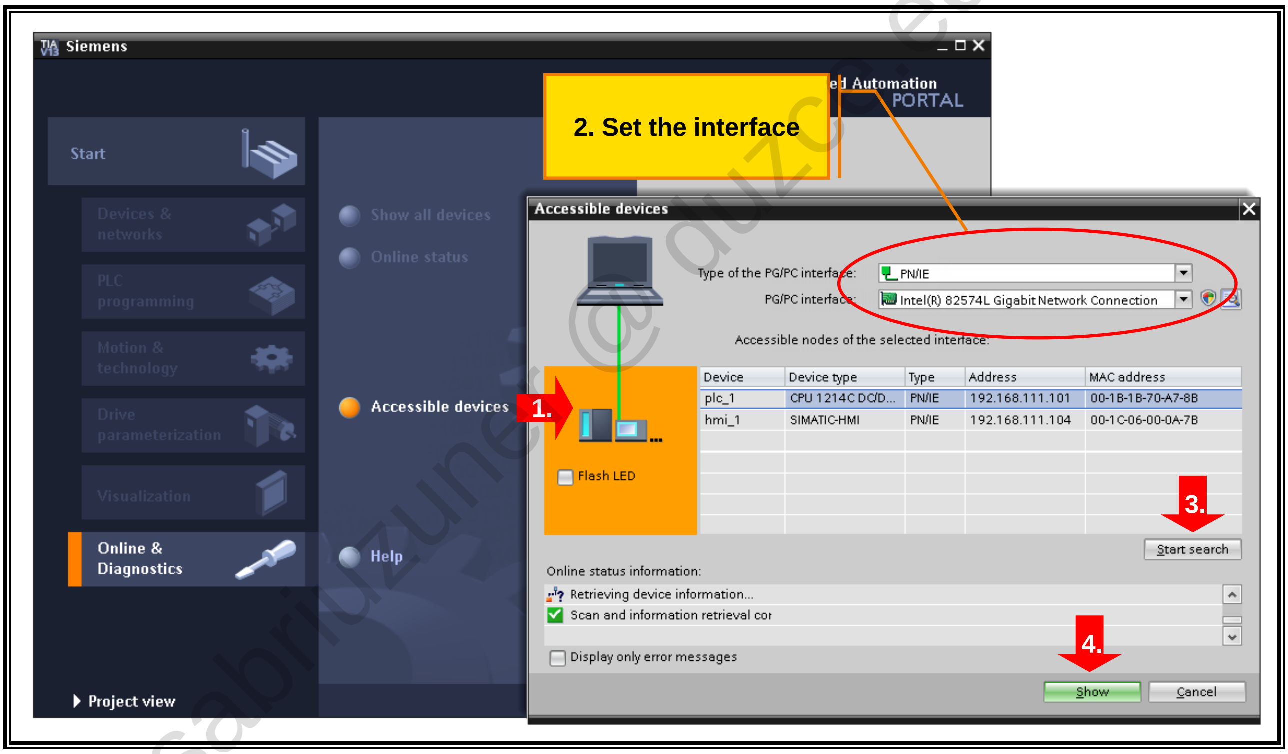
Task

Delete all existing projects.

What to Do

1. Start the TIA Portal
2. Switch to the Project view.
3. Close any open projects.
4. Let the system show you the existing projects and delete them as shown.

2.10. Exercise 2: Establishing an Online Connection to the CPU



Accessible devices

Type of the PG/PC interface: PN/IE

PG/PC interface: Intel(R) 82574L Gigabit Network Connection

Accessible nodes of the selected interface:

Device	Device type	Type	Address	MAC address
plc_1	CPU 1214C DC/D...	PN/IE	192.168.111.101	00-1B-1B-70-A7-8B
hmi_1	SIMATIC-HMI	PN/IE	192.168.111.104	00-1C-06-00-0A-7B

□ Flash LED

Start search

Online status information:

Retrieving device information...

✓ Scan and information retrieval cor

□ Display only error messages

Show   Cancel

Task

Establish a connection to the CPU 1214 and use the automatic IP address assignment of the TIA Portal for this.

What to Do

1. Switch to the Portal view.
2. Start the "Accessible devices" function.
   "Online & Diagnostics" → "Accessible devices"
3. Select "PN/IE" for the "Type of the PG/PC interface".
4. Select the respective Ethernet interface of your field PG.

☞ Field PG M3: This device has 2 Ethernet interfaces! Make sure that you use the correct interface!

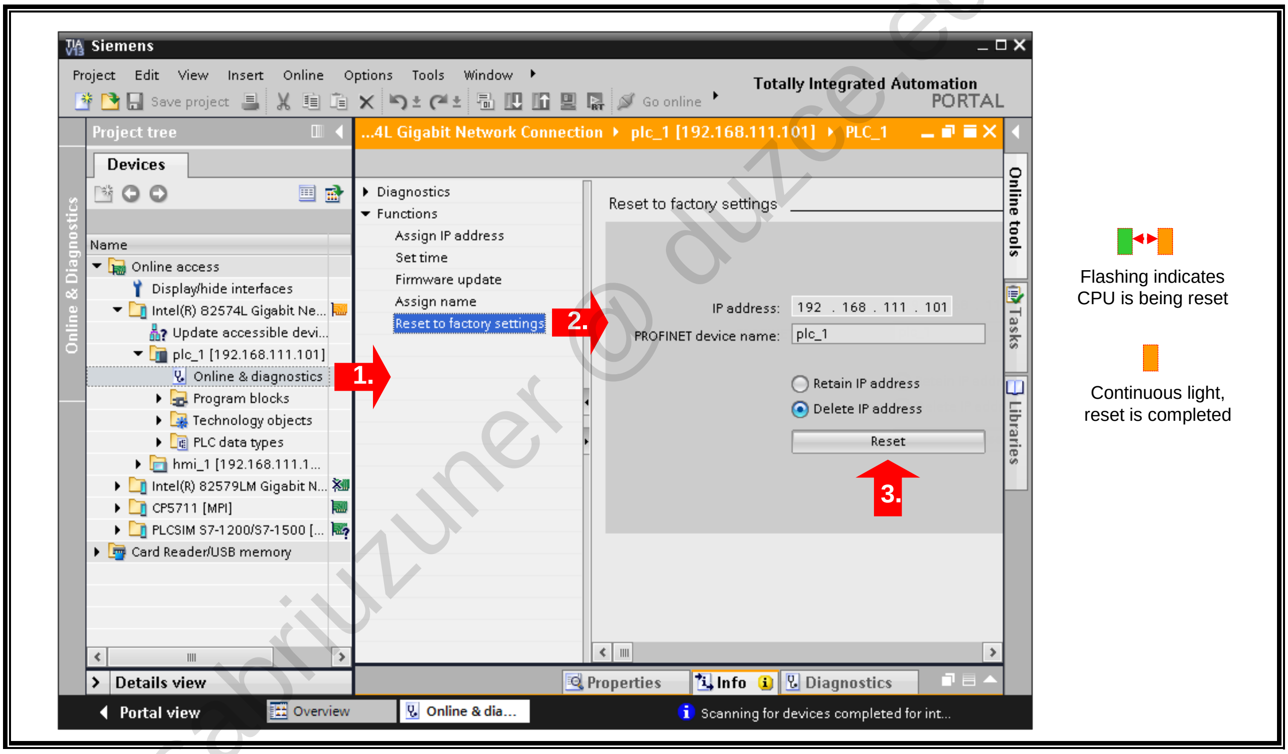
5. After searching, select the CPU 1214 → Device type "S7-1200".

☞ Should more than one S7-1200 be found, you can find out whether you have selected the correct CPU through the function "Flash LED". The Status LEDs of the respective CPU flash.

6. Click on "Show" and confirm the follow-up prompt with "Yes".

☞ The telegram service "TCP/IP" is used to execute certain functions. For this, the CPU and PG must be located in the same IP subnet. The TIA Portal assigns your PG a temporary, alternative IP address which is located in the same subnet as the CPU. That way you can work freely.

2.11. Exercise 3: Resetting the CPU to Factory Settings



Reset to factory settings

IP address: 192 . 168 . 111 . 101

PROFINET device name: plc_1

○ Retain IP address

● Delete IP address

Reset

Scanning for devices completed for int...

Flashing indicates CPU is being reset

Continuous light, reset is completed

Task

Establish a defined initial state for the following configuration steps by resetting the CPU to factory settings.

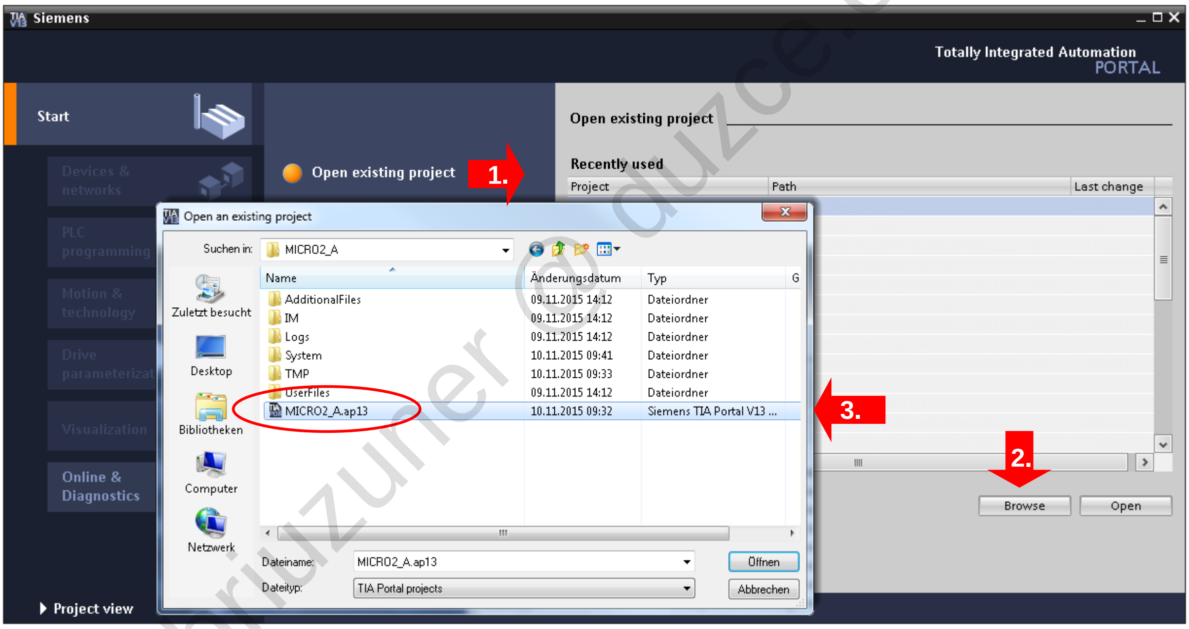
What to Do

1. In the Project view, in the ‘Online access’, navigate to your CPU and open "Online & diagnostics".

   "Online access" → your CPU → Double-click on "Online & diagnostics"

2. Open the dialog "Reset to factory settings".
3. Select the item "Delete IP address" and click on "Reset".

2.12. Exercise 4.1: Opening an Existing Project



Suchen in: MICRO2_A

Name	Änderungsdatum	Typ
AdditionalFiles	09.11.2015 14:12	Dateiordner
IM	09.11.2015 14:12	Dateiordner
Logs	09.11.2015 14:12	Dateiordner
System	10.11.2015 09:41	Dateiordner
TMP	10.11.2015 09:33	Dateiordner
UserFiles	09.11.2015 14:12	Dateiordner
MICRO2_A.ap13	10.11.2015 09:32	Siemens TIA Portal V13 ...

Dateiname: MICRO2_A.ap13

Dateityp: TIA Portal projects

The required basic project "MICRO2_A" is already located in <Drive>:Archives\TIA_Portal\TIA-MICRO2 on your hard drive and must, if necessary, be adjusted to the existing hardware and loaded into the CPU or the touchpanel.

Task

Open the basic project and save it using the name "My Project".

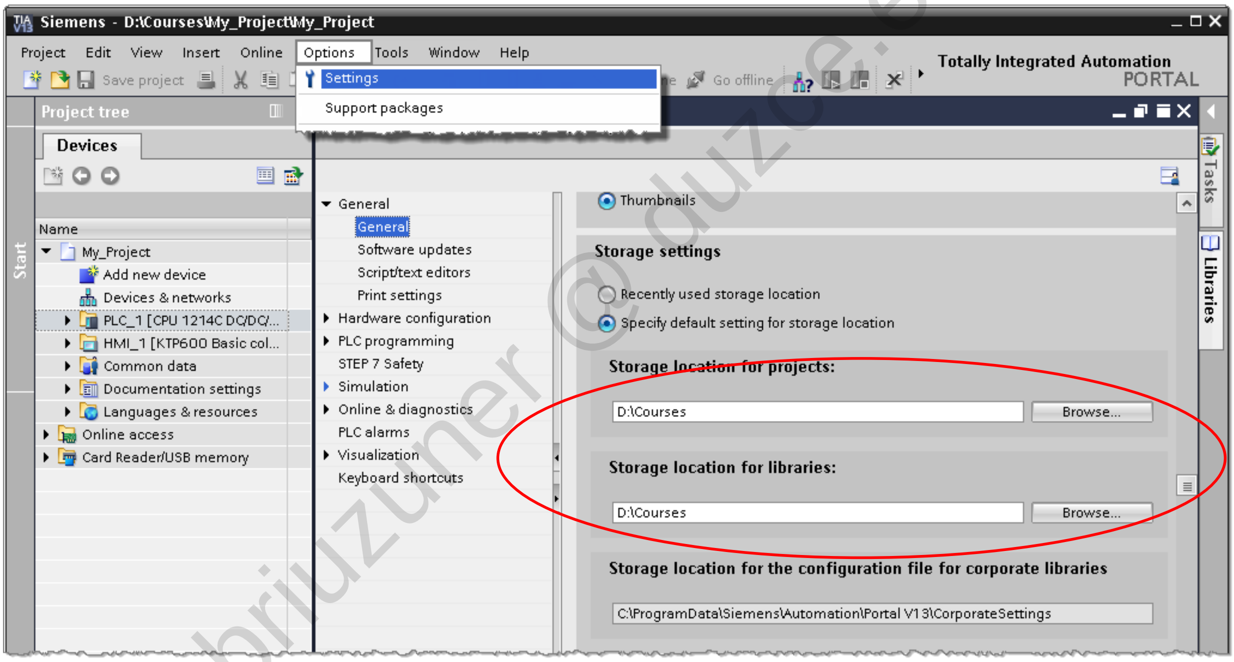
What to Do

1. Switch to the Portal view and select the menu item "Open existing project".
2. Click on "Browse" and navigate to the basic project folder. Open the project.

   `<Drive>:\Archives\TIA_Portal\TIA-MICRO2\MICRO2_A` → *Double-click on MICRO2_A.ap13*
3. Open the Project view.
4. Save your project using the name "My_Project".

   Project → Save as… → "My_Project"

2.13. Exercise 4.2: Saving the Project in…



● Thumbnails

Storage settings

○ Recently used storage location

● Specify default setting for storage location

Storage location for projects:

D:\Courses Browse...

Storage location for libraries:

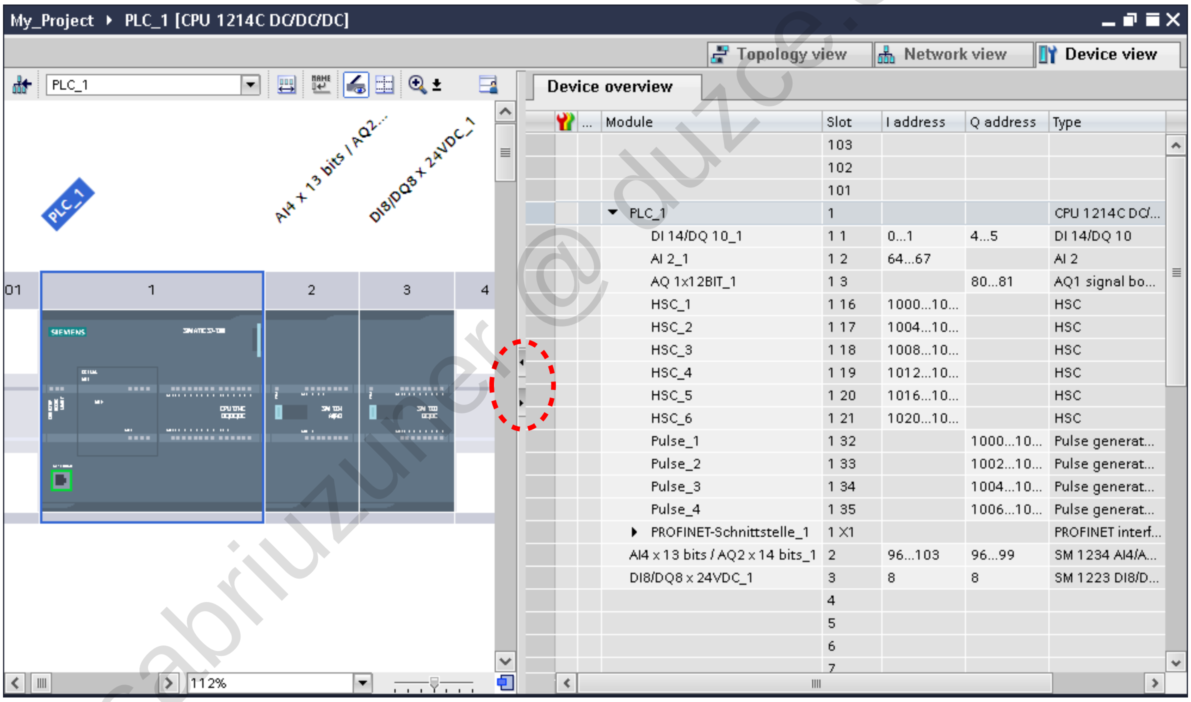
D:\Courses Browse...

Storage location for the configuration file for corporate libraries

C:\ProgramData\Siemens\Automation\Portal V13\CorporateSettings

Save the project in D:\Courses\My_Project

2.14. Exercise 5: Checking the Device Configuration and, if necessary, Adjusting It



	...	Module	Slot	I address	Q address	Type
			103			
			102			
			101			
		▾ PLC_1	1			CPU 1214C DC/...
		DI 14/DQ 10_1	1 1	0...1	4...5	DI 14/DQ 10
		AI 2_1	1 2	64...67		AI 2
		AQ 1x12BIT_1	1 3		80...81	AQ1 signal bo...
		HSC_1	1 16	1000...10...		HSC
		HSC_2	1 17	1004...10...		HSC
		HSC_3	1 18	1008...10...		HSC
		HSC_4	1 19	1012...10...		HSC
		HSC_5	1 20	1016...10...		HSC
		HSC_6	1 21	1020...10...		HSC
		Pulse_1	1 32		1000...10...	Pulse generat...
		Pulse_2	1 33		1002...10...	Pulse generat...
		Pulse_3	1 34		1004...10...	Pulse generat...
		Pulse_4	1 35		1006...10...	Pulse generat...
		▸ PROFINET-Schnittstelle_1	1 X1			PROFINET interf...
		AI4 x 13 bits / AQ2 x 14 bits_1	2	96...103	96...99	SM 1234 AI4/A...
		DI8/DQ8 x 24VDC_1	3	8	8	SM 1223 DI8/D...
			4			
			5			
			6			
			7			

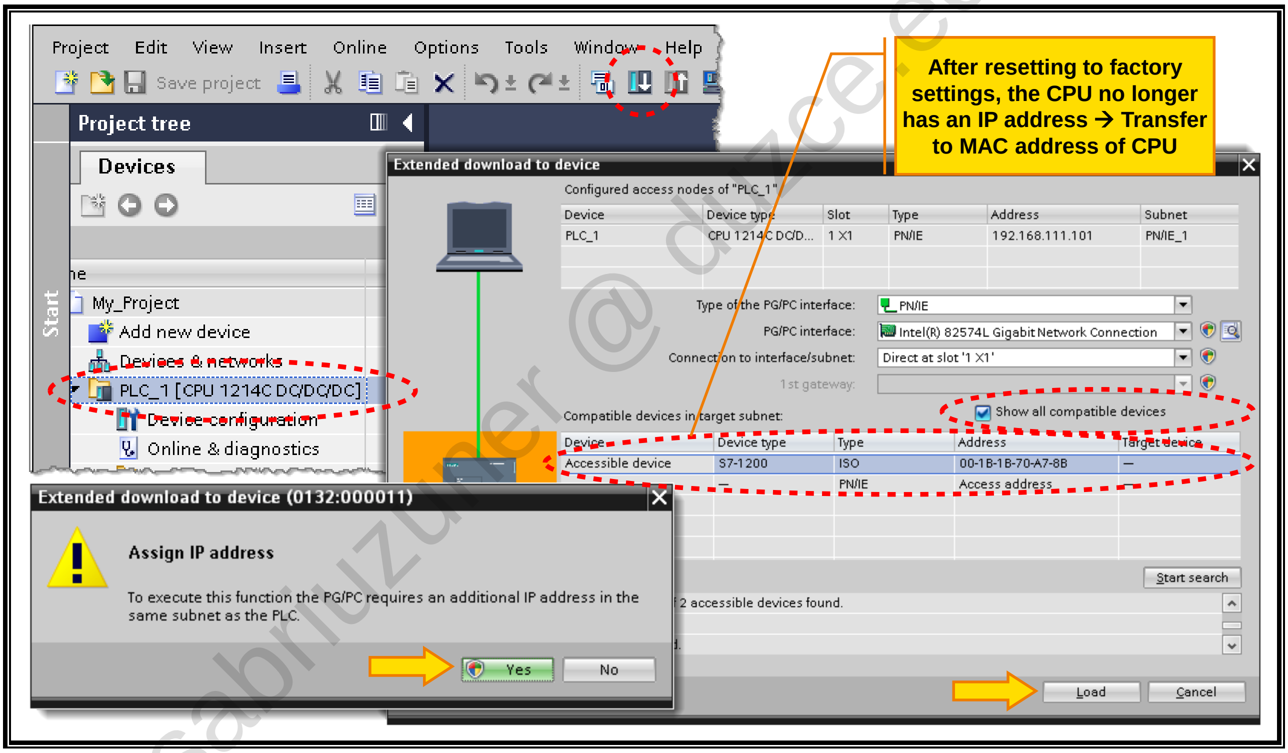
Task

Check the already existing CPU device configuration and the set I/O addresses.

What to Do

If necessary, exchange deviating modules and check the I/O addresses of the CPU and SMs shown in the picture.

2.15. Exercise 6: Downloading the Device Configuration and User Program into the CPU



After resetting to factory settings, the CPU no longer has an IP address → Transfer to MAC address of CPU

Extended download to device

Configured access nodes of "PLC_1"

Device	Device type	Slot	Type	Address	Subnet
PLC_1	CPU 1214C DC/D...	1 X1	PN/IE	192.168.111.101	PN/IE_1

Type of the PG/PC interface: PN/IE

PG/PC interface: Intel(R) 82574L Gigabit Network Connection

Connection to interface/subnet: Direct at slot '1 X1'

1st gateway:

Compatible devices in target subnet: ☑ Show all compatible devices

Device	Device type	Type	Address	Target device
Accessible device	S7-1200	ISO	00-1B-1B-70-A7-8B	—
—		PN/IE	Access address	—

Start search

2 accessible devices found.

Load Cancel

Extended download to device (0132:000011)

Assign IP address

To execute this function the PG/PC requires an additional IP address in the same subnet as the PLC.

Yes No

Task

Transfer the PLC_1 (hardware and software) to the S7-1214.

What to Do

1. Open the context menu of the CPU and then the Download dialog.
   Right-click on "PLC_1" → "Download to device" → "All"
2. Make the same PG/PC interface settings as in Exercise 2.

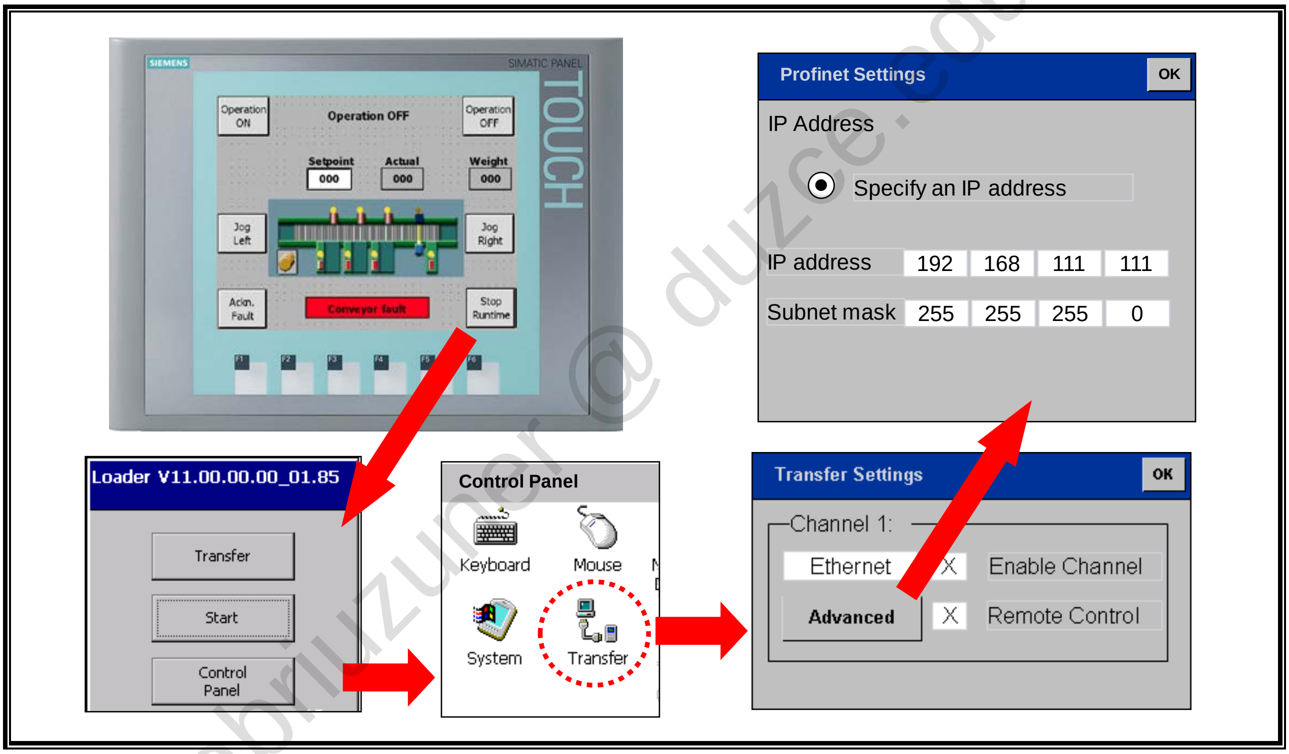
☞ A first search starts automatically. The CPU with the configured IP address is searched for. Since there is currently no station with this address, no results are shown.

3. Check the option "Show all compatible devices".

☞ The checkmark causes all accessible stations to be shown and not just the one with the configured IP address.

4. Select the device "S7-1200 | ISO | MAC address". If this applies to more than one station, you can use the function "Flash LED" (see Exercise 2).
5. Confirm the follow-up prompt with "Yes".
6. In the following dialog, click on "Load".

2.16. Exercise 7: Setting the IP Address of the Touchpanel



Loader V11.00.00.00_01.85

Transfer

Start

Control Panel

Profinet Settings OK

IP Address

● Specify an IP address

IP address 192 168 111 111

Subnet mask 255 255 255 0

Transfer Settings OK

Channel 1:

Ethernet ☒ Enable Channel

Advanced ☒ Remote Control

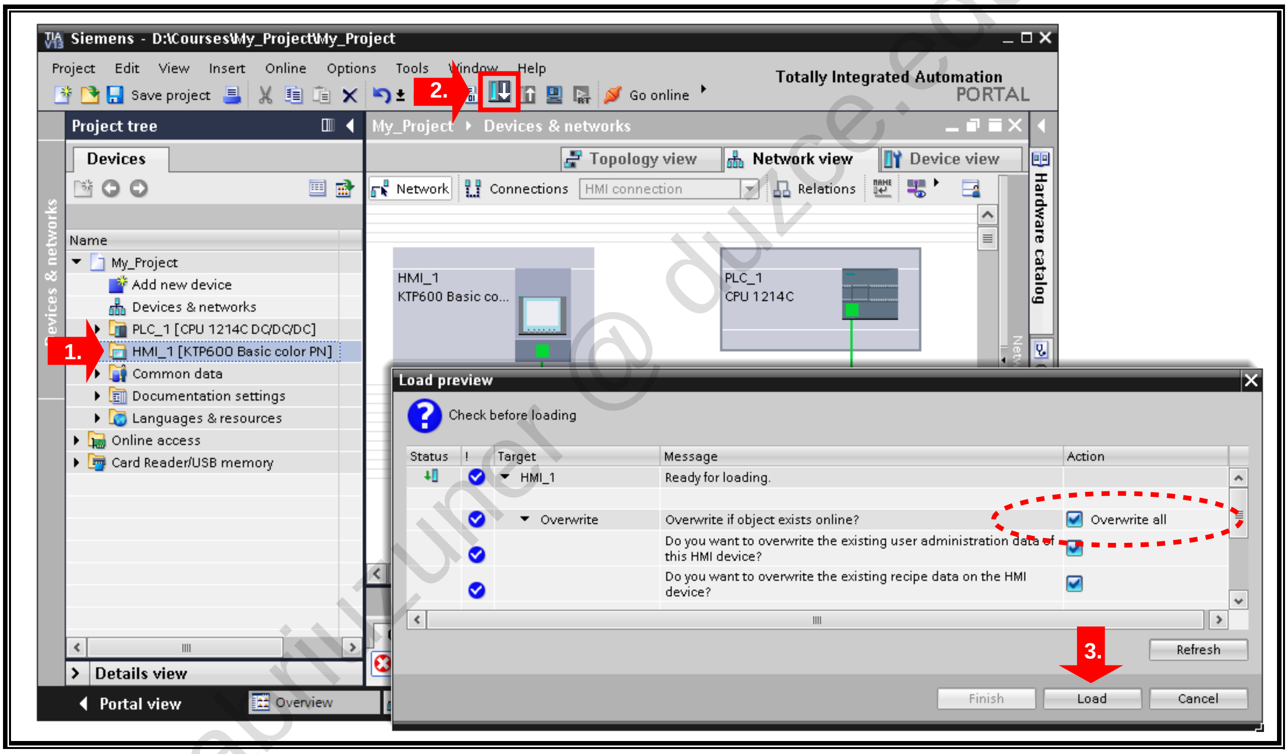
Task

On the touchpanel, set the IP address stored in the project.

What to Do

1. Stop the current Runtime
2. Follow the instructions shown in the picture.

2.17. Exercise 8: Transferring the Touchpanel Project



Load preview

Check before loading

Status	!	Target	Message	Action
	✓	HMI_1	Ready for loading.	
	✓	Overwrite	Overwrite if object exists online?	☑ Overwrite all
	✓		Do you want to overwrite the existing user administration data of this HMI device?	☑
	✓		Do you want to overwrite the existing recipe data on the HMI device?	☑

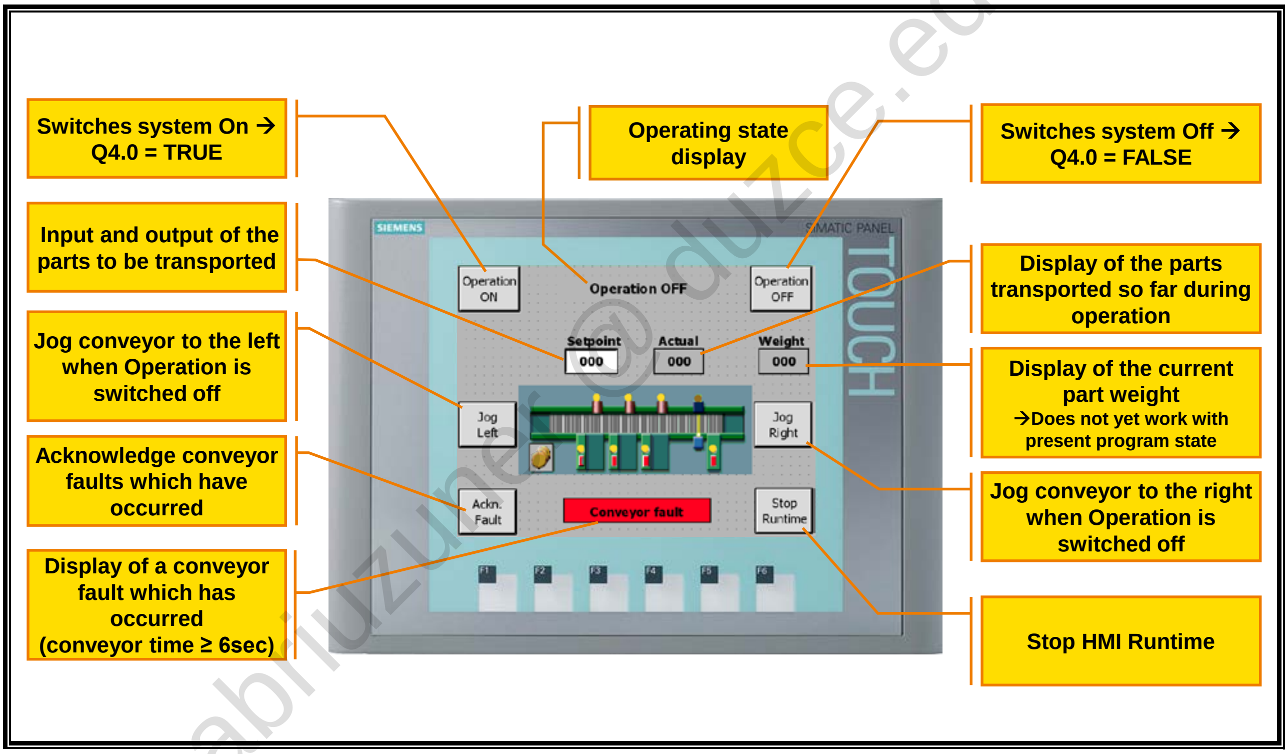
Task

Download the existing HMI project into the touchpanel.

What to Do

1. Select the touchpanel in the Project tree.
2. Click on the "Download to device" button as shown in the picture.
3. Check the option "Overwrite all" and then click on "Load".

2.18. Exercise 9: Function Test Touchpanel Project and CPU Program



Switches system On → Q4.0 = TRUE

Operating state display

Switches system Off → Q4.0 = FALSE

Input and output of the parts to be transported

Display of the parts transported so far during operation

Jog conveyor to the left when Operation is switched off

Display of the current part weight
→Does not yet work with present program state

Acknowledge conveyor faults which have occurred

Jog conveyor to the right when Operation is switched off

Display of a conveyor fault which has occurred (conveyor time ≥ 6sec)

Stop HMI Runtime

What to Do

Check the system functions described in the following.

Operation

The system can be switched on and off via the buttons "Operation ON" and "Operation OFF".

Operation OFF

The conveyor can be moved in the appropriate direction via the buttons "Jog Right" and "Jog Left".

Operation ON

A value > 0 must be entered via the input field "Setpoint (quantity)".

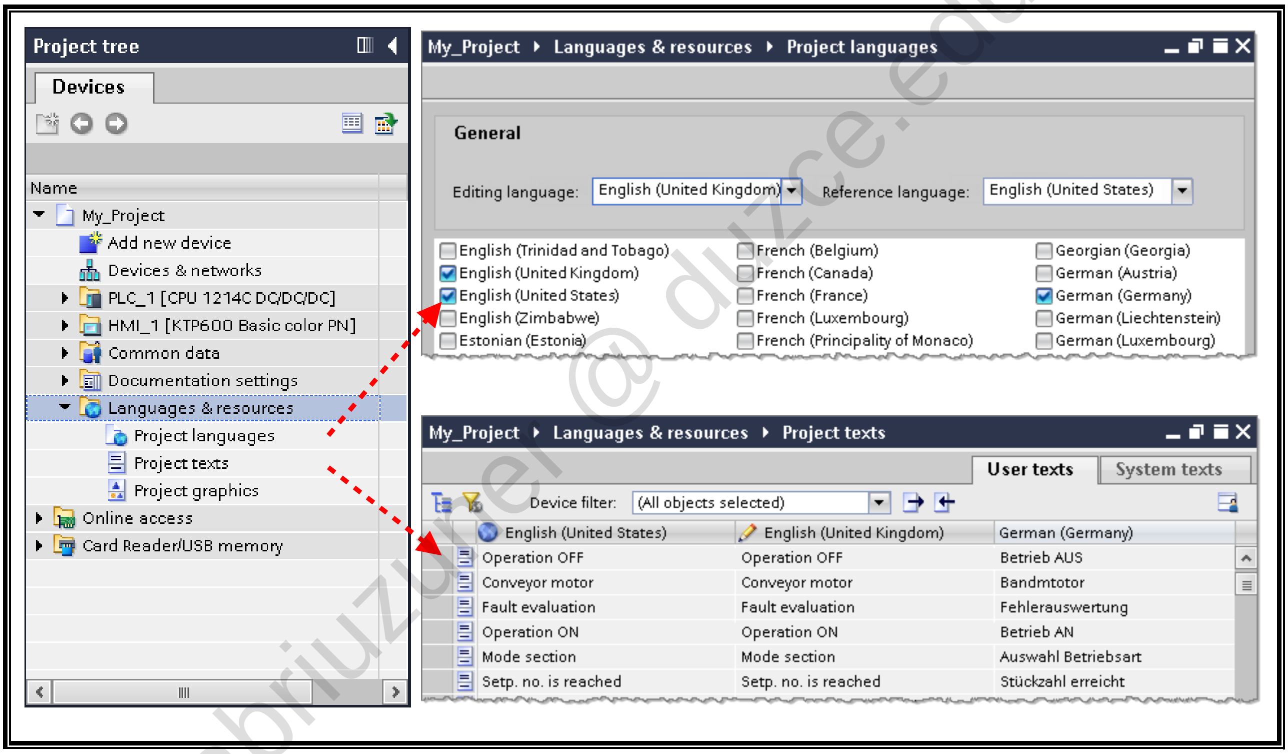
When the conveyor is standing, the indicator lights show with a continuous light at Bays 1and 2 that a new part can be placed on the conveyor. If this has happened, the indicator light at the particular bay shows with a 1Hz flashing light that the transport sequence can be started by pressing the bay pushbutton. The part is transported until it has passed through the light barrier. During transport, the bay indicator lights show a 2Hz flashing light.

If a transport sequence takes longer than 6 seconds, the conveyor is automatically stopped and the fault is indicated with a flashing light on the touchpanel and the simulator LED "P_Fault" (Q5.0). Only after the fault has been acknowledged via the appropriate touchpanel button or via the simulator pushbutton "S_Ackn" (I 1.0), can a new transport sequence be started.

The transported parts are counted as they pass through the light barrier and the number is displayed on the output field "Actual". If the number, which has been input via the input field "Setpoint", is reached, it is indicated via the conveyor indicator light "P_Bay-LB" (Q 8.4). Only after this is acknowledged via the conveyor pushbutton "S_Bay-LB" (I 8.4), can a new transport sequence be started.

The output field "Weight" does not yet display any values with the present program state. The associated programming is done in the chapter "Analog Value Processing".

2.19. Exercise 10: Selecting the Editing Language



My_Project ▸ Languages & resources ▸ Project languages

General

Editing language: English (United Kingdom)   Reference language: English (United States)

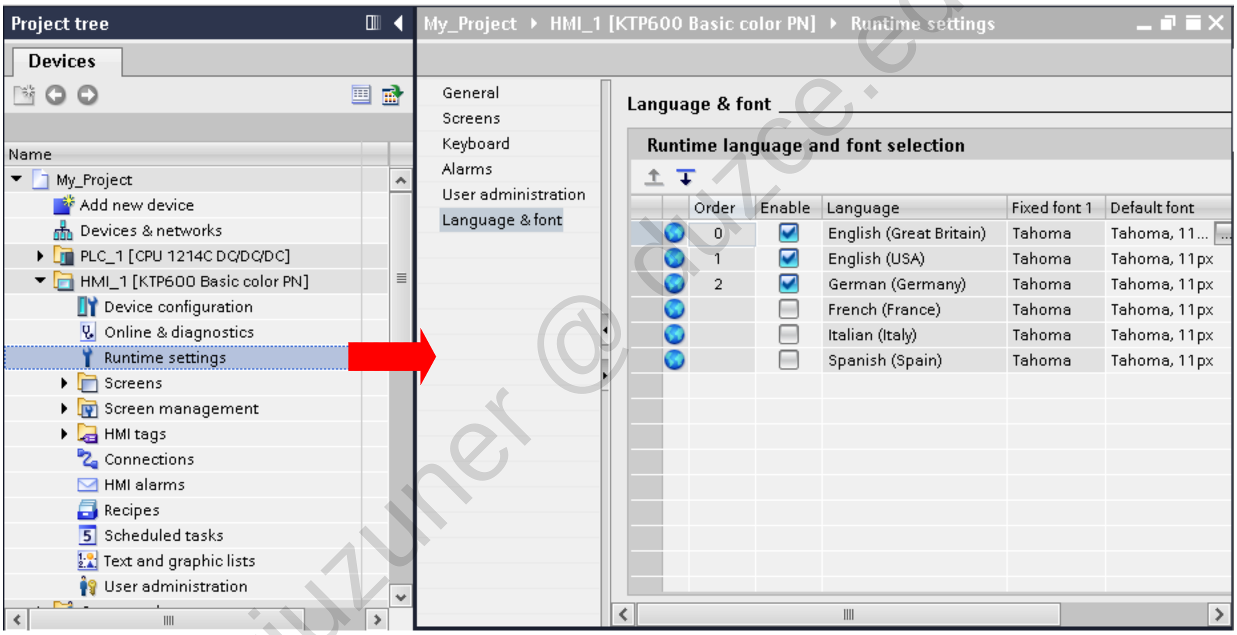
□ English (Trinidad and Tobago)	□ French (Belgium)	□ Georgian (Georgia)
☑ English (United Kingdom)	□ French (Canada)	□ German (Austria)
☑ English (United States)	□ French (France)	☑ German (Germany)
□ English (Zimbabwe)	□ French (Luxembourg)	□ German (Liechtenstein)
□ Estonian (Estonia)	□ French (Principality of Monaco)	□ German (Luxembourg)

My_Project ▸ Languages & resources ▸ Project texts

Device filter: (All objects selected)

English (United States)	English (United Kingdom)	German (Germany)
Operation OFF	Operation OFF	Betrieb AUS
Conveyor motor	Conveyor motor	Bandmtotor
Fault evaluation	Fault evaluation	Fehlerauswertung
Operation ON	Operation ON	Betrieb AN
Mode section	Mode section	Auswahl Betriebsart
Setp. no. is reached	Setp. no. is reached	Stückzahl erreicht

2.20. Exercise 11: Runtime Settings

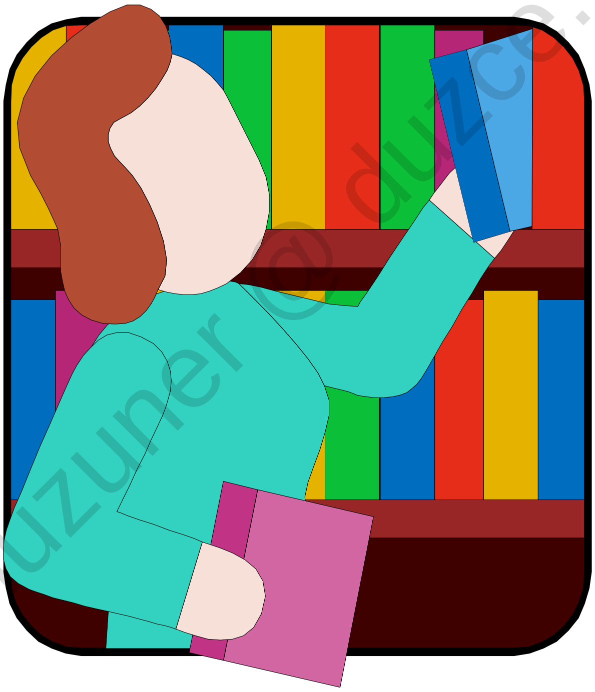


Language & font

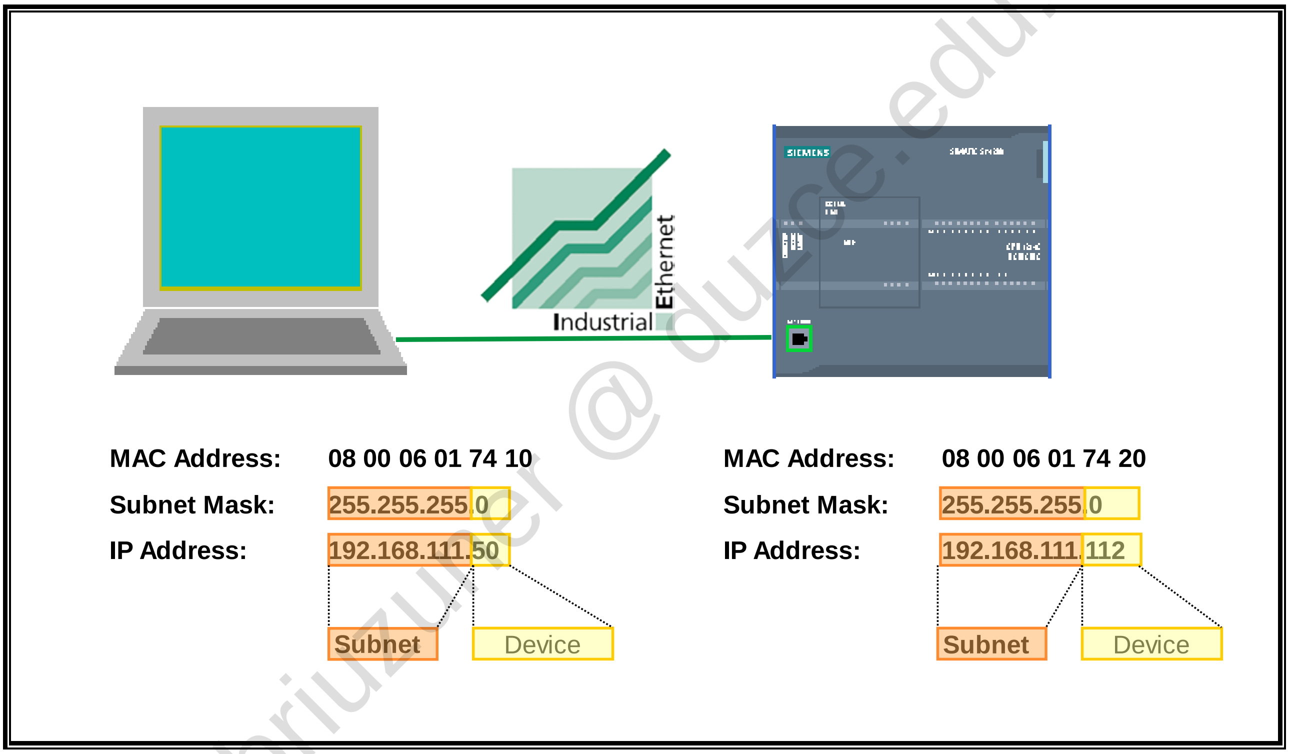
Runtime language and font selection

Order	Enable	Language	Fixed font 1	Default font
0	☑	English (Great Britain)	Tahoma	Tahoma, 11...
1	☑	English (USA)	Tahoma	Tahoma, 11px
2	☑	German (Germany)	Tahoma	Tahoma, 11px
	□	French (France)	Tahoma	Tahoma, 11px
	□	Italian (Italy)	Tahoma	Tahoma, 11px
	□	Spanish (Spain)	Tahoma	Tahoma, 11px

2.21. Additional Information



2.22. Industrial Ethernet: IP Address and Subnet Mask



MAC Address: 08 00 06 01 74 10
Subnet Mask: 255.255.255.0
IP Address: 192.168.111.50
Subnet
Device

MAC Address: 08 00 06 01 74 20
Subnet Mask: 255.255.255.0
IP Address: 192.168.111.112
Subnet
Device

Internet Protocol

The **I**nternet **P**rotocol (**IP**) is the basis for all TCP/IP networks. It creates the so-called datagrams (data packets specially tailored to the Internet protocol) and handles their transport within the local subnet or their "routing" (forwarding) to other subnets.

IP Addresses

IP addresses are not assigned to a specific computer, but rather to the network interfaces of the computer. A computer with several network connections (for example routers) must therefore be assigned an IP address for each connection.

IP addresses consist of 4 bytes. With the dot notation, each byte of the IP address is expressed by a decimal number between 0 and 255. The four decimal numbers are separated by dots (see picture).

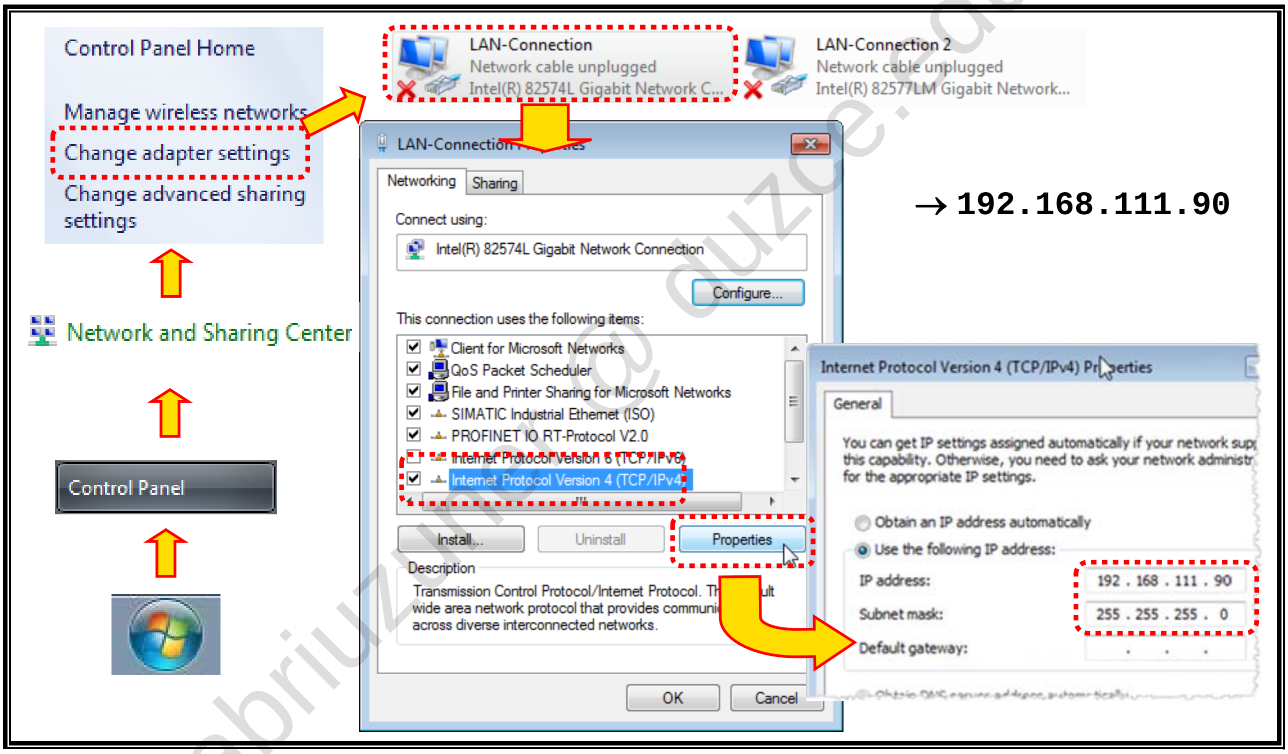
MAC Address

Every Ethernet interface is assigned a fixed address by the manufacturer that is unique worldwide. This address is referred to as the hardware or MAC address (**M**edia **A**ccess **C**ontrol). It is stored on the network card and uniquely identifies the Ethernet interface in a local network. Cooperation among the manufacturers ensures that the address is unique worldwide.

Subnet Mask

The subnet mask separates the IP address into network and device (computer) address.

2.23. Online Access: Assigning an IP Address for the PG


Control Panel Home
Manage wireless networks
Change adapter settings
Change advanced sharing settings
Network and Sharing Center
Control Panel
LAN-Connection
Network cable unplugged
Intel(R) 82574L Gigabit Network C...
LAN-Connection 2
Network cable unplugged
Intel(R) 82577LM Gigabit Network...
LAN-Connection Properties
Networking
Sharing
Connect using:
Intel(R) 82574L Gigabit Network Connection
Configure...
This connection uses the following items:
☑ Client for Microsoft Networks
☑ QoS Packet Scheduler
☑ File and Printer Sharing for Microsoft Networks
☑ SIMATIC Industrial Ethernet (ISO)
☑ PROFINET IO RT-Protocol V2.0
□ Internet Protocol Version 6 (TCP/IPv6)
☑ Internet Protocol Version 4 (TCP/IPv4)
Install...
Uninstall
Properties
Description
OK
Cancel
Internet Protocol Version 4 (TCP/IPv4) Properties
General
○ Obtain an IP address automatically
● Use the following IP address:
IP address:
Subnet mask:
Default gateway:
Obtain DNS server address automatically

→ 192.168.111.90

Transmission Control Protocol/Internet Protocol. Th... wide area network protocol that provides communi... across diverse interconnected networks.

You can get IP settings assigned automatically if your network sup... this capability. Otherwise, you need to ask your network administr... for the appropriate IP settings.

IP address: 192 . 168 . 111 . 90

Subnet mask: 255 . 255 . 255 . 0

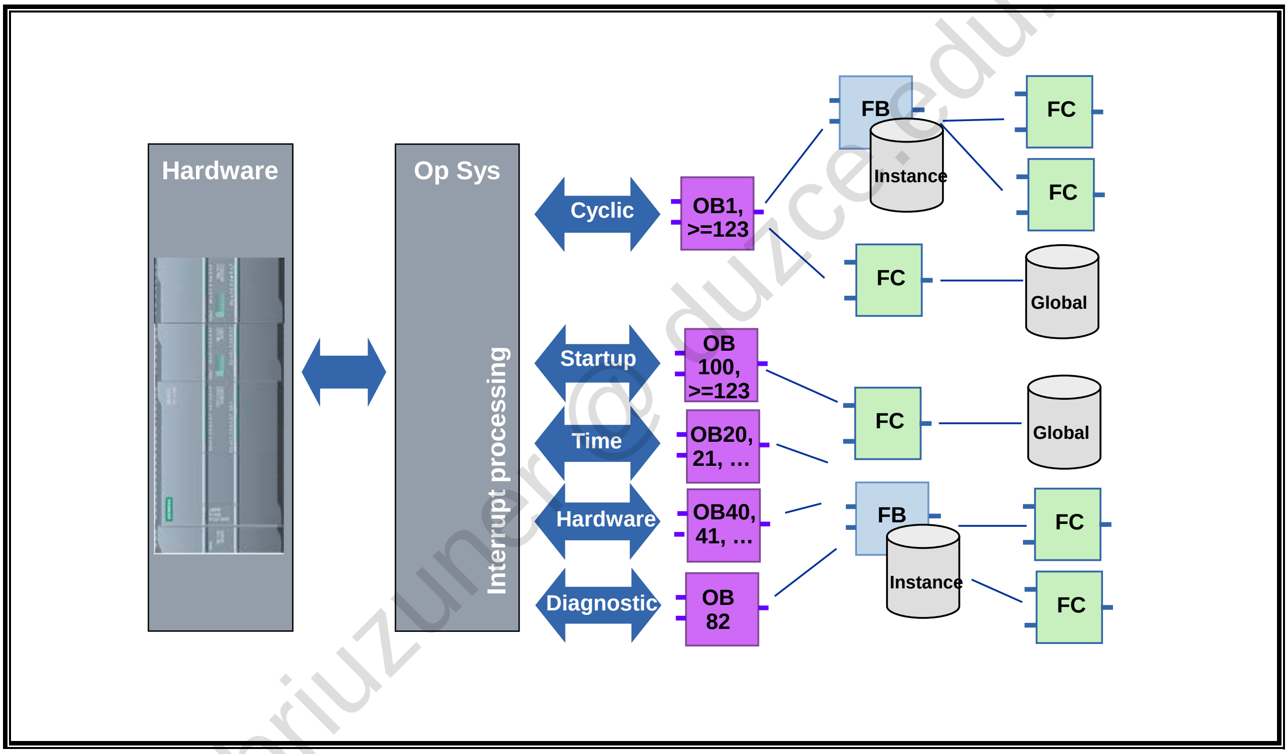
Default gateway: . . .

IP Address of the Programming Device

You can set the IP address of the PG as shown in the picture.

If an online connection between the programming device and the CPU is to be established, the same subnet mask must be assigned to the two devices. The assigned IP addresses have to be located in the same subnet.

2.24. OB – Organization Blocks



OBs

Organization blocks form the interface between the user program and the CPU’s operating system.

Organization blocks are called exclusively by the operating system. There are various start events (time interrupt, hardware interrupt, ...see picture).

Startup Program

After a restart, a startup program is executed. In the startup OBs you can, for example, carry out a pre-assignment of communication connections.

Cyclic Program Execution

The program stored in the cyclic OB is executed cyclically, that is, after it is executed completely it is executed again. With this cyclic program execution, the reaction time results from the execution time for the CPU’s operating system and the sum of the command runtimes of all executed instructions. The reaction time, that is, how fast an output can be switched in relation to an input signal, amounts to a minimum of one time and a maximum of two times the cycle time.

Periodic Program Execution

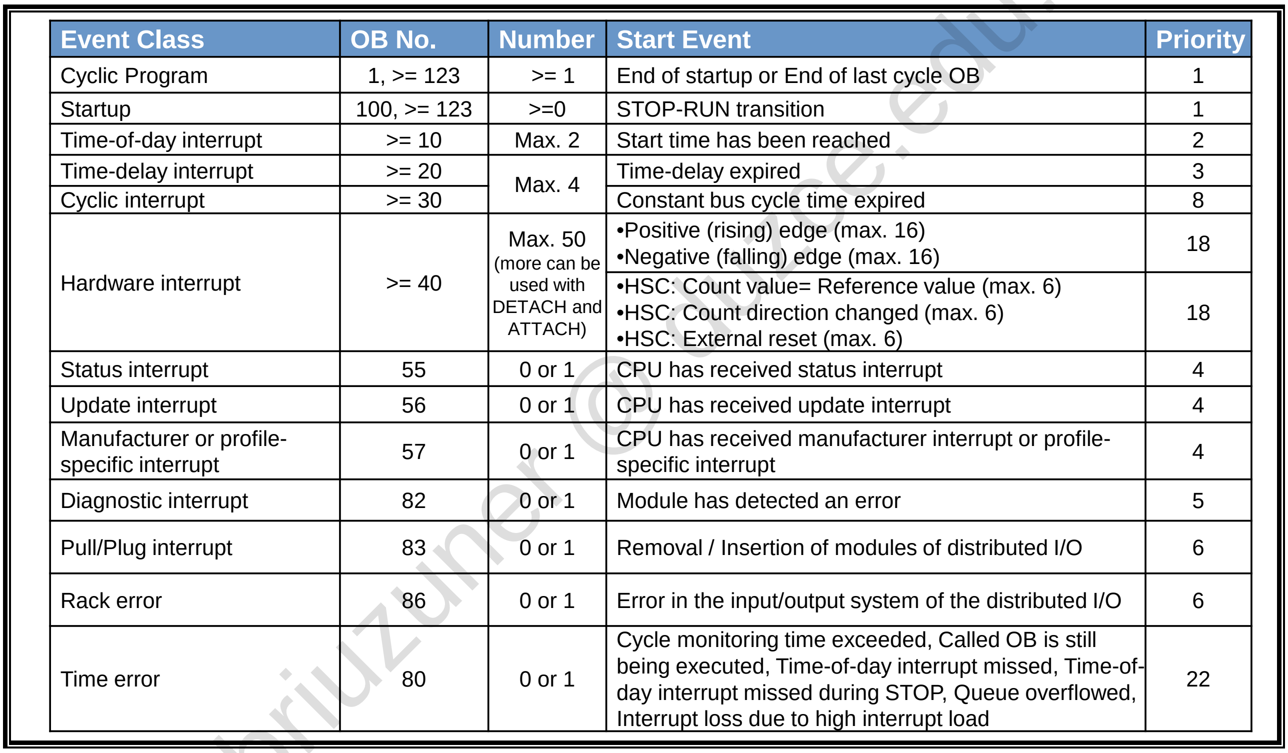
This makes it possible to interrupt the cyclic program execution at fixed intervals. With the cyclic interrupts, an organization block (for example OB235) is executed after an adjustable time base (for example, every 100ms) has expired. In these blocks, closed-loop control blocks with their sampling time, for example, are called.

Event-driven Program Execution

In order to be able to react quickly to a process event, the hardware interrupt can be used. After an event occurs, the cycle is immediately interrupted and an interrupt program is executed.

With time-delay interrupts, a freely definable event can be reacted to with a time-delay; with an error OB, the user can influence the behavior of the controller in case there is an error.

2.24.1. Events which Start an OB



Event Class	OB No.	Number	Start Event	Priority
Cyclic Program	1, >= 123	>= 1	End of startup or End of last cycle OB	1
Startup	100, >= 123	>=0	STOP-RUN transition	1
Time-of-day interrupt	>= 10	Max. 2	Start time has been reached	2
Time-delay interrupt	>= 20	Max. 4	Time-delay expired	3
Cyclic interrupt	>= 30		Constant bus cycle time expired	8
Hardware interrupt	>= 40	Max. 50 (more can be used with DETACH and ATTACH)	•Positive (rising) edge (max. 16) •Negative (falling) edge (max. 16)	18
			•HSC: Count value= Reference value (max. 6) •HSC: Count direction changed (max. 6) •HSC: External reset (max. 6)	18
Status interrupt	55	0 or 1	CPU has received status interrupt	4
Update interrupt	56	0 or 1	CPU has received update interrupt	4
Manufacturer or profile-specific interrupt	57	0 or 1	CPU has received manufacturer interrupt or profile-specific interrupt	4
Diagnostic interrupt	82	0 or 1	Module has detected an error	5
Pull/Plug interrupt	83	0 or 1	Removal / Insertion of modules of distributed I/O	6
Rack error	86	0 or 1	Error in the input/output system of the distributed I/O	6
Time error	80	0 or 1	Cycle monitoring time exceeded, Called OB is still being executed, Time-of-day interrupt missed, Time-of-day interrupt missed during STOP, Queue overflowed, Interrupt loss due to high interrupt load	22

Events

The operating system of S7-1200-CPUs is based on events. There are two types of events:

- Events which can start an OB
- Events which cannot start an OB

An event which can start an OB triggers the following reaction:

- If you have assigned an OB to the event, this OB is called.
  If it is currently not possible to call this OB, the event is entered into a queue according to its priority.
- If you have not assigned an OB to the event, the predefined default system reaction is carried out.

An event which cannot start an OB triggers the predefined default system reaction for the associated event class.

The user program cycle is therefore based on events, the assignment of OBs to those events, and on the code which is either contained in the OB or called in the OB.

The table above gives an overview of the events which can start an OB. It is sorted according to OB priority. 1 is the lowest priority.

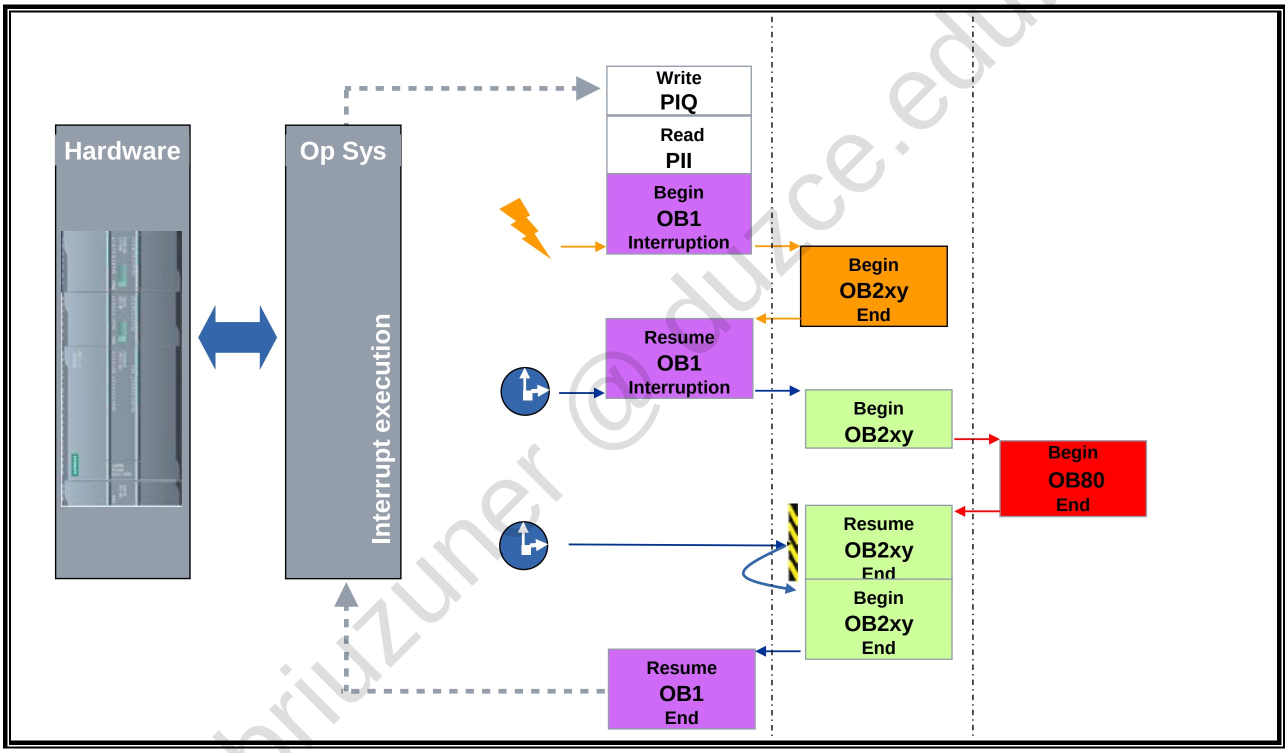
OB Priority

With the exception of the startup and cyclic OBs, all OBs have a priority which can be changed between 2 and 24. In all, the priorities are staggered from 1 - 27, whereby 1 is the lowest priority and 27 is the highest priority.

2.24.2. Events which Cannot Start an OB

Event Class	Event	Event Priority	System Reaction
Insert / Remove central modules	Insert / Remove a module	21	STOP
I/O access error during process image update	I/O access error during process image update	22	Ignore
Programming error	Programming error in a block for which you use the system reactions provided by the operating system (Note: If you have activated the local error handling, the error handling routine programmed in the block takes effect.)	23	RUN
I/O access error	I/O access error in a block for which you use the system reactions provided by the operating system (Note: If you have activated the local error handling, the error handling routine programmed in the block takes effect.)	24	RUN
Maximum cycle time exceeded twice	Cycle monitoring time was exceeded twice	27	STOP

2.24.3. Interrupting the Cyclic Program



Interruption of OBs

Every OB program execution can be interrupted between instructions by an event (OB) with a higher priority if this is set in the properties of the CPU. (CPU > Properties > Startup > OBs should be interruptible).

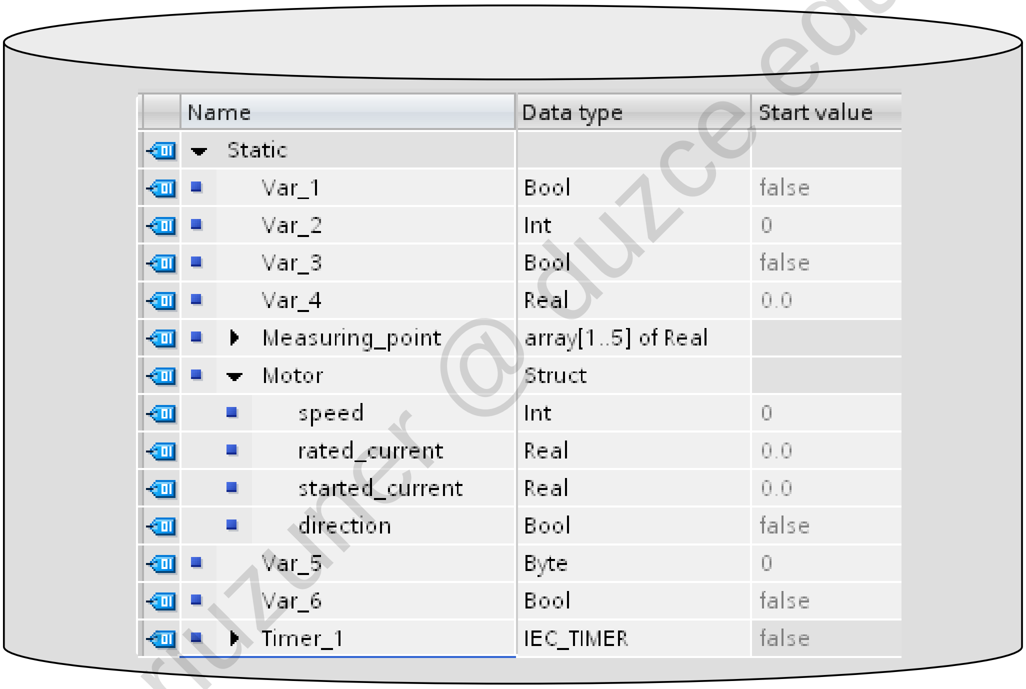
Queue

If the OBs (with the exception of cyclic OBs) are not parameterized as interruptible or have the same or a lower priority, then this event is entered into a queue according to its priority. The start events of a queue are processed at a later point in time in the order they occurred.

Interruption of the Cycle Program

Cyclic OBs have the lowest priority and are therefore interrupted when there are call requests from all other OBs, even if the OBs are not parameterized as interruptible in the CPU properties.

2.25. DB – Data Block



Name	Data type	Start value
▼ Static		
Var_1	Bool	false
Var_2	Int	0
Var_3	Bool	false
Var_4	Real	0.0
▶ Measuring_point	array[1..5] of Real	
▼ Motor	Struct	
speed	Int	0
rated_current	Real	0.0
started_current	Real	0.0
direction	Bool	false
Var_5	Byte	0
Var_6	Bool	false
▶ Timer_1	IEC_TIMER	false

Overview

Data blocks are used to store user data. Data blocks occupy memory space in the user memory of the CPU. Variable data (e.g. numeric values) with which the user program works is located in the data blocks.

The user program can access the data of a data block via bit, byte, word or double-word operations. Access can occur symbolically or absolutely.

Applications

Data blocks can be used in different ways by the user depending on their contents. Differentiation is made between:

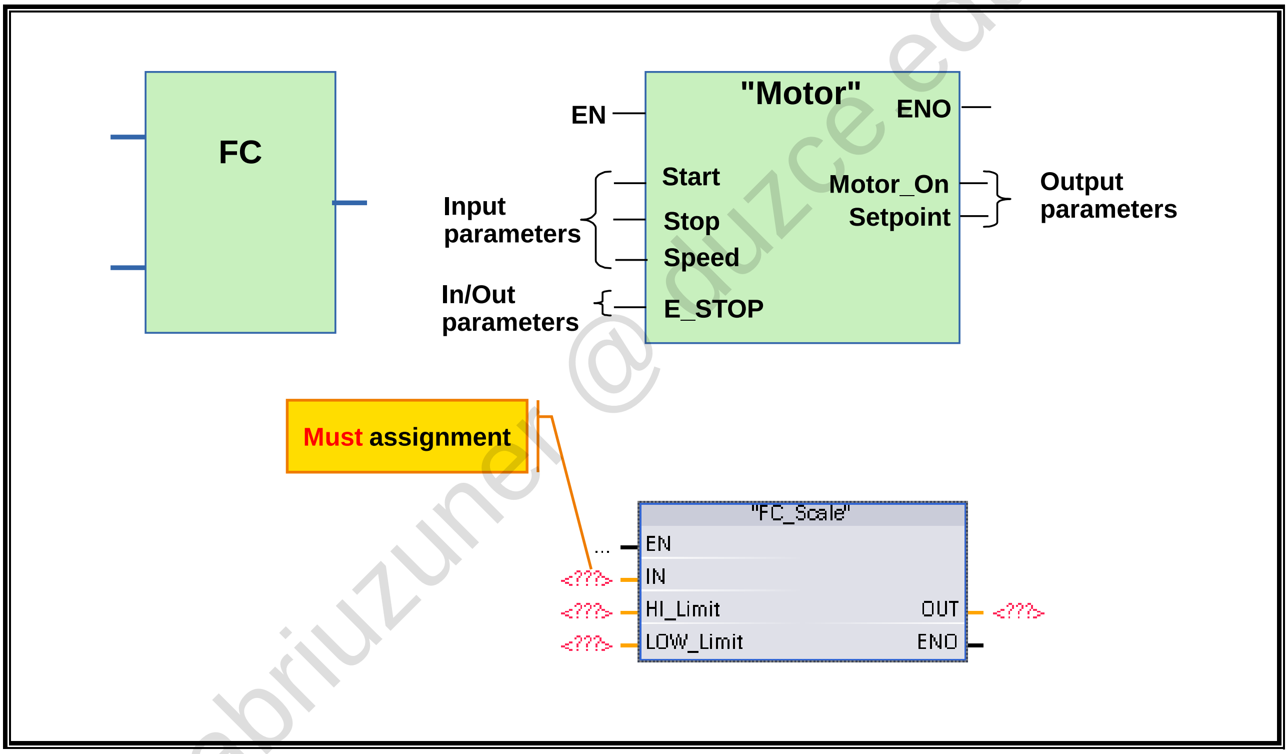
- Global data blocks: they contain information which can be accessed by all logic (code) blocks in the user program.
- Instance data blocks: they are always assigned to an FB. The data of this DB should only be processed by the associated FB.

Creating DBs

Global DBs are created either via the Program Editor or according to a previously created "PLC data type".

Instance data blocks are generated when a function block is called.

2.26. FC – Function



Overview

Functions represent parameter-assignable blocks without memory. In STEP 7 they can have as many input parameters, output parameters and in/out parameters as are required.

Functions have no memory; no separate, permanent data area for storing results exists. Temporary results that occur during function execution can only be stored in the temporary variables of the respective local data stack.

Application

Functions are primarily used when function values are to be returned to the calling blocks. (for example, mathematical functions, single control with binary logic operation).

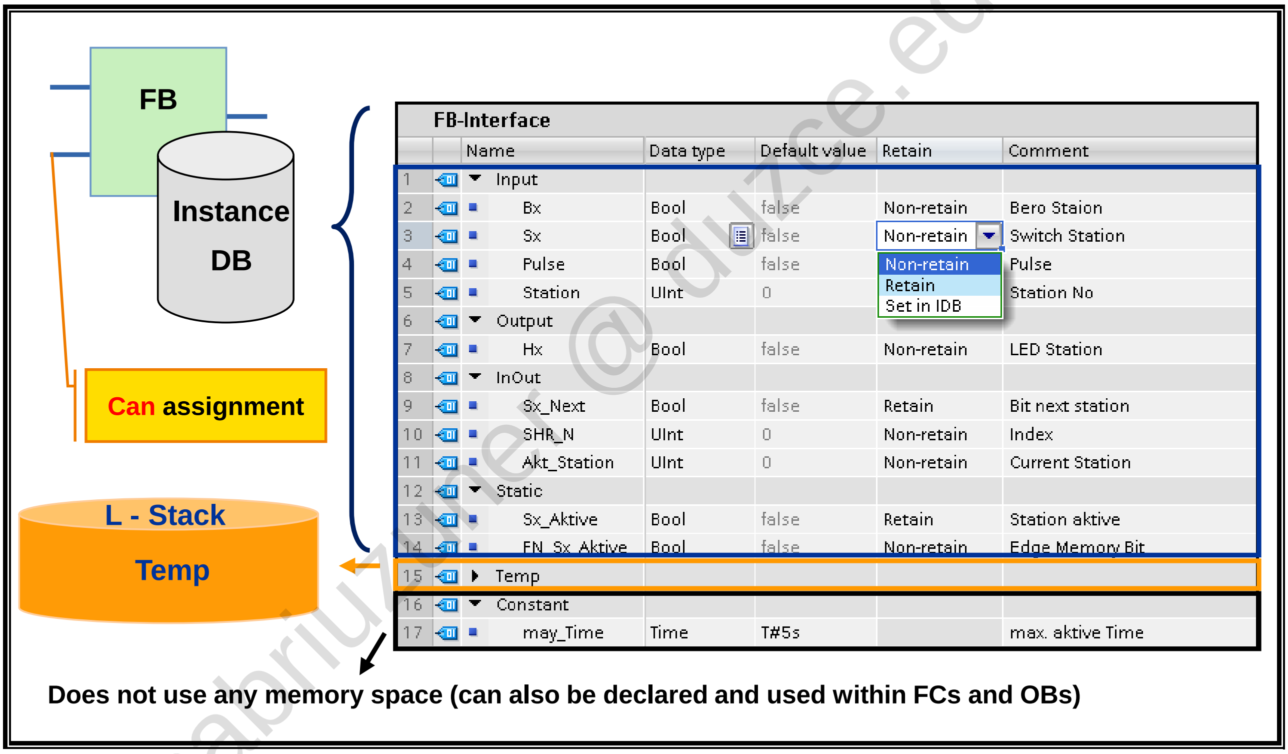
IEC-61131-Conforming Functions

1. Functions can have as many input parameters as is required. They can, however, only return one result to the output parameter RET_VAL.
2. Global operands can neither be read nor written within functions.
3. No instances of function blocks can be called within functions.
4. Because of the missing "memory", the returned result of a norm-conforming function is solely dependent on the values of the input parameter. For identical values of the input parameter, a function always returns the identical result.

It is therefore up to the programming person to create norm-conforming functions or to individually do the block programming and block structuring in STEP 7.

☞ If 2 to 4 is fulfilled, then this is recognized in the Properties under the attribute “Block can be used as know-how protected library element” after compilation and the block can be used in every other project.

2.27. FB – Function Block



	Name	Data type	Default value	Retain	Comment
1	Input				
2	Bx	Bool	false	Non-retain	Bero Staion
3	Sx	Bool	false	Non-retain	Switch Station
4	Pulse	Bool	false		Pulse
5	Station	UInt	0		Station No
6	Output				
7	Hx	Bool	false	Non-retain	LED Station
8	InOut				
9	Sx_Next	Bool	false	Retain	Bit next station
10	SHR_N	UInt	0	Non-retain	Index
11	Akt_Station	UInt	0	Non-retain	Current Station
12	Static				
13	Sx_Aktive	Bool	false	Retain	Station aktive
14	FN_Sx_Aktive	Bool	false	Non-retain	Edge Memory Bit
15	Temp				
16	Constant				
17	may_Time	Time	T#5s		max. aktive Time

Does not use any memory space (can also be declared and used within FCs and OBs)

Overview

Function blocks (FB) are blocks of the user program and represent logic blocks with memory according to the IEC Standard 61131-3. They can be called by all blocks.

Function blocks can each have as many input, output and in/out parameters as well as static and temporary variables as are required.

Unlike FCs, FBs are instantiated, that is, an FB is assigned its own data area in which the FB can "remember" process states from call to call, for example. In the simplest form, this private data area is its own DB (Instance DB).

"Memory"

You have the opportunity to declare static variables in the declaration section of a function block. The function block can remember information from call to call in these variables.

The ability of a function block to remember information over several calls is the essential difference to functions.

Application

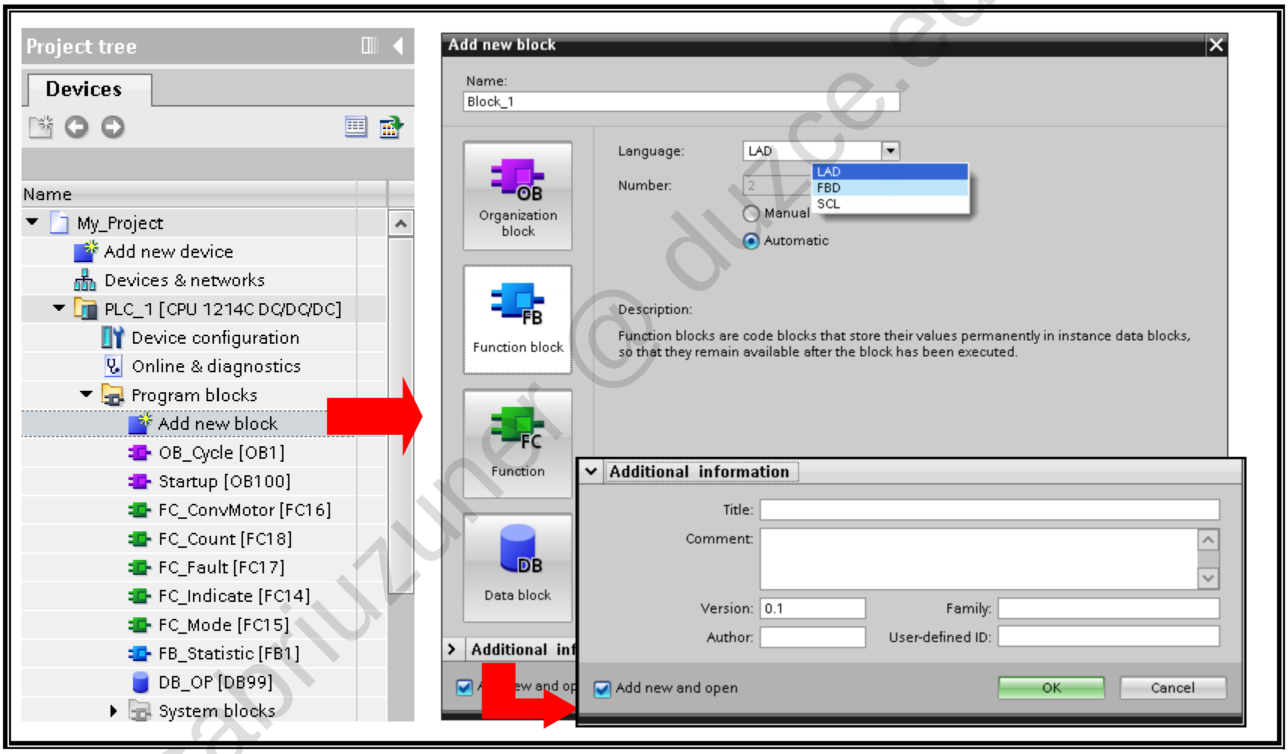
With the help of this "memory", a function block can implement counter and timer functions or control process units, such as processing stations, drives, boilers etc., for example.

In particular, function blocks are well suited for controlling all those process units whose performance depends not only on outside influences but also on internal states, such as processing step, speed, temperature etc.

When controlling such units, the internal status data of the process unit are then copied to the static variables of the function block.

☞ Within an FB, if no global variables are used and only multiple instances are used for FB calls, then this is recognized in the Properties under the attribute “Block can be used as know-how protected library element” after compilation and the block can be used in every other project.

2.28. Adding a New Block



Name: Block_1

Language: LAD

Number: 2

○ Manual

● Automatic

Description:

Function blocks are code blocks that store their values permanently in instance data blocks, so that they remain available after the block has been executed.

Additional information

Title:

Comment:

Version: 0.1

Family:

Author:

User-defined ID:

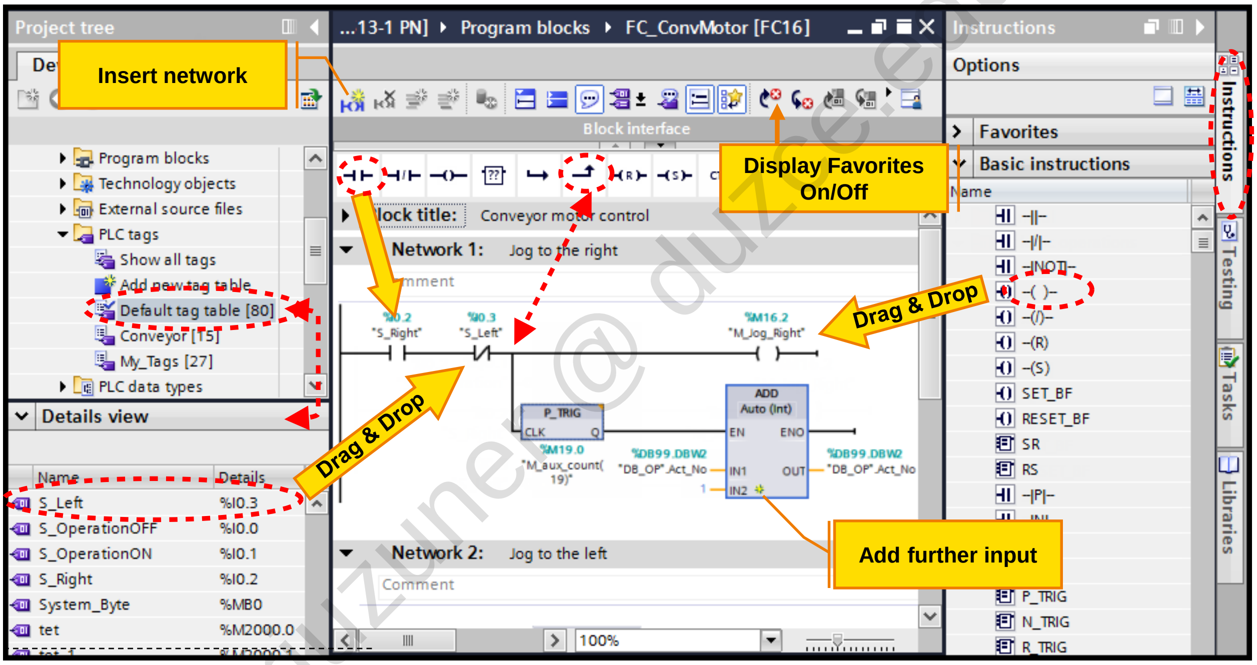
☑ Add new and open

Inserting a Block

A new block is created as shown in the picture. When you create a block, the type of block (OB, FB, FC or DB), the programming language, the symbolic name and number, among other things, must be defined. The block numbers can also be assigned automatically or manually.

In "Additional information", the block can be documented in more detail, among other things, with a Version number and an Author.

2.29. Block Programming



Block Programming

The instructions within a block can be programmed as follows:

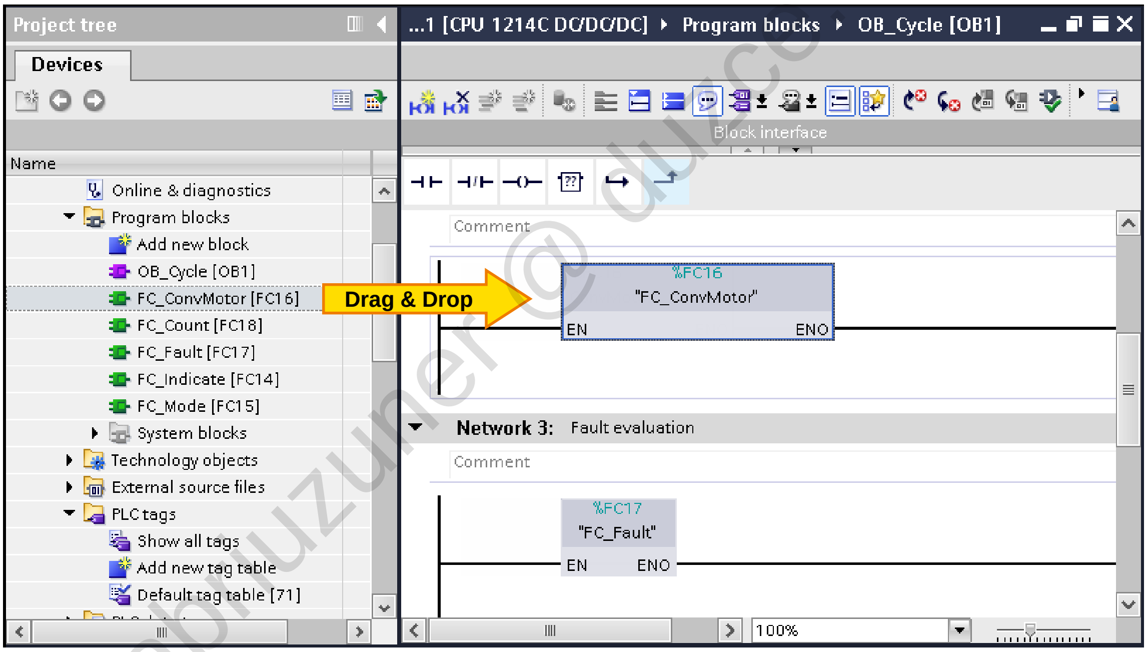
- using drag & drop from the Favorites or the Instructions catalog to anywhere in the program
- by first selecting the location in the program and then double-clicking on the desired instruction in the Favorites or the Instructions catalog

Operands can be entered with an absolute or a symbolic address. If a tag table of a data block is highlighted (not opened!) in the Project tree, tags (variables) can also be pulled from the Details view using drag & drop to the appropriate location in the program.

Favorites

Frequently used LAD (FBD) elements are available in the symbol bar which can be expanded individually using drag & drop from the Instructions catalog.

2.30. Block Calls

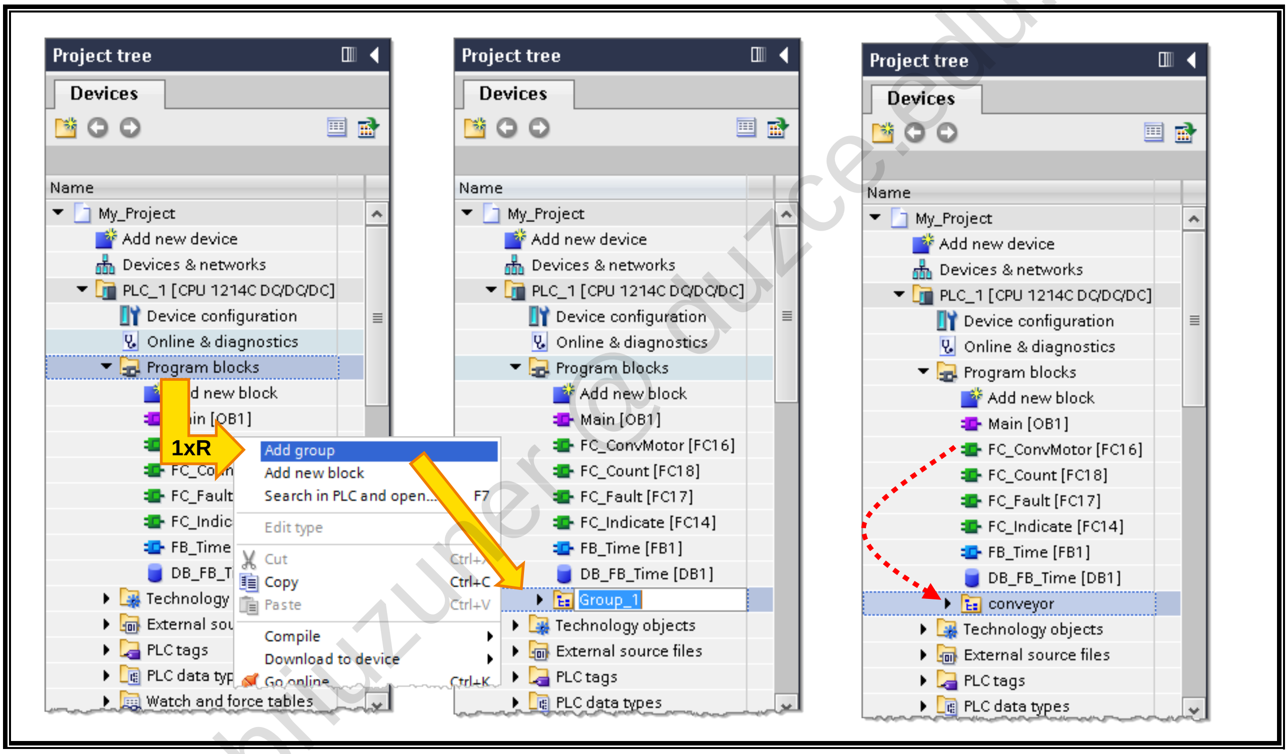


Block Calls

If one block calls another block, the instructions of the called block are executed. Only when the execution of the called block is completed, is the execution of the calling block taken up again and continues with the instruction that follows the block call.

The block call can be programmed using drag & drop or copy & paste.

2.31. Block Groups

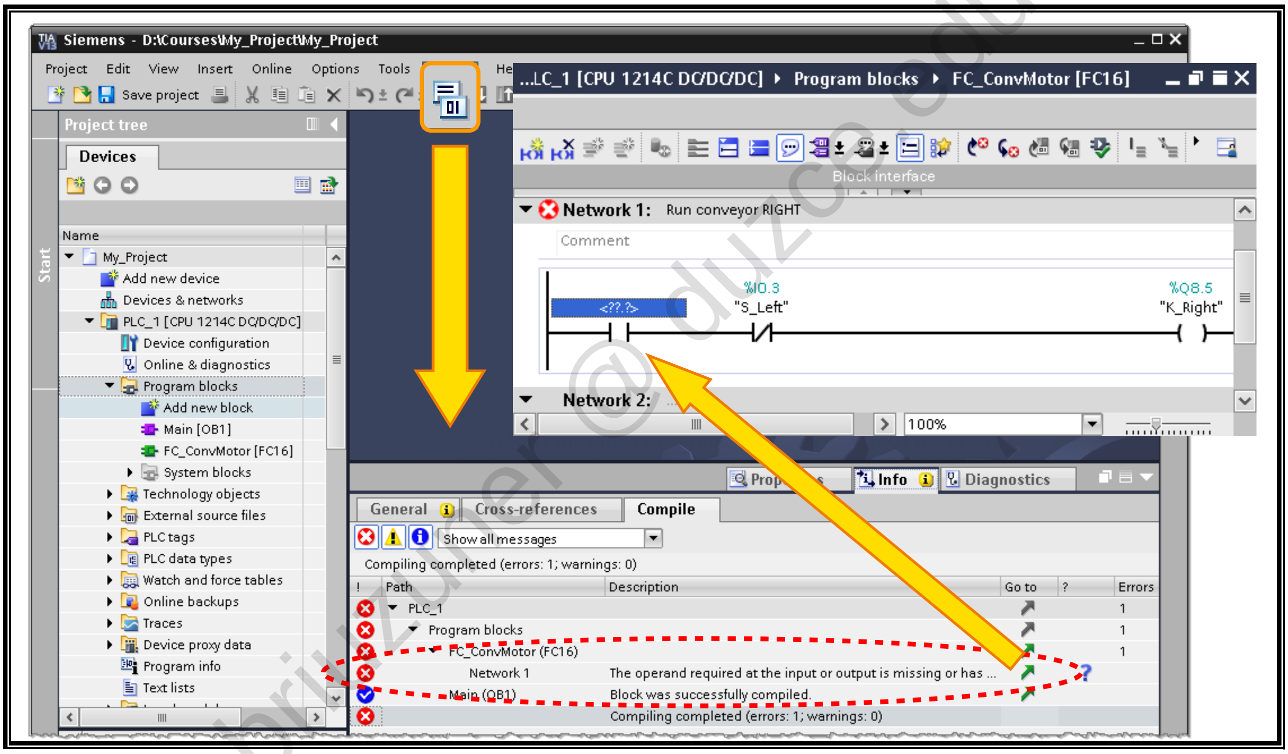


Block Groups

To achieve more clarity, large programs with many blocks can be divided into different block groups. The groupings can, for example, be related to the structure of the system to be controlled. Even if the blocks are managed in different groups, each block must have a unique symbolic name. Regardless of the groupings, the sum of all blocks represents the user program of the controller.

The blocks can simply be shifted between the block groups using drag & drop.

2.32. Compiling a Block



Compiling completed (errors: 1; warnings: 0)

!	Path	Description	Go to	?	Errors
☒	PLC_1		↗		1
☒	Program blocks		↗		1
☒	FC_ConvMotor (FC16)		↗		1
☒	Network 1	The operand required at the input or output is missing or has ...	↗	?	
✓	Main (OB1)	Block was successfully compiled.	↗		
☒		Compiling completed (errors: 1; warnings: 0)			

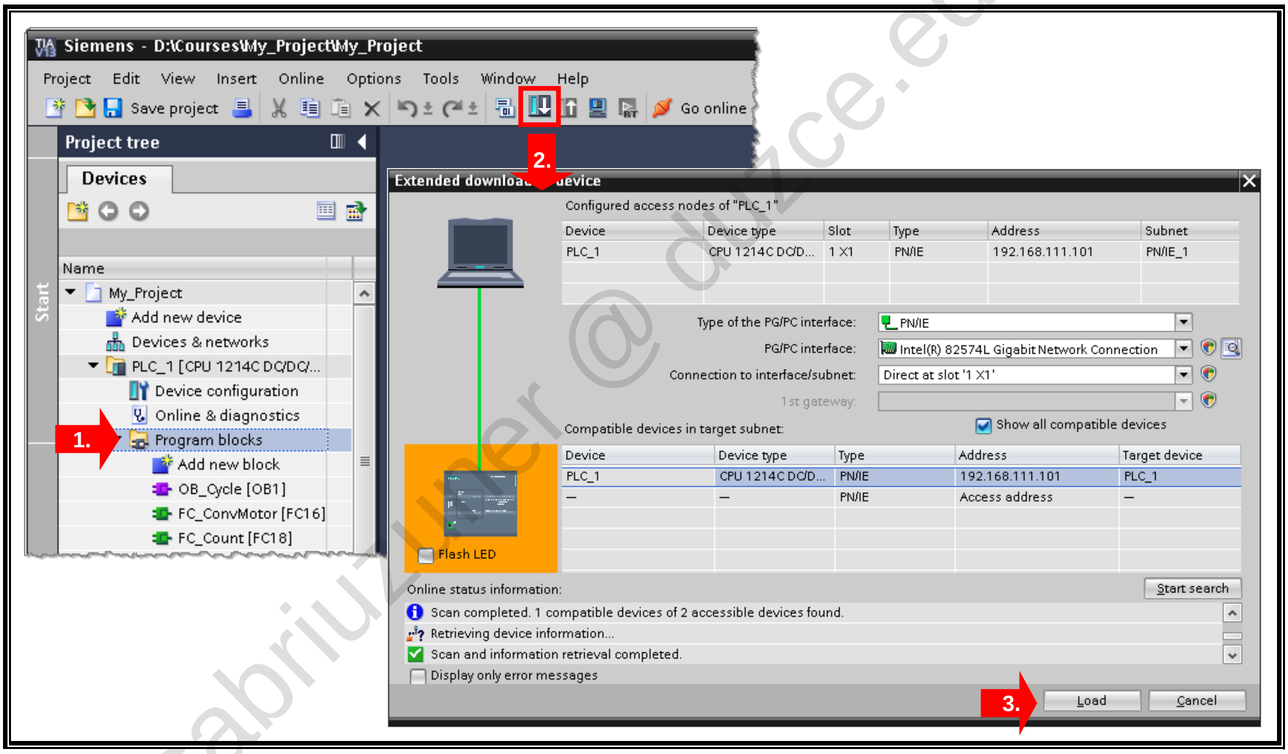
Compiling a Block

With the Compile icon, all changes of whatever is selected (highlighted) in the Project tree are compiled (in the example shown, all changes of the entire program are compiled). The changes of individual blocks (select relevant block), the changes of the entire program or the changes of the entire station with software and hardware ("Station" is selected) can be compiled.

To completely compile the blocks or the station, the context menu (right click) of the folder "Program blocks" or the station is opened and the function "Software (rebuild all blocks)" or the function "Hardware (rebuild all)" is selected in the menu "Compile".

In the Inspector window "Info -> Compile", the status of the compilation is displayed. If errors occurred during compilation, you can jump directly from the error entry to the error location by double-clicking.

2.33. Downloading Blocks into the CPU



Configured access nodes of "PLC_1"

Device	Device type	Slot	Type	Address	Subnet
PLC_1	CPU 1214C DC/D...	1 X1	PN/IE	192.168.111.101	PN/IE_1

Type of the PG/PC interface: PN/IE

PG/PC interface: Intel(R) 82574L Gigabit Network Connection

Connection to interface/subnet: Direct at slot '1 X1'

1st gateway:

Compatible devices in target subnet: ☑ Show all compatible devices

Device	Device type	Type	Address	Target device
PLC_1	CPU 1214C DC/D...	PN/IE	192.168.111.101	PLC_1
–	–	PN/IE	Access address	–

Online status information:

Scan completed. 1 compatible devices of 2 accessible devices found.

Retrieving device information....

Scan and information retrieval completed.

□ Display only error messages

Downloading into the CPU

The project data which is downloaded into the devices is divided into hardware and software project data.

Hardware project data results from configuring the hardware, networks, and connections. The first time you download the data using the icon "Download to device" the hardware project data is completely loaded. In subsequent downloads, only configuration changes are downloaded.

In order to once more download the entire configuration, you open the context menu of the station and select the function "Download to device > Hardware configuration".

Software project data involves the blocks of the user program. The first time you download, the software project data is completely loaded. In subsequent downloads, either by means of the icon "Download to device" or via the context menu, only changes are downloaded.

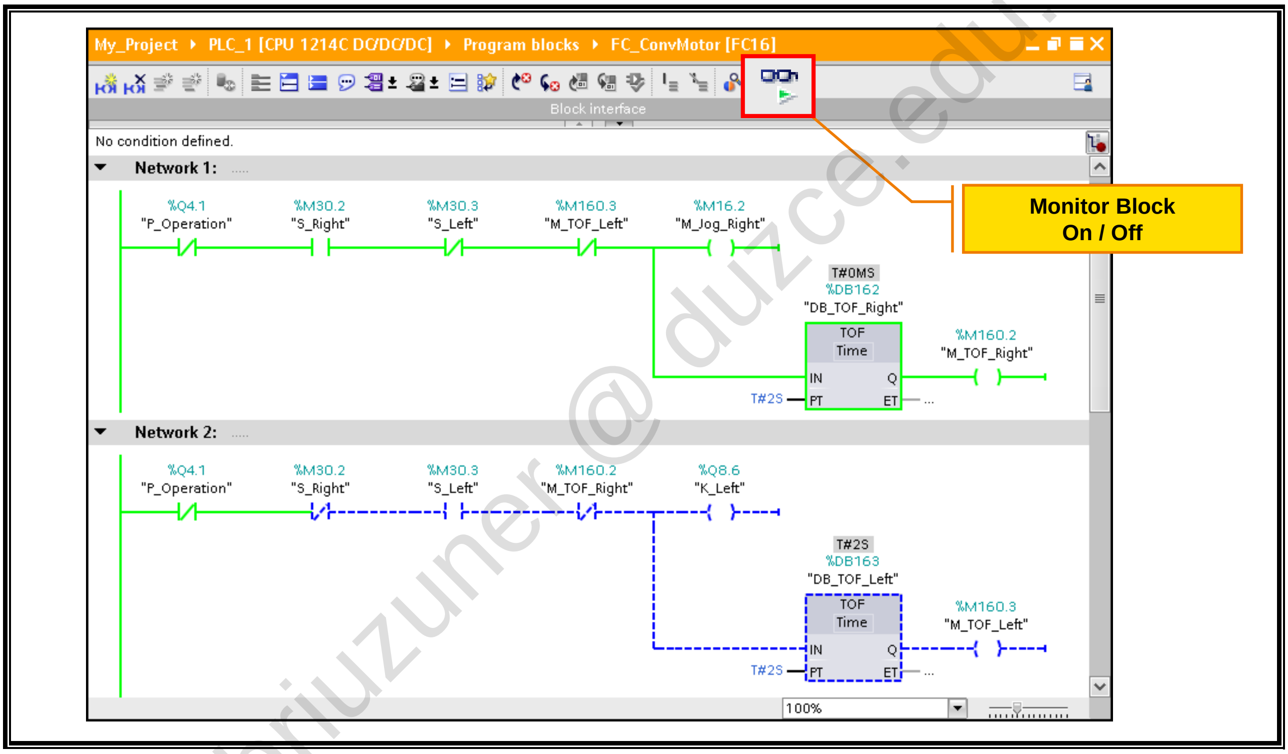
What is to be downloaded?
Selection via: Context menu of device > Download to device

– Hardware and Software (only changes):	Download all new and modified software project data as well as the new and modified hardware configuration
– Hardware configuration:	Download the entire hardware configuration
– Software (only changes):	Download all new and modified software project data

☞ If the changes to the objects to be downloaded were not compiled before the loading, then the compilation is automatically carried out before the download.

☞ The download is only carried out if the compilation is error-free.

2.34. Monitoring a Block



Monitoring Blocks

The test function *Monitor block* is used to be able to follow the program execution within a block. The statuses or contents of the operands used in the block at the time of program execution are displayed on the monitor.

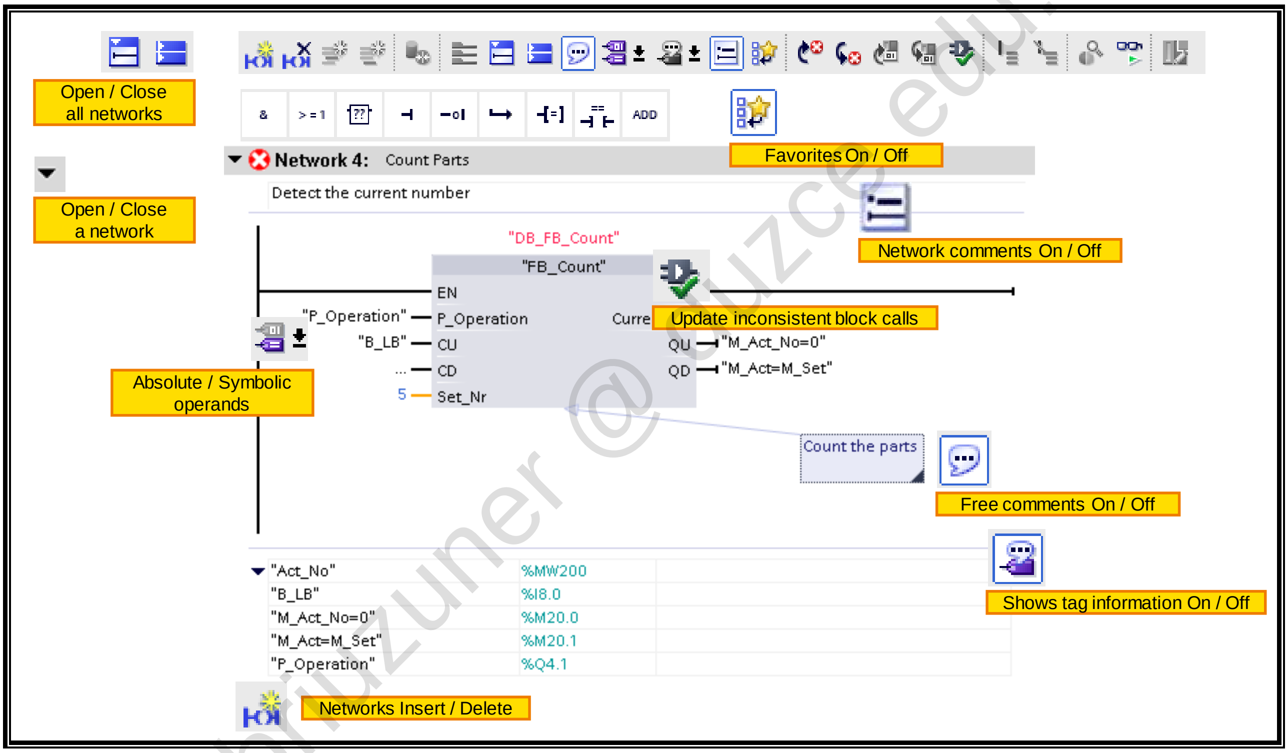
Monitoring

Blocks can only be monitored if an online connection to the CPU exists. Furthermore, the offline block must be identical to the online block. If the offline opened block does not match the block stored online in the CPU, either the online stored block must be opened, or the offline opened block must be downloaded into the CPU before you can monitor and then you can monitor the block.

Examples:

Status fulfilled	→	"Element is represented with a green color"
Status not fulfilled	→	"Element is represented with a blue color"

2.35. Block Networks



"Act_No"	%MW200	
"B_LB"	%I8.0	
"M_Act_No=0"	%M20.0	
"M_Act=M_Set"	%M20.1	
"P_Operation"	%Q4.1	

Open / Close all networks

Open / Close a network

Absolute / Symbolic operands

Favorites On / Off

Network comments On / Off

Update inconsistent block calls

Free comments On / Off

Shows tag information On / Off

Networks Insert / Delete

Contents

6

6. Introduction to PROFINET

6.1. Objectives

At the end of the chapter the participant will ...

… be familiar with the PROFINET IO device types

… understand the term "Switched Ethernet"

… be able to explain the principle of PROFINET IO device addressing

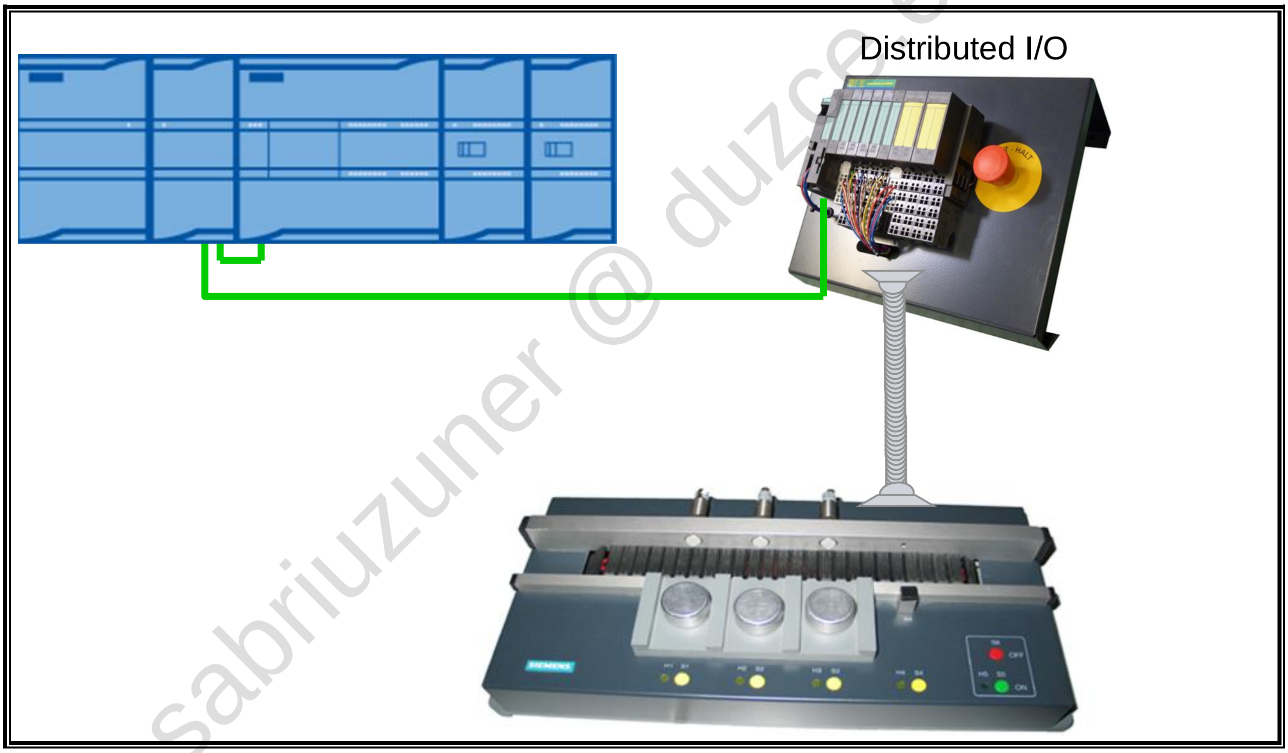
… be familiar with the procedure and necessary editors for configuring a PROFINET IO device

… be able to commission a distributed I/O (peripheral) with PROFINET IO

Objectives

In this chapter, the basics of PROFINET are handled. This includes not only the addressing procedure and the device types, but also the configuration of distributed field devices in the TIA-Portal.

6.2. Task Description: Replacing a Central I/O Module with Distributed I/O



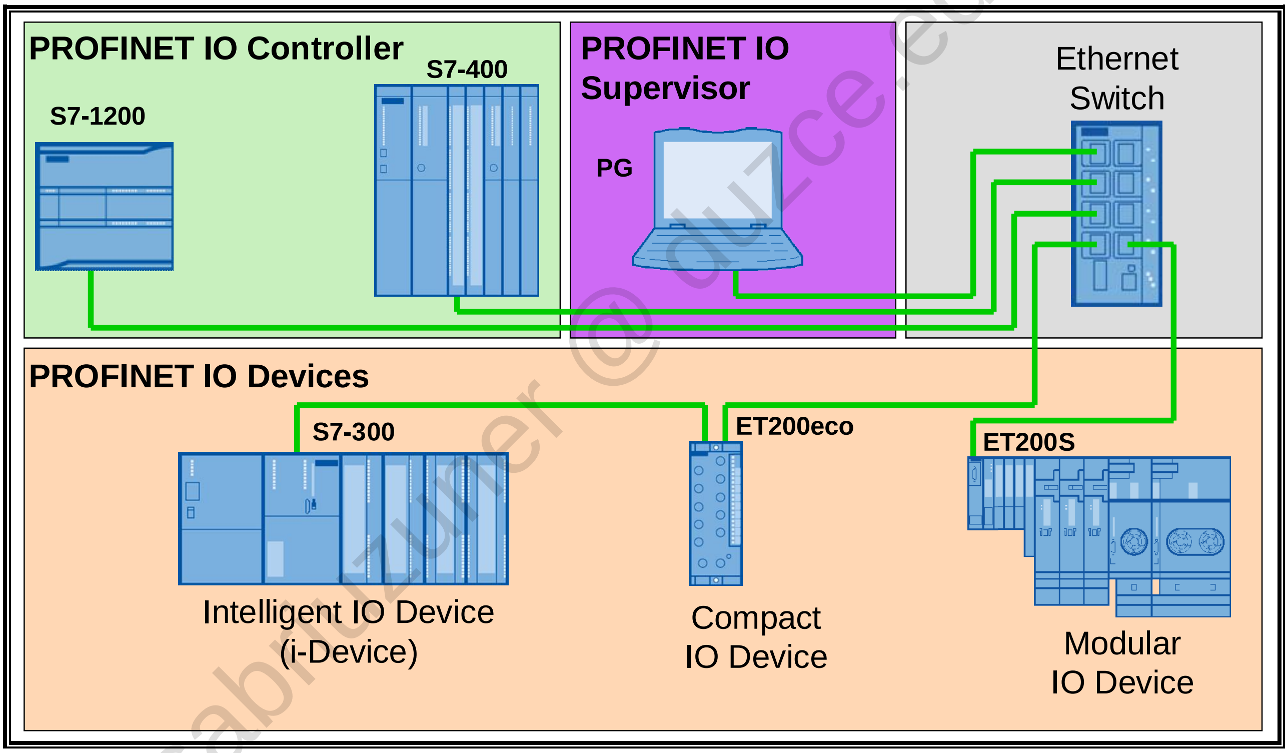
Situation Up Until Now

The conveyor is controlled through the 8DI/8DO module of the central rack.

Goal

You are to commission the PROFINET system for your training device in such a way that the conveyor model can be controlled via the ET 200S with the same functionality without having to change the S7 program.

6.3. PROFINET IO Device Types



PROFINET IO Controller

The IO controller (typically the PLC) establishes a logical connection to the connected IO devices after Power-On and subsequently parameterizes these (module parameters, address, etc.). (This corresponds to the function of a Class 1 Master in PROFIBUS).

PROFINET IO Device

An IO device is a distributed IO device that is connected via PROFINET IO (this corresponds to the function of a slave in PROFIBUS).

Differentiation is made for the following IO device types:

- Compact IO device: Fixed degree of expansion.
- Modular IO device: Variable degree of expansion; can be expanded or reduced as required.
- Intelligent IO device: A PLC is configured not as an IO controller but as an IO device and provides a higher-level controller with I/O data.

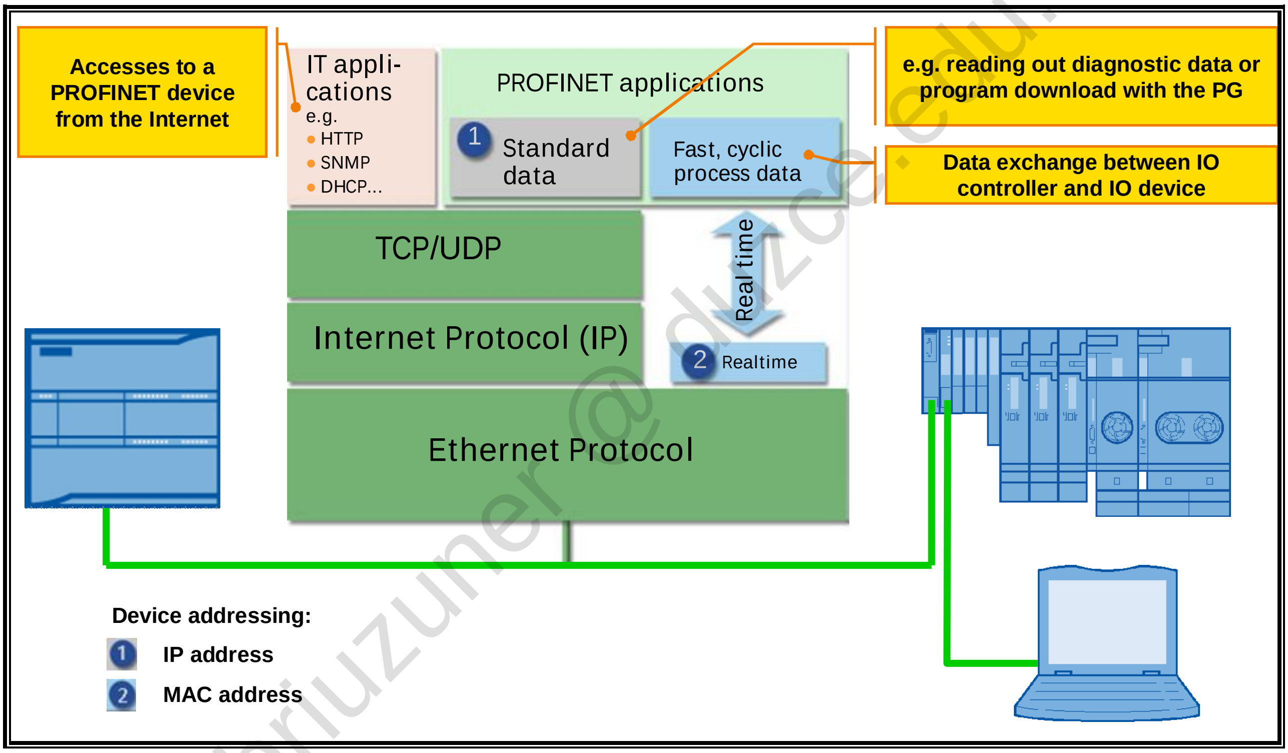
IO Supervisor

This can be a programming device (PG), personal computer (PC) or Human Machine Interface (HMI) for commissioning or diagnostic purposes. (This corresponds to a Class 2 Master in PROFIBUS).

Ethernet Switch

PROFINET is based on Ethernet. For that reason, switches are always used as "network distributors". Every node device is connected to a switch via a so-called "point-to-point" connection. This is also referred to as a **"Switched Ethernet"**. In most PROFINET devices, a 2 or multi-port switch is already integrated so that it is very easy to establish a line structure (comparable to PROFIBUS).

6.4. PROFINET Communications Model


Accesses to a PROFINET device from the Internet
IT appli-cations
e.g.
HTTP
SNMP
DHCP...
PROFINET applications
1 Standard data
Fast, cyclic process data
e.g. reading out diagnostic data or program download with the PG
Data exchange between IO controller and IO device
TCP/UDP
Real time
Internet Protocol (IP)
2 Realtime
Ethernet Protocol
Device addressing:
1 IP address
2 MAC address

The PROFINET Communications Model

The PROFINET concept is modularly designed so that you can choose the necessary functionality. Essentially the functionality differs in the type of data exchange, in order to satisfy the very stringent requirements for data transmission speed. The picture shows the correlation between the communication channels for PROFINET IO. Both communication channels can be used in parallel.

PROFINET IO Standard Data

Included in the standard data for PROFINET is the data which occurs, for example, when monitoring online states, when downloading a user program or when reading diagnostic data. This data is transmitted via the protocols User Datagram Protocol (UDP) and Internet Protocol (IP). Since Internet Protocol (IP) is used, the device requires an IP address.

Fast, Cyclic Process Data

The exchange of fast, cyclic data takes place via the so-called Real-Time channel. This builds directly on the Ethernet layer and is thus significantly faster since less protocol information has to be evaluated here. The addressing of the devices takes place via the MAC address which is fixed in every device by the factory.

IT Standards

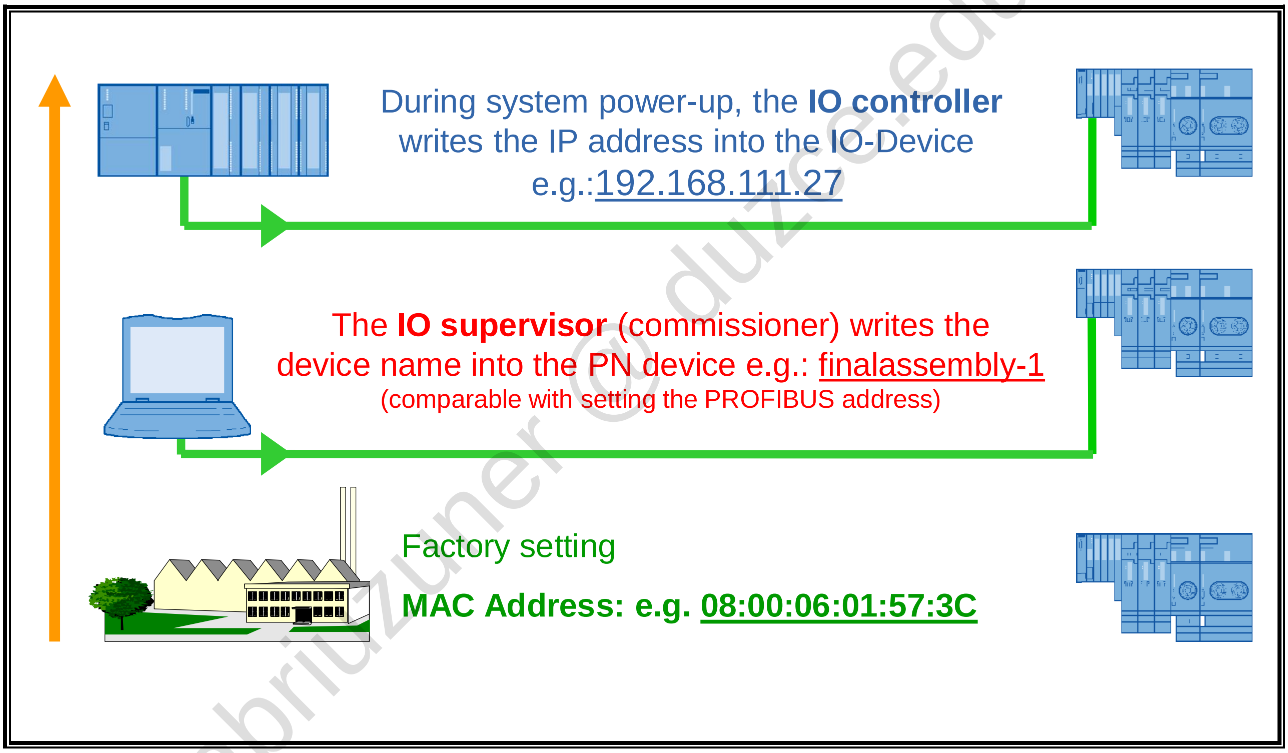
The design of PROFINET WEB Integration focuses on commissioning and diagnostics. Within these areas of application, WEB-based concepts can be used particularly efficiently. The WEB services describe mechanisms to integrate PROFINET devices in the Internet/Intranet world. The essential features are:

Access to a PROFINET device from the Internet or Intranet is done with standard protocols (for example, http). The data is transmitted in standard formats such as HTML or XML and can be presented with standard browsers such as Opera or Internet Explorer.

This worldwide accessibility makes it easy for the application manufacturer to support the user with commissioning. Access to the data is done via "Web Pages". Possible applications for WEB Integration are, among others,

- Test and commissioning,
- Overview of the device data (PROFINET IO),
- Device diagnostics and
- System and device documentation.

6.5. PROFINET Device Addressing



During system power-up, the **IO controller** writes the IP address into the IO-Device e.g.:192.168.111.27

The **IO supervisor** (commissioner) writes the device name into the PN device e.g.: finalassembly-1 (comparable with setting the PROFIBUS address)

Factory setting

MAC Address: e.g. 08:00:06:01:57:3C

PROFINET Device Addressing

So that the IO devices are accessible for the IO controller, they must be supplied with unique address parameters. After the addressing, every IO device has three address parameters

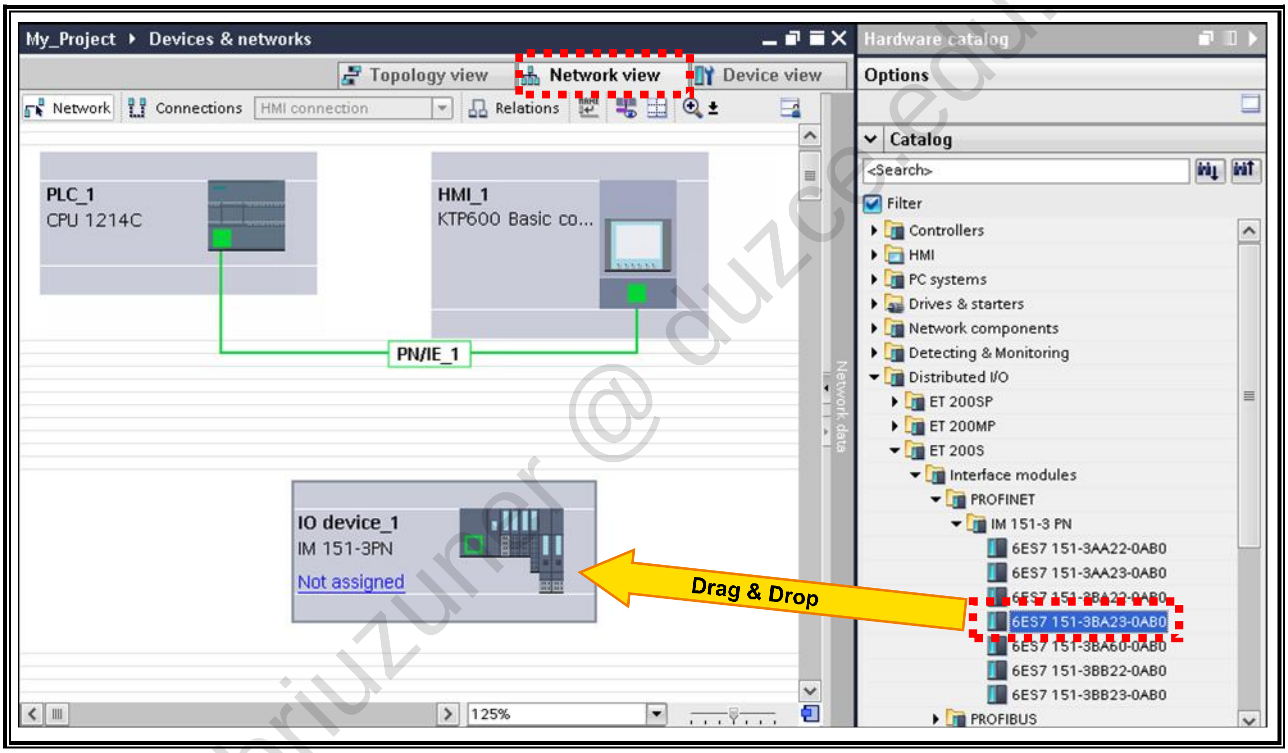
- MAC Address
  In its factory settings, every PROFINET device already has a fixed, world-wide unique MAC address. As a rule, this cannot be changed. It is required for the Real-Time communication.
- Device Name
  Before an IO device can be addressed by an IO controller it must have a device name. This procedure was chosen for PROFINET because names are easier to handle than complex IP addresses.
  So that the individual devices are accessible during the IO controller's system power-up, they are given device names. This occurs through the Supervisor. The device names which were given for the individual IO devices offline must match the online device names. This is comparable with setting the PROFIBUS address. If errors are made here, the IO controller cannot reach the IO device.

The rules for the converted names are listed in the following. If the converted name is not to be different from the name of the module, then the name of the module must comply with these rules.

- The name consists of one or more labels which are separated by a dot [.].
- Total length of the name: 1 to 240 characters
- Length of a label: 1 to 63 characters
- A label consists of the characters [a-z; 0-9]
- Labels must not begin or end with the characters "-"
- The first label must not begin with "port-xyz" or "port-xyz-abcde" (a,b,c,d, e,x,y,z=0-9)
- The name must not have the following format: n.n.n.n (n=0-999)

- IP Address
  In addition to the device name and the MAC address, an IO device also requires an IP address so that acyclic Read/Write services can be executed, for example. During system power-up, the IO controller assigns the IP addresses stored in the device configuration to the IO devices after checking the device names.

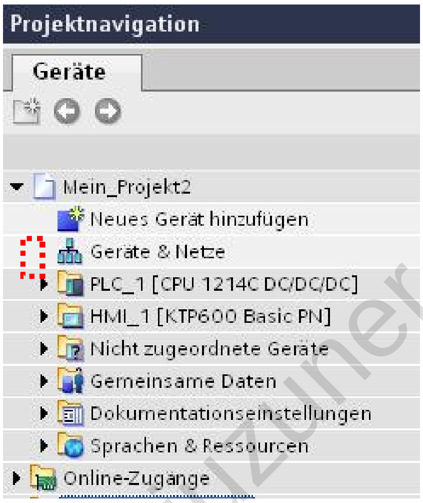
6.6. Inserting Distributed I/O into the Project (Network View)



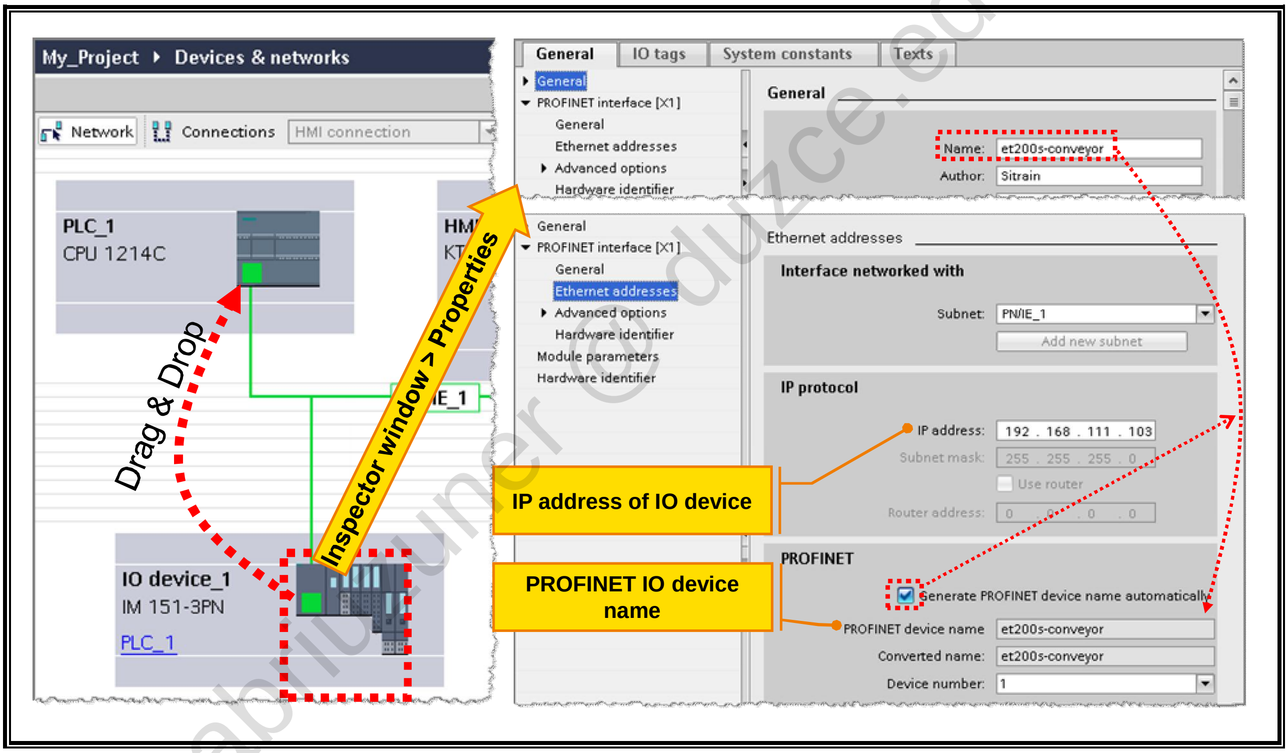
Inserting Distributed I/O into the Project

PROFINET IO devices are added in the Network view. Here, you can insert the relevant devices into the project by dragging & dropping them from the Hardware catalog.

In the beginning, the added device is not assigned to any controller and therefore appears in the Project tree as (an) "Unassigned devices" in the same level as the PLCs and HMIs.



6.7. Configuring a Connection to the CPU and Setting the Address Parameters



General

Name: et200s-conveyor

Author: Sitrain

Ethernet addresses

Interface networked with

Subnet: PN/IE_1

IP protocol

IP address: 192 . 168 . 111 . 103

Subnet mask: 255 . 255 . 255 . 0

□ Use router

Router address: 0 . 0 . 0 . 0

PROFINET

☑ Generate PROFINET device name automatically

PROFINET device name: et200s-conveyor

Converted name: et200s-conveyor

Device number: 1

Configuring the Connection to the CPU

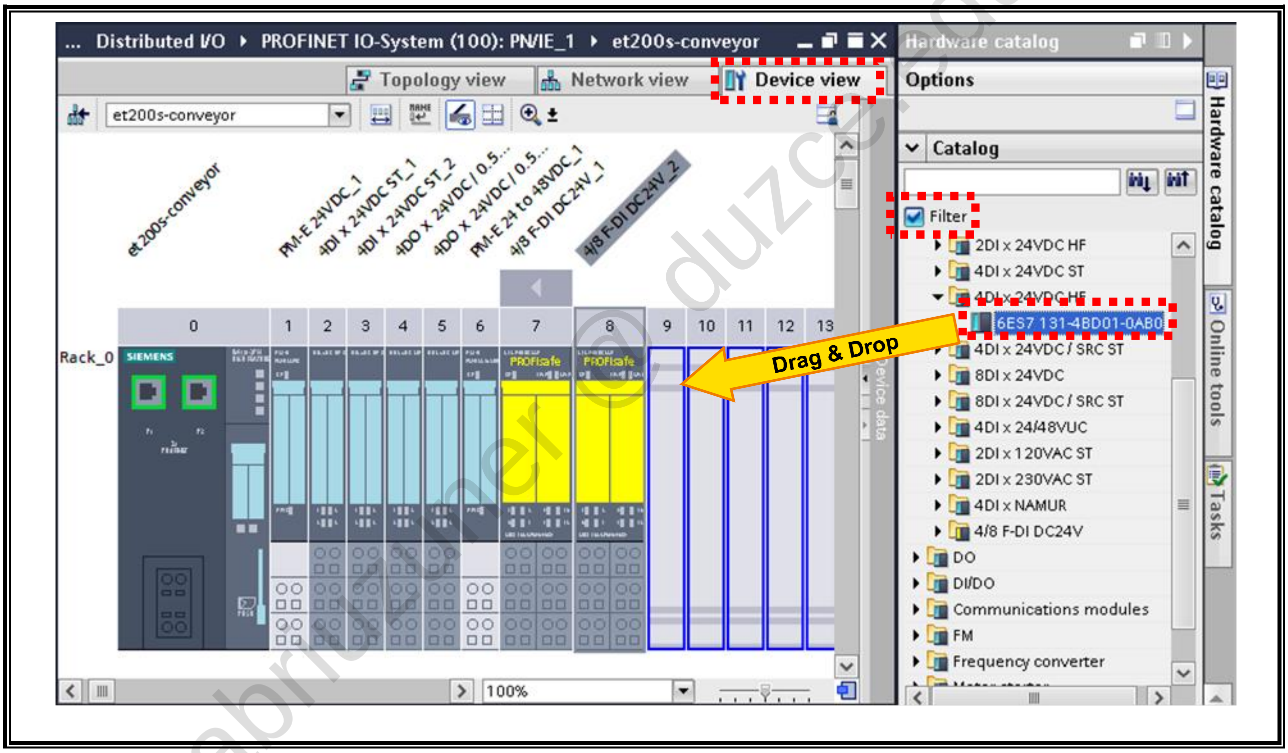
IO devices must be assigned to an IO controller. This is done by dragging & dropping the PROFINET interface of the IO device onto the PROFINET interface of the IO controller. The IO controller now recognizes the assignment and, during system power-up, queries the IO device with the configured device name.

Address Parameters

- Device Name
  The PROFINET device name is entered in the General part of the PROFINET interface of the respective IO device.

- IP Address
  The IP address is automatically assigned (first free subnet address) through the assignment of the IO device to the IO controller. It can be changed later on in the Properties of the PROFINET interface of the IO device.

6.8. Configuring Distributed I/Os (Device View)

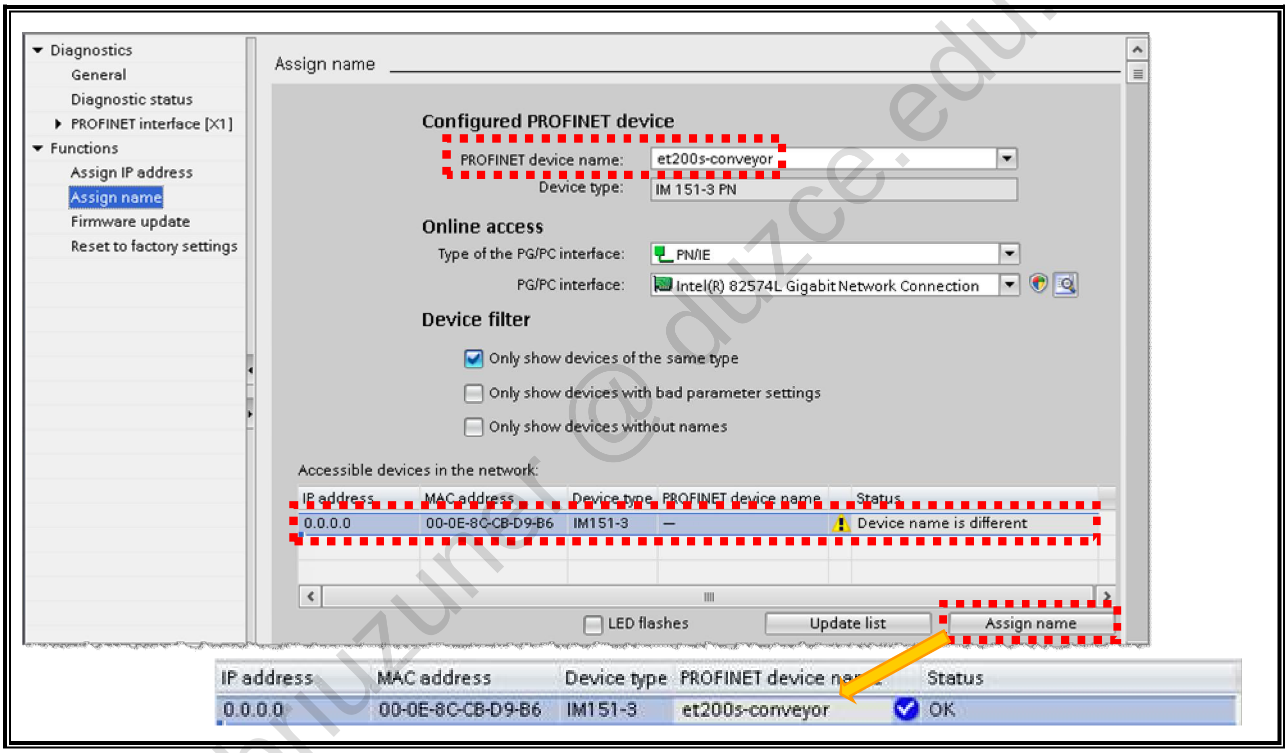


Configuring Distributed I/Os

Distributed I/Os are configured in the Device view by dragging the desired components from the Hardware Catalog onto the individual module slots.

If the checkmark at "Filter" is set, only the compatible modules are automatically displayed.

6.9. Writing the Device Name in the IO Device (Device Initialization)



Configured PROFINET device

PROFINET device name: et200s-conveyor

Device type: IM 151-3 PN

Online access

Type of the PG/PC interface: PN/IE

PG/PC interface: Intel(R) 82574L Gigabit Network Connection

Device filter

☑ Only show devices of the same type

☐ Only show devices with bad parameter settings

☐ Only show devices without names

Accessible devices in the network:

IP address	MAC address	Device type	PROFINET device name	Status
0.0.0.0	00-0E-8C-CB-D9-B6	IM151-3	—	Device name is different

☐ LED flashes

IP address	MAC address	Device type	PROFINET device name	Status
0.0.0.0	00-0E-8C-CB-D9-B6	IM151-3	et200s-conveyor	✓ OK

Device Initialization

The assignment of the device name to an IO device is the most important step in PROFINET addressing. The device name configured offline and the device name that exists online must match since the IO controller first checks the device names of the connected IO devices and then assigns the configured IP addresses during system power-up. If an IO device is not accessible under its configured device name, the IO controller cannot establish a connection to the IO device.

☞ The IP address does not have to be assigned manually. It is assigned by the IO controller during system power-up (after checking the device name). If, however, an IP address is assigned manually, it is then overwritten by the IO controller.

Ways of Assigning a Name

In principle, there are three ways of writing (assigning) the device name in an IO device, whereby only two of the three ensure that the offline configured device name really gets written into the IO device without errors.

- Version 1 and 2 (sure) → if possible, use one of these procedures!

  The assignment of the device name is triggered from the device configuration of the IO device.

  1. *Device configuration of IO device → Right-click on the Interface module (Slot 0) → Online & diagnostics → Functions → Assign name (see picture)*

  Or

  2. *Device configuration of IO device → Right-click on the Interface module (Slot0) → Assign device name*

  For both versions, the IO device is selected based on the MAC address and, to be sure that you have selected the right one, you can make the LEDs on the selected IO device flash by checking “LED flashes”. Since the configured device name is adopted, you can’t make any typing errors.

- Version 3 (possible typing errors)

  The assignment of the device name is triggered via "Accessible devices" or "Online accesses".

  Project view → Online accesses → Ethernet interface → IO device → Online & diagnostics → Functions → Assign name

  Here, any device name you choose can be assigned. This procedure can be used if you do not have the original project. The disadvantage vis-à-vis Version 1 or 2 is that you can make typing errors, that is, the device name does not match the offline configuration. In this way, the IO device is no longer accessible to the IO controller.

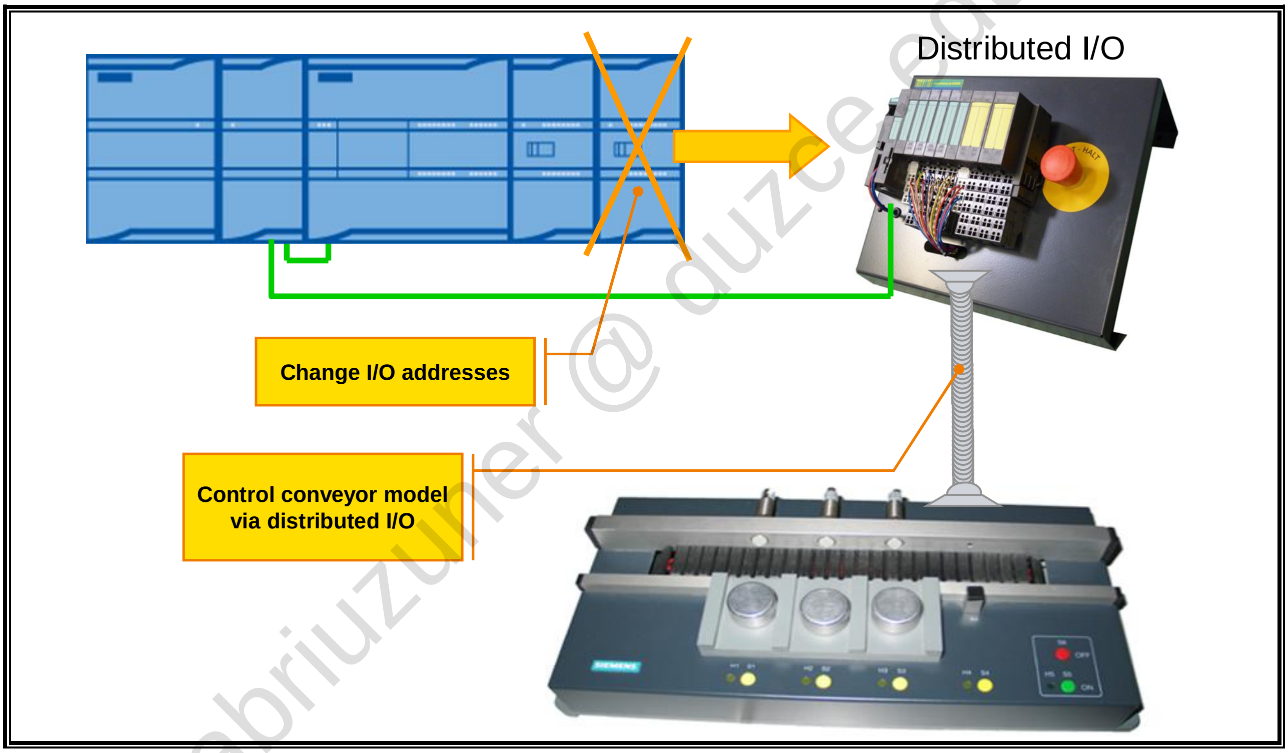
More than one PROFINET Device of the Same Type

Should several nameless PROFINET devices of the same type be available on the network, they can only be differentiated by their MAC address. This is printed on the Interface modules.

Alternatively, the function "LED flashes" can also be used to differentiate the PROFINET devices from one another. For this, a device is selected and the function "LED flashes" is started.

The LINK-LED(s) of the selected device always flash. This/These LED(s) normally show(s) that there is a physical connection between the device and the next switch.

6.10. Task Description: Controlling the Conveyor Model via the ET200S

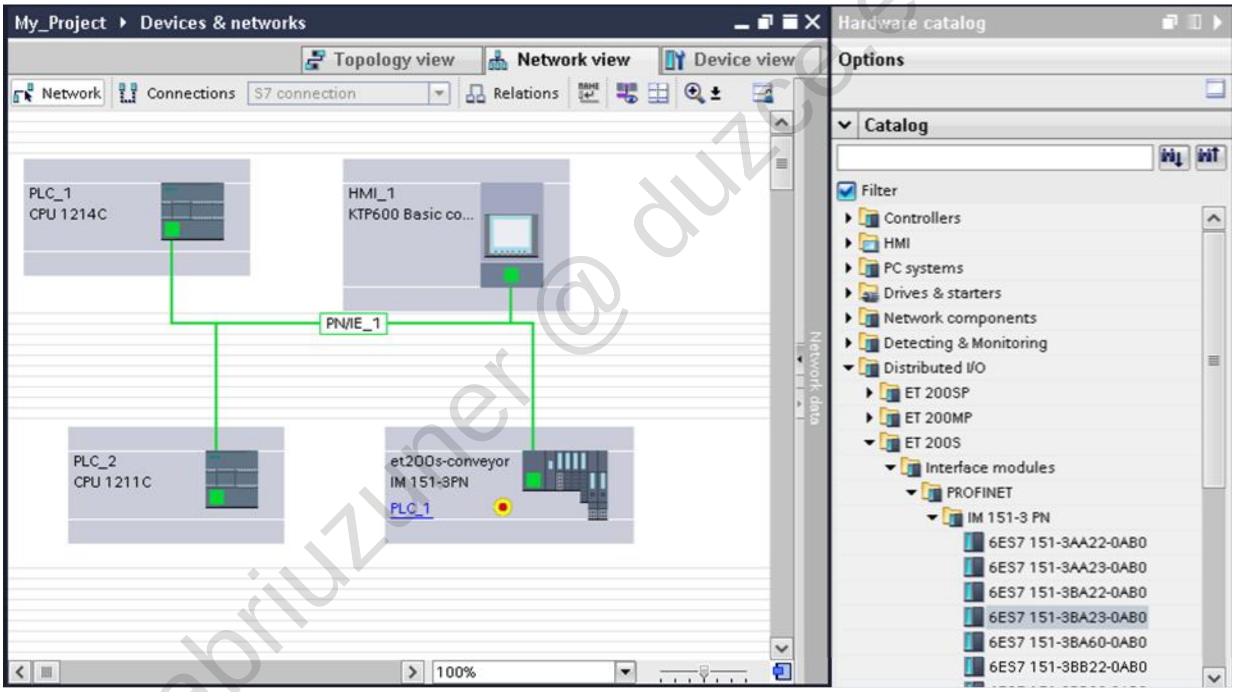


Task Description

The control of the conveyor model through the central 8DI/8DO module is now to be replaced by the I/O modules of the ET200S.

For this, you must insert the new IO device in your project and configure it. Furthermore, the current I/O addresses of the conveyor model are to be used (QB8 and IB8) so that the user program doesn’t have to be changed.

6.10.1. Exercise 1: Inserting the ET200S in the Project and Networking It



Task

Insert the Interface module of your ET200S into your project and network it with the CPU Station "PLC_1".

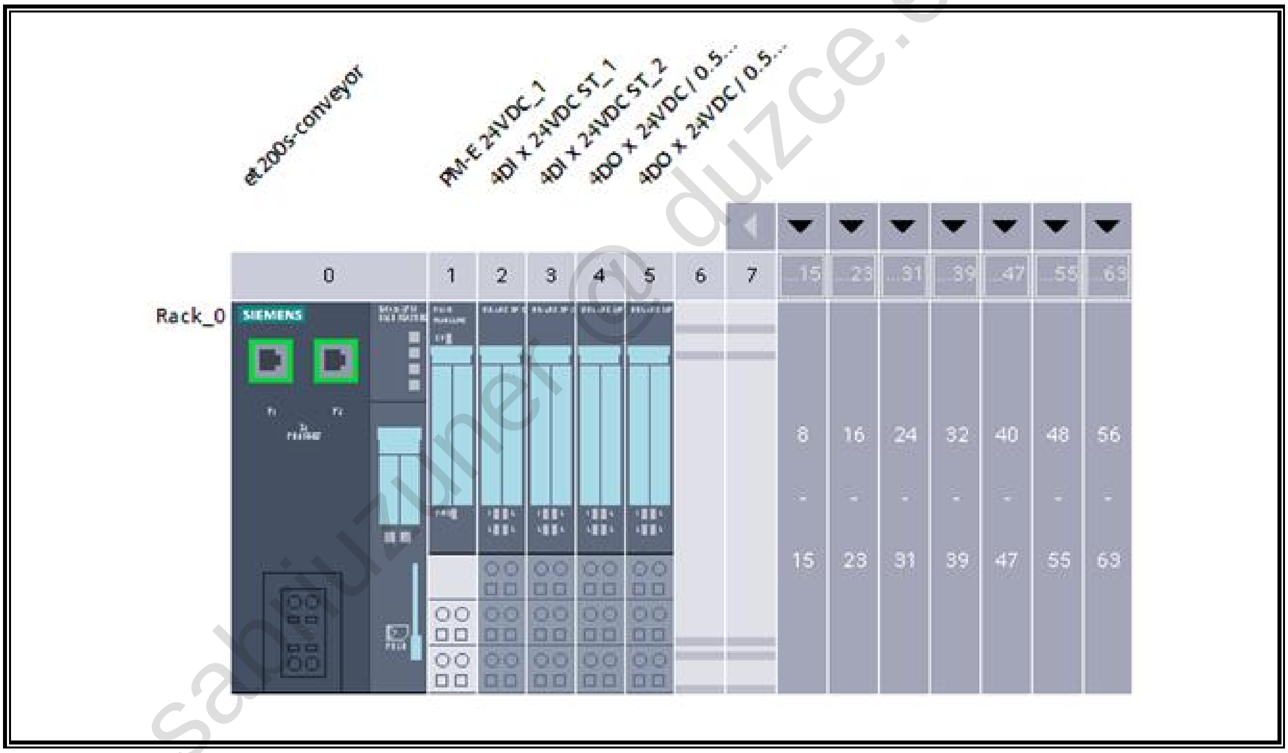
What to Do

1. Open the Network view of your project.
My_Project → Devices & networks → Network view

2. From the Hardware catalog, select the head module of your ET200S and drag it onto the network plan using drag & drop.

3. Network ET200S with the CPU.
Drag the PROFINET interface of the ET200S to the PROFINET interface of the CPU and drop it there.

6.10.2. Exercise 2: Configuring the ET200S and Setting the PROFINET Address Parameters



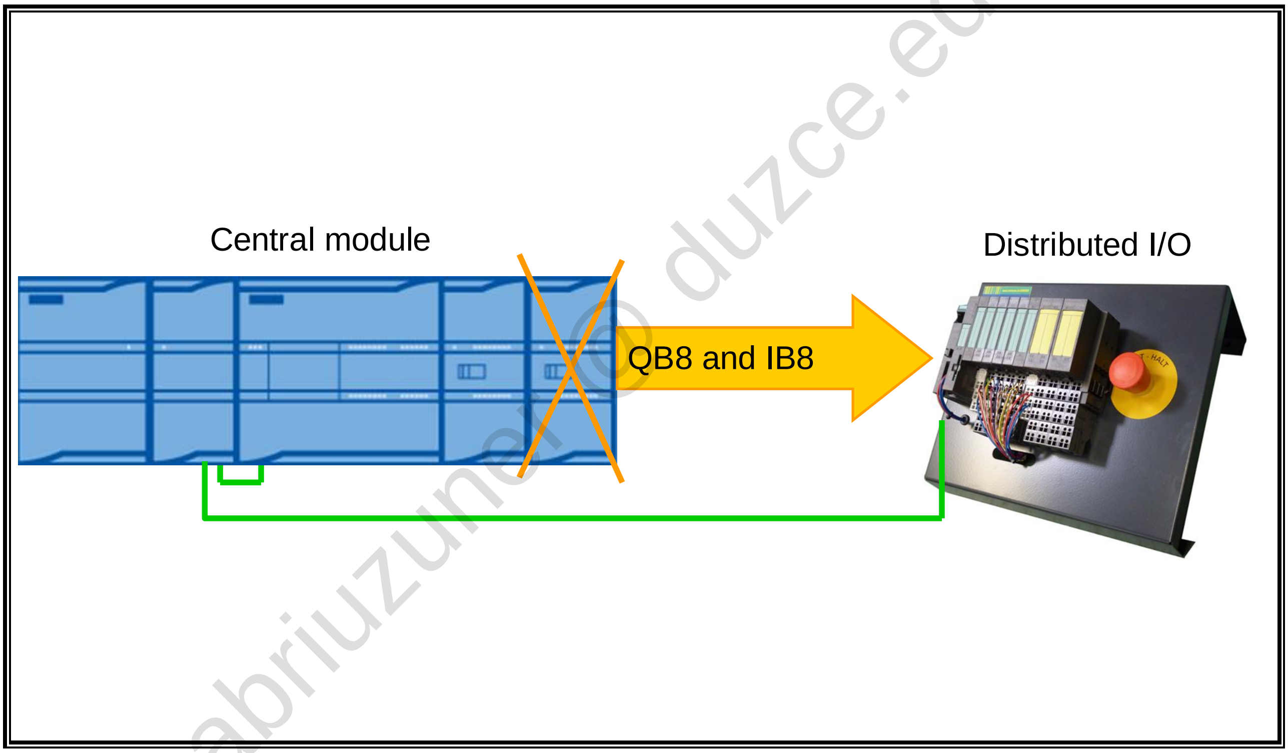
Task

Configure the modules of the ET200S and set the address parameters.

What to Do

1. Open the Device view of the ET200S.
2. Equip the ET200S with the modules according to the order number on the individual modules.
3. Set "**`et200s-conveyor`**" as the PROFINET device name
   Select the Interface module (Slot 0) → Inspector window → Properties → General → Name
4. Set the IP address **`192.168.111.103`** for the IO device.
   Select the Interface module (Slot 0) → Inspector window → Properties → PROFINET interface → Ethernet addresses → IP protocol

6.10.3. Exercise 3: Changing the I/O Addresses



Situation Up Until Now

The binary input signals of the conveyor model are in IB8; the output signals in QB8. Currently, QB8 and IB8 are processed by the central 8DI/8DO module.

Task

QB8 and IB8 are to be assigned to the I/O modules of the ET200S.

What to Do

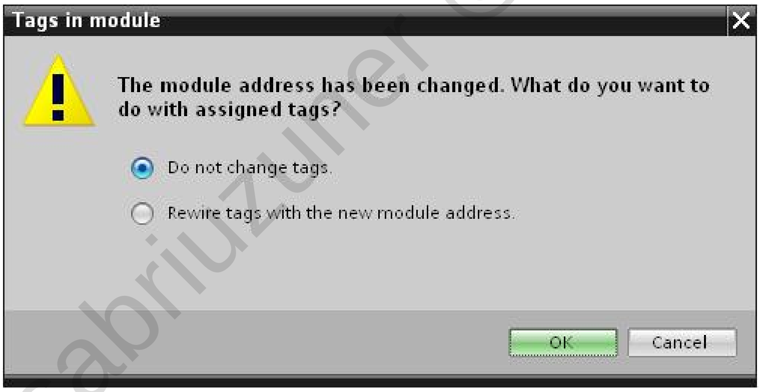
1. Open the Device view of "PLC_1" and navigate to the address properties of the 8DI/8DO module and assign the following addresses:

   Module → 8DI/8DQ → I/O addresses

   Input addresses → Start address: **88**

   Output addresses → Start address: **88**

2. In the pop-up message, select "Do not change tags", since the PLC tags of IB8 or QB8 are otherwise rewired to IB88 or QB88.



Tags in module

The module address has been changed. What do you want to do with assigned tags?

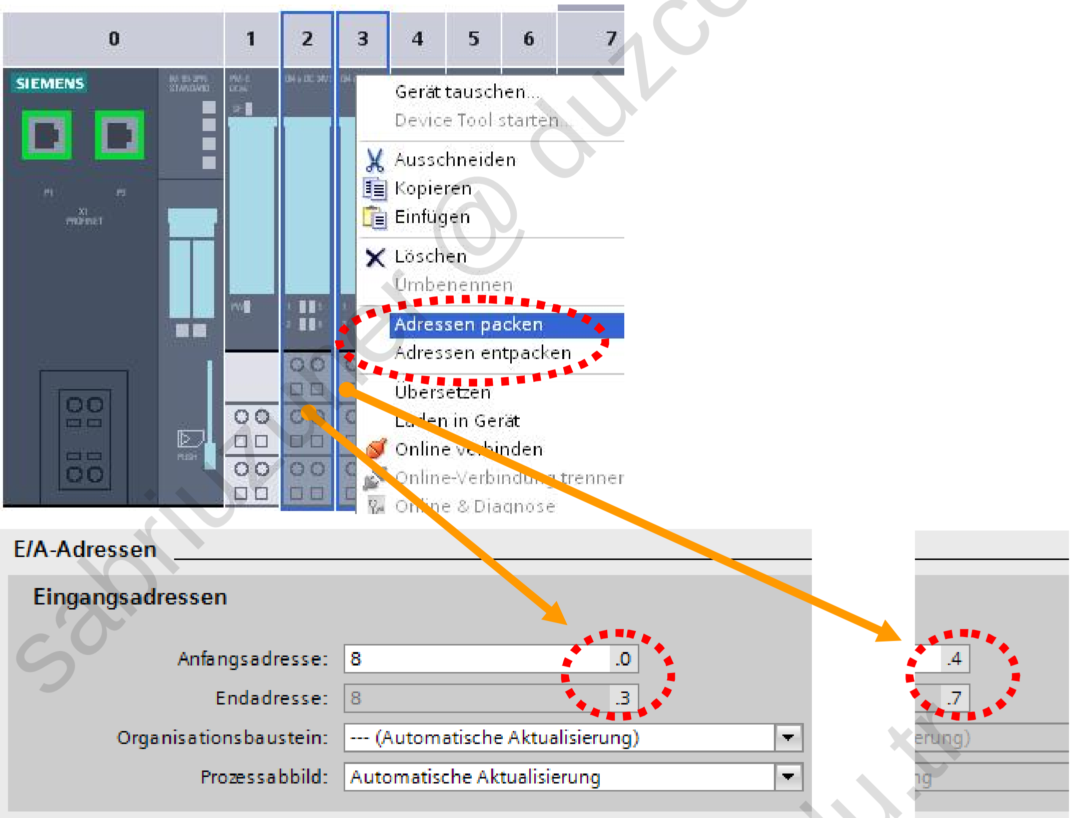
● Do not change tags.

○ Rewire tags with the new module address.

3. Open the Device view of the ET200S.

4. For the input and output addresses of the ET200S, configure the addresses IB8 and QB8.

☞ So that the 8 channels of a module are in one and the same byte, you have to pack the addresses. For this, select both modules, right-click on one of the modules and then click on "Pack addresses"



E/A-Adressen

Eingangsadressen

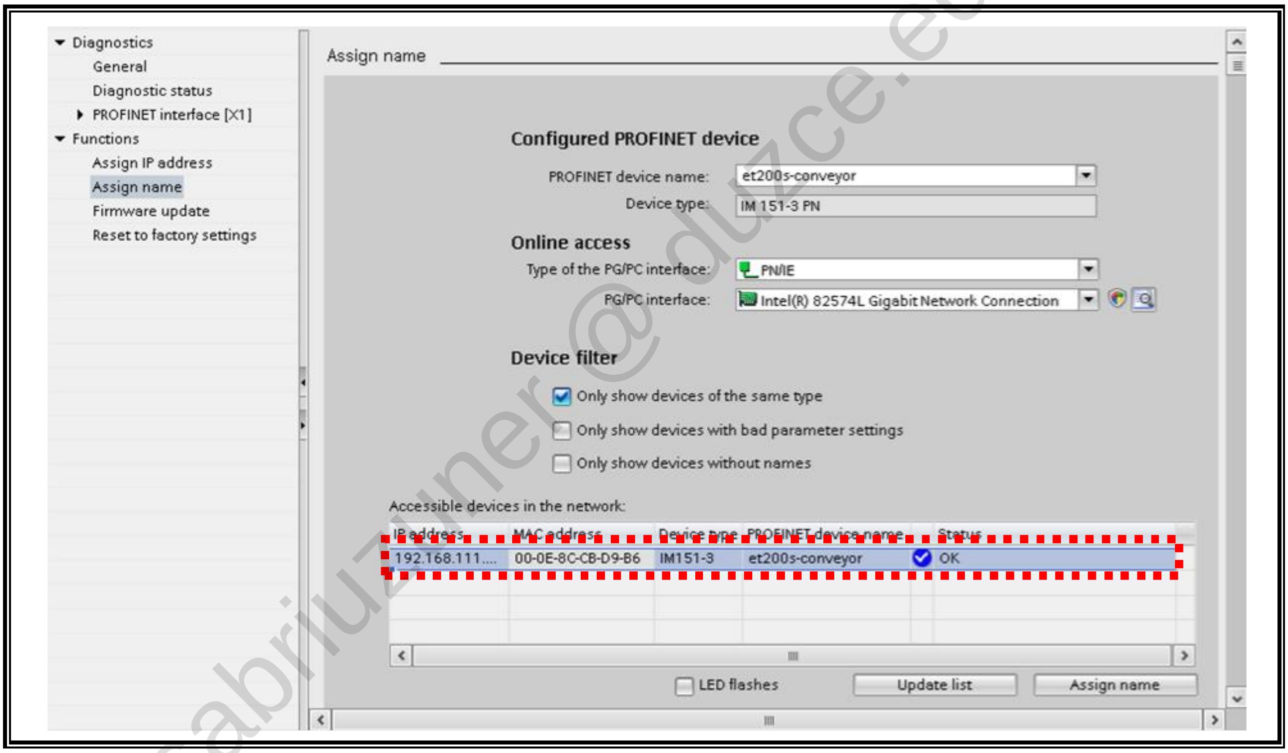
Anfangsadresse: 8 .0 .4

Endadresse: 8 .3 .7

Organisationsbaustein: --- (Automatische Aktualisierung)

Prozessabbild: Automatische Aktualisierung

6.10.4. Exercise 4: Writing the Device Name in the IO Device (Device Initialization)



Assign name

Configured PROFINET device

PROFINET device name: et200s-conveyor

Device type: IM 151-3 PN

Online access

Type of the PG/PC interface: PN/IE

PG/PC interface: Intel(R) 82574L Gigabit Network Connection

Device filter

☑ Only show devices of the same type

☐ Only show devices with bad parameter settings

☐ Only show devices without names

Accessible devices in the network:

IP address	MAC address	Device type	PROFINET device name	Status
192.168.111....	00-0E-8C-CB-D9-B6	IM151-3	et200s-conveyor	✓ OK

Task

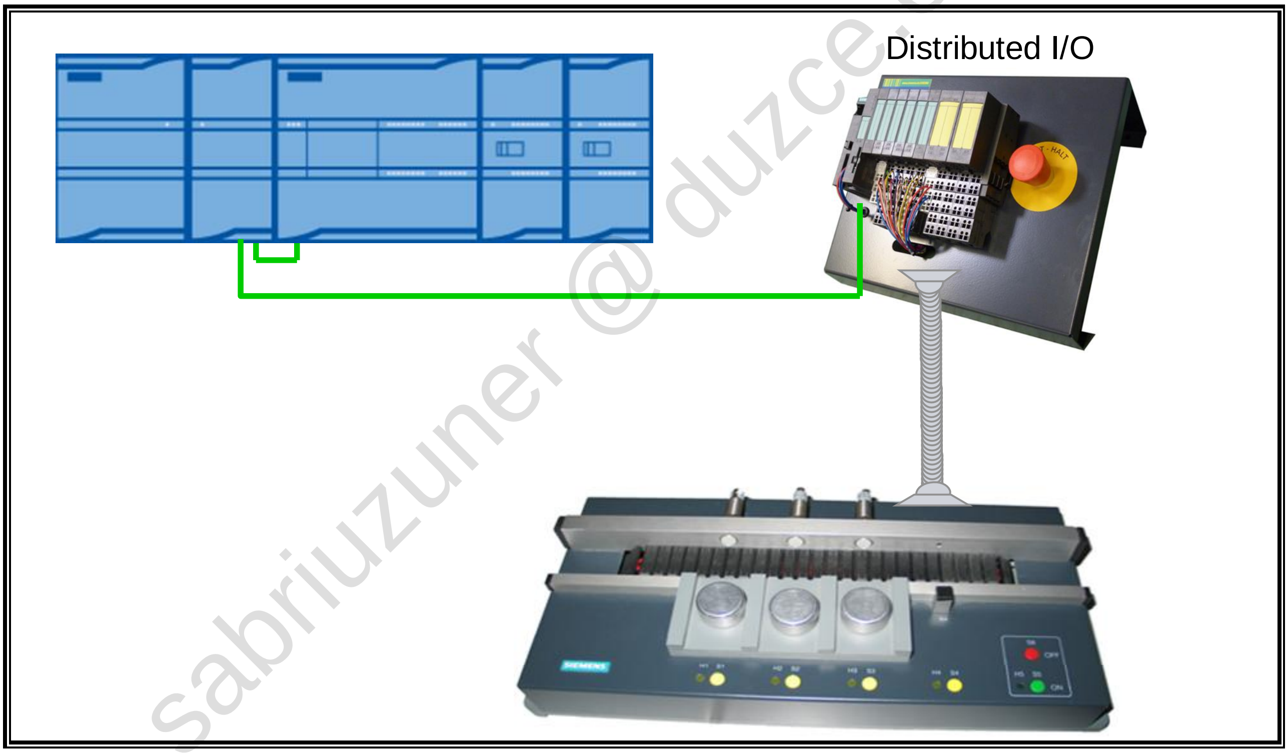
Assign the device name "et200s-conveyor" to your ET200S.

What to Do

1. Open the device configuration of the ET200S
2. Open the context menu of the Interface module and select "Online & diagnostics"
   Right-click on the Interface module (Slot 0) → "Online & diagnostics"
3. Navigate to "Assign name"
   "Online & diagnostics" → Functions → Assign name
4. From the list of accessible devices select the device of the type "IM151-3" and click on "Assign name"
5. Check whether the device name was assigned correctly by updating the list of accessible devices.

☞ The ET200S still doesn't have an IP address. This will be assigned by the IO controller after you have transferred the modified hardware configuration of "PLC_1" and the IO controller (CPU 1214) starts up again.

6.10.5. Exercise 5: Compiling the Modified Device Configuration and Testing the Program



Preparation

Remove the conveyor cable connector from the central training device and insert it in the ET200S' socket.

Task

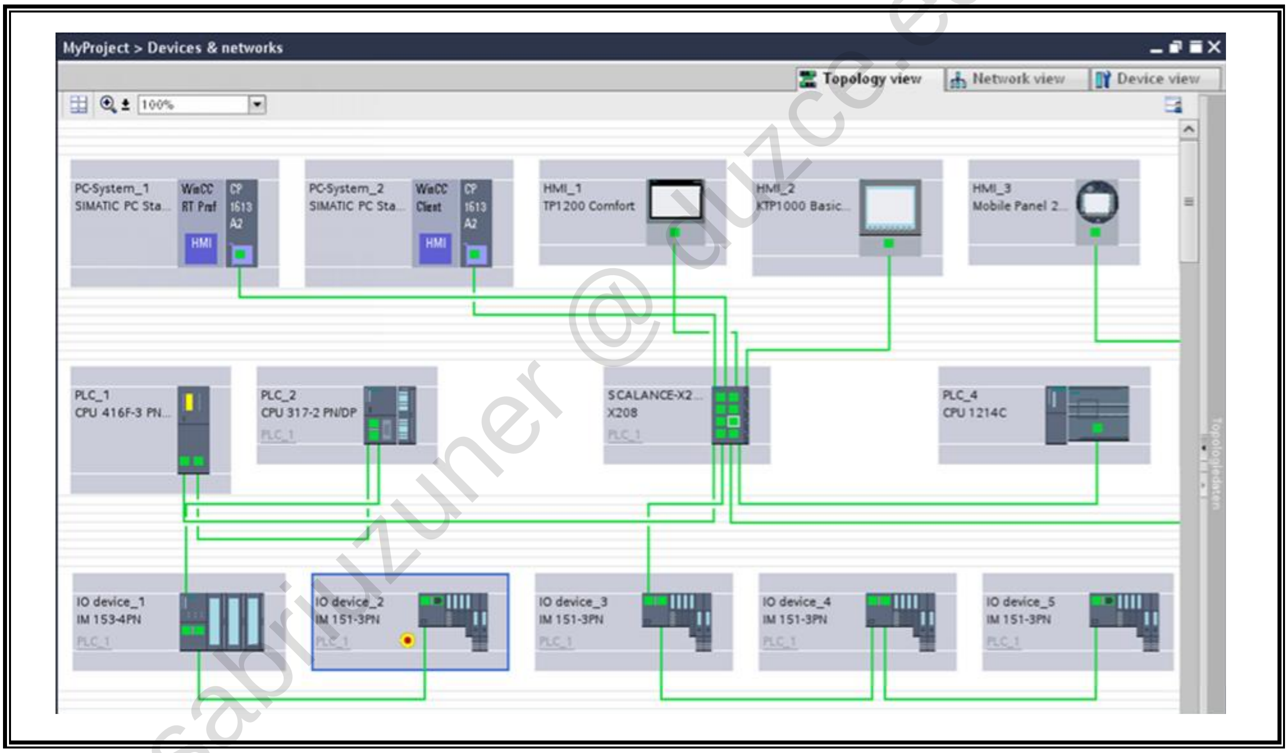
Download the modified hardware configuration of "PLC_1" into the controller and test whether your user program behaves as usual.

What to Do

1. Transfer the modified hardware configuration of "PLC_1" into the controller.
2. Test your user program by producing at least 1 part and test whether the conveyor model can be jogged to the left and the right when "P_Operation" = FALSE.

6.11. Additional Information

6.11.1. Topology Editor



Topology Editor

The configuration of the network topology is required for certain PROFINET functions. This includes, for example, the functions "Enabling device replacement without exchangeable medium" or "PROFINET IRT".

⚠ **Caution!**
The planned topology is loaded into the IO controllers involved. If the real topology is changed later on, the functions that are built up on it are no longer executed correctly. The LEDs "BF" and/or "MAINT" light up.

Functions of the Topology View

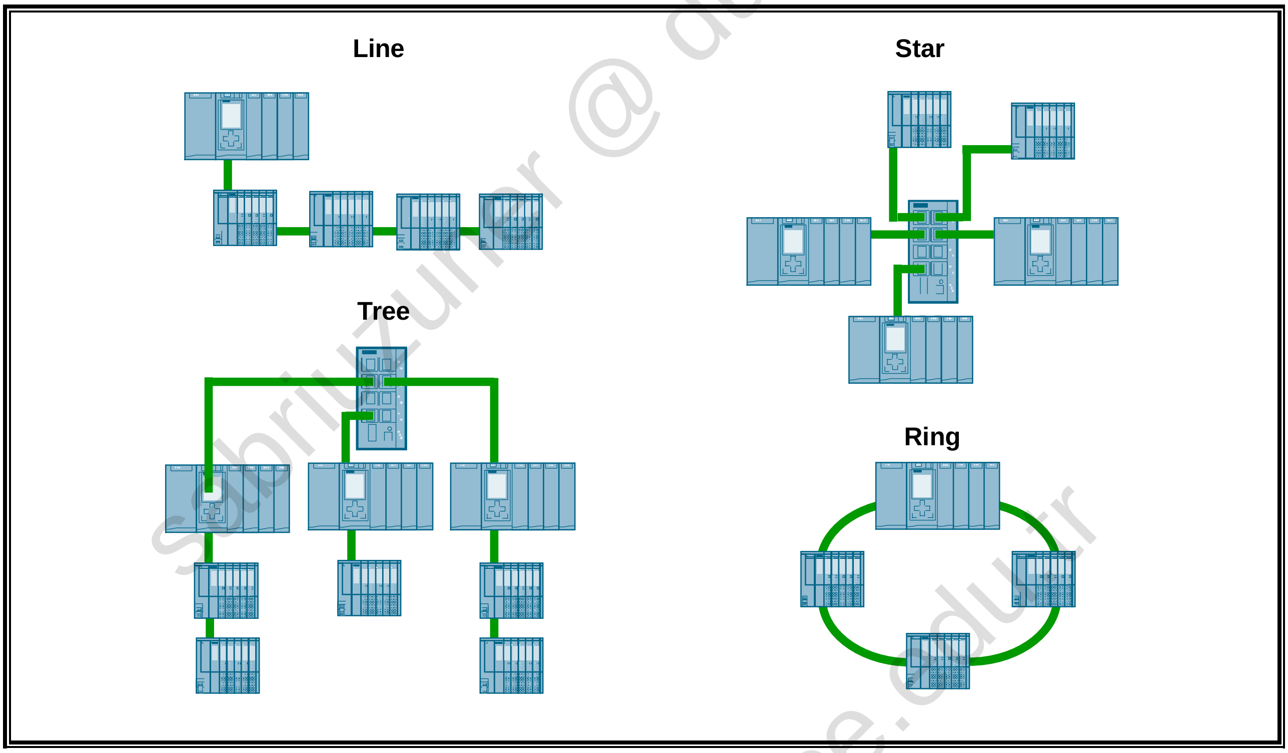
The Topology view is one of three work areas of the Devices & networks editor. The following tasks can be carried out:

- Display Ethernet topology
  - Display all PROFINET devices and passive Ethernet components of the project including ports
  - Display interconnections between the ports
  - Display associated logical networks
  - Display diagnostic information of all ports
- Configure Ethernet topology
  - Create, change and delete port interconnections
  - Rename stations, devices, interfaces, ports
  - Add PROFINET devices and passive Ethernet components to the project from the Hardware catalog
- Determine and minimize differences between the setpoint topology and the actual topology
  - Carry out offline/online comparison of Ethernet modules, Ethernet ports and Ethernet port interconnections
  - Adopt topology information existing online into the offline project

Differences between Network View and Topology View

- The Network view displays all logical subnets of the project.
- The Topology view displays all Ethernet components of the project. This also includes passive components such as switches and media converters and cables.
- The position of a device in the Network view and its position in the Topology view are independent of one another, that is, as a rule, one and the same device is located in a different position in each of the two views.

6.11.2. Topologies



Star Structure

The simplest network structure is a central Switch that enables the data transmission between the connected devices.

- Advantages
  - Easy managing, monitoring and diagnosis in the network
  - Flexible adding and removing of connections
- Disadvantages:
  - Single point of failure (one single critical point in case of error)
  - Wiring costs
- Appropriate for:
  - Small production areas
  - Individual production machines
  - Host system of a larger system

Line Structure

The devices are arranged in series.

- Advantages:
  - Cable cost savings for larger installations
  - Traditional fieldbus structure
- Disadvantages:
  - The transmission time can be influenced by the routing

Tree Structure

In principle, the tree structure is a combination of the line and star structure.

- Advantages:
  - The system is very transparent
  - Little data traffic since local data is restricted to the source location
  - Greater safety since local data remains in the originating area
  - Is also used to divide complex systems into logical subsystems
- Disadvantages:
  - Single point of failure (one single critical point in case of error)
  - Wiring costs

Ring Structure

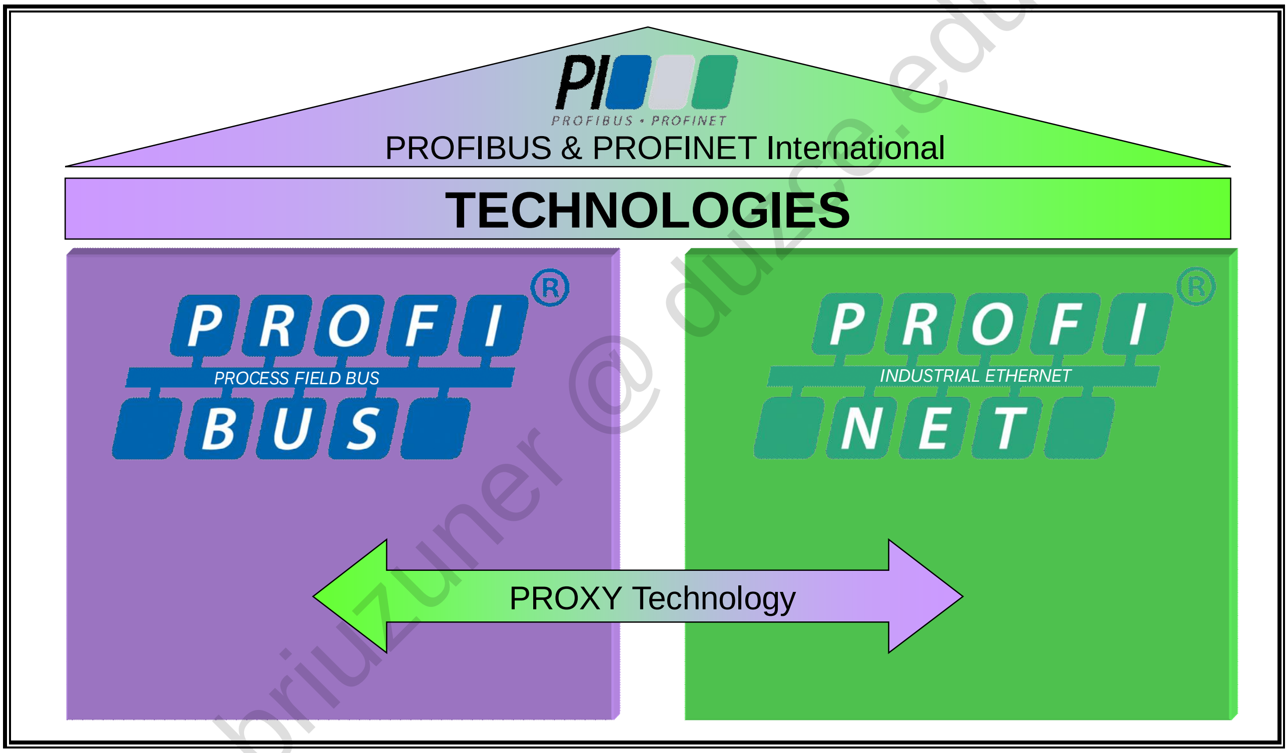
The ring structure is the result of connecting the ends of a line structure.

An internal interruption of the ring at a switch ensures that data packages do not circulate. With an interruption at another location, it is automatically closed.

The redundant path should not be combined with the outgoing path.

- Advantages:
  - Additional failure security
- Disadvantages:
  - Increased expenditures for hardware

6.11.3. The PROFIBUS User Organization

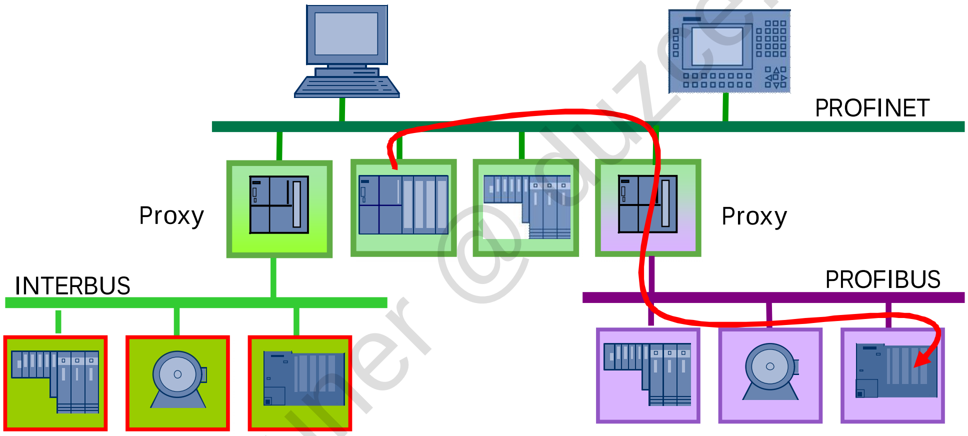


The PROFIBUS User Organization

The leading automation vendors have joined forces in the PROFIBUS User Organization (PNO). By now there are national user organizations in 25 countries which have united under the umbrella organization PROFIBUS & PROFINET International (PI) and which represent the interests of PROFIBUS users and manufacturers. PROFINET was integrated in the PI. The international promotion of PROFIBUS and PROFINET is supported in cooperation with all members. Over 1400 companies belong to the association. Included in the main tasks are:

- Continuous development of the technology
- Definition of user profiles
- Creation of test guidelines as specification for the test laboratories
- Certification activities to prove the conformance of Standards of PROFIBUS and PROFINET products
- Assignment and management of PROFIBUS identity numbers
- Assignment of PNO-MAC addresses for PROFINET
- Member support
- Information dissemination
- Public relations activities and
- Common marketing activities

6.11.4. PROFINET Proxy Concept



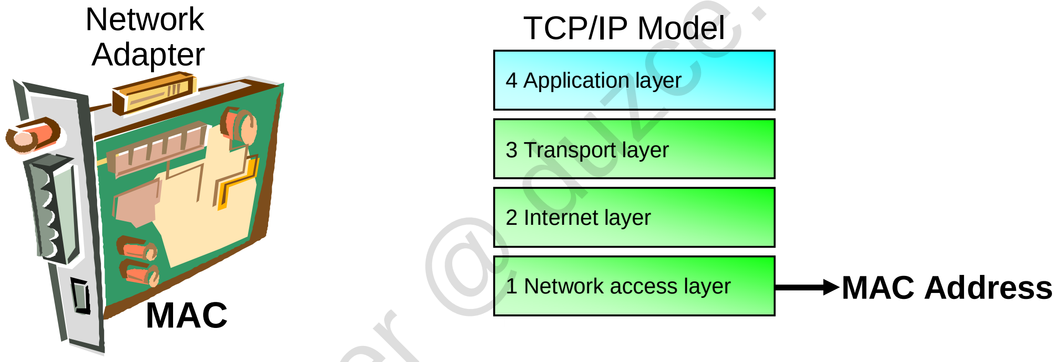
Field bus systems such as PROFIBUS or INTERBUS can be integrated in PROFINET systems via Proxies.

PN Proxy Concept

Through the use of Proxies/Gateways/Links, it is possible to integrate existing bus systems in a PROFINET network. For this, the manufacturers offer different solutions, for example the "IE/PB-Link" from Siemens for the integration of PROFIBUS devices.

From the point of view of PROFINET, a proxy is an IO device. For the connected PROFIBUS slaves, the proxy represents the PROFIBUS master.

6.11.5. The MAC Address



Manufacturer-specific part						Consecutive number					
47	40	39	32	31	24	23	16	15	8	7	0
0	0	0	E	C	F	X	X	X	X	X	X

MAC Address

Every Ethernet device (node) requires, for the identification in the network on Layer 2 of the ISO/OSI model, a unique address as a network access point for the layers above it. For that reason, each device has a fixed, world-wide unique address which is given by the factory. This is called the MAC (Media Access Control) address or MAC for short.

A MAC address has a length of 48 bits and is usually depicted in "canonical representation" (LSB format) (e.g. with "ipconfig /all"). For the transmission of data it is defined that the least significant bit of an octet (LSB) is sent first.

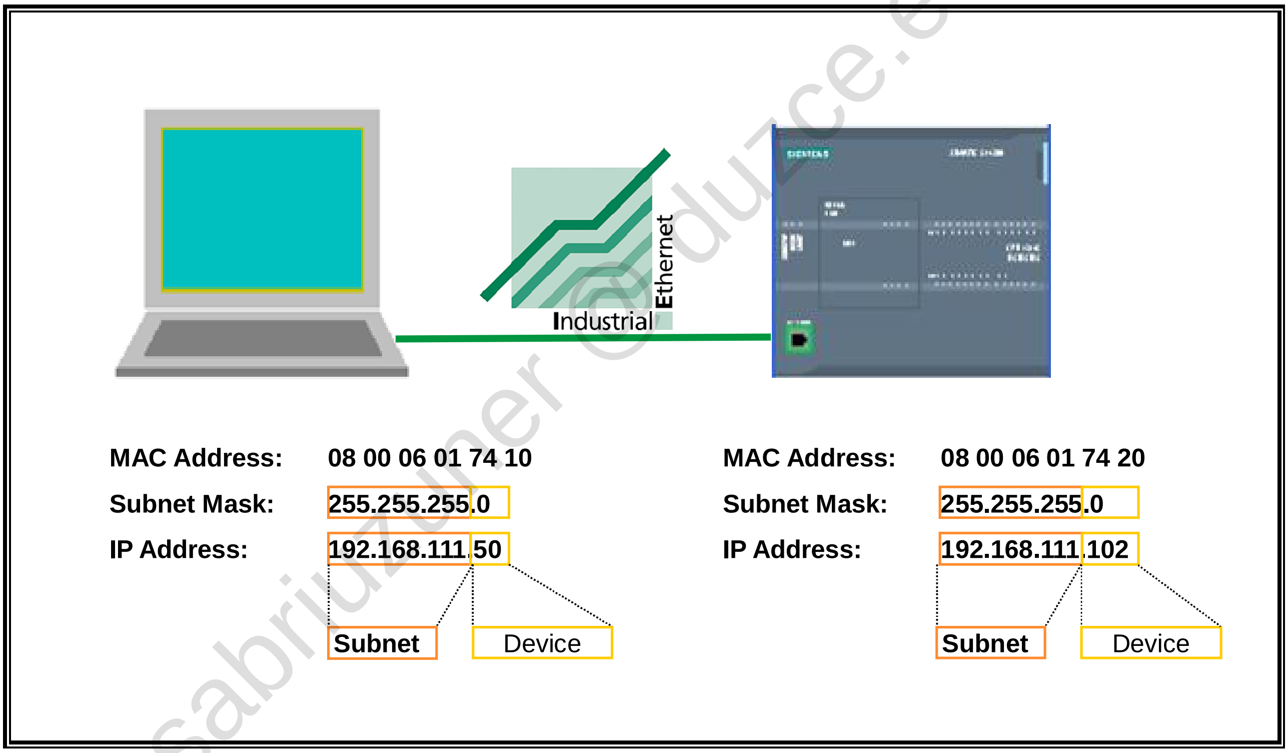
Ethernet A dress

The Ethernet address is 6 bytes long in hexadecimal notation. It is divided into a manufacturer-specific part and a consecutive number. For smaller companies it may make sense to use the PNO Ethernet address. That way, they don’t have to apply for their own Ethernet address. The OUI (Organizationally Unique Identifier) of the PNO is 00-0E-CF

Examples

• 00-0E-CF	PROFIBUS User Organization
• 00-0E-8C	Siemens AG A&D ET
• 08-00-06	Siemens AG IT Solutions
• 00-01-E3	Siemens AG
• 00-0E-F0	Festo AG & Co. KG

6.11.6. Industrial Ethernet: IP Address and Subnet Mask



MAC Address: 08 00 06 01 74 10
Subnet Mask: 255.255.255.0
IP Address: 192.168.111.50
Subnet
Device

MAC Address: 08 00 06 01 74 20
Subnet Mask: 255.255.255.0
IP Address: 192.168.111.102
Subnet
Device

Internet Protocol

The Internet Protocol (IP) is the basis for all TCP/IP networks. It creates the so-called datagrams (data packets specially tailored to the Internet protocol) and handles their transport within the local subnet or their "routing" (forwarding) to other subnets.

IP Addresses

IP addresses are not assigned to a specific computer, but rather to the network interfaces of the computer. A computer with several network connections (for example routers) must therefore be assigned an IP address for each connection.

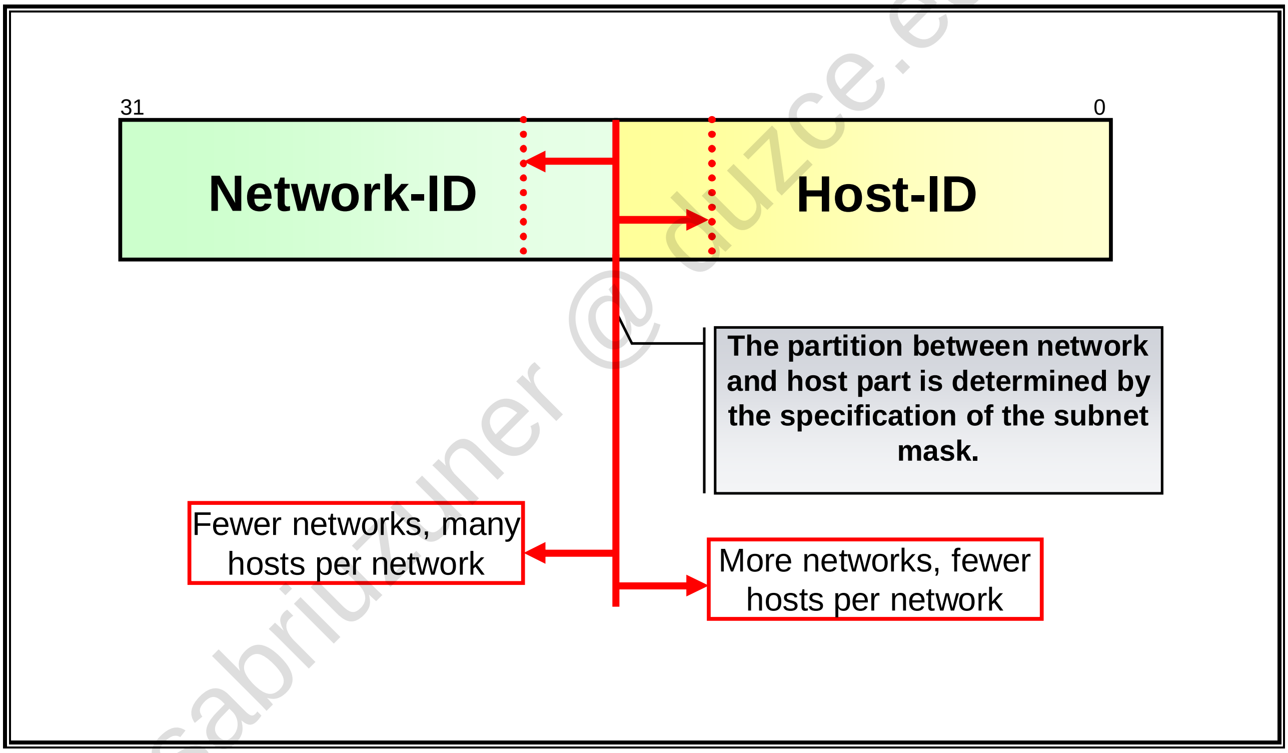
IP addresses consist of 4 bytes. With the dot notation, each byte of the IP address is expressed by a decimal number between 0 and 255. The four decimal numbers are separated by dots (see picture).

Subnet Mask

The subnet mask (also net mask or network mask) is (in binary notation) a sequence of ones followed by a sequence of zeros. The partition between ones and zeros marks the separation between the network part and the computer (host) part of the IP address.

.

6.11.7. The Partitioning of the IP Address


31
0
Network-ID
Host-ID
The partition between network and host part is determined by the specification of the subnet mask.
Fewer networks, many hosts per network
More networks, fewer hosts per network

The Partitioning of the IP Address

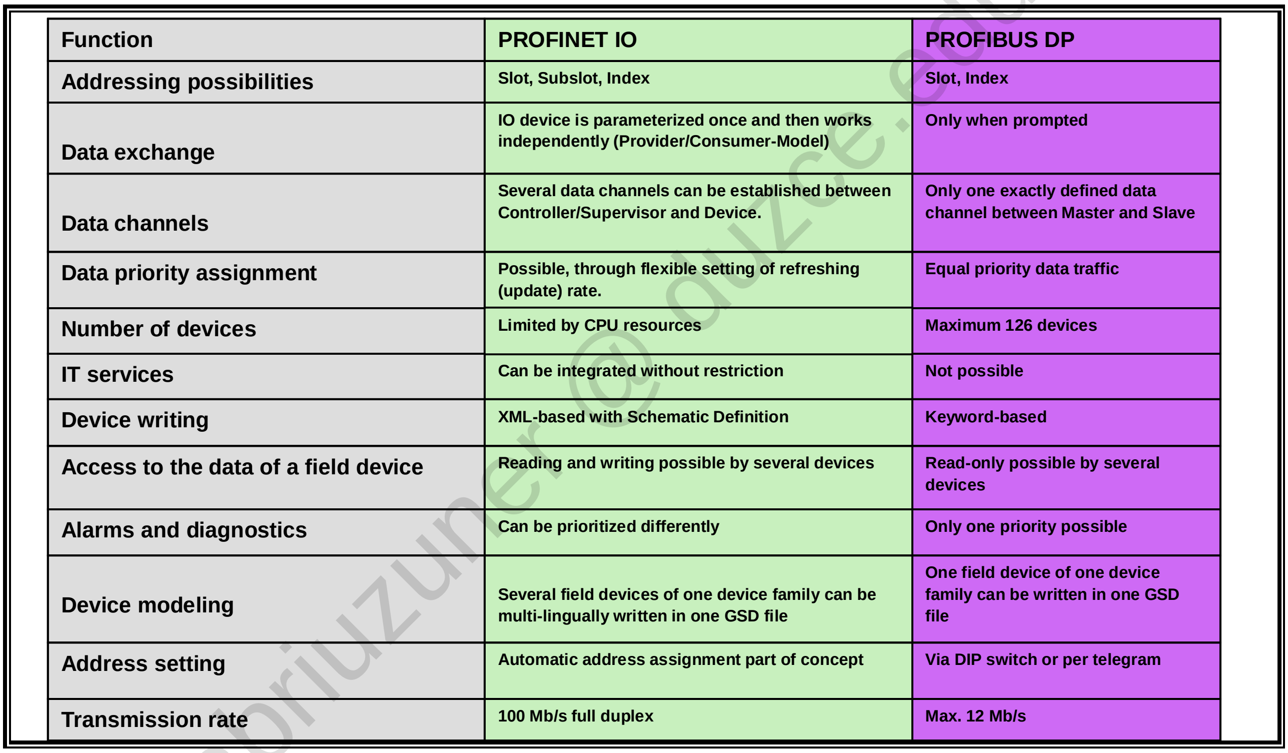
The IP address is divided into a device part (Host-ID) and a network part (Network-ID). Originally, 5 different network classes were defined worldwide (A,B,C,D,E). For the network classes A to C, it was uniquely defined which part of the IP address represents the Network-ID and which part represents the Host-ID.

Attention!
Today, IP addresses are no longer divided into classes. So that the partition between Network-ID and Host-ID can still be determined, a subnet mask is also specified.

Network Size

If you shift the partition between Network part and Host part to the left, you have fewer networks with many devices per network. If, however, you shift it to the right, you have many networks with few devices per network.

6.11.8. Comparison of PROFINET IO and PROFIBUS DP



Function	PROFINET IO	PROFIBUS DP
Addressing possibilities	Slot, Subslot, Index	Slot, Index
Data exchange	IO device is parameterized once and then works independently (Provider/Consumer-Model)	Only when prompted
Data channels	Several data channels can be established between Controller/Supervisor and Device.	Only one exactly defined data channel between Master and Slave
Data priority assignment	Possible, through flexible setting of refreshing (update) rate.	Equal priority data traffic
Number of devices	Limited by CPU resources	Maximum 126 devices
IT services	Can be integrated without restriction	Not possible
Device writing	XML-based with Schematic Definition	Keyword-based
Access to the data of a field device	Reading and writing possible by several devices	Read-only possible by several devices
Alarms and diagnostics	Can be prioritized differently	Only one priority possible
Device modeling	Several field devices of one device family can be multi-lingually written in one GSD file	One field device of one device family can be written in one GSD file
Address setting	Automatic address assignment part of concept	Via DIP switch or per telegram
Transmission rate	100 Mb/s full duplex	Max. 12 Mb/s

Comparison

PROFINET IO and PROFIBUS DP differentiate themselves in the features shown in the picture.

Contents 3

3. Analog Value Processing

3.1. Objectives

At the end of the chapter the participant will ...

… be familiar with the principle of analog value processing

... be able to assign parameters to an analog module

... be able to address an analog module

... be able to interpret the resolution of a module

… be familiar with the operations for the analog value conversion

… be able to program a simple analog value conversion

... be able to evaluate the diagnostics interrupt of the analog module

… be familiar with the principle of interrupt processing

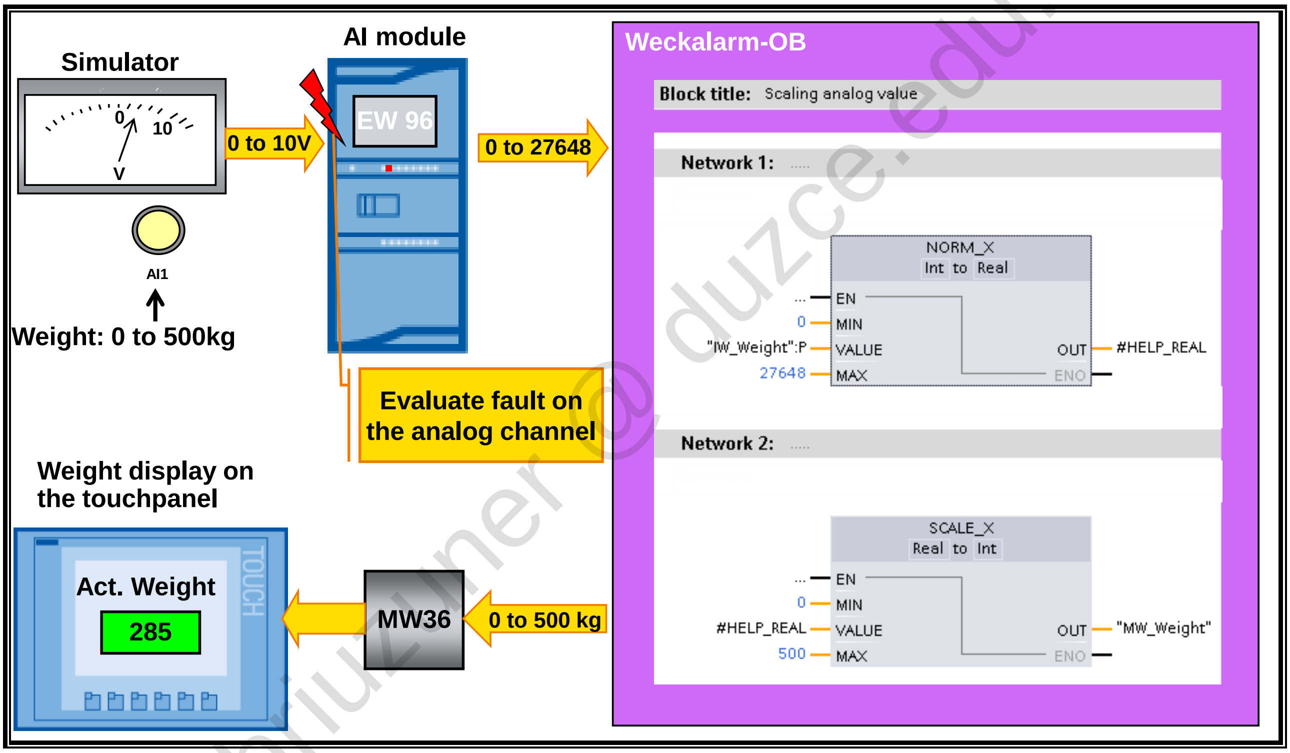
… be able to generate and program a cyclic interrupt

Objectives

In this chapter, the principle of analog value processing is presented. The goal is that the participant is capable of parameterizing an analog module and of interpreting the resolution.

Furthermore, the necessary conversion operations are presented in order to be able to process an analog value. The participant should be able to program a simple analog value conversion and be able to interpret a diagnostics interrupt of an analog module.

3.2. Task Description



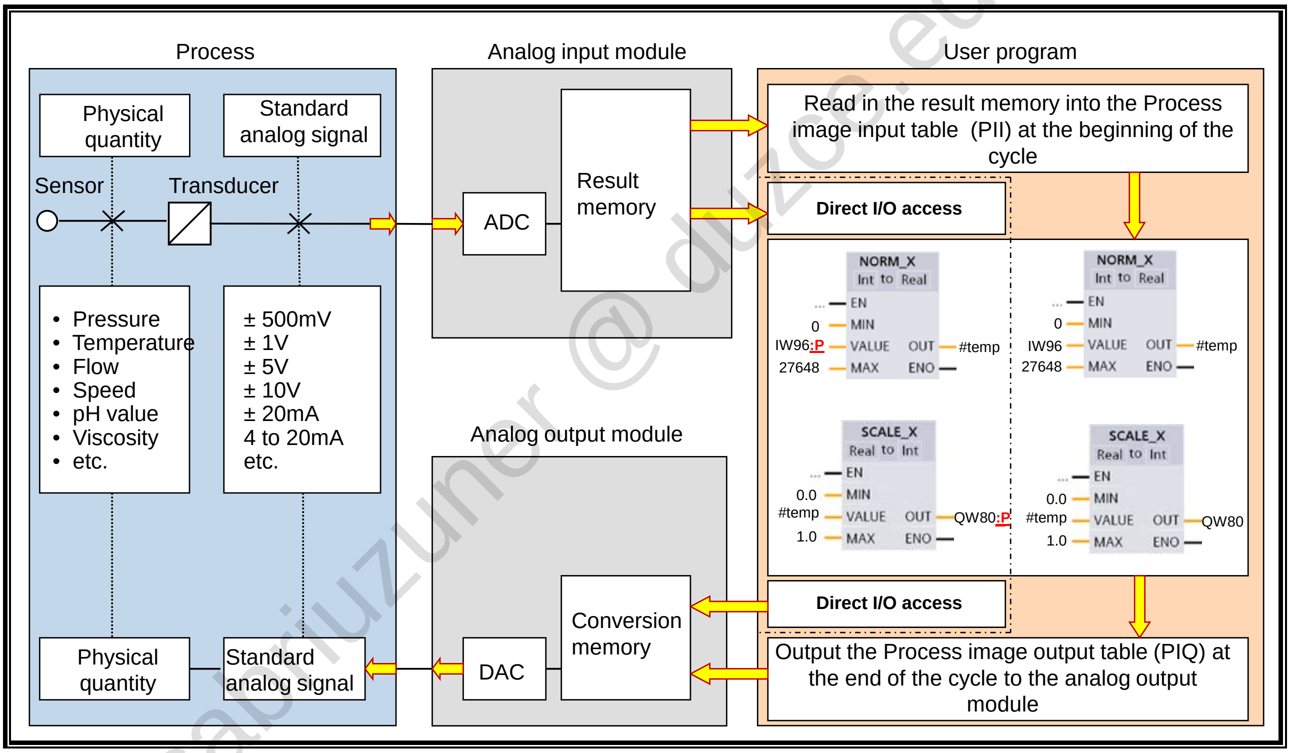
Task Description

In this chapter, the conversion and processing of analog signals is handled.

For this, a voltage is to be set and read in on the simulator potentiometer. This voltage simulates part weight values. It will be your task to convert the read-in values every 250ms in the cyclic interrupt into weight values between 0 kg and 500 kg using the operations NORM_X and SCALE_X. The weight is only valid in the range of 100kg to 400kg. If the weight of the part exceeds or falls below these limits, the part is considered invalid and no further transport sequence can be started (Bay LEDs remain dark and conveyor movement to the right cannot be started).

As well, you will learn how you must proceed when there is a channel fault of an analog module in order to get detailed information on the fault event.

3.3. Principle of Analog Value Processing



Principle of Analog Value Processing

In a production process, there are a variety of physical quantities (such as pressure, temperature, speed, rotational speed, pH value, and viscosity etc.) that need to be processed in the PLC for automation purposes.

Sensor

Measuring sensors respond to changes in the quantity to be measured by such things as linear expansion, angular ductability, and alteration of electrical conductivity.

Transducer

Measuring transducers convert these above-mentioned changes into standard analog signals, such as: ± 500mV, ± 10V, ± 20mA, 4 to 20mA.

These signals are supplied to the analog input modules.

ADC

Before these analog values can be processed in the CPU, they must be converted to digital form. The ADC (Analog-to-Digital Converter) on the analog input module handles this conversion.

The analog-to-digital conversion is performed sequentially. This means the signals are converted for each analog input channel in turn.

Result Memory

The result of the conversion is stored in the result memory and remains there until it is overwritten by a new value.

You can use the “IW...:P” addressing to read the converted analog value directly from the I/O.

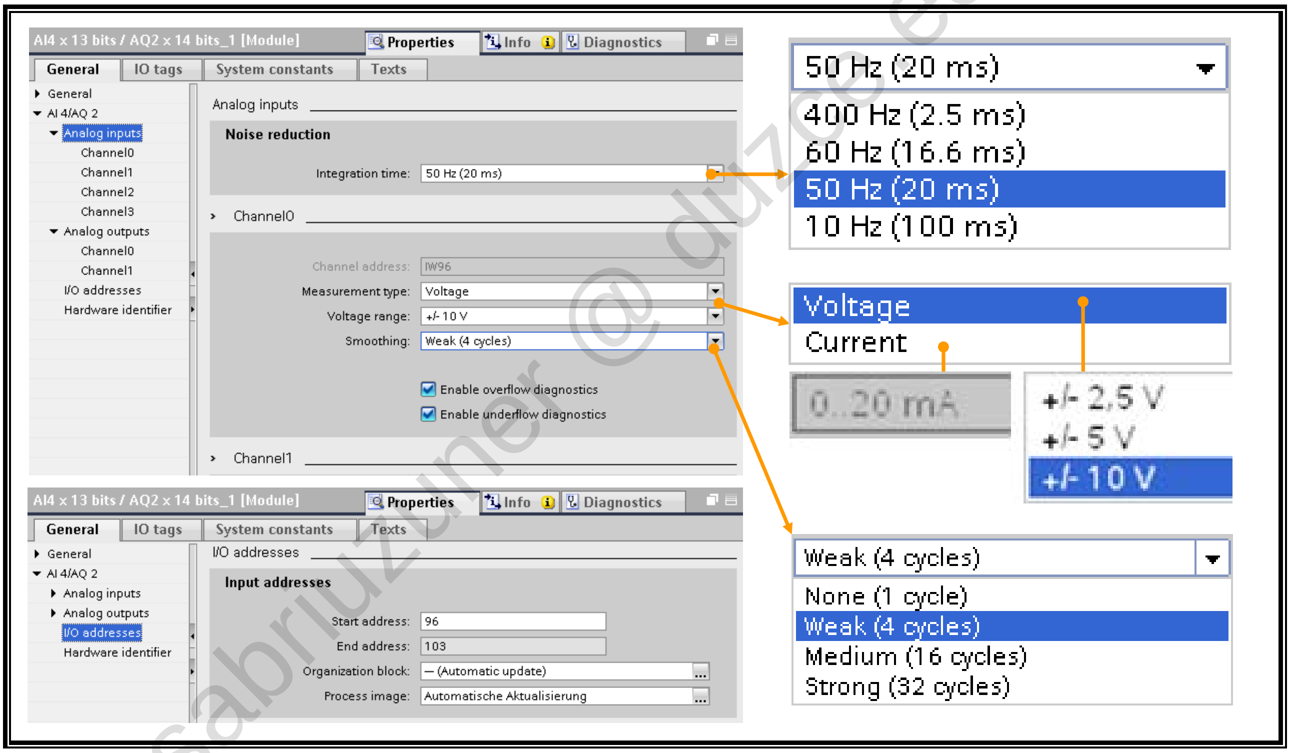
Analog Output

The (MOVE) transfer instruction is used to write the analog values the user program calculated to an analog output module, where a DAC (Digital-to-Analog Converter) converts them to standard analog signals.

Analog Actuators

You can connect standard actuators directly to the analog output modules.

3.4. Properties of Analog Input Modules



AI4 x 13 bits / AQ2 x 14 bits_1 [Module]

Analog inputs

Noise reduction

Integration time: 50 Hz (20 ms)

Channel0

Channel address: IW96

Measurement type: Voltage

Voltage range: +/- 10 V

Smoothing: Weak (4 cycles)

☑ Enable overflow diagnostics

☑ Enable underflow diagnostics

Channel1

AI4 x 13 bits / AQ2 x 14 bits_1 [Module]

I/O addresses

Input addresses

Start address: 96

End address: 103

Organization block: --- (Automatic update)

Process image: Automatische Aktualisierung

Analog Input Modules

In STEP7, analog input modules are configured and assigned parameters in the Device configuration of the respective PLC. The settings or parameters of all modules are downloaded into the CPU. The CPU must be in the STOP state to do this. In a subsequent CPU warm restart, the CPU transfers these parameters to the relevant modules.

Parameters

For the respective module, differentiation is made between module parameters and channel parameters.

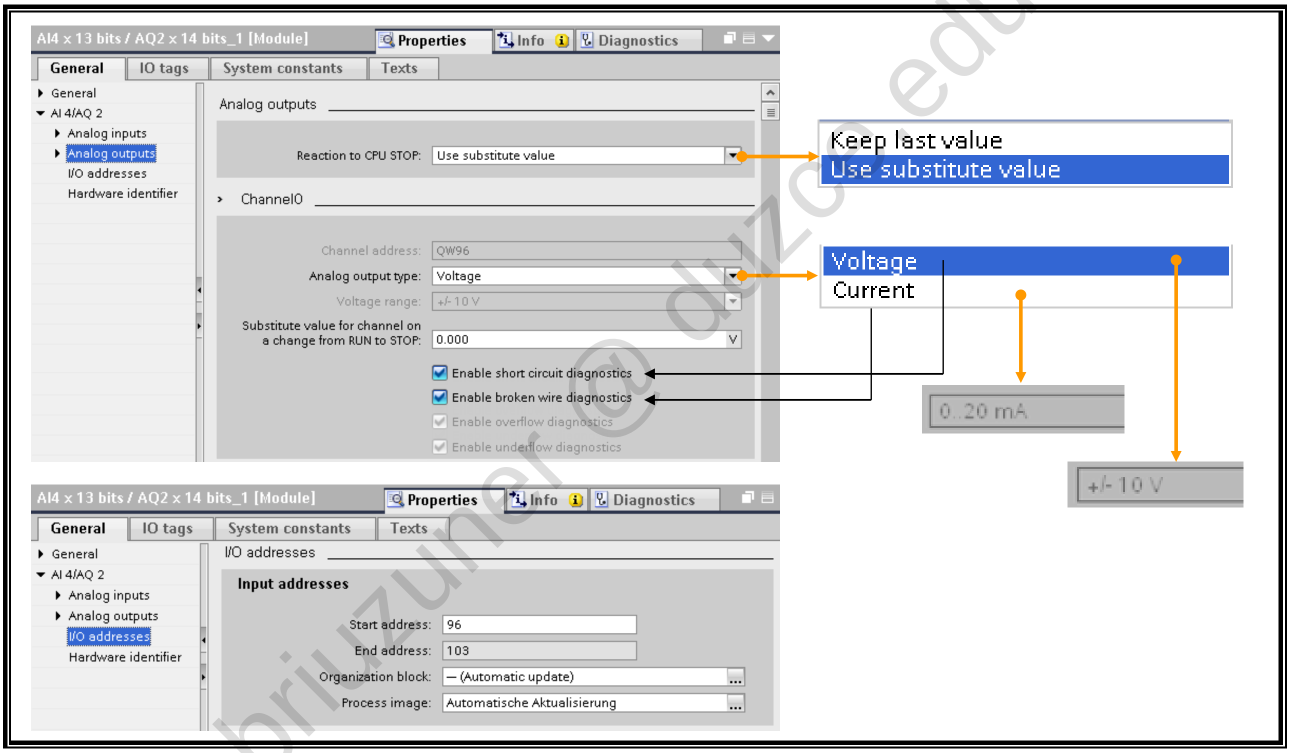
Module Parameters

- General
  Name and comment for the integrated analog inputs of the CPU.
- Noise Reduction
  In the noise reduction, the noise frequencies of the specified frequency (in Hz) are suppressed by the integration time which is set.
- I/O Addresses and Hardware Identifier
  The address space of the entry addresses as well as the process image is defined. The hardware identity of the device is displayed.

Channel Parameters

- Measurement Type
  The type of measurement, such as voltage, is set with this parameter. An unused channel must then be deactivated since it is otherwise also converted which would result in a longer total conversion time of the module.
- Measuring Range (in the picture – Voltage range)
  With this parameter, the measuring range of the selected type of measurement is set.
- Smoothing
  The smoothing of analog values generates a stable analog signal for further processing. Smoothing the analog values is recommended in case of fast signal changes (measured value changes), for example, in the level measurement of fluctuating liquids.
- Underflow Diagnostics
  Through this parameter, the underflow diagnostics is activated. If the measured value falls below the underflow range of the channel, a diagnostic interrupt is triggered.
- Overflow Diagnostics
  Through this parameter, the overflow diagnostics is activated. If the measured value exceeds the overflow range of the channel, a diagnostic interrupt is triggered.

3.5. Properties of Analog Output Modules



AI4 x 13 bits / AQ2 x 14 bits_1 [Module]

Properties | Info | Diagnostics

General | IO tags | System constants | Texts

- General
- AI 4/AQ 2
  - Analog inputs
  - Analog outputs
  - I/O addresses
  - Hardware identifier

Analog outputs

Reaction to CPU STOP: Use substitute value

Channel0

Channel address: QW96

Analog output type: Voltage

Voltage range: +/- 10 V

Substitute value for channel on a change from RUN to STOP: 0.000

☑ Enable short circuit diagnostics

☑ Enable broken wire diagnostics

☑ Enable overflow diagnostics

☑ Enable underflow diagnostics

AI4 x 13 bits / AQ2 x 14 bits_1 [Module]

Properties | Info | Diagnostics

General | IO tags | System constants | Texts

- General
- AI 4/AQ 2
  - Analog inputs
  - Analog outputs
  - I/O addresses
  - Hardware identifier

I/O addresses

Input addresses

Start address: 96

End address: 103

Organization block: --- (Automatic update)

Process image: Automatische Aktualisierung

Analog Output Modules

In STEP7, analog output modules are configured and assigned parameters in the Device configuration of the respective PLC. The settings or parameters of all modules are downloaded into the CPU. The CPU must be in the STOP state to do this. In a subsequent CPU warm restart, the CPU transfers these parameters to the relevant modules.

Parameters

For the respective module, differentiation is made between module parameters and channel parameters.

Module Parameters

- General
  Name and comment for the integrated analog outputs of the CPU.
- Reaction to CPU STOP
  - Use substitute value
    The peripheral device outputs the value previously set for the channel.
  - Keep last value
    The peripheral device retains the value last put out before STOP.

Caution !
Make sure that the system is always in safe mode in the case of "Keep last value"!

- I/O Addresses and Hardware Identifier
  The address space of the entry addresses as well as the process image is defined. The hardware identity of the device is displayed.

Channel Parameters

- Output Type
  The type of output, such as voltage, is set with this parameter. Unused outputs must be deactivated since these are otherwise also converted which would result in a longer total conversion time of the module.
- Output Range (in the picture – Voltage range)
  The output range of the selected type of output is set with this parameter.
- Broken Wire Diagnostics (in Current mode)
  With this parameter, the diagnostic Wire break is generated when there is a wire break. This diagnostic is not noticeable in the zero range.
- Short-circuit Diagnostics (in Voltage mode)
  With this parameter, a diagnostic is generated when there is a short-circuit of the output wire. This diagnostic is not noticeable in the zero range.
- Overload Diagnostics
  With this parameter, a diagnostic is generated when there is an overload.
- Substitute value
  With this parameter, a substitute value is specified which the module is to output when the CPU goes into STOP. The substitute value must be in the rated range, the over range or the under range.

3.6. Analog Value Representation and Measured Value Resolution

Bit no.		min. units		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Bit value		Dec.	Hex.	VZ	2^{14}	2^{13}	2^{12}	2^{11}	2^{10}	2^{9}	2^{8}	2^{7}	2^{6}	2^{5}	2^{4}	2^{3}	2^{2}	2^{1}	2^{0}
Reso-lution in bits + sign (VZ)	8	128	80	*	*	*	*	*	*	*	*	*	0	0	0	0	0	0	0
	9	64	40	*	*	*	*	*	*	*	*	*	*	0	0	0	0	0	0
	10	32	20	*	*	*	*	*	*	*	*	*	*	*	0	0	0	0	0
	11	16	10	*	*	*	*	*	*	*	*	*	*	*	*	0	0	0	0
	12	8	8	*	*	*	*	*	*	*	*	*	*	*	*	*	0	0	0
	13	4	4	*	*	*	*	*	*	*	*	*	*	*	*	*	*	0	0
	14	2	2	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	0
	15	1	1	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*

* = **0** or **1**

Representation

Negative analog values are represented as the two's complement.
The value is positive if bit No. 15=0 and negative if bit No.15=1.

Resolution

If the resolution of an analog module is less than 16 bits, the analog value is written into the accumulator (module result memory) left-justified. The unused less significant bit positions are filled with "0"s.

Accuracy

Resolutions of between 8 and 16 bits are possible, depending on the type of module.

3.7. Analog Value Representation of Different Measuring Ranges

Range	Voltage such as:		Current such as:		Resistance such as:		Temperature e.g. Pt100 (Standard)	
	Meas.range ± 10V	Units	Meas.range 4 to 20mA	Units	Meas.range 0 to300Ohm	Units	Meas.range -200 to+850ºC	Units
Overflow	>= 11.76	32767	>= 22.815	32767	>=352.778	32767	>= 1000.1	32767
Over range	11.7589 : 10.0004	32511 : 27649	22.810 : 20.0005	32511 : 27649	352.767 : 300.011	32511 : 27649	1000.0 : 850.1	10000 : 8501
Rated range	10.00 7.50 : -7.5 -10.00	27648 20736 : -20736 -27648	20.000 16.000 : : 4.000	27648 20736 : : 0	300.000 225.000 : : 0.000	27648 20736 : : 0	850.0 : : : -200.0	8500 : : : -2000
Under range	- 10.0004 : - 11.759	- 27649 : - 32512	3.9995 : 1.1852	- 1 : - 4864	negative values not possible	- 1 : - 4864	- 200.1 : - 243.0	- 2001 : - 2430
Underflow	<= - 11.76	- 32768	<= 1.1845	- 32768		- 32768	<= - 243.1	- 32768

Voltage, Current (Symmetrical)

Converting the symmetrical voltage or current ranges

- ± 80mV
- ± 250 mV
- ± 500 mV
- ± 1 V
- ± 2,5 V
- ± 5V
- ± 10V
- ± 3,2 mA
- ± 10 mA
- ± 20 mA

results in a rated range of -27648 to +27648.

Voltage, Current (Asymmetrical)

Converting the asymmetrical voltage or current ranges

- 0 ... 2 V
- 1 ... 5 V
- 0 ... 20 mA
- 4 ... 20 mA

results in a rated range of 0 to +27648.

Resistance

Converting the resistance ranges

- 0 to 150 Ohm
- 0 to 300 Ohm
- 0 to 600 Ohm

results in a rated range of 0 to +27648.

Temperature

Temperatures are measured with resistance thermometers or thermocouples. Converting results in a rated range of ten times the temperature range:

Sensor:	Temperature range:	Rated range when converted:
• Pt 100	-200 to + 850 ºC	-2000 to + 8500
• Ni 100	-60 to + 250 ºC	-600 to + 2500
• Thermocouple Type K	-270 to + 1372 ºC	-2700 to + 13720
• Thermocouple Type N	-270 to + 1300 ºC	-2700 to + 13000
• Thermocouple Type J	-210 to + 1200 ºC	-2100 to + 12000
• Thermocouple Type E	-270 to + 1000 ºC	-2700 to + 10000

3.8. Analog Value Representation for the Analog Outputs

Range	Units	Voltage Output ranges: 0 to 10V	 1 to 5V	 ± 10V	Current Output ranges: 0 to 20mA	 4 to 20mA	 ± 20mA
Overflow	>=32767	0	0	0	0	0	0
Over range	32511 : 27649	11.7589 : 10.0004	5.8794 : 5.0002	11.7589 : 10.0004	23.515 : 20.0007	22.81 : 20.005	23.515 : 20.0007
Rated range	27648 : 0	10.0000 : 0	5.0000 : 1.0000	10.0000 : 0	20.000 : 0	20.000 : 4.000	20.000 : 0
	: - 6912	0	0.9999	:	0	3.9995	:
	- 6913		0	:		0	:
	: : - 27648		0	: -10.0000		0	: -20.000
Under range	- 27649 : - 32512			- 10.0004 : - 11.7589			- 20.007 : - 23.515
Underflow	<=- 32513			0			0

Voltage, Current (Symmetrical)

For symmetrical voltage or current ranges, a rated range of -27648 to +27648 is converted to:

- ± 10V
- ± 20mA.

Voltage, Current (Asymmetrical)

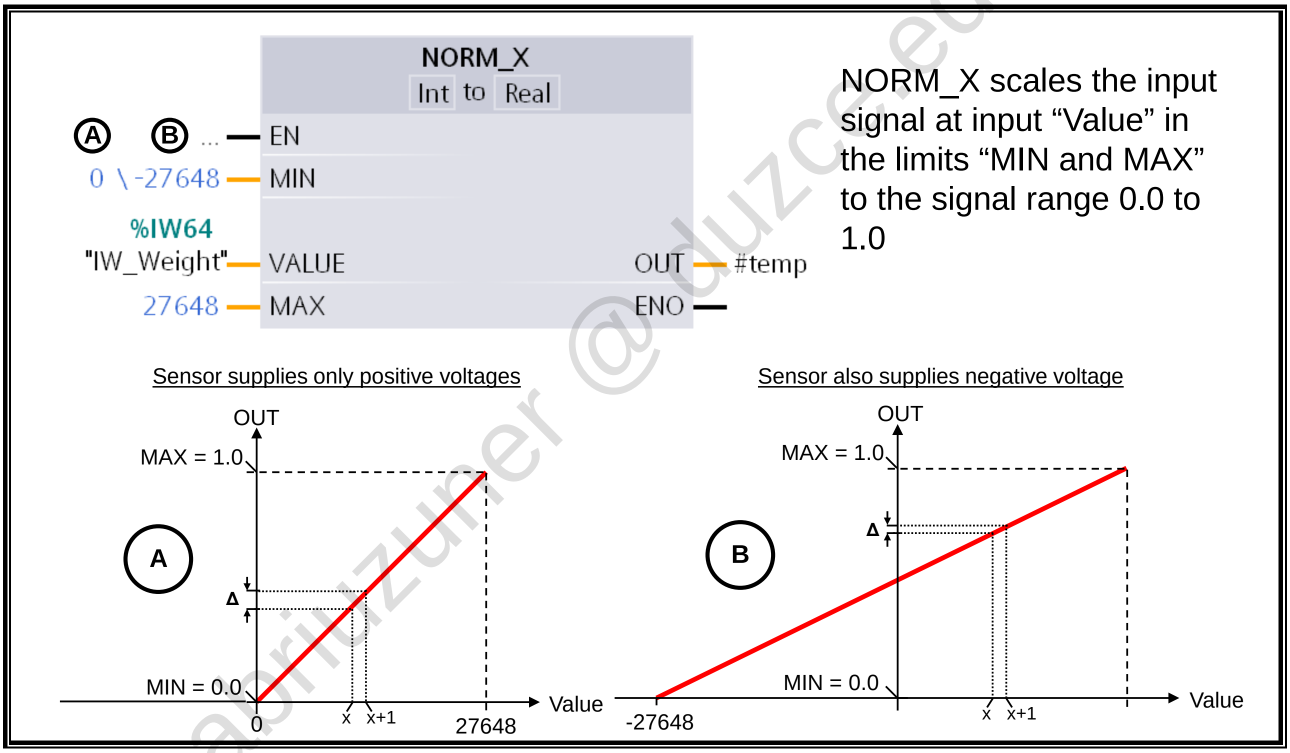
For asymmetrical voltage or current ranges, a rated range of 0 to +27648 is converted to:

- 0 to 10V
- 1 to 5V
- 0 to 20mA
- 4 to 20mA.

Overflow

If the value to be converted reaches the overflow range, the analog output module is disabled (0V, 0mA).

3.9. Scaling Analog Inputs with NORM_X and SCALE_X (1)



NORM_X scales the input signal at input "Value" in the limits "MIN and MAX" to the signal range 0.0 to 1.0

Norm_X

The analog module converts the voltage range of -10V to +10V into the value range of -27648 to +27648. The "Normalize" instruction scales a value by mapping it to a linear scale. You can use the MIN and MAX parameters to define the limits of a value range that is applied to the scale. Depending on the position of this value to be scaled in the value range, the result is calculated and stored as a floating-point number. If the value to be scaled is equal to the value at the MIN input, the instruction returns the value "0.0" as the result. If the value to be scaled is equal to the value at the MAX input, the instruction supplies the result "1.0".

Resolution

In example **B,** the measurement occurs with twice the resolution or with half as much measuring tolerance **Δ**, since the measured value is mapped to the greater units range of -27648 to +27648.

Data Types

- The parameters on the input-side can be one of the following data types:

  SINT, INT, DINT, USINT, UINT, UDINT or REAL
- The parameter OUT can be one of the following data types: REAL or LREAL

Sensor

In the following it is assumed that a sensor is used which has a measuring range of 0 to 10V (case A) or -10V to 10V (case B).

Example **A** shows the scaling when a sensor is used that supplies a measured voltage of 0V as the smallest measured value and +10V for the maximum measured value. Example **B** shows the scaling when a sensor is used which supplies -10V as the smallest measured value and 10V as the largest measured value.

Parameters

- VALUE: Value which is scaled.
- MIN: Lower limit of the value range
- MAX: Upper limit of the value range
- OUT: Scaled signal 0.0 to 1.0

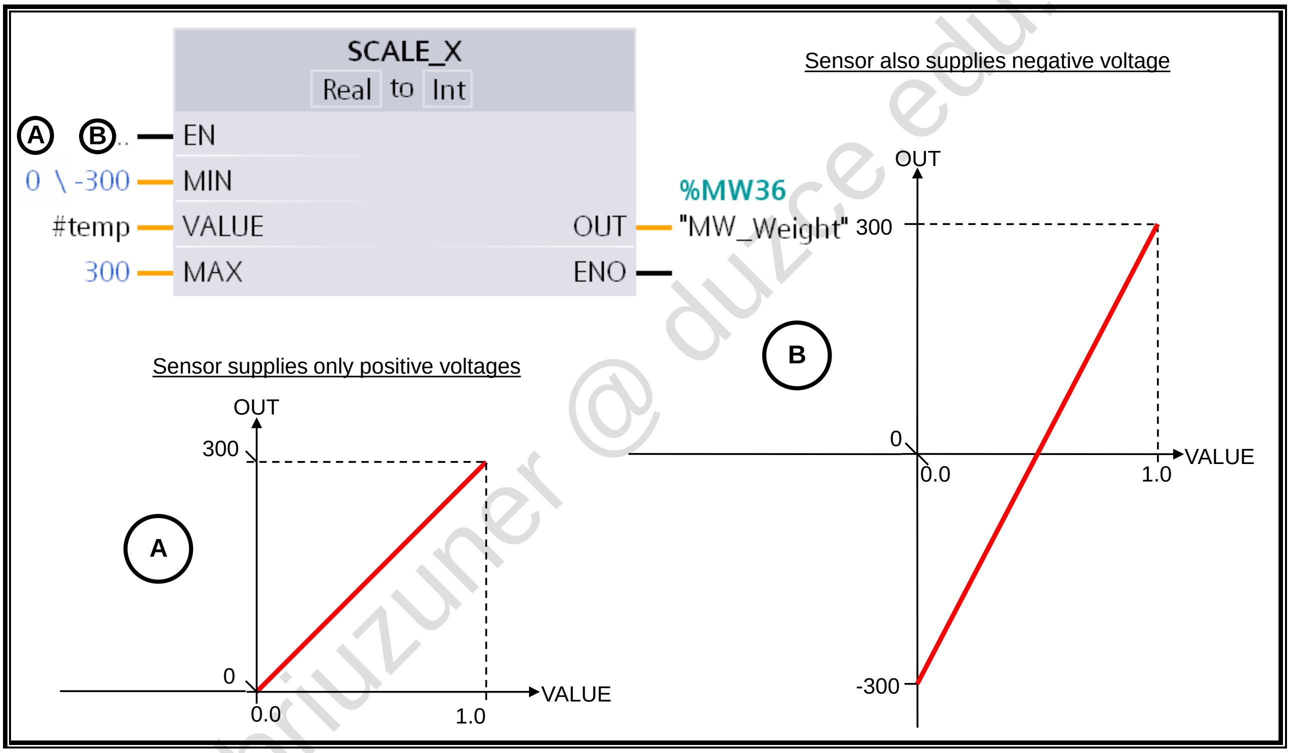
EN / ENO

The "Normalize" instruction is only executed if the signal state is "1" at the enable input EN. In this case, the enable output ENO has signal state "1".

The enable output ENO returns signal state "0" if one of the following conditions applies:

- The EN input has signal state "0".
- The value at the MIN input is greater than or equal to the value at the MAX input.
- The value of a specified floating-point number is outside of the range of the scaled numbers according to IEEE-754.
- The value at input VALUE is NaN (Not a number = result of an invalid arithmetic operation).

3.10. Scaling Analog Inputs with NORM_X and SCALE_X (2)



SCALE_X

The "Scale" instruction scales the value at the VALUE input linearly by mapping it to a specified value range. When the "Scale" instruction is executed, the floating-point value at the VALUE input is scaled to the value range which was defined by the MIN and MAX parameters. The result of the scaling is an integer which is stored at the OUT output.

Example

In the example shown, the value at the VALUE input is scaled within the limits 0 to 300 for case A. In case B, VALUE is scaled to the limits -300 to 300.

☞ The VALUE input may only be within the limits 0.0 to 1.0!

Parameters

- VALUE: Value which is scaled.
- MIN: Lower limit of the value range
- MAX: Upper limit of the value range
- OUT: Result of scaling.

EN / ENO

The "Scale" instruction is only executed if the signal state is "1" at the enable input EN. In this case, the enable output ENO also has signal state "1".

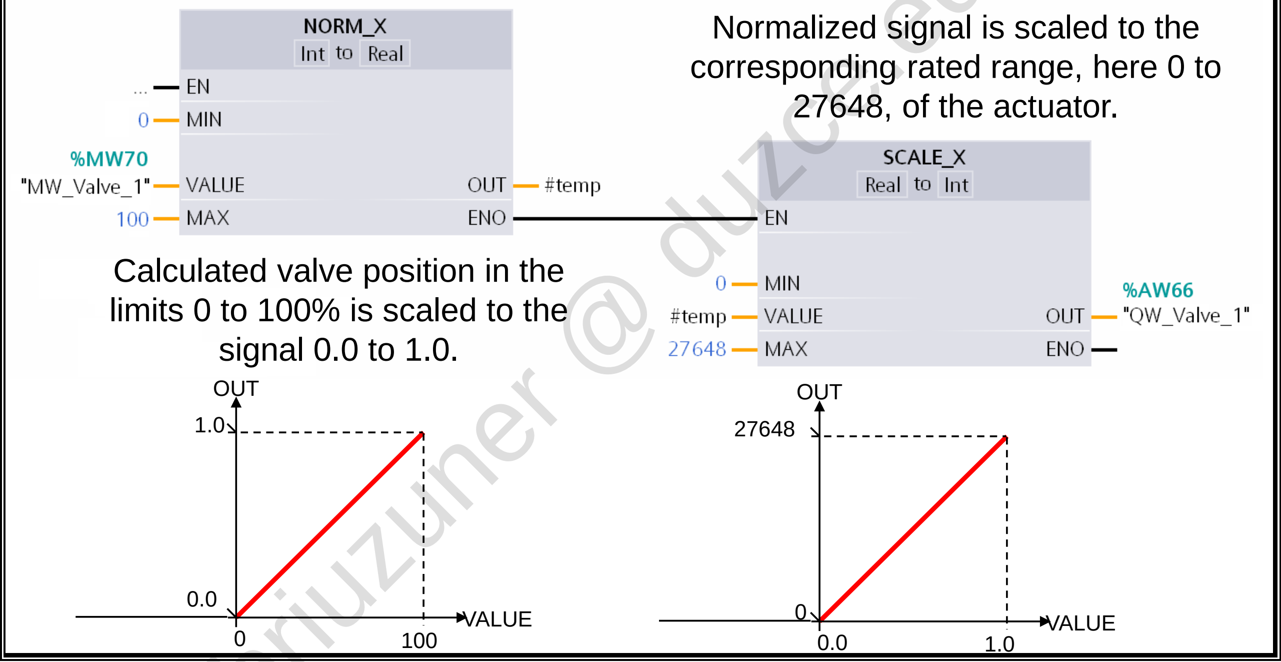
The enable output ENO returns signal state "0" if one of the following conditions applies:

- The EN input has signal state "0".

- The value at the MIN input is greater than or equal to the value at the MAX input.
- The value of a specified floating-point number is outside of the range of the scaled numbers according to IEEE-754.
- An overflow occurs.
- The value at input VALUE is NaN (Not a number = result of an invalid arithmetic operation).

3.11. Controlling Analog Outputs with NORM_X and SCALE_X

Example: Control valve 0 to 100%



Normalized signal is scaled to the corresponding rated range, here 0 to 27648, of the actuator.

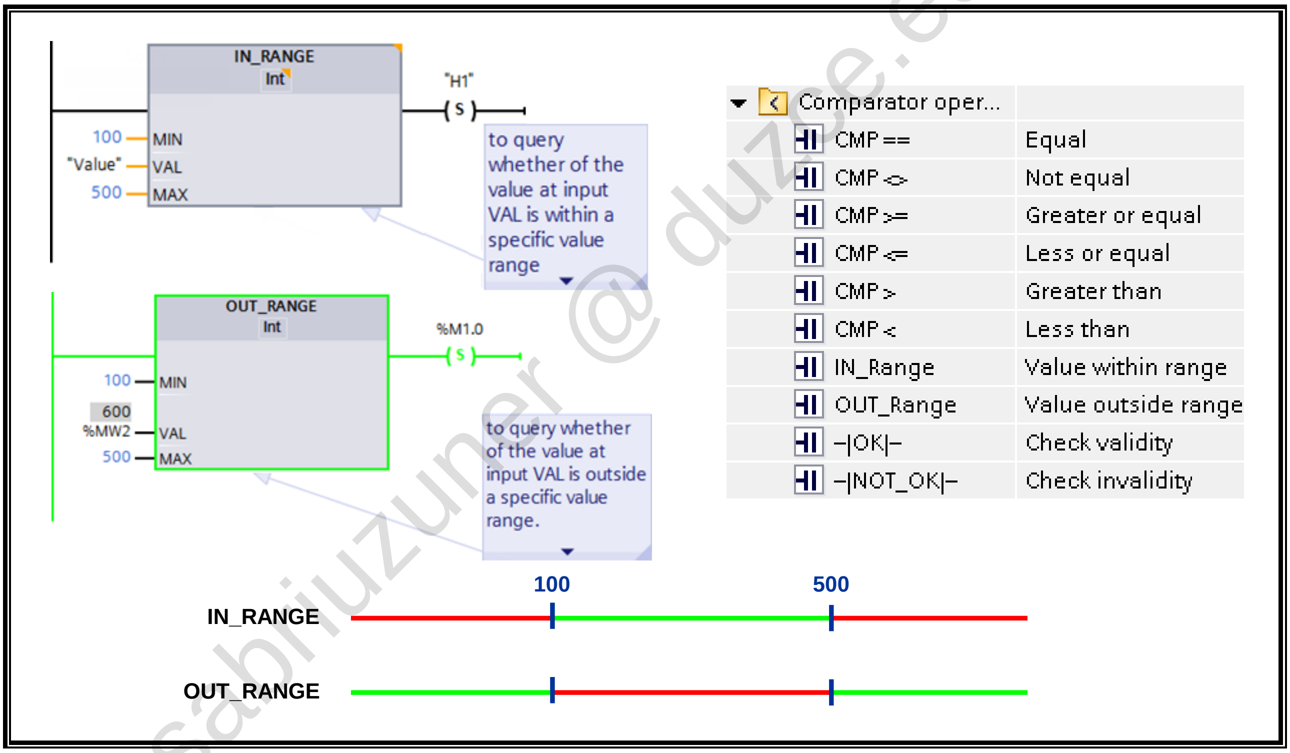
Calculated valve position in the limits 0 to 100% is scaled to the signal 0.0 to 1.0.

Controlling Analog Outputs (Example)

An analog value (valve position) calculated by the user program in the range 0 to 100% is converted to the range 0 to +27648 through the combination of NORM_X and SCALE_X. In outputting the unscaled value to an analog output module, it will control the analog actuator (for example, a servo valve) with, for example, 0V to +10V (depending on the output range set).

The example shows the scaling for an actuator that is to be controlled with the value 0 (0V or 0mA) when the program value is 0%, and with the maximum value (for example, +10V or 20mA) when it is 100%.

3.12. Comparator Operations: IN_RANGE and OUT_RANGE



to query whether the value at input VAL is within a specific value range

to query whether of the value at input VAL is outside a specific value range.

Comparator oper...	
CMP ==	Equal
CMP <>	Not equal
CMP >=	Greater or equal
CMP <=	Less or equal
CMP >	Greater than
CMP <	Less than
IN_Range	Value within range
OUT_Range	Value outside range
–|OK|–	Check validity
–|NOT_OK|–	Check invalidity

IN_RANGE

With the "Value within range" instruction you can query whether the value at the VAL input is within a specific value range. You define the limits of the value range with the parameters MIN and MAX. In executing the query, the "Value within range" instruction compares the value at the VAL input with the values of the parameters MIN and MAX and assigns the result to the box output. If the value at the VAL input fulfills the comparison MIN <= VAL <= MAX, the box output has signal state "1". When the comparison is not fulfilled, the box output has signal state "0".

The compare function is only executed if the values to be compared are of the same data type and the box output is used.

OUT_RANGE

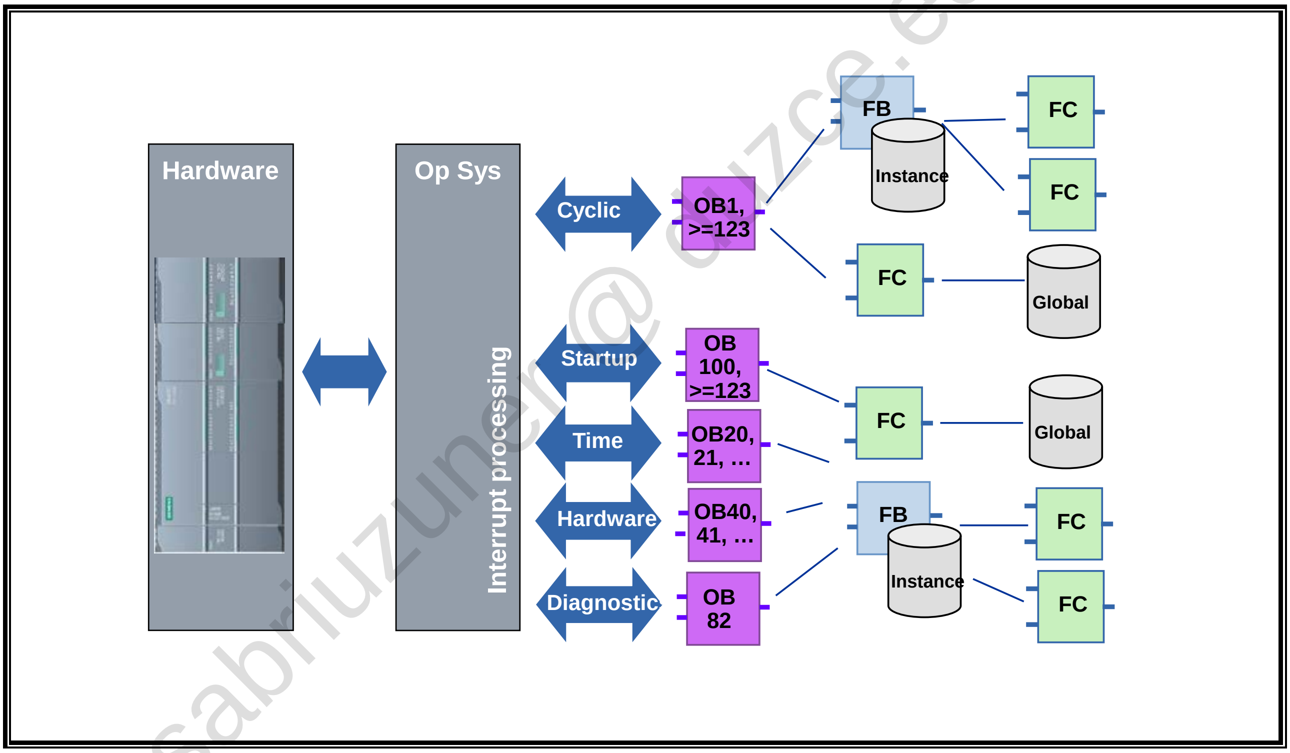
With the "Value outside range" instruction you can query whether the value at the VAL input is outside of a specific value range. The limits of the value range are defined through the parameters MIN and MAX. In executing the query, the "Value outside range" instruction compares the value at the VAL input with the values of the parameters MIN and MAX and assigns the result to the box output. If the value at the VAL input fulfills the comparison MIN > VAL or VAL > MAX, the box output has signal state "1". When the comparison is not fulfilled, the box output has signal state "0".

The compare function is only executed if the values to be compared are of the same data type and the box output is used.

OK / NOT_OK

The OK (NOT_OK) instruction checks whether the value of the variable specified through the box corresponds to a valid REAL or LREAL. If this is the case, the box supplies RLO '1' at its output.

3.13. Organization Blocks: Overview



OBs

Organization blocks form the interface between the user program and the CPU's operating system.

Organization blocks are called exclusively by the operating system. There are various start events (time interrupt, hardware interrupt, ...see picture).

Startup Program

After a restart, a startup program is executed. In the startup OBs you can, for example, carry out a pre-assignment of communication connections.

Cyclic Program Execution

The program stored in the cyclic OB is executed cyclically, that is, after it is executed completely it is executed again. With this cyclic program execution, the reaction time results from the execution time for the CPU's operating system and the sum of the command runtimes of all executed instructions. The reaction time, that is, how fast an output can be switched in relation to an input signal, amounts to a minimum of one time and a maximum of two times the cycle time.

Periodic Program Execution

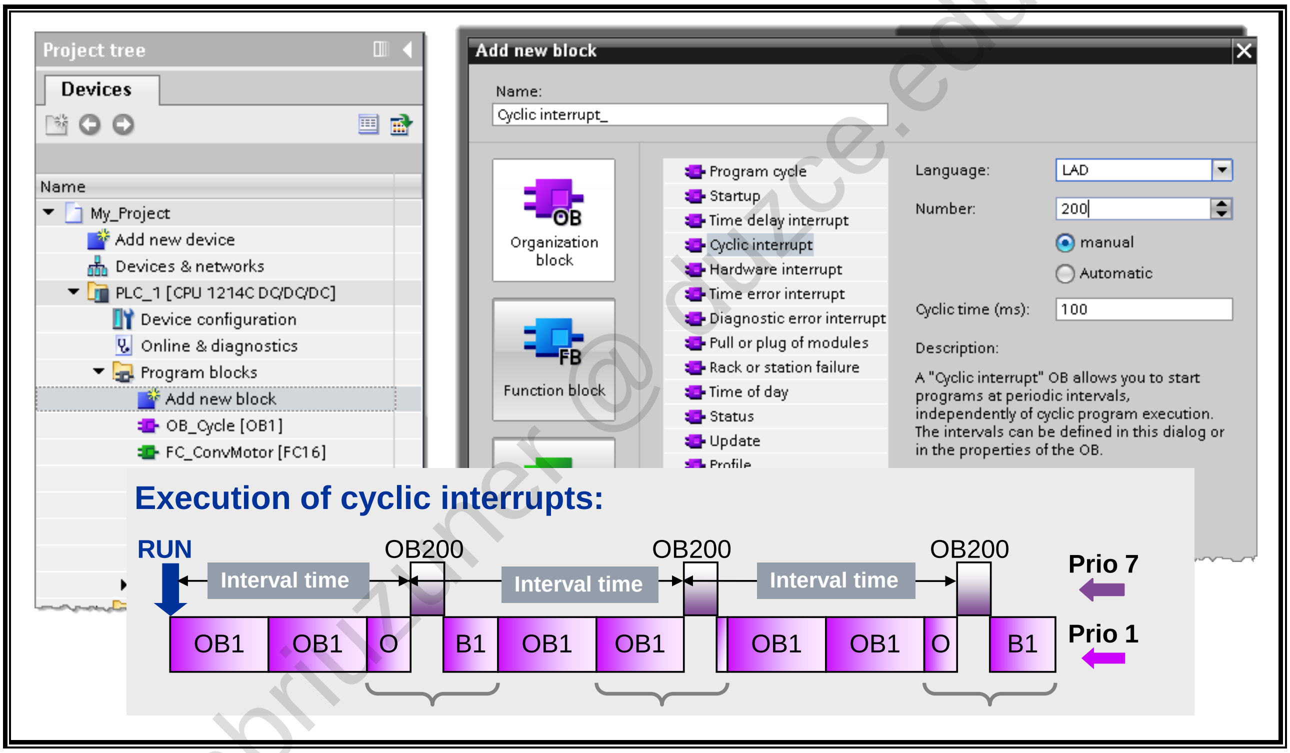
This makes it possible to interrupt the cyclic program execution at fixed intervals. With the cyclic interrupts, an organization block (for example OB235) is executed after an adjustable time base (for example, every 100ms) has expired. In these blocks, closed-loop control blocks with their sampling time, for example, are called.

Event-driven Program Execution

In order to be able to react quickly to a process event, the hardware interrupt can be used. After an event occurs, the cycle is immediately interrupted and an interrupt program is executed.

With time-delay interrupts, a freely definable event can be reacted to with a time-delay; with an error OB, the user can influence the behavior of the controller in case there is an error.

3.13.1. Cyclic Interrupts



Name: Cyclic interrupt_

Language: LAD

Number: 200

● manual

○ Automatic

Cyclic time (ms): 100

Description:

A "Cyclic interrupt" OB allows you to start programs at periodic intervals, independently of cyclic program execution. The intervals can be defined in this dialog or in the properties of the OB.

Description

Cyclic interrupt OBs are used to start programs in equidistant intervals regardless of the cyclic program execution.

The time interval defines the intervals in which the cyclic interrupt OB is started and is an integral multiple of the basic clock of 1ms. The phase offset is the time by which the start time is shifted vis-à-vis the basic clock. When several cyclic interrupt OBs are used, you can use this offset to prevent a simultaneous starting time should the time intervals of the cyclic interrupt OBs have a common multiple. You can specify a period between 1 ms and 60000 ms as the time interval.

Note

The runtime of every cyclic interrupt OB must be considerably less than its time interval. If a cyclic interrupt OB is not yet completed but is once again pending for processing because the clock has run out, the time error interrupt OB is started. After that, the error-causing cyclic interrupt is carried out or discarded.

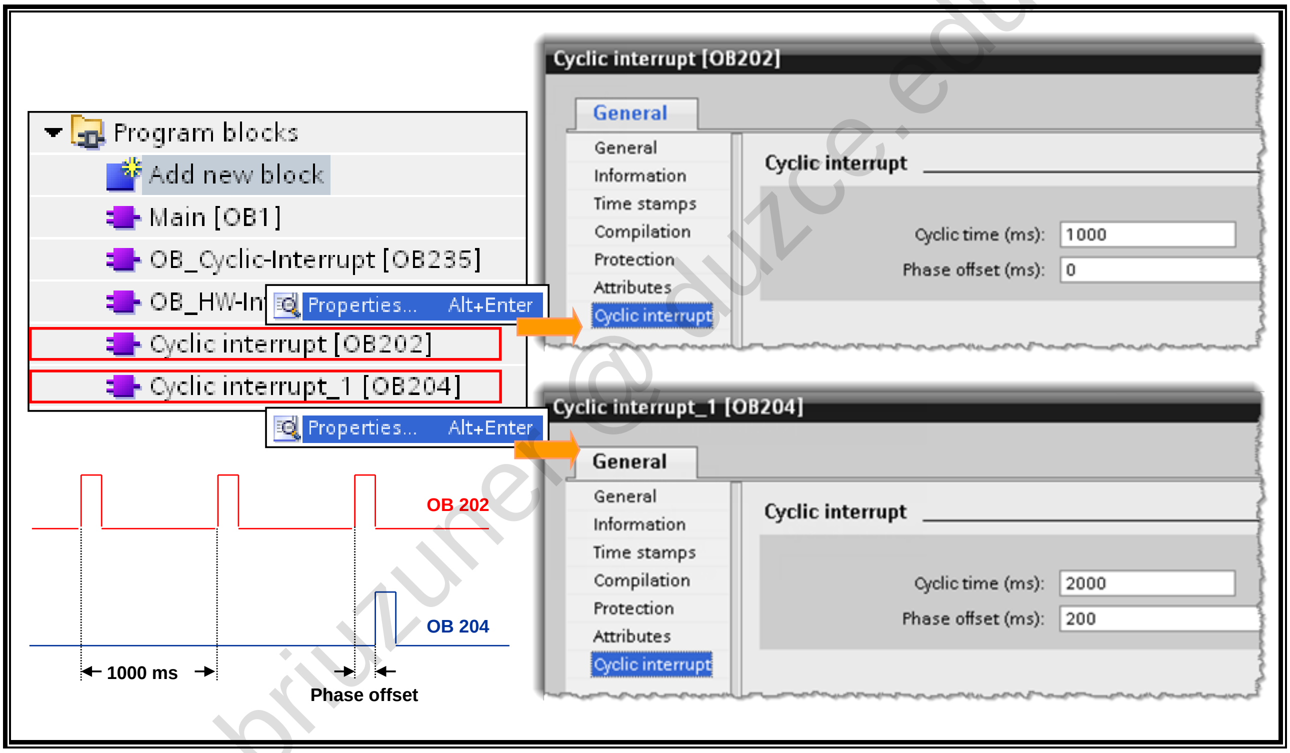
Example

You have inserted two cyclic interrupt OBs into your program:

- cyclic interrupt OB1
- cyclic interrupt OB2

For cyclic interrupt OB1, you have set a time interval of 20 ms and for cyclic interrupt OB2, a time interval of 100 ms. After the time interval of 100 ms has run out, cyclic interrupt OB1 reaches its starting time for the fifth time, cyclic interrupt OB2 for the first time. In order to nevertheless process the cyclic interrupt OBs with a time delay, enter a phase offset for one of the two cyclic interrupt OBs.

3.13.2. Phase Offsets for Cyclic Interrupts



Cyclic interrupt [OB202]

Cyclic interrupt

Cyclic time (ms): 1000

Phase offset (ms): 0

Cyclic interrupt_1 [OB204]

Cyclic interrupt

Cyclic time (ms): 2000

Phase offset (ms): 200

Phase Offset

With cyclic interrupt OBs, you can start programs at regular (equidistant) intervals. For this, you have to enter a time interval and a phase offset for every cyclic interrupt OB used.

Note

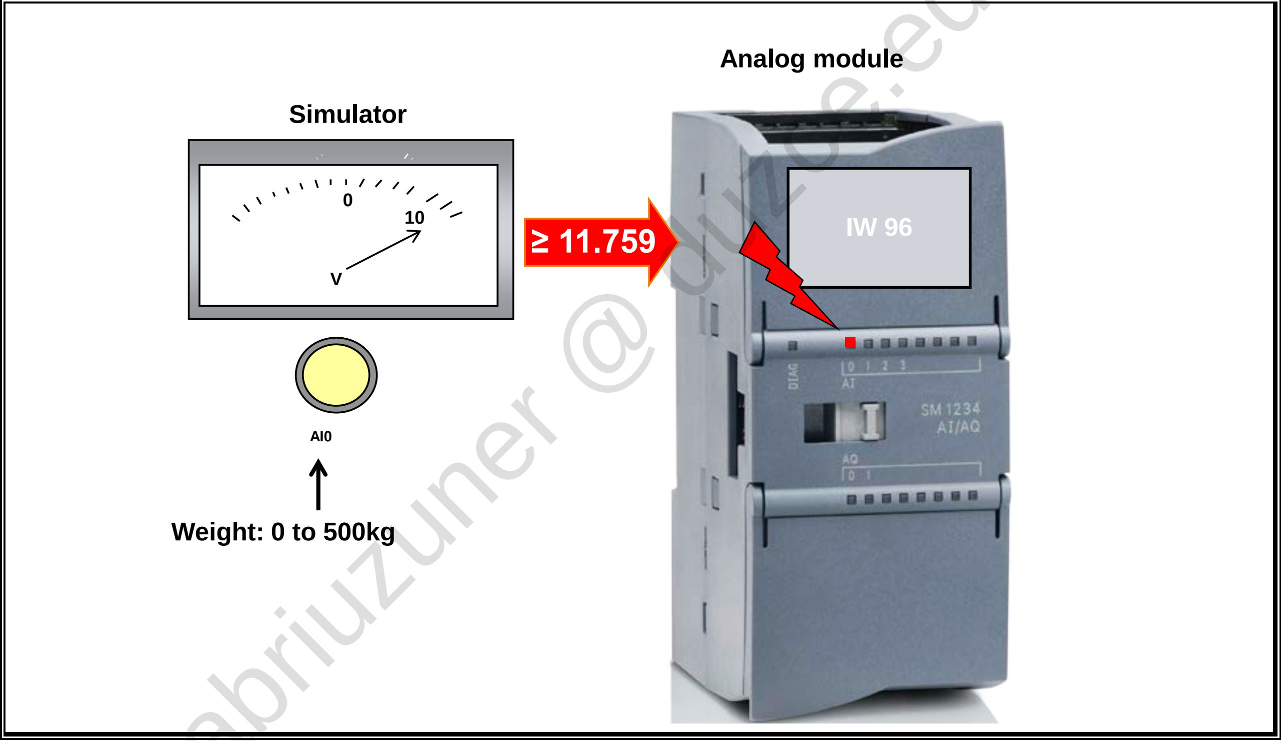
When you assign parameters to several cyclic interrupt OBs, you must give each cyclic interrupt OB a different cyclic time or phase offset in order to prevent a simultaneous execution or a queue. When a cyclic interrupt OB is created, the cyclic time of 100 and the phase offset of 0 are entered as the start value.

Procedure

To enter a time interval (cyclic time) and a phase offset for a cyclic interrupt OB, please proceed as follows:

- in the Project tree, open the folder "Program blocks".
- right-click on an existing cyclic interrupt OB.
- in the context menu select the command "Properties".
- the dialog "<Name of the cyclic interrupt OB>" is opened.
- in the area tree, click on the group "Cyclic interrupt".
- the input fields for the time interval (cyclic time) and the phase offset are displayed.
- enter the time interval and the phase offset.
- confirm the entries with "OK".

3.14. Task Description: Fault Evaluation on the Analog Channel

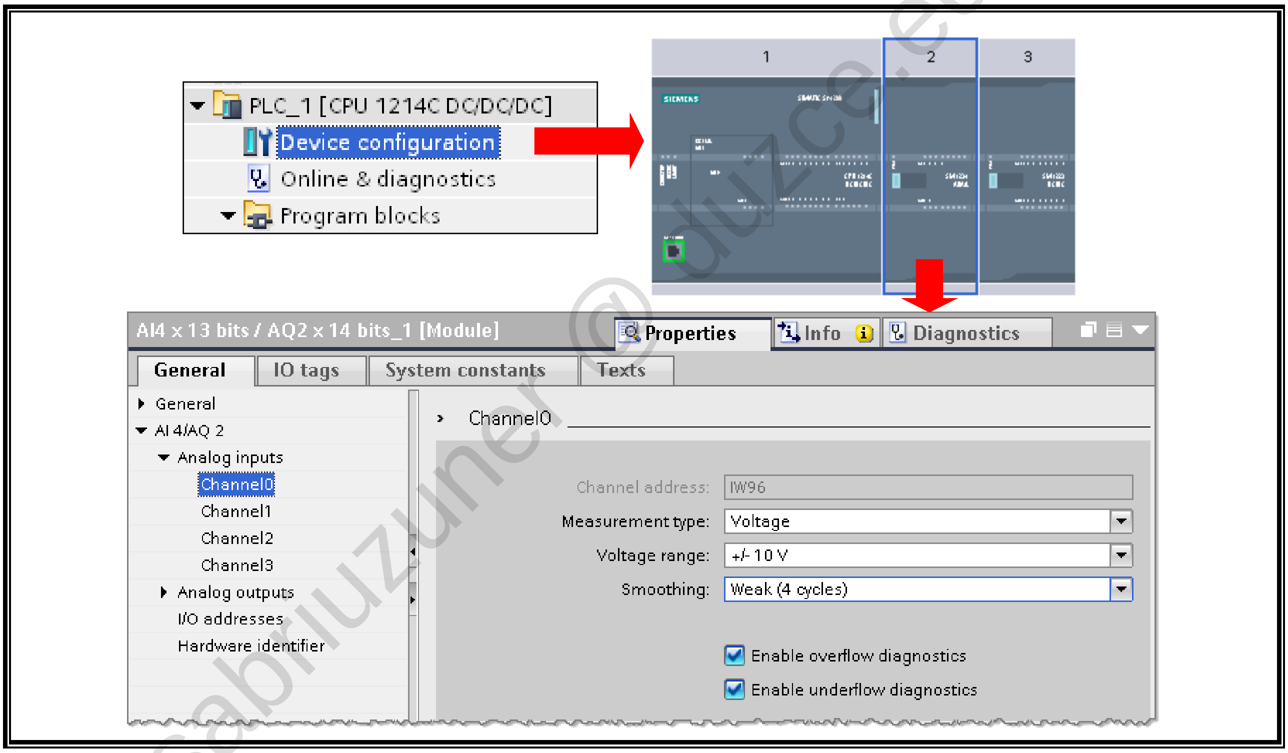


Task Description

You are to provoke a channel fault on analog channel 0 of the AI4/AO2 module and then evaluate it. For this, the measuring range of the analog input is first to be set to ± 5V and then you are to set an input voltage which is too high on the simulator potentiometer.

You are to investigate the fault condition which occurs using the STEP7 online functions.

3.14.1. Exercise 1: Parameterizing the Analog Module SM 1234



Channel0

Channel address: IW96

Measurement type: Voltage

Voltage range: +/- 10 V

Smoothing: Weak (4 cycles)

☑ Enable overflow diagnostics

☑ Enable underflow diagnostics

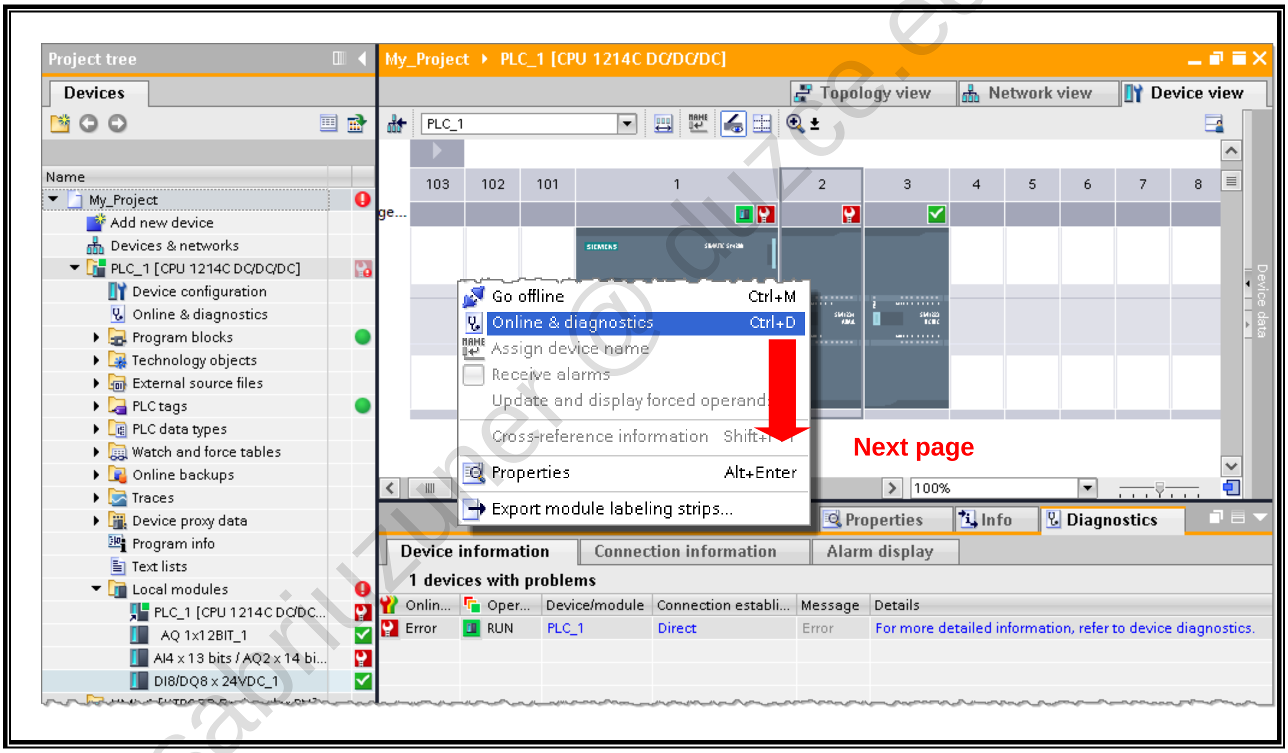
Task

Parameterize the analog module SM 1234.

What to Do

Make the settings for the analog module as shown in the picture.

3.14.2. Exercise 2: Hardware Diagnostics for Diagnostic Interrupt



1 devices with problems

Onlin...	Oper...	Device/module	Connection establi...	Message	Details
Error	RUN	PLC_1	Direct	Error	For more detailed information, refer to device diagnostics.

Task

After you have assigned parameters to your analog module in the previous exercise and have activated the diagnostics interrupt, you are now to initiate a diagnostic interrupt by knowingly setting the voltage too high.

After the CPU signals an ERROR because the voltage is too high at the input of the analog module, you are to localize the "error" that occurred by using a simple online connection and, in the next step, read out detailed information from the CPU (see picture).

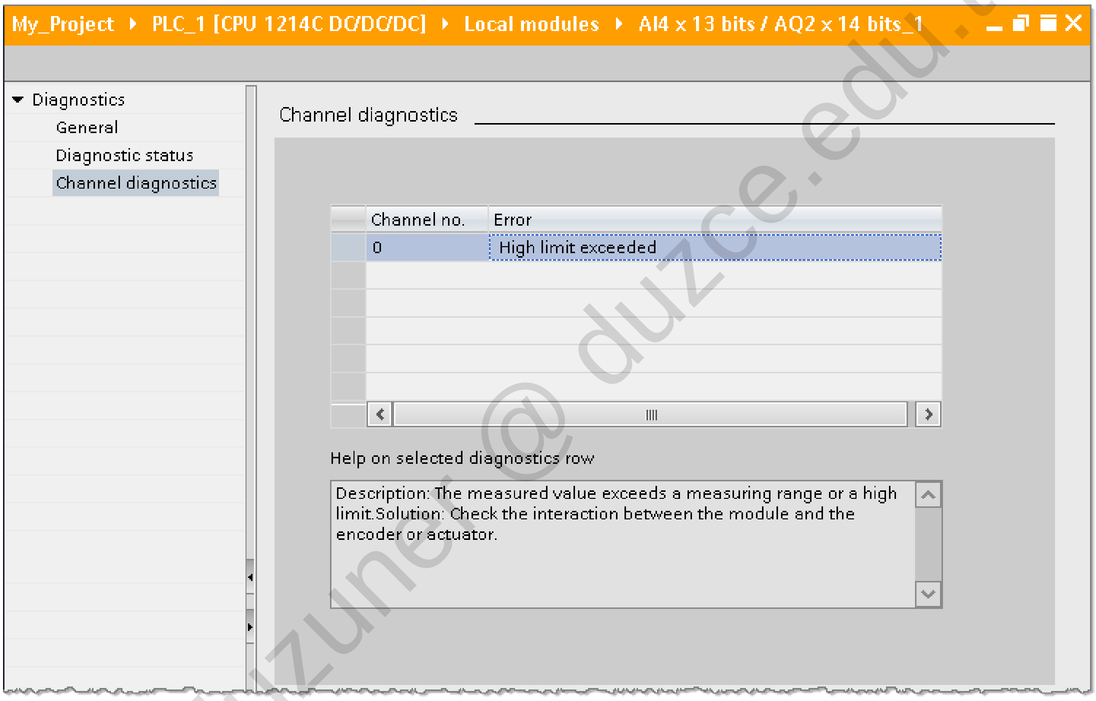
What to Do

1. On the simulator, set a voltage which is either too low or too high (voltage < -11.759V or voltage > +11.759V)
2. Establish an online connection to the CPU

   My_Project → select Station → Go online
3. Open the list of *Local modules*

   My_Project → PLC_1 → Local modules
4. Open the faulty analog module
   Double-click on the module
5. In the Inspector window, display the "Diagnostics" tab and click the link in "Details". → The diagnostics buffer of the CPU opens.

☞ The **Diagnostic Interrupt OB 82** can be programmed to give you detailed information about the error event. (evaluation of the OB 82 start information).



My_Project ▸ PLC_1 [CPU 1214C DC/DC/DC] ▸ Local modules ▸ AI4 x 13 bits / AQ2 x 14 bits_1
Diagnostics
General
Diagnostic status
Channel diagnostics

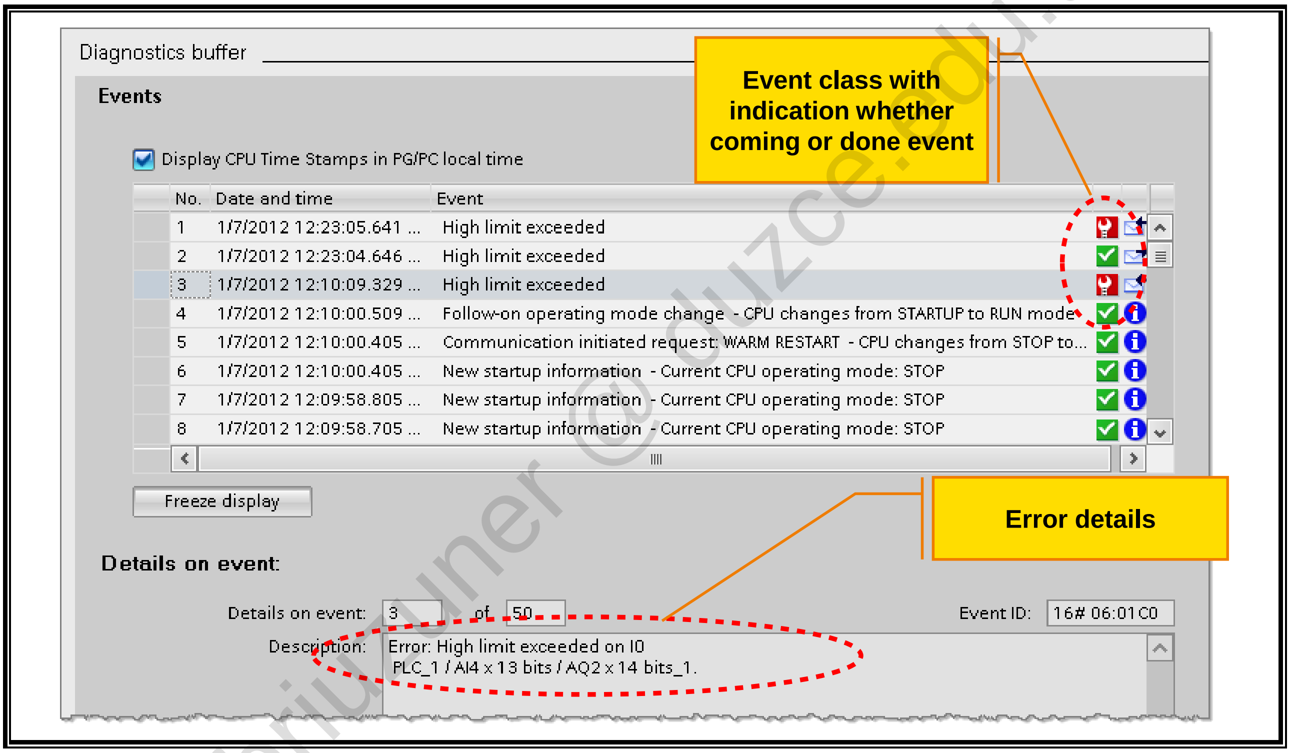
Channel diagnostics

Channel no.	Error
0	High limit exceeded

Help on selected diagnostics row

Description: The measured value exceeds a measuring range or a high limit.Solution: Check the interaction between the module and the encoder or actuator.

3.14.3. Exercise 3: Evaluating the Diagnostics Buffer of the CPU



Diagnostics buffer

Events

☑ Display CPU Time Stamps in PG/PC local time

No.	Date and time	Event	
1	1/7/2012 12:23:05.641 ...	High limit exceeded	🟥 ✉
2	1/7/2012 12:23:04.646 ...	High limit exceeded	✓ ✉
3	1/7/2012 12:10:09.329 ...	High limit exceeded	🟥 ✉
4	1/7/2012 12:10:00.509 ...	Follow-on operating mode change - CPU changes from STARTUP to RUN mode	✓ ℹ
5	1/7/2012 12:10:00.405 ...	Communication initiated request: WARM RESTART - CPU changes from STOP to...	✓ ℹ
6	1/7/2012 12:10:00.405 ...	New startup information - Current CPU operating mode: STOP	✓ ℹ
7	1/7/2012 12:09:58.805 ...	New startup information - Current CPU operating mode: STOP	✓ ℹ
8	1/7/2012 12:09:58.705 ...	New startup information - Current CPU operating mode: STOP	✓ ℹ

Details on event:

Details on event: 3 of 50

Event ID: 16# 06:01C0

Description: Error: High limit exceeded on I0
PLC_1 / AI4 x 13 bits / AQ2 x 14 bits_1.

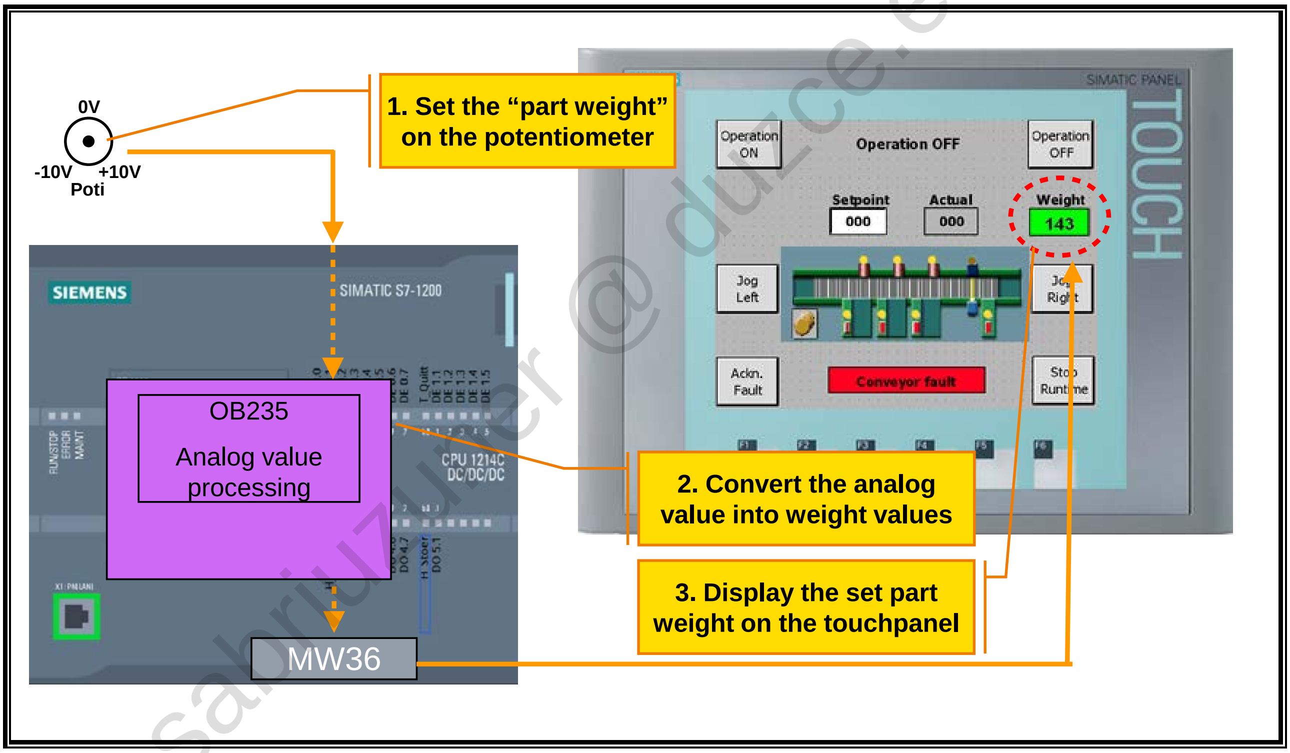
Event class with indication whether coming or done event

Error details

Task

Evaluate the information highlighted in the picture.

3.15. Task Description: Converting the Analog Value and Outputting It on the Touchpanel



Task Description

An analog value processing is to be programmed in a cyclic interrupt (OB235). The converted analog value (part weight: 0 to 500kg) is stored in memory word MW36 ("MW_Weight") and is then to be displayed on the touchpanel.

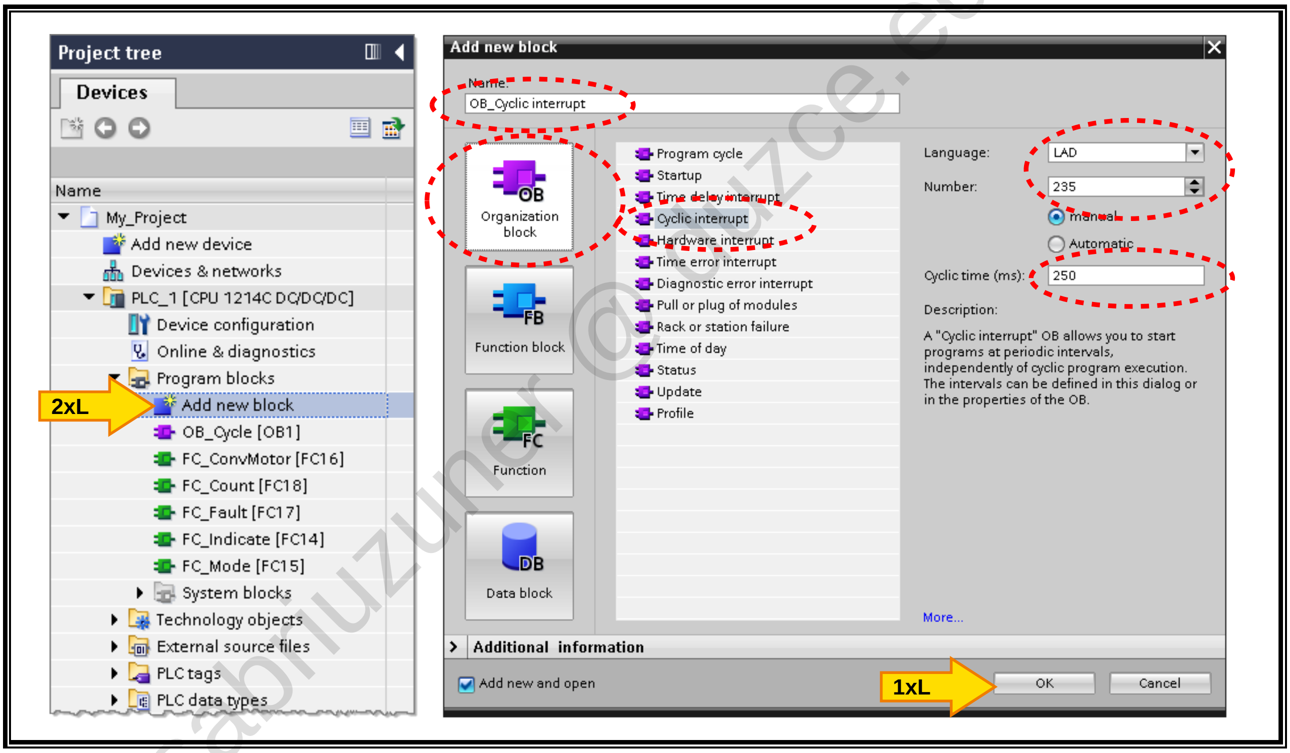
Furthermore, the part weight is to be checked for validity and the result is to be assigned to bit memory M35.0 ("M_Weight_OK"):

- 100kg ≤ MW36 ≤ 400kg → M35.0 ("M_Weight_OK") = TRUE
- otherwise → M35.0 ("M_Weight_OK") = FALSE

The bit memory is already linked to the I/O field "Act. Weight" on the touchpanel and influences its background color.

If bit memory M35.0 delivers the value FALSE, the bay indicator lights on the conveyor must remain dark and no new transport sequence can be started. The lock outs in FC14 and FC16 necessary for this must be programmed by you.

3.15.1. Exercise 4: Inserting "OB_Cyclic interrupt" (OB235)



Name: OB_Cyclic interrupt

Language: LAD

Number: 235

● manual

○ Automatic

Cyclic time (ms): 250

Description:

A "Cyclic interrupt" OB allows you to start programs at periodic intervals, independently of cyclic program execution. The intervals can be defined in this dialog or in the properties of the OB.

☑ Add new and open

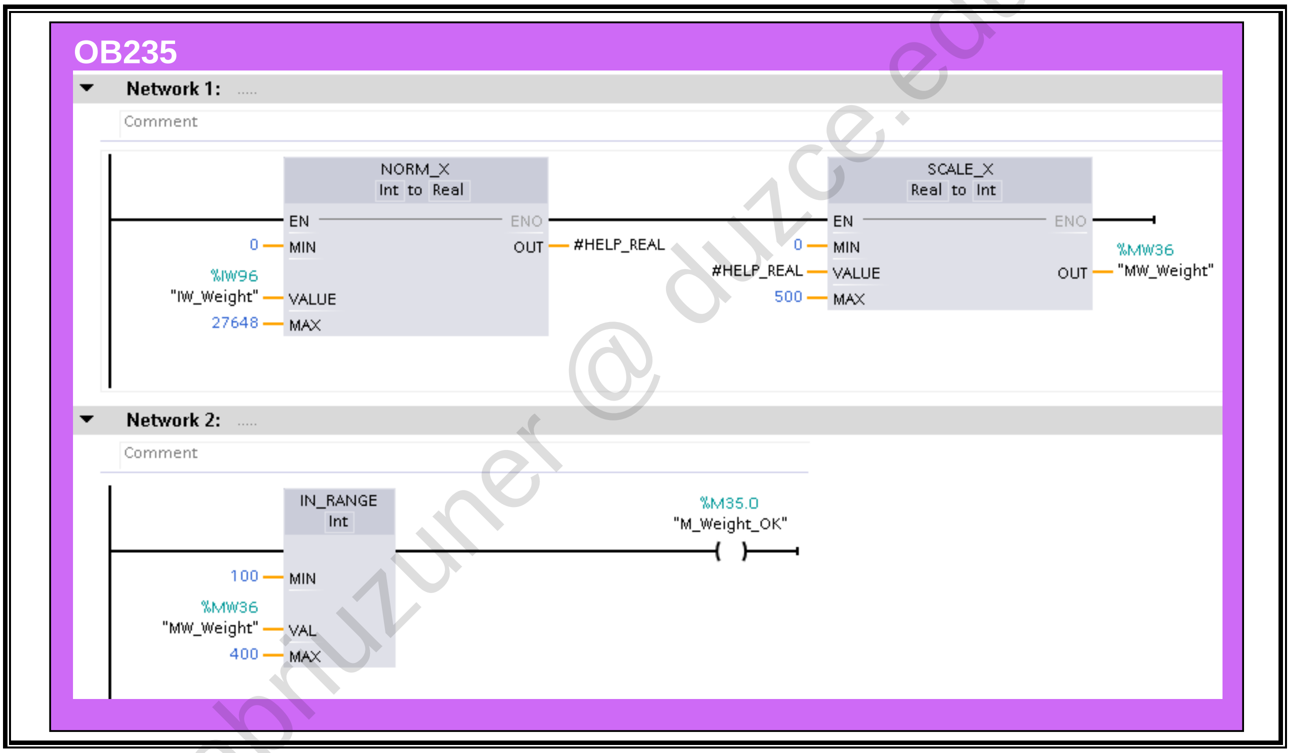
Task

Insert the Cyclic interrupt OB 235 into your user program.

What to Do

1. Open the "Add new block" dialog
   My_Project → PLC_1 → Program blocks → Double-click on "Add new block"
2. Select the OB type "Cyclic interrupt".
3. Assign the block name "OB_Cyclic interrupt".
4. Assign the manual block number 235.
5. Set a cyclic time of 250ms.
6. Click on OK.

3.15.2. Exercise 5: Programming Analog Value Processing and Lock Outs (FC14, FC16)



Task

In OB235, program the analog value processing represented in the picture and then the lock outs dependent on the part weight in FC14 and FC16.

What to Do

1. Activate the instructions NORM_X and SCALE_X and assign the parameters as shown in the picture.

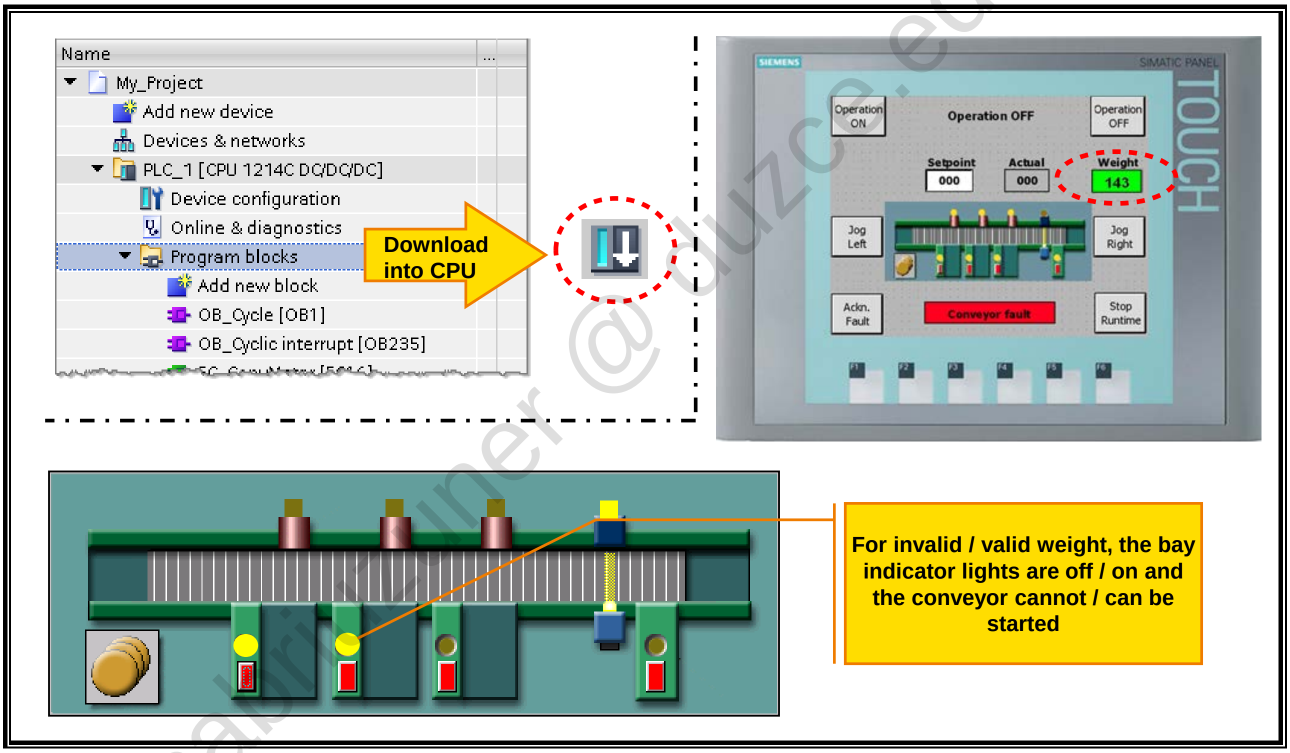
#HELP_REAL is a local, temporary tag variable of the type REAL

PLC tags are already created for the tag variables `"IW_Weight"` and `"MW_Weight"`.

2. Program the comparison to find out whether the weight is within the limits 100 to 400 kg. For this, use the instruction "IN_Range" and assign the RLO to bit memory M35.0.

3. Insert bit memory M35.0 ("M_Weight_OK") as a lock out in the correct locations in "FC_Indicate" and "FC_ConvMotor".

3.15.3. Exercise 6: Downloading Blocks into the CPU and Testing the Display on the Touchpanel



Download into CPU

For invalid / valid weight, the bay indicator lights are off / on and the conveyor cannot / can be started

Task

Check the functions you previously programmed.

What to Do

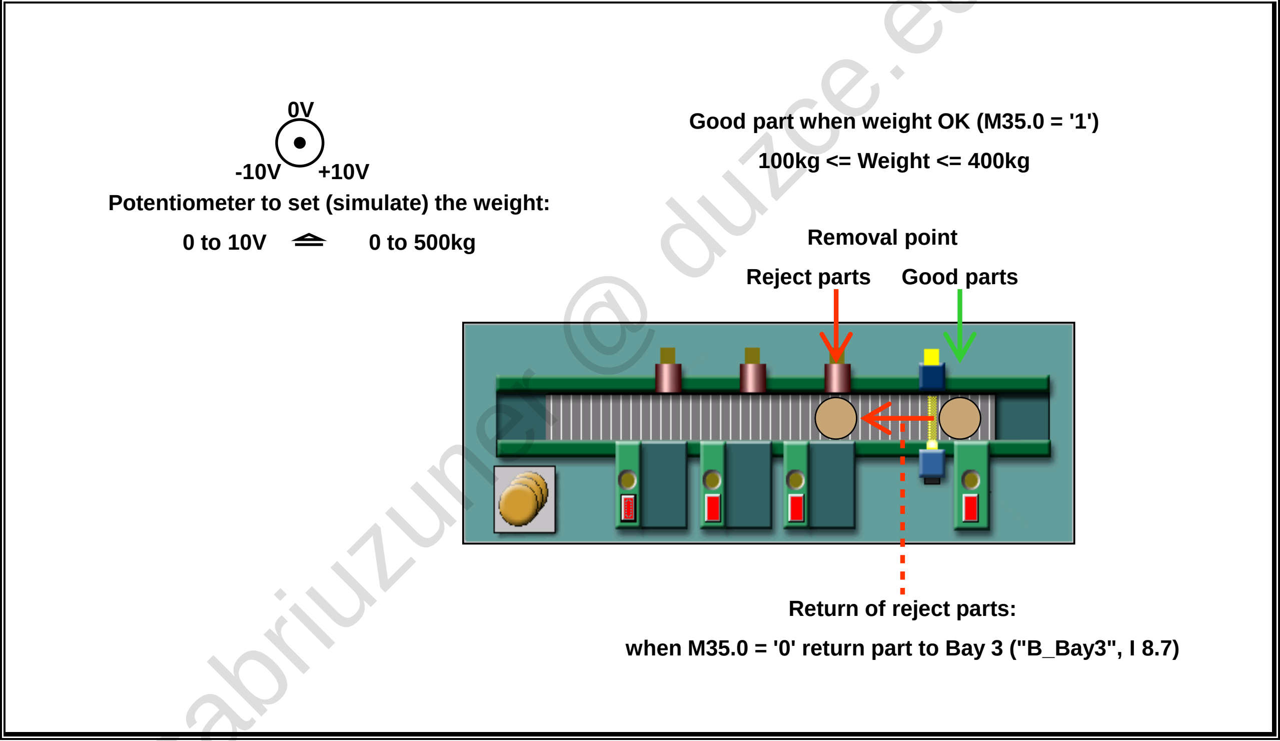
1. Download all modified blocks into the CPU.
   Right-click on PLC_1 → Download *to device* → *Software*
2. On the touchpanel, check whether the weight value is displayed correctly and whether the background color changes when there is an invalid weight.
3. Switch the operation on and enter a setpoint quantity.
4. Set an invalid weight and check whether the bay indicator lights on the conveyor model remain dark and whether the conveyor motor can no longer be started.

3.16. Additional Information

Note

The following pages contain either further information or are for reference to complete a topic.

3.17. Additional Exercise: Return of Reject Parts


0V
-10V
+10V
Potentiometer to set (simulate) the weight:
0 to 10V ≙ 0 to 500kg
Good part when weight OK (M35.0 = '1')
100kg <= Weight <= 400kg
Removal point
Reject parts
Good parts
Return of reject parts:
when M35.0 = '0' return part to Bay 3 ("B_Bay3", I 8.7)

Function Up Until Now

Parts are transported from Bay 1 or 2 through the light barrier. A transport sequence is started as soon as a part is placed on the conveyor at Bay 1 or 2 and the associated bay pushbutton is pressed. The transport sequence ends as soon as the part has passed through the light barrier.

The acquisition of the weight (to be set on the potentiometer) of the transported parts is already programmed in "OB_Cyclic interrupt" (OB235). If the part weight is outside of the allowable range of 100 to 400kg, the bit memory "M_Weight_OK" (M35.0) is assigned the status '0'.

Task

Parts whose weights lie outside of the allowable range are to be returned to Bay 3 ("B_Bay3", I 8.7). As well, these parts are not to be counted.

What to Do

1. Expand the block "FC_ConvMotor" (FC16) to include the described return function.
2. Expand the block "FC_Count" (FC18) in such a way that the reject parts are not counted.
3. Save your project and download all blocks into the CPU.

Contents

8

8. Technology Objects

8.1. Objectives

At the end of the chapter the participant will ...

... know the difference between PWM and PTO outputs.

… be familiar with the structure of a simple closed-loop control.

… be familiar with the procedure for creating a PID controller.

… be able to commission a PID controller with automatic optimization.

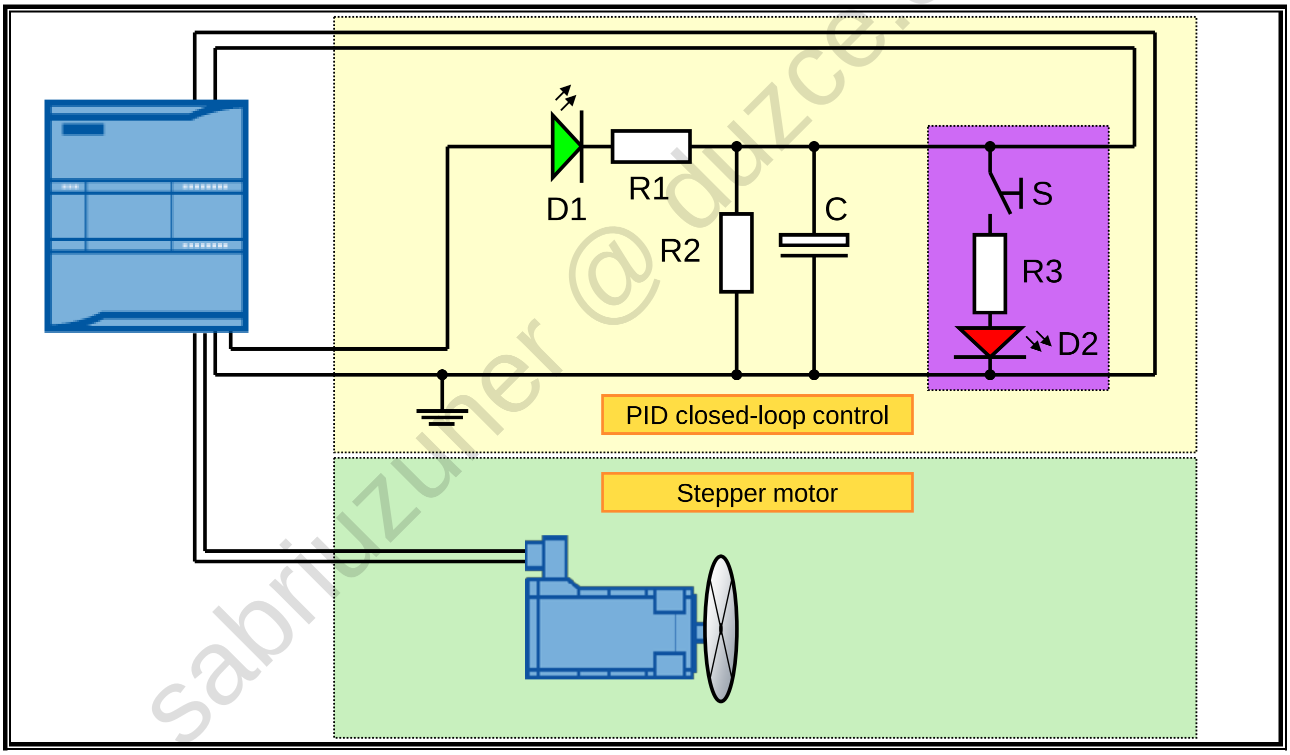
… understand the principle of controlling a stepper motor.

… be able to commission a stepper motor for positioning a turntable.

Objectives

In this chapter, the technology objects "PID (Control)" and "Axis" are dealt with. The necessary, theoretical basics and the procedure for configuring are presented.

8.2. Task Description: Commissioning a PID Controller and Controlling a Stepper Motor



Task Description

In the first step, a PID controller is to be commissioned. It is to control the voltage at Capacitor C to a constant voltage of 10V, even when a fault, in the form of a load resistance R3 is switched in via the switch S.

The manipulated variable (PWM output) is thereby controlled by the "PID_Compact" controller block, by evaluating the fed back capacitor voltage at the analog input of the CPU.

After the control loop has been commissioned with the technology object PID, the stepper motor of the training device is then to be commissioned. For this, the technology object "Positioning Axis" is to be used, which is to be configured by you. On the hardware side, the so-called "PTO" output of the CPU is used for this as well as a Boolean output for the direction setting.

There are Motion Control instructions for the programming of the axis control.

Your task is to commission a prefabricated function block and to monitor the movement sequences on an online screen.

8.3. Introduction to Pulse Generators

CPUs

Signal board

100kHz

20kHz / 200kHz

Each CPU with 2 pulse generators

Description	Default output assignment		
		Pulse	Direction
PTO 1	Integrated in CPU	Q0.0	Q0.1
	Signal board	Q4.0	Q4.1
PWM 1	Integrated in CPU	Q0.0	--
	Signal board	Q4.0	--
PTO 2	Integrated in CPU	Q0.2	Q0.3
	Signal board	Q4.2	Q4.3
PWM 2	Integrated in CPU	Q0.2	--
	Signal board	Q4.2	--

Pulse Generators

All CPU types of the SIMATIC S7-1200 series are equipped with two pulse generators. These can be used independent of each other either for Pulse Width Modulation (PWM) or pulse train (Pulse-Train-Output – PTO).

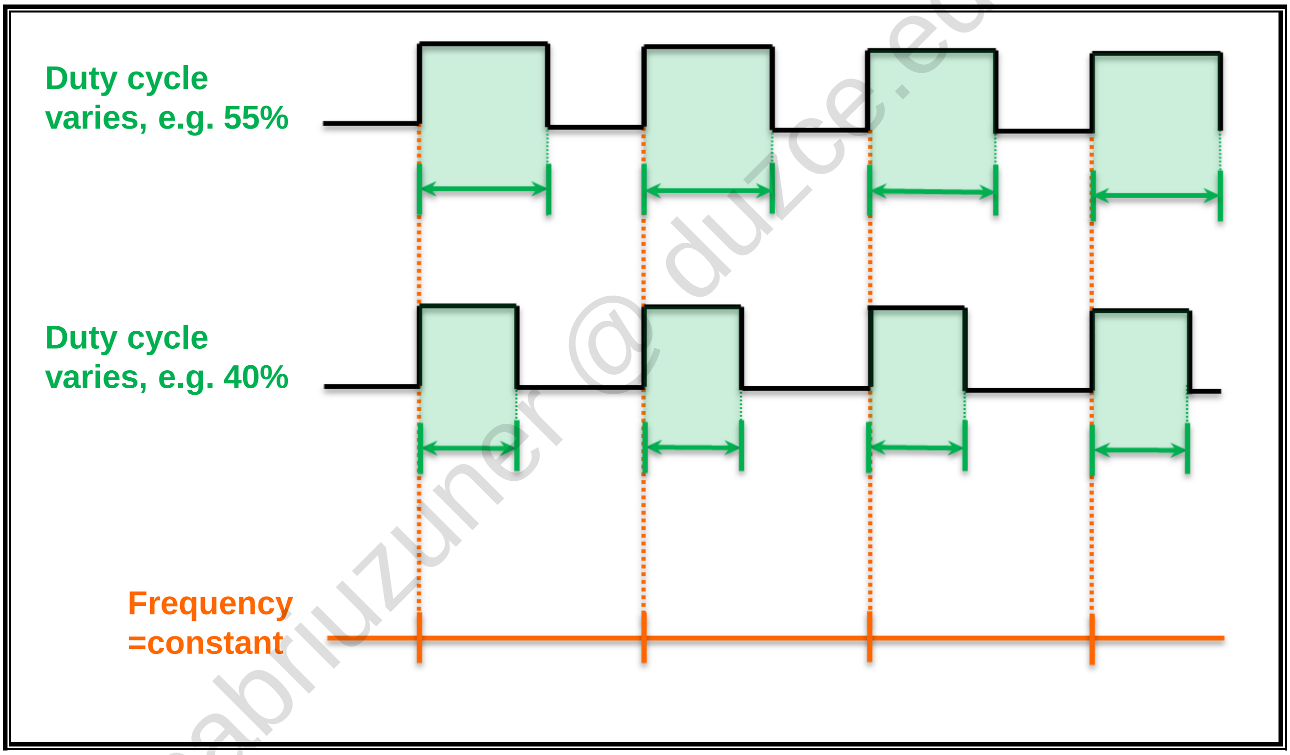
The two pulse generators are assigned specific digital outputs by default (see table above) and are activated in the device configuration of the respective CPU. Integrated CPU outputs or the outputs of an optional signal board can be used. If the addresses of the outputs were changed, the addresses correspond to the newly assigned ones.

Regardless, PTO1/PWM1 always uses the first two digital outputs of the configured addresses and PTO2/PWM2 uses the next two digital outputs, either on the CPU or the inserted signal board. When an output is not required for a pulse function, it is available for other purposes.

The maximum pulse frequency of the pulse generators is 100 kHz for the digital outputs of the CPU and 20 kHz or 200 kHz for the digital outputs of the signal board.

⚠ STEP7 gives no warning when an axis is configured with a maximum speed or frequency that exceeds this hardware limitation. This can lead to problems in the application. You must always make sure that the maximum pulse frequency of the hardware is not exceeded.

8.3.1. Pulse Width Modulation (PWM)



Pulse Width Modulation

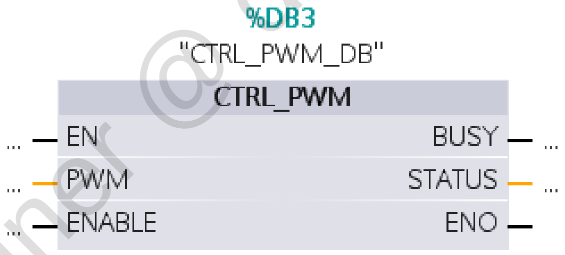
With the Pulse Width Modulation (PWM), the cycle time, that is, the time from one positive edge to the next, remains constant. The duty cycle (pulse width), however, represents the variable size of the modulation.

The duty cycle can be specified as hundredth of the cycle time (0 – 100), as thousandth (0 – 1000), as ten thousandth (0 – 10000) or as S7 analog format. The pulse duration can lie between 0 (no pulse, always off) and full scale (continuous pulse, always on).

Since the duty cycle can lie between 0 and full scale with the PWM, it provides a digital output that in many ways is the same as an analog output. The PWM output can, for example, serve to control the speed of a motor from standstill to full speed or it can be used to control the position of a valve from closed to fully open.

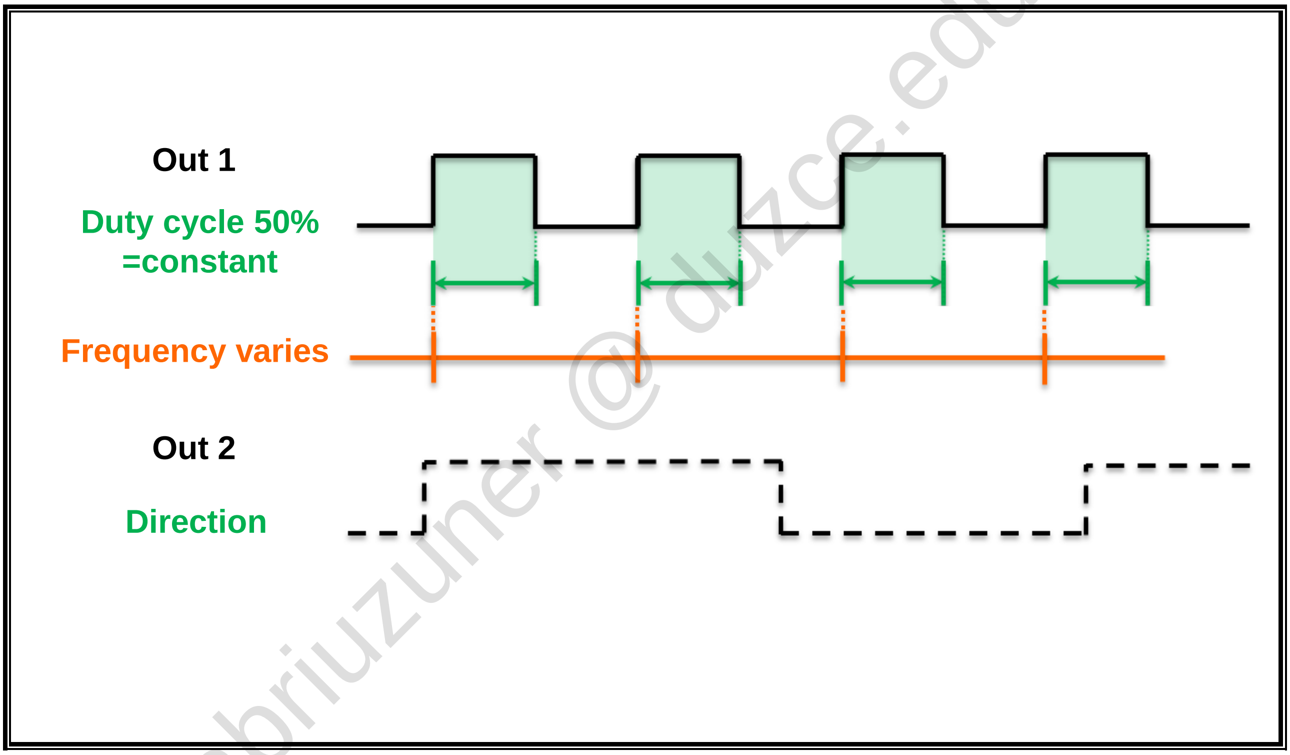
Controlling PWM Outputs

The "CTRL_PWM" block is used to control PWM outputs.



The first time the target system switches to RUN, the PWM duty cycle ratio is set to the start value specified in the device configuration. To change the pulse duration during program runtime, the desired values are written in the output addresses ("Start address") specified in the device configuration, for example, with the command "MOVE".

8.3.2. Pulse Train Output (PTO)

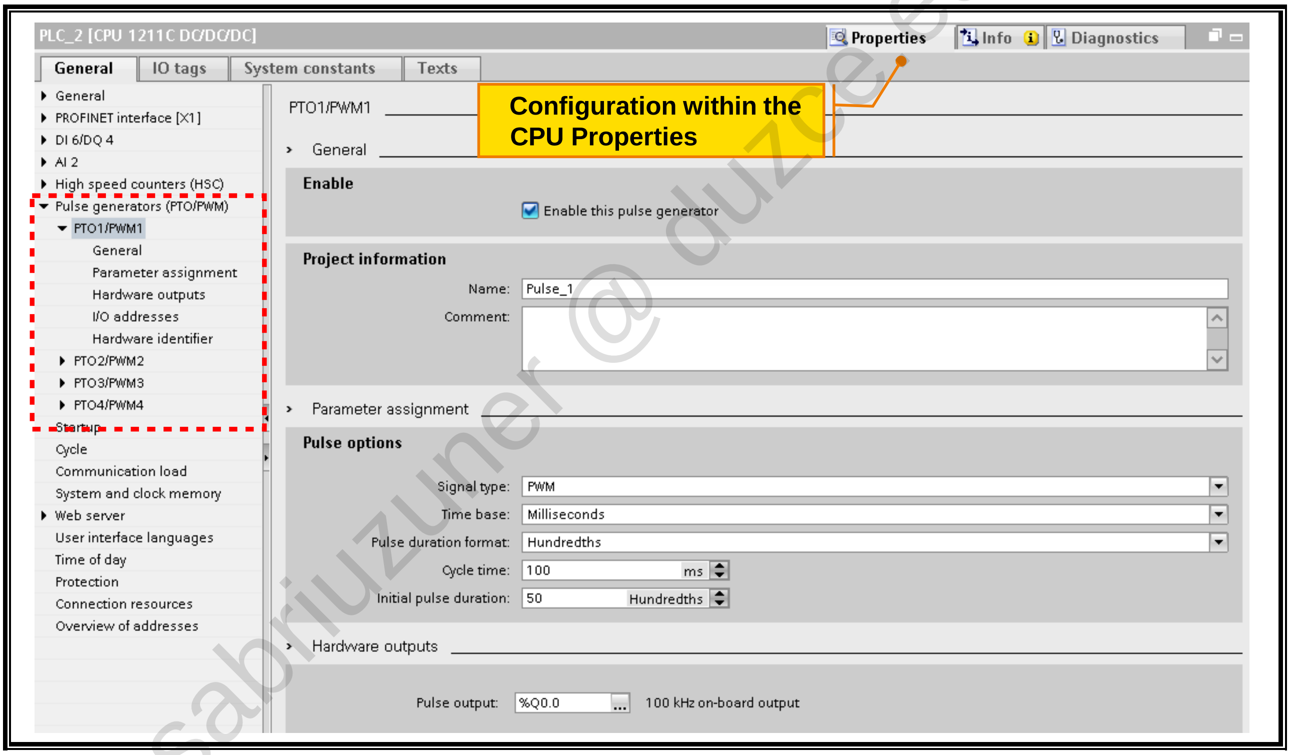


Pulse Train Output

Unlike the PWM, the Pulse Train Output has a fixed duty cycle of 50% and a variable frequency. Through this, the speed of the connected drive can be controlled.

The turning direction of the drive can be specified via the direction output.

8.3.3. Configuring a Pulse Generator



☑ Enable this pulse generator

Name: Pulse_1

Comment:

Signal type: PWM

Time base: Milliseconds

Pulse duration format: Hundredths

Cycle time: 100 ms

Initial pulse duration: 50 Hundredths

Pulse output: %Q0.0 ... 100 kHz on-board output

Configuration within the CPU Properties

Configuring a Pulse Generator

To activate the pulse generator, you need to proceed as follows:

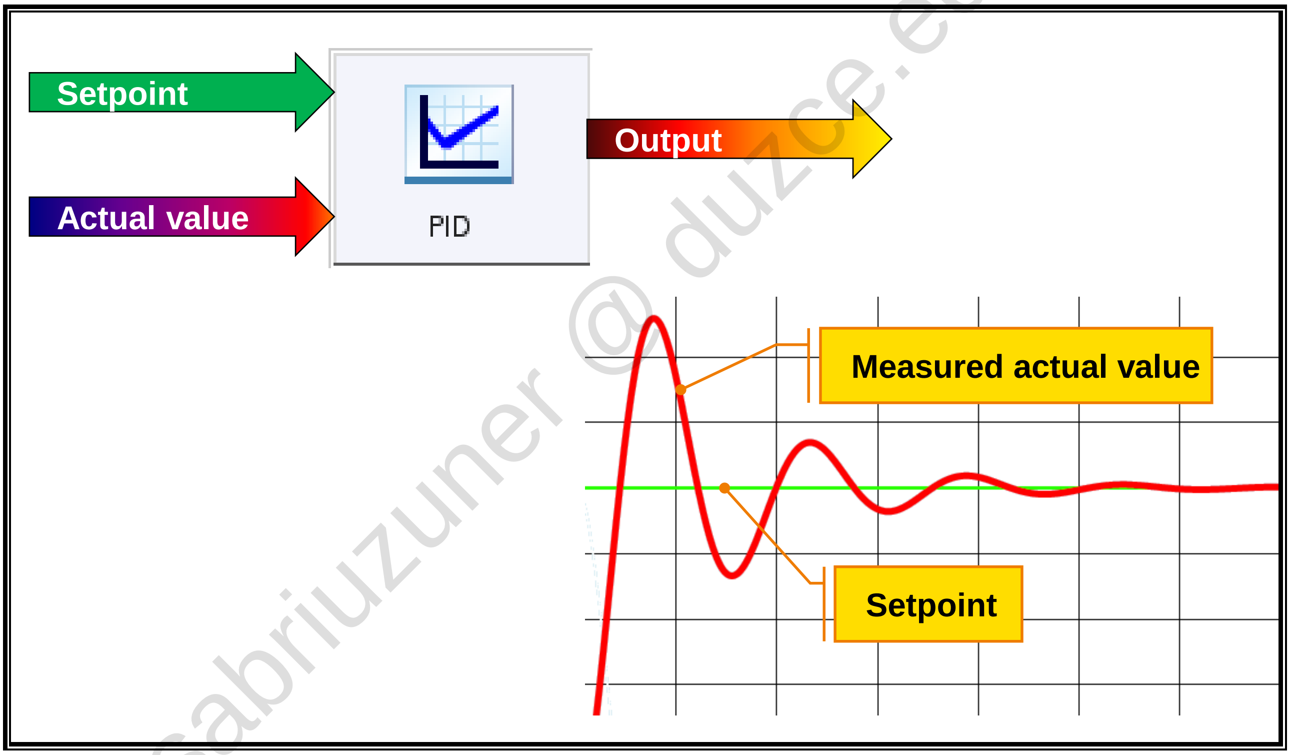
1. First of all, the respective pulse generator must be activated.

2 Another name than the default assigned name as well as a comment can be entered.

3. Set pulse options
   - Use as PWM or PTO output
   - Time base
   - Format for the pulse duration
   - Cycle time specification
   - Initial pulse duration

4. The hardware outputs used by the pulse generator are displayed in the field "Pulse output"

8.4. Introduction to the PID (Controller)



PID (Controller)

PID stands for "Proportional Integral Differential". A PID controller has a proportional component, an integrating component and a differentiating component. For each of the three components, a specific equation is in force:

- The equation of the proportional component results in a value that is proportional to the control deviation
- The result of the integral equation increases with the duration of the control deviation
- The speed of the control deviation influences the differential component; the steeper the increase or fall of the change, the greater the D-component is

The three equations are then combined and result in the output value (Output).

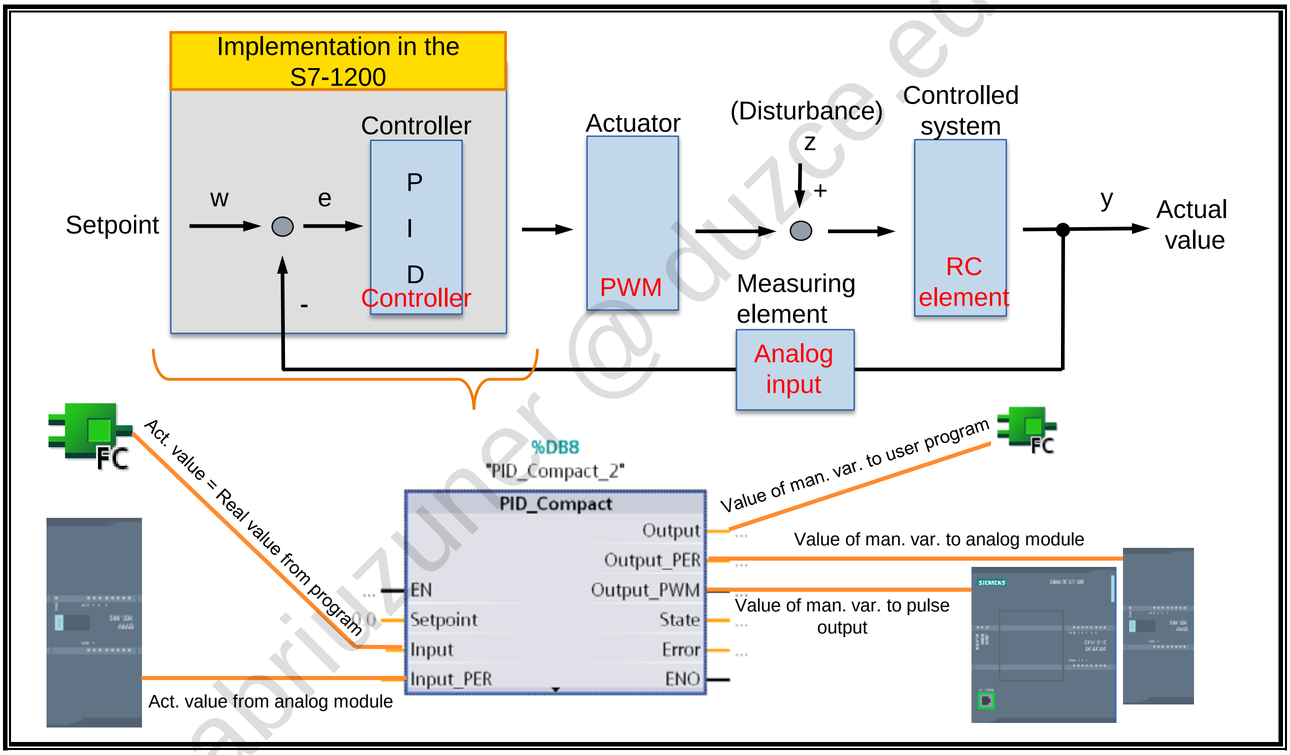
PID controllers are used in industry, for example, to control the temperature of welding systems when it is important to retain a constant temperature value in spite of possible disturbances.

Put very simply, a PID controller serves to align a changing, measured actual value with a setpoint value as quickly as possible and as exactly as possible.

This is done by readjusting the output variable whereby the overshoots and undershoots keep getting smaller until the actual value equals the setpoint value as exactly as possible.

For this, there is a wizard in STEP7 which, in conjunction with the S7-1200, enables you to configure the necessary settings of a controlled system quickly and easily as well as without extensive prior knowledge.

8.4.1. Implementation of a PID Controller in the S7-1200



PID Controller in the S7-1200

Based on the example of a complete control system, the picture above shows the PID controller implemented in the SIMATIC 1200 station in symbolic representation and the block that results from it.

In the S7-1200, the actual controller of a PID control system is implemented. For this, the TIA Portal provides a prefabricated block "PID_Compact" which can be inserted in the user program and then assigned.

At the same time, a "PID" technology object is available with which the controller can be configured in the user program and then commissioned.

⚠ **The switch output for the pulse width modulation is controlled by the instruction PID_Compact. The pulse generators integrated in the CPU are not used.**

Closed-loop Control

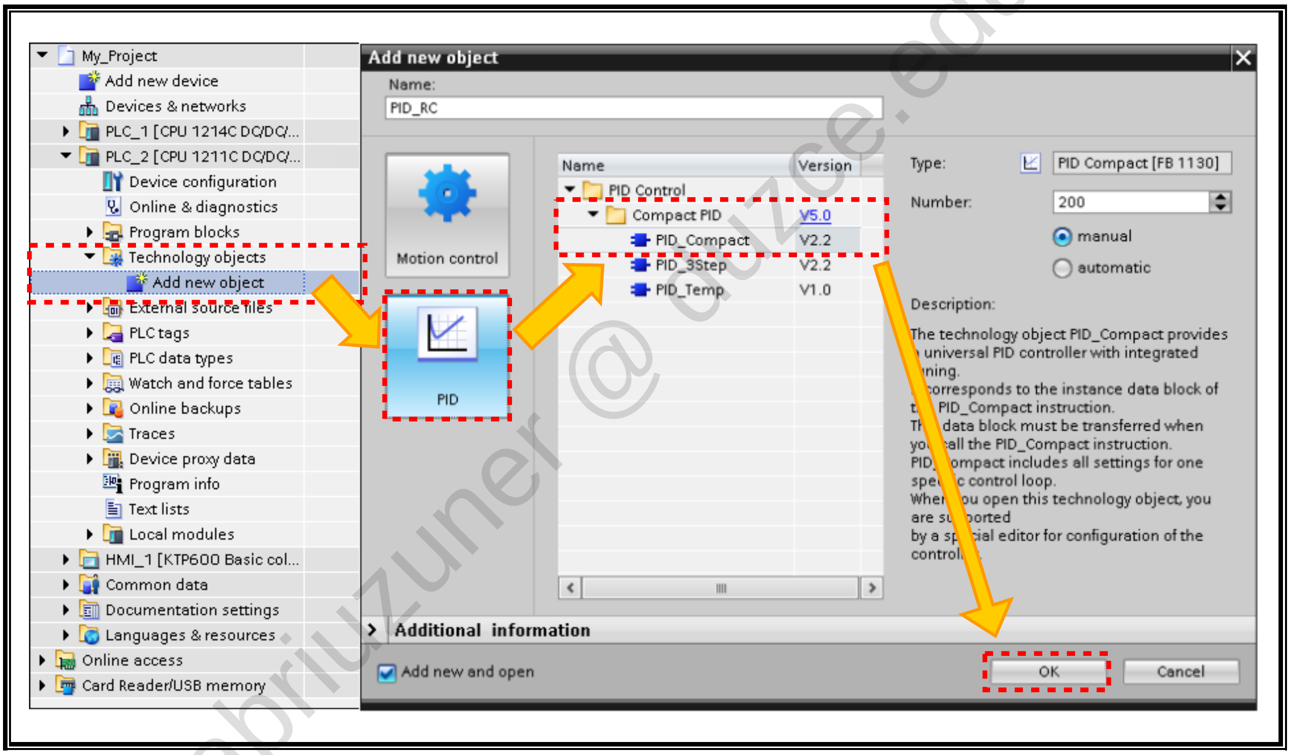
- Blocks:

  In the simplest case, a PID control system consists of a PID controller, an actuator as well as the system to be controlled. The output signal of the controlled system is fed back via a measuring element on the PID controller.

- Signals/Values:

  In the simplest case, distinction is made in a PID control system between the setpoint value (w), the input value (e), the correcting variable that results from it. Together with an influencing disturbance (z), the actual value (y) results from this in the closed-loop control, which is once again fed back to the PID controller via the measuring element.

8.4.2. Creating a "PID" Technology Object



Name: PID_RC

Name	Version
PID Control	
Compact PID	V5.0
PID_Compact	V2.2
PID_3Step	V2.2
PID_Temp	V1.0

Type: PID Compact [FB 1130]

Number: 200

● manual

○ automatic

Description:

The technology object PID_Compact provides universal PID controller with integrated ning. corresponds to the instance data block of t PID_Compact instruction. Th data block must be transferred when yo call the PID_Compact instruction. PID ompact includes all settings for one spe c control loop. When ou open this technology object, you are su ported by a sp cial editor for configuration of the control.

☑ Add new and open

Creating a PID Controller

After a PID controller has been created under "Technology objects", the view is transferred to the wizard. It uses the following identifiers:

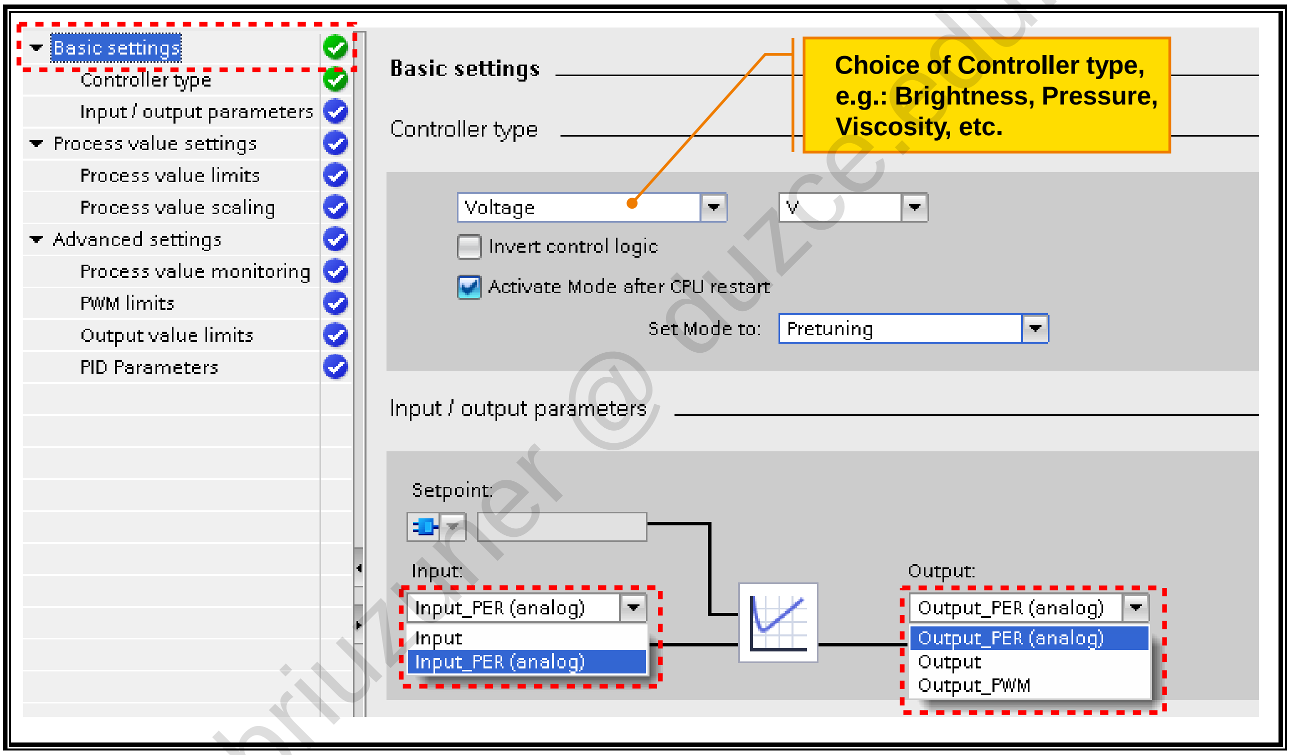
✓ The settings were successfully configured

✓ The settings are only occupied with default values, functioning is not hindered by this

✗ The settings are still faulty

As well, a function block for the PID controller is automatically created which contains all input values and output values in its interface.

8.4.2.1. Configuring a PID Controller (1) - Basic Settings



- Basic settings ✓
  - Controller type ✓
  - Input / output parameters ✓
- Process value settings ✓
  - Process value limits ✓
  - Process value scaling ✓
- Advanced settings ✓
  - Process value monitoring ✓
  - PWM limits ✓
  - Output value limits ✓
  - PID Parameters ✓

Basic settings

Controller type

Voltage ▾ V ▾

□ Invert control logic

☑ Activate Mode after CPU restart

Set Mode to: Pretuning ▾

Input / output parameters

Setpoint:

Input: Input_PER (analog) ▾ (Input, Input_PER (analog))

Output: Output_PER (analog) ▾ (Output_PER (analog), Output, Output_PWM)

Choice of Controller type, e.g.: Brightness, Pressure, Viscosity, etc.

Basic Settings

The configuration of the basic settings of the PID controller offers the following options:

Type of Controller

The preselection "Controller type" sets the desired unit for the controller.
If the checkbox "Invert the control logic" is activated (checked), it causes an increase of the manipulated value when a decrease of the actual value occurs (for example, falling water level through an increase of the valve position of the outlet valve or decreasing temperature through an increase of the cooling capacity).

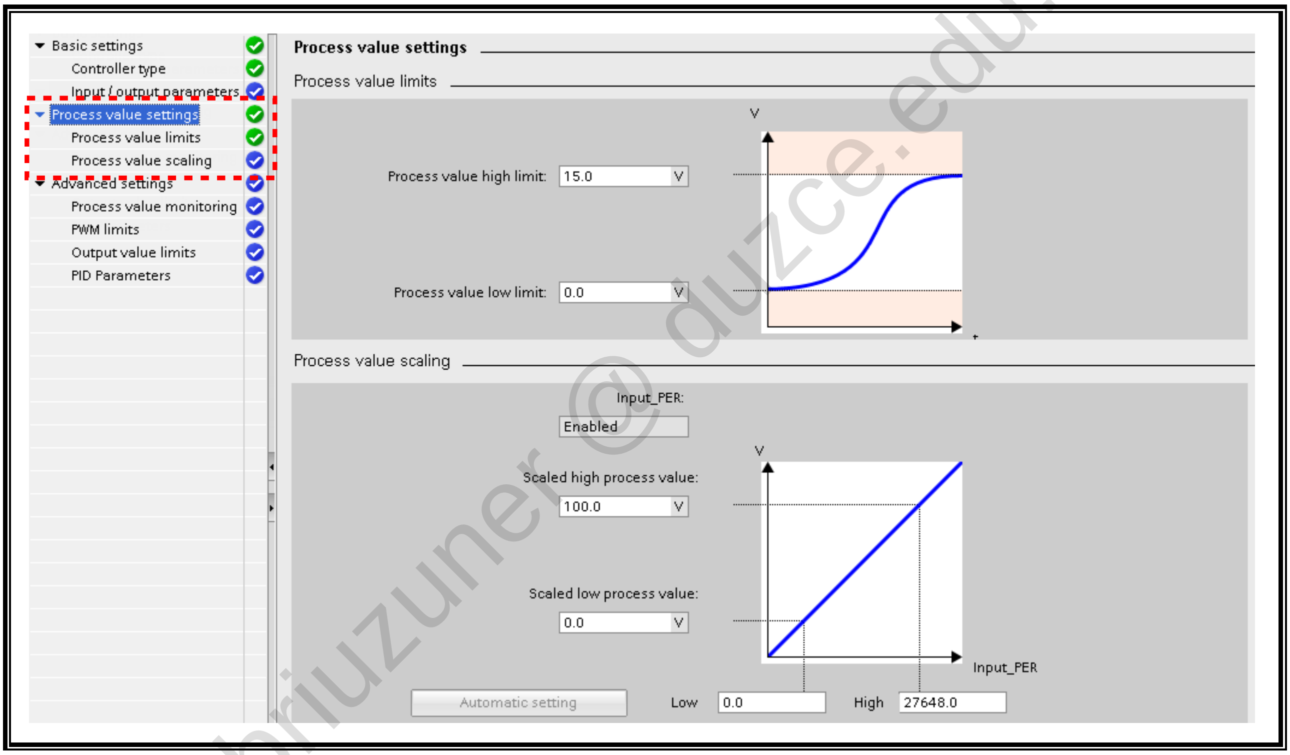
Setting the Input / Output Parameters

- Setpoint:
  Choose whether the value at the function block or the value of the instance DB is to be used (insofar as it exists and is only available in the Inspector window of the program editor).

- Input:
  Choose whether the input parameter "Input" or "Input_PER" is to be used.
  - "Input" is used when an actual value from the user program is to be used.
  - "Input_PER" is used when the actual value of an analog input is to be used.

- Output
  Select the manipulated value output of the instruction "PID_Compact". The following possibilities are available:
  - Output: uses a variable of the user program as the manipulated value output. (Real format)
  - Output_PER: uses an analog output as the manipulated value output (analog output value).
  - Output_PWM: uses a digital switch output and controls it via a pulse width modulation. (The manipulated value is formed via variable switch-on and switch-off times.)

8.4.2.2. Configuring a PID Controller (2) - Process Value Settings



Process value settings

Process value limits

Process value high limit: 15.0 V

Process value low limit: 0.0 V

Process value scaling

Input_PER: Enabled

Scaled high process value: 100.0 V

Scaled low process value: 0.0 V

Low 0.0 High 27648.0

Process Value Settings

For the configuration of the process value settings, the following options are available:

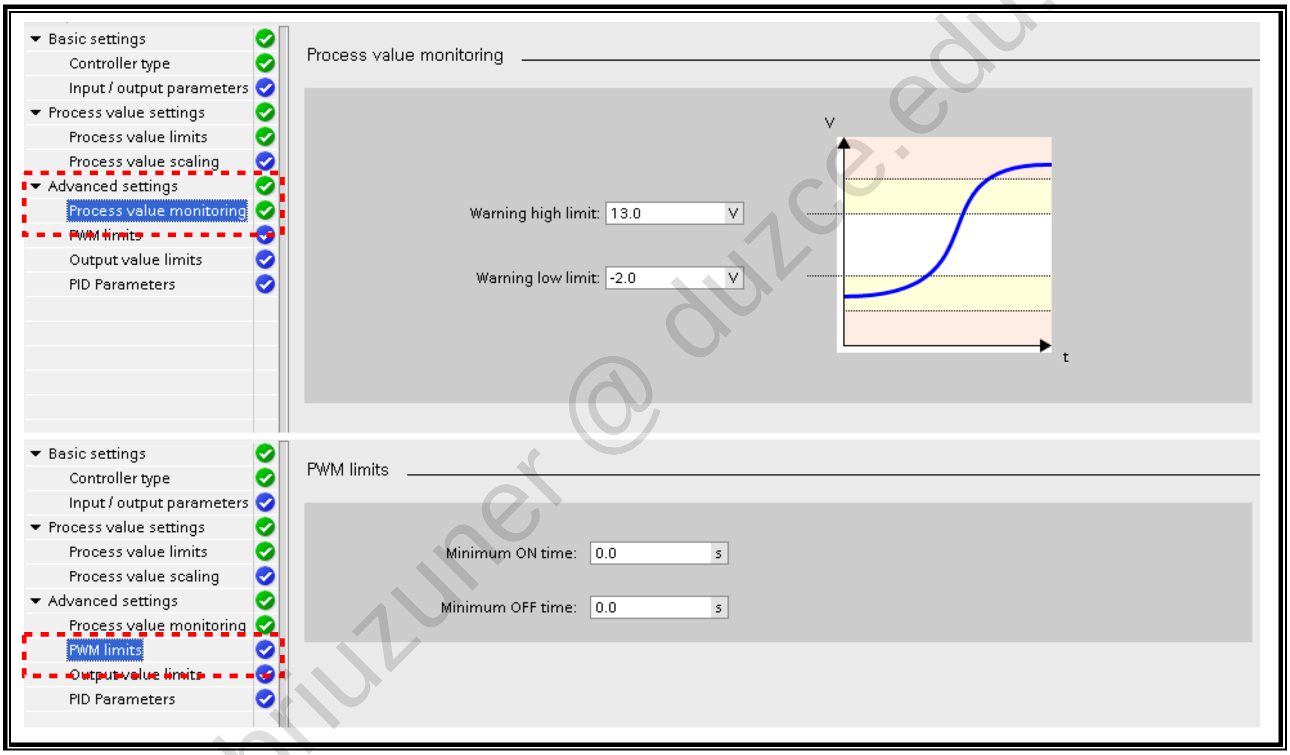
High Limit and Low Limit

They define the absolute upper and lower limit of the process value. During operation, as soon as these limits are exceeded or fall below, the controller switches off and the value of the manipulated variable is set to 0%.

Scaling

Through scaling, the process values (actual values) are defined by a lower and an upper value pair. Each value pair consists of the value of the analog input and the physical value of the respective scaling point. Depending on the configuration of the basic setting, a process value of the user program can also be used instead of the analog value of the analog input.

8.4.2.3. Configuring a PID Controller (3) - Process Value Monitoring and PWM Limits



Process value monitoring

Warning high limit: 13.0 V

Warning low limit: -2.0 V

PWM limits

Minimum ON time: 0.0 s

Minimum OFF time: 0.0 s

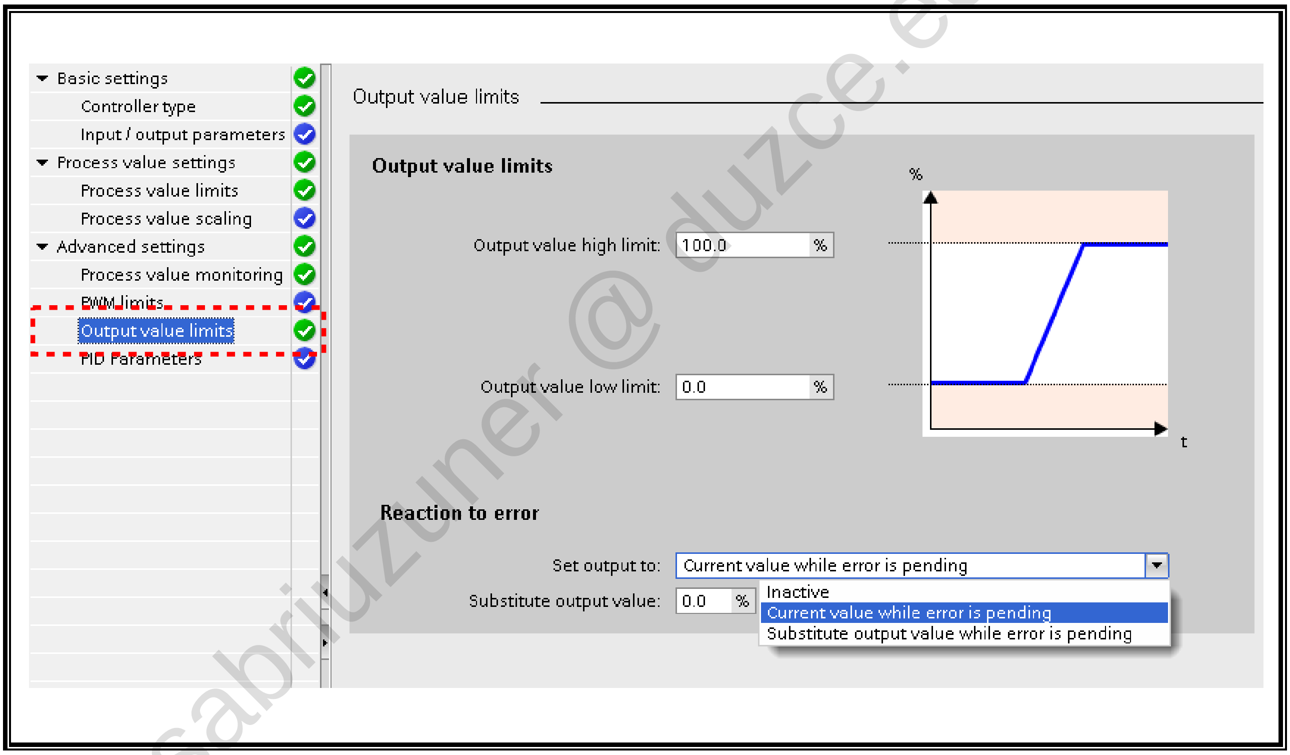
Process Value Monitoring

The monitoring of the process value is preset by two limits. If, during controller runtime, the process value exceeds the high limit or falls below the low limit, a message is output at the Boolean output parameters "InputWarning_H" or "InputWarning_L" of the block "PID_COMPACT".

PWM Limits

In the window "PWM limits", the minimum permitted switch ON and switch OFF times of the pulse width modulation are set. The minimum ON and OFF times can be extended when, for example, the number of switching cycles is to be reduced. This makes sense, for example, for the delayed control of a tank level when you want to avoid the valve reacting to every small change in the level.

8.4.2.4. Configuring a PID Controller (4) - Output Value Limits



Output value limits

Output value limits

Output value high limit: 100.0 %

Output value low limit: 0.0 %

Reaction to error

Set output to: Current value while error is pending

Substitute output value: 0.0 %

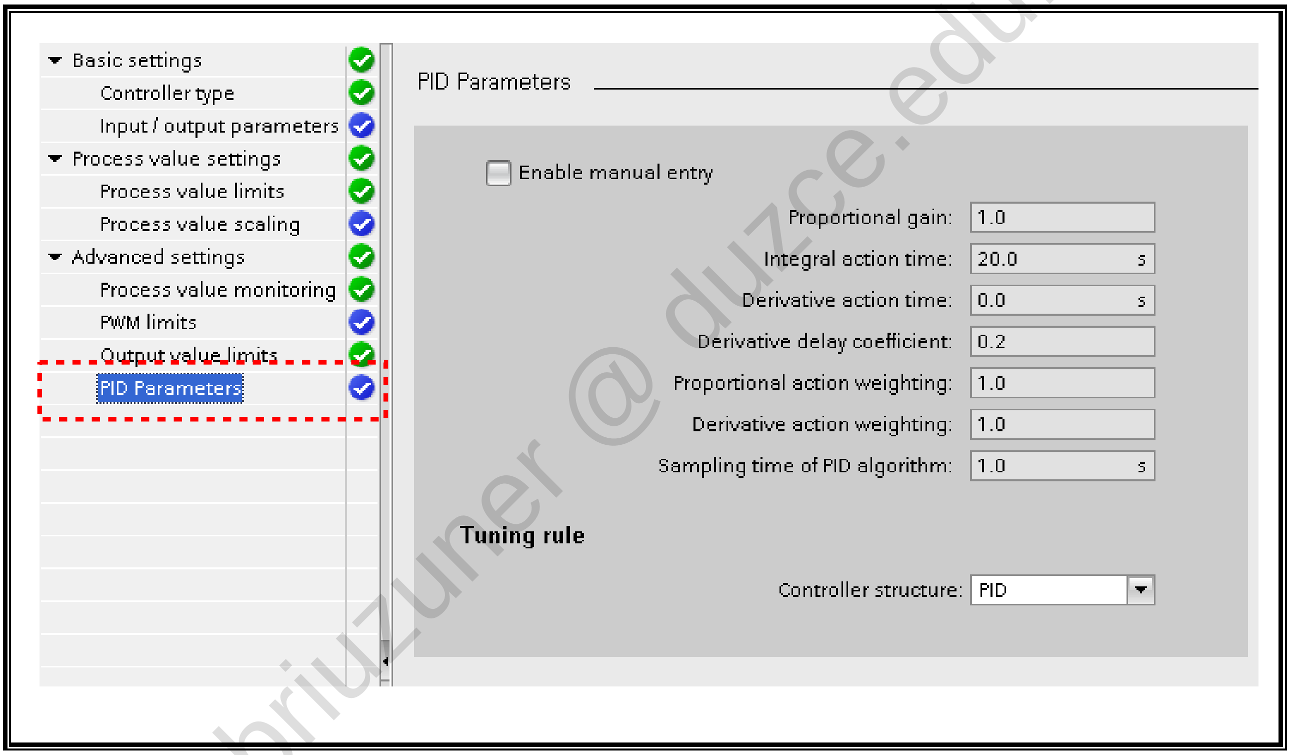
Output Value Limits

In the configuration window "Output value limits", the absolute limits of the manipulated value are specified. Neither in manual mode nor in automatic mode can absolute manipulated value limits be exceeded nor can they fall below. If in manual mode, a manipulated value is specified outside of the limits, the effective value in the CPU is limited to the configured limits.

Reaction to Error

If an error occurs during processing (output parameter ERROR = TRUE), then a substitute value can be output, the old value can be held (pending) or the output can be deactivated (inactive) at OUTPUT.

8.4.2.5. Configuring a PID Controller (5) - PID Parameters



PID Parameters

□ Enable manual entry

Proportional gain: 1.0

Integral action time: 20.0 s

Derivative action time: 0.0 s

Derivative delay coefficient: 0.2

Proportional action weighting: 1.0

Derivative action weighting: 1.0

Sampling time of PID algorithm: 1.0 s

Tuning rule

Controller structure: PID

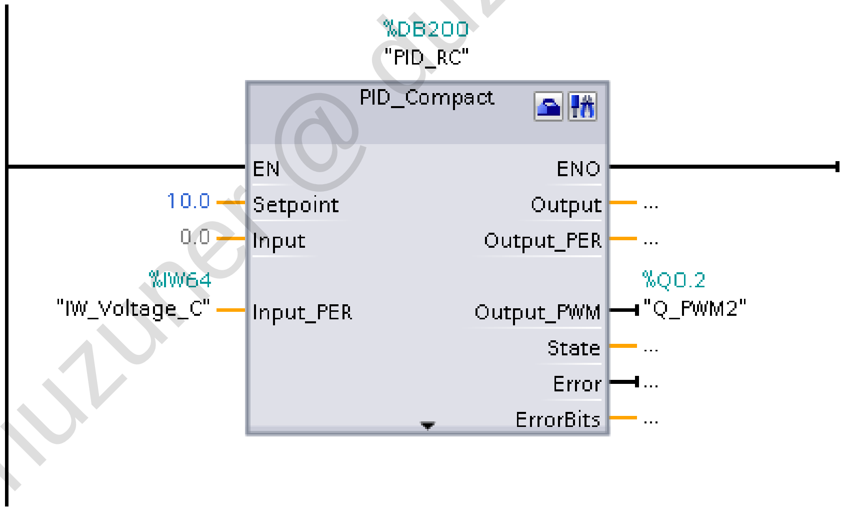
PID Parameters

The PID parameters are grayed out by default; they can, if need be, be changed. This, however, is only recommended for users with experience in PID control.

The PID parameters are determined automatically when the automatic auto-tuning has been run through.

8.4.3. "PID_Compact" Call

Call in the cyclic interrupt OB



PID_Compact

The instruction PID_Compact provides a PID controller with integrated optimization for automatic and manual mode operation.

Call

PID_Compact is called in the time base of the cycle time of the calling OB. This must be constant to ensure that the PID controller can sample in equidistant intervals. For that reason, PID_Compact is preferably called in a cyclic interrupt OB since the cycle time in the cyclic user program can vary significantly because of conditional program execution, for example.

Start-up Behavior

When the CPU starts up, it starts PID_Compact in the operating mode in which it was last active.

Monitoring the Sampling Time PID_Compact

Ideally, the sampling time corresponds to the cycle time of the calling OB. The instruction PID_Compact measures, in each case, the interval between two calls. That is the current sampling time. Every time the operating mode changes and in the first start-up, the mean value of the first 10 sampling times is formed. When the current sampling time deviates too greatly from this mean value, an error occurs (Error = 0800 hex) and PID_Compact switches into the "inactive" mode.

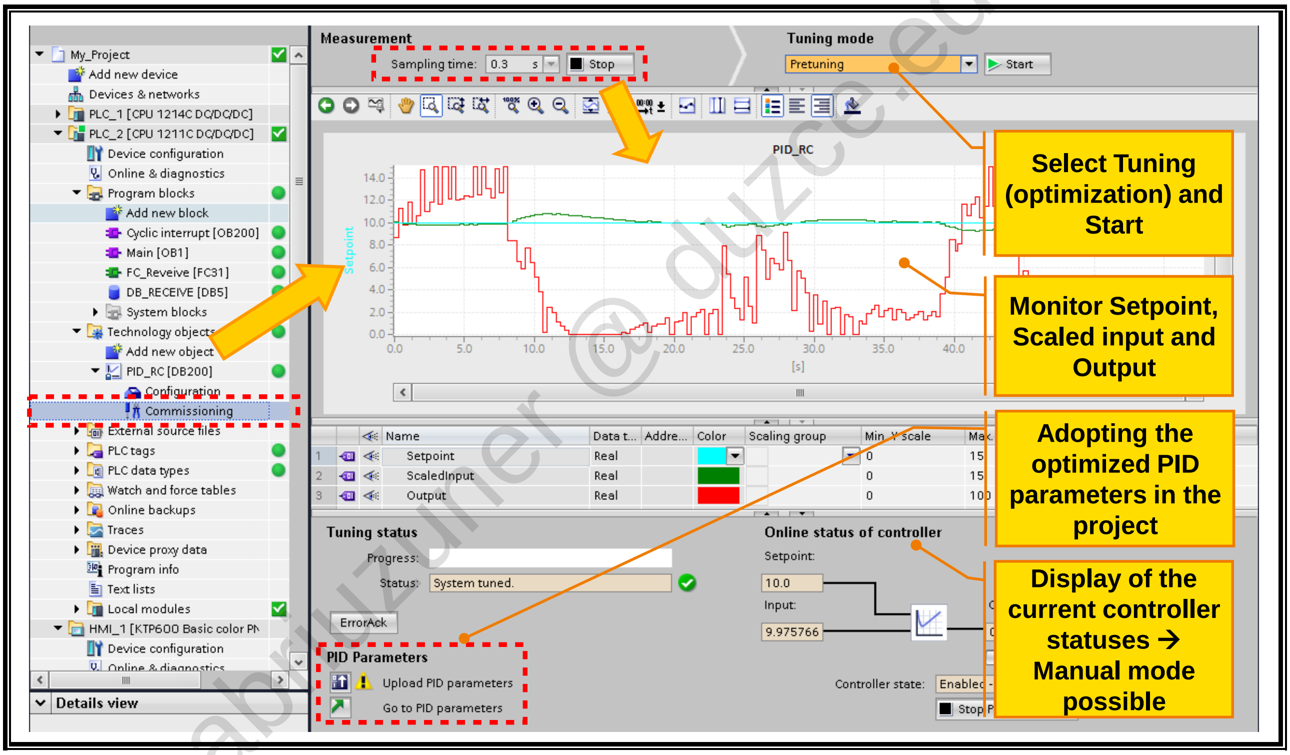
During tuning (optimization), the following conditions put PID_Compact in the "inactive" mode:

- New mean value >= 1.1 x old mean value
- New mean value <= 0.9 x old mean value

In automatic mode, the following conditions put PID_Compact in the "inactive" mode:

- New mean value >= 1.5 x old mean value
- New mean value <= 0.5 x old mean value

8.4.4. Using the Commissioning Panel



Sampling time: 0.3 s

Tuning mode: Pretuning

Name	Data t...	Addre...	Color	Scaling group	Min. Y scale	Max.
1 Setpoint	Real		🟦		0	15
2 ScaledInput	Real		🟩		0	15
3 Output	Real		🟥		0	100

Tuning status

Progress:

Status: System tuned.

Online status of controller

Setpoint: 10.0

Input: 9.975766

Controller state: Enabled

PID Parameters

Select Tuning (optimization) and Start

Monitor Setpoint, Scaled input and Output

Adopting the optimized PID parameters in the project

Display of the current controller statuses → Manual mode possible

Commissioning Panel

In the configuration of the PID controller, you can carry out an automatic tuning (optimization) and you can monitor the current measured values.

Commissioning

As soon as Measurement is switched on through a click on "Start", actual value and setpoint value as well as manipulated value are graphically represented (see picture).

Under "Tuning mode" the auto-tuning can be started. This must first of all occur in the first start-up. In the second step, you can then tune in the operating point. The status and the progress of the running tuning (optimization) can be read from the bar graph.

Required for automatic fine tuning:

- PID_Compact is called in a cyclic interrupt OB
- "Manual mode" is deactivated
- The difference between current actual value and setpoint is >50%

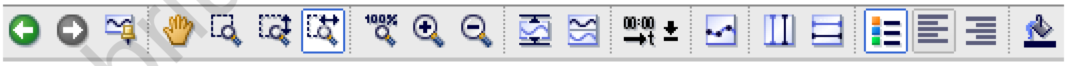
The operation can take some minutes. During this time you cannot work with the CPU.

Subsequently, the ascertained data must be adopted in the project via "Upload PID parameters".

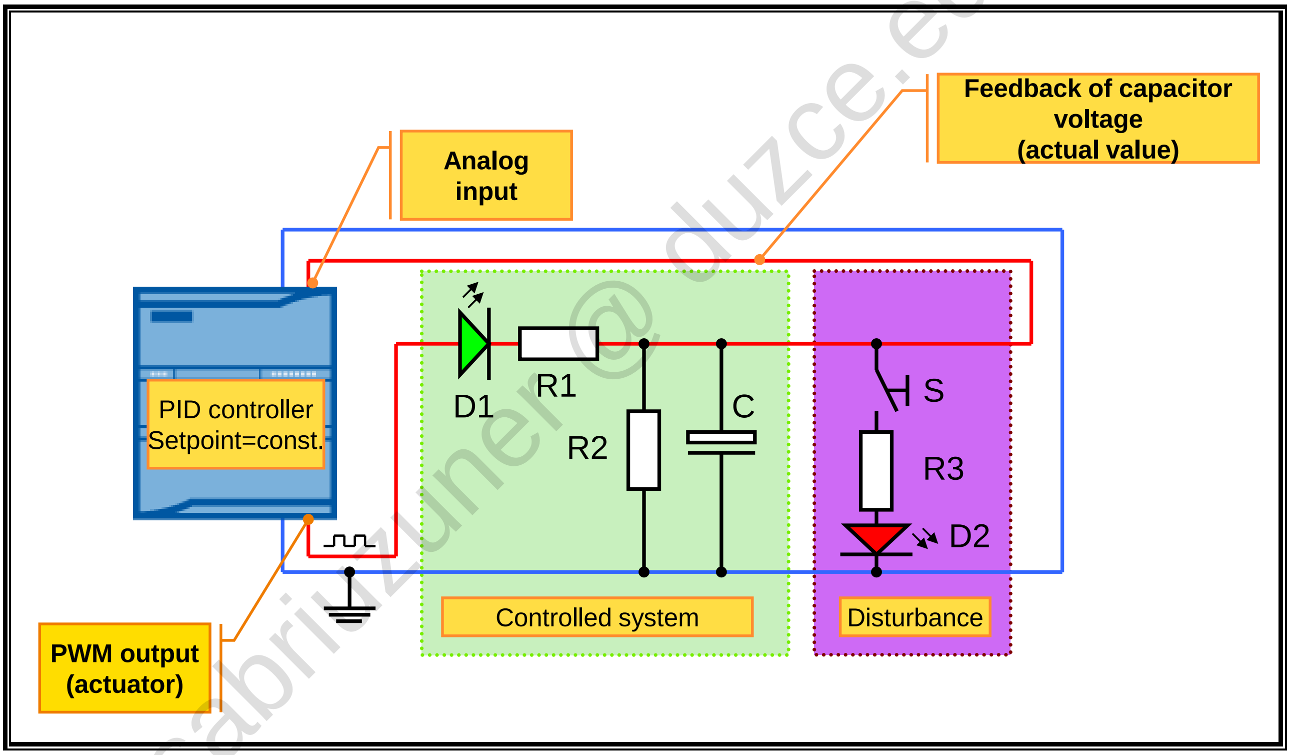
Through "Online status of controller" you can monitor the current actual value, the setpoint as well as the output in % and you can specify a manual manipulated value.

Representation

Using the following buttons, you can stretch or compress the value axes, select a type of representation for the value diagram, shift the view etc.



8.5. Task Description: Controlling the Capacitor Voltage


Analog input
Feedback of capacitor voltage (actual value)
PID controller Setpoint=const.
D1
R1
R2
C
S
R3
D2
PWM output (actuator)
Controlled system
Disturbance

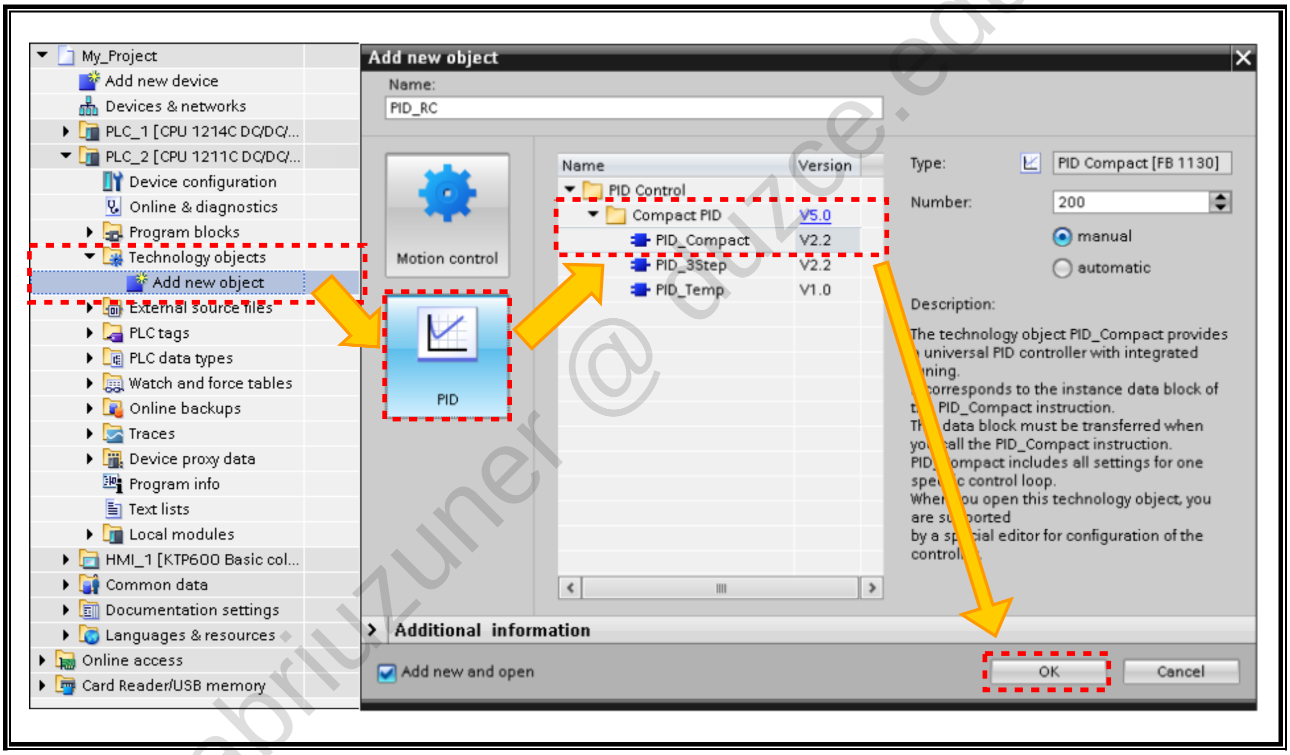
Task Description

In the first step, the PLC with CPU 1211C is to be commissioned.

Then, a PID controller is to be commissioned. This is to control the voltage at Capacitor C to a constant voltage of 10.0V, even when a fault in the form of a load resistance R3 is switched in via the switch S.

The manipulated variable (PWM output) is controlled by the "PID_Compact" controller block, by evaluating the fed-back capacitor voltage at analog input AI0.

8.5.1. Exercise 1: Creating and Configuring the "PID" Technology Object



Name: PID_RC

Type: PID Compact [FB 1130]

Number: 200

● manual

○ automatic

Description:

The technology object PID_Compact provides a universal PID controller with integrated tuning. It corresponds to the instance data block of the PID_Compact instruction. The data block must be transferred when you call the PID_Compact instruction. PID_Compact includes all settings for one specific control loop. When you open this technology object, you are supported by a special editor for configuration of the controller.

☑ Add new and open

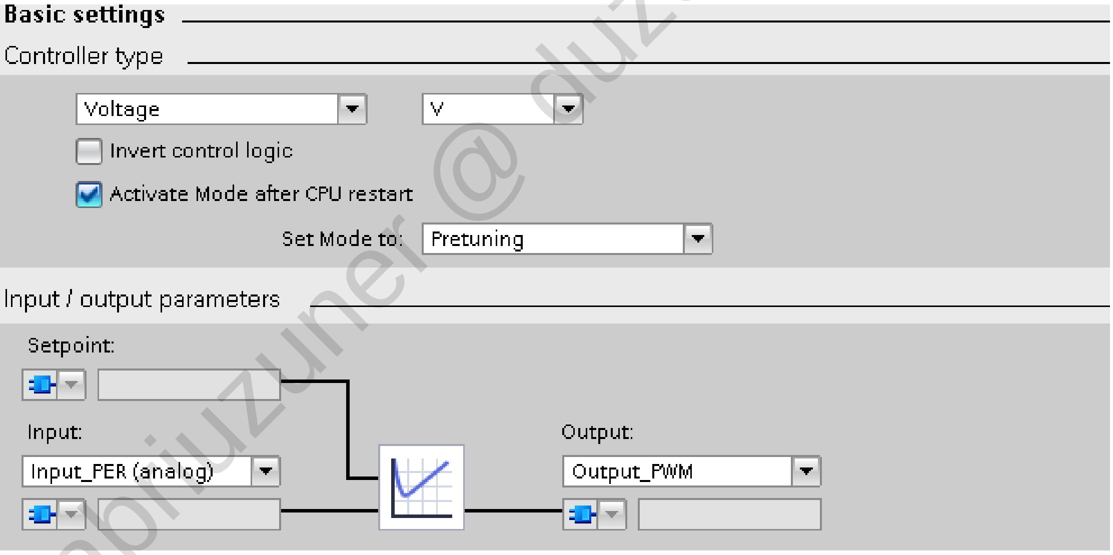
Task

In PLC_2 (CPU 1211C), create a new technology object of the type "PID" and give it the name "PID_RC".

What to Do

1. Start the Technology objects wizard and create a new PID controller.
   PLC_2 →Technology objects → Double-click on "Add new object" → PID → PID_Compact
   - Name: PID_RC
   - Number: 200

2. Implement the settings shown in the following:



Voltage | V

□ Invert control logic

☑ Activate Mode after CPU restart

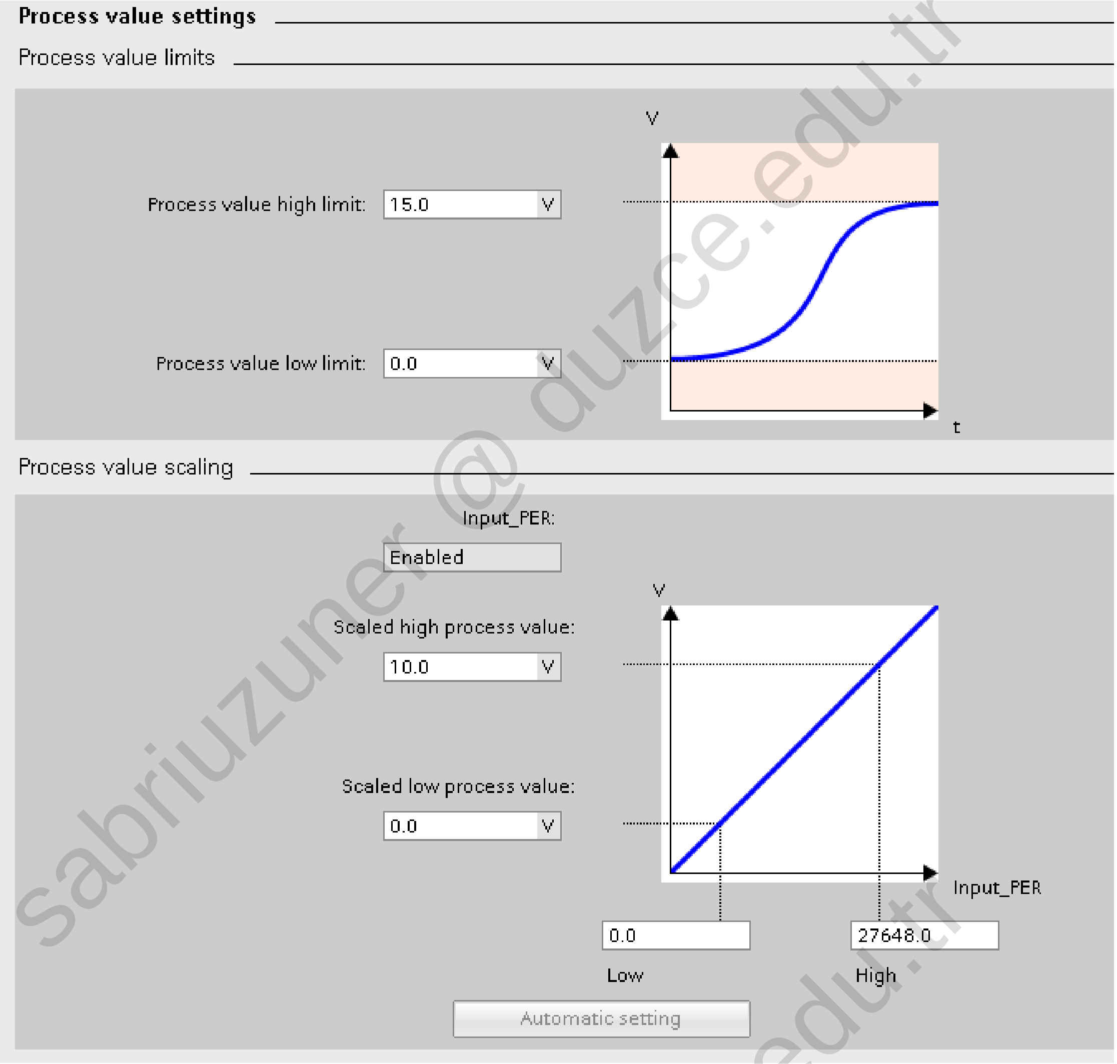
Set Mode to: Pretuning

Setpoint:

Input: Input_PER (analog)

Output: Output_PWM

Continued on the next page →



Process value settings

Process value limits

Process value high limit: 15.0 V

Process value low limit: 0.0 V

Process value scaling

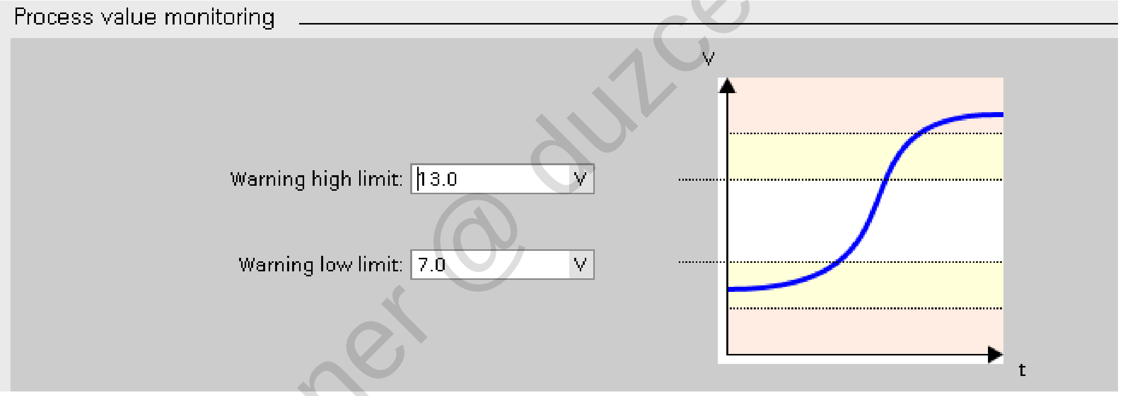
Input_PER: Enabled

Scaled high process value: 10.0 V

Scaled low process value: 0.0 V

Low: 0.0

High: 27648.0



Process value monitoring

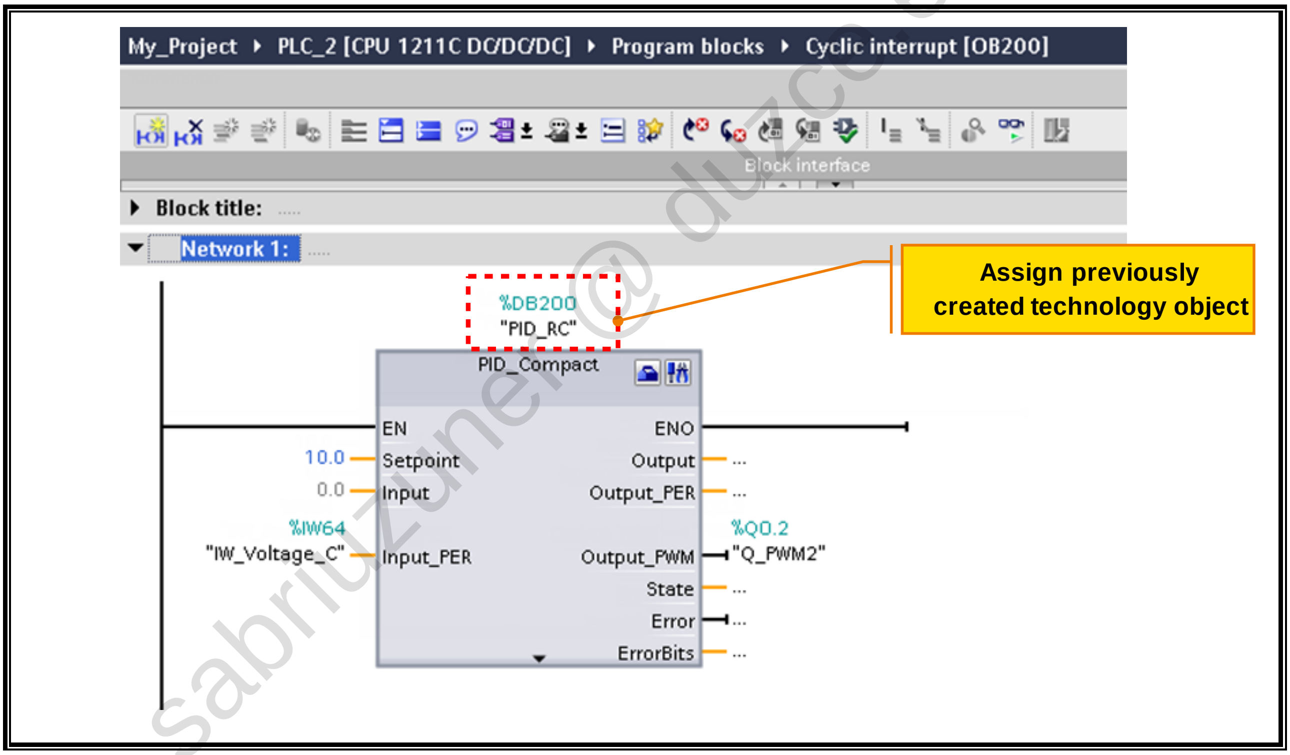
Warning high limit: 13.0 V

Warning low limit: 7.0 V

3. PWM limits, Output value limits, Reaction to error, and PID Parameters remain unchanged.

4. Save your project.

8.5.2. Exercise 2: Calling the "PID_Compact" Block in the Cyclic Interrupt "Cyclic Interrupt" (OB200)



Assign previously created technology object

Task

Create "Cyclic Interrupt" (OB200), call the block PID_Compact and assign the previously created technology object "PID_RC" (DB200) to it.

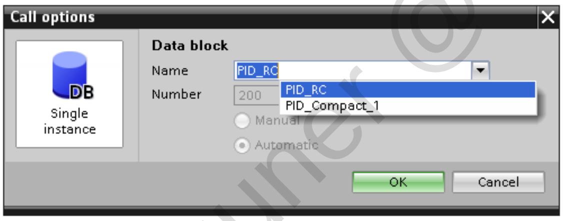
What to Do

1. Create the cyclic interrupt OB "Cyclic Interrupt" (OB200). Cycle time 250ms.

☞ In order to be able to react faster to disturbances, the sampling time of the closed-loop control (cycle time of the cyclic interrupt) can be reduced, for example, to 100ms or less. So that it is easier to monitor the control process, a relatively high value of 250ms is set for the exercise.

2. In OB200, call the block "PID_Compact" from the Instructions catalog.
   Instructions Task Card → Technology → PID Control → Compact PID → PID_Compact

3. In the "Call options" dialog which opens, select "PID_RC":


Call options

Data block

Name: PID_RC

Number: 200

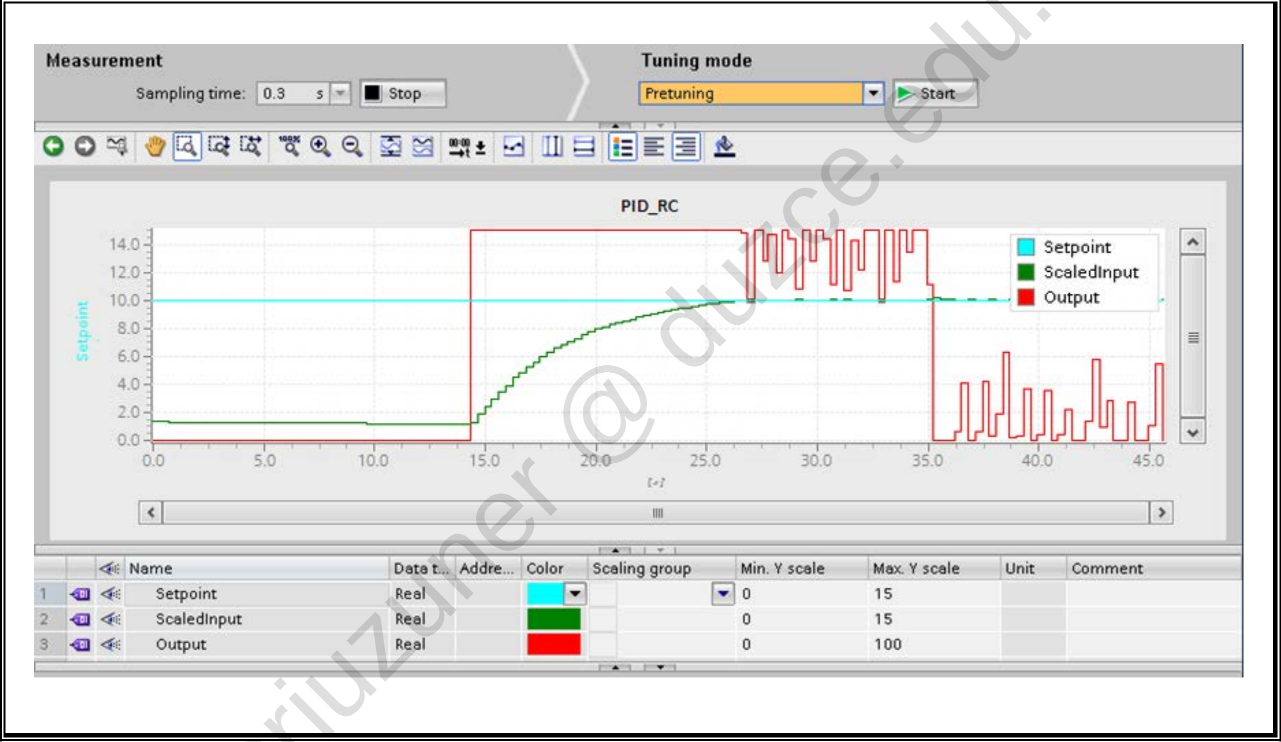
○ Manual

● Automatic

4. Assign the block as shown in the picture.
   – EW_Spg_RC (IW64) is the analog input 0 (0-10V) on the CPU.
   – PWM_Q02 (Q0.2) is the digital output for the PWM.
   – The setpoint is assigned constantly with 10.0 (V).

5. Save your project and transfer the complete user program into the CPU.

8.5.3. Exercise 3: Commissioning the PID Controller



Measurement

Sampling time: 0.3 s

Tuning mode

Pretuning

	Name	Data t...	Addre...	Color	Scaling group	Min. Y scale	Max. Y scale	Unit	Comment
1	Setpoint	Real		🟦		0	15		
2	ScaledInput	Real		🟩		0	15		
3	Output	Real		🟥		0	100		

Task

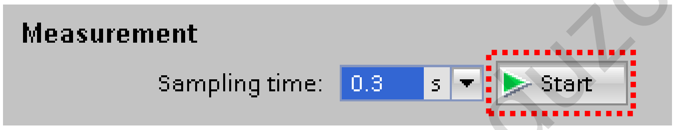
Carry out a first commissioning of the PID controller and save the determined PID parameters in your project.

What to Do

1. Open the Commissioning panel

   PLC_2 → Technology objects → PID_RC → Commissioning

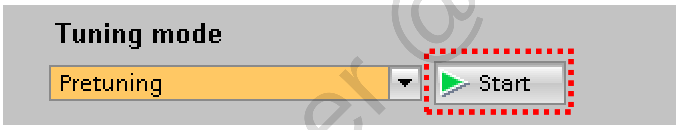
2. Start the measurement



Measurement

Sampling time: 0.3 s

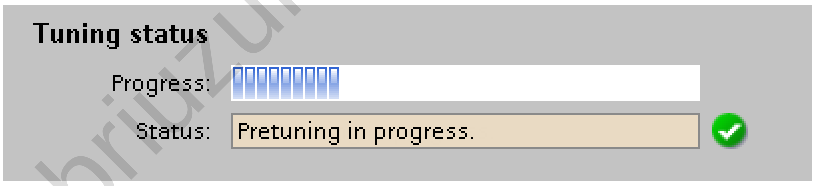
3. Select the Tuning mode "Pretuning" and click on Start



Tuning mode

Pretuning

4. Monitor the progress of the tuning

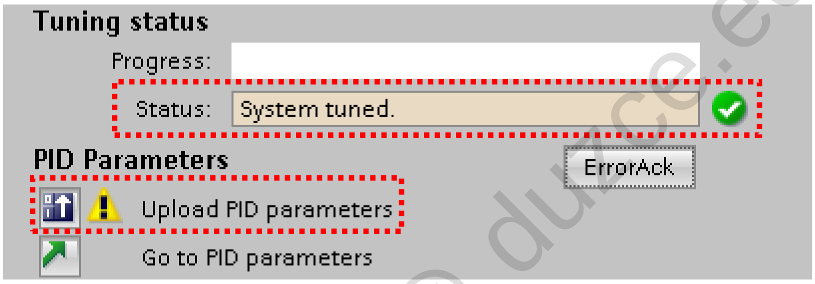


Tuning status

Progress:

Status: Pretuning in progress.

5. After the system is tuned, click on "Upload PID parameters", in order to save the determined PID parameters in your project.



Tuning status

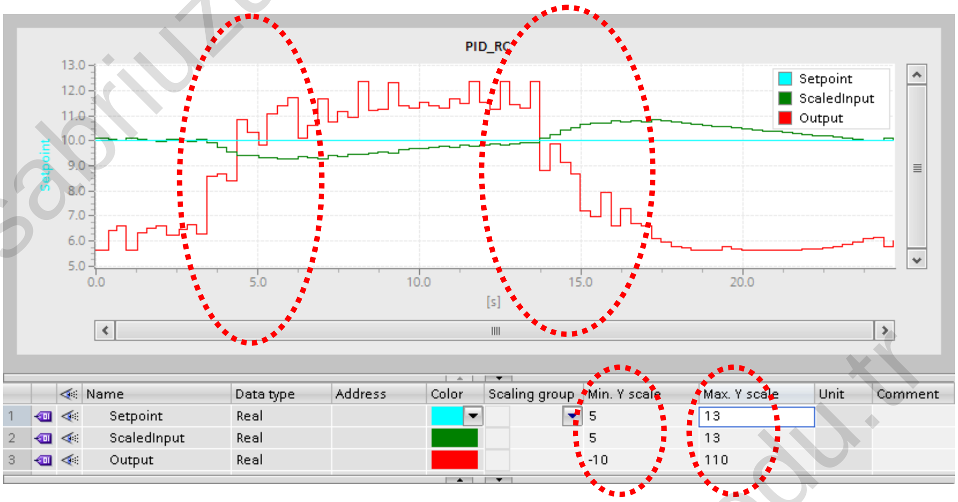
Progress:

Status: System tuned. ✓

PID Parameters

6. Monitor the measured value trends of actual value (green) and manipulated value (red) and switch in the disturbance and after approximately 5 seconds, switch it off again.

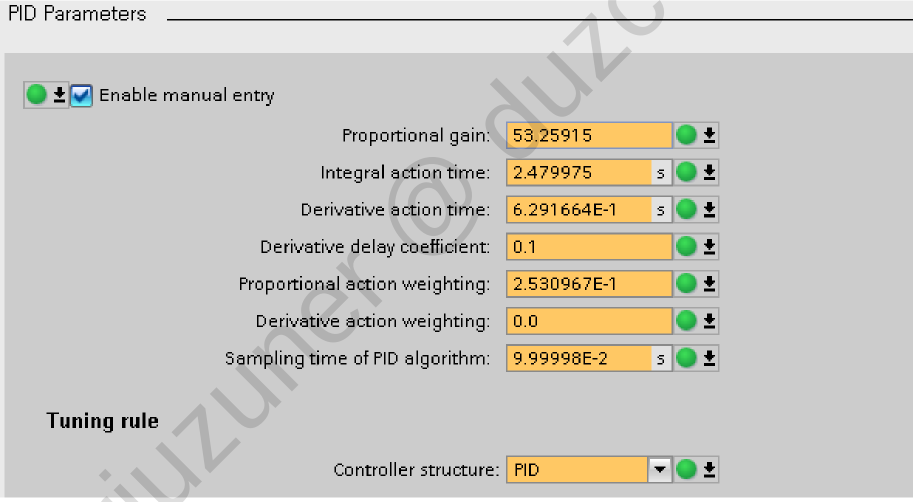
   Result: After switching on the disturbance, the manipulated value immediately shoots up to compensate for the dropping of the voltage at the capacitor. While the disturbance is pending, the manipulated value remains at a high level (approximately 85%).
   When the disturbance is switched off, an overshoot of the actual value develops, whereupon the manipulated value immediately drops.



	Name	Data type	Address	Color	Scaling group	Min. Y scale	Max. Y scale	Unit	Comment
1	Setpoint	Real		🟦		5	13		
2	ScaledInput	Real		🟩		5	13		
3	Output	Real		🟥		-10	110		

7. Check the adoption of the PID parameters in the configuration of the technology object "PID_RC"

   PLC_2 → Technology objects → PID_RC → Configuration → PID Parameters



PID Parameters

☑ Enable manual entry

Proportional gain: 53.25915

Integral action time: 2.479975 s

Derivative action time: 6.291664E-1 s

Derivative delay coefficient: 0.1

Proportional action weighting: 2.530967E-1

Derivative action weighting: 0.0

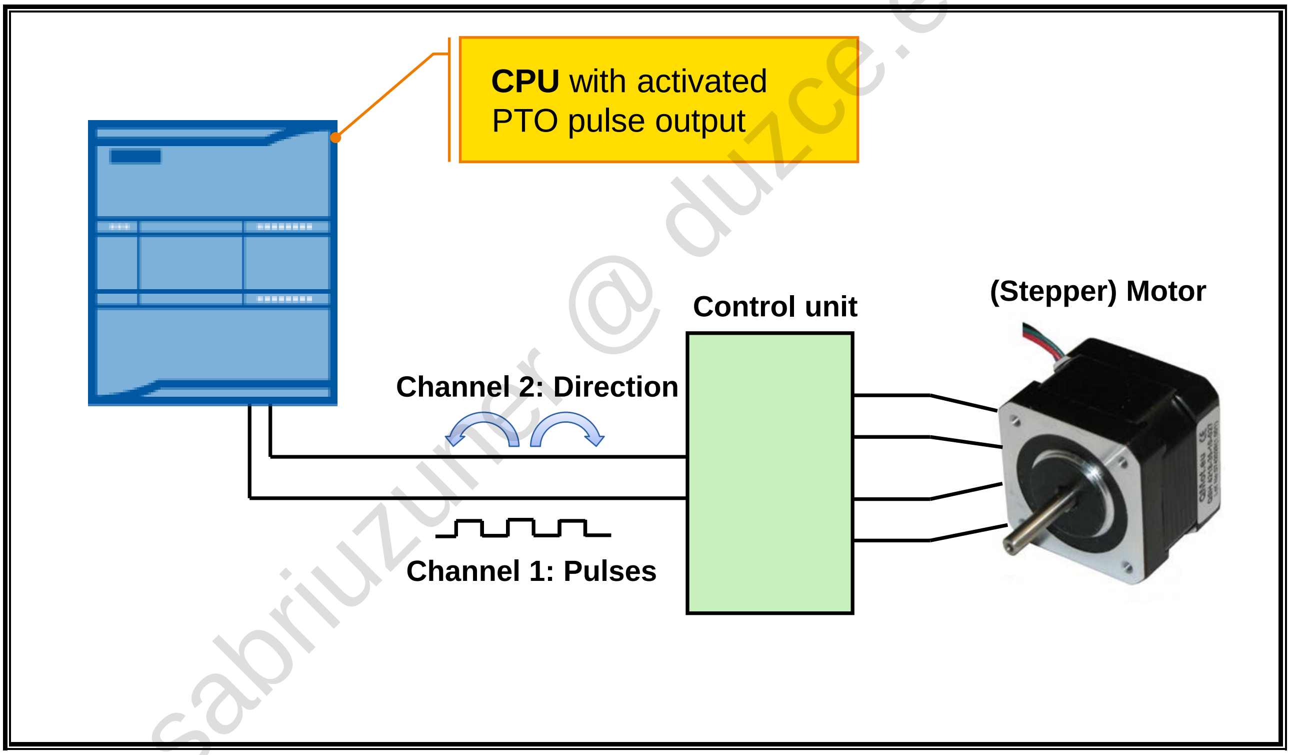
Sampling time of PID algorithm: 9.99998E-2 s

Tuning rule

Controller structure: PID

8. Save your project.

8.6. Introduction to the "Axis" Technology Object (Controlling the Stepper Motor)



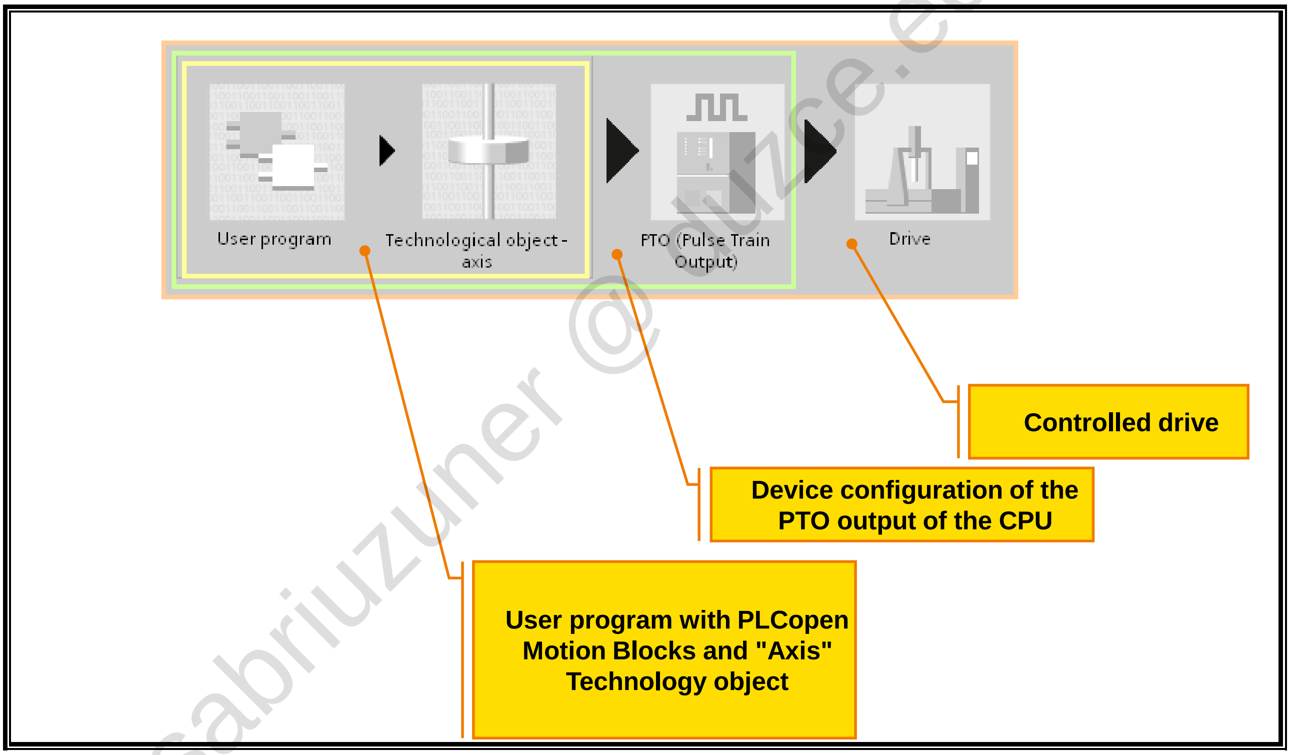
"Axis" Technology Object

The "Axis" technology object represents an axis in the controller and is suitable for controlling stepper motors and servo motors with pulse interface. The "Axis" technology object is controlled via Motion Control instructions.

Suitable are all drives or control units which support a control via a pulse/direction interface.

Typical areas of use are adjustable axes and operating axes as well as feed axes and transport axes. These are used, for example, in the steel, automobile and food and beverage industry and are used, among others, in packaging machines.

8.6.1. Principle of Axis Control


User program
Technological object - axis
PTO (Pulse Train Output)
Drive
Controlled drive
Device configuration of the PTO output of the CPU
User program with PLCopen Motion Blocks and "Axis" Technology object

Principle of Axis Control

Within the S7-1200 there is a model grouped into four sections for the control of axes in which you are supported by wizards and diagnostic screens for the commissioning and diagnoses of axes.

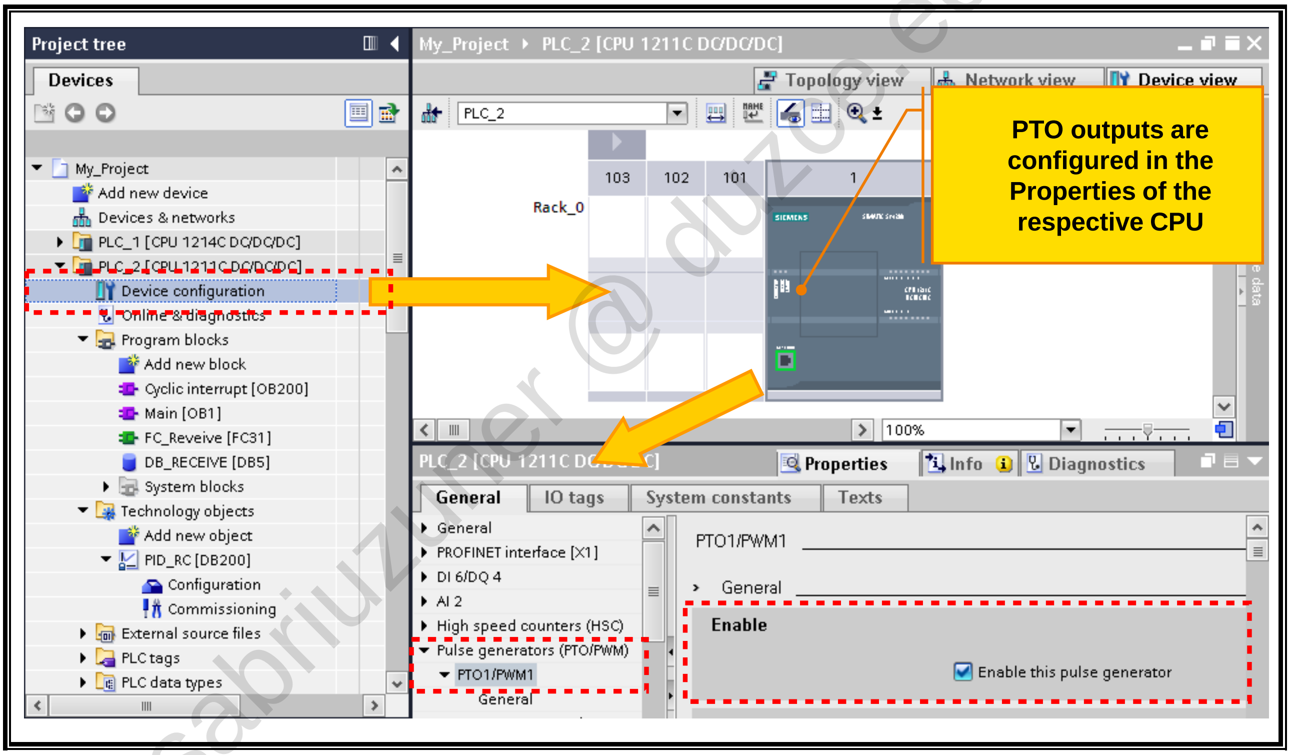
- User Program
  The user program utilizes standardized Motion Control instructions for the control of the "Axis" technology object and thus the axis or the drive.

- Technology Object "Axis"
  The "Axis" technology object represents an axis in the controller. Through dialog boxes (wizard), the parameters of the axis can very easily be specified. In the proper sense, the technology object is a data block with an exactly defined structure in which the parameters input by the user are entered. The specification of the DB is then required for the programming in the user program for the Motion Control instructions.

- PTO (Output)
  The PTO (output) is activated in the CPU device configuration and then assigned to the technology object "Axis". The output is subsequently controlled by the instructions in the user program.

- Drive
  Suitable are all drives or control units which support a control via a pulse/direction interface.

8.6.2. Configuring a PTO Output (1)

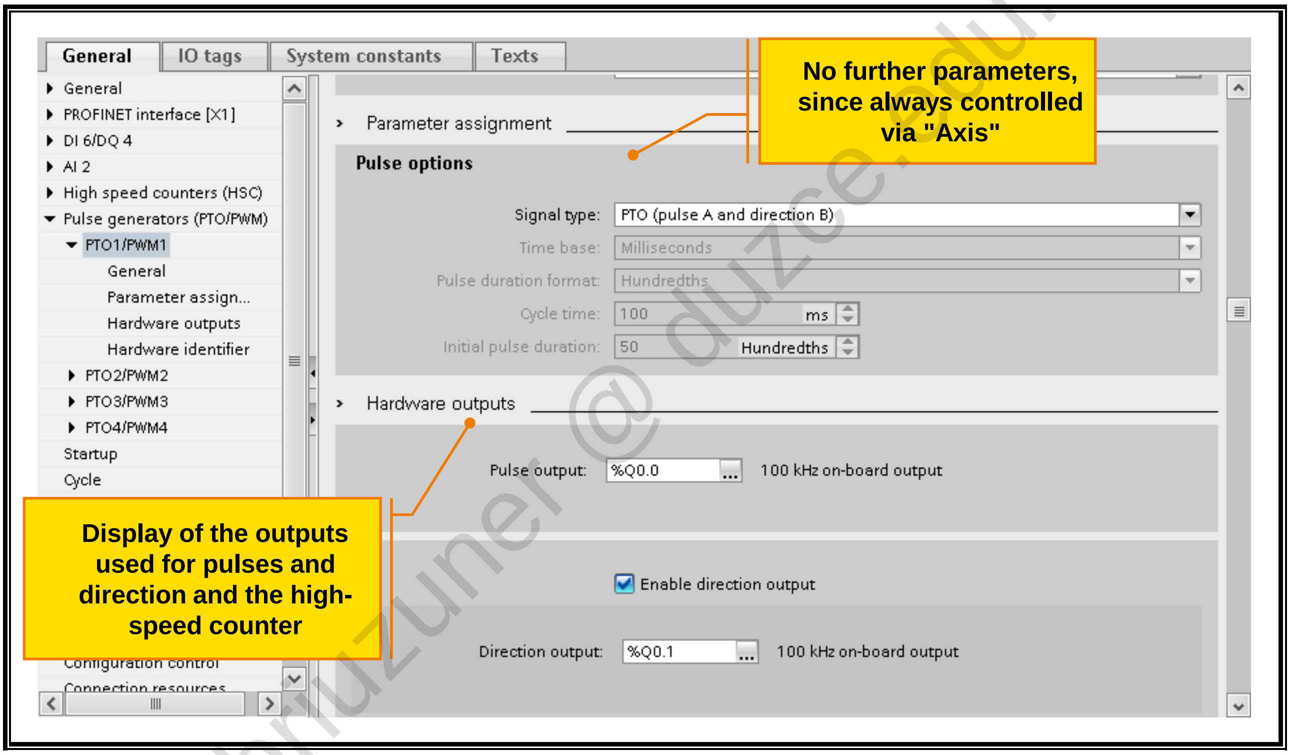


PTO outputs are configured in the Properties of the respective CPU

Configuring a PTO (Output)

PTOs are activated via the Device configuration of the respective CPU. In the Properties window "Pulse generators", the respective pulse generator must first of all be activated.

8.6.3. Configuring a PTO Output (2)



Parameter assignment

Pulse options

Signal type: PTO (pulse A and direction B)

Time base: Milliseconds

Pulse duration format: Hundredths

Cycle time: 100 ms

Initial pulse duration: 50 Hundredths

Hardware outputs

Pulse output: %Q0.0 100 kHz on-board output

☑ Enable direction output

Direction output: %Q0.1 100 kHz on-board output

No further parameters, since always controlled via "Axis"

Display of the outputs used for pulses and direction and the high-speed counter

Configuring a PTO Output

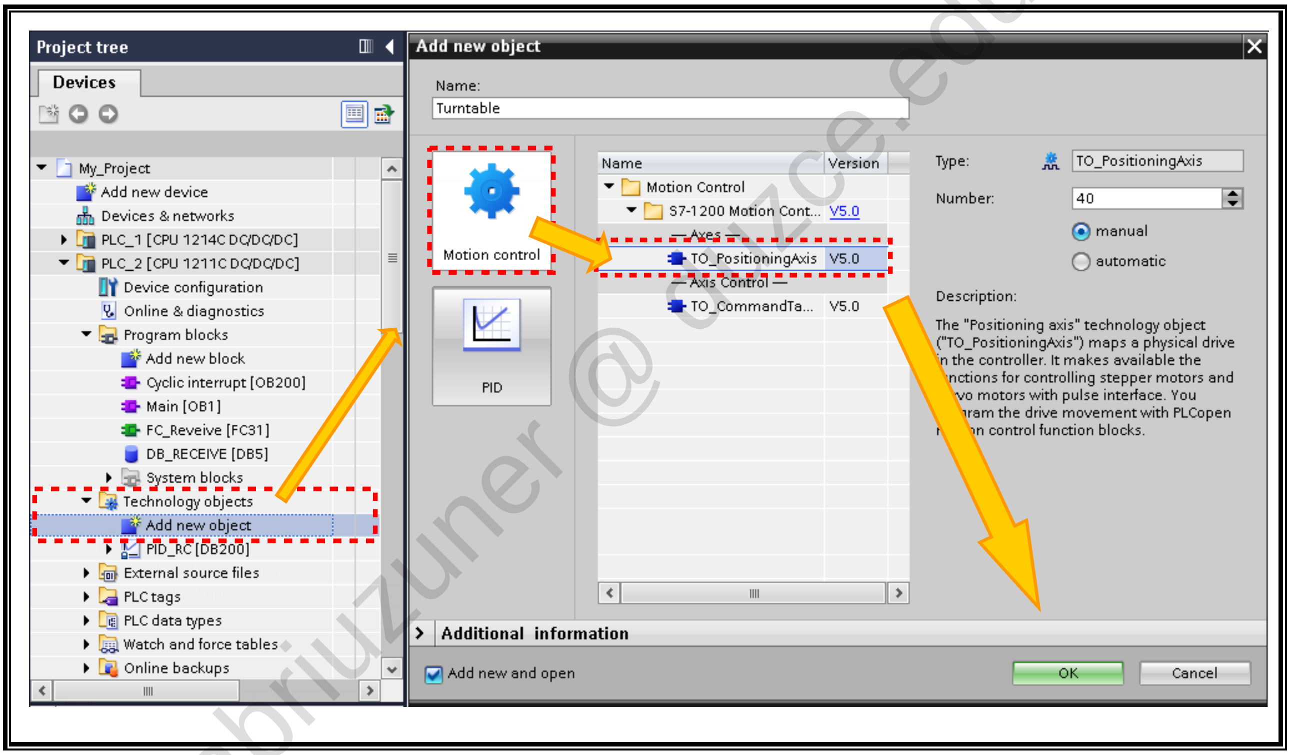
In the second step, the type of pulse generator must be selected. You can choose between "PTO" and "PWM".

To control an axis, you have to choose "PTO". Since PTO outputs are always controlled via the "Axis" technology object, no further settings can be made in the Device configuration.

There are 4 Possibilities for Controlling:

- PTO (pulse A and direction B)
  A pulse output and a direction output are used to control the stepper motor.
- PTO (count up A, count down B)
  One pulse output each for movement in positive direction and negative direction are used to control the stepper motor.
- PTO (A/B phase-shifted)
  Both pulse outputs for Phase A and for Phase B use the same frequency.
  On the drive side, the interval of the pulse outputs is evaluated as a step.
  The phase shifting between Phase A and Phase B determines the direction of movement.
- PTO (A/B phase-shifted four-fold)
  Both pulse outputs for Phase A and for Phase B use the same frequency.
  On the drive side, all rising and all falling edges of Phase A and Phase B are evaluated as a step.
  The phase shifting between Phase A and Phase B determines the direction of movement.

8.7. Creating a "Positioning Axis" Technology Object



Project tree

Devices

- My_Project
  - Add new device
  - Devices & networks
  - PLC_1 [CPU 1214C DC/DC/DC]
  - PLC_2 [CPU 1211C DC/DC/DC]
    - Device configuration
    - Online & diagnostics
    - Program blocks
      - Add new block
      - Cyclic interrupt [OB200]
      - Main [OB1]
      - FC_Reveive [FC31]
      - DB_RECEIVE [DB5]
      - System blocks
    - Technology objects
      - Add new object
      - PID_RC [DB200]
    - External source files
    - PLC tags
    - PLC data types
    - Watch and force tables
    - Online backups

Add new object

Name: Turntable

Name	Version
Motion Control	
S7-1200 Motion Cont...	V5.0
— Axes —	
TO_PositioningAxis	V5.0
— Axis Control —	
TO_CommandTa...	V5.0

Type: TO_PositioningAxis

Number: 40

● manual

○ automatic

Description:

The "Positioning axis" technology object ("TO_PositioningAxis") maps a physical drive in the controller. It makes available the nctions for controlling stepper motors and vo motors with pulse interface. You ram the drive movement with PLCopen n control function blocks.

> Additional information

☑ Add new and open

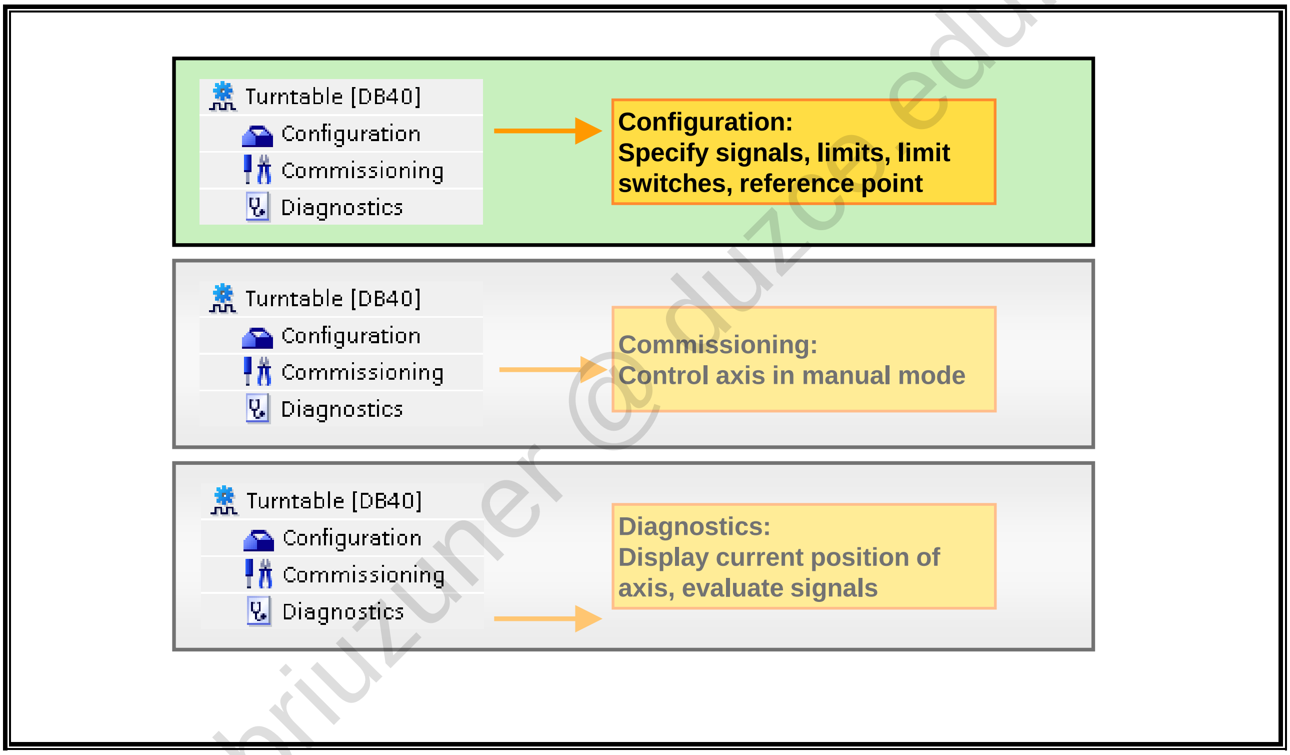
Creating a "Positioning Axis" Technology Object

Just as with PID (controller), the "Add new object" dialog box (wizard) in the "Technology objects" folder is called for the creation.

First of all "Motion" is selected and then the object "TO_PositioningAxis".

With a click on "OK", the configuration wizard of the axis is started.

8.7.1. Properties of "Axis": Configuration



Configuration:
Specify signals, limits, limit switches, reference point

Commissioning:
Control axis in manual mode

Diagnostics:
Display current position of axis, evaluate signals

Properties of "Axis"

After the "Axis" technology object has been created, there are three selection possibilities available for handling:

- **Configuration**
  - Selection of the PTO (output) to be used and configuration of the drive interface
  - Properties of the mechanics and gear ratio of the drive (or the machine or plant)
  - Properties for position monitoring, for dynamic parameters and for referencing (homing)

  The configuration is stored in the data block of the technology object

- **Commissioning**

  With the "Commissioning" tool, the functioning of the axis is tested without having to have created a user program. When you start this tool, the Control panel opens. The following commands are available in the Control panel:

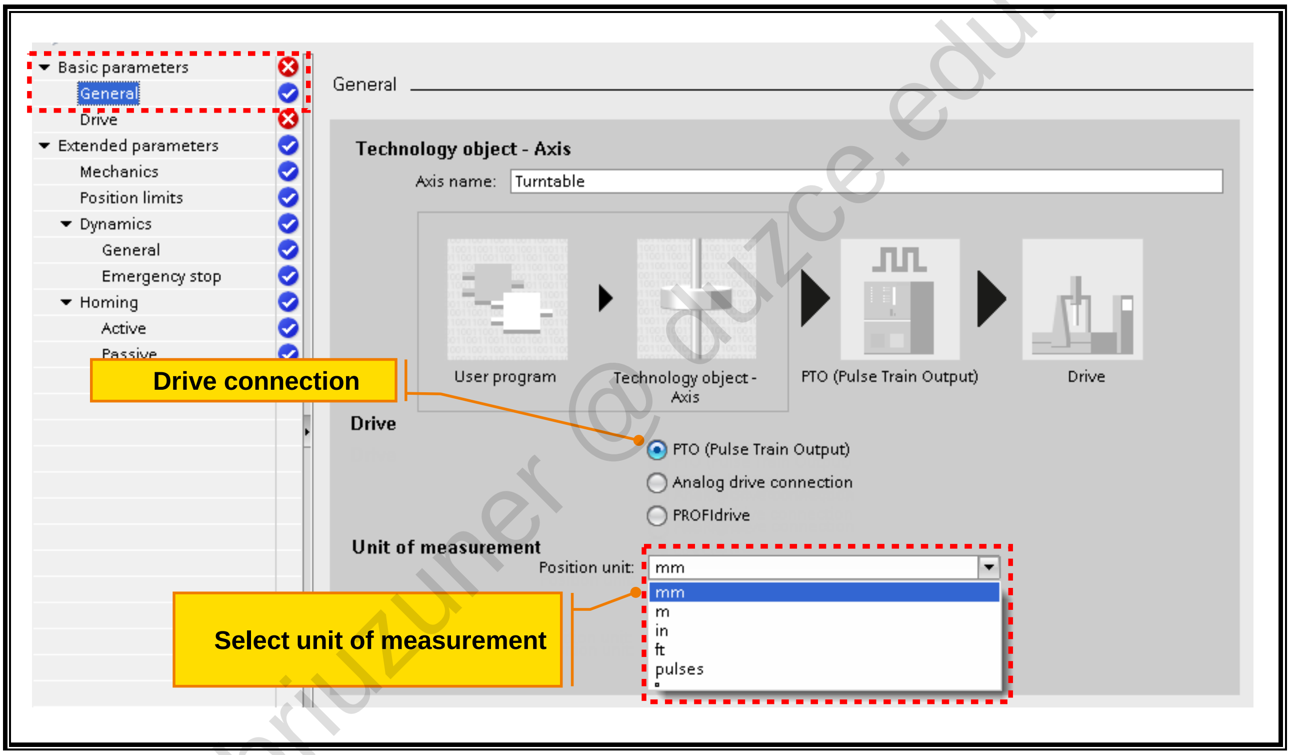
  - Enable and Disable the axis
  - Traversing the axis in Jog mode
  - Absolute and relative positioning of the axis
  - Referencing (homing) the axis
  - Acknowledgement of errors

- **Diagnostics**

  With the "Diagnostics" tool you check the current Status and Error information of axis and drive.

In the following, the configuration of the axis is presented.

8.7.1.1. Configuring an "Axis" (1)



Axis name: Turntable

Drive

● PTO (Pulse Train Output)
○ Analog drive connection
○ PROFIdrive

Unit of measurement

Position unit: mm

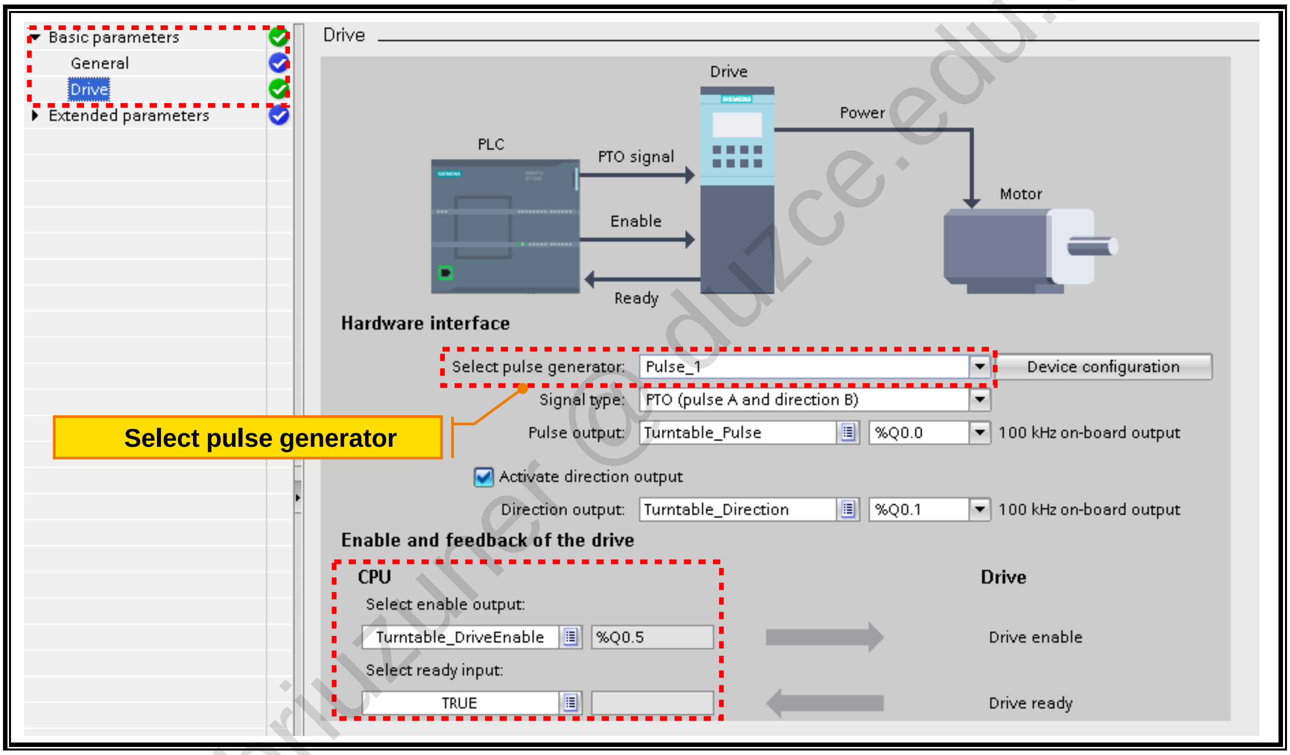
mm
m
in
ft
pulses

Basic Parameters / General

The properties of "Axis" are divided into Basic and Extended parameters.

In the basic parameters, under General, the drive connection and the unit of measurement that is used are set.

8.7.1.2. Configuring an "Axis" (2)



Drive

Hardware interface

Select pulse generator: Pulse_1

Signal type: PTO (pulse A and direction B)

Pulse output: Turntable_Pulse %Q0.0 100 kHz on-board output

☑ Activate direction output

Direction output: Turntable_Direction %Q0.1 100 kHz on-board output

Enable and feedback of the drive

Select enable output: Turntable_DriveEnable %Q0.5 → Drive enable

Select ready input: TRUE ← Drive ready

Basic Parameters - Drive

In the Basic parameters - Drive, the hardware interface of "Axis" as well as the output for the ‘drive enable’ and the input for the feedback "Drive ready" of the drive is specified.

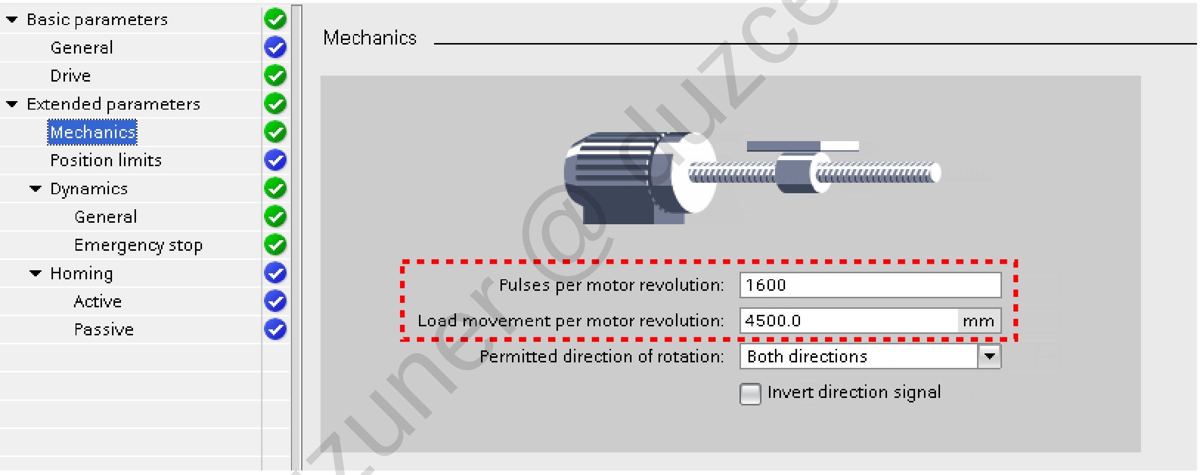
Enable and Feedback of the Drive

The ‘Drive enable’ is controlled by the Motion Control instruction "MC_Power" and it issues the drive the ‘Power Enable’. The signal for the drive is provided via the output which has to be configured.

If the drive is ready to execute movements after it receives the ‘Drive enable’, it signals "Drive ready" to the CPU. The signal "Drive ready" is reported back to the CPU via the input which has to be configured.

☞ If the drive doesn’t have any such interfaces, the parameters do not have to be configured. In this case, select the "TRUE" value for the Ready input.

8.7.1.3. Configuring an "Axis" (3)



Pulses per motor revolution: 1600

Load movement per motor revolution: 4500.0 mm

Permitted direction of rotation: Both directions

□ Invert direction signal

Mechanics

- Pulses per motor revolution

  In this field you specify how many pulses the motor requires for one revolution.

- Distance (Load movement) per motor revolution

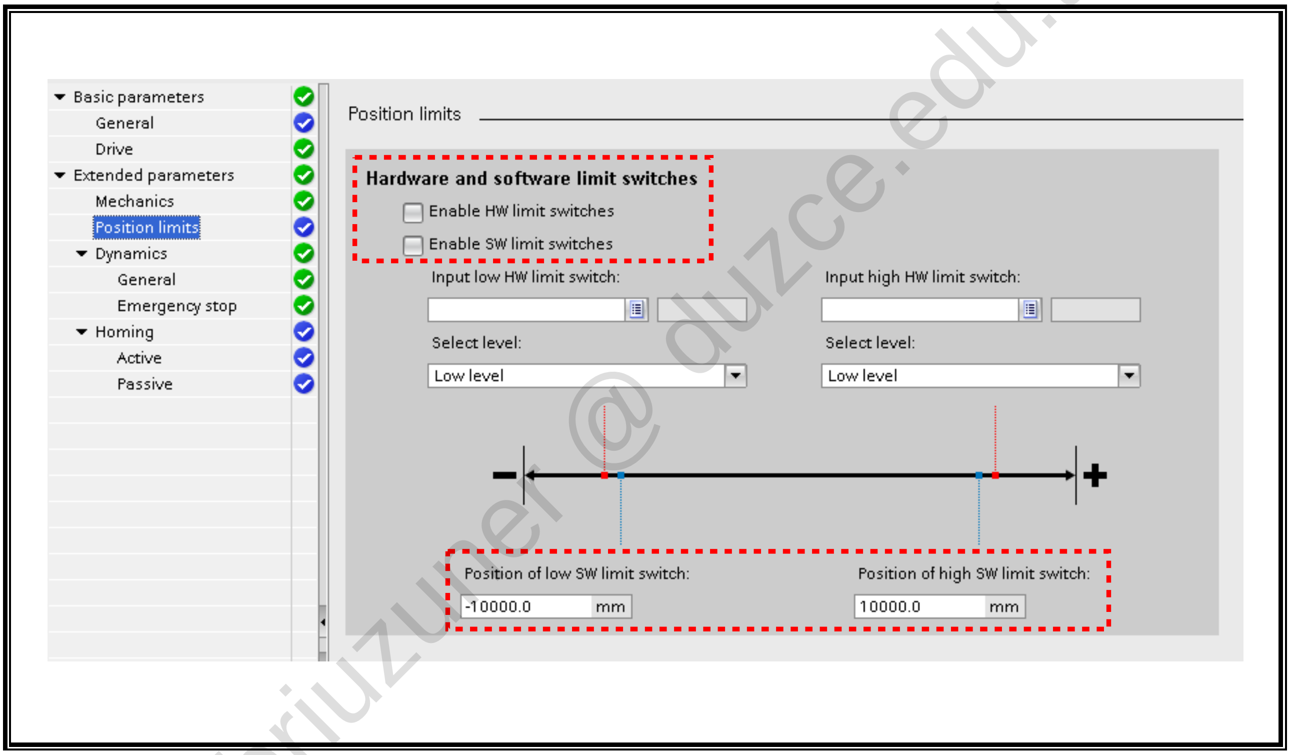
  In this field you specify what distance the mechanics of the system covers per motor revolution.

- Invert direction signal

  Through the checkbox "Invert direction signal" you can adjust the direction output to the direction logic of the drive.

Invert direction signal: deactivated	Invert direction signal: activated
0 V level = negative direction 5 V / 24 V level = positive direction	0 V level = positive direction 5 V / 24 V level = negative direction

8.7.1.4. Configuring an "Axis" (4)



Position limits

Hardware and software limit switches

□ Enable HW limit switches

□ Enable SW limit switches

Input low HW limit switch:

Select level: Low level

Input high HW limit switch:

Select level: Low level

Position of low SW limit switch: -10000.0 mm

Position of high SW limit switch: 10000.0 mm

Hardware Limit Switches

Input low / high HW limit switch

Through the drop-down lists, you can set the digital hardware limit switches. The inputs must be interrupt-capable. As inputs for the HW limit switches, the digital on-board CPU inputs and the digital inputs of an inserted signal board are available.

A filter time of the digital inputs of the CPU is set to 6.4 ms by default. This can lead to undesired delays when used as HW limit switches. In this case, the delay time must be shortened accordingly.

The filter time can be set in the Devices configuration of the digital inputs under "Input filters".

Software Limit Switches

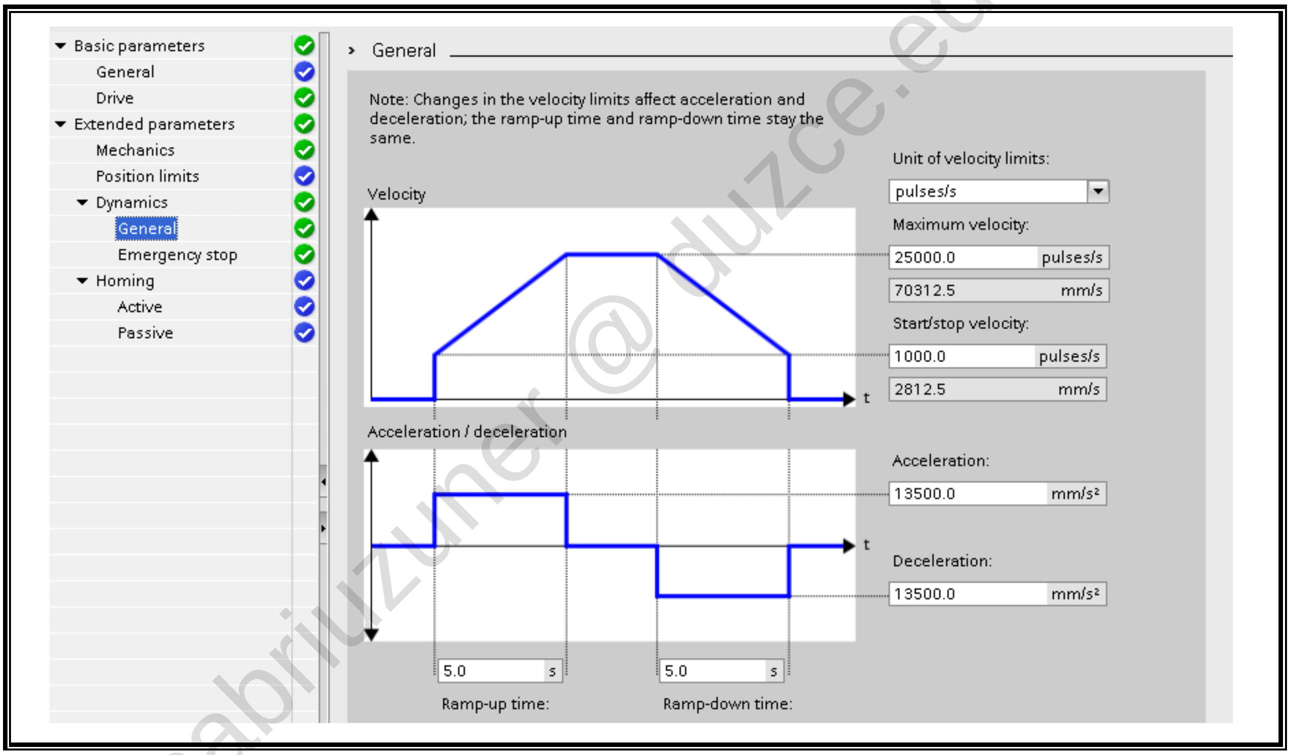
The software limit switches are activated via the checkbox "Enable SW limit switches". They are purely virtual and can be specified here as the distance travelled from the zero point.

The software limit switches are defined through the input fields "Position of low/high SW limit switch".

The value of the high software limit switch must be greater than or equal to the value of the low software limit switch.

Enabled software limit switches can only be operational with a referenced axis.

8.7.1.5. Configuring an "Axis" (5)



Note: Changes in the velocity limits affect acceleration and deceleration; the ramp-up time and ramp-down time stay the same.

Unit of velocity limits: pulses/s

Maximum velocity: 25000.0 pulses/s
70312.5 mm/s

Start/stop velocity: 1000.0 pulses/s
2812.5 mm/s

Acceleration: 13500.0 mm/s²

Deceleration: 13500.0 mm/s²

Ramp-up time: 5.0 s

Ramp-down time: 5.0 s

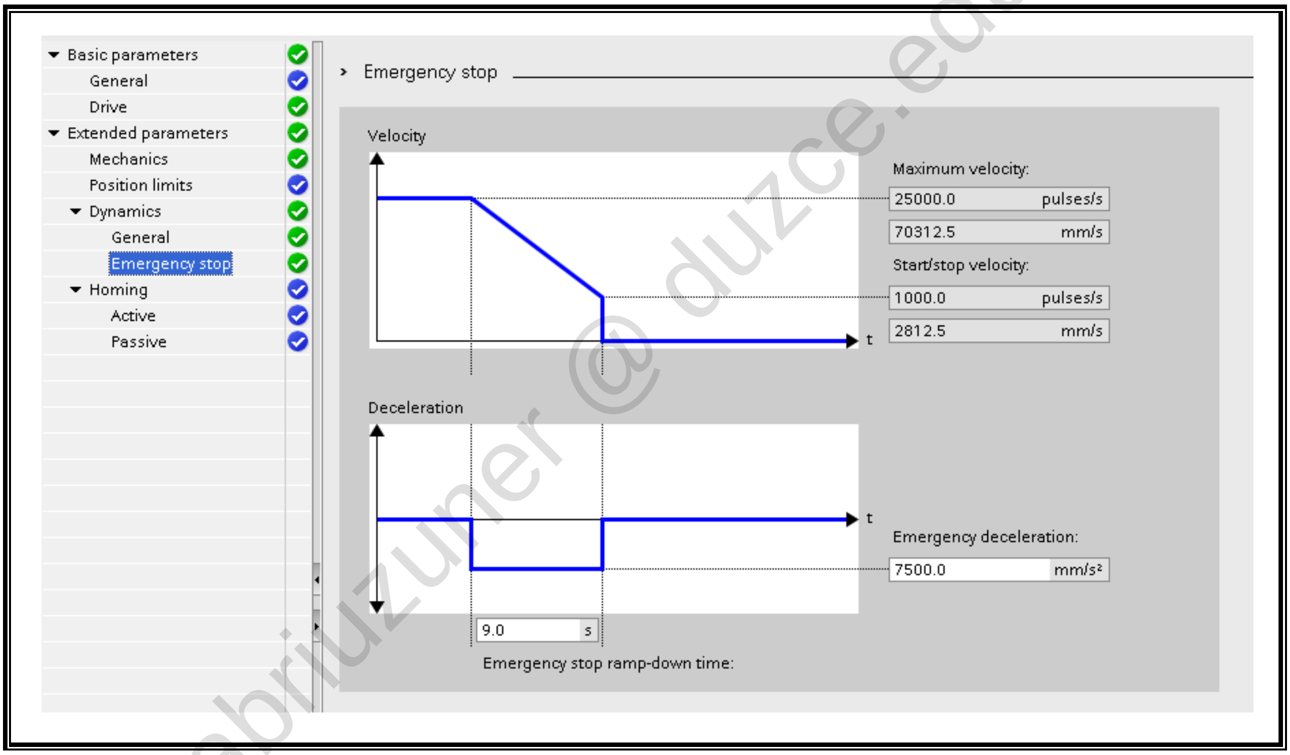
Dynamics / General

In the "Dynamics - General" configuration dialog you can specify the limits for the motion sequences:

- Velocity
  - In the field "Maximum velocity", the maximum allowable velocity of the axis is configured.
  - In the field "Start/stop velocity", the minimum allowable velocity of the axis is configured.
  - The value of the Maximum velocity must be greater than or equal to the value of the Start/stop velocity.
- Acceleration

  The desired acceleration values can either be specified directly via the fields "Acceleration" and "Deceleration" or indirectly via the specification of Ramp-up time or Ramp-down time.

8.7.1.6. Configuring an "Axis" (6)



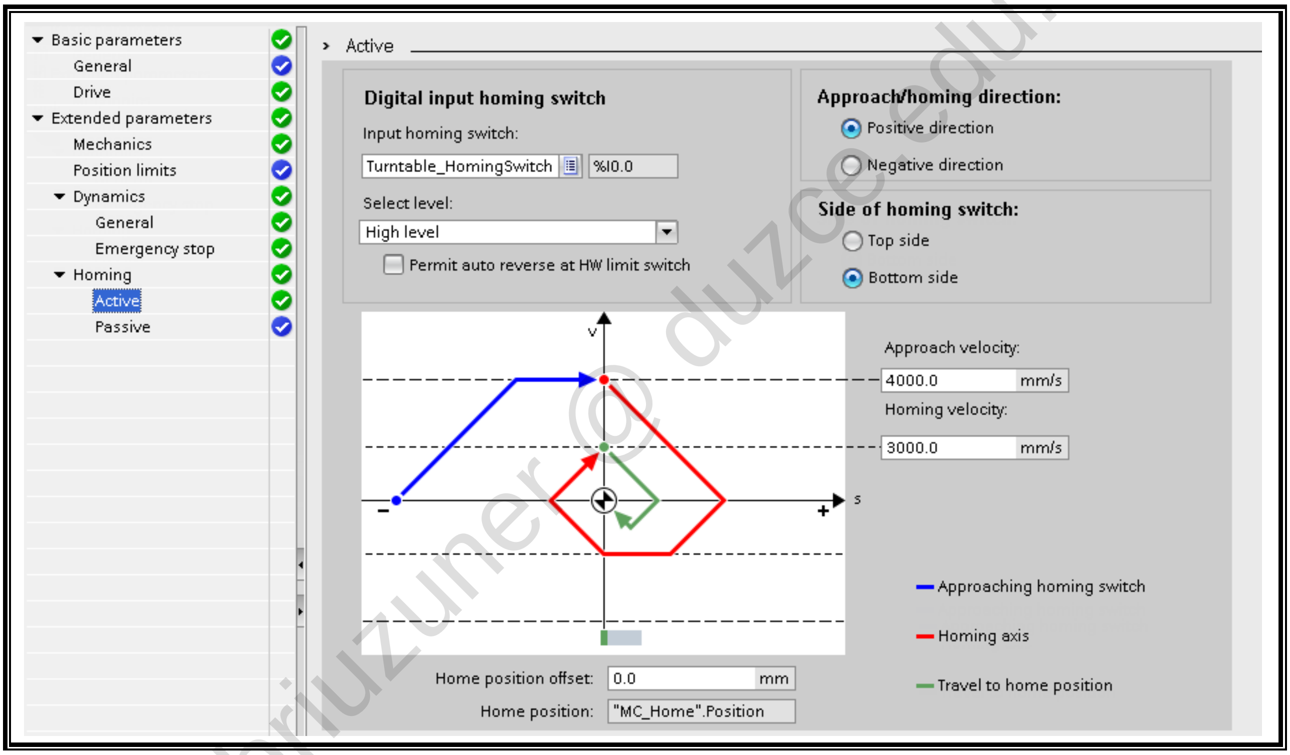
Maximum velocity: 25000.0 pulses/s
70312.5 mm/s
Start/stop velocity: 1000.0 pulses/s
2812.5 mm/s
Emergency deceleration: 7500.0 mm/s²
Emergency stop ramp-down time: 9.0 s

Dynamics / Emergency Stop

In the "Dynamics - Emergency stop" configuration wizard dialog, the Emergency deceleration of the axis can be set. In case of failure and when blocking the axis with the Motion-Control instruction "MC_Power" (Input parameter StopMode = 0), the axis is brought to a standstill with this deceleration.

⚠ The Emergency deceleration must be sufficiently large in order to bring the axis to a standstill in good time when there is an emergency (for example, when approaching the hardware limit switches, before reaching the mechanical stop).

8.7.1.7. Configuring an "Axis" (7)



- Basic parameters ✓
  - General ✓
  - Drive ✓
- Extended parameters ✓
  - Mechanics ✓
  - Position limits ✓
  - Dynamics ✓
    - General ✓
    - Emergency stop ✓
  - Homing ✓
    - Active ✓
    - Passive ✓

› Active

Digital input homing switch

Input homing switch: Turntable_HomingSwitch %I0.0

Select level: High level

□ Permit auto reverse at HW limit switch

Approach/homing direction:

● Positive direction

○ Negative direction

Side of homing switch:

○ Top side

● Bottom side

Approach velocity: 4000.0 mm/s

Homing velocity: 3000.0 mm/s

Home position offset: 0.0 mm

Home position: "MC_Home".Position

Homing (Referencing)

Compared to a closed-loop control system, you don’t have any feedback signal during traversing from which you can derive conclusions on the current position of the axis. For that reason, it is necessary that the axis, for example, every time the system is switched on, references a defined point, the so-called reference point. This reference point (also reference cam) is configured as an interrupt-capable input and marks the zero point of the axis. When the position of the reference point switch and the reference point position (home position) is different, the appropriate reference point offset is entered in the field "Home position offset". The axis approaches the reference (home) position with the referencing velocity.

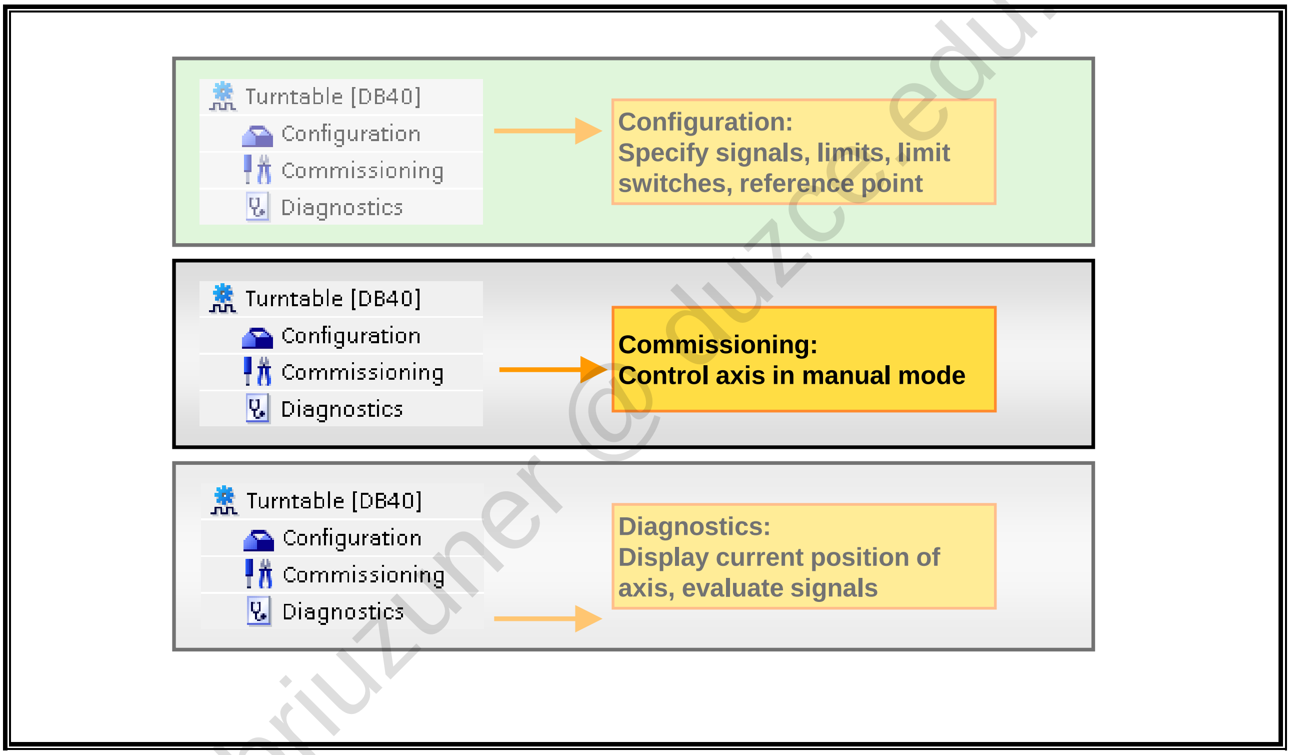
In order to approach the home position exactly there is the "active homing", whose sequence is represented in the picture above. The movement is divided into three steps:

- Seeking the referencing point (homing) switch (blue section of the graph)
  When active referencing is started, the axis accelerates to the configured "Approach velocity" and with it seeks the referencing point (homing) switch.

- Reference point travel (red section of the graph)
  When it detects the referencing point (homing) switch, the axis slows down in this example, turns around, in order to reference on the configured side of the referencing point (homing) switch with the configured "Homing velocity".

- Traversing the home position offset (green section of the graph)
  After referencing, the axis travels the distance of the home position offset with the referencing velocity. Arriving there, the axis finds itself in the home position which was specified at the input parameter "Position" of the Motion Control instruction "MC_Home".

8.7.2. Properties of "Axis": Commissioning



Configuration:
Specify signals, limits, limit switches, reference point

Commissioning:
Control axis in manual mode

Diagnostics:
Display current position of axis, evaluate signals

Properties of "Axis"

After the "Axis" technology object has been created, there are three selection possibilities available for handling:

- **Configuration**
  - Selection of the PTO (output) to be used and configuration of the drive interface
  - Properties of the mechanics and gear ratio of the drive (or the machine or plant)
  - Properties for position monitoring, for dynamic parameters and for referencing (homing)

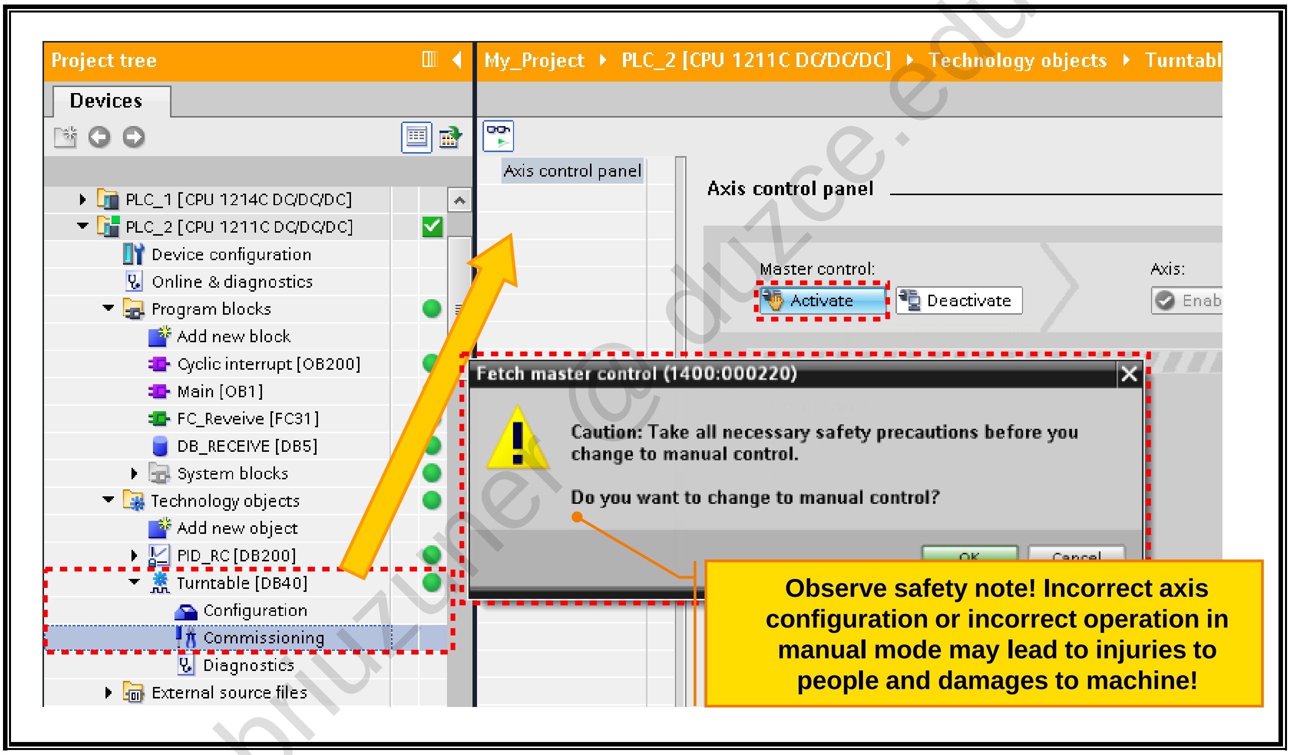
  The configuration is stored in the data block of the technology object
- **Commissioning**

  With the "Commissioning" tool, the functioning of the axis is tested without having to have created a user program. When you start this tool, the Control panel opens. The following commands are available in the Control panel:
  - Enable and Disable the axis
  - Traversing the axis in Jog mode
  - Absolute and relative positioning of the axis
  - Referencing (homing) the axis
  - Acknowledgement of errors
- **Diagnostics**

  With the "Diagnostics" tool you check the current Status and Error information of axis and drive.

In the following, the commissioning of the axis is presented.

8.7.3. Activating the Commissioning Panel



Fetch master control (1400:000220)

Caution: Take all necessary safety precautions before you change to manual control.

Do you want to change to manual control?

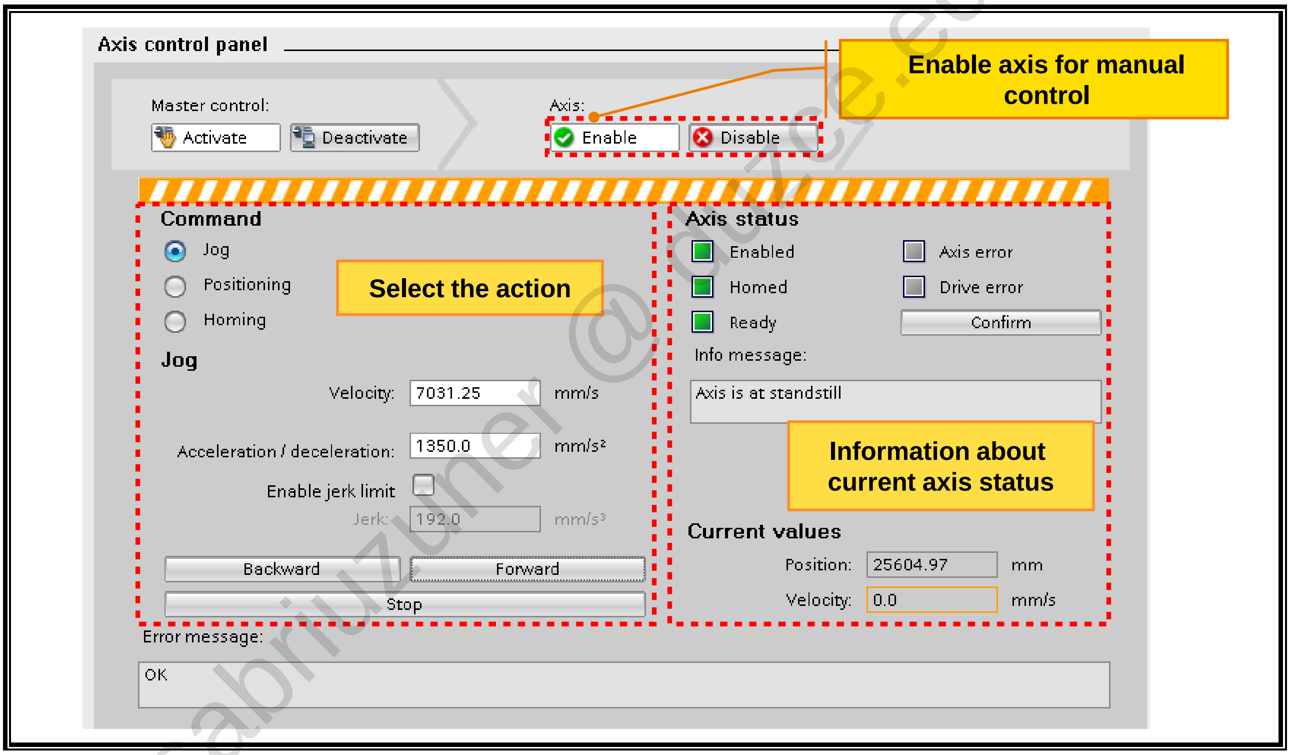
Observe safety note! Incorrect axis configuration or incorrect operation in manual mode may lead to injuries to people and damages to machine!

Commissioning Panel

In automatic mode, the Control Panel offers the opportunity to get an overview of the current status of the axis. The most important bits such as Enabled, Homed or Axis error as well as the current values on Position and Velocity are represented.

If the axis is to be controlled in manual mode, a warning appears which points out that all necessary safety precautions are to be taken since you are about to actively intervene in the process.

8.7.3.1. Using the Commissioning Panel (Manual Control)



Axis control panel

Master control: Activate | Deactivate

Axis: ✓ Enable | ☒ Disable

Enable axis for manual control

Command

● Jog

○ Positioning

○ Homing

Select the action

Jog

Velocity: 7031.25 mm/s

Acceleration / deceleration: 1350.0 mm/s²

Enable jerk limit □

Jerk: 192.0 mm/s³

Backward | Forward

Stop

Axis status

🟩 Enabled

🟩 Homed

🟩 Ready

□ Axis error

□ Drive error

Confirm

Info message:

Axis is at standstill

Information about current axis status

Current values

Position: 25604.97 mm

Velocity: 0.0 mm/s

Error message:

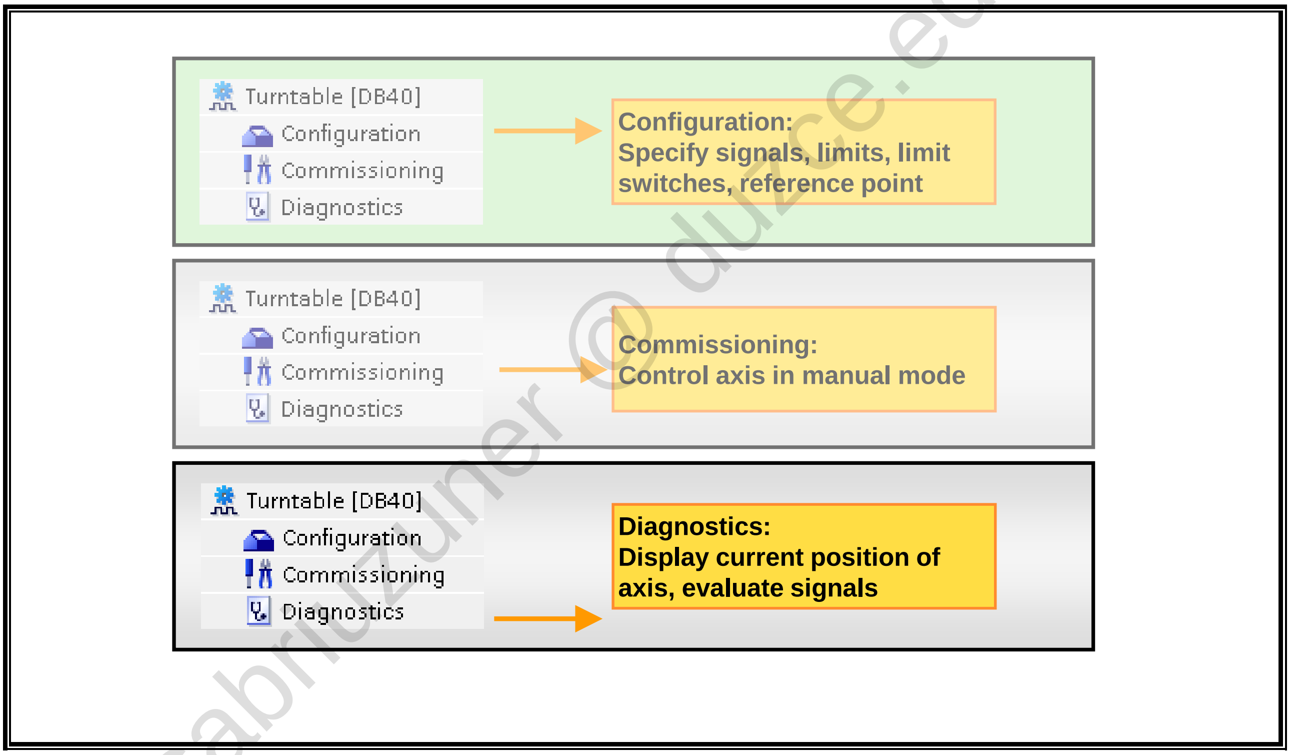
OK

Using the Commissioning Panel

If the axis is to change to manual control, the execution of the MC-Power command in the user program has to be deactivated first. Then, the manual control can be activated via the button "Enable".

Thereupon, the control passes from the user program to the Control Panel and it is possible to enable and disable the axis, move with the selected velocity in jog mode or to acknowledge errors of the control panel as soon as their cause is eliminated.

8.7.4. Properties of "Axis": Diagnostics



Configuration:
Specify signals, limits, limit switches, reference point

Commissioning:
Control axis in manual mode

Diagnostics:
Display current position of axis, evaluate signals

Properties of "Axis"

After the "Axis" technology object has been created, there are three selection possibilities available for handling:

- **Configuration**
  - Selection of the PTO (output) to be used and configuration of the drive interface
  - Properties of the mechanics and gear ratio of the drive (or the machine or plant)
  - Properties for position monitoring, for dynamic parameters and for referencing (homing)

  The configuration is stored in the data block of the technology object

- **Commissioning**

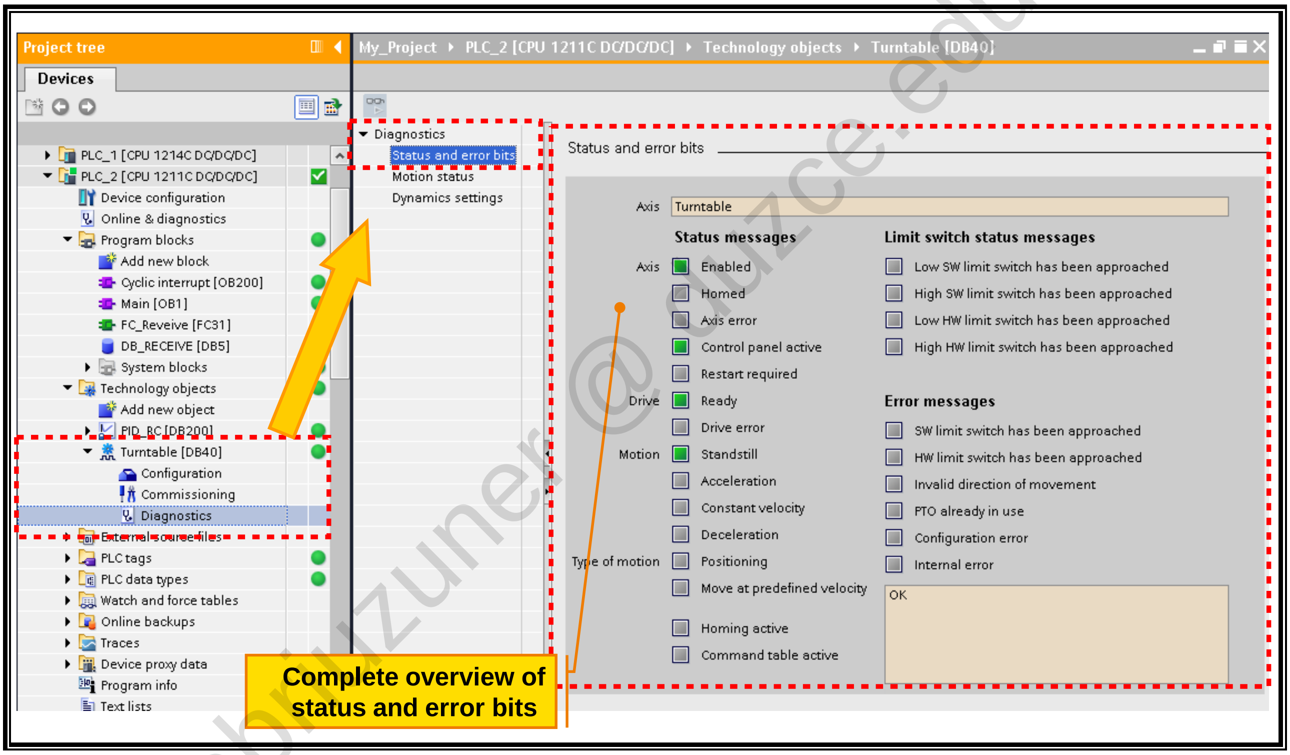
  With the "Commissioning" tool, the functioning of the axis is tested without having to have created a user program. When you start this tool, the Control panel opens. The following commands are available in the Control panel:
  - Enable and Disable the axis
  - Traversing the axis in Jog mode
  - Absolute and relative positioning of the axis
  - Referencing (homing) the axis
  - Acknowledgement of errors

- **Diagnostics**

  With the "Diagnostics" tool you check the current Status and Error information of axis and drive.

In the following, the diagnostics of the axis is presented.

8.7.4.1. Axis Diagnostics (1)



Status and error bits

Axis: Turntable

Status messages

Axis
- 🟩 Enabled
- □ Homed
- □ Axis error
- 🟩 Control panel active
- □ Restart required

Drive
- 🟩 Ready
- □ Drive error

Motion
- 🟩 Standstill
- □ Acceleration
- □ Constant velocity
- □ Deceleration

Type of motion
- □ Positioning
- □ Move at predefined velocity
- □ Homing active
- □ Command table active

Limit switch status messages
- □ Low SW limit switch has been approached
- □ High SW limit switch has been approached
- □ Low HW limit switch has been approached
- □ High HW limit switch has been approached

Error messages
- □ SW limit switch has been approached
- □ HW limit switch has been approached
- □ Invalid direction of movement
- □ PTO already in use
- □ Configuration error
- □ Internal error

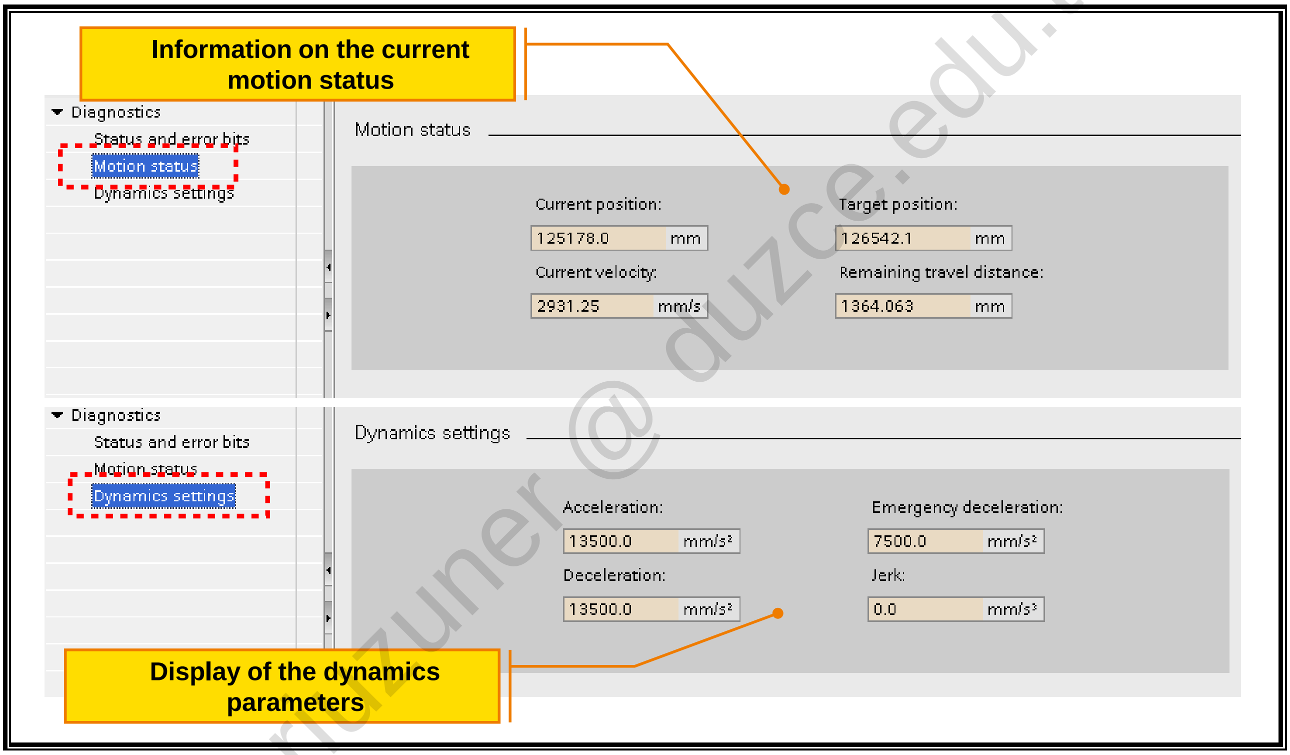
OK

Complete overview of status and error bits

Status and Error Bits

After the axis diagnostics is started, the current statuses of the axis are displayed under "Status and error bits". Among other things, you can read out the current axis and motion status. In addition, error events, such as, the reaching of limit switches are displayed.

8.7.4.2. Axis Diagnostics (2)



Information on the current motion status

Display of the dynamics parameters

Motion status

Current position: 125178.0 mm

Target position: 126542.1 mm

Current velocity: 2931.25 mm/s

Remaining travel distance: 1364.063 mm

Dynamics settings

Acceleration: 13500.0 mm/s²

Emergency deceleration: 7500.0 mm/s²

Deceleration: 13500.0 mm/s²

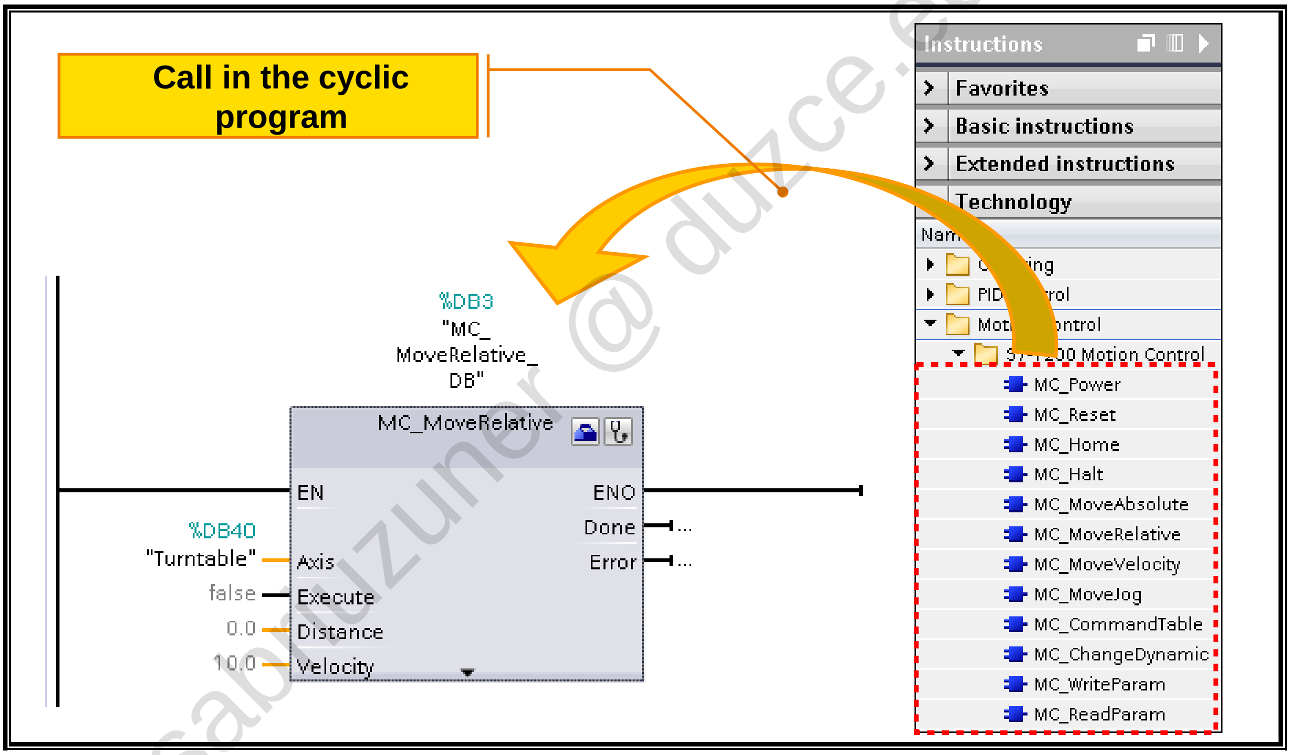
Jerk: 0.0 mm/s³

Motion Status and Dynamics Settings

- Through the menu point "Motion status" you can get information on the current movement. The values displayed are continuously updated.
- In the Dynamics settings, you can read out the configured values for Acceleration, Deceleration and Emergency deceleration.

☞ All values are available as read-only.

8.7.5. Blocks for Axis Control



Motion Control Instructions

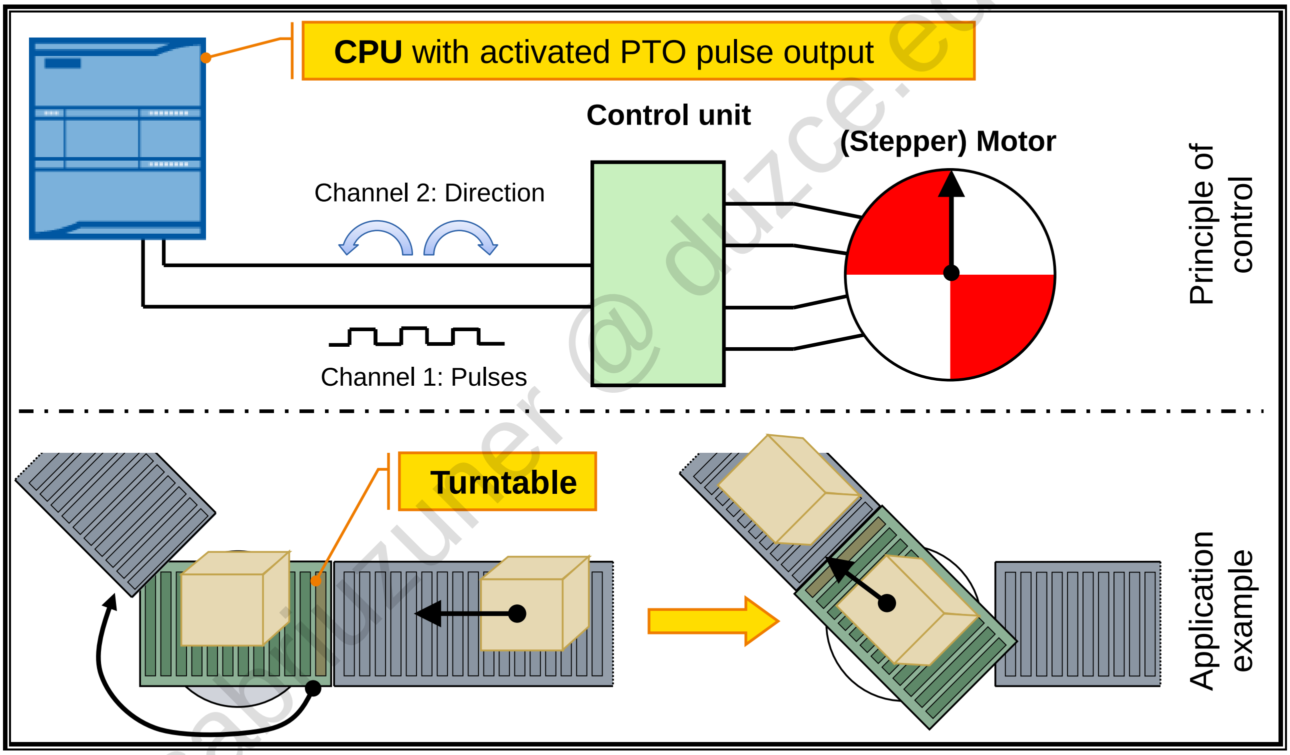
Through the Motion Control instructions, you control the axis from the user program. The instructions start Motion Control tasks which execute the desired functions.

The status of the Motion Control tasks as well as possible errors which occurred during processing can be queried at the output parameters of the Motion Control instructions. The following Motion Control instructions are available for selection:

Instruction	Function
MC_Power	Activate/deactivate axis
MC_Reset	Acknowledge error of the axis
MC_Home	Home (reference axis)
MC_Halt	Cancel all MC instructions (axis commands)
MC_MoveAbsolute	Move axis to an absolute position
MC_MoveRelative	Move axis to a position relative to the current one
MC_MoveVelocity	Move axis with a constant (defined) velocity
MC_MoveJog	Move axis with (manual) jog velocity
MC_CommandTable	Execute axis jobs as movement sequence
MC_ChangeDynamic	Change dynamic settings of the axis
MC_WriteRaram	Write variables of the positioning axis
MC_ReadParam	Continuously read movement data of a positioning axis

All blocks can be called in the cyclic program.

8.8. Task Description: Controlling a Stepper Motor



Task Description

The stepper motor of the training device is to be commissioned. For this, the "Turntable" technology object of the type "Axis" is to be created which is to be configured by you. On the hardware-side, the PTO (output) 1 of the CPU is used as well as a Boolean output for the specification of the direction.

The function block "FB_Turntable" (FB40) takes over the control of the axis. You are to call this function block in the user program.

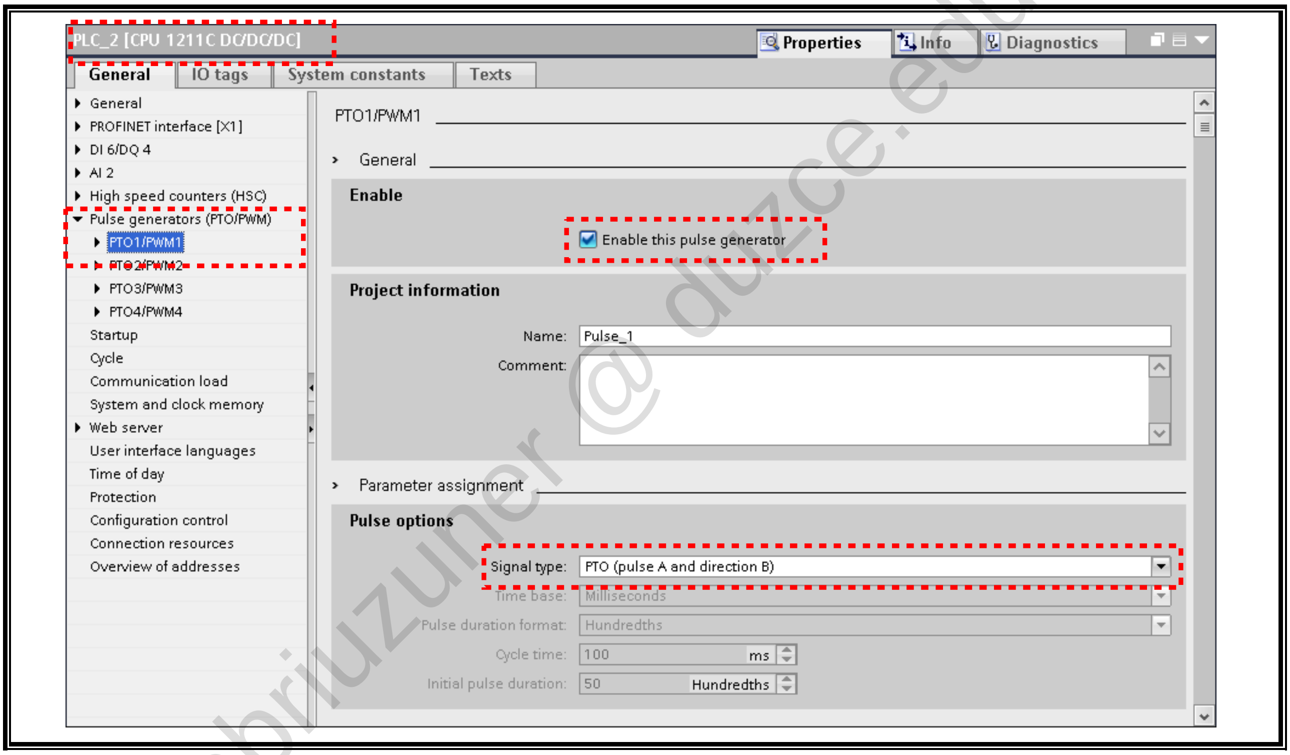
Scenario

A production piece is transported via a turn-lift table. The production pieces arrive at the lower level and are transported onto the turn-lift table (Position 1: 90°). Then, the turn-lift table approaches the upper level and executes a 225° turn to Position 2 (315°). Having arrived at the upper level, the production piece is moved off the turn-lift table and is transported away. Subsequently, the turn-lift table travels back in the opposite direction to the starting point (Position 1).

The starting point is approached for the first time after the system is switched on and subsequent referencing (homing) is done.

☞ The vertical movement was not programmed.

8.8.1. Exercise 4: Activating (Enabling) PTO 1 of the CPU



☑ Enable this pulse generator

Name: Pulse_1

Comment:

Signal type: PTO (pulse A and direction B)

Time base: Milliseconds

Pulse duration format: Hundredths

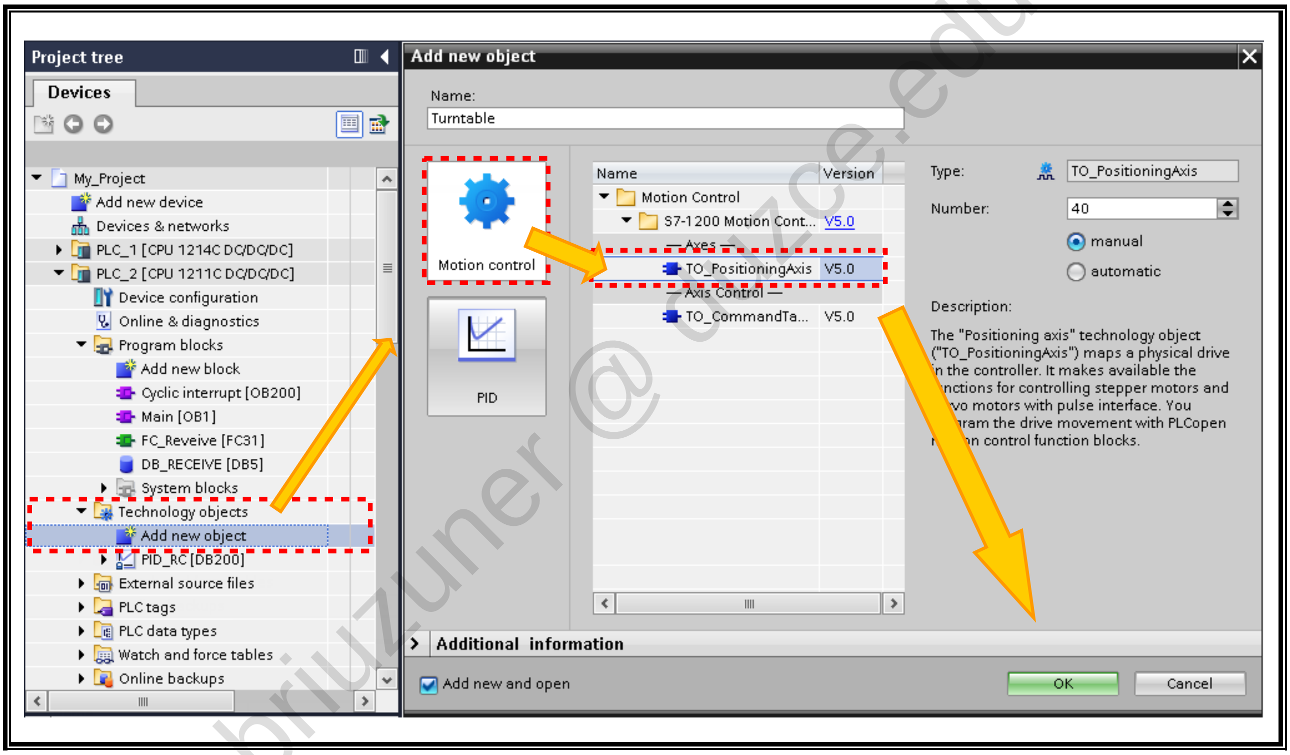
Cycle time: 100 ms

Initial pulse duration: 50 Hundredths

Task

The "Axis" technology object configured in the following accesses a PTO output. For this, activate (enable) the PTO 1 of PLC_2 as shown in the picture.

8.8.2. Exercise 5: Creating and Configuring the Technology Object "Axis"



Name: Turntable

Type: TO_PositioningAxis

Number: 40

● manual

○ automatic

Description:

The "Positioning axis" technology object ("TO_PositioningAxis") maps a physical drive in the controller. It makes available the functions for controlling stepper motors and servo motors with pulse interface. You program the drive movement with PLCopen motion control function blocks.

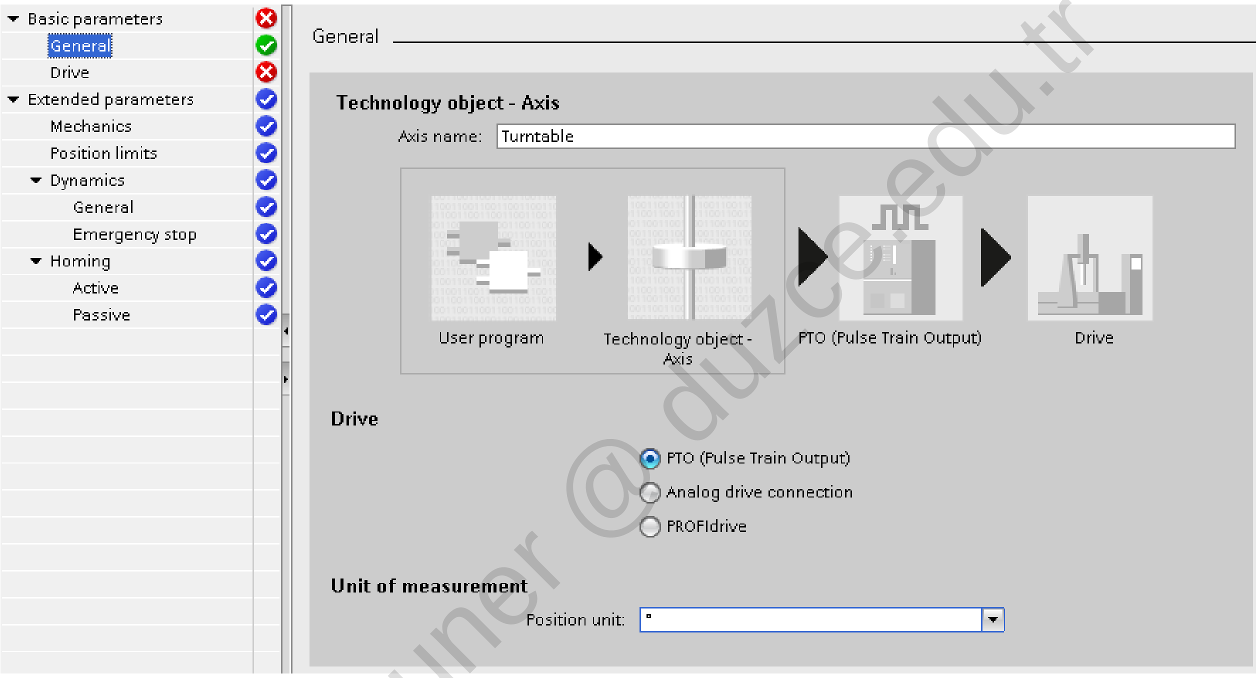
☑ Add new and open

Task

Create a technology object "Turntable" of the type "Axis" and configure it.

What to Do

1. Create a new technology object in PLC_2:
   PLC_2 → Technology objects → Add new object → Motion Control → TO_PositioningAxis
   - Name: Turntable
   - Number: 40
2. Implement the settings shown in the following:



- Basic parameters ✗
  - General ✓
  - Drive ✗
- Extended parameters ✓
  - Mechanics ✓
  - Position limits ✓
  - Dynamics ✓
    - General ✓
    - Emergency stop ✓
  - Homing ✓
    - Active ✓
    - Passive ✓

General

Technology object - Axis

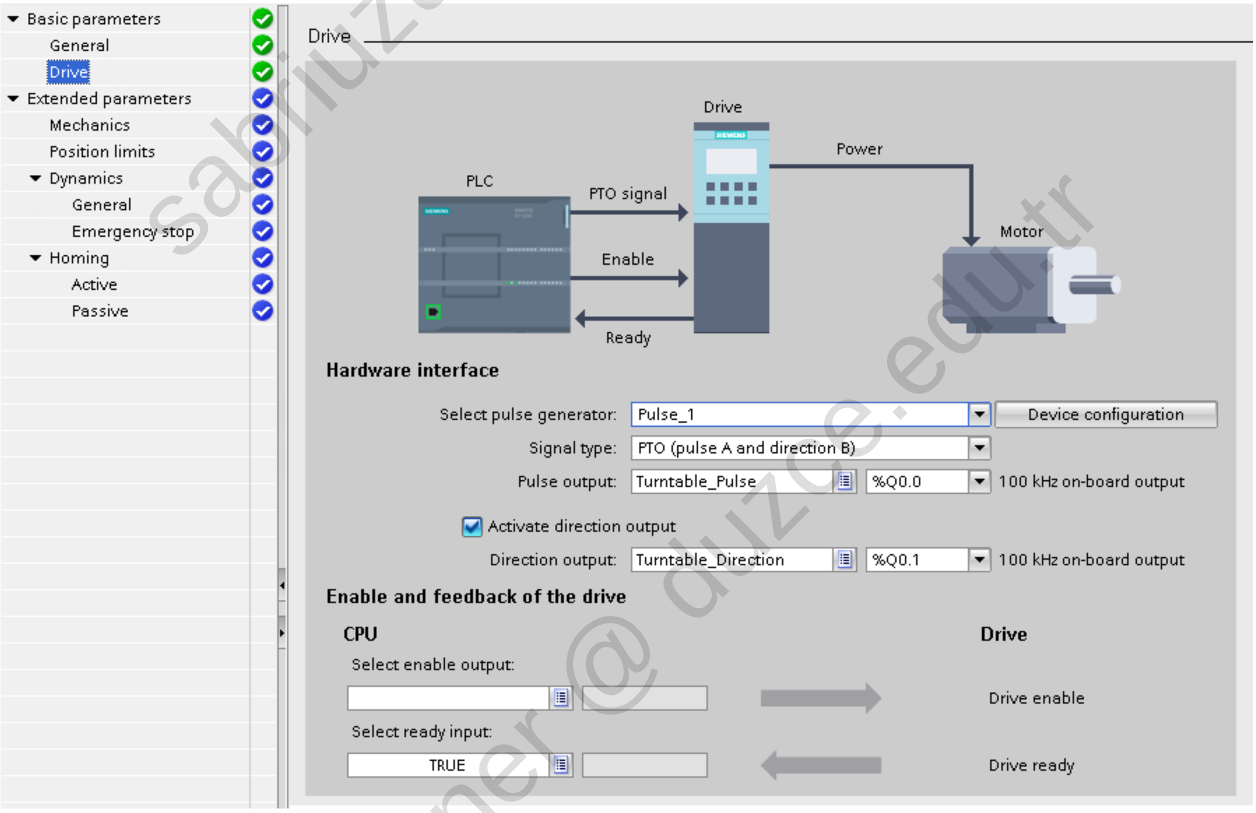
Axis name: Turntable

Drive

● PTO (Pulse Train Output)
○ Analog drive connection
○ PROFIdrive

Unit of measurement

Position unit: °



- Basic parameters ✓
  - General ✓
  - Drive ✓
- Extended parameters ✓
  - Mechanics ✓
  - Position limits ✓
  - Dynamics ✓
    - General ✓
    - Emergency stop ✓
  - Homing ✓
    - Active ✓
    - Passive ✓

Drive

Hardware interface

Select pulse generator: Pulse_1 [Device configuration]

Signal type: PTO (pulse A and direction B)

Pulse output: Turntable_Pulse %Q0.0 100 kHz on-board output

☑ Activate direction output

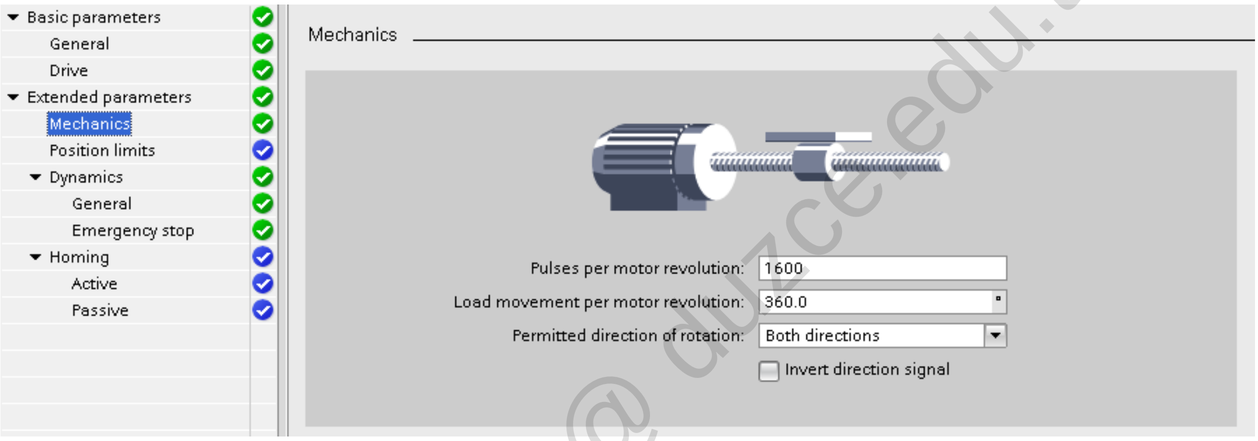
Direction output: Turntable_Direction %Q0.1 100 kHz on-board output

Enable and feedback of the drive

CPU **Drive**

Select enable output: → Drive enable

Select ready input: TRUE ← Drive ready



- Basic parameters ✓
  - General ✓
  - Drive ✓
- Extended parameters ✓
  - Mechanics ✓
  - Position limits ✓
  - Dynamics ✓
    - General ✓
    - Emergency stop ✓
  - Homing ✓
    - Active ✓
    - Passive ✓

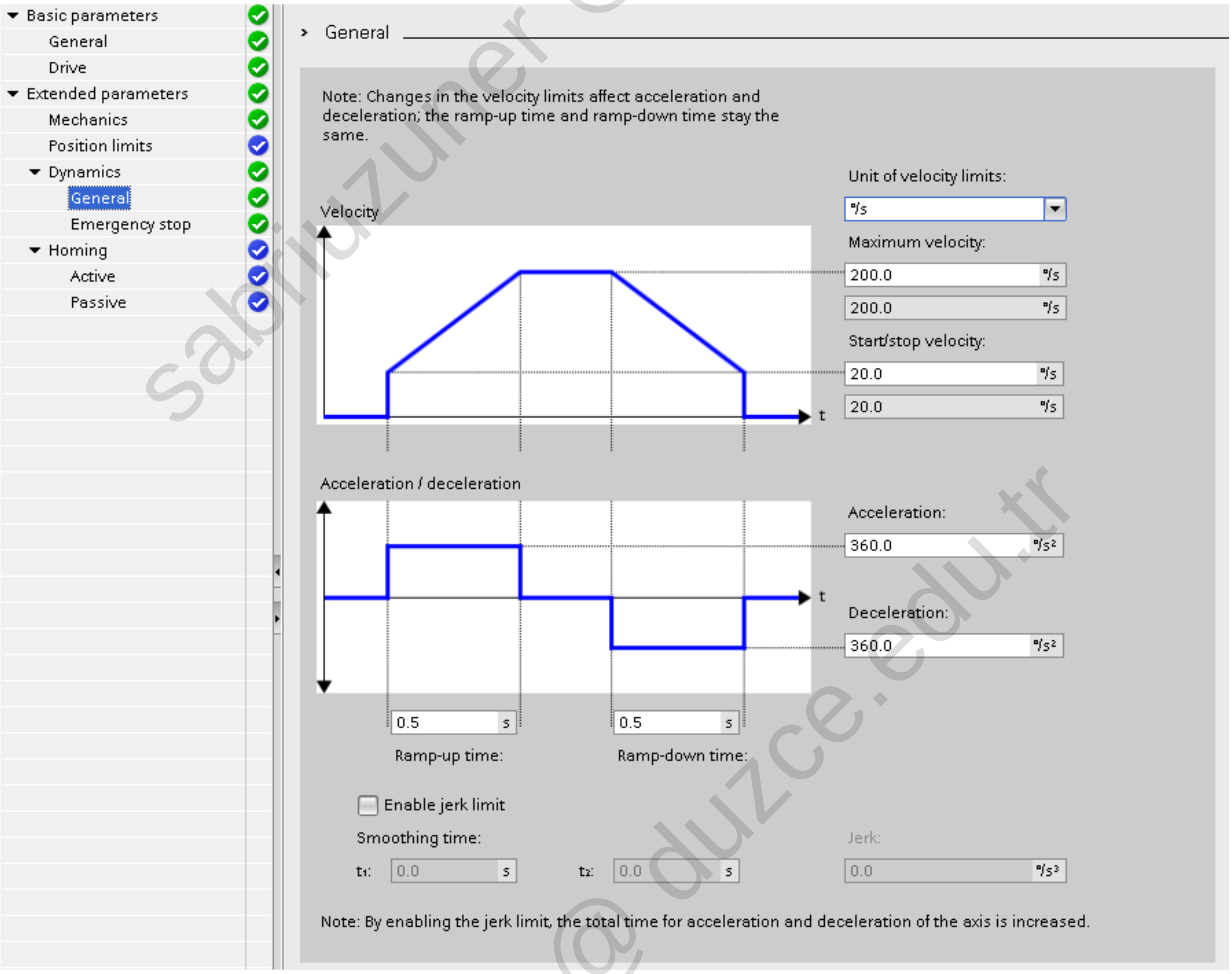
Mechanics

Pulses per motor revolution: 1600

Load movement per motor revolution: 360.0 °

Permitted direction of rotation: Both directions

□ Invert direction signal



- Basic parameters ✓
  - General ✓
  - Drive ✓
- Extended parameters ✓
  - Mechanics ✓
  - Position limits ✓
  - Dynamics ✓
    - General ✓
    - Emergency stop ✓
  - Homing ✓
    - Active ✓
    - Passive ✓

General

Note: Changes in the velocity limits affect acceleration and deceleration; the ramp-up time and ramp-down time stay the same.

Unit of velocity limits: °/s

Maximum velocity: 200.0 °/s; 200.0 °/s

Start/stop velocity: 20.0 °/s; 20.0 °/s

Acceleration: 360.0 °/s²

Deceleration: 360.0 °/s²

Ramp-up time: 0.5 s

Ramp-down time: 0.5 s

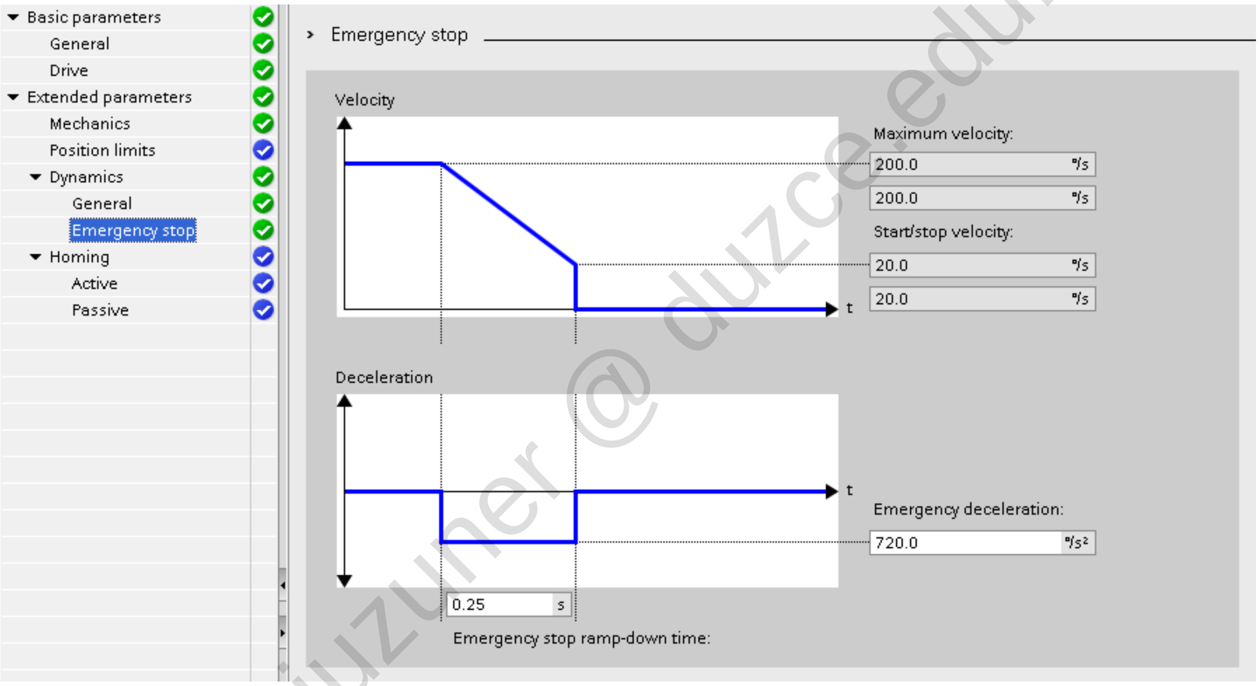
□ Enable jerk limit

Smoothing time:

t₁: 0.0 s   t₂: 0.0 s

Jerk: 0.0 °/s³

Note: By enabling the jerk limit, the total time for acceleration and deceleration of the axis is increased.



Emergency stop

Maximum velocity: 200.0 °/s

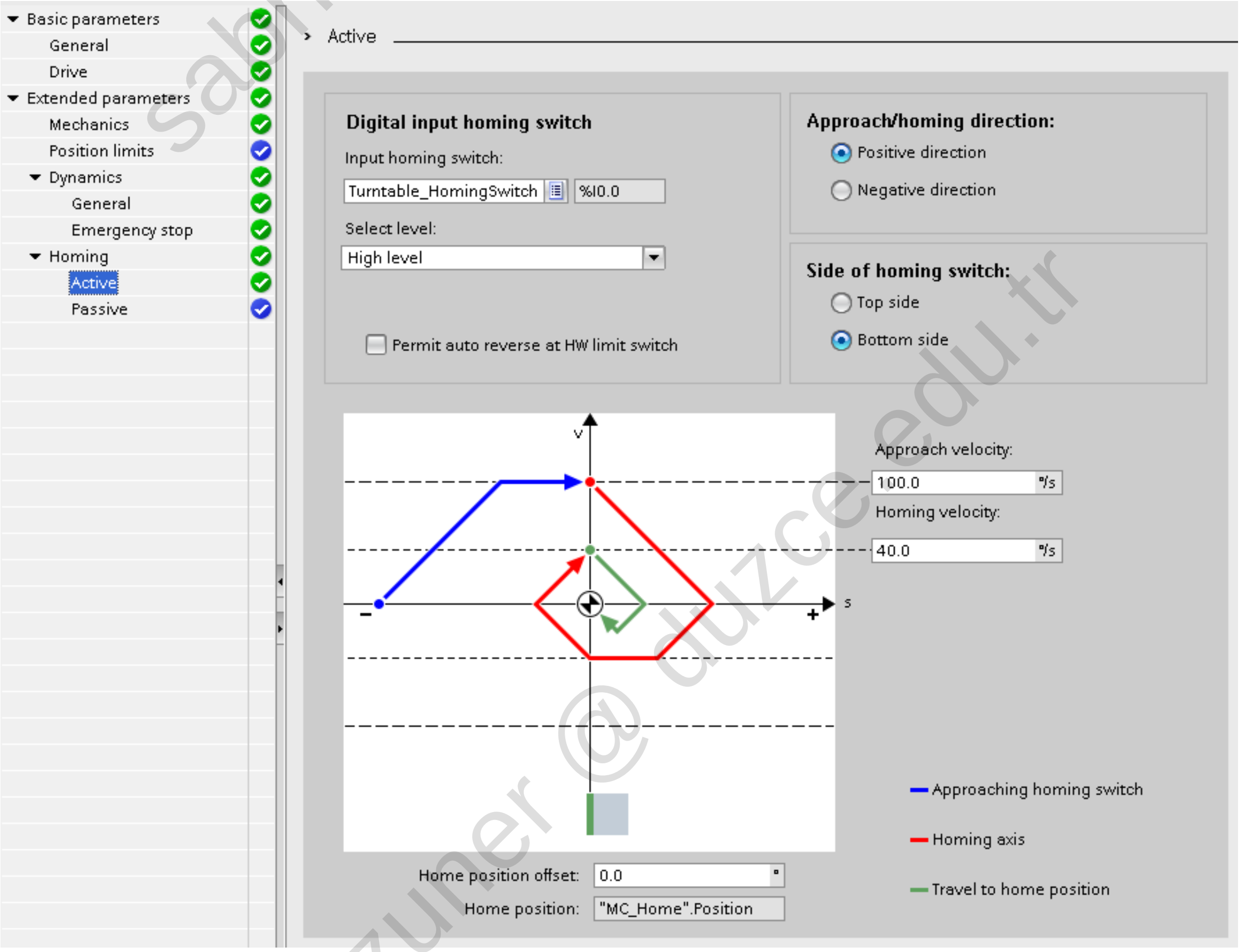
200.0 °/s

Start/stop velocity: 20.0 °/s

20.0 °/s

Emergency deceleration: 720.0 °/s²

Emergency stop ramp-down time: 0.25 s



Active

Digital input homing switch

Input homing switch: Turntable_HomingSwitch %I0.0

Select level: High level

□ Permit auto reverse at HW limit switch

Approach/homing direction:

● Positive direction

○ Negative direction

Side of homing switch:

○ Top side

● Bottom side

Approach velocity: 100.0 °/s

Homing velocity: 40.0 °/s

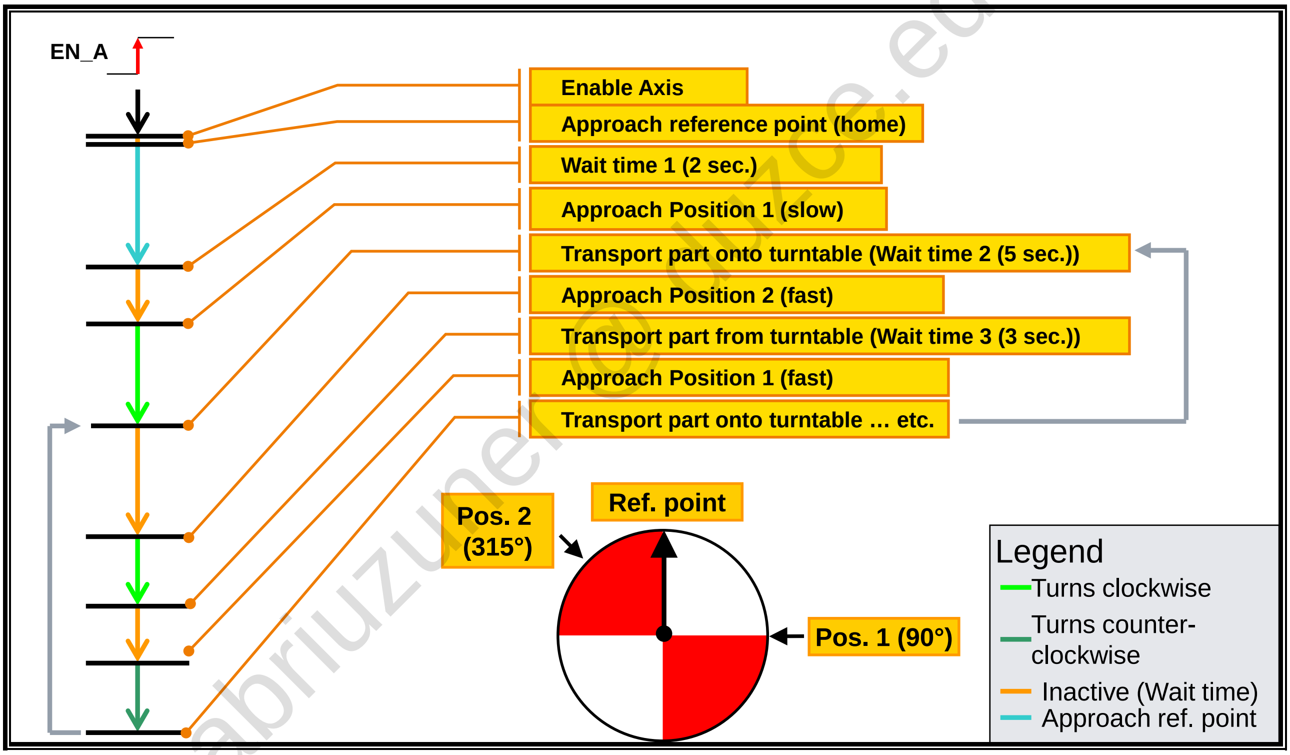
Home position offset: 0.0 °

Home position: "MC_Home".Position

3. The settings "Enable and feedback of the drive", "Position limits" as well as "Homing Passive" remain unchanged.

4. Save your project.

8.8.3. Exercise 6: Commissioning "FB_Turntable" (FB40)

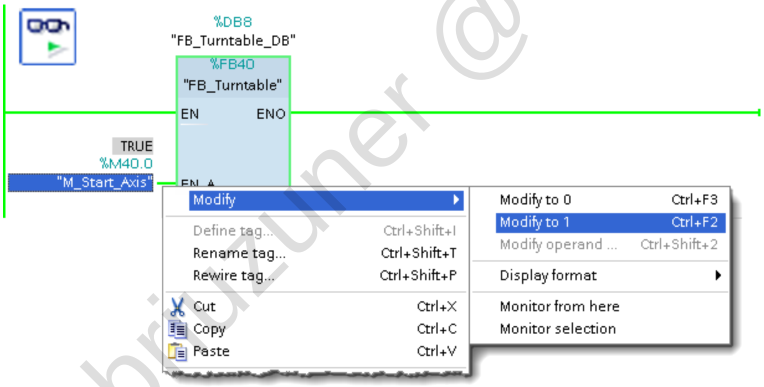

EN_A
Enable Axis
Approach reference point (home)
Wait time 1 (2 sec.)
Approach Position 1 (slow)
Transport part onto turntable (Wait time 2 (5 sec.))
Approach Position 2 (fast)
Transport part from turntable (Wait time 3 (3 sec.))
Approach Position 1 (fast)
Transport part onto turntable ... etc.
Pos. 2 (315°)
Ref. point
Pos. 1 (90°)
Legend
Turns clockwise
Turns counter-clockwise
Inactive (Wait time)
Approach ref. point

Task

The control of the technology object "Turntable" is implemented by the function block "FB_Turntable" (FB40). Insert the block from the Project library into your user program and call it in OB1.

What to Do

1. Using drag & drop, copy the block "FB_Turntable" from the Project library into the program folder of PLC_2.
2. Call "FB_Turntable" in OB1 with instance DB number ≠ 40.
3. Connect the input "EN_A" with the bit memory "M_Start_Axis" (M40.0).
4. Save your project and transfer both the hardware and the software to PLC_2.
5. Monitor the call of "FB_Turntable" in OB1 and control the bit memory "M_Start_Axis" as shown in the following.

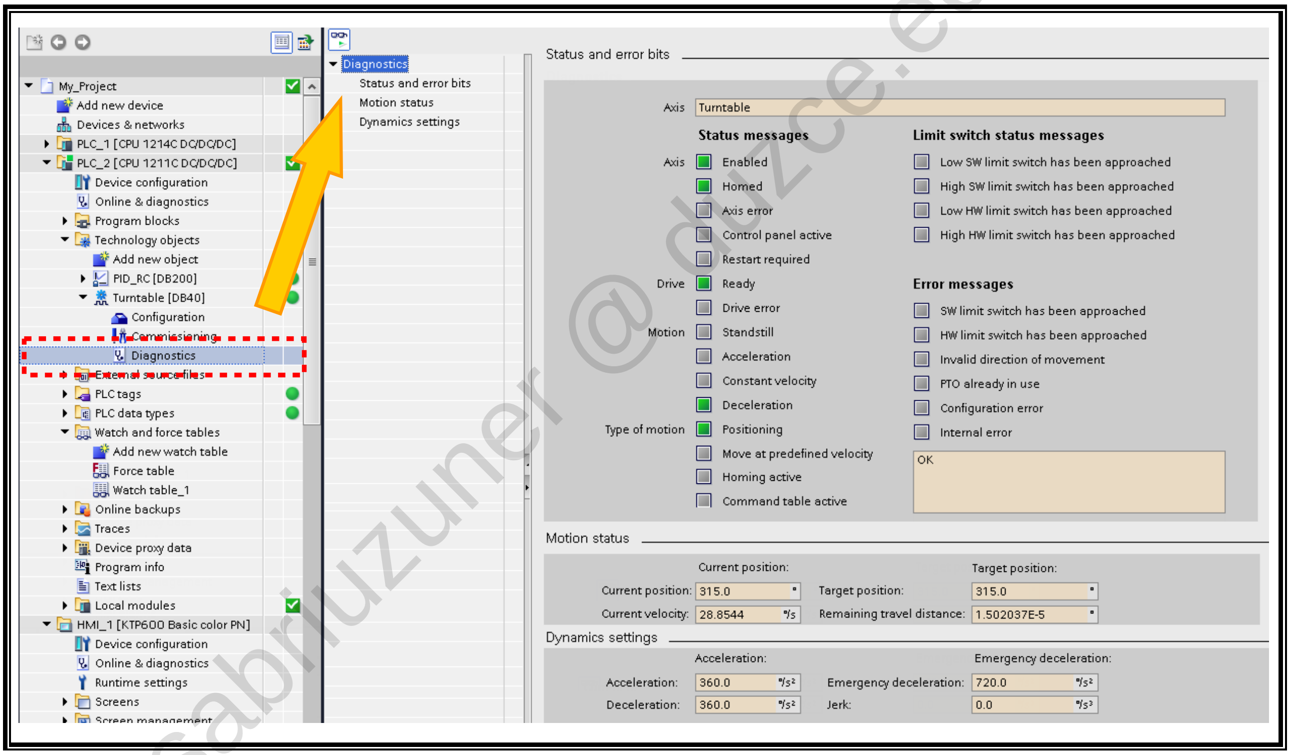


Result

The axis first of all carries out the active homing (referencing) and then begins with the motion sequence described in the task.

If the bit memory "M_Start_Axis" is reset, the axis ends the current execution and then stops at Position 1.

8.8.4. Exercise 7: Starting the Axis and Monitoring the Statuses with the Diagnostic Panel



Status and error bits

Axis: Turntable

Status messages

- Axis: 🟩 Enabled
- 🟩 Homed
- □ Axis error
- □ Control panel active
- □ Restart required
- Drive: 🟩 Ready
- □ Drive error
- Motion: □ Standstill
- □ Acceleration
- □ Constant velocity
- 🟩 Deceleration
- Type of motion: 🟩 Positioning
- □ Move at predefined velocity
- □ Homing active
- □ Command table active

Limit switch status messages

- □ Low SW limit switch has been approached
- □ High SW limit switch has been approached
- □ Low HW limit switch has been approached
- □ High HW limit switch has been approached

Error messages

- □ SW limit switch has been approached
- □ HW limit switch has been approached
- □ Invalid direction of movement
- □ PTO already in use
- □ Configuration error
- □ Internal error

OK

Motion status

Current position:		Target position:	
Current position:	315.0 °	Target position:	315.0 °
Current velocity:	28.8544 °/s	Remaining travel distance:	1.502037E-5 °

Dynamics settings

Acceleration:		Emergency deceleration:	
Acceleration:	360.0 °/s²	Emergency deceleration:	720.0 °/s²
Deceleration:	360.0 °/s²	Jerk:	0.0 °/s³

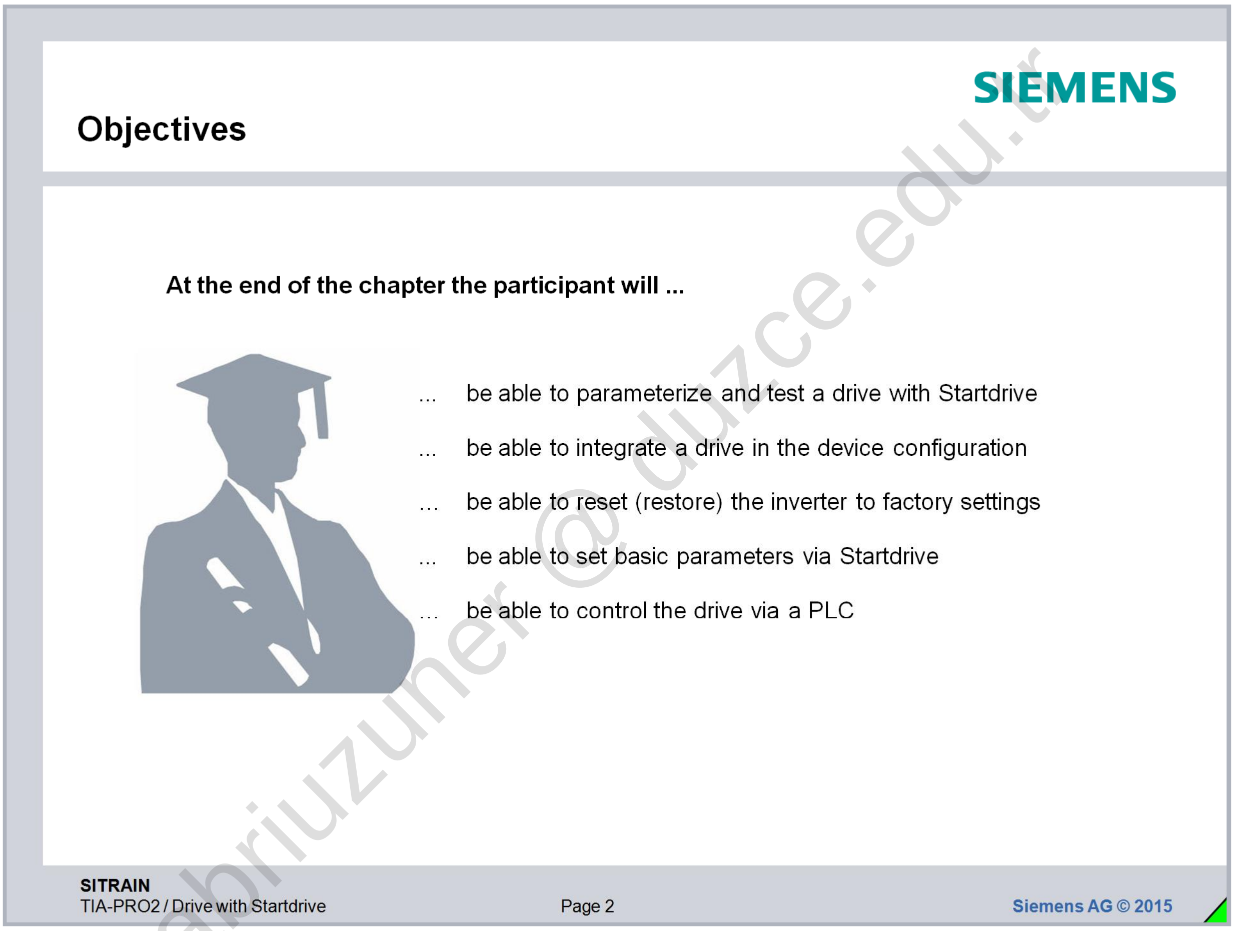
Task

Monitor the current status of the axis and track the motion sequences via the Diagnostic Panel.

SIEMENS

SITRAIN

Integrating and Commissioning a Drive with Startdrive



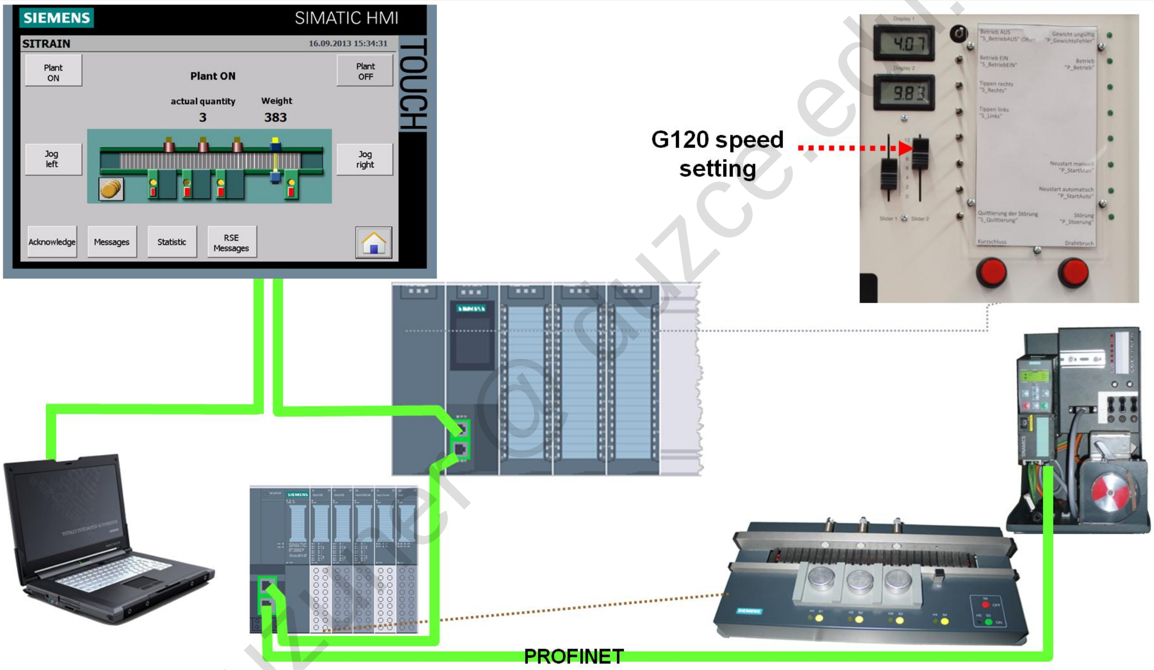
Objectives

At the end of the chapter the participant will ...

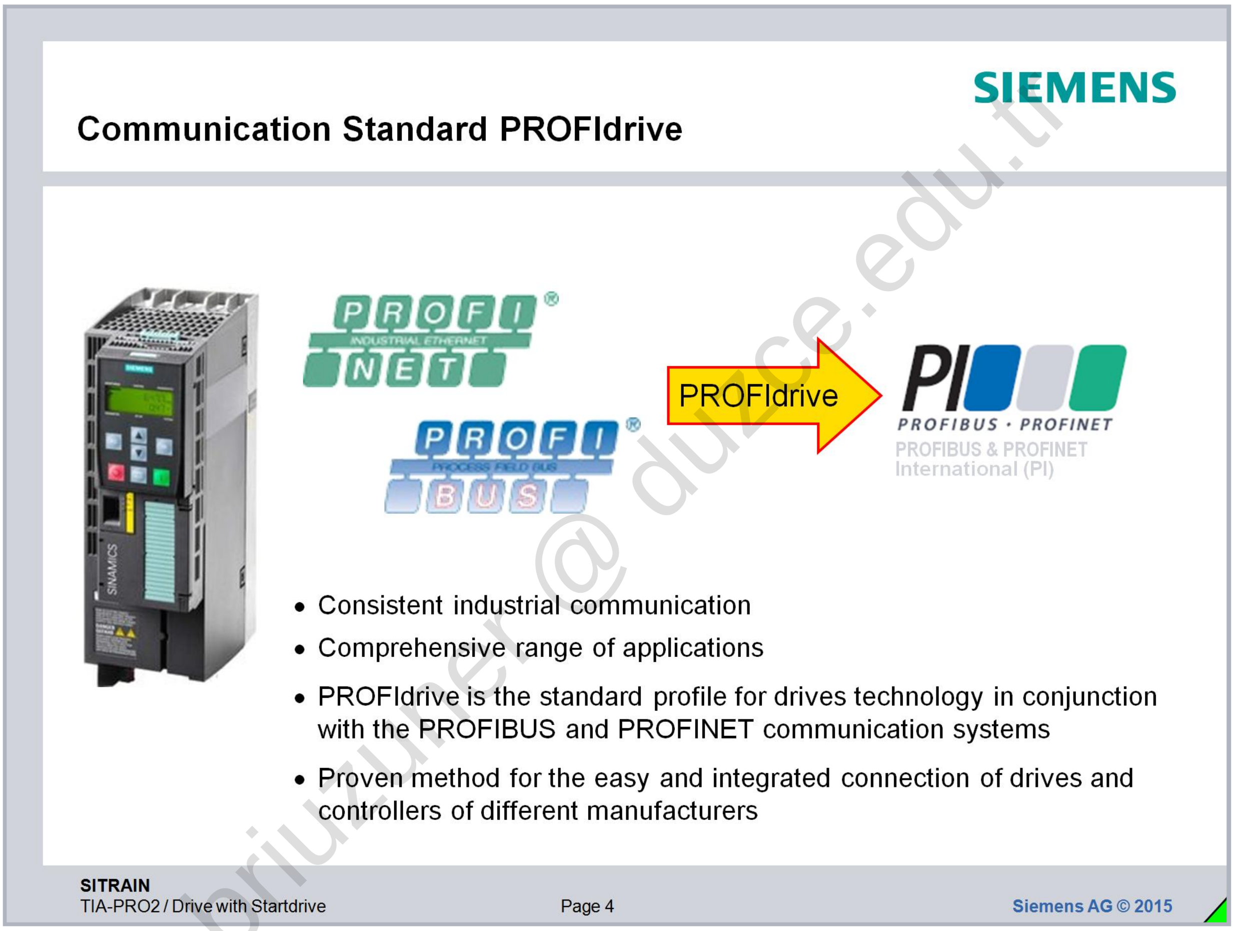
- ... be able to parameterize and test a drive with Startdrive
- ... be able to integrate a drive in the device configuration
- ... be able to reset (restore) the inverter to factory settings
- ... be able to set basic parameters via Startdrive
- ... be able to control the drive via a PLC

SITRAIN
TIA-PRO2 / Drive with Startdrive Page 2 Siemens AG © 2015

Task Description:
G120 as an Additional Conveyor Drive



SITRAIN
TIA-PRO2 / Drive with Startdrive



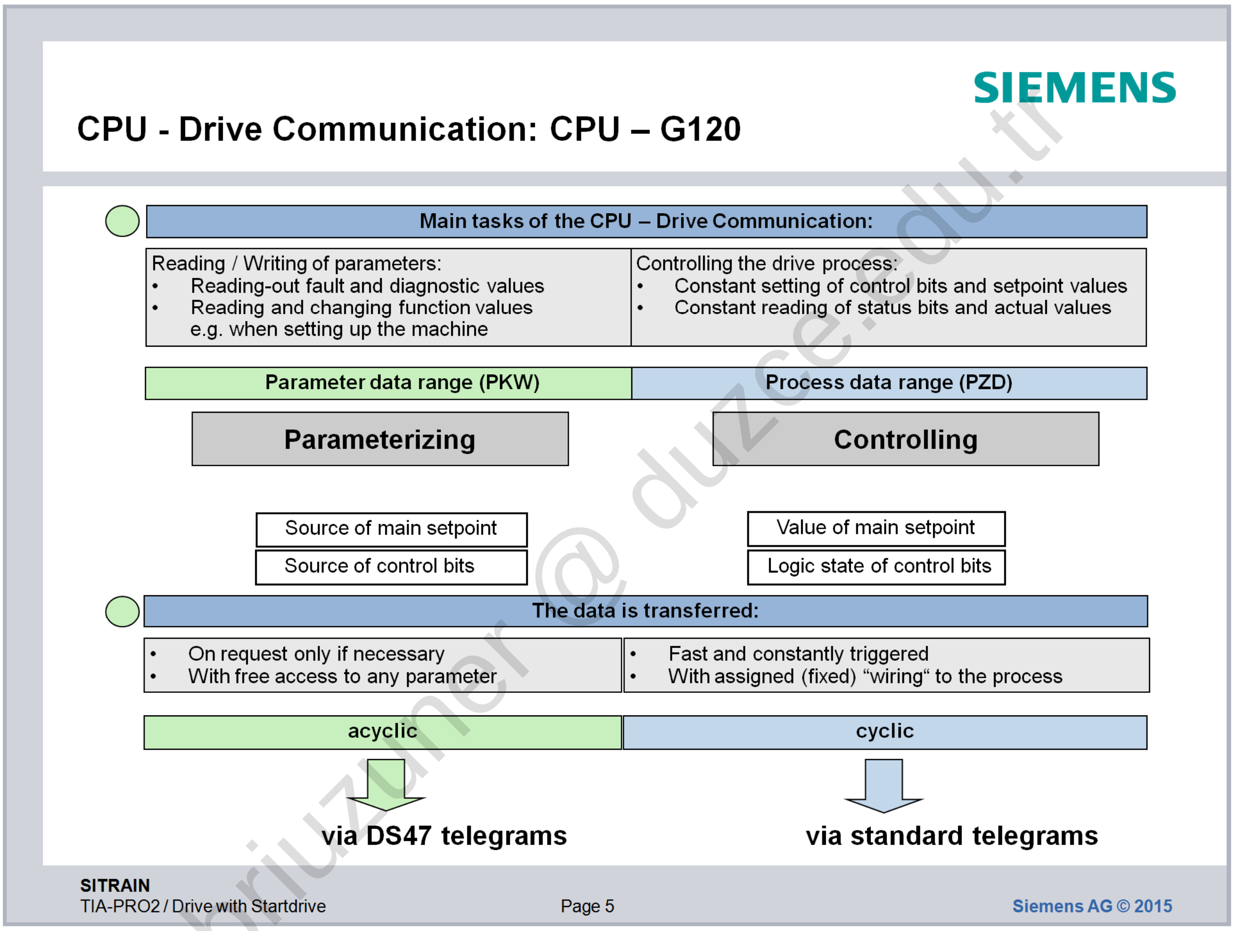
SIEMENS

Communication Standard PROFIdrive

PROFI NET INDUSTRIAL ETHERNET ®
PROFI BUS PROCESS FIELD BUS ®
PROFIdrive
PI
PROFIBUS · PROFINET
PROFIBUS & PROFINET International (PI)

- Consistent industrial communication
- Comprehensive range of applications
- PROFIdrive is the standard profile for drives technology in conjunction with the PROFIBUS and PROFINET communication systems
- Proven method for the easy and integrated connection of drives and controllers of different manufacturers

SITRAIN
TIA-PRO2 / Drive with Startdrive Page 4 Siemens AG © 2015



SIEMENS

CPU - Drive Communication: CPU – G120

Main tasks of the CPU – Drive Communication:

Reading / Writing of parameters:	Controlling the drive process:
• Reading-out fault and diagnostic values • Reading and changing function values e.g. when setting up the machine	• Constant setting of control bits and setpoint values • Constant reading of status bits and actual values
Parameter data range (PKW)	**Process data range (PZD)**

Parameterizing | **Controlling**

Source of main setpoint

Source of control bits

Value of main setpoint

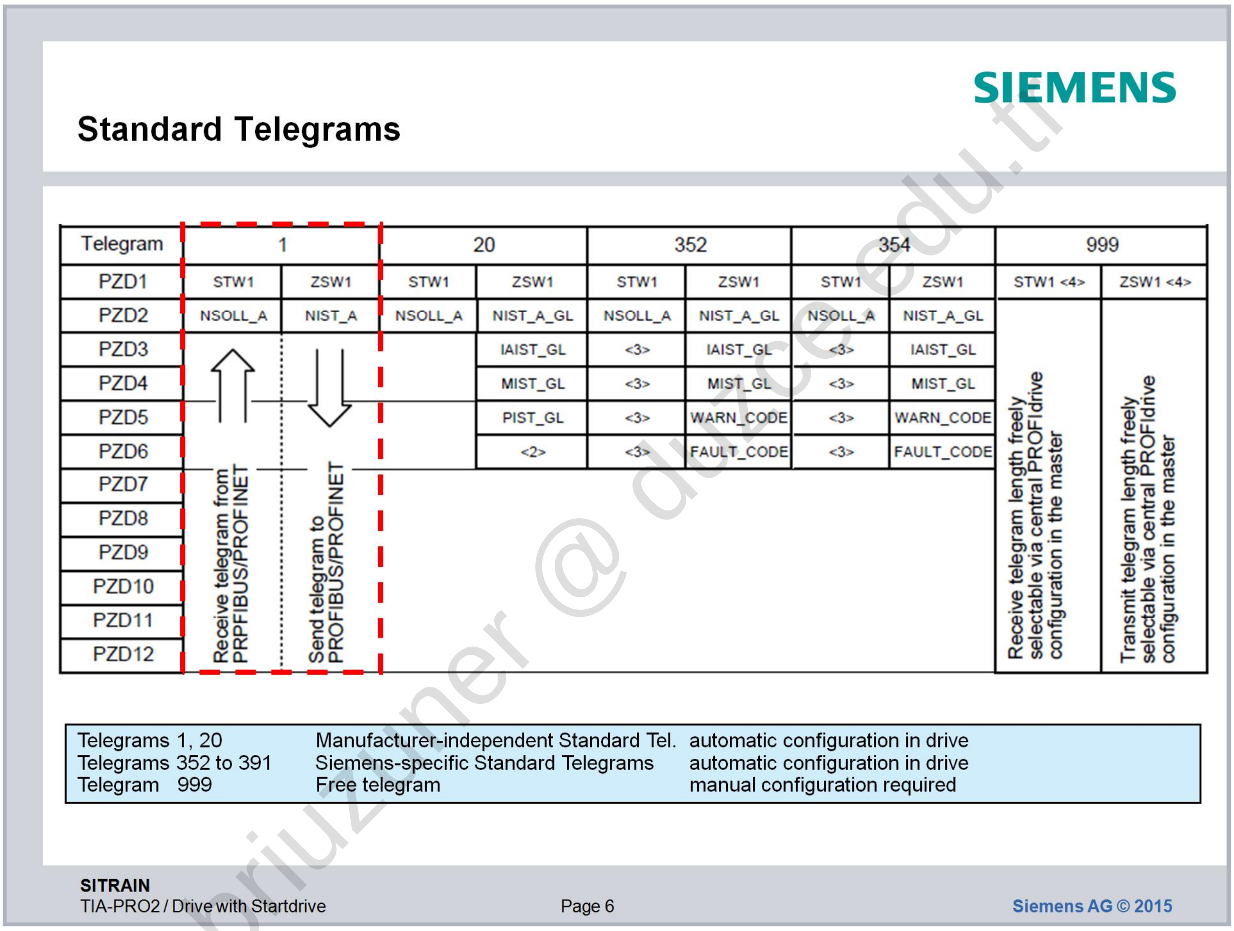
Logic state of control bits

The data is transferred:

• On request only if necessary • With free access to any parameter	• Fast and constantly triggered • With assigned (fixed) “wiring“ to the process
acyclic	**cyclic**

via DS47 telegrams

via standard telegrams



SIEMENS

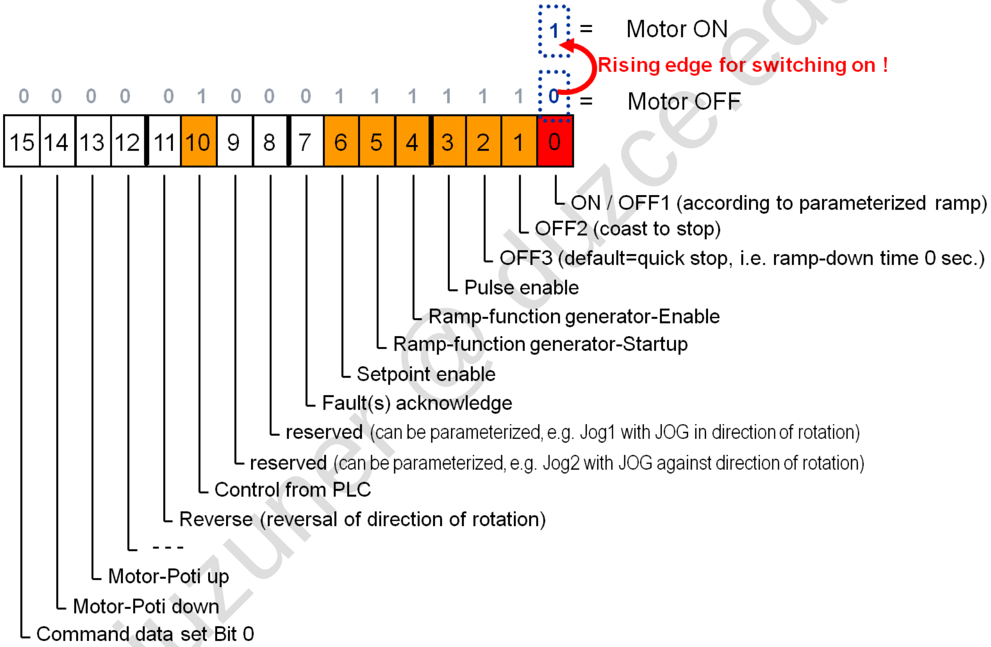
Standard Telegrams

Telegram	1		20		352		354		999	
PZD1	STW1	ZSW1	STW1	ZSW1	STW1	ZSW1	STW1	ZSW1	STW1 <4>	ZSW1 <4>
PZD2	NSOLL_A	NIST_A	NSOLL_A	NIST_A_GL	NSOLL_A	NIST_A_GL	NSOLL_A	NIST_A_GL	Receive telegram length freely selectable via central PROFIdrive configuration in the master	Transmit telegram length freely selectable via central PROFIdrive configuration in the master
PZD3	⇧ Receive telegram from PRPFIBUS/PROFINET	⇩ Send telegram to PROFIBUS/PROFINET		IAIST_GL	<3>	IAIST_GL	<3>	IAIST_GL		
PZD4				MIST_GL	<3>	MIST_GL	<3>	MIST_GL		
PZD5				PIST_GL	<3>	WARN_CODE	<3>	WARN_CODE		
PZD6				<2>	<3>	FAULT_CODE	<3>	FAULT_CODE		
PZD7										
PZD8										
PZD9										
PZD10										
PZD11										
PZD12										

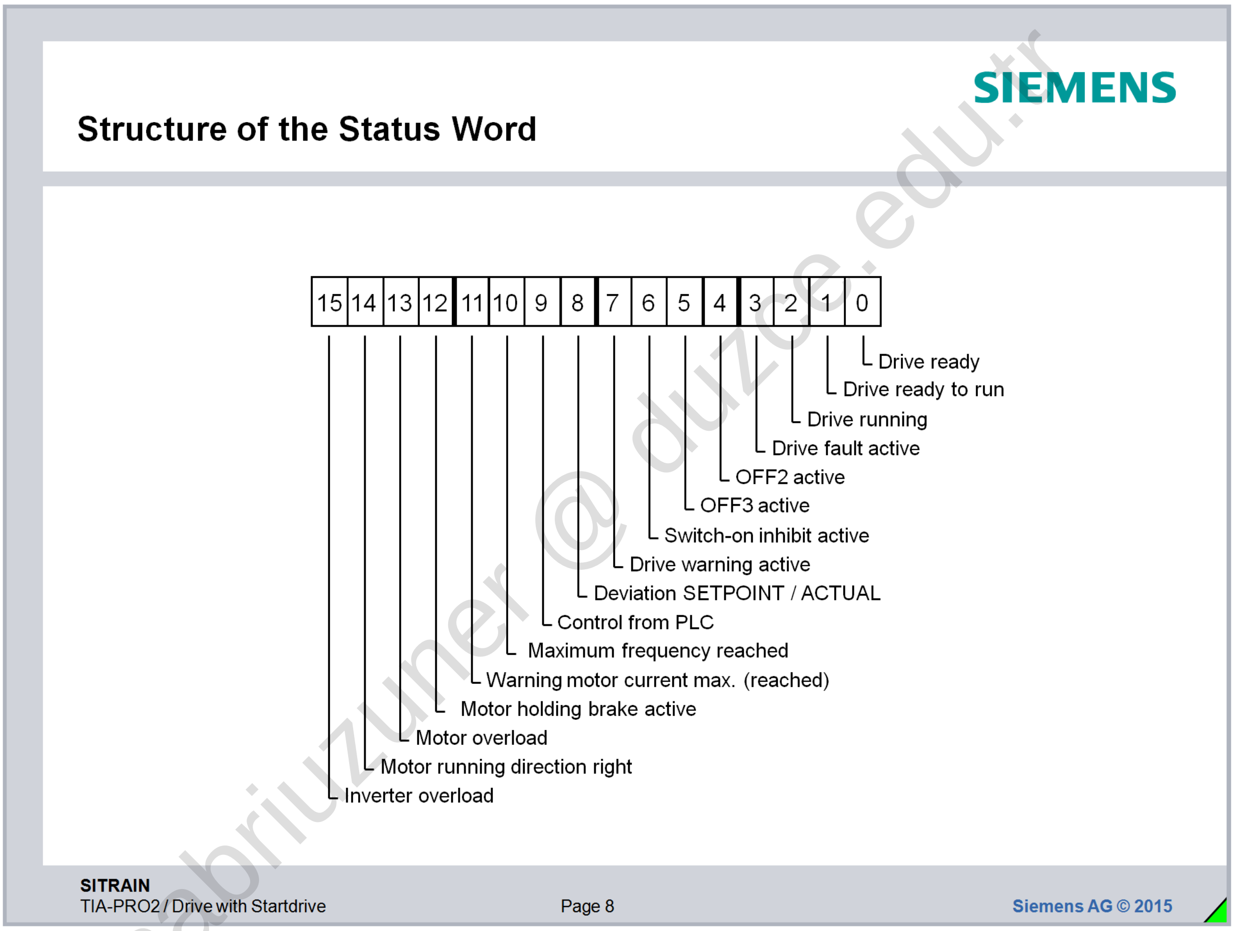
Telegrams 1, 20	Manufacturer-independent Standard Tel.	automatic configuration in drive
Telegrams 352 to 391	Siemens-specific Standard Telegrams	automatic configuration in drive
Telegram 999	Free telegram	manual configuration required

SIEMENS

Structure of the Control Word



- ON / OFF1 (according to parameterized ramp)
- OFF2 (coast to stop)
- OFF3 (default=quick stop, i.e. ramp-down time 0 sec.)
- Pulse enable
- Ramp-function generator-Enable
- Ramp-function generator-Startup
- Setpoint enable
- Fault(s) acknowledge
- reserved (can be parameterized, e.g. Jog1 with JOG in direction of rotation)
- reserved (can be parameterized, e.g. Jog2 with JOG against direction of rotation)
- Control from PLC
- Reverse (reversal of direction of rotation)
- - - -
- Motor-Poti up
- Motor-Poti down
- Command data set Bit 0



SIEMENS

Structure of the Status Word

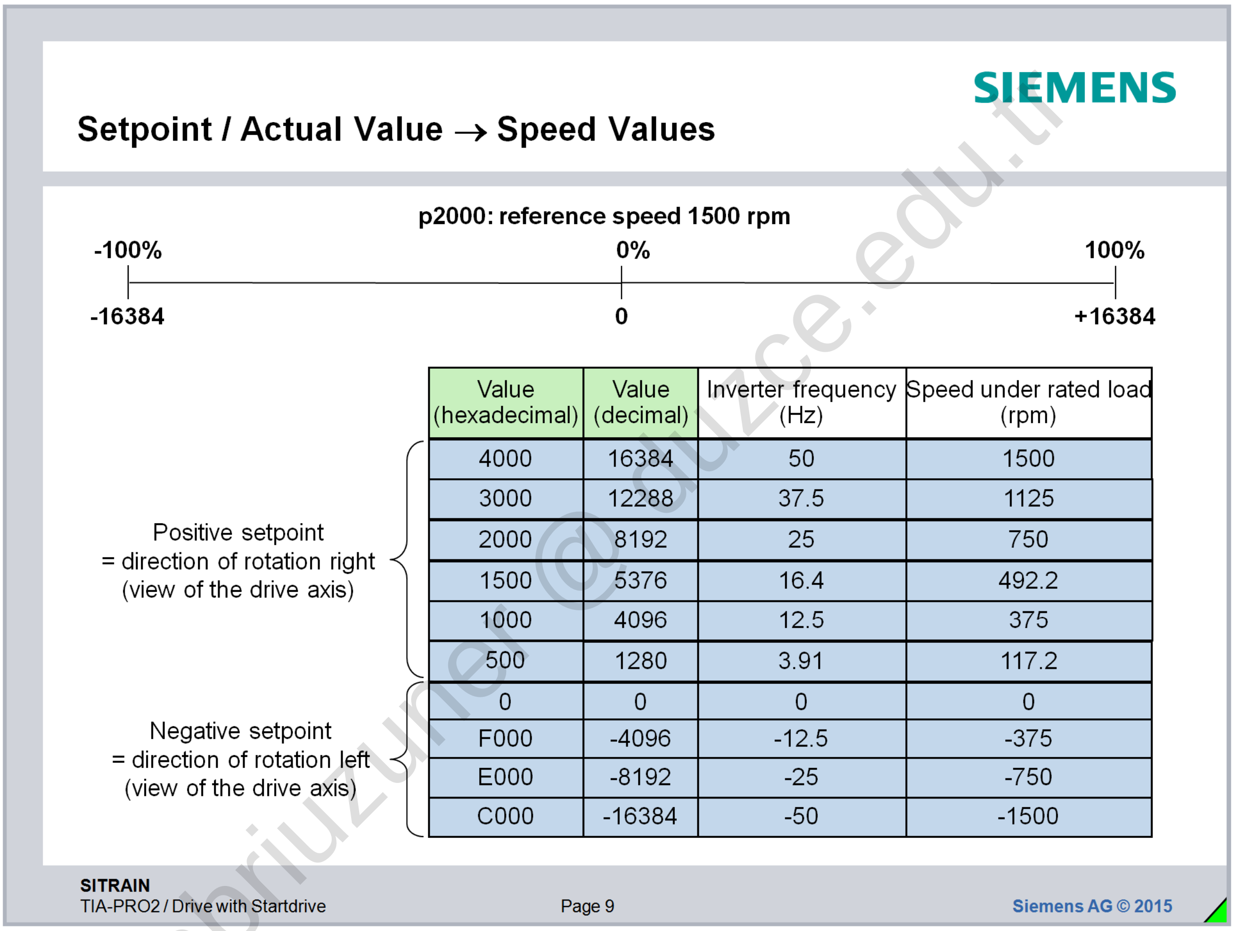
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

- Drive ready
- Drive ready to run
- Drive running
- Drive fault active
- OFF2 active
- OFF3 active
- Switch-on inhibit active
- Drive warning active
- Deviation SETPOINT / ACTUAL
- Control from PLC
- Maximum frequency reached
- Warning motor current max. (reached)
- Motor holding brake active
- Motor overload
- Motor running direction right
- Inverter overload

SITRAIN
TIA-PRO2 / Drive with Startdrive

Page 8

Siemens AG © 2015



SIEMENS

Setpoint / Actual Value → Speed Values

p2000: reference speed 1500 rpm

-100% — **0%** — **100%**

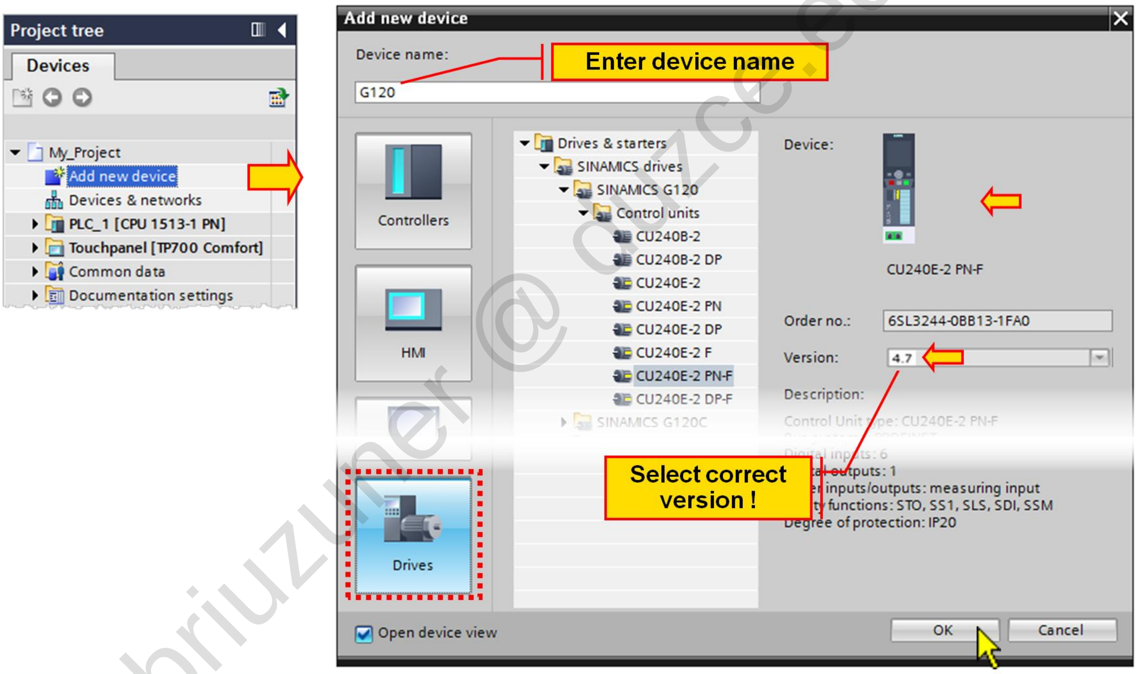
-16384 — **0** — **+16384**

Positive setpoint
= direction of rotation right
(view of the drive axis)

Negative setpoint
= direction of rotation left
(view of the drive axis)

Value (hexadecimal)	Value (decimal)	Inverter frequency (Hz)	Speed under rated load (rpm)
4000	16384	50	1500
3000	12288	37.5	1125
2000	8192	25	750
1500	5376	16.4	492.2
1000	4096	12.5	375
500	1280	3.91	117.2
0	0	0	0
F000	-4096	-12.5	-375
E000	-8192	-25	-750
C000	-16384	-50	-1500

Inserting a Drive into the Project



Project tree

Devices

- My_Project
  - Add new device
  - Devices & networks
  - PLC_1 [CPU 1513-1 PN]
  - Touchpanel [TP700 Comfort]
  - Common data
  - Documentation settings

Add new device

Device name: G120

Enter device name

- Drives & starters
  - SINAMICS drives
    - SINAMICS G120
      - Control units
        - CU240B-2
        - CU240B-2 DP
        - CU240E-2
        - CU240E-2 PN
        - CU240E-2 DP
        - CU240E-2 F
        - CU240E-2 PN-F
        - CU240E-2 DP-F
    - SINAMICS G120C

Device: CU240E-2 PN-F

Order no.: 6SL3244-0BB13-1FA0

Version: 4.7

Description:

Control Unit type: CU240E-2 PN-F

Digital inputs: 6

tal outputs: 1

er inputs/outputs: measuring input

ty functions: STO, SS1, SLS, SDI, SSM

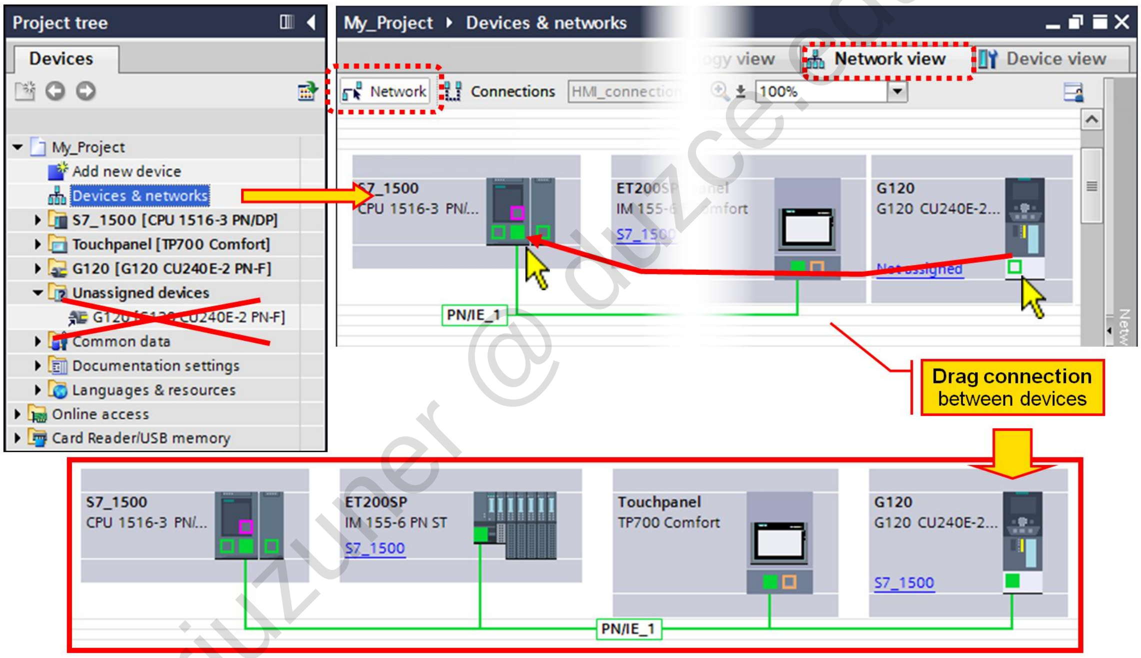
Degree of protection: IP20

Select correct version !

☑ Open device view

SIEMENS

Networking a Drive with a CPU



Drag connection between devices

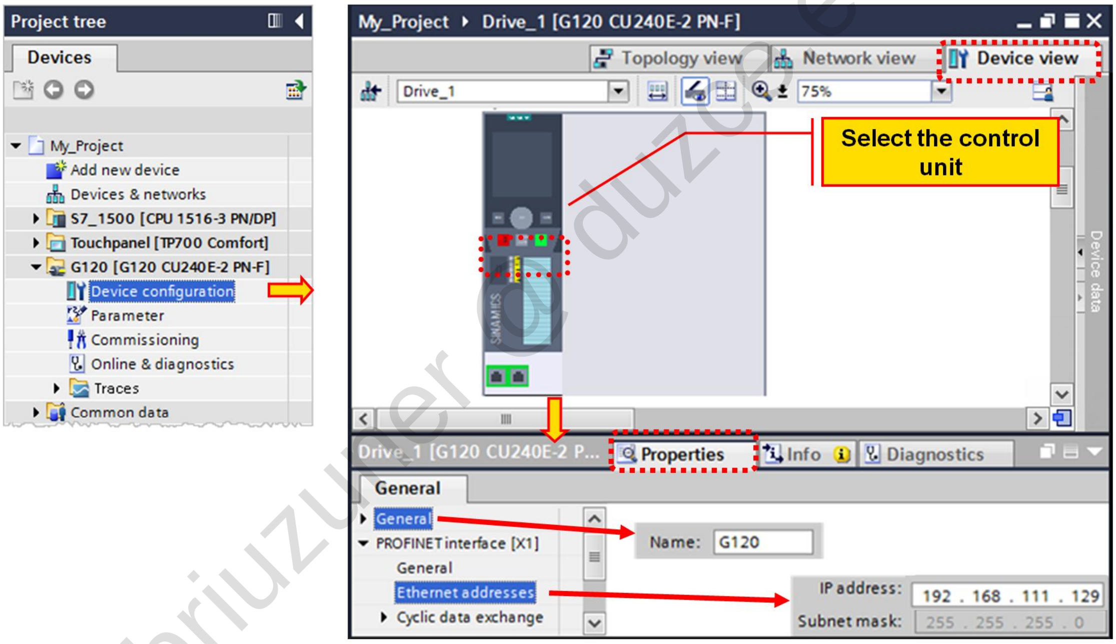
SITRAIN
TIA-PRO2 / Drive with Startdrive

Page 11

Siemens AG © 2015

Parameterizing the Module Address and Module Name

SIEMENS



Select the control unit

Name: G120

IP address: 192 . 168 . 111 . 129

Subnet mask: 255 . 255 . 255 . 0

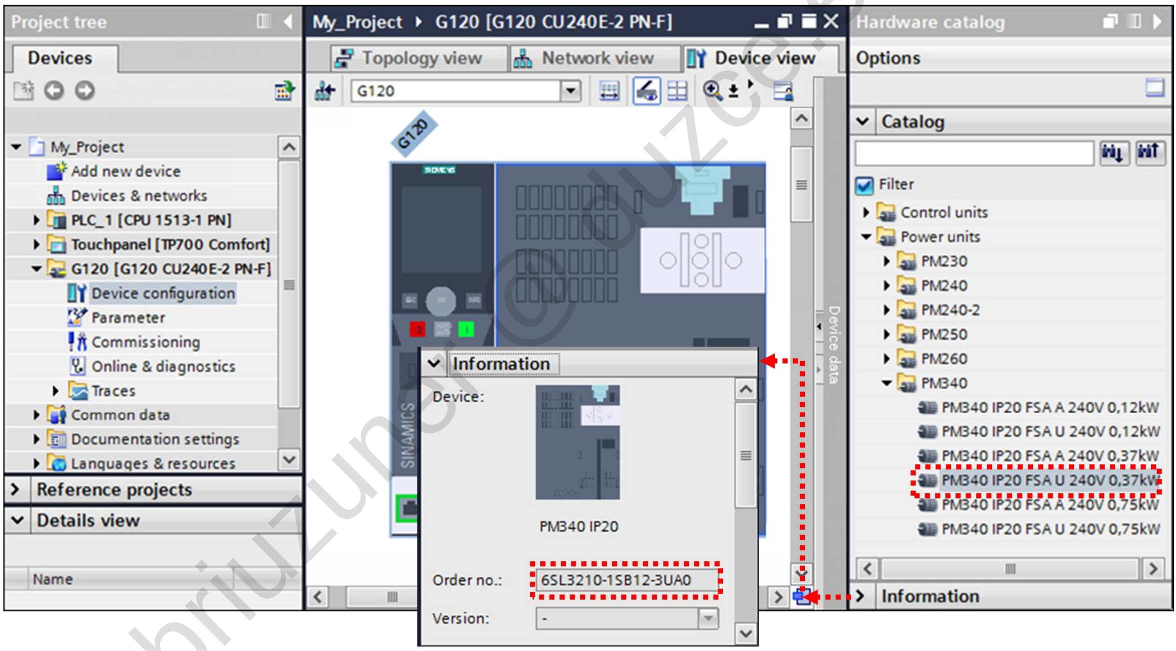
SITRAIN
TIA-PRO2 / Drive with Startdrive

Page 12

Siemens AG © 2015

SIEMENS

Configuring a Power Unit



Information

Device: PM340 IP20

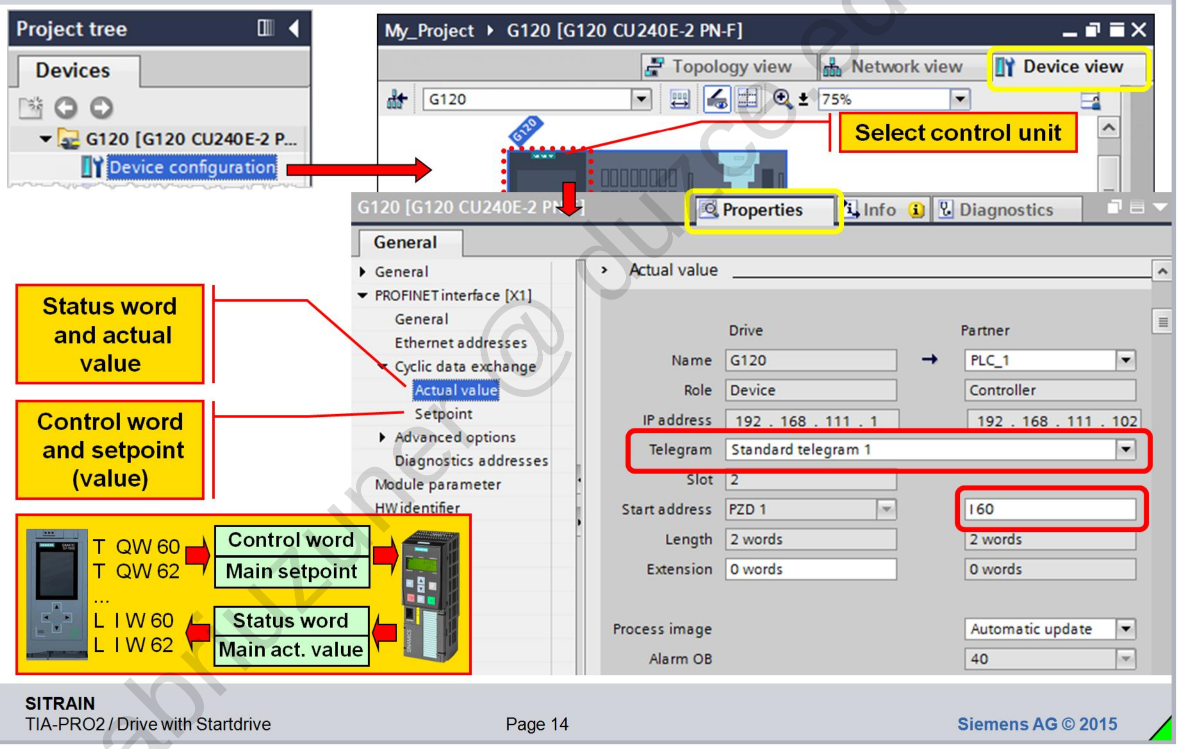
Order no.: 6SL3210-1SB12-3UA0

Version: -

SITRAIN
TIA-PRO2 / Drive with Startdrive
Page 13
Siemens AG © 2015

Parameterizing the Process Data Area (PZD)

SIEMENS



Select control unit

Status word and actual value

Control word and setpoint (value)

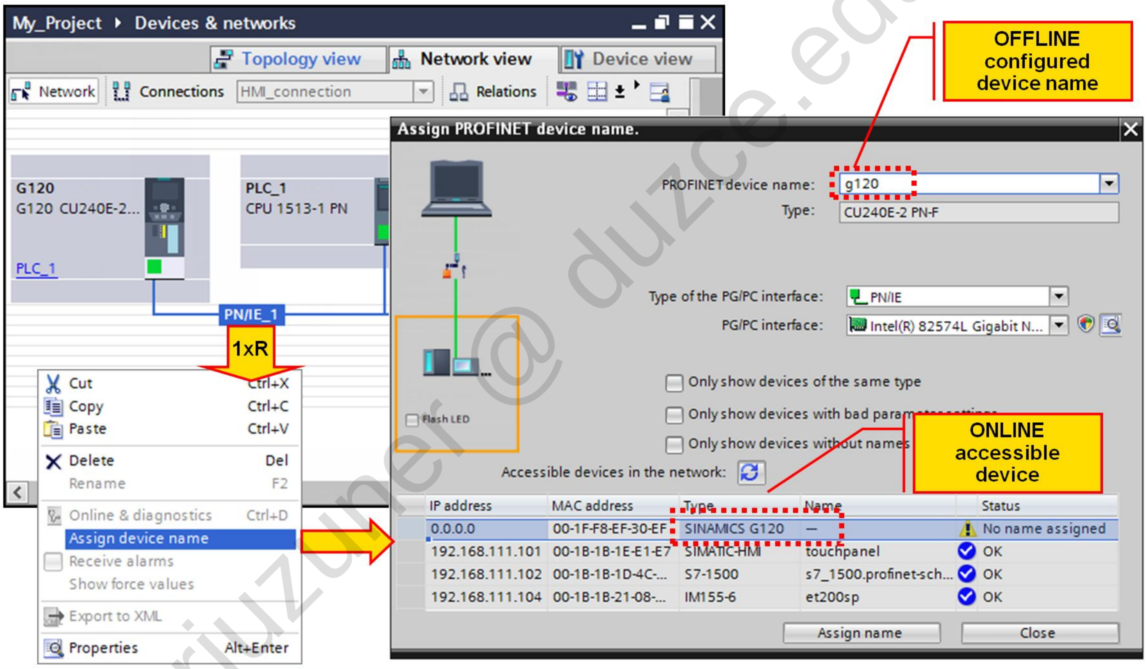
	Drive		Partner
Name	G120	→	PLC_1
Role	Device		Controller
IP address	192 . 168 . 111 . 1		192 . 168 . 111 . 102
Telegram	Standard telegram 1		
Slot	2		
Start address	PZD 1		I 60
Length	2 words		2 words
Extension	0 words		0 words
Process image			Automatic update
Alarm OB			40

T QW 60 → Control word
T QW 62 → Main setpoint
...
L I W 60 ← Status word
L I W 62 ← Main act. value

SITRAIN
TIA-PRO2 / Drive with Startdrive
Page 14
Siemens AG © 2015

Assigning the Device Name ONLINE (Module Initialization)

SIEMENS



Assign PROFINET device name.

PROFINET device name: g120

Type: CU240E-2 PN-F

Type of the PG/PC interface: PN/IE

PG/PC interface: Intel(R) 82574L Gigabit N...

□ Only show devices of the same type

□ Only show devices with bad parameter settings

□ Only show devices without names

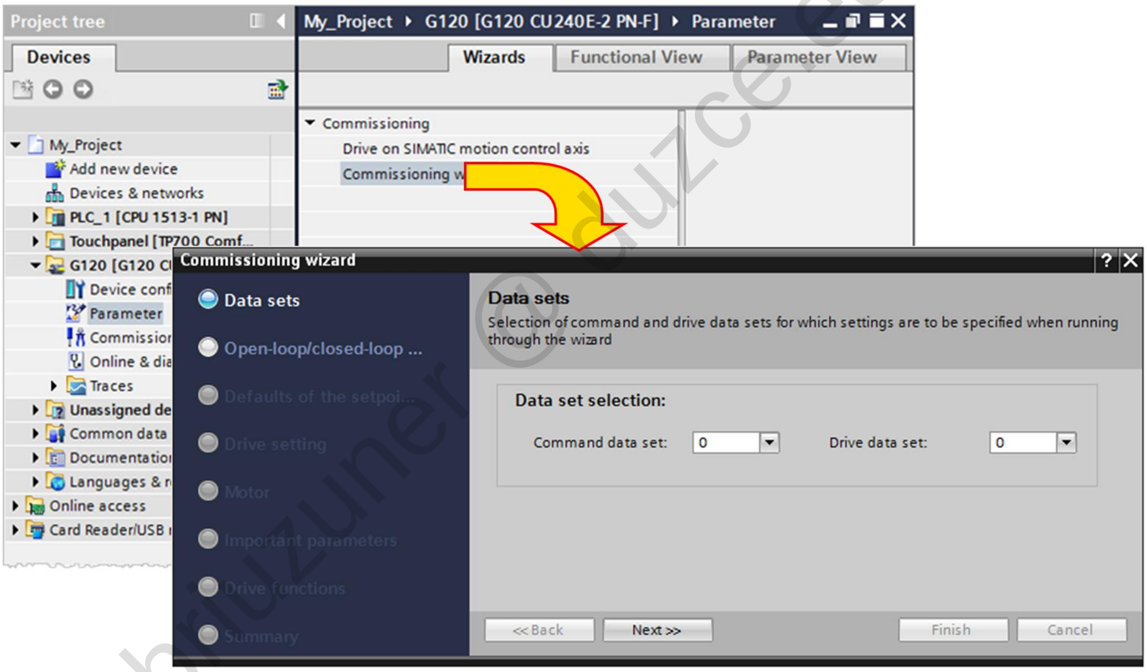
Accessible devices in the network:

IP address	MAC address	Type	Name	Status
0.0.0.0	00-1F-F8-EF-30-EF	SINAMICS G120	--	No name assigned
192.168.111.101	00-1B-1B-1E-E1-E7	SIMATIC-HMI	touchpanel	OK
192.168.111.102	00-1B-1B-1D-4C-...	S7-1500	s7_1500.profinet-sch...	OK
192.168.111.104	00-1B-1B-21-08-...	IM155-6	et200sp	OK

Assign name | Close

Parameterizing the Drive with the "Commissioning Wizard"

SIEMENS



Data sets

Selection of command and drive data sets for which settings are to be specified when running through the wizard

Data set selection:

Command data set: 0

Drive data set: 0

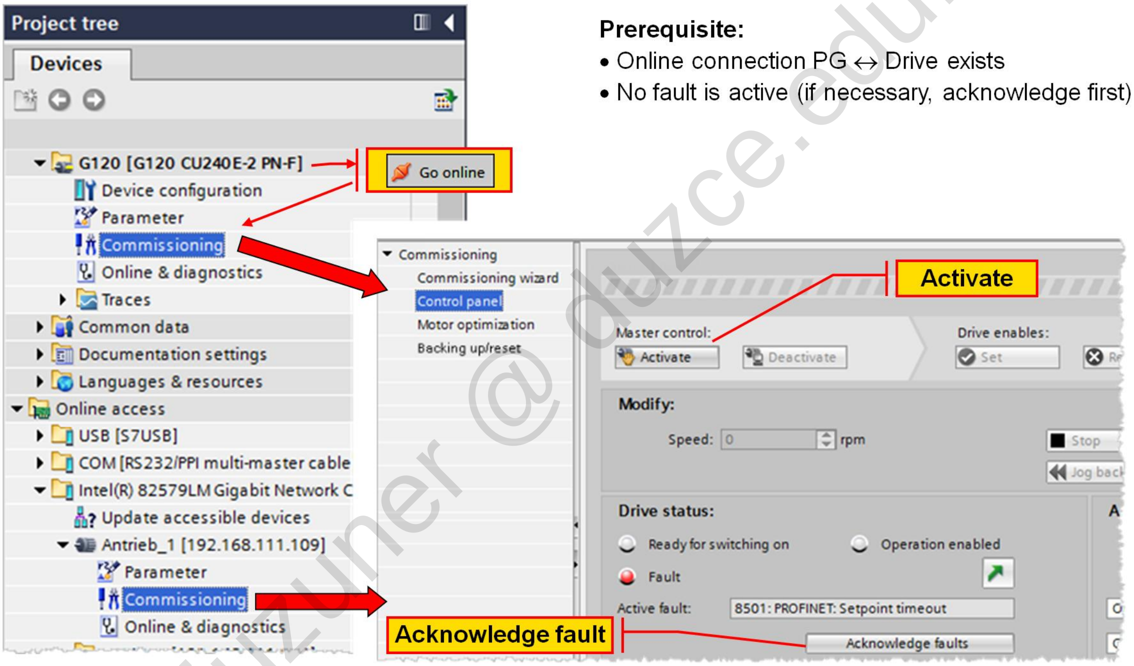
SITRAIN
TIA-PRO2 / Drive with Startdrive
Page 16
Siemens AG © 2015

Online Commissioning: Activating / Deactivating the Control Panel

SIEMENS

Prerequisite:

- Online connection PG ↔ Drive exists
- No fault is active (if necessary, acknowledge first)



Project tree

Devices

- G120 [G120 CU240E-2 PN-F]
  - Device configuration
  - Parameter
  - Commissioning
  - Online & diagnostics
  - Traces
- Common data
- Documentation settings
- Languages & resources
- Online access
  - USB [S7USB]
  - COM [RS232/PPI multi-master cable
  - Intel(R) 82579LM Gigabit Network C
    - Update accessible devices
    - Antrieb_1 [192.168.111.109]
      - Parameter
      - Commissioning
      - Online & diagnostics

Go online

- Commissioning
  - Commissioning wizard
  - Control panel
  - Motor optimization
  - Backing up/reset

Activate

Master control: Activate | Deactivate

Drive enables: Set

Modify:

Speed: 0 rpm

Stop

Jog back

Drive status:

○ Ready for switching on

○ Operation enabled

● Fault

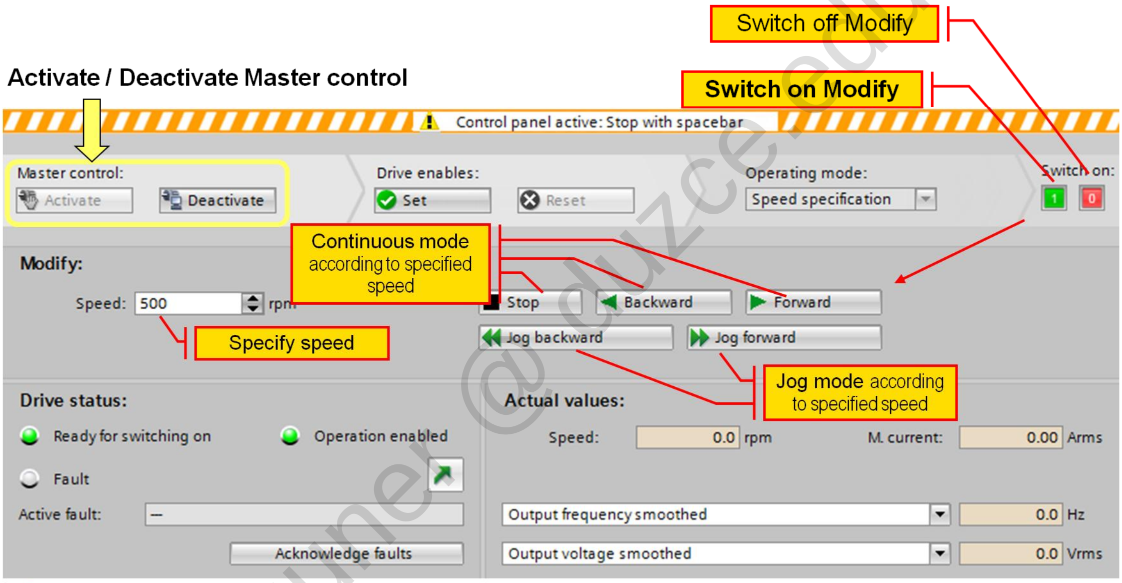
Active fault: 8501: PROFINET: Setpoint timeout

Acknowledge faults

Acknowledge fault

Operating the Control Panel for Commissioning

Activate / Deactivate Master control

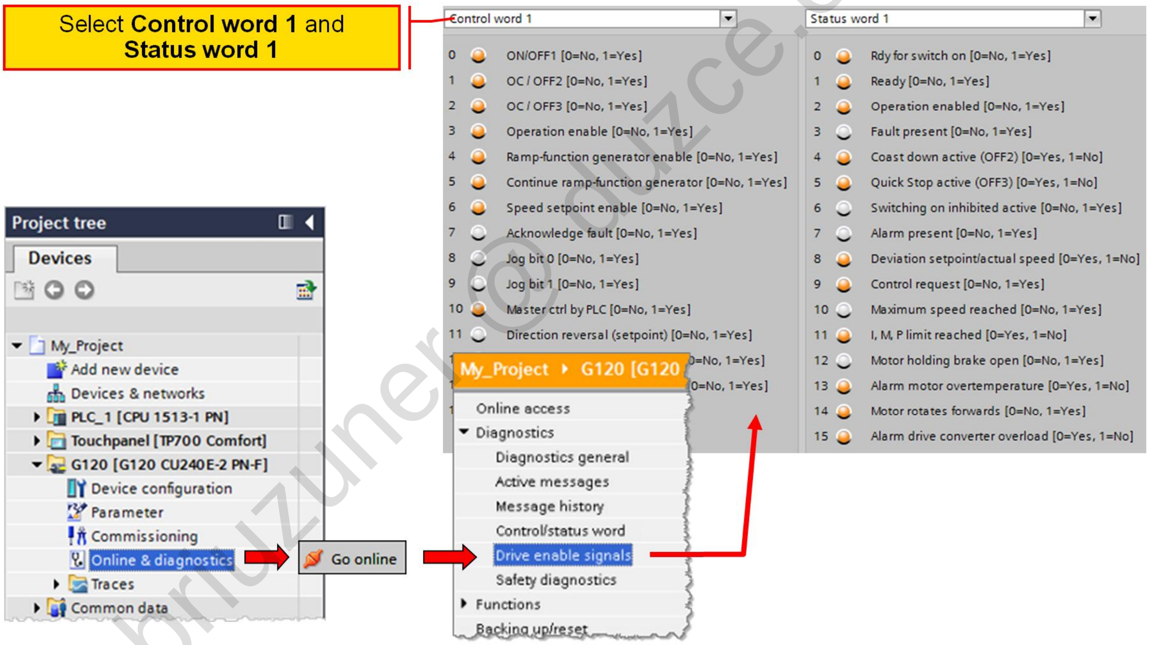


⚠ For reasons of safety, the drive is stopped when the editor is exited (e.g. Editor change, Windows task change)

SIEMENS

Monitoring Control and Status Word(s) Online

Select **Control word 1** and **Status word 1**



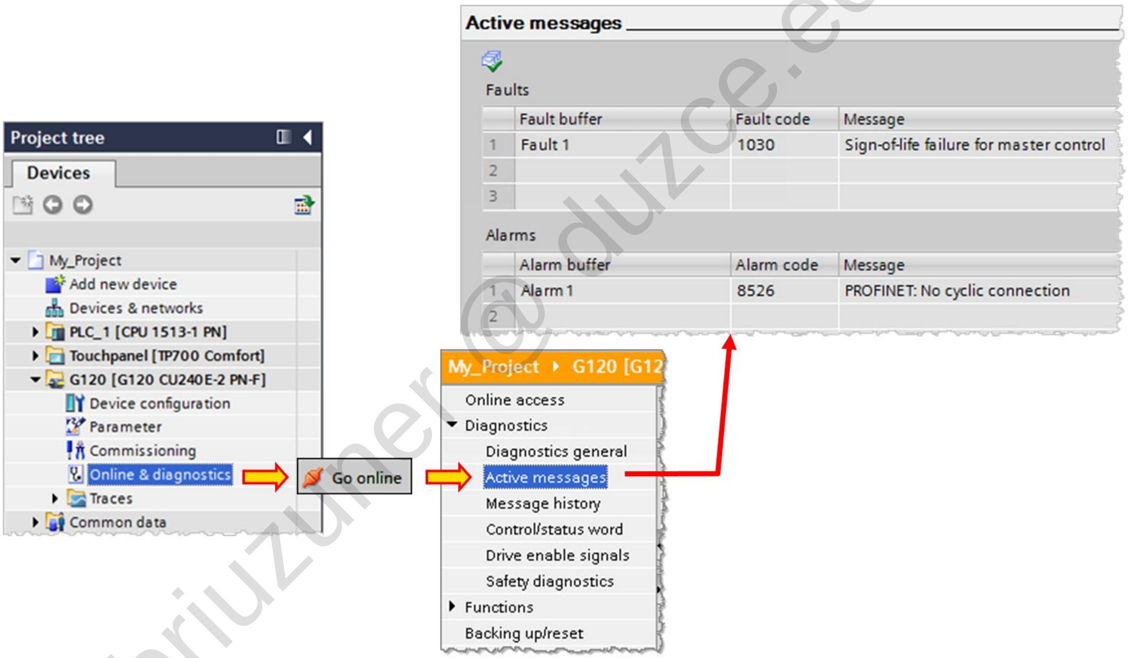
Control word 1

0	●	ON/OFF1 [0=No, 1=Yes]
1	●	OC / OFF2 [0=No, 1=Yes]
2	●	OC / OFF3 [0=No, 1=Yes]
3	●	Operation enable [0=No, 1=Yes]
4	●	Ramp-function generator enable [0=No, 1=Yes]
5	●	Continue ramp-function generator [0=No, 1=Yes]
6	●	Speed setpoint enable [0=No, 1=Yes]
7	○	Acknowledge fault [0=No, 1=Yes]
8	○	Jog bit 0 [0=No, 1=Yes]
9	○	Jog bit 1 [0=No, 1=Yes]
10	●	Master ctrl by PLC [0=No, 1=Yes]
11	○	Direction reversal (setpoint) [0=No, 1=Yes]
1		0=No, 1=Yes]
1		0=No, 1=Yes]
1		
1		

Status word 1

0	●	Rdy for switch on [0=No, 1=Yes]
1	●	Ready [0=No, 1=Yes]
2	●	Operation enabled [0=No, 1=Yes]
3	○	Fault present [0=No, 1=Yes]
4	●	Coast down active (OFF2) [0=Yes, 1=No]
5	●	Quick Stop active (OFF3) [0=Yes, 1=No]
6	○	Switching on inhibited active [0=No, 1=Yes]
7	○	Alarm present [0=No, 1=Yes]
8	●	Deviation setpoint/actual speed [0=Yes, 1=No]
9	●	Control request [0=No, 1=Yes]
10	○	Maximum speed reached [0=No, 1=Yes]
11	●	I, M, P limit reached [0=Yes, 1=No]
12	○	Motor holding brake open [0=No, 1=Yes]
13	●	Alarm motor overtemperature [0=Yes, 1=No]
14	●	Motor rotates forwards [0=No, 1=Yes]
15	●	Alarm drive converter overload [0=Yes, 1=No]

Monitoring Active Messages Online



Active messages

Faults

	Fault buffer	Fault code	Message
1	Fault 1	1030	Sign-of-life failure for master control
2			
3			

Alarms

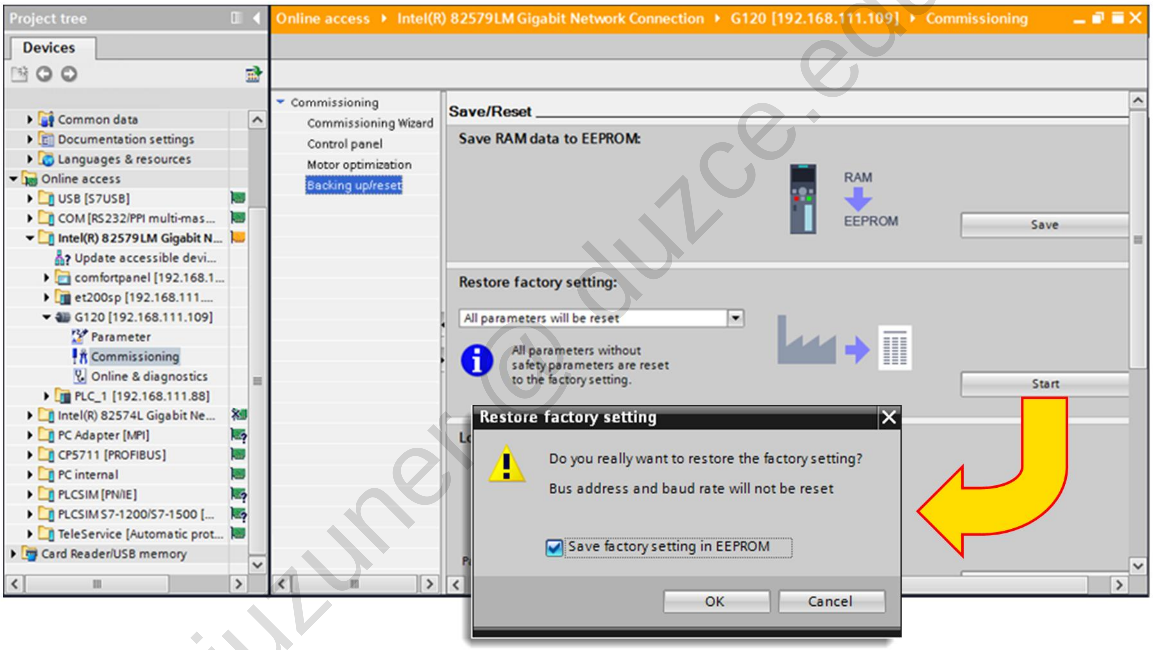
	Alarm buffer	Alarm code	Message
1	Alarm 1	8526	PROFINET: No cyclic connection
2			

SITRAIN
TIA-PRO2 / Drive with Startdrive

Page 20

Siemens AG © 2015

Exercise 1: Restoring the Factory Settings



Save/Reset

Save RAM data to EEPROM:

Restore factory setting:

All parameters will be reset

All parameters without safety parameters are reset to the factory setting.

Restore factory setting

Do you really want to restore the factory setting?

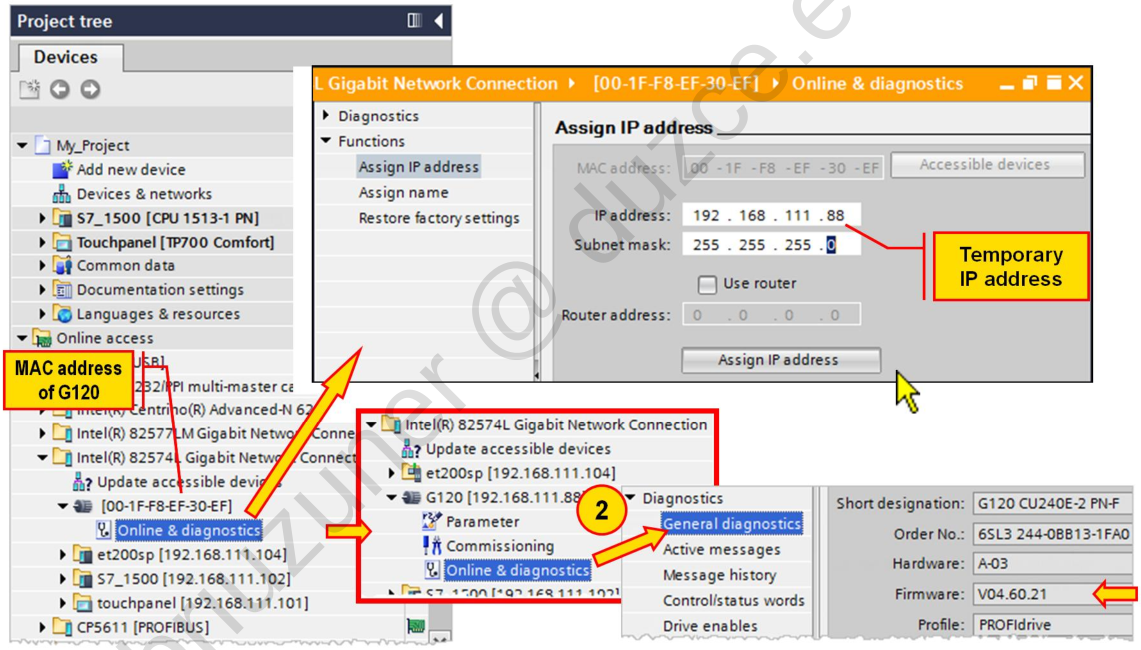
Bus address and baud rate will not be reset

☑ Save factory setting in EEPROM

OK Cancel

Exercise 2:
Reading-out the Firmware Version of the Drive

SIEMENS



Project tree

Devices

- My_Project
  - Add new device
  - Devices & networks
  - S7_1500 [CPU 1513-1 PN]
  - Touchpanel [TP700 Comfort]
  - Common data
  - Documentation settings
  - Languages & resources
- Online access
  - Intel(R) 82577LM Gigabit Network Connection
  - Intel(R) 82574L Gigabit Network Connection
    - Update accessible devices
    - [00-1F-F8-EF-30-EF]
      - Online & diagnostics
    - et200sp [192.168.111.104]
    - S7_1500 [192.168.111.102]
    - touchpanel [192.168.111.101]
  - CP5611 [PROFIBUS]

MAC address of G120

... Gigabit Network Connection ▸ [00-1F-F8-EF-30-EF] ▸ Online & diagnostics

- Diagnostics
- Functions
  - Assign IP address
  - Assign name
  - Restore factory settings

Assign IP address

MAC address: 00 -1F -F8 -EF -30 -EF

IP address: 192 . 168 . 111 . 88

Subnet mask: 255 . 255 . 255 . 0

□ Use router

Router address: 0 . 0 . 0 . 0

Temporary IP address

- Intel(R) 82574L Gigabit Network Connection
  - Update accessible devices
  - et200sp [192.168.111.104]
  - G120 [192.168.111.88]
    - Parameter
    - Commissioning
    - Online & diagnostics
  - S7_1200 [192.168.111.102]

2

- Diagnostics
  - General diagnostics
  - Active messages
  - Message history
  - Control/status words
  - Drive enables

Short designation: G120 CU240E-2 PN-F

Order No.: 6SL3 244-0BB13-1FA0

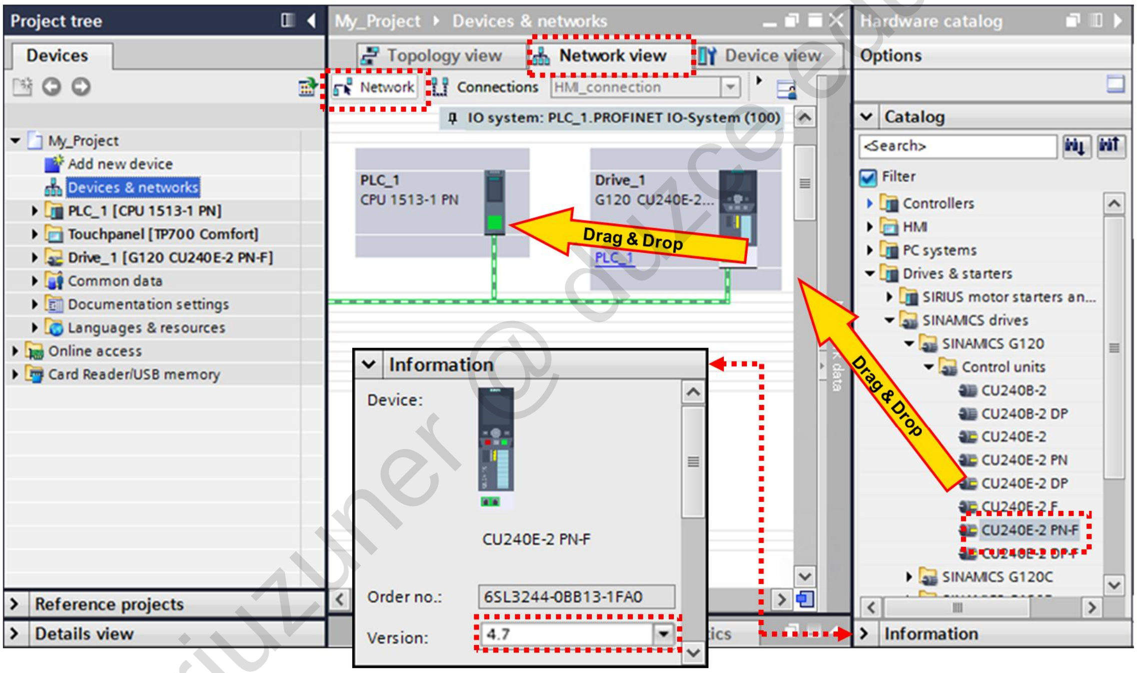
Hardware: A-03

Firmware: V04.60.21

Profile: PROFIdrive

Exercise 3: Inserting and Networking the Drive in the Offline Project

SIEMENS

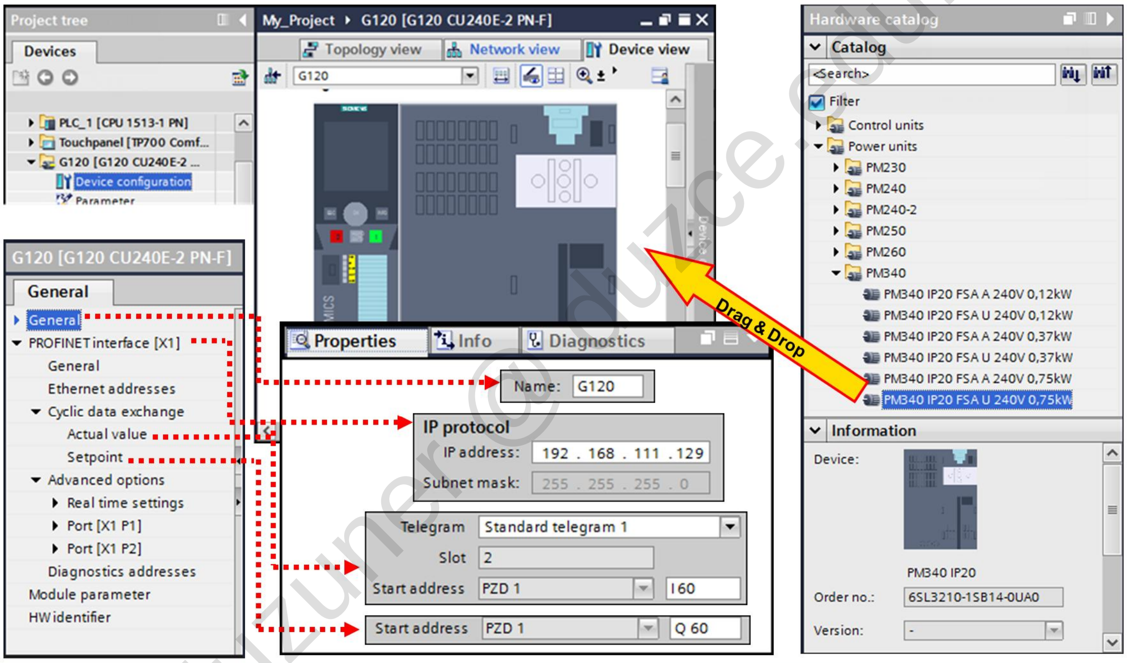


Order no.: 6SL3244-0BB13-1FA0

Version: 4.7

Exercise 4:
Configuring and Parameterizing the Drive

SIEMENS



Name: G120

IP protocol

IP address: 192 . 168 . 111 . 129

Subnet mask: 255 . 255 . 255 . 0

Telegram: Standard telegram 1

Slot: 2

Start address: PZD 1 | I 60

Start address: PZD 1 | Q 60

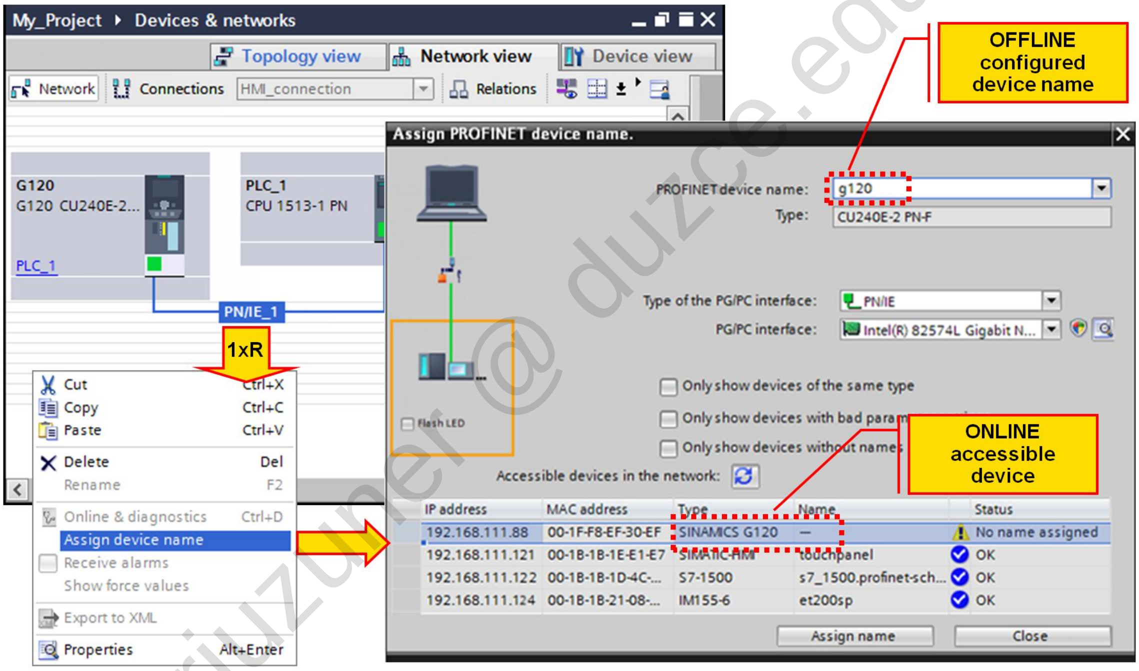
Device: PM340 IP20

Order no.: 6SL3210-1SB14-0UA0

Version: -

Exercise 5:
Assigning the PROFINET Device Name ONLINE

SIEMENS



OFFLINE configured device name

ONLINE accessible device

Assign PROFINET device name.

PROFINET device name: g120

Type: CU240E-2 PN-F

Type of the PG/PC interface: PN/IE

PG/PC interface: Intel(R) 82574L Gigabit N...

□ Only show devices of the same type

□ Only show devices with bad param...

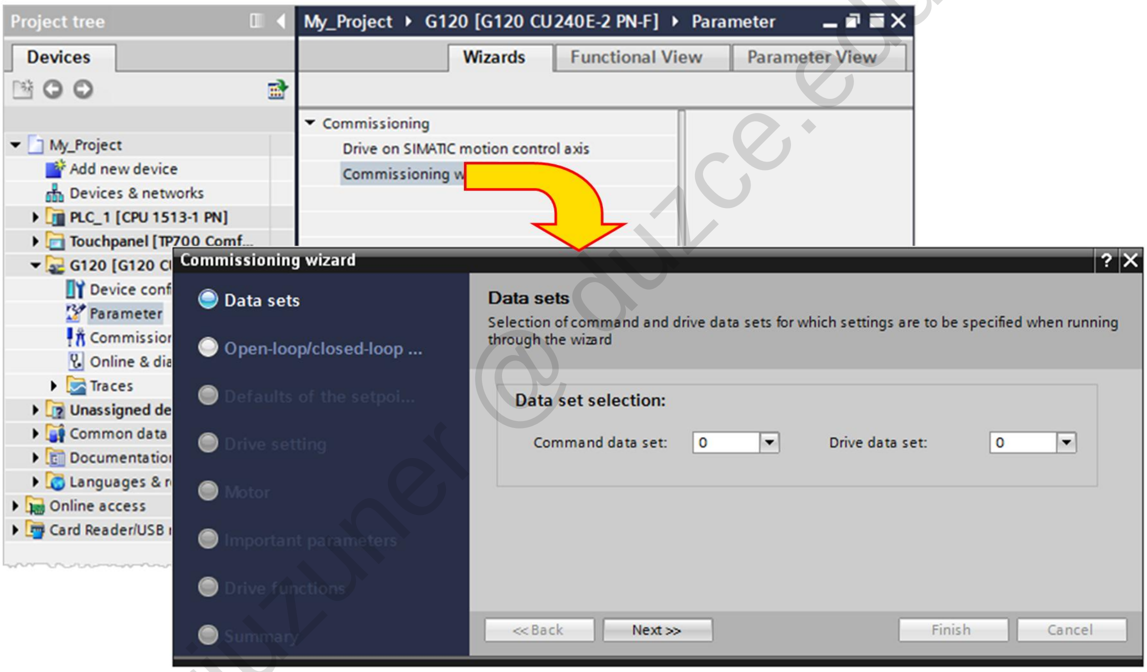
□ Only show devices without names

Accessible devices in the network:

IP address	MAC address	Type	Name	Status
192.168.111.88	00-1F-F8-EF-30-EF	SINAMICS G120	--	⚠ No name assigned
192.168.111.121	00-1B-1B-1E-E1-E7	SIMATIC-HMI	touchpanel	✓ OK
192.168.111.122	00-1B-1B-1D-4C-...	S7-1500	s7_1500.profinet-sch...	✓ OK
192.168.111.124	00-1B-1B-21-08-...	IM155-6	et200sp	✓ OK

SITRAIN
TIA-PRO2 / Drive with Startdrive Page 25 Siemens AG © 2015

Exercise 6: Parameterizing the Drive OFFLINE with the Commissioning Wizard



Data sets

Selection of command and drive data sets for which settings are to be specified when running through the wizard

Data set selection:

Command data set: 0

Drive data set: 0

SITRAIN

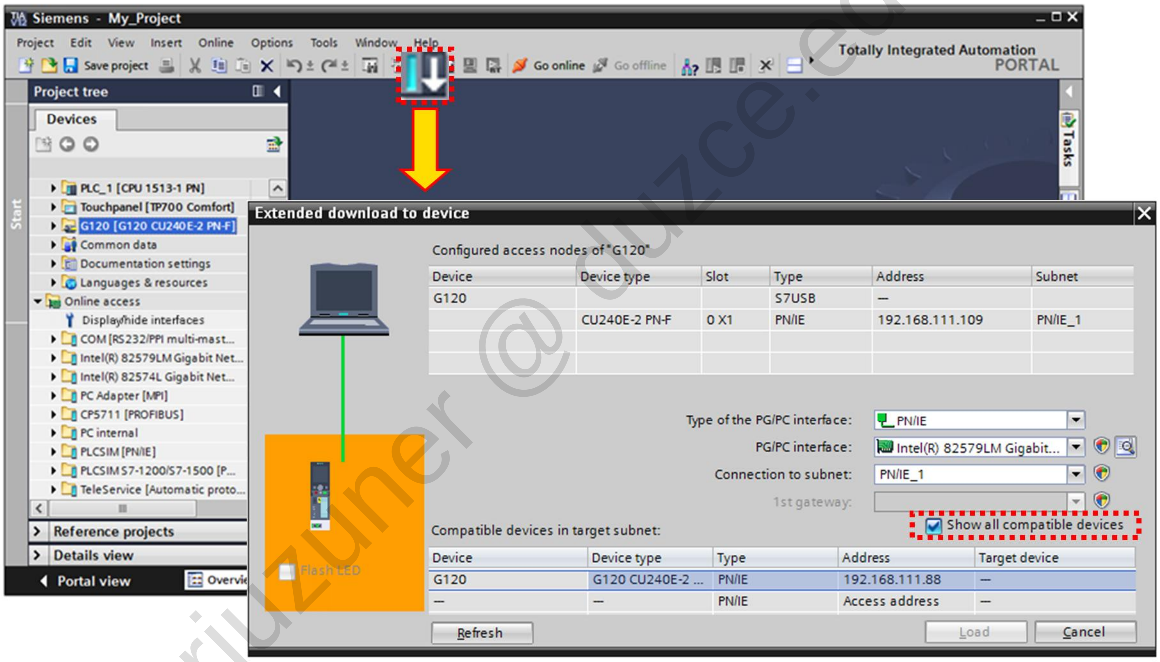
TIA-PRO2 / Drive with Startdrive

Page 26

Siemens AG © 2015

Exercise 7:
Downloading the Parameterization into the Drive

SIEMENS



Extended download to device

Configured access nodes of "G120"

Device	Device type	Slot	Type	Address	Subnet
G120			S7USB	---	
	CU240E-2 PN-F	0 X1	PN/IE	192.168.111.109	PN/IE_1

Type of the PG/PC interface: PN/IE

PG/PC interface: Intel(R) 82579LM Gigabit...

Connection to subnet: PN/IE_1

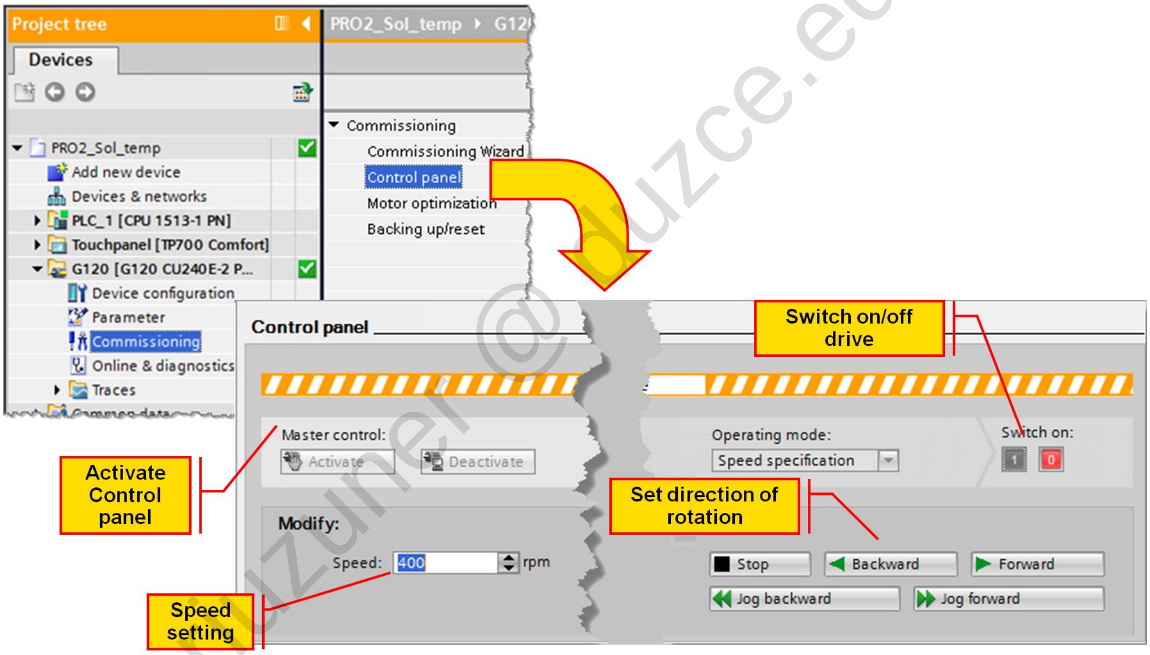
1st gateway:

☑ Show all compatible devices

Compatible devices in target subnet:

Device	Device type	Type	Address	Target device
G120	G120 CU240E-2 ...	PN/IE	192.168.111.88	---
---	---	PN/IE	Access address	---

Exercise 8:
Operating the Drive via the Control Panel



Operating mode: Speed specification

Speed: 400 rpm

Switch on/off drive

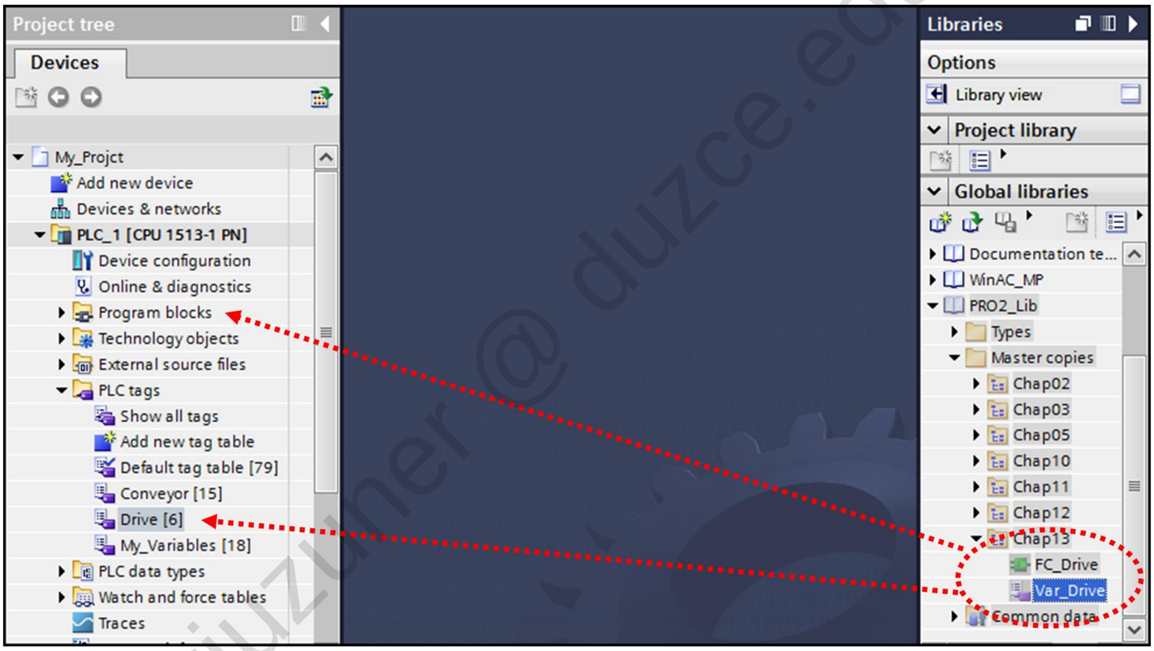
Activate Control panel

Set direction of rotation

Speed setting

SIEMENS

Exercise 9: Commissioning "FC_Drive" (FC120)



Project tree

Devices

- My_Projct
  - Add new device
  - Devices & networks
  - PLC_1 [CPU 1513-1 PN]
    - Device configuration
    - Online & diagnostics
    - Program blocks
    - Technology objects
    - External source files
    - PLC tags
      - Show all tags
      - Add new tag table
      - Default tag table [79]
      - Conveyor [15]
      - Drive [6]
      - My_Variables [18]
    - PLC data types
    - Watch and force tables
    - Traces

Libraries

Options

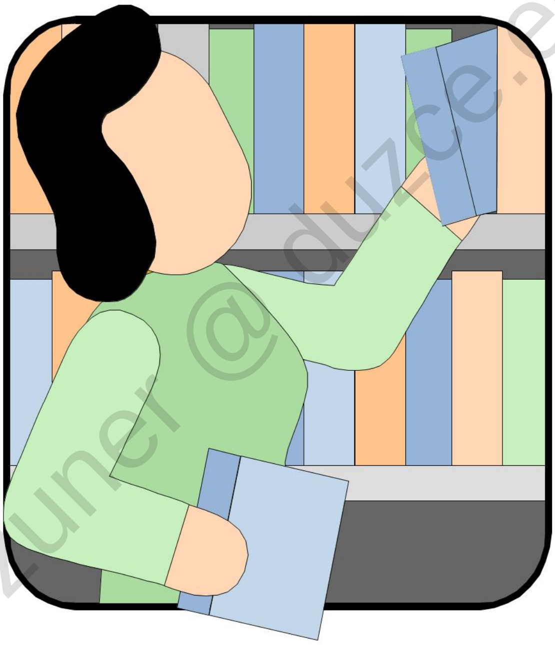
Library view

Project library

Global libraries

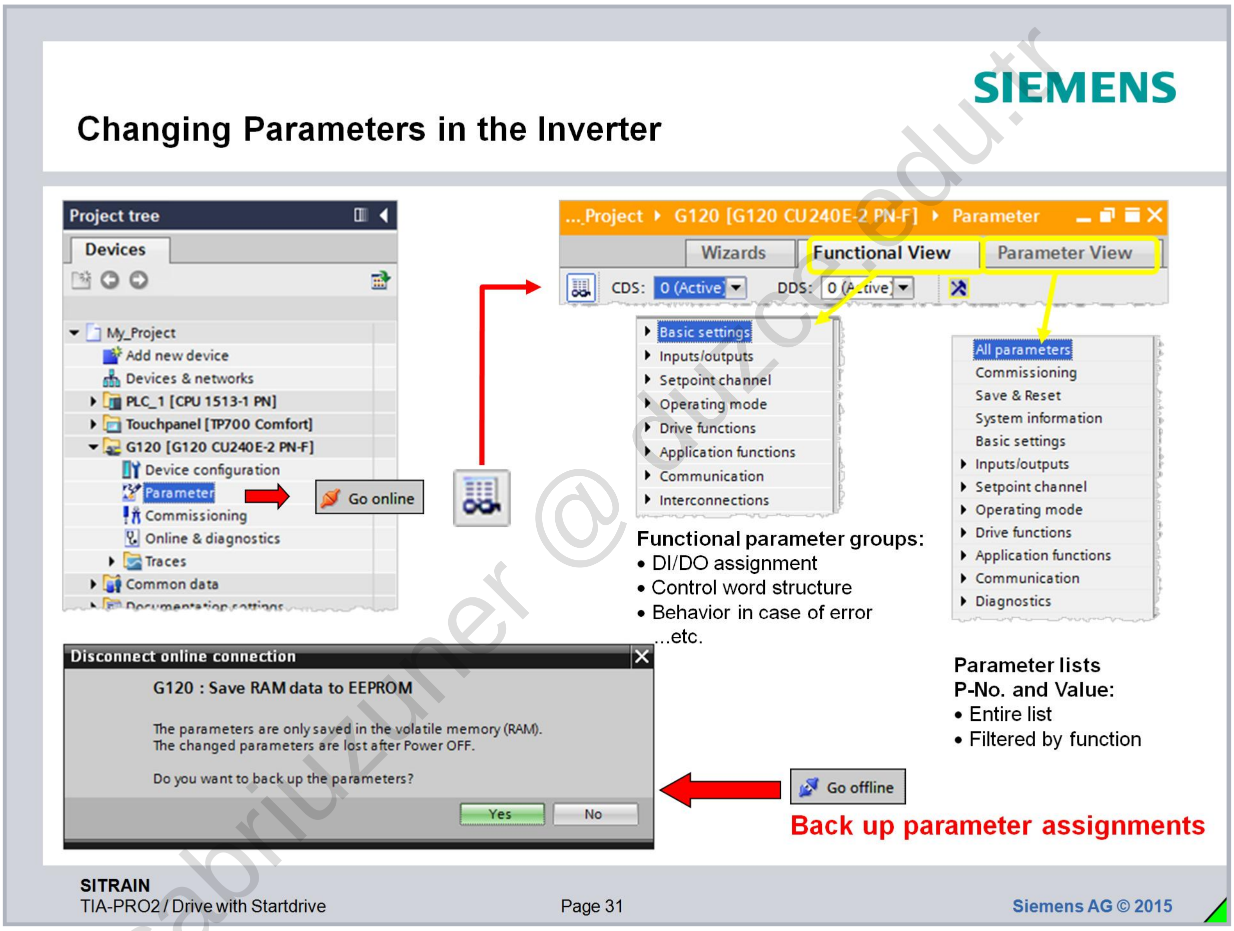
- Documentation te...
- WinAC_MP
- PRO2_Lib
  - Types
  - Master copies
    - Chap02
    - Chap03
    - Chap05
    - Chap10
    - Chap11
    - Chap12
    - Chap13
      - FC_Drive
      - Var_Drive
  - Common data

Additional Information



SIEMENS

Changing Parameters in the Inverter



Functional parameter groups:
- DI/DO assignment
- Control word structure
- Behavior in case of error

...etc.

Parameter lists P-No. and Value:
- Entire list
- Filtered by function

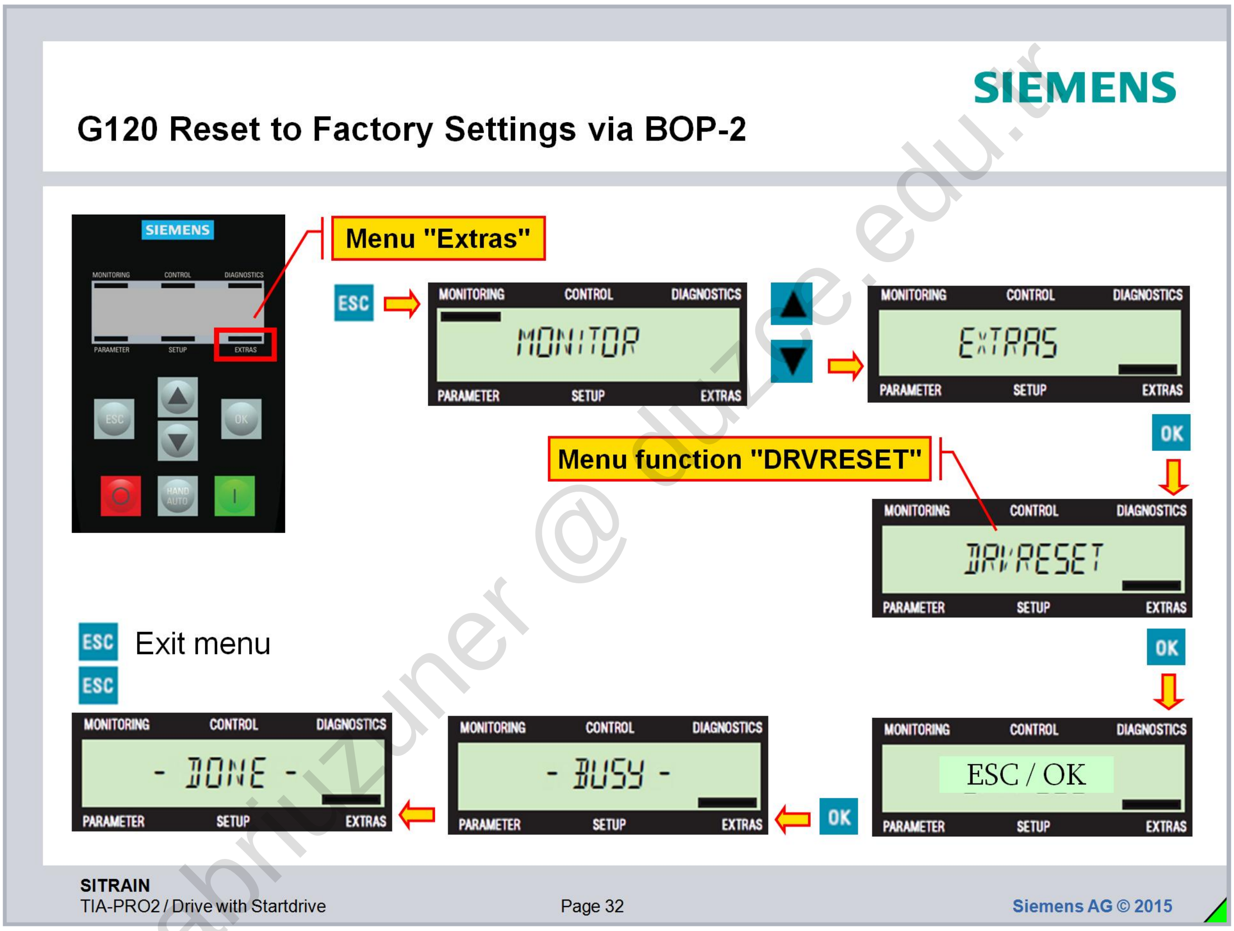
Disconnect online connection

G120 : Save RAM data to EEPROM

The parameters are only saved in the volatile memory (RAM). The changed parameters are lost after Power OFF.

Do you want to back up the parameters?

Back up parameter assignments



SIEMENS

G120 Reset to Factory Settings via BOP-2

Menu "Extras"

Menu function "DRVRESET"

ESC Exit menu

SITRAIN
TIA-PRO2 / Drive with Startdrive Page 32 Siemens AG © 2015

Contents 5

5. Introduction to Industrial Communication

5.1. Objectives

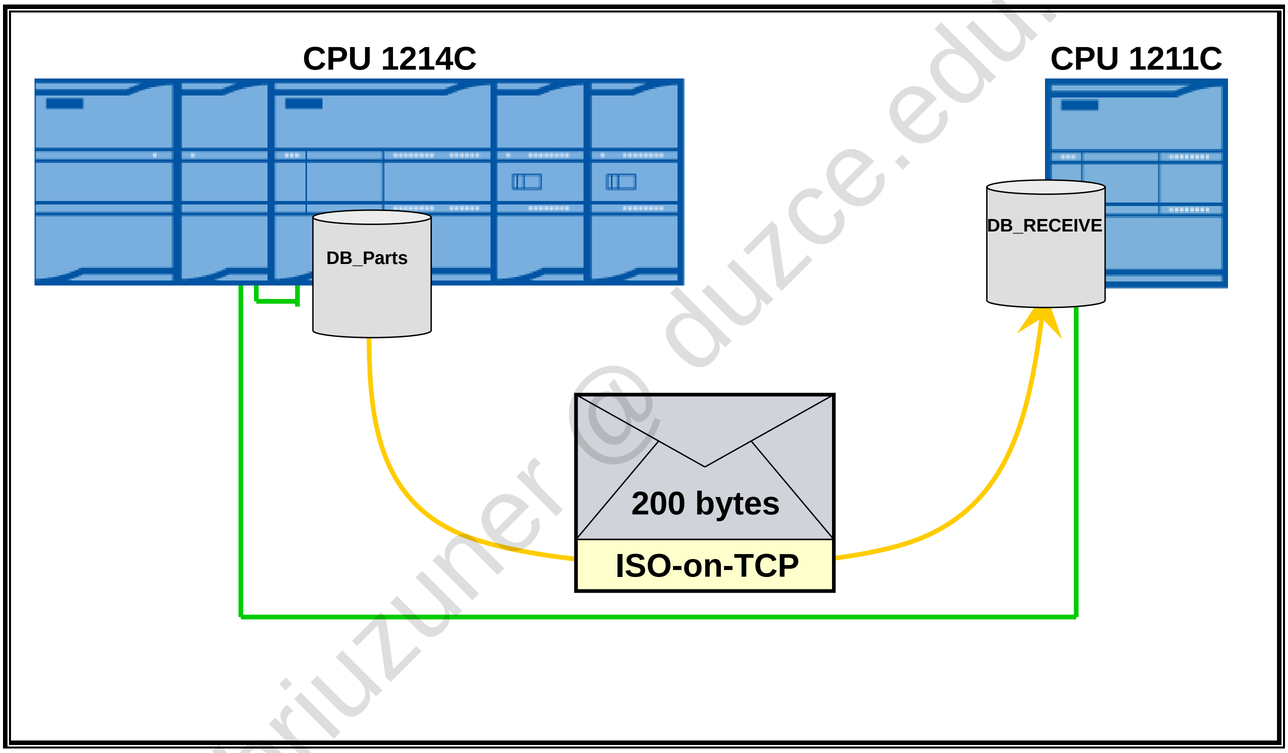
At the end of the chapter the participant will ...

- … understand the principle of CPU-CPU communication.
- … be familiar with the features of the transport protocol "ISO-on-TCP".
- … understand the difference between packet-oriented and data flow-oriented communication.
- … be able to create a communications connection between 2 CPUs.

Objectives

In this chapter, the industrial communication via Ethernet between two S7-1200 CPUs is dealt with. The available communication services are compared. In a concluding exercise, an ISO-on-TCP connection between the two CPUs of the training area is programmed.

5.2. Task Description: Creating an "ISO-on-TCP" Connection

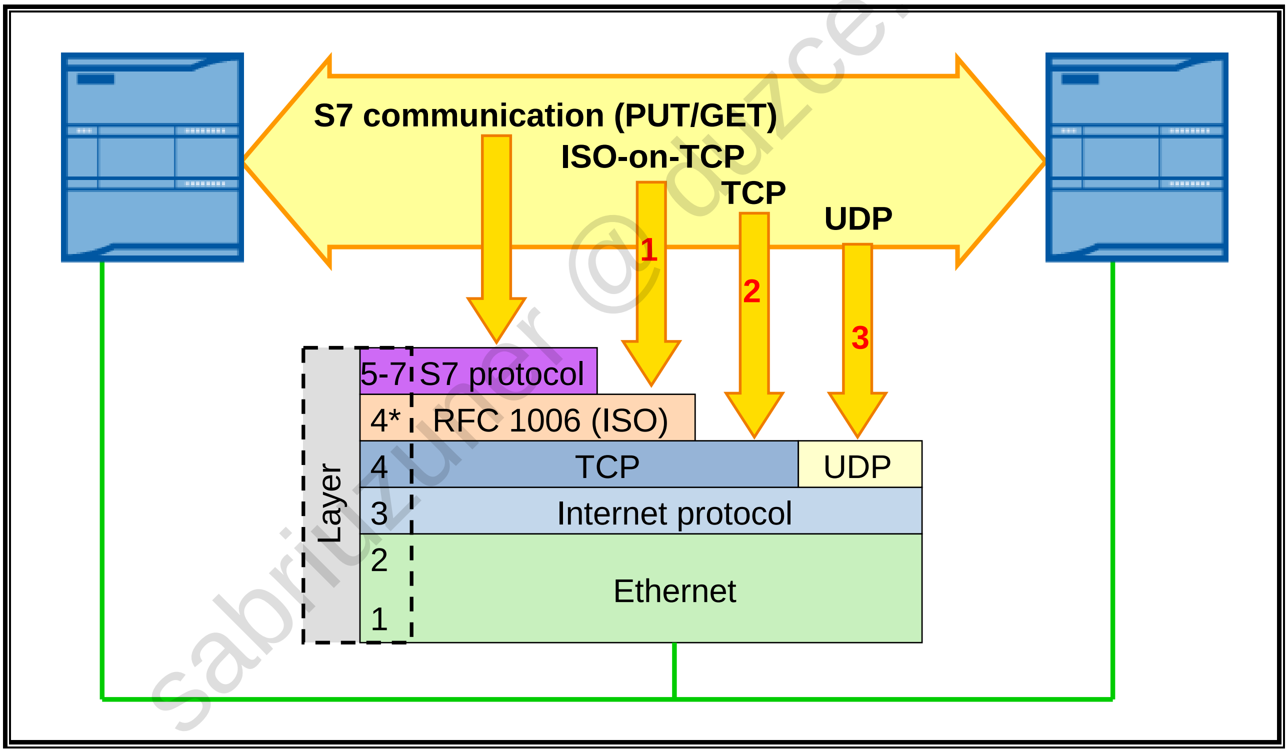


Task Description

An ISO-on-TCP connection between the CPUs of your training area is to be programmed. Via this connection, the ARRAY variable "DB_Parts".Part_Weight is to be sent from the controller "PLC_1" to the still to be configured controller "PLC_2" and stored there in the DB "DB_RECEIVE" in the ARRAY variable "Receive_buffer".

To minimize data traffic, sending is not to be continuous. Instead, sending is to occur under the same conditions as the saving of weight values in "DB_Parts".

5.3. S7-1200 Ethernet Communication Services in the ISO/OSI Communication Model



ISO/OSI Model

Communication tasks are divided for international comparison in 7 layers according to the ISO/OSI model. Included are also, among other things, the types of communication shown in the picture which are supported by the S7-1200.

Every layer has an exactly defined task area and, in each case, a defined interface to the higher-level and the subordinate layer.

Abbreviation

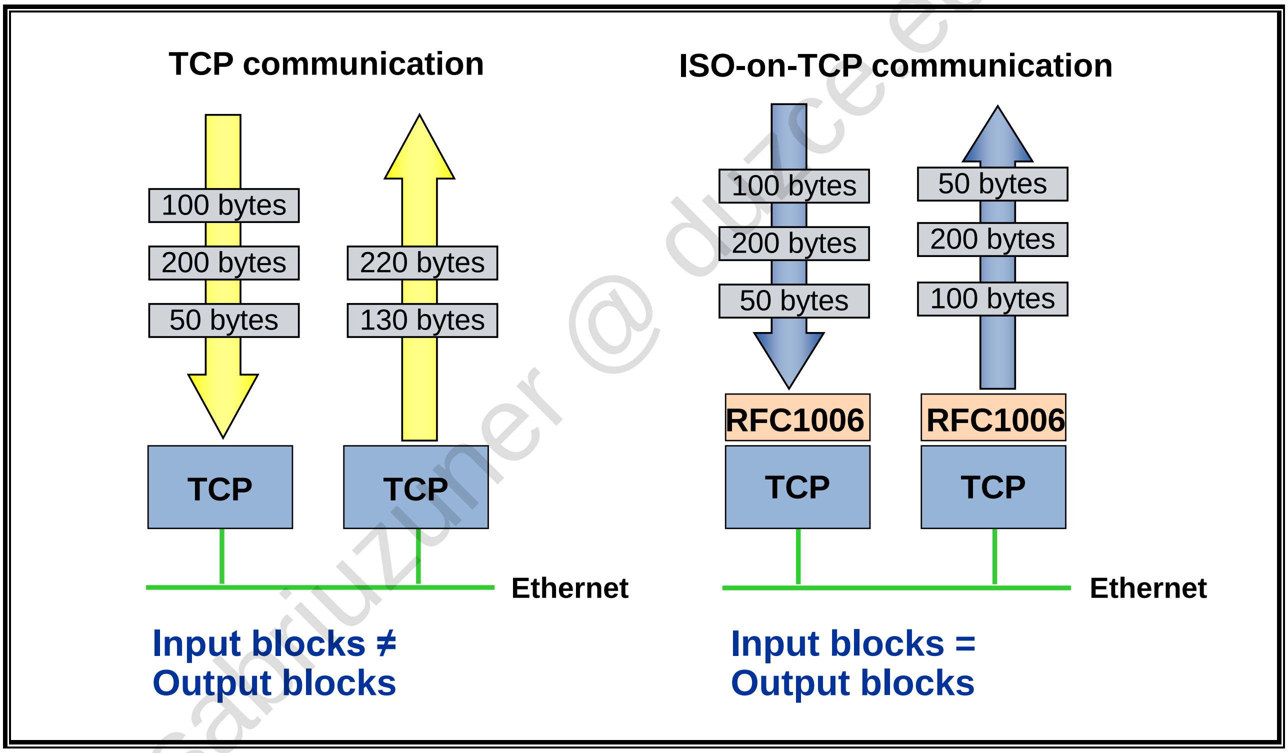
International Organization for Standardization / Open Systems Interconnection Reference Model

Ethernet Communication Services

For communication via (industrial) Ethernet, there are various services in the SIMATIC environment. These differ with regard to:

- Data security
- Amounts of data
- Data handling
- Routing capability and
- Engineering effort

5.3.6. Data Flow-oriented and Message-oriented Communication



Data Flow-oriented and Packet-oriented Communication

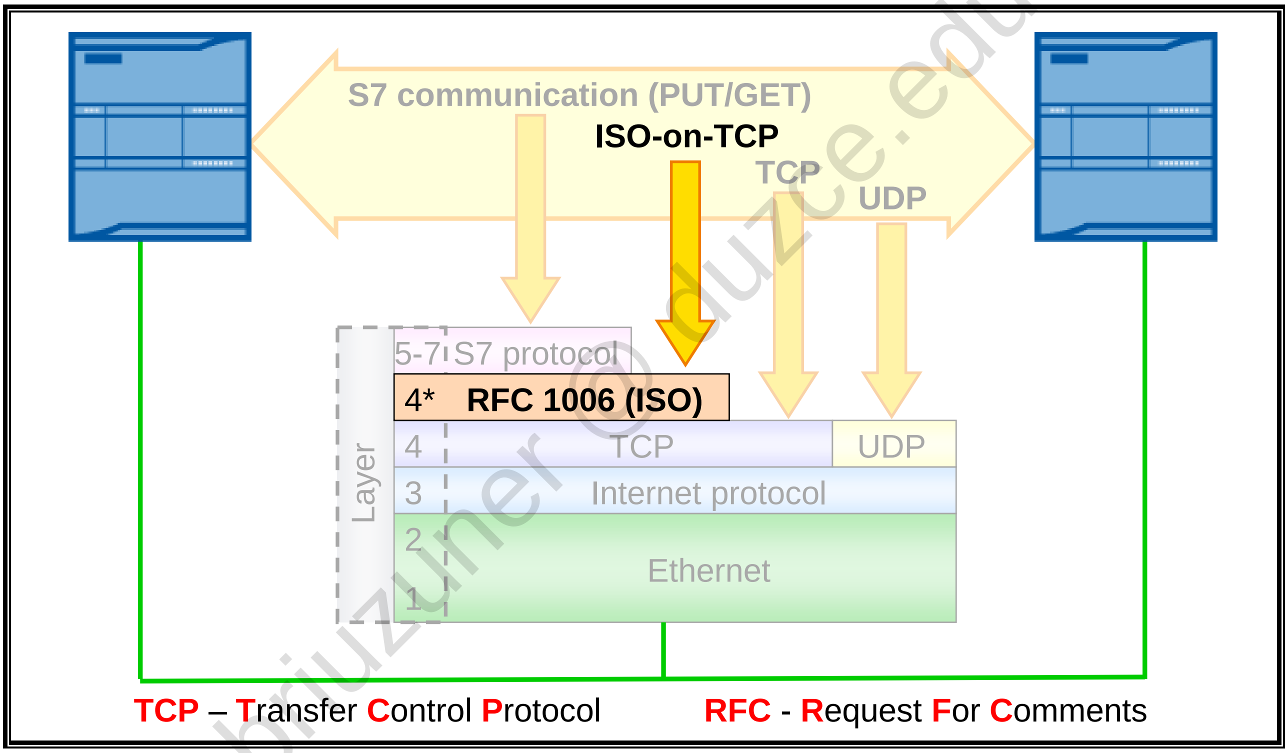
When data is transferred with the Transmission-Control-Protocol (TCP), the transmission takes place in the form of a data flow. Neither information on length nor information about the beginning and end of a message is transferred. The receiver, however, cannot recognize where a message ends in the data flow and where the next one begins in the data flow. A read task of the receiver thus only supplies as much data as is currently found in the receive buffer. This means that possibly more than one data block can be found in the receive buffer.
This process is well suited for communicating with third-party systems or computer systems.

Behavior of the RFC 1006 Protocol Expansion

In most automation applications it is, however, essential to work message-oriented. Self-contained message blocks are sent via a connection which is also recognized as such by the receiver. In order to ensure this, RFC 1006 specifies which information (in the form of a header) must be added to the data to be transferred.

RFC 1006 thus provides applications which are based on the data flow-oriented TCP protocol with a message-oriented transmission.

5.3.7. ISO-on-TCP Communication



Background

Historically seen, the ISO Transport Protocol, as Layer 4 interface of the ISO-OSI reference model, was the first Ethernet protocol in SIMATIC. The great advantage of this protocol lies in the message-oriented transmission of data whereby the processing within the automation system becomes easier.

Since, however, the Layer 3 implementation is missing (no IP addresses) with the ISO Transport Protocol, no network addressing and thus no routing is possible.

ISO-on-TCP

The packet-oriented transmission of data is the great advantage of the ISO Transport Protocol. However, because of the expanding networking, the missing routing functionality developed into an increasing disadvantage.

Since the routing-capable TCP/IP protocol became increasingly popular because of the internet, an attempt was made to combine the advantages of both protocols. In the expansion RFC1006 (RFC = Request for Comments) "ISO on top of TCP", also called "ISO-on-TCP", the mapping of characteristics of the ISO transport on the TCP/IP protocol is set down. This protocol is available in all current modules of SIMATIC S7.

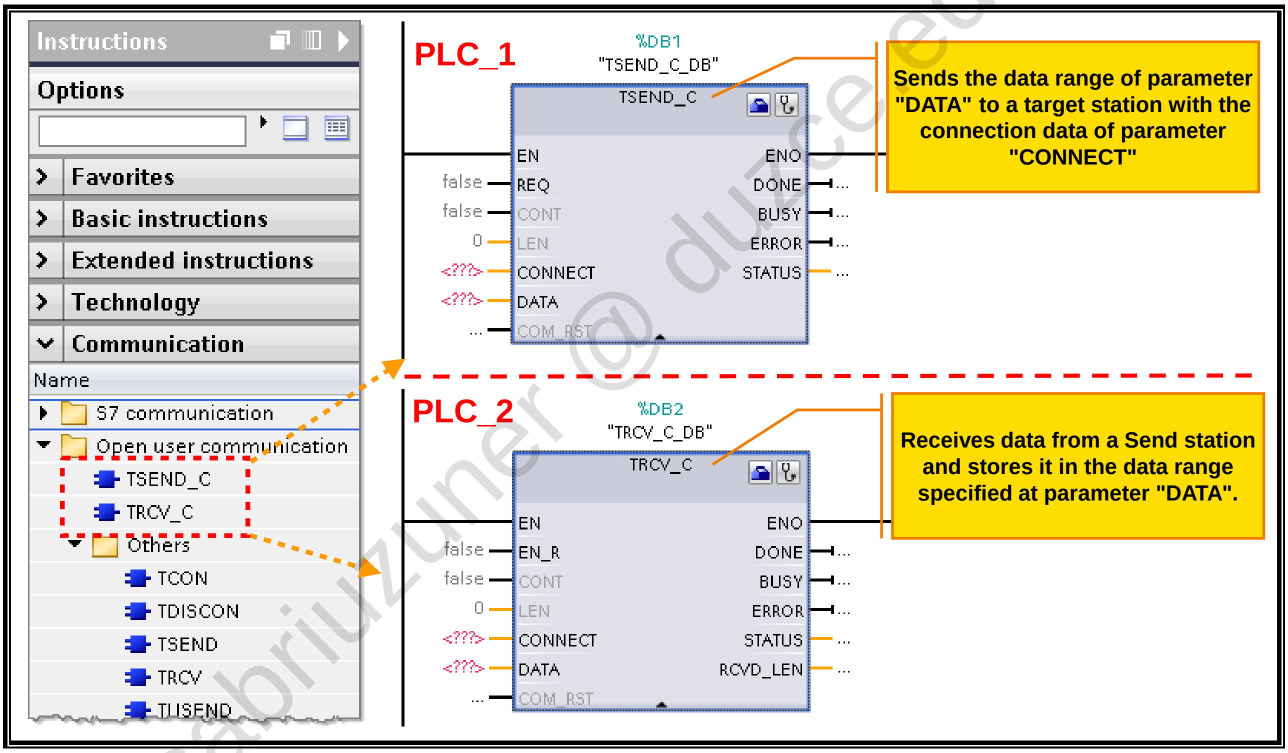
Advantages of the ISO-on-TCP Protocol:

- Fast communication
- Suitable for the transmission of medium to large amounts of data (<= 8192 bytes)
- Routing-capable (i.e. can be used in WAN)
- Packet-oriented data transmission
- Dynamic data lengths are possible

Disadvantages/Features of the ISO-on-TCP Protocol:

- Mainly applicable in SIMATIC homogenous structures
- Greater programming effort needed to manage the data

5.4. Combined Blocks for the Connection Programming



Sends the data range of parameter "DATA" to a target station with the connection data of parameter "CONNECT"

Receives data from a Send station and stores it in the data range specified at parameter "DATA".

Open User Communication

If the integrated Ethernet interface of the CPU is used for the Ethernet communication, the so-called "Open User Communication" is used.
Included in the "open" communication services, that is those whose inner structure is open, are TCP, UDP and ISO-on-TCP. Connections between SIMATIC controllers which use one of these services are not configured in the Network view of STEP7 (such as connections to HMI devices) but are programmed. For this, there are various blocks available in the Instruction Catalog.

Connection-oriented and Connectionless Services

Connection-oriented services, this includes TCP and ISO-on-TCP, first of all establish a connection to the communication partner and then send the data (also bidirectional). If the transmission process is completed, the connection is disconnected. All data is acknowledged by the receiver. The sender resends all unacknowledged telegrams. This is comparable to telephoning with a telephone. First, a connection is established (dial number + pick up), then, information is exchanged and in the end you hang up, that is, the connection is disconnected.

Connectionless services, such as UDP, send their data without first establishing a connection, similar to a "walkie-talkie" (blocks: TUSEND / TURCV). It can therefore not be ensured whether the data is received by the receiver. The advantage lies here in the speed, since less administrative data for flow control have to be sent and interpreted.

Combined Connection Blocks

Connection-oriented communication services can be used in two ways:

- Single blocks

  With the single blocks, targeted connection actions can be executed.

  – TCON / TDISCON: connect and disconnect a connection

  – TSEND / TRCV: send or receive data after establishing a connection

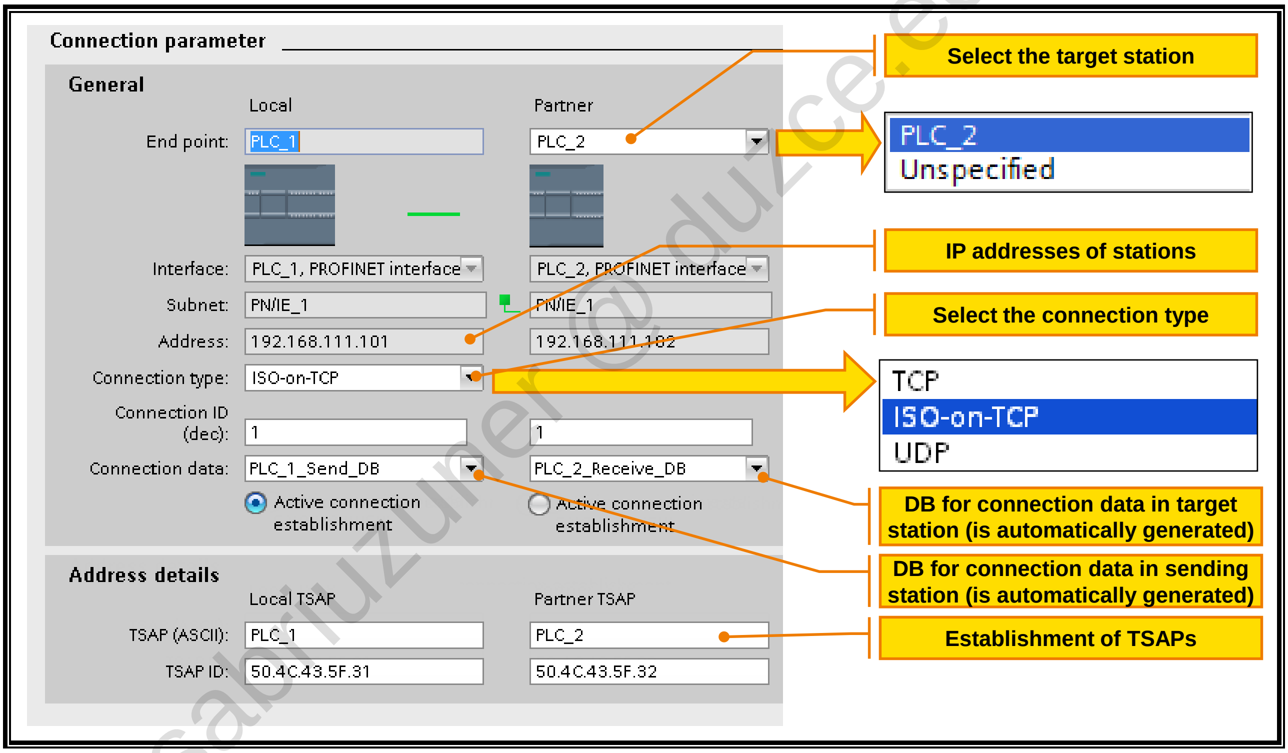
- Combined connection blocks

  – TSEND_C: Connect a connection (when Active connection establishment is active) , Send data and disconnect the connection (when Active connection establishment is active)

  – TRCV_C: Connect a connection (when Active connection establishment is active), Receive data and disconnect the connection (when Active connection establishment is active)

  With combined connection blocks, a connection to the communication partner is established, data is sent/received and the connection is disconnected with just one call.

5.4.1. Connection Parameterization via Block Properties (Sending Station with TSEND_C)



Connection parameter

General

	Local	Partner
End point:	PLC_1	PLC_2
Interface:	PLC_1, PROFINET interface	PLC_2, PROFINET interface
Subnet:	PN/IE_1	PN/IE_1
Address:	192.168.111.101	192.168.111.102
Connection type:	ISO-on-TCP	
Connection ID (dec):	1	1
Connection data:	PLC_1_Send_DB	PLC_2_Receive_DB
	● Active connection establishment	○ Active connection establishment

Address details

	Local TSAP	Partner TSAP
TSAP (ASCII):	PLC_1	PLC_2
TSAP ID:	50.4C.43.5F.31	50.4C.43.5F.32

Select the target station

IP addresses of stations

Select the connection type

DB for connection data in target station (is automatically generated)

DB for connection data in sending station (is automatically generated)

Establishment of TSAPs

Connection Parameterization via Block Properties

The TSEND_C block works connection-oriented, that is, a partner CPU must first of all be configured. After the block is called, this can be done very easily in the Inspector window under *Properties > Configuration > Connection parameter*.

Connection Parameters

- End point

  Here, the partner CPU is configured. It can be in the same project (here: "PLC_2"), or a "not specified" partner is created. In both cases, the addressing of the partner occurs via its IP address.

- Address

  Here, the IP address of the partner is specified. If a specified partner was selected before, its IP address is automatically adopted. For not specified partners, it must be entered manually.

- Connection type

  Via the connection type, a communication service is selected and thus the properties of the communication.

- Connection ID (dec)

  Via the connection ID, the number of the connection within the CPU is specified. Depending on the CPU used, several simultaneous connections are mastered. The connection ID must be unique within the CPU. The exact number of simultaneously possible Ethernet connections can be found in the manual.

- Connection data

  The connection data, which the "TSEND_C" block accesses, is stored in a DB. This DB contains the configured information on the connection type and the partner CPU (IP address etc.). The DB is automatically created when "New" is selected in the field "Connection data".

Address Details

- TSAP (ASCII)

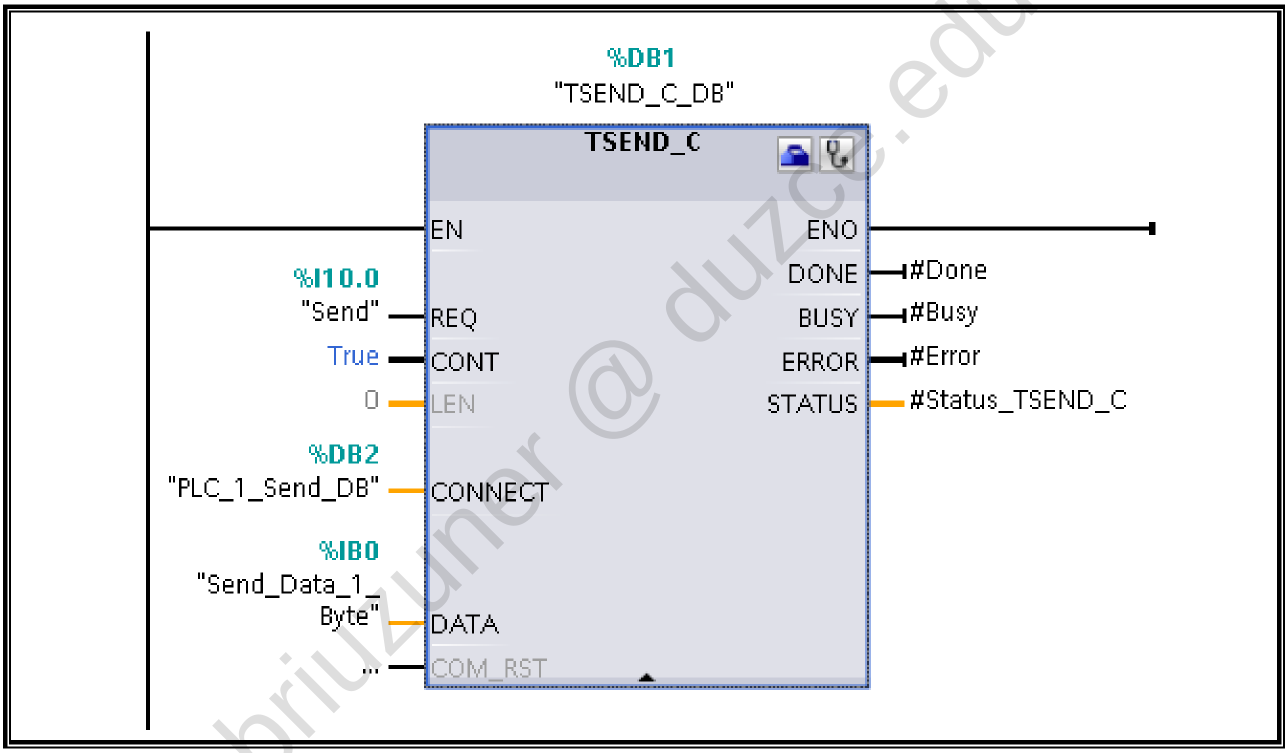
  The Transport Service Access Point (TSAP) is used for ISO-on-TCP communication in order to address the transmitted data at the receiver. In the network, the receiver is addressed by means of the MAC address. At the receiver, the received data is first of all stored in the receive buffer of the Ethernet interface and then fetched from there by the operating system of the CPU. The application within the CPU is addressed via the TASP-ID.

  This type of addressing is comparable to a multiple family dwelling with a collective mailbox and a caretaker who distributes the mail. In order to now address an occupant, the house number (MAC address) must first of all be specified. The letter (data) first lands in the mailbox (receive buffer of the Ethernet interface). So that the caretaker can deliver the letter, the name of the occupant (TSAP-IP) must also be on the letter.

- TSAP-ID

  The TASP-ID is automatically generated from the entered TSAP.

5.4.2. Parameterized Send Block TSEND_C



TSEND_C

The TSEND_C block processes send tasks as follows:

1. Establish connection to the communication partner
2. Send data to parameter DATA
3. Disconnect connection.

Parameterized Send Block TSEND_C

The TSEND_C block has several parameters which are explained in the following.

- REQ

  If a rising edge is detected at input REQ, a send task is started.

- CONT

  The input CONT controls the communication connection.

  FALSE: The connection is disconnected.
  TRUE: The connection is established.

- LEN

  The parameter LEN specifies the maximum number of bytes which can be sent with the task. If purely symbolic values are specified at parameter DATA, the parameter LEN must have the value "0".

- CONNECT

  In order to parameterize the communication connections for TCP, UDP and ISO-on-TCP, a connection describing DB with a fixed structure is used. The data structure contains the necessary parameters which are required to establish the connection. The connection describing DB is automatically created for a new connection by the connection parameterization of the Open User Communication when using the instructions TSEND_C, TRCV_C or TCON.

- DATA

  The send data range is specified via the parameter DATA. The addressing takes place via a pointer to the send range which contains the address and the length of the data to be sent (for example, "P#DB80.DBX0.0 byte 20" (pointer to a data range of 20 bytes in DB80, beginning from bit 0.0)) or symbolically, in order to address, for example, ARRAYs or structures (for example, ""DB_Parts".Parts_Management" (pointer to the structure variable "Parts_Management" in "DB_Parts")).

- COM_RST

  Causes a restart of the instruction:

  FALSE: Irrelevant
  TRUE: Complete restart of the instruction whereby the existing connection is disconnected and a new connection is established.

- DONE

  The status parameter DONE indicates the processing status of the current send task.

  FALSE: Task has not yet started or is still being processed.
  TRUE: Task was executed without error.

- BUSY

  The status parameter BUSY indicates whether the current call of the block has already been completed or is not active since "TSEND_C" is executed asynchronously. As long as BUSY = "TRUE", no new send task is accepted.

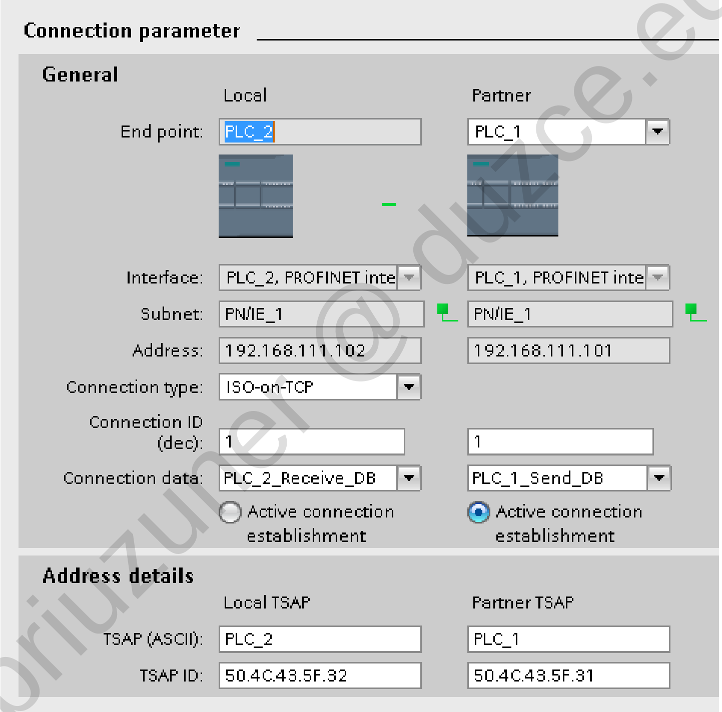
- ERROR

  The status parameter ERROR indicates with "TRUE" whether errors occurred during the send task. Error details can be queried at the parameter STATUS.

- STATUS

  The parameter STATUS delivers a 2 byte HEX code which contains information on the send state or errors that have occurred.

5.4.3. Connection Parameterization via Block Properties (Receiving Station with TRCV_C)



Connection parameter

General

	Local	Partner
End point:	PLC_2	PLC_1
Interface:	PLC_2, PROFINET inte	PLC_1, PROFINET inte
Subnet:	PN/IE_1	PN/IE_1
Address:	192.168.111.102	192.168.111.101
Connection type:	ISO-on-TCP	
Connection ID (dec):	1	1
Connection data:	PLC_2_Receive_DB	PLC_1_Send_DB
	○ Active connection establishment	● Active connection establishment

Address details

	Local TSAP	Partner TSAP
TSAP (ASCII):	PLC_2	PLC_1
TSAP ID:	50.4C.43.5F.32	50.4C.43.5F.31

Connection Parameterization via Block Properties

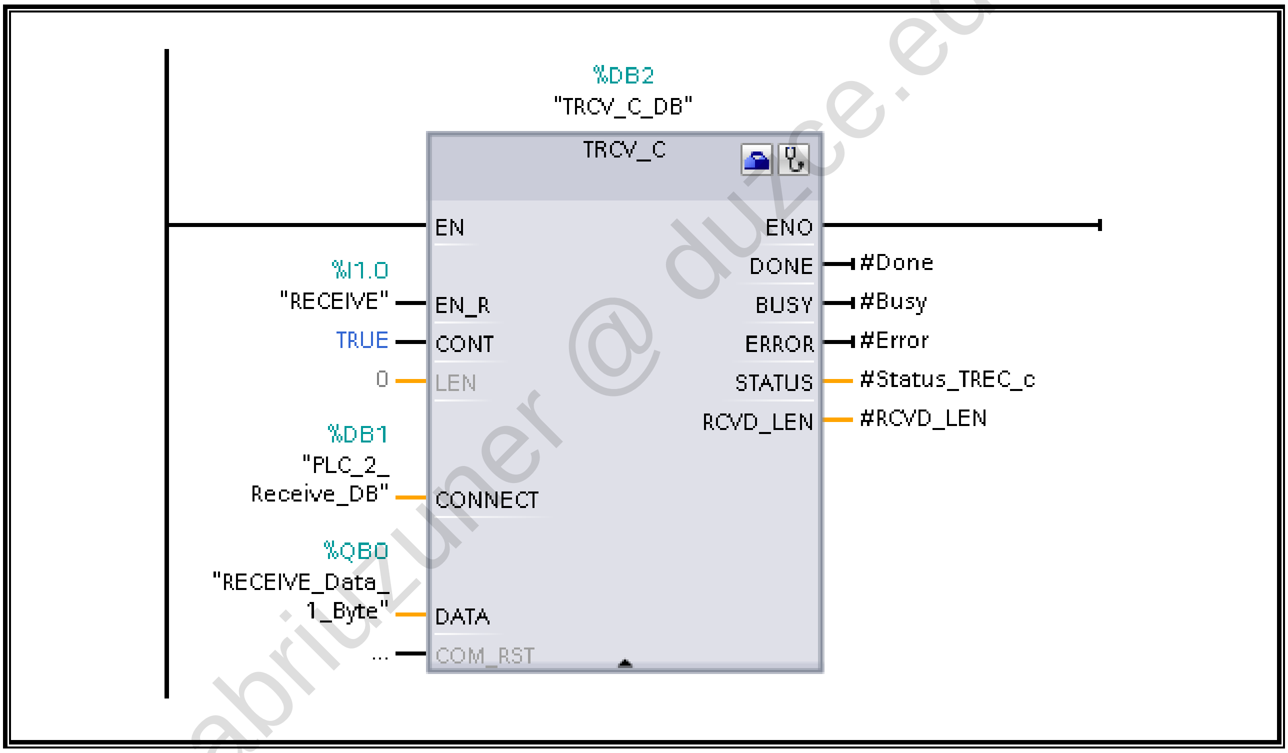
The TRCV_C block, as well, works connection-oriented, that is, a partner CPU must first of all be configured. After the block is called, this can be done very easily in the Inspector window under *Properties > Configuration > Connection parameter*.

If the partner CPU was already programmed beforehand, and both stations are located in the same project, only the "connection data" DBs still have to be selected. The rest of the connection parameters are automatically entered.

Connection Parameters

See page "Parameterized Send Block TSEND_C"

5.4.4. Parameterized Receive Block TRCV_C



TRCV_C

The TRCV_C block processes send tasks as follows:

1. Establish connection to the communication partner.
2. Receive data when EN_R = TRUE. When receiving data, the parameter CONT must have the value TRUE in order to maintain the existing connection.
3. Disconnect connection.

Parameterized Send Block TRCV_C

In the following, the differences to the "TSEND_C" block are presented.

- EN_R

  The receiving of data is enabled (EN_R = TRUE) via the parameter EN_R.

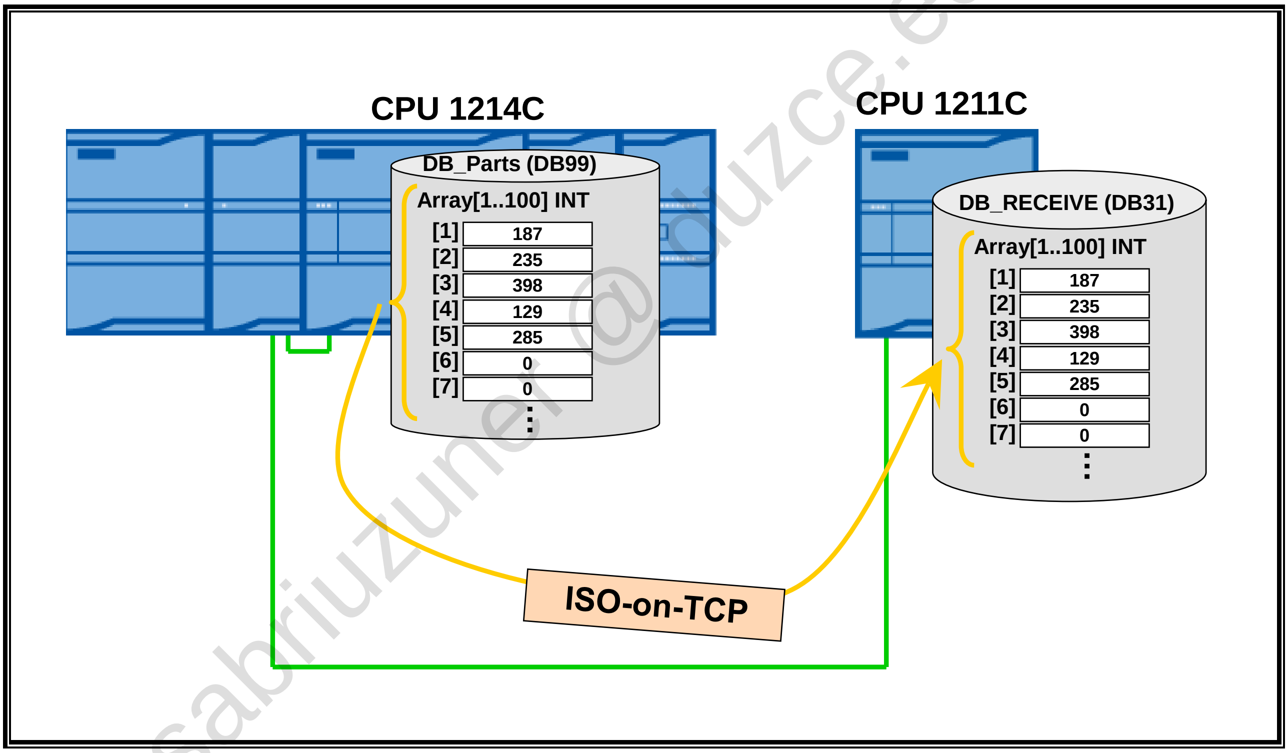
- DATA

  At the parameter DATA, the data range is specified in which the received data is to be stored.

- RCVD_LEN

  This parameter outputs the number of actually received bytes after successful receipt.

5.5. Task Description: Program CPU-CPU Communication and Send 200 Bytes of Data



Situation Up Until Now

The individual weight values of the produced, valid parts are stored in the ARRAY variable "DB_Parts".Part_Weight whenever the part has passed through the light barrier.

Task Description

A connection between the CPUs of your training area is to be programmed. Via this connection, the ARRAY variable "DB_Parts".Part_Weight is to be sent from the controller "PLC_1" to the still to be configured controller "PLC_2" and stored there in the DB "DB_RECEIVE" in the ARRAY variable "Receive_buffer".

To minimize data traffic, sending is not to be continuous. Instead, sending is to occur under the same conditions as the saving of weight values in "DB_Parts".

Blocks

- PLC_1

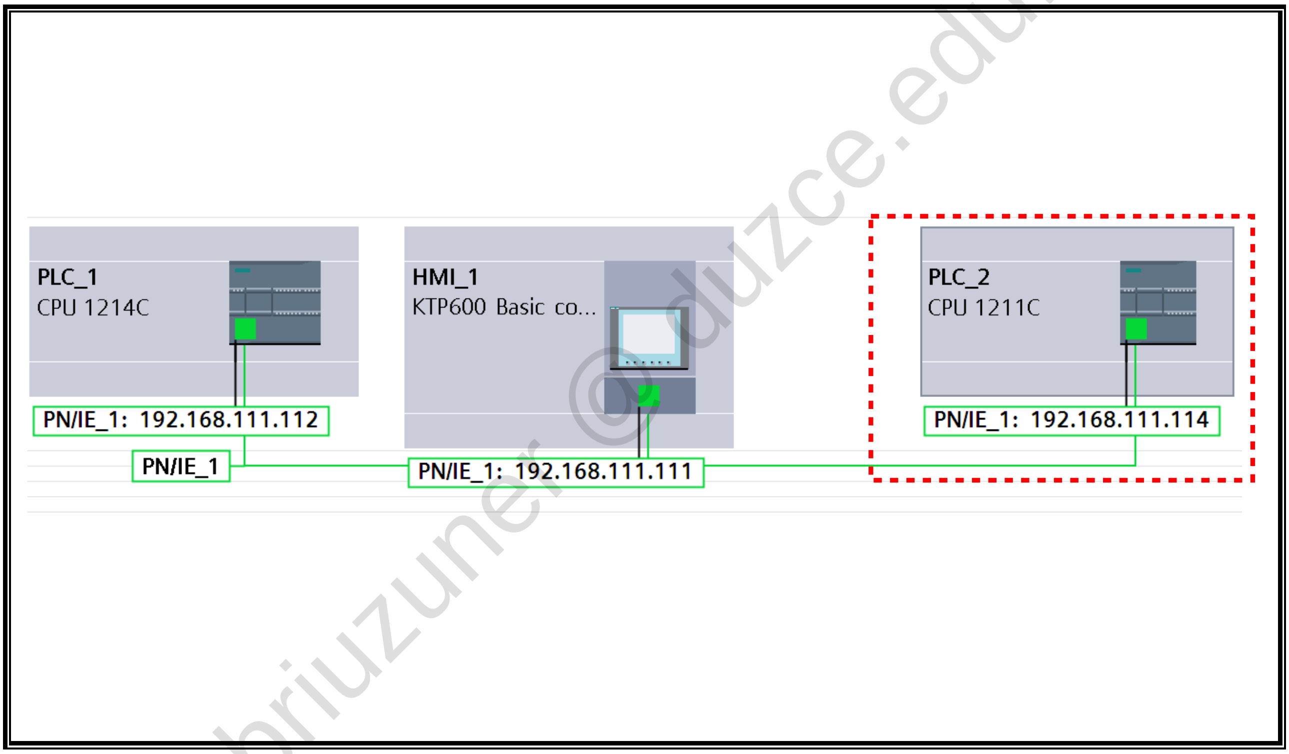
  FC_Send (FC30) → Call in FC_Count (FC18)

- PLC_2
  - FC_Receive (FC31) → Call in MAIN (OB1)
  - DB_RECEIVE (DB31)

Note

The "PLC_2" controller is given the IP address `192.168.111.114`

5.5.1. Exercise 1: Preparing the CPU 1211C

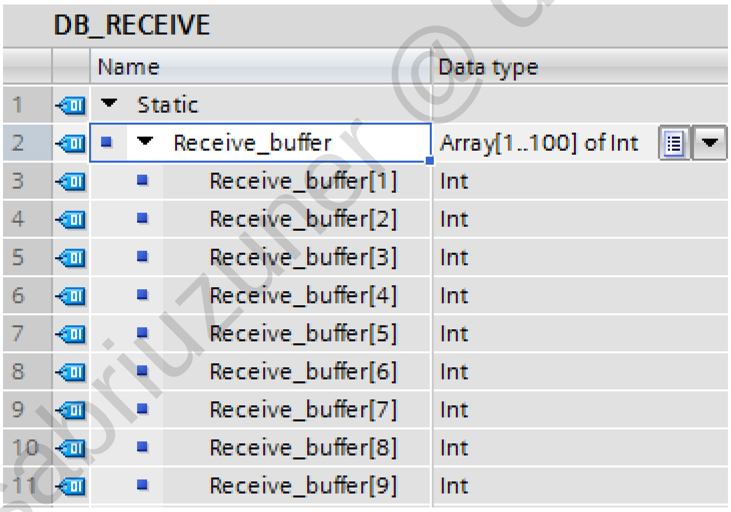


Task

Configure and network the so far unused controller of your training area.

What to Do

1. Reset the controller of your training area that hasn’t been used up until now to the factory settings to establish a defined initial state.
2. Add a CPU of the type S7-1211C to your project and assign the name "PLC_2".
3. In the Network view of your project, network the new station with the rest of the components.
4. Assign "PLC_2" the IP address 192.168.111.114.
5. Assign "PLC_2" the IP address (MB10) in "PLC_2".
6. Generate a new DB with the name "DB_RECEIVE" and in it create the ARRAY variable "Receive_buffer" (ARRAY[1..100] of INT).

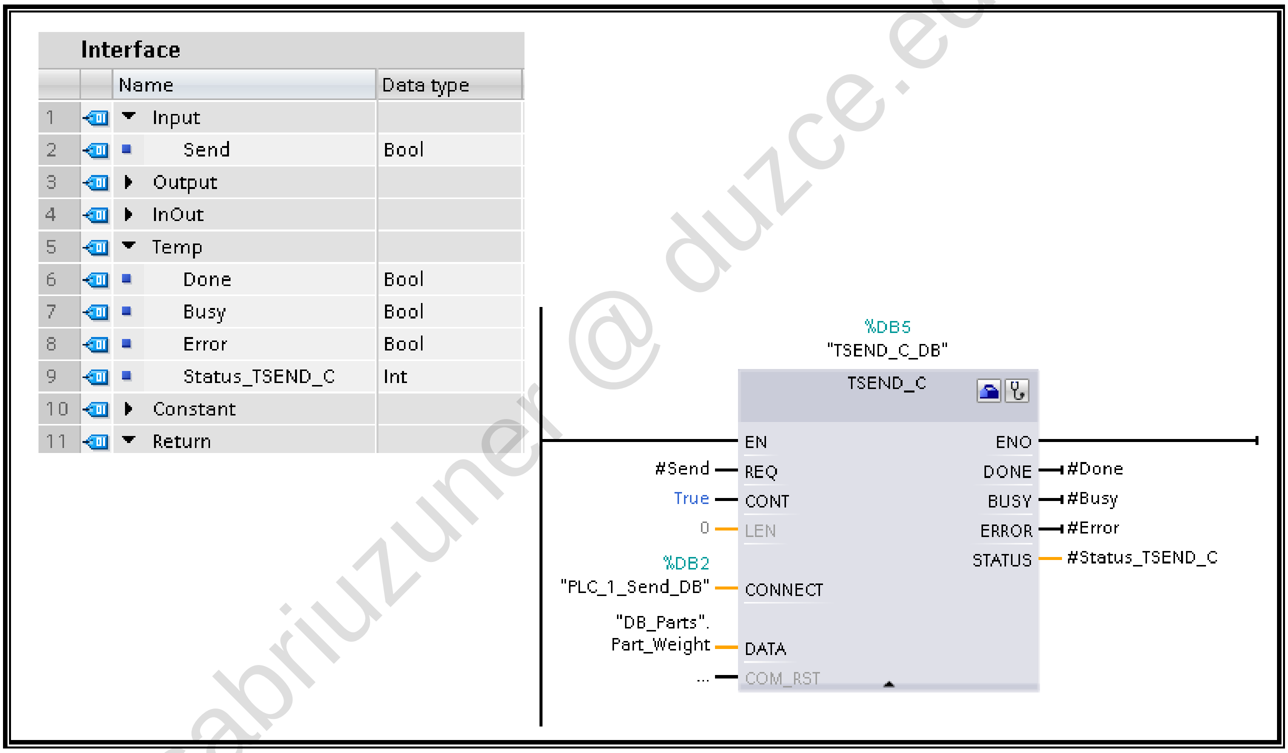


DB_RECEIVE

	Name	Data type
1	Static	
2	Receive_buffer	Array[1..100] of Int
3	Receive_buffer[1]	Int
4	Receive_buffer[2]	Int
5	Receive_buffer[3]	Int
6	Receive_buffer[4]	Int
7	Receive_buffer[5]	Int
8	Receive_buffer[6]	Int
9	Receive_buffer[7]	Int
10	Receive_buffer[8]	Int
11	Receive_buffer[9]	Int

7. Download the modified hardware and software into the controller.

5.5.2. Exercise 2: Calling TSEND_C ("PLC_1": "FC_Send" (FC30))



Interface

	Name	Data type
1	▼ Input	
2	Send	Bool
3	▶ Output	
4	▶ InOut	
5	▼ Temp	
6	Done	Bool
7	Busy	Bool
8	Error	Bool
9	Status_TSEND_C	Int
10	▶ Constant	
11	▼ Return	

%DB5
"TSEND_C_DB"
TSEND_C

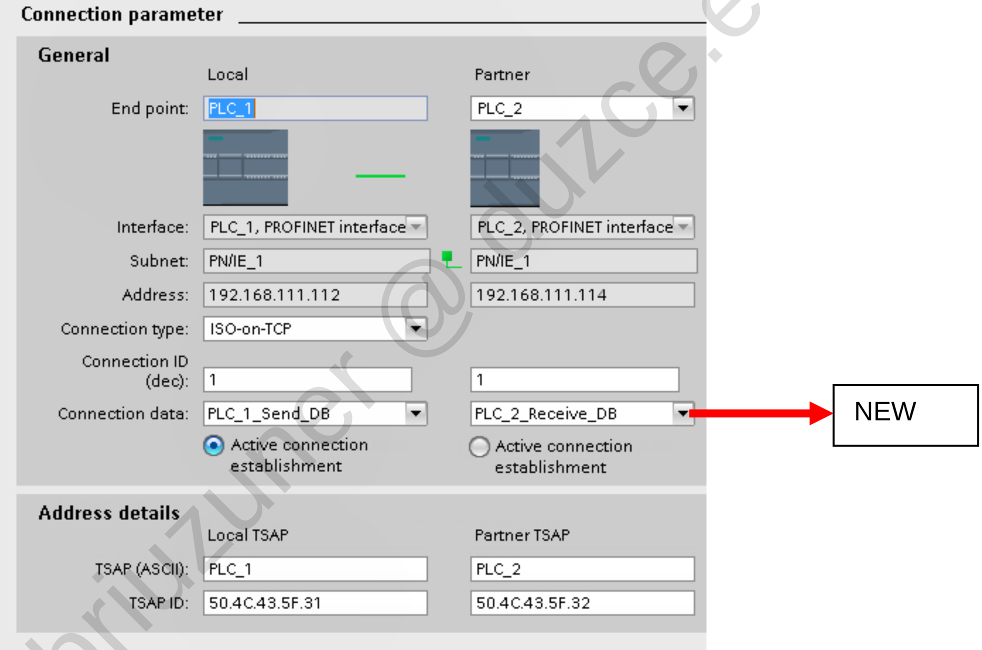
EN — ENO
#Send — REQ
True — CONT
0 — LEN
%DB2 "PLC_1_Send_DB" — CONNECT
"DB_Parts".Part_Weight — DATA
... — COM_RST
DONE — #Done
BUSY — #Busy
ERROR — #Error
STATUS — #Status_TSEND_C

Task

In PLC_1, create a new function "FC_Send" (FC30) and in it program the call of "TSEND_C".

What to Do

1. Add the new function "FC_Send" (FC30) to your user program.
2. Call the block "TSEND_C".
   Instructions → Communication → Open User Comm. → TSEND_C
3. Implement the connection parameter settings as shown in the following:



Connection parameter

General

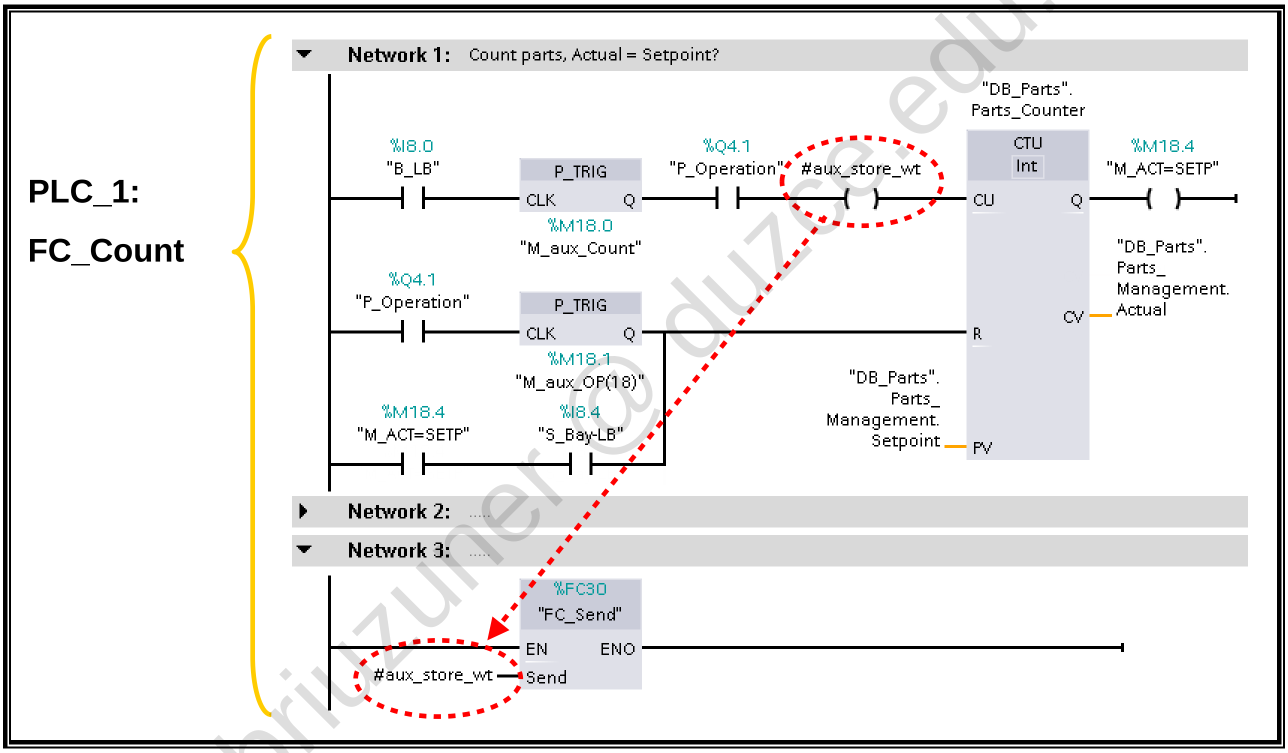
	Local	Partner
End point:	PLC_1	PLC_2
Interface:	PLC_1, PROFINET interface	PLC_2, PROFINET interface
Subnet:	PN/IE_1	PN/IE_1
Address:	192.168.111.112	192.168.111.114
Connection type:	ISO-on-TCP	
Connection ID (dec):	1	1
Connection data:	PLC_1_Send_DB	PLC_2_Receive_DB
	● Active connection establishment	○ Active connection establishment

Address details

	Local TSAP	Partner TSAP
TSAP (ASCII):	PLC_1	PLC_2
TSAP ID:	50.4C.43.5F.31	50.4C.43.5F.32

4. Parameterize the block call as shown in the picture above.

5.5.3. Exercise 3: Calling "FC_Send"



Task

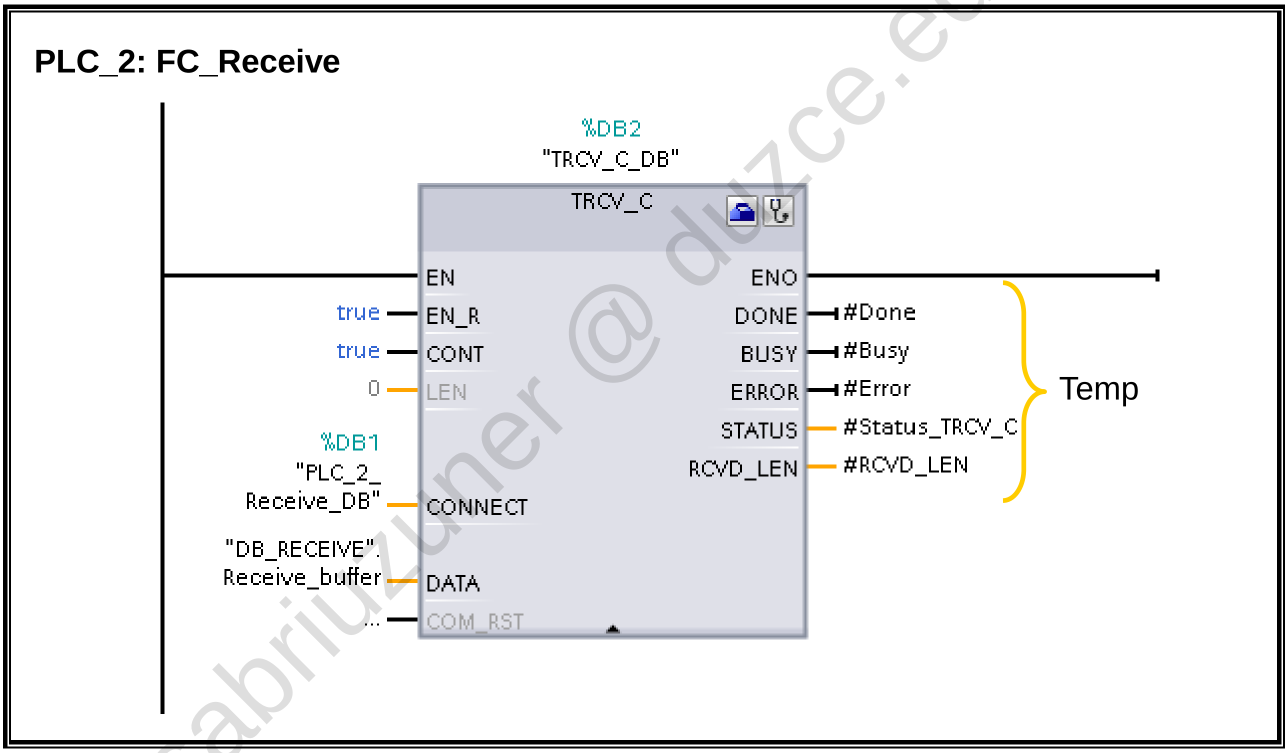
Call the new block "FC_Send" (FC30) in "FC_Count" (FC18) and assign the parameter "Send" the same RLO as the counter input "CU".

What to Do

1. Open "FC_Count" (FC18) and call "FC_Send".
2. Assign the input "Send" the RLO of the counter input "CU".
3. Download the entire user program into the controller "PLC_1".

☞ In the call of "FC_Send" pay attention to the call sequence within "FC_Count"! In NW2, "FC_Ind_Weight" is called. Only after the current part weight has been stored in DB_Parts, can "Part_Weight" be sent to PLC_2.

5.5.4. Exercise 4: Calling TRCV_C ("PLC_2": "FC_Receive" (FC31))



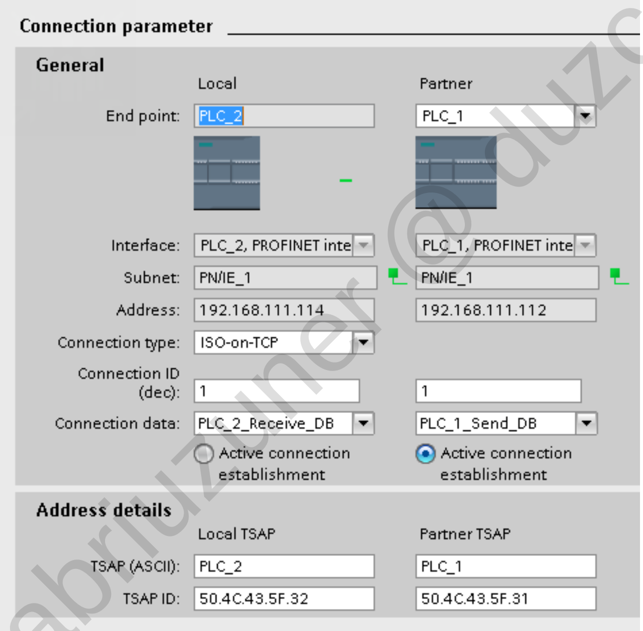
Task

In PLC_2, create a new function "FC_Receive" (FC31) and in it program the call of "TRCV_C". Then call "FC_Receive" in OB1 and transfer the entire user program.

What to Do

1. Add the new function "FC_Receive" (FC31) to your user program.
2. Call the block "TRCV_C".

   Instructions → Communication → Open User Comm. → TRCV_C
3. Implement the connection parameter settings as shown in the following:



Connection parameter

General

	Local	Partner
End point:	PLC_2	PLC_1
Interface:	PLC_2, PROFINET inte	PLC_1, PROFINET inte
Subnet:	PN/IE_1	PN/IE_1
Address:	192.168.111.114	192.168.111.112
Connection type:	ISO-on-TCP	
Connection ID (dec):	1	1
Connection data:	PLC_2_Receive_DB	PLC_1_Send_DB
	○ Active connection establishment	● Active connection establishment

Address details

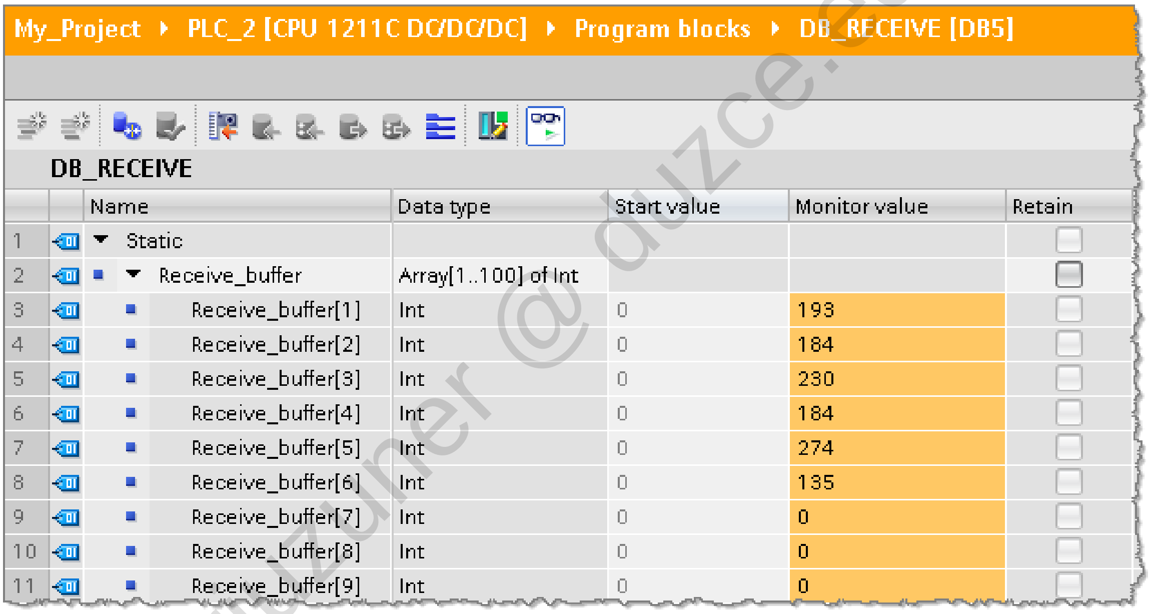
	Local TSAP	Partner TSAP
TSAP (ASCII):	PLC_2	PLC_1
TSAP ID:	50.4C.43.5F.32	50.4C.43.5F.31

4. Parameterize the block call as shown in the picture above.
5. Call "FC_Receive" in OB1.
6. Download the entire user program into PLC_2.

Result

Due to the TRUE signal at EN_R, receiving is continuous. The received data is stored in the DB variable "Receive_buffer".

5.5.5. Exercise 5: Function Test



My_Project ▸ PLC_2 [CPU 1211C DC/DC/DC] ▸ Program blocks ▸ DB_RECEIVE [DB5]

DB_RECEIVE

	Name	Data type	Start value	Monitor value	Retain
1	▼ Static				□
2	▼ Receive_buffer	Array[1..100] of Int			□
3	Receive_buffer[1]	Int	0	193	□
4	Receive_buffer[2]	Int	0	184	□
5	Receive_buffer[3]	Int	0	230	□
6	Receive_buffer[4]	Int	0	184	□
7	Receive_buffer[5]	Int	0	274	□
8	Receive_buffer[6]	Int	0	135	□
9	Receive_buffer[7]	Int	0	0	□
10	Receive_buffer[8]	Int	0	0	□
11	Receive_buffer[9]	Int	0	0	□

Task

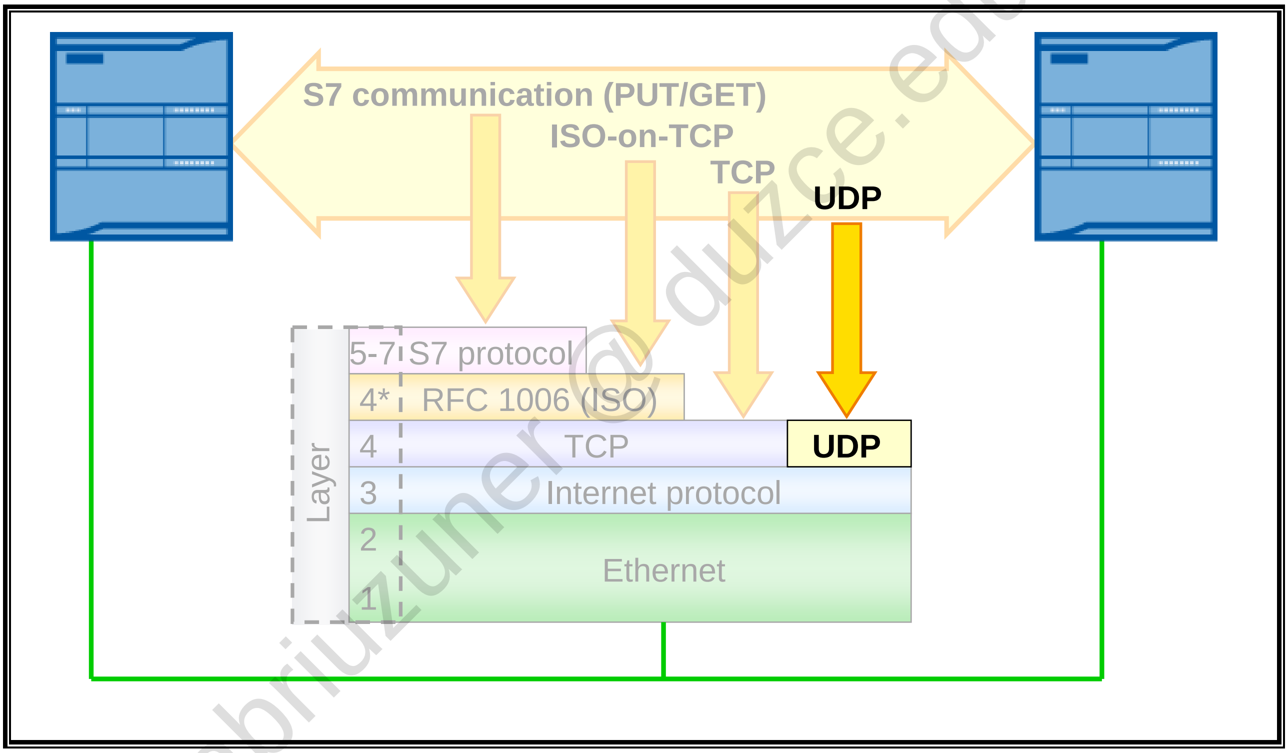
Check whether the communication between PLC_1 and PLC_2 works. For this, monitor "DB_RECEIVE" of PLC_2. First of all, no data is displayed. Only after a part has passed through the light barrier is a send sequence triggered and the 200 bytes in total of data is sent to PLC_2.

5.6. Additional Information

Note

The following pages contain either further information or are for reference to complete a topic.

5.6.1. UDP Communication



User Datagram Protocol

The UDP protocol was introduced to transfer data quickly and straightforward. The UDP protocol is located in Layer 4 (transport layer) of the ISO-OSI reference model and thus is also based on the IP layer. The receiver of data is therefore addressed with the help of IP addresses. The data packet to be sent is only increased by a minimum of administrative information so that a higher data throughput compared to TCP/IP results.

Because of the demand that data be transferred quickly, the UDP protocol merely provides basic functions. Hence, data can be exchanged between communicating partners with a minimum of effort. Security mechanisms as they exist for TCP/IP are thereby dispensed with. The UDP protocol is connectionless and packet-oriented.

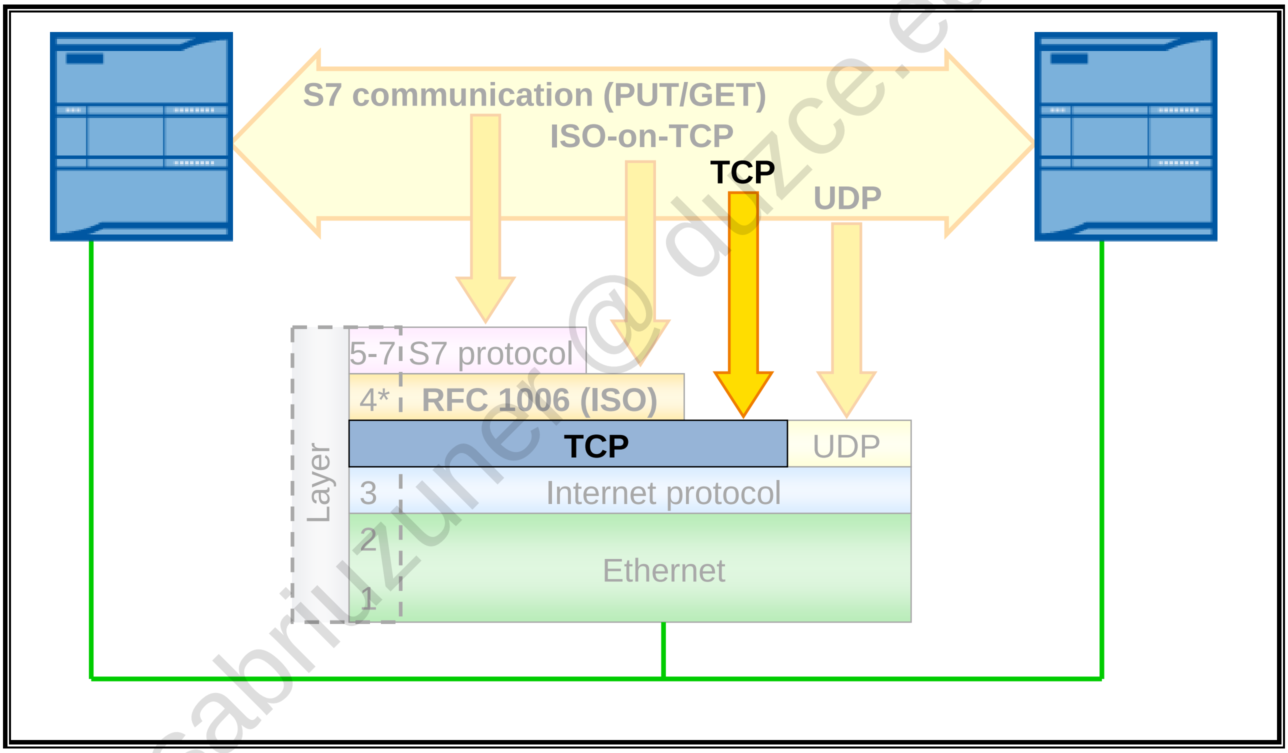
Advantages of UDP

- Very fast data transmission
- Very flexible, ideal for use with third-party systems
- Routing-capable
- Multicast / Broadcast capable
- Suitable for small to medium amounts of data (<= 2048 bytes)

Disadvantages of UDP

- Lost data packets are not resent
- Multiple deliveries of individual packets are possible
- The arrival sequence of packets at the receiver cannot be predicted

5.6.2. TCP Communication



Transmission Control Protocol

When data is transferred with TCP, the transmission takes place in the form of a data flow. Neither information on length nor information about the beginning and end of a message is transferred. The receiver, however, cannot recognize where a message ends in the data flow and where the next one begins in the data flow. For that reason, the sender must define a message structure which can be interpreted by the receiver. The message structure can be composed of, for example, the data and a concluding control character such as "carriage return", which signals the end of a message.

In most cases, TCP is based on the IP (Internet protocol) and so you also talk about the "TCP/IP protocol". It is established in Layer 4 of the ISO-OSI reference model.

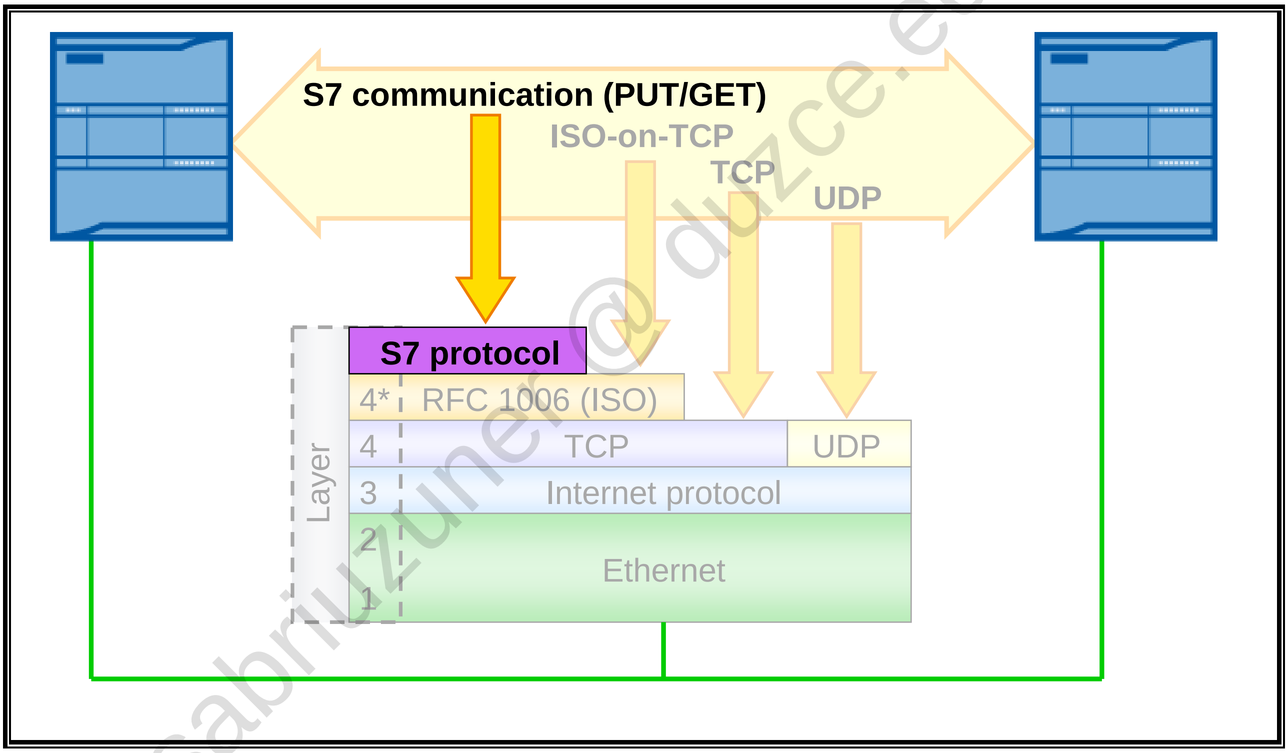
Advantages of the TCP Protocol:

- Fast communication
- Suitable for the transmission of medium to large amounts of data (<= 8192 bytes)
- Routing-capable (i.e. can be used in WAN)
- Flexible, can be used with third-party systems
- Acknowledged

Disadvantages of the TCP Protocol:

- Only static data lengths can be transmitted
- Greater programming effort required to manage data
- Data is transmitted as data flow

5.6.3. S7 Communication



S7 Communication

The S7 protocol is supported by all available S7 controllers and communications processors. As well, PC systems with the appropriate hardware and software equipment support communication via the S7 protocol. The S7-400 controllers use SFBs, the S7-300 and 1200 controllers use FBs. These functions are available regardless of the bus system used so that you can use the S7 communication via Industrial Ethernet, PROFIBUS or MPI.

Advantages of the S7 Protocol:

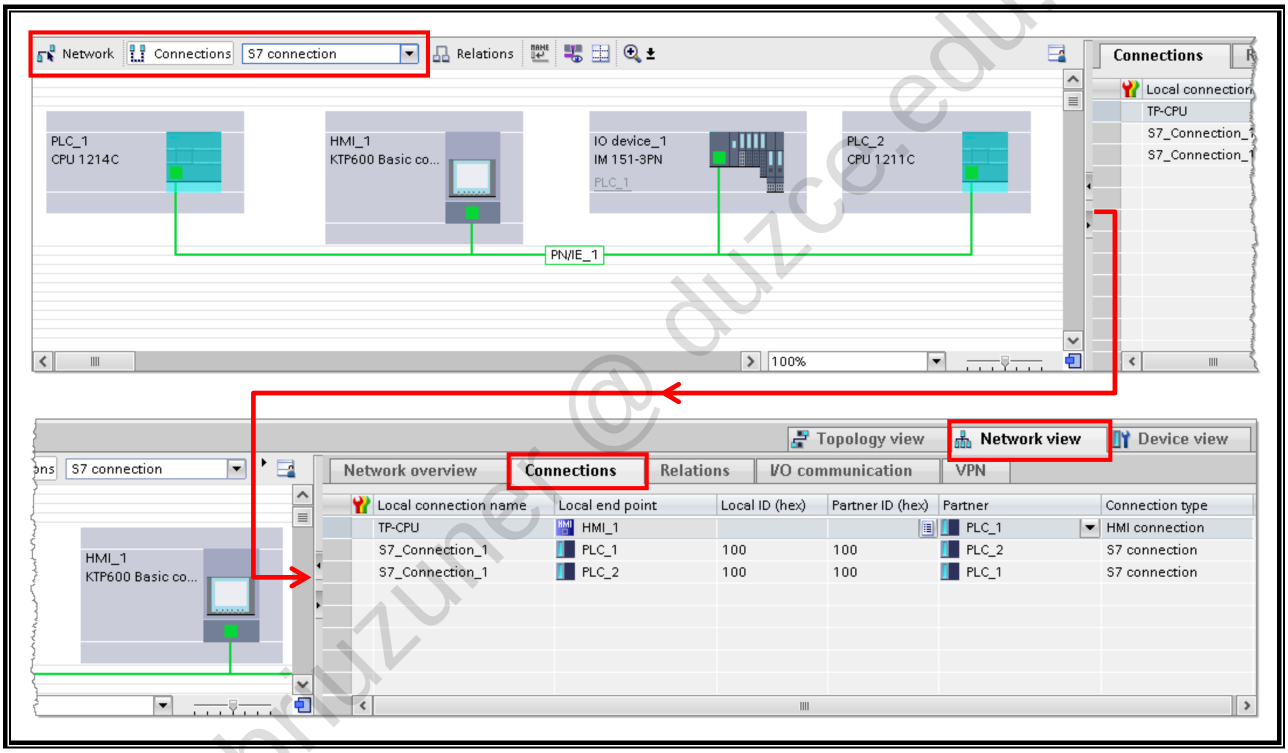
- Regardless of the bus medium (PROFIBUS, Industrial Ethernet (ISO o. TCP), MPI)
- Applies to all S7 data areas
- Transmission of up to 64KByte in one task
- Layer 7 protocol independently ensures for acknowledgement of the data records
- Low processor and bus load in the transmission of larger amounts of data since it is optimized for SIMATIC communication

Disadvantages of the S7 Protocol

Manufacturer-dependent, the S7 protocol is only implemented in the SIMATIC S7 spectrum.

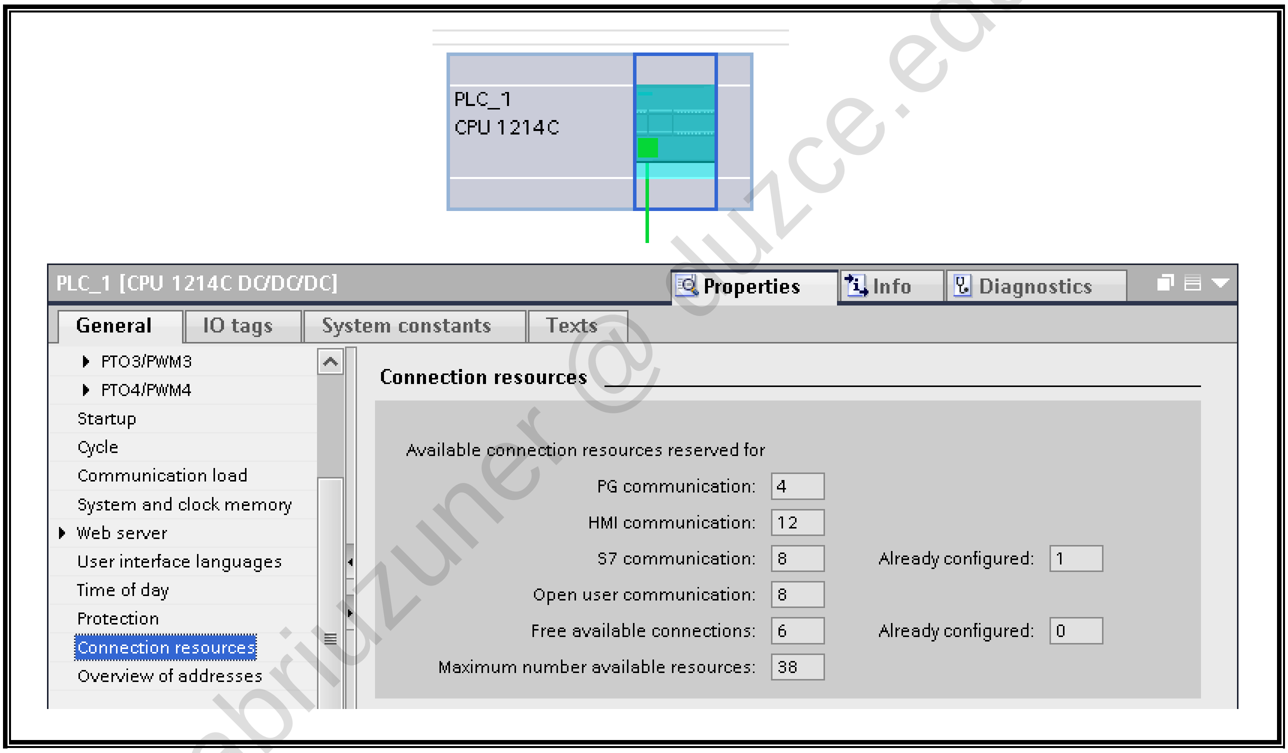
Not compatible with S5 communication.

5.7. Connections



Local connection name	Local end point	Local ID (hex)	Partner ID (hex)	Partner	Connection type
TP-CPU	HMI_1			PLC_1	HMI connection
S7_Connection_1	PLC_1	100	100	PLC_2	S7 connection
S7_Connection_1	PLC_2	100	100	PLC_1	S7 connection

5.7.1. Connection Resources



Connection resources

Available connection resources reserved for

PG communication: 4

HMI communication: 12

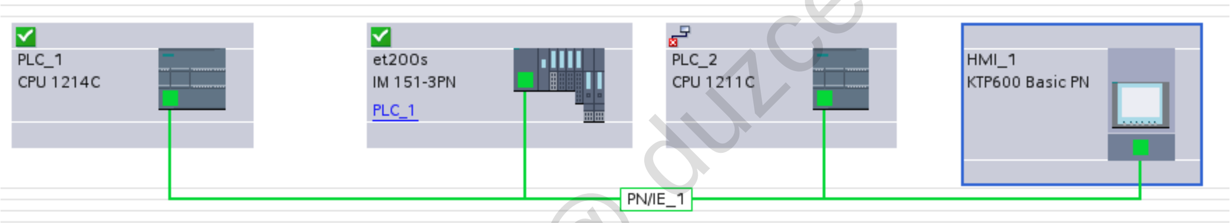
S7 communication: 8 Already configured: 1

Open user communication: 8

Free available connections: 6 Already configured: 0

Maximum number available resources: 38

5.8. Diagnosing the Open User Communication

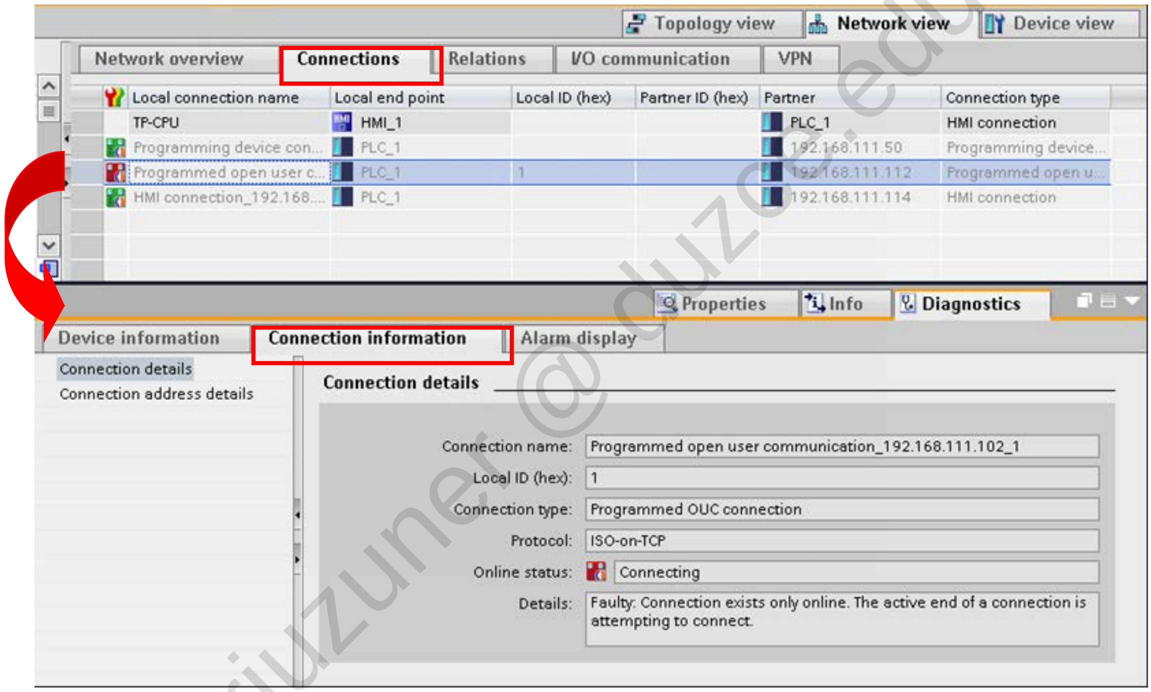


Go online

- **In the connection table you can see the status of the connections**
- **You will find further details in the Inspector window**

Next page

5.8.1. Connection Tables and Connection Information



Local connection name	Local end point	Local ID (hex)	Partner ID (hex)	Partner	Connection type
TP-CPU	HMI_1			PLC_1	HMI connection
Programming device con...	PLC_1			192.168.111.50	Programming device...
Programmed open user c...	PLC_1	1		192.168.111.112	Programmed open u...
HMI connection_192.168....	PLC_1			192.168.111.114	HMI connection

Connection details

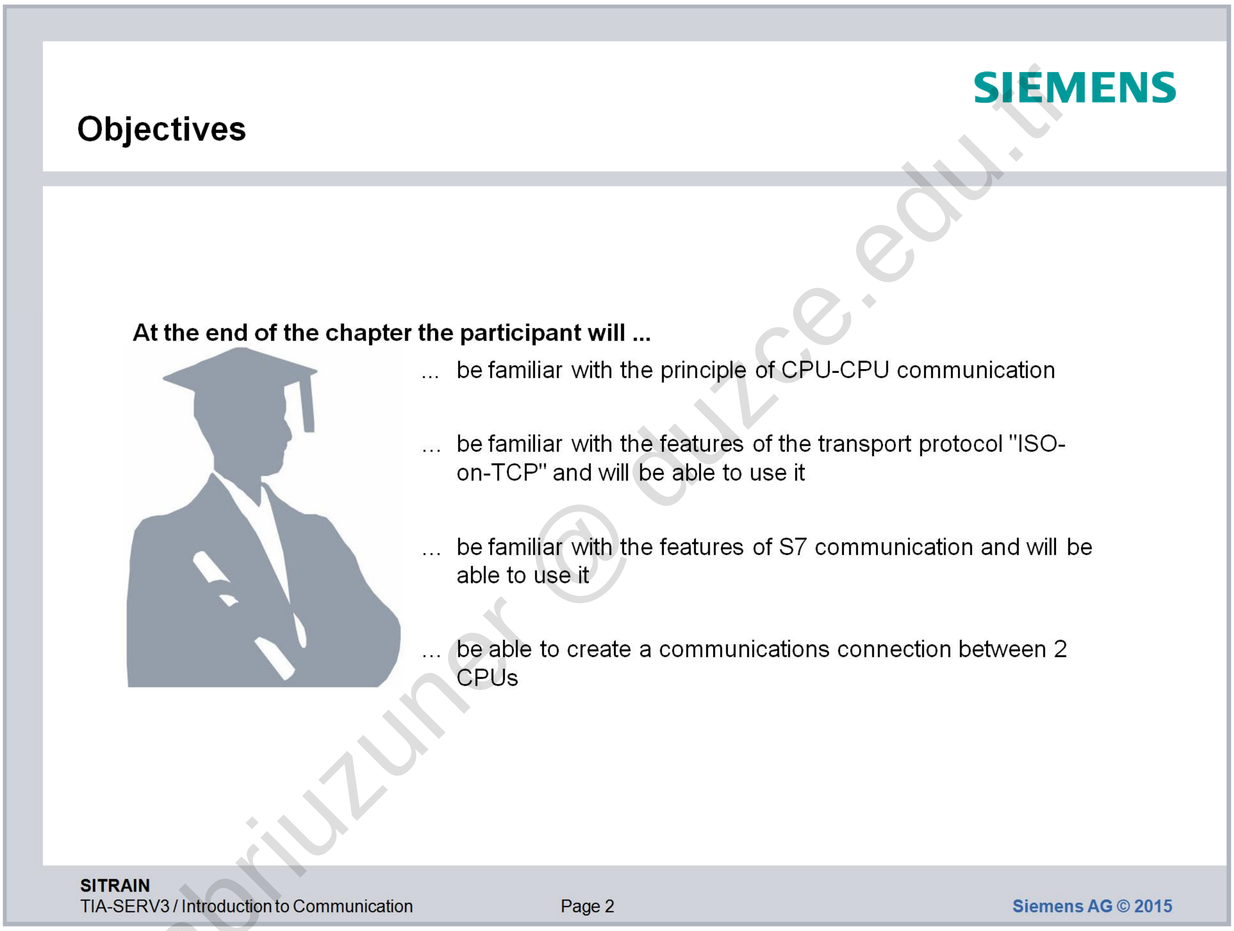
Connection name:	Programmed open user communication_192.168.111.102_1
Local ID (hex):	1
Connection type:	Programmed OUC connection
Protocol:	ISO-on-TCP
Online status:	Connecting
Details:	Faulty: Connection exists only online. The active end of a connection is attempting to connect.

SIEMENS

SITRAIN

Introduction to Communication

Inhalt **Seite**



SIEMENS

Objectives

At the end of the chapter the participant will ...

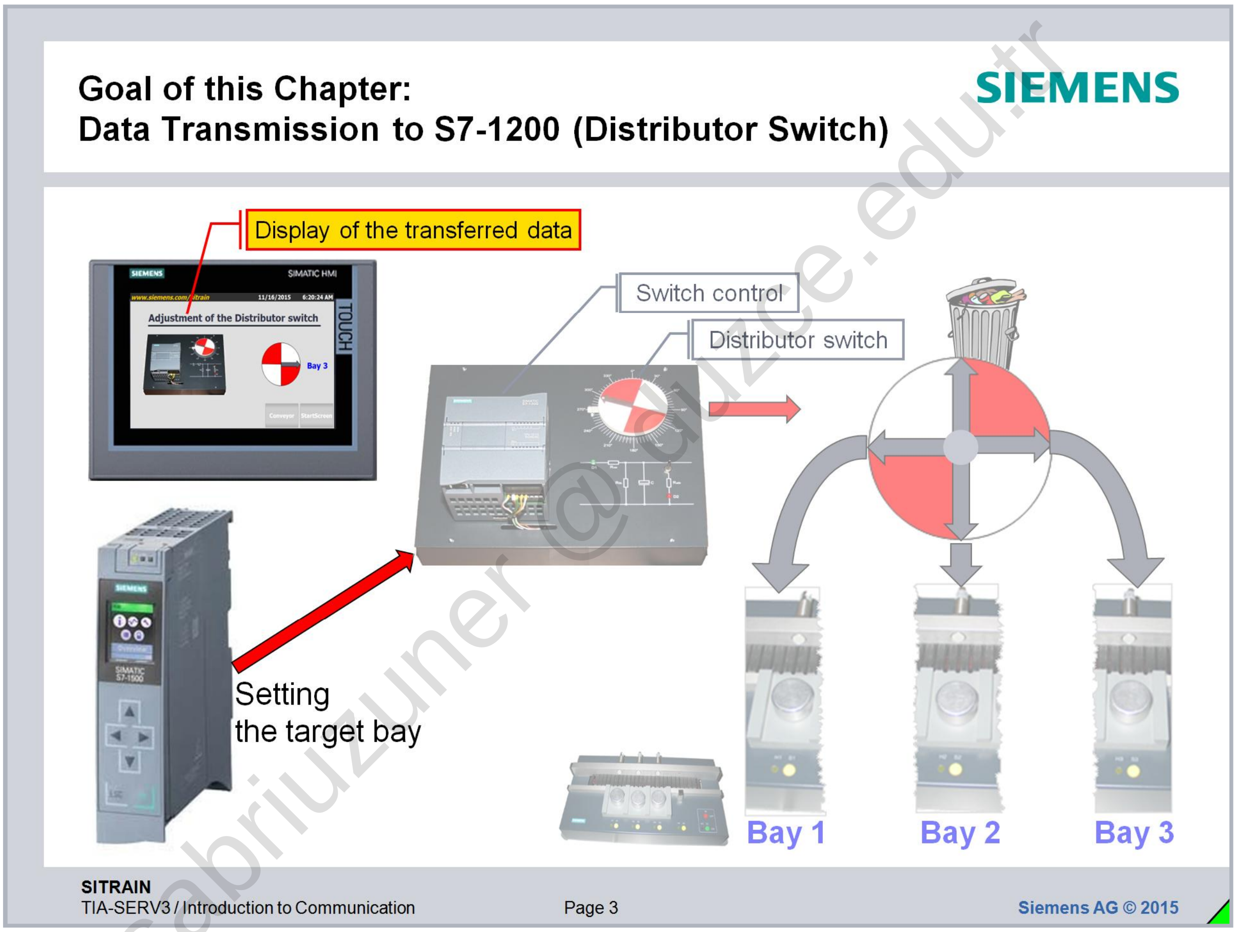
... be familiar with the principle of CPU-CPU communication

... be familiar with the features of the transport protocol "ISO-on-TCP" and will be able to use it

... be familiar with the features of S7 communication and will be able to use it

... be able to create a communications connection between 2 CPUs

SITRAIN
TIA-SERV3 / Introduction to Communication Page 2 Siemens AG © 2015



Goal of this Chapter: Data Transmission to S7-1200 (Distributor Switch)

Display of the transferred data

Switch control

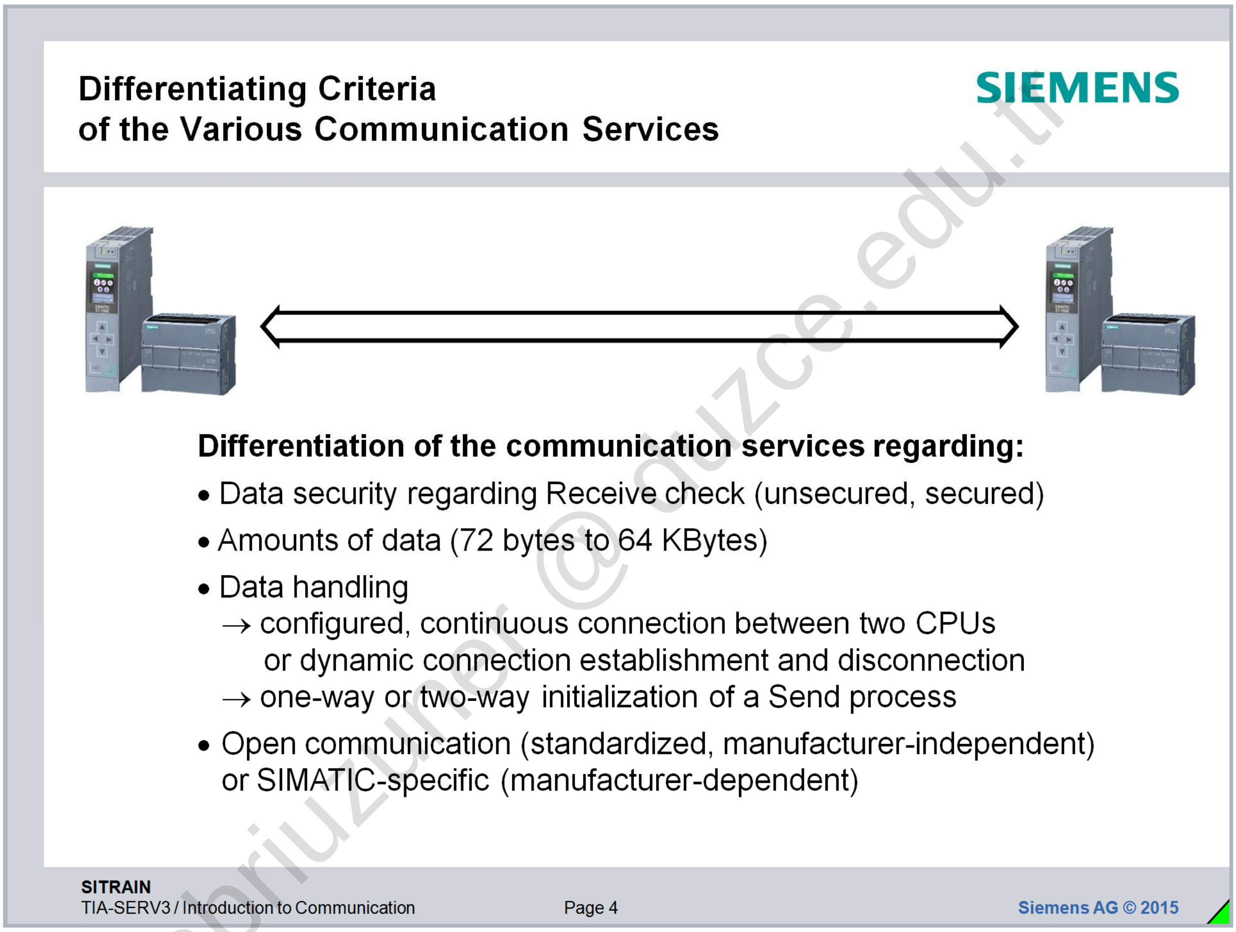
Distributor switch

Setting the target bay

Bay 1

Bay 2

Bay 3



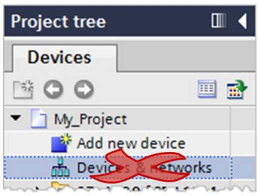
Differentiating Criteria of the Various Communication Services

Differentiation of the communication services regarding:

- Data security regarding Receive check (unsecured, secured)
- Amounts of data (72 bytes to 64 KBytes)
- Data handling
  → configured, continuous connection between two CPUs
  or dynamic connection establishment and disconnection
  → one-way or two-way initialization of a Send process
- Open communication (standardized, manufacturer-independent) or SIMATIC-specific (manufacturer-dependent)

Open Communication - Open User Communication → Instructions

SIEMENS



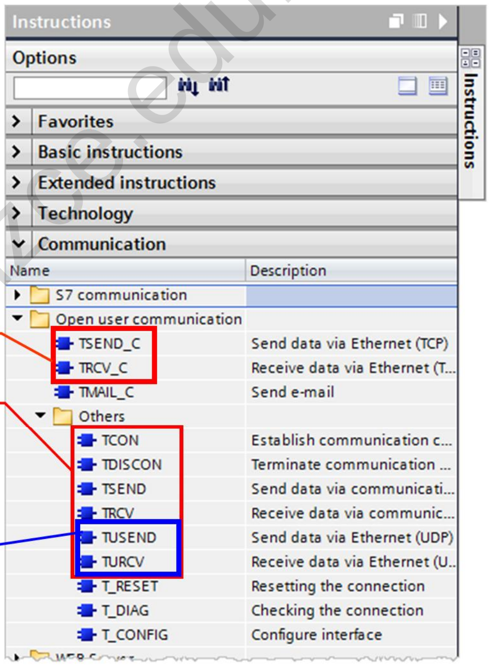
No configured S7 connection required

Combined instructions
(establish connection, disconnect and data transmission)

Separate instructions
(establish connection, disconnect, data transmission)

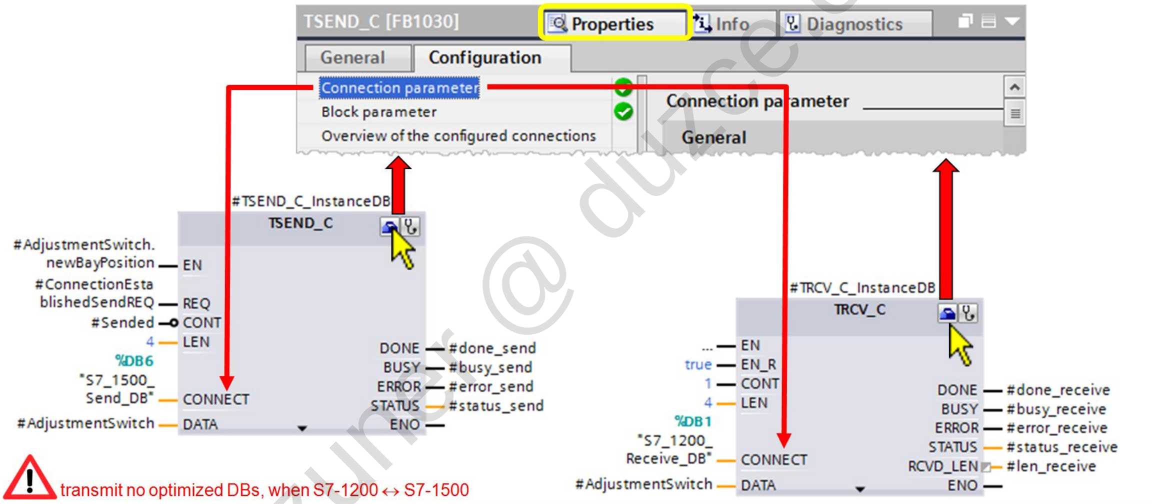
☺ Medium to large amounts of data (max. 8192 bytes)
☺ Secured communication possible
☹ One-way initiated connection establishment

☹ Unsecured data transmission
no repetition in case of transmission loss



Name	Description
S7 communication	
Open user communication	
TSEND_C	Send data via Ethernet (TCP)
TRCV_C	Receive data via Ethernet (T...
TMAIL_C	Send e-mail
Others	
TCON	Establish communication c...
TDISCON	Terminate communication ...
TSEND	Send data via communicati...
TRCV	Receive data via communic...
TUSEND	Send data via Ethernet (UDP)
TURCV	Receive data via Ethernet (U...
T_RESET	Resetting the connection
T_DIAG	Checking the connection
T_CONFIG	Configure interface

Open Communication - Open User Communication → Combined Instructions TSEND_C and TRCV_C



transmit no optimized DBs, when S7-1200 ↔ S7-1500

CONT Communication connection
1 = establish and hold, 0 = disconnect

CONNECT Pointer to the structure for connection description

DATA Pointer to the Send or Receive area

DONE 1 = completed without errors; 0 = no task or still in processing

BUSY 0 = finished or no task; 1 = in processing

LEN max. length of the Send or Receive task
=0 for optimized DBs

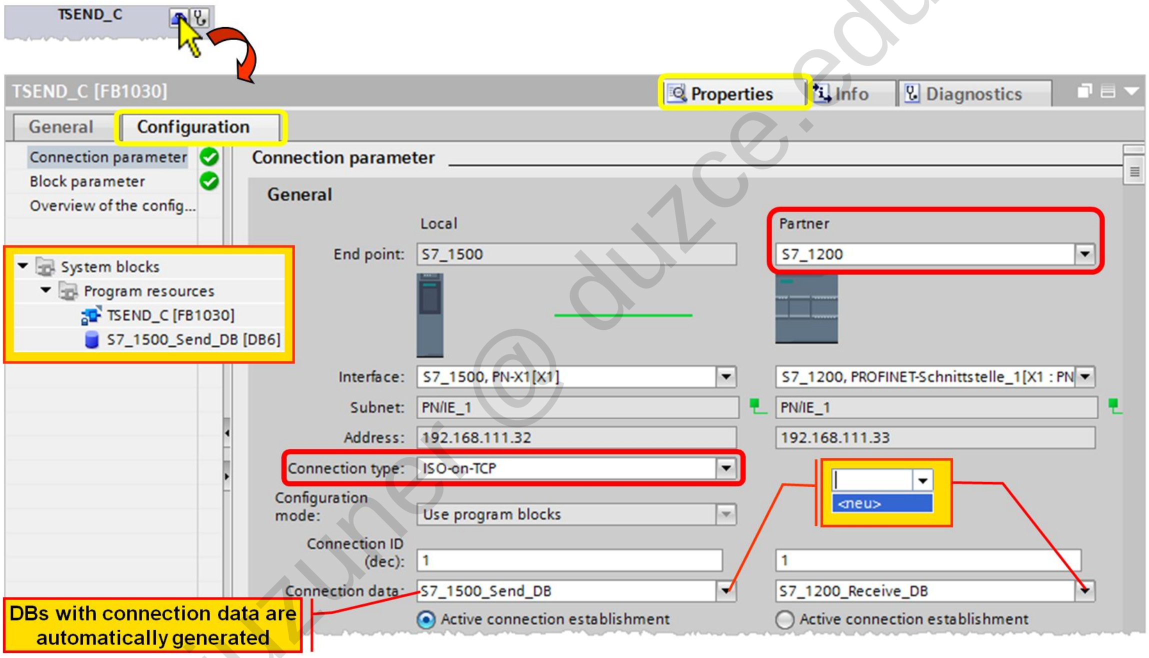
REQ Edge 0→1 starts Send task

EN_R 1 = Receive activated

RCVD_LEN actually received amount of data in bytes

Open Communication - Open User Communication → Connection Data for TSEND_C

SIEMENS



TSEND_C

TSEND_C [FB1030]

Connection parameter

General

	Local	Partner
End point:	S7_1500	S7_1200
Interface:	S7_1500, PN-X1 [X1]	S7_1200, PROFINET-Schnittstelle_1 [X1 : PN
Subnet:	PN/IE_1	PN/IE_1
Address:	192.168.111.32	192.168.111.33
Connection type:	ISO-on-TCP	
Configuration mode:	Use program blocks	
Connection ID (dec):	1	1
Connection data:	S7_1500_Send_DB	S7_1200_Receive_DB
	● Active connection establishment	○ Active connection establishment

DBs with connection data are automatically generated

SITRAIN

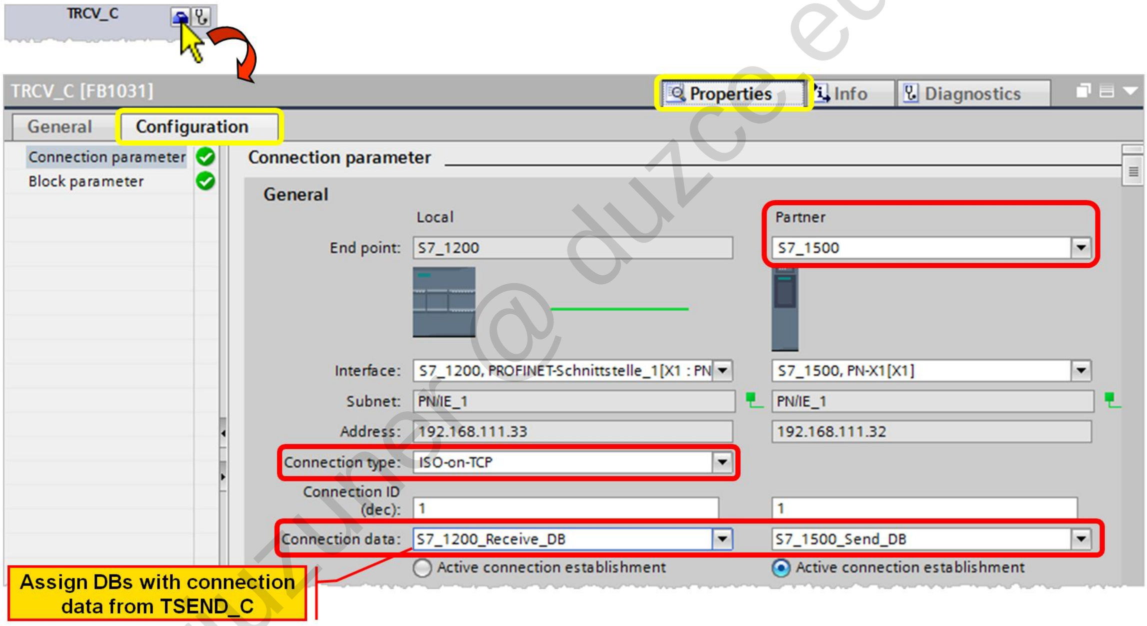
TIA-SERV3 / Introduction to Communication

Page 7

Siemens AG © 2015

Open Communication - Open User Communication → Connection Data for TRCV_C

SIEMENS



Connection parameter

General

	Local	Partner
End point:	S7_1200	S7_1500
Interface:	S7_1200, PROFINET-Schnittstelle_1[X1 : PN	S7_1500, PN-X1[X1]
Subnet:	PN/IE_1	PN/IE_1
Address:	192.168.111.33	192.168.111.32
Connection type:	ISO-on-TCP	
Connection ID (dec):	1	1
Connection data:	S7_1200_Receive_DB	S7_1500_Send_DB
	○ Active connection establishment	● Active connection establishment

Assign DBs with connection data from TSEND_C

SITRAIN
TIA-SERV3 / Introduction to Communication

Siemens AG © 2015

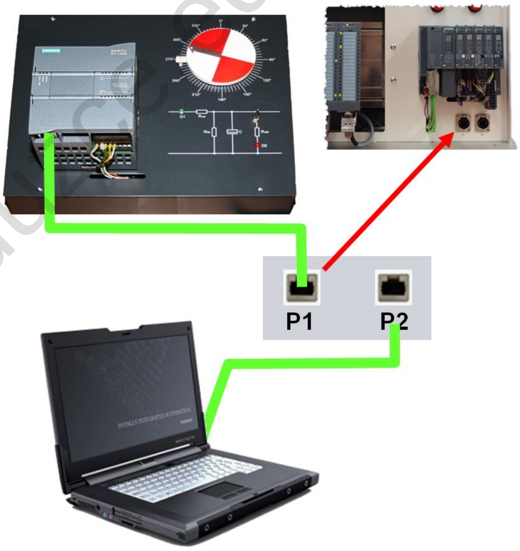
Exercise 1: Connecting the S7-1200 Training Device and Resetting It

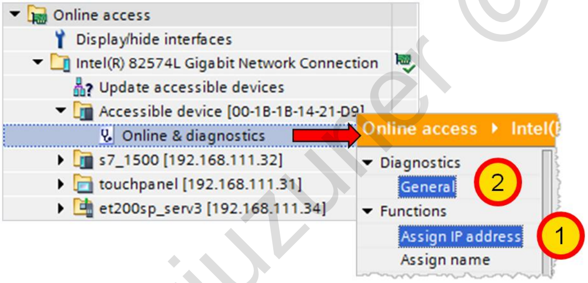
SIEMENS

Task Description:

The second training device with the S7-1200 is to be connected and reset.

- Connect the S7-1200 CPU
- Delete the S7-1200 CPU
  → Reset to factory settings
- Determine the S7-1200 CPU version
  → Assign a temporary IP address
  → Read-out the version online





- Online access
  - Display/hide interfaces
  - Intel(R) 82574L Gigabit Network Connection
    - Update accessible devices
    - Accessible device [00-1B-1B-14-21-D9]
      - Online & diagnostics
    - s7_1500 [192.168.111.32]
    - touchpanel [192.168.111.31]
    - et200sp_serv3 [192.168.111.34]

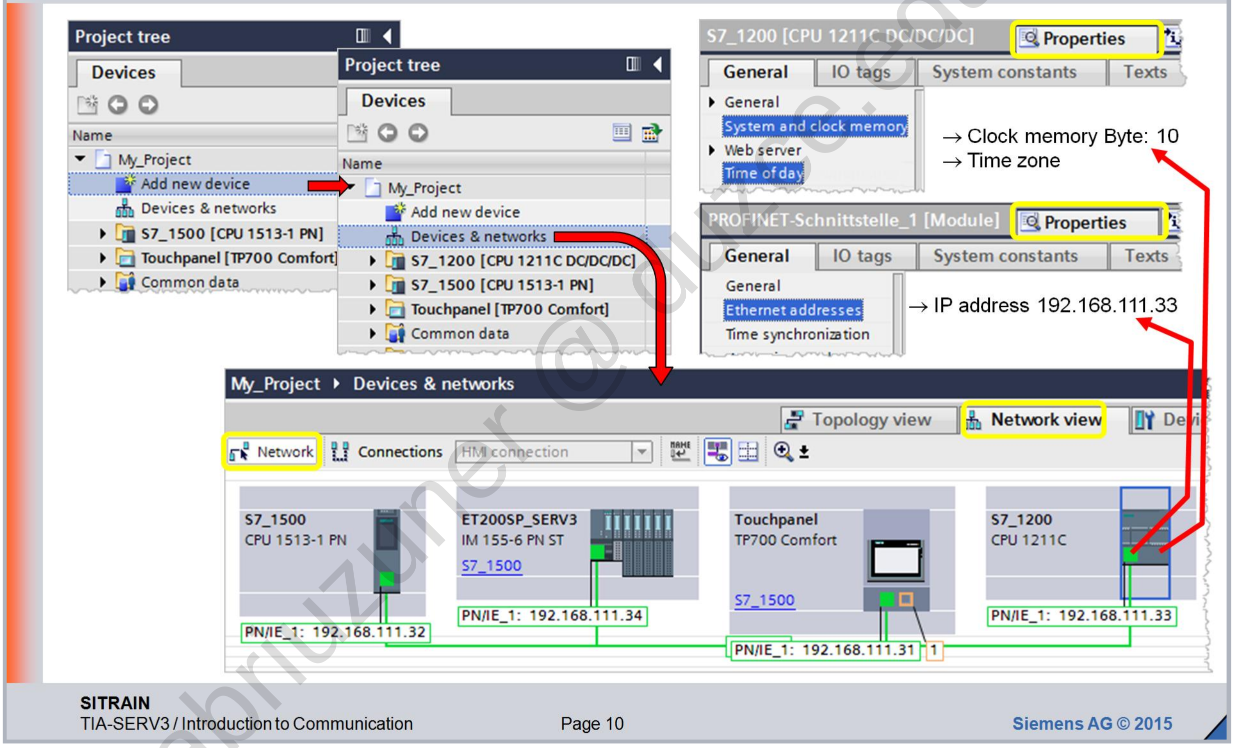
Online access ▸ Intel(

- Diagnostics
  - General 2
- Functions
  - Assign IP address 1
  - Assign name

Aufgabenbeschreibung

Exercise 2: Adding the S7-1200 CPU, Networking It, Assigning Parameters

SIEMENS



→ Clock memory Byte: 10

→ Time zone

→ IP address 192.168.111.33

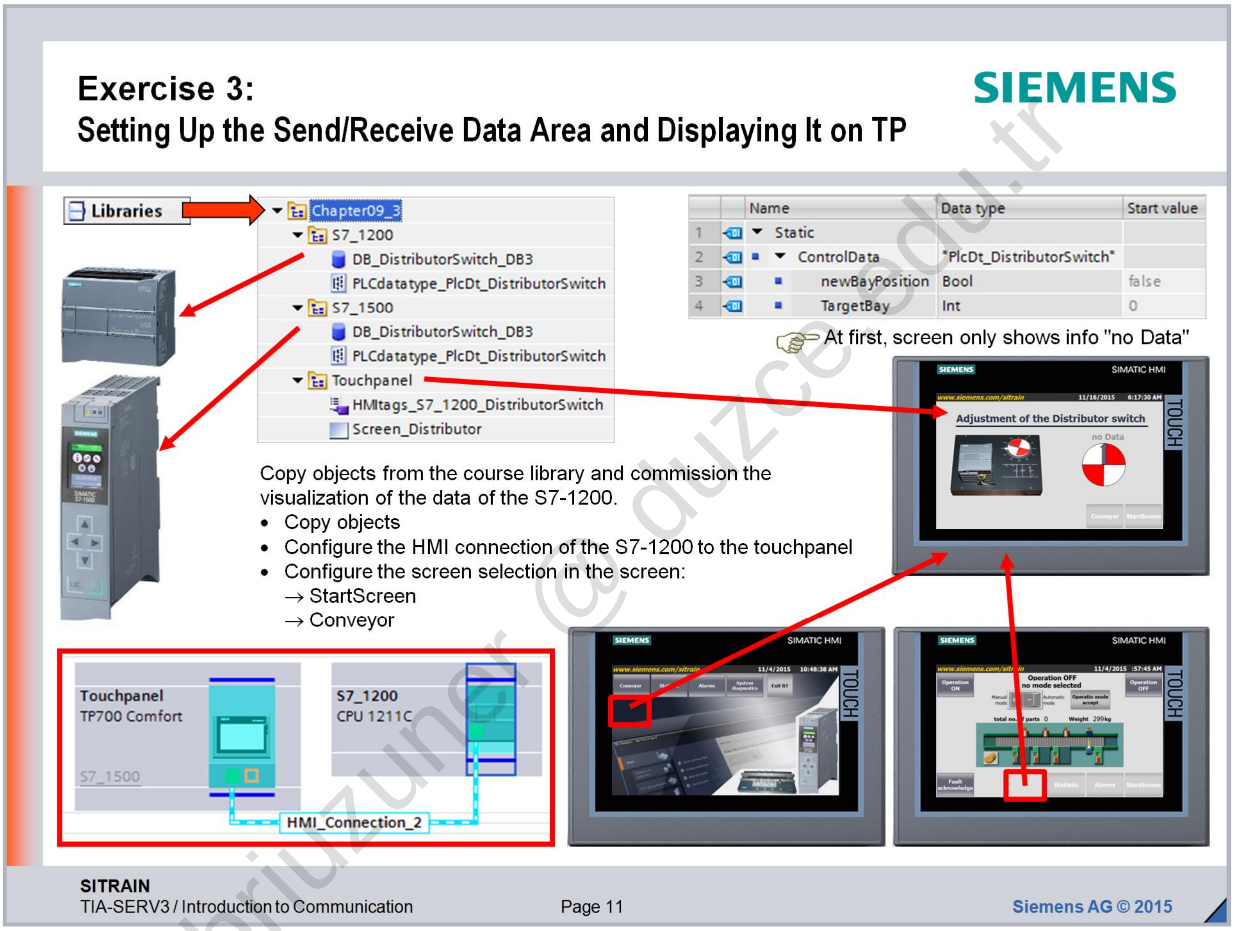
PN/IE_1: 192.168.111.32

PN/IE_1: 192.168.111.34

PN/IE_1: 192.168.111.31

PN/IE_1: 192.168.111.33

Aufgabenbeschreibung



Exercise 3:
Setting Up the Send/Receive Data Area and Displaying It on TP

- Libraries
- Chapter09_3
  - S7_1200
    - DB_DistributorSwitch_DB3
    - PLCdatatype_PlcDt_DistributorSwitch
  - S7_1500
    - DB_DistributorSwitch_DB3
    - PLCdatatype_PlcDt_DistributorSwitch
  - Touchpanel
    - HMItags_S7_1200_DistributorSwitch
    - Screen_Distributor

	Name	Data type	Start value
1	▼ Static		
2	▼ ControlData	"PlcDt_DistributorSwitch"	
3	newBayPosition	Bool	false
4	TargetBay	Int	0

At first, screen only shows info "no Data"

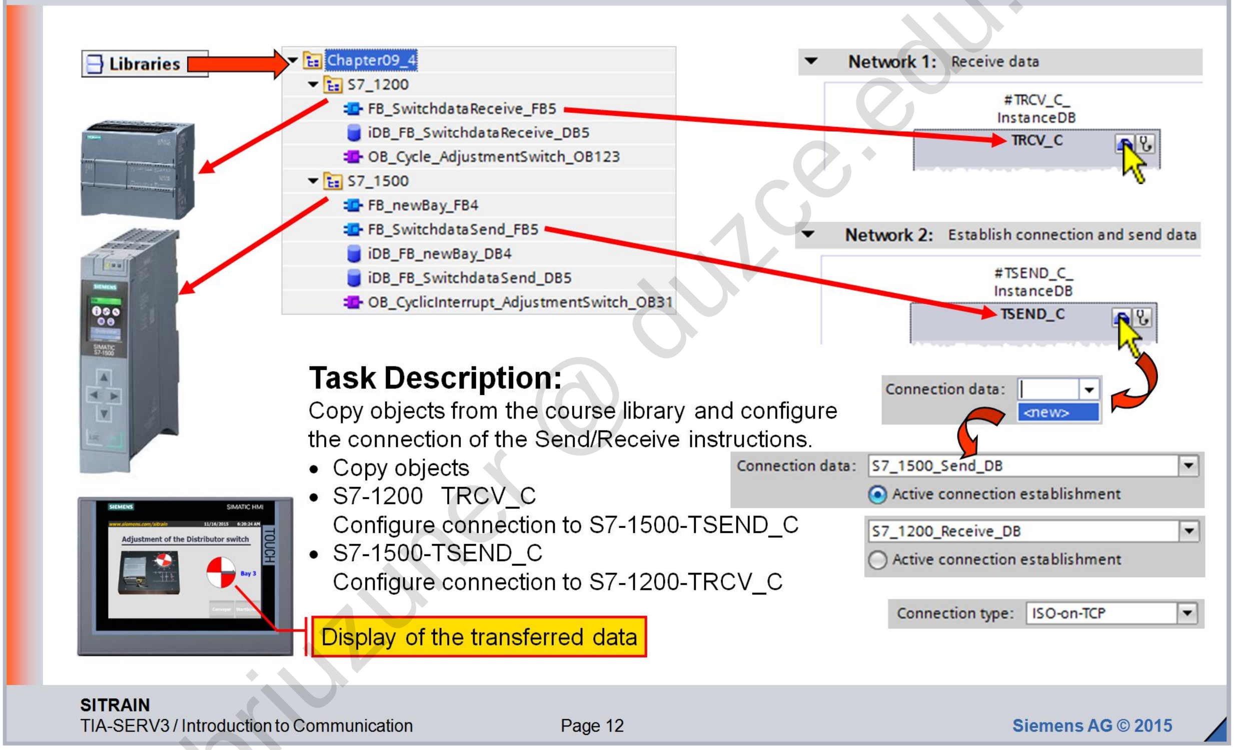
Copy objects from the course library and commission the visualization of the data of the S7-1200.

- Copy objects
- Configure the HMI connection of the S7-1200 to the touchpanel
- Configure the screen selection in the screen:
  → StartScreen
  → Conveyor

SITRAIN
TIA-SERV3 / Introduction to Communication Page 11 Siemens AG © 2015

Aufgabenbeschreibung

Exercise 4:
Sending the 'Adjustment Switch' Data to the S7-1200



Libraries
Chapter09_4
S7_1200
FB_SwitchdataReceive_FB5
iDB_FB_SwitchdataReceive_DB5
OB_Cycle_AdjustmentSwitch_OB123
S7_1500
FB_newBay_FB4
FB_SwitchdataSend_FB5
iDB_FB_newBay_DB4
iDB_FB_SwitchdataSend_DB5
OB_CyclicInterrupt_AdjustmentSwitch_OB31
Network 1: Receive data
#TRCV_C_ InstanceDB
TRCV_C
Network 2: Establish connection and send data
#TSEND_C_ InstanceDB
TSEND_C
Connection data:
<new>
Connection data: S7_1500_Send_DB
● Active connection establishment
S7_1200_Receive_DB
○ Active connection establishment
Connection type: ISO-on-TCP

Task Description:
Copy objects from the course library and configure the connection of the Send/Receive instructions.

- Copy objects
- S7-1200   TRCV_C
  Configure connection to S7-1500-TSEND_C
- S7-1500-TSEND_C
  Configure connection to S7-1200-TRCV_C

Display of the transferred data

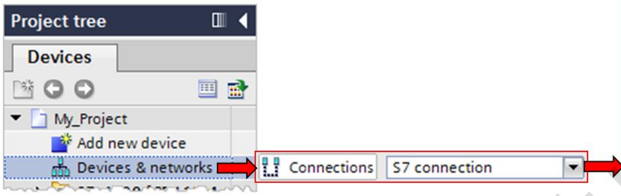
SITRAIN
TIA-SERV3 / Introduction to Communication   Page 12   Siemens AG © 2015

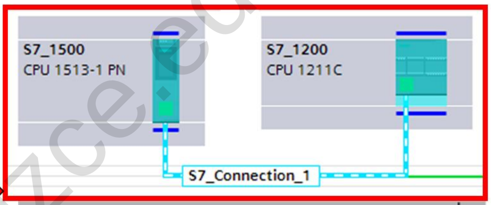
Aufgabenbeschreibung

S7 Communication
→ Instructions

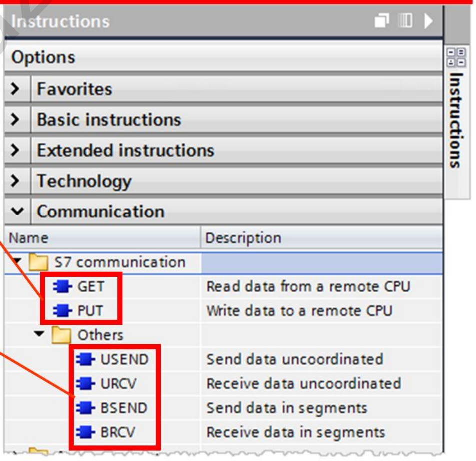
SIEMENS

Configured S7 connection required





☺ Fast communication optimized to SIMATIC
☺ Secured transmission
☹ SIMATIC-specific



Name	Description
S7 communication	
GET	Read data from a remote CPU
PUT	Write data to a remote CPU
Others	
USEND	Send data uncoordinated
URCV	Receive data uncoordinated
BSEND	Send data in segments
BRCV	Receive data in segments

One-way communication services

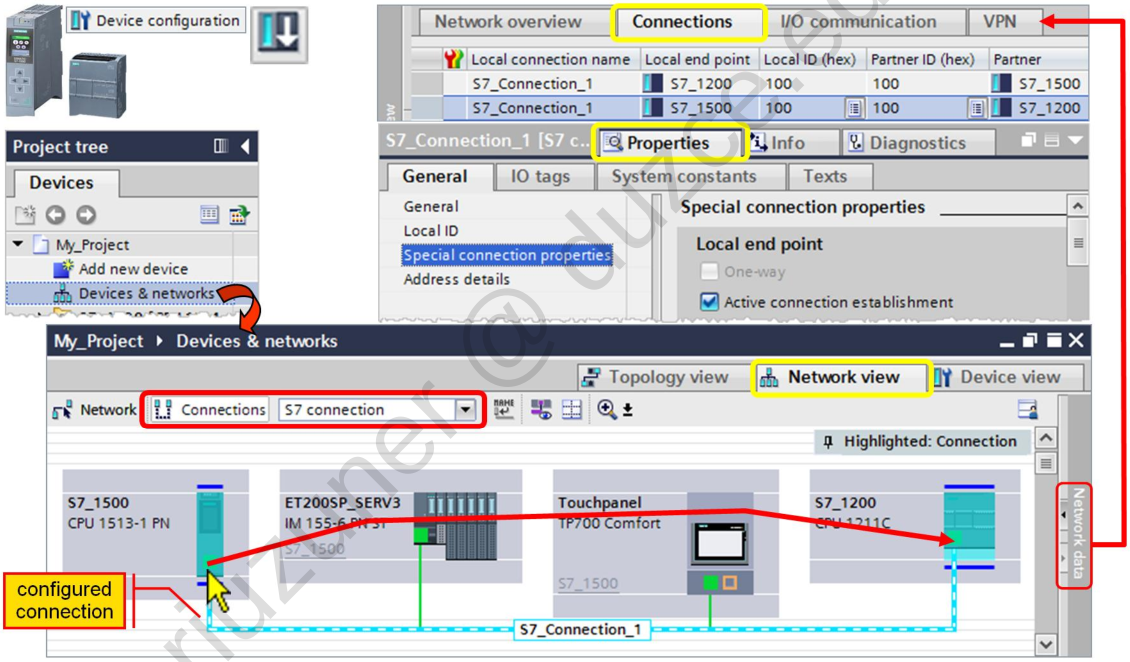
☺ One-way initiated data transmission
☹ Limited amounts of data max. 880Byte (depends on the CPU)

Two-way communication services

☺ Co-ordinated data transmission possible
☺ Large amounts of data max. 64KByte (BSEND/BRCV)
☹ Not supported by S7-1200

S7 Communication
→ Configuring an S7 Connection

SIEMENS



Local connection name	Local end point	Local ID (hex)	Partner ID (hex)	Partner
S7_Connection_1	S7_1200	100	100	S7_1500
S7_Connection_1	S7_1500	100	100	S7_1200

Special connection properties

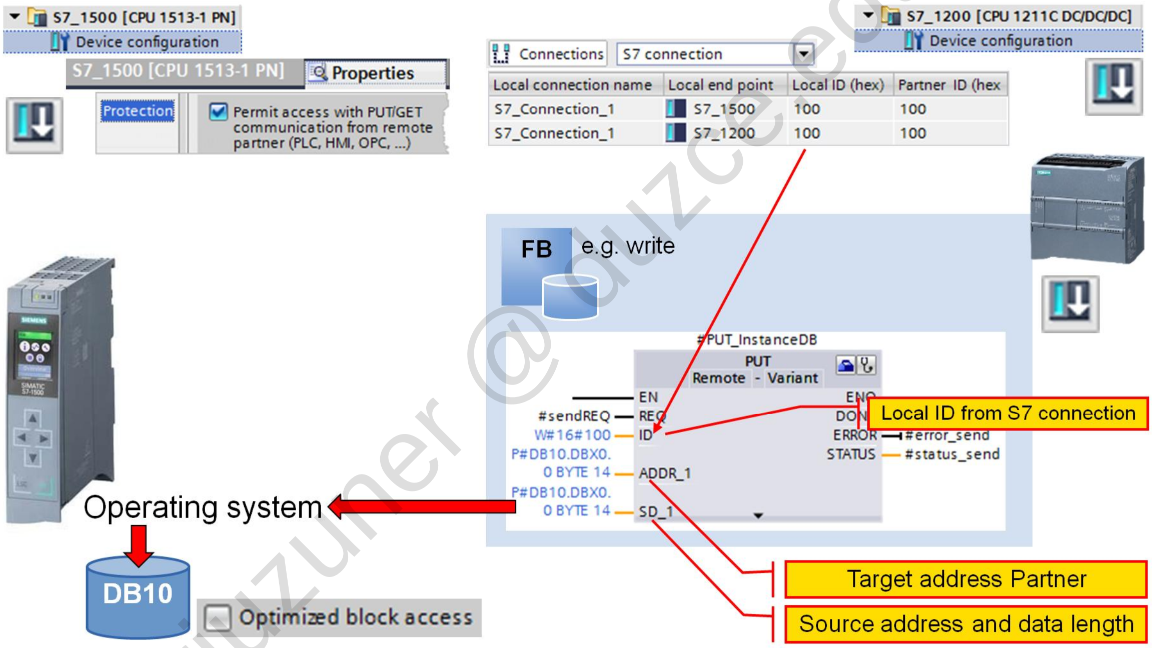
Local end point

□ One-way

☑ Active connection establishment

configured connection

S7 Communication
→ One-way Communication Service PUT/GET



☑ Permit access with PUT/GET communication from remote partner (PLC, HMI, OPC, ...)

Local connection name	Local end point	Local ID (hex)	Partner ID (hex
S7_Connection_1	S7_1500	100	100
S7_Connection_1	S7_1200	100	100

Local ID from S7 connection

Operating system

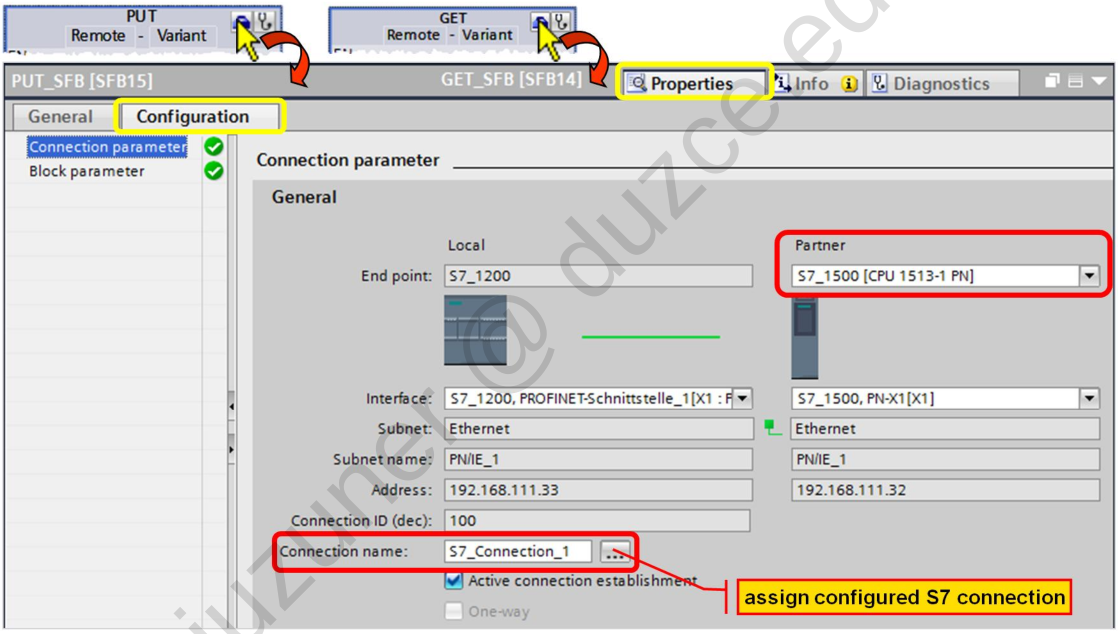
□ Optimized block access

Target address Partner

Source address and data length

SITRAIN
TIA-SERV3 / Introduction to Communication Page 15 Siemens AG © 2015

S7 Communication
→ Connection Data for PUT/GET



Connection parameter

General

	Local	Partner
End point:	S7_1200	S7_1500 [CPU 1513-1 PN]
Interface:	S7_1200, PROFINET-Schnittstelle_1[X1 : P	S7_1500, PN-X1[X1]
Subnet:	Ethernet	Ethernet
Subnet name:	PN/IE_1	PN/IE_1
Address:	192.168.111.33	192.168.111.32
Connection ID (dec):	100	
Connection name:	S7_Connection_1	

☑ Active connection establishment

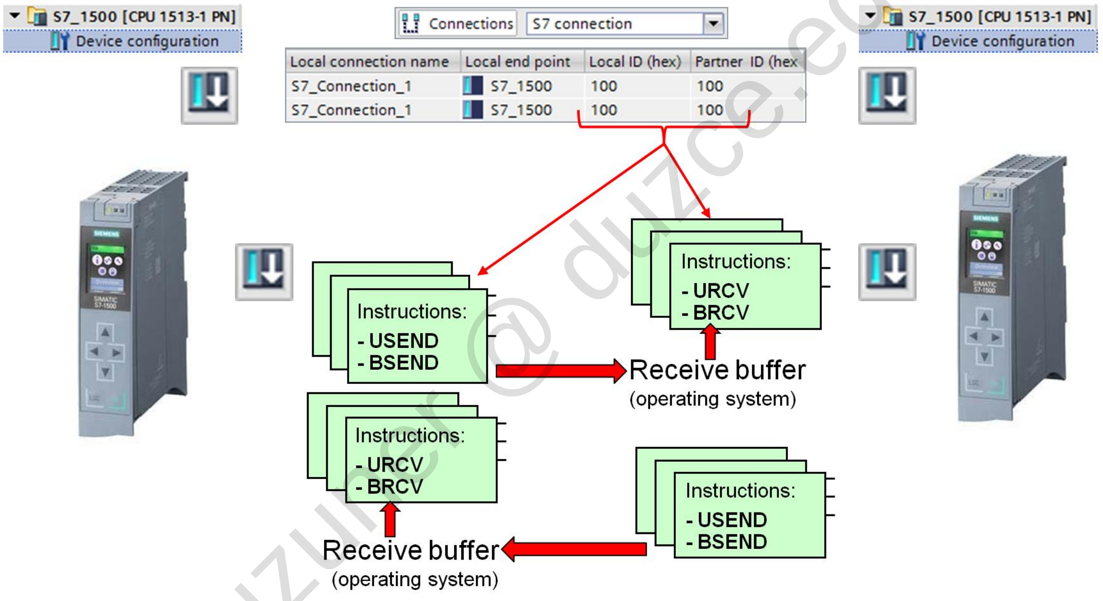
□ One-way

assign configured S7 connection

SITRAIN
TIA-SERV3 / Introduction to Communication
Page 16
Siemens AG © 2015

S7 Communication
→ Two-way Communication Service xSEND/xRCV

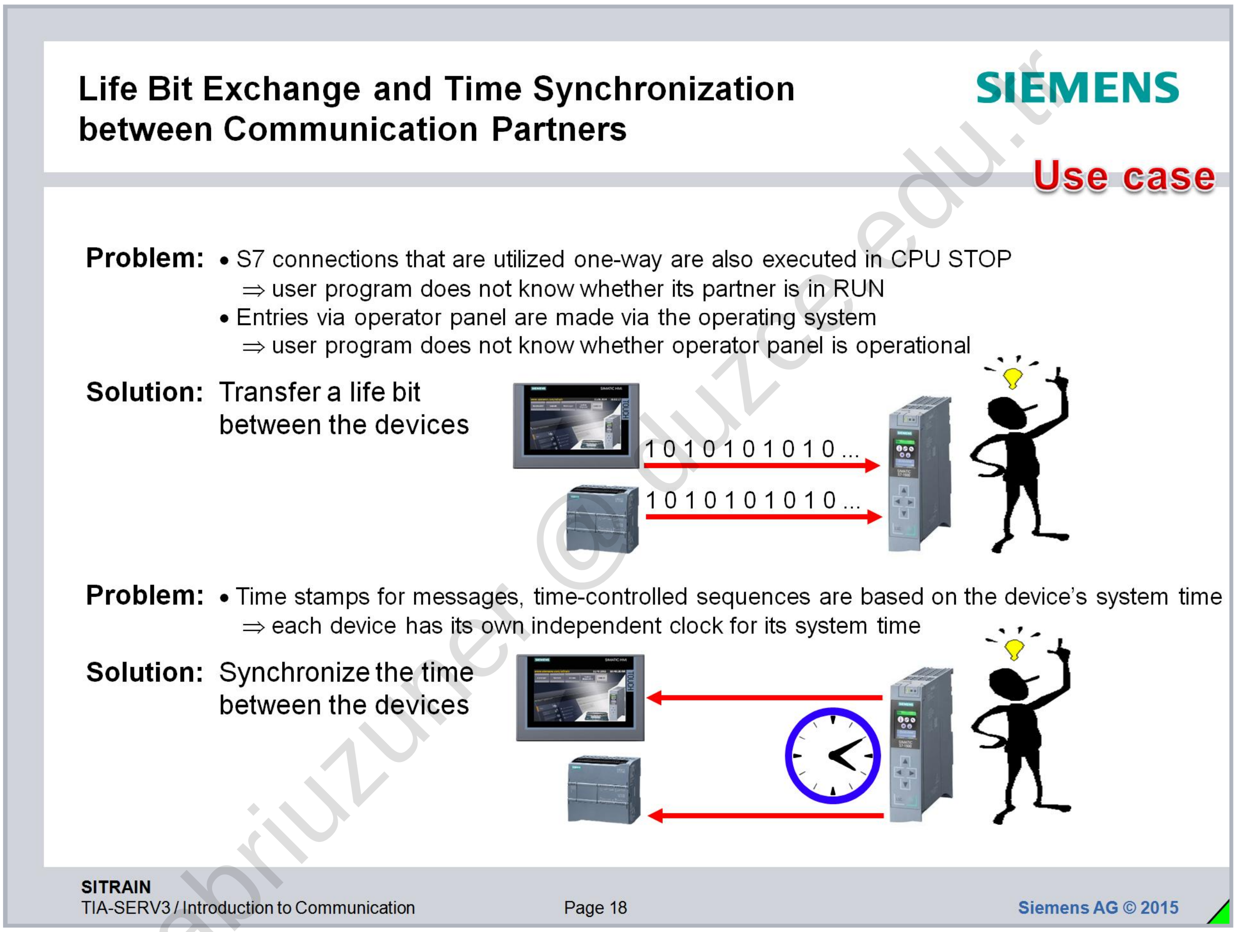
SIEMENS



Local connection name	Local end point	Local ID (hex)	Partner ID (hex
S7_Connection_1	S7_1500	100	100
S7_Connection_1	S7_1500	100	100

BSEND/BRCV – coordinated sending

USEND/URCV – uncoordinated sending



Life Bit Exchange and Time Synchronization between Communication Partners

SIEMENS

Use case

Problem:
- S7 connections that are utilized one-way are also executed in CPU STOP
  ⇒ user program does not know whether its partner is in RUN
- Entries via operator panel are made via the operating system
  ⇒ user program does not know whether operator panel is operational

Solution: Transfer a life bit between the devices

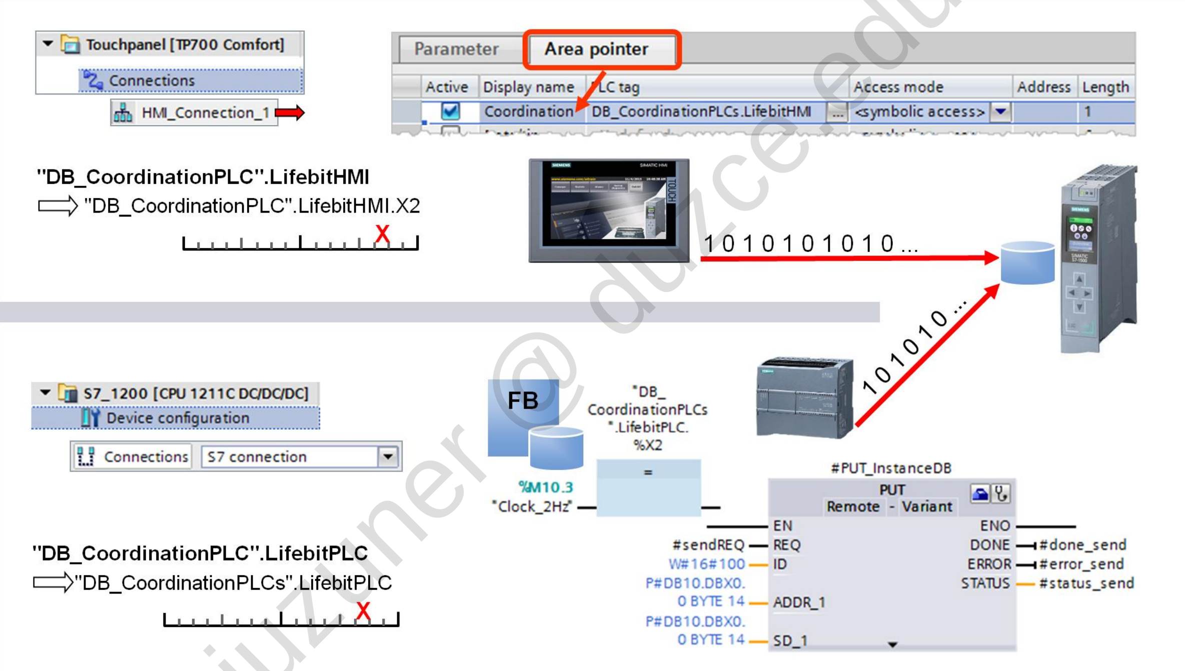
Problem:
- Time stamps for messages, time-controlled sequences are based on the device's system time
  ⇒ each device has its own independent clock for its system time

Solution: Synchronize the time between the devices

SITRAIN
TIA-SERV3 / Introduction to Communication Page 18 Siemens AG © 2015

Life Bit Exchange between Communication Partners

SIEMENS

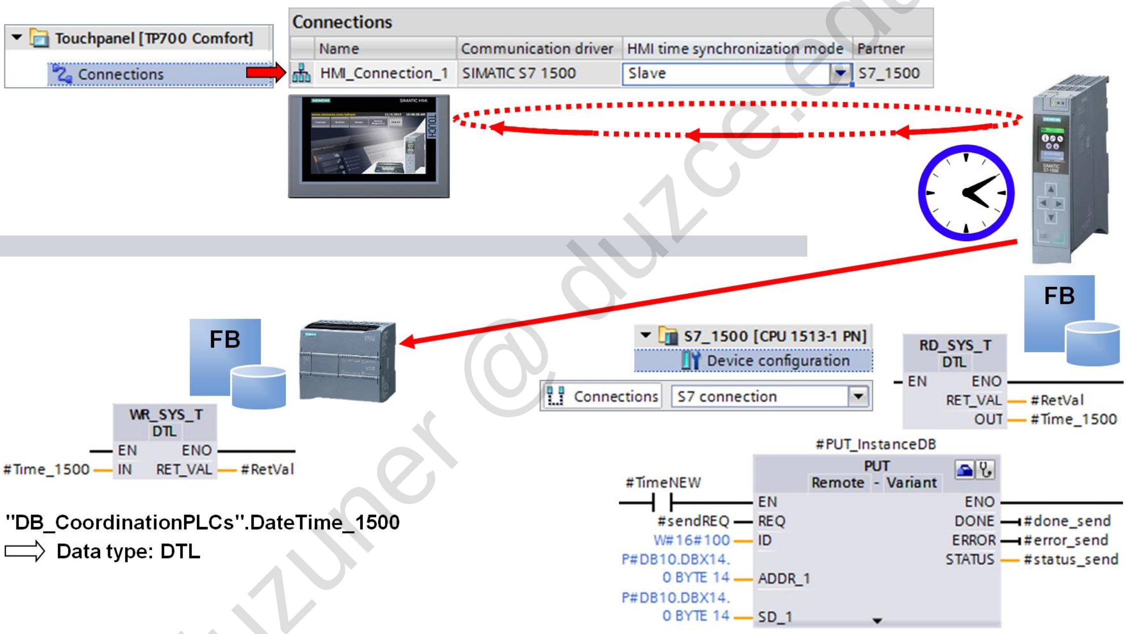


Active	Display name	PLC tag	Access mode	Address	Length
☑	Coordination	DB_CoordinationPLCs.LifebitHMI	<symbolic access>		1

"DB_CoordinationPLC".LifebitHMI
⇨ "DB_CoordinationPLC".LifebitHMI.X2

"DB_CoordinationPLC".LifebitPLC
⇨"DB_CoordinationPLCs".LifebitPLC

Time Synchronization
between Communication Partners



Touchpanel [TP700 Comfort]
- Connections

Connections

Name	Communication driver	HMI time synchronization mode	Partner
HMI_Connection_1	SIMATIC S7 1500	Slave	S7_1500

S7_1500 [CPU 1513-1 PN]
- Device configuration

Connections: S7 connection

```
        RD_SYS_T
          DTL
      EN        ENO
           RET_VAL — #RetVal
               OUT — #Time_1500
```

```
                    #PUT_InstanceDB
                         PUT
                   Remote - Variant
#TimeNEW
—| |—          EN                ENO
    #sendREQ — REQ              DONE — #done_send
    W#16#100 — ID              ERROR — #error_send
P#DB10.DBX14.                 STATUS — #status_send
     0 BYTE 14 — ADDR_1
P#DB10.DBX14.
     0 BYTE 14 — SD_1
```

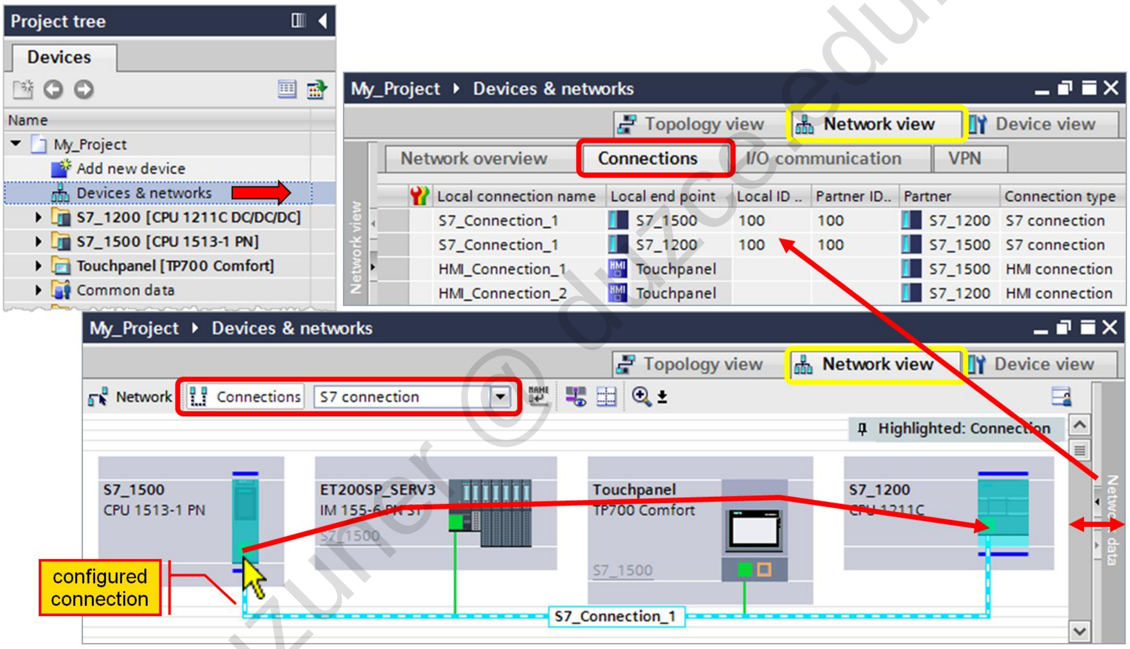
```
           WR_SYS_T
             DTL
          EN        ENO
#Time_1500 — IN   RET_VAL — #RetVal
```

"DB_CoordinationPLCs".DateTime_1500

⇨ Data type: DTL

SIEMENS

Exercise 5: Configuring an S7 Connection

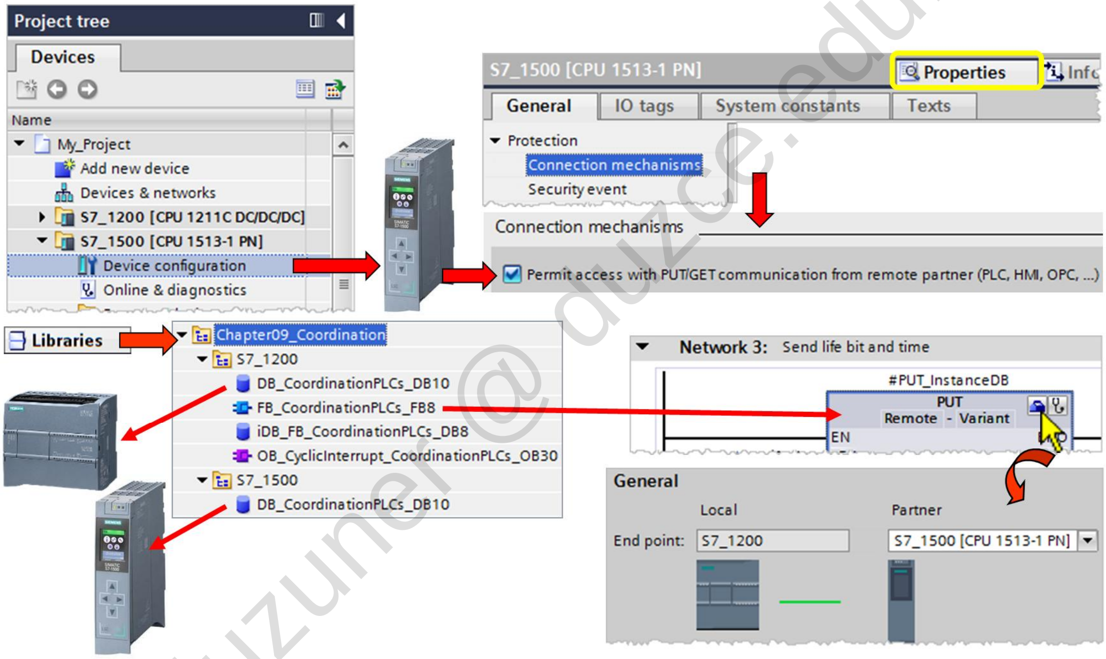


Local connection name	Local end point	Local ID ..	Partner ID..	Partner	Connection type
S7_Connection_1	S7_1500	100	100	S7_1200	S7 connection
S7_Connection_1	S7_1200	100	100	S7_1500	S7 connection
HMI_Connection_1	Touchpanel			S7_1500	HMI connection
HMI_Connection_2	Touchpanel			S7_1200	HMI connection

configured connection

Aufgabenbeschreibung

Exercise 6: Sending the Life Bit and Time



☑ Permit access with PUT/GET communication from remote partner (PLC, HMI, OPC, ...)

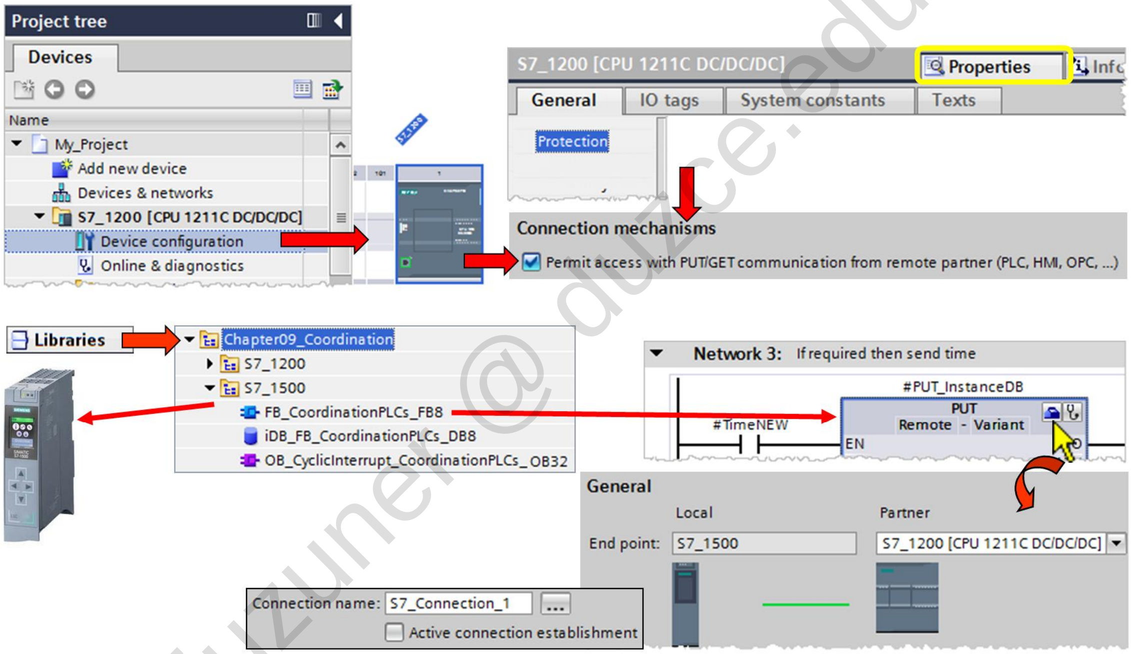
End point: S7_1200 | S7_1500 [CPU 1513-1 PN]

SITRAIN
TIA-SERV3 / Introduction to Communication Page 22 Siemens AG © 2015

Aufgabenbeschreibung

Exercise 7:
Checking the Time and, if necessary, Sending an Update

SIEMENS



☑ Permit access with PUT/GET communication from remote partner (PLC, HMI, OPC, ...)

End point: S7_1500 | S7_1200 [CPU 1211C DC/DC/DC]

Connection name: S7_Connection_1

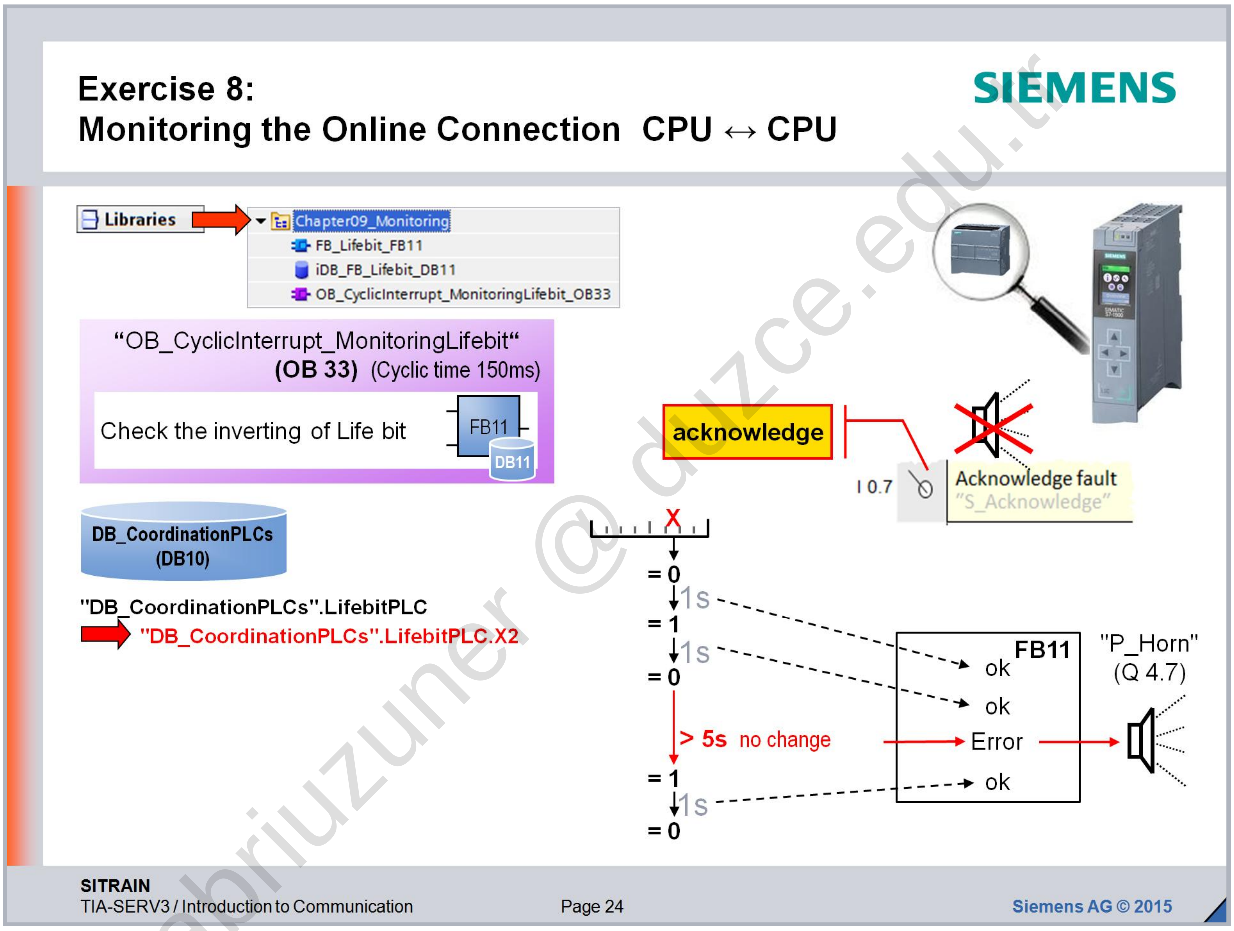
□ Active connection establishment

SITRAIN
TIA-SERV3 / Introduction to Communication

Page 23

Siemens AG © 2015

Aufgabenbeschreibung



Exercise 8:
Monitoring the Online Connection CPU ↔ CPU

SIEMENS

"OB_CyclicInterrupt_MonitoringLifebit"
(OB 33) (Cyclic time 150ms)

Check the inverting of Life bit

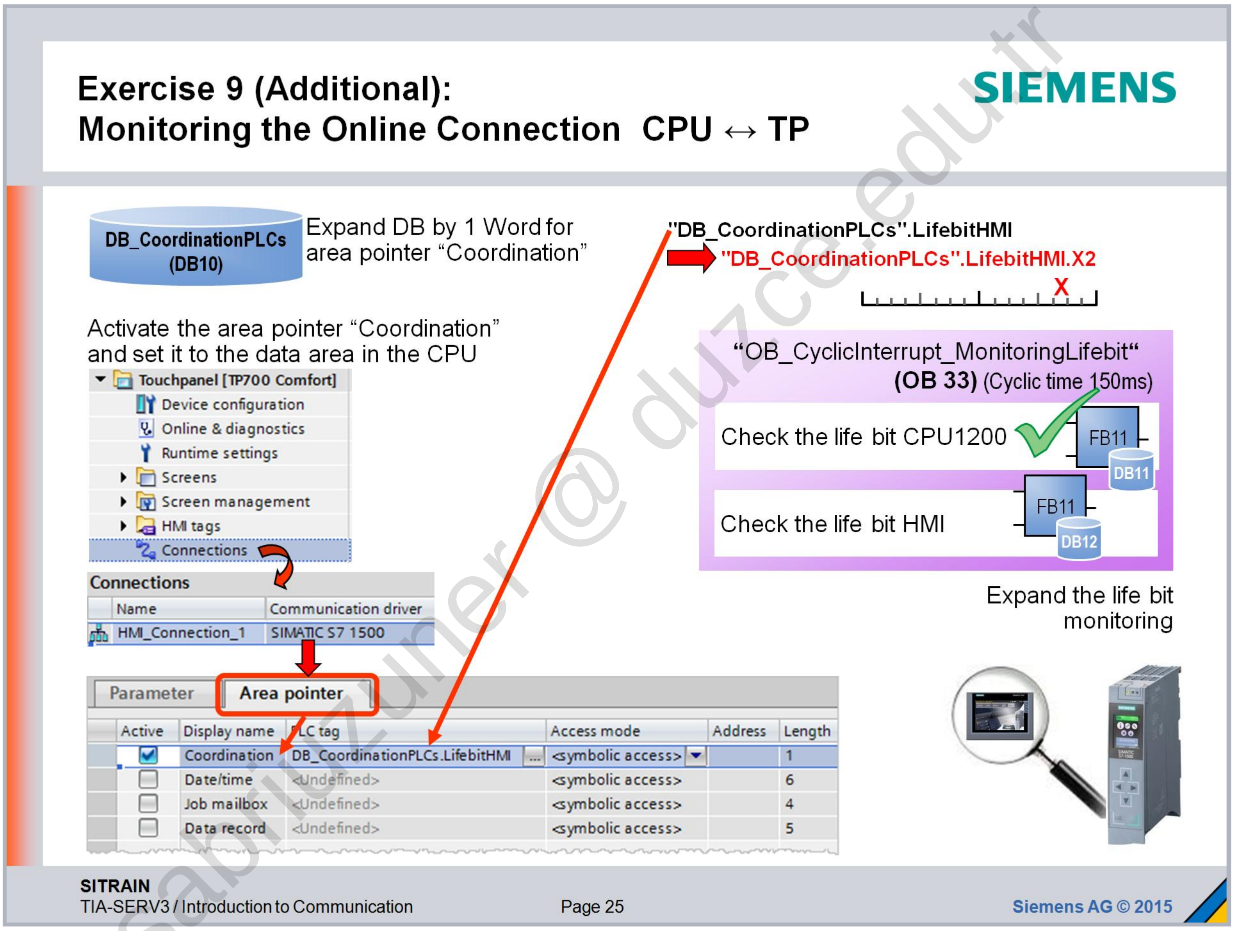
DB_CoordinationPLCs (DB10)

"DB_CoordinationPLCs".LifebitPLC

"DB_CoordinationPLCs".LifebitPLC.X2

SITRAIN
TIA-SERV3 / Introduction to Communication
Page 24
Siemens AG © 2015

Aufgabenbeschreibung



Exercise 9 (Additional): Monitoring the Online Connection CPU ↔ TP

SIEMENS

Expand DB by 1 Word for area pointer “Coordination”

Activate the area pointer “Coordination” and set it to the data area in the CPU

Name	Communication driver
HMI_Connection_1	SIMATIC S7 1500

Active	Display name	PLC tag	Access mode	Address	Length
☑	Coordination	DB_CoordinationPLCs.LifebitHMI	<symbolic access>		1
□	Date/time	<Undefined>	<symbolic access>		6
□	Job mailbox	<Undefined>	<symbolic access>		4
□	Data record	<Undefined>	<symbolic access>		5

"DB_CoordinationPLCs".LifebitHMI

"DB_CoordinationPLCs".LifebitHMI.X2

“OB_CyclicInterrupt_MonitoringLifebit“
(OB 33) (Cyclic time 150ms)

Check the life bit CPU1200

Check the life bit HMI

Expand the life bit monitoring

SITRAIN
TIA-SERV3 / Introduction to Communication Page 25 Siemens AG © 2015

Aufgabenbeschreibung

Additional Information

SITRAIN
TIA-SERV3 / Introduction to Communication

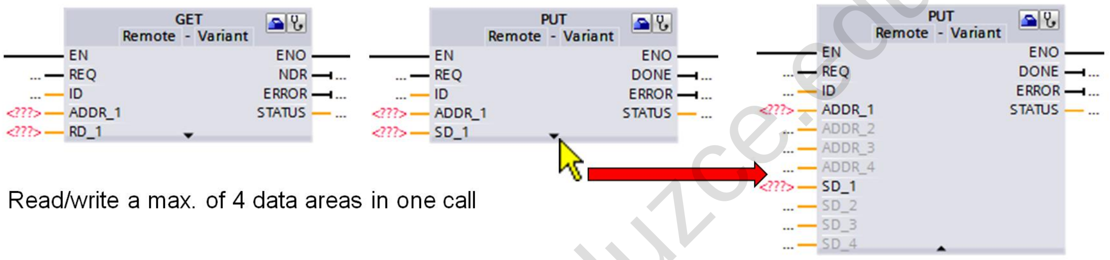
Page 26

Siemens AG © 2015

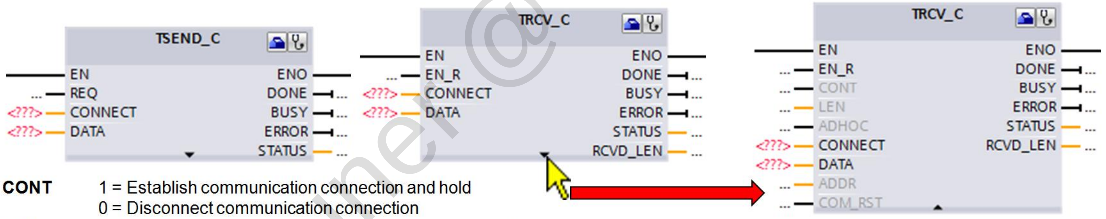
Anhang Die folgenden Seiten enthalten weiterführende Informationen bzw. dienen zum Nachschlagen oder zur Vervollständigung eines Themas.

Call Interface Communication Instructions → Optional Parameters

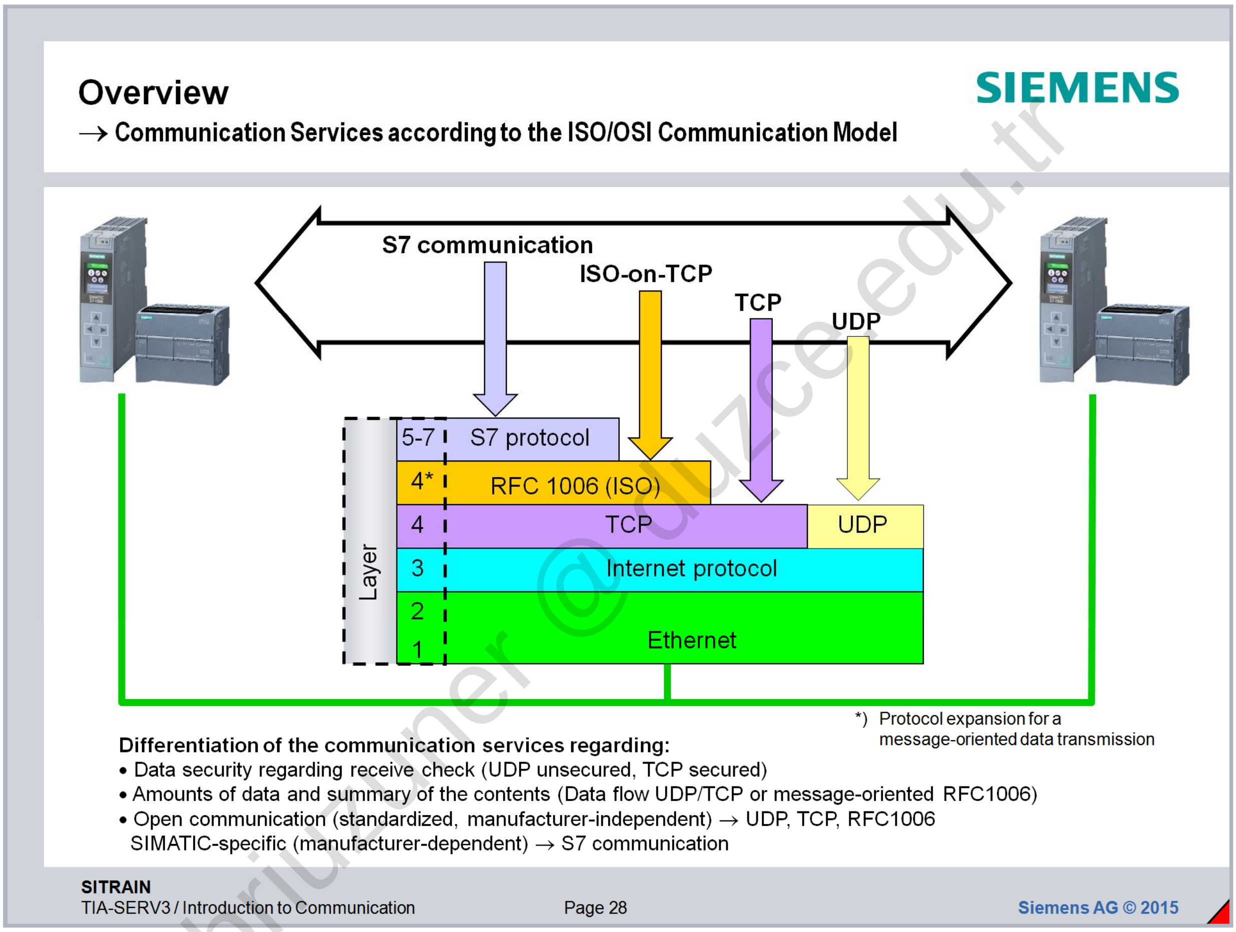
SIEMENS



Read/write a max. of 4 data areas in one call



CONT	1 = Establish communication connection and hold 0 = Disconnect communication connection
LEN	Data length, when no symbolic variable from optimized data area
ADHOC	Ad-hoc mode for connection type=TCP
ADDR	Pointer to the address of the Sender for connection type=UDP
COM_RST	0 = irrelevant; 1 = connection is reset, if necessary, active data transmission is aborted



Overview

→ Communication Services according to the ISO/OSI Communication Model

SIEMENS

*) Protocol expansion for a message-oriented data transmission

Differentiation of the communication services regarding:

- Data security regarding receive check (UDP unsecured, TCP secured)
- Amounts of data and summary of the contents (Data flow UDP/TCP or message-oriented RFC1006)
- Open communication (standardized, manufacturer-independent) → UDP, TCP, RFC1006
  SIMATIC-specific (manufacturer-dependent) → S7 communication