VIPA System 300S

SPEED7 - CPU SC | 312-5BE13 | Manual

HB140E_CPU-SC | RE_312-5BE13 | Rev. 11/27

July 2011

VIPA art of automation

Note

Every effort has been made to ensure that the information contained in this document was complete and accurate at the time of publishing. Nevertheless, the authors retain the right to modify the information. This customer document describes all the hardware units and functions known at the present time. Descriptions may be included for units which are not present at the customer site. The exact scope of delivery is described in the respective purchase contract.

CE Conformity

Hereby, VIPA GmbH declares that the products and systems are in compliance with the essential requirements and other relevant provisions of the following directives:

- 2004/108/EC Electromagnetic Compatibility Directive
- 2006/95/EC Low Voltage Directive

Conformity is indicated by the CE marking affixed to the product.

Conformity Information

For more information regarding CE marking and Declaration of Conformity (DoC), please contact your local VIPA customer service organization.

Trademarks

Information product support

Contact your local VIPA Customer Service Organization representative if you wish to report errors or questions regarding the contents of this document. If you are unable to locate a customer service center, contact VIPA as follows:

VIPA GmbH, Ohmstraße 4, 91074 Herzogenaurach, Germany

Telefax:+49 9132 744 1204
EMail: documentation@vipa.de

Technical support

Contact your local VIPA Customer Service Organization representative if you encounter problems with the product or have questions regarding the product. If you are unable to locate a customer service center, contact VIPA as follows:

VIPA GmbH, Ohmstraße 4, 91074 Herzogenaurach, Germany

Telephone: +49 9132 744 1150/1180 (Hotline)
EMail: support@vipa.de

Contents

About this manual

This manual describes the SPEED7 CPU 312SC from the System 300S. Here you may find every information for commissioning and operation.

Overview

Chapter 1: Principles

This chapter contains hints for the usage and information about the project engineering of a System 300 with the CPU 312SC from VIPA. General information like dimensions and environment conditions will also be found.

Chapter 2: Assembly and installation guidelines

In this chapter you will find all information, required for the installation and the cabling of a process control with the components of the System 300 and the CPU 312SC.

Chapter 3: Hardware description

Here the hardware components of the CPU 312SC are described.

The technical data may be found at the end of the chapter.

Chapter 4: Deployment CPU 312SC

This chapter describes the deployment of the CPU 312SC with SPEED7 technology in the System 300. The description refers directly to the CPU and to the employment in connection with peripheral modules that are mounted on a profile rail together with the CPU at standard bus.

Chapter 5: Deployment I/O periphery

This chapter contains all information necessary for the deployment of the in-/output periphery of the CPU 312SC. It describes functionality, project engineering and diagnostic of the analog and digital part.

Chapter 6: Deployment PtP communication

Content of this chapter is the deployment of the RS485 slot for serial PtP communication.

Here you'll find all information about the protocols and project engineering of the interface, which are necessary for the serial communication using the RS485 interface.

Chapter 7: WinPLC7

In this chapter the programming and simulation software WinPLC7 from VIPA is presented. WinPLC7 is suited for every with Siemens STEP®7 programmable PLC.
Besides the system presentation and installation here the basics for using the software is explained with a sample project.
More information concerning the usage of WinPLC7 may be found in the online help respectively in the online documentation of WinPLC7.

Objective and contents

The manual describes the SPEED7 CPU 312SC from VIPA. It contains a description of the construction, project implementation and usage.

This manual is part of the documentation package with order number HB140E_CPU_SC and relevant for:

Product	Order number	as of state: CPU-HW	CPU-FW
CPU 312SC	VIPA 312-5BE13	01	V354

Target audience

The manual is targeted at users who have a background in automation technology.

Structure of the manual

The manual consists of chapters. Every chapter provides a self-contained description of a specific topic.

Guide to the document

The following guides are available in the manual:

- an overall table of contents at the beginning of the manual
- an overview of the topics for every chapter

Availability

The manual is available in:

- printed form, on paper
- in electronic form as PDF-file (Adobe Acrobat Reader)

Icons Headings

Important passages in the text are highlighted by following icons and headings:

Danger!
Immediate or likely danger.
Personal injury is possible.

Attention!
Damages to property is likely if these warnings are not heeded.

Note!
Supplementary information and useful tips.

Safety information

Applications conforming with specifications

The SPEED7 CPU is constructed and produced for:

- all VIPA System 300 components
- communication and process control
- general control and automation applications
- industrial applications
- operation within the environmental conditions specified in the technical data
- installation into a cubicle

Danger!
This device is not certified for applications in

- in explosive environments (EX-zone)

Documentation

The manual must be available to all personnel in the

- project design department
- installation department
- commissioning
- operation

The following conditions must be met before using or commissioning the components described in this manual:

- Modification to the process control system should only be carried out when the system has been disconnected from power!
- Installation and modifications only by properly trained personnel
- The national rules and regulations of the respective country must be satisfied (installation, safety, EMC ...)

Disposal

National rules and regulations apply to the disposal of the unit!

Chapter 1 Basics

Overview

This chapter contains hints for the usage and information about the project engineering of a System 300 with the CPU 312SC from VIPA.

General information like dimensions and environment conditions will also be found.

Content

Safety Information for Users

Handling of electrostatic sensitive modules

VIPA modules make use of highly integrated components in MOS-Technology. These components are extremely sensitive to over-voltages that can occur during electrostatic discharges.

The following symbol is attached to modules that can be destroyed by electrostatic discharges.

The Symbol is located on the module, the module rack or on packing material and it indicates the presence of electrostatic sensitive equipment.

It is possible that electrostatic sensitive equipment is destroyed by energies and voltages that are far less than the human threshold of perception. These voltages can occur where persons do not discharge themselves before handling electrostatic sensitive modules and they can damage components thereby, causing the module to become inoperable or unusable.

Modules that have been damaged by electrostatic discharges can fail after a temperature change, mechanical shock or changes in the electrical load.

Only the consequent implementation of protection devices and meticulous attention to the applicable rules and regulations for handling the respective equipment can prevent failures of electrostatic sensitive modules.

Shipping of modules

Modules must be shipped in the original packing material.

Measurements and alterations on electrostatic sensitive modules

When you are conducting measurements on electrostatic sensitive modules you should take the following precautions:

- Floating instruments must be discharged before use.
- Instruments must be grounded.

Modifying electrostatic sensitive modules you should only use soldering irons with grounded tips.

Attention!
Personnel and instruments should be grounded when working on electrostatic sensitive modules.

General description of the System 300

The System 300

The System 300 is a modular automation system for middle and high performance needs, that you can use either central or decentralized. The single modules are directly clipped to the profile rail and are connected together with the help of bus clips at the backside.

The CPUs of the System 300 are instruction set compatible to S7-300 from Siemens.

System 300V
System 300S

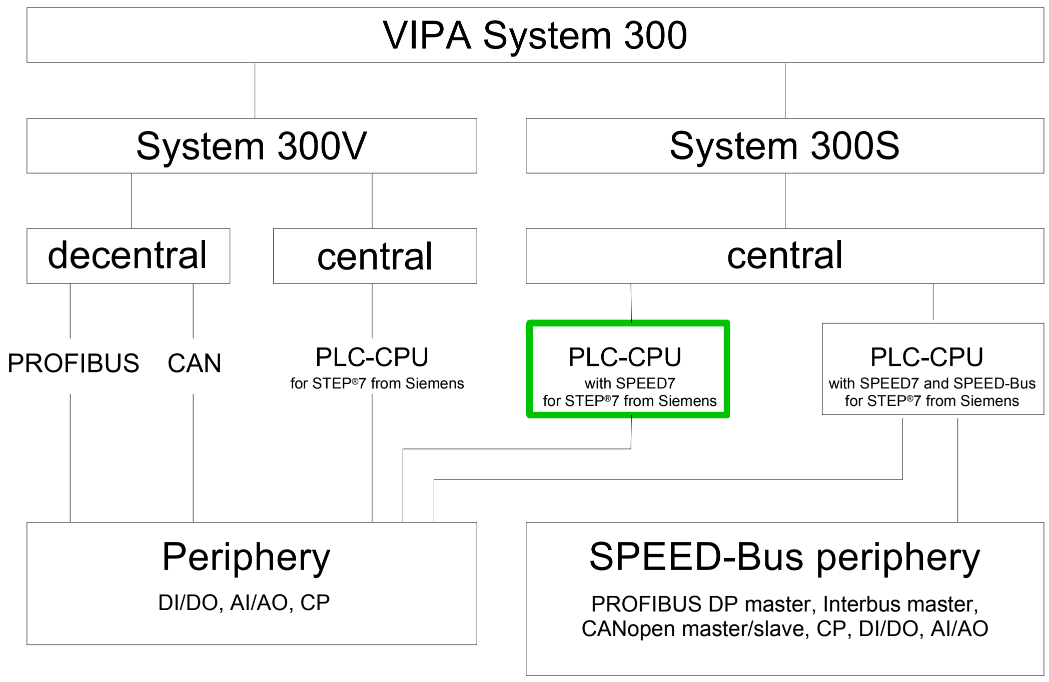
VIPA differentiates between System 300V and System 300S.

- System 300V

  The System 300V allows you to resolve automation tasks central and decentralized. The single modules of the System 300V from VIPA are similar in construction to Siemens. Due to the compatible backplane bus, the modules from VIPA and Siemens may be mixed.

- System 300S

  The System 300S extends the central area with high-speed CPUs that have the integrated SPEED7 chip. Additionally some CPUs have got a parallel SPEED-Bus that allows the modular connection of fast peripheral modules like IOs or bus master.


VIPA System 300
System 300V
System 300S
decentral
central
central
PROFIBUS
CAN
PLC-CPU for STEP®7 from Siemens
PLC-CPU with SPEED7 for STEP®7 from Siemens
PLC-CPU with SPEED7 and SPEED-Bus for STEP®7 from Siemens
Periphery
DI/DO, AI/AO, CP
SPEED-Bus periphery
PROFIBUS DP master, Interbus master, CANopen master/slave, CP, DI/DO, AI/AO

Manual overview

This manual describes the SC CPU from the System 300S. Here it concerns a CPU with input/output periphery and integrated SPEED7-Technology without SPEED-Bus.

The description of the System 300V CPU 31x without SPEED7 and the concerning peripheral modules like digital and analog in-/output modules, power supplies and bus coupler may be found in the HB 130.

Operating structure of a CPU

General

The CPU contains a standard processor with internal program memory. In combination with System 300S peripherals the unit provides a powerful solution for process automation applications within the System 300 family.

A CPU supports the following modes of operation:

- cyclic operation
- timer processing
- alarm controlled operation
- priority based processing

Cyclic processing

Cyclic processing represents the major part of all the processes that are executed in the CPU. Identical sequences of operations are repeated in a never-ending cycle.

Timer processing

Where a process requires control signals at constant intervals you can initiate certain operations based upon a **timer**, e.g. not critical monitoring functions at one-second intervals.

Alarm controlled processing

If a process signal requires a quick response you would allocate this signal to an **alarm controlled** procedure. An alarm can activate a procedure in your program.

Priority based processing

The above processes are handled by the CPU in accordance with their **priority**. Since a timer or an alarm event requires a quick reaction, the CPU will interrupt the cyclic processing when these high-priority events occur to react to the event. Cyclic processing will resume, once the reaction has been processed. This means that cyclic processing has the lowest priority.

CPU Applications

Overview

The program that is present in every CPU is divided as follows:

- System routine
- User application

System routine

The system routine organizes all those functions and procedures of the CPU that are not related to a specific control application.

User application

This consists of all the functions that are required for the processing of a specific control application. The operating modules provide the interfaces to the system routines.

Operands of the CPU

Overview

The following series of operands is available for programming the CPU:

- Process image and periphery
- Bit memory
- Timers and counters
- Data blocks

Process image and periphery

The user application can quickly access the process image of the inputs and outputs PAA/PAE. You may manipulate the following types of data:

- individual Bits
- Bytes
- Words
- Double Words

You may also gain direct access to peripheral modules via the bus from user application. The following types of data are available:

- Bytes
- Words
- Blocks

Bit Memory

The bit memory is an area of memory that is accessible by means of certain operations. Bit memory is intended to store frequently used working data.

You may access the following types of data:

- individual Bits
- Bytes
- Words
- Double words

Timers and counters

In your program you may load cells of the timer with a value between 10ms and 9990s. As soon as the user application executes a start-operation, the value of this timer is decremented by the interval that you have specified until it reaches zero.

You may load counter cells with an initial value (max. 999) and increment or decrement these when required.

Data Blocks

A data block contains constants or variables in the form of bytes, words or double words. You may always access the current data block by means of operands.

You may access the following types of data:

- individual Bits
- Bytes
- Words
- Double words

CPU 312SC

Overview

The SC CPU bases upon the SPEED7 technology. This supports the CPU at programming and communication by means of co-processors that causes a power improvement for highest needs.

The CPU is programmed in STEP®7 from Siemens. For this you may use WinPLC7 from VIPA or the Siemens SIMATIC Manager.

Due to the SPEED7 chipset the CPU behaves like a CPU 318. Here the instruction set of the S7-400 from Siemens is used.

The CPU with integrated Ethernet-PG/OP channel, a MPI- and RS485-slot simplifies the integration of the CPU into an existing network or the connection of additional peripheral equipment.

The user application is stored in the battery buffered RAM or on an additionally pluggable MMC storage module.

Memory management

The CPU has an integrated memory. Information about the capacity (min. capacity ... max capacity) of the memory may be found at the front of the CPU.

The memory is divided into the following 3 parts:

- Load memory 512kbyte
- Code memory (50% of the work memory)
- Data memory (50% of the work memory)

The work memory has 64kbyte. There is the possibility to extend the work memory to its maximum printed capacity 512kbyte by means of a MCC memory extension card.

Integrated Ethernet-PG/OP-channel

The CPU has an Ethernet interface for PG/OP communication. After the assignment of IP address parameters by "Assign Ethernet Address" respectively by a "minimum project" the Ethernet PG/OP channel may directly be addressed by means of the "PLC" functions to program and remote control the CPU. A max. of 4 PG/OP connections is available.

You may also access the CPU with a visualization software via these connections.

Operation Security

- Wiring by CageClamps at the front connector
- Core cross-section 0.08...2.5mm^2
- Total isolation of the wiring at module change
- Potential separation of all modules to the backplane bus
- ESD/Burst acc. IEC 61000-4-2/IEC 61000-4-4 (up to level 3)
- Shock resistance acc. IEC 60068-2-6 / IEC 60068-2-27 (1G/12G)

Environmental conditions

- Operating temperature: 0 ... +60°C
- Storage temperature: -25 ... +70°C
- Relative humidity: 5 ... 95% without condensation
- Ventilation by means of a fan is not required

Dimensions/ Weight

- Dimensions of the basic enclosure: 2tier width: (HxWxD) in mm: 80x125x120
- Available lengths of the profile rail in mm: 160, 482, 530, 830 and 2000

Compatibility

Modules and CPUs of the System 300 from VIPA and Siemens may be used at the "Standard" bus as a mixed configuration.

The project engineering takes place in WinPLC7 from VIPA or in the hardware configurator from Siemens.

The SPEED7 CPUs from VIPA are instruction compatible to the programming language STEP®7 from Siemens and may be programmed via WinPLC7 from VIPA or via the Siemens SIMATIC Manager.

Here the instruction set of the S7-400 from Siemens is used.

Note!

Please do always use the **CPU 312C (6ES7 312-5BE03-0AB0 V2.6)** from Siemens of the hardware catalog to project a CPU 312SC from VIPA.

For the project engineering, a thorough knowledge of the Siemens SIMATIC Manager and the hardware configurator from Siemens is required!

Integrated power supply

The CPU comes with an integrated power supply. The power supply has to be supplied with DC 24V. By means of the supply voltage, the internal electronic is supplied as well as the backplane bus for the peripherals modules. The power supply is protected against inverse polarity and overcurrent.

Chapter 2 Assembly and installation guidelines

Overview

In this chapter you will find all information, required for the installation and the cabling of a process control with the components of the System 300 and the CPU 312SC.

Content

Topic **Page**

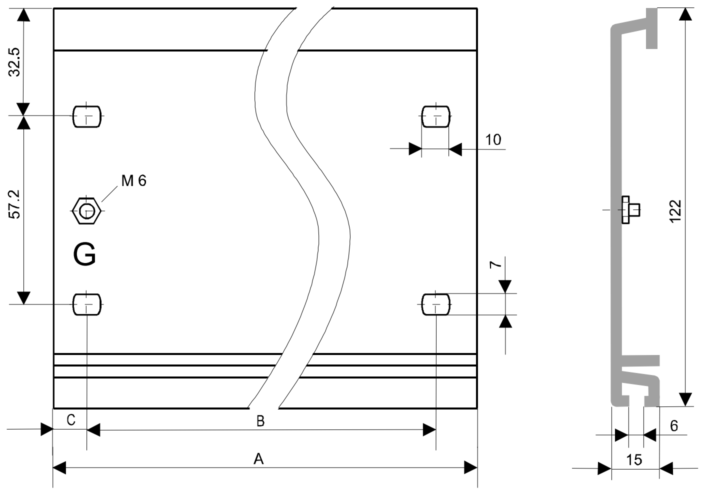
Overview

General

The single modules are directly installed on a profile rail and connected via the backplane bus connector. Before installing the modules you have to clip the backplane bus connector to the module from the backside.

The backplane bus connector is delivered together with the peripheral modules.

Profile rail

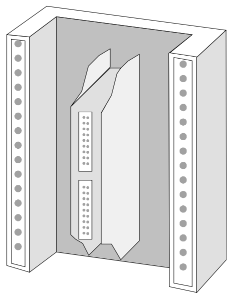


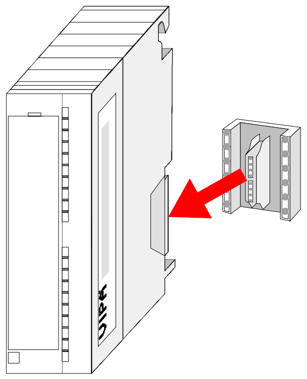
Order number	A	B	C
VIPA 390-1AB60	160mm	140mm	10mm
VIPA 390-1AE80	482mm	466mm	8.3mm
VIPA 390-1AF30	530mm	500mm	15mm
VIPA 390-1AJ30	830mm	800mm	15mm
VIPA 390-9BC00*	2000mm	Drillings only left	15mm

* Unit pack: 10 pieces

Bus connector

For the communication between the modules the System 300 uses a backplane bus connector. Backplane bus connectors are included in the delivering of the peripheral modules and are clipped at the module from the backside before installing it to the profile rail.



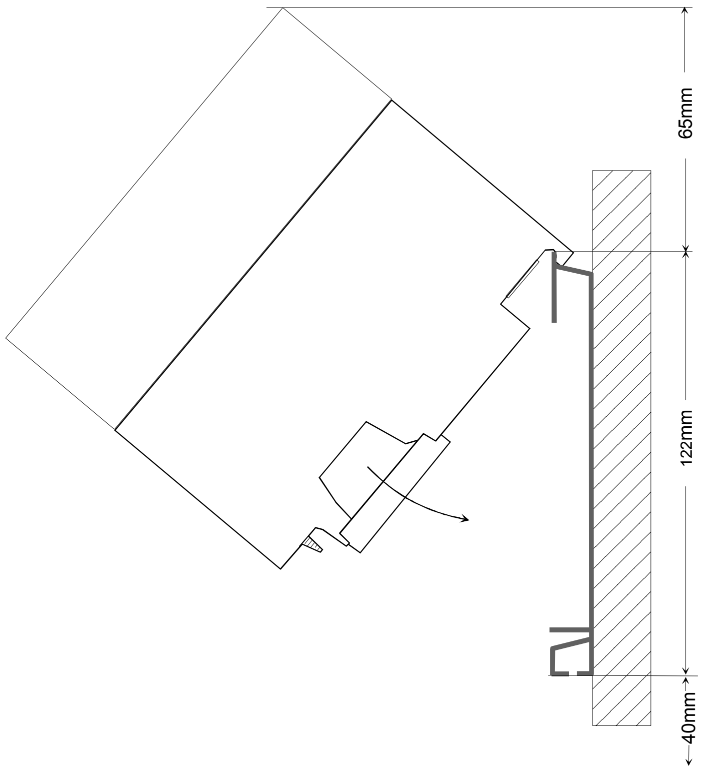


Installation dimensions

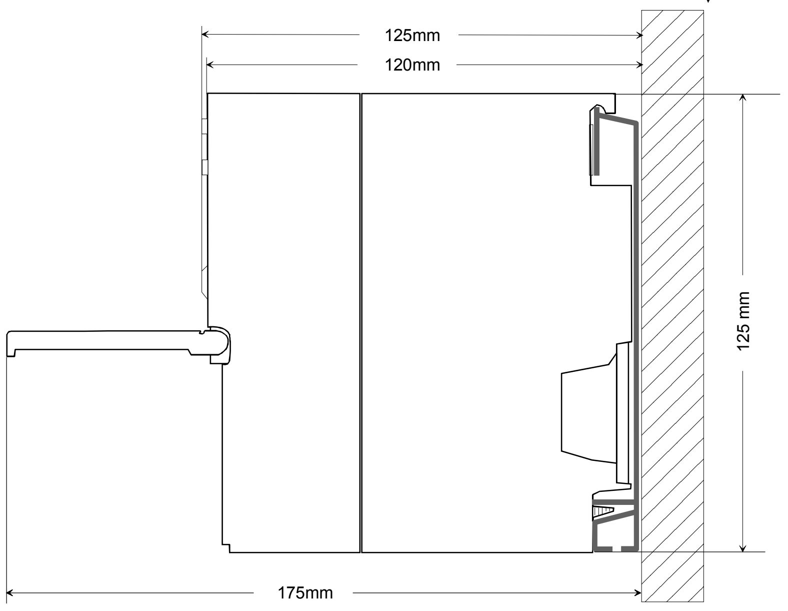
Dimensions Basic enclosure

2tier width (WxHxD) in mm: 80 x 125 x 120

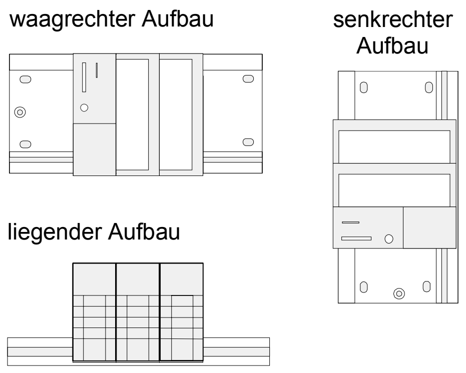
Dimensions



Installation dimensions



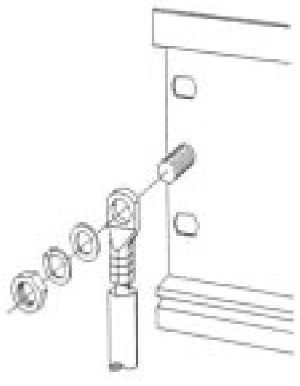
Installation



Assembly possibilities

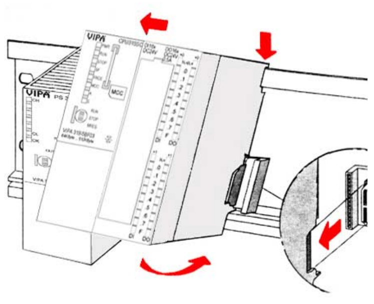
Please regard the allowed environment temperatures:

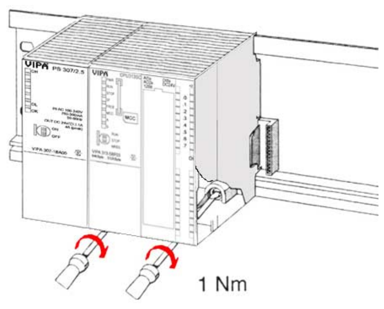
• horizontal assembly:	from 0 to 60°C
• vertical assembly:	from 0 to 40°C
• lying assembly:	from 0 to 40°C



Approach

- Bolt the profile rail with the background (screw size: M6), so that you still have minimum 65mm space above and 40mm below the profile rail.
- If the background is a grounded metal or device plate, please look for a low-impedance connection between profile rail and background.
- Connect the profile rail with the protected earth conductor. For this purpose there is a bolt with M6-thread.
- The minimum cross-section of the cable to the protected earth conductor has to be 10mm^2.
- Stick the power supply to the profile rail and pull it to the left side to the grounding bolt of the profile rail.
- Fix the power supply by screwing.
- Take a backplane bus connector and click it at the CPU from the backside like shown in the picture.
- Stick the CPU to the profile rail right from the power supply and pull it to the power supply.





- Click the CPU downwards and bolt it like shown.
- Repeat this procedure with the peripheral modules, by clicking a backplane bus connector, stick the module right from the modules you've already fixed, click it downwards and connect it with the backplane bus connector of the last module and bolt it.

Danger!

- The power supplies must be released before installation and repair tasks, i.e. before handling with the power supply or with the cabling you must disconnect current/voltage (pull plug, at fixed connection switch off the concerning fuse)!
- Installation and modifications only by properly trained personnel!

Cabling

Overview

The CPUs are exclusively delivered with CageClamp contacts. The connection of the I/O periphery happens by 40pole front screw connection.

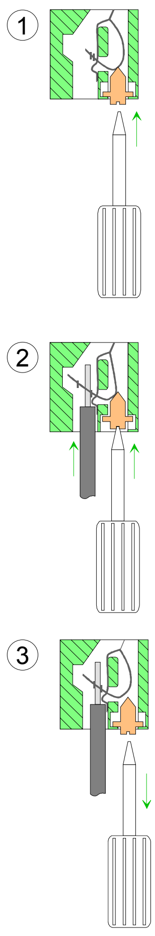
Danger!

- The power supplies must be released before installation and repair tasks, i.e. before handling with the power supply or with the cabling you must disconnect current/voltage (pull plug, at fixed connection switch off the concerning fuse)!
- Installation and modifications only by properly trained personnel!

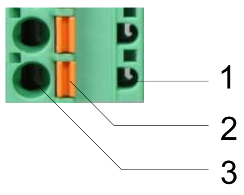
CageClamp technology (green)

For the cabling of power supply of a CPU, a green plug with CageClamp technology is deployed.

The connection clamp is realized as plug that may be clipped off carefully if it is still cabled.



Here wires with a cross-section of 0.08mm^2 to 2.5mm^2 may be connected. You can use flexible wires without end case as well as stiff wires.



[1]	Test point for 2mm test tip
[2]	Locking (orange) for screwdriver
[3]	Round opening for wires

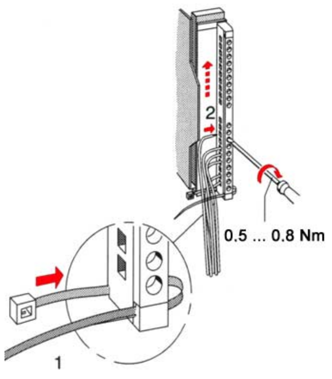
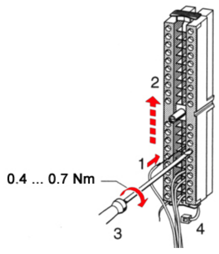
The picture on the left side shows the cabling step by step from top view.

- For cabling you push the locking vertical to the inside with a suiting screwdriver and hold the screwdriver in this position.
- Insert the de-isolated wire into the round opening. You may use wires with a cross-section from 0.08mm^2 to 2.5mm^2.
- By removing the screwdriver the wire is connected safely with the plug connector via a spring.

Front connectors of the in-/output modules

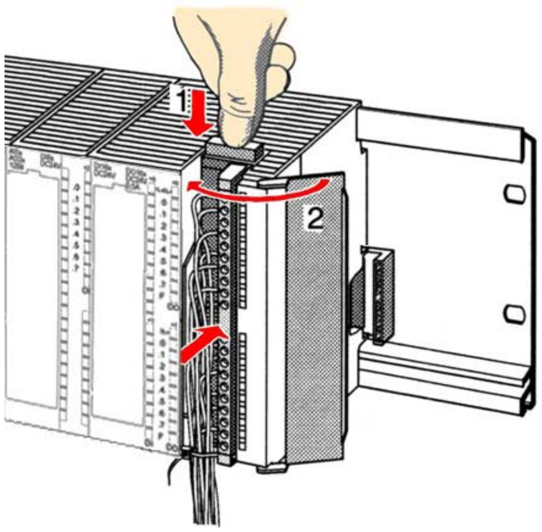
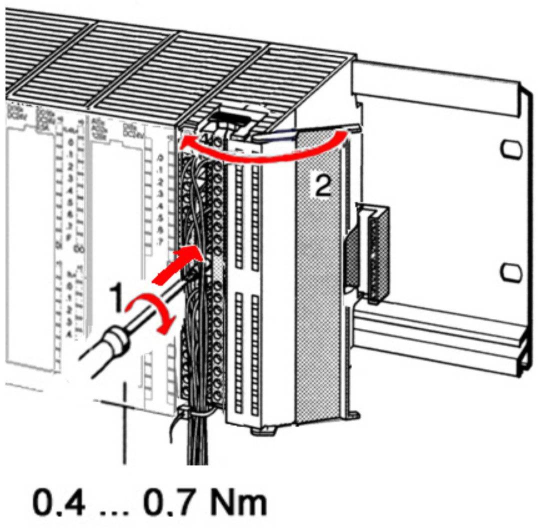
In the System 300 there are 20- and 40pole front connectors. The connection of the in/output periphery of the CPU happens by means of the 40pole front connector.

In the following the cabling of the two variants are shown:

20pole screw connection VIPA 392-1AJ00	**40pole screw connection** VIPA 392-1AM00
Open the front flap of your I/O module.	
Bring the front connector in cabling position. For this you plug the front connector on the module until it locks. In this position the front connector juts out of the module and has no contact yet.	
De-isolate your wires. If needed, use core end cases.	
Thread the included cable binder into the front connector.	
If you want to lead out your cables from the bottom of the module, start with the cabling from bottom to top, res. from top to bottom, if the cables should be led out at the top.	
Bolt also the connection screws of not cabled screw clamps.	
	Put the included cable binder around the cable bundle and the front connector. 
Fix the cable binder for the cable bundle.	

continued ...

... continued

20pole screw connection	40pole screw connection
Push the release key at the front connector on the upper side of the module and at the same time push the front connector into the module until it locks. 	Bolt the fixing screw of the front connector. 
Now the front connector is electrically connected with your module.	
Close the front flap.	
Fill out the labeling strip to mark the single channels and push the strip into the front flap.	

Installation Guidelines

General

The installation guidelines contain information about the interference free deployment of System 300V systems. There is the description of the ways, interference may occur in your control, how you can make sure the electromagnetic digestibility (EMC), and how you manage the isolation.

What means EMC?

Electromagnetic digestibility (EMC) means the ability of an electrical device, to function error free in an electromagnetic environment without being interferenced res. without interferencing the environment.

All System 300 components are developed for the deployment in hard industrial environments and fulfill high demands on the EMC. Nevertheless you should project an EMC planning before installing the components and take conceivable interference causes into account.

Possible interference causes

Electromagnetic interferences may interfere your control via different ways:

- Fields
- I/O signal conductors
- Bus system
- Current supply
- Protected earth conductor

Depending on the spreading medium (lead bound or lead free) and the distance to the interference cause, interferences to your control occur by means of different coupling mechanisms.

One differs:

- galvanic coupling
- capacitive coupling
- inductive coupling
- radiant coupling

Basic rules for EMC

In the most times it is enough to take care of some elementary rules to guarantee the EMC. Please regard the following basic rules when installing your PLC.

- Take care of a correct area-wide grounding of the inactive metal parts when installing your components.
  - Install a central connection between the ground and the protected earth conductor system.
  - Connect all inactive metal extensive and impedance-low.
  - Please try not to use aluminum parts. Aluminum is easily oxidizing and is therefore less suitable for grounding.
- When cabling, take care of the correct line routing.
  - Organize your cabling in line groups (high voltage, current supply, signal and data lines).
  - Always lay your high voltage lines and signal res. data lines in separate channels or bundles.
  - Route the signal and data lines as near as possible beside ground areas (e.g. suspension bars, metal rails, tin cabinet).
- Proof the correct fixing of the lead isolation.
  - Data lines must be laid isolated.
  - Analog lines must be laid isolated. When transmitting signals with small amplitudes the one sided laying of the isolation may be favorable.
  - Lay the line isolation extensively on an isolation/protected earth conductor rail directly after the cabinet entry and fix the isolation with cable clamps.
  - Make sure that the isolation/protected earth conductor rail is connected impedance-low with the cabinet.
  - Use metallic or metalized plug cases for isolated data lines.
- In special use cases you should appoint special EMC actions.
  - Wire all inductivities with suppressors, which are not addressed by the System 300 modules.
  - For lightening cabinets you should prefer incandescent lamps and avoid luminescent lamps.
- Create a homogeneous reference potential and ground all electrical operating supplies when possible.
  - Please take care for the targeted employment of the grounding actions. The grounding of the PLC is a protection and functionality activity.
  - Connect installation parts and cabinets with the System 300 in star topology with the isolation/protected earth conductor system. So you avoid ground loops.
  - If potential differences between installation parts and cabinets occur, lay sufficiently dimensioned potential compensation lines.

Isolation of conductors

Electrical, magnetic and electromagnetic interference fields are weakened by means of an isolation, one talks of absorption.

Via the isolation rail, that is connected conductive with the rack, interference currents are shunt via cable isolation to the ground. Hereby you have to make sure, that the connection to the protected earth conductor is impedance-low, because otherwise the interference currents may appear as interference cause.

When isolating cables you have to regard the following:

- If possible, use only cables with isolation tangle.
- The hiding power of the isolation should be higher than 80%.
- Normally you should always lay the isolation of cables on both sides. Only by means of the both-sided connection of the isolation you achieve a high quality interference suppression in the higher frequency area.

  Only as exception you may also lay the isolation one-sided. Then you only achieve the absorption of the lower frequencies. A one-sided isolation connection may be convenient, if:
  - the conduction of a potential compensating line is not possible
  - analog signals (some mV res. µA) are transferred
  - foil isolations (static isolations) are used.
- With data lines always use metallic or metalized plugs for serial couplings. Fix the isolation of the data line at the plug rack. Do not lay the isolation on the PIN 1 of the plug bar!
- At stationary operation it is convenient to de-insulate the isolated cable interruption free and lay it on the isolation/protected earth conductor line.
- To fix the isolation tangles use cable clamps out of metal. The clamps must clasp the isolation extensively and have well contact.
- Lay the isolation on an isolation rail directly after the entry of the cable in the cabinet. Lead the isolation further on to the System 300 module and **don't** lay it on there again!

Please regard at installation!

At potential differences between the grounding points, there may be a compensation current via the isolation connected at both sides.

Remedy: Potential compensation line

Chapter 3 Hardware description

Overview

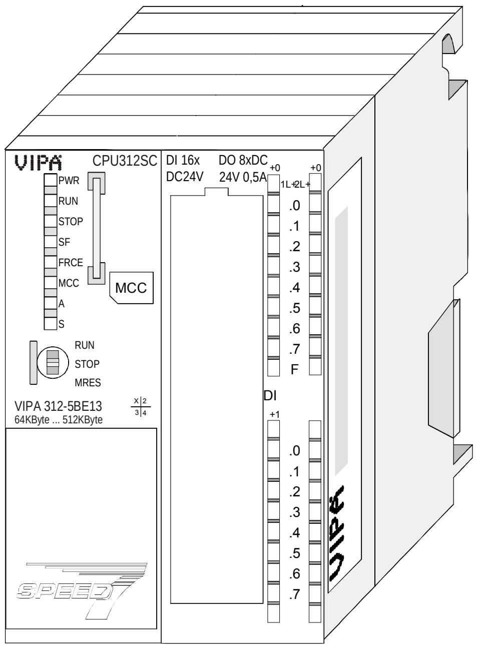
Here the hardware components of the CPU 312SC are described.
The technical data are at the end of the chapter.

Content

Properties

CPU 312SC
312-5BE13

- SPEED7 technology integrated
- Instruction set compatible to STEP®7 from Siemens with access to the peripheral modules of the System 300V for the standard bus
- Integrated DC24V power supply unit
- 64kbyte work memory (32kbyte code, 32kbyte data)
- Memory expandable to max. 512kbyte (256kbyte code, 256kbyte data)
- MCC slot for external memory cards and memory extension
- Status-LEDs for operating state and diagnosis
- Real-time clock battery buffered
- Ethernet PG/OP interface integrated
- MPI interface
- RS485 interface for PtP communication
- Digital I/Os: DI 16xDC24V / DO 8xDC 24V, 0.5A
- 2 counter (10kHz)
- 512 timer
- 512 counter
- 8192 bit memory

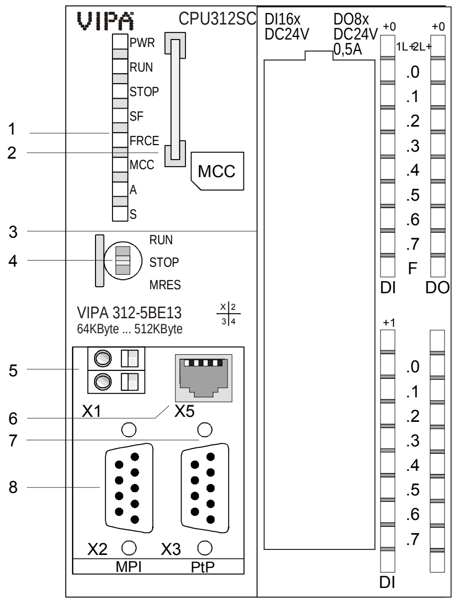


Order data

Type	Order number	Description
CPU 312SC	VIPA 312-5BE13	MPI interface, card slot, Real-time clock, Ethernet interface for PG/OP, PtP via RS485, DI 16xDC24V / DO 8xDC 24V, 0.5A, 2 counter

Structure

CPU 312SC
312-5BE13



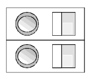
[1]	LEDs of the CPU part
[2]	MCC slot
[3]	LEDs of the I/O part
[4]	Operating mode switch CPU

The following components are under the front flap

[5]	Slot for DC 24V power supply
[6]	Ethernet interface for PG/OP channel
[7]	PtP interface
[8]	MPI interface

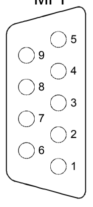
Interfaces

X1



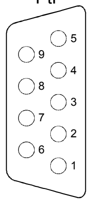
+	1	+ DC 24 V
-	2	0 V

X2
MPI



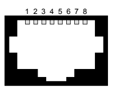
1	n. c.
2	M24V
3	RxD/TxD-P (line B)
4	RTS
5	M5V
6	P5V
7	P24V
8	RxD/TxD-N (line A)
9	n.c.

X3
PtP

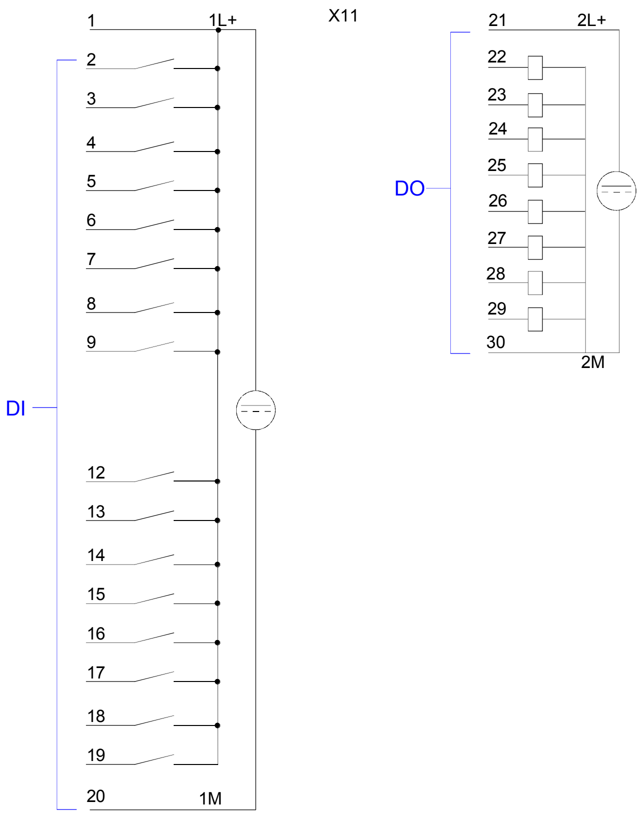


1	n. c.
2	M24V
3	RxD/TxD-P (line B)
4	RTS
5	M5V
6	P5V
7	P24V
8	RxD/TxD-N (line A)
9	n.c.

X5



1	Transmit +
2	Transmit -
3	Receive +
4	n. c.
5	n. c.
6	Receive -
7	n. c.
8	n. c.



Power supply X1

The CPU has an integrated power supply. The power supply has to be provided with DC 24V. For this serves the DC 24V slot, that is underneath the flap.

Via the power supply not only the internal electronic is provided with voltage, but by means of the backplane bus also the connected modules. The power supply is protected against polarity inversion and overcurrent. The internal electronic is galvanically connected with the supply voltage.

Please regard that the integrated power supply may provide the backplane bus with a sum of max. 5A depending on the CPU.

MPI interface X2

9-pin SubD jack:

The MPI interface serves for the connection between programming unit and CPU. By means of this the project engineering and programming happens. In addition MPI serves for communication between several CPUs or between HMIs and CPU.

Standard setting is MPI Address 2.

PtP interface X3

9-pin SubD jack:

The CPU has a RS 485 interface. The interface is fix set to PtP communication.

Using the *PtP* functionality the RS485 interface is allowed to connect via serial point-to-point connection to different source res. target systems. Here the following protocols are supported:

ASCII, STX/ETX, 3964R, USS and Modbus master (ASCII, RTU).

The PtP communication is configured during run-time by means of the SFC 216 (SER_CFG). The communication happens by means of the SFC 217 (SER_SND) and SFC 218 (SER_RCV).

Ethernet PG/OP channel X5

9-pin SubD jack:

The RJ45 jack serves the interface to the Ethernet PG/OP channel. This interface allows you to program res. remote control your CPU, to access the internal website or to connect a visualization via up to 4 PG/OP connections. Here a transfer rate of 100MBit (full duplex) is supported.

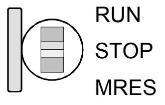
For online access to the CPU via Ethernet PG/OP channel valid IP address parameters have to be assigned to this. More may be found at chapter "Deployment CPU 31..." at "Initialization Ethernet PG/OP channel".

LEDs CPU part

The CPU has got one row of LEDs on the front side. The following table shows you the usage of the LEDs and the according colors:

Label	Color	Meaning
PWR	green	CPU part is provided with internal 5V
RUN	green	CPU is in the operating mode RUN
STOP	yellow	CPU is in the operating mode STOP
SF	red	On at system errors
FRCE	yellow	On as soon as variables are forced (fixed)
MCC	yellow	Blinks at storage media access
A	green	Activity: on: physically connected off: no physical connection blinks: shows Ethernet activity
S	green	Speed: on: 100MBit off: 10MBit

Operating mode switch



With the operating mode switch you may switch the CPU between STOP and RUN. During the transition from STOP to RUN the operating mode START-UP is driven by the CPU.

Placing the switch to MRES (Memory Reset), you request an overall reset with following load from MMC, if a project there exists.

Storage media slot

As external storage medium for applications and firmware you may use a MMC (**m**ulti**m**edia **c**ard) or a MMC for memory extension. The MCC can additionally be used as an external storage medium.

The VIPA storage media are pre-formatted with the PC format FAT and may be accessed via a card reader.

After PowerON (in operation mode STOP) respectively an overall reset the CPU checks, if there is a storage medium with data valid for the CPU.

Memory management

The CPU has an integrated memory. Information about the capacity (min. capacity ... max capacity) of the memory may be found at the front of the CPU.

The memory is divided into the following 3 parts:

- Load memory 512kbyte
- Code memory (50% of the work memory)
- Data memory (50% of the work memory)

The work memory has 64kbyte. There is the possibility to extend the work memory to its maximum printed capacity 512kbyte by means of a MCC memory extension card.

In-/Output range CPU 312SC

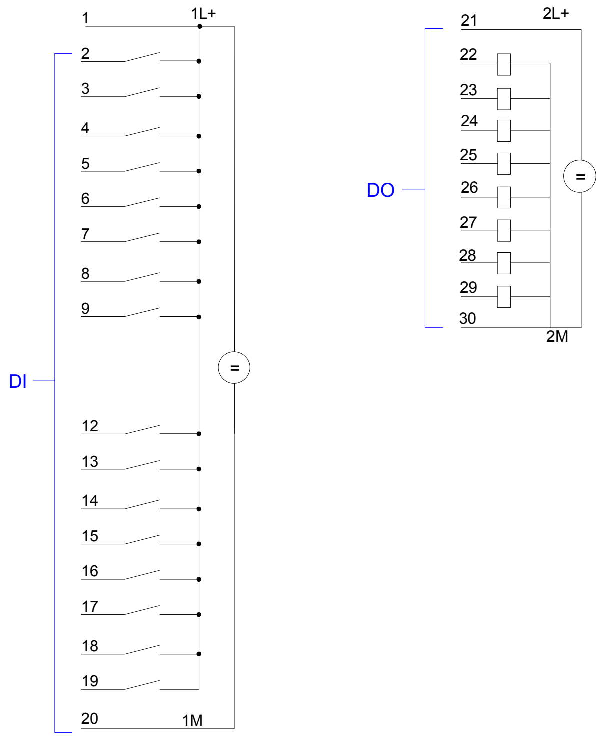
Overview CPU 312SC

The CPU 312SC has the following analog and digital in- and output ranges integrated in one casing:

• Digital Input:	16xDC 24V
• Digital Output:	8xDC 24V, 0.5A
• Technological functions:	2 Channels

Each of the digital in-/ outputs monitors its state via a LED. Via the parameterization you may assign alarm properties to every digital input. Additionally the digital inputs are parameterizable as counter.

X11:



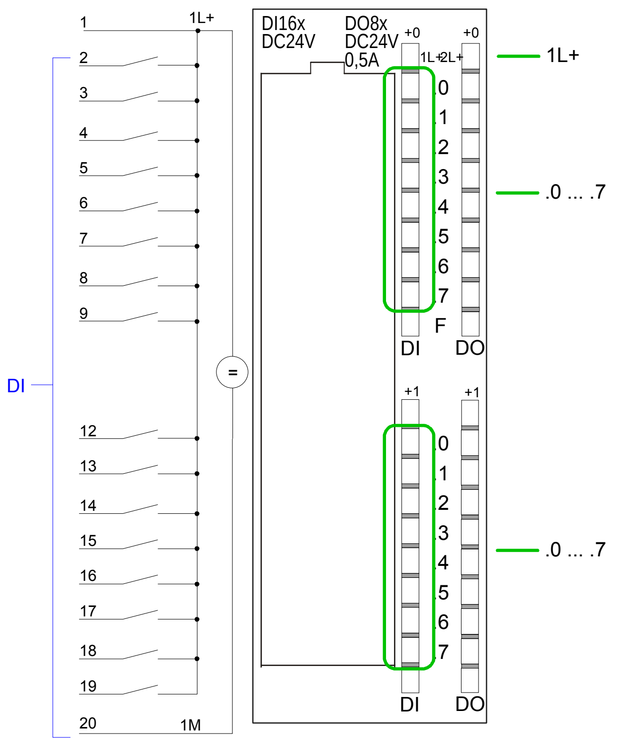
Attention!

Please take care that the voltage at an output channel always is ≤ the supply voltage via L+.

CPU 312SC: Pin assignment and status indicator

Pin	Assignment
1	Power supply +DC 24V
2	I+0.0 / Channel 0 (A)/Pulse
3	I+0.1 / Channel 0 (B)/Direction
4	I+0.2 / Channel 0 HW gate
5	I+0.3 / Channel 1 (A)/Pulse
6	I+0.4 / Channel 1 (B)/Direction
7	I+0.5 / Channel 1 HW gate
8	I+0.6
9	I+0.7
10	not connected
11	not connected
12	I+1.0
13	I+1.1
14	I+1.2
15	I+1.3
16	I+1.4 / Channel 0 Latch
17	I+1.5 / Channel 1 Latch
18	I+1.6
19	I+1.7
20	Ground 1M DI

Connection **LEDs**

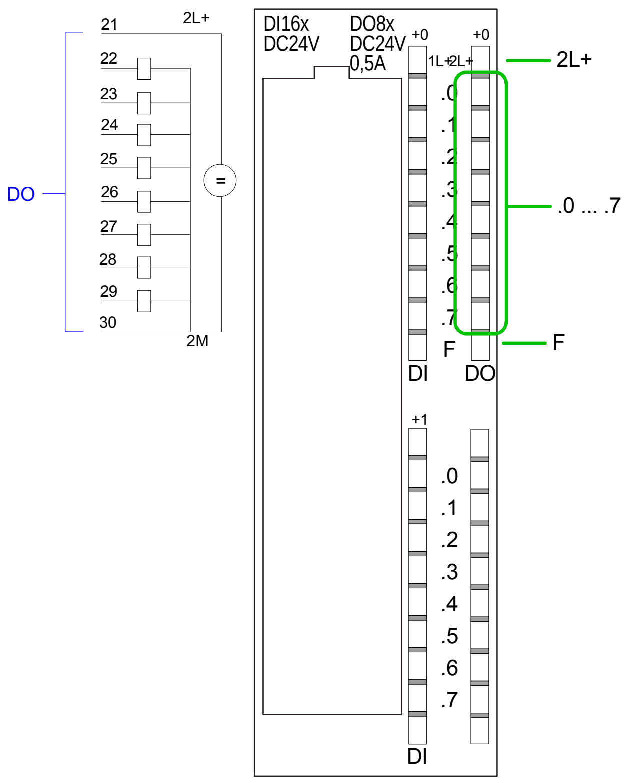


DI:		
L1+		LED (green) Supply voltage available for DI
.0 7		LEDs (green) I+0.0 to I+0.7 resp. I+1.0 to I+1.7 Starting with app. 15V the signal "1" at the input is recognized and the according LED

CPU 312SC: Pin assignment and status indicator

Pin	Assignment
21	Power supply +DC 24V
22	Q+0.0 / Channel 0 Output
23	Q+0.1 / Channel 1 Output
24	Q+0.2 / Channel 2 Output
25	Q+0.3
26	Q+0.4
27	Q+0.5
28	Q+0.6
29	Q+0.7
30	Ground 2M DO
31	not connected
32	not connected
33	not connected
34	not connected
35	not connected
36	not connected
37	not connected
38	not connected
39	not connected
40	not connected

Connection **LEDs**



DO:	
L2+	LED (green) Supply voltage available for DO
0 7	LEDs (green) Q+0.0 to Q+0.7 on at active output
F	LED (red) Overload or short circuit error

Technical Data

Order number	**312-5BE13**
Type	CPU 312SC
SPEED-Bus	-
Technical data power supply	
Power supply (rated value)	DC 24 V
Power supply (permitted range)	DC 20.4...28.8 V
Reverse polarity protection	✓
Current consumption (no-load operation)	135 mA
Current consumption (rated value)	500 mA
Inrush current	11 A
Technical data digital inputs	
Number of inputs	16
Cable length, shielded	1000 m
Cable length, unshielded	600 m
Rated load voltage	DC 24 V
Reverse polarity protection of rated load voltage	✓
Current consumption from load voltage L+ (without load)	70 mA
Rated value	DC 24 V
Input voltage for signal "0"	DC 0...5 V
Input voltage for signal "1"	DC 15...28.8 V
Input voltage hysteresis	-
Frequency range	-
Input resistance	-
Input current for signal "1"	6 mA
Connection of Two-Wire-BEROs possible	✓
Max. permissible BERO quiescent current	1.5 mA
Input delay of "0" to "1"	0.1 / 0.35 ms
Input delay of "1" to "0"	0.1 / 0.35 ms
Number of simultaneously utilizable inputs horizontal configuration	-
Number of simultaneously utilizable inputs vertical configuration	-
Input characteristic curve	IEC 61131, type 1
Initial data size	2 Byte
Technical data digital outputs	
Number of outputs	8
Cable length, shielded	1000 m
Cable length, unshielded	600 m
Rated load voltage	DC 24 V
Reverse polarity protection of rated load voltage	-
Current consumption from load voltage L+ (without load)	100 mA
Total current per group, horizontal configuration, 40°C	3 A
Total current per group, horizontal configuration, 60°C	2 A
Total current per group, vertical configuration	2 A
Output voltage signal "1" at min. current	L+ (-0.8 V)
Output voltage signal "1" at max. current	-
Output current at signal "1", rated value	0.5 A
Output current, permitted range to 40°C	5 mA to 0.6 A
Output current, permitted range to 60°C	5 mA to 0.6 A
Output current at signal "0" max. (residual current)	0.5 mA

Order number	**312-5BE13**
Output delay of "0" to "1"	-
Output delay of "1" to "0"	-
Minimum load current	-
Lamp load	5 W
Parallel switching of outputs for redundant control of a load	possible
Parallel switching of outputs for increased power	not possible
Actuation of digital input	✓
Switching frequency with resistive load	max. 2.5 kHz
Switching frequency with inductive load	max. 0.5 Hz
Switching frequency on lamp load	max. 2.5 kHz
Internal limitation of inductive shut-off voltage	L+ (-52 V)
Short-circuit protection of output	yes, electronic
Trigger level	1 A
Number of operating cycle of relay outputs	-
Switching capacity of contacts	-
Output data size	1 Byte
Technical data analog inputs	
Number of inputs	-
Cable length, shielded	-
Rated load voltage	-
Reverse polarity protection of rated load voltage	-
Current consumption from load voltage L+ (without load)	-
Voltage inputs	-
Min. input resistance (voltage range)	-
Input voltage ranges	-
Operational limit of voltage ranges	-
Basic error limit voltage ranges with SFU	-
Current inputs	-
Min. input resistance (current range)	-
Input current ranges	-
Operational limit of current ranges	-
Basic error limit current ranges with SFU	-
Resistance inputs	-
Resistance ranges	-
Operational limit of resistor ranges	-
Basic error limit	-
Resistance thermometer inputs	-
Resistance thermometer ranges	-
Operational limit of resistance thermometer ranges	-
Basic error limit thermoresistor ranges	-
Thermocouple inputs	-
Thermocouple ranges	-
Operational limit of thermocouple ranges	-
Basic error limit thermoelement ranges	-
Programmable temperature compensation	-
External temperature compensation	-
Internal temperature compensation	-
Resolution in bit	-
Measurement principle	-
Basic conversion time	-
Noise suppression for frequency	-
Initial data size	-
Technical data analog outputs	
Number of outputs	-
Cable length, shielded	-

Order number	**312-5BE13**
Rated load voltage	-
Reverse polarity protection of rated load voltage	-
Current consumption from load voltage L+ (without load)	-
Voltage output short-circuit protection	-
Voltage outputs	-
Min. load resistance (voltage range)	-
Max. capacitive load (current range)	-
Output voltage ranges	-
Operational limit of voltage ranges	-
Basic error limit voltage ranges with SFU	-
Current outputs	-
Max. in load resistance (current range)	-
Max. inductive load (current range)	-
Output current ranges	-
Operational limit of current ranges	-
Basic error limit current ranges with SFU	-
Settling time for ohmic load	-
Settling time for capacitive load	-
Settling time for inductive load	-
Resolution in bit	-
Conversion time	-
Substitute value can be applied	-
Output data size	-
Technical data counters	
Number of counters	2
Counter width	32 Bit
Maximum input frequency	10 kHz
Maximum count frequency	10 kHz
Mode incremental encoder	✓
Mode pulse / direction	✓
Mode pulse	✓
Mode frequency counter	-
Mode period measurement	-
Gate input available	✓
Latch input available	✓
Reset input available	-
Counter output available	✓
Load and working memory	
Load memory, integrated	512 KB
Load memory, maximum	512 KB
Work memory, integrated	64 KB
Work memory, maximal	512 KB
Memory divided in 50% program / 50% data	✓
Memory card slot	MMC-Card with max. 1 GB
Hardware configuration	
Racks, max.	1
Modules per rack, max.	8
Number of integrated DP master	0
Number of DP master via CP	4
Operable function modules	8
Operable communication modules PtP	8
Operable communication modules LAN	8
Status information, alarms, diagnostics	
Status display	yes
Interrupts	yes
Process alarm	yes

Order number	**312-5BE13**
Diagnostic interrupt	yes
Diagnostic functions	no
Diagnostics information read-out	possible
Supply voltage display	green LED
Group error display	red SF LED
Channel error display	red LED per group
Command processing times	
Bit instructions, min.	0.02 µs
Word instruction, min.	0.02 µs
Double integer arithmetic, min.	0.02 µs
Floating-point arithmetic, min.	0.12 µs
Timers/Counters and their retentive characteristics	
Number of S7 counters	512
Number of S7 times	512
Data range and retentive characteristic	
Number of flags	8192 Byte
Number of data blocks	4095
Max. data blocks size	64 KB
Max. local data size per execution level	510 Byte
Blocks	
Number of OBs	15
Number of FBs	2048
Number of FCs	2048
Maximum nesting depth per priority class	8
Maximum nesting depth additional within an error OB	4
Time	
Real-time clock buffered	✓
Clock buffered period (min.)	6 W
Accuracy (max. deviation per day)	10 s
Number of operating hours counter	8
Clock synchronization	✓
Synchronization via MPI	Master/Slave
Synchronization via Ethernet (NTP)	no
Address areas (I/O)	
Input I/O address area	1024 Byte
Output I/O address area	1024 Byte
Input process image maximal	128 Byte
Output process image maximal	128 Byte
Digital inputs	272
Digital outputs	264
Digital inputs central	272
Digital outputs central	264
Integrated digital inputs	16
Integrated digital outputs	8
Analog inputs	64
Analog outputs	64
Analog inputs, central	64
Analog outputs, central	64
Integrated analog inputs	0
Integrated analog outputs	0
Communication functions	
PG/OP channel	✓
Global data communication	✓
Number of GD circuits, max.	4
Size of GD packets, max.	22 Byte

Order number	**312-5BE13**
S7 basic communication	✓
S7 basic communication, user data per job	76 Byte
S7 communication	✓
S7 communication as server	✓
S7 communication as client	-
S7 communication, user data per job	160 Byte
Number of connections, max.	32
Functionality Sub-D interfaces	
Type	X2
Type of interface	RS485
Connector	Sub-D, 9-pin, female
Electrically isolated	-
MPI	✓
MP²I (MPI/RS232)	-
DP master	-
DP slave	-
Point-to-point interface	-
Type	X3
Type of interface	RS485
Connector	Sub-D, 9-pin, female
Electrically isolated	✓
MPI	-
MP²I (MPI/RS232)	-
DP master	-
DP slave	-
Point-to-point interface	✓
CAN	-
Functionality PROFIBUS master	
PG/OP channel	-
Routing	-
S7 basic communication	-
S7 communication	-
S7 communication as server	-
S7 communication as client	-
Equidistance support	-
Isochronous mode	-
SYNC/FREEZE	-
Activation/deactivation of DP slaves	-
Direct data exchange (slave-to-slave communication)	-
DPV1	-
Transmission speed, min.	-
Transmission speed, max.	-
Number of DP slaves, max.	-
Address range inputs, max.	-
Address range outputs, max.	-
User data inputs per slave, max.	-
User data outputs per slave, max.	-
Functionality PROFIBUS slave	
PG/OP channel	-
Routing	-
S7 communication	-
S7 communication as server	-
S7 communication as client	-
Direct data exchange (slave-to-slave communication)	-

Order number	**312-5BE13**
DPV1	-
Transmission speed, min.	-
Transmission speed, max.	-
Automatic detection of transmission speed	-
Transfer memory inputs, max.	-
Transfer memory outputs, max.	-
Address areas, max.	-
User data per address area, max.	-
Point-to-point communication	
PtP communication	✓
Interface isolated	✓
RS232 interface	-
RS422 interface	-
RS485 interface	✓
Connector	Sub-D, 9-pin, female
Transmission speed, min.	150 bit/s
Transmission speed, max.	115.5 kbit/s
Cable length, max.	500 m
Point-to-point protocol	
ASCII protocol	✓
STX/ETX protocol	✓
3964(R) protocol	✓
RK512 protocol	-
USS master protocol	✓
Modbus master protocol	✓
Modbus slave protocol	-
Special protocols	-
Functionality RJ45 interfaces	
Type	X5
Type of interface	Ethernet 10/100 MBit
Connector	RJ45
Electrically isolated	✓
PG/OP channel	✓
Productive connections	-
Mechanical data	
Dimensions (WxHxD)	80 x 125 x 120 mm
Weight	410 g
Environmental conditions	
Operating temperature	0 °C to 60 °C
Storage temperature	-25 °C to 70 °C
Certifications	
UL508 certification	in preparation

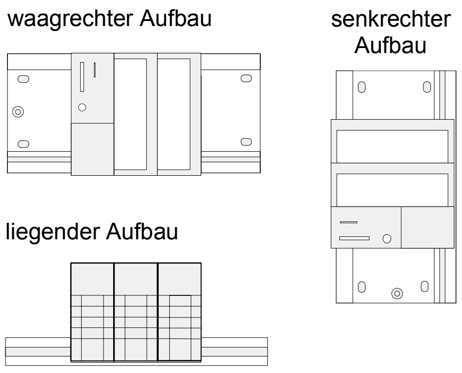
Chapter 4 Deployment CPU 312SC

Overview

This chapter describes the deployment of the CPU 312SC with SPEED7 technology in the System 300. The description refers directly to the CPU and to the employment in connection with peripheral modules that are mounted on a profile rail together with the CPU at standard bus.

Content

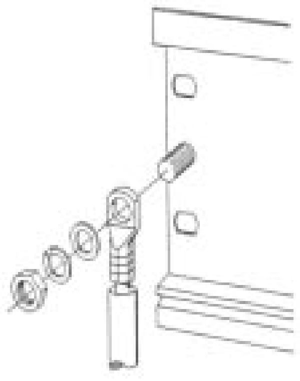
Installation

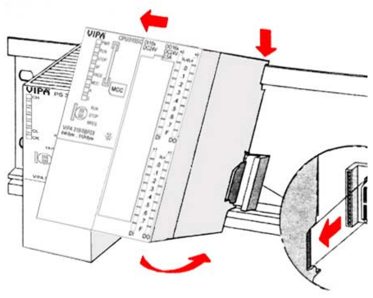


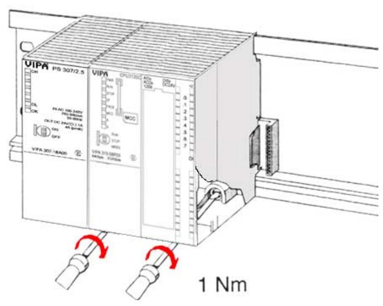
Assembly possibilities

Please regard the allowed environment temperatures:

• horizontal assembly:	from 0 to 60°C
• vertical assembly:	from 0 to 40°C
• lying assembly:	from 0 to 40°C







Approach

- Bolt the profile rail with the background (screw size: M6), so that you still have minimum 65mm space above and 40mm below the profile rail.
- If the background is a grounded metal or device plate, please look for a low-impedance connection between profile rail and background.
- Connect the profile rail with the protected earth conductor. For this purpose there is a bolt with M6-thread.
- The minimum cross-section of the cable to the protected earth conductor has to be 10mm^2.
- Stick the power supply to the profile rail and pull it to the left side to the grounding bolt of the profile rail.
- Fix the power supply by screwing.
- Take a backplane bus connector and click it at the CPU from the backside like shown in the picture.
- Stick the CPU to the profile rail right from the power supply and pull it to the power supply.

- Click the CPU downwards and bolt it like shown.
- Repeat this procedure with the peripheral modules, by clicking a backplane bus connector, stick the module right from the modules you've already fixed, click it downwards and connect it with the backplane bus connector of the last module and bolt it.

Danger!

- The power supplies must be released before installation and repair tasks, i.e. before handling with the power supply or with the cabling you must disconnect current/voltage (pull plug, at fixed connection switch off the concerning fuse)!
- Installation and modifications only by properly trained personnel!

Start-up behavior

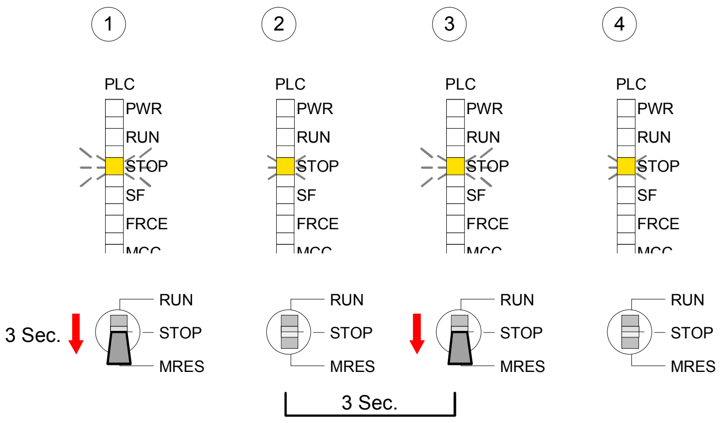
Turn on power supply

After the power supply has been switched on, the CPU changes to the operating mode the operating mode lever shows.

Now you may transfer your project to the CPU via MPI from your configuration tool res. plug in a MMC with your project and run an Overall reset.

Overall reset

The following picture shows the approach once more:



Note!
The transfer of the application program from the MMC into the CPU takes always place after an Overall reset!

Delivery status

When the CPU is delivered it has been reset. After a STOP→RUN transition the CPU switches to RUN without program.

Boot procedure with valid data in the CPU

The CPU switches to RUN with the program stored in the battery buffered RAM.

Boot procedure with empty battery

The accumulator/battery is automatically loaded via the integrated power supply and guarantees a buffer for max. 30 days. If this time is exceeded, the battery may be totally discharged. This means that the battery buffered RAM is deleted.

In this state, the CPU executes an overall reset. If a MMC is plugged, program code and data blocks are transferred from the MMC into the work memory of the CPU.

If no MMC is plugged, the CPU transfers permanent stored "protected" blocks into the work memory if available.

Information about storing protected blocks in the CPU is to find in this chapter at "Extended Know-how protection".

Depending on the position of the RUN/STOP lever, the CPU switches to RUN res. remains in STOP.

This event is stored in the diagnostic buffer as: "Start overall reset automatically (unbuffered POWER_ON)".

Addressing

Overview

To provide specific addressing of the integrated in-/output periphery and the installed peripheral modules, certain addresses must be allocated in the CPU.

At the start-up of the CPU, this assigns automatically peripheral addresses for digital in-/output modules starting with 0 and ascending depending on the slot location.

If no hardware project engineering is available, the CPU stores at the addressing analog modules to even addresses starting with 256.

The integrated in-/output periphery is also allocated to the address area of the CPU. More may be found at "Address assignment".

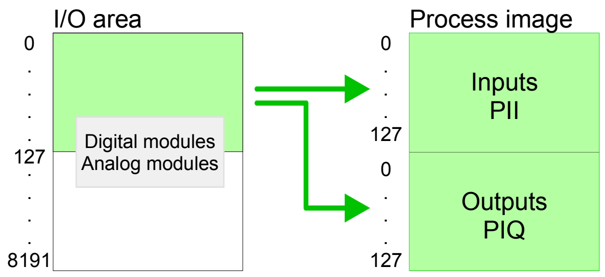
Addressing Backplane bus I/O devices

The CPU provides an I/O area (address 0 ... 8191) and a process image of the in- and outputs (each address 0 ... 127).

The process image stores the signal states of the lower address (0 ... 127) additionally in a separate memory area.

The process image this divided into two parts:

- process image to the inputs (PII)
- process image to the outputs (PIQ)



The process image is updated automatically when a cycle has been completed.

Max. number of pluggable modules

Maximally 8 modules may be addressed by the CPU 312SC. The extension by means of extension racks is not possible.

Define addresses by hardware configuration

You may access the modules with read res. write accesses to the peripheral bytes or the process image.

To define addresses a hardware configuration may be used. For this, click on the properties of the according module and set the wanted address.

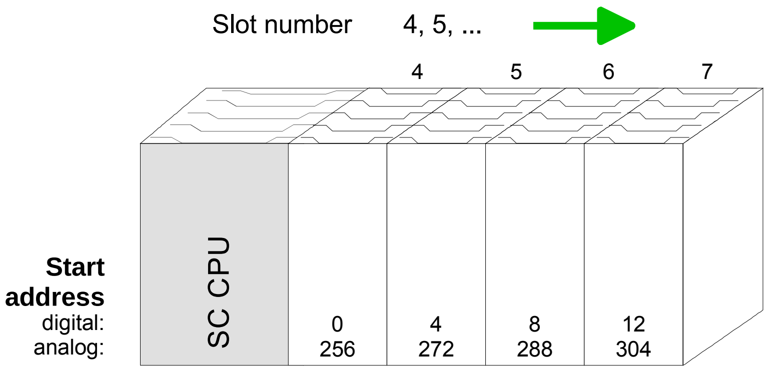
Automatic addressing

If you do not like to use a hardware configuration, an automatic addressing comes into force.

At the automatic address allocation DIOs occupy depending on the slot location always 4byte and AIOs, FMs, CPs always 16byte at the bus.

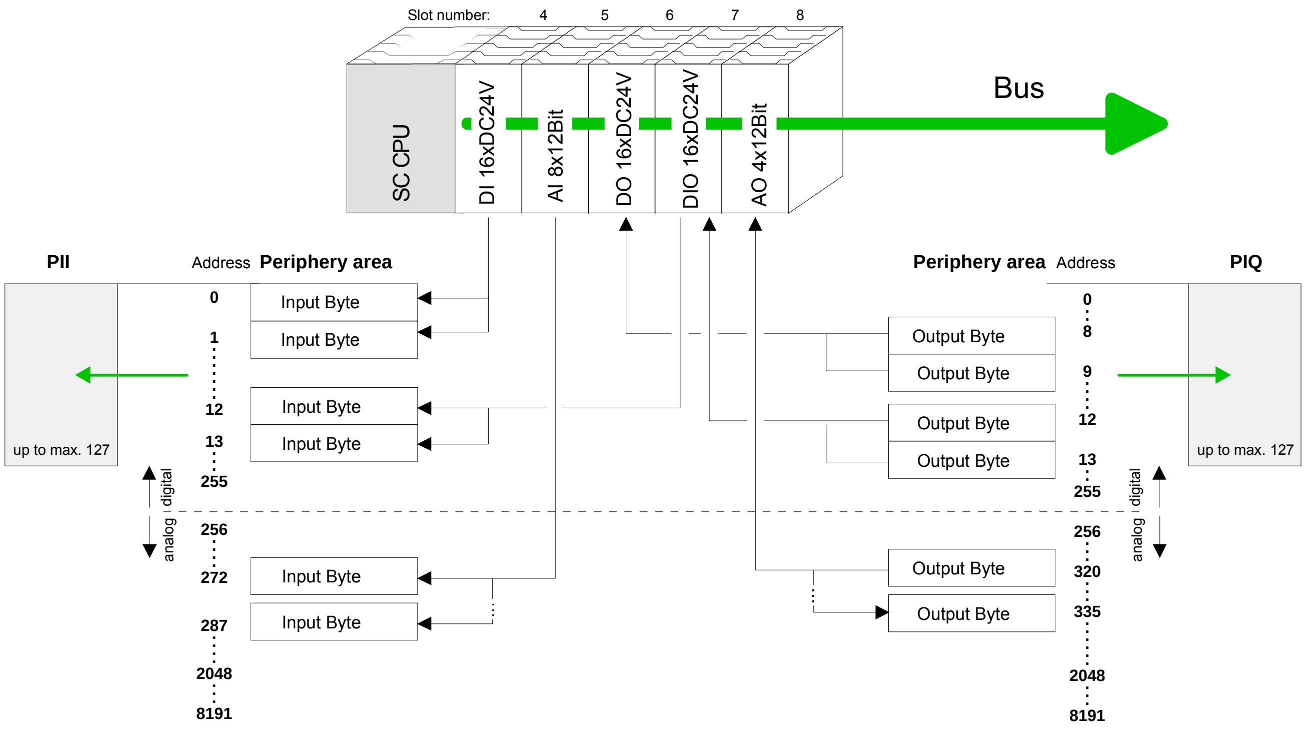
Depending on the slot location the start address from where on the according module is stored in the address range is calculated with the following formulas:

DIOs:	Start address = $4 \cdot (\text{slot} - 4)$
AIOs, FMs, CPs:	Start address = $16 \cdot (\text{slot} - 4) + 256$



Example for automatic address allocation

The following sample shows the functionality of the automatic address allocation:



Address assignment

Input range

Sub module	Default-Address	Access	Assignment
DI10/DO6	124	Byte	Digital Input I+0.0 ... I+0.7
	125	Byte	Digital Input I+1.0 ... I+1.7
Counter	768	DInt	Channel 0: Count value / Frequency value
	772	DInt	Channel 1: Count value / Frequency value
	776	DInt	reserved
	780	DInt	reserved

Output range

Sub module	Default-Address	Access	Assignment
DI10/DO6	124	Byte	Digital Output Q+0.0 ... Q+0.7
Counter	768	DWord	reserved
	772	DWord	reserved
	776	DWord	reserved
	780	DWord	reserved

Initialization Ethernet PG/OP channel

Overview

The CPU 312SC has an integrated Ethernet PG/OP channel. This channel allows you to program and remote control your CPU with up to 4 connections.

The PG/OP channel also gives you access to the internal web page that contains information about firmware version, connected I/O devices, current cycle times etc.

For online access to the CPU via Ethernet PG/OP channel valid IP address parameters have to be assigned to this by means of the Siemens SIMATIC manager. This is called "Initialization".

Possibilities for Initialization

There are the following possibilities for assignment of IP address parameters (initialization):

- PLC functions with *Assign Ethernet address*
- Hardware project engineering with CP (Minimal project)

Requirements

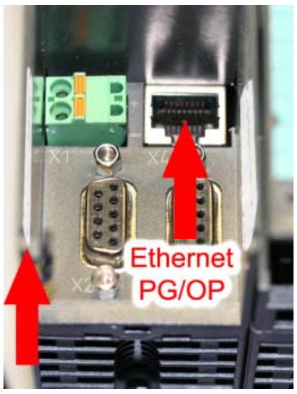
For the hardware configuration the following software is necessary:

- SIMATIC Manager from Siemens up to V. 5.1 and SIMATIC NET or
- SIMATIC Manager from Siemens up to V. 5.2 and SP1

Initialization via PLC functions

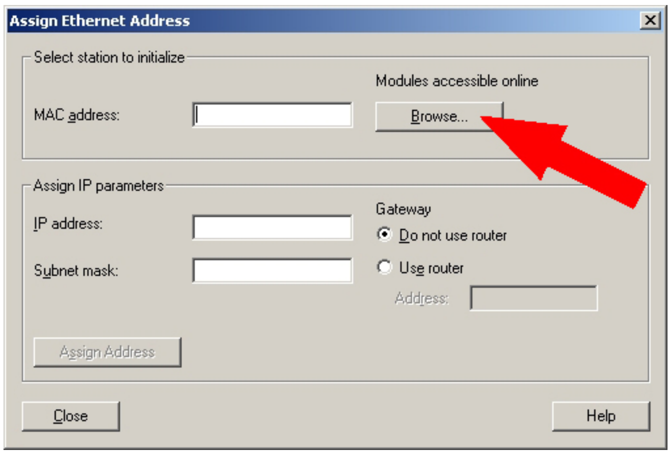
The initialization takes place after the following proceeding:

- Determine the current Ethernet (MAC) address of your Ethernet PG/OP channel. This always may be found as address under the front flap of the CPU on a sticker on the left side.

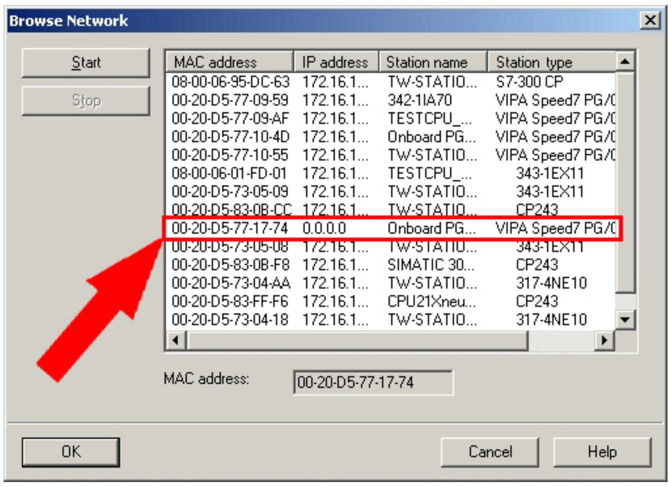


Ethernet address
Ethernet PG/OP

- Establish a network connection between Ethernet PG/OP channel of the CPU and PC.
- Start the Siemens SIMATIC manager at the PC.
- Set via **Options** > *Set PG/PC Interface* the Access Path to "TCP/IP -> Network card .... Protocol RFC 1006".
- Open with **PLC** > *Assign Ethernet Address* the dialog window for "initialization" of a station.



- Use the [Browse] button to determine the CPU components via MAC address.
  As long as the Ethernet PG/OP channel was not initialized yet, this owns the IP address 0.0.0.0 and the station name "Onboard PG/OP".



MAC address	IP address	Station name	Station type
08-00-06-95-DC-63	172.16.1...	TW-STATIO...	S7-300 CP
00-20-D5-77-09-59	172.16.1...	342-1IA70	VIPA Speed7 PG/(
00-20-D5-77-09-AF	172.16.1...	TESTCPU_...	VIPA Speed7 PG/(
00-20-D5-77-10-4D	172.16.1...	Onboard PG...	VIPA Speed7 PG/(
00-20-D5-77-10-55	172.16.1...	TW-STATIO...	VIPA Speed7 PG/(
08-00-06-01-FD-01	172.16.1...	TESTCPU_...	343-1EX11
00-20-D5-73-05-09	172.16.1...	TW-STATIO...	343-1EX11
00-20-D5-83-0B-CC	172.16.1...	TW-STATIO...	CP243
00-20-D5-77-17-74	0.0.0.0	Onboard PG...	VIPA Speed7 PG/(
00-20-D5-73-05-08	172.16.1...	TW-STATIO...	343-1EX11
00-20-D5-83-0B-F8	172.16.1...	SIMATIC 30...	CP243
00-20-D5-73-04-AA	172.16.1...	TW-STATIO...	317-4NE10
00-20-D5-83-FF-F6	172.16.1...	CPU21Xneu...	CP243
00-20-D5-73-04-18	172.16.1...	TW-STATIO...	317-4NE10

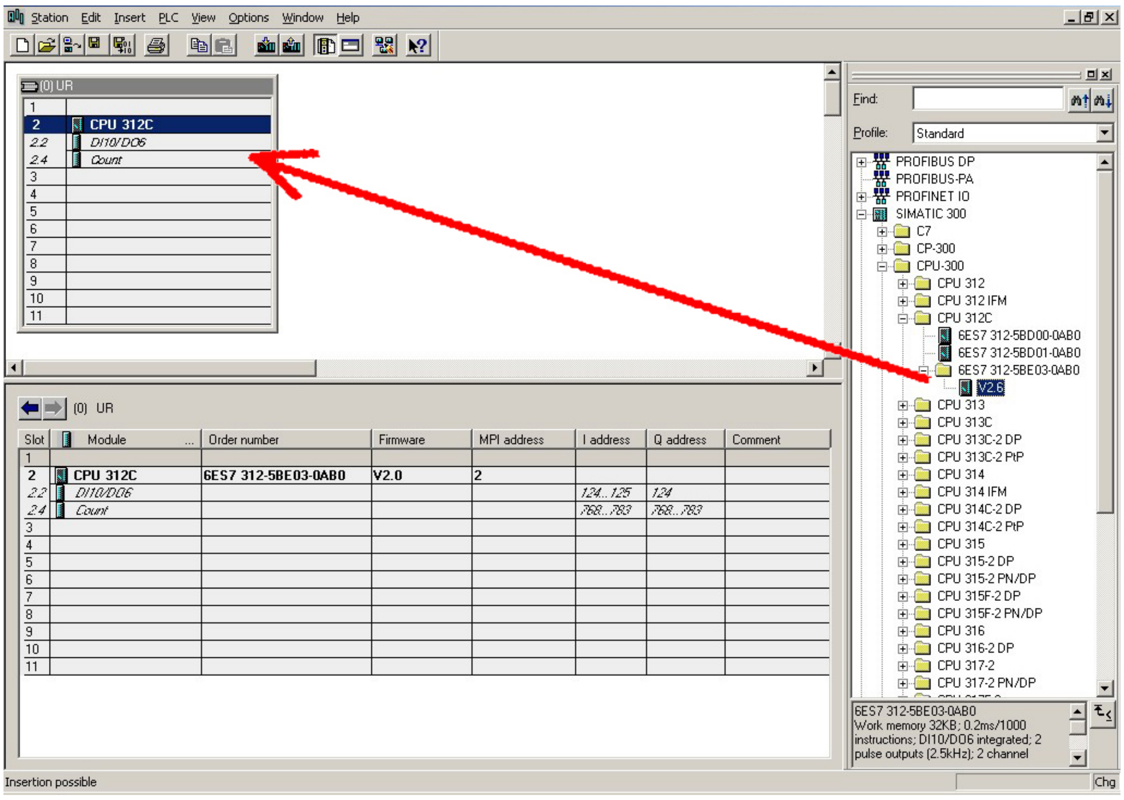
MAC address: 00-20-D5-77-17-74

- Choose the determined module and click to [OK].
- Set the IP configuration by entering IP address, subnet mask and net transition. In addition an IP address may be received from a DHCP server. For this depending upon the selected option the MAC address, device name or the Client ID, which may be entered here, is to be conveyed to the DHCP server. The Client-ID is a character sequence from maximally 63 characters.
  Here the following indications may be used: Dash "-", 0-9, A-z, A-Z
- Confirm your settings by button [Assign Address]

Direct after the assignment the Ethernet PG/OP channel may be reached by the Siemens SIMATIC manager by means of these IP address parameters and the *Access Path* "TCP/IP -> Network card .... Protocol RFC 1006".

Initialization via minimal project

- Establish a network connection between Ethernet PG/OP channel of the CPU and PC.
- Start the SIMATIC Manager from Siemens and create a new project.
- Add a new System 300 station via **Insert** > *Station* > *SIMATIC 300-Station*.
- Activate the station "SIMATIC 300" and open the hardware configurator by clicking on "Hardware".
- Engineer a rack (SIMATIC 300 \ Rack-300 \ Profile rail)
- Place the Siemens CPU 312C with the order no. 6ES7 312-5BE03-0AB0 V2.6 from the hardware catalog. This may be found at SIMATIC 300 \ CPU 300 \ CPU 312C.
- Include the CP 343-1EX11 at slot 4 (SIMATIC 300 \ CP 300 \ Industrial Ethernet \ CP 343-1).



(0) UR

Slot	Module	Order number	Firmware	MPI address	I address	Q address	Comment
1							
2	CPU 312C	6ES7 312-5BE03-0AB0	V2.0	2			
2.2	DI10/DO6				124...125	124	
2.4	Count				768...783	768...783	
3							
4							
5							
6							
7							
8							
9							
10							
11							

6ES7 312-5BE03-0AB0
Work memory 32KB; 0.2ms/1000 instructions; DI10/DO6 integrated; 2 pulse outputs (2.5kHz); 2 channel

- Type the wanted IP address and subnet mask into the dialog window of "Properties" of the CP 343-1 and connect the CP with "Ethernet".
- Save and compile your project.
- Transfer your project via MPI or MMC into your CPU. More information about transfer methods may be found in the chapter "Project transfer".

Direct after the assignment the Ethernet PG/OP channel may be reached by the Siemens SIMATIC manager by means of these IP address parameters and the *Access Path* "TCP/IP -> Network card .... Protocol RFC 1006".

Access to the internal web page

Access to the web page

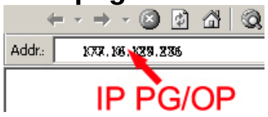
The Ethernet PG/OP channel provides a web page that you may access via an Internet browser by its IP address. The web page contains information about firmware versions, current cycle times etc.

The current content of the web page is stored on MMC by means of the MMC-Cmd WEBPAGE. More information may be found at "MMC-Cmd - Auto commands".

Requirements

A PG/OP channel connection should be established between PC with Internet browser and CPU 312SC. This may be tested by *Ping* to the IP address of the Ethernet PG/OP channel.

Web page



The access takes place via the IP address of the Ethernet PG/OP channel. The web page only serves for information output. The monitored values are not alterable.

`CPU WITH ETHERNET PG/OP`	
`Slot 100`	
`  VIPA 312-5BE13 V3.5.4 Px000135.pkg,` `  SERIALNUMBER 02118`	Order no., firmware vers., package, serial no.
`  SUPPORTDATA: PRODUCT V3290, HARDWARE ...`	Information for support
`  OnBoardEthernet : MacAddress : 0020d5771524,` `  IP-Address : , SubnetMask : , Gateway :`	Ethernet PG/OP: Addresses
`  Cpu state : RUN`	CPU state
`  FunctionRS485 X2 : MPI`	RS485 function X2
`  FunctionRS485 X3 : PtP`	RS485 function X3
`  Cycletime [microseconds] : min=17000` `  cur=17000 ave=17000 max=17000`	CPU cycle time: min= minimal, cur= current ave= average, max= maximal
`  MCC-Trial-Time: 70:23`	Remaining time for deactivation of the expansion memory if MCC is removed.
	Additional CPU components:
`Slot 202`	Slot 202 (Digital I/Os):
`  VIPA DI16/DO8 V3.2.9,SUPPORTDATA:PRODUCT...`	Name, firmware version, module type
`  SUPPORTDATA: PRODUCT V3290, Module Type ...`	Information for support
`  Address Input 124...125`	Configured input base addresses
`  Address Output 124...124`	Configured output base addresses
`Slot 204`	Slot 204 (Counter)
`  VIPA 2 COUNTERS V3.2.9,`	Name, firmware version, module type
`  SUPPORTDATA: PRODUCT V3290, Module Type ...`	Information for support
`  Address Input 768...783`	Configured input base addresses
`  Address Output 768...783`	Configured output base addresses
`Standard Bus` `8 Bit Mode`	*Modules at standard bus*

Project engineering as CPU 312C

Overview

The project engineering of the CPU 312SC takes place at the Siemens hardware configurator and is divided into the following parts:

- Project engineering CPU 312SC as CPU 312C from Siemens (6ES7 312-5BE03-0AB0 V2.6).
- Project engineering of the plugged modules at the bus.
- Project engineering Ethernet PG/OP channel always as last module as CP 343-1 (343-1EX11-0XE0)

Note!
Please consider that this SPEED7-CPU has 4 ACCUs. After an arithmetic operation (+I, -I, *I, /I, +D, -D, *D, /D, MOD, +R, -R, *R, /R) the content of ACCU 3 and ACCU 4 is loaded into ACCU 3 and 2.
This may cause conflicts in applications that presume an unmodified ACCU2.

For more information may be found in the manual "VIPA Operation list SPEED7" at "Differences between SPEED7 and 300V programming".

Requirements

The hardware configurator is a part of the Siemens SIMATIC Manager. It serves the project engineering. The modules that may be configured here are listed in the hardware catalog. If necessary you have to update the hardware catalog with **Options** > *Update Catalog*.

For the project engineering a thorough knowledge of the Siemens SIMATIC Manager and the hardware configurator from Siemens are required and assumed!

Fast introduction

To be compatible with the Siemens hardware configurator the following steps should be executed:

Slot	Module
1	
2	**CPU 312C**
2.2	*DI10/DO6*
2.4	*Count*
3	
Modules at the bus	
	343-1EX11 (Ethernet-PG/OP)

- Start the hardware configurator from Siemens.
- Configure CPU 312C (6ES7 312-5BE03-0AB0 V2.6) from Siemens.
- Starting with slot 4, place the System 300 modules in the plugged sequence.
- For the internal Ethernet PG/OP channel that is integrated to every SC CPU, you have to configure a Siemens CP 343-1 (343-1EX11) always as last module. Let at *options* the attitude "Save configuration data on the CPU" activated!

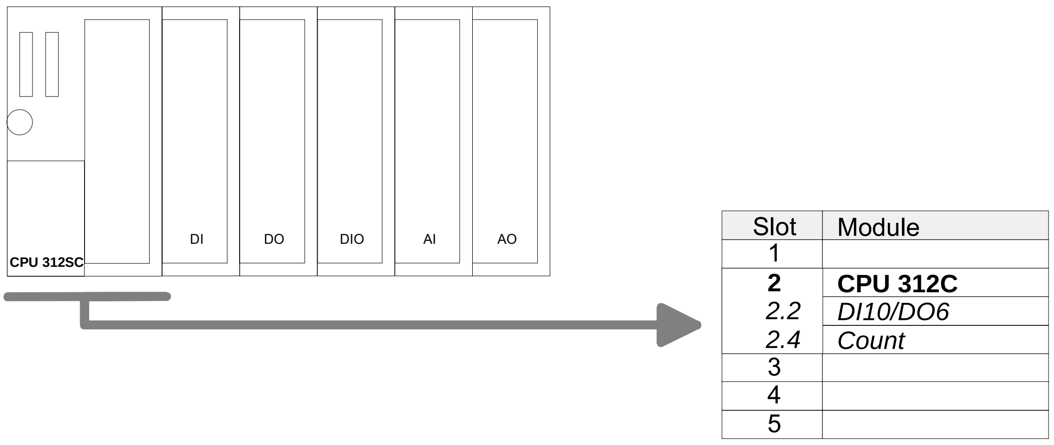
Steps of the project engineering

The project engineering is separated into 3 parts:

- Project engineering of the CPU
- Project engineering of the plugged modules
- Project engineering of the PG/OP channel

Project engineering CPU as CPU 312C

- Start the hardware configurator from Siemens with a new project and insert a profile rail from the hardware catalog.
- Place the following Siemens CPU at slot 2:
  CPU 312C (6ES7 312-5BE03-0AB0 V2.6)

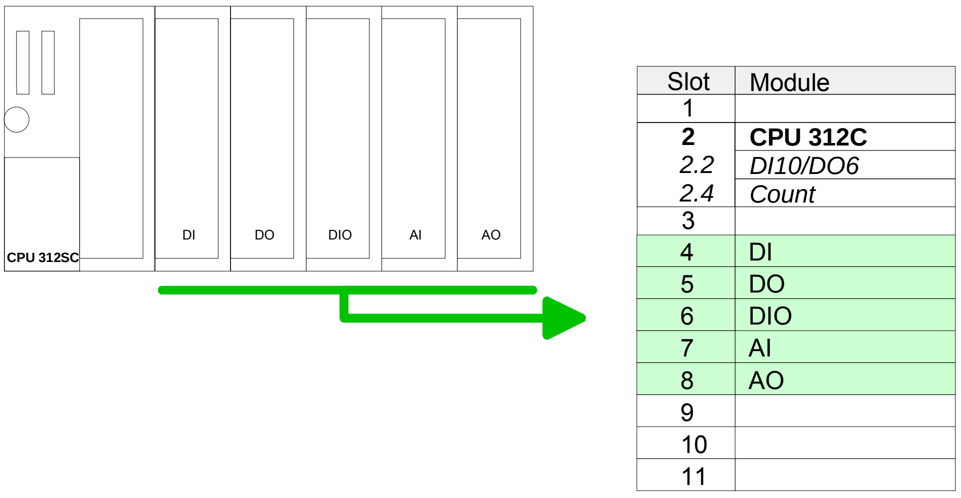


Slot	Module
1	
2	**CPU 312C**
2.2	*DI10/DO6*
2.4	*Count*
3	
4	
5	

Configuring of the modules at the bus

The modules at the bus are configured with the following approach:

- Include your System 300 modules at the bus in the plugged sequence starting with slot 4.
- Parameterize the CPU res. the modules where appropriate. The parameter window opens by a double click on the according module.

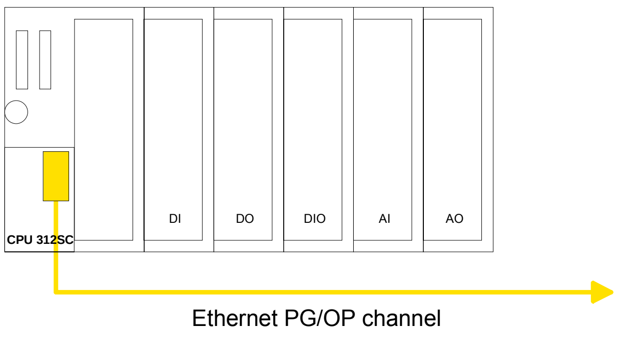


Slot	Module
1	
2	**CPU 312C**
2.2	*DI10/DO6*
2.4	*Count*
3	
4	DI
5	DO
6	DIO
7	AI
8	AO
9	
10	
11	

Project engineering of Ethernet PG/OP channel as 343-1EX11

For the internal Ethernet PG/OP channel you have to configure a Siemens CP 343-1 (343-1EX11) always as last module. This may be found at the hardware catalog at SIMATIC 300 \ CP 300 \ Industrial Ethernet \ CP 343-1 \ 6GK7 343-1EX11 0XE0.

Let with the CP343-1 at *options* the attitude "Save configuration data on the CPU" activated!



Slot	Module
1	
2	**CPU 312C**
2.2	*DI10/DO6*
2.4	*Count*
3	
4	DI
5	DO
6	DIO
7	AI
8	AO
9	343-1EX11
10	
11	

Set IP parameters

Open the property window via double-click on the CP 343-1EX11. Enter "General" and click at [Properties]. Type in the *IP address*, *subnet mask* and *gateway* for the CP and select the wanted *subnet*.

Maximum 8 modules addressable

Maximally 8 modules may be addressed by the CPU 312SC. The extension by means of extension racks is not possible.

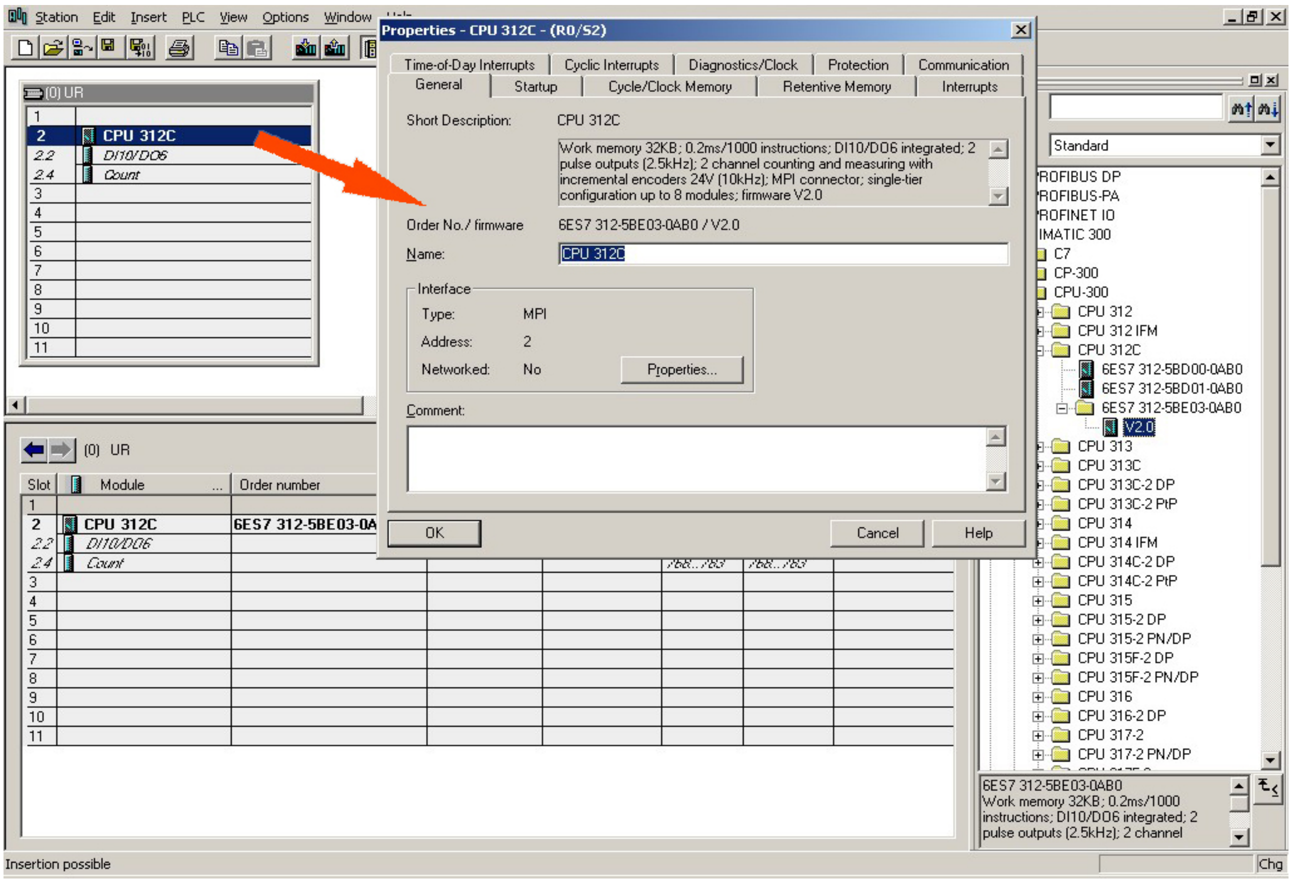
CPU parameterization

Overview

Since the CPU 312SC of VIPA is to be configured as Siemens CPU 312C in the Siemens hardware configurator, the parameters of the CPU 312SC may be set with "Object properties" during hardware configuration.

Via a double-click on the CPU 312C the parameter window may be accessed.

Using the registers you get access to all parameters of the CPU.



(0) UR

1		
2	CPU 312C	
2.2	DI10/DO6	
2.4	Count	
3		
4		
5		
6		
7		
8		
9		
10		
11		

Short Description: CPU 312C

Work memory 32KB; 0.2ms/1000 instructions; DI10/DO6 integrated; 2 pulse outputs (2.5kHz); 2 channel counting and measuring with incremental encoders 24V (10kHz); MPI connector; single-tier configuration up to 8 modules; firmware V2.0

Order No./ firmware: 6ES7 312-5BE03-0AB0 / V2.0

Name: CPU 312C

Interface

Type: MPI

Address: 2

Networked: No

Comment:

(0) UR

Slot	Module	Order number
1		
2	CPU 312C	6ES7 312-5BE03-0A
2.2	DI10/DO6	
2.4	Count	
3		
4		
5		
6		
7		
8		
9		
10		
11		

ROFIBUS DP
ROFIBUS-PA
ROFINET IO
IMATIC 300
C7
CP-300
CPU-300
CPU 312
CPU 312 IFM
CPU 312C
6ES7 312-5BD00-0AB0
6ES7 312-5BD01-0AB0
6ES7 312-5BE03-0AB0
V2.0
CPU 313
CPU 313C
CPU 313C-2 DP
CPU 313C-2 PtP
CPU 314
CPU 314 IFM
CPU 314C-2 DP
CPU 314C-2 PtP
CPU 315
CPU 315-2 DP
CPU 315-2 PN/DP
CPU 315F-2 DP
CPU 315F-2 PN/DP
CPU 316
CPU 316-2 DP
CPU 317-2
CPU 317-2 PN/DP

6ES7 312-5BE03-0AB0
Work memory 32KB; 0.2ms/1000 instructions; DI10/DO6 integrated; 2 pulse outputs (2.5kHz); 2 channel

Note!

A description of the parameters of the sub module *DI10/DO6* and *Counter* may be found at chapter "Deployment I/O periphery".

Supported parameters

The CPU does not evaluate all parameters that may be set at the hardware configuration.

The following parameters are supported at this time:

General

Short description	Since the CPU 312SC is configured as CPU 312C from Siemens, here the short description CPU 312C stands.
Order No. / Firmware	Order number and firmware are identical to the details in the "Hardware catalog" window.
Name	The *Name* field provides a short description of the module, which you can change to meet your requirements. If you change the description, the new description appears in the SIMATIC Manager.
Interface	Here the address of the MPI interface stands.
Properties	Click the "Properties" button to change the properties of the MPI interface.
Comment	In this field information about the module may be entered.

Startup

Startup when expected/actual configuration differ	If the checkbox for "Startup when expected/actual configuration differ" is *deselected* and at least one module is not located at its configured slot or if another type of module is inserted there instead, then the CPU switches to STOP mode. If the checkbox for "Startup when expected/actual configuration differ" is *selected*, then the CPU starts even if there are modules are not located in their configured slots of if another type of module is inserted there instead, such as during an initial system start-up.
Monitoring Time for Ready message by modules [100ms]	This operation specifies the maximum time for the ready message of all configured modules after PowerON. If the modules do not send a ready message to the CPU by the time the monitoring time has expired, the actual configuration becomes unequal to the preset configuration.
Monitoring Time for Transfer of parameters to modules [100ms]	The maximum time for the transfer of parameters to parameterizable modules. If not all of the modules have been assigned parameters by the time this monitoring time has expired, the actual configuration becomes unequal to the preset configuration.

Cycle/Clock memory

Scan Cycle Monitoring Time

Here the scan cycle monitoring time in milliseconds may be set. If the scan cycle time exceeds the scan cycle monitoring time, the CPU enters the STOP mode. Possible reasons for exceeding the time are:

- Communication processes
- a series of interrupt events
- an error in the CPU program

Scan Cycle Load from Communication

Using this parameter you can control the duration of communication processes, which always extend the scan cycle time so it does not exceed a specified length.

If there are no additional asynchronous events, the scan cycle time of OB1 is increased by following factor:

$$\frac{100}{100 - \text{cycle load from communication } \%}$$

If the cycle load from communication is set to 50%, the scan cycle time of OB 1 can be doubled. At the same time, the scan cycle time of OB 1 is still being influenced by asynchronous events (e.g. process interrupts) as well.

OB85-Call up at I/O Access Error

The preset reaction of the CPU may be changed to an I/O access error that occurs during the update of the process image by the system.

The CPU 312SC is preset such that OB 85 is not called if an I/O access error occurs and no entry is made in the diagnostic buffer either.

Clock Memory

Activate the check box if you want to use clock memory and enter the number of the memory byte.

Note!
The selected memory byte cannot be used for temporary data storage.

Retentive Memory

Number of Memory Bytes from MB0

Enter the number of retentive memory bytes from memory byte 0 onwards.

Number of S7 Timers from T0

Enter the number of retentive S7 timers from T0 onwards. Each S7 timer occupies 2 bytes.

Number of S7 Counters from C0

Enter the number of retentive S7 counter from C0 onwards.

Interrupts

Hardware Interrupts	Currently, the default priority may not be modified.

Time-of-Day interrupts

Priority	The priority may not be modified.
Active	Activate the check box of the time-of-day interrupt OBs if these are to be automatically started on complete restart.
Execution	Select how often the interrupts are to be triggered. Intervals ranging from every minute to yearly are available. The intervals apply to the settings made for *start date* and *time*.
Start date / Time	Enter date and time of the first execution of the time-of-day interrupt.
Process image partition	Is not supported.

Cyclic interrupts

Priority	The preset priority may not be modified.
Execution	Enter the time intervals in ms, in which the watchdog interrupt OBs should be processed. The start time for the clock is when the operating mode switch is moved from STOP to RUN.
Phase Offset	Not adjustable.
Process image partition	Is not supported.

Protection

Level of protection

Here 1 of 3 protection levels may be set to protect the CPU from unauthorized access.

Protection level 1 (default setting):

- No password adjustable, no restrictions

Protection level 2 with password:

- Authorized users: read and write access
- Unauthorized user: read access only

Protection level 3:

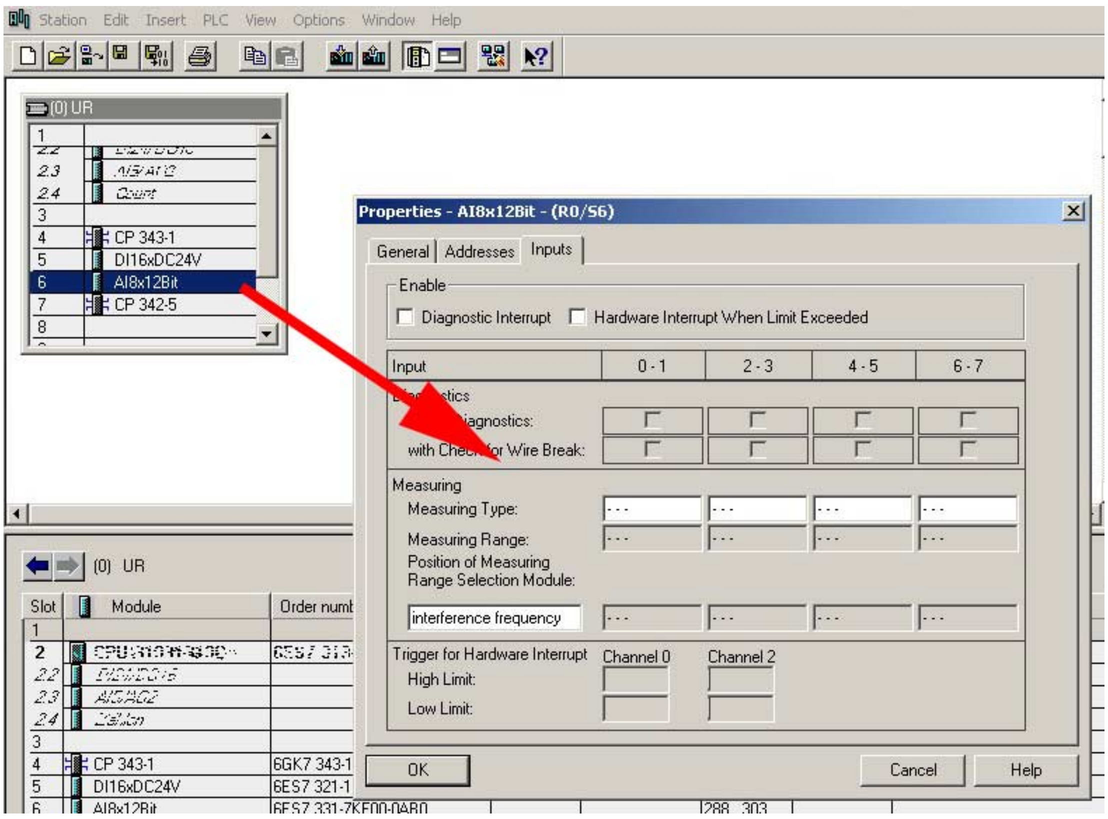
- Authorized users: read and write access
- Unauthorized user: no read and write access

Parameterization of modules

Approach

By using the SIMATIC Manager from Siemens you may set parameters for configurable System 300 modules at any time.

For this, double-click during the project engineering at the slot overview on the module you want to parameterize In the appearing dialog window you may set the wanted parameters.



Parameterization during runtime

By using the SFCs 55, 56 and 57 you may alter and transfer parameters for wanted modules during runtime.

For this you have to store the module specific parameters in so called "record sets".

More detailed information about the structure of the record sets is to find in the according module description.

Project transfer

Overview

There are the following possibilities for project transfer into the CPU:

- Transfer via MPI
- Transfer via MMC
- Transfer via Ethernet

Transfer via MPI

For transfer via MPI the CPU has a MPI interface. This MPI interface supports maximally 32 PG/OP channels.

MPI programming cable

The MPI programming cables are available at VIPA in different variants. The deployment of the cables is identical. The cables provide a bus enabled RS485 plug for the MPI jack of the CPU and a RS232 res. USB plug for the PC.

Due to the RS485 connection you may plug the MPI programming cables directly to an already plugged MPI plug on the MPI jack. Every bus participant identifies itself at the bus with an unique MPI address, in the course of which the address 0 is reserved for programming devices.

Net structure

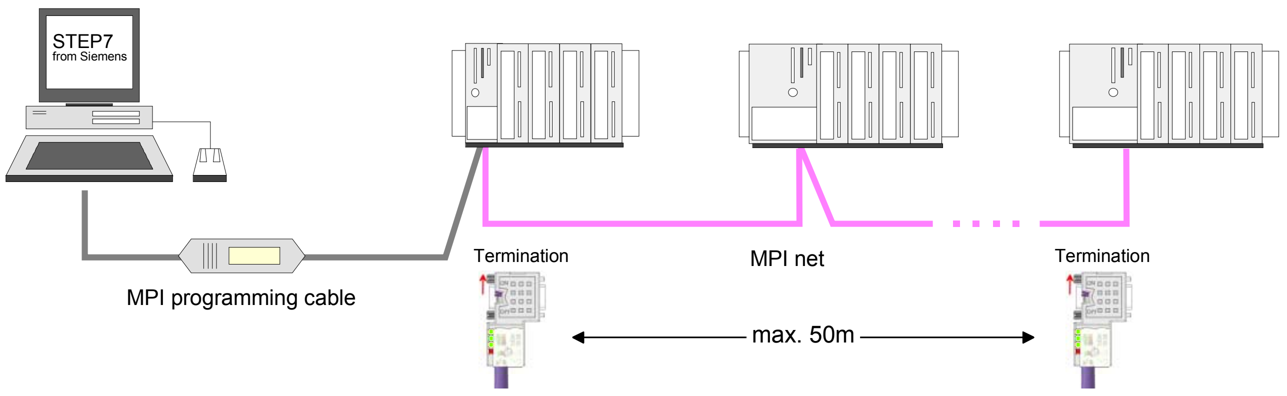
The structure of a MPI net is in the principal identical with the structure of a PROFIBUS net. This means the same rules are valid and you use the same components for the build-up. The single participants are connected with each other via bus connectors and PROFIBUS cables. Please consider with this CPU that the total extension of the MPI net does not exceed 50m.

Per default the MPI net runs with 187.5kbaud. VIPA CPUs are delivered with MPI address 2.

Terminating resistor

A cable has to be terminated with its surge impedance. For this you switch on the terminating resistor at the first and the last participant of a network or a segment.

Please make sure that the participants with the activated terminating resistors are always power supplied. Otherwise it may cause interferences on the bus.



Approach transfer via MPI

A maximum of 32 PG/OP connections is supported by MPI. The transfer via MPI takes place with the following proceeding:

- Connect your PC to the MPI jack of your CPU via a MPI programming cable.
- Load your project in the SIMATIC Manager from Siemens.
- Choose in the menu **Options** > *Set PG/PC interface.*
- Select in the according list the "PC Adapter (MPI)"; if appropriate you have to add it first, then click on [Properties].
- Set in the register *MPI* the transfer parameters of your MPI net and type a valid *address*.
- Switch to the register *Local connection.*
- Set the COM port of the PCs and the transfer rate 38400Baud for the MPI programming cable from VIPA.
- Via **PLC** > *Load to module* you may transfer your project via MPI to the CPU and save it on a MMC via **PLC** > *Copy RAM to ROM* if one is plugged.

Transfer via MMC

The MMC (**Mem**ory **C**ard) serves as external transfer and storage medium. There may be stored several projects and sub-directories on a MMC storage module. Please regard that your current project is stored in the root directory and has one of the following file names:

- *S7PROG.WLD* is read after overall reset respectively may be written by CPU by an order.
- *AUTOLOAD.WLD* is read after PowerON.

Transfer MMC → CPU

The transfer of the application program from the MMC into the CPU takes place depending on the file name after overall reset or PowerON. The blinking of the LED "MCC" of the CPU marks the active transfer.

Please regard that your user memory serves for enough space, otherwise your user program is not completely loaded and the SF LED gets on.

Transfer CPU → MMC

When the MMC has been installed, the write command stores the content of the battery buffered RAM as *S7PROG.WLD* at the MMC. The write command is controlled by means of the block area of the Siemens SIMATIC manager **PLC** > *Copy RAM to ROM*. During the write process the "MCC"-LED of the CPU is blinking. When the LED expires the write process is finished.

Transfer control

After a MMC access, an ID is written into the diagnostic buffer of the CPU. To monitor the diagnosis entries, you select **PLC** > *Module Information* in the Siemens SIMATIC Manager. Via the register "Diagnostic Buffer" you reach the diagnosis window.

When accessing a MMC, the following events may occur:

Event-ID	Meaning
0xE100	MMC access error
0xE101	MMC error file system
0xE102	MMC error FAT
0xE200	MMC writing finished successful
0xE21F	Error during reload, read error, out of memory

Transfer via Ethernet

For transfer via Ethernet the CPU has an Ethernet PG/OP channel. The Ethernet PG/OP channel supports maximally 4 PG/OP connections.

Initialization

So that you may access the Ethernet PG/OP channel you have to assign IP address parameters by means of the "initialization".

Determine Ethernet address

During initialization the Ethernet (MAC) address of the Ethernet PG/OP channel is to be assigned. This may be found beneath the front flap of the CPU on the left side on a sticker. The address begins with "EA: ...".

Proceeding

- Establish a network connection between the Ethernet PG/OP channel and your PC.
- Set at Siemens SIMATIC manager via **Options** > *Set PG/PC Interface* the access path to "TCP/IP -> Network card .... Protocol RFC 1006".
- Open with **PLC** > *Edit Ethernet Node* the dialog window for "initialization" of a station.
- Determine the station via MAC address and assign it to IP address parameters. As long as the Ethernet PG/OP channel was not initialized yet, this owns the IP address 0.0.0.0.

Directly after allocation the Ethernet PG/OP channel of the CPU may be accessed with the Siemens SIMATIC manager by the assigned IP address parameters. More information concerning this may be found at "Initialization Ethernet PG/OP channel".

Transfer

- For transfer open your project in the Siemens SIMATIC manager.
- If not already happen, set at Siemens SIMATIC manager via **Options** > *Set PG/PC Interface* the access path to "TCP/IP -> Network card .... Protocol RFC 1006".
- Click to **PLC** > Download → the dialog "Select target module" is opened. Select your target module and enter the IP address parameters of the Ethernet PG/OP channel for connection. Provided that no new hardware configuration is transferred to the CPU, the entered Ethernet connection is permanently stored in the project as transfer channel.
- With [OK] the transfer is started. System dependent you get a message that the projected system differs from target system. This message may be accepted by [OK] → your project is transferred and may be executed in the CPU after transfer.

Operating modes

Overview

The CPU can be in one of 4 operating modes:

- Operating mode STOP
- Operating mode START-UP
- Operating mode RUN
- Operating mode HOLD

Certain conditions in the operating modes START-UP and RUN require a specific reaction from the system program. In this case the application interface is often provided by a call to an organization block that was included specifically for this event.

Operating mode STOP

- The application program is not processed.
- If there has been a processing before, the values of counters, timers, flags and the process image are retained during the transition to the STOP mode.
- Outputs are inhibited, i.e. all digital outputs are disabled.
- RUN-LED off
- STOP-LED on

Operating mode START-UP

- During the transition from STOP to RUN a call is issued to the start-up organization block OB 100. The length of this OB is not limited. The processing time for this OB is not monitored. The START-UP OB may issue calls to other blocks.
- All digital outputs are disabled during the START-UP, i.e. outputs are inhibited.
- RUN-LED blinks for at least 3s, even if the start-up time is shorter or the CPU gets to STOP due to an error.
- STOP-LED off

When the CPU has completed the START-UP OB, it assumes the operating mode RUN.

Operating mode RUN

- The application program in OB 1 is processed in a cycle. Under the control of alarms other program sections can be included in the cycle.
- All timers and counters being started by the program are active and the process image is updated with every cycle.
- The BASP-signal (outputs inhibited) is deactivated, i.e. all digital outputs are enabled.
- RUN-LED on
- STOP-LED off

Operating mode HOLD

The CPU offers up to 3 breakpoints to be defined for program diagnosis. Setting and deletion of breakpoints happens in your programming environment. As soon as a breakpoint is reached, you may process your program step by step.

Precondition

For the usage of breakpoints, the following preconditions have to be fulfilled:

- Testing in single step mode is only possible with STL. If necessary switch the view via **View** > *STL* to STL.
- The block must be opened online and must not be protected.
- The open block must not be altered in the editor.

Approach for working with breakpoints

- Activate **View** > *Breakpoint Bar*.
- Set the cursor to the command line where you want to insert a breakpoint.
- Set the breakpoint with **Debug** > *Set Breakpoint*. The according command line is marked with a circle.
- To activate the breakpoint click on **Debug** > *Breakpoints Active*. The circle is changed to a filled circle.
- Bring your CPU into RUN. When the program reaches the breakpoint, your CPU switches to the state HOLD, the breakpoint is marked with an arrow and the register contents are monitored.
- Now you may execute the program code step by step via **Debug** > *Execute Next Statement* or run the program until the next breakpoint via **Debug** > *Resume*.
- Delete (all) breakpoints with the option **Debug** > *Delete All Breakpoints*.

Behavior in operating state HOLD

- The LED RUN blinks and the LED STOP is on.
- The execution of the code is stopped. No level is further executed.
- The real-time clock runs is just running.
- The outputs were disabled (BASP is activated).
- Configured CP connections remain exist.

Note!

The usage of breakpoints is always possible. Switching to the operating mode test operation is not necessary.

With more than 2 breakpoints, a single step execution is not possible.

Function security

The CPUs include security mechanisms like a Watchdog (100ms) and a parameterizable cycle time surveillance (parameterizable min. 1ms) that stop res. execute a RESET at the CPU in case of an error and set it into a defined STOP state.

The VIPA CPUs are developed function secure and have the following system properties:

Event	concerns	Effect
RUN → STOP	general	BASP (**B**efehls-**A**usgabe-**Sp**erre, i.e. command output lock) is set.
	central digital outputs	The outputs are set to 0V.
	central analog outputs	The voltage supply for the output channels is switched off.
	decentralized outputs	The outputs are set to 0V.
	decentralized inputs	The inputs are read constantly from the slave and the recent values are put at disposal.
STOP → RUN respectively PowerON	general	First the PII is deleted, then OB 100 is called. After the execution of the OB, the BASP is reset and the cycle starts with: Delete PIQ → Read PII → OB 1.
	central analog outputs	The behavior of the outputs at restart can be preset.
	decentralized inputs	The inputs are read constantly from the slave and the recent values are put at disposal.
RUN	general	The program execution happens cyclically and can therefore be foreseen: Read PII → OB 1 → Write PIQ.

PII = Process image inputs

PIQ = Process image outputs

Overall reset

Overview

During the overall reset the entire user memory (RAM) is erased. Data located in the memory card is not affected.

You have 2 options to initiate an overall reset:

- initiate the overall reset by means of the function selector switch
- initiate the overall reset by means of the Siemens SIMATIC Manager

Note!

You should always issue an overall reset to your CPU before loading an application program into your CPU to ensure that all blocks have been cleared from the CPU.

Overall reset by means of the function selector

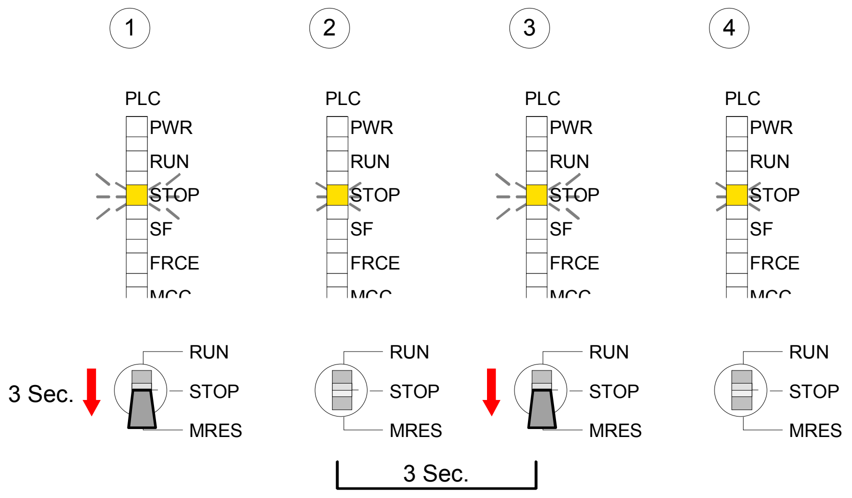
Condition

The operating mode of the CPU is STOP. Place the function selector on the CPU in position "STOP" → the STOP-LED is on.

Overall reset

- Place the function selector in the position MRES and hold it in this position for app. 3 seconds. → The STOP-LED changes from blinking to permanently on.
- Place the function selector in the position STOP and switch it to MRES and quickly back to STOP within a period of less than 3 seconds. → The STOP-LED blinks (overall reset procedure).
- The overall reset has been completed when the STOP-LED is on permanently. → The STOP-LED is on.

The following figure illustrates the above procedure:



Automatic reload

If there is a project S7PROG.WLD on the MMC, the CPU attempts to reload this project from MMC → the MCC LED is on.

When the reload has been completed the LED expires. The operating mode of the CPU will be STOP or RUN, depending on the position of the function selector.

Overall reset by means of the Siemens SIMATIC Manager

Condition

The operating mode of the CPU must be STOP.
You may place the CPU in STOP mode by the menu command **PLC** > *Operating mode*.

Overall reset

You may request the overall reset by means of the menu command **PLC** > *Clean/Reset*.

In the dialog window you may place your CPU in STOP mode and start the overall reset if this has not been done as yet.

The STOP-LED blinks during the overall reset procedure.

When the STOP-LED is on permanently the overall reset procedure has been completed.

Automatic reload

If there is a project S7PROG.WLD on the MMC, the CPU attempts to reload this project from MMC → the MCC LED is on.

When the reload has been completed, the LED expires. The operating mode of the CPU will be STOP or RUN, depending on the position of the function selector.

Set back to factory setting

The following approach deletes the internal RAM of the CPU completely and sets it back to the delivery state.

Please regard that the MPI address is also set back to default 2!

More information may be found at the part "Factory reset" further below.

Firmware update

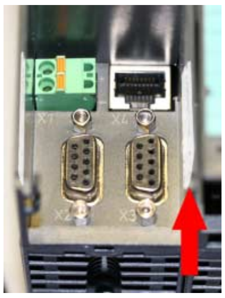
Overview

By means of a MMC there is the opportunity to execute a firmware update at the CPU and its components.

For this an accordingly prepared MMC must be in the CPU during the start-up.

Thus a firmware file may be recognized and assigned with start-up, a pkg file name is reserved for each updateable component and hardware release, which begins with "px" and differs in a number with six digits. The pkg file name of every updateable component may be found at a label right down the front flap of the module.

As soon as with start-up a pkg file is on the MMC and the firmware is more current than in the components, all the pkg file assigned components within the CPU get the new firmware.



Firmware package and version

Latest Firmware at www.vipa.de

The latest 2 firmware versions may be found in the service area at www.vipa.de.

For example the following file is necessary for the firmware update of the CPU 312-5BE13 with hardware release 1: Px000135_Vxxx.zip

Attention!

When installing a new firmware you have to be extremely careful. Under certain circumstances you may destroy the CPU, for example if the voltage supply is interrupted during transfer or if the firmware file is defective.

In this case, please call the VIPA-Hotline!

Please regard that the version of the update firmware is different to the existing firmware otherwise no update is executed.

Display the Firmware version of the SPEED7 system via web page

The SC CPU has an integrated web page that monitors information about firmware version of the I/O components. The Ethernet PG/OP channel provides the access to this web page.

To activate the PG/OP channel you have to enter according IP parameters.

This can be made in Siemens SIMATIC manager either by a hardware configuration, loaded by MMC respectively MPI or via Ethernet by means of the MAC address with **PLC** > *Assign Ethernet Address*.

After that you may access the PG/OP channel with a web browser via the IP address of the project engineering. More detailed information is to find in "Access to Ethernet PG/OP channel and website".

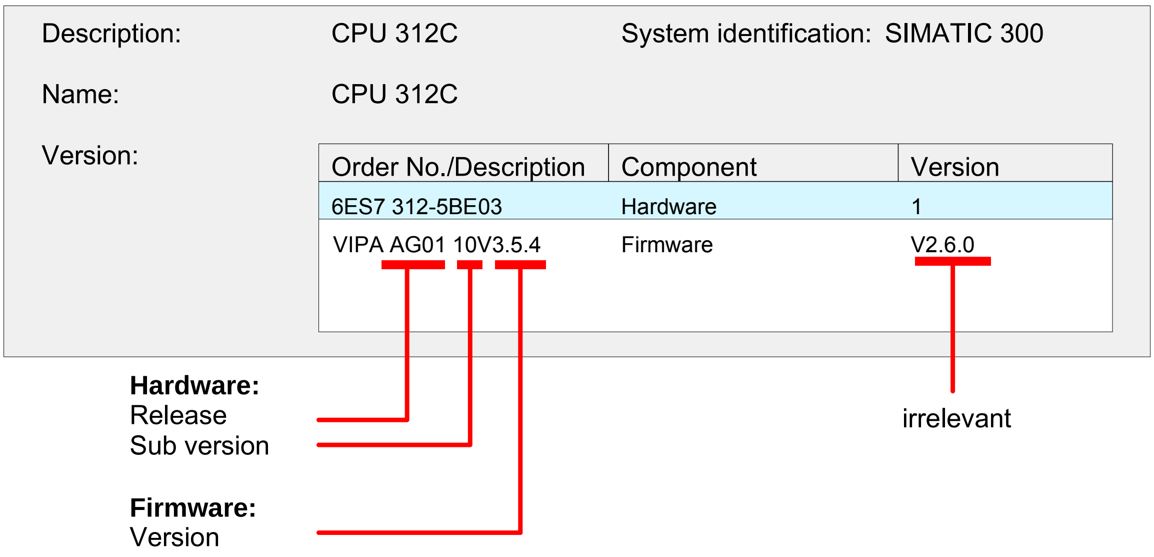
Determine CPU firmware version with module information

First establish an online connection to the CPU. To monitor the module information you choose the option **PLC** > *Module Information* in the Siemens SIMATIC Manager.

Via the register "General" the window with hardware and firmware version may be selected.

From software-technical reasons there is something different of the CPU 312SC to the CPU 312C from Siemens:

The releases of hard and software may be found at "Order No./Description". Here the number at "Version" is irrelevant.


Description: CPU 312C System identification: SIMATIC 300

Name: CPU 312C

Version:

Order No./Description	Component	Version
6ES7 312-5BE03	Hardware	1
VIPA AG01 10V3.5.4	Firmware	V2.6.0

Note!

Every register of the module information dialog is supported by the VIPA CPUs. More about these registers may be found in the online help of the Siemens SIMATIC manager.

Load firmware and transfer it to MMC

- Go to www.vipa.de.
- Click on Service > Download > Firmware Updates.
- Click on "Firmware for System 300S CPUs"
- Choose the according modules (CPU, DPM, CP...) and download the firmware Px......zip to your PC.
- Extract the zip-file and copy the extracted file to your MMC.
- Following this approach, transfer all wanted firmware files to your MMC.

Attention!

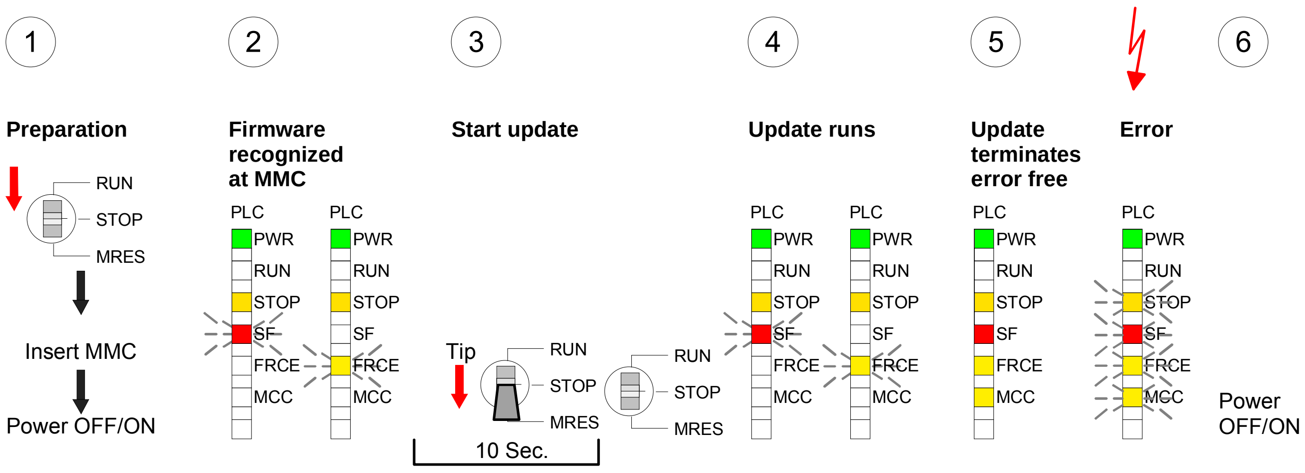
With a firmware update an overall reset is automatically executed. If your program is only available in the load memory of the CPU it is deleted! Save your program before executing a firmware update! After the firmware update you should execute a "Set back to factory settings" (see following page).

Transfer firmware from MMC into CPU

1. Get the RUN-STOP lever of your CPU in position STOP. Turn off the voltage supply. Plug the MMC with the firmware files into the CPU. Please take care of the correct plug-in direction of the MMC. Turn on the voltage supply.
2. After a short boot-up time, the alternate blinking of the LEDs SF and FRCE shows that at least a more current firmware file was found on the MMC.
3. You start the transfer of the firmware as soon as you tip the RUN/STOP lever downwards to MRES within 10s.
4. During the update process, the LEDs SF and FRCE are alternately blinking and MMC LED is on. This may last several minutes.
5. The update is successful finished when the LEDs PWR, STOP, SF, FRCE and MCC are on. If they are blinking fast, an error occurred.
6. Turn Power OFF and ON. Now it is checked by the CPU, whether further current firmware versions are available at the MMC. If so, again the LEDs SF and FRCE flash after a short start-up period. Continue with point 3.

   If the LEDs do not flash, the firmware update is ready.

   Now a *factory reset* should be executed (see next page). After that the CPU is ready for duty.



Factory reset

Proceeding

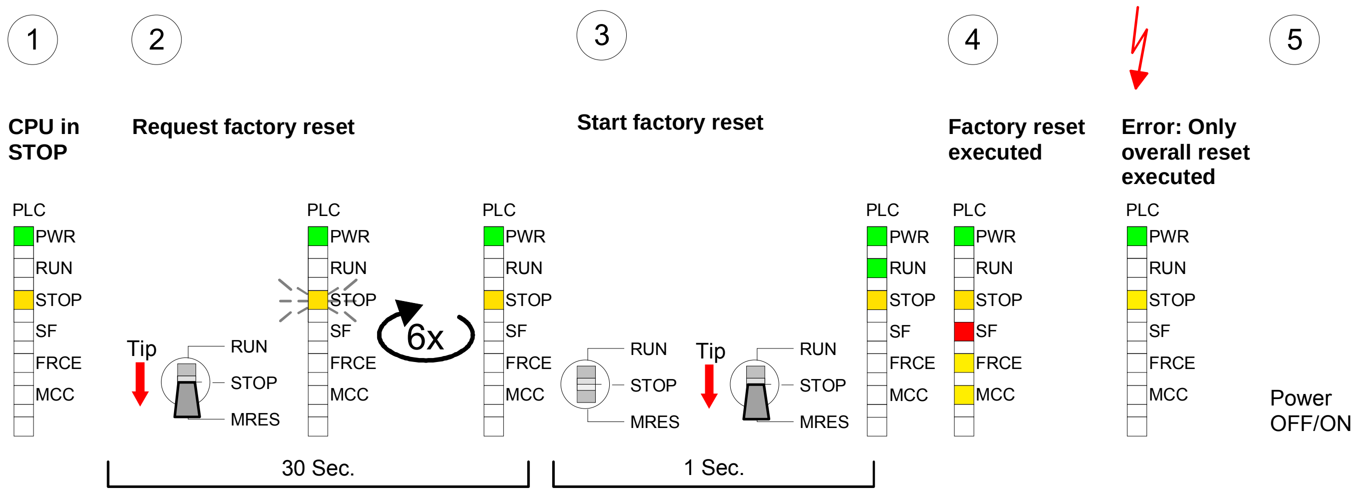
With the following proceeding the internal RAM of the CPU is completely deleted and the CPU is reset to delivery state.

Please note that here also the IP address of the Ethernet PG/OP channel is set to 0.0.0.0 and the MPI address is reset to the address 2!

A factory reset may also be executed by the MMC-Cmd FACTORY_RESET. More information may be found at "MMC-Cmd - Auto commands".

1. Switch the CPU to STOP.
2. Push the operating switch down to position MRES for 30s. Here the STOP-LED flashes. After a few seconds the stop LED changes to static light. Now the STOP LED changes between static light and flashing. Starting here count the static light states.
3. After the 6. static light release the operating mode switch and tip it downwards to MRES. Now the RUN LED lights up once. This means that the RAM was deleted completely.
4. For the confirmation of the resetting procedure the LEDs PWR, STOP, SF, FRCE and MCC get ON. If not, the factory reset has failed and only an overall reset was executed. In this case you can repeat the procedure. A factory reset can only be executed if the stop LED has static light for exactly 6 times.
5. The end of factory reset is shown by static light of the LEDs STOP, SF, FRCE and MCC. Switch the power supply off and on.

The proceeding is shown in the following Illustration:



Note!
After the firmware update you always should execute a *Factory reset*.

Slot for storage media

Overview

At the front of the CPU there is a slot for storage media. As external storage medium for applications and firmware you may use a multimedia card (MMC) or a VIPA MCC memory extension card. The MCC can additionally be used as an external storage medium.

It has the PC compatible FAT16 file format.

You can cause the CPU to load a project automatically respectively to execute a command file by means of pre-defined file names.

Accessing the storage medium

To the following times an access takes place on a storage medium:

After overall reset

- The CPU checks if there is a project S7PROG.WLD. If exists the project is automatically loaded.
- The CPU checks if there is a project PROTECT.WLD with protected blocks. If exists the project is automatically loaded. These blocks are stored in the CPU until the CPU is reset to factory setting or an empty PROTECT.WLD is loaded.
- The CPU checks if a MCC memory extension card is put. If exists the memory extension is enabled, otherwise a memory expansion, which was activated before, is de-activated.

After PowerON

- The CPU checks if there is a project AUTOLOAD.WLD. If exists an overall reset is established and the project is automatically loaded.
- The CPU checks if there is a command file with VIPA_CMD.MMC. If exists the command file is loaded and the containing instructions are executed.
- After PowerON and CPU STOP the CPU checks if there is a *.pkg file (firmware file). If exists this is indicated by blinking of the LEDs and the firmware may be installed by an update request (see "Firmware update).

Once in STOP

- If a storage medium is put, which contains a command file VIPA_CMD.MMC, the command file is loaded and the containing instructions are executed.

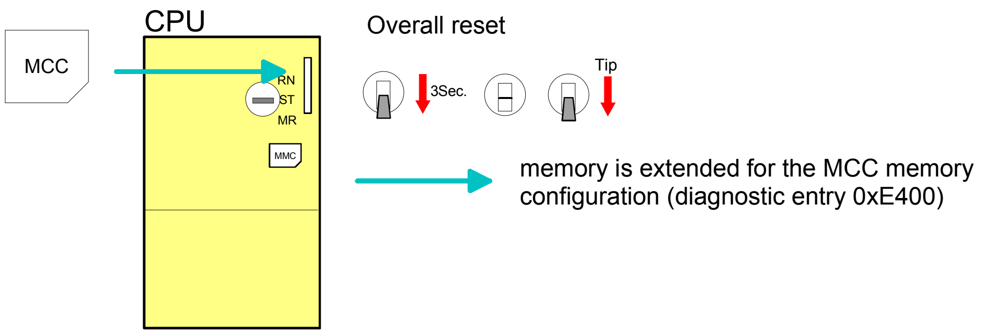
Memory extension with MCC

Overview

With the SC CPU there is the possibility to extend the work memory of your CPU.
For this, a MCC memory extension card is available from VIPA. The MCC is a specially prepared MMC (**M**ulti**m**edia **C**ard). By plugging the MCC into the MCC slot and then an overall reset the according memory expansion is released. There may only one memory expansion be activated at the time.
On the MCC there is the file *memory.key*. This file may not be altered or deleted. You may use the MCC also as "normal" MMC for storing your project.

Approach

To extend the memory, plug the MCC into the card slot at the CPU labeled with "MCC" and execute an overall reset.


MCC
CPU
RN
ST
MR
MMC
Overall reset
3Sec.
Tip
memory is extended for the MCC memory configuration (diagnostic entry 0xE400)

If the memory expansion on the MCC exceeds the maximum extendable memory range of the CPU, the maximum possible memory of the CPU is automatically used.

You may determine the recent memory extension via the Siemens SIMATIC Manager at *Module Information* - "Memory".

Attention!
Please regard that the MCC must remain plugged when you've executed the memory expansion at the CPU. Otherwise the CPU switches to STOP after 72h. The MCC can <u>not</u> be exchanged with a MCC of the same memory configuration.

Behavior

When the MCC memory configuration has been taken over you may find the diagnosis entry 0xE400 in the diagnostic buffer of the CPU.

After pulling the MCC the entry 0xE401 appears in the diagnostic buffer, the SF-LED is on and after 72h the CPU switches to STOP. A reboot is only possible after plugging-in the MCC again or after an overall reset.

After re-plugging the MCC, the SF-LED extinguishes and 0xE400 is entered into the diagnostic buffer.

You may reset the memory configuration of your CPU to the initial status at any time by executing an overall reset without MCC.

Extended know-how protection

Overview

Besides the "standard" Know-how protection the CPU from VIPA provide an "extended" know-how protection that serves a secure block protection for accesses of 3rd persons.

Standard protection

The standard protection from Siemens transfers also protected blocks to the PG but their content is not displayed. But with according manipulation the Know-how protection is not guaranteed.

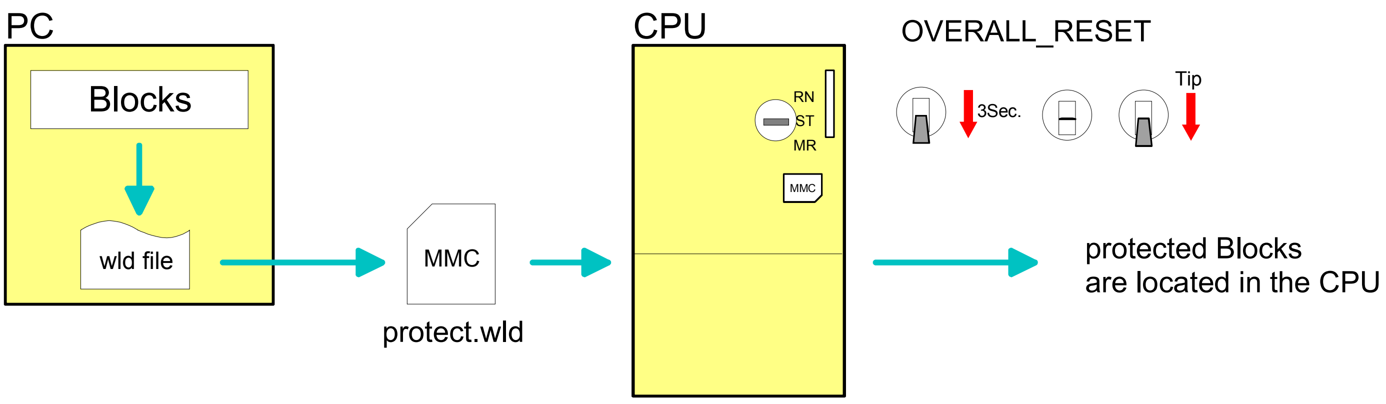
Extended protection

The "extended" know-how protection developed by VIPA offers the opportunity to store blocks permanently in the CPU.

At the "extended" protection you transfer the protected blocks into a WLD-file named protect.wld. By plugging the MMC and following overall reset, the blocks in the protect.wld are permanently stored in the CPU.

You may protect OBs, FBs and FCs.

When back-reading the protected blocks into the PG, exclusively the block header are loaded. The source remains in the CPU and is thus protected for accesses of 3rd persons.



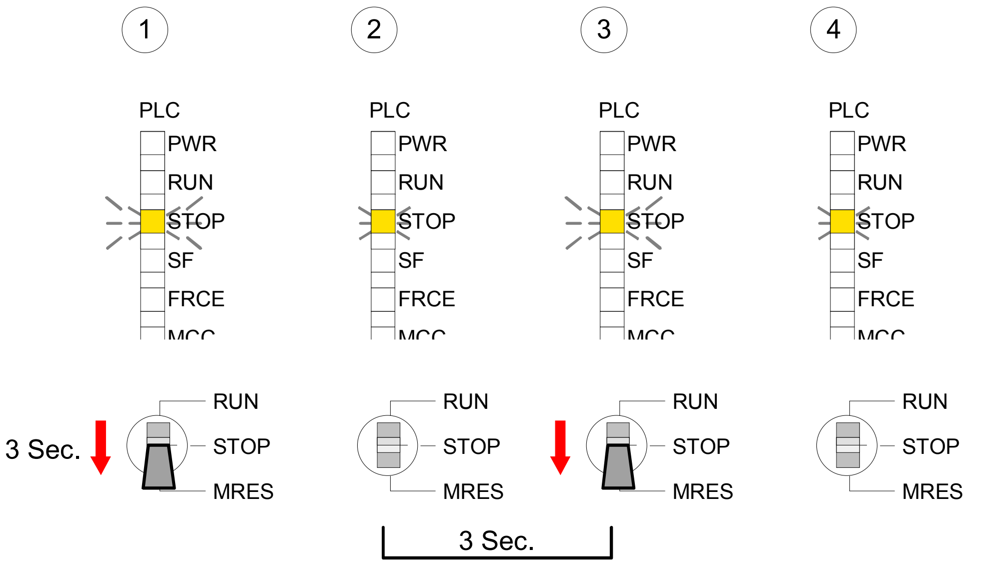
Protect blocks with protect.wld

Create a new wld-file in your project engineering tool with **File** > *Memory Card file* > *New* and rename it to "protect.wld".

Transfer the according blocks into the file by dragging them with the mouse from the project to the file window of protect.wld.

Transfer protect.wld to CPU with overall reset

Transfer the file protect.wld to a MMC storage module, plug the MMC into the CPU and execute an overall reset with the following approach:



The overall reset stores the blocks in protect.wld permanently in the CPU protected from accesses of 3. persons.

Protection behavior

Protected blocks are overwritten by a new protect.wld.

Using a PG 3rd persons may access protected blocks but only the block header is transferred to the PG. The block code that is to protect remains in the CPU and can not be read.

Change respectively delete protected blocks

Protected blocks in the RAM of the CPU may be substituted at any time by blocks with the same name. This change remains up to next overall reset. Protected blocks may permanently be overwritten only if these are deleted at the protect.wld before.

By transferring an empty protect.wld from the MMC you may delete all protected blocks in the CPU.

Usage of protected blocks

Due to the fact that reading of a "protected" block from the CPU monitors no symbol labels it is convenient to provide the "block covers" for the end user.

For this, create a project out of all protected blocks. Delete all networks in the blocks so that these only contain the variable definitions in the according symbolism.

MMC-Cmd - Auto commands

Overview

A *command file* at a MMC is once executed until the next PowerON under the following conditions:

- CPU is in STOP and MMC is stuck
- After PowerON with operating switch in STOP

Command file

The *command file* is a text file, which consists of a command sequence to be stored as ***vipa_cmd.mmc*** in the root directory of the MMC.

The file has to be started by *CMD_START* as 1. command, followed by the desired commands (no other text) and must be finished by CMD_END as last command.

Text after the last command *CMD_END* e.g. comments is permissible, because this is ignored. As soon as the command file is recognized and executed each action is stored at the MMC in the log file logfile.txt. In addition for each executed command a diagnostics entry may be found in the diagnostics buffer.

Commands

Please regard the command sequence is to be started with *CMD_START* and ended with CMD_END.

Command	Description	Diagnostics entry
CMD_START	In the first line *CMD_START* is to be located.	0xE801
	There is a diagnostic entry if *CMD_START* is missing	0xE8FE
WAIT1SECOND	Waits ca. 1 second.	0xE803
WEBPAGE	The current web page of the CPU is stored at the MMC as "webpage.htm".	0xE804
LOAD_PROJECT	The function "Overall reset and reload from MMC" is executed. The wld file located after the command is loaded else "s7prog.wld" is loaded.	0xE805
SAVE_PROJECT	The recent project (blocks and hardware configuration) is stored as "s7prog.wld" at the MMC. If the file just exists it is renamed to "s7prog.old". If your CPU is password protected so you have to add this as parameter. Otherwise there is no project written. Example: SAVE_PROJECT password	0xE806
FACTORY_RESET	Executes "factory reset".	0xE807
DIAGBUF	The current diagnostics buffer of the CPU is stored as "diagbuff.txt" at the MMC.	0xE80B
SET_NETWORK	IP parameters for Ethernet PG/OP channel may be set by means of this command. The IP parameters are to be given in the order IP address, subnet mask and gateway in the format xxx.xxx.xxx.xxx each separated by a comma. Enter the IP address if there is no gateway used.	0xE80E
CMD_END	In the last line *CMD_END* is to be located.	0xE802

Examples The structure of a command file is shown in the following. The corresponding diagnostics entry is put in parenthesizes.

Example 1

`CMD_START`	Marks the start of the command sequence (0xE801)
`LOAD_PROJECT proj.wld`	Execute an overall reset and load "proj.wld" (0xE805)
`WAIT1SECOND`	Wait ca. 1s (0xE803)
`WEBPAGE`	Store web page as "webpage.htm" (0xE804)
`DIAGBUF`	Store diagnostics buffer of the CPU as "diagbuff.txt" (0xE80B)
`CMD_END`	Marks the end of the command sequence (0xE802)
`... arbitrary  text ...`	Text after the command CMD_END is not evaluated.

Example 2

`CMD_START`	Marks the start of the command sequence (0xE801)
`LOAD_PROJECT proj2.wld`	Execute an overall reset and load "proj2.wld" (0xE805)
`WAIT1SECOND`	Wait ca. 1s (0xE803)
`WAIT1SECOND`	Wait ca. 1s (0xE803)
`SET_NETWORK 172.16.129.210,255.255.224.0,172.16.129.210`	IP parameter (0xE80E)
`WAIT1SECOND`	Wait ca. 1s (0xE803)
`WAIT1SECOND`	Wait ca. 1s (0xE803)
`WEBPAGE`	Store web page as "webpage.htm" (0xE804)
`DIAGBUF`	Store diagnostics buffer of the CPU as "diagbuff.txt" (0xE80B)
`CMD_END`	Marks the end of the command sequence (0xE802)
`... arbitrary  text ...`	Text after the command CMD_END is not evaluated.

Note!

The parameters IP address, subnet mask and gateway may be received from the system administrator.

Enter the IP address if there is no gateway used.

VIPA specific diagnostic entries

Entries in the diagnostic buffer

You may read the diagnostic buffer of the CPU via the Siemens SIMATIC Manager. Besides of the standard entries in the diagnostic buffer, the VIPA CPUs support some additional specific entries in form of event-IDs.

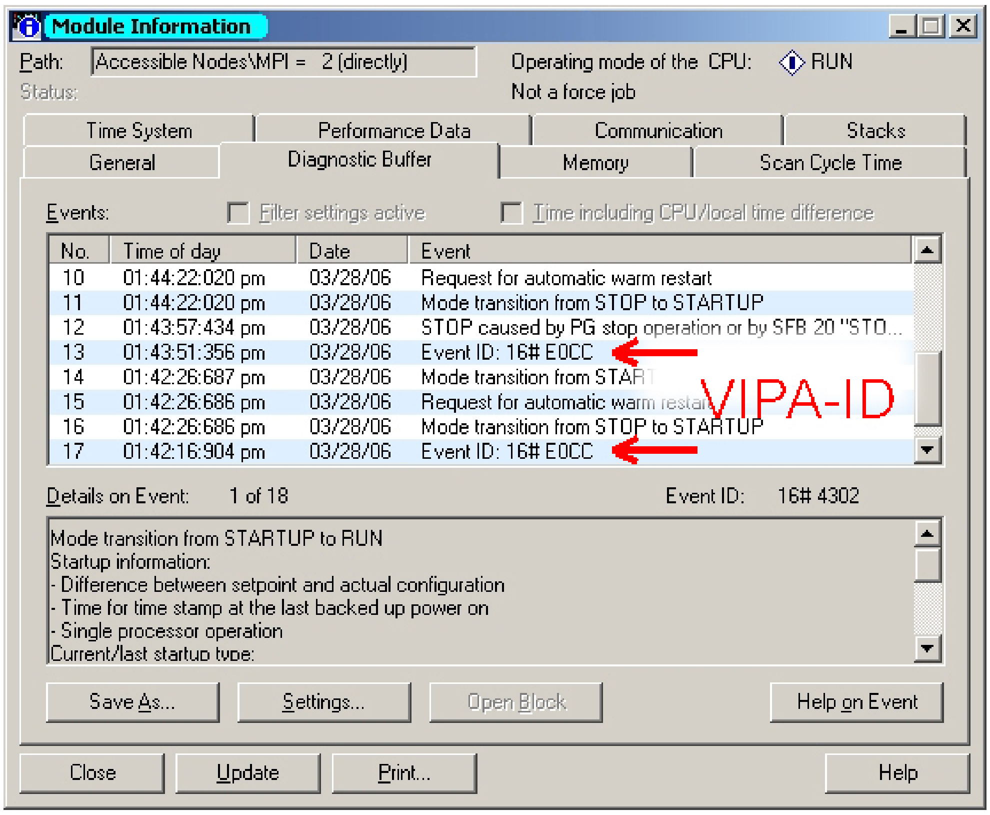
The current content of the diagnostics buffer is stored on MMC by means of the MMC-Cmd DIAGBUF. More information may be found at "MMC-Cmd - Auto commands".

Note!

Every register of the module information is supported by the VIPA CPUs. More information may be found at the online help of the Siemens SIMATIC manager.

Monitoring the diagnostic entries

To monitor the diagnostic entries you choose the option **PLC** > *Module Information* in the Siemens SIMATIC Manager. Via the register "Diagnostic Buffer" you reach the diagnostic window:



Path: Accessible Nodes\MPI = 2 (directly)
Operating mode of the CPU: RUN
Status:
Not a force job

Events: ☐ Filter settings active ☐ Time including CPU/local time difference

No.	Time of day	Date	Event
10	01:44:22:020 pm	03/28/06	Request for automatic warm restart
11	01:44:22:020 pm	03/28/06	Mode transition from STOP to STARTUP
12	01:43:57:434 pm	03/28/06	STOP caused by PG stop operation or by SFB 20 "STO...
13	01:43:51:356 pm	03/28/06	Event ID: 16# E0CC
14	01:42:26:687 pm	03/28/06	Mode transition from STAR
15	01:42:26:686 pm	03/28/06	Request for automatic warm restai
16	01:42:26:686 pm	03/28/06	Mode transition from STOP to STARTUP
17	01:42:16:904 pm	03/28/06	Event ID: 16# E0CC

Details on Event: 1 of 18 Event ID: 16# 4302

```
Mode transition from STARTUP to RUN
Startup information:
- Difference between setpoint and actual configuration
- Time for time stamp at the last backed up power on
- Single processor operation
Current/last startup type:
```

The diagnosis is independent from the operating mode of the CPU. You may store a max. of 100 diagnostic entries in the CPU.

The following page shows an overview of the VIPA specific Event-IDs.

Overview of the Event-IDs

Event-ID	Description
0xE003	Error at access to I/O devices Zinfo1: I/O address Zinfo2: Slot
0xE004	Multiple parameterization of a I/O address Zinfo1: I/O address Zinfo2: Slot
0xE005	Internal error - Please contact the VIPA-Hotline!
0xE006	Internal error - Please contact the VIPA-Hotline!
0xE007	Configured in-/output bytes do not fit into I/O area
0xE008	Internal error - Please contact the VIPA-Hotline!
0xE009	Error at access to standard back plane bus
0xE010	Not defined module group at backplane bus recognized Zinfo2: Slot Zinfo3: Type ID
0xE011	Master project engineering at Slave-CPU not possible or wrong slave configuration
0xE012	Error at parameterization
0xE013	Error at shift register access to standard bus digital modules
0xE014	Error at Check_Sys
0xE015	Error at access to the master Zinfo2: Slot of the master (32=page frame master)
0xE016	Maximum block size at master transfer exceeded Zinfo1: I/O address Zinfo2: Slot
0xE017	Error at access to integrated slave
0xE018	Error at mapping of the master I/O devices
0xE019	Error at standard back plane bus system recognition
0xE01A	Error at recognition of the operating mode (8 / 9 Bit)
0xE01B	Error - maximum number of plug-in modules exceeded
0xE030	Error of the standard bus
0xE0B0	Speed7 is not stoppable (probably undefined BCD value at timer)
0xE0C0	Not enough space in work memory for storing code block (block size exceeded)
0xE0CC	Communication error MPI / Serial
0xE0CD	Error at DPV1 job management
0xE0CE	Error: Timeout at sending of the i-slave diagnostics
0xE100	MMC access error
0xE101	MMC error file system
0xE102	MMC error FAT
0xE104	MMC error at saving

continued ...

... continue

Event-ID	Description
0xE200	MMC writing finished (Copy Ram2Rom)
0xE210	MMC reading finished (reload after overall reset)
0xE21F	MMC reading: error at reload (after overall reset), read error, out of memory
0xE300	Internal flash writing finished (Copy Ram2Rom)
0xE400	Memory expansion MCC has been plugged
0xE401	Memory expansion MCC has been removed
0xE801	MMC-Cmd: CMD_START recognized and successfully executed
0xE802	MMC-Cmd: CMD_END recognized and successfully executed
0xE803	MMC-Cmd: WAIT1SECOND recognized and successfully executed
0xE804	MMC-Cmd: WEBPAGE recognized and successfully executed
0xE805	MMC-Cmd: LOAD_PROJECT recognized and successfully executed
0xE806	MMC-Cmd: SAVE_ PROJECT recognized and successfully executed
0xE807	MMC-Cmd: FACTORY_RESET recognized and successfully executed
0xE808	MMC-Cmd : DPM_DEBUGLEVEL recognized and successfully executed
0xE80B	MMC-Cmd: DIAGBUF recognized and successfully executed
0xE80E	MMC-Cmd: SET_NETWORK recognized and successfully executed
0xE810	MMC-Cmd : BLOCK_CRC_ON recognized and successfully executed
0xE811	MMC-Cmd : BLOCK_CRC_OFF recognized and successfully executed
0xE812	MMC-Cmd : BLOCK_COMPRESS_ON recognized and successfully executed
0xE813	MMC-Cmd : BLOCK_COMPRESS_OFF recognized and successfully executed
0xE8FB	MMC-Cmd: Error: Initialization of the Ethernet PG/OP channel by means of SET_NETWORK is faulty.
0xE8FC	MMC-Cmd: Error: Not every IP-Parameter is set at SET_NETWORK.
0xE8FE	MMC-Cmd: Error: CMD_START was not found
0xE8FF	MMC-Cmd: Error: Reading the CMD file is faulty (MMC error)
0xE901	Check sum error
0xEA00	Internal error - Please contact the VIPA-Hotline!
0xEA01	Internal error - Please contact the VIPA-Hotline!
0xEA02	SBUS: Internal error (internal plugged sub module not recognized) Zinfo1: Internal slot
0xEA04	SBUS: Multiple parameterization of a I/O address Zinfo1: I/O address Zinfo2: Slot Zinfo3: Data width

continued ...

... continue

Event-ID	Description
0xEA05	Internal error - Please contact the VIPA-Hotline!
0xEA07	Internal error - Please contact the VIPA-Hotline!
0xEA08	SBUS: Parameterized input data width unequal to plugged input data width Zinfo1: Parameterized input data width Zinfo2: Slot Zinfo3: Input data width of the plugged module
0xEA09	SBUS: Parameterized output data width unequal to plugged output data width Zinfo1: Parameterized output data width Zinfo2: Slot Zinfo3: Output data width of the plugged module
0xEA10	SBUS: Input address outside input area Zinfo1: I/O address Zinfo2: Slot Zinfo3: Data width
0xEA11	SBUS: Output address outside output area Zinfo1: I/O address Zinfo2: Slot Zinfo3: Data width
0xEA12	SBUS: Error at writing record set Zinfo1: Slot Zinfo2: Record set number Zinfo3: Record set length
0xEA14	SBUS: Multiple parameterization of a I/O address (Diagnostic address) Zinfo1: I/O address Zinfo2: Slot Zinfo3: Data width
0xEA15	Internal error - Please contact the VIPA-Hotline!
0xEA18	SBUS: Error at mapping of the master I/O devices Zinfo2: Master slot
0xEA19	Internal error - Please contact the VIPA-Hotline!
0xEA20	Error - RS485 interface is not set to PROFIBUS DP master but there is a PROFIBUS DP master configured.
0xEA21	Error - Project engineering RS485 interface X2/X3: PROFIBUS DP master is configured but missing Zinfo2: Interface x
0xEA22	Error - RS485 interface X2 - value is out of range Zinfo: Configured value X2
0xEA23	Error - RS485 interface X3 - value is out of range Zinfo: Configured value X3

continued ...

... continue

Event-ID	Description	
0xEA24	Error - Project engineering RS485 interface X2/X3: Interface/Protocol is missing, the default settings are used. Zinfo2: Configured value X2 Zinfo2: Configured value X3	
0xEA30	Internal error - Please contact the VIPA-Hotline!	
0xEA40	Internal error - Please contact the VIPA-Hotline!	
0xEA41	Internal error - Please contact the VIPA-Hotline!	
0xEA97	Memory error SBUS service Channel (ZInfo3 = Slot)	
0xEA98	Timeout at waiting for reboot of a SBUS module (Server)	
0xEA99	Error at file reading via SBUS	
0xEE00	Additional information at UNDEF_OPCODE	
	1	Wrong Priority
	2	Buffer overflow
	3	Frame format errors
	4	Incorrect SSL request (SZL-ID invalid)
	5	Incorrect SSL request (SZL-SubID invalid)
	6	Incorrect SSL request (SZL-Index invalid)
	7	Incorrect value
	8	Incorrect RetVal
	9	Incorrect SAP
	10	Incorrect connection type
	11	Incorrect sequence number
	12	Faulty block number in the telegram
	13	Faulty block type in the telegram
	14	Inactive function
	15	Incorrect size in the telegram
	20	Error writing to MMC
	90	Incorrect Buffer size
	98	Unknown error
	99	Internal error

Using test functions for control and monitoring of variables

Overview

For troubleshooting purposes and to display the status of certain variables you can access certain test functions via the menu item **Debug** of the Siemens SIMATIC Manager.

The status of the operands and the VKE can be displayed by means of the test function **Debug** > *Monitor*.

You can modify and/or display the status of variables by means of the test function **PLC** > *Monitor/Modify Variables*.

Debug > *Monitor*

This test function displays the current status and the VKE of the different operands while the program is being executed.

It is also possible to enter corrections to the program.

Note!
When using the test function “Monitor” the PLC must be in RUN mode!

The processing of statuses can be interrupted by means of jump commands or by timer and process-related alarms. At the breakpoint the CPU stops collecting data for the status display and instead of the required data it only provides the PG with data containing the value 0.

For this reason, jumps or time and process alarms can result in the value displayed during program execution remaining at 0 for the items below:

- the result of the logical operation VKE
- Status / AKKU 1
- AKKU 2
- Condition byte
- absolute memory address SAZ. In this case SAZ is followed by a "?".

The interruption of the processing of statuses does not change the execution of the program. It only shows that the data displayed is no longer.

PLC > *Monitor/Modify Variables*

This test function returns the condition of a selected operand (inputs, outputs, flags, data word, counters or timers) at the end of program-execution.

This information is obtained from the process image of the selected operands. During the "processing check" or in operating mode STOP the periphery is read directly from the inputs. Otherwise only the process image of the selected operands is displayed.

Control of outputs

It is possible to check the wiring and proper operation of output-modules.

You can set outputs to any desired status with or without a control program. The process image is not modified but outputs are no longer inhibited.

Control of variables

The following variables may be modified:

I, Q, M, T, C and D.

The process image of binary and digital operands is modified independently of the operating mode of the SC CPU.

When the operating mode is RUN the program is executed with the modified process variable. When the program continues they may, however, be modified again without notification.

Process variables are controlled asynchronously to the execution sequence of the program.

Chapter 5 Deployment I/O periphery

Overview

This chapter contains all information necessary for the employment of the in-/output periphery of the CPU 312SC. It describes functionality, project engineering and diagnostic of the analog and digital part.

Content

Overview

Hardware

At the CPU 312SC the connectors for digital in-/output and technological functions are integrated to a 2tier casing.

Project engineering

The project engineering takes place in the Siemens SIMATIC manager as CPU 312C from Siemens (6ES7 312-5BE03-0AB0 V2.6).

Here the CPU is parameterized by the "Properties" dialog of the CPU 312C.

For parameterization of the digital I/O periphery and the technological functions the corresponding submodule of the CPU 312C may be used.

I/O periphery

The integrated I/Os of the CPU 312SC may be used for technological functions or as standard I/Os.

Technological functions and standard I/Os may be used simultaneously with appropriate hardware. Read access to inputs used by technological functions is possible. Write access to used outputs is not possible.

Technological functions

Up to 2 channels may be parameterized as technological function. The parameterization of the appropriate channel is made in the hardware configurator by the *count* submodule of the CPU 312C.

There are the following technological functions:

- Continuous count
- Single count
- Periodic count

The controlling of the corresponding counter mode happens by means of the SFB COUNT (SFB 47) of the user program.

In-/Output range CPU 312SC

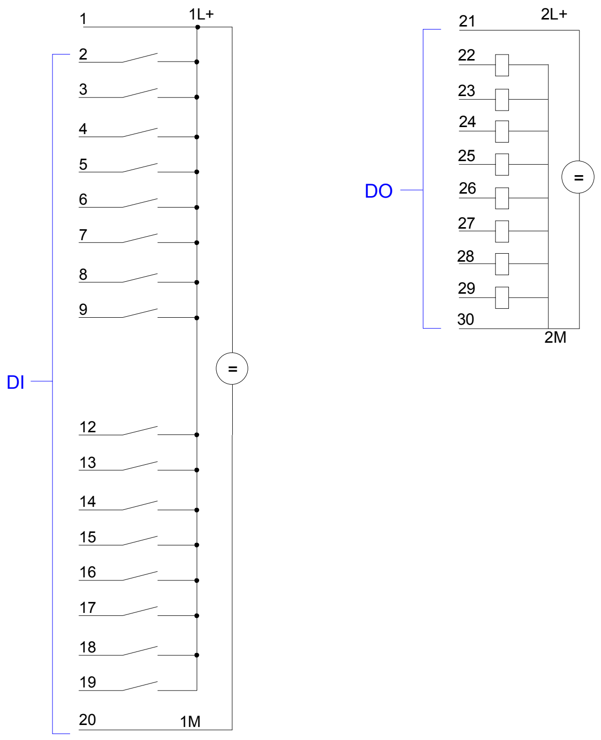
Overview CPU 312SC

The CPU 312SC has the following analog and digital in- and output ranges integrated in one casing:

• Digital Input:	16xDC 24V
• Digital Output:	8xDC 24V, 0.5A
• Technological functions:	2 Channels

Each of the digital in-/ outputs monitors its state via a LED. Via the parameterization you may assign alarm properties to every digital input. Additionally the digital inputs are parameterizable as counter.

X11:



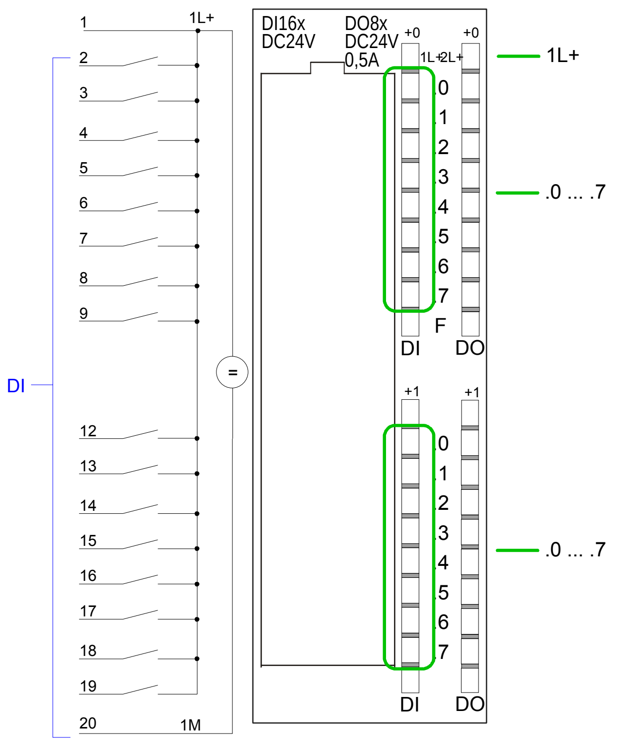
Attention!

Please take care that the voltage at an output channel always is ≤ the supply voltage via L+.

CPU 312SC: Pin assignment and status indicator

Pin	Assignment
1	Power supply +DC 24V
2	I+0.0 / Channel 0 (A)/Pulse
3	I+0.1 / Channel 0 (B)/Direction
4	I+0.2 / Channel 0 HW gate
5	I+0.3 / Channel 1 (A)/Pulse
6	I+0.4 / Channel 1 (B)/Direction
7	I+0.5 / Channel 1 HW gate
8	I+0.6
9	I+0.7
10	not connected
11	not connected
12	I+1.0
13	I+1.1
14	I+1.2
15	I+1.3
16	I+1.4 / Channel 0 Latch
17	I+1.5 / Channel 1 Latch
18	I+1.6
19	I+1.7
20	Ground 1M DI

Connection **LEDs**

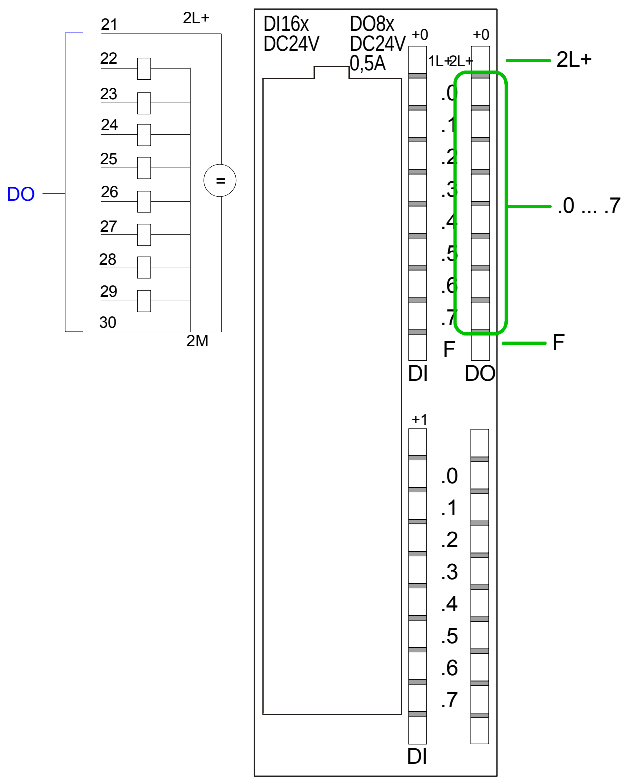


DI:		
L1+		LED (green) Supply voltage available for DI
.0 7		LEDs (green) I+0.0 to I+0.7 resp. I+1.0 to I+1.7 Starting with app. 15V the signal "1" at the input is recognized and the according LED

CPU 312SC: Pin assignment and status indicator

Pin	Assignment
21	Power supply +DC 24V
22	Q+0.0 / Channel 0 Output
23	Q+0.1 / Channel 1 Output
24	Q+0.2 / Channel 2 Output
25	Q+0.3
26	Q+0.4
27	Q+0.5
28	Q+0.6
29	Q+0.7
30	Ground 2M DO
31	not connected
32	not connected
33	not connected
34	not connected
35	not connected
36	not connected
37	not connected
38	not connected
39	not connected
40	not connected

Connection **LEDs**



DO:	
L2+	LED (green) Supply voltage available for DO
0 7	LEDs (green) Q+0.0 to Q+0.7 on at active output
F	LED (red) Overload or short circuit error

Address assignment

Input range

Sub module	Default-Address	Access	Assignment
DI10/DO6	124	Byte	Digital Input I+0.0 ... I+0.7
	125	Byte	Digital Input I+1.0 ... I+1.7
Counter	768	DInt	Channel 0: Count value / Frequency value
	772	DInt	Channel 1: Count value / Frequency value
	776	DInt	reserved
	780	DInt	reserved

Output range

Sub module	Default-Address	Access	Assignment
DI10/DO6	124	Byte	Digital Output Q+0.0 ... Q+0.7
Counter	768	DWord	reserved
	772	DWord	reserved
	776	DWord	reserved
	780	DWord	reserved

Digital part

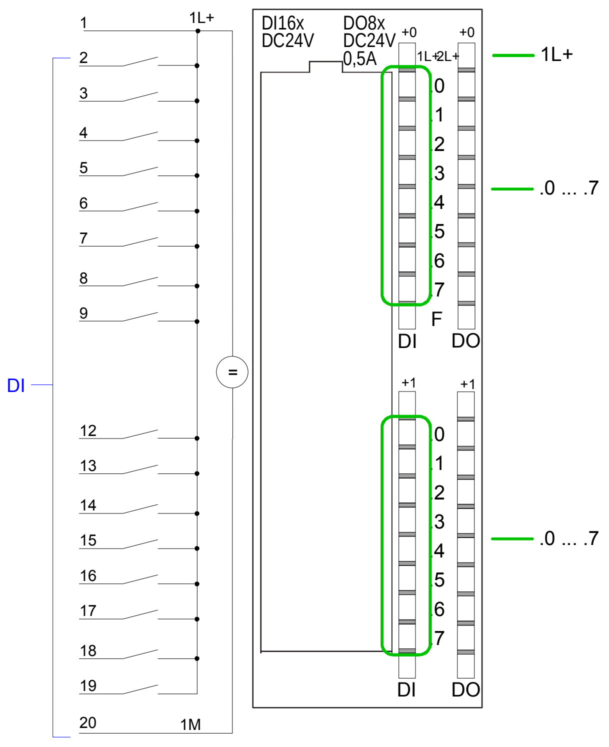
Digital part *CPU 312SC*

The digital part consists of 16 input -, 8 output channels and 2 channels for technological functions. Each of these digital input- respectively output channels show its state via a LED. By means of the parameterization you may assign interrupt properties to the inputs I+0.0 to I+1.7.

CPU 312SC: Pin assignment and status indicator

Pin	Assignment
1	Power supply +DC 24V
2	I+0.0 / Channel 0 (A)/Pulse
3	I+0.1 / Channel 0 (B)/Direction
4	I+0.2 / Channel 0 HW gate
5	I+0.3 / Channel 1 (A)/Pulse
6	I+0.4 / Channel 1 (B)/Direction
7	I+0.5 / Channel 1 HW gate
8	I+0.6
9	I+0.7
10	not connected
11	not connected
12	I+1.0
13	I+1.1
14	I+1.2
15	I+1.3
16	I+1.4 / Channel 0 Latch
17	I+1.5 / Channel 1 Latch
18	I+1.6
19	I+1.7
20	Ground 1M DI

Connection **LEDs**



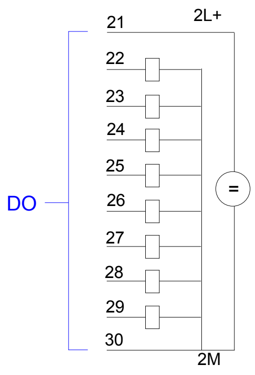
DI:

L1+	LED (green) Supply voltage available for DI
.0 7	LEDs (green) I+0.0 to I+0.7 resp. I+1.0 to I+1.7 Starting with app. 15V the signal "1" at the input is recognized and the according LED

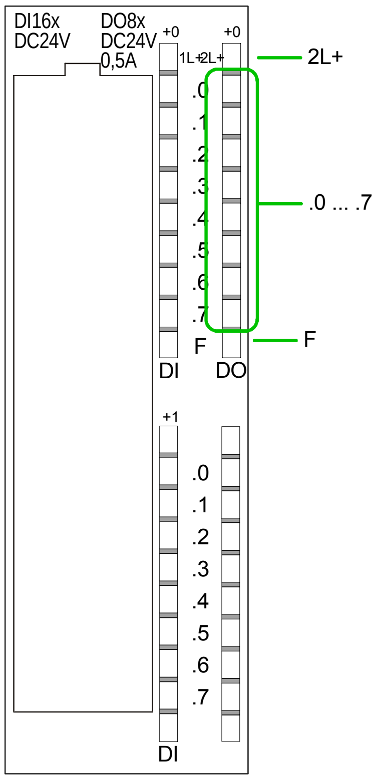
CPU 312SC: Pin assignment and status indicator

Pin	Assignment
21	Power supply +DC 24V
22	Q+0.0 / Channel 0 Output
23	Q+0.1 / Channel 1 Output
24	Q+0.2 / Channel 2 Output
25	Q+0.3
26	Q+0.4
27	Q+0.5
28	Q+0.6
29	Q+0.7
30	Ground 2M DO
31	not connected
32	not connected
33	not connected
34	not connected
35	not connected
36	not connected
37	not connected
38	not connected
39	not connected
40	not connected

Connection



LEDs



DO: L2+	LED (green) Supply voltage available for DO
0 7	LEDs (green) Q+0.0 to Q+0.7 resp. Q+1.0 to Q+1.7 on at active output
F	LED (red) Overload or short circuit error

Access to the digital part

The CPU 312SC creates in its peripheral area an area for input respectively output data. Without a hardware configuration the in the following specified default addresses are used.

Input range

Sub module	Default address	Access	Assignment
DI10/DO8	124	Byte	Digital Input I+0.0 ... I+0.7
	125	Byte	Digital Input I+1.0 ... I+1.7
Count	768	DInt	Channel 0: Count value / Frequency value
	772	DInt	Channel 1: Count value / Frequency value
	776	DInt	reserved
	780	DInt	reserved

Output range

Sub module	Default address	Access	Assignment
DI10/DO8	124	Byte	Digital Output Q+0.0 ... Q+0.7
Count	768	DWord	reserved
	772	DWord	reserved
	776	DWord	reserved
	780	DWord	reserved

Digital part - Parameterization

Parameter data

Parameters of the digital part may be set by means of the *DI10/DO6* submodule of the CPU 312C from Siemens during hardware configuration.

In the following all parameters are specified, which may be used with the hardware configuration of the digital periphery.

General

This provides the short description of the digital periphery. At *Comment* information about the module such as purpose may be entered.

Addresses

At this register the start address of the in-/output periphery may be set.

Inputs

Here there are the following adjustment possibilities:

- Hardware interrupt
- Input delay

For the digital output channels there are no parameters.

Hardware interrupt

A hardware interrupt may be optionally triggered on the rising or falling edge of an input.

A diagnostic interrupt is only supported together with hardware interrupt lost.

Select with the arrow keys the input and activate the desired hardware interrupt.

Input delay

The input delay may be configured per channel in groups of four. Please note that in the parameter window only the value 0.1ms may be set. At the other values 0.35ms is internally used for input delay.

Counter - Fast introduction

Overview

The CPU 312SC has in-/outputs, which may be used for technological functions respectively as standard periphery. Technological functions and standard I/O may be used simultaneously with appropriate hardware.

Read access to inputs used by technological functions is possible. Write access to used outputs is not possible.

The parameterization of the corresponding channel is made in the hardware configurator by means of the *Count* submodule of the CPU 312C from Siemens.

Now the following technological functions at 2 channels are at the disposal:

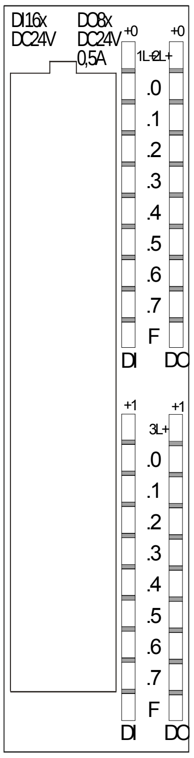
- Continuous count, e.g. for position decoding with Incremental encoder
- Single count, e.g. for unit decoding to a maximum limit
- Periodical count, e.g. for applications with repeated counting operations

Independent of the number of activated counters for the CPU 312SC the maximum frequency amounts to 10kHz.

The controlling of the appropriate modes of operation is made from the user program by the SFB COUNT (SFB 47).

Pin assignment

Pin	Assignment
1	Power supply +DC 24V
2	I+0.0 / Ch. 0 (A)/Pulse
3	I+0.1 / Ch. 0 (B)/Direction
4	I+0.2 / Ch. 0 Hardware gate
5	I+0.3 / Ch. 1 (A)/ Pulse
6	I+0.4 / Ch. 1 (B)/ Direction
7	I+0.5 / Ch. 1 Hardware gate
8	I+0.6
9	I+0.7
10	not connected
11	not connected
12	I+1.0
13	I+1.1
14	I+1.2
15	I+1.3
16	I+1.4 / Channel 0 Latch
17	I+1.5 / Channel 1 Latch
18	I+1.6
19	I+1.7
20	Ground DI



Pin	Assignment
21	Power supply +DC 24V
22	Q+0.0 / Channel 0 Output
23	Q+0.1 / Channel 1 Output
24	Q+0.2
25	Q+0.3
26	Q+0.4
27	Q+0.5
28	Q+0.6
29	Q+0.7
30	Ground DO
31	not connected
32	not connected
33	not connected
34	not connected
35	not connected
36	not connected
37	not connected
38	not connected
39	not connected
40	not connected

Preset respectively parameterize counter

The counter signal is detected and evaluated during counting operation. Every counter occupies one double word in the input range for the *counter register*. In the operating modes "single count" and "periodical count" an end respectively start value may be defined according to the counting direction up respectively down.

Each counter has parameterizable additional functions as gate function, latch function, comparison value, hysteresis and hardware interrupt.

Each counter parameter may be set by the *Count* submodule of the Siemens CPU 312C. Here is defined among others:

- Interrupt behavior
- max. Frequency
- Counter mode respectively behavior
- Stat, end, comparison value and hysteresis

Controlling the counter functions

The SFB COUNT (SFB 47) should cyclically be called (e.g. OB 1) for controlling the counter functions. The SFB is to be called with the corresponding instance DB. Here the parameters of the SFB are stored.

Among others the SFB 47 contains a request interface. Hereby you get read and write access to the registers of the appropriate counter.

So that a new job may be executed, the previous job must have be finished with JOB_DONE = TRUE. Per channel you may call the SFB in each case with the same instance DB, since the data necessary for the internal operational are stored here. Writing accesses to outputs of the instance DB is not permissible.

Note!

You must not call an SFB you have configured in your program in another program section under another priority class, because the SFB must not interrupt itself.

Example: It is not allowed to call the same SFB both in OB 1 and in the interrupt OB.

Controlling the counter

The counter is controlled by the internal gate (i gate). The i gate is the result of logic operation of hardware gate (HW gate) and software gate (SW gate), where the HW gate evaluation may be deactivated by the parameterization.

HW gate:	open (activate):	Edge 0-1 at hardware gate$_x$ input of the module
	close (deactivate):	Edge 1-0 at hardware gate$_x$ input of the module
SW gate:	open (activate):	In application program by setting SW_GATE of the SFB 47
	close (deactivate):	In application program by resetting SW_GATE of the SFB 47

Read counter

The counter values may be read by the output parameter COUNTVAL of the SFB 47. There is also the possibility for direct access to the counter values by means of the input address of the *Count* submodule.

Counter inputs (Connections)

There are the following possibilities for connection to the technological functions:

- 24V incremental encoder, equipped with two tracks with 90° phase offset
- 24V pulse generator with direction signal
- 24V proximity switch (e.g. BERO or light barrier)

For not all inputs are available at the same time, you may set the input assignment for every counter via the parameterization. For each counter the following inputs are available:

Channel$_x$ (A)

Pulse input for count signal res. track A of an encoder. Here you may connect encoder with 1-, 2- or 4-tier evaluation.

Channel$_x$ (B)

Direction signal res. track B of the encoder. Via the parameterization you may invert the direction signal.

Hardware gate$_x$

This input allows you to open the HW gate with a high peek and thus start a count process. The usage of the HW gate may be parameterized.

Latch$_x$

With an edge 0-1 at Latch$_x$ the recent counter value is stored in a memory that you may read at need.

Counter outputs

Every counter has an assigned output channel. The following behavior for the output channel may be set via parameterization:

- No comparison: Output is not controlled and is switched in the same way as a normal output.
- Count value ≥ comparison value:
  Output is set as long as counter value ≥ comparison value.
- Count value ≤ comparison value:
  Output is set as long as counter value ≤ comparison value.
- Pulse at comparison value: You can specify a pulse period for adaptation to the actuators you are using. The output is set for the given pulse duration, as soon as the counter reached the comparison value. If you have parameterized a main count direction the output is only set when reaching the comparison value from the main counting direction. The maximum pulse duration may amount to 510ms. By setting 0 as *pulse duration* the output gets set as long as the comparison conditions are fulfilled.

Parameter overview

In the following the parameters are listed which may be used for counter configuration during hardware configuration.

General

Here the short description of the counter function may be found. At *Comment* information about the module such as purpose may be entered.

Addresses

Here the start address of the in- output periphery is set.

Basic parameters

Here the interrupts the counter functions should trigger may be selected. You have the following options:

- None: There is no interrupt triggered.
- Process: The counting function triggers a hardware interrupt.
- Diagnostics and Process: With the CPU 312SC the diagnostic interrupt of the digital in-/output periphery is only supported in connection with "hardware interrupt lost".

Count

Parameters	Description	Range of values	Default
Main count direction	• *None*: No restriction of the counting range • *Up*: Restricts the up-counting range. Counter starts at 0 or load value, counts in positive direction up to the declaration end value -1 and then jumps back to load value at the next positive transducer pulse. • *Down*: Restricts the down-counting range. The Counter starts at the declared start value or load value in negative direction, counts to 1 and then jumps to start value at the next negative encoder pulse.	• None • Up • Down (not with continuous count)	None
End value/ Start value	*End value,* with up-count as default. *Start value,* with down-count as default.	2...2147483647 (2^{31}-1)	2147483647 (2^{31}-1)
Gate function	• *Cancel count*: The count starts when the gate opens and resumes at the load value when the gate opens again. • *Stop count*: The count is interrupted when the gate closes and resumed at the last actual value when the gate opens again.	• Abort the count operation • Interrupt the count operation	Cancel count
Comparison value	The count value is compared with the comparison value. see also the parameter "Characteristics of the output": • No main direction of count • Up-count as default • Down-count as default	-2^{31} to $+2^{31}$-1 -2^{31} to End value-1 1 to $+2^{31}$-1	0
Hysteresis	A hysteresis is used to eliminate frequent output jitter if the count value lies within the range of the comparison value. 0 and 1 means: Hysteresis switched off	0 to 255	0
max. frequency: counting signals/hard-ware gate	You can set the maximum frequency of the track A/pulse, track B/direction and hardware gate signals in fixed steps.	10, 5, 2, 1kHz	10kHz

continue ...

... continued

Parameters	Description	Range of value	Default
max. frequency: Latch	You can set the maximum frequency of the latch signal in fixed steps.	10, 5, 2, 1kHz	10kHz
Signal evaluation	The count and direction signals are connected to the input. A rotary transducer is connected to the input (single, dual or quadruple evaluation).	• Pulse/Direction • Rotary encoder single • Rotary encoder, double • Rotary encoder quadruple	Pulse/Direction
Hardware gate	In the activated state the Gate control is made via SW-gate and HW-gate, otherwise via SW-gate only.	• activated • deactivated	deactivated
Count direction inverted	In the activated state the "direction" input signal is inverted.	• activated • deactivated	deactivated
Characteristics of the output	The output and the "Comparator" (STS_CMP) status bit are set, dependent on this parameter.	• No comparison • Count ≥ comparison value • Count ≤ comparison value • Pulse at comparison value	No comparison
Pulse duration	With the setting "Characteristics of the output: Pulse at comparison value" the pulse duration of the output signal may be specified. Only even values are possible. The value is internal multiplied with 1.024ms.	0 to 510	0
Hardware interrupt: Hardware gate opening	In the activated state a hardware interrupt is generated when the hardware gate opens while the software gate is open.	• activated • deactivated	
Hardware interrupt: Hardware gate closing	In the activated state a hardware interrupt is generated when the hardware gate closes while the software gate is open.	• activated • deactivated	deactivated
Hardware interrupt: On reaching comparator	In the activated state a hardware interrupt is triggered on reaching the comparator (reaction) value. The process interrupt may only be released if in addition the value of "Characteristics of the output" is <u>not</u> "no comparison".	• activated • deactivated	deactivated
Hardware interrupt: Overflow	In the activated state a hardware interrupt is generated in the event of an overflow (exceeding the upper count limit).	• activated • deactivated	deactivated
Hardware interrupt: Underflow	In the activated state a hardware interrupt is generated in the event of an underflow (undershooting the lower count limit).	• activated • deactivated	deactivated

Counter - Controlling

Overview

The controlling of the appropriate counter is made from the user program by the SFB COUNT (SFB 47). The SFB is to be called with the corresponding instance DB. Here the parameters of the SFB are stored.

With the SFB COUNT (SFB 47) you have following functional options:

- Start/Stop the counter via software gate SW_GATE
- Enable/control output DO
- Read the status bit
- Read the actual count and latch value
- Request to read/write internal counter registers

Parameter SFB 47

Name	Decla-ration	Data type	Address (Inst.-DB)	Default value	Comment
LADDR	INPUT	WORD	0.0	300h	This parameter is not evaluated. Always the internal I/O periphery is addressed.
CHANNEL	INPUT	INT	2.0	0	Channel number
SW_GATE	INPUT	BOOL	4.0	FALSE	Enables the Software gate
CTRL_DO	INPUT	BOOL	4.1	FALSE	Enables the output False: Standard Digital Output
SET_DO	INPUT	BOOL	4.2	FALSE	Parameter is not evaluated
JOB_REQ	INPUT	BOOL	4.3	FALSE	Initiates the job (edge 0-1)
JOB_ID	INPUT	WORD	6.0	0	Job ID
JOB_VAL	INPUT	DINT	8.0	0	Value for write jobs
STS_GATE	OUTPUT	BOOL	12.0	FALSE	Status of the internal gate
STS_STRT	OUTPUT	BOOL	12.1	FALSE	Status of the hardware gate (is only refreshed if "HW gate" is activated in hardware configuration before)
STS_LTCH	OUTPUT	BOOL	12.2	FALSE	Status of the latch input
STS_DO	OUTPUT	BOOL	12.3	FALSE	Status of the output
STS_C_DN	OUTPUT	BOOL	12.4	FALSE	Status of the down-count Always indicates the last direction of count. After the first SFB call STS_C_DN is set FALSE.
STS_C_UP	OUTPUT	BOOL	12.5	FALSE	Status of the up-count Always indicates the last direction of count. After the first SFB call STS_C_UP is set TRUE.
COUNTVAL		DINT	14.0	0	Actual count value
LATCHVAL		DINT	18.0	0	Actual latch value
JOB_DONE		BOOL	22.0	TRUE	New job can be started.
JOB_ERR		BOOL	22.1	FALSE	Job error
JOB_STAT		WORD	24.0	0	Job error ID

Local data only in instance DB

Name	Data type	Address (Instance DB)	Default value	Comment
RES00	BOOL	26.0	FALSE	reserved
RES01	BOOL	26.1	FALSE	reserved
RES02	BOOL	26.2	FALSE	reserved
STS_CMP	BOOL	26.3	FALSE	Comparator Status *) Status bit STS_CMP indicates that the comparison condition of the comparator is or was reached. STS_CMP also indicates that the output was set. (STS_DO = TRUE). This parameter is only refreshed if in the hardware configuration a comparison value is set at "Characteristics of the output".
RES04	BOOL	26.4	FALSE	reserved
STS_OFLW	BOOL	26.5	FALSE	Overflow status - is only set at range overflow *)
STS_UFLW	BOOL	26.6	FALSE	Underflow status - is only set at range underflow *)
STS_ZP	BOOL	26.7	FALSE	Status of the zero mark *) The bit is only set when counting without main direction. Indicates the zero mark. This is also set when the counter is set to 0 or if is start counting.
JOB_OVAL	DINT	28.0		Output value for read request.
RES10	BOOL	32.0	FALSE	reserved
RES11	BOOL	32.1	FALSE	reserved
RES_STS	BOOL	32.2	FALSE	Reset status bits: Resets the status bits: STS_CMP, STS_OFLW, STS_ZP. The SFB must be twice to reset the status bit.

*) Reset with RES_STS

Note!

Per channel you may call the SFB in each case with the same instance DB, since the data necessary for the internal operational are stored here. Writing accesses to outputs of the instance DB is not permissible.

Counter request interface

To read/write counter registers the request interface of the SFB 47 may be used.

So that a new job may be executed, the previous job must have be finished with JOB_DONE = TRUE.

Proceeding

The deployment of the request interface takes place at the following sequence:

- Edit the following input parameters:

Name	Data type	Address (DB)	Default	Comment
JOB_REQ	BOOL	4.3	FALSE	Initiates the job (edges 0-1)
JOB_ID	WORD	6.0	0	Job ID: 00h Job without function 01h Writes the count value 02h Writes the load value 04h Writes the comparison value 08h Writes the hysteresis 10h Writes the pulse duration 20h Writes the end value 82h Reads the load value 84h Reads the comparison value 88h Reads the hysteresis 90h Reads the pulse duration A0h Reads the end value
JOB_VAL	DINT	8.0	0	Value for write jobs (see table at the following page)

- Call the SFB. The job is processed immediately. JOB_DONE only applies to SFB run with the result FALSE. JOB_ERR = TRUE if an error occurred. Details on the error cause are indicated at JOB_STAT.

Name	Data type	Address (DB)	Default	Comment
JOB_DONE	BOOL	22.0	TRUE	New job can be started
JOB_ERR	BOOL	22.1	FALSE	Job error
JOB_STAT	WORD	24.0	0000h	Job error ID 0000h No error 0121h Compare value too low 0122h Compare value too high 0131h Hysteresis too low 0132h Hysteresis too high 0141h Pulse duration too low 0142h Pulse duration too high 0151h Load value too low 0152h Load value too high 0161h Count value too low 0162h Count value too high 01FFh Invalid job ID

- A new job may be started with JOB_DONE = TRUE.
- A value to be read of a read job may be found in JOB_OVAL in the instance DB at address 28.

Permitted value range for JOB_VAL

Continuous count:

Job	Valid range
Writing counter directly	-2147483647 (-2^{31}+1) to +2147483646 (2^{31}-2)
Writing the load value	-2147483647 (-2^{31}+1) to +2147483646 (2^{31}-2)
Writing comparison value	-2147483648 (-2^{31}) to +2147483647 (2^{31}-1)
Writing hysteresis	0 to 255
Writing pulse duration*	0 to 510ms

Single/periodic count, no main count direction:

Job	Valid range
Writing counter directly	-2147483647 (-2^{31}+1) to +2147483646 (2^{31}-2)
Writing the load value	-2147483647 (-2^{31}+1) to +2147483646 (2^{31}-2)
Writing comparison value	-2147483648 (-2^{31}) to +2147483647 (2^{31}-1)
Writing hysteresis	0 to 255
Writing pulse duration*	0 to 510ms

Single/periodic count, main count direction up:

Job	Valid range
End value	2 to +2147483646 (2^{31}-1)
Writing counter directly	-2147483648 (-2^{31}) to end value -2
Writing the load value	-2147483648 (-2^{31}) to end value -2
Writing comparison value	-2147483648 (-2^{31}) to end value -1
Writing hysteresis	0 to 255
Writing pulse duration*	0 to 510ms

Single/periodic count, main count direction down:

Job	Valid range
Writing counter directly	2 to +2147483647 (2^{31}-1)
Writing the load value	2 to +2147483647 (2^{31}-1)
Writing comparison value	1 to +2147483647 (2^{31}-1)
Writing hysteresis	0 to 255
Writing pulse duration*	0 to 510ms

*) Only even values allowed. Odd values are automatically rounded.

Latch function

As soon as during a count process an edge 0-1 is recognized at the "Latch" input of a counter, the recent counter value is stored in the according latch register.

You may access the latch register via LATCHVAL of the SFB 47.

A just in LATCHVAL loaded value remains after a STOP-RUN transition.

Counter - Functions

Parameterization

- Start the Siemens SIMATIC Manager with your project and open the hardware configurator.
- Place a profile rail.
- Configure at slot 2 the corresponding CPU from Siemens CPU 31xC.
- Open the dialog window "Properties" by a double click to the *Count* submodule of the CPU.
- As soon as an operating mode to the corresponding channel is selected, a dialog window for this operating mode is created and displayed and filled with default parameters.
- Execute the wished parameterization.
- Store the project with **Station** > *Save and compile*.
- Transfer the project to the CPU.

Load value, End value

Via the parameterization you have the opportunity to define a main counting direction for every counter. If "none" or "endless" is chosen, the complete counting range is available:

Limit counter	Valid value range
Lower count limit	-2 147 483 648 (-2^{31})
Upper count limit	+2 147 483 647 ($2^{31}-1$)

Otherwise this range may be limited in both directions by a start value as *load value* and an *end value*.

Main counting direction

Main counting direction forward

Upper restriction of the count range. The counter counts 0 res. load value in positive direction until the parameterized end value –1 and jumps then back to the load value with the next following encoder pulse.

Please note a load value may exclusively be set by the request interface of the counter.

Main counting direction backwards

Lower restriction of the count range. The counter counts from the parameterized start- res. load value in negative direction to the parameterized end value +1 and jumps then back to the start value with the next following encoder pulse.

Please note an end value may exclusively be set by the request interface of the counter.

Count Continuously

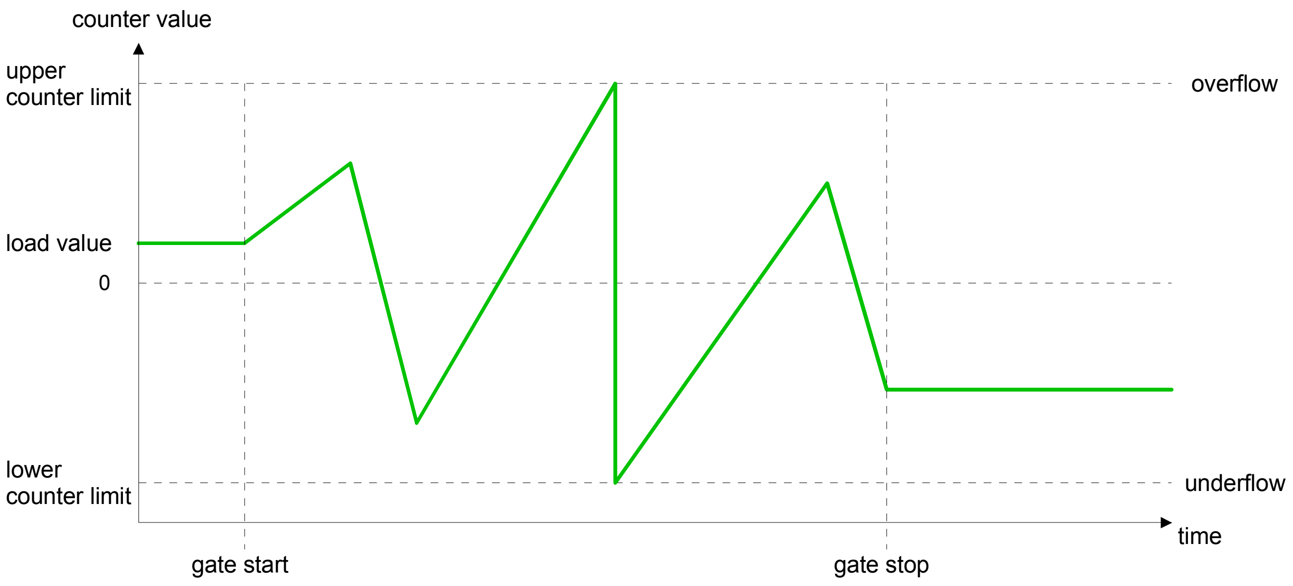
In this operating mode, the counter counts from 0 res. from the load value.

When the counter counts forward and reaches the upper count limit and another counting pulse in positive direction arrives, it jumps to the lower count limit and counts from there on.

When the counter counts backwards and reaches the lower count limit and another counting pulse in negative direction arrives, it jumps to the upper count limit and counts from there on.

The count limits are set to the maximum count range.

	Valid value range
Lower count limit	-2 147 483 648 (-2^{31})
Upper count limit	+2 147 483 647 ($2^{31}-1$)



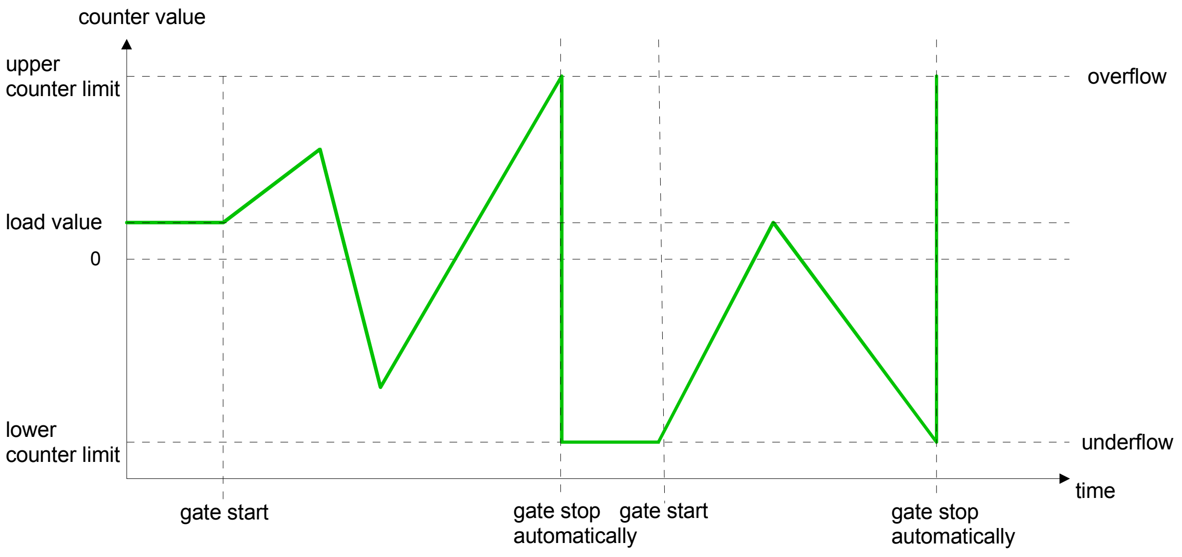
Count Once

No main counting direction

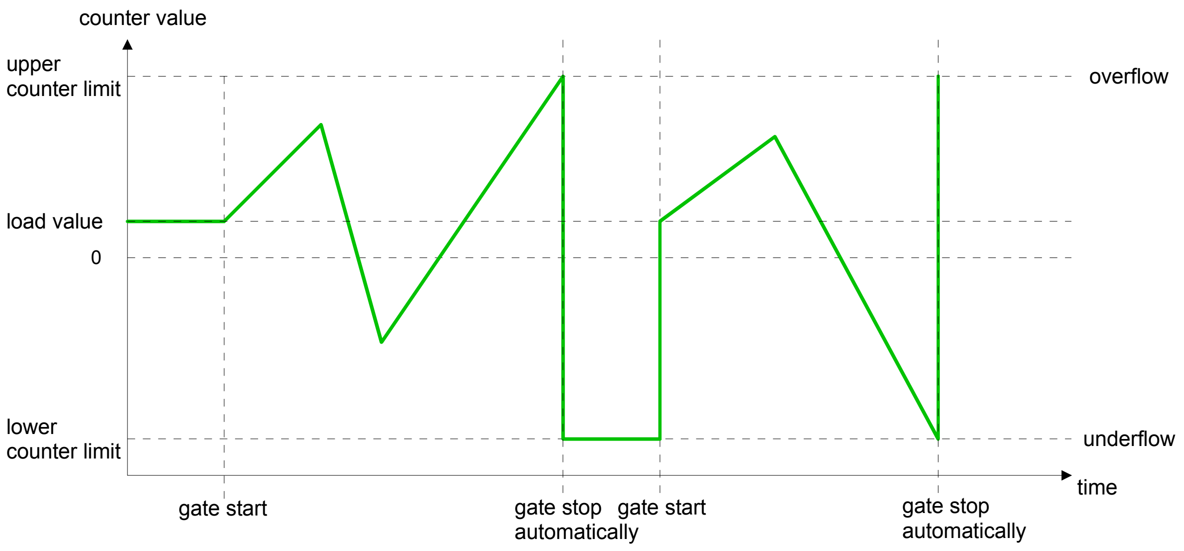
- The counter counts once starting with the load value.
- You may count forward or backwards.
- The count limits are set to the maximum count range.
- At over- or underrun at the count limits, the counter jumps to the according other count limit and the gate is automatically closed.
- To restart the count process, you must create an edge 0-1 of the gate.
- At interrupting gate control, the count process continuous with the last recent counter value.
- At aborting gate control, the counter starts with the load value.

	Valid value range
Lower count limit	-2 147 483 648 (-2^{31})
Upper count limit	+2 147 483 647 ($2^{31}-1$)

Interrupting gate control:



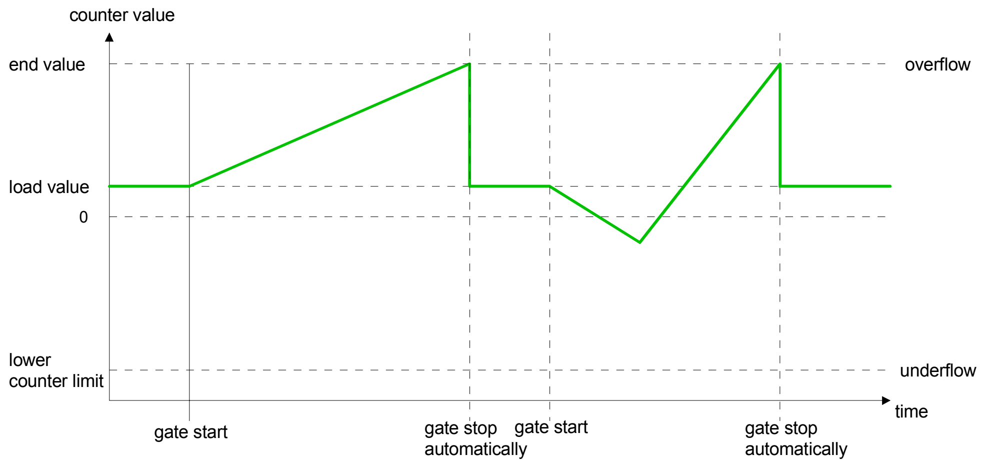
Aborting gate control:



Main counting direction forward

- The counter counts starting with the load value.
- When the counter reaches the end value –1 in positive direction, it jumps to the load value at the next positive count pulse and the gate is automatically closed.
- To restart the count process, you must create an edge 0-1 of the gate. The counter starts with the load value.

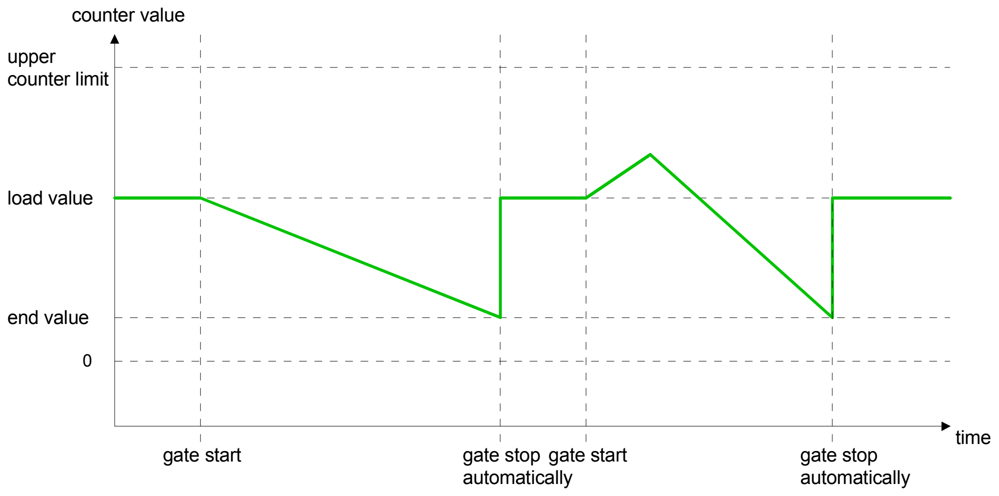
	Valid value range
Limit value	-2 147 483 647 (-2^{31}+1) to +2 147 483 647 (2^{31}-1)
Lower count limit	-2 147 483 648 (-2^{31})



Main counting direction backwards

- The counter counts backwards starting with the load value.
- When the counter reaches the end value +1 in negative direction, it jumps to the load value at the next negative count pulse and the gate is automatically closed.
- To restart the count process, you must create an edge 0-1 of the gate. The counter starts with the load value.

	Valid value range
Limit value	-2 147 483 648 (-2^{31}) to +2 147 483 646 (2^{31}-2)
Upper count limit	+2 147 483 647 (2^{31}-1)

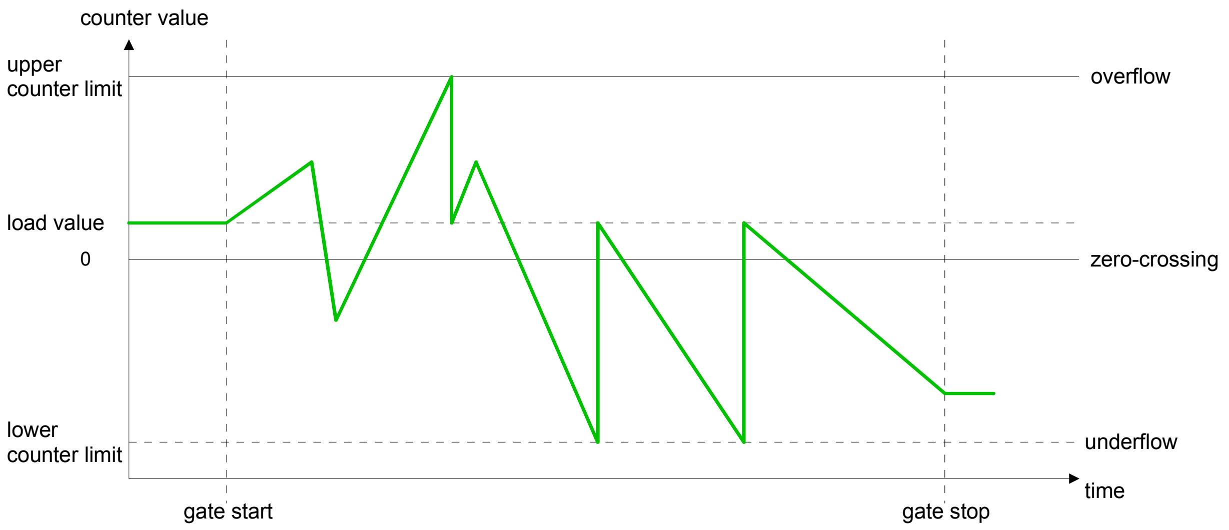


Count Periodically

No main counting direction

- The counter counts forward or backwards starting with the load value.
- At over- or underrun at the count limits, the counter jumps to the according other count limit and counts from there on.
- The count limits are set to the maximum count range.

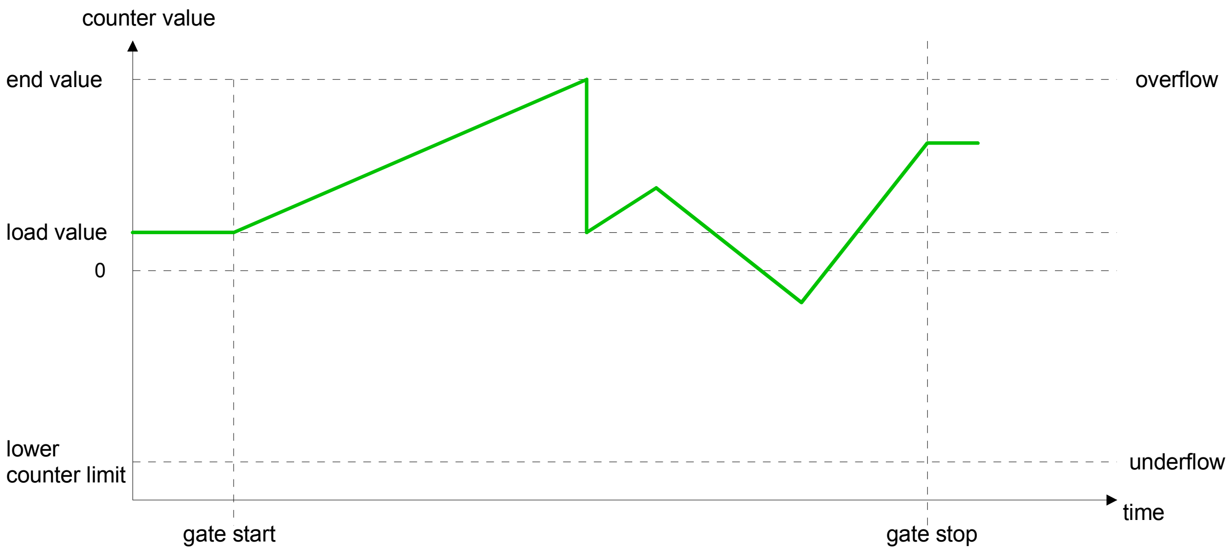
	Valid value range
Lower count limit	-2 147 483 648 (-2^{31})
Upper count limit	+2 147 483 647 (2^{31}-1)



Main counting direction forward

- The counter counts forward starting with the load value.
- When the counter reaches the end value -1 in positive direction, it jumps to the load value at the next positive count pulse.

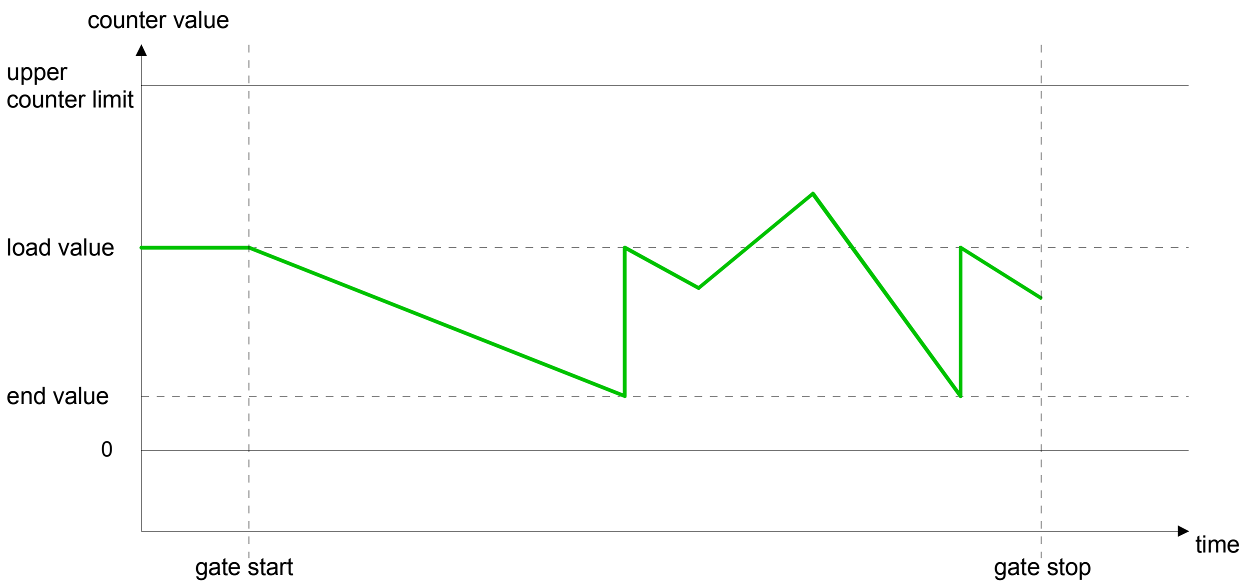
	Valid value range
Limit value	-2 147 483 647 (-2^{31}+1) to +2 147 483 647 (2^{31}-1)
Lower count limit	-2 147 483 648 (-2^{31})



Main counting direction backwards

- The counter counts backwards starting with the load value.
- When the counter reaches the end value +1 in negative direction, it jumps to the load value at the next negative count pulse.
- You may exceed the upper count limit.

	Valid value range
Limit value	-2 147 483 648 (-2^{31}) to +2 147 483 646 ($2^{31}-2$)
Upper count limit	+2 147 483 647 ($2^{31}-1$)



Counter - Additional functions

Overview

The following additional functions may be set via the parameterization for every counter:

- Gate function

  The gate function serves the start, stop and interrupt of a count function.
- Latch function

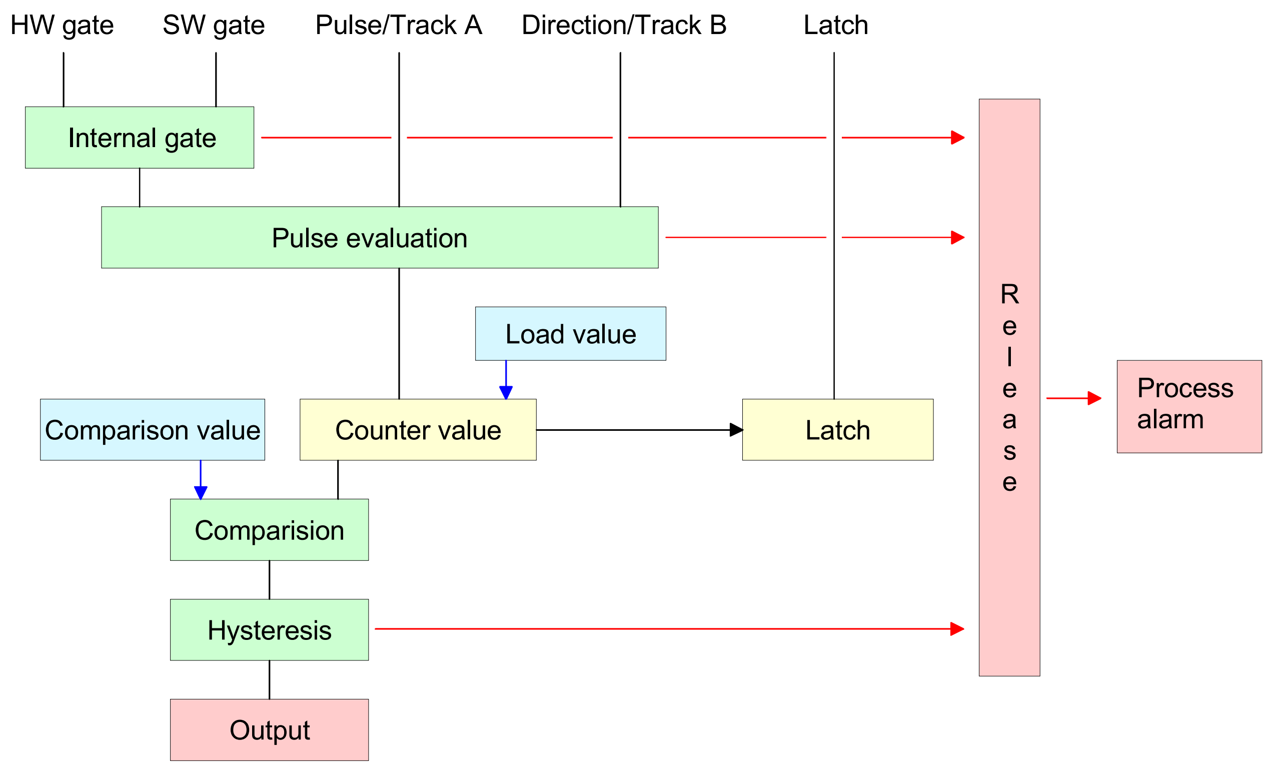
  An edge 0-1 at the digital input "Latch" stores the recent counter value in the latch register.
- Comparison

  You may set a comparison value that activates res. de-activates a digital output res. releases a hardware interrupt depending on the counter value.
- Hysteresis

  The setting of a hysteresis avoids for example a high output toggling when the value of an encoder signal shifts around a comparison value.

Schematic structure

The illustration shows how the additional functions influence the counting behavior. The following pages describe these functions in detail:



Gate function

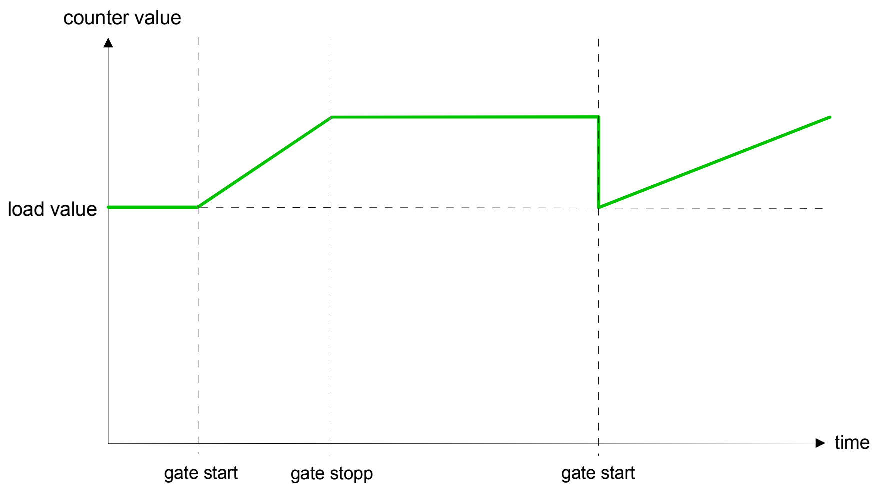
The counter is controlled by the internal gate (i gate). The i gate is the result of logic operation of hardware gate (HW gate) and software gate (SW gate), where the HW gate evaluation may be deactivated by the parameterization.

HW gate: open (activate):	Edge 0-1 at hardware gate$_x$ input of the module
close (deactivate):	Edge 1-0 at hardware gate$_x$ input of the module
SW gate: open (activate):	In application program by setting SW_GATE of the SFB 47
close (deactivate):	In application program by resetting SW_GATE of the SFB 47

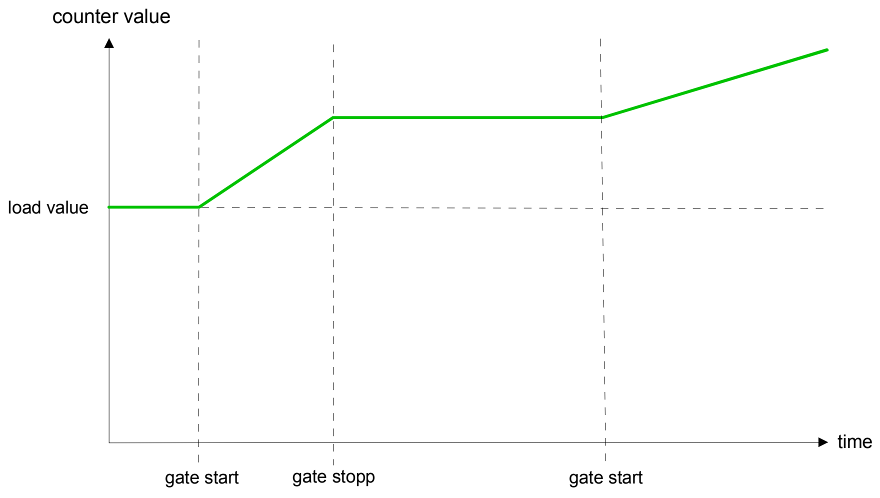
Gate function cancel and stop

The parameterization defines if the gate cancels or stops the counter process.

- At *cancel function* the counter starts counting with the load value after gate restart.



- At *stop function*, the counter continues counting with the last recent counter value after gate restart.



Gate control abort, interruption

How the CPU should react at opening of the SW gate may be set with the parameter *Gate function*. The usage of the HW gate may be determined by the parameter *Hardware gate*.

Gate control via SW gate, canceling
(HW gate deactivated, gate function: Cancel count)

SW gate	HW gate	Reaction Counter
edge 0-1	de-activated	Restart with load value

Gate control via SW gate, stopping
(HW gate deactivated, gate function: Stop count)

SW gate	HW gate	Reaction Counter
edge 0-1	de-activated	Continue

Gate control via SW/HW gate, canceling
(HW gate activated, gate function: Cancel count)

SW gate	HW gate	Reaction Counter
edge 0-1	1	Continue
1	edge 0-1	Restart with load value

Gate control via SW/HW gate, stopping
(HW gate activated, gate function: Stop count)

SW gate	HW gate	Reaction Counter
edge 0-1	1	Continue
1	edge 0-1	Continue

Gate control "Count once"

Gate control via SW/HW gate, operating mode "Count once"

If the internal gate has been closed automatically it may only be opened again under the following conditions:

SW gate	HW gate	Reaction I gate
1	edge 0-1	1
edge 0-1 (after edge 0-1 at HW gate)	edge 0-1	1

Latch function

As soon as during a count process an edge 0-1 is recognized at the "Latch" input of a counter, the recent counter value is stored in the according latch register.

The latch value may be accessed by the parameter LATCHVAL of the SFB 47.

A just in LATCHVAL loaded value remains after a STOP-RUN transition.

Comparator

In the CPU a comparison value may be stored that is assigned to the digital output, to the status bit "Status Comparator" STS_CMP and to the hardware interrupt. The digital output may be activated depending on the count value and comparison value. A comparison value may be entered by the parameter assignment screen form respectively by the request interface of the SFB 47.

Characteristics of the output

You pre-define the behavior of the counter output via the parameterization:

- output never switches
- output switch when counter value ≥ comparison value
- output switch when counter value ≤ comparison value
- output switch at comparison value

No comparison

The output is set as normal output. The SFB input parameter CTRL_DO is effect less. The status bits STS_DO and STS_CMP (Status comparator in the instance DB) remain reset.

Count ≥ comparison value respectively *Count ≤ comparison value*

The output remains set as long as the counter value is higher or equal comparison value respectively lower or equal comparison value. For this the control bit must be set.

The comparison result is shown by the status bit STS_CMP.

This status bit may only be reset if the comparison condition is no longer fulfilled.

Pulse at comparison value

When the counter reaches the comparison value the output is set for the parameterized pulse duration. If you have configured a main count direction the output is only activated when the comparison value is reached with the specified main count direction. For this the control bit CTRL_DO should be set first.

The status of the digital output may be shown by the status bit ST_DO.
The comparison result is shown by the status bit STS_CMP. This status bit may only be reset if the pulse duration has run off. comparison condition is no longer fulfilled.

With pulse time = 0 the output is as set as the comparison condition is fulfilled.

Pulse duration

For adaptation to the used actors a pulse duration may be specified. The pulse duration defines how long the output should be set. It may be preset in steps of 2ms between 0 and 510ms. The pulse duration starts with the setting of the according digital output. The inaccuracy of the pulse duration is less than 1ms.

There is no past triggering of the pulse duration when the comparison value has been left and reached again during pulse output. A change of the pulse period during runtime is not applied until the next pulse.

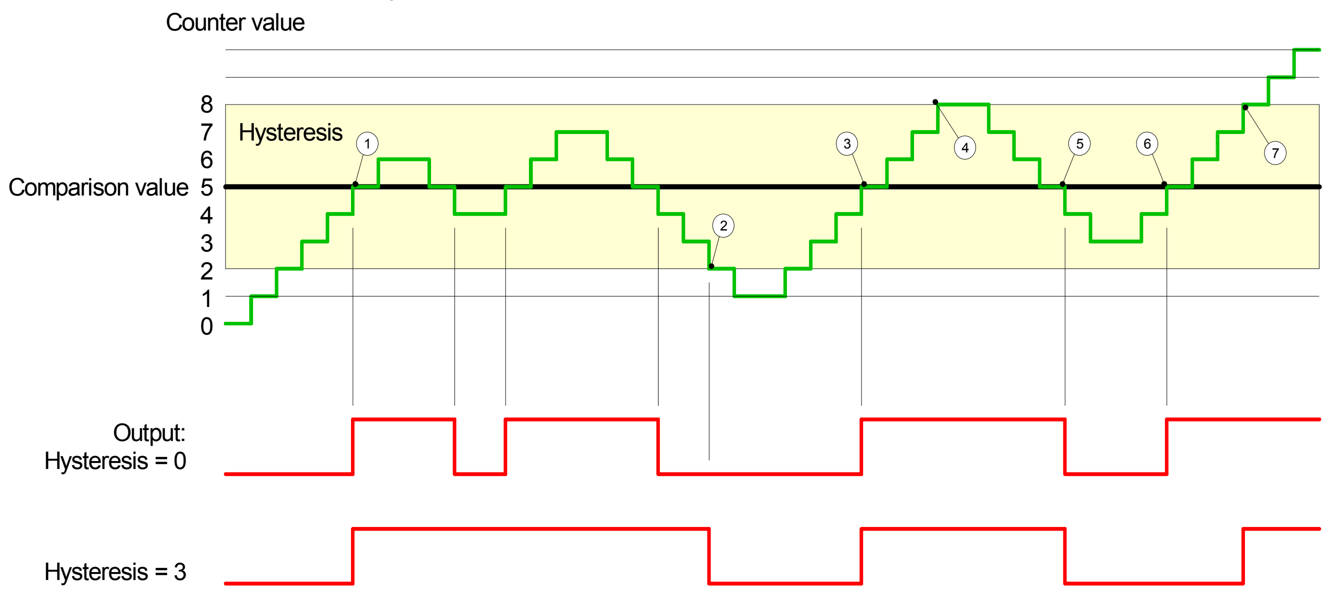
Hysteresis

The hysteresis serves e.g. the avoidance of many toggle processes of the output and the interrupt, if the counter value is in the range of the comparison value. You may set a range of 0 to 255. The settings 0 and 1 deactivate the hysteresis. The hysteresis influences the zero run, over- and underflow.

An activated hysteresis remains active after a change. The new hysteresis range is taken over at the next reach of the comparison value.

The following pictures illustrate the output behavior for hysteresis 0 and hysteresis 3 for the according conditions:

Effect at counter value ≥ comparison value



① Counter value ≥ comparison value → output is set and hysteresis activated

② Leave hysteresis range → output is reset

③ Counter value ≥ comparison value → output is set and hysteresis activated

④ Leave hysteresis range, output remains set for counter value ≥ comparison value

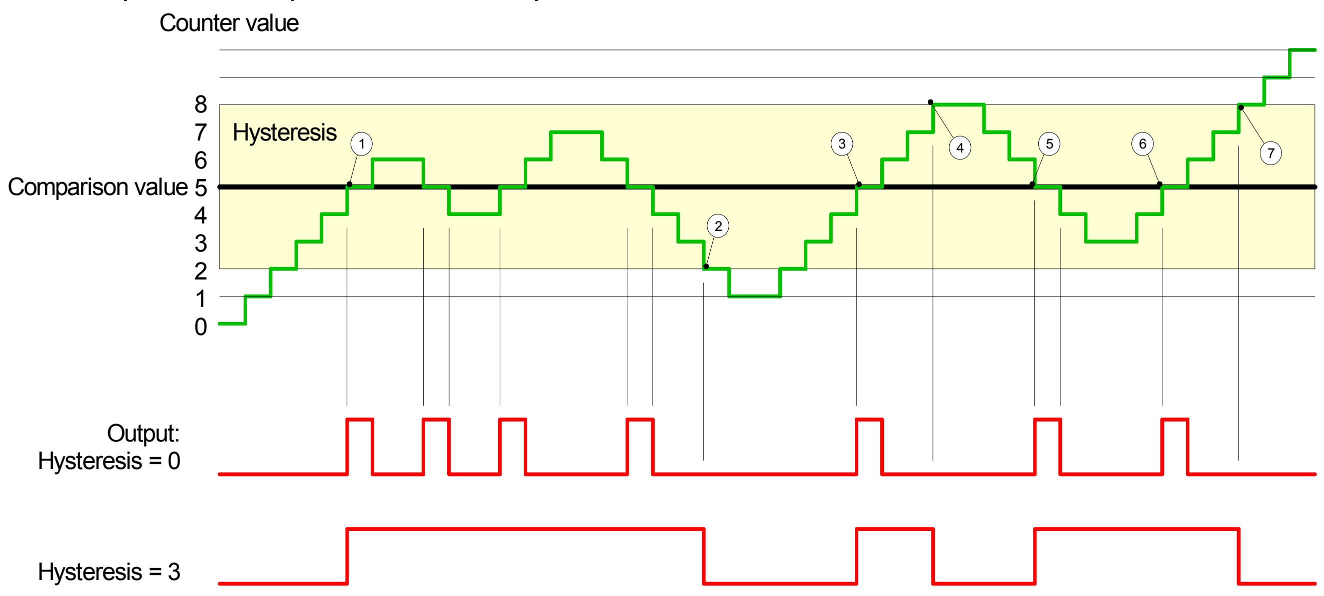
⑤ Counter value < comparison value and hysteresis active → output is reset

⑥ Counter value ≥ comparison value → output is not set for hysteresis active

⑦ Leave hysteresis range, output remains set for counter value ≥ comparison value

With reaching the comparison condition the hysteresis gets active. At active hysteresis the comparison result remains unchanged until the counter value leaves the set hysteresis range. After leaving the hysteresis range a new hysteresis is only activated with again reaching the comparison conditions.

Effect at pulse at comparison value with pulse duration Zero



① Counter value = comparison value → output is set and hysteresis activated

② Leave hysteresis range → output is reset and counter value < comparison value

③ Counter value = comparison value → output is set and hysteresis activated

④ Output is reset for leaving hysteresis range and counter value > comparison value

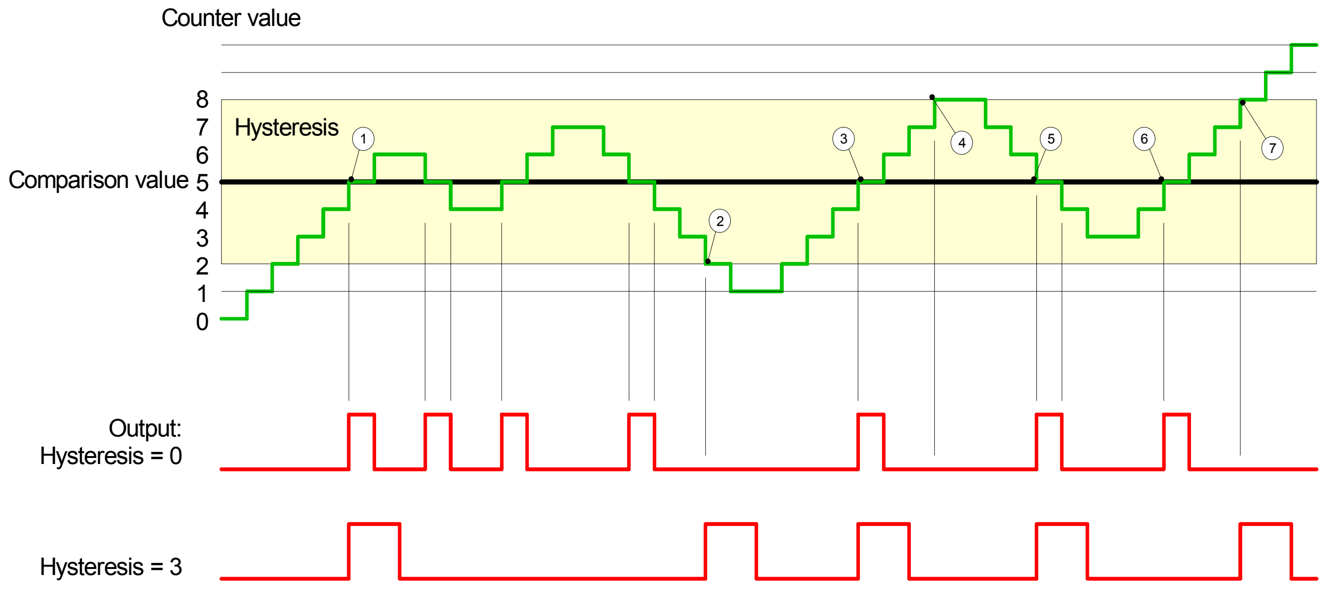
⑤ Counter value = comparison value → output is set and hysteresis activated

⑥ Counter value = comparison value and hysteresis active → output remains set

⑦ Leave hysteresis range and counter value > comparison value → output is reset

With reaching the comparison condition the hysteresis gets active. At active hysteresis the comparison result remains unchanged until the counter value leaves the set hysteresis range. After leaving the hysteresis range a new hysteresis is only activated with again reaching the comparison conditions.

Effect at pulse at comparison value with pulse duration not zero



① Counter value = comparison value → pulse of the parameterized duration is put out, the hysteresis is activated and the counting direction stored

② Leaving the hysteresis range contrary to the stored counting direction → pulse of the parameterized duration is put out, the hysteresis is de-activated

③ Counter value = comparison value → pulse of the parameterized duration is put out, the hysteresis is activated and the counting direction stored

④ Leaving the hysteresis range without changing counting direction → hysteresis is de-activated

⑤ Counter value = comparison value → pulse of the parameterized duration is put out, the hysteresis is activated and the counting direction stored

⑥ Counter value = comparison value and hysteresis active → no pulse

⑦ Leaving the hysteresis range contrary to the stored counting direction → pulse of the parameterized duration is put out, the hysteresis is de-activated

With reaching the comparison condition the hysteresis gets active and a pulse of the parameterized duration is put out. As long as the counter value is within the hysteresis range, no other pulse is put out. With activating the hysteresis the counting direction is stored in the CPU. If the counter value leaves the hysteresis range contrary to the stored counting direction, a pulse of the parameterized duration is put out. Leaving the hysteresis range without direction change, no pulse is put out.

Counter - Diagnostic and interrupt

Overview

The parameterization allows you to define the following trigger for a hardware interrupt that may initialize a diagnostic interrupt:

- Status changes at an input (at opened SW gate)
- Status changes at the HW-gate (at opened SW gate)
- Reaching a comparison value
- Overflow respectively at overrun upper counter limit
- Underflow respectively at underrun lower counter limit

Hardware interrupt

A hardware interrupt causes a call of the OB 40. Within the OB 40 you may find the logical basic address of the module that initialized the hardware interrupt by using the *Local word 6*. More detailed information about the initializing event is to find in the *local double word 8*.

Local double word 8 of the OB 40

The *local double word 8* of the OB 40 has the following structure:

Local byte	Bit 7 ... Bit 0
8	Bit 0: Edge at I+0.0 Bit 1: Edge at I+0.1 Bit 2: Edge at I+0.2 Bit 3: Edge at I+0.3 Bit 4: Edge at I+0.4 Bit 5: Edge at I+0.5 Bit 6: Edge at I+0.6 Bit 7: Edge at I+0.7
9	Bit 0: Edge at I+1.0 Bit 1: Edge at I+1.1 Bit 2: Edge at I+1.2 Bit 7 ... 3: reserved
10	Bit 0: Gate counter 0 open (activated) Bit 1: Gate counter 0 closed Bit 2: Over-/underflow/end value counter 0 Bit 3: Counter 0 reached comparison value Bit 4: Gate counter 1 open (activated) Bit 5: Gate counter 1 closed Bit 6: Over-/underflow/ end value counter 1 Bit 7: Counter 1 reached comparison value
11	Bit 7 ... 0: reserved

Diagnostic interrupt

Via the parameterization (record set 7Fh) you may activate a global diagnostic interrupt for the analog and digital part.

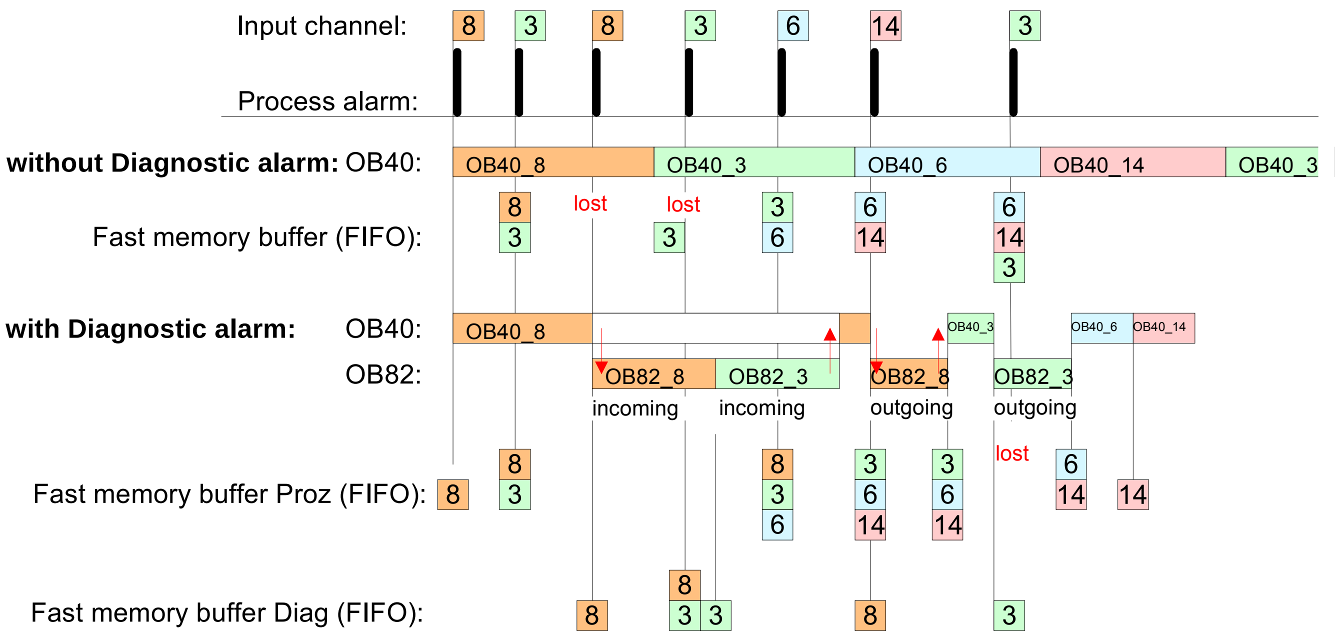
A diagnostic interrupt occurs when during a hardware interrupt execution in OB 40 another hardware interrupt is thrown for the same event. The initialization of a diagnostic interrupt interrupts the recent hardware interrupt execution in OB 40 and branches in OB 82 to diagnostic interrupt processing$_{\text{incoming}}$. If during the diagnostic interrupt processing other events are occurring at other channels that may also cause a process res. diagnostic interrupt, these are interim stored.

After the end of the diagnostic interrupt processing at first all interim-stored diagnostic interrupts are processed in the sequence of their occurrence and then all hardware interrupts.

If a channel where currently a diagnostic interrupt$_{\text{incoming}}$ is processed res. interim stored initializes further hardware interrupts, these get lost. When a hardware interrupt for which a diagnostic interrupt$_{\text{incoming}}$ has been released is ready, the diagnostic interrupt processing is called again as diagnostic interrupt$_{\text{going}}$.

All events of a channel between diagnostic interrupt$_{\text{incoming}}$ and diagnostic interrupt$_{\text{going}}$ are not stored and get lost. Within this time window (1$^{\text{st}}$ diagnostic interrupt$_{\text{incoming}}$ until last diagnostic interrupt$_{\text{going}}$) the SF-LED of the CPU is on. Additionally for every diagnostic interrupt$_{\text{incoming/going}}$ an entry in the diagnostic buffer of the CPU occurs.

Example



Diagnostic interrupt processing

Every OB 82 call causes an entry in the diagnostic buffer of the CPU containing error cause and module address.

By using the SFC 59 you may read the diagnostic bytes.

At de-activated diagnostic interrupt you have access to the last recent diagnostic event.

If you've activated the diagnostic function in your hardware configuration, the contents of record set 0 are already in the local double word 8 when calling the OB 82. The SFC 59 allows you to also read the record set 1 that contains additional information.

After leaving the OB 82 a clear assignment of the data to the last diagnostic interrupt is not longer possible.

The record sets of the diagnostic range have the following structure:

Record set 0 Diagnostic$_{\text{incoming}}$

Byte	Bit 7 ... Bit 0
0	Bit 0: set at module failure Bit 1: 0 (fix) Bit 2: set at external error Bit 3: set at channel error Bit 4: set when external auxiliary supply is missing Bit 7 ... 5: 0 (fix)
1	Bit 3 ... 0: Module class 0101b: Analog 1111b: Digital Bit 4: Channel information present Bit 7 ... 5: 0 (fix)
2	Bit 3 ... 0: 0 (fix) Bit 4: Failure module internal supply voltage (output overload) Bit 7 ... 5: 0 (fix)
3	Bit 5 ... 0: 0 (fix) Bit 6: Hardware interrupt lost Bit 7: 0 (fix)

Record set 0 Diagnostic$_{\text{going}}$

After the removing error a diagnostic message$_{\text{going}}$ takes place if the diagnostic interrupt release is still active.

Byte	Bit 7 ... Bit 0
0	Bit 0: set at module failure Bit 1: 0 (fix) Bit 2: set at external error Bit 3: set at channel error Bit 4: set when external auxiliary supply is missing Bit 7 ... 5: 0 (fix)
1	Bit 3 ... 0: Module class 0101b: Analog module 1111b: Digital Bit 4: Channel information present Bit 7 ... 5: 0 (fix)
2	00h (fix)
3	00h (fix)

Diagnostic Record set 1 (Byte 0 ... 15)

The record set 1 contains the 4Byte of the record set 0 and additionally 12Byte module specific diagnostic data.

The diagnostic bytes have the following assignment:

Byte	Bit 7 ... Bit 0
0 ... 3	Contents record set 0 (see page before)
4	Bit 6 ... 0: channel type (here 70h) 70h: Digital input 71h: Analog input 72h: Digital output 73h: Analog output 74h: Analog in-/output Bit 7: More channel types present 0: no 1: yes
5	Number of diagnostic bits per channel (here 08h)
6	Number of channels of a module (here 08h)
7	Bit 0: Error in channel group 0 (I+0.0 ... I+0.3) Bit 1: Error in channel group 1 (I+0.4 ... I+0.7) Bit 2: Error in channel group 2 (I+1.0 ... I+1.3) Bit 3: Error in channel group 3 (I+1.4 ... I+I.7) Bit 4: Error in channel group 4 (Counter 0) Bit 5: Error in channel group 5 (Counter 1) Bit 6: reserved Bit 7: reserved
8	Diagnostic interrupt due to hardware interrupt lost at... Bit 0: ... input I+0.0 Bit 1: 0 (fix) Bit 2: ... input I+0.1 Bit 3: 0 (fix) Bit 4: ... input I+0.2 Bit 5: 0 (fix) Bit 6: ... input I+0.3 Bit 7: 0 (fix)
9	Diagnostic interrupt due to hardware interrupt lost at... Bit 0: ... input I+0.4 Bit 1: 0 (fix) Bit 2: ... input I+0.5 Bit 3: 0 (fix) Bit 4: ... input I+0.6 Bit 5: 0 (fix) Bit 6: ... input I+0.7 Bit 7: 0 (fix)
10	Diagnostic interrupt due to hardware interrupt lost at... Bit 0: ... input I+1.0 Bit 1: 0 (fix) Bit 2: ... input I+1.1 Bit 3: 0 (fix) Bit 4: ... input I+1.2 Bit 5: 0 (fix) Bit 7 ... 6: reserved

continued ...

... continue Record set 1

Byte	Bit 7 ... Bit 0
11	Bit 7 ... 0: reserved
12	Diagnostic interrupt due to hardware interrupt lost at... Bit 0: ... Gate Counter 0 closed Bit 1: 0 (fix) Bit 2: ... Gate Counter 0 open Bit 3: 0 (fix) Bit 4: ... Over-/underflow/end value Counter 0 Bit 5: 0 (fix) Bit 6: ... Counter 0 reached comparison value Bit 7: 0 (fix)
13	Diagnostic interrupt due to hardware interrupt lost at... Bit 0: ... Gate Counter 1 closed Bit 1: 0 (fix) Bit 2: ... Gate Counter 1 open Bit 3: 0 (fix) Bit 4: ... Over-/underflow/end value Counter 1 Bit 5: 0 (fix) Bit 6: ... Counter 1 reached comparison value Bit 7: 0 (fix)
14	reserved
15	reserved

Chapter 6 Deployment PtP communication

Overview

Content of this chapter is the deployment of the RS485 slot for serial PtP communication.

Here you'll find all information about the protocols and project engineering of the interface, which are necessary for the serial communication using the RS485 interface.

Content

Fast introduction

General

The CPU 312SC has a RS485 interface, which is fix set to PtP communication (**p**oint-**t**o-**p**oint. This supports the serial process connection to different source or destination systems.

Protocols

The protocols res. procedures ASCII, STX/ETX, 3964R, USS and Modbus are supported.

Parameterization

The parameterization happens during runtime using the SFC 216 (SER_CFG). For this you have to store the parameters in a DB for all protocols except ASCII.

Communication

The communication is controlled by SFCs. Send takes place via SFC 217 (SER_SND) and receive via SFC 218 (SER_RCV).

The repeated call of the SFC 217 SER_SND delivers a return value for 3964R, USS and Modbus via RetVal that contains, among other things, recent information about the acknowledgement of the partner station.

The protocols USS and Modbus allow to evaluate the receipt telegram by calling the SFC 218 SER_RCV after SER_SND.

The SFCs are included in the consignment of the CPU.

Overview SFCs for serial communication

The following SFCs are used for the serial communication:

SFC		Description
SFC 216	SER_CFG	RS485 parameterize
SFC 217	SER_SND	RS485 send
SFC 218	SER_RCV	RS485 receive

Principals of the data transfer

Overview

The data transfer is handled during runtime by using SFCs. The principles of data transfer are the same for all protocols and is shortly illustrated in the following.

Principle

Data that is into the according data channel by the PLC, is stored in a FIFO send buffer (**f**irst **i**n **f**irst **o**ut) with a size of 2x1024Byte and then put out via the interface.

When the interface receives data, this is stored in a FIFO receive buffer with a size of 2x1024Byte and can there be read by the PLC.

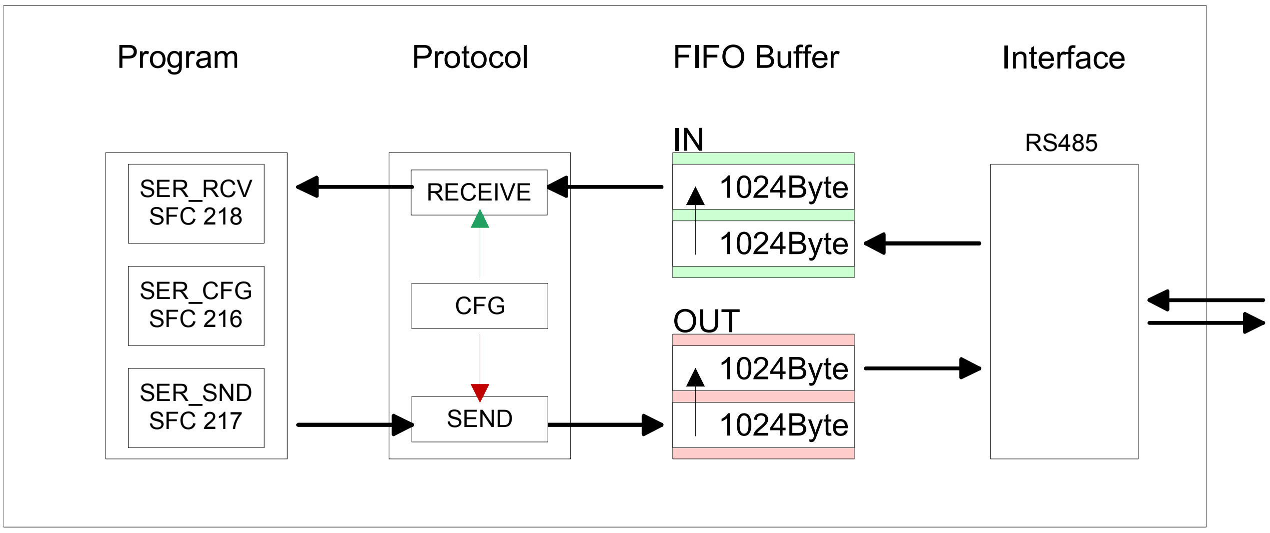
If the data is transferred via a protocol, the adoption of the data to the according protocol happens automatically.

In opposite to ASCII and STX/ETX, the protocols 3964R, USS and Modbus require the acknowledgement of the partner.

An additional call of the SFC 217 SER_SND causes a return value in RetVal that includes among others recent information about the acknowledgement of the partner.

Further on for USS and Modbus after a SER_SND the acknowledgement telegram must be evaluated by call of the SFC 218 SER_RCV.

RS485 PtP communication



Deployment RS485 interface

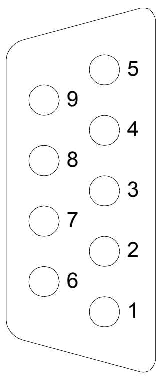
Properties RS485

The RS485 interface from the CPU 312SC is fix set to PtP communication. Parameterization and communication happens by means of SFCs.

The following characteristics distinguish the RS485 interface:

- Logical states represented by voltage differences between the two cores of a twisted pair cable
- Serial bus connection in two-wire technology using half duplex mode
- Data communications up to a max. distance of 500m
- Data communication rate up to 115.2kBaud

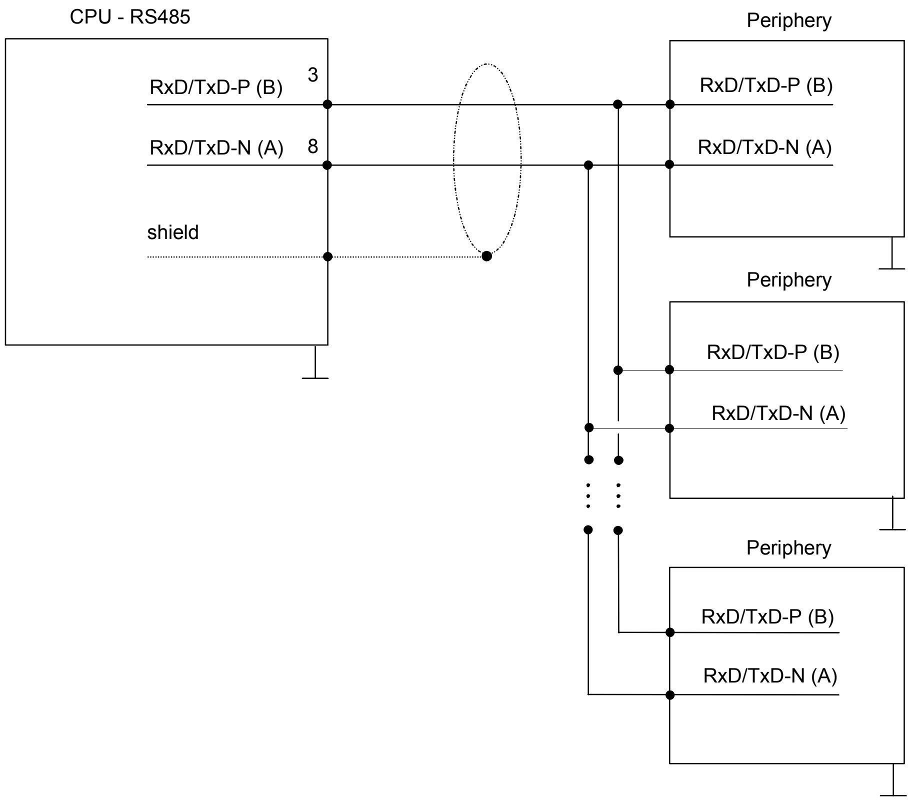
Connection RS485



9pin SubD jack

Pin	RS485
1	n.c.
2	M24V
3	RxD/TxD-P (Line B)
4	RTS
5	M5V
6	P5V
7	P24V
8	RxD/TxD-N (Line A)
9	n.c.

Connection



Parameterization

SFC 216 (SER_CFG)

The parameterization happens during runtime deploying the SFC 216 (SER_CFG). You have to store the parameters for STX/ETX, 3964R, USS and Modbus in a DB.

Name	Declaration	Type	Comment
Protocol	IN	BYTE	1=ASCII, 2=STX/ETX, 3=3964R
Parameter	IN	ANY	Pointer to protocol-parameters
Baudrate	IN	BYTE	Number of the baudrate
CharLen	IN	BYTE	0=5Bit, 1=6Bit, 2=7Bit, 3=8Bit
Parity	IN	BYTE	0=None, 1=Odd, 2=Even
StopBits	IN	BYTE	1=1Bit, 2=1.5Bit, 3=2Bit
FlowControl	IN	BYTE	1 (fix)
RetVal	OUT	WORD	Return value (0 = OK)

Parameter description

All time settings for timeouts must be set as hexadecimal value. Find the hex value by multiply the wanted time in seconds with the baudrate.

Example: Wanted time 8ms at a baudrate of 19200Baud
Calculation: 19200Bit/s x 0,008s ≈ 154Bit → (9Ah)
Hex value is 9Ah.

Protocol

Here you fix the protocol to be used. You may choose between:
1: ASCII
2: STX/ETX
3: 3964R
4: USS Master
5: Modbus RTU Master
6: Modbus ASCII Master

Parameter (as DB)

At ASCII protocol, this parameter is ignored.

At STX/ETX, 3964R, USS and Modbus you fix here a DB that contains the communication parameters and has the following structure for the according protocols:

Data block at STX/ETX

DBB0:	STX1	BYTE	(1. Start-ID in hexadecimal)
DBB1:	STX2	BYTE	(2. Start-ID in hexadecimal)
DBB2:	ETX1	BYTE	(1. End-ID in hexadecimal)
DBB3:	ETX2	BYTE	(2. End-ID in hexadecimal)
DBW4:	TIMEOUT	WORD	(max. delay time between 2 telegrams)

Note!

The start res. end sign should always be a value <20, otherwise the sign is ignored!

Data block at 3964R

DBB0:	Prio	BYTE	(The priority of both partners must be different)
DBB1:	ConnAttmptNr	BYTE	(Number of connection trials)
DBB2:	SendAttmptNr	BYTE	(Number of telegram retries)
DBW4:	CharTimeout	WORD	(Character delay time)
DBW6:	ConfTimeout	WORD	(Acknowledgement delay time)

Data block at USS

DBW0:	Timeout	WORD	(Delay time in)

Data block at Modbus-Master

DBW0:	Timeout	WORD	(Respond delay time)

Baud rate

Velocity of data transfer in Bit/s (Baud).

04h:	1200Baud	05h:	1800Baud	06h:	2400Baud	07h:	4800Baud
08h:	7200Baud	09h:	9600Baud	0Ah:	14400Baud	0Bh:	19200Baud
0Ch:	38400Baud	0Dh:	57600Baud	0Eh:	115200Baud		

CharLen

Number of data bits where a character is mapped to.

0: 5Bit 1: 6Bit 2: 7Bit 3: 8Bit

Parity

The parity is -depending on the value- even or odd. For parity control, the information bits are extended with the parity bit, that amends via its value ("0" or "1") the value of all bits to a defined status. If no parity is set, the parity bit is set to "1", but not evaluated.

0: NONE 1: ODD 2: EVEN

StopBits

The stop bits are set at the end of each transferred character and mark the end of a character.

1: 1Bit 2: 1.5Bit 3: 2Bit

FlowControl

The parameter FlowControl is ignored. When sending RTS=1, when receiving RTS=0.

RetVal SFC 216 (Error message SER_CFG)

Return values send by the block:

Error code	Description
0000h	no error
809Ah	interface not found
8x24h	Error at SFC-Parameter x, with x: 1: Error at "Protocol" 2: Error at "Parameter" 3: Error at "Baudrate" 4: Error at "CharLength" 5: Error at "Parity" 6: Error at "StopBits" 7: Error at "FlowControl" (Parameter missing)
809xh	Error in SFC parameter value x, where x: 1: Error at "Protocol" 3: Error at "Baudrate" 4: Error at "CharLength" 5: Error at "Parity" 6: Error at "StopBits" 7: Error at "FlowControl"
8092h	Access error in parameter DB (DB too short)
828xh	Error in parameter x of DB parameter, where x: 1: Error 1. parameter 2: Error 2. parameter ...

Communication

Overview

The communication happens via the send and receive blocks SFC 217 (SER_SND) and SFC 218 (SER_RCV).

The SFCs are included in the consignment of the CPU.

SFC 217 (SER_SND)

This block sends data via the serial interface.

The repeated call of the SFC 217 SER_SND delivers a return value for 3964R, USS and Modbus via RetVal that contains, among other things, recent information about the acknowledgement of the partner station.

The protocols USS and Modbus require to evaluate the receipt telegram by calling the SFC 218 SER_RCV after SER_SND.

Parameter

Name	Declaration	Type	Comment
DataPtr	IN	ANY	Pointer to Data Buffer for sending data
DataLen	OUT	WORD	Length of data sent
RetVal	OUT	WORD	Return value (0 = OK)

DataPtr

Here you define a range of the type Pointer for the send buffer where the data that has to be sent is stored. You have to set type, start and length.

Example: Data is stored in DB5 starting at 0.0 with a length of 124Byte.

DataPtr:=P#DB5.DBX0.0 BYTE 124

DataLen

Word where the number of the sent Bytes is stored.

At **ASCII** if data were sent by means of SFC 217 faster to the serial interface than the interface sends, the length of data to send could differ from the *DataLen* due to a buffer overflow. This should be considered by the user program.

With **STX/ETX**, **3964R**, **Modbus** and **USS** always the length set in DataPtr is stored or 0.

RetVal SFC 217 (Error message SER_SND)

Return values of the block:

Error code	Description
0000h	Send data - ready
1000h	Nothing sent (data length 0)
20xxh	Protocol executed error free with xx bit pattern for diagnosis
7001h	Data is stored in internal buffer - active (busy)
7002h	Transfer - active
80xxh	Protocol executed with errors with xx bit pattern for diagnosis (no acknowledgement by partner)
90xxh	Protocol not executed with xx bit pattern for diagnosis (no acknowledgement by partner)
8x24h	Error in SFC parameter x, where x:
	1: Error in "DataPtr"
	2: Error in "DataLen"
8122h	Error in parameter "DataPtr" (e.g. DB too short)
807Fh	Internal error
809Ah	Interface not found or interface is used for PROFIBUS
809Bh	Interface not configured

Protocol specific RetVal values

ASCII

Value	Description
9000h	Buffer overflow (no data send)
9002h	Data too short (0Byte)

STX/ETX

Value	Description
9000h	Buffer overflow (no data send)
9001h	Data too long (>1024Byte)
9002h	Data too short (0Byte)
9004h	Character not allowed

3964R

Value	Description
2000h	Send ready without error
80FFh	NAK received - error in communication
80FEh	Data transfer without acknowledgement of partner or error at acknowledgement
9000h	Buffer overflow (no data send)
9001h	Data too long (>1024Byte)
9002h	Data too short (0Byte)

... Continue RetVal SFC 217 SER_SND

USS

Error code	Description
2000h	Send ready without error
8080h	Receive buffer overflow (no space for receipt)
8090h	Acknowledgement delay time exceeded
80F0h	Wrong checksum in respond
80FEh	Wrong start sign in respond
80FFh	Wrong slave address in respond
9000h	Buffer overflow (no data send)
9001h	Data too long (>1024Byte)
9002h	Data too short (<2Byte)

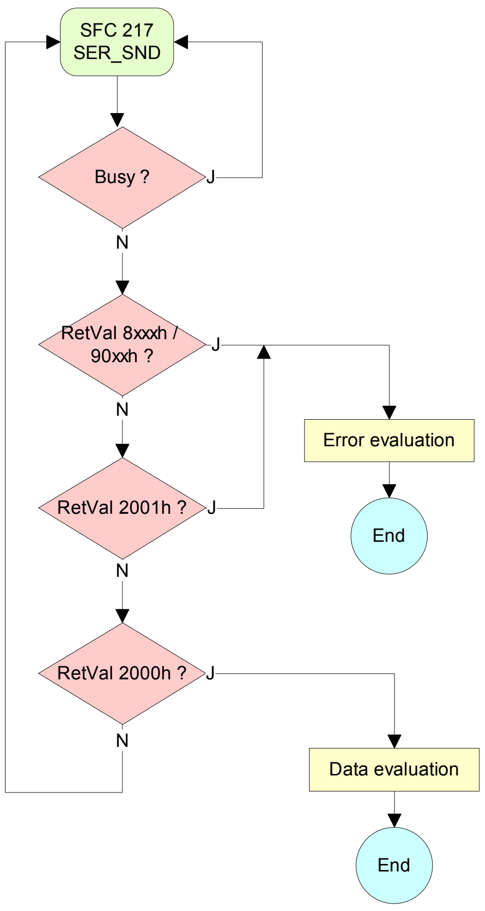
Modbus RTU/ASCII Master

Error code	Description
2000h	Send ready (positive slave respond)
2001h	Send ready (negative slave respond)
8080h	Receive buffer overflow (no space for receipt)
8090h	Acknowledgement delay time exceeded
80F0h	Wrong checksum in respond
80FDh	Length of respond too long
80FEh	Wrong function code in respond
80FFh	Wrong slave address in respond
9000h	Buffer overflow (no data send)
9001h	Data too long (>1024Byte)
9002h	Data too short (<2Byte)

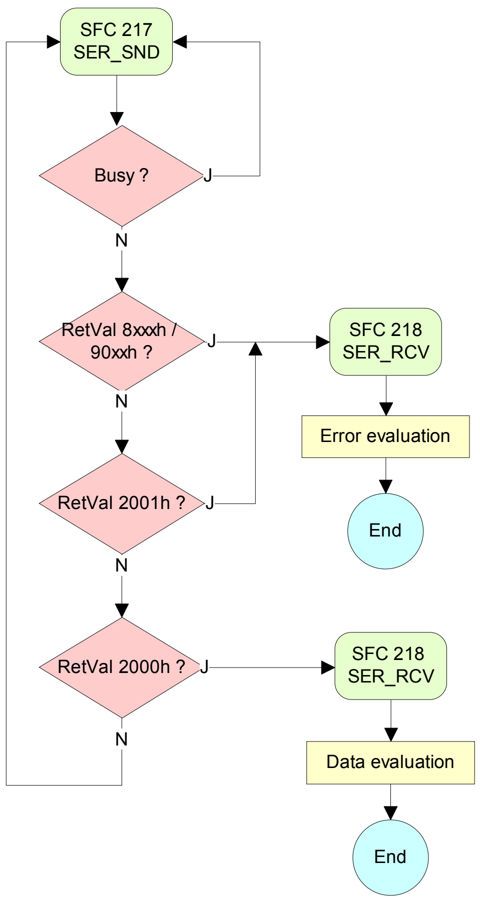
Principles of programming

The following text shortly illustrates the structure of programming a send command for the different protocols.

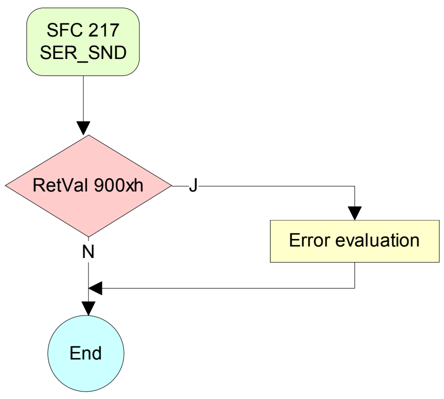
3964R



USS / Modbus



ASCII / STX/ETX



SFC 218 (SER_RCV)

This block receives data via the serial interface.

Using the SFC 218 SER_RCV after SER_SND with the protocols USS and Modbus the acknowledgement telegram can be read.

Parameter

Name	Declaration	Type	Comment
DataPtr	IN	ANY	Pointer to Data Buffer for received data
DataLen	OUT	WORD	Length of received data
Error	OUT	WORD	Error Number
RetVal	OUT	WORD	Return value (0 = OK)

DataPtr

Here you set a range of the type Pointer for the receive buffer where the reception data is stored. You have to set type, start and length.

Example: Data is stored in DB5 starting at 0.0 with a length of 124Byte.
DataPtr:=P#DB5.DBX0.0 BYTE 124

DataLen

Word where the number of received Bytes is stored.

At **STX/ETX** and **3964R**, the length of the received user data or 0 is entered.

At **ASCII**, the number of read characters is entered. This value may be different from the read telegram length.

Error

This word gets an entry in case of an error. The following error messages may be created depending on the protocol:

ASCII

Bit	Error	Description
0	overrun	Overflow, a sign couldn't be read fast enough from the interface
1	framing error	Error that shows that a defined bit frame is not coincident, exceeds the allowed length or contains an additional Bit sequence (Stopbit error)
2	parity	Parity error
3	overflow	Buffer is full

STX/ETX

Bit	Error	Description
0	overflow	The received telegram exceeds the size of the receive buffer.
1	char	A sign outside the range 20h...7Fh has been received.
3	overflow	Buffer is full

3964R / Modbus RTU/ASCII Master

Bit	Error	Description
0	overflow	The received telegram exceeds the size of the receive buffer.

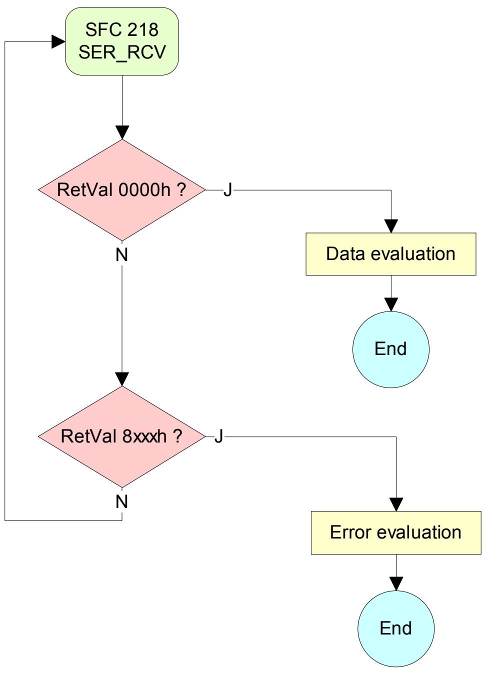
RetVal SFC 218 (Error message SER_RCV)

Return values of the block:

Error code	Description
0000h	no error
1000h	Receive buffer too small (data loss)
8x24h	Error at SFC-Parameter x, with x: 1: Error at "DataPtr" 2: Error at "DataLen" 3: Error at "Error"
8122h	Error in parameter "DataPtr" (e.g. DB too short)
809Ah	Serial interface not found res. interface is used by PROFIBUS
809Bh	Serial interface not configured

Principles of programming

The following picture shows the basic structure for programming a receive command. This structure can be used for all protocols.



Protocols and procedures

Overview

The CPU supports the following protocols and procedures:

- ASCII communication
- STX/ETX
- 3964R
- USS
- Modbus

ASCII

ASCII data communication is one of the simple forms of data exchange.

Incoming characters are transferred 1 to 1.

At ASCII, with every cycle the read-SFC is used to store the data that is in the buffer at request time in a parameterized receive data block. If a telegram is spread over various cycles, the data is overwritten. There is no reception acknowledgement. The communication procedure has to be controlled by the concerning user application. An according Receive_ASCII-FB may be found in the service area of www.vipa.de.

STX/ETX

STX/ETX is a simple protocol with start and end ID, where STX stands for **S**tart of **T**e**x**t and ETX for **E**nd of **T**e**x**t.

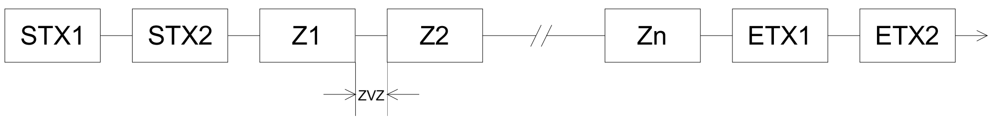
The STX/ETX procedure is suitable for the transfer of ASCII characters. It does not use block checks (BCC). Any data transferred from the periphery must be preceded by a start followed by the data characters and the end character.

Depending of the byte width the following ASCII characters can be transferred: 5Bit: not allowed: 6Bit: 20...3Fh, 7Bit: 20...7Fh, 8Bit: 20...FFh.

The effective data, which includes all the characters between Start and End are transferred to the PLC when the End has been received.

When data is send from the PLC to a peripheral device, any user data is handed to the SFC 217 (SER_SND) and is transferred with added Start- and End-ID to the communication partner.

Message structure:



You may define up to 2 Start- and End-IDs.

You may work with 1, 2 or no Start- and with 1, 2 or no End-ID. As Start- res. End-ID all Hex values from 01h to 1Fh are permissible. Characters above 1Fh are ignored. In the user data, characters below 20h are not allowed and may cause errors. The number of Start- and End-IDs may be different (1 Start, 2 End res. 2 Start, 1 End or other combinations). For not used start and end characters you have to enter FFh in the hardware configuration. If no End-ID is defined, all read characters are transferred to the PLC after a parameterizable character delay time (Timeout).

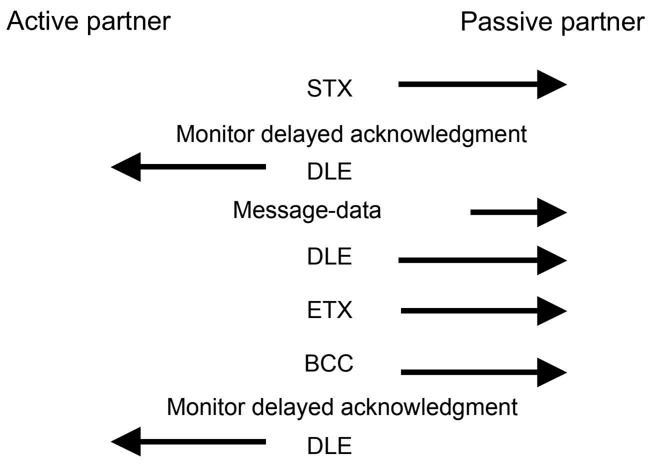
3964R

The 3964R procedure controls the data transfer of a point-to-point link between the CPU and a communication partner. The procedure adds control characters to the message data during data transfer. These control characters may be used by the communication partner to verify the complete and error free receipt.

The procedure employs the following control characters:

• STX	**S**tart of **Tex**t
• DLE	**D**ata **L**ink **E**scape
• ETX	**E**nd of **Tex**t
• BCC	**B**lock **C**heck **C**haracter
• NAK	**N**egative **Ack**nowledge

Procedure


Active partner
Passive partner
STX
Monitor delayed acknowledgment
DLE
Message-data
DLE
ETX
BCC
Monitor delayed acknowledgment
DLE

You may transfer a maximum of 255Byte per message.

Note!

When a DLE is transferred as part of the information it is repeated to distinguish between data characters and DLE control characters that are used to establish and to terminate the connection (DLE duplication). The DLE duplication is reversed in the receiving station.

The 3964R procedure <u>requires</u> that a lower priority is assigned to the communication partner. When communication partners issue simultaneous send commands, the station with the lower priority will delay its send command.

USS

The USS protocol (**U**niverselle **s**erielle **S**chnittstelle = universal serial interface) is a serial transfer protocol defined by Siemens for the drive and system components. This allows to build-up a serial bus connection between a superordinated master and several slave systems.

The USS protocol enables a time cyclic telegram traffic by presetting a fix telegram length.

The following features characterize the USS protocol:

- Multi point connection
- Master-Slave access procedure
- Single-Master-System
- Max. 32 participants
- Simple and secure telegram frame

You may connect 1 master and max. 31 slaves at the bus where the single slaves are addressed by the master via an address sign in the telegram. The communication happens exclusively in half-duplex operation.

After a send command, the acknowledgement telegram must be read by a call of the SFC 218 SER_RCV.

The telegrams for send and receive have the following structure:

Master-Slave telegram

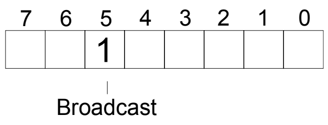
STX	LGE	ADR	PKE		IND		PWE		STW		HSW		BCC
02h			H	L	H	L	H	L	H	L	H	L	

Slave-Master telegram

STX	LGE	ADR	PKE		IND		PWE		ZSW		HIW		BCC
02h			H	L	H	L	H	L	H	L	H	L	

where	STX:	Start sign	STW:	Control word
	LGE:	Telegram length	ZSW:	State word
	ADR:	Address	HSW:	Main set value
	PKE:	Parameter ID	HIW:	Main effective value
	IND:	Index	BCC:	Block Check Character
	PWE:	Parameter value		

Broadcast with set Bit 5 in ADR-Byte



A request may be directed to a certain slave ore be sent to all slaves as broadcast message. For the identification of a broadcast message you have to set Bit 5 to 1 in the ADR-Byte. Here the slave addr. (Bit 0 ... 4) is ignored. In opposite to a "normal" send command, the broadcast does not require a telegram evaluation via SFC 218 SER_RCV. Only write commands may be sent as broadcast.

Modbus

The Modbus protocol is a communication protocol that fixes a hierarchic structure with one master and several slaves.

Physically, Modbus works with a serial half-duplex connection.

There are no bus conflicts occurring, because the master can only communicate with one slave at a time. After a request from the master, this waits for a preset delay time for an answer of the slave. During the delay time, communication with other slaves is not possible.

After a send command, the acknowledgement telegram must be read by a call of the SFC 218 SER_RCV.

The request telegrams send by the master and the respond telegrams of a slave have the following structure:

Start sign	Slave address	Function Code	Data	Flow control	End sign

Broadcast with slave address = 0

A request can be directed to a special slave or at all slaves as broadcast message. To mark a broadcast message, the slave address 0 is used.

In opposite to a "normal" send command, the broadcast does not require a telegram evaluation via SFC 218 SER_RCV.

Only write commands may be sent as broadcast.

ASCII, RTU mode

Modbus offers 2 different transfer modes:

- ASCII mode: Every Byte is transferred in the 2 sign ASCII code. The data are marked with a start and an end sign. This causes a transparent but slow transfer.
- RTU mode: Every Byte is transferred as one character. This enables a higher data pass through as the ASCII mode. Instead of start and end sign, a time control is used.

The mode selection happens during runtime by using the SFC 216 SER_CFG.

Supported Modbus protocols

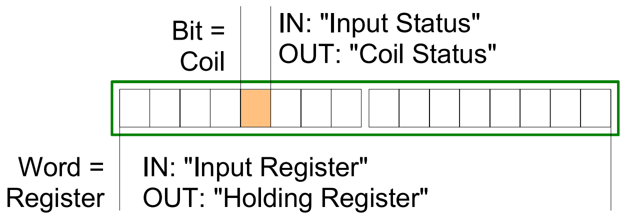
The following Modbus Protocols are supported by the RS485 interface:

- Modbus RTU Master
- Modbus ASCII Master

Modbus - Function codes

Naming convention

Modbus has some naming conventions:



- Modbus differentiates between bit and word access;
  Bits = "Coils" and Words = "Register".
- Bit inputs are referred to as "Input-Status" and Bit outputs as "Coil-Status".
- Word inputs are referred to as "Input-Register" and Word outputs as "Holding-Register".

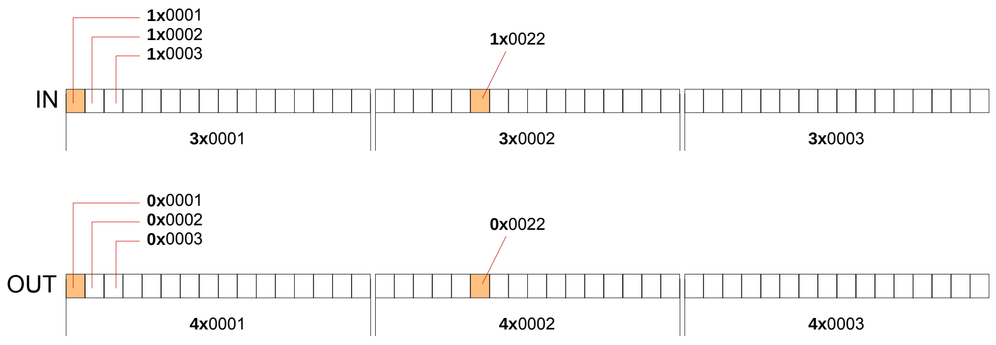
Range definitions

Normally the access at Modbus happens by means of the ranges 0x, 1x, 3x and 4x.

0x and 1x gives you access to *digital* Bit areas and 3x and 4x to *analog* word areas.

For the CPs from VIPA is not differentiating digital and analog data, the following assignment is valid:

0x:	Bit area for master output data Access via function code 01h, 05h, 0Fh
1x:	Bit area for master input data Access via function code 02h
3x:	Word area for master input data Access via function code 04h
4x:	Word area for master output data Access via function code 03h, 06h, 10h



A description of the function codes follows below.

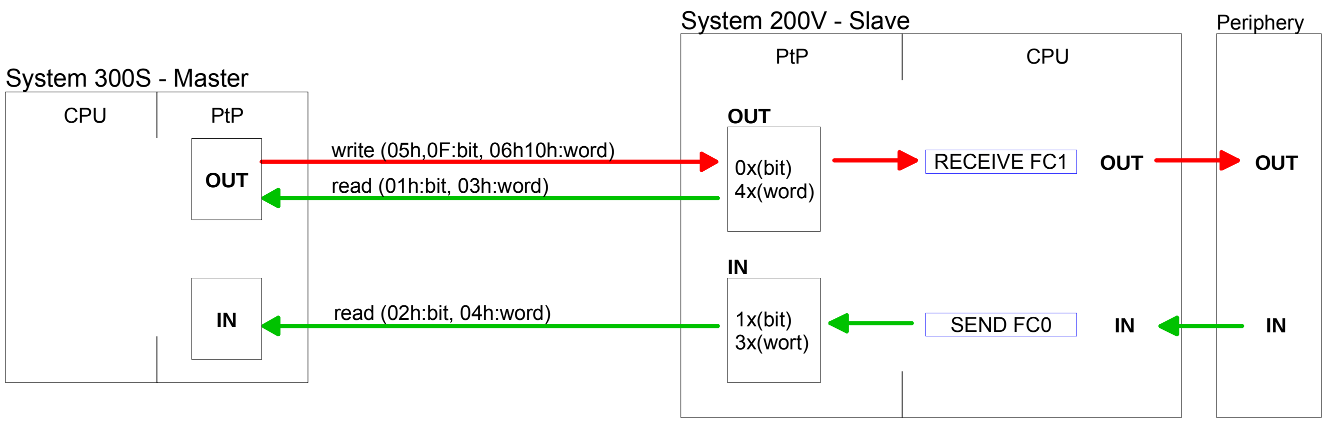
Overview

With the following Modbus function codes a Modbus master can access a Modbus slave: With the following Modbus function codes a Modbus master can access a Modbus slave. The description always takes place from the point of view of the master:

Code	Command	Description
01h	Read n Bits	Read n Bits of master output area 0x
02h	Read n Bits	Read n Bits of master input area 1x
03h	Read n Words	Read n Words of master output area 4x
04h	Read n Words	Read n Words master input area 3x
05h	Write 1 Bit	Write 1 Bit to master output area 0x
06h	Write 1 Word	Write 1 Word to master output area 4x
0Fh	Write n Bits	Write n Bits to master output area 0x
10h	Write n Words	Write n Words to master output area 4x

Point of View of "Input" and "Output" data

The description always takes place from the point of view of the master. Here data, which were sent from master to slave, up to their target are designated as "output" data (OUT) and contrary slave data received by the master were designated as "input" data (IN).



Respond of the slave

If the slave announces an error, the function code is send back with an "ORed" 80h. Without an error, the function code is sent back.

Slave answer:	Function code OR 80h	→ Error
	Function code	→ OK

Byte sequence in a Word

For the Byte sequence in a Word is always valid:

1 Word	
High Byte	Low Byte

Check sum CRC, RTU, LRC

The shown check sums CRC at RTU and LRC at ASCII mode are automatically added to every telegram. They are not shown in the data block.

Read n Bits 01h, 02h

Code 01h: Read n Bits of master output area 0x
Code 02h: Read n Bits of master input area 1x

Command telegram

Slave address	Function code	Address 1. Bit	Number of Bits	Check sum CRC/LRC
1 Byte	1 Byte	1 Word	1 Word	1 Word

Respond telegram

Slave address	Function code	Number of read Bytes	Data 1. Byte	Data 2. Byte	...	Check sum CRC/LRC
1 Byte	1 Byte	1 Byte	1 Byte	1 Byte		1 Word
			max. 250 Byte			

Read n Words 03h, 04h

03h: Read n Words of master output area 4x
04h: Read n Words master input area 3x

Command telegram

Slave address	Function code	Address 1. Bit	Number of Words	Check sum CRC/LRC
1 Byte	1 Byte	1 Word	1 Word	1 Word

Respond telegram

Slave address	Function code	Number of read Bytes	Data 1. Word	Data 2. Word	...	Check sum CRC/LRC
1 Byte	1 Byte	1 Byte	1 Word	1 Word		1 Word
			max. 125 Words			

Write 1 Bit 05h

Code 05h: Write 1 Bit to master output area 0x
A status change is via "Status Bit" with following values:

"Status Bit" = 0000h $\rightarrow$ Bit = 0
"Status Bit" = FF00h $\rightarrow$ Bit = 1

Command telegram

Slave address	Function code	Address Bit	Status Bit	Check sum CRC/LRC
1 Byte	1 Byte	1 Word	1 Word	1 Word

Respond telegram

Slave address	Function code	Address Bit	Status Bit	Check sum CRC/LRC
1 Byte	1 Byte	1 Word	1 Word	1 Word

Write 1 Word 06h

Code 06h: Write 1 Word to master output area 4x

Command telegram

Slave address	Function code	Address word	Value word	Check sum CRC/LRC
1 Byte	1 Byte	1 Word	1 Word	1 Word

Respond telegram

Slave address	Function code	Address word	Value word	Check sum CRC/LRC
1 Byte	1 Byte	1 Word	1 Word	1 Word

Write n Bits 0Fh

Code 0Fh: Write n Bits to master output area 0x

Please regard that the number of Bits has additionally to be set in Byte.

Command telegram

Slave address	Function code	Address 1. Bit	Number of Bits	Number of Bytes	Data 1. Byte	Data 2. Byte	...	Check sum CRC/LRC
1 Byte	1 Byte	1 Word	1 Word	1 Byte	1 Byte	1 Byte	1 Byte	1 Word
					max. 250 Byte			

Respond telegram

Slave address	Function code	Address 1. Bit	Number of Bits	Check sum CRC/LRC
1 Byte	1 Byte	1 Word	1 Word	1 Word

Write n Words 10h

Code 10h: Write n Words to master output area 4x

Command telegram

Slave address	Function code	Address 1. Word	Number of words	Number of Bytes	Data 1. Word	Data 2. Word	...	Check sum CRC/LRC
1 Byte	1 Byte	1 Word	1 Word	1 Byte	1 Word	1 Word	1 Word	1 Word
					max. 125 Words			

Respond telegram

Slave address	Function code	Address 1. Word	Number of Words	Check sum CRC/LRC
1 Byte	1 Byte	1 Word	1 Word	1 Word

Modbus - Example communication

Outline

The example establishes a communication between a master and a slave via Modbus. The following combination options are shown:

Modbus master (M)	Modbus slave (S)
CPU 31xS	CPU 21xSER-1

Components

The following components are required for this example:

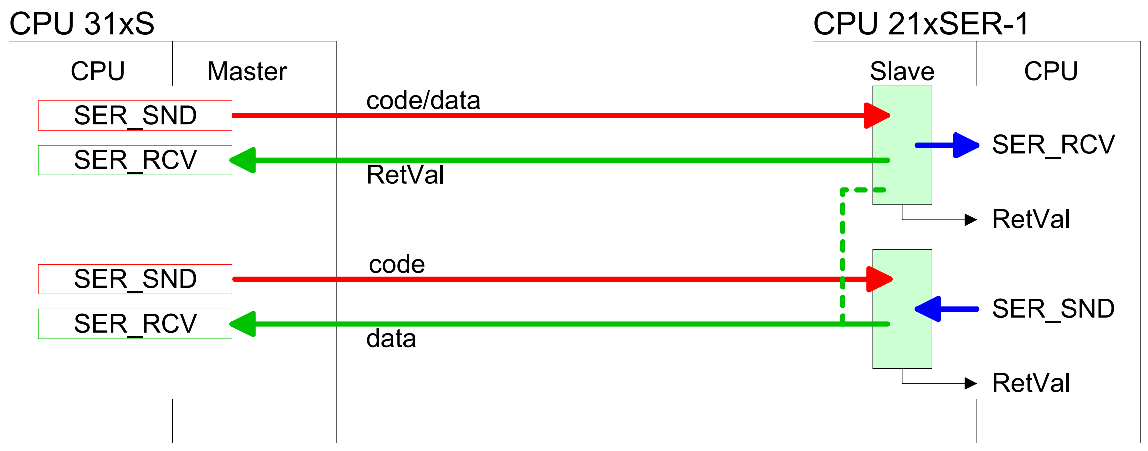
- CPU 31xS as Modbus RTU master
- CPU 21xSER-1 as Modbus RTU slave
- Siemens SIMATIC Manager and possibilities for the project transfer
- Modbus cable connection

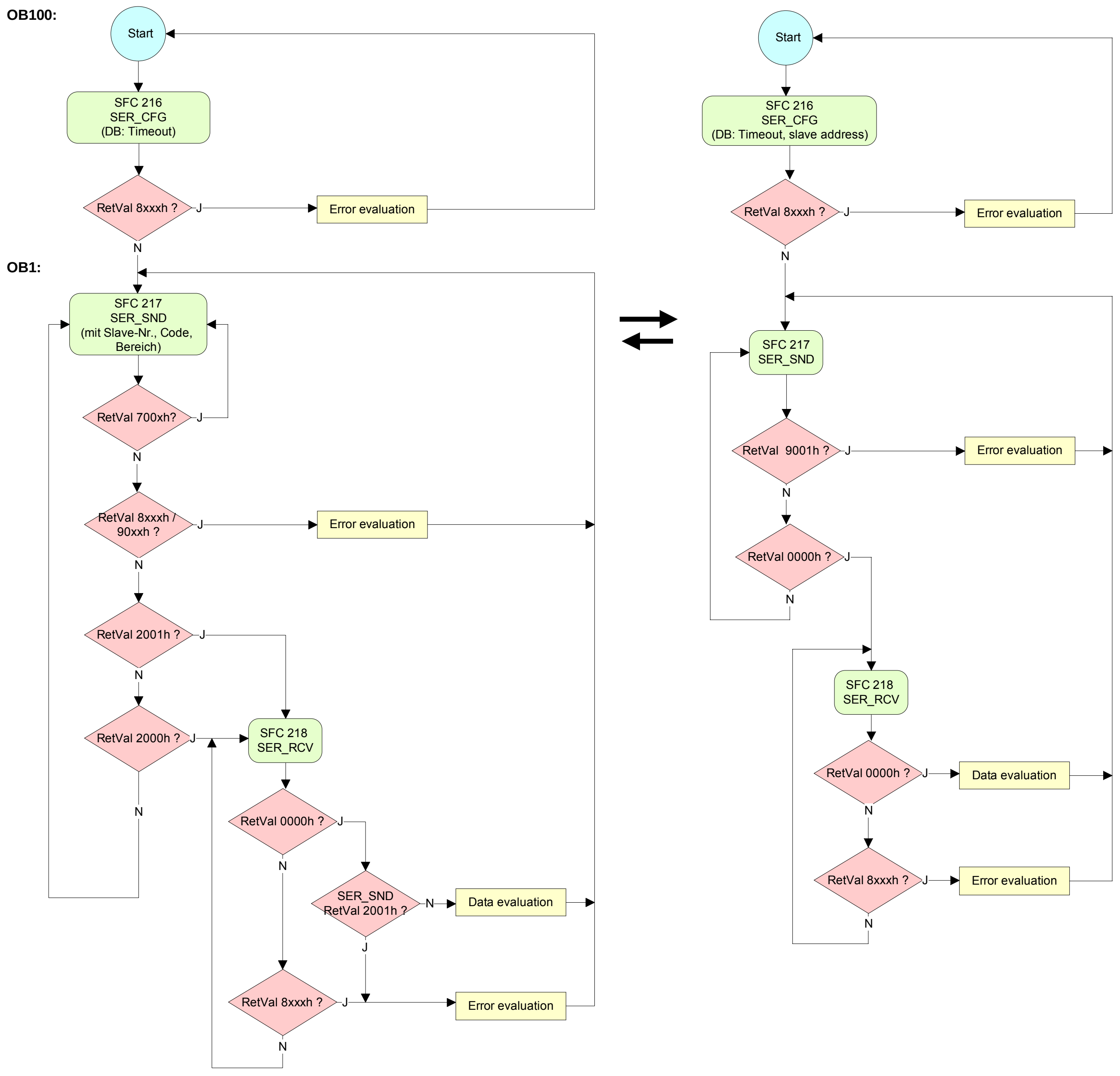
Approach

- Assemble a Modbus system consisting of a CPU 31xS as Modbus master and a CPU 21xSER-1 as Modbus slave and Modbus cable.
- Execute the project engineering of the master!
  For this you create a PLC user application with the following structure:
  OB 100: Call SFC 216 (configuration as Modbus RTU master) with timeout setting and error evaluation.
  OB 1: Call SFC 217 (SER_SND) where the data is send with error evaluation. Here you have to build up the telegram according to the Modbus rules.
  Call SFC 218 (SER_RECV) where the data is received with error evaluation.
- Execute the project engineering of the slave!
  The PLC user application at the slave has the following structure:
  OB 100: Call SFC 216 (configuration as Modbus RTU slave) with timeout setting and Modbus address in the DB and error evaluation.
  OB 1: Call SFC 217 (SER_SND) for data transport from the slave CPU to the output buffer.
  Call SFC 218 (SER_RECV) for the data transport from the input buffer to the CPU. Allow an according error evaluation for both directions.

The following page shows the structure for the according PLC programs for master and slave.

Master **Slave**





Chapter 7 WinPLC7

Overview

In this chapter the programming and simulation software WinPLC7 from VIPA is presented. WinPLC7 is suited for every with Siemens STEP®7 programmable PLC.

Besides the system presentation and installation here the basics for using the software is explained with a sample project.

More information concerning the usage of WinPLC7 may be found in the online help respectively in the online documentation of WinPLC7.

Content

System presentation

General

WinPLC7 is a programming and simulation software from VIPA for every PLC programmable with Siemens STEP®7.

This tool allows you to create user applications in FBD, LAD and STL.

Besides of a comfortable programming environment, WinPLC7 has an integrated simulator that enables the simulation of your user application at the PC without additional hardware.

This “Soft-PLC” is handled like a real PLC and offers the same error behavior and diagnosis options via diagnosis buffer, USTACK and BSTACK.

Note!

Detailed information and programming samples may be found at the online help respectively in the online documentation of WinPLC7.

Alternatives

There is also the possibility to use the Siemens SIMATIC manager instead of WinPLC7 from VIPA. Here the proceeding is part of this manual.

System requirements

- Pentium with 233MHz and 64Mbyte work space
- Graphics card with at least 16bit color - we recommend a screen resolution of at least 1024x768 pixel.
- Windows 98SE/ME, Windows 2000,
  Windows XP (Home and Professional), Windows Vista

Source

You may receive a *demo version* from VIPA. Without any activation with the *demo version* the CPUs 11x of the System 100V from VIPA may be configured.

To configure the SPEED7 CPUs a license for the "profi" version is necessary. This may be online received and activated.

There are the following sources to get WinPLC7:

Online

At www.vipa.de in the service area at *Downloads* a link to the current demo version and the updates of WinPLC7 may be found.

CD

Order no.	Description
SW211C1DD	WinPLC7 Single license, CD, with documentation in german
SW211C1ED	WinPLC7 Single license, CD, with documentation in english
SW900T0LA	ToolDemo VIPA software library free of charge respectively demo versions, which may be activated

Installation

Preconditions

The project engineering of a SPEED7 CPU from VIPA with WinPLC7 is only possible using an activated "Profi" version of WinPLC7.

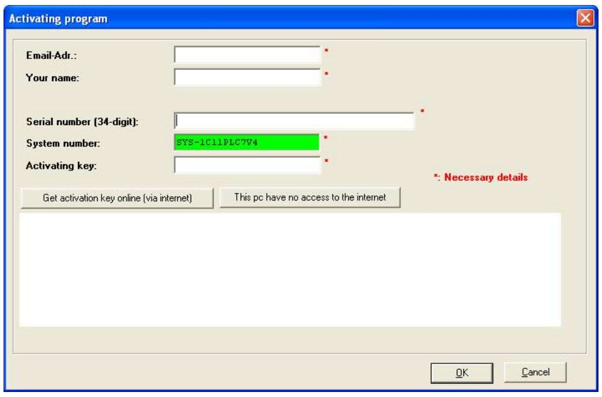
Installation WinPLC7 Demo

The installation and the registration of WinPLC7 has the following approach:

- For installation of WinPLC7 start the setup program of the corresponding CD respectively execute the online received exe file.
- Choose the according language.
- Agree to the software license contract.
- Set an installation directory and a group assignment and start the installation.

Activation of the "Profi" version

- Start WinPLC7. A "Demo" dialog is shown.
- Press the <q> key. The following dialog for activation is shown:


Activating program

Email-Adr.: *
Your name: *
Serial number (34-digit): *
System number: SYS-1C11PLC7V4 *
Activating key: *
*: Necessary details

Get activation key online (via internet)
This pc have no access to the internet
OK
Cancel

- Fill in the following fields:
  Email-Addr., *Your Name* und *Serial number*. The serial number may be found on a label at the CD case.
- If your computer is connected to Internet you may online request the *Activation Key* by [Get activation key via Internet]. Otherwise click at [This PC has no access to the internet] and follow the instructions.
- With successful registration the activation key is listed in the dialog window respectively is sent by email.
- Enter the activation key and click to [OK]. Now, WinPLC7 is activated as "Profi" version.

Installation of WinPCAP for station search via Ethernet

To find a station via Ethernet (accessible nodes) you have to install the WinPCAP driver. This driver may be found on your PC in the installation directory at WinPLC7-V4/WinPcap_4_0.exe.

Execute this file and follow the instructions.

Example project engineering

Job definition

In the example a FC 1 is programmed, which is cyclically called by the OB 1. By setting of 2 comparison values (*value1* and *value2*) during the FC call, an output of the PLC-System should be activated depending on the comparison result.

Here it should apply:

if *value1* = *value2* activate output Q 124.0

if *value1* > *value2* activate output Q 124.1

if *value1* < *value2* activate output Q 124.2

Precondition

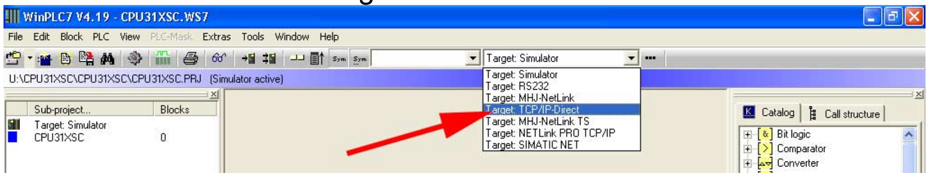
- You have administrator rights for your PC.
- WinPLC7 is installed and activated as "Profi" version.
- One CPU and one digital output module are installed and cabled.
- The Ethernet PG/OP channel of the CPU is connected to your Ethernet network. Your CPU may be connected to your PC with an Ethernet cable either directly or via hub/switch.
- WinPCap for station search via Ethernet is installed.
- The power supply of the CPU and the I/O periphery are activated and the CPU is in STOP state.

Project engineering

- Start WinPLC7 ("Profi" version)
- Create and open a new project by **File** > *Open/create a project*.

Hardware configuration

- For the call of the hardware configurator it is necessary to set WinPLC7 from the *Simulator*-Mode to the *Offline*-Mode. For this and the communication via Ethernet set "Target: TCP/IP Direct".



- Start the hardware configurator with . Please regard an object is selected with a double click at an object in the hardware configurator.
- Choose in the register *Select PLC-System* the parameter "VIPA SPEED7" and click to [Create]. A new station is created.
- Save the empty station. A station name and a comment may be entered before saving.
- By double click choose the according VIPA CPU in the hardware catalog at CPU SPEED7.
- For output place a digital output module and assign the start address 124.
- Save the hardware configuration.

Online access via Ethernet PG/OP channel

- Open the *CPU-Properties*, by double clicking to the CPU at slot 2 in the hardware configurator.
- Click to the button [Ethernet CP-Properties (PG/OP-channel)]. The *Properties CP343* is opened.
- Chose the register *Common Options*.
- Click to [Properties Ethernet].
- Choose the subnet "PG_OP_Ethernet".
- Enter a valid IP address-and a subnet mask. You may get this from your system administrator.
- Close every dialog window with [OK].
- Select, if not already done, "Target: External TCP/IP direct".
- Open with **Online** > *Send configuration to the CPU* a dialog with the same name.
- Click to [Accessible nodes]. Please regard to use this function it is necessary to install WinPCap before!
- Choose your network card and click to [Determining accessible nodes]. After a waiting time every accessible station is listed. Here your CPU with IP 0.0.0.0 is listed, too. To check this the according MAC address is also listed. This MAC address may be found at a label beneath the front flap of the CPU.
- For the temporary setting of an IP address select you CPU and click to [Temporary setting of the IP parameters]. Please enter the same IP parameters, you configured in the CPU properties and click to [Write Parameters].
- Confirm the message concerning the overall reset of the CPU. The IP parameters are transferred to the CPU and the list of accessible stations is refreshed.
- Select you CPU and click to [Confirm]. Now you are back in the dialog "Send configuration".

Transfer hardware configuration

- Choose your network card and click to [Send configuration]. After a short time a message is displayed concerning the transfer of the configuration is finished.

Note!

Usually the online transfer of the hardware configuration happens within the hardware configurator.

With **File** > *Save active station in the WinPL7 sub project* there is also the possibility to store the hardware configuration as a system file in WinPLC7 to transfer it from WinPLC7 to the CPU.

The hardware configuration is finished, now and the CPU may always be accessed by the IP parameters as well by means of WinPLC7.

Programming of the FC 1

The PLC programming happens by WinPLC7. Close the hardware configurator and return to your project in WinPLC7.

The PLC program is to be created in the FC 1.

Creating block FC 1

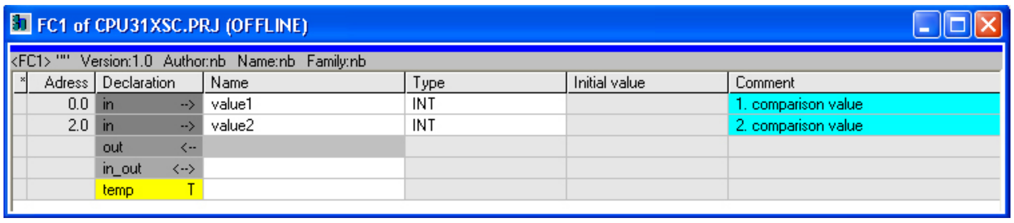
- Choose **File** > *Create new block*.
- Enter "FC1" as block and confirm with [OK]. The editor for FC 1 is called.

Creating parameters

In the upper part of the editor there is the *parameter table*. In this example the 2 integer values *value1* und *value2* are to be compared together. Since both values are read only by the function, these are to be defined as "in".

- Select the "in   -->" row at the *parameter table* and enter at the field *Name* "value1". Press the [Return] key. The cursor jumps to the column with the data type.
- The data type may either directly be entered or be selected from a list of available data types by pressing the [Return] key. Set the data type to INT and press the [Return] key. Now the cursor jumps to the *Comment* column.
- Here enter "1. compare value" and press the [Return] key. A new "in -->" row is created and the cursor jumps to *Name*.
- Proceed for *value2* in the same way as described for *value1*.
- Save the block.

The parameter table shows the following entries, now:



FC1 of CPU31XSC.PRJ (OFFLINE)

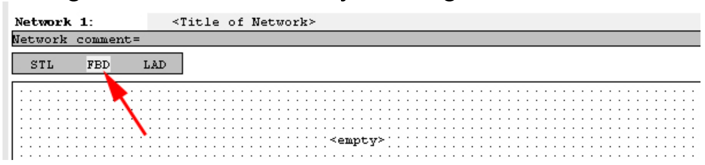
<FC1> "" Version:1.0 Author:nb Name:nb Family:nb

Adress	Declaration		Name	Type	Initial value	Comment
0.0	in	-->	value1	INT		1. comparison value
2.0	in	-->	value2	INT		2. comparison value
	out	<--				
	in_out	<-->				
	temp	T				

Enter the program

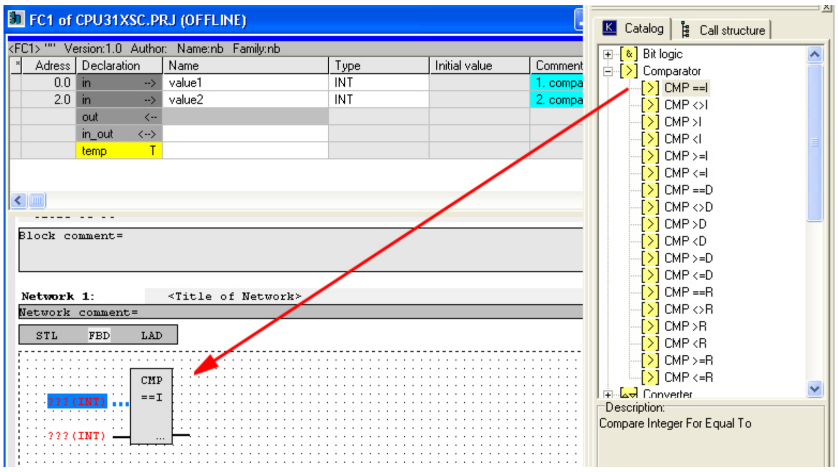
As requested in the job definition, the corresponding output is activated depending on the comparison of *value1* and *value2*. For each comparison operation a separate network is to be created.

- The program is to be created as FBD (function block diagram). Here change to the FBD view by clicking at FBD.



- Click to the input field designated as "empty".
  The available operations may be added to your project by drag&drop from the hardware catalog or by double click at them in the hardware catalog.

- Open in the catalog the category "Comparator" and add the operation "CMP==I" to your network.



FC1 of CPU31XSC.PRJ (OFFLINE)

<FC1> "" Version:1.0 Author: Name:nb Family:nb

Adress	Declaration		Name	Type	Initial value	Comment
0.0	in	-->	value1	INT		1. compa
2.0	in	-->	value2	INT		2. compa
	out	<--				
	in_out	<-->				
	temp	T				

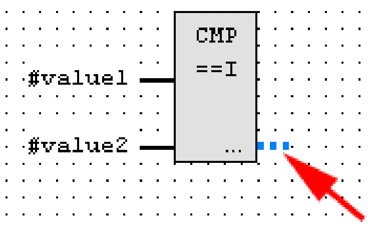
```
Block comment=

Network 1:        <Title of Network>
Network comment=
```

Description:
Compare Integer For Equal To

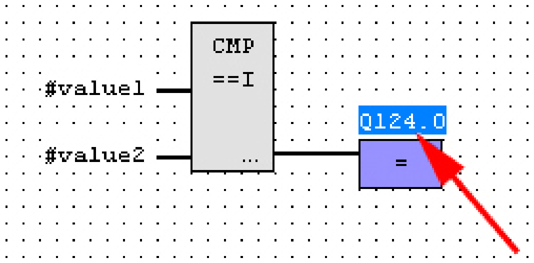
- Click to the input left above and insert *value1*. Since these are block parameters a selection list of block parameters may be viewed by entering "#".
- Type in "#" and press the [Return] key.
- Choose the corresponding parameter and confirm it with the [Return] key.
- Proceed in the same way with the parameter *value2*.

The allocation to the corresponding output, here Q 124.0, takes place with the following proceeding:

- Click to the output at the right side of the operator.



- Open in the catalog the category "Bit logic" and select the function "--[=]". The inserting of "--=" corresponds to the WinPLC7 shortcut [F7].
- Insert the output Q 124.0 by clicking to the operand.



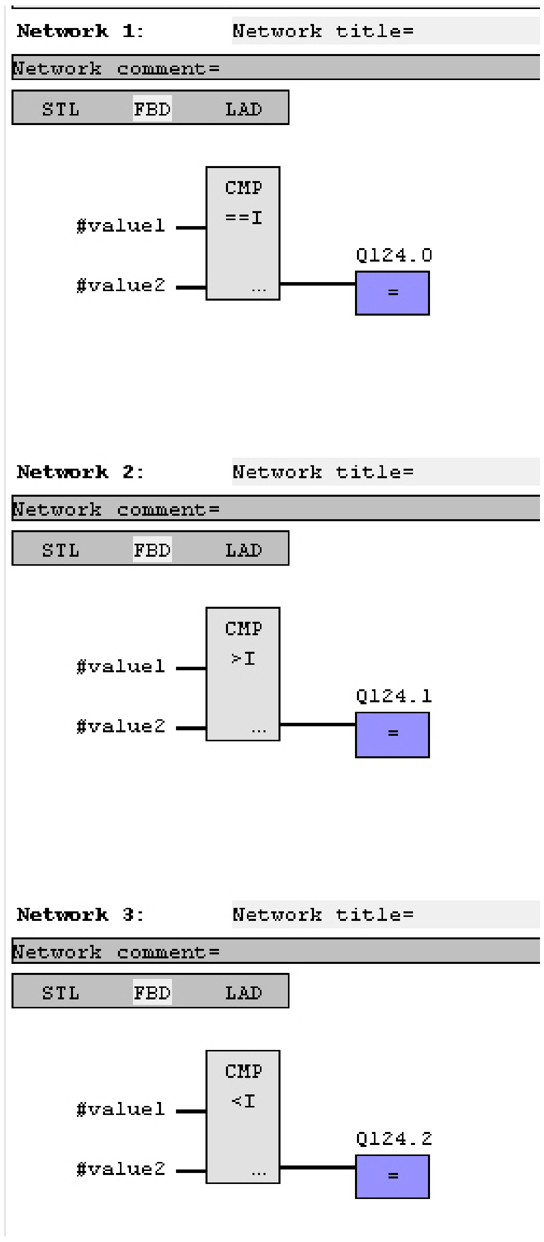
Network1 is finished, now.

Adding a new network

For further comparisons the operations "CMP>I" at Q 124.1 and "CMP<I" at Q 124.2 are necessary. Create a network for both operations with the following proceeding:

- Move your mouse at an arbitrary position on the editor window and press the right mouse key.
- Select at the context menu "Insert new network". A dialog field is opened to enter the position and number of the networks.
- Proceed as described for "Network 1".
- Save the FC 1 with **File** > *Save content of focused window* respectively press [Strg]+[S].

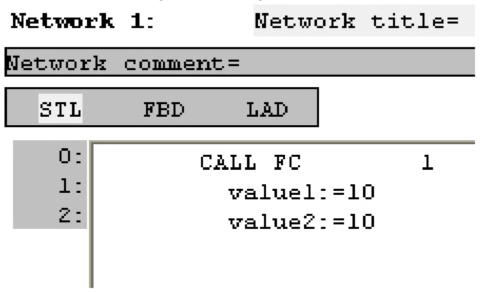
After you have programmed the still missing networks, the FC 1 has the following structure:



Creating the block OB 1

The FC 1 is to be called from the cycle OB 1.

- To create the OB 1 either you select **File** > *Create new block* or click to button [Display OB 1] and create the OB 1.
- Change to the STL view.
- Type in "Call FC 1" and press the [Return] key. The FC parameters are automatically displayed and the following parameters are assigned:



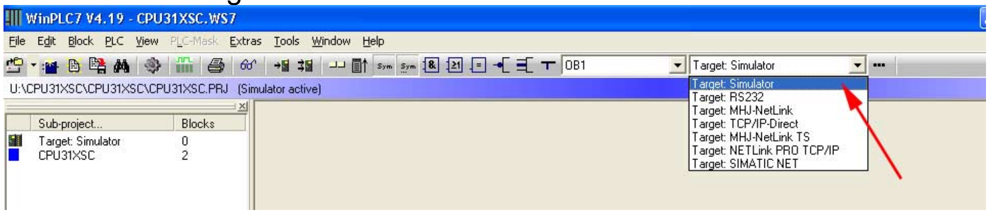
```
Network 1:        Network title=
Network comment=
STL     FBD     LAD
0:      CALL FC         1
1:        value1:=10
2:        value2:=10
```

- Save the OB 1 with **File** > *Save content of focused window* respectively *press* [Strg]+[S].

Test the PLC program in the *Simulator*

With WinPLC7 there is the possibility to test your project in a *simulator*.

- Here select "Target: Simulator".



WinPLC7 V4.19 - CPU31XSC.WS7

U:\CPU31XSC\CPU31XSC\CPU31XSC.PRJ (Simulator active)

Sub-project...	Blocks
Target: Simulator	0
CPU31XSC	2

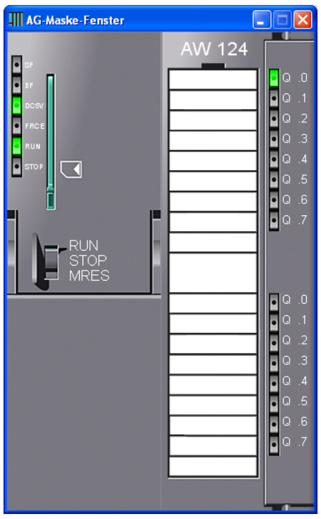
- Transfer the blocks to the simulator with **PLC** > *Send all* blocks.
- Switch the CPU to RUN, by clicking to the photo "Switch/Operating mode" and select in the dialog window the button [Warm restart]. The displayed state changes from STOP to RUN.
- To view the process image select **View** > *Display process image window*.
- Double click to the process image and enter at "Line 2" the address PQB124. Confirm with [OK]. A value marked by red color corresponds to a logical "1".
- Open the OB 1 with the button [Display OB 1].
- Change the value of one variable, save the OB 1 and transfer it to the simulator. According to your settings the process image changes immediately. The status of your blocks may be displayed with **Block** > *Monitoring On/Off*.

Visualization via PLC mask

A further component of the simulator is the *PLC mask*. Here a CPU is graphically displayed, which may be expanded by digital and analog peripheral modules.

As soon as the CPU of the simulator is switched to RUN state, inputs may be activated by mouse and outputs may be displayed.

- Open the PLC mask with **view** > *PLC mask*. A CPU is graphically displayed.
- By clicking the right mouse button within the PLC mask the context menu is opened. Choose for this example "Insert 16-port digital input module". The module is displayed at the right side of the CPU.
- Double-click to the output module, open its properties dialog and enter the *Module address* 124.
- Switch the operating mode switch to RUN by means of the mouse. Your program is executed and displayed in the simulator, now.



Transfer PLC program to CPU and its execution

- For transfer to the CPU set the transfer mode to "Target: TCP/IP-Direct".
- For presetting the Ethernet data click to [...] and click to [Accessible nodes].
- Choose your network card and click to [Determining accessible nodes]. After a waiting time every accessible station is listed.
- Choose your CPU, which was provided with TCP/IP address parameters during the hardware configuration and click to [Confirm].
- Close the "Ethernet properties" dialog with [OK].
- Transfer the blocks to your CPU with **PLC** > *Send all blocks*.
- Switch your CPU to RUN state.
- Open the OB 1 with the button [Display OB 1].
- Change the value of one variable, save the OB 1 and transfer it to the CPU. According to your settings the process image changes immediately. The status of your blocks may be displayed with **Block** > *Monitoring On/Off*.